Stochastic optimization of maintenance scheduling : blackbox methods, decomposition approaches - Theoretical and numerical aspects

Thomas Bittar

École doctorale Mathématiques et Sciences et Technologies de l'Information et de la Communication

THÈSE DE DOCTORAT

Spécialité : Mathématiques

Présentée par

Thomas BITTAR

Pour obtenir le grade de

Docteur de l'École des Ponts ParisTech

Stochastic Optimization of Maintenance Scheduling: Blackbox Methods, Decomposition Approaches - Theoretical and Numerical Aspects

---

Soutenance le 5 Février 2021 devant le jury composé de :

Président du jury:	M. Michel DE LARA	École des Ponts ParisTech
Directeur de thèse:	M. Jean-Philippe CHANCELIER	École des Ponts ParisTech
Rapporteurs:	M. Sébastien LE DIGABEL	Polytechnique Montréal
	M. Jérôme LELONG	Grenoble INP - ENSIMAG
Examinateurs:	M. Pierre CARPENTIER	ENSTA Paris
	Mme Nadia OUDJANE	EDF R&D
	Mme Delphine SINOQUET	IFPEN
Invité:	M. Jérôme LONCHAMPT	EDF R&D

Cette thèse a été préparée conjointement au CERMICS à l'École des Ponts ParisTech et dans le département PRISME d'EDF R&D sous la forme d'un contrat CIFRE.

Résumé

Le but de cette thèse est de développer des algorithmes pour la planification optimale de la maintenance. On s'intéresse à des systèmes de grande taille constitués de plusieurs composants liés par un stock commun de pièces de rechange. Les tests numériques sont effectués sur des systèmes de composants d'une même centrale hydroélectrique.

La première partie est consacrée à l'étude des méthodes de type boîte noire qui sont souvent utilisées pour la planification de la maintenance. On s'intéresse à un algorithme basé sur le krigeage, Efficient Global Optimization (EGO), et à une méthode de recherche directe, Mesh Adaptive Direct Search (MADS). On présente le fonctionnement des algorithmes aussi bien d'un point de vue théorique que pratique et on propose quelques améliorations pour l'implémentation d'EGO. On compare MADS et EGO sur un banc d'essai académique et sur des cas industriels de petite taille, montrant la supériorité de MADS mais aussi les limites des méthodes boîte noire lorsque l'on veut s'attaquer à des problèmes de grande taille.

Dans une deuxième partie, on veut prendre en compte la structure du système, constitué de plusieurs composants liés par un stock commun, afin de pouvoir résoudre des problèmes d'optimisation de maintenance en grande dimension. Dans ce but, on développe un modèle de la dynamique du système étudié et on formule explicitement un problème de contrôle optimal stochastique. On met en place un schéma de *décomposition par prédiction*, basé sur le Principe du Problème Auxiliaire (PPA), qui permet de ramener la résolution du problème en grande dimension à la résolution itérative d'une suite de sous-problèmes de plus petite taille. La décomposition est d'abord appliquée sur des cas tests académiques où elle se révèle très performante. Dans le cas industriel, il est nécessaire de procéder à une « relaxation » du système pour appliquer la méthode de décomposition. Lors des tests numériques, on résout une approximation de Monte-Carlo du problème. La décomposition permet d'obtenir des gains substantiels par rapport à l'algorithme de référence.

Pour résoudre l'approximation de Monte-Carlo du problème de maintenance, on a utilisé une version déterministe du PPA. Dans la troisième partie, on étudie le PPA dans le cadre de l'approximation stochastique. On se place dans un espace de Banach et on prouve la mesurabilité des itérés de l'algorithme, on étend des résultats de convergence existant dans les espaces de Hilbert et on donne des vitesses de convergence.

Mots-clés. Optimisation stochastique, Méthodes boîte noire, Principe du Problème Auxiliaire, Décomposition-coordination, Planification de la maintenance.

Abstract

The aim of the thesis is to develop algorithms for optimal maintenance scheduling. We focus on the specific case of large systems that consist of several components linked by a common stock of spare parts. The numerical experiments are carried out on systems of components from a single hydroelectric power plant.

The first part is devoted to blackbox methods which are commonly used in maintenance scheduling. We focus on a kriging-based algorithm, Efficient Global Optimization (EGO), and on a direct search method, Mesh Adaptive Direct Search (MADS). We present a theoretical and practical review of the algorithms as well as some improvements for the implementation of EGO. MADS and EGO are compared on an academic benchmark and on small industrial maintenance problems, showing the superiority of MADS but also the limitation of the blackbox approach when tackling large-scale problems.

In a second part, we want to take into account the fact that the system is composed of several components linked by a common stock in order to address large-scale maintenance optimization problems. For that purpose, we develop a model of the dynamics of the studied system and formulate an explicit stochastic optimal control problem. We set up a scheme of *decomposition by prediction*, based on the Auxiliary Problem Principle (APP), that turns the resolution of the large-scale problem into the iterative resolution of a sequence of subproblems of smaller size. The decomposition is first applied on synthetic test cases where it proves to be very efficient. For the industrial case, a "relaxation" of the system is needed and developed to apply the decomposition methodology. In the numerical experiments, we solve a Sample Average Approximation (SAA) of the problem and show that the decomposition leads to substantial gains over the reference algorithm.

As we use a SAA method, we have considered the APP in a deterministic setting. In the third part, we study the APP in the stochastic approximation framework in a Banach space. We prove the measurability of the iterates of the algorithm, extend convergence results from Hilbert spaces to Banach spaces and give efficiency estimates.

Keywords. Stochastic optimization, Blackbox methods, Auxiliary Problem Principle, Decomposition-coordination, Maintenance scheduling.

Remerciements

Ces trois années de thèse ont été d'une grande richesse tant d'un point de vue scientifique que personnel. Au travers de ces quelques lignes, je tiens à remercier les nombreuses personnes sans qui ce travail n'aurait pas pu voir le jour.

Tout d'abord, je remercie mes deux rapporteurs, Sébastien Le Digabel et Jérôme Lelong, pour avoir pris le temps de relire ma thèse en détail et formulé des commentaires instructifs.

Mes plus vifs remerciements vont à mon directeur de thèse, Jean-Philippe Chancelier, ainsi qu'à Pierre Carpentier pour m'avoir guidé tout au long de ces trois années, toujours avec sympathie et bienveillance. Ajoutant à cela votre rigueur scientifique et votre disponibilité – j'ai toujours pu trouver une porte ouverte lorsque j'avais besoin de conseils avisés – je peux sans nul doute affirmer que j'ai bénéficié d'un encadrement de grande qualité. Cette thèse CIFRE est le résultat d'une symbiose entre un encadrement académique et industriel. À ce titre, je remercie chaleureusement Jérôme Lonchampt, mon encadrant à EDF, qui m'a permis de concilier les aspects théoriques de mes travaux avec les enjeux industriels d'EDF, tout en me laissant une grande liberté. Vous avez constitué un trio d'encadrants avec lequel ce fut un réel plaisir de travailler.

Du côté du CERMICS, je tiens à remercier Michel De Lara, qui m'a beaucoup appris sur la manière de présenter mes travaux de manière claire et pédagogique. J'étais accompagné lors de réunions hebdomadaires par les doctorants du pôle *Optimisation Stochastique* du CERMICS : Adrien, Benoît, Cyrille, Maël et Thomas avec qui nous nous efforcions de mettre toujours plus d'`underbrace` dans les présentations, ce qui nous a amené jusqu'en Norvège d'où je garde des souvenirs précieux. Je remercie plus généralement l'ensemble des doctorants du CERMICS, trop nombreux pour être cités, qui contribuent à mettre une ambiance studieuse, agréable et conviviale au sein du laboratoire ou sur Teams en ces temps de distanciation sociale. Je tiens également à remercier Isabelle Simunic qui est d'une efficacité inégalée pour le traitement de tous les petits tracas administratifs du quotidien.

J'ai passé la majeure partie de mon temps à Chatou, entouré d'une équipe dans laquelle je me suis rapidement senti comme un poisson dans l'eau. Cette intégration a notamment été facilitée par Soufien Hourrig, mon chef de groupe, toujours bienveillant, sachant allier professionnalisme et moments de rigolade. Je cite encore Jérôme Lonchampt auprès de qui j'ai fait mes premiers pas à EDF ainsi qu'Émilie Dautrême, chef du projet AMPH, qui a toujours été enthousiaste lors de mes présentations. Mention spéciale à Bertrand Iooss qui parvient à suivre un nombre incalculable de travaux en même temps et qui m'a suggéré de nombreuses pistes de réflexions intéressantes et fructueuses. Il m'est impossible de dresser la liste de tous les collègues du groupe P17 ou plus généralement du département PRISME mais je vous remercie tous de créer cette ambiance de travail chaleureuse, agrémentée de discussions souvent passionnantes

autour d'un café ou parfois d'une bière. Eh oui, des bières j'ai eu l'occasion d'en partager avec nombre d'entre vous, que vous soyez ingénieurs : Claire, Elias, Laura, Louis, Pauline, Thibault, Vanessa, Vincent, ou éternels stagiaires : Anne, Clément, Justine. Évidemment, la team thésard était toujours présente dans ces moments et a grandement participé à mon épanouissement lors de ces trois années : merci à Antoine, Azénor, Pablo, Sami. Dédicace spéciale à Jérôme qui a égayé ces trois années et auprès de qui j'ai appris beaucoup d'un point de vue personnel. Enfin, je remercie mes compagnons de bureau, Alvaro, l'espagnol qui maîtrise les subtilités de la langue française mieux que les français eux-mêmes, et Paul, avec qui nous avons exploré les capacités insoupçonnées du SI d'EDF et dont il faut se méfier lorsqu'il vous invite innocemment à boire un petit verre.

Je voudrais aussi remercier mes amis, les centraliens sur Paris – Katchu, Loki, Rootz, Stasch – toujours partants pour une randonnée à vélo ou pour aller courir, ainsi que Piot dont j'aimerais bien avoir les talents culinaires. Je n'oublie pas la team foot de Saint-Maur, qui se reconnaîtra, et avec qui c'est toujours un plaisir de vibrer devant le ballon rond. Il y a aussi Clément et Gabriel, présents à mes côtés depuis tant d'années, de près ou de loin. Mes derniers mots vont aux incrustes – Arnaud, Claire, Julien, Maroua, Pia et Karen – chez qui je sais que je trouverai en toutes circonstances un matelas pour m'accueillir.

Enfin, j'adresse un immense merci à ma famille qui a toujours veillée à ce que je puisse étudier dans les meilleures conditions. Être aux côtés de mes parents, mon frère et ma sœur pendant les périodes de confinement m'a permis de continuer efficacement ma thèse et de rester motivé pour une séance de sport quotidienne.

Contents

List of Figures

List of Tables

List of Algorithms

Acronyms

APP	Auxiliary Problem Principle.
BLUP	Best Linear Unbiased Predictor.
CM	Corrective Maintenance.
CMA-ES	Covariance Matrix Adaptation Evolution Strategy.
COCO	COmparing Continuous Optimizers.
DOE	Design Of Experiments.
EGO	Efficient Global Optimization.
EI	Expected Improvement.
FSSF	Fully Sequential Space-Filling.
GP	Gaussian Process.
GPS	Generalized Pattern Search.
LCC	Life Cycle Cost.
LHS	Latin Hypercube Sampling.
MADS	Mesh Adaptive Direct Search.
MLE	Maximum Likelihood Estimation.
NPV	Net Present Value.
OAT	One-At-a-Time.
PM	Preventive Maintenance.
PVA	Predictive Variance Adequation.
SA	Stochastic Approximation.
SAA	Sample Average Approximation.
VME	Valorisation Maintenance Exceptionnelle.

Notations

$\lvert A\rvert$	Cardinal of the set A.
A^{c}	Complement of the set A.
$\mathrm{dist}(x, A)$	Distance between x and the set A.
$\boldsymbol{X}$	A capital bold symbol denotes a random variable.
$\mathrm{Cov}(\boldsymbol{X}, \boldsymbol{Y})$	Covariance between the random variables $\boldsymbol{X}$ and $\boldsymbol{Y}$.
$\mathbb{E}(\boldsymbol{X})$	Expectation of the random variable $\boldsymbol{X}$.
$\langle \cdot\,, \cdot \rangle$	Inner product in a Hilbert space or duality pairing between a Banach space and its topological dual.
$\lVert \cdot \rVert$	Euclidean norm or norm induced by the inner product $\langle \cdot\,, \cdot \rangle$.
$\lceil x \rceil$	Ceiling function: smallest integer greater than or equal to x.
$\lfloor x \rfloor$	Floor function: greatest integer less than or equal to x.
∇	Gradient.
∇_u	Partial gradient with respect to u.
∂_u	Partial derivative with respect to u.
I_d	Identity matrix of size d.
$\mathbf{1}_A$	Indicator function of the set A.
$\mathcal{N}(\mu, \Sigma)$	Normal distribution with mean μ and covariance matrix Σ.
$\mathrm{proj}_A(u)$	Projection of the vector u onto the set A.
$\mathbb{P}$-a.s.	$\mathbb{P}$-almost surely.
$\mathbb{N}$	Set of all natural numbers.
$\mathbb{Q}$	Set of all rational numbers.
$\overline{\mathbb{R}}$	Extended real line $\overline{\mathbb{R}} = \mathbb{R} \cup \{-\infty, +\infty\}$.
$\mathbb{R}$	Set of all real numbers.
$\mathbb{Z}$	Set of all integers.

1 Introduction (Version française)

En mathématiques, l'art de poser des questions est plus précieux que de résoudre des problèmes.

Georg Cantor

Contents

Cette thèse est le fruit d'une collaboration entre le département PRISME[1] d'EDF R&D[2] et le CERMICS[3], laboratoire de recherche en mathématiques appliquées de l'École des Ponts ParisTech. Elle s'inscrit dans le cadre du projet AMPH[4] dont le but est de fournir à la filière hydraulique des outils décisionnels pour une maintenance adaptée et optimisée, rendue nécessaire par le vieillissement des aménagements du parc dans un contexte de contraintes toujours plus fortes sur les ressources. Plusieurs axes de travail sont abordés dans le projet AMPH :

1. Exploiter au mieux les connaissances à disposition pour évaluer les risques de défaillance des matériels.

2. Optimiser les politiques de maintenance pour assurer les performances du parc sur la durée.

3. Être en appui sur les activités de surveillance, diagnostic et pronostic des matériels.

Ce travail de thèse s'intègre dans le deuxième axe ci-dessus et touche donc au domaine de l'optimisation mathématique avec une application à la gestion d'actifs industriels.

Un *actif physique* est un élément physique quelconque qui produit ou qui a de la valeur pour une entreprise (par exemple une centrale électrique). La *gestion d'actifs industriels* est le domaine qui s'intéresse à l'analyse du cycle de vie des actifs physiques en utilisant des outils et méthodes pour la quantification des risques technico-économiques, dans le but de fournir une aide aux décisions d'investissement. Plus précisément, on se focalise sur les décisions d'investissement liées aux politiques de maintenance des actifs physiques. La Section 1.1 décrit le problème industriel de planification optimale de la maintenance qui a donné naissance à ce projet de recherche, ainsi que les verrous mathématiques associés. La Section 1.2 est une revue de la littérature sur l'optimisation de la maintenance. On donne aussi un aperçu des deux principales approches développées dans la thèse. Enfin, la Section 1.3 décrit le plan du manuscrit.

1.1 Contexte industriel

Pour produire de l'électricité, EDF exploite de nombreuses installations physiques comme des centrales nucléaires ou hydroélectriques, des éoliennes ou des panneaux solaires par exemple. EDF cherche à optimiser l'exploitation de ces installations de manière à en améliorer la fiabilité et la performance technico-économique. Optimiser la maintenance est un des principaux leviers d'actions pour atteindre ce but.

1.1.1 Le problème d'optimisation de la maintenance

Dans cette thèse, on s'intéresse à des composants de centrale hydroélectrique tels que des turbines, des transformateurs ou des alternateurs, voir Figure 1.1.

Ces équipements sont coûteux (de 1 à 10 millions d'euros), de grande taille et sont au cœur du processus de production d'électricité. On étudie des systèmes qui comprennent jusqu'à 80 composants du même type et partageant un stock commun de pièces de

[1] PRISME : Performance, Risque Industriel et Surveillance pour la Maintenance et l'Exploitation
[2] EDF R&D : Électricité De France, Recherche et Développement
[3] CERMICS : Centre d'Enseignement et de Recherche en MathématIques et Calcul Scientifique
[4] AMPH : Asset Management Pour l'Hydraulique

(a) Une turbine Pelton
©EDF - Christophe Huret

(b) Un transformateur
©EDF - Lionel Astruc

(c) Un alternateur
©EDF - Lionel Astruc

FIGURE 1.1: Exemples de composants d'intérêt dans cette thèse.

rechange. Étant donné le prix et la longue durée de vie de ces composants, le stock ne comprend que peu de pièces. Les systèmes sont étudiés sur un horizon de long terme de plusieurs décennies. Au cours du temps, les composants sont sujets à des défaillances aléatoires, dont les dates sont distribuées selon des lois de défaillance supposées connues. Le calcul de ces lois de défaillance est un domaine de recherche en soi qui n'entre pas dans le cadre de ce travail. Ainsi, on suppose que ces lois sont des paramètres d'entrée de notre problème. Si un composant défaillant ne peut pas être remplacé – parce que le stock est vide par exemple – on dit que le système est en *indisponibilité fortuite*, ce qui engendre de lourdes pertes de production. Une politique de maintenance efficace est donc nécessaire pour améliorer la disponibilité des équipements et donc la production électrique, ce qui amène des gains substantiels. On considère deux types de maintenance dans ce travail :

- Une Maintenance Corrective (MC) correspond à la réparation ou au remplacement d'un composant après une défaillance. C'est une opération non planifiée, on réagit à l'occurrence d'un événement aléatoire (la défaillance d'un composant).
- Une Maintenance Préventive (MP) correspond à la réparation ou au remplacement d'un composant avant l'occurrence d'une défaillance. C'est une opération planifiée dont le but est d'éviter une défaillance future. Les MP pouvant être anticipées, elles sont moins coûteuses que les MC.

Définir une *stratégie de maintenance préventive*, ou plus simplement une *stratégie de maintenance*, consiste à fixer les dates de MP pour tous les composants du système. Même si les composants sont indépendants entre eux, le stock commun de pièces de rechange induit un couplage, ce qui implique que la stratégie de maintenance doit être définie à l'échelle du système global et non composant par composant. Une stratégie de maintenance efficace doit prendre en compte les interactions entre les composants et le stock.

L'objectif industriel est de trouver une stratégie de maintenance *optimale*. Pour préciser ce que l'on entend par *optimale*, on introduit les indicateurs économiques qui permettent de quantifier la performance d'un actif, avec une attention particulière pour l'évaluation de la rentabilité des stratégies de maintenance.

1.1.2 Performance d'une politique de maintenance

L'indicateur principal pour évaluer la performance d'un actif est le coût du cycle de vie (CCV). Le CCV est le coût cumulé généré par un actif durant sa vie. Par exemple, la Figure 1.2 représente le coût généré par un actif à chaque année de sa vie, le CCV est

alors la somme de tous les coûts annuels. On prend en compte les coûts de conception et de mise en service qui interviennent avant le fonctionnement nominal de l'actif. Au cours des premières années d'exploitation, la majeure partie des coûts est due au coût d'exploitation et, au fil du temps, les coûts de maintenance et de remise en état deviennent importants. Enfin, après la durée d'exploitation du système, il convient d'envisager les coûts de mise hors service ou de démantèlement.

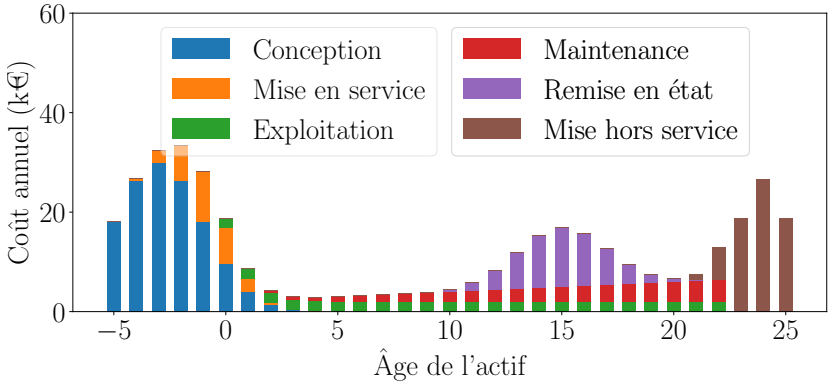


FIGURE 1.2: Exemple de coûts annuels générés par un actif physique. Le coût du cycle de vie est la somme des coûts annuels.

La politique de maintenance d'un actif est un facteur majeur qui influence le CCV. Pour un système donné, on définit une *stratégie de référence*, par exemple une stratégie purement corrective qui consiste à ne réaliser que des MC. Cette stratégie ne requiert pas d'investissement. En revanche, une MP représente un investissement pour l'entreprise. Pour évaluer la rentabilité d'une stratégie de MP, aussi appelée *stratégie d'investissement*, on utilise la Valeur Actuelle Nette (VAN), définie par :

$$\text{VAN} = \text{CCV}_{\text{ref}} - \text{CCV}_{\text{invest}} \ ,$$

où CCV_{ref} (resp. $\text{CCV}_{\text{invest}}$) est le CCV du système lorsque la stratégie de référence (resp. la stratégie d'investissement) est appliquée. La calcul de la VAN est illustrée sur la Figure 1.3. Une VAN positive signifie que l'investissement est rentable, une VAN négative signifie que l'investissement est plus élevé que les gains réalisés. Dans notre cas, la majeure partie des gains correspond en fait à des pertes évitées : on évite des pertes de production dues aux indisponibilités fortuites (en vert) et une partie des coûts de MC (en orange).

Il faut noter que le CCV, et donc la VAN, dépendent des occurrences des défaillances qui sont des événements aléatoires. Ainsi, la VAN est elle-même une variable aléatoire. Une stratégie de maintenance optimale est donc une politique qui optimise un critère de risque sur la VAN. Dans la thèse, on recherche une stratégie de maintenance qui maximise l'espérance de la VAN. Par la linéarité de l'espérance, on a :

$$\mathbb{E}\left(\text{VAN}\right) = \mathbb{E}\left(\text{CCV}_{\text{ref}}\right) - \mathbb{E}\left(\text{CCV}_{\text{invest}}\right) \ ,$$

où l'espérance est calculée sur l'ensemble des scénarios de défaillance. Étant donné que l'espérance du CCV de la stratégie de référence, $\mathbb{E}\left(\text{CCV}_{\text{ref}}\right)$, ne dépend pas de la stratégie évaluée, maximiser l'espérance de la VAN équivaut à minimiser l'espérance du CCV de la stratégie d'investissement, $\mathbb{E}\left(\text{CCV}_{\text{invest}}\right)$. Dans la thèse, l'objectif industriel est donc formulé comme la minimisation de l'espérance du CCV.

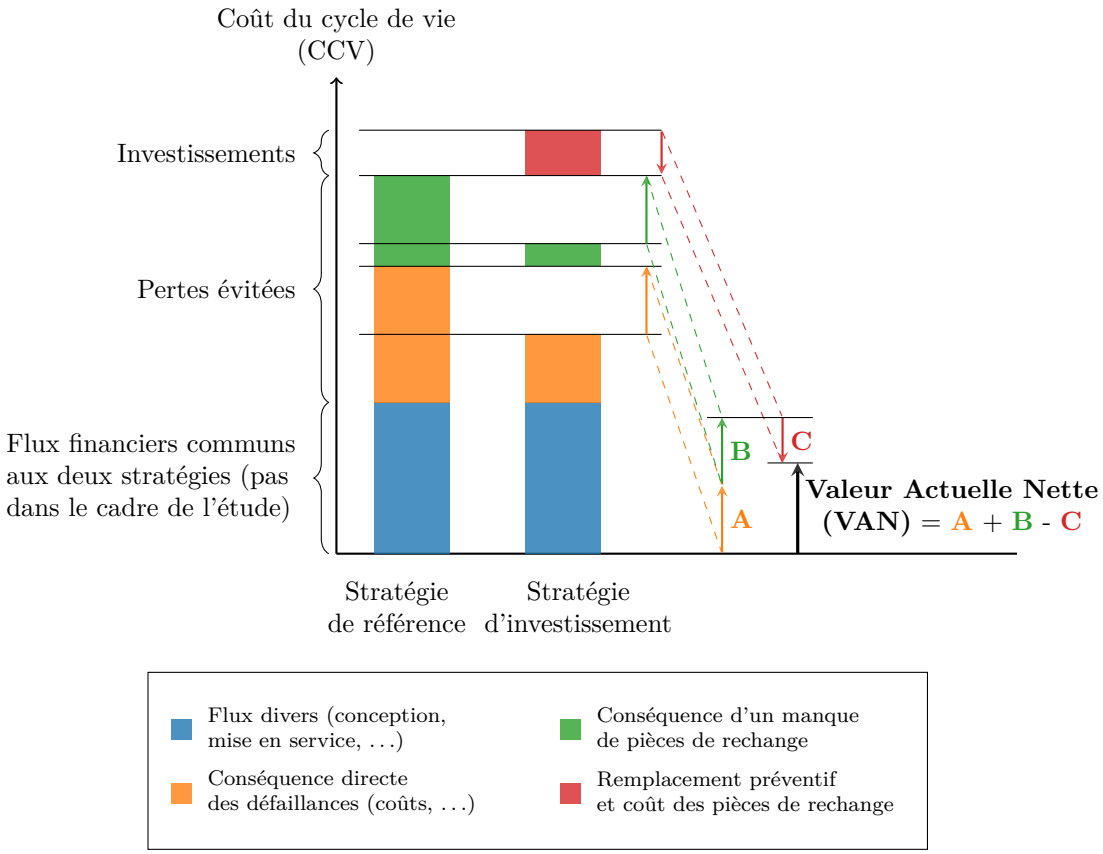


FIGURE 1.3: Calcul de la VAN pour une stratégie d'investissement.

Méthodologie actuelle pour le calcul du CCV. EDF a développé le logiciel Valorisation Maintenance Exceptionnelle (VME) pour l'évaluation de la performance des stratégies d'investissement. Cet outil permet de modéliser des actifs physiques ainsi que leurs interactions et de calculer des indicateurs de risque technico-économiques pour une stratégie de maintenance donnée. VME estime la distribution du CCV avec une méthode de Monte-Carlo où les dates de défaillances sont tirées aléatoirement à partir de la loi de défaillance correspondante. En l'état actuel des choses, on peut exécuter VME avec différentes stratégies de maintenance conçues « à la main » et comparer leur performance. Une exécution de VME correspond en fait à une évaluation de la fonction objectif (l'espérance du CCV). La version actuelle du logiciel ne permet pas d'effectuer l'optimisation de la maintenance. L'objectif industriel de cette thèse est de mettre en place une méthodologie efficace dans ce but.

1.1.3 Verrous mathématiques du problème d'optimisation

Le problème industriel de planification optimale de la maintenance est un problème d'*optimisation stochastique*. L'optimisation stochastique est un domaine des mathématiques qui s'intéresse aux problèmes d'optimisation ayant un caractère aléatoire, que ce soit dans le problème en lui-même (objectif ou contraintes aléatoires) ou dans l'algorithme utilisé pour sa résolution (itérés aléatoires) [Spall, 2003]. Dans notre problème, l'aléa vient des défaillances des composants. Plusieurs défis doivent être relevés d'un point de vue mathématique :

- On considère des systèmes comprenant jusqu'à 80 composants qui sont couplés par un stock commun. Il s'agit d'un problème difficile en grande dimension pour

lequel l'objectif est plutôt d'améliorer la solution trouvée par des méthodes heuristiques que de trouver le véritable optimum.

- Comme VME utilise des simulations de Monte-Carlo pour calculer le CCV, on a seulement accès à une estimation de l'espérance du CCV et pas à sa vraie valeur. Les évaluations sont dites *bruitées*. Il y a un lien fort entre l'importance du bruit et le temps requis pour une évaluation de la fonction : lorsque l'on augmente le nombre de simulations de Monte-Carlo, le bruit est réduit mais les évaluations sont plus coûteuses en temps de calcul. Pour une estimation précise de l'espérance du CCV, i.e. avec un intervalle de confiance à 95% qui est plus petit que 1% du coût moyen, l'évaluation de l'espérance du CCV prend environ 30 minutes pour le cas industriel avec 80 composants.

- VME est vu comme une *boîte noire* : pour une stratégie de maintenance donnée, on ne connaît que la valeur de l'objectif (l'espérance du CCV). On n'a pas de formule analytique liant la stratégie de maintenance au CCV, ni d'informations sur le gradient, qui n'est peut-être même pas défini.

On se trouve dans le contexte *boîte noire à précision variable* [Alarie et al., 2019, Polak and Wetter, 2006] dans le sens où le bruit de la boîte noire VME peut être contrôlé par le nombre de simulations de Monte-Carlo effectuées.

1.2 Approches mathématiques pour l'optimisation de la maintenance

Les politiques de maintenance ont un impact économique majeur et sont étudiées dans de nombreux secteurs industriels comme le secteur de l'électricité [Froger et al., 2016], l'industrie manufacturière [Ding and Kamaruddin, 2015] ou le domaine du génie civil [Sánchez-Silva et al., 2016]. On présente une revue de la littérature des méthodes d'optimisation pour des problèmes de maintenance de systèmes à plusieurs composants. Ensuite, on détaille les deux approches qui sont étudiées dans la thèse.

1.2.1 Revue des méthodes d'optimisation pour la planification de la maintenance

Il y a une grande diversité dans la modélisation des problèmes de maintenance car les objectifs et contraintes varient d'une étude à l'autre [Dekker, 1996]. Néanmoins, de nombreuses revues de littérature existent dans le domaine de la planification de la maintenance [Alrabghi et al., 2013, Cho and Parlar, 1991, Nicolai and Dekker, 2008]. Elles donnent en particulier un résumé des techniques d'optimisation utilisées. Ces techniques peuvent être divisées en deux catégories principales : la programmation mathématique et les méthodes heuristiques [Froger et al., 2016].

Du côté de la programmation mathématique, un processus de décision markovien combiné à un algorithme d'itération sur la valeur pour optimiser conjointement la maintenance et la commande de pièces de rechange est utilisé dans [Olde Keizer et al., 2017]. Pour des problèmes en grande dimension, une résolution frontale est impossible, il est donc souhaitable d'utiliser des méthodes de décomposition [Froger et al., 2016]. Dans [Lusby et al., 2013], un planning d'arrêt de centrales nucléaires est établi en utilisant une décomposition de Benders couplée à diverses heuristiques. La programmation

mixte est utilisée dans [Grigoriev et al., 2006] pour modéliser un problème d'optimisation de maintenance périodique. Une relaxation linéaire du problème est ensuite résolue avec une décomposition de Dantzig-Wolfe.

Étant donnée la difficulté de modéliser des systèmes industriels toujours plus complexes, les méthodes boîte noire ont été largement développées. Elles peuvent s'attaquer à des problèmes avec des objectifs ou des contraintes non-linéaires plus facilement que les approches analytiques [Froger et al., 2016] et peuvent être facilement couplées à des modèles de simulation. Les techniques d'optimisation boîte noire qui sont utilisées pour la planification optimale de la maintenance sont des méthodes heuristiques, comme les algorithmes génétiques [Almakhlafi and Knowles, 2012], l'optimisation par essaims particulaires [Suresh and Kumarappan, 2013] et le recuit simulé [Fattahi et al., 2014]. Dans une thèse précédente à EDF [Demgne, 2015], le système étudié est modélisé par un processus de Markov déterministe par morceaux et l'optimisation de la maintenance est effectuée avec un algorithme génétique. Pour des problèmes de grande taille, la génération de solutions satisfaisantes avec les méthodes boîte noire peut être lente à cause de la taille de l'espace de recherche.

1.2.2 Deux approches pour l'optimisation de la maintenance

Pour résoudre des problèmes d'optimisation industriels, la traduction des besoins opérationnels en un problème mathématique est une étape fondamentale. L'objectif défini par EDF pour l'optimisation de la maintenance est la minimisation de l'espérance du CCV mais l'ensemble des stratégies de maintenance à considérer doit être défini. Par exemple, on peut considérer les stratégies de maintenance périodiques, fixer un nombre maximum de MP pour chaque composant ou alors considérer des stratégies très générales où l'on prend une décision de maintenance chaque année pour chacun des composants. Le choix d'un ensemble approprié pour les stratégies de maintenance est essentiel : avec un ensemble de grande taille pour les stratégies, le coût optimal est meilleur que sur un plus petit sous-ensemble mais la résolution du problème d'optimisation est plus difficile. L'ensemble des stratégies de maintenance admissibles doit être adapté aux enjeux industriels. Pour des opérations courantes, comme la lubrification des composants, il est suffisant de restreindre la recherche aux stratégies de maintenance périodiques. En revanche, pour des maintenances exceptionnelles comme le remplacement d'un composant de grande taille, il peut être intéressant de rechercher des politiques de maintenance plus générales.

Actuellement, l'évaluation de la fonction objectif est faite avec le modèle de simulation VME, considéré comme une boîte noire. L'idée naturelle est de se pencher sur les méthodes d'optimisation boîte noire, qui peuvent être facilement couplées à VME. Cette étude est l'objet de la Partie I du manuscrit. Ces méthodes peuvent être efficaces pour des systèmes de petite taille ou lorsque l'ensemble des stratégies de maintenance admissibles est petit, mais elles sont sujettes à la malédiction de la dimension. Ainsi, lorsque l'on veut aborder des problèmes mettant en jeu des maintenances exceptionnelles – et donc lorsque l'on doit considérer des stratégies très générales – pour des systèmes de grande taille, une autre approche est nécessaire.

Pour s'attaquer aux problèmes d'optimisation de maintenance en grande dimension, on sort du contexte boîte noire. On développe un modèle analytique de la dynamique du système. Ensuite, on met en place une méthode de décomposition qui utilise des informations sur la structure du système afin d'effectuer l'optimisation de manière efficace. Cette approche est présentée dans la Partie II du manuscrit. La dynamique

analytique du système, développée dans la Partie II, est plus simple que celle qui est utilisée au sein de VME et nous permet donc de considérer des stratégies de maintenance plus générales lors de l'optimisation. D'une certaine manière, on perd un peu de la précision de la modélisation de la dynamique au profit d'un espace de recherche plus grand pour les stratégies de maintenance. Les solutions trouvées après cette optimisation qui utilise la dynamique analytique sont ensuite évaluées avec VME pour s'assurer que l'espérance du CCV, calculée avec la dynamique la plus précise, reste satisfaisante.

1.3 Plan du manuscrit

Le manuscrit est composé de trois parties qui sont détaillées ci-dessous.

Approches boîtes noires pour l'optimisation de la maintenance : comparaison entre un algorithme basé sur le krigeage et une méthode de recherche directe. Dans la première partie, on étudie l'algorithme Efficient Global Optimization (EGO) [Jones et al., 1998] qui utilise des métamodèles, et une méthode de recherche directe, Mesh Adaptive Direct Search (MADS) [Audet and Dennis, 2006], pour la résolution du problème de planification optimale de la maintenance.

- Le Chapitre 3 est une revue bibliographique des principes du krigeage et de EGO, abordant à la fois les aspects théoriques et pratiques. Ensuite, on présente une version d'EGO où l'on effectue un étape finale d'optimisation locale car cette méthode a déjà prouvé son efficacité dans [Mohammadi, 2016].

- Dans le Chapitre 4, on s'intéresse aux méthodes par recherche directe. Ce chapitre est une revue bibliographique des méthodes de recherche par motifs et présente les idées qui ont mené à la conception de l'algorithme MADS, qui jouit de fortes garanties de convergence théoriques.

- Dans le Chapitre 5, on présente deux contributions originales. Dans un premier temps, on propose une nouvelle version d'EGO, appelée EGO-FSSF, où le plan d'expérience initial de taille fixe est remplacé par un plan séquentiel ayant de bonnes propriétés de remplissage de l'espace, couplé avec une étape de validation du métamodèle. Le but de cette variante est d'utiliser le budget d'évaluation plus efficacement en permettant l'adaptation du plan initial à la difficulté du problème d'optimisation. La seconde contribution est la comparaison de la performance de nombreux solveurs pour l'étape de maximisation de l'Expected Improvement (EI) au sein d'EGO. Le choix d'un solveur pour la maximisation de l'EI est souvent fondé sur des arguments heuristiques dans la littérature, c'est pourquoi on donne des arguments quantitatifs pour guider cette décision.

- Dans le Chapitre 6, les performances d'EGO et MADS sont comparées sur le banc d'essai COmparing Continuous Optimizers (COCO) [Hansen et al., 2021]. Les deux algorithmes sont ensuite couplés à VME et sont utilisés pour résoudre des cas industriels de petite taille. On compare EGO et MADS et on met en lumière les limites des méthodes boîte noire dès lors que la dimension du problème augmente.

Une méthode de décomposition par prédiction pour l'optimisation de la maintenance. Les méthodes boîte noire sont limitées pour s'attaquer à des problèmes de grande taille. Dans cette seconde partie, on utilise une autre approche qui consiste à formuler explicitement le problème de maintenance en un problème de contrôle optimal stochastique. On résout ce problème à l'aide d'une méthode de décomposition-coordination.

- Le Chapitre 7 est une revue du Principe du Problème Auxiliaire (PPA) et de son utilisation en décomposition. Ce chapitre fournit tous les outils théoriques nécessaires à la compréhension de la méthode de décomposition que l'on va implémenter pour résoudre le problème d'optimisation de la maintenance.

- Dans le Chapitre 8, on développe un modèle analytique d'un système industriel de grande taille. On formule un problème de contrôle optimal stochastique qui permet de considérer des stratégies de maintenance très générales. D'une certaine façon, on *ouvre la boîte noire* pour utiliser des informations sur la fonction à optimiser.[5]

- Dans le Chapitre 9, on applique le PPA sur deux cas tests académiques qui partagent la même structure que le système industriel mais pour lesquels l'implémentation de méthode de décomposition est plus aisée. Les résultats numériques sur ces cas tests permettent de valider l'implémentation de la méthode avant d'aborder le cas industriel.

- Dans le Chapitre 10, on applique la technique de décomposition-coordination sur un problème d'optimisation industriel de grande taille. Les expériences numériques prouvent l'efficacité de la méthodologie. À notre connaissance, c'est la première fois que le PPA est appliqué pour un problème de planification de maintenance.

Le PPA stochastique dans les espaces de Banach : mesurabilité et convergence. La troisième partie explore des aspects théoriques du PPA stochastique.

- Dans le Chapitre 11, on étudie le PPA stochastique, qui est un cadre général regroupant de nombreux algorithmes classiques (par exemple la descente de gradient stochastique ou l'algorithme *mirror descent* stochastique). On prouve la mesurabilité des itérés de l'algorithme et on étend des résultats de convergence dans le cadre des espaces de Banach. On donne aussi des vitesses de convergence pour la fonction objectif en considérant soit la suite des itérés moyennés soit le dernier itéré, ce dernier résultat étant obtenu en adaptant le concept de monotonie de Fejér à nos besoins.

[5] Pour la plupart des problèmes boîte noire, le changement de paradigme que l'on entreprend ici en *ouvrant la boîte noire* ne peut être reproduit. Par exemple, lorsque la boîte noire est un code de mécanique des fluides qui modélise un processus physique gouverné par des équations aux dérivées partielles couplées, il est impossible d'utiliser une méthode analytique pour l'optimisation. Ici, on est dans la situation peu commune où il peut être intéressant de sortir du cadre boîte noire pour concevoir une méthode d'optimisation spécifique au problème et adaptée aux systèmes de grande taille.

1.4 Publications

Les travaux de cette thèse font l'objet de deux articles soumis à des revues à comité de lecture.

- L'article suivant reprend les travaux des Chapitres 7, 8 et 10, et constitue la contribution principale de cette thèse :

  Bittar, T., Carpentier, P., Chancelier, J.-P., and Lonchampt, J. (2020). A Decomposition Method by Interaction Prediction for the Optimization of Maintenance Scheduling. *arXiv:2002.10719 [math]*. (Submitted to Annals of Operations Research).

- Le travail présenté dans le Chapitre 11 fait l'objet de la publication suivante :

  Bittar, T., Carpentier, P., Chancelier, J.-P., and Lonchampt, J. (2021). The stochastic Auxiliary Problem Principle in Banach spaces: measurability and convergence. *arXiv:2101.08073 [math]*. (Submitted to SIAM Journal on Optimization).

2 Introduction

In mathematics the art of asking questions is more valuable than solving problems.

Georg Cantor

Contents

This thesis stems from the collaboration between the PRISME[1] department of EDF R&D[2] and CERMICS[3], the research center in applied mathematics at École des Ponts ParisTech. It is part of the AMPH[4] project, the aim of which is to provide the hydraulic industry with decision-making tools for adapted and optimized maintenance, made necessary by the ageing of the hydroelectric fleet in a context of ever-increasing constraints on resources. The AMPH project addresses several areas of work:

1. Make the most of the available knowledge to assess the risks of equipment failure.
2. Optimize maintenance policies to ensure the long-term performance of the fleet.
3. Be proactive on the activities of monitoring, diagnosis and prognosis of the equipments.

This thesis is part of the second point above and falls within the field of mathematical optimization, with an industrial application to engineering asset management.

A *physical asset* is any physical element that has or produces value for a company (for example a power plant). *Engineering asset management* can be defined as a field that focuses on the integrative analysis of the life cycle of physical assets by using tools and methods for the quantification of technical and economic risks in order to support investment decisions. More precisely, we are concerned with investment decisions related to the maintenance policy of physical assets. In Section 2.1, we describe the industrial problem of optimal maintenance scheduling that gave birth to this research project and the related mathematical challenges. In Section 2.2, we give an overview of the literature on optimal maintenance scheduling and we describe the two main approaches that we develop in this work. Finally, in Section 2.3, we give the outline of the manuscript.

2.1 Industrial context

In order to produce electricity, EDF operates many physical facilities such as nuclear or hydroelectric power plants, wind turbines, solar panels and so on. EDF seeks to optimize the exploitation of these physical assets in order to improve their reliability as well as their economic and technical performance. Maintenance optimization is one of the main levers of action towards this goal.

2.1.1 An optimal maintenance scheduling problem

We are concerned with components of hydroelectric power plants such as turbines, transformers or generators, see Figure 2.1.

These equipments are expensive large components (from 1 to 10 million euros) that are at the heart of the electricity production process. We study systems that consist in a set of up to 80 components of the same kind sharing a common stock of spare parts. Given the price and the long lifespan of these components, the stock consists only in very few parts. The systems are studied on a long term horizon of several decades. Over time, the components experience random failures that occur according to known

[1] PRISME: Performance, Risque Industriel et Surveillance pour la Maintenance et l'Exploitation
[2] EDF R&D: Électricité De France, Recherche et Développement
[3] CERMICS: Centre d'Enseignement et de Recherche en MathématIques et Calcul Scientifique
[4] AMPH: Asset Management Pour l'Hydraulique

(a) A Pelton turbine
©EDF - Christophe Huret

(b) A transformer
©EDF - Lionel Astruc

(c) A generator
©EDF - Lionel Astruc

Figure 2.1: Examples of components of interest in this thesis.

failure distributions. The computation of these failure distributions constitutes a field of research itself and is not in the scope of this work, hence we assume that they are input parameters for our problem. If a failed component cannot be replaced – because the stock is empty for example – the system is in *forced outage* which causes a heavy loss of production. An efficient maintenance policy is then needed to increase the availability of the equipments and hence the electricity production, which can lead to important gains. We consider two kinds of maintenance in this work:

- A Corrective Maintenance (CM) occurs when we repair or replace a component after a failure. This is an unplanned operation, we react to a random event (the failure of a component).
- A Preventive Maintenance (PM) consists in repairing or replacing a component before the occurrence of a failure. This is a planned operation whose goal is to avoid a possible failure. Due to its anticipated scheduling, a PM is cheaper than a CM.

Defining a *preventive maintenance strategy*, or simply a *maintenance strategy*, consists in setting the dates of PM for all the components of the system. Even if the components are independent, the common stock of spare parts introduces a coupling between them, which implies that the maintenance strategy must be designed at the scale of the whole system, and not component by component. An efficient maintenance strategy must take into account the interactions between the components and the stock.

The industrial challenge is to design a methodology that enables to find an *optimal* maintenance strategy. In order to precise what we mean by *optimal*, we introduce the economic indicators that are used to assess the performance of an asset, with a particular focus on the evaluation of the profitability of a maintenance strategy.

2.1.2 Economic performance of a maintenance policy

A major indicator that is used to assess the performance of an asset is the life cycle cost (LCC). The LCC is the cumulative cost generated by the asset during its whole life. For example, Figure 2.2 represents the cost generated by an asset at each year of its life, the LCC is then the sum of all the yearly costs. We take into account the design and commissioning costs that occur before the running time of the asset. In the first operating years, the major part of the cost is due to the operating costs, and, as time goes on, maintenance and refurbishment costs become significant. Finally, after the operating time of the system, decommissioning or dismantling cost should be considered.

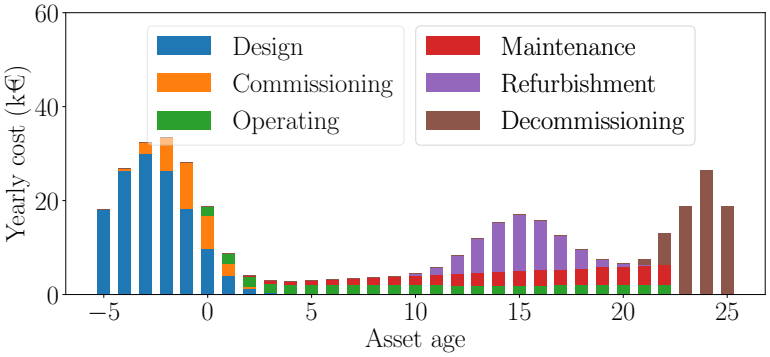


Figure 2.2: Example of yearly costs generated by a physical asset. The life cycle cost is the sum of all the yearly costs.

The maintenance policy of an asset is a major factor that influences the LCC. For a given system, we define a *reference strategy*, for example a pure corrective policy that consists in performing only CMs. This strategy does not require an investment. On the other hand, a PM represents an investment for the company. To evaluate the profitability of a PM strategy, also called *investment strategy*, we use the Net Present Value (NPV) computed as:

$$\mathrm{NPV} = \mathrm{LCC}_{\mathrm{ref}} - \mathrm{LCC}_{\mathrm{invest}} \ ,$$

where $\mathrm{LCC}_{\mathrm{ref}}$ (resp. $\mathrm{LCC}_{\mathrm{invest}}$) is the LCC of the system when the reference strategy (resp. the investment strategy) is applied. The computation of the NPV is illustrated on Figure 2.3. A positive NPV means that the investment is profitable, a negative NPV means that the investment is higher than the realized gains. In our case, the major part of the gains corresponds in fact to avoided losses: we avoid losses of production due to forced outages (in green) and a part of the CM cost (in orange).

Note that the LCC, and in consequence the NPV, depend on the occurrence of failures that are random events. Thus, the NPV is itself a random variable. An optimal maintenance strategy is then a policy that optimizes a risk criterion on the NPV. In the thesis, we focus on finding a strategy that maximizes the expected NPV. By linearity of the expectation, we have:

$$\mathbb{E}\left(\mathrm{NPV}\right) = \mathbb{E}\left(\mathrm{LCC}_{\mathrm{ref}}\right) - \mathbb{E}\left(\mathrm{LCC}_{\mathrm{invest}}\right) \ ,$$

where the expectation is taken over all failure scenarios. As the expectation of the LCC of the reference strategy, $\mathbb{E}\left(\mathrm{LCC}_{\mathrm{ref}}\right)$, does not depend on the evaluated strategy, maximizing the expected NPV is equivalent to minimizing the expected LCC of the investment strategy, $\mathbb{E}\left(\mathrm{LCC}_{\mathrm{invest}}\right)$. In the thesis, the industrial objective for a given system is then formulated as the minimization of the expected LCC.

Current methodology for the computation of the LCC. EDF has developed the software Valorisation Maintenance Exceptionnelle (VME) for the valuation of investment strategies. This tool allows to model physical assets as well as their interactions and to compute technical and economic risk indicators for a given maintenance strategy. VME estimates the distribution of the LCC with a Monte-Carlo method where failure dates of the components are drawn at random from the corresponding known failure distributions. We say that VME provides a *simulation-based model* for

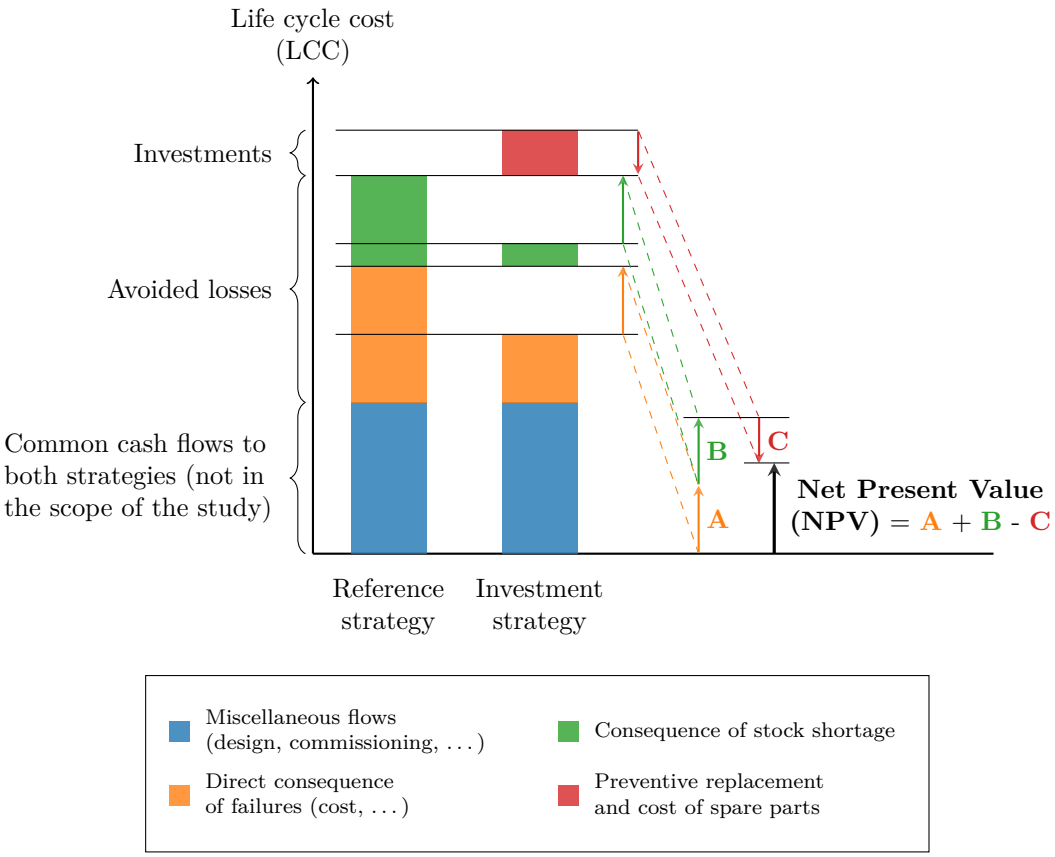


Figure 2.3: Computation of the NPV for a investment strategy.

the industrial system. As things stand, we can run VME with different handcrafted maintenance strategies and compare the performance of the evaluated strategies. In fact, one run of VME corresponds to one evaluation of the objective function (the expected LCC). The current version of the software is not able to perform maintenance optimization. The goal of this thesis is to provide an efficient methodology to do it.

2.1.3 Optimization challenges

The industrial problem of optimal maintenance scheduling is a *stochastic optimization* problem. Stochastic optimization is a field of mathematics concerned with optimization problems that involve randomness, either in the problem (random objective or constraints) or in the algorithm used for the resolution (random iterates) [Spall, 2003]. In our problem, randomness comes from the failures of the component. Some mathematical challenges must be addressed:

- We consider systems with up to 80 components coupled with a common stock. This is a difficult large-scale optimization problem for which the goal is to improve the solution given by heuristic methods rather than to find the true optimum.
- As VME uses Monte-Carlo simulations to compute the LCC, we only have access to an estimation of the expected LCC and not to the true value of the objective. The evaluations are said to be *noisy*. There is a strong link between the amount of noise and the computation time for a function evaluation: when the number of Monte-Carlo simulations is increased, the noise is reduced but the evaluations are more time-consuming. For an accurate estimation of the expected LCC, *i.e.*

with a 95% confidence interval that is smaller than 1% of the expected cost, the evaluation of the expected LCC takes around 30 minutes for the largest industrial case with 80 components.

- VME is a simulation model and is considered to be a *blackbox*: for a given maintenance strategy, we only know the value of the objective function (the expected LCC). We have no analytical formula linking the maintenance strategy to the LCC and no information on the gradient, which may not even be defined.

VME is said to be a *noisy blackbox with adaptive precision* [Alarie et al., 2019, Polak and Wetter, 2006] in the sense that it is a blackbox for which the amplitude of the noise can be controlled by the number of Monte-Carlo simulations.

2.2 Mathematical approaches for optimal maintenance scheduling

Maintenance policies have a major economic impact and are thus studied in many areas of the industry such as the electricity sector [Froger et al., 2016], the manufacturing industry [Ding and Kamaruddin, 2015] or civil engineering [Sánchez-Silva et al., 2016]. We give a literature overview of the optimization methods for multi-component maintenance scheduling problems. Then, we detail the two approaches that are studied in this thesis.

2.2.1 Literature overview of the optimization techniques for maintenance scheduling

There is a great diversity in the modeling of maintenance problems because the objective and constraints vary from one study to another [Dekker, 1996]. Nevertheless, many reviews exist on optimal maintenance scheduling [Alrabghi and Tiwari, 2015, Cho and Parlar, 1991, Nicolai and Dekker, 2008]. In particular, they give a summary from the literature of the optimization techniques used for maintenance problems. They can be split in two main categories: mathematical programming and heuristic methods [Froger et al., 2016].

On the mathematical programming side, a Markov decision process combined with the value iteration algorithm is used in [Olde Keizer et al., 2017] for a joint maintenance and spare part ordering optimization problem. For high-dimensional problems, a frontal resolution is impracticable, and resorting to decomposition methods is relevant [Froger et al., 2016]. In [Lusby et al., 2013], a shutdown planning for the refueling of nuclear power plants is designed with a Benders decomposition coupled with various heuristics. In [Grigoriev et al., 2006], mixed integer programming is used to model a periodic maintenance optimization problem. A linear relaxation of the problem is then solved using Dantzig-Wolfe decomposition.

Due to the difficulty in modeling ever more complex industrial systems, blackbox methods have been largely developed. They can deal with non-linear objectives and contraints more easily than analytical approaches [Froger et al., 2016] and are easy to couple with simulation-based models. The blackbox optimization techniques that are mainly used for optimal maintenance scheduling are heuristic methods, including genetic algorithms [Almakhlafi and Knowles, 2012], particle swarm optimization [Suresh and Kumarappan, 2013] and simulated annealing [Fattahi et al., 2014].

In a previous thesis at EDF [Demgne, 2015], the author models the system of interest with piecewise deterministic Markov processes and optimizes the maintenance with a genetic algorithm. For large-scale problems, generating satisfying solutions with black-box methods may be slow due to the size of the solution space.

2.2.2 Two approaches for maintenance optimization

When tackling industrial optimization problems, the translation of the operational needs into a mathematical problem is a fundamental step. The goal defined by EDF for the optimization of maintenance scheduling is to minimize the expected LCC but the space of maintenance strategies to consider needs to be defined. For example, we can choose to consider only periodic maintenance strategies, to set a maximum number of PMs for each component or to allow for very general strategies where a maintenance decision can be taken each year for each component. Choosing the appropriate space for the maintenance strategies is fundamental: with a large space of strategies, the optimal cost is better than on a smaller subset but the resolution of the optimization problem is harder. The set of admissible maintenance strategies must be adapted to the industrial stakes. For common operations, such as lubrication of the components, it is sufficient to restrict the search to periodic maintenance strategies. On the other hand, for exceptional maintenance such as the replacement of a large component, it may be worth to look for more general policies.

Currently, the evaluation of the objective function is done with the simulation-based model VME that is seen as a blackbox. A natural idea is then to investigate blackbox optimization methods that can be easily plugged to VME. This is the object of Part I of the manuscript. These methods may be efficient for small systems or when the set of admissible maintenance strategies is small, but they are known to be subject to the curse of dimensionality. This is why, if we aim at tackling problems involving exceptional maintenance – hence considering very general strategies – for large-scale systems, another approach is needed.

To tackle high-dimensional maintenance optimization problems, we get out of the blackbox context. We develop an analytical model of the dynamics of the system. Then, we set up a decomposition methodology that uses information on the structure of the system to perform the optimization efficiently. This is the object of Part II of the manuscript. The analytical dynamics of the system developed in Part II is simpler than what is used within VME and allows us to consider more general maintenance strategies in the optimization process. There is a trade-off between the accuracy of the modeling of the dynamics and the size of the search space for the maintenance strategies we can consider. The solutions found after this optimization process using the analytical dynamics can then be evaluated with VME to ensure that the expected LCC, computed with the accurate dynamics, is satisfying.

2.3 Outline of the thesis

The manuscript is composed of three parts that are detailed below.

Blackbox approaches for optimal maintenance scheduling: a comparison between a kriging-based algorithm and a direct search method. In the first part, we investigate a surrogate-based approach, the Efficient Global Optimization

(EGO) algorithm [Jones et al., 1998], and a direct search algorithm, Mesh Adaptive Direct Search (MADS) [Audet and Dennis, 2006], for the resolution of the optimal maintenance scheduling problem.

- In Chapter 3, we carry out a bibliographical review of the principles of kriging and EGO, with considerations both from the practical and the theoretical side. Finally, we present a version of EGO where we do a final optimization with a local solver as it has proved to be efficient in [Mohammadi, 2016].

- In Chapter 4, we focus on direct search methods. This chapter is a bibliographical review of pattern search methods and exposes the ideas that have led to the design of the MADS algorithm, which enjoys strong theoretical guarantees.

- In Chapter 5, we present two original contributions. First, we propose a new version of EGO, called EGO-FSSF, where the classical fixed size initial design is replaced with a fully sequential space-filling initial design coupled with a metamodel validation step. This variant aims at using the evaluation budget more efficiently by allowing the size of the initial design to be adapted to the difficulty of the optimization problem. The second contribution is a benchmark of solvers for the Expected Improvement (EI) maximization step within EGO. The choice of a solver for EI maximization is often based on heuristic arguments in the literature, so we provide quantitative arguments to guide the decision.

- In Chapter 6, the performance of EGO and MADS are compared on the extensive blackbox optimization benchmark COmparing Continuous Optimizers (COCO) [Hansen et al., 2021]. Then, both methods are plugged on VME and run on small industrial test cases. We compare EGO and MADS and highlight the limits of blackbox methods when the dimension of the problem grows.

A decomposition method by interaction prediction for optimal maintenance scheduling. Blackbox methods are limited when it comes to tackle large-scale problems. In this second part, we take another approach by framing the maintenance problem into an explicit stochastic optimal control problem. We solve this problem with a decomposition-coordination method.

- In Chapter 7, we review the APP and its use in decomposition. This chapter provides the necessary theoretical tools in order to understand the decomposition methodology that we will implement to solve the optimal maintenance scheduling problem.

- In Chapter 8, we develop an analytical model of a large-scale industrial system. We formulate a stochastic optimal control problem that allows to consider very general maintenance policies. In a way, we *open the blackbox* to use information on the function to optimize.[5]

[5] For most blackbox problems, the paradigm shift we undertake here by *opening the blackbox* cannot be reproduced. For example, when the blackbox is a computational fluid dynamics code that models a physical process governed by coupled partial differential equations, it is impossible to use analytical methods for the optimization. Here, we are in the uncommon situation where it may be worth getting out of the blackbox framework in order to design a problem-specific optimization method that can tackle large-scale problems.

- In Chapter 9, we apply the APP on two synthetic test cases that share a similar structure as the industrial system but for which the decomposition methodology can be implemented more easily. The numerical results on these synthetic test cases allow to validate the implementation of the method before moving on to the industrial case.

- In Chapter 10, we apply the decomposition-coordination technique on a large-scale industrial optimization problem. Numerical experiments prove the efficiency of the methodology. As far as we know, this is the first time that the APP is applied for a maintenance scheduling problem.

The stochastic APP in Banach spaces: measurability and convergence. The third part explores theoretical aspects of the stochastic APP.

- In Chapter 11, we study the stochastic APP, which is a general framework that gathers many well-known algorithms (for instance stochastic gradient descent or stochastic mirror descent). We prove the measurability of the iterates of the algorithm and extend convergence results to the Banach case. We also give efficiency estimates for the function value taken for the averaged sequence of iterates or for the last iterate, the latter being obtained by adapting the concept of Fejér monotonicity to our needs.

2.4 Publications

The work of this thesis is the subject of two journal papers.

- The following paper summarizes the work of Chapters 7, 8 and 10, and is the main contribution of this thesis:

  Bittar, T., Carpentier, P., Chancelier, J.-P., and Lonchampt, J. (2020). A Decomposition Method by Interaction Prediction for the Optimization of Maintenance Scheduling. *arXiv:2002.10719 [math]*. (Submitted to Annals of Operations Research).

- The work presented in Chapter 11 is the subject of the following publication:

  Bittar, T., Carpentier, P., Chancelier, J.-P., and Lonchampt, J. (2021). The stochastic Auxiliary Problem Principle in Banach spaces: measurability and convergence. *arXiv:2101.08073 [math]*. (Submitted to SIAM Journal on Optimization).

Part I

Blackbox approaches for optimal maintenance scheduling: comparison between a kriging-based algorithm and a direct search method

Introduction to blackbox methods

In this part, we are interested in the following general problem:

$$\min_{u \in U^{\mathrm{ad}}} f(u) \,, \tag{2.1}$$

where $f : \mathbb{R}^d \to \mathbb{R}$ is referred as the *objective function*, with $d \in \mathbb{N}$ being the dimension of the input space. The set $U^{\mathrm{ad}} \subset \mathbb{R}^d$ is assumed to be compact. If moreover f is continuous, then the Weierstrass theorem guarantees the existence of a global minimum for Problem (2.1), that is, a point $u^\sharp \in U^{\mathrm{ad}}$ such that $f(u^\sharp) \leq f(u)$ for all $u \in U^{\mathrm{ad}}$. We assume that we are in the expensive blackbox framework, that is, the evaluations of f are time consuming and we have no information on the gradients, which may not even be defined. The only available information is the value of f at evaluation points. This situation occurs when f is the output of an expensive computer code or the result of physical experiments. An introduction to blackbox optimization can be found in [Audet and Hare, 2017].

In our case, f represents the function that takes a maintenance strategy u as input and returns the expectation of the life cycle cost (LCC), computed with the blackbox computer code VME. We investigate blackbox optimization methods, with the objective of plugging them to the VME software. In this way, we will be able to perform maintenance optimization for small industrial systems.

Blackbox optimization algorithms only use function values to perform the optimization as the derivative information is unavailable. A thorough review and classification of unconstrained blackbox algorithms can be found in [Rios and Sahinidis, 2013]. Following this classification, we can distinguish between *model-based* and *direct search* algorithms. Model-based methods construct approximations of the objective function that are updated and used at each iteration to select the next evaluation point whereas direct search algorithms only use the values of f to determine the search directions. Among model-based methods, we can mention:

- Derivative-free trust-region algorithms [Powell, 2002]. These methods build a local surrogate of the objective function, usually a quadratic approximation, that is supposed to be accurate in the neighborhood – the so-called *trust region* – of the current iterate. The next iterate is chosen by minimizing the surrogate within the trust region. Constructing a quadratic approximation of f requires many function evaluations, which is unaffordable in the expensive setting.

- Response surface methods [Jones, 2001]. These methods construct a global approximation of the objective function, called *metamodel*, *surrogate* or *response surface*, which is a linear combination of some basis functions. Several functional forms can be used for the metamodel (splines, radial-basis functions or defined by

kriging for instance). The next iterate can be chosen by minimizing the response surface directly or by finding a point that achieves the maximum of an acquisition function. We refer to [Conn et al., 2009, Chapter 12] for a review of the management of surrogates for optimization. Kriging has a statistical interpretation that allows to derive confidence bounds for the estimator. This measure of uncertainty of the metamodel is used to design efficient search strategies.

Among direct search methods, we mention:

- Heuristic methods such as genetic algorithms (GA) [Holland, 1975], simulated annealing (SA) [Metropolis et al., 1953] or particle swarm optimization (PSO) [Kennedy and Eberhart, 1995]. These methods define a trajectory in the search space (SA) or the evolution of a population of candidate solutions (GA, PSO) based on probabilistic arguments inspired from biology (Darwin's theory on evolution for GA, social behavior of a bird flock for PSO) or physics (annealing in metallurgy for SA). These methods are often hard to tune as good settings are problem-dependent and not based on theoretical grounds but rather on heuristic considerations. Moreover, these methods may require a large number of evaluations to find an acceptable solution which is not adapted to the expensive framework.

- The DIRECT (DIvding RECTangles) algorithm [Jones et al., 1993] divides the search space into hyperrectangles. The objective function is evaluated at the center of potentially optimal rectangles which are then further subdivided.

- Pattern search methods [Torczon, 1997] generate search points only on a scaled lattice. An iteration starts with a global exploratory search followed by a local poll step. These algorithms enjoy strong theoretical convergence guarantees.

The goal of this part is to investigate blackbox optimization algorithms and to compare their performance both on a comprehensive benchmark and on a small industrial maintenance optimization problem. We choose to focus on one model-based and one direct search method. Kriging has the advantage over other model-based methods to provide a measure of uncertainty of the metamodel. Hence, we start, in Chapter 3, with a bibliographical review of kriging and of the Efficient Global Optimization (EGO) algorithm. For the direct search method, we focus on pattern search methods that are more flexible than DIRECT, thanks to the exploratory search step that can be arbitrarily defined by the user. Chapter 4 is a review of pattern search methods, with a particular focus on the Mesh Adaptive Direct Search (MADS) algorithm [Audet and Dennis, 2006], which has the strongest theoretical convergence results. In Chapter 5, we present an original variant of the EGO algorithm, called EGO-FSSF, that uses a sequential initial space-filling design coupled with a metamodel validation step. The insight is to adapt the size of the initial design in EGO to the difficulty of the problem in order to improve the efficiency of the algorithm. We also present a benchmark of solvers for the maximization of the Expected Improvement within EGO in order to give quantitative arguments for the choice of a solver for this task. Finally, Chapter 6 provides a numerical comparison between EGO, EGO-FSSF and MADS on the COmparing Continuous Optimizers (COCO) platform and on a small industrial case.

3 Overview of kriging and of the EGO algorithm

He who thinks little errs much...

Leonardo da Vinci

Contents

3.1 Introduction

Gaussian process (GP) regression, also known as kriging, is an interpolation method that was initially developed by [Matheron, 1962] for mineral exploration. The term kriging originates from the name of the South-African engineer Danie G. Krige who tried to predict the concentration of ores in deposits from a small number of drillings. This approach has since been popularized and applied in many fields such as numerical simulation [Sacks et al., 1989] or machine learning [Rasmussen and Williams, 2006].

We consider the framework of Problem (2.1). Kriging aims at approximating f on U^{ad} by a metamodel, knowing only its values on a subset $U^{\text{o}} = \{u_1, \dots, u_l\} \subset U^{\text{ad}}$, called the *observation set*. The metamodel can provide a prediction of the value of $f(u)$ for an arbitrary $u \in U^{\text{ad}}$ at a cheap computational cost. More than that, the properties of GPs allow to derive prediction intervals or more generally a prediction distribution for the estimate of $f(u)$. The theory of GP modeling, as well as some practical considerations are exposed in Section 3.2. As the industrial problem driving this thesis is an optimization problem, we are interested in using GP models for optimization purpose. In Section 3.3, we describe the Efficient Global Optimization (EGO) algorithm introduced by [Jones et al., 1998]. EGO uses the GP model and an acquisition function, that takes advantage of the statistical properties of the GP models, to build an efficient sequential evaluation strategy for the minimization of f.

Contributions. This chapter consists in a bibliographical review of kriging and of the EGO algorithm. As GP regression is widely used in industrial contexts, for example in [Marrel et al., 2008, Iooss and Marrel, 2019], we aim at providing a global understanding of the method, as well as some practical implementation considerations. We hope that in the light of this chapter, a practitioner would be able to use GP regression. We also review some theoretical results on the EGO algorithm.

3.2 Overview of Gaussian process regression

This section is a review of GP regression. We start by some general definitions. In §3.2.1, we expose the kriging equations that are at the heart of the theory of GP regression. Then, we focus on some considerations that influence the practical performance of kriging, namely the initial design of experiments, in §3.2.2, and the choice of the covariance function, in §3.2.3. Finally, in §3.2.4, we present some criteria for the assessment of the quality of the regression process. Throughout the manuscript, random variables are denoted with capital bold letters.

Definition 3.1. Let $(\Omega, \mathcal{A}, \mathbb{P})$ be a probability space and U^{ad} be a subset of $\mathbb{R}^d$. A *Gaussian process* on U^{ad} is a collection $\boldsymbol{Z} = \left\{\boldsymbol{Z}_u : \Omega \to \mathbb{R}\right\}_{u \in U^{\text{ad}}}$ of real-valued random variables such that for any finite subset $\{u_1, \dots, u_l\} \subset U^{\text{ad}}$, the vector $(\boldsymbol{Z}_{u_1}, \dots, \boldsymbol{Z}_{u_l})$ is a multivariate Gaussian random vector. The behavior of a GP is completely determined by its *mean function* defined as:

$$\mu : U^{\text{ad}} \ni u \mapsto \mathbb{E}\left(\boldsymbol{Z}_u\right) \in \mathbb{R} \,,$$

and its *covariance function* denoted by k and defined as follows:

$$\begin{aligned} k : U^{\text{ad}} \times U^{\text{ad}} &\to \mathbb{R} \\ (u, u') &\mapsto \operatorname{Cov}(\boldsymbol{Z}_u, \boldsymbol{Z}_{u'}) = \mathbb{E}\left(\left(\boldsymbol{Z}_u - \mu(u)\right)\left(\boldsymbol{Z}_{u'} - \mu(u')\right)\right) \,. \end{aligned}$$

In the kriging framework, we assume that the mean is of the form:

$$\mu(u) = \sum_{j=1}^{p} \beta_j \psi_j(u) = \psi(u)^\top \beta \ ,$$

where $\beta = (\beta_1, \ldots, \beta_p)^\top \in \mathbb{R}^p$ and $\psi = (\psi_1, \ldots, \psi_p)^\top$ is a set of known basis functions from U^{ad} to $\mathbb{R}$.

Definition 3.2. We say that a Gaussian process $\boldsymbol{Z}$ is *stationary* when its mean μ is a constant function and its covariance function satisfies:

$$k(u, u') = k(u - u', 0) \ ,$$

for all $u, u' \in U^{\mathrm{ad}}$, *i.e.* the covariance depends only on the difference $u - u'$ but not on the particular location of u and u'.

The covariance function k specifies the dependence structure of the random variables of the process and controls the smoothness of the functions $u \in U^{\mathrm{ad}} \mapsto \boldsymbol{Z}_u(\omega)$, for $\omega \in \Omega$. In §3.2.3, we will see that the covariance function has a strong influence on the behavior of a GP.

3.2.1 Kriging equations

The core assumption of kriging is that $f : U^{\mathrm{ad}} \to \mathbb{R}$ is the realization of a GP $\boldsymbol{Z}$, that is to say, there exists $\omega \in \Omega$ such that for all $u \in U^{\mathrm{ad}}$, we have:

$$f(u) = \boldsymbol{Z}_u(\omega) \ .$$

Definition 3.3. Assume that the values of f are known on $U^{\mathrm{o}} = \{u_1, \ldots, u_l\} \subset U^{\mathrm{ad}}$. The set U^{o} is called the *observation set*. We introduce the *vector of observations* as:

$$f_l = (f(u_1), \ldots, f(u_l))^\top \ ,$$

and the *Gaussian vector at observation points* defined by:

$$\boldsymbol{Z}_l^o = \left(\boldsymbol{Z}_{u_1}, \ldots, \boldsymbol{Z}_{u_l}\right)^\top \ .$$

The goal of kriging is to predict the value of f on U^{ad} knowing its values only on the observation set U^{o}. Note that for any $u \in U^{\mathrm{ad}}$, we can compute the conditional distribution of the random variable $\boldsymbol{Z}_u$ knowing $\boldsymbol{Z}_l^o = f_l$ denoted by $\left[\boldsymbol{Z}_u \middle| \boldsymbol{Z}_l^o = f_l\right]$. We obtain that it is still a Gaussian distribution. Moreover, its mean $m_l(u)$ and its standard deviation $s_l(u)$ are given by the so-called kriging equations. We derive these equations in two different frameworks:

1. First, the vector β, and therefore the mean μ of the GP, are known. This is simple kriging.

2. Second, the vector β is unknown and estimated from the observations. This is universal kriging or ordinary kriging.

3.2.1.1 Simple kriging

We start with *simple kriging* which is defined as kriging under the assumption that the mean μ of the GP $\boldsymbol{Z}$ is known. The simple kriging equations are given by the following theorem.

Theorem 3.4 (Simple kriging equations)**.** *Let $U^{\mathrm{o}} = \{u_1, \ldots, u_l\}$, f_l and $\boldsymbol{Z}_l^o$ be as in Definition 3.3 and let $u \in U^{\mathrm{ad}}$. Under the assumption that the mean μ of the GP $\boldsymbol{Z}$ is known, the conditional distribution of $\boldsymbol{Z}_u$ given that $\boldsymbol{Z}_l^o = f_l$ is:*

$$\left[\boldsymbol{Z}_u \middle| \boldsymbol{Z}_l^o = f_l\right] \sim \mathcal{N}\left(m_l(u), s_l^2(u)\right) , \tag{3.1}$$

where the kriging mean*, $m_l(u)$, and the* kriging variance*, $s_l^2(u)$, are given by:*

$$m_l(u) = \psi(u)^\top \beta + k_l(u)^\top K_l^{-1} \left(f_l - \Psi\beta\right) , \tag{3.2}$$

$$s_l^2(u) = k(u,u) - k_l(u)^\top K_l^{-1} k_l(u) . \tag{3.3}$$

where $\Psi = \left(\psi_j(u_i)\right)_{i,j}$ is the matrix of explanatory variables for the mean, $K_l = \left(k(u_i,u_j)\right)_{i,j}$ is the covariance matrix of $\boldsymbol{Z}_l^o$ and $k_l(u) = \left(k(u_1,u), \ldots, k(u_l,u)\right)^\top$. The conditional GP characterized by the kriging mean and the kriging variance is referred as the kriging metamodel*.*

Proof. The joint distribution of $\boldsymbol{Z}_l^o$ and $\boldsymbol{Z}_u$, which arises directly from the definition of $\boldsymbol{Z}$ is given by:

$$\begin{pmatrix} \boldsymbol{Z}_l^o \\ \boldsymbol{Z}_u \end{pmatrix} \sim \mathcal{N}_{n+1}\left(\begin{pmatrix} \Psi\beta \\ \psi^\top(u)\beta \end{pmatrix}, \begin{pmatrix} K_l & k_l(u)^\top \\ k_l(u) & k(u,u) \end{pmatrix}\right) ,$$

Then, Equation (3.1) giving the conditional distribution of $\boldsymbol{Z}_u$ given that $\boldsymbol{Z}_l^o = f_l$ is a straightforward consequence of [Eaton, 1983, Proposition 3.13]. □

We take $m_l(u)$ as the prediction of $f(u)$. The *prediction interval* of level α is then:

$$\left[m_l(u) - \Phi^{-1}(1-\alpha/2)s_l(u), m_l(u) + \Phi^{-1}(1-\alpha/2)s_l(u)\right] , \tag{3.4}$$

where Φ is the cumulative distribution function of the standard normal distribution. We make some comments on the kriging mean and the kriging variance:

- The kriging mean $m_l(u)$ is the sum of two terms. The term $\psi(u)^\top\beta$ is the mean of the GP at u. The second term $k_l(u)^\top K_l^{-1}\left(f_l - \Psi\beta\right)$ is proportional to the difference between the observations and the mean of the process. In the case of a stationary process, if we assume that:

  $$k(u-u',0) \to 0 \quad \text{when} \quad \|u-u'\| \to +\infty ,$$

  then,

  $$m_l(u) \to \psi(u)^\top\beta \quad \text{when} \quad \mathrm{dist}(u, U^{\mathrm{o}}) \to +\infty ,$$

  where $\|\cdot\|$ is the Euclidean norm and $\mathrm{dist}(u, U^{\mathrm{o}})$ is the distance between u and the set U^{o}. Thus, the kriging mean converges towards the mean of the process $\psi(u)^\top\beta$ when u is far from the observation points U^{o}.

- The variance term $s_l^2(u)$ is composed of the variance $k(u,u)$ of the GP at u corrected by a term that only depends on the position of the observations points $\{u_1, \ldots, u_l\}$ and not on the value of f at these points.

Figure 3.1 gives an example of kriging prediction. The main advantages of GP regression are the following:

1. Kriging only relies on the assumption that f is the realization of a GP and is therefore a flexible non-parametric regression method. The regression function corresponds to the kriging mean, which is the mean of a conditional GP. Hence, it can be very general and is not constrained to belong to some parametric family, contrary to polynomial regression for instance.

2. In the case where there is no uncertainty at the observation points, the kriging mean $m_l(u)$ interpolates the observations and the variance $s_l(u)$ vanishes at these points. Hence, the prediction of f at a point $u \in U^{\mathrm{o}}$ returns the exact value $f(u)$. This is indeed the case on Figure 3.1.

3. Finally, the uncertainty on the prediction of f at any point $u \in U^{\mathrm{ad}}$ can be quantified with the prediction interval (3.4).

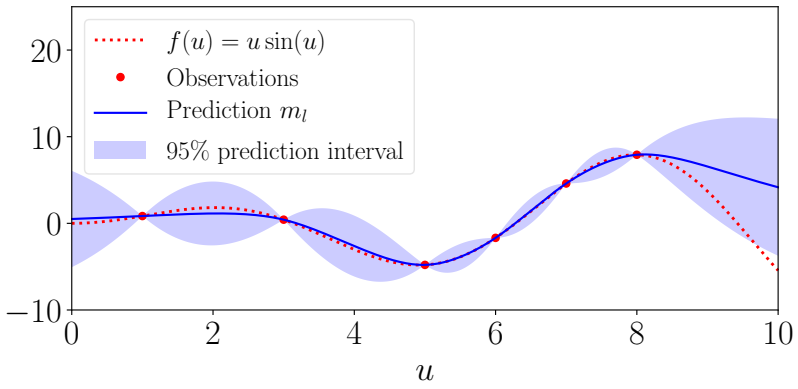


Figure 3.1: Illustrative example of a kriging prediction.

There are other interpretations of GP regression that allow to retrieve the kriging equations (3.2) and (3.3). We can show that the kriging mean is the Best Linear Unbiased Predictor (BLUP) from the observations [Stein, 1999]. To do so, we must work with the centered GP $\boldsymbol{Y} = \left\{\boldsymbol{Y}_u\right\}_{u \in U^{\mathrm{ad}}}$ and the related quantities defined as follows:

$$
\begin{aligned}
\boldsymbol{Y}_u &= \boldsymbol{Z}_u - \mu(u) \,, \\
\boldsymbol{Y}_l^o &= \left(\boldsymbol{Y}_{u_1}, \ldots, \boldsymbol{Y}_{u_l}\right)^\top \,, \\
y(u) &= f(u) - \mu(u) \\
y_l &= f_l - \Psi\beta \,, \\
\bar{m}_l(u) &= m_l(u) - \mu(u) \,.
\end{aligned}
$$

The process $\boldsymbol{Y}$ is a zero mean GP with the same covariance as $\boldsymbol{Z}$. The BLUP of $y(u)$ from the observations y_l is $\lambda^\sharp(u)^\top y_l$ where:

$$
\lambda^\sharp(u) = \min_{\lambda \in \mathbb{R}^n} \mathbb{E}\left(\left(\boldsymbol{Y}_u - \lambda^\top \boldsymbol{Y}_l^o\right)^2\right) \,.
$$

We have:

$$\mathbb{E}\left(\left(\boldsymbol{Y}_u - \lambda^\top \boldsymbol{Y}_l^o\right)^2\right) = k(u,u) - 2\lambda^\top k_l(u) + \lambda^\top K_l \lambda \;, \tag{3.5}$$

leading to $\lambda^\sharp(u) = K_l^{-1} k_l(u)$. The BLUP is then given by:

$$\lambda^\sharp(u)^\top y_l = K_l^{-1} k_l(u)\,(f_l - \Psi\beta) = \bar{m}_l(u) \;.$$

We find that the prediction of $y(u)$ is indeed given by the kriging mean. Plugging the value of $\lambda^\sharp(u)$ into (3.5), we also get that the kriging variance is $s_l(u)$. This interpretation of kriging allows to retrieve the same results as the direct computation with the conditional distributions.

3.2.1.2 Universal and ordinary kriging

Now, consider the framework where the kriging mean is unknown, referred as *universal kriging*. We will also present *ordinary kriging*, that is a special case of universal kriging.

Universal kriging. We give a prediction for $f(u)$ while estimating the value of the vector β at the same time. We use the BLUP approach to get the kriging equations. We are looking for a prediction of the form:

$$m_l(u) = \lambda^\sharp(u)^\top f_l \;, \tag{3.6}$$

where $\lambda^\sharp(u)$ is the solution of:

$$\begin{aligned} &\min_{\lambda\in\mathbb{R}^n} \mathbb{E}\left(\left(\boldsymbol{Z}_u - \lambda^\top \boldsymbol{Z}_l^o\right)^2\right) \;, \\ &\text{s.t. } \Psi^\top \lambda = \psi(u) \;. \end{aligned} \tag{3.7}$$

The equality constraint in (3.7) arises from the fact that we are looking for an unbiased predictor. The resolution can be done by the method of Lagrange multipliers, see for example [Le Gratiet, 2013]. We get:

$$\lambda^\sharp(u) = K_l^{-1} k_l(u) + K_l^{-1}\Psi\left(\Psi^\top K_l^{-1}\Psi\right)^{-1}\left(\psi(u) - \Psi^\top K_l^{-1} k_l(u)\right) \;.$$

We use (3.6) to deduce the universal kriging mean. For the universal kriging variance, we note that under the constraint $\Psi^\top\lambda = \psi(u)$, we have:

$$\mathbb{E}\left(\left(\boldsymbol{Z}_u - \lambda^\top \boldsymbol{Z}_l^o\right)^2\right) = k(u,u) - 2\lambda^\top k_l(u) + \lambda^\top K_l \lambda \;, \tag{3.8}$$

and we substitute the value of $\lambda^\sharp(u)$ in (3.8). Finally, the universal kriging equations write:

$$\begin{aligned} m_l(u) &= \psi(u)^\top \hat{\beta}_l + k_l(u)^\top K_l^{-1}(f_l - \Psi\hat{\beta}_l) \;, \\ s_l^2(u) &= k(u,u) - k_l(u)^\top K_l^{-1} k_l(u) \\ &\quad + \left(\psi(u)^\top - k_l(u)^\top K_l^{-1}\Psi\right)\left(\Psi^\top K_l^{-1}\Psi\right)^{-1}\left(\psi(u)^\top - k_l(u)^\top K_l^{-1}\Psi\right)^\top \;. \end{aligned} \tag{3.9}$$

with $\hat{\beta}_l = \left(\Psi^\top K_l^{-1}\Psi\right)^{-1}\Psi^\top K_l^{-1} f_l$ being the estimation of the vector β. The universal kriging equations are similar to those of simple kriging, the unknown value β being replaced by its estimate $\hat{\beta}_l$. There is also an additional term in the kriging variance compared to the simple kriging case, accounting for the uncertainty on the estimation of β.

Ordinary kriging. This is a special case of universal kriging where the set of basis functions $\psi = (\psi_1, \ldots, \psi_p)^\top$ reduces to only one constant function ψ_0 such that $\psi_0(u) = \gamma \in \mathbb{R}$ for all $u \in U^{\mathrm{ad}}$. The ordinary kriging equations can be derived from (3.9) using that $\Psi = \gamma \mathbf{1}_l$ with $\mathbf{1}_l = (1, \ldots, 1)^\top$:

$$m_l(u) = \hat{\mu}_l + k_l(u)^\top K_l^{-1} \left(f_l - \hat{\mu}_l \mathbf{1}_l\right) , \tag{3.10}$$

$$s_l^2(u) = k(u,u) - k_l(u)^\top K_l^{-1} k_l(u) + \frac{\left(1 - k_l(u)^\top K_l^{-1} \mathbf{1}_l\right)^2}{\mathbf{1}_l^\top K_l^{-1} \mathbf{1}_l} , \tag{3.11}$$

with $\hat{\mu}_l = \hat{\beta}_l \psi_0(u) = \hat{\beta}_l \gamma = \frac{\mathbf{1}_l^\top K_l^{-1} f_l}{\mathbf{1}_l^\top K_l^{-1} \mathbf{1}_l}$. In the remainder of the chapter, we work in the framework of ordinary kriging.

3.2.1.3 A few words on the noisy case

In some situations, given $u \in U^{\mathrm{ad}}$, obtaining the exact value of $f(u)$ is not possible as any attempt to obtain $f(u)$ is corrupted by noise. If f is the result of an experiment, this noise can represent measurement errors. In the case where f is the output of a stochastic simulation code, the noise represents the variability of the response between different runs with the same input data. We address this situation by explaining how kriging can take into account these evaluation errors. We assume that the user has access to l observations of the form:

$$\widetilde{f}(u_i) = f(u_i) + \varepsilon_i, \quad i \in \{1, \ldots, l\} ,$$

where ε_i is a realization of a real Gaussian random variable $\boldsymbol{\varepsilon}_i \sim \mathcal{N}(0, \zeta_i^2)$. The random errors $\boldsymbol{\varepsilon}_1, \ldots, \boldsymbol{\varepsilon}_l$ are independent. They are also independent from the Gaussian process $\boldsymbol{Z}$. Let $\widetilde{\boldsymbol{Z}}_{u_i} = \boldsymbol{Z}_{u_i} + \boldsymbol{\varepsilon}_i$ and $\widetilde{\boldsymbol{Z}}_l^o = \left(\widetilde{\boldsymbol{Z}}_{u_1}, \ldots, \widetilde{\boldsymbol{Z}}_{u_l}\right)^\top$. We derive the kriging equations in the noisy case for the simple kriging framework. Note that:

$$\mathrm{Cov}\left(\widetilde{\boldsymbol{Z}}_{u_i}, \boldsymbol{Z}_u\right) = k(u_i, u) ,$$
$$\mathrm{Cov}\left(\widetilde{\boldsymbol{Z}}_l^o, \widetilde{\boldsymbol{Z}}_l^o\right) = K_l + \Delta_l ,$$

where Δ_l is the following diagonal matrix:

$$\Delta_l = \begin{pmatrix} \zeta_1^2 & & 0 \\ & \ddots & \\ 0 & & \zeta_l^2 \end{pmatrix} .$$

In the case of simple kriging, we have the following joint distribution:

$$\begin{pmatrix} \widetilde{\boldsymbol{Z}}_l^o \\ \boldsymbol{Z}_u \end{pmatrix} \sim \mathcal{N}_{n+1}\left(\begin{pmatrix} \Psi\beta \\ \psi^\top(u)\beta \end{pmatrix}, \begin{pmatrix} K_l + \Delta_l & k_l(u)^\top \\ k_l(u) & k(u,u) \end{pmatrix}\right) .$$

Let $\widetilde{f}_l = \left(\widetilde{f}(u_1), \ldots, \widetilde{f}(u_l)\right)^\top$ be the vector of noisy observations, then we have:

$$\left[\boldsymbol{Z}_u \middle| \widetilde{\boldsymbol{Z}}_l^o = \widetilde{f}_l\right] \sim \mathcal{N}\left(m_l(u), s_l^2(u)\right) ,$$

with

$$m_l(u) = \psi(u)^\top \beta + k_l(u)^\top \left(K_l + \Delta_l\right)^{-1} \left(\widetilde{f}_l - \Psi\beta\right) ,$$
$$s_l^2(u) = k(u,u) - k_l(u)^\top \left(K_l + \Delta_l\right)^{-1} k_l(u) .$$

The kriging mean and the kriging variance are similar as in the noiseless case, except that K_l is replaced by $K_l + \Delta_l$. This also applies in the universal and ordinary kriging frameworks. In the noisy case, we lose the interpolation property of the kriging mean $m_l(u)$ with the observations and the variance $s_l(u)$ is non zero even at the observation points, see Figure 3.2. These features are natural as there is still some uncertainty on the true value of f at the observation points.

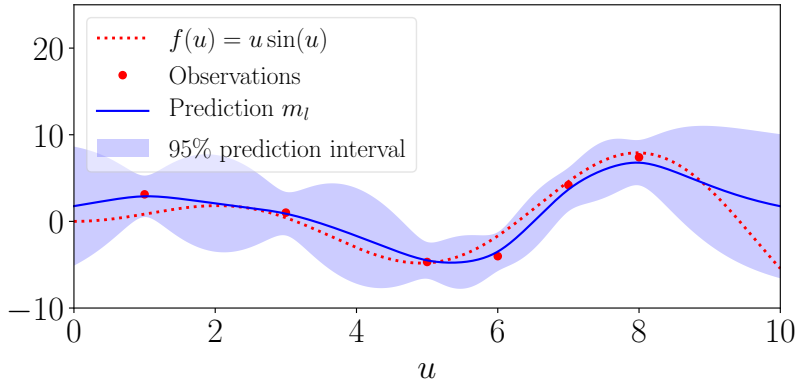


Figure 3.2: Kriging prediction in the noisy case.

The theory of GP regression is based on the kriging equations that have been presented in this section. However, for an efficient implementation of kriging, other considerations must be taken into account. In the following section, we emphasize the importance of the initial observation set for the quality of the metamodel.

3.2.2 Initial design of experiments

The quality of the predictions given by the metamodel depends on the initial design of experiments (DOE), that is, the initial set $U^{\mathrm{o}} = \{u_1, \ldots, u_l\}$ where f is evaluated before constructing the metamodel. We start by an example to illustrate this insight.

3.2.2.1 Illustrative example

For $u = (u^1, u^2) \in U^{\mathrm{ad}} = [-1, 1]^2$, we consider the Ackley function $f : U^{\mathrm{ad}} \to \mathbb{R}$ defined by:

$$f(u) = -20\exp\left(-0.2\sqrt{\frac{1}{2}\sum_{i=1}^{2}(u^i)^2}\right) - \exp\left(\frac{1}{2}\sum_{i=1}^{2}\cos(2\pi u^i)\right) + 20 + \exp(1)\ .$$

Two different initial DOEs with 40 points are constructed on Figure 3.3a and 3.3b, respectively with a crude Monte-Carlo strategy and an optimized Latin Hypercube Sampling (LHS) strategy [Damblin et al., 2013]. The LHS is a sampling method that ensures that all portions of the range of each input variable is represented. With the crude Monte-Carlo design, large regions of the space are empty whereas the LHS design has better space-filling properties. Then, Figures 3.3c and 3.3d represent the metamodel constructed from the corresponding initial DOE and Figures 3.3e and 3.3f show the corresponding kriging variance. We notice that the metamodel resulting from the LHS design is better that the one from the Monte-Carlo design. Moreover, the kriging variance, representing the uncertainty on the prediction, is larger for the metamodel constructed from the Monte-Carlo design. This is due to the fact that some

regions of the space are far from the set of observation points. In these regions, the observations do not give much information to build the prediction, and the metamodel may not be accurate.

3.2.2.2 Brief review of space-filling designs

The importance of the space-filling properties of a design for the predictivity performance of a metamodel is confirmed by theoretical results of [Wang et al., 2020] where upper bounds are derived for the supremum of the estimation error:

$$\sup_{u \in U^{\mathrm{ad}}} |f(u) - m_l(u)| \ .$$

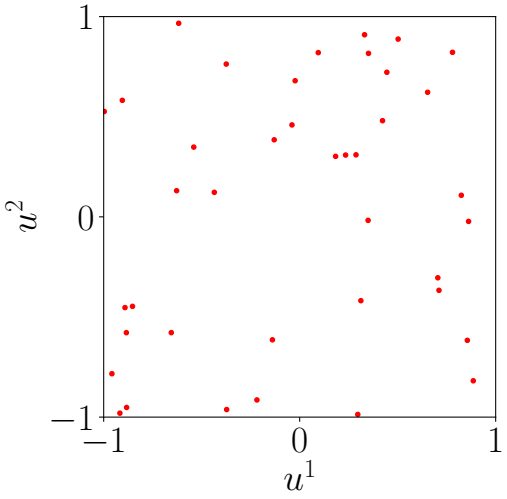
These bounds are increasing function of the *fill distance*:

$$\sup_{u \in U^{\mathrm{ad}}} \mathrm{dist}(u, U^{\mathrm{o}}) \ , \tag{3.12}$$

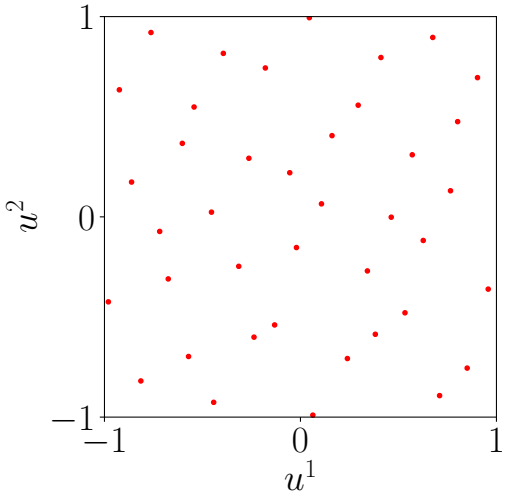
which quantifies the space-filling property of a design. The fill distance is the maximal distance between a point of the domain U^{ad} and the set of observation points U^{o}. The designs that minimize the fill distance, called minimax designs [Johnson et al., 1990], lead to the best theoretical bound for the maximum estimation error.

Many other space-filling designs are considered in the literature, a comprehensive review can be found in [Abtini, 2018]. We quickly mention some popular designs. The maximin design [Johnson et al., 1990] is constructed by maximizing the minimal distance between two points of the design U^{o}. We can also construct a design with the l first points of a low-discrepancy sequence such as the Halton sequence [Halton, 1960] or the Sobol sequence [Sobol, 1967]. These designs ensure that the distribution of the points is close to the uniform distribution. Another important class of designs are stratified designs that consist in splitting the space U^{ad} in multiple strata and drawing a point at random in each stratum. The LHS design [McKay et al., 1979], which is one of the most popular designs for kriging, is an extension of stratified sampling.

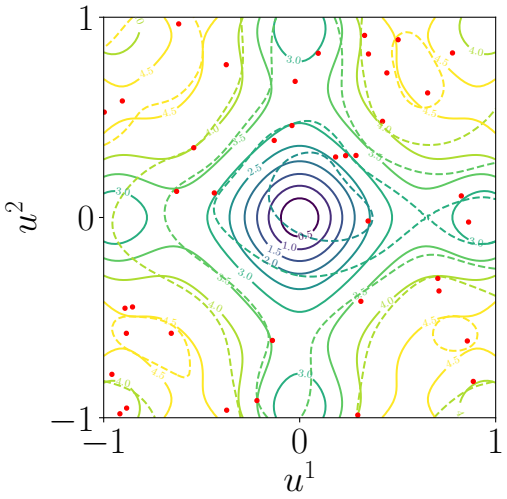
We quickly detail the principle of LHS and some of its refinements. The goal of LHS is to ensure a good repartition of the points when projected on a one-dimensional subspace. Suppose we construct a LHS design with l points. Each of the d input dimensions is divided into l equiprobable intervals. Then, for each input dimension, one sample is generated in each interval. This results in l scalar samples for each dimension. Finally, we randomly combine these scalar samples to obtain l samples of dimension d. However, this procedure may lead to designs with poor space-filling properties. The LHS design can then be optimized with respect to some criterion (minimax or maximin distance, discrepancy) to get these space-filling properties [Damblin et al., 2013]. However, most of LHS designs do not guarantee a good repartition of samples when projected on multi-dimensional subspaces [Damblin et al., 2013]. Robustness to projection on multi-dimensional subspaces is relevant to capture interaction effects between inputs. It is also essential in the case where the metamodel fitting is made on a subspace of the input space. This situation may occur after a screening step has discarded non influent variables, so that the metamodel is fitted only on the subspace of influent variables. Among the criteria tested in [Damblin et al., 2013], only discrepancy optimized LHS achieves robustness to projections on high-dimensional subspaces. Finally, we mention that there is another class of robust designs: the class of maximum projection designs [Joseph et al., 2015].



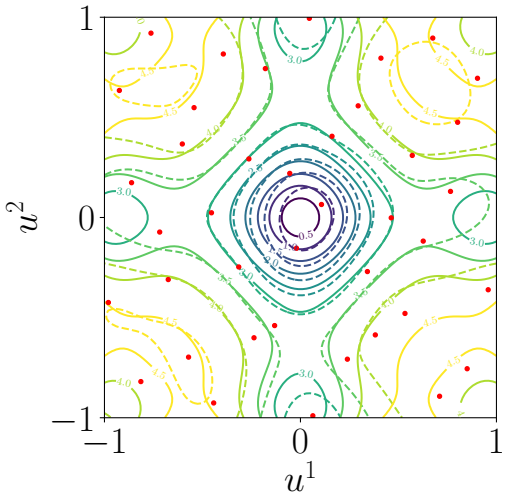
(a) Crude Monte-Carlo design.



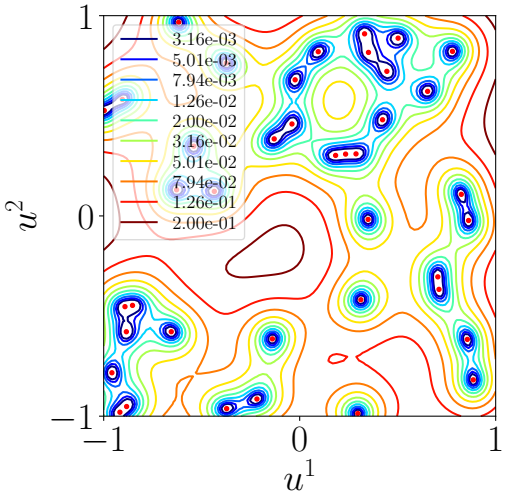
(b) Optimal LHS design.



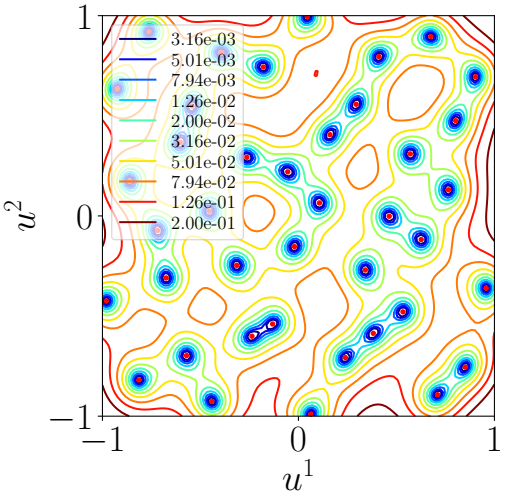
(c) Ackley function (solid lines) and kriging mean m_l (dashed lines) resulting from the Monte-Carlo (MC) design.



(d) Ackley function (solid lines) and kriging mean m_l (dashed lines) resulting from the optimal LHS design.



(e) Kriging variance s_l^2 with the MC design.



(f) Kriging variance s_l^2 with the LHS design.

Figure 3.3: Influence of the initial design of experiments on the quality of the metamodel.

In this section, we have seen that a good space-filling design can improve the quality of the kriging metamodel. The choice of the initial DOE is therefore an important consideration when using GP regression in practice. In the next section, we focus on the choice of the covariance function, which is another aspect that influences the performance of kriging.

3.2.3 Covariance functions

The covariance function can be interpreted as a similarity measure between variables. When the points u and u' are *close*, we expect $\boldsymbol{Z}_u$ and $\boldsymbol{Z}_{u'}$ to be strongly correlated. Conversely, when u and u' are *far away* from each other, the values of $\boldsymbol{Z}_u$ and $\boldsymbol{Z}_{u'}$ are not correlated. In §3.2.1, we have derived the kriging equations assuming that the covariance function k is known. However, k is chosen by the user so that it fits the structure of the observed data. In this section, we highlight the implications of this choice on the properties of the underlying GP. First, we recall that a covariance function satisfies the following properties [Rasmussen and Williams, 2006].

Definition 3.5. Let U be an arbitrary subset of $\mathbb{R}^d$. A kernel is a mapping $k : U \times U \to \mathbb{R}$. The kernel k is said to be an *admissible covariance function* if it satisfies the following conditions:

- k is symmetric, *i.e.* $k(u, u') = k(u', u)$ for all $u, u' \in U$.
- k is positive semi-definite:

$$\forall l \in \mathbb{N},\ \forall u_1, \ldots, u_l \in U,\ \forall a_1, \ldots, a_l \in \mathbb{R},\ \sum_{i=1}^{l} \sum_{j=1}^{l} a_i a_j k(u_i, u_j) \geq 0 \ .$$

  This is equivalent to the fact that for all $l \in \mathbb{N}$ and for all $u_1, \ldots, u_l \in U$, the matrix $K_l = (k(u_i, u_j))_{i,j}$ is positive semi-definite, *i.e.* K_l is a covariance matrix.

3.2.3.1 Choosing a covariance function

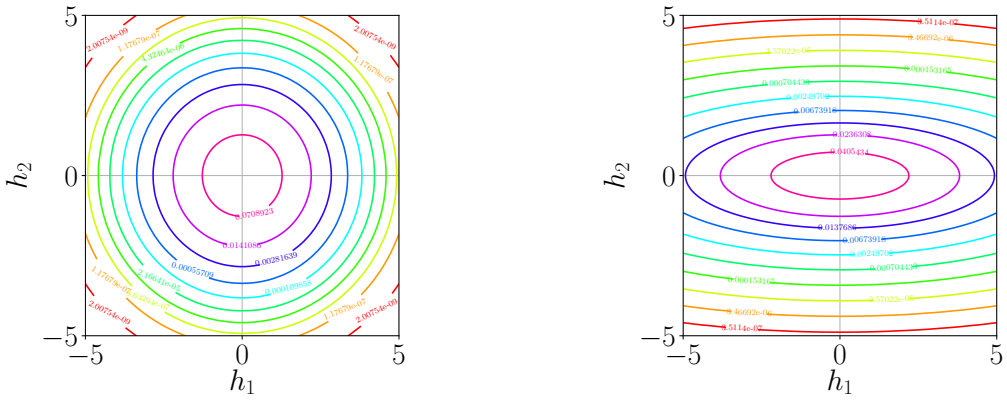
In practice, it is difficult to check that an arbitrary function is positive definite and to do a non-parametric estimation of the kernel [Roustant et al., 2012]. Thus, the choice of the covariance function is done among parametric families which are known to be positive definite.

Definition 3.6. Consider a covariance function of the form:

$$k(u, u') = \sigma^2 r_\theta(u, u'), \quad u, u' \in U^{\text{ad}} \ ,$$

where $\sigma > 0$ controls the variance of the GP and $r_\theta : U^{\text{ad}} \times U^{\text{ad}} \to \mathbb{R}$ is a correlation function parametrized by the vector $\theta = (\theta_1, \ldots, \theta_d) \in \mathbb{R}^d$ with $\theta_j > 0$ for all $j \in \{1, \ldots, d\}$.

- The parameter $\theta_i,\ i \in \{1, \ldots, d\}$, is called the *characteristic length* of the kernel in the i-th direction.
- When the characteristic lengths are the same in every direction, the kernel is *isotropic*, otherwise it is *anisotropic*, see Figure 3.4.



(a) Isotropic kernel: $\theta_1 = \theta_2$.

(b) Anisotropic kernel: $\theta_1 \neq \theta_2$.

Figure 3.4: Gaussian kernels $k(h,0) = \sigma^2 \exp\left(-\frac{h_1^2}{2\theta_1} - \frac{h_2^2}{2\theta_2}\right)$.

- The variables θ and σ^2 are called *hyperparameters* as they are parameters for the construction of the kriging metamodel.

In practice, we choose a parametrized covariance function of the form given in Definition 3.6. The characteristic lengths $\theta = (\theta_1, \ldots, \theta_d)$ give a precise meaning of the words *close* and *far away* as they define the zone of influence of the variable $\boldsymbol{Z}_u$. Figure 3.5 gives examples of realizations of a GP with different characteristic lengths. With a large θ, $\boldsymbol{Z}_u$ and $\boldsymbol{Z}_{u'}$ are highly correlated even when u and u' are far away, whereas with a small θ, $\boldsymbol{Z}_u$ and $\boldsymbol{Z}_{u'}$ are almost not correlated, even when u and u' are close.

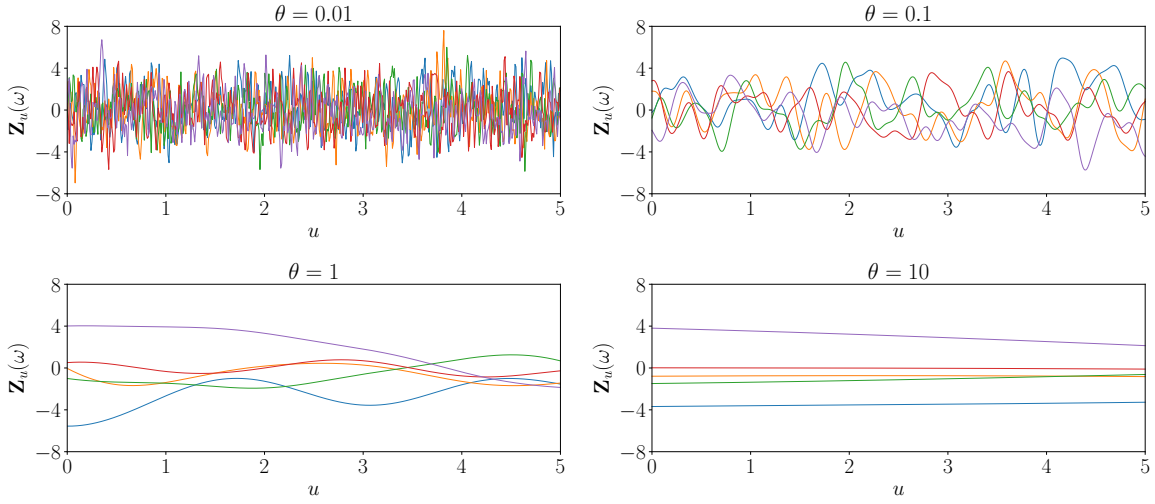


Figure 3.5: Realizations of a Gaussian process with different characteristic lengths θ.

To get admissible covariance functions in large dimensions, we simply take the product of admissible 1D-kernels, resulting in separable kernels. We assume that $U^{\text{ad}} = U_1^{\text{ad}} \times \ldots \times U_d^{\text{ad}}$ with $U_i^{\text{ad}} \subset \mathbb{R}$ for $i \in \{1, \ldots, d\}$. All the kernels that are mentioned below are stationary, *i.e.* they only depend on $h = u - u'$. Then, the structure of the kernels we consider is the following:

$$k(h,0) = k(u-u',0) = \sigma^2 r_\theta(h,0) = \sigma^2 \prod_{j=1}^{d} c_{\theta_j}(h_j) \ ,$$

where $h_j \in U_j^{\text{ad}}$ is the j-th coordinate of h. The function $c_{\theta_j} : U_j^{\text{ad}} \to \mathbb{R}$ is a 1D correlation function. Examples of kernels, from [Rasmussen and Williams, 2006], are given in Table 3.1. The function Γ is the Euler gamma function and K_ν is the modified Bessel function of the second kind of order ν [Abramowitz and Stegun, 1964]. Matérn kernels have a simpler expression when $\nu = p + 1/2$, $p \in \mathbb{N}$ as they can be written as the product of an exponential and a polynomial function of order p. When $\nu = 1/2$, the Matérn kernel corresponds to the exponential kernel and, as $\nu \to \infty$, the Matérn kernel converges to the Gaussian kernel.

Matérn $\nu > 0$	$c_\theta(h) = \frac{2^{1-\nu}}{\Gamma(\nu)} \left(\frac{\sqrt{2\nu}\lvert h\rvert}{\theta} \right)^\nu K_\nu \left(\frac{\sqrt{2\nu}\lvert h\rvert}{\theta} \right)$
Gaussian (Matérn $\nu \to \infty$)	$c_\theta(h) = \exp\left(-\frac{h^2}{2\theta} \right)$
Matérn $\nu = 5/2$	$c_\theta(h) = \left(1 + \frac{\sqrt{5}\lvert h\rvert}{\theta} + \frac{5h^2}{3\theta^2} \right) \exp\left(\frac{-\sqrt{5}\lvert h\rvert}{\theta} \right)$
Matérn $\nu = 3/2$	$c_\theta(h) = \left(1 + \frac{\sqrt{3}\lvert h\rvert}{\theta} \right) \exp\left(\frac{-\sqrt{3}\lvert h\rvert}{\theta} \right)$
Exponential (Matérn $\nu = 1/2$)	$c_\theta(h) = \exp\left(-\frac{h}{\theta} \right)$
Rational quadratic $\alpha > 0$	$c_\theta(h) = \left(1 + \frac{h^2}{2\alpha\theta^2} \right)^{-\alpha}$

Table 3.1: Examples of admissible 1D correlation functions $c_\theta : \mathbb{R} \to \mathbb{R},\ \theta > 0$.

Now that we are able to construct admissible kernels in arbitrary dimension from parametric families, the question is how to choose a kernel that is adapted to our data. To guide this choice, we should note that the covariance function determines the smoothness of the realizations of a GP. Figure 3.6 shows examples of GP realizations with different kernels. With the exponential covariance function, the realizations are only continuous, with the Matérn kernel 3/2 (resp. 5/2) they are $\mathcal{C}^1$-differentiable (resp. $\mathcal{C}^2$-differentiable), and finally with the Gaussian kernel they are $\mathcal{C}^\infty$-differentiable. More generally, the realizations of a GP with a Matérn covariance function of parameter ν are $\lceil \nu \rceil - 1$ differentiable, where $\lceil \nu \rceil$ is the smallest integer greater than or equal to ν [Stein, 1999]. In practice, the choice of a Matérn kernel with $\nu = 3/2$ or $\nu = 5/2$ is justified when the function we try to predict is not smooth.

After the choice of a parametric family for the covariance, we have to compute the hyperparameters θ and σ^2 in order to get the explicit form of the kernel and to be able to use the kriging equations. The hyperparameters computation is usually done by Maximum Likelihood Estimation (MLE). The expression of the likelihood and an efficient algorithm to solve the maximization problem can be found in [Park and Baek, 2001].

In this part, we have argued that it is easier to choose covariance functions from a parametric family and have highlighted that they determine the smoothness of the underlying GP. The next paragraph focuses on numerical considerations and gives some additional arguments to guide the choice of the kernel.

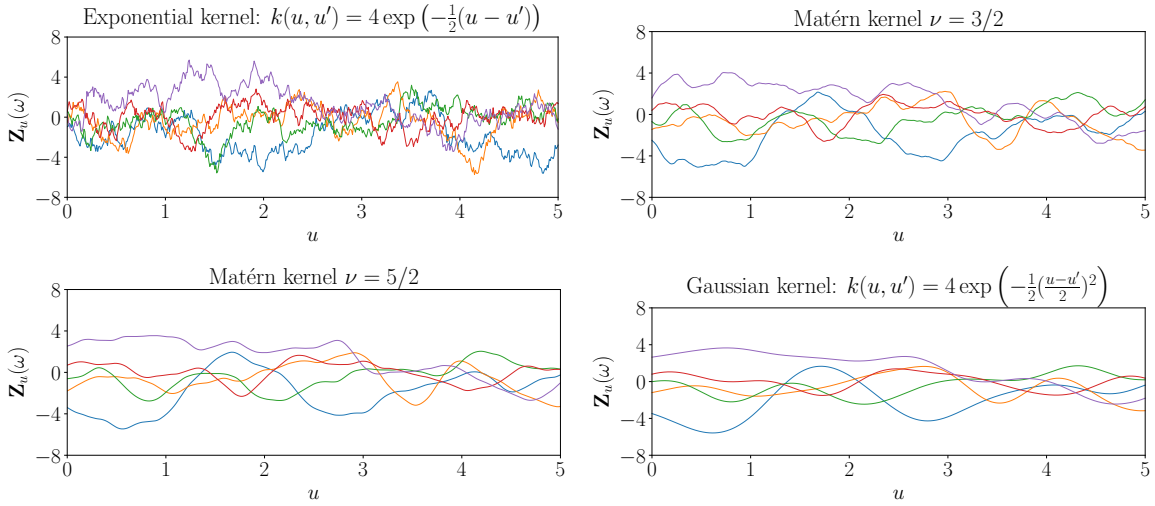


Figure 3.6: Realizations of a Gaussian process with different covariance functions.

3.2.3.2 Numerical considerations

The computation of the kriging equations and of the likelihood for the estimation of hyperparameters involve the inverse of the covariance matrix K_l^{-1}. The difficulty of the numerical computation of K_l^{-1} is linked to the *condition number* of the matrix K_l.

Definition 3.7. Let K_l be a symmetric positive semi-definite matrix. Let $\lambda_{\min}(K_l)$ and $\lambda_{\max}(K_l)$ be respectively the smallest and the largest eigenvalue of K_l. We define the *condition number* of K_l as:

$$\kappa(K_l) = \lambda_{\max}(K_l)/\lambda_{\min}(K_l) \ .$$

When several observation points are close to each other relatively to the characteristic length θ of the kernel, the covariance matrix K_l is ill-conditioned, meaning that $\kappa(K_l)$ is large. This makes the computation of K_l^{-1} numerically difficult. This phenomenon is all the more important when k is a Gaussian kernel as it enforces the smoothness of the realizations of the GP. To overcome this issue, [Stein, 1999] recommends to choose a Matérn kernel with $\nu = 3/2$ or $\nu = 5/2$.

We also mention the work of [Mohammadi, 2016, Chapter 3] which extensively studies two methods to avoid the ill-conditioning of the covariance matrix, namely the *pseudo-inverse* and the *nugget* regularizations. The pseudo-inverse regularization consists in replacing K_l^{-1} by its Moore-Penrose pseudo-inverse [Penrose, 1955] in the kriging equations. The nugget regularization consists in adding a positive value τ^2 to the diagonal of the covariance matrix K_l. This amounts to consider that the observations are perturbed with an additive Gaussian noise $\mathcal{N}(0, \tau^2)$, similarly as in §3.2.1.3. With the nugget, the interpolation property of the kriging mean with the data points is lost and the variance is non-zero at these points. [Mohammadi, 2016] argues that the pseudo-inverse regularization is equivalent to the nugget regularization when τ is sufficiently small. In the sequel, we use the nugget regularization given its simplicity and the fact that it is implemented in the software OpenTURNS [Baudin et al., 2017] that is used for the numerical experiments.

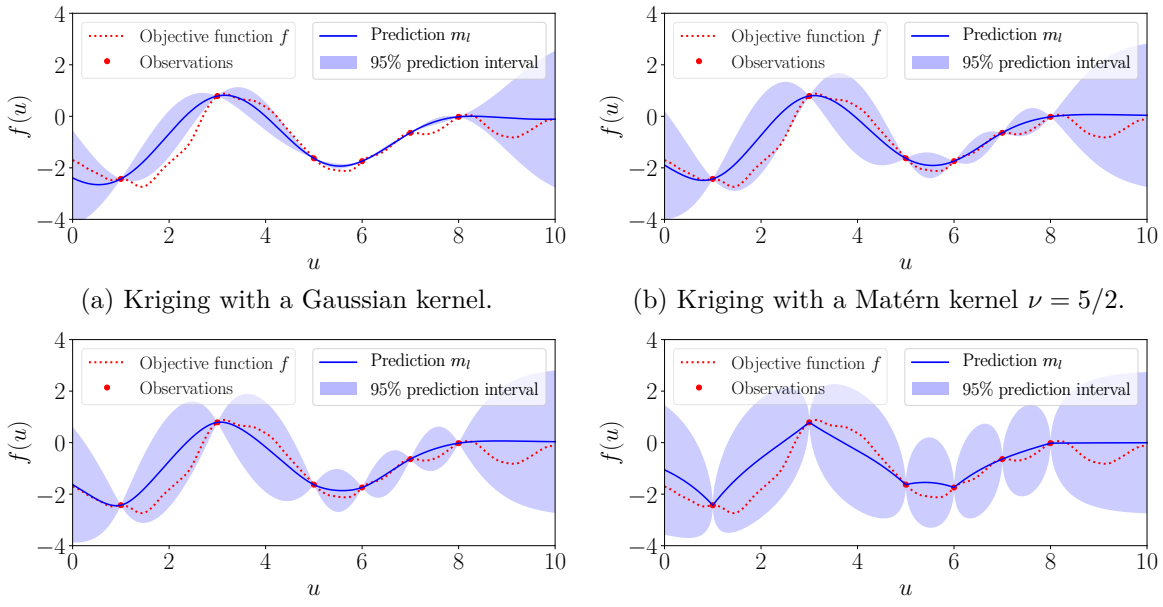
These numerical considerations give additional arguments in favor of Matérn kernels and justify the addition of a nugget in the covariance function, even in the case where the observations are not corrupted by noise. However, even if we aim at choosing a

covariance function that represents well the dependence structure of our data, it is difficult to ensure that this is effectively the case in practice. This is why, we focus on the effect of covariance misspecification in the next paragraph.

3.2.3.3 Effect of covariance misspecification

In order to build a kriging metamodel, the covariance function k is chosen by the user in a parametric family and then the hyperparameters θ and σ^2 are estimated by maximum likelihood. It is interesting to understand the effects of a bad covariance choice on the kriging metamodel.

A detailed study in this direction is done in [Stein, 1999]. One of the main conclusions is that a misspecification of the covariance structure has a greater impact on the computation of the kriging variance s_l^2 than on the efficiency of the prediction given by m_l. More precisely, the smoother the covariance function, the more optimistic the kriging variance. For example, in Figure 3.7, the true covariance structure of the objective function is a Matérn kernel with $\nu = 5/2$. When the kriging metamodel is computed using a Gaussian kernel (Figure 3.7a), the kriging variance, is overoptimistic *i.e.* smaller than the true one. On the other hand, if a Matérn covariance with $\nu = 3/2$ (Figure 3.7c) or an exponential kernel is used (Figure 3.7d), the kriging variance is too conservative, which is still more desirable than being overoptimistic. This is why [Stein, 1999] recommends to avoid Gaussian kernels and to use Matérn kernels instead.



(a) Kriging with a Gaussian kernel.

(b) Kriging with a Matérn kernel $\nu = 5/2$.

(c) Kriging with a Matérn kernel $\nu = 3/2$.

(d) Kriging with a Matérn kernel $\nu = 1/2$.

Figure 3.7: Kriging with different kernels. The objective function is a realization of a Gaussian process with a Matérn covariance function $\nu = 5/2$. We observe that the prediction interval is too optimistic when the regression is done with the Gaussian kernel and too pessimistic with the Matérn kernel $\nu = 3/2$ or $\nu = 1/2$.

Regarding the efficiency of the kriging prediction m_l, upper and lower bounds are derived in [Tuo and Wang, 2019]. The results are derived in the case where the underlying function is a random realization of a Gaussian process with a Matérn kernel of parameter ν_0. The kriging metamodel is constructed assuming a Matérn covariance of

parameter ν. Under some assumptions on the space-filling properties of the DOE, we have, as the size l of the design tends to infinity:

$$\sup_{u \in U^{\text{ad}}} |f(u) - m_l(u)| = \begin{cases} \mathcal{O}\left(n^{-\nu/d}\sqrt{\log(n)}\right) & \text{if } \nu \leq \nu_0 \ , \\ \mathcal{O}\left(n^{-\nu_0/d}\sqrt{\log(n)}\right) & \text{if } \nu > \nu_0 \ , \end{cases}$$

the rate for $\nu > \nu_0$ being optimal. This result confirms the insight of [Stein, 1999] in that the error of kriging prediction is not dramatically affected by covariance misspecification. In fact, if the covariance function is oversmooth ($\nu > \nu_0$), the predictive performance of the kriging metamodel do not deteriorate. Non-asymptotic results depending on space-filling metrics of the DOE are also given in [Tuo and Wang, 2019]. These theoretical bounds confirm the importance of the choice of the DOE for the predictivity performance of the kriging metamodel, as already highlighted in §3.2.2.

In summary, oversmooth covariance functions give good prediction performance but lead to an overoptimistic evaluation of the uncertainty of this estimation. On the other hand, undersmooth covariance functions give slightly less efficient predictions of the objective function and lead to a conservative estimation of the uncertainty, which in our personal view is much more reassuring than the opposite. In light of these observations and of the arguments already given in §3.2.3.1 and §3.2.3.2, it seems that using a Matérn kernel with $\nu = 3/2$ or $\nu = 5/2$ is a good generic choice. The value of ν may be adjusted by the user if some properties on the smoothness of the objective function are known.

3.2.4 Metamodel validation

Recall that in kriging, the goal is to predict the value of a function $f : U^{\text{ad}} \to \mathbb{R}$ from its values on an observation set U^{o}. After the study of space-filling designs (§3.2.2) and of covariance functions (§3.2.3), we are able to compute the kriging equations, given by Theorem 3.4, meaning that we have constructed a metamodel for the function f. In this section, we focus on metamodel validation, which is a process that allows to evaluate the quality of the regression. The validation step is essential when the metamodel is used for prediction or sensitivity analysis, especially for critical industrial applications such as nuclear safety in [Iooss et al., 2010].

3.2.4.1 The predictivity coefficient

Definition 3.8. Let $\{v_1, \ldots, v_p\} \subset U^{\text{ad}}$ be a test sample that is disjoint from the observation set U^{o}. We define the predictivity coefficient Q^2 as:

$$Q^2 = 1 - \frac{\sum_{i=1}^{p} \left(f(v_i) - m_l(v_i)\right)^2}{\sum_{i=1}^{p} \left(f(v_i) - \frac{1}{p}\sum_{j=1}^{p} f(v_j)\right)^2} \ ,$$

The predictivity coefficient is used to assess the metamodel accuracy. The closer to one the predictivity coefficient, the better the metamodel accuracy. Note that Q^2 can be negative, meaning that the mean of the observations is a better estimator than the metamodel estimation. The test sample can be drawn at random with a crude Monte-Carlo strategy. A natural question that arises is how to choose the number p of samples to get an accurate value for Q^2. Moreover, the computation of the predictivity coefficient on the test sample requires some new evaluations of f. When the function

f is the output of a time consuming computer code, it may be impossible to use a sufficiently large test sample. This is why in practice, the predictivity coefficient is computed by K-fold cross validation [Hastie et al., 2001]. More precisely, we use the leave-one-out cross validation:

$$Q^2 = 1 - \frac{\sum_{i=1}^{n} \left(f(u_i) - m_{-i}(u_i)\right)^2}{\sum_{i=1}^{n} \left(f(u_i) - \frac{1}{n}\sum_{j=1}^{n} f(u_j)\right)^2} ,$$

where m_{-i} is the kriging mean of the metamodel constructed from the set of observations $U^{\mathrm{o}} \setminus \{u_i\}$. The idea of leave-one-out cross validation is to construct, for each $i \in \{1, \ldots, l\}$, a metamodel without taking into account the i-th observation $f(u_i)$ as it will be used for validation. The kriging mean $m_{-i}(u_i)$ is used as a prediction of $f(u_i)$. As $f(u_i)$ is known, we can then compute the predictivity coefficient without using new evaluations of f. This method involves the construction of l different metamodels. However, the kriging equations, given by Theorem 3.4, involve the covariance function, for which the hyperparameters θ and σ^2 are computed by MLE (see §3.2.3), which can be computationally expensive. In this case, it is possible to construct the l metamodels while keeping the same hyperparameters as in the metamodel constructed from U^{o}. These fixed hyperparameters should be close to the optimal hyperparameters of each metamodel as the sets U^{o} and $U^{\mathrm{o}} \setminus \{u_i\}$ do not differ that much.

Note that [Iooss et al., 2010] warns that the test sample method may provide an optimistic Q^2 if the test sample is too small whereas the cross validation method may provide a pessimistic Q^2. The authors provide a sequential validation design that aims at minimizing the number of necessary points in the test sample method to capture the right metamodel predictivity. The idea consists in choosing test points in the unfilled zone of the training sample. This method gives better results than cross validation but is more computationally demanding.

In the thesis, our goal is to use GP regression for optimization purpose. Hence, the metamodels are not used directly for estimation or sensitivity analysis, they are rather just part of an optimization procedure. For the numerical experiments of Chapter 6, reducing the computational effort is the main concern. Thus, the computation of the predictivity coefficient will be done with the leave-one-out cross validation method.

3.2.4.2 The predictive variance adequation

Definition 3.9. Let $\{v_1, \ldots, v_p\} \subset U^{\mathrm{ad}}$ be a test sample that is disjoint from the observation set U^{o}. We define the Predictive Variance Adequation (PVA) [Bachoc, 2013]:

$$\mathrm{PVA} = \left| \log_{10} \left(\frac{1}{p} \sum_{i=1}^{p} \frac{\left(f(v_i) - m_l(v_i)\right)^2}{s_l^2(v_i)} \right) \right| ,$$

where $\log_{10}$ is the logarithm with base 10.

The PVA is used to evaluate the quality of the kriging variance given by the metamodel. The smaller the PVA, the better, because this means that the kriging variance is of the same order as the prediction errors, and hence the prediction intervals are reliable. The logarithm is used to equally weight underestimations and overestimations of the kriging variance. A PVA equals to 1 means that we underestimate or overestimate the prediction errors by a factor 10. For the same reasons as for the predictivity coefficient, the PVA can be estimated by leave-one-out cross validation:

$$\mathrm{PVA} = \left| \log_{10} \left(\frac{1}{n} \sum_{i=1}^{n} \frac{\left(f(u_i) - m_{-i}(u_i)\right)^2}{s_{-i}^2(u_i)} \right) \right| .$$

where m_{-i} and s^2_{-i} are respectively the kriging mean and the kriging variance of the metamodel constructed from the set of observations $U^{\mathrm{o}} \setminus \{u_i\}$.

The predictivity coefficient Q^2 and the PVA are complementary indicators of the quality of a metamodel, respectively to quantify the performance in terms of prediction of the function f and in terms of reliability of the prediction intervals.

3.3 The Efficient Global Optimization algorithm

In Section 3.2, we have presented the theory of GP regression, that allows to predict the value of a function $f : U^{\mathrm{ad}} \to \mathbb{R}$, knowing only the value of f on an observation set U^{o}. In Part I of the thesis, the goal is to tackle Problem (2.1): we aim at finding the minimum of f, which is considered to be a blackbox. In this section, we describe the Efficient Global Optimization (EGO) algorithm that exploits the kriging predictions for optimization purpose. The general idea is to use the predictions given by the metamodel to smartly choose the evaluation points of the objective function f.

In §3.3.1, we give the general structure of the EGO algorithm. In §3.3.2, we focus on the choice of the infill criterion, which defines the strategy to *smartly* choose the sequence of evaluation points in EGO. The infill criterion has therefore a great influence on the behavior of the algorithm. Then, in §3.3.3, we review the few available theoretical results for the EGO algorithm. Finally, in §3.3.4, we consider a local optimization step after the EGO iterations in order to improve the efficiency of the algorithm.

3.3.1 Description of the EGO algorithm

The EGO algorithm is described by Algorithm 1. We assume that we are in the ordinary kriging framework, so the computation of the metamodel at each iteration is done with Equations (3.10) and (3.11). The termination criterion is a maximum number M of evaluations of the objective function f.

Algorithm 1 General description the EGO algorithm

Step 0. Choose a class of covariance functions parametrized by the hyperparameters θ and σ^2 (see §3.2.3). Choose an infill criterion (see §3.3.2).

Step 1. (*Initial design step*) Let $U^{\mathrm{o}} = \{u_1, \ldots, u_l\} \subset U^{\mathrm{ad}}$ be an initial DOE (§3.2.2).

(a) Evaluate f on U^{o}.

(b) Estimate the hyperparameters θ and σ^2 with the maximum likelihood.

(c) Compute the initial metamodel with (3.10) and (3.11).

Step 2. (*Infill step*) For $i = l + 1, \ldots, M$:

(a) Choose a new evaluation point $u_i \in U^{\mathrm{ad}}$ by optimizing the infill criterion. Let $U^{\mathrm{o}} \leftarrow U^{\mathrm{o}} \bigcup \{u_i\}$ and evaluate $f(u_i)$.

(b) Update the estimation of hyperparameters θ and σ^2 with the maximum likelihood.

(c) Update the metamodel with (3.10) and (3.11).

Return. The best evaluated value $\min_{u \in U^{\mathrm{o}}} f(u)$.

3.3.2 Infill criterion

The sequence of evaluation points is determined by the optimization of an *infill criterion*, also called *acquisition function*. This criterion uses information from the current metamodel to quantify the interest of evaluating any candidate point. Depending on the infill criterion, the behavior of the EGO algorithm can range from pure exploration to intensive evaluations near the current best solution (exploitation). We give a brief overview of common infill criteria with a focus on the EI which is the most popular.

3.3.2.1 Minimizing the kriging mean

A first naive idea is to choose the $(l+1)$-th evaluation point as the minimizer of the kriging mean:

$$u_{l+1} \in \arg\min_{u \in U^{\mathrm{ad}}} m_l(u) \ .$$

However, [Jones, 2001] shows that this criterion is not efficient as the sequence of evaluation points could be stuck near a local minimum. The flaw of this criterion is that it does not take into account the uncertainty of the metamodel. We must choose an infill criterion that uses both the information on the kriging mean and on the uncertainty of the prediction, given by the kriging variance, so that the algorithm is more exploratory. This criterion should not be used in practice.

3.3.2.2 The Expected Improvement (EI) criterion

The EI criterion is defined in [Jones et al., 1998].

Definition 3.10. Let $\boldsymbol{Z} = \{\boldsymbol{Z}_u\}_{u \in U^{\mathrm{ad}}}$ be a GP and $f_l^\sharp = \min_{1 \le i \le l} f(u_i)$ be the best function value on an observation set $U^{\mathrm{o}} = \{u_1, \dots, u_l\} \in U^{\mathrm{ad}}$.

1. We define the *improvement* as:

$$\boldsymbol{I}_u^l = \left(f_l^\sharp - \boldsymbol{Z}_u\right)^+, \quad u \in U^{\mathrm{ad}} \ ,$$

   where $(\cdot)^+ = \max(\cdot, 0)$. The improvement $\boldsymbol{I}_u^l$ is a random variable. If $\boldsymbol{Z}_u$ is larger than the best observation, there is no improvement. Otherwise, the improvement is the difference between $\boldsymbol{Z}_u$ and the best observation.

2. The *Expected Improvement* is defined as the expectation of the improvement conditionally to the observations:

$$\begin{aligned} EI_l(u) &= \mathbb{E}\left(\boldsymbol{I}_u^l \;\middle|\; \boldsymbol{Z}_l^o = f_l\right) \ , \\ &= \mathbb{E}\left(\left(\min_{1 \le i \le l} f(u_i) - \boldsymbol{Z}_u\right)^+ \;\middle|\; \boldsymbol{Z}_l^o = f_l\right) \ , \end{aligned}$$

   where $\boldsymbol{Z}_l^o$ and f_l have been introduced in Definition 3.3.

In the EGO algorithm, the $(l+1)$-th point of evaluation of f is chosen as a maximizer of the EI:

$$u_{l+1} \in \arg\max_{u \in U^{\mathrm{ad}}} EI_l(u) \ .$$

The EI always vanishes at the observation points U^{o} and is always strictly positive on $U^{\mathrm{ad}} \setminus U^{\mathrm{o}}$. Thus, f is never evaluated twice at the same point. Note also that the EI is increasing with m_l and s_l. Using the computational properties of GPs, we can derive an explicit formula for the EI [Jones et al., 1998]:

$$EI_l(u) = \left(f_l^\sharp - m_l(u)\right) \Phi\left(\frac{f_l^\sharp - m_l(u)}{s_l(u)}\right) + s_l(u)\phi\left(\frac{f_l^\sharp - m_l(u)}{s_l(u)}\right) ,$$

where Φ and ϕ are respectively the cumulative distribution function and the probability density function of the standard normal distribution. The popularity of the EI is partly due to the availability of the explicit formula, which allows for gradient computation.

Let us illustrate the features of the EI on an example. The upper subplot of Figure 3.8 shows the objective function f (dotted curve), the kriging metamodel (in blue), and the associated EI (green curve). On the lower subplot, f is evaluated at the point that maximizes the EI, the metamodel is updated accordingly and the new EI function is plotted. The EI takes high values in the regions where the kriging mean is low and the prediction intervals are wide, indeed these regions are the ones where the objective function can be potentially improved. The EI function is multimodal, *i.e.* it has multiple local maxima. Moreover, outside these local maxima, the EI is very flat and is almost zero, but still strictly positive outside the observation points. These characteristics make the maximization of the EI a challenging problem, even though an analytical form of the function and the gradients are available. The choice of an algorithm for the maximization of the EI is the subject of a benchmark in Section 5.3.

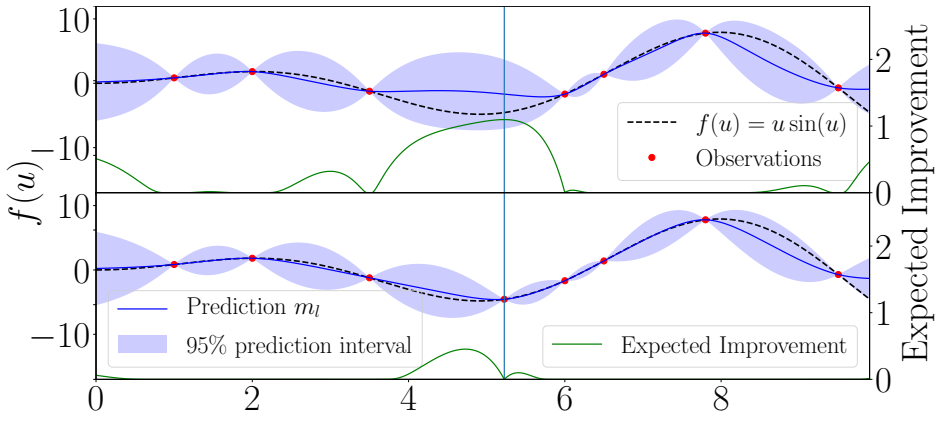


Figure 3.8: Metamodel update after the evaluation at a point that maximizes the EI.

3.3.2.3 Other existing criteria

Other choices for the infill criterion are possible. [Srinivas et al., 2010] casts the GP optimization problem as a multi-armed bandit problem and uses an upper-confidence bound (for a maximization problem) as acquisition function. This algorithm is referred as GP-UCB. For a minimization problem, we consider the lower-confidence bound:

$$u_{l+1} \in \operatorname*{arg\,min}_{u \in U^{\mathrm{ad}}} m_l(u) - \sqrt{\rho_l} s_l(u) ,$$

where $\rho_l \geq 0$ influences the exploration-exploitation tradeoff. Values for ρ_l are given in [Srinivas et al., 2010] so as to minimize regret bounds in different contexts. The use of upper or lower confidence bounds was already suggested by [Cox and John, 1992] in a simplified version (ρ_l is not iteration dependent and tuned by the user).

The EI and GP-UCB use the prediction of f to define the trade off between exploiting the kriging mean and exploring regions with large kriging variance. Another class of infill criteria are information-based acquisition functions. The goal is instead to maximize the information on the location of a global minimizer of f, by minimizing the entropy of the predictive distribution of $f(u)$, or of some related quantity. Examples include the Stepwise Uncertainty Reduction strategy [Villemonteix et al., 2009] or the Predictive Entropy Search [Hernández-Lobato et al., 2014]. These strategies are however more difficult to implement in practice as they rely on numerical integration whereas the EI or the lower-confidence bound can be computed in closed form.

3.3.2.4 Choice of the infill criterion within the thesis

Comparing the performance of infill criteria is out of the scope of this work, but many studies are done in this purpose. In [Picheny et al., 2013], the authors conclude that several criteria have similar performance and that the choice *may be based on user preference without a critical deterioration of performance*. In [Rehbach et al., 2020], the EI is compared with kriging mean minimization. Interestingly, this study concludes that for the expensive setting in high dimension, the exploratory behavior of the EI may be less efficient than the greedy minimization of the kriging mean. The authors acknowledge that EGO with a direct minimization of the kriging mean can be stuck in a local optimum, especially in low dimension, but identify scenarios where this criterion might be preferred over the EI. We also mention the study in [Talgorn et al., 2015], which compares several EI-based infill criteria that are adapted to the case of a constrained optimization problem.

Nevertheless, in Chapters 5 and 6, we use the EI criterion as it is frequently used among practitioners: the EI is the most common default infill criterion in surrogate-based optimization softwares [Rehbach et al., 2020].

So far, we have tackled all the essential aspects for a basic implementation of the EGO algorithm: the choice of the initial DOE §3.2.2, the covariance function §3.2.3 and the infill criterion §3.3.2. In the next section, we focus on some theoretical aspects of the algorithm.

3.3.3 Theoretical considerations

The EGO algorithm has become very popular for industrial applications in the expensive blackbox setting, yet only a few theoretical results are available. We give an overview of these results as they may help to understand the behavior of the algorithm besides their own theoretical interest.

A first convergence result is given by [Vazquez and Bect, 2010] for EGO with the EI and fixed mean and covariance functions, *i.e.* in the case of simple kriging and with fixed values of the hyperparameters θ and σ^2 along the iterations. Under an assumption on the covariance function called the *no-empty-ball* property, the algorithm converges to the optimum for $\mathbb{P}$-almost all continuous functions, where $\mathbb{P}$ is the prior distribution of the Gaussian process $\boldsymbol{Z}$. The no-empty-ball property is defined as follows.

Definition 3.11. [Vazquez and Bect, 2010, Definition 3] A Gaussian process $\boldsymbol{Z}$, or equivalently its covariance function k, has the *no-empty-ball* property if for all sequences $\{u_l\}_{l\in\mathbb{N}}$ in U^{ad} and all $v \in U^{\mathrm{ad}}$, the following statements are equivalent:

(i) v is an adherent point of the set $\{u_l, l \in \mathbb{N}\}$,

(ii) $s_l^2(v) \to 0$ when $l \to +\infty$, where $s_l^2(v)$ is the kriging variance, computed by (3.3).

For continuous covariance functions (such as the examples of Table 3.1), (i) always implies (ii). The other implication means that if $s_l^2(v)$ goes to 0 for some v, then necessarily, for all $\varepsilon > 0$, there is an observation in a ball of radius ε centered in v, *i.e.* there is *no empty ball* centered in v. The Matérn covariance functions have the no-empty-ball property but this is not the case of the Gaussian kernel. Hence, the convergence of EGO is not ensured with the Gaussian covariance function. In fact, [Yarotsky, 2013] even exhibits an explicit example where EGO with a Gaussian covariance function does not converge. This observation is another argument to avoid using Gaussian kernels in EGO besides the numerical difficulties that have been highlighted in §3.2.3.

The result of [Vazquez and Bect, 2010] holds only with fixed hyperparameters. However, in practice, the hyperparameters are often estimated from the observations. In this case, [Bull, 2011] shows that the algorithm may not converge. Nevertheless, under some additional assumptions, [Bull, 2011] proves a convergence rate of $\mathcal{O}(l^{-1/d})$ for EGO, where d is the dimension of the input space.

These theoretical convergence results allow to justify the use of EGO in practice as, in some settings, it will asymptotically get close to a global optimizer. However, in the expensive blackbox framework, only a limited budget is used, so the behavior of EGO may not be well-described by asymptotical results. The performance of EGO depends heavily on the user-defined settings of the algorithm (initial DOE, covariance function, infill criterion, fixed or estimated hyperparameters). The choice of a particular setting for EGO is based more upon empirical considerations than theoretical grounds.

3.3.4 An additional local optimization step for EGO

By design, the EGO algorithm is an exploratory algorithm as, asymptotically, it will tend to reduce the uncertainty on the prediction at every point of the space (except if we use an infill criterion that does not take into account the uncertainty, which is not recommended in practice, see for instance §3.3.2.1). This space-filling behavior is confirmed by the theoretical result of [Vazquez and Bect, 2010], given in §3.3.3, which states that under some mild assumptions, the EGO algorithm produces a dense sequence of evaluation points in the input domain. The EGO algorithm is then efficient in detecting regions of interest for the minimization of the objective function but is not designed to find the precise location of a local minimum inside these regions. This behavior was already pointed out in [Mohammadi, 2016, Chapter 4] where EGO is efficiently coupled with the Covariance Matrix Adaptation Evolution Strategy (CMA-ES) algorithm [Hansen and Ostermeier, 1996]. In the same fashion, we allocate a proportion $0 < p_{\text{local}} < 1$ of the overall evaluation budget to a local search step that is done with the L-BFGS algorithm [Nocedal, 1980] after the EGO iterations. The EGO algorithm coupled with L-BFGS is summarized in Algorithm 2, where the maximum number of evaluations of f is set to $M \in \mathbb{N}$.

3.4 Conclusion

In this chapter, we have presented an overview of GP regression and of the EGO algorithm, which is adapted for blackbox optimization. We have focused on the theory of GP modeling and on many practical aspects for the implementation of kriging, such

Algorithm 2 The EGO algorithm with a final local optimization

Step 0. Choose an infill criterion and a class of covariance functions parametrized by the hyperparameters θ and σ^2. Let $0 < p_{\text{local}} < 1$ be the proportion of the budget to allocate to the local search.

Step 1. (*Initial design step*) Let $U^{\text{o}} = \{u_1, \ldots, u_l\} \subset U^{\text{ad}}$ be an initial DOE.

(a) Evaluate f on U^{o}.

(b) Estimate the hyperparameters θ and σ^2 by maximum likelihood.

(c) Compute the initial metamodel with (3.10) and (3.11).

Step 2. (*Infill step*) For $i = l+1, \ldots, (1-p_{\text{local}})M$:

(a) Choose a new evaluation point $u_i \in U^{\text{ad}}$ by optimizing the infill criterion. Let $U^{\text{o}} \leftarrow U^{\text{o}} \bigcup \{u_i\}$ and evaluate $f(u_i)$.

(b) Update the estimation of hyperparameters θ and σ^2 by maximum likelihood.

(c) Update the metamodel with (3.10) and (3.11).

Step 3. (*Final local search*) Run a local optimization with L-BFGS starting from the current best point $\arg\min_{u \in U^{\text{o}}} f(u)$ with the remaining evaluation budget $p_{\text{local}}M$.

Return. The best iterate of the L-BFGS algorithm.

as the choice of the initial DOE and of the covariance function. It comes out that – without prior information on the function we wish to optimize – using an initial optimized LHS design and a Matérn covariance function seems to be a relevant choice. Theoretical arguments of §3.2.3.3 also support this choice. We emphasize that the validation of the metamodel is essential to ensure that the kriging predictions are of good quality. In an optimization purpose, we have described the EGO algorithm, that takes advantage of the kriging predictions to build a sequence of evaluation points of the objective function. The behavior of EGO is controlled by an infill criterion. We have reviewed several existing infill criteria and have justified that the EI is a relevant choice. We also consider a version of EGO coupled with a final optimization step to find the precise location of a minimum once a region of interest is identified.

We aim at applying EGO to a small industrial maintenance optimization problem in Chapter 6, where it will be compared to the MADS algorithm presented in Chapter 4. The understanding of kriging and of the numerical and theoretical aspects developed in this chapter provides the keys for an informed choice of the settings of EGO. Before the industrial application of EGO, we introduce, in Chapter 5, a new variant to adaptively choose the size of the initial DOE, which should make the algorithm more efficient than with the fixed-size initial DOE used in this chapter.

4 Direct search algorithms

Knowledge is a process of piling up facts; wisdom lies in their simplification.

Martin Henry Fischer

Contents

4.1 Introduction to direct search algorithms

Similarly as in Chapter 3, we consider the problem of minimizing an expensive blackbox function $f : U^{\text{ad}} \to \mathbb{R}$ with $U^{\text{ad}} \subset \mathbb{R}^d$. In this chapter, we focus on direct search methods that gather algorithms sharing the following characteristics [Wright, 1996]:

1. They only use function values.
2. They do not approximate the gradient.

The second criterion might be slightly ambiguous but intends to exclude methods that use finite differences to approximate the gradient. These features make direct search methods a viable choice to solve optimization problems in the expensive blackbox setting. A more pictorial description of direct search methods is given in [Powell, 1994]: it consists in *finding the deepest point of a muddy lake, given a boat and a plumb line, when there is a price to be paid for each sounding*. The term *direct search* already appears in [Hooke and Jeeves, 1961] where a first description of this class of methods can be found. The original motivation for direct search algorithms was to solve practical problems that were unsuccessfully attacked by classical methods such as Newton-based algorithms. The first direct search methods were proposed in the 1950s and 1960s, at the time of the first digital computers. As direct methods require less computational effort than classical methods – the choice of evaluation points is based on simple rules and do not require to construct a local approximation of the objective function – they were particularly appealing in these early years of numerical optimization where only a limited computational power was available. The very first example of a direct search method is the coordinate search attributed to [Fermi et al., 1954] (and concisely described in [Davidon, 1991]) which used one of the first computers to fit some theoretical parameters (phase shifts) to experimental data (scattering cross sections).

The development of direct search methods really took off in the 1960s with simplex-based methods introduced by [Spendley et al., 1962] and the popular Nelder-Mead algorithm [Nelder and Mead, 1965]. Simplex-based methods evaluate f on the vertices of a simplex. At iteration $l+1$, a new simplex is constructed using a simple transformation such as a reflection, a contraction or an expansion of the current simplex. The transformations are chosen so that the simplex moves and shrinks around the optimum. The Nelder-Mead algorithm has encountered a large success in practice but shows a wide range of performance depending on the problem at hand [Wright, 1996]. In his thesis [Woods, 1985], Woods even provides an example where the procedure converges to a non-critical point. The Nelder-Mead algorithm was designed so that the simplex *adapts to the local landscape* of the function, with the drawback that it is drastically deformed with a small volume for ill-conditioned functions. This feature may be responsible for both the successes and the failures of the method [Wright, 1996].

The lack of theoretical guarantees for the convergence of the Nelder-Mead algorithm is one of the main flaws of the method. A first direct search algorithm with strong convergence properties is the multidirectional search method proposed by [Torczon, 1989]. This simplex-based method also belongs to the class of pattern search methods, introduced in [Hooke and Jeeves, 1961], and for which theoretical guarantees are given in [Torczon, 1997]. The idea of pattern search methods is to generate points only on a scaled lattice, hence avoiding the degeneration of the simplex. This lattice structure is at the heart of the convergence proofs of these algorithms. The interested reader can refer to [Torczon and Trosset, 1998] for a more detailed exposition of the key features of pattern search methods that the Nelder-Mead algorithm lacks.

The ultimate goal of Part I of the thesis is to solve a maintenance optimization problem where the objective function is given as a blackbox. In Chapter 3, we have presented the EGO algorithm for this task. In this chapter, we study direct search methods, that are another class of algorithms adapted to blackbox optimization.

Contributions. We carry out a bibliographical review of the Generalized Pattern Search (GPS) algorithm in Section 4.2 and of its generalization in Section 4.3, called the Mesh Adaptive Direct Search (MADS) algorithm. We give a detailed description of the functioning of these methods as well as the associated theoretical guarantees. The role of this chapter is to give the reader all the necessary material to understand the philosophy behind the MADS algorithm. We will then be able, in Chapter 6, to give a critical interpretation of the comparative performance of MADS and EGO when applied to blackbox optimization problems.

4.2 Generalized Pattern Search (GPS)

The GPS algorithm has first been introduced in [Torczon, 1997] and exhibits theoretical convergence properties. In §4.2.1, we describe the GPS algorithm using the terminology of [Audet and Dennis, 2002] as in our view it is easier to work with. Then, in §4.2.2, we give the associated theoretical convergence results. The GPS algorithm falls into the general class of *pattern search methods* as defined by [Torczon, 1997].

Definition 4.1. A *lattice* $M \subset \mathbb{R}^d$ is a set of the form:

$$M = \left\{ \sum_{i=1}^{d} a_i v_i \;\middle|\; a_i \in \mathbb{Z},\ \forall i \in \{1, \dots, n\} \right\},$$

where the family of vectors $(v_1, \dots, v_d)$ is a basis of $\mathbb{R}^d$. Hence, a lattice is the set of all linear combinations with integer coefficients of the elements of the basis $(v_1, \dots, v_d)$.

Figure 4.1 shows three examples of lattice in $\mathbb{R}^2$. The basis vectors are drawn in red and the black points represent a finite subset of the elements of the lattice.

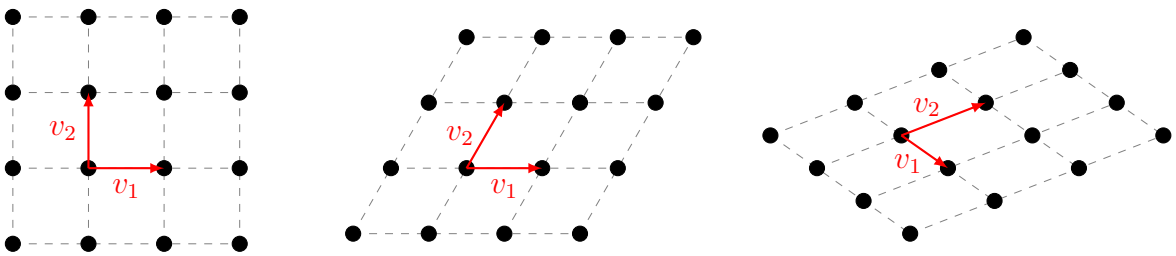


Figure 4.1: Examples of lattice in $\mathbb{R}^2$.

Pattern search methods are formally defined in [Torczon, 1997]. Here, we slightly reformulate the original definition in order to make it clearer in our view, using the concept of ρ-M *compatibility* defined below.

Definition 4.2. Let $\rho = \{\rho_n\}_{n\in\mathbb{N}}$ be a strictly positive real sequence and M be a lattice in $\mathbb{R}^d$. A sequence $\{u_l\}_{l\in\mathbb{N}}$ of elements of $\mathbb{R}^d$ is said to be ρ-M *compatible* if for all $n \in \mathbb{N}$, the steps $\{u_{l+1} - u_l\}_{1\leq l\leq n-1}$ lie in the scaled lattice $\rho_n M$.

Definition 4.3. An algorithm with iterates $\{u_l\}_{l\in\mathbb{N}}$ is said to be a *pattern search algorithm* if there exists a strictly positive real sequence $\rho = \{\rho_n\}_{n\in\mathbb{N}}$ and a lattice M such that the sequence $\{u_l\}_{l\in\mathbb{N}}$ is ρ-M compatible.

4.2.1 Description of the GPS algorithm

We start with some definitions for the description of the GPS algorithm.

Definition 4.4. A set $S \subset \mathbb{R}^d$ is called a *positive spanning set* if every element of $\mathbb{R}^d$ can be written as a nonnegative linear combination of elements of S.

Definition 4.5. A *pattern* is a positive spanning set $S = \{s^1, \dots, s^k\} \subset \mathbb{Z}^d$ *i.e.* the elements of S are integer vectors.[1] By abuse of notation, we also denote by $S \in \mathbb{R}^{d \times k}$ the matrix with columns $s^1, \dots, s^k$.

Figure 4.2 shows examples of pattern in $\mathbb{R}^2$. In dimension d, the smallest pattern contains $d+1$ vectors. We can consider arbitrarily rich patterns in general, but for the algorithms of this chapter, we always consider the pattern $S = \{s^1, \dots, s^{2d}\} = \{\pm e_i\}_{1 \le i \le d}$ that consists of the canonical basis of $\mathbb{R}^d$ completed with its opposite, represented on the left plot of Figure 4.2.

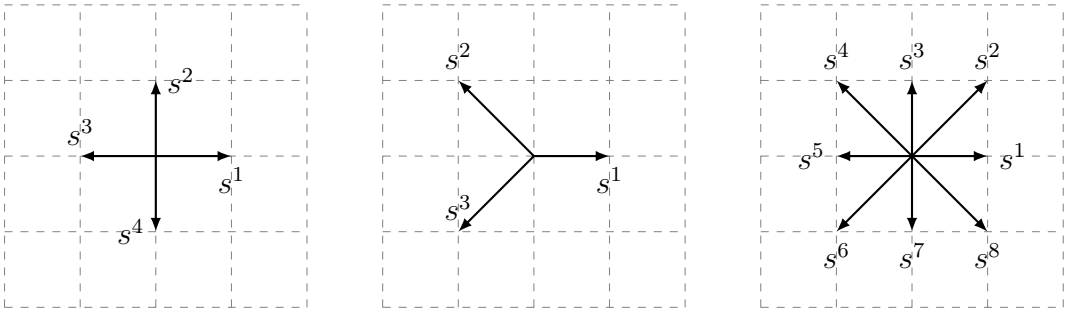


Figure 4.2: Examples of pattern in $\mathbb{R}^2$.

Now, we introduce the notions of *mesh* and *frame* that are essential to describe the iterations of the GPS algorithm.

Definition 4.6. Let S be a pattern and u_l be the l-th iterate of the GPS algorithm.

1. We define the *current mesh* M_l as:

$$M_l = \left\{ u_l + \Delta_l^{\mathrm{m}} S y,\ y \in \mathbb{N}^k \right\} , \tag{4.1}$$

   where $\Delta_l^{\mathrm{m}} > 0$ is called the *mesh size parameter*.

2. Let $S_l \subset S$ be a positive spanning set. We define the *frame* P_l centered in u_l as:

$$P_l = \{ u_l + \Delta_l^{\mathrm{m}} s,\ s \in S_l \} . \tag{4.2}$$

   The set S_l is referred as the set of *polling directions* at iteration l.

An iteration of the GPS algorithm consists of two steps: a *search step* and a *poll step*. We denote by u_l the iterate of the GPS algorithm at iteration l.

[1] In fact, the general definition of a pattern allows S to be of the form $S = GZ$ with $G \in \mathbb{R}^{d \times d}$ being a invertible matrix and $Z \in \mathbb{Z}^{d \times k}$. However, as G is taken to be the identity in the manuscript, we choose a simplified definition of a pattern.

1. The search step consists in evaluating the objective function f on a finite number of points (which can be none) lying on the mesh M_l. The requirement that evaluation points belong to M_l ensures that the iterates lie on a scaled lattice, see [Audet and Dennis, 2002, Proposition 3.4]. The search step is very flexible and acts as an exploratory phase: any user-defined strategy or heuristics can be used to choose some evaluation points. For instance, [Booker et al., 1999] uses a surrogate to choose promising points. We just need to ensure that the search step terminates after a finite number of evaluations of f. During the search step, if we find a point u_{l+1} such that $f(u_{l+1}) < f(u_l)$, the iteration terminates and the mesh size parameter $\Delta_l^{\mathfrak{m}}$ is unchanged or increased. The iteration is said to be *successful* and u_{l+1} is said to be an *improved mesh point*. Otherwise, we go on to the poll step.

2. In the poll step, we evaluate f on the frame P_l. Note that $P_l \subset M_l$. The frame P_l consists of points of the mesh M_l that are in the neighborhood of the current iterate u_l.

   - If we find a point $u_l^i \in P_l$ such that $f(u_l^i) < f(u_l)$, then u_l^i is called an *improved mesh point*. We set $u_{l+1} = u_l^i$ and $\Delta_l^{\mathfrak{m}}$ is unchanged or increased. The iteration is said to be *successful*.
   - Otherwise, the iteration is *unsuccessful*. We set $u_{l+1} = u_l$ and we decrease the mesh size parameter. This allows to consider evaluation points closer to the current best solution at the next iteration.

We will see in §4.2.2 that the convergence analysis of GPS only relies on the poll step, this is why the search step is very flexible. However, this search step has a important influence on the practical performance of the algorithm as good heuristics or surrogate techniques can greatly speed up the method.

Remark 4.7. We give some precisions on the requirements and the choice for the positive spanning sets S and S_l.

1. The requirement that S_l (and therefore S) is a positive spanning set ensures that it contains at least one vector in each half space. If f is Gateaux-differentiable and u_l is not a critical point, then there exists in S_l at least one descent direction for f at u_l. The reduction of the mesh size parameter guarantees that there will be only a finite number of unsuccessful iterations.

2. The frame P_l is defined by a positive spanning set $S_l \subset S$ that is chosen by the user at each iteration of the algorithm. When the problem of interest (2.1) is a bound constrained problem, we can simply take $S = \{\pm e_i\}_{1 \leq i \leq d}$ being the canonical basis of $\mathbb{R}^d$ completed with its opposite. Then, we take $S_l = S$ at each iteration of the GPS algorithm. The generality of Definition 4.6 is useful for linearly constrained problem. In this case, when iterates are generated near the boundary of the admissible set U^{ad}, we must be able to choose a set $S_l \subset S$ of polling directions that is adapted to the geometry of U^{ad}, *i.e.* we require that S_l is a positive spanning set of some tangent cone of U^{ad}. Hence, the set S must be rich enough so that an adapted choice for S_l is possible. We can refer to [Audet and Dennis, 2002, §3.5] for more details. ◊

In order to precise the mesh size parameter update rule, let $q > 1$ be a rational number and $a^- \leq -1$, $a^+ \geq 0$ be two integers. The mesh size parameter is updated as follows:

$$\Delta_{l+1}^{\mathfrak{m}} = q^{a_l} \Delta_l^{\mathfrak{m}} , \tag{4.3}$$

with:

$$a_l \in \begin{cases} \{0, 1, \ldots, a^+\} & \text{if the iteration is successful ,} \\ \{a^-, a^- + 1, \ldots, -1\} & \text{if the iteration is unsuccessful .} \end{cases}$$

This formulation includes very general rules for the mesh size parameter update. The general idea is to increase or keep the same mesh size parameter when the iteration is successful ($q^{a_l} \geq 1$) and to decrease the mesh size parameter when the iteration is unsuccessful ($q^{a_l} < 1$). The strategy to choose a_l within $\{0, \ldots, a^+\}$ (for a successful iteration) or $\{a^-, \ldots, -1\}$ (unsuccessful iteration) is defined by the user.

Example 4.8. The case where $\Delta_l^{\mathfrak{m}}$ remains unchanged for a successful iteration and is divided by 2 for an unsuccessful iteration corresponds to $q = 2,\ a^+ = 0,\ a^- = -1$. △

The GPS algorithm is summarized in Algorithm 3. The termination criterion is a maximum number of iterations or a maximum number of evaluations of the objective function.

Algorithm 3 The GPS algorithm

Initialization: Let $N > 0$ be the maximum number of iterations. Choose a pattern S and the parameters $q,\ a^+,\ a^-$ of the mesh size update rule. Let $u_0 \in U^{\text{ad}}$ and $\Delta_0^{\mathfrak{m}} > 0$.

Iteration: For $l = 0, \ldots, N-1$, let u_l be the current iterate. Let M_l and P_l be the mesh and the frame defined by (4.1) and (4.2) respectively.

Search step: Evaluate f on a finite number n of points, possibly zero, of the mesh M_l with any user-defined strategy. Denote by $\{u_l^1, \ldots, u_l^n\}$ the set of search points.

– If an improved mesh point u_l^i is found, $1 \leq i \leq n$, *i.e.* $f(u_l^i) < f(u_l)$, set $u_{l+1} = u_l^i$: the iteration is successful, go to the mesh size parameter update step.

– Otherwise:

Poll step: Denote by $\{u_l^{n+1}, \ldots, u_l^p\}$ the set of points in the frame P_l. Evaluate f on $\{u_l^{n+1}, \ldots, u_l^p\}$.

∗ If an improved mesh point u_l^i is found, $n+1 \leq i \leq p$, *i.e.* $f(u_l^i) < f(u_l)$, set $u_{l+1} = u_l^i$: the iteration is successful.

∗ If no improved mesh point is found, set $u_{l+1} = u_l$: the iteration is unsuccessful.

Mesh size parameter update: Compute $\Delta_{l+1}^{\mathfrak{m}}$ according to (4.3).

Return the current best point u_N.

Proposition 4.9. *The GPS algorithm is a pattern search algorithm in the sense of Definition 4.3.*

Proof. Since the iterates of the algorithm satisfy $u_{l+1} \in M_l$ for all $l \in \mathbb{N}$, there exists $y_l \in \mathbb{N}^k$ such that:

$$u_{l+1} - u_l = \Delta_l^{\mathfrak{m}} S y_l .$$

Moreover, from the update rule (4.3), we deduce that for all $l \in \mathbb{N}$, there exists $b_l \in \mathbb{Z}$ such that $\Delta_l^{\mathfrak{m}} = q^{b_l}\Delta_0^{\mathfrak{m}}$. Let $n \in \mathbb{N}$ and introduce:

$$b_n^- = \min\{b_0, \dots, b_n\} \quad \text{and} \quad b_n^+ = \max\{b_0, \dots, b_n\} .$$

As $q \in \mathbb{Q}$, we can write $q = \frac{\alpha}{\beta}$ with α and β being relatively prime integers, so that:

$$u_{l+1} - u_l = \frac{\alpha^{b_n^-}}{\beta^{b_n^+}}\Delta_0^{\mathfrak{m}} \left(\alpha^{b_l - b_n^-}\beta^{b_n^+ - b_l} S y_l\right) .$$

By definition, the matrix S has integer coefficients. Moreover, $\alpha^{b_l - b_n^-}\beta^{b_n^+ - b_l} \in \mathbb{N}$ and $y_l \in \mathbb{N}^k$, therefore the vector $\left(\alpha^{b_l - b_n^-}\beta^{b_n^+ - b_l} S y_l\right)$ has integer coefficients, showing that it belongs to the lattice M generated by the canonical basis of $\mathbb{R}^d$. With $\rho_n = \frac{\alpha^{b_n^-}}{\beta^{b_n^+}}\Delta_0^{\mathfrak{m}} > 0$, we get that $u_{l+1} - u_l \in \rho_n M$ so that the GPS algorithm is indeed a pattern search method. □

Specific pattern search methods can be distinguished by three factors: the choice of the pattern S, the manner in which the research of an iterate among the set of evaluation points is conducted and the rule for the mesh size parameter update (*i.e.* the values of q, a^+, a^-).

Example 4.10. We specify the pattern S and the mesh size parameter update rule for the coordinate search algorithm. Coordinate search is described concisely by [Davidon, 1991] when he reports the methodology of [Fermi et al., 1954]: *they varied one theoretical parameter at a time by steps of the same magnitude, and when no such increase or decrease in any one parameter further improved the fit to the experimental data, they halved the step size and repeated the process until the steps were deemed sufficiently small.* Coordinate search is a pattern search algorithm with the pattern $S = \{\pm e_i\}_{1 \le i \le d}$. At iteration l, let u_l be the current iterate and $\Delta_l^{\mathfrak{m}}$ be the mesh size parameter. No search step is performed and in the poll step, we take $S_l = S$. The objective function is then evaluated on the points of the frame:

$$P_l = \{u_l \pm \Delta_l^{\mathfrak{m}} e_i, 1 \le i \le d\} .$$

If an improved mesh point is found, $\Delta_l^{\mathfrak{m}}$ is unchanged, otherwise it is divided by 2. This corresponds to the update rule (4.3) with $q = 2$, $a^+ = 0$ and $a^- = -1$.

We can find the explicit values of S and of the mesh size parameter update rule in [Torczon, 1997] for some other pattern search algorithms. △

Example 4.11. Figure 4.3 represents an example for the mesh M_l and the frame $P_l = \{p_l^1, p_l^2, p_l^3\}$ at successive iterations of the GPS algorithm. We have supposed that $S = \{(a, b) \neq (0, 0),\ a, b \in \{-1, 0, 1\}\}$, this pattern corresponds to the right plot of Figure 4.2. At each iteration, the set of polling directions S_l is chosen as a subset of S that is a positive spanning set of $\mathbb{R}^2$.

4.2.2 Convergence results

The convergence of pattern search methods has been established in the unconstrained case [Torczon, 1997], in the bound constrained case [Lewis and Torczon, 1999] and in the linearly constrained case [Lewis and Torczon, 2000]. A convergence analysis of the GPS algorithm is also done in [Audet and Dennis, 2002] in the linearly constrained

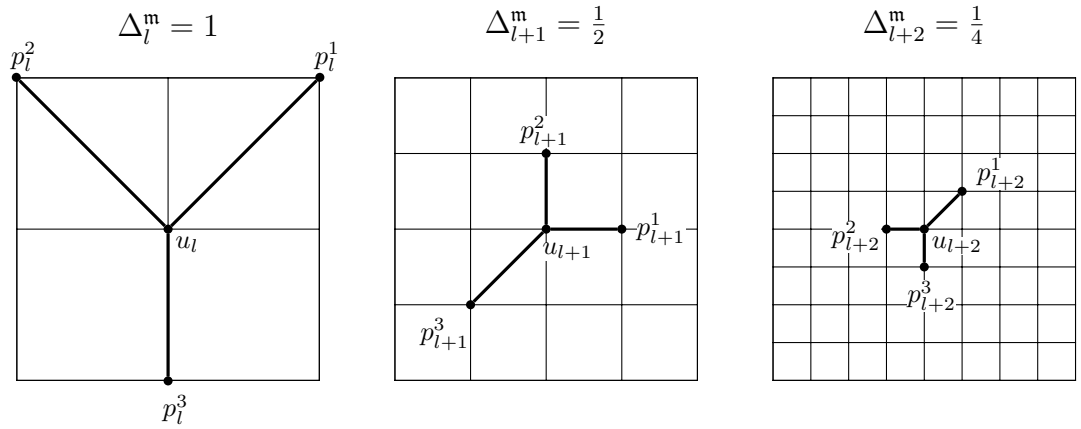


Figure 4.3: Example of mesh M_l and frame $P_l = \{p_l^1, p_l^2, p_l^3\}$ at successive (unsuccessful) iterations of the GPS algorithm. The mesh M_l consists of the intersections of the lines, the trial points of P_l are represented by black points. △

case. The results show that when f is smooth, the GPS algorithm converges to a stationary point of f. The theoretical convergence analysis relies only on the poll step, this is why much flexibility is allowed in the search step. We start by some definitions.

Definition 4.12. Let $\{u_l\}_{l\in\mathbb{N}}$ be the sequence of iterates of the GPS algorithm:

- If iteration l is successful (*i.e.* $f(u_{l+1}) < f(u_l)$), u_{l+1} is said to be an *improved mesh point*.
- If iteration l is unsuccessful (*i.e.* $u_{l+1} = u_l$), then u_l is said to be a *mesh local optimizer*: u_{l+1} has a lowest objective function value than its neighboring mesh points.
- A subsequence $\{u_{\sigma(l)}\}_{l\in\mathbb{N}}$ consisting of mesh local optimizers is a *refining subsequence* if $\{\Delta_{\sigma(l)}^{\mathbb{m}}\}_{l\in\mathbb{N}}$ converges to zero.

Definition 4.13. A function $f : \mathbb{R}^d \to \mathbb{R}$ is *locally Lipschitz near* $u \in \mathbb{R}^d$ if there exists $A > 0$ and $\varepsilon > 0$ such that:

$$|f(v) - f(w)| \leq A \|v - w\| \ , \quad \forall v, w \in \mathcal{B}(u, \varepsilon) \ ,$$

where $\mathcal{B}(u, \varepsilon)$ is the open ball of radius ε centered in u.

Definition 4.14. [Clarke, 1990] Let $f : \mathbb{R}^d \to \mathbb{R}$ be a locally Lipschitz function near $u \in \mathbb{R}^d$.

- The *Clarke generalized directional derivative* of f at u in the direction $s \in \mathbb{R}^d$ is defined as:

$$f^\circ(u; s) = \limsup_{\substack{v \to u,\ v \in U^{\text{ad}} \\ t \downarrow 0,\ v + ts \in U^{\text{ad}}}} \frac{f(v + ts) - f(v)}{t} \ .$$

- A point $u \in \mathbb{R}^d$ is a *Clarke stationary point*[2] of f if $f^\circ(u; s) \geq 0$ for all $s \in \mathbb{R}^d$.

[2] The definition of a Clarke stationary point is given here in the unconstrained case. In the constrained case, u is a Clarke stationary point of f if and only if $f^\circ(u; s) \geq 0$ for all s in the Clarke tangent cone to U^{ad} at u. We refer to [Audet and Dennis, 2006, Definition 3.5] for the definition of the Clarke tangent cone.

We can now give the main convergence result of the GPS algorithm.

Theorem 4.15. *[Audet and Dennis, 2002] There exists at least one converging refining subsequence. Let $u^\sharp$ be the limit of such a sequence. Assume that f is locally Lipschitz near $u^\sharp$, then the Clarke generalized directional derivative of f at $u^\sharp$ is nonnegative for a finite set of directions $\hat{S} \subset S$, that is $f^\circ(u^\sharp; s) \geq 0$ for all $s \in \hat{S}$.*[3]

In fact, the results of Theorem 4.15 cannot be improved in the sense that the GPS algorithm does not converge to a Clarke stationary point in general, as shown by Example F of [Audet, 2004]. The fact that the generalized derivative is provably nonnegative only for a finite set directions comes from the restriction to a finite set S of polling directions [Audet and Dennis, 2006]. This observation is the motivation for considering a version of the algorithm that can generate a set of normalized polling directions that is dense in the unit sphere of $\mathbb{R}^d$: this is the goal of the Mesh Adaptive Direct Search algorithm.

4.3 Mesh Adaptive Direct Search (MADS)

In this section, we present the MADS algorithm of [Audet and Dennis, 2006]. MADS is similar to GPS with the only difference that a dense set of polling directions are considered.

4.3.1 Description of the algorithm

In GPS, the fineness of the mesh M_l and of the frame P_l is defined by a unique mesh size parameter $\Delta_l^{\mathfrak{m}}$. In MADS, a new parameter $\Delta_l^{\mathfrak{p}} > 0$ is introduced, called the *poll size parameter*. The poll size parameter controls the maximal distance between the current iterate u_l and the evaluation points that are considered during the poll step. In a sense, $\Delta_l^{\mathfrak{p}}$ defines what is considered as a neighborhood of the current iterate. The poll size parameter $\Delta_l^{\mathfrak{p}}$ must satisfy two conditions:

(a) $\forall l \in \mathbb{N},\ \Delta_l^{\mathfrak{m}} \leq \Delta_l^{\mathfrak{p}}$.

(b) For any subsequence index $\{\sigma(l)\}_{l\in\mathbb{N}}$:

$$\lim_{l\in\mathbb{N}} \Delta_{\sigma(l)}^{\mathfrak{m}} = 0 \quad \text{if and only if} \quad \lim_{l\in\mathbb{N}} \Delta_{\sigma(l)}^{\mathfrak{p}} = 0 \,.$$

In GPS, the frame P_l (4.2) is defined from a set of polling directions $S_l \subset S$, usually $S_l = S = \{\pm e_i\}_{1\leq i\leq d}$. In MADS, the pattern S is similar as in GPS but more freedom is allowed for the set of the polling directions S_l. Before introducing the new polling directions, we need some technical definitions.

Definition 4.16. Let $S = \{s^1, \ldots, s^k\}$ be a subset of $\mathbb{R}^d$. We introduce the *normalized set* $\bar{S}$ as the set of normalized vectors from S:

$$\bar{S} = \left\{ \frac{s^1}{\|s^1\|}, \ldots, \frac{s^k}{\|s^k\|} \right\} .$$

[3] The set of directions $\hat{S}$ will become clear in §4.3.2, this is the set of *refining directions* of the GPS algorithm, see Definition 4.19.

Definition 4.17. [Coope and Price, 2000] Let $\{S_l\}_{l\in\mathbb{N}}$ be a sequence of subsets of $\mathbb{R}^d$. A set $S_\infty = \{s^1_\infty, \ldots, s^k_\infty\}$ is *a CP-limit* of the sequence of sets $\{S_l\}_{l\in\mathbb{N}}$ if there exists a subsequence $\{S_{\sigma(l)}\}_{l\in\mathbb{N}}$ of $\{S_l\}_{l\in\mathbb{N}}$ such that the sets $S_{\sigma(l)} = \{s^1_{\sigma(l)}, \ldots, s^k_{\sigma(l)}\}$ are all of cardinality k and that:

$$\lim_{l\to\infty} s^i_{\sigma(l)} = s^i_\infty, \quad i \in \{1, \ldots, k\} .$$

We can now present the conditions that must be satisfied by the polling directions in the MADS algorithm.

Definition 4.18. Let S be pattern and $\Delta^{\mathfrak{m}}_l, \Delta^{\mathfrak{p}}_l > 0$ be mesh and poll size parameters satisfying conditions (a) and (b). At iteration l of MADS, the set of polling directions S_l is a positive spanning set such that for all $s \in S_l$:

(i) s is a nonnegative integer combination of elements of S *i.e.* there exists $y \in \mathbb{N}^k$ such that $s = Sy$.

(ii) $\Delta^{\mathfrak{m}}_l \|s\| \leq \Delta^{\mathfrak{p}}_l \max\{\|s'\|,\ s' \in S\}$, where $\|\cdot\|$ is the Euclidean norm in $\mathbb{R}^d$.

(iii) The CP-limits of the normalized sets $\bar{S}_l$ are positive spanning sets.

In MADS, the frame P_l is defined as:

$$P_l = \{u_l + \Delta^{\mathfrak{m}}_l s,\ s \in S_l\} \ , \tag{4.4}$$

with polling directions S_l taken as in Definition 4.18. Therefore, S_l is not restricted to be subset of the pattern S as in GPS and we have more freedom for the poll step. The parameter $\Delta^{\mathfrak{p}}_l$ determines how far on the mesh we can go to construct these polling directions as illustrated on Figure 4.4. The bold black square allows to visualize condition (ii) of the definition of P_l: the evaluation points of the poll step must be points of the mesh that are inside the square. We see that when $\Delta^{\mathfrak{m}}_l$ decreases faster than $\Delta^{\mathfrak{p}}_l$, MADS can choose among a larger and larger set of polling directions: the possible directions are defined by all the mesh points inside the black square. In Figure 4.4, three of these possible directions are selected at each iteration.

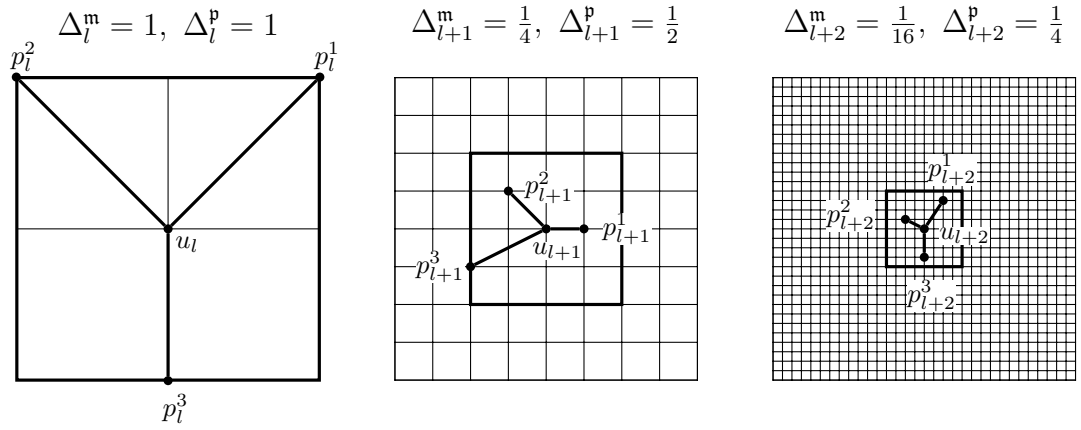


Figure 4.4: Example of mesh M_l and frame $P_l = \{p^1_l, p^2_l, p^3_l\}$ at successive (unsuccessful) iterations of the MADS algorithm. The mesh M_l consists of the intersections of the lines, the evaluation points of P_l are represented by black points.

MADS is summarized in Algorithm 4 and has the same structure as GPS. In the case where $\Delta^{\mathfrak{m}}_l = \Delta^{\mathfrak{p}}_l$ for all l, we exactly get the GPS algorithm.

Algorithm 4 The MADS algorithm

Initialization: Let $N > 0$ be the maximum number of iterations. Choose a pattern S and the parameters q, a^+, a^- of the mesh size update rule. Let $u_0 \in U^{\text{ad}}$ and $\Delta_0^{\mathfrak{m}}, \Delta_0^{\mathfrak{p}} > 0$.

Iteration: For $l = 0, \ldots, N-1$, let u_l be the current iterate. Let M_l and P_l be the mesh and the frame defined by (4.1) and (4.4) respectively.

Search step: Evaluate f on a finite number n of points, possibly zero, of the mesh M_l with any user-defined strategy. Denote by $\{u_l^1, \ldots, u_l^n\}$ the set of search points.

- If an improved mesh point u_l^i is found, $1 \leq i \leq n$, *i.e.* $f(u_l^i) < f(u_l)$, set $u_{l+1} = u_l^i$: the iteration is successful, go to the mesh size parameter update step.
- Otherwise:

  Poll step: Denote by $\{u_l^{n+1}, \ldots, u_l^p\}$ the set of points in the frame P_l. Evaluate f on $\{u_l^{n+1}, \ldots, u_l^p\}$.

  * If an improved mesh point u_l^i is found, $n+1 \leq i \leq p$, *i.e.* $f(u_l^i) < f(u_l)$, set $u_{l+1} = u_l^i$: the iteration is successful.
  * If no improved mesh point is found, set $u_{l+1} = u_l$: the iteration is unsuccessful.

Mesh size parameter update: Compute $\Delta_{l+1}^{\mathfrak{m}}$ and $\Delta_{l+1}^{\mathfrak{p}}$ according to (4.3) and in order to satisfy conditions (a) and (b).

Return the current best point u_N.

4.3.2 Convergence results

The convergence analysis of MADS is carried out in [Audet and Dennis, 2006]. The notions introduced in Definition 4.12 are still valid for MADS. We define the concept of *refining direction* that is at the heart of the convergence results of MADS.

Definition 4.19. Let $u^\sharp$ be the limit of a refining subsequence $\{u_{\sigma(l)}\}_{l \in \mathbb{N}}$. Suppose that there exists a subsequence of normalized polling directions $\left\{ s_{\tau(\sigma(l))} / \left\| s_{\tau(\sigma(l))} \right\| \right\}_{l \in \mathbb{N}}$, with $s_{\tau(\sigma(l))} \in S_{\tau(\sigma(l))}$, which converges to a limit $s \in \mathbb{R}^d$ and that $u_{\tau(\sigma(l))} + \Delta_{\tau(\sigma(l))}^{\mathfrak{m}} s_{\tau(\sigma(l))} \in U^{\text{ad}}$ for all l. Then, the limit s is a *refining direction* for $u^\sharp$.

We give the convergence result of MADS in the case where the limit point of the algorithm is in the interior of the domain. In particular, the following result is valid in the unconstrained case $U^{\text{ad}} = \mathbb{R}^d$.

Theorem 4.20. *Let $u^\sharp$ be the limit of a refining subsequence. Assume that $u^\sharp \in \text{int}(U^{\text{ad}})$ and that f is Lipschitz continuous near $u^\sharp$. If the set of refining directions for $u^\sharp$ is dense in the unit sphere of $\mathbb{R}^d$, then $u^\sharp$ is a Clarke stationary point.*

Remark 4.21. The convergence result is still valid if $u^\sharp$ is on the boundary of the domain U^{ad}. We have only cited the case where $u^\sharp \in \text{int}(U^{\text{ad}})$ in order to avoid some technicalities that may not be useful to capture the essence of the point. The interested reader can refer to [Audet and Dennis, 2006] for the complete results and proofs. ◊

The key point to guarantee the convergence of MADS to a stationary point is the density of the refining directions. In the convergence result of GPS (Theorem 4.15), the set $\hat{S}$ is in fact the set of refining directions for GPS. As refining directions are a subset of normalized polling directions, they can only be in finite number for GPS.

4.3.3 The OrthoMADS implementation of MADS

In the general description of MADS, nothing ensures that the set of refining directions is dense. For example, GPS is a valid instance of MADS. In order to use MADS in practice, we need to specify an implementation of the algorithm, that is, an explicit strategy for the mesh size parameters update and an appropriate choice for the polling directions S_l. The first implementation of MADS that has been described is LTMADS [Audet and Dennis, 2006] where the polling directions are constructed from a random lower triangular basis matrix. LTMADS is a stochastic instance of MADS as some randomness is used in the choice of the polling directions. Here, we present the implementation OrthoMADS [Abramson et al., 2009] that generates at each iteration a set of polling directions which consists of an orthogonal basis and its opposite. The advantage of OrthoMADS is that the polling directions are chosen deterministically which allows for repeatability of the algorithm on a given problem.

4.3.3.1 Construction of the polling directions

In OrthoMADS, we take the pattern $S = \{\pm e_i\}_{1 \leq i \leq d}$, that consists of the canonical basis of $\mathbb{R}^d$ and its opposite. The construction of the set of polling directions S_l is based the Halton sequence in $[0,1]^d$, here denoted by $\{v_t\}_{t \in \mathbb{N}}$ [Halton, 1960]. The density of the polling directions will follow from the density of the Halton sequence in the hypercube $[0,1]^d$. Recall that the elements of S_l must be nonnegative integer combinations of elements of S (condition (i) in Definition 4.18), that is, integer vectors. This property is not satisfied by the Halton sequence so we cannot use it directly. The construction of S_l is done in three steps, that are illustrated on Figure 4.5:

1. The direction v_t is translated, scaled and rounded to generate an adjusted Halton direction $a_{t,r} \in \mathbb{Z}^d$, where $r \in \mathbb{N}$ is a parameter of the transformation such that the norm of $a_{t,r}$ is *close* to $2^{|r|/2}$ without exceeding it. The vector $a_{t,r}$ is constructed such that the normalized direction $\frac{a_{t,r}}{\|a_{t,r}\|}$ is close to $\frac{2v_t - \mathbf{1}_d}{\|2v_t - \mathbf{1}_d\|}$ where $\mathbf{1}_d = (1, \ldots, 1)^\top$.

2. We apply the Householder transform [Householder, 1958] to $a_{t,r} \in \mathbb{Z}^d$ in order to construct an orthogonal basis of $\mathbb{R}^d$ composed of integer vectors:

$$H_{t,r} = \|a_{t,r}\|^2 \, \mathrm{I}_d - 2 a_{t,r} a_{t,r}^\top \, .$$

   The columns of the matrix $H_{t,r}$ form an integer orthogonal basis for $\mathbb{R}^d$.

3. The set S_l is only determined by values $t_l \in \mathbb{N}$ and $r_l \in \mathbb{Z}$ that parametrize the construction of the orthogonal basis H_{t_l,r_l}. The set S_l is composed of the columns of H_{t_l,r_l} and their opposite:

$$S_l = \begin{pmatrix} H_{t_l,r_l} & -H_{t_l,r_l} \end{pmatrix} \, .$$

4.3.3.2 Mesh size update rule and choice of the parameters t_l and r_l

In order to fully specify the OrthoMADS instance, we need to give the explicit rule for the mesh size and poll size parameters update and for the choice of t_l and r_l. At iteration $l = 0$, we set $\Delta_0^{\mathfrak{m}} = \Delta_0^{\mathfrak{p}} = 1$ and $l_0 = 0$.

- If iteration l is unsuccessful, then $r_{l+1} = r_l + 1$.

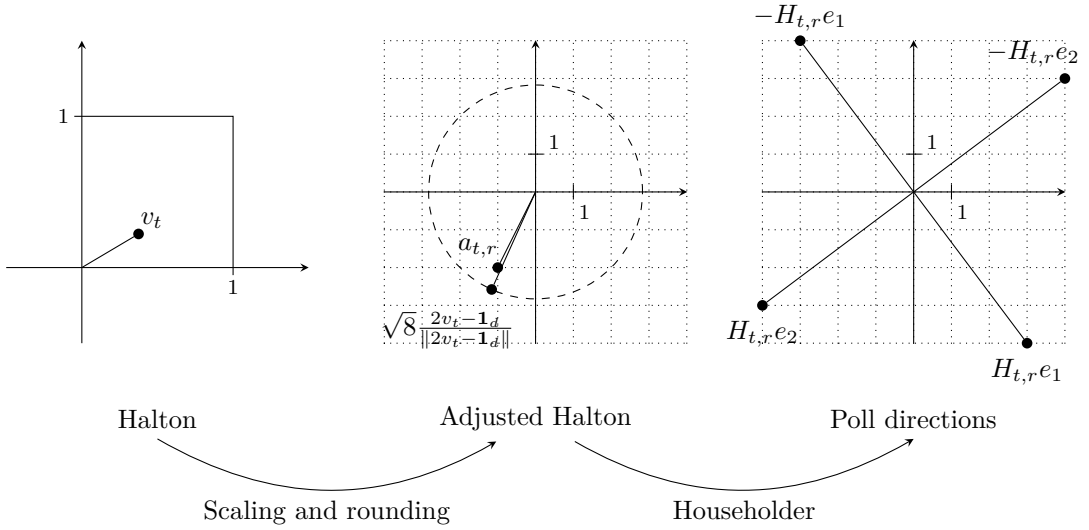


Figure 4.5: Example of the construction of polling directions from [Abramson et al., 2009] with $d = 2$ and $(t, r) = (6, 3)$. The Halton direction is $v_t = (3/8, 2/9)^\top$ and the adjusted Halton direction is $a_{t,r} = (-1, -2)^\top$. We have $H_{t,r}e_1 = (3, -4)^\top$ and $H_{t,r}e_2 = (-4, -3)^\top$.

- Otherwise, $r_{l+1} = r_l - 1$.

The mesh size parameters are then updated as follows:

$$\Delta_l^{\mathfrak{p}} = 2^{-r_l} \quad \text{and} \quad \Delta_l^{\mathfrak{m}} = \begin{cases} 4^{-r_l} & \text{if } r_l > 0 \ , \\ 1 & \text{otherwise} \ . \end{cases} \tag{4.5}$$

We always have $\Delta_l^{\mathfrak{m}} \leq \Delta_l^{\mathfrak{p}}$ and $2^{|r_l|}\Delta_l^{\mathfrak{m}} = \Delta_l^{\mathfrak{p}}$. The update rule (4.5) fits into the general framework (4.3) with $q = 4$, $a^+ = 1$ and $a^- = -1$. We see that in unsuccessful iterations, the parameter $\Delta_l^{\mathfrak{m}}$ decreases faster than $\Delta_l^{\mathfrak{p}}$ which corresponds to the illustration of Figure 4.4 and is key to generate a dense set of polling directions.

The choice of t_l is also important to generate a dense set of polling directions. We store the smallest poll size value that is used before iteration l:

$$\Delta_{\leq l,\min}^{\mathfrak{p}} = \min\{\Delta_j^{\mathfrak{p}},\ 0 \leq j \leq l\} \ .$$

- If, at iteration l, we have $\Delta_l^{\mathfrak{p}} = \Delta_{\leq l,\min}^{\mathfrak{p}}$, *i.e.* the current poll size is the smallest poll size used so far, then we set $t_l = r_l + d + 1$.

- For the other iterations, we just set $t_{l+1} = t_l + 1$.

Consider the sequence of ordered indices:

$$L = \{l_1, l_2, \ldots\} = \left\{l \in \mathbb{N},\ \text{iteration } l \text{ is unsuccessful and } \Delta_l^{\mathfrak{p}} = \Delta_{\leq l,\min}^{\mathfrak{p}}\right\} \ .$$

From our choice of t_l and r_l, we have $(t_{l_j}, r_{l_j}) = (d + j, j - 1)$ for $j \in \mathbb{N}$. The Halton sequence $\{v_{t_l}\}_{l\in L}$ is then exactly $\{v_t\}_{t\geq d+1}$. We can then use the density of the Halton sequence to show that for $i \in \{1, \ldots, 2d\}$, the set of normalized polling directions $\{S_l e_i / \|S_l e_i\|\}_{l\in L}$ is dense in the unit sphere of $\mathbb{R}^d$. Finally, note that we have $\|S_l e_i\| \leq 2^{|r_l|}$, therefore:

$$\Delta_l^{\mathfrak{m}} \|S_l e_i\| \leq \Delta_l^{\mathfrak{p}} \ .$$

This corresponds to condition (ii) of Definition 4.18 meaning that the trial points of the poll step indeed lie in the black square of Figure 4.4.

OrthoMADS is a valid implementation of MADS that satisfies all the necessary conditions to ensure the convergence of the algorithm to a Clarke stationary point of the objective function f.

4.4 Conclusion

This chapter gives an overview of direct search methods and more precisely of pattern search algorithms. This class of methods is suited to tackle expensive blackbox optimization problems. We have described the GPS algorithm and highlighted the ideas that have led to the design of MADS. The key feature of MADS is that it is possible to generate a dense set of polling directions, which allows to strengthen the convergence results compared to GPS. We have presented the explicit implementation OrthoMADS that satisfies the requirements for these strong theoretical results. We must also keep in mind that the practical performance of a pattern search method depends on the user-defined strategy in the search step. In Chapter 6, we present a comparison of the performance of MADS and EGO, introduced in Chapter 3, both on an academic benchmark and on a industrial maintenance optimization problem.

5 Two contributions for EGO

To doubt everything or to believe everything are two equally convenient solutions; both dispense with the necessity of reflection.

Henri Poincaré

Contents

5.1 Introduction

We are still concerned with the optimization problem (2.1) of minimizing $f\colon U^{\mathrm{ad}} \to \mathbb{R}$ in an expensive blackbox setting. In Chapter 3, we have presented the EGO algorithm that is adapted to this framework and focused on several practical aspects for the implementation of the method. In this chapter, we come back on the two following points.

1. We have highlighted that the initial design of experiments (DOE) should have space-filling properties for an efficient implementation of the method. The reason is that the infill criterion driving the choice of the next evaluation point in EGO is only computed from the metamodel. Hence, the algorithm achieves good performance only if the metamodel of f built from the initial design is sufficiently accurate. Most of the time, the size of the initial DOE is fixed using heuristic criteria that depend on the dimension d of the input space but not on the function to be optimized [Jones et al., 1998, Loeppky et al., 2009]. Hence, the size of the initial DOE may be either too small or too large for an optimal efficiency of the algorithm.

2. As stated in §3.3.2, we use the EI as infill criterion for EGO. However, the EI is difficult to optimize as it has many local maximizers and is very flat outside these local maxima. Choosing an efficient solver for the EI maximization is then a challenging task. This choice is almost always based on heuristic arguments in the literature, yet it heavily influences the behavior of the EGO algorithm.

Contributions. In order to tackle the aforementioned challenges, we propose the following approach.

1. In Section 5.2, we use a fully sequential space-filling (FSSF) initial design and assess the quality of the metamodel after each evaluation of the objective function f. Once the metamodel is accurate enough, we launch the infill step of the EGO algorithm. Hence, the size of the initial design is adapted to the difficulty of the problem and the evaluation budget is efficiently allocated between the initial experimental design and the core step of the EGO procedure. This new variant of EGO is called EGO-FSSF, standing for EGO with fully sequential space-filling design.

2. Section 5.3 is devoted to the choice of a solver for the EI maximization. We carry out a benchmark to assess the performance of various solvers for this particular task and provide quantitative arguments to choose the most efficient solver.

5.2 The EGO-FSSF algorithm

In this section, we present a new variant of the EGO algorithm, called EGO-FSSF, where the usual fixed-size initial design is replaced by a sequential design of adaptive size that still preserves space-filling properties.

Step 1 of the EGO algorithm 1 consists in evaluating the objective function on a fixed-size initial DOE. Possible choices for the DOE have been presented in §3.2.2. A natural question we face is the choice of the number of points in the initial design. If this number is too low, the metamodel may not be accurate enough and the EGO

procedure may fail [Jones, 2001], especially in the case where f presents multiple local minima. On the other hand, if too many points are used, the remaining budget for the EGO procedure might be too low to get a satisfactory result. In the literature, a common rule of thumb is to choose a number of samples for the initial DOE that is proportional to the input dimension d. Some authors explicitly choose $10d$ samples [Jones et al., 1998, Loeppky et al., 2009]. However, for a given input dimension d, this rule does not adapt to the difficulty of the problem. For example, in order to construct a metamodel with a given accuracy, fewer points may be needed for a smooth convex function than for a multimodal function. This is why we propose to use a FSSF design [Shang and Apley, 2020] instead of a fixed-size initial design in order to adaptively choose the number of samples in the initial DOE.

5.2.1 Description of the algorithm

In the original EGO procedure (Algorithm 1), the first metamodel is constructed after l_{doe} evaluations of the objective function f where l_{doe} is the predetermined size of the initial DOE. In EGO-FSSF, the initial step consists in sequential evaluations of the objective function f with an update of the metamodel after each observation. The sequence of evaluation points is determined by the FSSF-fr design of [Shang and Apley, 2020]. Details on the construction of this design are given in §5.2.2. The quality of the metamodel is assessed after each evaluation of f by computing the predictivity coefficient Q^2 and the PVA by leave-one-out cross validation (see §3.2.4). Once the metamodel accuracy reaches user-defined thresholds Q^2_{min} and PVA_{max}, or when a maximum number of evaluations $p_{\text{init}}M$ is done, with $0 < p_{\text{init}} < 1$ and M the overall budget, we go on to the infill step. We expect that this procedure efficiently splits the budget between the initial design (Step 1) and the infill step (Step 2).

The EGO-FSSF is summarized in Algorithm 5. We also consider the local optimization step of Algorithm 2 as it has proved its efficiency in [Mohammadi, 2016]. The performance of EGO-FSSF is compared to the classical EGO algorithm in Chapter 6.

5.2.2 Construction of the FSSF design

The sequential design we use is the FSSF design of [Shang and Apley, 2020], more precisely the FSSF-fr variant ("fr" stands for *forward-reflected*). We have already seen in §3.2.2 that space-filling properties are essential to get a metamodel of good quality. The challenge in constructing a *sequential* space-filling design is that it must retain good space-filling properties for each size. For example, designs produced with 10 points and 20 points must be as space-filling as possible, the design of size 10 being a subset of the design of size 20. This is illustrated in Figure 5.1. The construction of these designs is conceptually different from designs that are optimized for a predefined size, such as LHS designs, where the location of all points is decided at the same time. If we remove one point from a LHS design, it will have poor space-filling properties.

We summarize the construction of the FSSF-fr design. We start with a candidate set of points $S \subset U^{\text{ad}}$ that covers the domain U^{ad} fairly densely, for example a Sobol sequence with l_{sob} points. We choose l_{sob} so that $l_{\text{sob}} \gg l_{\text{doe}}$ where l_{doe} is the maximum number of points needed in the sequential design. The value $l_{\text{sob}} = 1000d + 2l_{\text{doe}}$ (where d is the dimension of the domain) is used in [Shang and Apley, 2020] and $l_{\text{doe}} = p_{\text{init}}M$ for the EGO-FSSF algorithm. Denote by $S_i = \{u_1, \ldots, u_i\} \subset S$ the design with i points. The design S_{i+1} is constructed from S_i to which we add the point $u_{i+1} \in S$

Algorithm 5 The EGO-FSSF algorithm

Initialisation. Choose an infill criterion and a class of covariance functions parametrized by the hyperparameters θ and σ^2. Define:

- $M > 0$ the maximum number of evaluations of the objective function f,
- $0 < p_{\text{init}}, p_{\text{local}} < 1$ respectively the maximum proportion of the budget used for the sequential initial DOE and the proportion of the budget allocated to the final local search.
- $Q^2_{\text{min}} < 1$ the predictivity threshold and $\text{PVA}_{\text{max}} > 0$ the PVA threshold,

Evaluate f at $u_1, u_2 \in U^{\text{ad}}$ and construct a first metamodel. Set $i = 2$.

Step 1. While ($Q^2 < Q^2_{\text{min}}$ or $\text{PVA} > \text{PVA}_{\text{max}}$) and $i < p_{\text{init}} M$:

(a) Add a new point u_i in the initial design according to a FSSF strategy.

(b) Evaluate $f(u_i)$, estimate the hyperparameters θ and σ^2 by maximum likelihood and update the metamodel with (3.10) and (3.11).

(c) Compute the predictivity coefficient Q^2 and the PVA by cross validation.

(d) $i \leftarrow i + 1$.

Step 2. While $i < (1 - p_{\text{local}})M$:

(a) Choose a new evaluation point u_i by optimizing the infill criterion.

(b) Evaluate $f(u_i)$, update the estimation of hyperparameters θ and σ^2 by maximum likelihood and update the metamodel with (3.10) and (3.11).

(c) $i \leftarrow i + 1$.

Step 3. With the remaining evaluation budget $p_{\text{local}} M$, run a local optimization with L-BFGS starting from the current best point.

Return. The best iterate of the L-BFGS algorithm.

that maximizes the minimum pairwise distance (maximin criterion):

$$u_{i+1} \in \arg\max_{u \in S} \ \text{dist}(u, S_i) \quad \text{where} \quad \text{dist}(u, S_i) = \min_{j \in \{1,\dots,i\}} \|u - u_j\| \ .$$

Choosing u_{i+1} based on this sole criteria leads to the FSSF-f design ("f" stands for *forward*) but it has the drawback that too many points are near the boundary, see Figure 5.2a. The FSSF-f design indeed optimizes the maximin distance but this is not satisfactory as large areas of the interior of the domain are empty, *i.e.* the fill distance (3.12) between design points is large.

To avoid selecting points too close to the boundary, we pretend that when we consider adding a point $u \in U^{\text{ad}}$ to the design, the *reflected point* $R(u)$ with respect to the boundary of U^{ad} is also added to the design, see Figure 5.3. The point u_{i+1} is then chosen such that:

$$u_{i+1} \in \arg\max_{u \in S} \ \min \Big\{ \text{dist}(u, S_i), \sqrt{2d} \times \text{dist}(u, R(u)) \Big\} \ .$$

If u is close to the boundary, it is close to its reflected point $R(u)$ so $\text{dist}(u, R(u))$ will be small and the point will not be added to the design. The scaling factor $\sqrt{2d}$ is used to appropriately scale the distance $\text{dist}(u, R(u))$ so that it can be fairly compared to pairwise distances between design points. An example of FSSF-fr design is given in Figure 5.2b.

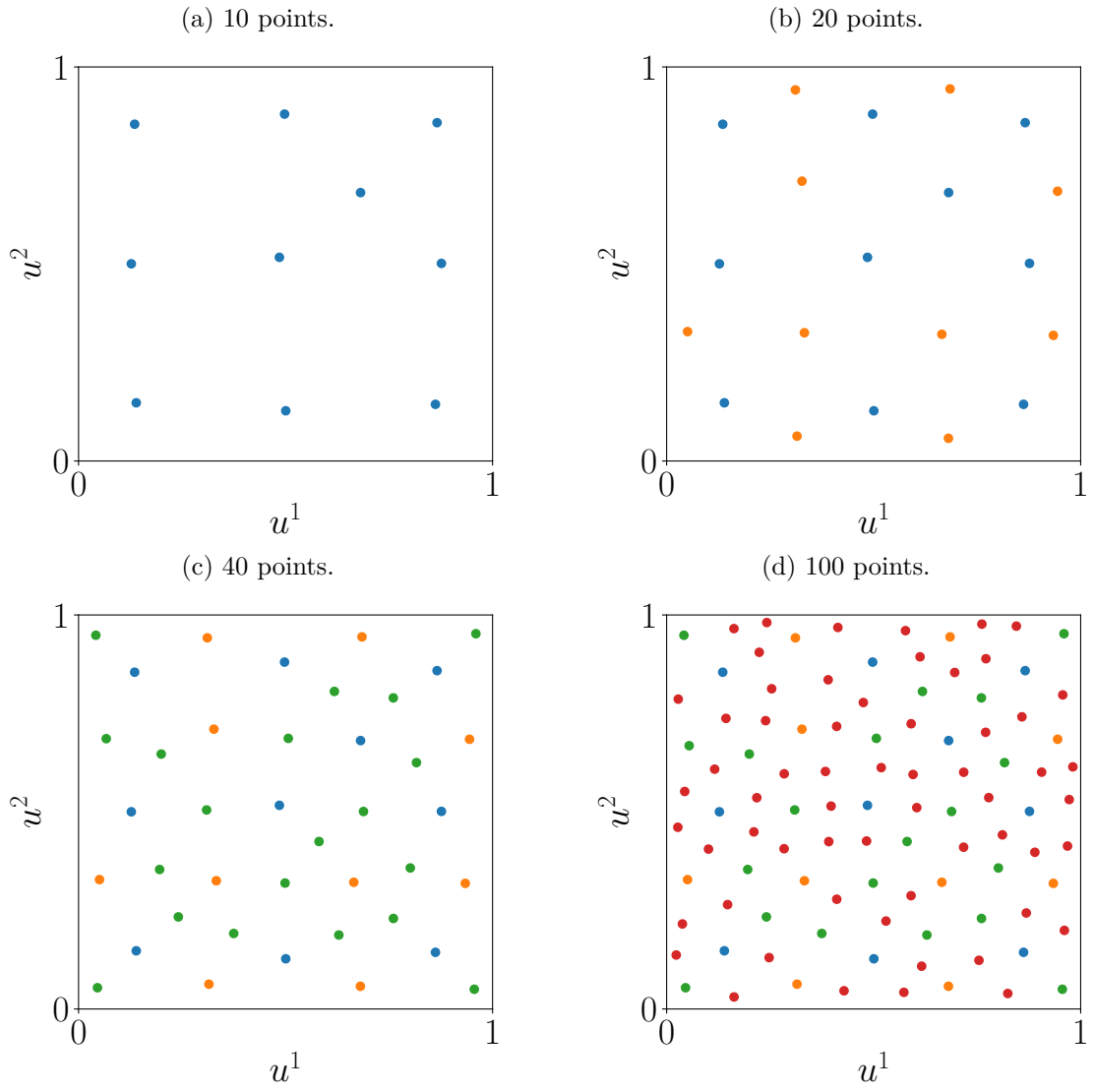


Figure 5.1: Sequential construction of a FSSF-fr design. For $l < m$, the design with l points is included in the design with m points. Each color represents the points that are added to go from one design to the next.

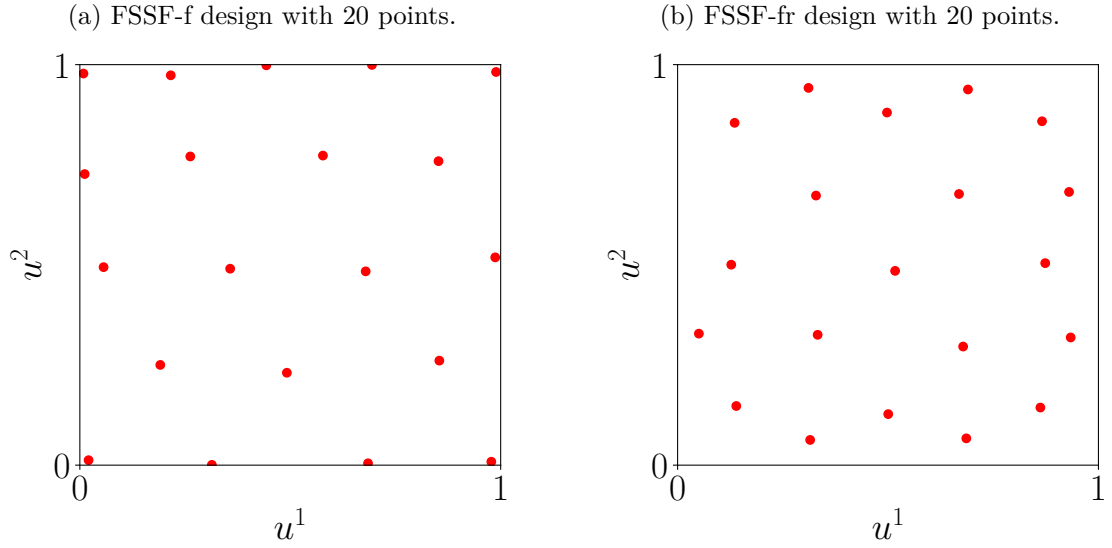


Figure 5.2: Two variants of FSSF designs on $U^{\text{ad}} = [0, 1]^2$.

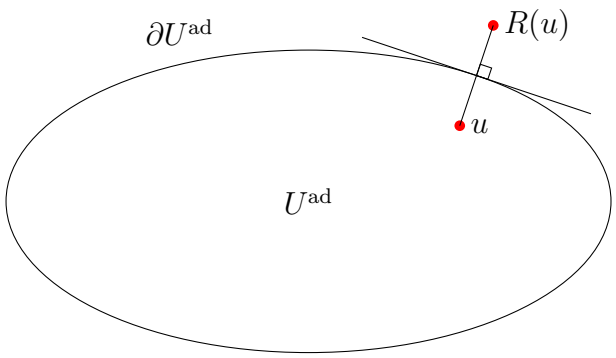


Figure 5.3: The reflected point $R(u)$ is the closest mirror image of u with respect to all hyperplanes that form the boundary of the domain ∂U^{ad}.

5.3 A comparison of solvers for EI maximization

In the numerical experiments with the EGO or the EGO-FSSF algorithm in Chapter 6, we will use the EI as infill criterion. In this section, we aim at providing quantitative arguments to select the most appropriate solver for EI maximization. To do so, we design a benchmark of EI functions and compare the performance of various solvers. We describe the experimental setting in §5.3.1 and present the numerical results in §5.3.2.

The EI maximization problem is always challenging, as highlighted in §3.3.2.2. Nevertheless, this task must be performed efficiently because the infill step directly drives the behavior of EGO. Various algorithms are used for EI maximization in the literature:

- CMA-ES [Hansen and Ostermeier, 1996] is used in [Mohammadi, 2016].
- The DIRECT algorithm of [Jones et al., 1993] is used in [Brochu et al., 2010].
- The package `DiceKriging` [Roustant et al., 2012] uses GENOUD (GENetic Optimization Using Derivatives) [Sekhon and Mebane, 1998], which is a hybrid combination between an evolutionary algorithm for global search and BFGS for local search.

However, there is hardly any quantitative analysis on the performance of these algorithms for the particular task of EI maximization. A notable contribution in this direction can be found in [Munoz Zuniga and Sinoquet, 2020]. The authors consider the EGO algorithm with mixed categorical-continuous variables and present an extensive comparison between five algorithms to justify the choice of the EI maximization solver. To our knowledge, no such work has been carried out for the continuous variables framework. We aim at giving a first contribution to fill this gap.

5.3.1 Experimental setting

To carry out a benchmark for EI maximization, the tricky part is to design *EI-like functions*, *i.e.* functions that exhibit the same characteristics as the EI functions that are encountered during the EGO algorithm. To construct our EI test bed, we take advantage of the COmparing Continuous Optimizers (COCO) platform [Hansen et al., 2021]. COCO provides a test bed of 24 functions that are scalable with the dimension. For each function, 15 instances are defined, corresponding to translations of the original

function. For each instance of a COCO function, we run the initial step of the EGO-FSSF design. The maximum number of evaluations is set to $20d$, where d is the dimension of the input space. The quality threshold is $(Q^2_{\min}, \mathrm{PVA}_{\max}) = (0.7,\ 2)$. We store the resulting kriging metamodel along with the location and value of the observation points. This information is sufficient to compute the EI criterion. This procedure is repeated for the 15 instances of the 24 functions of the COCO test bed, in dimension 2, 3, 5, 10, 20. We have then created a total of $15 \times 24 \times 5 = 1800$ *real* EI functions, that we denote by $g_1, \ldots, g_{1800}$. Figure 5.4 shows examples of EI functions that have been generated.

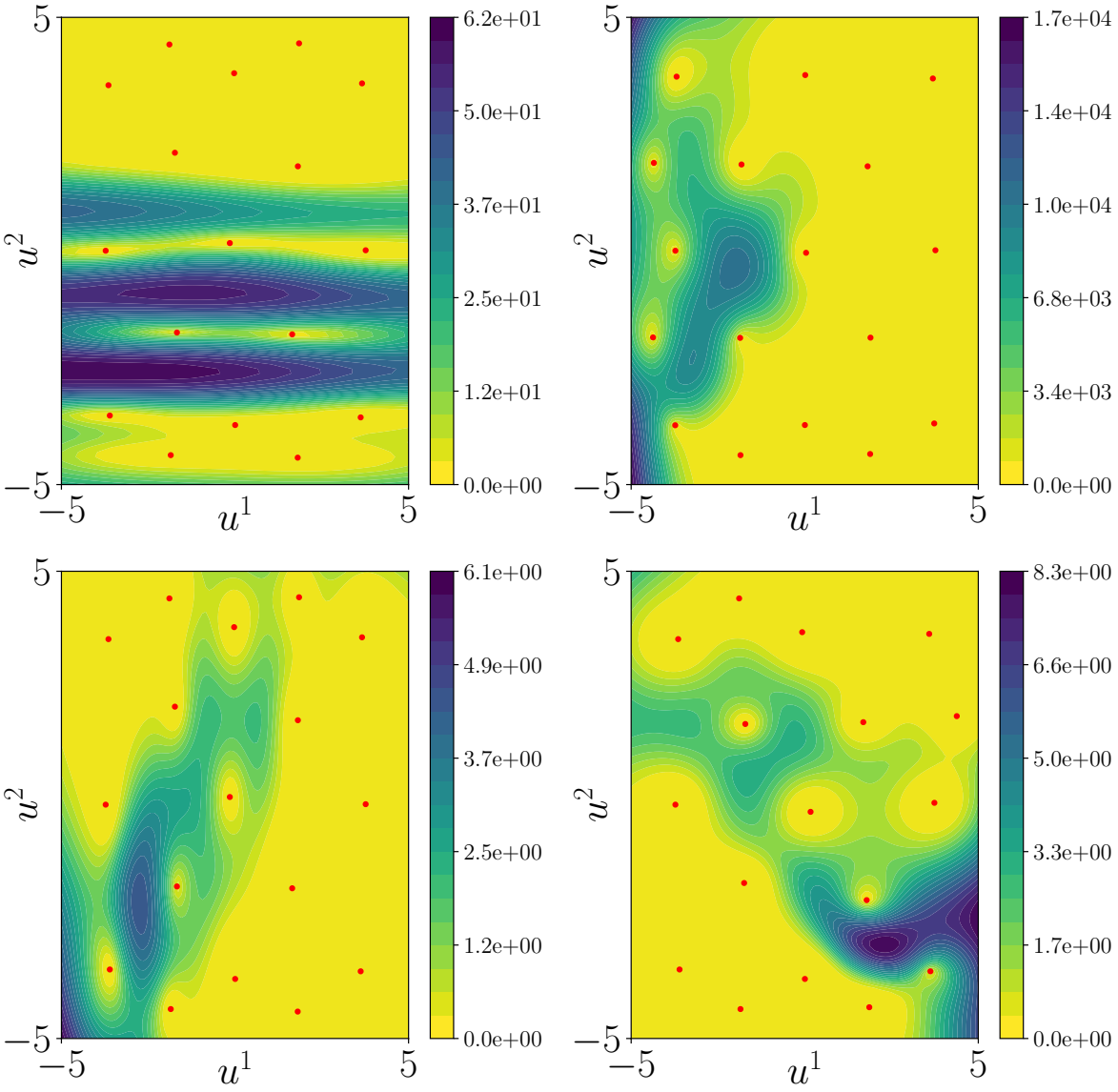


Figure 5.4: Typical examples of generated EI functions. The red dots indicate the location of the sample points of the objective function that are used to create the kriging metamodel and derive the EI function. We can check that the EI is zero at these points.

We work with the OpenTURNS software [Baudin et al., 2017]. OpenTURNS allows to choose the EI maximization solver among a large variety of algorithms, including solvers from the NLopt library [Johnson, 2009]. We run each algorithm of the NLopt library, as well as TNC [Nash, 2000], Cobyla [Powell, 1994] and the global optimization algorithm of the Dlib library (Dlib-Global) [King, 2009] on the 1800 EI functions. Some

solvers are only designed for local optimization. For these solvers, we use a multistart with $10d$ points, the initial points being chosen as an optimized LHS design. This design is identical for each local optimization solver. The global optimization solvers only use one starting point. This starting point is the same for all global solvers. The algorithms stop after $200d$ calls to the EI function. The solvers TNC, Cobyla and Dlib-Global use no information on the derivative of the EI. The names of the NLopt solvers are of the form NLopt-{G,L}{N,D}-XXX (see Figures 5.5 and 5.6) where G (resp. L) denotes a global (resp. local) solver and N (resp. D) denotes a derivative-free (resp. gradient-based) solver. For gradient-based solvers, OpenTURNS provides a finite difference approximation of the gradient of the EI.

5.3.2 Numerical results of the EI maximization benchmark

In this section, we present the results of the EI maximization benchmark. For each EI function g_p, $p \in \{1, \ldots, 1800\}$, the optimum $g_p^\sharp$ is taken as the maximum value of g_p found over all solvers. Then, for each algorithm, we compute the relative error:

$$r_p^{algo} = \frac{\left| g_p^\sharp - g_p^{algo} \right|}{g_p^\sharp},$$

where g_p^{algo} is the maximum value of g_p found by the algorithm $algo$. As the EI is always positive, the relative error is always between 0 and 1. For each solver, the distribution of the relative errors $\{r_p^{algo}\}_{p\in\{1,\ldots,1800\}}$ is represented with box plots in Figure 5.5. The boxes range from the first to the third quartile, with the inner line indicating the median of the values. The whiskers represent the 10-th and 90-th percentile.

The upper and lower graphs of Figure 5.5 emphasize different aspects of the performance of the algorithms. The linear scale (upper graph) allows to clearly compare the third quartile and the 90-th percentile of the algorithms. We see that the MLSL variants [Kucherenko and Sytsko, 2005] are the most robust, as the box is not even visible and the 90-th percentile is around 0.2 compared to 0.8 for most other solvers. The Dlib-Global algorithm is also performing well, with a third quartile close to zero but a 90-th percentile of 0.6 which is higher than for MLSL. We can also see that the STOGO variants do not perform well on the test bed with more than 25% of the function with a relative error greater than 0.8.

In order to have a clearer view on the lower percentiles and to compare the 75-th percentile of Dlib-Global and MLSL, we use the log scale representation of the lower graph of Figure 5.5. The different variants of the MLSL solver are the most efficient: for 75% of the functions, the relative error is in fact below 10^{-7}. This is several order of magnitude better than for all other algorithms, the second best being Dlib-Global with 10^{-4}.

As the location of the EI maximizer is what really determines the behavior of a subsequent EGO iteration, we also compute the L^2-distance between the location of the maximizer given by a solver and the overall maximizer, up to a scaling factor:

$$d_p^{algo} = \frac{\left\| u_p^\sharp - u_p^{algo} \right\|}{\sqrt{d}},$$

where $u_p^\sharp$ is the maximizer of g_p *i.e.* $g_p(u_p^\sharp) = g_p^\sharp$ and u_p^{algo} is the maximizer returned by the algorithm $algo$ for the function g_p. The distribution of $\{d_p^{algo}\}_{p\in\{1,\ldots,1800\}}$ is represented in Figure 5.6. The scaling factor $\sqrt{d}$ ensures that the values of d_p^{algo} are

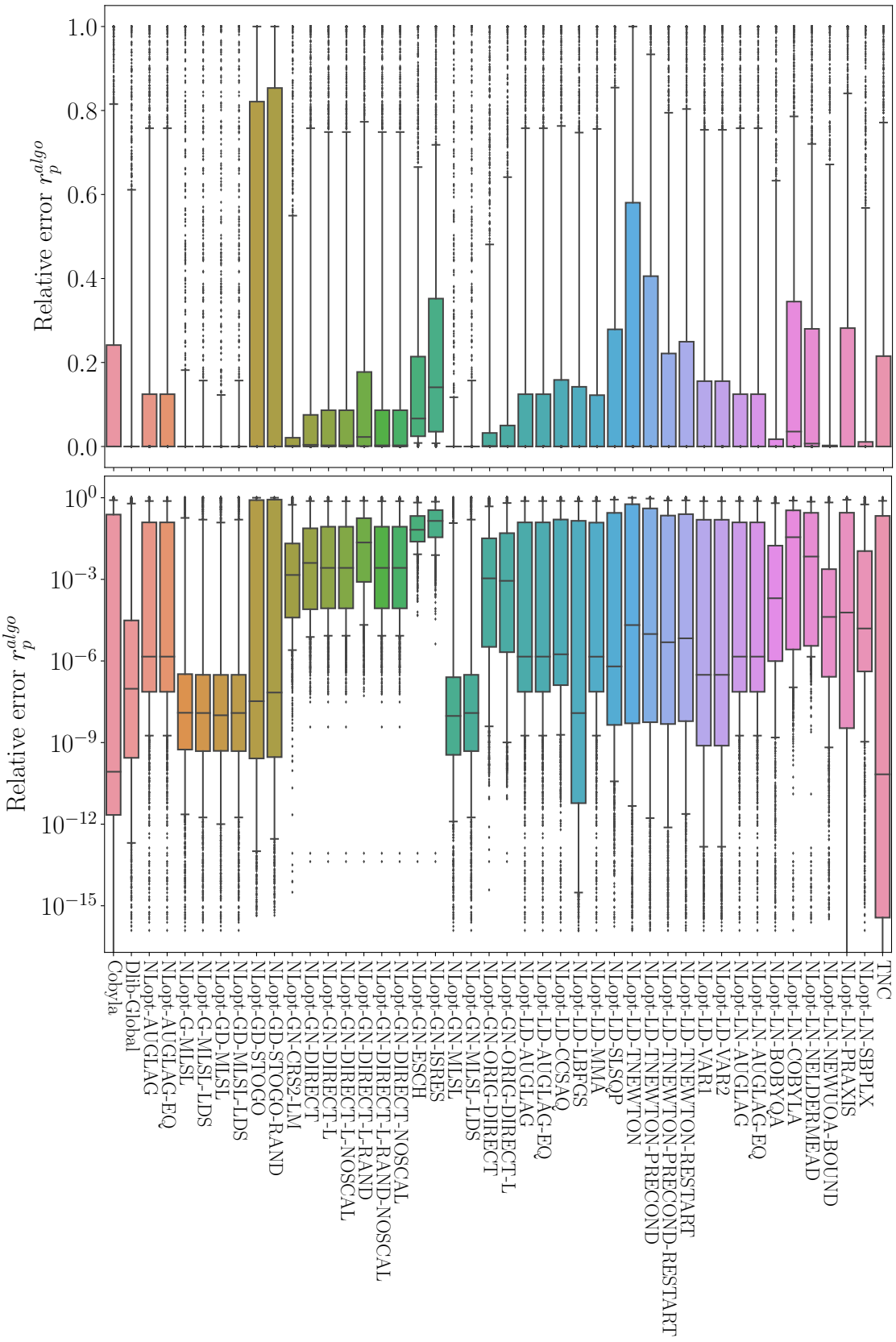


Figure 5.5: Relative error between the maximum EI value found by the solver and the optimal EI for all the benchmarked solvers: the upper graph is in linear scale, the lower graph is in log scale.

of the same order of magnitude for functions with different input dimensions and can be fairly aggregated in Figure 5.6. The conclusions remain the same with MLSL being the most efficient algorithm, followed by Dlib-Global.

Remark 5.1. The log-scale graph of Figure 5.5 allows to compare the small relative errors r_p and is useful to rank the best algorithms. However, we argue that two solvers with a small relative error but of different order of magnitude on a given EI function may not lead to a real difference in the subsequent iteration of EGO. Suppose that we have two solvers with respective relative errors r_p of 10^{-7} and 10^{-4} on a given EI function. Such small errors may only reflect that one of the solvers has better exploited the local area around the optimum than the other, meaning that the distance between the maximizers of the EI proposed by both solvers is small. In this case, the behavior of the next EGO iteration is not affected.

On the other hand, if the relative error r_p is large for some solver, it means that the local area of the optimum has not been found. If this error is close to 1, this even means that the solver has not managed to escape the flat areas of the EI function. In this case, the maximizer provided by the solver can be far from the optimum, hence completely modifying the behavior of EGO. This is why we have provided Figure 5.6 that shows the error on the maximizer.

Ranking the algorithms with respect to a high percentile is then more relevant than looking at the low percentiles. For example, even if the lower quartile of r_p is around 10^{-4} for the DIRECT algorithm against 10^{-7} for most algorithms (of which TNEWTON for instance), the third quartile is around 0.1 and the 90-th percentile is around 0.8 which makes it better than TNEWTON (third quartile greater than 0.2 and 90-th percentile greater than 0.8). ◊

To conclude the benchmark, the solver MLSL is the best performing algorithm of the benchmark with a 90-th percentile that is around 0.2, which is much better than any other solver. Hence, when using EGO in Chapter 6, the EI is maximized with a MLSL solver, more precisely we choose the GD-MLSL-LDS variant.

5.4 Conclusion

In this chapter, we have focused on two practical aspects for the implementation of EGO.

First, we propose a variant of the EGO algorithm, called EGO-FSSF where the initial design is built sequentially while retaining space-filling properties. Hence, we can adaptively choose the size of the initial design, based on criteria assessing the quality of the metamodel, whereas in the classical version of EGO, the size of the initial DOE is fixed with some heuristic criteria. With the adaptive initial design size, the budget of evaluations of the objective function is efficiently split between the initial exploratory step and the infill step. A comparison of the performance of EGO and EGO-FSSF is carried out in Chapter 6.

The second focus of this chapter is the choice of an efficient solver for EI maximization, which is an essential task within EGO. A specific benchmark of EI functions has been designed and we have assessed many solvers available through the software OpenTURNS. This benchmark provides quantitative arguments for the selection of a solver for EI maximization, whereas it is usually based on heuristic considerations in the literature. The MLSL algorithm is the best performing solver of the benchmark and it will be used in the practical implementation of EGO in Chapter 6.

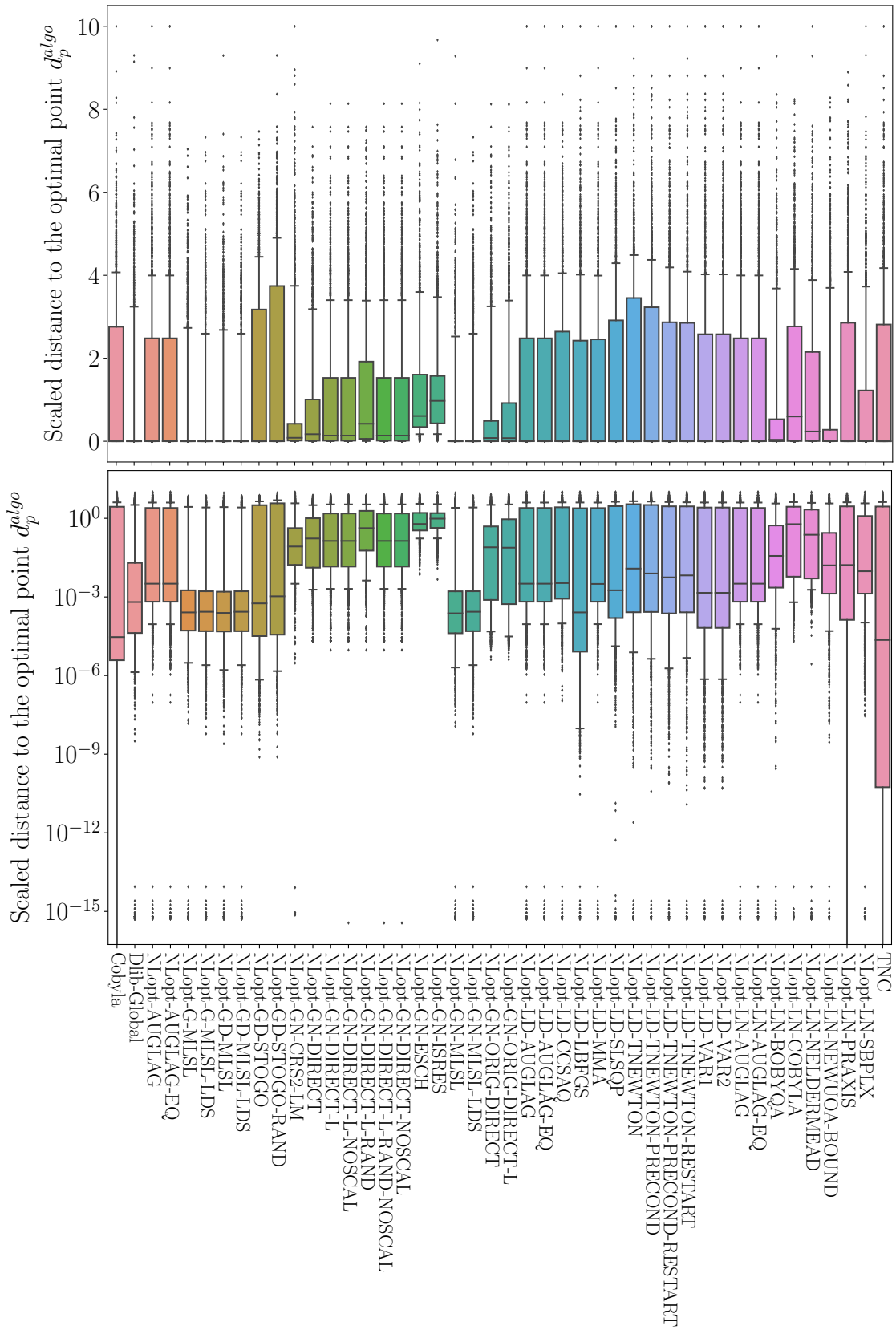


Figure 5.6: Distance (scaled by the dimension) between the EI maximizer found by the solver and the optimal point for all the benchmarked solvers: the upper graph is in linear scale, the lower graph is in log scale.

6 Benchmark and industrial application of EGO and MADS

The biggest risk is not taking any risk... In a world that is changing really quickly, the only strategy that is guaranteed to fail is not taking risks.

Mark Zuckerberg

Contents

6.1 Introduction

In Chapters 3 and 4, we have investigated two algorithms, EGO and MADS, that are suited to solve the blackbox minimization problem (2.1). The interest in studying the general problem (2.1) is that the industrial maintenance optimization problem described in Section 2.1 fits into this framework.

Contributions. We consider the EGO algorithm 2, the EGO-FSSF algorithm 5 and the MADS algorithm 4. In Section 6.2, these algorithms are benchmarked on the COmparing Continuous Optimizers (COCO) platform [Hansen et al., 2021], a tool that provides an experimental setup for blackbox optimization algorithms. This benchmark allows to fairly compare the performance of blackbox methods in a very general setting. In Section 6.3, we plug the algorithms EGO-FSSF and MADS to the software VME, which is the tool used for the evaluation of the cost of a maintenance strategy at EDF. We compare the performance of EGO-FSSF and MADS on small industrial test cases. To our knowledge, this is the first time that EGO is used for maintenance optimization. We mention that MADS has been applied for the maintenance scheduling of turbines in [Aoudjit, 2010, Alarie et al., 2019].

6.2 Benchmarking EGO and MADS on COCO

Benchmarking blackbox optimization algorithms is not an easy task as their performance may depend on some properties of the objective function, such as its smoothness, eventual symmetries or the presence of local minima. However, these features are not known by the algorithm in the blackbox context. The optimization algorithms may also be stochastic, meaning that two runs on the same problem may not give the same output. Hence, appropriate metrics must be used for performance assessment.

In order to rigorously carry out this tedious benchmarking task, we use the COmparing Continuous Optimizers (COCO) platform [Hansen et al., 2021]. The COCO framework provides an automatized benchmarking procedure (design of the test bed, design of the experimental set up, generation of data output and post-processing of results) that allows for the reproducibility of results and a fair performance comparison between algorithms.

6.2.1 Presentation of the test bed

Several test beds are available in COCO for different purpose: noiseless or noisy, single or multi-objective, real parameter or mixed-integer. In this chapter, we use the standard noiseless benchmark (single objective, continuous variables). Each function in the benchmark is designed so as to have different features. Thus, we can analyze the behavior of the optimization algorithms in different situations. For example, can our algorithm exploit the fact that the function is linear, or escape local minima? We should keep in mind that we are in the blackbox context so the properties of the function are not given to the algorithm. Note also that some problems of the test bed are very challenging and may be more difficult than practical applications.

The test bed is constituted of 24 functions defined on $[-5, 5]^d$ where the dimension d is chosen by the user. The functions are designed so that the location and the value of the optimum are known. Now, we describe the properties that are considered for

the construction of the test bed. The presentation is based on [Hansen et al., 2009], where the explicit formula for each function of the test bed can be found.

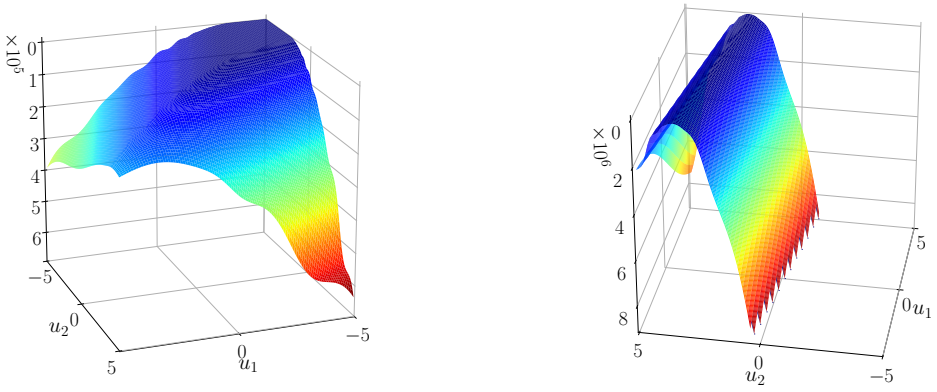
6.2.1.1 Conditioning

In the case of a convex quadratic function $f(u) = \frac{1}{2}u^\top Hu$, where H is a symmetric definite positive matrix, the conditioning of f is the condition number of the Hessian matrix H, see Definition 3.7. Loosely, for more general functions, the conditioning $\kappa_f(u)$ of f at $u \in U^{\text{ad}}$ is the ratio:

$$\kappa_f(u) = \lim_{\varepsilon \to 0} \frac{|f(u + \varepsilon e_1) - f(u)|}{|f(u + \varepsilon e_2) - f(u)|},$$

where $e_1, e_2 \in \mathbb{R}^d$ are the unit directions where f is the "steepest" and the "flattest" respectively. The directions e_1 and e_2 depend on u, the conditioning is a local measure. The test bed contains functions with conditionings up to 10^6. Figure 6.1a shows a function with a low conditioning whereas the function of Figure 6.1b has a high conditioning. Ill conditioning is one of the most common challenges in optimization.

Remark 6.1. In all the plots representing functions from the COCO test bed (Figures 6.1-6.5), the z-axis is reversed for a better visualization. ◊



(a) Low conditioning.

(b) High conditioning.

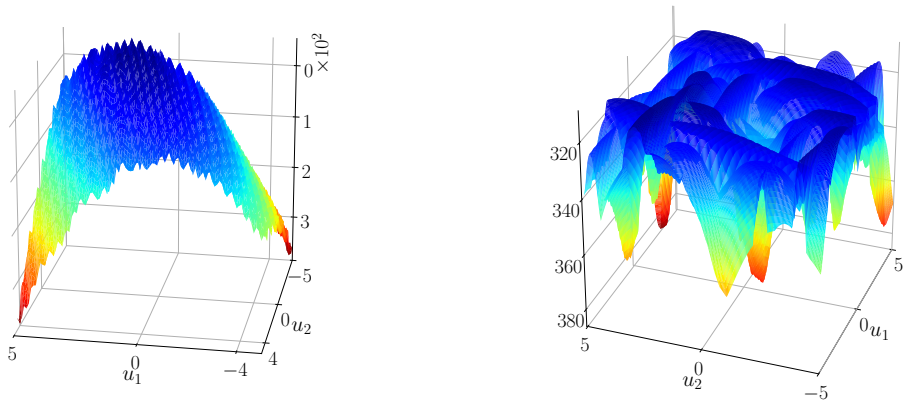
Figure 6.1: Functions of the test bed with different conditionings.

6.2.1.2 Multimodality

A function is said to be multimodal when it has several local optima. Multimodality is the other main difficulty of global optimization besides ill conditioning. Figure 6.2 shows two examples of multimodal functions from the test bed. The Rastrigin function (Figure 6.2a) has a regular structure whereas the Gallagher function (Figure 6.2b) is designed so as to present 101 local minima with randomly chosen location and value.

6.2.1.3 Regularity

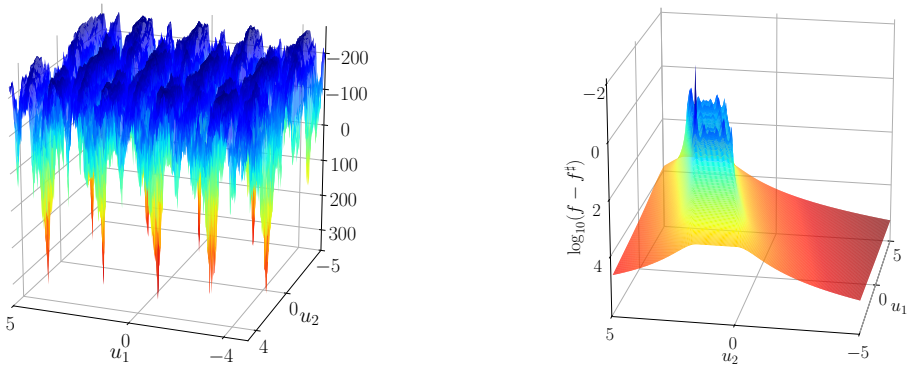
The test bed contains some functions that are irregular, but still continuous. As functions that are obtained from formula are often highly regular, a non-linear transformation is applied to introduce some irregularities. Figure 6.3 shows two examples of



(a) Rastrigin function. (b) Gallagher function.

Figure 6.2: Examples of multimodal functions of the test bed.

irregular functions from the test bed. The Weierstrass function (Figure 6.3a) has a perturbed periodic structure. The Schwefel function (Figure 6.3b) is irregular only near the local minima. Note that for the Schwefel function, the plot represents $\log_{10}(f - f^\sharp)$ where $f^\sharp$ is the optimal function value.



(a) Weierstrass function. (b) Schwefel function.

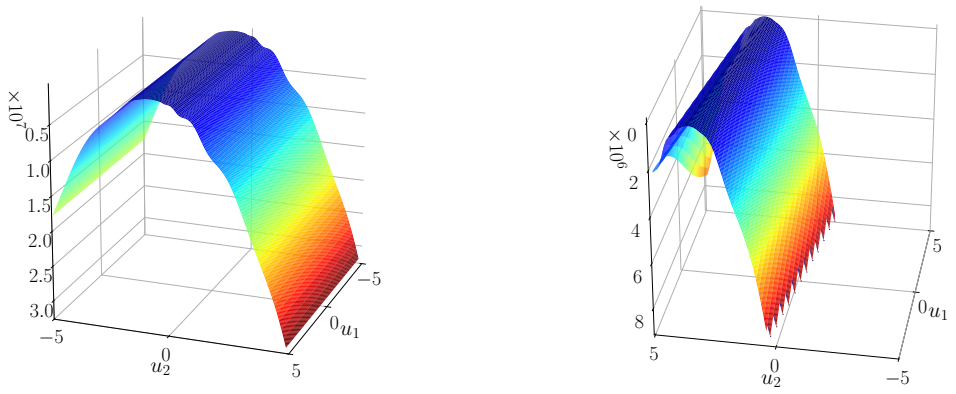
Figure 6.3: Examples of irregular functions of the test bed.

6.2.1.4 Separability

A function is separable if it is the sum of functions of one scalar variable. The optimization of a d-dimensional separable function boils down to solving d problems in dimension 1. Non separable functions are then much more difficult to optimize than separable functions. In the test bed, non separable functions are constructed from separable functions to which a rotation operator is applied. Figure 6.4a and 6.4b represent respectively a separable ellipsoidal function and its non separable counterpart.

6.2.1.5 Symmetry

Some stochastic optimization algorithms rely on the Gaussian distribution to select iterates. Symmetric functions may favor these operators [Hansen et al., 2009]. To avoid

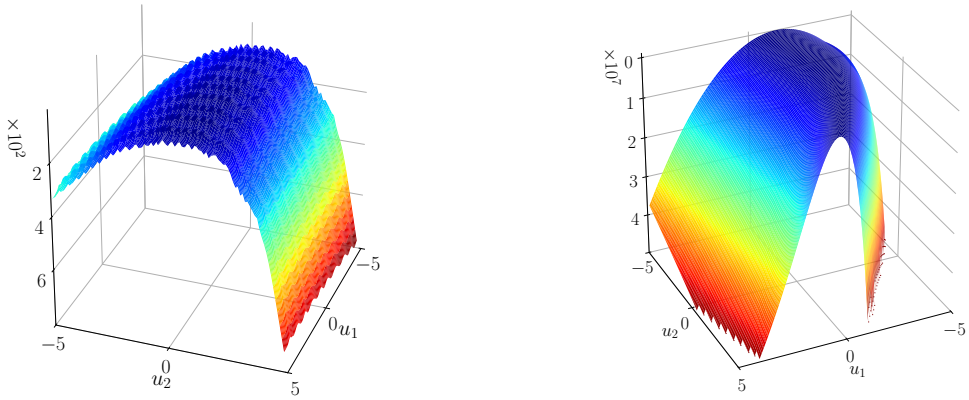


(a) Separable version.

(b) Non separable version.

Figure 6.4: Ellipsoidal function.

this bias, the test bed includes asymmetric functions that are generated by applying a symmetry breaking operator to symmetric functions. Figure 6.5 gives examples of asymmetric functions from the test bed.



(a) Rastrigin function.

(b) Bent cigar function.

Figure 6.5: Examples of asymmetric functions of the test bed.

6.2.1.6 Final remarks on the test bed

For each of the 24 functions of the test bed, 15 *instances* are defined, corresponding to translations of the original function. This allows to randomly choose the location and the value of the optimum. Using different instances of the same function is particularly interesting for stochastic algorithms as we can then estimate their probability of reaching a given target. The benchmarked algorithms run on 15 instances of the 24 functions of the test bed in dimension 2, 3, 5 and 10, resulting in $15 \times 24 \times 4 = 1440$ optimization problems to solve.

6.2.2 Parameters of the algorithms

In this section, we specify the experimental setting for the benchmark. We use the version 2.3.3 of COCO. The different variants of EGO are implemented with the software OpenTURNS (version 1.15) [Baudin et al., 2017]. The algorithm MADS is available through the software NOMAD (version 3.9.1) [Le Digabel, 2011]. The numerical experiments run on a processor Intel® Core™ i7-6700HQ @2.60GHz. We give the parameters that are used for each algorithm in the benchmark.

6.2.2.1 For EGO

Three variants of the EGO algorithm are considered. First, we consider the classical EGO algorithm 2, referred as EGO-LHS, which uses a fixed size LHS initial design. Then, we consider two variants called EGO-FSSF-lo and EGO-FSSF-hi, described by Algorithm 5, which use a FSSF initial design but with different values of the quality threshold $(Q^2_{\min}, \text{PVA}_{\max})$. EGO-FSSF-hi requires a *highly* accurate metamodel to start the infill step whereas EGO-FSSF-lo only requires a coarse metamodel (*i.e.* of *lower* quality). The three variants of EGO use the final local optimization step with L-BFGS. Parameters of the algorithms are summarized in Table 6.1.

Algorithm	EGO-LHS	EGO-FSSF-lo	EGO-FSSF-hi
Initial design type	Optimized LHS	FSSF-fr	FSSF-fr
Initial design size	$10d$	Adaptive	Adaptive
Threshold $(Q^2_{\min}, \text{PVA}_{\max})$	/	$(0.4, 1)$	$(0.7, 0.3)$
Max budget proportion for the initial design p_{init}	/	0.5	0.5
Budget proportion the final local search p_{local}	0.2	0.2	0.2
Common parameters			
Evaluation budget	$100d$		
Infill criterion	EI		
EI maximization algorithm	NLopt-GD-MLSL-LDS		
Covariance function	Matérn 3/2		
Nugget	10^{-6}		
Local search algorithm	L-BFGS		

Table 6.1: Parameters of the EGO algorithm.

We add some comments on the chosen parameters:

- The maximum proportion of the budget p_{init} that can be used for the sequential initial design is set to 50%. We use this large value for p_{init} in order to ensure that the infill step in EGO-FSSF is most of the time launched thanks to the good accuracy of the metamodel rather than because of budget limitation.

- For each variant, the infill criterion is the EI. We use the MLSL-LDS algorithm [Kucherenko and Sytsko, 2005] for EI maximization as it is the best performing algorithm from the benchmark of Section 5.3.
- We choose a Matérn covariance function with $\nu = 3/2$ as non differentiable functions are present in the benchmark. Moreover, as explained in §3.2.3, this kernel makes the numerical inversion of the covariance matrix easier than with a Gaussian kernel for instance and is also a robust choice in case of covariance misspecification.
- A fixed nugget value of 10^{-6} is used, as recommended by [Mohammadi, 2016, Chapter 5].

We also give complementary information on the solvers that are used for the estimation of the hyperparameters θ and σ^2 of the metamodel at each iteration of EGO. During the initial design step, it is particularly important to get the best possible metamodel to efficiently start the infill step. By default, OpenTURNS uses TNC (a truncated Newton method) which is a local optimization algorithm while the likelihood maximization problem may be non-convex. This is why we choose to change the default solver for maximum likelihood estimation.

- In EGO-LHS, the metamodel is computed only at the end of the initial step. We use a multistart L-BFGS algorithm [Nocedal, 1980] for the estimation of the hyperparameters. The starting points of L-BFGS are generated by a LHS design.
- In EGO-FSSF, the metamodel is updated after each new evaluation in the initial sequential design. As the the hyperparameters estimation should not change much between two evaluations, we still use the TNC algorithm but initialized at the current estimated hyperparameters. However, it may happen that the estimated correlation lengths θ are small with respect to the distance between design points, meaning that the metamodel is noisy and of poor quality. In this case, the TNC algorithm fails to provide better hyperparameters at the following iteration because it is trapped in a local optimum. In this situation, we estimate the hyperparameters with the multistart L-BFGS algorithm.

These choices are solely based on heuristic arguments but they might slightly improve the overall performance of EGO compared to the default implementation of OpenTURNS.

6.2.2.2 For MADS

We use the OrthoMADS implementation of MADS presented in §4.3.3, which details the choice of the pattern, the polling directions, the initial mesh parameters and the mesh size update rule. We use the default search step strategy of NOMAD, based on quadratic models [Conn and Le Digabel, 2013]. Similarly as for EGO, the maximum evaluation budget is set to $100d$ calls to the objective function. The parameters of MADS are summarized in Table 6.2.

6.2.3 Performance assessment with runlength-based targets

The indicators used for performance assessment in the COCO platform as well as the computation methodology are exhaustively described in [Hansen et al., 2016]. Here, we

Algorithm	MADS
Implementation	OrthoMADS
Search step strategy	Based on quadratic models
Evaluation budget	$100d$

Table 6.2: Parameters of the MADS algorithm.

present the notion of runlength-based target that is used for a meaningful comparison of the algorithms in the benchmark.

Usually, the run of an algorithm is declared *successful* when it reaches a target of the form $f_{tar} = f^\sharp + \Delta f$ where $f^\sharp$ is the optimal value and Δf is a desired precision. For a given precision Δf, the difficulty of reaching the target varies greatly for the different functions of the test bed. For example, reaching a target with a precision $\Delta f = 10^{-6}$ is much easier for the regular, low conditioned function of Figure 6.1a than for the Gallagher function of Figure 6.2b. As we are in the expensive blackbox setting, only a low budget is available for the optimization. Hence, defining similar fixed targets for all functions of the benchmark may not give relevant indicators on the performance of the algorithms. A given target may indeed be too easy for some functions of the test bed, so that each algorithm reach it in few iterations, whereas, it may be too hard for other functions, so that no algorithm can reach it. No information on the comparative performance of the algorithms is gained from those runs.

This is why COCO uses the notion of *runlength-based targets*. It aims at defining target values that depend on the objective function in order to represent the *same level of difficulty*. Instead of choosing a set of target precisions, we rather choose a set of *reference budgets*. We also suppose that a reference algorithm has already run on the benchmark. For a reference budget $b > 0$, we look, for each function, at the precision $\Delta f(b)$ that has been reached by the reference algorithm. The precision $\Delta f(b)$ is referred as the runlength-based target b for the function f. Runlength-based targets are different for each function but represent the same level of difficulty as they require the same budget to be reached by the reference algorithm. In the numerical experiments, 31 runlength-based targets are considered, with reference budgets evenly spaced between $0.5d$ and $50d$ on the log scale, where d is the problem dimension. For a given dimension d, we denote by:

- $\{f_{i,j}^d\}_{(i,j)\in I\times J}$, with $I = \{1,\ldots,24\}$ and $J = \{1,\ldots,15\}$, the set of all function instances of the benchmark where $f_{i,j}^d$ is the j-th instance of the i-th function in dimension d.

- $\{b_k^d\}_{k\in K}$, with $K = \{1,\ldots,31\}$, the set of reference budgets.

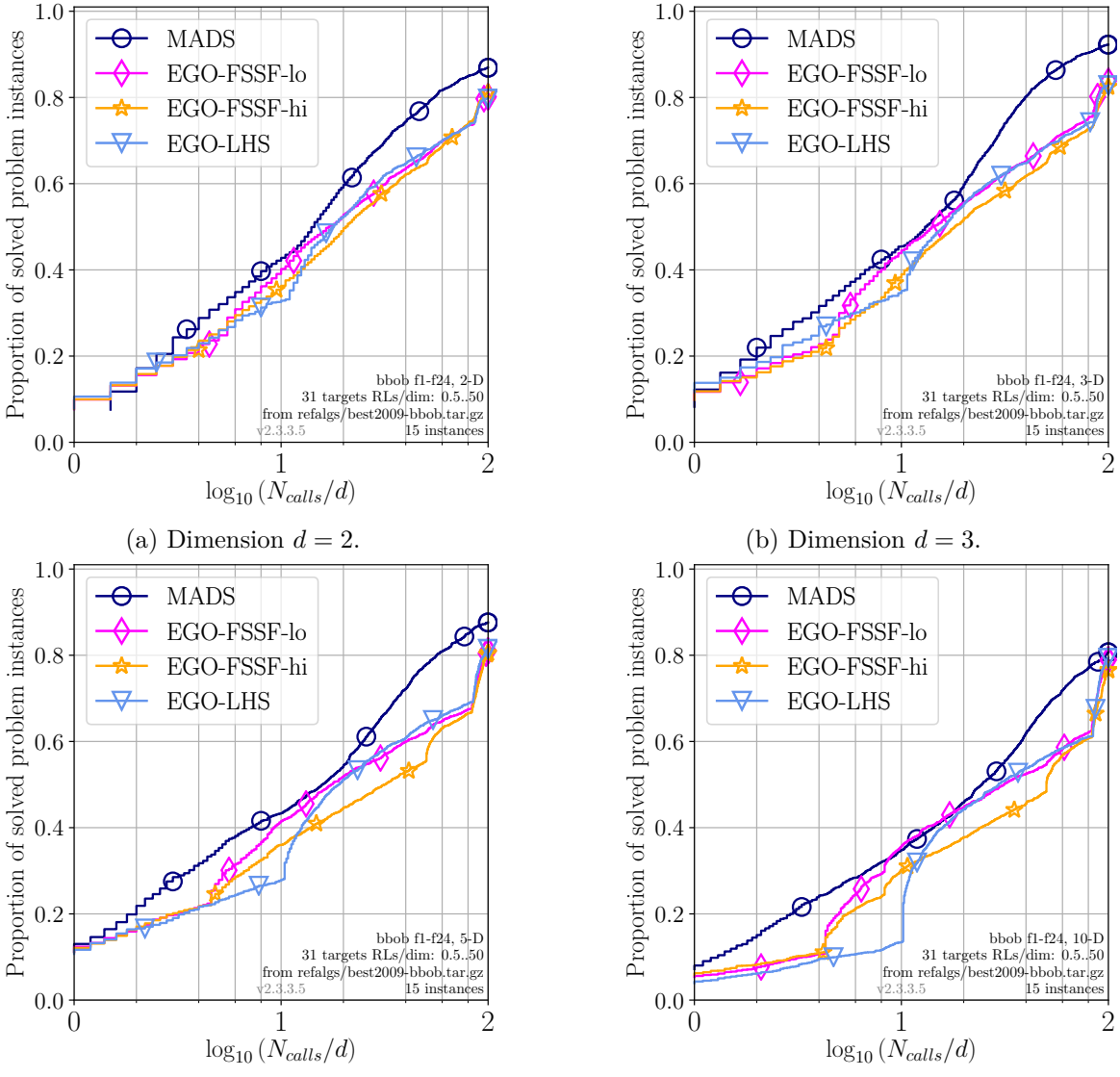
The reference algorithm in COCO is the *artificial* best algorithm of the workshop BBOB-2009 [Hansen et al., 2010]. In the BBOB-2009 workshop, 31 algorithms have run on the benchmark. For each reference budget and function, the runlength-based target is taken as the best precision attained by the algorithms of the benchmark. The *real* algorithm that defines the runlength-based target can then be different for each budget and function.

6.2.4 Benchmark results

We can now perform a comparative study of the algorithms using the data generated by COCO.

6.2.4.1 Aggregated results over all functions and targets

We start by analyzing aggregated results over all functions and all targets. Figure 6.6 shows the proportion of problems solved within a given budget. Each pair $(f^d_{i,j}, b^d_k)$, with $(i, j, k) \in I \times J \times K$, defines a problem instance with the runlength-based target $\Delta f^d_{i,j}(b^d_k)$. The graphs of Figure 6.6 give, for a given budget b on the x-axis, the proportion of problem instances among the set $\{(f^d_{i,j}, b^d_k)\}_{(i,j,l)\in I\times J\times K}$ that are solved within the budget b. Each subplot corresponds to a different dimension d. Overall, MADS solves around 5% more instances than any of the EGO variants.



(a) Dimension $d = 2$.

(b) Dimension $d = 3$.

(c) Dimension $d = 5$.

(d) Dimension $d = 10$.

Figure 6.6: Proportion of problems solved within a given budget, aggregated over all targets and functions, N_{calls} is the number of calls to the objective function.

We can clearly distinguish the three steps of the EGO algorithm on the graphs. Let us focus on Figure 6.6d as the behavior is the most easily visible. With a low budget,

the proportion of solved problems increases slowly, this is the initial exploration step, done either with a LHS or a FSSF design. Then, we see a clear change of slope showing that we enter the infill step, driven by the EI maximization. The EI exploits the metamodel built in the initial step to solve many instances in few iterations. Then, the slope slowly decreases before another break, slightly before the maximum budget is used. This corresponds to the beginning of the local search step, that allows to go from 60% to 80% of solved instances (in dimension 10) and to almost reach the same efficiency as MADS. These plots confirm the usefulness of the final local search that has already been highlighted in [Mohammadi, 2016].

There is little difference in the final performance of the three variants of EGO. However, we notice differences at intermediate budgets. The initial design size is adaptive for the EGO-FSSF variants. Hence, for the easiest functions, the EGO-FSSF variants start the infill step sooner than EGO-LHS. This explains that the change of slope arrives earlier with the EGO-FSSF variants. Moreover, for EGO-FSSF-hi, we notice another change of slope slightly before the local search, corresponding to a budget of $50d$ evaluations, which is the maximum budget allocated for the initial design. This change of slope is particularly visible in dimension $d = 5$ and $d = 10$. This reflects the fact that for some functions, the metamodel has not reached the quality threshold. No such change of slope is visible for EGO-FSSF-lo, meaning that the quality threshold is reached for a greater number of functions.

After using the whole budget of $100d$ evaluations, the performance of the three EGO variants are similar. If we compare two variants with different budgets allocated to the initial design, the variant with the greater budget for the initial design has a more efficient infill step due to a metamodel of better quality. This compensates the additional budget used in the initial step compared to a variant with a coarser initial design. This observation is consistent with the conclusions of [Picheny et al., 2013] that compares EGO with LHS designs of different sizes and finds that the size of the initial DOE is not critical.

The progress of MADS is much more regular than that of EGO, which makes MADS more robust to interruptions of an experiment. Indeed, suppose that we launch the algorithms with a predefined evaluation budget and that for some reason the run is interrupted before the end. Then, if we are still in the initial design step or just before the final search for EGO, the performance gap between EGO and MADS is significantly larger than at the end of the run (see Figure 6.6): up to 20% more instances are solved by MADS for a budget of $80d$ evaluations, *i.e.* before launching the local search in EGO. The performance of EGO is conditioned by the complete realization of the three steps of the algorithm (initial design, infill step, local search), whereas MADS can be interrupted without consequences.

6.2.4.2 Aggregated results over all targets and by function group

Figure 6.7 shows the results aggregated by function group in dimension $d = 10$. The EGO algorithms perform particularly well compared to MADS on multimodal functions with global structure (Figure 6.7d), such as the Rastrigin function (Figure 6.2a). MADS only solves around 60% of problems of this kind compared to more than 80% for all the other groups. Our interpretation is that MADS does not manage to escape local minima in this case. Indeed, suppose that MADS has found a local minimum for a function like the Rastrigin function (Figure 6.2a). In order to escape this minimum, the search step must produce a candidate point in another basin of attraction which is

better than the current best point. However, the search step of MADS uses a quadratic model around the current best point which may not be the best strategy for this task.

Figure 6.7 allows to highlight some differences between the behavior of the three EGO variants, especially in the budget that is used for the initial design. Separable functions (Figure 6.7a) correspond to the easiest problems of the test bed. Hence, only few iterations are needed in the initial step to get the coarse metamodel required in EGO-FSSF-lo. For EGO-LHS, the initial design always uses $10d$ evaluations, which, in this particular case, is more than for EGO-FSSF-lo but less than for EGO-FSSF-hi. For multimodal functions without global structure (Figure 6.7e), it is more difficult to have a metamodel of good quality. In this case, even the coarse requirement on the metamodel is not met for most functions: there is a clear change of slope after $50d$ evaluations for EGO-FSSF-lo. These observations confirm that the size of the initial design in the EGO-FSSF variants is adapted to the difficulty of the problem. However, no matter how much effort is devoted to the initial design, the three variants exhibits very similar performance after the $100d$ evaluations.

6.2.4.3 Conclusions of the benchmark

The COCO benchmark provides fair quantitative indicators to guide the choice of an optimization algorithm in a blackbox context. Overall, MADS is more efficient than EGO. Among the three variants of EGO, the performance are very similar after $100d$ evaluations. However, in the case where an expensive simulation is interrupted before the end, for example suppose that only $50d$ evaluations are done over the $100d$ that were initially planned, EGO-FSSF-lo will perform better in most cases as the infill step may have already been launched. This is why we advise to use EGO-FSSF-lo among the three EGO variants.

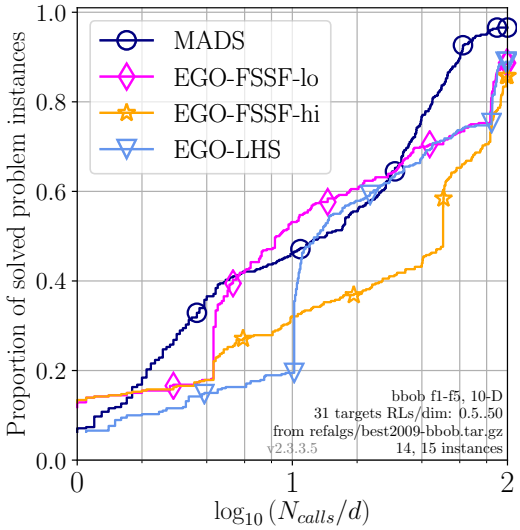
These conclusions only hold when no information is known about the objective function. We have seen that for highly multimodal functions with global structure, EGO performs better than MADS[1]. Other studies [Mohammadi, 2016, Rehbach et al., 2020] confirm that multimodal functions are a favorable framework for EGO with EI as infill criterion. More generally, if some properties of the function are known, we can refer to the aggregated results by function group to guide the decision. The performance of a blackbox algorithm is in fact highly dependent on the problem at hand.

For a given practical problem, we strongly believe that, when possible, we should try to gain information on the objective function in order to guide the choice of the algorithm. If no information is available, we choose MADS. In the next section, we apply EGO and MADS to solve a first industrial maintenance optimization problem.

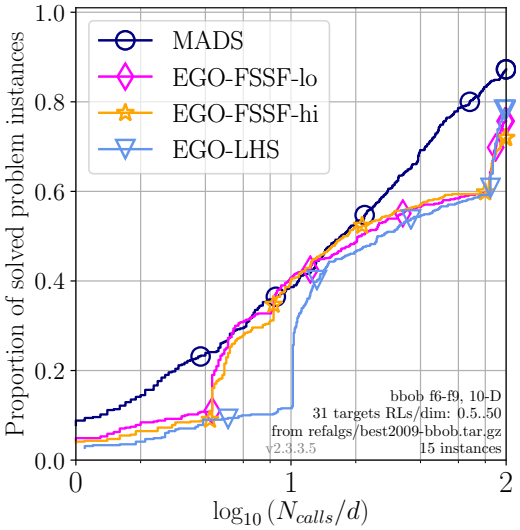
6.3 Application to an industrial maintenance optimization problem

We are interested in the industrial maintenance optimization problem described in Section 2.1. In this section, we solve a first industrial case by plugging the blackbox algorithms EGO-FSSF-lo and MADS to the software VME, which is the simulation model that allows to evaluate the expected LCC of a maintenance strategy. The

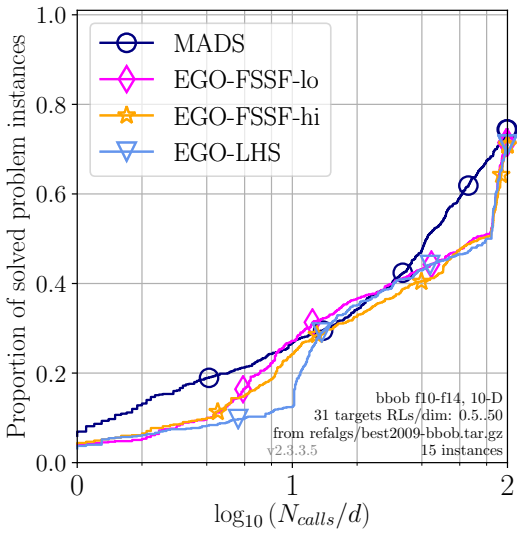
[1] The NOMAD software provides the Variable Neighborhood Search (VNS) option for MADS. VNS is a strategy to espace local minima and has not been used in the thesis. Enabling the VNS option would have probably improved the performance of MADS on multimodal functions.



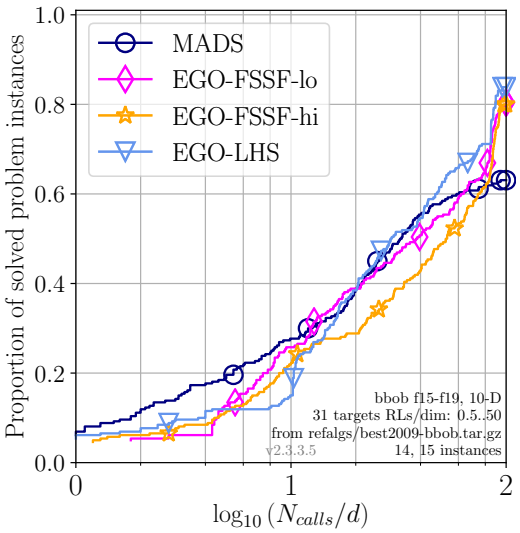
(a) Separable functions.



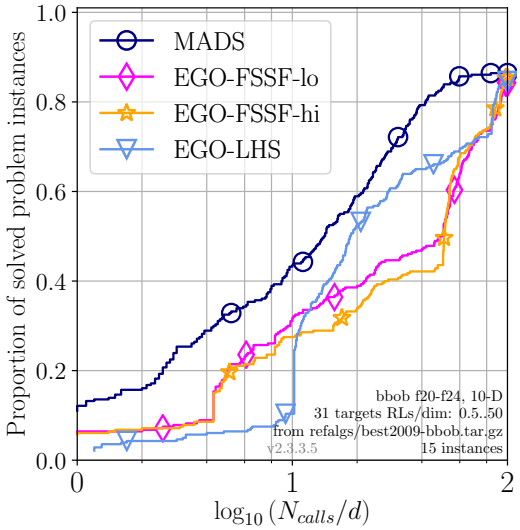
(b) Low or moderate conditioning.



(c) High conditioning and unimodal.



(d) Multimodal with global structure.



(e) Multimodal without global structure.

Figure 6.7: Proportion of problems solved within a given budget aggregated by function group in dimension $d = 10$, N_{calls} is the number of calls to the objective function.

expected LCC is the objective function we seek to minimize. As the EGO variants exhibit very similar performance on the COCO benchmark, we only consider the EGO-FSSF-lo variant, following the recommendation of §6.2.4.3.

6.3.1 Description of the test cases

We consider a system of n components with $n \geq 1$ sharing a common stock of spare parts, on a horizon of $T = 40$ years. For this test case, we only consider periodic PM strategies. A maintenance strategy is represented by a vector:

$$u = (u_1, \ldots, u_n) \in U^{\text{ad}} = [0, T]^n \ , \tag{6.1}$$

where u_i is a continuous variable that represents the periodicity of the PMs of component $i \in \{1, \ldots, n\}$. More precisely, this means that component i undergoes a PM at dates $\left\{k \times u_i,\ 1 \leq k \leq \lfloor \frac{T}{u_i} \rfloor \right\}$.

We focus on periodic maintenance strategies as they can be described using only one parameter per component. These strategies are a subset of general maintenance strategies where maintenance decisions can be made each year for each component. Therefore, the optimal solution on the set of periodic strategies is worse than on the set of general strategies. However, if we were to consider general strategies, the search space would be described with T parameters per component, which is too large to perform the optimization with blackbox methods. This is why we restrict the search to periodic strategies in this part.

The industrial problem of minimizing the expected LCC can expressed as:

$$\min_{u \in U^{\text{ad}}} \mathbb{E}\left(j(u, \boldsymbol{W})\right) \ . \tag{6.2}$$

where $\boldsymbol{W}$ is a random variable on a probability space $(\Omega, \mathcal{A}, \mathbb{P})$ that models the random failures of the components. The cost $j(u, \boldsymbol{W})$ is the LCC for the maintenance strategy u. In practice, the software VME uses the Monte-Carlo method to estimate $\mathbb{E}\left(j(u, \boldsymbol{W})\right)$. Hence, we do not have access to the true value of the objective function. This is why, in Section 2.1.3, we say that the evaluations made by VME are *noisy*.

In the numerical experiments, the *noisy* aspect is handled by considering a Monte-Carlo approximation of Problem (6.2). We fix a set of $Q = 100$ realizations $w_1, \ldots, w_Q$ of the random variable $\boldsymbol{W}$ and solve:

$$\min_{u \in U^{\text{ad}}} \frac{1}{Q} \sum_{q=1}^{Q} j(u, w_q) \ . \tag{6.3}$$

Problem (6.3) is deterministic and the objective function can be evaluated exactly with VME. Therefore, we can use the algorithms EGO and MADS that have been presented in the *noiseless* case in Chapters 3 an 4.

We consider systems with the characteristics given in Table 6.3. We consider four different test cases, respectively with 2, 3, 5 and 10 components. These test cases represent only small instances of the industrial problems that can be encountered at EDF. From (6.1), a maintenance strategy is parametrized with one variable for each component, hence the number of optimization variables in the problem is equal to the number of components n. The failure distributions of the components are Weibull distributions of parameters $\beta > 0$ and $\lambda > 0$, denoted by $\text{Weib}(\beta, \lambda)$. The definition of the Weibull distribution is given in Definition B.8.

Parameter	**Value**		
Number of components n	2, 3, 5 or 10		
Initial number of spare parts	$\lfloor \frac{n}{5} \rfloor$		
Horizon	40 years		
Delay for the spare parts supply	0.1 year		
Forced outage cost	30000 k€/ month		
	Comp. 1	**Comp.** 2	**Comp.** $i \geq 3$
PM cost	50 k€	50 k€	50 k€
CM cost	100 k€	250 k€	200 k€
Failure distribution	Weib(2.3, 10)	Weib(4, 20)	Weib(3, 10)
Mean time to failure	8.85 years	18.13 years	8.93 years

Table 6.3: Characteristics of the industrial system.

For EGO-FSSF-lo and MADS, we use the same settings as for the COCO benchmark, described in Tables 6.1 and 6.2. In particular, the maximum number of evaluations of the objective function is set to $100n$ where n is the number of components of the system.

6.3.2 Numerical results

We present the results of the optimization with EGO-FSSF-lo and MADS plugged on VME for the four test cases. In addition to the comparison of the performance between EGO-FSSF-lo and MADS, the analysis strongly focuses on the behavior of the algorithms with respect to the dimension of the problem, in order to highlight the potential difficulties to solve the largest instances of the maintenance problem we wish to consider in this thesis.

In Figure 6.8 we show the evolution of the objective function value along the iterations for the two algorithms. The bullets on the curve denote the evaluations after which an improvement effectively occurs. In Table 6.4, we report the best function value and the running time for each algorithm.

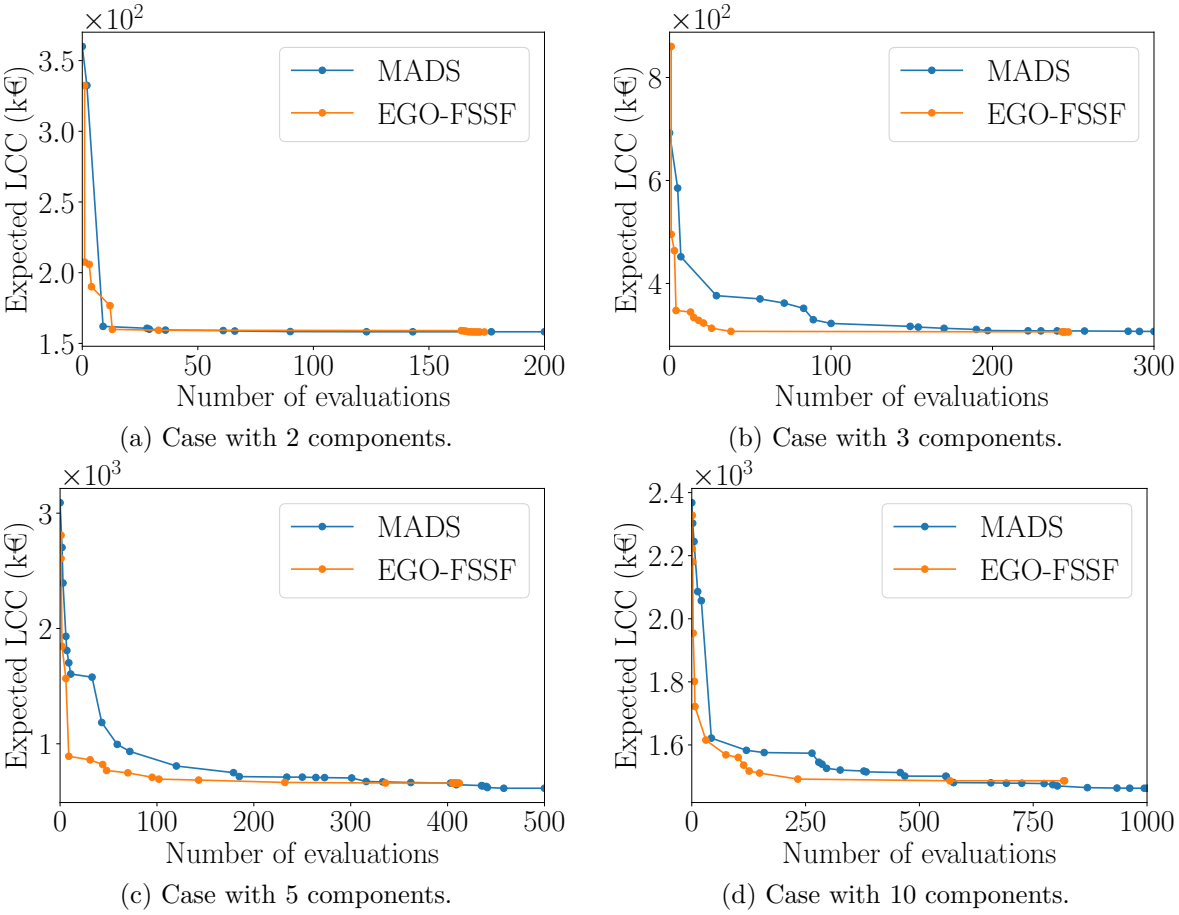
	EGO		**MADS**	
Test case	Optimal value	Running time	Optimal value	Running time
2 components	158.07	49.6s	158.22	6.49s
3 components	305.70	234s	306.83	11.7s
5 components	659.27	1830s	614.22	23.4s
10 components	1486.4	3.54×10^4s	1462.7	66.2s

Table 6.4: Optimal function value and running time of EGO and MADS.

On the 2-component case (Figure 6.8a), both algorithms find a good solution after around 10 evaluations. For the other test cases, we see that EGO is more efficient than MADS in the first iterations, probably due to its fully exploratory behavior during the initial design step. However, as more evaluations are carried out MADS manages to improve the objective and returns a solution that is better than EGO by 7% in the case with 5 components and by 1% in the case with 10 components. For the cases with

2 and 3 components, there is less than 0.5% difference in the optimal value found by EGO and MADS, but this time in favour of EGO. Therefore, EGO is slightly better than MADS in low dimension but when the number of components grows, MADS has a significant advantage.

For EGO, after an efficient start in the first evaluations, not much improvement of the objective can be observed (Figure 6.8). We point out that the last group of bullets on the EGO curves corresponds to the final local search. The orange curves do not extend to the maximum allowed evaluation budget, meaning that the local search stops before using all the budget. This behavior can be explained by the fact that the dates of PM in VME are specified with a precision to the day. This means that in a small neighborhood of a given $u \in U^{\text{ad}}$, all the points correspond to the same day of PM in VME and therefore have the same objective function value. Hence, if the local search is done in a neighborhood of the current iterate that is small enough, the local optimizer sees that the function has a zero gradient and stops.



(a) Case with 2 components.

(b) Case with 3 components.

(c) Case with 5 components.

(d) Case with 10 components.

Figure 6.8: Evolution of the objective function value along the iterations for the algorithms EGO-FSSF-lo and MADS plugged on VME.

Now, we comment on the running times of the algorithms. We notice that EGO is much slower than MADS. An analysis of the runs of MADS shows that $88 \pm 1\%$ of the computation time is spent in the evaluations of the objective function with VME, and this proportion is the same for all test cases. The number of calls to VME is of the same order of magnitude in EGO and MADS, meaning that for EGO most of the computation time is spent in the inner working of the algorithm. The calls to VME when EGO is used indeed represent only 11% of the computation time for the case with

2 components and less than 0.2% with 10 components. A lot of computational resources is used for the estimation of the hyperparameters, the inversion of the covariance matrix and for the maximization of the EI in EGO. These tasks become hard quickly when the dimension grows, hence EGO will not be able to tackle the industrial maintenance optimization problem for large-scale systems with up to 80 components.

The algorithm MADS converges after $100n$ evaluations, where n is the number of components (Figure 6.8) and the running time stays low, even for the case with 10 components (Table 6.4). However, when running MADS from different initial points[2], we observe that it does not converge to the same maintenance strategy: it is stuck too early in a local optimum and is not exploratory enough, confirming the behavior we have seen on the multimodal functions of the COCO benchmark in §6.2.4.2. In large dimension, MADS may be able to converge in reasonable time to a local optimum but the quality of the solution will be questionable.[3]

To conclude on the small cases considered in this section, MADS returns maintenance strategies that perform similarly or better than those returned by EGO but within a much lower running time. It is clear that EGO will not be able to scale up to larger systems. The drawback of MADS in larger dimension is its insufficient exploratory behavior, which may lead to solutions of low-quality. Hence, a simple application of these blackbox methods will not be appropriate to tackle large-scale instances of the industrial maintenance optimization problem.

Remark 6.2. In the numerical experiments, we have used $Q = 100$ as it allows for fast runs of the optimization algorithms. However, in an operational context, maintenance strategies are often compared using 10^4 or 10^5 scenarios so that the confidence interval on the expected LCC is small enough to allow for robust decision making. This is why, for a real small industrial case, say with less than 10 components, we advise to use MADS with $Q = 10^4$ or $Q = 10^5$ so that the execution can be done in no more than few hours and with an objective function that is an accurate estimation of the LCC. ◊

6.4 Conclusion, limits of the blackbox methods and perspectives

In this chapter, we have assessed the performance of two blackbox algorithms, EGO and MADS, on a comprehensive benchmark and on small industrial test cases. Overall, MADS is performing better than EGO. On the COCO benchmark, only the highly multimodal functions with global structure are more favorable for EGO. On the industrial cases, MADS does always as good or better than EGO and is much faster. From the user perspective, MADS requires less tuning and can be used more easily. Moreover, as stated by [Conn and Le Digabel, 2013], if no structure of the objective function is present or known, direct search methods may be preferred over model-based methods. All these points argue in favor of MADS over EGO if a blackbox algorithm is to be chosen for a small industrial maintenance scheduling problem.

[2] The results of Table 6.4 have all been obtained with u^0 being the center of the admissible set $U^{\text{ad}} = [0, T]^n$ *i.e.* $u^0 = (T/2, \ldots, T/2)$, the results of the runs with other initial point are not presented in the manuscript.

[3] This general remark is confirmed with the numerical results of Part II, where we argue that MADS cannot sufficiently explore the whole space in Section 10.8.

However, both EGO and MADS are limited for an application to large-scale problems. In EGO, some steps like the estimation of the hyperparameters or the maximization of the EI are computationally demanding. In MADS, the exploration of the search space may not be sufficient to provide a solution of satisfying quality in large dimension. Several ideas can be explored to improve the efficiency of the proposed blackbox methods.

- We have highlighted that the performance of MADS is dependent of the search step, which is currently done using a quadratic approximation of the objective function. We can try a hybrid algorithm that uses EGO within MADS: EGO would be used in the search step of MADS to take advantage of its exploratory behavior, whereas the poll step of MADS ensures a good local exploitation.[4] We note that [Munoz Zuniga and Sinoquet, 2020] combines EGO and MADS in the other way around: MADS is used as the EI solver within EGO. We can also use the variable neighborhood search [Audet et al., 2008a] in MADS, that is designed to escape local minima, to improve the exploratory behavior of the algorithm.

- The other path for improvement is parallelization. Parallel versions of MADS [Audet et al., 2008b] and EGO [Ginsbourger et al., 2008] already exist but have not been tested. Even though more evaluations are possible with parallel algorithms, the complexity of the problem grows exponentially with the dimension which may prevent from using these methods for large problems.

In this part, we have only considered periodic maintenance strategies in order to reduce the size of the search space, so that the problem can be solved by a blackbox approach. If we are to consider more general policies with larger systems, we would not be able to run EGO in reasonable time. We may still be able to run MADS but the direct approach with the blackbox method may be outperformed by more subtle algorithms. In the next part, we use a different approach, based on decomposition methods, to alleviate the curse of dimensionality.

[4] This idea is implemented in the NOMAD software through the SGTELIB library.

Part II

Decomposition by prediction for optimal maintenance scheduling

From blackbox optimization to stochastic optimal control

The blackbox methods presented in Part I are limited when it comes to tackle large-scale optimization problems. In this part, we use a different approach: we give an explicit modeling of the industrial problem as a stochastic optimal control problem and we design a decomposition method that takes advantage of the structure of the problem.

Optimal control is concerned with the study of problems that involve dynamical systems. When decision making is faced with uncertainties, either in the cost function, or the constraints of a system, a problem is said to be *stochastic* and when we have both a dynamics and uncertainties, we enter the realm of *stochastic optimal control*.

When it comes to optimization, and that we need to choose a cost function, several attitudes are possible to deal with uncertainties. For instance, we can try to optimize the worst case situation, or find the best decision in expectation. Hence, there is no generic formulation of a stochastic optimization problem, unlike for deterministic linear programming or convex optimization for example. We classify stochastic optimal control problems given their *information structure*:

- In the *open-loop* framework, the decision is deterministic, it is only computed once and for all, knowing the *a priori* probability distributions of the uncertainties.

- In the *closed-loop* framework, the decision may depend on past realizations of the uncertainties, which amounts to say that the decision can be based on online observations of the system.

For the interested reader, a more comprehensive typology of information structures is given in [Carpentier et al., 2015]. The distinction between open-loop and closed-loop is essential as it determines the methods that can be used for the problem resolution. Open-loop problems can be tackled by stochastic gradient algorithms [Robbins and Monro, 1951], whereas closed-loop problems often use stochastic dynamic programming [Bertsekas and Shreve, 1996].

In the industrial problem described in Section 2.1, we are constrained to use open-loop controls. The controls correspond to PMs and as we consider large critical components of the electricity production process, the maintenance operations require a shut down of the unit and the mobilization of important human resources, which must be planned several years in advance. Hence, we cannot afford to adapt the PM strategy to online observations of the system. Therefore, we stick to the open-loop framework in this part.

We are concerned with problems of the following form:

$$\min_{u \in U^{\text{ad}}} J(u) \quad \text{where} \quad J(u) = \mathbb{E}\left(j(u, \boldsymbol{W})\right) , \tag{6.4}$$

where $\boldsymbol{W}$ is a random variable, defined on a probability space $(\Omega, \mathcal{A}, \mathbb{P})$, that represents the uncertainties in the system. The function j is a real-valued cost function, assumed to be differentiable with respect to u, and $U^{\text{ad}} \subset \mathbb{U}$ is the space of admissible controls, with $\mathbb{U}$ being a Hilbert space. The main difficulties to tackle Problem (6.4) are the evaluations of J and ∇J, which involve the computation of an expectation. Two approaches have been developed to get around this difficulty, namely Stochastic Approximation (SA) and Sample Average Approximation (SAA), both based on Monte-Carlo sampling techniques. A comparison between SA and SAA methods can be found in [Nemirovski et al., 2009].

1. In a SA method, each iteration uses a different sample of the random variable $\boldsymbol{W}$. For instance, the l-th iteration of the stochastic gradient method, written in terms of random variables, is given by:

$$\boldsymbol{U}^{l+1} = \text{proj}_{U^{\text{ad}}}\left(\boldsymbol{U}^l - \varepsilon^l \nabla_u j(\boldsymbol{U}^l, \boldsymbol{W}^{l+1})\right) ,$$

   where $\{\boldsymbol{U}^l\}_{l \in \mathbb{N}}$ is the sequence of random iterates of the algorithm, $\varepsilon^l > 0$ is the step size and $\{\boldsymbol{W}^l\}_{l \in \mathbb{N}}$ are independent random variables, identically distributed as $\boldsymbol{W}$. The projection operator must be understood "ω by ω". A numerical execution of the algorithm, with iterates $\{u^l\}_{l \in \mathbb{N}}$, corresponds to a realization of the random sequence $\{\boldsymbol{U}^l\}_{l \in \mathbb{N}}$, that is:

$$\exists \omega \in \Omega, \ \forall l \in \mathbb{N}, \ u^l = \boldsymbol{U}^l(\omega) .$$

2. The SAA approach consists in approximating Problem (6.4) by the sample average problem:

$$\min_{u \in U^{\text{ad}}} \frac{1}{Q} \sum_{q=1}^{Q} j(u, w_q) , \tag{6.5}$$

   where $Q \in \mathbb{N}$ and $(w_1, \ldots, w_Q)$ is a realization of a sample of $\boldsymbol{W}$ of size Q. The SAA problem (6.5) can then be solved by any appropriate deterministic algorithm.

In Section 2.1, we consider industrial systems with up to 80 components. To solve such large instances of the maintenance optimization problem, we must resort to decomposition methods. Decomposition-coordination algorithms consist in splitting an optimization problem into a sequence of smaller subproblems that are iteratively solved and coordinated. Many decomposition schemes exist and they can formulated within a unified framework called the Auxiliary Problem Principle (APP) [Cohen, 1980]. The APP is a general principle that consists in turning the resolution of a *master* optimization problem into the resolution of a sequence of auxiliary problems whose solutions converge to the master's one. The APP can be formulated both for a use in the SA or the SAA context.

SA and SAA have the same efficiency estimates with respect to the sample size [Nemirovski et al., 2009] but each approach requires the tuning of some parameters. We think that an operational user of our maintenance optimization algorithm will

more easily understand the stakes in tuning the sample size Q in the SAA approach than the step size ε^l in a SA scheme, which is more technical due to the requirements on the decrease rate of the sequence $\{\varepsilon^l\}_{l\in\mathbb{N}}$. This is why, we focus on the SAA approach to solve the industrial problem of Section 2.1.

Part II of the thesis is organized as follows. Chapter 7 is a bibliographical review of the APP, with a focus on the particular case of the so-called *decomposition by prediction*. In Chapter 8, we model the industrial maintenance problem as a stochastic optimal control problem. Then, in Chapters 9 and 10, we apply the APP respectively on synthetic test cases and on the industrial system of Chapter 8. The APP allows to design a decomposition by prediction of these large-scale systems. Numerical experiments are carried out to show the efficiency of the decomposition method. In Appendix A, we recall some basic definitions in convex analysis that are used in this part.

Communication. The work presented in Chapters 7, 8 and 10 constitutes an extended version of the preprint [Bittar et al., 2020].

Notations. We introduce some notations that are used throughout this part and which are related to the industrial system described in Section 2.1. We denote by $n \in \mathbb{N}$ the number of physical components of the system and by $N = n+1$ the number of *entities*, gathering the physical components and an added stock. We denote by:

- $\mathbb{I} = \{1, \dots, N\}$ the entity index set,
- $I = \{1, \dots, n\}$ the component index set.

Note that $\mathbb{I} = I \bigcup \{N\}$. Let $T \in \mathbb{N}$ be a time horizon. We denote by:

- $\mathbb{T} = \{0, \dots, T\}$ the time index set,
- $\mathbb{T}_{-1} = \{0, \dots, T-1\}$.

For any $v = \{v_{i,t}\}_{(i,t)\in\mathbb{I}\times\mathbb{T}}$, we use the following notations:

- $v_i = \{v_{i,0}, \dots, v_{i,T}\}$ denotes the value of v for the entity $i \in \mathbb{I}$ over all time steps.
- $v_{i:j} = \{v_i, \dots, v_j\}$ denotes the value of v for the entities $i, i+1, \dots, j \in \mathbb{I}$ over all time steps, with the convention that $v_{i:j}$ is empty if $i > j$.
- $v_{\cdot,t} = \{v_{1,t}, \dots, v_{n,t}\}$ denotes the value of v over all entities at time $t \in \mathbb{T}$.
- $v_{\cdot,a:b} = \{v_{\cdot,a}, \dots, v_{\cdot,b}\}$ denotes the value of v over all entities at time steps $a, a+1, \dots, b \in \mathbb{T}$.

7 The Auxiliary Problem Principle and its use in decomposition

Any intelligent fool can make things bigger, more complex, and more violent. It takes a touch of genius – and a lot of courage – to move in the opposite direction.

Ernst Friedrich Schumacher

Contents

7.1 Introduction

The industrial system of interest, presented in Section 2.1, presents some characteristics that makes it a *large system* with a *complex spatial structure* according to the definition of [Carpentier and Cohen, 2017, Chapter 1]:

- The system is described by a large number of variables or constraints, leading to a high-dimensional optimization problem, that requires large computational resources. We face the well-known *curse of dimensionality*.
- The global system is composed of several interconnected subsystems: different components share a common stock of spare parts.

The interest in mathematical optimization for large systems appears in the 1960s, in particular with the work of [Arrow and Hurwicz, 1960, Lasdon and Schoeffler, 1965, Mesarović et al., 1970, Takahara, 1964], that introduces decomposition-coordination methods. The idea of decomposition is to formulate subproblems involving only smaller subsystems of the original large system. Each subproblem is easier to solve than the global optimization problem and provides its *local* solution. Then, the goal of coordination is to ensure that gathering the local solutions leads to a global solution. Decomposition-coordination methods usually result in an iterative process alternating an optimization step on the subsystems and a coordination step that updates the subproblems. Different types of decomposition-coordination schemes have been designed, by prices, by quantities or by prediction. They have been unified within the Auxiliary Problem Principle (APP) [Cohen, 1978].

As we use a SAA approach, we only focus on the deterministic APP in this chapter. Still, we mention that the APP can be adapted to the stochastic case:

- In the open-loop stochastic case, the APP has been combined with stochastic gradient descent [Culioli and Cohen, 1990].
- The closed-loop stochastic case is more challenging because the coordination variables are stochastic processes (whereas they are deterministic in the open-loop case), which complicates the resolution of the local subproblems. The Dual Approximate Dynamic Programming algorithm [Barty et al., 2010] has been designed to handle this situation.

The main advantage of decomposition is that solving all the small subproblems is easier than solving the original problem. More than that, the computational complexity of an optimization problem is often superlinear or even exponential in the size of the problem. Hence, the sum of the computational efforts required for the resolution of all subproblems will be lower than for the global problem, even if the resolution of the subproblems must be carried out multiple times. Another feature of decomposition methods is that they are naturally adapted to parallelization as each subproblem is independent. This leads to a reduction of computation time.

Contributions. This chapter consists in a bibliographical review of the general framework of the APP (Section 7.2) and its specialization to the decomposition by prediction (Section 7.3). The decomposition scheme is implemented through a fixed-point algorithm for which we recall a theoretical convergence result. This review is essential as the APP is the tool that allows for the design of decomposition schemes in the practical cases presented in Chapters 9 and 10.

7.2 Presentation of the APP

Based on [Carpentier and Cohen, 2017], we present the main ideas of the APP. Consider the following problem, which we call the master problem:

$$\min_{u \in U^{\mathrm{ad}}} J^{\Sigma}(u) + J^{\Delta}(u) \;\text{ such that }\; \Theta(u) \in -C \;, \tag{7.1}$$

where:

- U^{ad} is a non-empty, closed, convex subset of a Hilbert space $\mathbb{U}$,
- C is a pointed closed convex cone of a Hilbert space $\mathcal{C}$.
- $J^{\Sigma} : \mathbb{U} \to \mathbb{R}$ and $J^{\Delta} : \mathbb{U} \to \mathbb{R}$ are convex and lower semi-continuous (l.s.c.). The function $J^{\Sigma} + J^{\Delta}$ is coercive on U^{ad}. Moreover, J^{Δ} is differentiable[1].
- $\Theta : \mathbb{U} \to \mathcal{C}$ is continuous, differentiable and C-convex, where C-convexity means that:

$$\forall u, v \in \mathbb{U},\ \forall \rho \in [0,1],\ \rho\Theta(u) + (1-\rho)\Theta(v) - \Theta(\rho u + (1-\rho)v) \in C \;.$$

Under constraint qualification conditions, solving Problem (7.1) is equivalent to finding a saddle point of the Lagrangian:

$$\begin{aligned} L : \mathbb{U} \times \mathcal{C} \to &\ \mathbb{R} \\ (u, \lambda) \mapsto &\ L(u,\lambda) = J^{\Sigma}(u) + J^{\Delta}(u) + \langle \lambda \,, \Theta(u) \rangle \;, \end{aligned}$$

on $U^{\mathrm{ad}} \times C^{\star}$, where $C^{\star}$ is the dual cone of C:

$$C^{\star} = \{ \lambda \in \mathcal{C},\ \langle \lambda \,, \mu \rangle \geq 0 \text{ for all } \mu \in C \} \;.$$

We write:

$$L = L^{\Sigma} + L^{\Delta} \;,$$

with $L^{\Sigma}(u,\lambda) = J^{\Sigma}(u)$ and $L^{\Delta}(u,\lambda) = J^{\Delta}(u) + \langle \lambda \,, \Theta(u) \rangle$. From the assumptions on J^{Σ}, J^{Δ} and Θ, we get that L^{Σ} and L^{Δ} are convex in u, concave in λ and L^{Δ} is differentiable. It is well-known that there is an equivalence between $(u^{\sharp}, \lambda^{\sharp}) \in U^{\mathrm{ad}} \times C^{\star}$ being a saddle point of L and the following pair of variational inequalities:

$$\begin{aligned} &\forall u \in U^{\mathrm{ad}},\ \left\langle \nabla_u L^{\Delta}(u^{\sharp}, \lambda^{\sharp}) \,, u - u^{\sharp} \right\rangle + L^{\Sigma}(u, \lambda^{\sharp}) - L^{\Sigma}(u^{\sharp}, \lambda^{\sharp}) \geq 0 \;, \\ &\forall \lambda \in C^{\star},\ \left\langle \nabla_{\lambda} L^{\Delta}(u^{\sharp}, \lambda^{\sharp}) \,, \lambda - \lambda^{\sharp} \right\rangle \leq 0 \;, \end{aligned} \tag{7.2}$$

Definition 7.1. *(Auxiliary problem)* Let $(\bar{u}, \bar{\lambda}) \in \mathbb{U} \times \mathcal{C}$ and $\varepsilon > 0$. Let $Q : \mathbb{U} \times \mathcal{C} \to \mathbb{R}$ be an auxiliary function that is differentiable, strongly convex in its first argument and concave in its second argument. Define:

$$\begin{aligned} G_{\bar{u},\bar{\lambda}}(u,\lambda) = Q(u,\lambda) &+ \left\langle \varepsilon \nabla_u L^{\Delta}(\bar{u},\bar{\lambda}) - \nabla_u Q(\bar{u},\bar{\lambda}) \,, u \right\rangle \\ &+ \left\langle \varepsilon \nabla_{\lambda} L^{\Delta}(\bar{u},\bar{\lambda}) - \nabla_{\lambda} Q(\bar{u},\bar{\lambda}) \,, \lambda \right\rangle + \varepsilon L^{\Sigma}(u,\lambda) \;. \end{aligned} \tag{7.3}$$

[1] In this chapter, when a function is said to be differentiable, we mean that it is Gateaux-differentiable, see Definition A.4.

Assume that $G_{\bar{u},\bar{\lambda}}$ admits a saddle point. Then, the problem of finding a saddle point of $G_{\bar{u},\bar{\lambda}}$, that writes:

$$\max_{\lambda\in C^\star}\min_{u\in U^{\mathrm{ad}}} G_{\bar{u},\bar{\lambda}}(u,\lambda) = \min_{u\in U^{\mathrm{ad}}}\max_{\lambda\in C^\star} G_{\bar{u},\bar{\lambda}}(u,\lambda) \tag{7.4}$$

is an *auxiliary problem* for (7.1).

The following statement is the fundamental lemma for the theory of the APP.

Lemma 7.2. *Let $(u^\sharp,\lambda^\sharp)$ be a solution of the auxiliary problem (7.4). If $(u^\sharp,\lambda^\sharp) = (\bar{u},\bar{\lambda})$, then $(u^\sharp,\lambda^\sharp)$ is a saddle point of the Lagrangian L of the master problem (7.1).*

Proof. The fact that $(u^\sharp,\lambda^\sharp)$ is solution of (7.4), implies that $(u^\sharp,\lambda^\sharp)$ satisfies the following variational inequalities:

$$\begin{aligned}
\forall u\in U^{\mathrm{ad}},\ &\left\langle \nabla_u Q(u^\sharp,\lambda^\sharp) + \varepsilon\nabla_u L^\Delta(\bar{u},\bar{\lambda}) - \nabla_u Q(\bar{u},\bar{\lambda})\,,\, u-u^\sharp\right\rangle \\
&\qquad + \varepsilon\left(L^\Sigma(u,\lambda^\sharp) - L^\Sigma(u^\sharp,\lambda^\sharp)\right)\geq 0\,,\\
\forall\lambda\in C^\star,\ &\left\langle \nabla_\lambda Q(u^\sharp,\lambda^\sharp) + \varepsilon\nabla_\lambda L^\Delta(\bar{u},\bar{\lambda}) - \nabla_\lambda Q(\bar{u},\bar{\lambda})\,,\, \lambda-\lambda^\sharp\right\rangle \leq 0\,.
\end{aligned}$$

Then, using $(u^\sharp,\lambda^\sharp) = (\bar{u},\bar{\lambda})$, we see that $(u^\sharp,\lambda^\sharp)$ satisfies the variational inequalities (7.2). This is equivalent to $(u^\sharp,\lambda^\sharp)$ being a saddle point of L. □

Lemma 7.2 suggests to use the fixed-point algorithm 6 to find a saddle point of L and hence solve the master problem (7.1). There is a great flexibility in the construction of the auxiliary problem. Some particular choices allow to retrieve well-known algorithms such as Uzawa or Arrow-Hurwicz algorithms, see [Carpentier and Cohen, 2017, §3.3]. In the following section, we design the auxiliary problem so that the APP fixed-point algorithm results in a scheme of decomposition by prediction.

Algorithm 6 APP fixed-point algorithm

1: Let $(\bar{u},\bar{\lambda}) = (u^0,\lambda^0)\in U^{\mathrm{ad}}\times C^\star$ and set $l=0$.
2: At iteration $l+1$:
- Solve the auxiliary problem (7.4), *i.e.*, find a saddle point (u^{l+1},λ^{l+1}) of $G_{\bar{u},\bar{\lambda}}$.
- Set $(\bar{u},\bar{\lambda}) = (u^{l+1},\lambda^{l+1})$.

3: If the maximum number of iterations is reached or $\left\|u^{l+1}-u^l\right\| + \left\|\lambda^{l+1}-\lambda^l\right\|$ is *sufficiently small* then stop, else $l\leftarrow l+1$ and go back to step 2.

7.3 The decomposition by prediction as a specific instance of the APP

In this section, still based on [Carpentier and Cohen, 2017], we show that the APP allows to retrieve the *decomposition by prediction* introduced in [Mesarović et al., 1970]. This decomposition scheme relies on a decomposition of both the primal space $\mathbb{U}$ and the dual space $\mathcal{C}$.

- We assume that $\mathbb{U} = \mathbb{U}_1 \times \ldots \times \mathbb{U}_N$, with $N \in \mathbb{N}$, and that $U^{\text{ad}} = U_1^{\text{ad}} \times \cdots \times U_N^{\text{ad}}$, where for all $i \in \mathbb{I} = \{1, \ldots, N\}$, $U_i^{\text{ad}} \subset \mathbb{U}_i$ is a closed convex set. The decomposition of the admissible set $U^{\text{ad}} = U_1^{\text{ad}} \times \cdots \times U_N^{\text{ad}}$ defines the subset on which each subproblem is solved.
- We assume that $\mathcal{C} = \mathcal{C}_1 \times \cdots \times \mathcal{C}_N$ and that $C = C_1 \times \cdots \times C_N$, where for all $i \in \mathbb{I}$, $C_i \subset \mathcal{C}_i$ is a closed convex cone. The decomposition of the cone $C = C_1 \times \cdots \times C_N$ specifies which part of the constraint is assigned to each subproblem.

We also assume that:

- J^Σ is additive with respect to the decomposition of the admissible space, meaning that for $u = (u_1, \ldots, u_N) \in U^{\text{ad}}$ with $u_i \in U_i^{\text{ad}}$ for all $i \in \mathbb{I}$, we have:
$$J^\Sigma(u) = \sum_{i=1}^{N} J_i^\Sigma(u_i) \ ,$$
where $J_i^\Sigma : \mathbb{U}_i \to \mathbb{R}$.
- $\Theta(u) = (\Theta_1(u), \ldots, \Theta_N(u))$ with $\Theta_i : \mathbb{U} \to \mathcal{C}_i$ for $i \in \mathbb{I}$.

Let $\bar{u} = (\bar{u}_1, \ldots, \bar{u}_N) \in \mathbb{U}$ and $\bar{\lambda} = (\bar{\lambda}_1, \ldots, \bar{\lambda}_N) \in C^\star$. We choose an auxiliary function of the form[2]:
$$Q(u, \lambda) = K(u) + \langle \lambda \, , \Phi(u) \rangle \, ,$$
where K is an auxiliary cost and Φ is an auxiliary constraint satisfying the following properties:

- $K : \mathbb{U} \to \mathbb{R}$ is convex, l.s.c., differentiable and additive: $K(u) = \sum_{i=1}^{N} K_i(u_i)$;
- $\Phi : \mathbb{U} \to \mathcal{C}$ is differentiable and block-diagonal: $\Phi(u) = \Big(\Phi_1(u_1), \ldots, \Phi_N(u_N)\Big)$.

In the decomposition by prediction, we interpret the saddle point problem (7.4) as a constrained optimization problem. Taking $\varepsilon = 1$ in (7.3), the auxiliary problem can be written as:
$$\begin{aligned} \min_{u \in U^{\text{ad}}} \ & K(u) + J^\Sigma(u) + \Big\langle \nabla J^\Delta(\bar{u}) - \nabla K(\bar{u}) \, , u \Big\rangle + \Big\langle \bar{\lambda} \, , \Big(\Theta'(\bar{u}) - \Phi'(\bar{u})\Big) \cdot u \Big\rangle \\ \text{s.t. } & \Phi(u) - \Phi(\bar{u}) + \Theta(\bar{u}) \in -C \ . \end{aligned} \tag{7.5}$$

Choosing K additive and Φ block-diagonal ensures that Problem (7.5) decomposes in N independent subproblems that can be solved in parallel. For $i \in \mathbb{I}$, the i-th subproblem is given by:
$$\begin{aligned} \min_{u_i \in U_i^{\text{ad}}} & K_i(u_i) + J_i^\Sigma(u_i) + \Big\langle \nabla_{u_i} J^\Delta(\bar{u}) - \nabla K_i(\bar{u}_i) \, , u_i \Big\rangle \\ & - \Big\langle \bar{\lambda}_i \, , \Phi_i'(\bar{u}_i) \cdot u_i \Big\rangle + \sum_{j=1}^{N} \Big\langle \bar{\lambda}_j \, , \partial_{u_i} \Theta_j(\bar{u}) \cdot u_i \Big\rangle \\ \text{s.t. } & \Phi_i(u_i) - \Phi_i(\bar{u}_i) + \Theta_i(\bar{u}) \in -C_i \ . \end{aligned} \tag{7.6}$$

This subproblem only depends on $u_i \in U_i^{\text{ad}}$ and inherits only the i-th component of the constraint.

[2] Other choices for the auxiliary function Q allow to retrieve different decomposition schemes. For instance, using $Q(u, \lambda) = K(u) - \frac{1}{2\alpha} \|\lambda\|^2$, with $\alpha > 0$ leads to price decomposition, see [Carpentier and Cohen, 2017, §4.2].

Example 7.3. Let $\bar{u} \in \mathbb{U}$. A canonical choice for the additive auxiliary cost function K is:

$$K(u) = \sum_{i=1}^{N} K_i(u_i) \quad \text{with} \quad K_i(u_i) = J^{\Delta}(\bar{u}_{1:i-1}, u_i, \bar{u}_{i+1:n}) \ ,$$

where $u_{i:j} = (u_i, \dots, u_j)$ for $i \leq j$ and the convention that $u_{i:j}$ is empty if $j > i$. Similarly, a canonical choice for the block-diagonal auxiliary constraint Φ is:

$$\Phi(u) = (\Phi_1(u_1), \dots, \Phi_N(u_N)) \quad \text{with} \quad \Phi_i(u_i) = \Theta_i(\bar{u}_{1:i-1}, u_i, \bar{u}_{i+1:n}) \ .$$

The general idea is to construct the i-th term of the auxiliary function from the original function where only the i-th component is allowed to vary. △

We can now write the APP fixed-point algorithm for the decomposition by prediction. We set the maximum of iterations to $M \in \mathbb{N}$.

Algorithm 7 APP fixed-point algorithm for decomposition by prediction

1: Start with $(\bar{u}, \bar{\lambda}) = (u^0, \lambda^0)$ and set $l = 0$.
2: **for all** $i \in \{1, \dots, N\}$ **do in parallel:**
3: | Solve the subproblem i defined by (7.6). Let $(u_i^{l+1}, \lambda_i^{l+1})$ be a solution.
4: **end for**
5: Set $\bar{u} = (u_1^{l+1}, \dots, u_N^{l+1})$ and $\bar{\lambda} = (\lambda_1^{l+1}, \dots, \lambda_N^{l+1})$.
6: If $\|u^{l+1} - u^l\| + \|\lambda^{l+1} - \lambda^l\|$ is *sufficiently small*, then stop, else $l \leftarrow l + 1$ and go back to step 2.
7: **return** $(\bar{u}, \bar{\lambda})$.

A convergence result for Algorithm 7 is given in [Cohen, 1980].

Theorem 7.4. *[Cohen, 1980, Theorem 5.1 with $\epsilon = 1$] Assume that:*

- *the admissible space U^{ad} is equal to the whole space $\mathbb{U}$,*
- *the constraints are equality constraints, that is $C = \{0\}$,*
- *$K(u) = \frac{1}{2}\langle u \, , \mathsf{K}u \rangle$, $J^{\Sigma}(u) = 0$, $J^{\Delta}(u) = \frac{1}{2}\langle u \, , \mathsf{J}u \rangle + \langle \mathsf{j} \, , u \rangle$ where K and J are linear self-adjoint strongly monotone and Lipschitz continuous operators, j is a vector in $\mathbb{U}$, and $2\mathsf{K} - \mathsf{J}$ is assumed to be strongly monotone,*
- *$\Phi(u) = \mathsf{O}u$, $\Theta(u) = \mathsf{T}u + \mathsf{t}$ where the operators O and T are linear and surjective and t is a vector in $\mathcal{C}$,*
- *(Geometric condition) the operator $2\left(\mathsf{T}\mathsf{J}^{-1}\mathsf{O}^{\top} + \mathsf{O}\mathsf{J}^{-1}\mathsf{T}^{\top}\right) - \mathsf{T}\mathsf{J}^{-1}(2\mathsf{K} + \mathsf{J})\mathsf{J}^{-1}\mathsf{T}^{\top}$ is strongly monotone.*

Then, the sequence $\left\{(u^l, \lambda^l)\right\}_{l \in \mathbb{N}}$ generated by the APP fixed-point algorithm for the decomposition by prediction converges strongly to the unique optimal solution $(u^\sharp, \lambda^\sharp)$ of the original problem (7.1).

The convergence theorem 7.4 for Algorithm 7 holds only in the restrictive case where the costs J^{Δ} and K are quadratic and the constraints Θ and Φ are linear. The difficulty for the convergence of the decomposition by prediction comes from the fact that the auxiliary function Q is not strongly concave in λ, it is linear in λ. This choice of Q was made to be able to interpret the auxiliary problem as a constrained minimization problem. The geometric condition of Theorem 7.4 is the assumption that allows to overcome the difficulty arising from this choice of Q.

7.4 Conclusion

In this part, we have recalled the theoretical foundations of the APP which allows to turn the resolution of an optimization problem into the iterative resolution of auxiliary problems. In order to use the APP for decomposition purpose, we construct a decomposable auxiliary problem. Hence, solving the auxiliary problem boils down to the resolution of independent subproblems of smaller size, which can be done in parallel. We have presented the particular case of the decomposition by prediction along with a convergence result that holds when the cost function is quadratic and the constraints are linear. The APP allows us to overcome the curse of dimensionality when it comes to solve large-scale optimization problems, such as the industrial maintenance scheduling problem, that we formulate in Chapter 8. The APP is applied on synthetic test cases in Chapter 9 and on the industrial system in Chapter 10.

8 Modeling of the industrial maintenance optimization problem

Life is like riding a bicycle. To keep your balance you must keep moving.

Albert Einstein

Contents

8.1 Introduction

This thesis is driven by the industrial problem, presented in Section 2.1, of maintenance optimization for a system of components from a single hydroelectric power plant. We recall that the components share a common stock of spare parts. The goal is to find a deterministic maintenance strategy that minimizes the expected life cycle cost (LCC). We consider systems with up to 80 components, which fall into the category of large-scale systems.

In Part I, the optimal maintenance scheduling problem is tackled with blackbox optimization algorithms. However, these methods are limited when the number of components is large. This is why, in Chapter 7, we have introduced the APP, a general framework that can be used to produce decomposition-coordination schemes, such as the decomposition by prediction, described in Section 7.3.

The ultimate goal of Part II is to apply the APP on the industrial case. To do so, some modeling effort is required to formulate the maintenance optimization problem.[1] We model the dynamics and the cost generated by the system and consider a more general space of maintenance strategies than in Part I, where we have only considered periodic maintenance strategies. In the model of this chapter, we will be able to decide whether or not to do a Preventive Maintenance (PM) each year for each component.

Contributions. In this chapter, we develop a model for the industrial system described in Section 2.1. In Section 8.2, we introduce variables to characterize the components, the stock, the PM strategy and the random failures of the components. Then, in Sections 8.3 and 8.4, we give an analytical expression of the dynamics and of the cost generated by the system. Finally, in Section 8.5, we formulate the maintenance optimization problem. This chapter is key to exit from the blackbox context, currently used at EDF, which prevents from tackling large-scale optimization problems. The modeling we propose consists in *opening the blackbox* and is a necessary step towards the implementation of the decomposition method presented in Chapter 7.

8.2 Description of the system

The system under study has been physically described in Section 2.1. In this section, we start the mathematical modeling of the system by introducing some variables to characterize the different elements that make it up: the stock and the physical components. Then, we define the space of admissible maintenance strategies and we present a model for the random failures of the components. The modeling of the system is necessary to provide a mathematical formulation of the maintenance optimization problem. Before the description of the system, we give some definitions.

Definition 8.1. Let $p \in \mathbb{N}$, $\mathcal{A} \subset \mathbb{R}^p$ and $x \in \mathbb{R}^p$. The indicator function of the set $\mathcal{A}$ is:

$$\mathbf{1}_{\mathcal{A}}(x) = \begin{cases} 1 \text{ if } x \in \mathcal{A}\,, \\ 0 \text{ if } x \notin \mathcal{A}\,. \end{cases}$$

We introduce the following notations for the space of random variables.

[1] This modeling effort was not needed in Part I as the problem was solved in a blackbox context: only the function evaluations are used to guide the search.

Definition 8.2. Let $(\Omega, \mathcal{A}, \mathbb{P})$ be a probability space and $(\mathbb{Y}, \mathcal{Y})$ be a measurable space. The set of measurable functions from $(\Omega, \mathcal{A}, \mathbb{P})$ to $(\mathbb{Y}, \mathcal{Y})$ is denoted by $\mathcal{Y} = L^0(\Omega, \mathcal{A}, \mathbb{P}; \mathbb{Y})$, where we omit the σ-algebra $\mathcal{Y}$ in the notation.

Remark 8.3. Any random variable $\boldsymbol{Y} : \Omega \to \mathbb{Y}$ is an element of $\mathcal{Y}$. ◊

We consider a system of $n \in \mathbb{N}^\star$ physical components of a single hydropower plant (generators, turbines or transformers) sharing a common stock of spare parts. A sketch of the system with $n = 2$ components is represented in Figure 8.1. A Corrective Maintenance (CM) consists in the replacement of a component after a failure. A Preventive Maintenance (PM) is a planned replacement of a healthy component (before a failure).

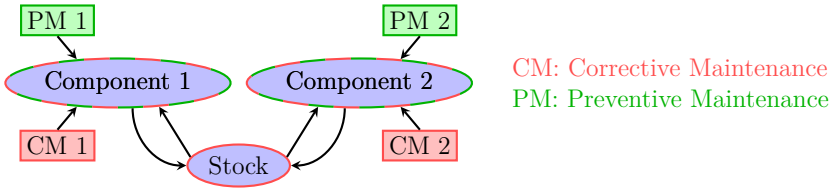


Figure 8.1: System of two components sharing the same stock of spare parts.

We use the notations defined in the introductory chapter of Part II. In the sequel, $i \in I = \{1, \dots, n\}$ is a component index, $t \in \mathbb{T} = \{0, \dots, T\}$ is a time step. In this chapter, we use the component index set I rather than the entity index set $\mathbb{I}$ as the variables describing the stock will be treated separately from the components.

8.2.1 Characterization of the stock and the components

The stock over time is characterized by the sequence of random variables

$$\boldsymbol{S} = (\boldsymbol{S}_0, \dots, \boldsymbol{S}_T) \in \mathcal{S} \,,$$

defined on a given probability space $(\Omega, \mathcal{A}, \mathbb{P})$ and where $\boldsymbol{S}_t$ is the random variable representing the number of available spare parts at time t. We have $\mathcal{S} = L^0(\Omega, \mathcal{A}, \mathbb{P}; \mathbb{S})$ the set of all random variables taking values in $\mathbb{S} = \{0, \dots, s\}^{T+1}$. The parameter $s \in \mathbb{N}$ is the maximum number of spare parts. The value of the initial stock is set to $\boldsymbol{S}_0 = s$. The replenishment delay for the parts, that is, the time from order to delivery of a part, is known and denoted by $D \in \mathbb{N}$.

At time t, component i is characterized by random variables representing:

- its regime

$$\boldsymbol{E}_{i,t} = \begin{cases} 0 \text{ if the component is broken,} \\ 1 \text{ if the component is healthy.} \end{cases}$$

  A component has only two regimes: in the healthy regime, it runs in its nominal operating point. In the broken regime, it stops working completely. Initially, all components are healthy *i.e.* $\boldsymbol{E}_{i,0} = 1$ for all $i \in I$.

- its age (if healthy) or the time for which it has failed (if broken) denoted by the real-valued random variable $\boldsymbol{A}_{i,t}$. Initially, the components are new *i.e.* $\boldsymbol{A}_{i,0} = 0$ for all $i \in I$.

- the time elapsed since its last D failures:

$$\boldsymbol{P}_{i,t} = (\boldsymbol{P}_{i,t}^1, \dots, \boldsymbol{P}_{i,t}^D) \ ,$$

  where D is the number of time steps for the supply of spare parts. For $d \in \{1, \dots, D\}$, $\boldsymbol{P}_{i,t}^d$ is the number of time steps elapsed since the d-th undiscarded failure of component i. $\boldsymbol{P}_{i,t}^d$ takes a default value δ if the component has failed fewer than d times. Hence, $\boldsymbol{P}_{i,t}^d$ takes values in $\{\delta\} \cup \mathbb{R}_+$ and $\boldsymbol{P}_{i,0} = (\delta, \dots, \delta)$. The random vector $\boldsymbol{P}_{i,t}$ is useful to compute the dates of replenishment of the stock. It is enough to store at most the dates of the last D failures to describe the supply of the stock. More details are given in §8.3.1.

The characteristics of component i at time t are gathered in:

$$\boldsymbol{X}_{i,t} = (\boldsymbol{E}_{i,t}, \boldsymbol{A}_{i,t}, \boldsymbol{P}_{i,t}) \in \mathcal{X}_{i,t} \ ,$$

where $\mathcal{X}_{i,t} = L^0(\Omega, \mathcal{A}, \mathbb{P}; \mathbb{X}_{i,t})$ with $\mathbb{X}_{i,t} = \{0,1\} \times \mathbb{R}_+ \times (\{\delta\} \cup \mathbb{R}_+)^D$. The state of the system is then described at t by $(\boldsymbol{X}_{1,t}, \dots, \boldsymbol{X}_{n,t}, \boldsymbol{S}_t)$. Finally, to describe the components over the whole study period we introduce:

$$\boldsymbol{X} = (\boldsymbol{X}_1, \dots, \boldsymbol{X}_n) = ((\boldsymbol{X}_{1,0}, \dots, \boldsymbol{X}_{1,T}), \dots, (\boldsymbol{X}_{n,0}, \dots, \boldsymbol{X}_{n,T})) \in \mathcal{X} \ ,$$

where $\mathcal{X} = \prod_{i=1}^n \mathcal{X}_i$ and $\mathcal{X}_i = \prod_{t=0}^T \mathcal{X}_{i,t}$, for all $i \in I$. In order to emphasize that $\boldsymbol{X}$ depends on all the components of the system, we sometimes use the notation $\boldsymbol{X}_{1:n}$ instead of $\boldsymbol{X}$.

8.2.2 Preventive maintenance strategy

A PM consists in repairing a component although it is in the healthy regime. The dates of PM can be different for each component. They define the preventive maintenance strategy of the system. Operational constraints impose to look for deterministic strategies. This means that the dates of PM are chosen without any knowledge on the state of the system after the beginning of the time horizon and cannot be changed during the study. The maintenance strategy is defined by a vector:

$$u = (u_1, \dots, u_n) = ((u_{1,0}, \dots, u_{1,T-1}), \dots, (u_{n,0}, \dots, u_{n,T-1})) \in \mathbb{U} = [0,1]^{nT} \ , \quad (8.1)$$

where $u_{i,t}$ characterizes the PM for component i at time t. More precisely, we set a threshold $0 < \nu < 1$: a control $u_{i,t} \geq \nu$ corresponds to a rejuvenation of the component proportional to $u_{i,t}$ and a value $u_{i,t} < \nu$ corresponds to not performing a maintenance. We consider that the maintenance operation is instantaneous and that it does not use parts from the stock. The reason is that PMs are planned in advance, hence it is possible to order the parts so that they arrive just in time for the maintenance operation.

Note that a PM decision could *a priori* be represented by a discrete variable taking values in $\{0,1\}$. As we intend to apply the APP which is a continuous optimization algorithm, we choose to model the PM strategy with continuous decision variables $u_{i,t} \in [0,1]$, $(i,t) \in I \times \mathbb{T}_{-1}$. The use of a continuous u and of the threshold ν will become clearer in §8.3.1.

The modeling of §8.1 allows to consider very general maintenance strategies as for each component, we can decide whether or not to do a PM at each time step. The space $\mathbb{U}$ is then much larger than the space of periodic maintenance strategies considered in Part I.

8.2.3 Failures of the components

In our study, the distribution of the time to failure for component i is a known Weibull distribution[2] of parameters (β_i, λ_i) denoted by $\text{Weib}(\beta_i, \lambda_i)$. The probability of failure of a component at a given time step only depends on its age and its failure distribution.

Proposition 8.4. *Let $i \in I$. Assume that component i has age $a \geq 0$ at time t. Let F_i be the cumulative distribution function of the time to failure for component i. Then, the probability of failure of component i at time $t + \Delta t$ conditionally to the component being healthy at t is given by:*

$$p_i(a) = \frac{F_i(a + \Delta t) - F_i(a)}{1 - F_i(a)} ,$$

The proof of this result can be found in [Meeker and Escobar, 2014, Section 2.2.1]. We introduce the random sequence:

$$\boldsymbol{W} = (\boldsymbol{W}_1, \dots, \boldsymbol{W}_n) = ((\boldsymbol{W}_{1,1}, \dots, \boldsymbol{W}_{1,T}), \dots, (\boldsymbol{W}_{n,1}, \dots, \boldsymbol{W}_{n,T})) \in \mathcal{W} ,$$

where $\mathcal{W} = L^0\left(\Omega, \mathcal{A}, \mathbb{P}; [0,1]^{nT}\right)$. The random process $\boldsymbol{W}$ is an exogenous noise that affects the dynamics of the regime $\boldsymbol{E}$ and the age $\boldsymbol{A}$. We assume that all $\boldsymbol{W}_{i,t}$ are independent random variables and follow a uniform distribution on $[0, 1]$. At time step t, component i has age $\boldsymbol{A}_{i,t}$. If $\boldsymbol{W}_{i,t+1} < p_i(\boldsymbol{A}_{i,t})$, component i fails at $t + 1$, otherwise no failure occurs.

8.3 Dynamics of the system

Now, we describe the dynamics of the system, that is, we explain how the variables characterizing the system, presented in Section 8.2, evolve between two time steps.

8.3.1 Dynamics of a component

Let $i \in I$ and $t \in \mathbb{T}_{-1}$. The dynamics of component i between t and $t + 1$ is described as follows.

1. If component i is healthy *i.e.* $\boldsymbol{E}_{i,t} = 1$:

   (a) If $u_{i,t} \geq \nu$, then a PM is performed. After a PM, component i stays healthy and is rejuvenated so that:

   $$(\boldsymbol{E}_{i,t+1}, \boldsymbol{A}_{i,t+1}) = (1, (1 - u_{i,t})(\boldsymbol{A}_{i,t} + 1)) .$$

   Note that $u_{i,t} = 1$ makes the component as good as new: in this case we have $\boldsymbol{A}_{i,t+1} = 0$.

   (b) If $u_{i,t} < \nu$, then no PM is performed. Component i fails with probability $p_i(\boldsymbol{A}_{i,t})$:

   $$(\boldsymbol{E}_{i,t+1}, \boldsymbol{A}_{i,t+1}) = \begin{cases} (0, 0) & \text{if } \boldsymbol{W}_{i,t+1} < p_i(\boldsymbol{A}_{i,t}) , \\ (1, \boldsymbol{A}_{i,t} + 1) & \text{otherwise} . \end{cases}$$

[2] The definition of the Weibull distribution is given in the Appendix, see Definition B.8.

2. If component i is broken *i.e.* $\boldsymbol{E}_{i,t} = 0$:

   (a) If a spare is available in the stock, a CM is performed to replace the component. We assume that the CM is an identical replacement, which implies that the component becomes as good as new. We get:

   $$(\boldsymbol{E}_{i,t+1}, \boldsymbol{A}_{i,t+1}) = (1, 0) \ .$$

   (b) If no spare part is available, the defective component stays in the broken regime:

   $$(\boldsymbol{E}_{i,t+1}, \boldsymbol{A}_{i,t+1}) = (0, \boldsymbol{A}_{i,t} + 1) \ .$$

   As all components belong to the same power plant, when at least one component is broken, the unit is shut down until the arrival of a spare part and the execution of the CM. Such a situation is a *forced outage*. During the shut down no electricity is produced.

We have to express formally that a spare part is available for the replacement of component i. At time t, suppose that the stock has $\boldsymbol{S}_t = r$ parts and that m components are broken. If $r \geq m$ then all components can be replaced immediately. When $r < m$, we must choose which components to replace. Our modeling choice is to replace the broken components following the order of their index: if $i_1 \leq \ldots \leq i_r \leq \ldots \leq i_m$ are the indices of the broken components, we replace only the components with index $i_1, \ldots, i_r$, the others stay in the broken regime and wait for new available parts. Using this choice, the availability of a spare part for component i corresponds to the condition:

$$\boldsymbol{S}_t \geq \sum_{j=1}^{i} \mathbf{1}_{\{0\}}(\boldsymbol{E}_{j,t}) \ . \tag{8.2}$$

The right hand side of (8.2) simply counts the number of broken components with index smaller or equal than i.

To completely describe the dynamics of a component, we have to specify the dynamics of the vector $\boldsymbol{P}_{i,t}$. It has been introduced in §8.2.1 to store the dates of failures of the component and compute the dates for the replenishment of the stock.

- If $\boldsymbol{P}_{i,t} = (t_1, \ldots, t_d, \delta, \ldots, \delta)$ with $t_1, \ldots, t_d \geq 0$, meaning that component i has undergone $d < D$ failures so far, then:

$$\boldsymbol{P}_{i,t+1} = \begin{cases} (t_1 + 1, \ldots, t_d + 1, 0, \delta, \ldots, \delta) & \text{if failure at } t+1 \ , \quad (8.3a) \\ (t_1 + 1, \ldots, t_d + 1, \delta, \delta, \ldots, \delta) & \text{otherwise .} \quad (8.3b) \end{cases}$$

- If $\boldsymbol{P}_{i,t} = (t_1, \ldots, t_D)$ with $t_1, \ldots, t_D \geq 0$, meaning that component i has undergone at least D failures so far, then:

$$\boldsymbol{P}_{i,t+1} = \begin{cases} (t_2 + 1, \ldots, t_D + 1, 0) & \text{if failure at } t+1 \ , \quad (8.4a) \\ (t_1 + 1, \ldots, t_D + 1) & \text{otherwise .} \quad (8.4b) \end{cases}$$

  In (8.4a), note that t_1 is discarded. As $\boldsymbol{P}_{i,t} = (t_1, \ldots, t_D)$ and $t_1 > \ldots > t_D \geq 0$, we get that $t_1 \geq D - 1$. At time step $t+1$, the part ordered from the failure at t_1

has arrived. Then, storing t_1 is not useful anymore. So if a failure occurs at $t+1$, we can discard t_1 to make room for the new date of failure. This proves that it is enough to have $\boldsymbol{P}_{i,t}$ of size D to compute the replenishment of the stock as stated in §8.2.1. Note that the dates are not discarded if there is no failure (see (8.4b)), so it is possible to have $t_d > D$ for some $d \in \{1, \dots, D\}$. Such variables have no influence on the dynamics of the system.

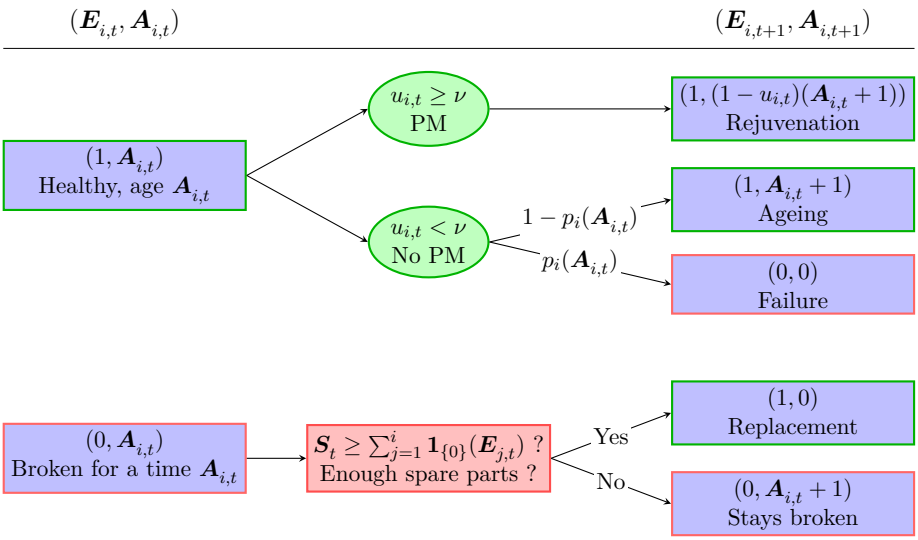


Figure 8.2: Dynamics of component i.

Figure 8.2 summarizes the dynamics of component i from t to $t+1$. Recall that we have $\boldsymbol{X}_{i,t} = (\boldsymbol{E}_{i,t}, \boldsymbol{A}_{i,t}, \boldsymbol{P}_{i,t})$. We write the *dynamics of component i* on the whole time horizon as:

$$\Theta_i(\boldsymbol{X}_{1:i}, \boldsymbol{S}, u_i, \boldsymbol{W}_i) = 0 \ ,$$

where $\Theta_i = \{\Theta_{i,t}\}_{t \in \mathbb{T}}$ such that:

$$\begin{cases} \Theta_{i,t+1}(\boldsymbol{X}_{1:i}, \boldsymbol{S}, u_i, \boldsymbol{W}_i) = \boldsymbol{X}_{i,t+1} - f_i(\boldsymbol{X}_{1:i,t}, \boldsymbol{S}_t, u_{i,t}, \boldsymbol{W}_{i,t+1}), \quad t \in \mathbb{T}_{-1} \ , \\ \Theta_{i,0}(\boldsymbol{X}_{1:i}, \boldsymbol{S}, u_i, \boldsymbol{W}_i) = \boldsymbol{X}_{i,0} - x_i \ , \end{cases} \tag{8.5}$$

with $x_i = (1, 0, \delta, \dots, \delta)^\top$ and f_i represents the dynamics we just described for component i. An explicit expression of f_i is given in Appendix D.1.

Note that there is a coupling between the dynamics of component i and the stock. There is also a coupling with components $j < i$. This is due to the choice (8.2) of replacing the broken components with the smallest indices first if there are not enough spare parts.

8.3.2 Dynamics of the stock

For the stock, the initial number of spare parts is $\boldsymbol{S}_0 = s$. As PMs can be anticipated, we consider that the needed spares are ordered so that they arrive just in time for the scheduled maintenance. Therefore, they do not appear in the dynamics of the stock. A part is used for each CM and a new part is ordered only after the failure of a component. The number of time steps for the supply of a part is D. Hence, the part ordered after the d-th undiscarded failure of component i arrives in the stock at $t+1$

if $\boldsymbol{P}^d_{i,t+1} = D$. This is equivalent to $\boldsymbol{P}^d_{i,t} = D - 1$. On the other hand, the number of broken components is $\sum_{i=1}^n \mathbf{1}_{\{0\}}(\boldsymbol{E}_{i,t})$ and we replace as many of them as possible given the current level of stock $\boldsymbol{S}_t$. Thus, we have:

$$\boldsymbol{S}_{t+1} = \boldsymbol{S}_t + \sum_{i=1}^{n}\sum_{d=1}^{D} \mathbf{1}_{\{D-1\}}(\boldsymbol{P}^d_{i,t}) - \min\left\{\boldsymbol{S}_t,\ \sum_{i=1}^{n} \mathbf{1}_{\{0\}}(\boldsymbol{E}_{i,t})\right\}, \quad t \in \mathbb{T}_{-1}\ . \tag{8.6}$$

We write the *dynamics of the stock* in compact form as:

$$\Theta_{\boldsymbol{S}}(\boldsymbol{X}_{1:n}, \boldsymbol{S}) = 0\ ,$$

where $\Theta_{\boldsymbol{S}} = \{\Theta_{\boldsymbol{S},t}\}_{t\in\mathbb{T}}$ such that:

$$\begin{cases} \Theta_{\boldsymbol{S},t+1}(\boldsymbol{X}_{1:n}, \boldsymbol{S}) = \boldsymbol{S}_{t+1} - f_{\boldsymbol{S}}(\boldsymbol{X}_{1:n,t}, \boldsymbol{S}_t), \quad t \in \mathbb{T}_{-1}\ , \\ \Theta_{\boldsymbol{S},0}(\boldsymbol{X}_{1:n}, \boldsymbol{S}) = \boldsymbol{S}_0 - s\ , \end{cases} \tag{8.7}$$

with f_S corresponding to the right-hand side of (8.6). Note that $\boldsymbol{S}_{t+1}$ depends on the current level of stock $\boldsymbol{S}_t$ but also on $\boldsymbol{X}_{i,t}$ for all $i \in I$. The stock is coupling all the components of the system.

Finally, the *dynamics of the whole system* is summarized by the almost sure equality constraint $\Theta(\boldsymbol{X}, \boldsymbol{S}, u, \boldsymbol{W}) = 0$, where we have $\Theta : \mathcal{X} \times \mathcal{S} \times \mathbb{U} \times \mathcal{W} \to \mathcal{L}$, with $\Theta = \left\{\{\Theta_i\}_{i\in I}, \Theta_{\boldsymbol{S}}\right\}$ and $\mathcal{L} = (\prod_{i=1}^n \mathcal{L}_i) \times \mathcal{L}_{\boldsymbol{S}}$ where $\mathcal{L}_i = L^0\left(\Omega, \mathcal{A}, \mathbb{P}; \mathbb{R}^{(D+2)(T+1)}\right)$ for $i \in I$ and $\mathcal{L}_{\boldsymbol{S}} = L^0\left(\Omega, \mathcal{A}, \mathbb{P}; \mathbb{R}^{(T+1)}\right)$.

We have now completely described the dynamics of the system. In the next part we specify the costs associated to the system.

8.4 Costs generated by the system

The costs generated by the system are due to PMs, CMs and forced outages of the unit. In practice, as PMs are scheduled in advance, they are cheaper than unpredictable CMs. A forced outage of the unit induces a loss of production. It is characterized by a yearly cost which is higher than that of a PM or a CM. We consider a discount rate τ meaning that a cost c occurring at time t will be valued $\eta_t c$ with the discount factor $\eta_t := \frac{1}{(1+\tau)^t}$. We introduce the following notations:

- $j^P_{i,t}(u_{i,t})$ is the PM cost incurred at time $t \in \mathbb{T}_{-1}$ for component $i \in I$. Let C^P_i be the cost of a PM operation on component i. We set:

  $$j^P_{i,t}(u_{i,t}) = \eta_t C^P_i u^2_{i,t}\ .$$

  We use a quadratic cost as it is strongly convex and should favor numerical convergence. In particular, in the case when $0 < u_{i,t} < \nu$, which models a situation where no PM is performed, we have $j^P_{i,t}(u_{i,t}) > 0$.[3]

[3] The fact that $j^P_{i,t}(u_{i,t}) > 0$ when $0 < u_{i,t} < \nu$ is favorable from a numerical point of view. For $0 < u_{i,t} < \nu$, we always have $j^P_{i,t}(u_{i,t}) > j^P_{i,t}(0)$ while no PM is performed. Hence, the system dynamics is the same with $u_{i,t} = 0$ and $0 < u_{i,t} < \nu$ while the cost is higher for $0 < u_{i,t} < \nu$. Hence, the control $u_{i,t} = 0$ is always better to $0 < u_{i,t} < \nu$. This feature will allow us to clearly distinguish the steps where a PM is performed from the others.

- $j^C_{i,t}(\boldsymbol{X}_{i,t})$ is the CM cost. It is due at the time of the failure of a component, even if there is no spare part to perform the operation immediately. Hence it only occurs when $(\boldsymbol{E}_{i,t}, \boldsymbol{A}_{i,t}) = (0, 0)$. Let C^C_i be the cost of a CM operation on component i. We have:

$$j^C_{i,t}(\boldsymbol{X}_{i,t}) = \eta_t C^C_i \mathbf{1}_{\{0\}}(\boldsymbol{E}_{i,t}) \mathbf{1}_{\{0\}}(\boldsymbol{A}_{i,t}) \ .$$

- $j^F_t(\boldsymbol{X}_{1:n,t})$ is the forced outage cost. As all components belong to the same production unit, a forced outage occurs when at least one component is in a failed state and the CM has not been performed immediately because of a lack of spare part. Let C^F be the forced outage cost per time unit. We have:

$$j^F_t(\boldsymbol{X}_{1:n,t}) = \eta_t C^F \min \left\{ 1, \ \sum_{i=1}^{n} \mathbf{1}_{\{0\}}(\boldsymbol{E}_{i,t}) \mathbf{1}_{\mathbb{R}^*_+}(\boldsymbol{A}_{i,t}) \right\} \ .$$

In order to consider the previous costs over the whole study period we introduce:

- the *total maintenance cost* (preventive and corrective) generated by component $i \in I$ on the studied period:

$$j_i(\boldsymbol{X}_i, u_i) = \sum_{t=0}^{T-1} j^P_{i,t}(u_{i,t}) + \sum_{t=0}^{T} j^C_{i,t}(\boldsymbol{X}_{i,t}) \ , \tag{8.8}$$

- the *total forced outage cost* generated by the system during the studied period:

$$j^F(\boldsymbol{X}_{1:n}) = \sum_{t=0}^{T} j^F_t(\boldsymbol{X}_{1:n,t}) \ , \tag{8.9}$$

8.5 Formulation of the maintenance optimization problem

The dynamics of the system is stochastic as it depends on the failure of the components, modeled by the random vector $\boldsymbol{W}$. The cost function is then stochastic as well. The objective is to find the deterministic maintenance strategy $u \in \mathbb{U}$ that minimizes the expected cost generated by the system over all failure scenarios. Hence, the industrial optimal maintenance scheduling problem is formulated as follows:

$$\begin{aligned} \min_{(\boldsymbol{X}, \boldsymbol{S}, u) \in \mathcal{X} \times \mathcal{S} \times \mathbb{U}} & \ \mathbb{E}\Big(\sum_{i=1}^{n} j_i(\boldsymbol{X}_i, u_i) + j^F(\boldsymbol{X}_{1:n}) \Big) \\ \text{s.t. } & \Theta(\boldsymbol{X}, \boldsymbol{S}, u, \boldsymbol{W}) = 0 \ , \quad \mathbb{P}\text{-a.s.} \ . \end{aligned} \tag{8.10}$$

The goal of Part II of the thesis is to solve Problem (8.10) with the scheme of decomposition by prediction introduced in Chapter 7. In this way, we should be able to solve the maintenance optimization problem for systems with a large number of components and in particular, to tackle the largest industrial case of interest with 80 components.

However, the description of the industrial system involves several integer variables – such as the regime $\boldsymbol{E}_{i,t}$ or the stock $\boldsymbol{S}_t$ – which seems incompatible with the application of the APP, a method based on variational techniques. The auxiliary problem (7.5)

indeed involves the derivatives of the dynamics Θ which are not defined given that integer variables appear in the expressions (8.5) and (8.7). Moreover, the convergence result for the decomposition by prediction, given in Theorem 7.4, applies only when the cost function is quadratic and the dynamics is linear. However, the forced outage cost j^F is not quadratic, nor even convex and the dynamics Θ is non linear. In the light of these remarks, the application of the APP to the industrial problem is not straightforward and we have no guarantee that a decomposition by prediction of Problem (8.10) will lead to a solution that achieves good performance.

8.6 Conclusion

In this chapter, we have developed a mathematical model for an industrial system of several components from a single hydropower plant sharing a common stock of spare parts. The system is described by a mix of integer and continuous variables and we give analytical expressions for the dynamics and cost of the system. In the model, we allow for very general maintenance strategies where one maintenance decision can be taken at each time step for each component. Therefore, we are not restricted to periodic maintenance strategies. The resulting optimization problem is a mixed-integer program for which the application of the APP is not straightforward and without any guarantee of performance. This is why, in Chapter 9, we choose to apply the APP first on two synthetic test cases, that share some common characteristics with the industrial problem (8.10) but that are closer to the theoretical scope of convergence of the decomposition method. The performance of the decomposition by prediction on these test cases will give a first indicator of whether it is relevant to use the APP for the industrial problem. In Chapter 10, we show how to implement the APP for Problem (8.10) and carry out some numerical experiments.

9 Application of the APP on two synthetic test cases

Theory is when you know everything but nothing works. Practice is when everything works but no one knows why. Here, theory and practice are combined: nothing works and no one knows why.

Unknown

Contents

9.1 Introduction

In Chapter 8, we have formulated the industrial maintenance optimization problem. The goal of Part II of the thesis is to solve Problem (8.10) with the APP in order to optimize the maintenance for large-scale industrial systems. However, we have highlighted in Section 8.5 that the application of APP to the industrial problem is not straightforward.

In this chapter, we design two synthetic test cases for which we apply the decomposition by prediction presented in Chapter 7. The test cases are designed so as to display the following characteristics.

1. They fit the structure of the industrial problem. By this, we mean that the couplings that appear in the cost function and in the dynamics of the synthetic system are chosen to be similar to those of the industrial system.

2. They have an analytical solution.

3. They have a linear dynamics and quadratic costs as required in Theorem 7.4, but do not satisfy the geometric condition that is hard to check in practice.

We implement the decomposition by prediction for the test cases and compare the numerical results given by the fixed-point algorithm with the analytical optimum. This is a preliminary validation step of the decomposition methodology before moving on to the application of the APP for the industrial case in Chapter 10. We consider the two following test cases.

1. In Section 9.2, we design a system with a quadratic cost function and a deterministic linear dynamics for which we can directly use the decomposition by prediction introduced in Section 7.3.

2. In Section 9.3, we design a system with a stochastic dynamics for which we aim at minimizing the expectation of a quadratic cost function. We use a SAA approach – as described in the introduction of Part II – to solve this problem and apply the decomposition by prediction on a deterministic approximate problem.

In this chapter, we reuse the vocabulary and notations presented in the introductory chapter of Part II.

Contributions. We present a progressive approach where the APP is applied on two test cases of increasing difficulty. We tailor these synthetic cases to be counterparts of the industrial problem but for which the application of the APP is easier. From a practical point of view, our contributions include the design of the synthetic problems, the result and the proof of the formula for the analytical solution of the problems, the choice of the auxiliary problem for the application of the APP and the numerical implementation of the fixed-point algorithm.

9.2 The deterministic synthetic test case

In this section, we apply the APP on a test case that evolves with a deterministic linear dynamics and that generates a deterministic quadratic cost. We start by describing the dynamics and cost of the synthetic system and we formulate the associated optimization

problem. Then, we give the analytical optimal solution of the problem. Next, in order to apply the APP, we give an explicit choice for the auxiliary function that leads to a scheme of decomposition by prediction. Finally, we apply the APP fixed-point algorithm on the test case and compare the numerical solution with the analytical one.

9.2.1 Description of the deterministic test case

We consider a system that consists in $N = 3$ entities, say two physical components and a stock, to do the analogy with the industrial case. The system is studied on a discrete time horizon $T \in \mathbb{N}$. In the sequel, $i \in \mathbb{I} = \{1, 2, 3\}$ denotes an entity, $I = \{1, 2\}$ is the components index set and $t \in \mathbb{T} = \{0, \dots, T\}$ denotes a time step. The entities of the system are characterized by a vector:

$$x = (x_1, x_2, x_3) = ((x_{1,0}, \dots, x_{1,T}), (x_{2,0}, \dots, x_{2,T}), (x_{3,0}, \dots, x_{3,T})) \in \mathbb{X} = \mathbb{R}^{3(T+1)} ,$$

where for $i \in \mathbb{I}$, we have $x_i \in \mathbb{X}_i = \mathbb{R}^{(T+1)}$ and $\mathbb{X} = \prod_{i \in \mathbb{I}} \mathbb{X}_i$. On the system, we can apply a control:

$$u = (u_1, u_2) = ((u_{1,0}, \dots, u_{1,T-1}), (u_{2,0}, \dots, u_{2,T-1})) \in \mathbb{U} = \mathbb{R}^{2T} , \tag{9.1}$$

where for $i \in I$, we have $u_i \in \mathbb{U}_i = \mathbb{R}^T$ and $\mathbb{U} = \prod_{i \in I} \mathbb{U}_i$. In the industrial case, controls represent the PMs and can only be applied on the components but not on the stock, justifying the size of the vector u.

Dynamics. The system evolves with a linear deterministic dynamics written as:

$$\Theta(x, u) = 0 ,$$

where $\Theta = \{\Theta_i\}_{i \in \mathbb{I}}$ with Θ_i representing the dynamics of entity i. We have $\Theta_i = \{\Theta_{i,t}\}_{t \in \mathbb{T}}$ with:

$$\begin{cases} \Theta_{i,t+1}(x, u) = x_{i,t+1} - A_i x_{\cdot,t} - B_i u_{\cdot,t} , & t \in \mathbb{T}_{-1} , \\ \Theta_{i,0}(x, u) = x_{i,0} - x_{i,\text{init}} , \end{cases}$$

where $x_{\cdot,t} = (x_{1,t}, x_{2,t}, x_{3,t})$, $u_{\cdot,t} = (u_{1,t}, u_{2,t})$. We also have $x_{i,\text{init}} \in \mathbb{R}$, $A_i \in \mathbb{R}^3$ and $B_i \in \mathbb{R}^2$. In the sequel, we denote by $A \in \mathbb{R}^{3\times3}$ (resp. $B \in \mathbb{R}^{3\times2}$) the matrix with rows $\{A_i\}_{i \in \mathbb{I}}$ (resp. $\{B_i\}_{i \in \mathbb{I}}$).

Cost. The cost generated by the system at time t is quadratic and given by:

$$\begin{cases} j_t(x_{\cdot,t}, u_{\cdot,t}) = \langle x_{\cdot,t} , Rx_{\cdot,t} \rangle + \langle u_{\cdot,t} , Ou_{\cdot,t} \rangle , & t \in \mathbb{T}_{-1} , \\ j_T(x_{\cdot,T}) = \langle x_{\cdot,T} , Rx_{\cdot,T} \rangle , \end{cases}$$

where $R \in \mathbb{R}^{3\times3}$ and $O \in \mathbb{R}^{2\times2}$ are symmetric matrices with R being positive semi-definite and O being positive definite. These requirements ensure that the cost function is positive and convex. The overall cost is then:

$$j(x, u) = \sum_{t=0}^{T-1} j_t(x_{\cdot,t}, u_{\cdot,t}) + j_T(x_{\cdot,T}) . \tag{9.2}$$

Problem of interest. We aim at solving the following optimization problem:

$$\begin{aligned} \min_{(x,u)\in\mathbb{X}\times\mathbb{U}} \; & j(x,u) \\ \text{s.t. } & \Theta(x,u) = 0 \; . \end{aligned} \tag{9.3}$$

One interest of studying the test case (9.3) is that an analytical solution can be computed thanks to the following theorem.

Theorem 9.1. *The optimal control $u^\sharp$ for Problem (9.3) is given by:*

$$u^\sharp = C^{-1} d \; .$$

Here $u^\sharp$ is to be understood as the vector $(u^\sharp_{1,0}, u^\sharp_{2,0}, \dots, u^\sharp_{1,T-1}, u^\sharp_{2,T-1})^\top \in \mathbb{R}^{2T}$ where the coordinates are sorted by time step *whereas in (9.1) the coordinates are sorted by* component*. The matrix $C \in \mathbb{R}^{2T\times 2T}$ is defined by blocks of size 2×2 such that:*

$$\begin{cases} C_{t,t} = O + \displaystyle\sum_{p=t}^{T-1} (A^{p-t}B)^\top R (A^{p-t}B) \in \mathbb{R}^{2\times 2}, \quad t \in \mathbb{T}_{-1} \; , \\ C_{t,k} = \displaystyle\sum_{p=\max(k,t)}^{T-1} (A^{p-t}B)^\top R (A^{p-k}B) \in \mathbb{R}^{2\times 2}, \quad t,k \in \mathbb{T}_{-1},\ t \neq k \; , \end{cases}$$

and $d \in \mathbb{R}^{2T}$ is defined by blocks of size 2×1 such that:

$$d_t = -\sum_{p=t}^{T-1} (A^{p-t}B)^\top R A^{p+1} x_{\,\cdot,\mathrm{init}} \in \mathbb{R}^2, \quad t \in \mathbb{T}_{-1} \; .$$

Proof. In Problem (9.3), when the constraint $\Theta(x,u) = 0$ is satisfied, we can express x as a function of u and substitute its value in $j(x,u)$. The cost function of Problem (9.3) can then be expressed as a function $\tilde{j} : \mathbb{U} \to \mathbb{R}$ that depends only on u. By induction, we get:

$$x_{\,\cdot,t+1} = A^{t+1} x_{\,\cdot,0} + \sum_{k=0}^{t} A^{t-k} B u_{\,\cdot,k}, \quad t \in \mathbb{T}_{-1} \; .$$

Therefore:

$$\begin{aligned} \tilde{j}(u) = \sum_{t=0}^{T} \left\langle A^t x_{\,\cdot,\mathrm{init}} + \sum_{k=0}^{t-1} A^{t-1-k} B u_{\,\cdot,k} \, , R \left(A^t x_{\,\cdot,0} + \sum_{k=0}^{t-1} A^{t-1-k} B u_{\,\cdot,k} \right) \right\rangle \\ + \sum_{t=0}^{T-1} \langle u_{\,\cdot,t} \, , O u_{\,\cdot,t} \rangle \; . \end{aligned} \tag{9.4}$$

The cost function $\tilde{j}$ is strictly convex. Moreover, the feasible set is convex and non empty, hence $\tilde{j}$ admits a unique minimizer $u^\sharp$. The optimal value $x^\sharp$ is then determined by the constraint $\Theta(x^\sharp, u^\sharp) = 0$. To compute $u^\sharp$, we simply use that:

$$\nabla_{u_{\,\cdot,t}} \tilde{j}(u^\sharp) = 0, \quad t \in \mathbb{T}_{-1} \; .$$

This leads to:

$$\begin{aligned} \left(O + \sum_{p=t+1}^{T} (A^{p-t-1}B)^\top R A^{p-t-1} B \right) u^\sharp_{\,\cdot,t} + \sum_{p=t+1}^{T} (A^{p-t-1}B)^\top R \sum_{\substack{k=0 \\ k\neq t}}^{p-1} A^{p-k-1} B u^\sharp_{\,\cdot,k} \\ + \sum_{p=t+1}^{T} (A^{p-t-1}B)^\top R A^p x_{\,\cdot,\mathrm{init}} = 0 \; . \end{aligned}$$

Exchanging the two summation signs in the second term and using the change of indices $p \leftarrow p+1$, we get:

$$\left(O + \sum_{p=t}^{T-1} (A^{p-t}B)^\top R A^{p-t} B\right) u^\sharp_{\cdot,t} + \sum_{\substack{k=0 \\ k\neq t}}^{T-1} \sum_{p=\max(k,t)}^{T-1} (A^{p-t}B)^\top R A^{p-k} B u^\sharp_{\cdot,k} + \sum_{p=t}^{T-1} (A^{p-t}B)^\top R A^{p+1} x_{\cdot,\text{init}} = 0 \ ,$$

which corresponds exactly to:

$$C_{t,t} u^\sharp_{\cdot,t} + \sum_{\substack{k=0 \\ k\neq t}}^{T-1} C_{t,k} u^\sharp_{\cdot,k} = d_t \ . \tag{9.5}$$

Equality (9.5) is valid for $t \in \mathbb{T}_{-1}$, so that we can write:

$$u^\sharp = C^{-1} d \ .$$

This concludes the proof. □

9.2.2 Application of the APP for a decomposition by prediction

In this section, we apply the APP as described in Section 7.3. We aim at decomposing Problem (9.3) in three subproblems involving respectively only the spaces $(\mathbb{X}_1 \times \mathbb{U}_1)$ (first subproblem), $(\mathbb{X}_2 \times \mathbb{U}_2)$ (second subproblem) and $\mathbb{X}_3$ (third subproblem). In practice, this means that we consider the following decomposition of the space $\mathbb{X} \times \mathbb{U}$:

$$\mathbb{X} \times \mathbb{U} = (\mathbb{X}_1 \times \mathbb{U}_1) \times (\mathbb{X}_2 \times \mathbb{U}_2) \times \mathbb{X}_3 \ . \tag{9.6}$$

The constraints of Problem (9.3) are represented by the function $\Theta : \mathbb{X} \times \mathbb{U} \to \mathbb{X}$. Our goal is to assign the constraint on the dynamics of entity i to the subproblem involving this same entity. This means that we consider the following decomposition of the space of constraints $\mathbb{X}$:

$$\mathbb{X} = \mathbb{X}_1 \times \mathbb{X}_2 \times \mathbb{X}_3 \ . \tag{9.7}$$

Then, we can identify the additive part and the non-additive coupling part in the cost j with respect to the decomposition (9.6) by writing:

$$j = j^\Sigma + j^\Delta \ ,$$

with

$$j^\Sigma(x,u) = \sum_{i=1}^{2} j_i^\Sigma(x_i,u_i) + j_3^\Sigma(x_3) \quad \text{where} \quad \begin{cases} j_i^\Sigma(x_i,u_i) = \sum_{t=0}^{T} R_{i,i} x_{i,t}^2 + \sum_{t=0}^{T-1} O_{i,i} u_{i,t}^2 \ , \\ j_3^\Sigma(x_3) = \sum_{t=0}^{T} R_{3,3} x_{3,t}^2 \ , \end{cases}$$

$$j^\Delta(x,u) = \sum_{t=0}^{T} \left\langle x_{\cdot,t} \, , \bar{R} x_{\cdot,t} \right\rangle + \sum_{t=0}^{T-1} \left\langle u_{\cdot,t} \, , \bar{O} u_{\cdot,t} \right\rangle \ , \tag{9.8}$$

where we write $R = R_{\text{diag}} + \bar{R}$ and $O = O_{\text{diag}} + \bar{O}$ with R_{diag} and O_{diag} being respectively the matrices R and O where only the diagonal elements are kept, the others being set to zero.

Remark 9.2. The additivity of the cost with respect to the decomposition (9.6) corresponds to the additivity with respect to the entity index i. We notice that both j^{Σ} and j^{Δ} are additive with respect to the time t but this property plays no role in the decomposition scheme entity by entity we are trying to set up. ◊

Now, we introduce an auxiliary cost function K that is additive with respect to the decomposition (9.6) of the primal space and a auxiliary dynamics Φ that is block-diagonal with respect to the decompositions (9.6) and (9.7) of the primal and the dual spaces. We use the canonical technique from Example 7.3. Let $\bar{x} \in \mathbb{X}$ and $\bar{u} \in \mathbb{U}$, we consider:

- An auxiliary cost function $K : \mathbb{X} \times \mathbb{U} \to \mathbb{R}$ such that:

$$K(x,u) = K_1(x_1,u_1) + K_2(x_2,u_2) + K_3(x_3)\ , \tag{9.9}$$

  with

$$\begin{cases} K_1(x_1,u_1) = j^{\Delta}((x_1,\bar{x}_2,\bar{x}_3),(u_1,\bar{u}_2))\ , \\ K_2(x_2,u_2) = j^{\Delta}((\bar{x}_1,x_2,\bar{x}_3),(\bar{u}_1,u_2))\ , \\ K_3(x_3) = j^{\Delta}((\bar{x}_1,\bar{x}_2,x_3),(\bar{u}_1,\bar{u}_2))\ . \end{cases} \tag{9.10}$$

- An auxiliary dynamics $\Phi : \mathbb{X} \times \mathbb{U} \to \mathbb{X}$ such that:

$$\Phi(x,u) = (\Phi_1(x_1,u_1), \Phi_2(x_2,u_2), \Phi_3(x_3))\ , \tag{9.11}$$

  with

$$\begin{cases} \Phi_1(x_1,u_1) = \Theta_1((x_1,\bar{x}_2,\bar{x}_3),(u_1,\bar{u}_2))\ , \\ \Phi_2(x_2,u_2) = \Theta_2((\bar{x}_1,x_2,\bar{x}_3),(\bar{u}_1,u_2))\ , \\ \Phi_3(x_3) = \Theta_3((\bar{x}_1,\bar{x}_2,x_3),(\bar{u}_1,\bar{u}_2))\ . \end{cases}$$

Now that the auxiliary functions are chosen, we can derive the auxiliary problem that is to be solved at each iteration of the APP fixed-point algorithm.

Proposition 9.3. *Let $(\bar{x},\bar{\lambda}) \in \mathbb{X}^2$ and $\bar{u} \in \mathbb{U}$. The auxiliary problem that results from the choices (9.9) and (9.11) of the auxiliary cost and dynamics is given by:*

$$\begin{aligned} \min_{(x,u)\in\mathbb{X}\times\mathbb{U}} \ & K(x,u) + j^{\Sigma}(x,u) + \left\langle \bar{\lambda}\ , (\Theta'(\bar{x},\bar{u}) - \Phi'(\bar{x},\bar{u}))\cdot(x,u) \right\rangle \\ s.t.\ & \Phi(x,u) = 0\ . \end{aligned} \tag{9.12}$$

Proof. The test case corresponds to the theoretical problem (7.1) where we identify j^{Σ} with J^{Σ} and j^{Δ} with J^{Δ}. Moreover, with the choice (9.9) for the auxiliary cost we have:

$$\nabla j^{\Delta}(\bar{x},\bar{u}) - \nabla K(\bar{x},\bar{u}) = 0\ .$$

With the choice (9.11) for the auxiliary dynamics we have:

$$\Theta(\bar{x},\bar{u}) - \Phi(\bar{x},\bar{u}) = 0\ .$$

Thus, the auxiliary problem (9.12) for the synthetic system corresponds exactly to the theoretical auxiliary problem (7.5). □

By construction, the auxiliary problem (9.12) is decomposable into independent subproblems that are given for $i \in I$ by:

$$\begin{aligned} \min_{(x_i,u_i)\in\mathbb{X}_i\times\mathbb{U}_i} \; & K_i(x_i,u_i) + j_i^{\Sigma}(x_i,u_i) + \sum_{\substack{j=1\\ j\neq i}}^{3} \left\langle \bar{\lambda}_j \, , \partial_{(x_i,u_i)}\Theta_j(\bar{x},\bar{u})\cdot(x_i,u_i)\right\rangle \\ \text{s.t. } & \Phi_i(x_i,u_i) = 0 \; . \end{aligned} \tag{9.13}$$

The third subproblem writes:

$$\begin{aligned} \min_{x_3\in\mathbb{X}_3} \; & K_3(x_3) + j_3^{\Sigma}(x_3) + \sum_{j=1}^{2} \left\langle \bar{\lambda}_j \, , \partial_{x_3}\Theta_j(\bar{x},\bar{u})\cdot x_3\right\rangle \\ \text{s.t. } & \Phi_3(x_3) = 0 \; . \end{aligned} \tag{9.14}$$

We are now set to apply Algorithm 7 which reduces the resolution of the synthetic problem (9.3) into the iterative resolution of the subproblems defined by (9.13) and (9.14). The subproblems (9.13) are solved with the blackbox algorithm MADS [Audet and Dennis, 2006] in order to stick to what we aim at doing for the industrial case in Chapter 10. MADS only outputs a primal solution $(x_i^\sharp, u_i^\sharp)$. Then, we need to compute the optimal multiplier $\lambda_i^\sharp = \{\lambda_{i,t}^\sharp\}_{t\in\mathbb{T}}$. To do so, we use the backward recursion that defines the adjoint state.

Lemma 9.4. *Let $i \in \mathbb{I}$. The optimal multipliers of subproblems $i \in I$ (9.13) and subproblem $i = 3$ (9.14) are given by the following backward recursion:*

$$\begin{aligned} \lambda_{i,T}^\sharp &= -2\bar{R}_i\bar{x}_{\cdot,T} - 2R_{i,i}x_{i,T}^\sharp \; , \\ \lambda_{i,t}^\sharp &= -2\bar{R}_i\bar{x}_{\cdot,t} - 2R_{i,i}x_{i,t}^\sharp + \bar{A}_i^\top\bar{\lambda}_{\cdot,t+1} + A_{i,i}\lambda_{i,t+1}^\sharp, \quad t\in\mathbb{T}_{-1} \; , \end{aligned} \tag{9.15}$$

where $\bar{A}_i$ and $\bar{R}_i$ denote the i-th row of $\bar{A}$ and $\bar{R}$ respectively.

Proof. Let L_i be the Lagrangian of subproblem $i \in \mathbb{I}$. As subproblem i is convex with linear constraints, L_i admits a saddle point. Let $(x_i^\sharp, u_i^\sharp, \lambda_i^\sharp) \in \mathbb{X}_i \times \mathbb{U}_i \times \mathbb{X}_i$ be a saddle point of L_i, we have:

$$\nabla L_i(x_i^\sharp, u_i^\sharp, \lambda_i^\sharp) = 0 \; .$$

The backward recursion (9.15) results from the computation of the partial derivatives of L_i with respect to $x_{i,t}$ for $t \in \mathbb{T}$. □

Remark 9.5. Note that the third subproblem (9.14) is easy to solve numerically. The constraint $\Phi_3(x_3) = 0$ indeed completely determines the dynamics of the system in this subproblem. Solving (9.14) then just amounts to simulate the dynamics and evaluate the corresponding cost function. Hence, we easily get the primal solution $x_3^\sharp$. The optimal multiplier $\lambda_3^\sharp = \{\lambda_{3,t}^\sharp\}_{t\in\mathbb{T}}$ is then computed with the adjoint state given by Lemma 9.4. ◊

9.2.3 Numerical results

In this section, we present the numerical setting of the synthetic problem, the parameters of the algorithms that are used for the practical implementation of the decomposition method and a comparison between the numerical results given by the fixed-point algorithm and the analytical solution.

9.2.3.1 Choice of the numerical values for the test case

Problem (9.3) is studied on a horizon of $T = 10$ time steps. Then, we need to specify the numerical values for the dynamics and cost of the test case. The location of the non-zero values in the matrices A, B, R and O is chosen so as to confer a structure to the test case that is similar to the structure of the industrial problem presented in Chapter 8. This means that we introduce couplings between the entities in the test case that are the same as the couplings that occur between the components and the stock in the industrial system. Recall that in the synthetic test case, entities $i = 1$ and $i = 2$ are similar to physical components of the industrial system and that entity $i = 3$ plays the role of a common stock of spare parts. We set:

$$\begin{aligned} A &= \frac{1}{4}\begin{pmatrix} 2 & 0 & -1 \\ 0.2 & 2 & -1 \\ -1 & -1 & 1 \end{pmatrix}, \qquad B = \begin{pmatrix} -0.5 & 0 \\ 0 & -0.5 \\ 0 & 0 \end{pmatrix}, \\ R &= \begin{pmatrix} 5 & -1 & 0 \\ -1 & 2.5 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \qquad O = \begin{pmatrix} 2 & 0 \\ 0 & 1 \end{pmatrix}. \end{aligned} \tag{9.16}$$

We can check that the matrix O is positive definite and that R is positive semi-definite. Let us detail precisely how the choice of these numerical values has been carried out.

For the matrix A. First, the stock appears in the dynamics of every component, see (8.5), and conversely all components appear in the dynamics of the stock, see (8.6). Hence, the third line and the third column of A do not contain any zeros. We choose the same coefficients for $A_{3,1}$ and $A_{3,2}$ to model a case where the influence of the two components on the stock is the same. Similarly, $A_{1,3}$ and $A_{2,3}$ are chosen to be identical as the influence of the stock on the two components is the same. From (8.5), we see that the dynamics of component i only depends on components $j < i$. Therefore, $A_{1,2}$ is set to zero as the dynamics of component 1 does not depend on component 2. However, $A_{2,1}$ takes a non-zero value as the dynamics of component 2 depends on component 1. We choose a low value for $A_{2,1}$ compared to the diagonal elements of A. The reason is that the coupling between the components in the dynamics is weak. This coupling indeed only occurs in the rare situation where many components must be replaced at the same time without enough spare parts being available.

For the matrix B. Note that a control (*i.e.* a PM) on a component does not affect the other component nor the stock, hence the only non-zero coefficients of B are $B_{1,1}$ and $B_{2,2}$. The diagonal coefficients are chosen to be identical as a PM has the same effect on every component.

For the cost matrices R and O. The matrix R represents the cost due to CMs and forced outages. The coupling part, represented by off-diagonal coefficients $R_{1,2}$ and $R_{2,1}$, is due to forced outages. Moreover, no cost is linked to the stock, therefore we set the third line and third column of R to zero. Finally, O represents the PM cost. This cost does not create any coupling between the components. Therefore, the matrix O is chosen to be diagonal.

Remarks on the violation of some requirements of Theorem 7.4. With the choice (9.16) for the matrices A, B, R and O, some assumptions of Theorem 7.4 for the convergence of fixed-point algorithm are not satisfied. The operator J that appears in Theorem 7.4 is given by:

$$\mathsf{J} = 2 \times \begin{pmatrix} \mathsf{J}_0 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \mathsf{J}_T \end{pmatrix} \quad \text{where} \quad \begin{cases} \mathsf{J}_t = \begin{pmatrix} R & 0 \\ 0 & O \end{pmatrix}, \quad t \in \mathbb{T}_{-1} \,, \\ \mathsf{J}_T = R \,. \end{cases}$$

With the choice (9.16) for the matrix R, and more precisely because no cost is incurred by the stock in the industrial problem, the operator J is only monotone and not strongly monotone as required in Theorem 7.4. Hence, even for the synthetic problem, we are not within the theoretical scope of convergence of the fixed-point algorithm.

9.2.3.2 Algorithm setting

We run 25 iterations of the fixed-point algorithm 7. At iteration l, the two subproblems defined by (9.13) are solved with MADS in its default configuration with a budget of 1000 evaluations, which leads to a primal solution (x_i^l, u_i^l) of subproblem $i \in I$. Then, the optimal multiplier λ_i^l is computed using the backward recursion of Lemma 9.4. As mentioned in Remark 9.5, solving the subproblem (9.14) just amounts to simulate the dynamics $\Phi_3(x_3) = 0$. This gives a primal solution x_3^l of the subproblem. The optimal multiplier λ_3^l is also computed with Lemma 9.4.

9.2.3.3 Results of the numerical experiments

Figure 9.1 shows the evolution of the value of the objective function j during the iterations of the fixed-point algorithm. The analytical optimum is 111.15 and the solution given by the decomposition method is 111.21 which represents a gap of 0.05%. This is a very good result, especially accounting for the fact that some hypotheses of Theorem 7.4 are violated.

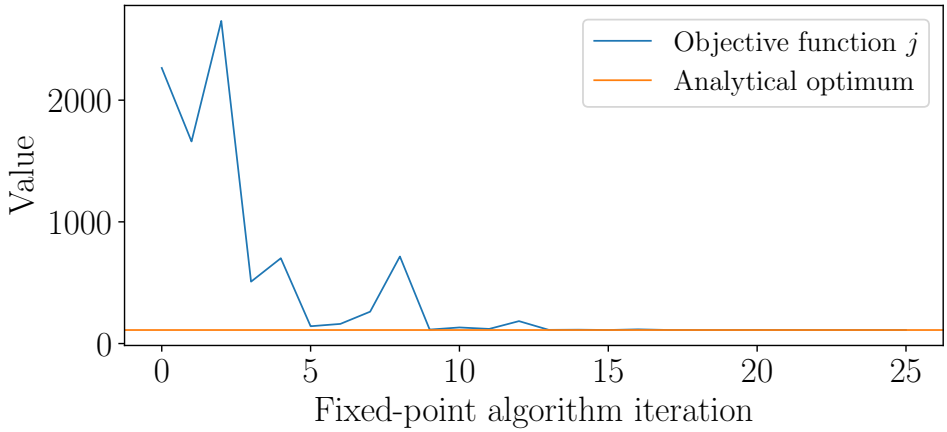


Figure 9.1: Convergence of the objective function j during the fixed-point algorithm in the deterministic case.

Figure 9.2 shows the convergence of some controls from the family $\{u_{i,t}\}_{(i,t)\in I\times\mathbb{T}_{-1}}$. The convergence plots for the whole family $\{u_{i,t}\}_{(i,t)\in I\times\mathbb{T}_{-1}}$ are given in Appendix C.1. We see that the controls indeed converge after around 20 iterations of the fixed-point

algorithm. However, the analysis shows that the fixed-point algorithm does not converge to the analytical optimum $u^\sharp$ (plotted in orange) but to another control u that is close to $u^\sharp$. This can be confirmed if we look carefully at the convergence plots given in Appendix C.1. This behavior is still not completely understood, as we can check numerically that the matrix C defined in Theorem 9.1 to compute the analytical optimum of the synthetic problem (9.3) is invertible and that $u^\sharp$ is indeed uniquely defined. Nevertheless, the fixed-point algorithm finds a solution with a very satisfying cost as already shown in Figure 9.1.

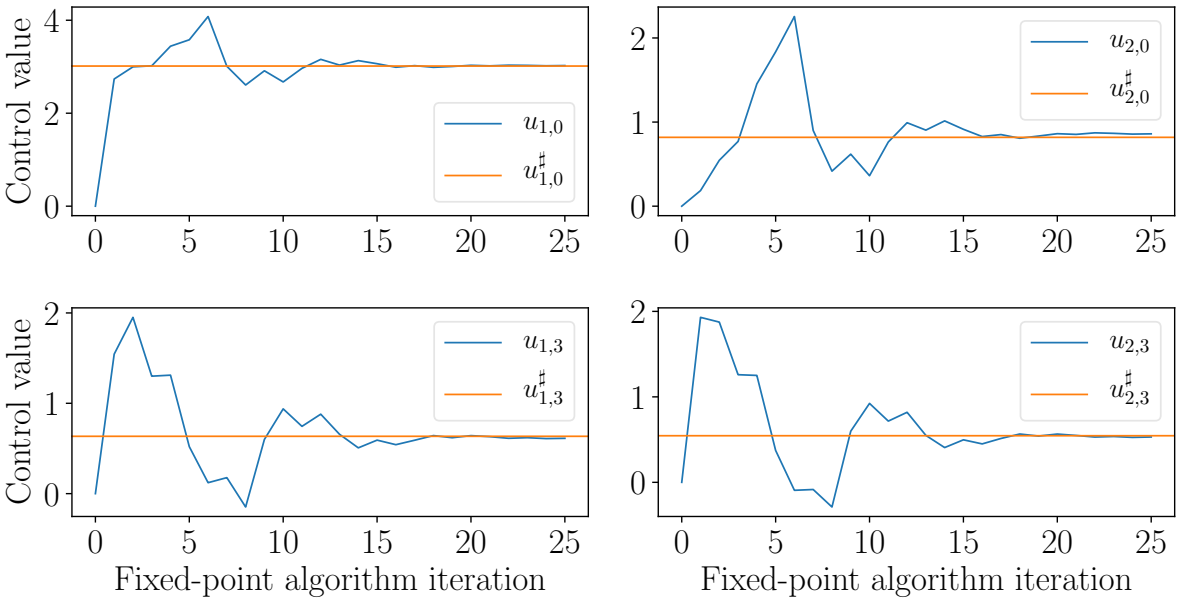


Figure 9.2: Convergence of the controls $\{u_{1,0}, u_{2,0}, u_{1,3}, u_{2,3}\}$ during the fixed-point algorithm in the deterministic case.

We can also look at the convergence of the multipliers $\{\lambda_{i,t}\}_{(i,t)\in\mathbb{I}\times\mathbb{T}}$ in Figure 9.3. As $\mathbb{I} = \{1, 2, 3\}$ and $\mathbb{T} = \{0, \dots, 10\}$, there are $3 \times 11 = 33$ multipliers. All multipliers converge after few iterations, confirming that the solutions of the subproblems are well coordinated and that the algorithm has indeed reached a fixed point.

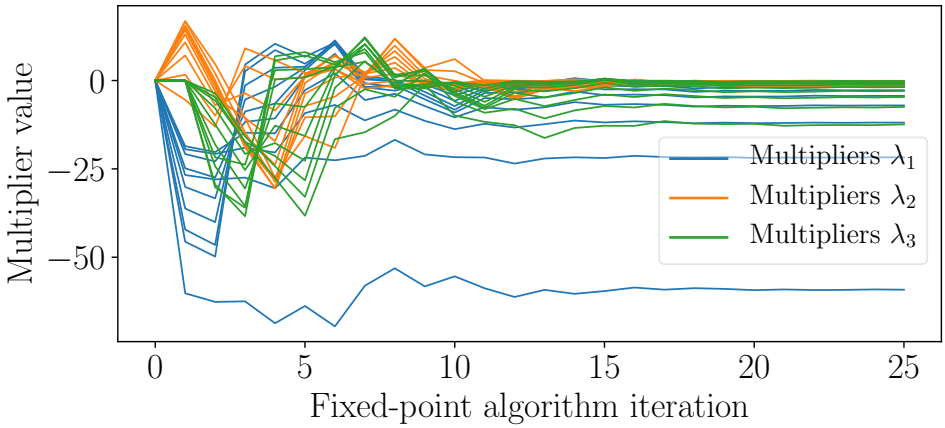


Figure 9.3: Convergence of the multipliers $\{\lambda_{i,t}\}_{(i,t)\in\mathbb{I}\times\mathbb{T}}$ during the fixed-point algorithm in the deterministic case.

9.2.3.4 Conclusion of the numerical experiments

This first test case for the fixed-point algorithm is promising as the decomposition-coordination scheme finds a solution with a cost that is very close to the analytical optimum, despite that the conditions of Theorem 7.4 are not all satisfied. In this section, we have considered a system with a deterministic dynamics. In the next section, we get closer to the industrial maintenance problem by considering a more challenging synthetic case, where the dynamics of the problem is stochastic.

9.3 The stochastic synthetic test case

The dynamics of the industrial system studied in Chapter 8 is stochastic due to the random failures of the components. This feature motivates the study of a test case with a stochastic dynamics. The stochastic synthetic problem is very similar to the deterministic one of Section 9.2. The difference is that the dynamics of the stochastic system is perturbed with an additive Gaussian noise. The objective function is then the expectation of the cost generated by the system.

We follow the same organization as in Section 9.2. We start by describing the synthetic system and we formulate the stochastic optimization problem. Then, we give the relevant analytical results. We go on by choosing an auxiliary function for the APP, leading to a scheme of decomposition by prediction. Finally, we run the fixed-point algorithm and compare the numerical and analytical solutions. A notable difference with the deterministic case is that in the numerical experiments, we only solve a Monte-Carlo approximation of the stochastic problem. The reason for using the SAA approach instead of solving the exact problem – which could be theoretically possible for the synthetic case – is that we aim at solving the problem as if we were in the realistic situation where the computation of the expected cost is intractable numerically.

9.3.1 Description of the stochastic test case

Similarly as in the deterministic case of Section 9.2, we consider a system of $N = 3$ entities on a discrete time horizon $T \in \mathbb{N}$. We still use the notations $\mathbb{I}$, I and $\mathbb{T}$ for the index sets and the notations $\mathbb{X}, \mathbb{X}_i, \mathbb{U}$ and $\mathbb{U}_i$ defined in Section 9.2.

In this part, we focus on a system with a stochastic dynamics, therefore the entities are characterized by random variables. Let $(\Omega, \mathcal{A}, \mathbb{P})$ be a probability space, the entities of the system are represented by the random vector:

$$\boldsymbol{X} = (\boldsymbol{X}_1, \boldsymbol{X}_2, \boldsymbol{X}_3) = ((\boldsymbol{X}_{1,0}, \dots, \boldsymbol{X}_{1,T}), (\boldsymbol{X}_{2,0}, \dots, \boldsymbol{X}_{2,T}), (\boldsymbol{X}_{3,0}, \dots, \boldsymbol{X}_{3,T})) \in \mathcal{X} \ ,$$

where $\mathcal{X} = L^2(\Omega, \mathcal{A}, \mathbb{P}; \mathbb{X})$ is the space of measurable square-integrable functions from Ω to $\mathbb{X}$. For all $i \in \mathbb{I}$, we have $\boldsymbol{X}_i \in \mathcal{X}_i = L^2(\Omega, \mathcal{A}, \mathbb{P}; \mathbb{X}_i)$ so that $\mathcal{X} = \prod_{i \in \mathbb{I}} \mathcal{X}_i$.

The controls are considered to be deterministic, as this is the case for the industrial application. They are represented by a vector:

$$u = (u_1, u_2) = ((u_{1,0}, \dots, u_{1,T-1}), (u_{2,0}, \dots, u_{2,T-1})) \in \mathbb{U} \ . \tag{9.17}$$

The admissible set for the controls is then the same as for the deterministic test case of Section 9.2.

The system evolves with a linear dynamics perturbed with an additive Gaussian noise. Let $\boldsymbol{W} = \{\boldsymbol{W}_{i,t}\}_{(i,t) \in \mathbb{I} \times \mathbb{T}}$ be a sequence of Gaussian random variables so that

$\boldsymbol{W}$ takes values in $\mathbb{X}$. We assume that the random vectors $\{\boldsymbol{W}_{\cdot,t}\}_{t\in\mathbb{T}}$ are independent, where $\boldsymbol{W}_{\cdot,t} = (\boldsymbol{W}_{1,t}, \boldsymbol{W}_{2,t}, \boldsymbol{W}_{3,t})$. Moreover, we suppose that for all $t \in \mathbb{T}$, $\boldsymbol{W}_{\cdot,t}$ has mean zero and variance $\Sigma \in \mathbb{R}^{3\times 3}$, that is:

$$\boldsymbol{W}_{\cdot,t} \sim \mathcal{N}(0, \Sigma) \ . \tag{9.18}$$

Dynamics. The dynamics of the system is given by the almost sure equality:

$$\Theta(\boldsymbol{X}, u, \boldsymbol{W}) = 0, \quad \mathbb{P}\text{-a.s. .} \tag{9.19}$$

where, similarly as in the deterministic case, $\Theta = \{\Theta_i\}_{i\in\mathbb{I}}$ with Θ_i representing the dynamics of entity i. We have $\Theta_i = \{\Theta_{i,t}\}_{t\in\mathbb{T}}$ with:

$$\begin{cases} \Theta_{i,t+1}(\boldsymbol{X}, u, \boldsymbol{W}) = \boldsymbol{X}_{i,t+1} - A_i \boldsymbol{X}_{\cdot,t} - B_i u_{\cdot,t} - \boldsymbol{W}_{i,t+1}, \quad \mathbb{P}\text{-a.s.}, \quad t \in \mathbb{T}_{-1} \ , \\ \Theta_{i,0}(\boldsymbol{X}, u, \boldsymbol{W}) = \boldsymbol{X}_{i,0} - x_{i,\text{init}}, \quad \mathbb{P}\text{-a.s. .} \end{cases} \tag{9.20}$$

We still have $x_{i,\text{init}} \in \mathbb{R}$, $A_i \in \mathbb{R}^3$ and $B_i \in \mathbb{R}^2$ and we denote by $A \in \mathbb{R}^{3\times 3}$ (resp. $B \in \mathbb{R}^{3\times 2}$) the matrix with rows $\{A_i\}_{i\in\mathbb{I}}$ (resp. $\{B_i\}_{i\in\mathbb{I}}$).

Cost. We consider again the quadratic cost function (9.2). This means that for a given outcome $\omega \in \Omega$, the cost generated by the system is:

$$j(\boldsymbol{X}(\omega), u) = \sum_{t=0}^{T} \left\langle \boldsymbol{X}_{\cdot,t}(\omega) \, , R\boldsymbol{X}_{\cdot,t}(\omega) \right\rangle + \sum_{t=0}^{T-1} \langle u_{\cdot,t} \, , Ou_{\cdot,t} \rangle \ .$$

Problem of interest. We aim at minimizing the expectation of the cost j, so the test case writes:

$$\begin{aligned} \min_{(\boldsymbol{X},u)\in\mathcal{X}\times\mathbb{U}} \ & J(\boldsymbol{X}, u) \\ \text{s.t. } & \Theta(\boldsymbol{X}, u, \boldsymbol{W}) = 0, \quad \mathbb{P}\text{-a.s. ,} \end{aligned} \tag{9.21}$$

where $J : (\boldsymbol{X}, u) \in \mathcal{X} \times \mathbb{U} \mapsto \mathbb{E}\left(j(\boldsymbol{X}, u)\right)$.

For the test case (9.21), when $\boldsymbol{X}$ satisfies the constraint $\Theta(\boldsymbol{X}, u, \boldsymbol{W}) = 0$, we can compute analytically the cost $J(\boldsymbol{X}, u)$ for any $u \in \mathbb{U}$ as shown in the following lemma.

Lemma 9.6. *Let $u \in \mathbb{U}$, and introduce for $t \in \mathbb{T}$:*

$$y_{\cdot,t} = A^t x_{\cdot,\text{init}} + \sum_{k=0}^{t-1} A^{t-1-k} B u_{\cdot,k} \ .$$

For the synthetic system (9.21), hence assuming that $\boldsymbol{X}$ satisfies the constraint (9.19), the expected cost can be expressed as a function $\tilde{J} : \mathbb{U} \to \mathbb{R}$ that depends only on u:

$$\tilde{J}(u) = \sum_{t=0}^{T-1} \langle u_{\cdot,t} \, , Ou_{\cdot,t} \rangle + \sum_{t=0}^{T} \left(\langle y_{\cdot,t} \, , Ry_{\cdot,t} \rangle + \sum_{k=0}^{t-1} \operatorname{tr}\left(RA^{t-1-k} \Sigma {A^{t-1-k}}^\top \right) \right) \ .$$

Proof. We have:

$$J(\boldsymbol{X}, u) = \sum_{t=0}^{T-1} \langle u_{\cdot,t}, Ou_{\cdot,t} \rangle + \sum_{t=0}^{T} \mathbb{E}\left(\left\langle \boldsymbol{X}_{\cdot,t}, R\boldsymbol{X}_{\cdot,t} \right\rangle\right) . \tag{9.22}$$

We just need to compute $\mathbb{E}\left(\left\langle \boldsymbol{X}_{\cdot,t}, R\boldsymbol{X}_{\cdot,t} \right\rangle\right)$. Using the dynamics (9.20), we get by induction that for $t \in \mathbb{T}$:

$$\boldsymbol{X}_{\cdot,t} = y_{\cdot,t} + \sum_{k=0}^{t-1} A^{t-1-k} \boldsymbol{W}_{\cdot,k+1}, \quad \mathbb{P}\text{-a.s.} .$$

Then:

$$\begin{aligned} \left\langle \boldsymbol{X}_{\cdot,t}, R\boldsymbol{X}_{\cdot,t} \right\rangle = \langle y_{\cdot,t}, Ry_{\cdot,t} \rangle + 2y_{\cdot,t}^{\top} R \sum_{k=0}^{t-1} A^{t-1-k} \boldsymbol{W}_{\cdot,k+1} \\ + \left(\sum_{k=0}^{t-1} A^{t-1-k} \boldsymbol{W}_{\cdot,k+1}\right)^{\top} R \sum_{k=0}^{t-1} A^{t-1-k} \boldsymbol{W}_{\cdot,k+1} . \end{aligned} \tag{9.23}$$

Using that $\mathbb{E}\left(\boldsymbol{W}_{\cdot,t}\right) = 0$ for all t, we get:

$$\mathbb{E}\left(2y_{\cdot,t}^{\top} R \sum_{k=0}^{t-1} A^{t-1-k} \boldsymbol{W}_{\cdot,k+1}\right) = 0 . \tag{9.24}$$

Finally, for the last term, let $\boldsymbol{Z}_{\cdot,t} = \sum_{k=0}^{t-1} A^{t-1-k} \boldsymbol{W}_{\cdot,k+1}$. We have:

$$\begin{aligned} \mathbb{E}\left(\boldsymbol{Z}_{\cdot,t}{}^{\top} R \boldsymbol{Z}_{\cdot,t}\right) &= \mathbb{E}\left(\operatorname{tr}\left(\boldsymbol{Z}_{\cdot,t}{}^{\top} R \boldsymbol{Z}_{\cdot,t}\right)\right) = \mathbb{E}\left(\operatorname{tr}\left(R \boldsymbol{Z}_{\cdot,t} \boldsymbol{Z}_{\cdot,t}{}^{\top}\right)\right) \\ &= \mathbb{E}\left(\sum_{k \neq l} \operatorname{tr}\left(R A^{t-1-k} \boldsymbol{W}_{\cdot,k+1} \boldsymbol{W}_{\cdot,l+1}{}^{\top} A^{t-1-l^{\top}}\right)\right) \\ &\quad + \mathbb{E}\left(\sum_{k=0}^{t-1} \operatorname{tr}\left(R A^{t-1-k} \boldsymbol{W}_{\cdot,k+1} \boldsymbol{W}_{\cdot,k+1}{}^{\top} A^{t-1-k^{\top}}\right)\right) . \end{aligned}$$

Now, note that for $k \neq l$, $\boldsymbol{W}_{\cdot,k}$ and $\boldsymbol{W}_{\cdot,l}$ are independent, which implies that:

$$\mathbb{E}\left(\boldsymbol{W}_{\cdot,k} \boldsymbol{W}_{\cdot,l}{}^{\top}\right) = \mathbb{E}\left(\boldsymbol{W}_{\cdot,k}\right) \mathbb{E}\left(\boldsymbol{W}_{\cdot,l}\right)^{\top} = 0, \quad k \neq l ,$$

and recall from (9.18) that $\mathbb{E}\left(\boldsymbol{W}_{\cdot,k} \boldsymbol{W}_{\cdot,k}{}^{\top}\right) = \Sigma$ for all k. Therefore:

$$\mathbb{E}\left(\boldsymbol{Z}_{\cdot,t}{}^{\top} R \boldsymbol{Z}_{\cdot,t}\right) = \sum_{k=0}^{t-1} \operatorname{tr}\left(R A^{t-1-k} \Sigma A^{t-1-k^{\top}}\right) . \tag{9.25}$$

Combining (9.23) with (9.24) and (9.25), we get:

$$\mathbb{E}\left(\left\langle \boldsymbol{X}_{\cdot,t}, R\boldsymbol{X}_{\cdot,t} \right\rangle\right) = \langle y_{\cdot,t}, Ry_{\cdot,t} \rangle + \sum_{k=0}^{t-1} \operatorname{tr}\left(R A^{t-1-k} \Sigma A^{t-1-k^{\top}}\right) . \tag{9.26}$$

Therefore, $\mathbb{E}\left(\left\langle \boldsymbol{X}_{\cdot,t}, R\boldsymbol{X}_{\cdot,t} \right\rangle\right)$ depends only on u and plugging (9.26) into (9.22) yields the desired result. □

We can also compute the analytical solution $u^\sharp$ of the stochastic problem (9.21). In fact, the optimal control is the same as in the deterministic case.

Theorem 9.7. *The optimal control $u^\sharp$ for Problem (9.21) is given by:*

$$u^\sharp = C^{-1} d \, .$$

Here $u^\sharp$ is to be understood as the vector $(u^\sharp_{1,0}, u^\sharp_{2,0}, \dots, u^\sharp_{1,T-1}, u^\sharp_{2,T-1})^\top \in \mathbb{R}^{2T}$ where the coordinates are sorted by time step *whereas in (9.17) the coordinates are sorted by* component*. The matrix $C \in \mathbb{R}^{2T \times 2T}$ is defined by blocks of size 2×2 such that:*

$$\begin{cases} C_{t,t} = O + \displaystyle\sum_{p=t}^{T-1} (A^{p-t}B)^\top R (A^{p-t}B) \in \mathbb{R}^{2\times 2}, \quad t \in \mathbb{T}_{-1} \, , \\ C_{t,k} = \displaystyle\sum_{p=\max(k,t)}^{T-1} (A^{p-t}B)^\top R (A^{p-k}B) \in \mathbb{R}^{2\times 2}, \quad t,k \in \mathbb{T}_{-1}, \ t \neq k \, , \end{cases}$$

and $d \in \mathbb{R}^{2T}$ is defined by blocks of size 2×1 such that:

$$d_t = -\sum_{p=t}^{T-1} (A^{p-t}B)^\top R A^{p+1} x_{\cdot,\text{init}} \in \mathbb{R}^2, \quad t \in \mathbb{T}_{-1} \, .$$

Proof. We have $\tilde{J} = \tilde{\jmath} + \alpha$ with $\tilde{\jmath}$ defined in (9.4) and $\alpha = \sum_{k=0}^{t-1} \operatorname{tr}\left(R A^{t-1-k} \Sigma {A^{t-1-k}}^\top\right)$ does not depend on u. Therefore $\tilde{J}$ and $\tilde{\jmath}$ have the same minimizer $u^\sharp$, which is given by Theorem 9.1. □

Similarly as in the deterministic case, the analytical optimal control $u^\sharp$ and the associated optimal cost $\tilde{J}(u^\sharp)$ will be used to check the quality of the numerical results given by the decomposition by prediction.

9.3.2 Application of the APP for a decomposition by prediction

The process to apply the APP for a decomposition by prediction of the stochastic test case (9.21) is the same as in the deterministic case of §9.2.2. We aim at designing three subproblems involving respectively the spaces $\mathcal{X}_1 \times \mathbb{U}_1$, $\mathcal{X}_2 \times \mathbb{U}_2$ and $\mathcal{X}_3$. This means that we consider the following decomposition of the space $\mathcal{X} \times \mathbb{U}$:

$$\mathcal{X} \times \mathbb{U} = (\mathcal{X}_1 \times \mathbb{U}_1) \times (\mathcal{X}_2 \times \mathbb{U}_2) \times \mathcal{X}_3 \, . \tag{9.27}$$

In the stochastic case, we have $\Theta : \mathcal{X} \times \mathbb{U} \to \mathcal{X}$. Similarly as in the deterministic case, we consider the following decomposition of the dual space $\mathcal{X}$:

$$\mathcal{X} = \mathcal{X}_1 \times \mathcal{X}_2 \times \mathcal{X}_3 \, . \tag{9.28}$$

We still treat the additive and non-additive part of the cost separately by writing $j = j^\Sigma + j^\Delta$ as defined in (9.8) and we introduce:

$$\begin{aligned} J^\Sigma(\boldsymbol{X}, u) &= \mathbb{E}\left(j^\Sigma(\boldsymbol{X}, u)\right), & J^\Delta(\boldsymbol{X}, u) &= \mathbb{E}\left(j^\Delta(\boldsymbol{X}, u)\right) \\ J_i^\Sigma(\boldsymbol{X}_i, u_i) &= \mathbb{E}\left(j_i^\Sigma(\boldsymbol{X}_i, u_i)\right), \quad i \in I, & J_3^\Sigma(\boldsymbol{X}_3) &= \mathbb{E}\left(j_3^\Sigma(\boldsymbol{X}_3)\right) \, . \end{aligned}$$

Then, we introduce an auxiliary cost function K that is additive with respect to the decomposition (9.27) of the primal space and an auxiliary dynamics Φ that is block-diagonal with respect to the decompositions (9.27) and (9.28) of the primal and dual spaces. The difference with the deterministic case is that the auxiliary functions K and Φ are defined on $\mathcal{X} \times \mathbb{U}$ rather than on $\mathbb{X} \times \mathbb{U}$. The deterministic cost that appears in (9.10) is replaced by an expected cost. Let $\overline{\boldsymbol{X}} \in \mathcal{X}$ and $\bar{u} \in \mathbb{U}$ and define:

- An auxiliary cost function $K : \mathcal{X} \times \mathbb{U} \to \mathbb{R}$ such that:

$$K(\boldsymbol{X}, u) = K_1(\boldsymbol{X}_1, u_1) + K_2(\boldsymbol{X}_2, u_2) + K_3(\boldsymbol{X}_3) \ , \tag{9.29}$$

  with

$$\begin{cases} K_1(\boldsymbol{X}_1, u_1) = \mathbb{E}\left(j^{\Delta}\left((\boldsymbol{X}_1, \overline{\boldsymbol{X}}_2, \overline{\boldsymbol{X}}_3), (u_1, \bar{u}_2)\right)\right) \ , \\ K_2(\boldsymbol{X}_2, u_2) = \mathbb{E}\left(j^{\Delta}\left((\overline{\boldsymbol{X}}_1, \boldsymbol{X}_2, \overline{\boldsymbol{X}}_3), (\bar{u}_1, u_2)\right)\right) \ , \\ K_3(\boldsymbol{X}_3) = \mathbb{E}\left(j^{\Delta}\left((\overline{\boldsymbol{X}}_1, \overline{\boldsymbol{X}}_2, \boldsymbol{X}_3), (\bar{u}_1, \bar{u}_2)\right)\right) \ . \end{cases}$$

- An auxiliary dynamics $\Phi : \mathcal{X} \times \mathbb{U} \times \mathcal{X} \to \mathcal{X}$ such that:

$$\Phi(\boldsymbol{X}, u, \boldsymbol{W}) = (\Phi_1(\boldsymbol{X}_1, u_1, \boldsymbol{W}), \Phi_2(\boldsymbol{X}_2, u_2, \boldsymbol{W}), \Phi_3(\boldsymbol{X}_3, \boldsymbol{W})) \ , \tag{9.30}$$

  with

$$\begin{cases} \Phi_1(\boldsymbol{X}_1, u_1, \boldsymbol{W}) = \Theta_1((\boldsymbol{X}_1, \overline{\boldsymbol{X}}_2, \overline{\boldsymbol{X}}_3), (u_1, \bar{u}_2), \boldsymbol{W}) \ , \\ \Phi_2(\boldsymbol{X}_2, u_2, \boldsymbol{W}) = \Theta_2((\overline{\boldsymbol{X}}_1, \boldsymbol{X}_2, \overline{\boldsymbol{X}}_3), (\bar{u}_1, u_2), \boldsymbol{W}) \ , \\ \Phi_3(\boldsymbol{X}_3, \boldsymbol{W}) = \Theta_3((\overline{\boldsymbol{X}}_1, \overline{\boldsymbol{X}}_2, \boldsymbol{X}_3), (\bar{u}_1, \bar{u}_2), \boldsymbol{W}) \ . \end{cases}$$

In the same way as in §9.2.2, now that the auxiliary functions are chosen, we can derive the auxiliary problem.

Proposition 9.8. *Let $(\overline{\boldsymbol{X}}, \overline{\boldsymbol{\Lambda}}) \in \mathcal{X}^2$ and $\bar{u} \in \mathbb{U}$. The auxiliary problem that results from the choices (9.29) and (9.30) of the auxiliary cost and dynamics is given by:*

$$\begin{aligned} \min_{(\boldsymbol{X}, u) \in \mathcal{X} \times \mathbb{U}} \ & K(\boldsymbol{X}, u) + J^{\Sigma}(\boldsymbol{X}, u) + \left\langle \overline{\boldsymbol{\Lambda}} \ , (\Theta'(\overline{\boldsymbol{X}}, \bar{u}) - \Phi'(\overline{\boldsymbol{X}}, \bar{u})) \cdot (\boldsymbol{X}, u) \right\rangle \\ & s.t. \ \Phi(\boldsymbol{X}, u, \boldsymbol{W}) = 0 \end{aligned} \tag{9.31}$$

Proof. With the choice (9.29) for the auxiliary cost, we have:

$$\nabla J^{\Delta}(\overline{\boldsymbol{X}}, \bar{u}) - \nabla K(\overline{\boldsymbol{X}}, \bar{u}) = 0 \ .$$

With the choice (9.30) for the auxiliary dynamics, we have:

$$\Theta(\overline{\boldsymbol{X}}, \bar{u}, \boldsymbol{W}) - \Phi(\overline{\boldsymbol{X}}, \bar{u}, \boldsymbol{W}) = 0, \quad \mathbb{P}\text{-a.s.} \ .$$

Thus, the auxiliary problem (9.31) for the stochastic synthetic system corresponds exactly to the theoretical auxiliary problem (7.5). □

The auxiliary problem (9.31) is decomposable into independent subproblems that are given for $i \in I$ by:

$$\begin{aligned} \min_{(\boldsymbol{X}_i, u_i) \in \mathcal{X}_i \times \mathbb{U}_i} \ & K_i(\boldsymbol{X}_i, u_i) + J_i^{\Sigma}(\boldsymbol{X}_i, u_i) + \mathbb{E}\left(\sum_{\substack{j=1 \\ j \neq i}}^{3} \left\langle \overline{\boldsymbol{\Lambda}}_j \ , \partial_{(\boldsymbol{X}_i, u_i)} \Theta_j(\overline{\boldsymbol{X}}, \bar{u}, \boldsymbol{W}) \cdot (\boldsymbol{X}_i, u_i) \right\rangle \right) \\ & \text{s.t. } \Phi_i(\boldsymbol{X}_i, u_i, \boldsymbol{W}) = 0 \ . \end{aligned} \tag{9.32}$$

The third subproblem writes:

$$\begin{aligned} \min_{\boldsymbol{X}_3 \in \mathcal{X}_3} \; & K_3(\boldsymbol{X}_3) + J_3^{\Sigma}(\boldsymbol{X}_3) + \sum_{j=1}^{2} \left\langle \overline{\boldsymbol{\Lambda}}_j \, , \partial_{\boldsymbol{X}_3} \Theta_j(\overline{\boldsymbol{X}}, \bar{u}, \boldsymbol{W}) \cdot \boldsymbol{X}_3 \right\rangle \\ \text{s.t. } & \Phi_3(\boldsymbol{X}_3, \boldsymbol{W}) = 0 \; . \end{aligned} \tag{9.33}$$

We can now apply Algorithm 7 to solve Problem (9.21). For a real-world stochastic optimization problem, such as the industrial problem presented in Chapter 8, the exact computation of the expectation in the cost is intractable numerically. This is why, in order to proceed as if we were in a realistic situation, we choose to solve a Monte-Carlo approximation of the test case (9.21) with $Q = 1000$ outcomes $\omega_1, \ldots, \omega_Q \in \Omega$:

$$\begin{aligned} \min_{(\boldsymbol{X}, u) \in \mathcal{X} \times \mathbb{U}} \; & \frac{1}{Q} \sum_{q=1}^{Q} j(\boldsymbol{X}(\omega_q), u) \\ \text{s.t. } & \Theta(\boldsymbol{X}(\omega_q), u, \boldsymbol{W}(\omega_q)) = 0, \quad q \in \{1, \ldots, Q\} \; . \end{aligned} \tag{9.34}$$

In the sequel, we only use the analytical value of J, given by Lemma 9.6, to assess the quality of the numerical solution, but not in the resolution procedure itself.

The Monte-Carlo approximation of subproblems (9.32) are solved with the black-box algorithm MADS, which returns a primal solution $\left(\{\boldsymbol{X}_i^\sharp(\omega_q)\}_{q \in \{1,\ldots,Q\}}, u_i^\sharp\right)$. Then, we compute the optimal multipliers $\{\boldsymbol{\Lambda}_i^\sharp(\omega_q)\}_{q \in \{1,\ldots,Q\}}$ with the following backward recursion, valid for $i \in \mathbb{I}$:

$$\begin{cases} \boldsymbol{\Lambda}_{i,T}^\sharp = -2\bar{R}_i \overline{\boldsymbol{X}}_{\,\cdot,T} - 2R_{i,i} \boldsymbol{X}_{i,T}^\sharp \; , \\ \boldsymbol{\Lambda}_{i,t}^\sharp = -2\bar{R}_i \overline{\boldsymbol{X}}_{\,\cdot,t} - 2R_{i,i} \boldsymbol{X}_{i,t}^\sharp + \bar{A}_i^\top \overline{\boldsymbol{\Lambda}}_{\,\cdot,t+1} + A_{i,i} \boldsymbol{\Lambda}_{i,t+1}^\sharp, \quad t \in \mathbb{T}_{-1} \; , \end{cases} \tag{9.35}$$

The recursion (9.35) can be derived similarly as in Lemma 9.4 as we only consider a finite number of scenarios in the Monte-Carlo approximation of the subproblems. Remark 9.5, stating that the third subproblem is easy to solve numerically, still holds in the stochastic case. Therefore, we just need to simulate the dynamics of subproblem (9.33) to get the primal solution $\{\boldsymbol{X}_3^\sharp(\omega_q)\}_{q \in \{1,\ldots,Q\}}$ and we again compute the multipliers $\{\boldsymbol{\Lambda}_3^\sharp(\omega_q)\}_{q \in \{1,\ldots,Q\}}$ with the backward recursion (9.35).

9.3.3 Numerical results

In this section, we describe the settings of the problem and of the algorithm used to run the APP fixed-point algorithm and we compare the analytical and numerical solutions of the synthetic problem (9.21).

9.3.3.1 Settings of the numerical experiments

We use the same settings for the system as in the deterministic case, that is we take $T = 10$ and use the matrices A, B, R and O defined by (9.16). We run 25 iterations of the fixed-point algorithm 7. At each iteration of the algorithm, the Monte-Carlo approximation of the two subproblems defined by (9.32) is solved with MADS in its default configuration with 1000 evaluations. Solving the third subproblem just amounts to simulate the auxiliary dynamics, see Remark 9.5. The optimal multipliers are computed with the backward recursion (9.35).

9.3.3.2 Results of the numerical experiments

Figure 9.4 shows the evolution of the value of the objective function $\tilde{J}$ during the iterations of the fixed-point algorithm: after each iteration of the algorithm, we evaluate $\tilde{J}(u^l)$ using Lemma 9.6, where u^l is the current iterate. The analytical optimum is 841.73 and the solution given by the decomposition method is 843.15, which represents a gap of 0.17%. The gap between the analytical and numerical solutions can be explained by the fact that we solve the approximate problem (9.34) instead of the original problem (9.21). This approximation is new compared to the deterministic case of Section 9.2. However, even with the use of a Monte-Carlo approximation of Problem (9.21), the optimality gap remains low and the numerical solution is very satisfying.

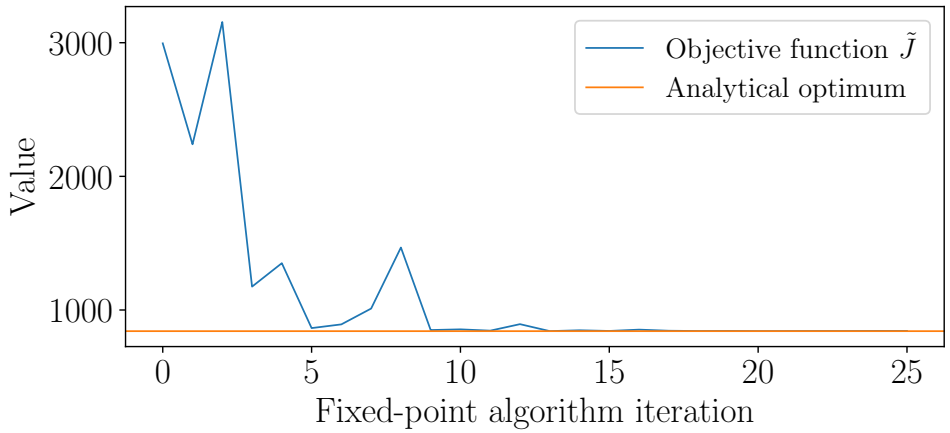


Figure 9.4: Convergence of the objective function during the fixed-point algorithm in the stochastic case.

Figure 9.5 shows the convergence of some controls from the family $\{u_{i,t}\}_{(i,t)\in I\times\mathbb{T}_{-1}}$. The convergence plots for the whole family $\{u_{i,t}\}_{(i,t)\in I\times\mathbb{T}_{-1}}$ are given in Appendix C.2. The same analysis and conclusion as in the deterministic case of Section 9.2 applies, namely, that the controls converge but not to the analytical optimum $u^\sharp$ (plotted in orange). This can be confirmed by a careful observation of the plots given in Appendix C.2. The solution found by the decomposition by prediction is nevertheless very good as it generates a cost that is only 0.17% higher than the optimal cost as observed on Figure 9.4.

We also look at the convergence of the multipliers. For each scenario ω_q, there is a family of multipliers $\{\mathbf{\Lambda}_{i,t}(\omega_q)\}_{(i,t)\in\mathbb{I}\times\mathbb{T}}$. Hence, as $\mathbb{I}=\{1,2,3\}$, $\mathbb{T}=\{0,\dots,10\}$ and $Q=1000$, we manipulate a total of $|\mathbb{I}|\times|\mathbb{T}|\times Q = 3.3\times10^4$ multipliers. Figure 9.6 shows the evolution of the $|\mathbb{I}|\times|\mathbb{T}| = 33$ multipliers obtained for the scenario ω_1 during the iterations of the fixed-point algorithm. Similarly as in the deterministic case of Section 9.2, the multipliers converge after few iterations. The plots of the multipliers for the other scenarios are not given in the manuscript, but we can check that they behave similarly as in Figure 9.6. Therefore, all the multipliers converge, showing that the coordination of the subproblems is ensured.

9.3.3.3 Conclusion of the numerical experiments

These numerical tests show that after few iterations, the fixed-point algorithm converges to a solution u that is very close, but different, of the analytical solution $u^\sharp$. For

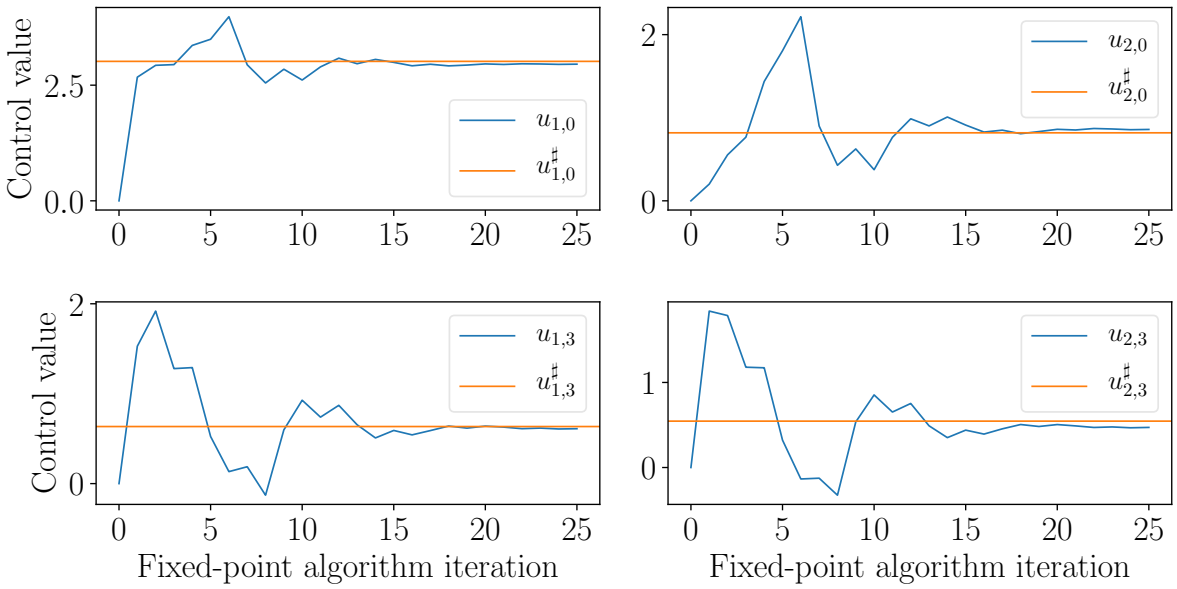


Figure 9.5: Convergence of the controls $\{u_{1,0}, u_{2,0}, u_{1,3}, u_{2,3}\}$ during the fixed-point algorithm in the stochastic case.

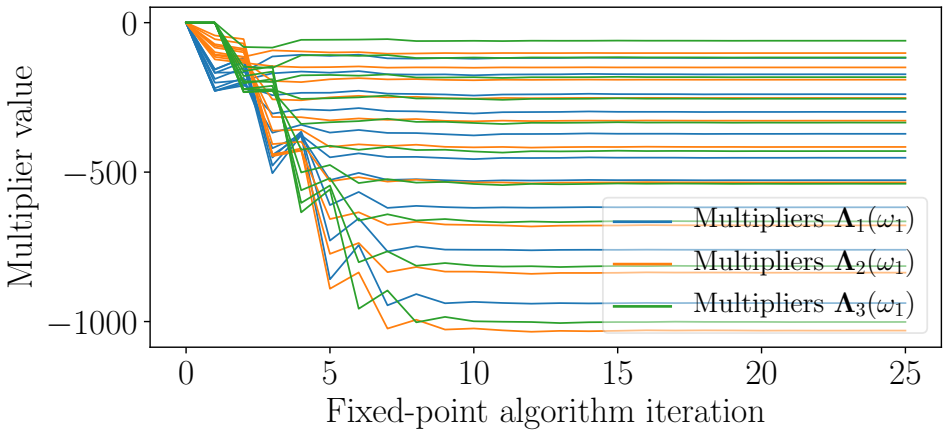


Figure 9.6: Convergence of the multipliers during the fixed-point algorithm in the stochastic case.

the same reasons as in the deterministic case, this behavior is not yet fully understood. In the stochastic case, we solve the Monte-Carlo approximation (9.34) of the original problem (9.21). With $Q = 1000$ scenarios, the numerical solution of the approximate problem is a good solution for the original problem.

9.4 Conclusion

We have applied the APP on a deterministic and on a stochastic test case. The motivation is to implement the decomposition by prediction on problems with increasing difficulty. In this way, we validate the decomposition approach before tackling the industrial maintenance optimization problem in Chapter 10. We have designed test cases with a linear dynamics and quadratic cost functions that have the same coupling structure as the industrial system. However, even for these synthetic test cases, the assumptions of Theorem 7.4 – that are sufficient for the convergence of the fixed-point algorithm towards the optimal solution – are not satisfied. Nevertheless, both in the

deterministic and the stochastic case, the fixed-point algorithm achieves a very good performance with less than 0.17% difference between the analytical optimal cost and the numerical solution. Therefore, these results allow to validate our implementation of the decomposition by prediction and are very promising for the industrial application.

10 Application of the APP on an industrial maintenance optimization problem

Sub-optimization is when everyone is for himself. Optimization is when everyone is working to help the company.

William Edwards Deming

Contents

10.1 Introduction

The goal of this chapter is to solve the industrial maintenance optimization problem, introduced in Chapter 8, with the APP. We aim at tackling the most demanding case of a system with 80 components. In Chapter 9, we have successfully applied the decomposition methodology on two test cases that have the same coupling structure as the industrial system. However, the application of the APP to the industrial problem is not as straightforward as for the synthetic cases. Indeed, the model of the industrial system involves several integer variables whereas the APP is based on variational techniques.

Let us expose the approach used for the industrial application of the APP. The beginning of the chapter follows the same guidelines as Chapter 9. We start, in Section 10.2, by specifying a decomposition of the admissible space and of the space of contraints. Then, in Section 10.3, we construct an auxiliary problem that is adapted to the proposed decomposition. The main novelty compared to the synthetic case turns up in Section 10.4, where we proceed to a continuous relaxation of the system in order to compute the gradients that appear in the auxiliary problem. This is where we overcome the difficulty caused by integer variables. In Section 10.5, we give the explicit formulation of the independent subproblems arising from the decomposition of this auxiliary problem. In Section 10.6, we present an efficient implementation of the APP fixed-point algorithm mixing a parallel and sequential strategy for the resolution of the subproblems. We will see that some parameters, that either appear in the auxiliary problem or linked to the relaxation of the system, influence the performance of the fixed-point algorithm. Section 10.7 is devoted to a comprehensive study for the tuning of these parameters in the algorithm. Finally, in Section 10.8, we present the results of the numerical experiments on the large-scale industrial system of 80 components.

Contributions. This chapter presents the industrial application of the APP. To our knowledge, this is the first time that a scheme of decomposition by prediction is applied for a maintenance optimization problem. The other contributions of the chapter are the following:

1. The original formulation of the maintenance optimization problem (8.10) is not suited to the application of the APP. To overcome this difficulty, we propose a continuous relaxation of the system.

2. In order to be computationally efficient, the implementation of the fixed-point algorithm is tailored to the industrial problem with its mixed parallel/sequential strategy.

3. The performance of the algorithm is dependent on some parameters, for which the tuning is carefully studied using tools for sensitivity analysis.

4. Finally, we manage to run the fixed-point algorithm on the most demanding industrial case of 80 components. A comparative study shows that we achieve sensible gains compared to the blackbox method currently used at EDF.

10.2 Decomposition of the space by component

This section specifies the decomposition of the admissible space and of the cone of constraints that is considered for the design of the decomposition by prediction. The notations used in this chapter have been introduced in Chapter 8.

Considering the physical nature of the industrial system composed of n components and a stock, we choose to decompose the problem in $n+1$ subproblems and call this decomposition a *decomposition by component*. More precisely, for $i \in I = \{1, \dots, n\}$, the i-th subproblem is called *subproblem on component i* since it is solved on $\mathcal{X}_i \times \mathbb{U}_i$ and only involves the dynamics of component i. The $(n+1)$-th subproblem is called *subproblem on the stock* since it is solved on $\mathcal{S}$ and only involves the dynamics of the stock. This means that the admissible space $\mathcal{X} \times \mathcal{S} \times \mathbb{U}$ of Problem (8.10) is decomposed as a product of a $n+1$ subspaces:

$$\mathcal{X} \times \mathcal{S} \times \mathbb{U} = (\mathcal{X}_1 \times \mathbb{U}_1) \times \ldots \times (\mathcal{X}_n \times \mathbb{U}_n) \times \mathcal{S} \,, \tag{10.1}$$

where, for $(\boldsymbol{X}, \boldsymbol{S}, u) = ((\boldsymbol{X}_1, \dots, \boldsymbol{X}_n), \boldsymbol{S}, (u_1, \dots, u_n)) \in \mathcal{X} \times \mathcal{S} \times \mathbb{U}$, we have:

$$(\boldsymbol{X}_i, u_i) \in \mathcal{X}_i \times \mathbb{U}_i \text{ for all } i \in I \text{ and } \boldsymbol{S} \in \mathcal{S} \,.$$

The constraint in Problem (8.10), that is $\Theta(\boldsymbol{X}, \boldsymbol{S}, u, \boldsymbol{W}) \in -C$ with $C = \{0\}_{\mathcal{L}}$ is decomposed through the following cone decomposition:

$$C = \{0\}_{\mathcal{L}} = \{0\}_{\mathcal{L}_1} \times \ldots \times \{0\}_{\mathcal{L}_n} \times \{0\}_{\mathcal{L}_{\boldsymbol{S}}} = C_1 \times \ldots C_n \times C_{\boldsymbol{S}} \,, \tag{10.2}$$

where for $i \in I$, $\{0\}_{\mathcal{L}_i}, \{0\}_{\mathcal{L}_{\boldsymbol{S}}}$ and $\{0\}_{\mathcal{L}}$ denote the null function of $\mathcal{L}_i, \mathcal{L}_{\boldsymbol{S}}$ and $\mathcal{L}$ respectively. We recall that Θ_i takes values in $\mathcal{L}_i$ and $\Theta_{\boldsymbol{S}}$ takes values in $\mathcal{L}_{\boldsymbol{S}}$.

10.3 Construction of an auxiliary problem

In this section, we choose the auxiliary functions that lead to the construction of a decomposable auxiliary problem, as described in Section 7.3.

Problem (8.10) is not directly decomposable by component because of couplings, highlighted in Chapter 8, that we recall now.

- The expected maintenance cost $\mathbb{E}\left(\sum_{i=1}^{n} j_i(\boldsymbol{X}_i, u_i)\right)$ is additive with respect to the decomposition by component and can be identified with J^{Σ} in Chapter 7.
- The forced outage cost j^F induces a non-additive coupling between the components. The expected forced outage cost $\mathbb{E}\left(j^F(\boldsymbol{X}_{1:n})\right)$ can be identified with J^{Δ} in Chapter 7.
- The dynamics Θ_i of component i induces a coupling with the stock and all components with index $j < i$. The stock dynamics, $\Theta_{\boldsymbol{S}}$, is coupling the stock with all components.

In order to obtain a decomposition of Problem (8.10) by component, we use the canonical technique from Example 7.3. We define an additive mapping K and a block diagonal mapping Φ so that the resulting auxiliary problem is decomposable. We also choose to augment the auxiliary cost K with a strongly convex term in order to ease the numerical convergence of the method. Let $\overline{\boldsymbol{X}} = (\overline{\boldsymbol{X}}_1, \dots, \overline{\boldsymbol{X}}_n) \in \mathcal{X}$, $\overline{\boldsymbol{S}} \in \mathcal{S}$, $\overline{u} \in \mathbb{U}$, $\overline{\boldsymbol{\Lambda}} \in C^{\star}$ and $\gamma_x, \gamma_s, \gamma_u > 0$. We consider:

- An additive auxiliary cost function $K : \mathcal{X} \times \mathcal{S} \times \mathbb{U} \to \mathbb{R}$:

$$K(\boldsymbol{X}, \boldsymbol{S}, u) = \sum_{i=1}^{n} K_i(\boldsymbol{X}_i, u_i) + K_{\boldsymbol{S}}(\boldsymbol{S}) \ ,$$

with

$$K_i(\boldsymbol{X}_i, u_i) = \mathbb{E}\Big(j^F(\overline{\boldsymbol{X}}_{1:i-1}, \boldsymbol{X}_i, \overline{\boldsymbol{X}}_{i+1:n}) + \frac{\gamma_x}{2} \left\| \boldsymbol{X}_i \right\|^2 + \frac{\gamma_u}{2} \left\| u_i \right\|^2 \Big), \quad i \in I \ ,$$
$$K_{\boldsymbol{S}}(\boldsymbol{S}) = \mathbb{E}\Big(\frac{\gamma_s}{2} \left\| \boldsymbol{S} \right\|^2 \Big) \ ,$$

- A block-diagonal auxiliary dynamics mapping $\Phi : \mathcal{X} \times \mathcal{S} \times \mathbb{U} \times \mathcal{W} \to \mathcal{L}$:

$$\Phi(\boldsymbol{X}, \boldsymbol{S}, u, \boldsymbol{W}) = (\Phi_1(\boldsymbol{X}_1, u_1, \boldsymbol{W}_1), \dots, \Phi_n(\boldsymbol{X}_n, u_n, \boldsymbol{W}_n), \Phi_{\boldsymbol{S}}(\boldsymbol{S})) \ ,$$

with

$$\Phi_i(\boldsymbol{X}_i, u_i, \boldsymbol{W}_i) = \Theta_i(\overline{\boldsymbol{X}}_{1:i-1}, \boldsymbol{X}_i, \overline{\boldsymbol{S}}, u_i, \boldsymbol{W}_i), \quad i \in I \ ,$$
$$\Phi_{\boldsymbol{S}}(\boldsymbol{S}) = \Theta_{\boldsymbol{S}}(\overline{\boldsymbol{X}}_{1:n}, \boldsymbol{S}) \ ,$$

with $\overline{\boldsymbol{X}}_{1:0}$ being by convention an empty vector.

We can now write an auxiliary problem for (8.10). Assume that the dynamics Θ is differentiable. In this case, Φ is differentiable and the auxiliary problem writes:

$$\begin{aligned} \min_{(\boldsymbol{X}, \boldsymbol{S}, u) \in \mathcal{X} \times \mathcal{S} \times \mathbb{U}} \ & \mathbb{E}\Bigg(\sum_{i=1}^{n} \Big(j_i(\boldsymbol{X}_i, u_i) + j^F(\overline{\boldsymbol{X}}_{1:i-1}, \boldsymbol{X}_i, \overline{\boldsymbol{X}}_{i+1:n}) \Big) \\ & \qquad + \frac{\gamma_x}{2} \left\| \boldsymbol{X} - \overline{\boldsymbol{X}} \right\|^2 + \frac{\gamma_s}{2} \left\| \boldsymbol{S} - \overline{\boldsymbol{S}} \right\|^2 + \frac{\gamma_u}{2} \left\| u - \overline{u} \right\|^2 \\ & \qquad + \Big\langle \overline{\boldsymbol{\Lambda}} \, , (\Theta'(\overline{\boldsymbol{X}}, \overline{\boldsymbol{S}}, \overline{u}, \boldsymbol{W}) - \Phi'(\overline{\boldsymbol{X}}, \overline{\boldsymbol{S}}, \overline{u}, \boldsymbol{W})) \cdot (\boldsymbol{X}, \boldsymbol{S}, u) \Big\rangle \Bigg) \\ \text{s.t. } & \Phi(\boldsymbol{X}, \boldsymbol{S}, u, \boldsymbol{W}) = 0 \ . \end{aligned} \tag{10.3}$$

By construction, the auxiliary problem (10.3) is decomposable with respect to the decompositions (10.1) and (10.2).

10.4 Relaxation of the system

The APP relies on variational techniques and requires the mappings Θ and Φ to be differentiable as the derivatives Θ' and Φ' appear in Problem (10.3). However, the dynamics Θ in Problem (8.10) involves integer variables so Θ' is not defined. To overcome this difficulty, we propose a continuous relaxation of the system with relaxed cost and dynamics that are differentiable almost everywhere. It is possible to use a differentiable relaxation of the system but this requires more implementation efforts. As we are far from the conditions of convergence of the algorithm (see Section 8.5), nothing ensures that a differentiable relaxation would give better results than the simple one that is defined below.

10.4.1 State variable relaxation

Let $i \in I$ and $t \in \mathbb{T}$. Recall from Section 8.2 that:

- $\boldsymbol{X}_{i,t} = (\boldsymbol{E}_{i,t}, \boldsymbol{A}_{i,t}, \boldsymbol{P}_{i,t})$ takes values in $\{0,1\} \times \mathbb{R} \times (\{\delta\} \cup \mathbb{R}_+)^D$,
- $\boldsymbol{S}_t$ takes values in $\mathbb{N}$.

We relax the integrity constraint on $\boldsymbol{E}_{i,t}$ and $\boldsymbol{S}_t$, so we allow:

- $\boldsymbol{X}_{i,t} = (\boldsymbol{E}_{i,t}, \boldsymbol{A}_{i,t}, \boldsymbol{P}_{i,t})$ to take values in $[0,1] \times \mathbb{R} \times (\{\delta\} \cup \mathbb{R}_+)^D$,
- $\boldsymbol{S}_t$ to take values in $\mathbb{R}$.

Remark 10.1. We lose the physical interpretation of the relaxed variables.

- If $0 < \boldsymbol{E}_{i,t} < 1$, we could think that component i is in a degraded regime where the closer $\boldsymbol{E}_{i,t}$ is to 1 the healthier it is. This interpretation is however not exact as the health of a component is only characterized by its age $\boldsymbol{A}_{i,t}$. The probability of failure of a component is indeed only a function of $\boldsymbol{A}_{i,t}$.
- A value $\boldsymbol{S}_t \in \mathbb{R}$ means that there can be a non-integer number of parts in the stock. ◊

10.4.2 Relaxation of the dynamics

The dynamics of the original system has been described in Section 8.3 and an explicit expression is given in Appendix D.1. This expression involves indicator functions $\mathbf{1}_{\mathcal{A}}$ for some set $\mathcal{A}$. The dynamics is then non-continuous. Replacing the original indicator function $\mathbf{1}_{\mathcal{A}}$ with a continuous relaxed version allows to define a relaxed dynamics for the system.

Definition 10.2. Let $\mathcal{A} \subset \mathbb{R}^p$, $x \in \mathbb{R}^p$ and $\alpha > 0$. We define a continuous relaxation $\mathbf{1}^{\alpha}_{\mathcal{A}}$ with parameter α of the indicator function $\mathbf{1}_{\mathcal{A}}$ as:

$$\mathbf{1}^{\alpha}_{\mathcal{A}}(x) = \begin{cases} 1 - 2\alpha \times \operatorname{dist}(x, \mathcal{A}) & \text{if } \operatorname{dist}(x, \mathcal{A}) \leq \dfrac{1}{2\alpha}\ , \\ 0 & \text{if } \operatorname{dist}(x, \mathcal{A}) > \dfrac{1}{2\alpha}\ , \end{cases}$$

where $\operatorname{dist}(x, \mathcal{A})$ is the Euclidean distance between x and the set $\mathcal{A}$.

Figure 10.1 illustrates the relaxation of the indicator function.

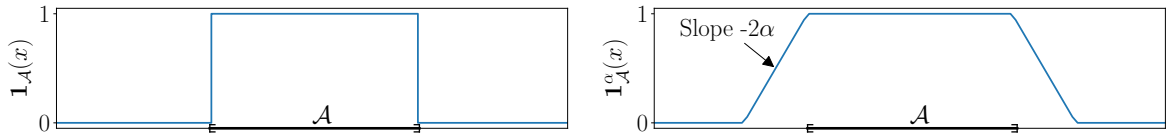


Figure 10.1: Illustration of the relaxation of the indicator function.

The parameter α quantifies the stiffness of the relaxation. As $\alpha \to +\infty$, the relaxed indicator $\mathbf{1}^{\alpha}_{\mathcal{A}}$ converges pointwise towards the original indicator function $\mathbf{1}_{\mathcal{A}}$. As $\alpha \to 0$, the relaxed indicator $\mathbf{1}^{\alpha}_{\mathcal{A}}$ converges pointwise towards the constant function 1. Note that for all $\alpha > 0$, the relaxed indicator $\mathbf{1}^{\alpha}_{\mathcal{A}}$ is continuous and differentiable almost everywhere.

A continuous relaxation of the system dynamics of parameter α is obtained by replacing the occurrences of the indicator function by its relaxed version. For instance, the relaxed dynamics of the stock is obtained from (8.6):

$$\boldsymbol{S}_{t+1} = \boldsymbol{S}_t + \sum_{i=1}^{n}\sum_{d=1}^{D} \mathbf{1}^{\alpha}_{\{D-1\}}(\boldsymbol{P}^d_{i,t}) - \min\left(\boldsymbol{S}_t,\ \sum_{i=1}^{n} \mathbf{1}^{\alpha}_{\{0\}}(\boldsymbol{E}_{i,t})\right) .$$

Additional technical details and an explicit expression for the relaxed dynamics of the components are given in Appendix D.2. The relaxed dynamics and relaxed auxiliary dynamics are denoted by Θ^α and Φ^α respectively.

10.4.3 Cost relaxation

The relaxation of the maintenance cost and forced outage cost is constructed using the same technique as for the dynamics. Let $i \in I$, $t \in \mathbb{T}$ and $\alpha > 0$ be given.

- The relaxed maintenance cost for component i at time t with parameter α is defined as:

$$\begin{cases} j^{\alpha}_{i,t}(\boldsymbol{X}_{i,t}, u_{i,t}) = \eta_t C^P_i u^2_{i,t} + \eta_t C^C_i \mathbf{1}^{\alpha}_{\{0\}}(\boldsymbol{E}_{i,t}) \mathbf{1}^{\alpha}_{\{0\}}(\boldsymbol{A}_{i,t}), & t \in \mathbb{T}_{-1} , \\ j^{\alpha}_{i,T}(\boldsymbol{X}_{i,T}) = \eta_T C^C_i \mathbf{1}^{\alpha}_{\{0\}}(\boldsymbol{E}_{i,T}) \mathbf{1}^{\alpha}_{\{0\}}(\boldsymbol{A}_{i,T}) . \end{cases}$$

- The relaxed forced outage cost at time t with parameter α is defined as:

$$j^{F,\alpha}_t(\boldsymbol{X}_{1:n,t}) = \eta_t C^F \min\left\{1,\ \sum_{i=1}^{n} \mathbf{1}^{\alpha}_{\{0\}}(\boldsymbol{E}_{i,t}) \mathbf{1}^{\alpha}_{\mathbb{R}^*_+}(\boldsymbol{A}_{i,t})\right\} .$$

Similarly, as in (8.8) and (8.9), we set:

$$j^{\alpha}_i(\boldsymbol{X}_i, u_i) = \sum_{t=0}^{T-1} j^{\alpha}_{i,t}(\boldsymbol{X}_{i,t}, u_{i,t}) + j^{\alpha}_{i,T}(\boldsymbol{X}_{i,T}) \quad \text{and} \quad j^{F,\alpha}(\boldsymbol{X}_{1:n}) = \sum_{t=0}^{T} j^{F,\alpha}_t(\boldsymbol{X}_{1:n,t}) .$$

We use the relaxed version of the dynamics Θ^α, of the auxiliary dynamics Φ^α, and of the costs j^α_i and $j^{F,\alpha}$ in the auxiliary problem (10.3). Hence, the objective function and the dynamics are continuous and differentiable almost everywhere with respect to $(\boldsymbol{X}, \boldsymbol{S}, u)$. Note that the choice of the relaxation parameter α plays an important role. We can guess that choosing a low value for α leads to a problem that may be easier to solve numerically than with a larger value of α. However, with a low value of α the relaxed dynamics and cost do not represent well the industrial problem whereas with a large α the original and relaxed dynamics are close. More details on the influence of the parameter α and how it is chosen in practice are given in Section 10.7.

10.5 Explicit expression of the subproblems

By construction, the auxiliary problem (10.3) can be decomposed into n independent subproblems on the components and a subproblem on the stock. In this section, we provide the explicit expression of these subproblems. The APP fixed-point algorithm 7 will be applied on the relaxed system, so we use the relaxed dynamics and cost in the writing of the subproblems. Formally, the gradients of Θ^α and Φ^α are not defined everywhere. By abuse of notation, the subproblems are given as if Θ^α and Φ^α were

differentiable. Appendix D.4 gives details on how we handle the points where the relaxed indicator is not differentiable.

The subproblem on component $i \in I$ is:

$$\begin{aligned}\min_{(\boldsymbol{X}_i,u_i)\in\mathcal{X}_i\times\mathbb{U}_i} \mathbb{E}\Bigg(& j_i^{\alpha}(\boldsymbol{X}_i,u_i) + j^{F,\alpha}(\overline{\boldsymbol{X}}_{1:i-1},\boldsymbol{X}_i,\overline{\boldsymbol{X}}_{i+1:n}) \\ & + \frac{\gamma_x}{2}\left\|\boldsymbol{X}_i-\overline{\boldsymbol{X}}_i\right\|^2 + \frac{\gamma_u}{2}\left\|u_i-\overline{u}_i\right\|^2 \\ & + \left\langle \overline{\boldsymbol{\Lambda}}_{\boldsymbol{S}}\,, \partial_{\boldsymbol{X}_i}\Theta_{\boldsymbol{S}}^{\alpha}(\overline{\boldsymbol{X}}_{1:n},\overline{\boldsymbol{S}})\cdot\boldsymbol{X}_i\right\rangle \\ & + \sum_{j=i+1}^{n}\left\langle \overline{\boldsymbol{\Lambda}}_j\,, \partial_{\boldsymbol{X}_i}\Theta_j^{\alpha}(\overline{\boldsymbol{X}}_{1:j},\overline{\boldsymbol{S}},\overline{u}_j,\boldsymbol{W}_j)\cdot\boldsymbol{X}_i\right\rangle\Bigg) \\ \text{s.t. } & \Phi_i^{\alpha}(\boldsymbol{X}_i,u_i,\boldsymbol{W}_i)=0\,.\end{aligned} \tag{10.4}$$

The subproblem on the stock is:

$$\begin{aligned}\min_{\boldsymbol{S}\in\mathcal{S}}\ & \mathbb{E}\left(\frac{\gamma_s}{2}\left\|\boldsymbol{S}-\overline{\boldsymbol{S}}\right\|^2 + \sum_{i=1}^{n}\left\langle\overline{\boldsymbol{\Lambda}}_i\,,\partial_{\boldsymbol{S}}\Theta_i^{\alpha}(\overline{\boldsymbol{X}}_{1:i},\overline{\boldsymbol{S}},\overline{u}_i,\boldsymbol{W}_i)\cdot\boldsymbol{S}\right\rangle\right) \\ \text{s.t. }& \Phi_{\boldsymbol{S}}^{\alpha}(\boldsymbol{S})=0\,.\end{aligned} \tag{10.5}$$

10.6 The APP fixed-point algorithm for the industrial system

We can now solve the original maintenance optimization problem (8.10) using the fixed-point algorithm 7. In this section, we give details on the practical implementation of the algorithm and present an efficient mixed parallel and sequential strategy for the resolution of the subproblems, that is tailored for the industrial optimization problem.

At each iteration, we solve the auxiliary problem (10.3). In a fully parallel version of the APP fixed-point algorithm, solving the auxiliary problem (10.3) boils down to the parallel resolution of the $n+1$ independent subproblems defined by (10.4)–(10.5).

1. The subproblems on the components (10.4) are solved with the blackbox algorithm MADS [Audet and Dennis, 2006], presented in Chapter 4. At iteration l, we solve subproblem $i \in I$ with:

$$(\overline{\boldsymbol{X}},\overline{\boldsymbol{S}},\overline{u}) = (\boldsymbol{X}^l,\boldsymbol{S}^l,u^l) \quad \text{and} \quad \overline{\boldsymbol{\Lambda}} = \boldsymbol{\Lambda}^l\,.$$

   MADS outputs a primal solution $(\boldsymbol{X}_i^{l+1},u_i^{l+1})$. The optimal multiplier $\boldsymbol{\Lambda}_i^{l+1}$ is computed afterwards using the adjoint state, in the same fashion as for the synthetic cases of Chapter 9. The full derivation of the backward recursion for the multipliers in the industrial case is carried out in Appendix D.3.

2. The subproblem on the stock (10.5) is very easy to solve numerically for an analogous reason as in Remark 9.5. At iteration l of the fully parallel APP fixed-point algorithm, we use:

$$(\overline{\boldsymbol{X}},\overline{\boldsymbol{S}},\overline{u}) = (\boldsymbol{X}^l,\boldsymbol{S}^l,u^l) \quad \text{and} \quad \overline{\boldsymbol{\Lambda}} = \boldsymbol{\Lambda}^l\,.$$

   The constraint $\Phi_{\boldsymbol{S}}^{\alpha}(\boldsymbol{S}) = 0$ represents the dynamics of the stock with $\overline{\boldsymbol{X}} = \boldsymbol{X}^l$ being fixed. The value of $\overline{\boldsymbol{X}}$ completely determines the dynamics of the

stock. Hence, solving the subproblem on the stock just boils down to simulate its dynamics, we get a primal solution $\boldsymbol{S}^{l+1}$. The optimal multiplier $\boldsymbol{\Lambda}_{\boldsymbol{S}}^{l+1}$ is also computed using the adjoint state, see Appendix D.3.

The features of the subproblem on the stock suggest to change the fully parallel strategy into a mixed parallel/sequential strategy.

1. At iteration l, the n subproblems on the components are still solved in parallel using:

$$(\overline{\boldsymbol{X}}, \overline{\boldsymbol{S}}, \overline{u}) = (\boldsymbol{X}^l, \boldsymbol{S}^l, u^l) \quad \text{and} \quad \overline{\boldsymbol{\Lambda}} = \boldsymbol{\Lambda}^l \ .$$

   This yields a solution $(\boldsymbol{X}_i^{l+1}, u_i^{l+1}, \boldsymbol{\Lambda}_i^{l+1})$ for each subproblem $i \in I$.

2. The difference arises for the subproblem on the stock. At iteration l, we immediately use the output of the subproblems on the components. This means that we set before solving the subproblem on the stock at iteration l:

$$(\overline{\boldsymbol{X}}, \overline{u}) = (\boldsymbol{X}^{l+1}, u^{l+1}) \quad \text{and} \quad (\overline{\boldsymbol{\Lambda}}_1, \dots, \overline{\boldsymbol{\Lambda}}_n) = (\boldsymbol{\Lambda}_1^{l+1}, \dots, \boldsymbol{\Lambda}_n^{l+1}) \ .$$

   This implies that the subproblem on the stock at iteration l can be solved only after all subproblems on the components have been solved.

With this strategy, we see experimentally that the number of iterations for convergence is reduced without penalizing the computation time per iteration as the subproblem on the stock can be solved in negligible time. This results in an overall speed up of the algorithm. The APP fixed-point algorithm with the mixed parallel/sequential strategy is presented in Algorithm 8. The termination criteria is a maximum number of iterations $M \in \mathbb{N}$. This is the version that is used for the numerical experiments of Sections 10.7 and 10.8.

Algorithm 8 APP fixed-point algorithm with a mixed parallel/sequential strategy

Start with $(\overline{\boldsymbol{X}}, \overline{\boldsymbol{S}}, \overline{u}) = (\boldsymbol{X}^0, \boldsymbol{S}^0, u^0)$, $\overline{\boldsymbol{\Lambda}} = \boldsymbol{\Lambda}^0$
for $l = 0, \dots, M-1$ **do:**
  for all $i \in \{1, \dots, n\}$ **do in parallel:**
    Solve the subproblem (10.4) on component i.
    Let $(\boldsymbol{X}_i^{l+1}, u_i^{l+1})$ be a solution and $\boldsymbol{\Lambda}_i^{l+1}$ be an optimal multiplier.
  end for
  Set $(\overline{\boldsymbol{X}}, \overline{u}) = ((\boldsymbol{X}_1^{l+1}, \dots, \boldsymbol{X}_n^{l+1}), (u_1^{l+1}, \dots, u_n^{l+1}))$
  Set $(\overline{\boldsymbol{\Lambda}}_1, \dots, \overline{\boldsymbol{\Lambda}}_n) = (\boldsymbol{\Lambda}_1^{l+1}, \dots, \boldsymbol{\Lambda}_n^{l+1})$
  Solve the subproblem (10.5) on the stock.
  Let $\boldsymbol{S}^{l+1}$ be a solution and $\boldsymbol{\Lambda}_{\boldsymbol{S}}^{l+1}$ be an optimal multiplier.
  Set $\overline{\boldsymbol{S}} = \boldsymbol{S}^{l+1}$ and $\overline{\boldsymbol{\Lambda}}_{\boldsymbol{S}} = \boldsymbol{\Lambda}_{\boldsymbol{S}}^{l+1}$
end for
return the maintenance strategy u^M

Remark 10.3. We could also have tried a fully sequential version of the fixed-point algorithm, where each subproblem is solved sequentially. Then, when solving subproblem i at iteration l we can use:

$$(\overline{\boldsymbol{X}}_{1:i-1}, \overline{\boldsymbol{X}}_{i:n}, \overline{u}_{1:i-1}, \overline{u}_{i:n}, \overline{\boldsymbol{S}}) = (\boldsymbol{X}_{1:i-1}^{l+1}, \boldsymbol{X}_{i:n}^{l}, u_{1:i-1}^{l+1}, u_{i:n}^{l}, \boldsymbol{S}^l)$$
$$(\overline{\boldsymbol{\Lambda}}_{1:i-1}, \overline{\boldsymbol{\Lambda}}_{i:n}, \overline{\boldsymbol{\Lambda}}_{\boldsymbol{S}}) = (\boldsymbol{\Lambda}_{1:i-1}^{l+1}, \boldsymbol{\Lambda}_{i:n}^{l}, \boldsymbol{\Lambda}_{\boldsymbol{S}}^{l})$$

Even though this strategy may converge with fewer iterations than the mixed parallel/sequential strategy, we lose all the interest of the decomposition as the subproblems must be solved in a sequential manner, resulting in an overwhelmingly large computation time. ◊

10.7 Tuning of the APP fixed-point algorithm

The practical resolution of the industrial problem with the decomposition by prediction consists in applying Algorithm 8 to Problem (8.10). However, the expectation in (8.10) cannot be evaluated exactly, so we solve a Monte-Carlo approximation of the problem with $Q = 100$ fixed failure scenarios $\omega_1, \ldots, \omega_Q \in \Omega$:

$$\begin{aligned} \min_{(\boldsymbol{X},\boldsymbol{S},u)\in\mathcal{X}\times\mathcal{S}\times\mathbb{U}} \ & \frac{1}{Q}\sum_{q=1}^{Q}\left(\sum_{i=1}^{n} j_i\left(\boldsymbol{X}_i(\omega_q), u_i\right) + j^F\left(\boldsymbol{X}_{1:n}(\omega_q)\right)\right) \\ \text{s.t. } & \Theta\left(\boldsymbol{X}(\omega_q), \boldsymbol{S}(\omega_q), u, \boldsymbol{W}(\omega_q)\right) = 0, \quad q \in \{1, \ldots, Q\} \ . \end{aligned} \tag{10.6}$$

We consider the industrial system with the characteristics given in Table 10.1, which is typical of the most demanding cases at EDF. As a maintenance strategy is parametrized by the vector $u \in \mathbb{U}$ given in (8.1), the optimization problem for the 80-component case involves $nT = 80 \times 40 = 3200$ variables. Several parameters have to be tuned in order to apply the APP fixed-point algorithm on this large-scale system. The parameters $\gamma_x, \gamma_s, \gamma_u$ appear in the auxiliary problem (10.3) and α characterizes the relaxation of the system, see Section 10.4. In this section, we study the influence of these parameters on the performance of the fixed-point algorithm, using tools for sensitivity analysis. The cobwebs, presented in §10.7.3 and the Morris method presented in §10.7.4 are normally used to assess the sensibility of a simulation code (for physical phenomena for instance) to some input parameters. Here, these tools are used in a different context, for the tuning of an optimization algorithm.

Parameter	**Value**		
Number of components n	80		
Initial number of spare parts $\boldsymbol{S}_0$	16		
Horizon T	40 years		
Time step Δt	1 year		
Number of time steps for supply D	2		
Discount rate τ	0.08		
Maintenance threshold ν	0.9		
Yearly forced outage cost C^F	10000 k€/ year		
	Comp. 1	**Comp.** 2	**Comp.** $i \geq 3$
PM cost C^P	50 k€	50 k€	50 k€
CM cost C^C	100 k€	250 k€	200 k€
Failure distribution	Weib$(2.3, 10)$	Weib$(4, 20)$	Weib$(3, 10)$
Mean time to failure	8.85 years	18.13 years	8.93 years

Table 10.1: Characteristics of the industrial system.

Remark 10.4. In Section 10.8, the performance of the APP fixed-point algorithm is compared with a *reference algorithm*, which is the blackbox algorithm MADS applied directly on Problem (10.6). The maintenance threshold ν is used both in the reference algorithm and the decomposition method, which is not the case for $\gamma_x, \gamma_s, \gamma_u$ and α that are only used in the APP fixed-point algorithm. Hence, we fix ν to a value that gives good performance with the reference algorithm and use it for the fixed-point algorithm. We do not consider changing the value of ν as it would mean that the reference algorithm changes for each different ν, making a fair comparison harder and the results less clear to analyze. ◊

10.7.1 Description of the parameters and tuning methodology

In the auxiliary problem (10.3), the value of $\gamma = (\gamma_x, \gamma_s, \gamma_u)$ influences the numerical behavior of the algorithm. We choose to increase the values of γ_x, γ_s and γ_u at each iteration of the APP fixed-point algorithm. The insight is that we can use low values of γ in the first iterations to get close to a good solution. Then, we use high values of γ to avoid oscillations of the solution of the auxiliary problem. Indeed, with a large value of γ, the solution $(\boldsymbol{X}^{l+1}, \boldsymbol{S}^{l+1}, u^{l+1})$ of the auxiliary problem at iteration $l+1$ is close to the previous solution $(\boldsymbol{X}^{l}, \boldsymbol{S}^{l}, u^{l})$. The value of γ_u evolves from iteration l to $l+1$ of the APP fixed-point algorithm with an additive step $\Delta\gamma > 0$, so that:

$$\gamma_u^{l+1} = \gamma_u^l + \Delta\gamma \ .$$

Then, γ_x and γ_s are chosen to be proportional to γ_u with ratios $r_x > 0$ and $r_s > 0$ respectively, so that:

$$\gamma_x^{l+1} = \gamma_u^{l+1}/r_x \ \text{ and } \ \gamma_s^{l+1} = \gamma_u^{l+1}/r_s \ .$$

The motivation for this choice is that the vectors $\boldsymbol{X}$, $\boldsymbol{S}$ and u need some rescaling so that their norms are of the same order of magnitude. The parameters $\gamma_u^0, r_x, r_s, \Delta\gamma$ have to be tuned.

The other parameter that requires attention is the relaxation parameter α. Similarly as for γ, we choose to increase the value of α at each iteration. A low value of α makes the problem easier to solve numerically but does not represent well the real problem. As α increases, the relaxed problem is closer and closer to the real one but becomes harder to solve. To ease the resolution, we use the solution of the auxiliary problem at iteration l as a warm start in MADS for iteration $l+1$. The value of α varies from iteration l to $l+1$ of the APP fixed-point algorithm with a step $\Delta\alpha > 0$ so that:

$$\alpha^{l+1} = \alpha^l + \Delta\alpha \ .$$

The values of α^0 and $\Delta\alpha$ have to be tuned. We denote by:

$$p = (\gamma_u^0, r_x, r_s, \Delta\gamma, \alpha^0, \Delta\alpha) \in \mathbb{R}^6 \ ,$$

the vector of parameters that have to be adjusted for the algorithm.

Tuning methodology. Choosing a good value for p is difficult in practice. We cannot afford to test the fixed-point algorithm with many different values of p directly on the 80-component case, as one run of the optimization takes around 22 hours. In order to find an appropriate value for p, we will rely on a smaller system than the

80-component case. This *small system* is designed so that the runs of the fixed-point algorithm take a reasonable amount of time, 4 hours in our case. The small system consists of 10 components with 2 spare parts initially. All the other characteristics of this small system are the same as in Table 10.1. In a way, the 10-component system can be seen as a downscaling of the 80-component case of interest.

The idea of the tuning procedure is to run the decomposition method on the 10-component system several times, but with a different value for p at each run. To do so, we start by defining bounds on the value of the parameters, they are given in Table 10.2. These bounds are chosen to be wide in order to ensure that they contain good values of the parameters.

Parameter	γ_u^0	r_x	r_s	$\Delta\gamma$	α^0	$\Delta\alpha$
Bounds	$[1, 100]$	$[1, 10^4]$	$[1, 10^3]$	$[0, 100]$	$[2, 200]$	$[0, 200]$

Table 10.2: Bounds for the parameters of the fixed-point algorithm.

Table 10.2 defines a hypercube $H \subset \mathbb{R}^6$, in which we look for a good value of p:

$$H = [1, 100] \times [1, 10^4] \times [1, 10^3] \times [0, 100] \times [2, 200] \times [0, 200] \ .$$

Then, we choose a sampling strategy to draw values of p within the hypercube H, we run the fixed-point algorithm with each of the sampled values and we analyze the impact of p on the output of the algorithm.

In the following sections, we use different sampling strategies and visualization tools in order to evaluate the influence of p and to guide the tuning of the algorithm. In each case, we use around 200 samples of p. This means that the fixed-point algorithm is executed 200 times. These 200 runs are launched in parallel and terminate in around 4 hours. Note that each run of the fixed-point algorithm consists in solving the optimization problem for the small system of 10 components and is therefore itself parallelized in 10 processes. Hence, there are two levels of parallelization and the whole procedure runs with $200 \times 10 = 2000$ processes.[1]

10.7.2 Histogram of the optimization outputs

In this paragraph, we aim at quantifying whether the influence of p on the output of the optimization algorithm is significant. Here, the sampling strategy for p is an optimized Latin Hypercube Sampling (LHS) [Damblin et al., 2013], assuming a uniform distribution of the values between the bounds given in Table 10.2. The optimized LHS strategy allows to ensure good space-filling properties of the design, meaning that the samples of p efficiently cover the hypercube H.[2]

In order to see if p impacts the output of the algorithm, we show, in Figure 10.2, a histogram of the optimal cost given by the algorithm on the 200 runs. Each run corresponds to a different value of p. For example, if we look at the leftmost bin of the histogram, there are two runs for which the fixed-point algorithm returns an optimal cost that is between 1250 and 1300 k€. Hence, we notice from Figure 10.2 that the choice of p heavily impacts the performance of the fixed-point algorithm: the output of

[1] These intensive parallel runs have been carried out on the cluster EOLE from EDF, that allows the use of up to 2688 cores per user simultaneously.

[2] We can refer to §3.2.2.2 for more details on space-filling designs.

the algorithm varies between 1255 and 2955 k€. The best performance of the algorithm is therefore 2.3 times better than its worst performance. This difference is only due to the value of p, which confirms that we must pay attention to its tuning.

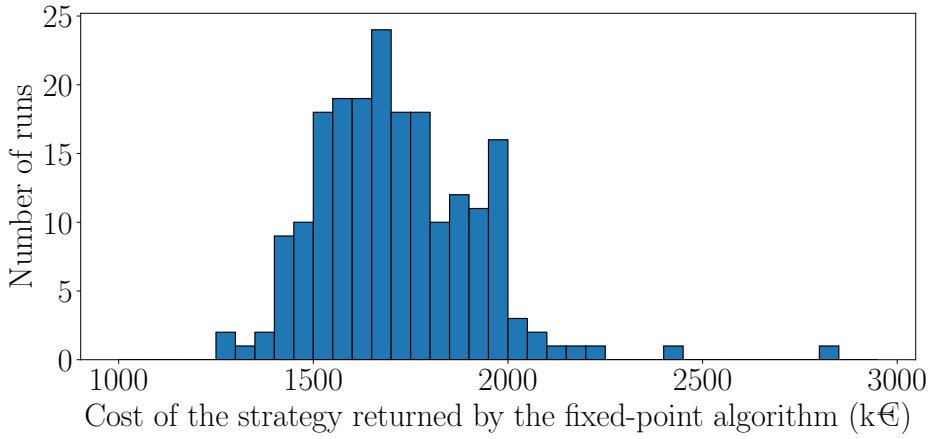


Figure 10.2: Histogram of the output of the 200 runs of the fixed-point algorithm, each run is done with a different value of p.

Hence, Figure 10.2 allows to see that p greatly influences the output of the algorithm. Now, the question is how to choose a value of p that will give a good performance on the 80-component case. To do so, we try to detect patterns linking the value of p and the performance of the algorithm. This is the object of the next section.

10.7.3 A qualitative approach with cobweb plots

The easiest strategy to tune the parameter p, using the 200 runs that have been performed in the previous section, is to simply take the value that has given the best performance. However, we have only sampled 200 values of p in the hypercube H and there may be better values than the ones that have been tested. This is why we aim at detecting some patterns in the values of p that lead to good performance of the algorithm. These patterns could then guide us towards an efficient choice of p that may be outside the set of sampled values.

For this analysis, we use cobweb plots, that give a qualitative representation of the output of the optimization given the input parameter p. The plots of this section are drawn using the same runs as in §10.7.2.

In Figure 10.3, there are 7 vertical axes representing respectively each coordinate of the vector p and the outputs of the runs of the fixed-point algorithm. On each axis, the 200 values of the sampled parameters (or of the optimal cost for the last axis) are sorted in increasing order. Each line (grey or red) represents a combination of the parameters $\gamma_u^0, r_x, r_s, \Delta\gamma, \alpha^0, \Delta\alpha$ from the sample (*i.e.* a value of p), along with its associated output. Therefore, there are 200 lines representing the 200 samples of p. Note that the visualization is only qualitative as the axes only show the ranking of the values in increasing order but not the actual numerical values. In red, we have highlighted the samples that lead to an optimal cost that is below 1450 k€. Then, we can see if some trends are emerging for good values of p. In Figure 10.3, we notice that most of the best samples have a low value for $\Delta\gamma$, *i.e.* $\Delta\gamma < 20$. We also remark that good values for r_x are mainly in the upper half of the graph, although the pattern is not as clear as for $\Delta\gamma$. For the other parameters, we can just say that extremely low

values (for r_s and α_0) or extremely high values (for $\Delta\alpha$ and γ_u^0) do no lead to a highly performing algorithm.

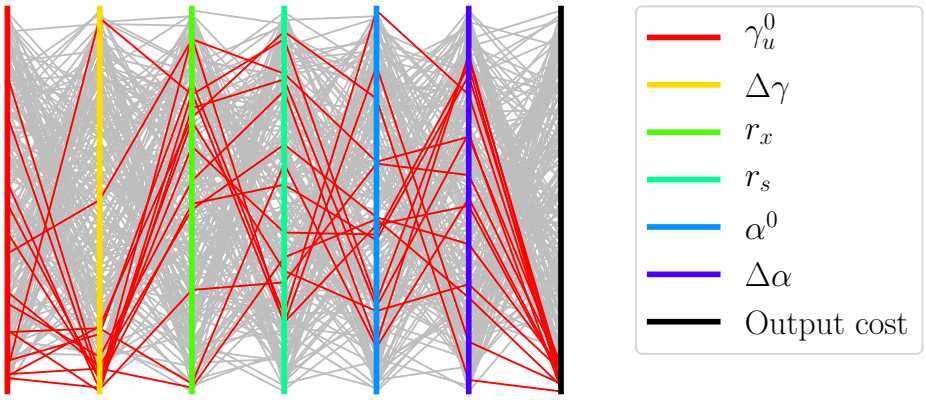


Figure 10.3: Cobweb plot highlighting the values of the parameters for which the output cost is better than 1450 k€.

It is difficult to detect a general trend from Figure 10.3 that could help us choosing a good value of p. In Figure 10.4, we highlight the worst parameter values in order to see if there are any regions of the hypercube H that should be avoided when choosing p. The red lines show the parameter values for which the output cost is worse than 1975 k€. Contrary to Figure 10.3, we see that high values of $\Delta\gamma$ are highlighted. We also notice that low values of $\Delta\alpha$ are not very efficient. Otherwise, we cannot distinguish a clear pattern in bad values of p.

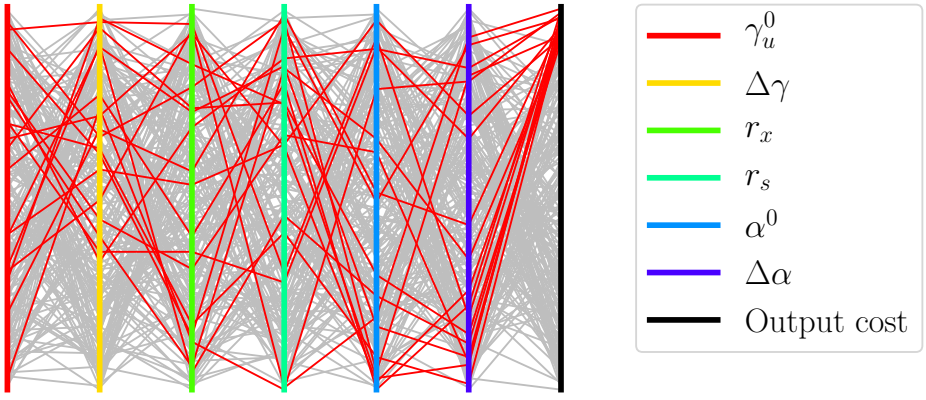


Figure 10.4: Cobweb plot highlighting the values of the parameters for which the output cost is worse than 1975 k€.

In this section, we have seen that cobweb plots are a good tool to visualize qualitative patterns of the effect of an input parameter on the output of an algorithm. In our case, although we have highlighted some regions in which $\Delta\gamma$ or r_x should be chosen, the pattern is not clear enough to design new values of p that will give good performance for sure.

10.7.4 A quantitative approach with the Morris method

The qualitative approach with cobweb plots in the previous section did not enable us to detect clear patterns for the tuning of the components of p. In this section, we use a quantitative approach, called the *Morris method* [Morris, 1991], to evaluate both the

individual influence and the interaction effects of the parameters $\gamma_u^0, r_x, r_s, \Delta\gamma, \alpha^0$ and $\Delta\alpha$ on the output of the fixed-point algorithm.

Contrary to §10.7.2 and §10.7.3, we do not use an optimized LHS to draw the values of p in this section. The Morris method is indeed based on randomized One-At-a-Time (OAT) experiments. An OAT experiment consists in doing some runs of the algorithm where the value of p used in two successive executions differs only by the value of one of its coordinate, the others being kept to a baseline value. This means that we only vary one value among $\{\gamma_u^0, r_x, r_s, \Delta\gamma, \alpha^0, \Delta\alpha\}$ between two runs.

In the Morris method, we start by mapping the hypercube H to the unit hypercube $[0,1]^6$. Then, the unit hypercube is discretized in M levels by input, hence generating a grid over $[0,1]^6$. The next step is to randomly generate N *paths* of length 7 (*i.e.* the dimension of H increased by one) along the grid.[3] The following definitions are given for the particular unit hypercube $[0,1]^6$ but hold in any finite dimension.

Definition 10.5. A *path* P is a set of samples $P = \{p^{(1)}, \dots, p^{(7)}\} \subset [0,1]^6$ where, for $i \in \{1, \dots, 6\}$, $p^{(i+1)} - p^{(i)}$ has only one non-zero coordinate, of length $\delta > 0$, where δ is a multiple of the discretization step $1/(M-1)$. We also impose that each coordinate is changed only once along the path.

In the remainder of this section, we denote by $\mathsf{A} : \mathbb{R}^6 \to \mathbb{R}$ the function such that $\mathsf{A}(p)$ is the optimal cost given by the fixed-point algorithm running with the parameter $p \in \mathbb{R}^6$.

Definition 10.6. Let $P_1, \dots, P_N$ be the N paths that are generated in the Morris method. We introduce the *elementary effect* $d_i(P_j)$ of perturbating coordinate $i \in \{1, \dots, 6\}$ on path $j \in \{1, \dots, N\}$ as:

$$d_i(P_j) = \frac{\mathsf{A}(p + \delta e_i) - \mathsf{A}(p)}{\delta} \ ,$$

where e_i is the i-th vector of the canonical Euclidean basis and $p \in P_j$ is the only element in P_j such that $p + \delta e_i \in P_j$.

Remark 10.7. The elementary effect $d_i(P_j)$ can be interpreted as an approximation on a coarse grid of the partial derivative of the function A with respect to its i-th input variable at the point $p \in P_j$ where p is such that $p + \delta e_i \in P_j$. ◊

Definition 10.8. For each coordinate $i \in \{1, \dots, 6\}$, we define two indices, the mean elementary effect μ_i and the standard deviation of the elementary effects σ_i:

$$\mu_i = \frac{1}{N} \sum_{j=1}^{N} d_i(P_j) \quad \text{and} \quad \sigma_i = \sqrt{\frac{1}{N} \sum_{j=1}^{N} \left(d_i(P_j) - \mu_i\right)^2} \ . \tag{10.7}$$

We also introduce the mean of the absolute elementary effects $\mu_i^\star$:

$$\mu_i^\star = \frac{1}{N} \sum_{j=1}^{N} |d_i(P_j)| \ .$$

[3] We can refer to [Campolongo et al., 2007] for a way of generating these paths so as to ensure a good covering of the input hypercube $[0,1]^6$.

The index μ_i quantifies the influence of the coordinate i of the parameter p on the optimal cost given by the algorithm. Indeed μ_i is the mean of the elementary effects $\{d_i(P_j)\}_{1\leq j\leq N}$ which can be seen as coarse approximations of the partial derivative of A with respect to its i-th input variable. Using the index $\mu_i^\star$ enables to avoid possible vanishing effects due to terms of opposite sign in the sum $\sum_{j=1}^N d_i(P_j)$. Hence, the higher $\mu_i^\star$, the higher the approximations of the partial derivatives of A in absolute value, meaning that the coordinate i of the parameter p strongly influences the output of the algorithm. A low value of $\mu_i^\star$ means that the coordinate i has little influence on the output and can be discarded from the tuning procedure.

The index σ_i measures the spread of the elementary effects from the mean μ_i and therefore the contribution of non-linear effects and of interactions between coordinates on the output of the algorithm. Indeed, if the influence of the i-th coordinate is linear with no interactions with others, then all $d_i(P_j)$, $1 \leq j \leq N$ take the same value, which is then equal to μ_i. In this case, we get that σ_i equals zero. A non-zero value of σ_i is then the sign of non-linear or interaction effects.

We use $N = 28$ paths of size 7, generated so as to efficiently cover the input space, using the technique of [Campolongo et al., 2007]. Thus, we perform a total of $28 \times 7 = 196$ runs of the fixed-point algorithm. Figure 10.5 shows the value of σ versus μ or $\mu^\star$ for each of the coordinates of p. Let us first focus on the absolute elementary effects (blue points). They are all in the upper right corner of the plot, that is, the region with high $\mu^\star$ and high σ. This means that all inputs are influential and with either non-linear effects and/or interactions with other coordinates. In particular, we cannot discard any coordinate from the tuning procedure and we cannot set the 6 parameters independently of others because of interaction effects. Let us now explain the interest in looking at the mean elementary effects μ. We see that μ is much smaller than $\mu^\star$ for all the coordinates. For a given coordinate i, this means that in the sum defining μ_i (10.7) there are terms of opposite signs, therefore the approximate partial derivative $d_i(P_j)$ can be positive or negative depending on the path P_j. Then – taking the example of α_0 – we deduce that increasing α^0 could either improve or degrade the performance of the algorithm depending on the point p at which we vary α^0. The same conclusion is true for the 6 parameters. Hence, we cannot exhibit a pattern for choosing a good value of p for sure.

10.7.5 Conclusion of the tuning

The study performed in this section shows that the fixed-point algorithm is sensitive to the value of p that parametrizes the auxiliary problem and the relaxation of the system. As we are not able to design a rule for a good choice of the parameter p, we simply use, for the 80-component case, the parametrization that gives the best results on the 10-component case. The corresponding value of p was obtained in the LHS used in §10.7.2 and §10.7.3:

$$p = (17.32,\ 7434,\ 815.3,\ 1.360 \times 10^{-1},\ 46.51,\ 135.5)\ . \tag{10.8}$$

We give the value of p with 4 significant digits to emphasize that the performance of the APP fixed-point algorithm is very sensitive to this value.

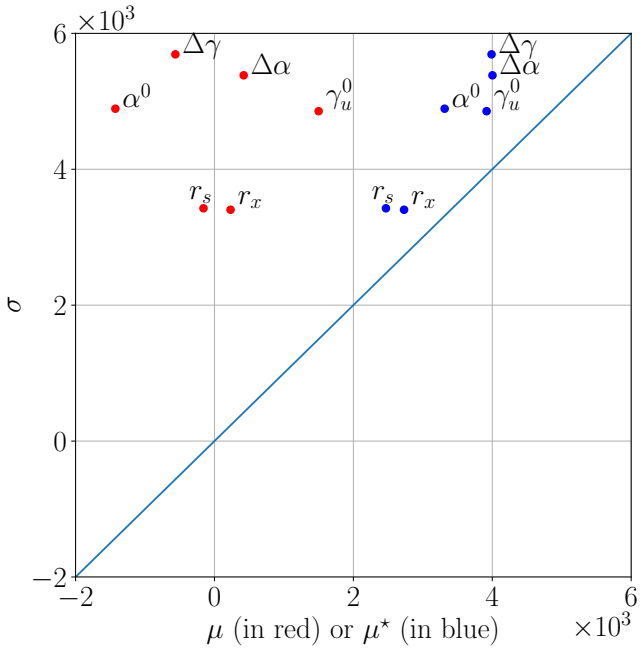


Figure 10.5: Standard deviation of the elementary effects with respect to the mean of the elementary effects in red (respectively absolute elementary effects in blue).

10.8 Numerical results on the 80-component case

Now that all the parameters of the fixed-point algorithm 8 are set, we can solve Problem (10.6). The performance of the fixed-point algorithm is compared with a reference algorithm, which is the blackbox algorithm MADS applied directly on Problem (10.6). We consider the system of 80 components described in Table 10.1. Parameters of the computation are given in Table 10.3. When running the optimization, the reference algorithm uses the original dynamics as it does not use gradient information. The APP fixed-point algorithm runs with the value of p in (10.8) that parametrizes the auxiliary problem and the relaxation of the system.

	Decomposition	MADS
Fixed-point iterations	50	/
Cost function calls	10^3/subproblem/iteration	8×10^5
Cost and dynamics	Relaxed	Original
Processor model	Intel® Xeon® Processor E5-2680 v4, 2.4 GHz	
Computation time	18h24min	22h30min

Table 10.3: Parameters of the computation for the two algorithms.

Remark 10.9. The APP fixed-point algorithm solves a decomposable auxiliary problem at each iteration, this algorithm is designed to be parallelized. It runs on 80 processors so that the subproblems on the components are solved in parallel. The reference algorithm MADS runs only on one processor. Note that it is also possible to parallelize MADS [Audet et al., 2008b], although the implementation is not as straightforward as for the decomposition method. The parallel version of MADS has not been tested in this thesis. ♢

The maintenance strategies returned by the two algorithms are then evaluated on a common set of 10^5 failure scenarios, distinct from those used for the optimization. For the two strategies, the evaluation is done with the original dynamics of the system in order to ensure a fair comparison. The two algorithms return a maintenance strategy with $u_{i,t} \in [0,1]$ for $(i,t) \in I \times \mathbb{T}$. From the operational perspective, PMs make the components as good as new. Hence, for the evaluation of the strategy, the controls are projected on $\{0,1\}$: we consider that if $u_{i,t} \geq \nu$, then the PM makes the component as good as new, otherwise no PM is performed. The comparison between the two maintenance strategies is fair as we use the same procedure for their evaluation.

The mean cost is 12902 k€ with MADS and 11483 k€ with the decomposition which represents a gain of 11%. The values of some quantiles are gathered in Table 10.4 and the distribution of the cost is represented on Figure 10.6.

	1%	5%	25%	50%	75%	95%	99%
Decomposition	10385	10683	11136	11472	11809	12326	12713
MADS	11858	12151	12588	12894	13211	13674	14010

Table 10.4: Quantiles of the cost of the two maintenance strategies (k€).

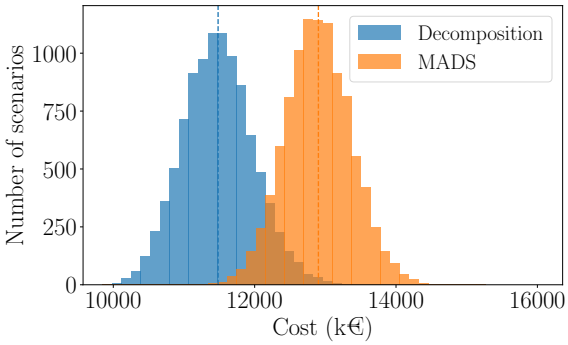


Figure 10.6: Distribution of the cost for the two maintenance strategies.

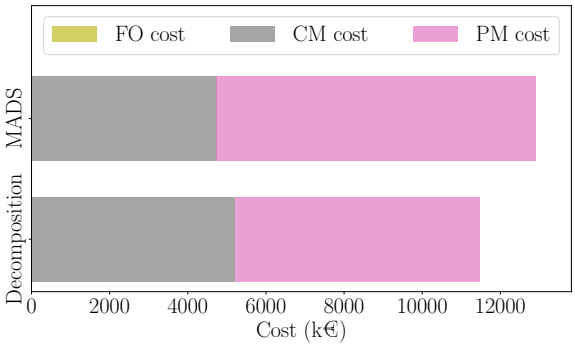


Figure 10.7: Part of the PM, CM and forced outage cost in the total expected cost.

Figure 10.7 outlines that the average CM cost is higher with the decomposition strategy. However, a much lower PM cost makes the decomposition more efficient than MADS. This is due to the fact that fewer PMs are performed with the decomposition strategy than with MADS strategy (Table 10.5). The counterpart is that failures and forced outages occur more often with the decomposition strategy (Table 10.5). The forced outage cost is not visible on Figure 10.7 as it represents 0.05 k€ for MADS strategy (3.9×10^{-6} of the total expected cost) and 4.09 k€ for the decomposition strategy (3.5×10^{-4} of the total expected cost). There are more forced outages with the decomposition strategy (63 occurrences in 10^5 failure scenarios versus 1 for MADS) but they almost all occur in the last two time steps of the study horizon. Therefore, the cost of forced outages is low because of the discount factor.

The cumulative number of PMs can be visualized on Figure 10.8. As already noticed, there are fewer PMs with the decomposition strategy. A striking feature with the decomposition strategy is that there are almost no PM in the first three years. This exploits the fact that the components are new. The reference algorithm MADS applied directly on the original problem does not detect this feature. In fact, the region of the space corresponding to not doing any PM in the first three years jointly for all

	Decomposition	MADS
Total number of PMs	447	558
Mean number of PMs/component	5.6	7.0
Mean time between PMs	6.1 years	5.0 years
Mean number of failures/component	1.40	1.18
Number of forced outages/Number of scenarios	63/10000	1/10000

Table 10.5: Overview of the number of PMs, failures and forced outages for each strategy.

components is a very small subset of the admissible space of the original problem and is not explored by MADS. On the other hand, the subproblems in the APP fixed-point algorithm act on an individual component, it is then easier to figure out that doing no PM in the first three years is profitable.

There is also a significant reduction of the number of PMs in the last five years of the study horizon. It is indeed useless to invest money to repair a component for the last few years, as the payback period could exceed the remaining life of the plant. Moreover, the discount factor at the end of the horizon greatly reduces the incurred cost so that a forced outage is not too penalizing. This is why some forced outages occur with the decomposition strategy at the end of the study period.

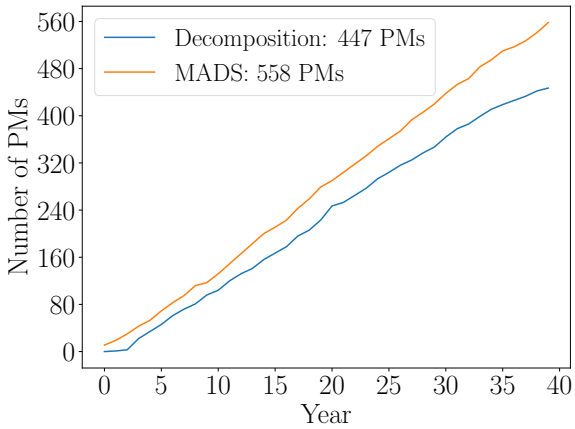


Figure 10.8: Cumulative number of PMs.

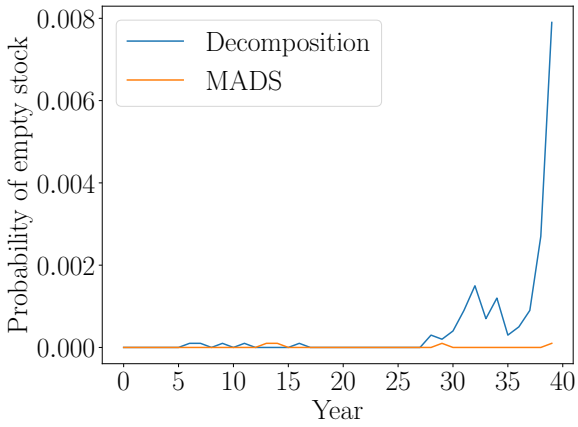


Figure 10.9: Evolution of the probability of having an empty stock.

Another indicator that is monitored by decision makers is the level of stock. A necessary condition for the occurrence of a forced outage is that the stock is empty. Hence, we look at the probability of having an empty stock. The higher this probability, the higher the probability of forced outage. The probability of having an empty stock is very low for both strategies in the first 30 years and then increases for the decomposition strategy (Figure 10.9). Again, because of the discount factor, forced outages in the last few years do not have important financial consequences. It is then more profitable to do fewer PMs and allow for a higher risk of failure. This is what the decomposition strategy does.

Overall, the strategy returned by the decomposition by prediction is more cost effective than MADS strategy. For a decision maker, the decomposition strategy requires less investment as we do fewer PMs. It also has the best expected cost. Even in the case of extreme events, it is more robust than MADS strategy, as shown by the 99% quantile in Table 10.4. The forced outages indeed only occur at the end of the horizon.

10.9 Conclusion

We have applied the APP to the industrial maintenance optimization problem described in Chapter 8. As the problem involves integer variables, we have introduced a continuous relaxation of the system so that the gradients that appear in the auxiliary problem are correctly defined. The fixed-point algorithm runs on the relaxed system and consists in the iterative resolution of independent subproblems involving only a single component or the stock. Hence, the decomposition allows to overcome the curse of dimensionality, that has been encountered in Part I. We have implemented the fixed-point algorithm with an efficient mixed parallel and sequential strategy. This implementation is particularly adapted to the structure of the industrial problem, for which the subproblem on the stock is easy to solve numerically. The numerical performance of the algorithm is sensitive to the tuning of some parameters that appear in the auxiliary problem or control the relaxation of the system. A detailed study, using sensitivity analysis tools, has been performed, but no clear pattern for a good choice of the parameters has been detected. The selected method for the tuning consists in choosing the parameters that lead to the best performance on a low-dimensional test case.

We have presented a numerical application of the APP on an industrial case with 80 components, which is representative of the most challenging cases for EDF. The decomposition method outperforms the blackbox algorithm MADS applied directly on the full problem. The strategy returned by the decomposition involves fewer PMs especially at the beginning and the end of the time horizon, hence considerably reducing the investment. More forced outages occur but only at the end of the time horizon so without heavy financial consequences. The decomposition methodology manages to exploit the fact that the discount factor makes the forced outages not too penalizing at the end of the time horizon and that there is no need to replace new components. MADS applied on the full problem cannot detect these features because the search space is too large. The strategy returned by the decomposition is also robust to extreme events as the 99% quantile is better than for MADS.

This work proves the interest of the modeling effort needed to apply the decomposition method. Some challenges still remain for an application in an operational context. Here, the dynamics is simulated with a time step of one year. A smaller time step must be used for an accurate evaluation of the costs. This will not increase the complexity of the problem as maintenance decisions are always made on a yearly basis, so that the space of admissible maintenance strategies is still of the same dimension. However, the time needed for the evaluation of the cost function will increase. It is also possible to model more complex systems, by adding a control on the time of the order of spare parts or dependence between the failures of the components for instance. We could also consider imperfect preventive maintenance. A balance must be found between the simplicity of the model and its adequation to reality given the industrial application in mind.

Part III

The stochastic APP in Banach spaces: measurability and convergence

From Sample Average Approximation to Stochastic Approximation

In this part, we are concerned with a stochastic optimization problem of the form:

$$\min_{u \in U^{\mathrm{ad}}} J(u) \quad \text{with} \quad J(u) = \mathbb{E}\left(j(u, \boldsymbol{W})\right) , \tag{10.9}$$

where U^{ad} is a non-empty closed convex subset of a Banach space $\mathbb{U}$, $\boldsymbol{W}$ is a random variable defined on a probability space $(\Omega, \mathcal{A}, \mathbb{P})$, taking values in a measurable space $\mathbb{W}$ and $j : U^{\mathrm{ad}} \times \mathbb{W} \to \mathbb{R}$ is a function. We also assume that J is differentiable.

The main difficulty to solve Problem (10.9) lies in the computation of $\nabla J(u)$ that involves an expectation and is often intractable numerically. Two approaches, based on Monte-Carlo sampling techniques, have been developed to overcome this difficulty, namely Stochastic Approximation (SA) and Sample Average Approximation (SAA). These methods have already been quickly mentioned in the introductory chapter of Part II, and SAA has been chosen for the resolution of the industrial maintenance optimization problem. In this part, we come back to the SA technique and its translation within the APP. Let us first recall the insight of each approach.

The SAA approach is described in details in [Shapiro et al., 2009, Chapter 5]. The idea is to generate a Q-sample $w_1, \ldots, w_Q$ of the random variable $\boldsymbol{W}$ and to approximate Problem (10.9) with the sample average problem:

$$\min_{u \in U^{\mathrm{ad}}} \frac{1}{Q} \sum_{q=1}^{Q} j(u, w_q) . \tag{10.10}$$

The approximate problem (10.10) can be solved efficiently with an appropriate deterministic algorithm. The main drawback of SAA is that the sample size Q must be fixed before solving the approximate problem. With a low value of Q, the solution of (10.10) may be an inaccurate approximation of the solution of the true problem (10.9), whereas a large value of Q leads to a high computation time for the gradient of the cost function of (10.10).

On the other hand, a SA algorithm, first introduced in [Robbins and Monro, 1951], is defined by a recursive stochastic update rule. For $l \in \mathbb{N}$, the l-th iterate of a SA algorithm is a mapping $\boldsymbol{U}_l : \Omega \to \mathbb{U}$, where the range of $\boldsymbol{U}_l$ is included in U^{ad}. The most basic SA scheme is the stochastic gradient descent algorithm. Assume that $\mathbb{U}$ is a Hilbert space, that $U^{\mathrm{ad}} = \mathbb{U}$ and that j is differentiable with respect to u, then the l-th iteration of stochastic gradient descent computes an iterate u_{l+1} such that:

$$u_{l+1} = u_l - \varepsilon_l \nabla_u j(u_l, w_{l+1}) ,$$

where $\varepsilon_l > 0$ is a positive real and w_{l+1} is a realization of the random variable $\boldsymbol{W}$. The insight of the SA method is to use an approximation of ∇J by sequentially incorporating samples of $\boldsymbol{W}$. Therefore, contrary to SAA, there is no need to choose a sample size *a priori*. Moreover, the gradient $\nabla_u j(u_l, w_{l+1})$ does not involve an expectation and can be computed easily numerically.

A comparison between SA and SAA is carried out in [Nemirovski et al., 2009]. The authors show that robust versions of SA algorithms perform similarly as the SAA method and argue that SA may constitute a viable alternative to SAA.

In Part II, we have studied the APP, a unifying framework for decomposition-coordination methods, but also for classical algorithms such as gradient descent, the proximal gradient algorithm or the Newton method [Carpentier and Cohen, 2017, Section 3.3]. The APP has been used in the deterministic case as the optimization problems of Part II are solved with the SAA approach. In this part, we focus on SA schemes. We carry out a theoretical study of the measurability and the convergence of the iterates of the stochastic APP introduced by [Culioli and Cohen, 1990]. The stochastic APP framework encompasses in particular the robust SA methods of [Nemirovski et al., 2009].

Communication. The work presented in Chapter 11 is the subject of the preprint [Bittar et al., 2021].

11 Measurability and convergence of the stochastic APP

Creativity is the ability to introduce order into the randomness of nature.

Eric Hoffer

Contents

11.1 Introduction

Let $\mathbb{U}$ be a Banach space with a norm $\|\cdot\|$, $(\Omega, \mathcal{A}, \mathbb{P})$ be a probability space and $(\mathbb{W}, \mathcal{B}(\mathbb{W}))$ be a measurable topological vector space with $\mathcal{B}(\mathbb{W})$ being the Borel σ-field on $\mathbb{W}$. We refer to [Bauschke and Combettes, 2011, Billingsley, 1995] for the definitions of basic concepts in analysis and probability theory. We consider a stochastic optimization problem of the form:

$$\min_{u \in U^{\mathrm{ad}}} \left\{ J(u) := J^{\Delta}(u) + J^{\Sigma}(u) \right\} \text{ where } \begin{cases} J^{\Delta}(u) = \mathbb{E}\left(j^{\Delta}(u, \boldsymbol{W}) \right) , \\ J^{\Sigma}(u) = \mathbb{E}\left(j^{\Sigma}(u, \boldsymbol{W}) \right) . \end{cases} \tag{11.1}$$

where $U^{\mathrm{ad}} \subset \mathbb{U}$ is a non-empty closed convex set, $\boldsymbol{W} : \Omega \to \mathbb{W}$ is a random variable, $j^{\Sigma}(\cdot, w)$ and $j^{\Delta}(\cdot, w)$ are proper, convex, lower-semicontinuous (l.s.c.) real-valued functions for all $w \in \mathbb{W}$.

Stochastic Approximation (SA) algorithms are the workhorse for solving Problem (11.1). The SA technique has been introduced in [Kiefer and Wolfowitz, 1952, Robbins and Monro, 1951] as an iterative method to find the root of a monotone function which is known only through noisy estimates. SA algorithms have been the subject of many theoretical studies [Bach and Moulines, 2011, Karimi et al., 2019, Nemirovski et al., 2009, Polyak and Juditsky, 1992] and have applications in various disciplines such as machine learning, signal processing or stochastic optimal control [Benveniste et al., 2012, Kushner and Yin, 1997]. Back in 1990, with decomposition applications in mind, [Culioli and Cohen, 1990] proposed a general SA scheme in an infinite dimensional Hilbert space based on the so-called Auxiliary Problem Principle (APP), called the stochastic APP algorithm. This algorithm also encompasses several well-known algorithms such as stochastic gradient descent, the stochastic proximal gradient algorithm or stochastic mirror descent. Recently, [Geiersbach and Pflug, 2019, Martin et al., 2019] apply SA methods to solve PDE-constrained optimization problems. In this chapter, we extend the stochastic APP algorithm to the Banach case.

We study the measurability of the iterates of the stochastic APP algorithm, that is, we prove that $\boldsymbol{U}_l$ is a random variable. The issue of measurability is not often addressed in the literature, yet it is essential from a theoretical point of view. When convergence results or efficiency estimates are derived for SA algorithms, the iterates must be random variables so that the probabilities or the expectations that appear in the computation are well-defined.

We denote by $\langle \cdot \, , \cdot \rangle$ the duality pairing between $\mathbb{U}$ and the topological dual space $\mathbb{U}^{\star}$. In the case where j^{Δ} is differentiable, the l-th iteration of the stochastic APP algorithm computes a minimizer u_{l+1} such that:

$$u_{l+1} \in \operatorname*{arg\,min}_{u \in U^{\mathrm{ad}}} K(u) + \left\langle \varepsilon_l \nabla_u j^{\Delta}(u_l, w_{l+1}) - \nabla K(u_l) \, , u \right\rangle + \varepsilon_l j^{\Sigma}(u, w_{l+1}) \, , \tag{11.2}$$

where $\varepsilon_l > 0$ is a positive real, w_{l+1} is a realization of the random variable $\boldsymbol{W}$ and K is a user-defined Gateaux-differentiable convex function. The role of the function K is made clear in Section 11.2. In the context of the APP, Problem (11.2) is called the *auxiliary problem* and the function K is called the *auxiliary function*. Let us now briefly expose how this scheme reduces to well-known algorithms for particular values of K and j^{Σ}.

Assume that $\mathbb{U}$ is a Hilbert space, $U^{\mathrm{ad}} = \mathbb{U}$ and $j^{\Sigma} = 0$. The l-th iteration of

stochastic gradient descent is given by:

$$u_{l+1} = u_l - \varepsilon_l \nabla_u j^{\Delta}(u_l, w_{l+1}) \; . \tag{11.3}$$

This is exactly the stochastic APP algorithm (11.2) with $j^{\Sigma} = 0$ and $K = \frac{1}{2}\|\cdot\|^2$ where $\|\cdot\|$ is the norm induced by the inner product in $\mathbb{U}$.

When j^{Δ} is differentiable and j^{Σ} is non-smooth but with a proximal operator that is easy to compute, proximal methods [Atchade et al., 2017, Parikh and Boyd, 2014] are particularly efficient, even in a high-dimensional Hilbert space $\mathbb{U}$. An iteration of the stochastic proximal gradient algorithm is:

$$u_{l+1} \in \arg\min_{u \in \mathbb{U}} \frac{1}{2\varepsilon_l} \|u_l - u\|^2 + \left\langle \nabla_u j^{\Delta}(u_l, w_{l+1}) \, , u - u_l \right\rangle + j^{\Sigma}(u, w_{l+1}) \; . \tag{11.4}$$

This is again the stochastic APP algorithm with $K = \frac{1}{2}\|\cdot\|^2$ but with a non zero function j^{Σ}. The proximal term $\frac{1}{2\varepsilon_l}\|u_l - u\|^2$ forces the next iterate u_{l+1} to be close to u_l with respect to the norm $\|\cdot\|$. When j^{Σ} is the indicator of a convex set, the stochastic proximal gradient method reduces to stochastic projected gradient descent and when $j^{\Sigma} = 0$, this is just the regular stochastic gradient descent (11.3). Proximal methods are well-suited for regularized regression problems in machine learning for example.

When $\mathbb{U}$ is only a Banach space and not a Hilbert space, Equation (11.3) does not make sense as $u_l \in \mathbb{U}$ while $\nabla_u j^{\Delta}(u_l, w_{l+1}) \in \mathbb{U}^{\star}$ the topological dual of $\mathbb{U}$, thus the minus operation is not defined. This difficulty is addressed with the mirror descent algorithm [Nemirovski and Yudin, 1983]. The original insight of the method is to map the iterate u_l to $\nabla K(u_l) \in \mathbb{U}^{\star}$, where K is a user-defined Gateaux-differentiable function. Then, we do a gradient step in $\mathbb{U}^{\star}$ and we map back the resulting point to the primal space $\mathbb{U}$. The function K is called the *mirror map* in this setting [Bubeck, 2015]. There is also a proximal interpretation of mirror descent: instead of defining the proximity with the norm $\|\cdot\|$, the mirror descent algorithm and its stochastic counterpart [Nemirovski et al., 2009] use a Bregman divergence [Bregman, 1967] that captures the geometric properties of the problem:

$$u_{l+1} \in \arg\min_{u \in U^{\mathrm{ad}}} \frac{1}{\varepsilon_l} D_K(u, u_l) + \left\langle \nabla_u j^{\Delta}(u_l, w_{l+1}) \, , u - u_l \right\rangle \; , \tag{11.5}$$

where D_K is the Bregman divergence associated with K:

$$D_K(u, u') = K(u) - K(u') - \langle \nabla K(u') \, , u - u' \rangle \, , \quad u, u' \in \mathbb{U} \, ,$$

The function K is sometimes called the *distance-generating function* as it defines the proximity between u and u'. With $K = \frac{1}{2}\|\cdot\|^2$, we get back to the setting of stochastic gradient descent. The mirror descent algorithm is particularly suited to the case where $\nabla_u j^{\Delta}$ has a Lipschitz constant which is large with respect to the norm $\|\cdot\|$ but small with respect to some other norm that is better suited to the geometry of the problem [Nemirovski et al., 2009]. For example, in the finite-dimensional case, the performance of stochastic gradient descent depends on the Lipschitz constant of $\nabla_u j^{\Delta}$ in the Euclidean geometry. Hence, if the problem exhibits a non-Euclidean geometric structure, stochastic mirror descent may be more efficient. Note that stochastic mirror descent corresponds to the stochastic APP with a general function K and $j^{\Sigma} = 0$.

In fact, the stochastic APP algorithm combines the ideas of mirror descent and of the proximal gradient method. The iteration defined by (11.2) can be equivalently

written as:

$$u_{l+1} \in \operatorname*{arg\,min}_{u \in U^{\mathrm{ad}}} \frac{1}{\varepsilon_l} D_K(u, u_l) + \left\langle \nabla_u j^{\Delta}(u_l, w_{l+1}) \, , u - u_l \right\rangle + j^{\Sigma}(u, w_{l+1}) \, ,$$

In the sequel, we stick to the formulation (11.2) and we consider a more general version as j^{Δ} is only assumed to be subdifferentiable and we allow for an additive error on the subgradient $\partial_u j^{\Delta}(u_l, w_{l+1})$. The complete setting and description of the algorithm is given in §11.2.1. Some applications of the stochastic APP algorithm are given in §11.2.2.

Figure 11.1 summarizes the relationship between the four stochastic approximation algorithms that we have introduced.

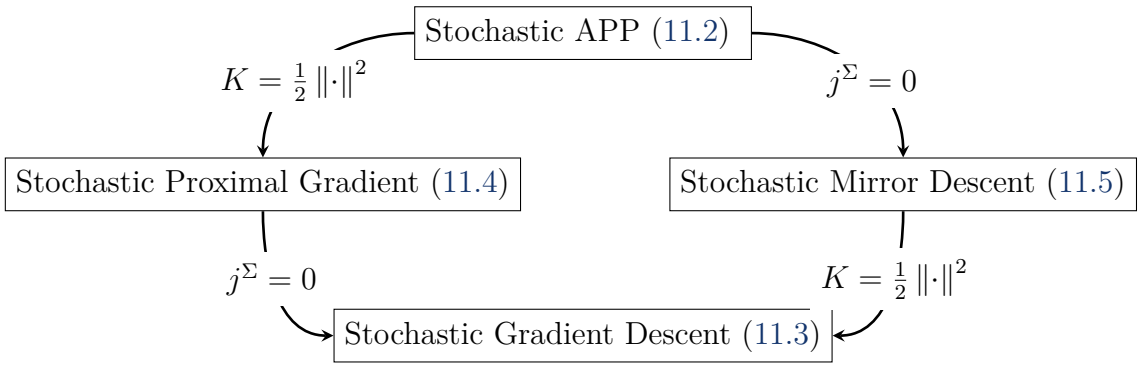


Figure 11.1: Links between various stochastic approximation algorithms.

Contributions. The main contributions of this chapter are the following:

- In Section 11.3, we prove the measurability of the iterates of the stochastic APP algorithm in a Banach space. For that purpose, we carry out a precise study based on [Castaing and Valadier, 1977, Hess, 1995] and we adapt some results of [Rockafellar and Wets, 2004] to the Banach case.
- In §11.4.1, convergence results for the iterates and for the function values of the stochastic APP algorithm are extended to the Banach case. These results already appear in [Culioli and Cohen, 1990] for the Hilbert case. They are also given, again in the Hilbert case, for stochastic mirror descent in [Nemirovski et al., 2009] or stochastic projected gradient in [Geiersbach and Pflug, 2019].
- In §11.4.2, we derive efficiency estimates for the function values taken either for the averaged sequence of iterates or for the last iterate. These efficiency estimates take into account the additive error on the subgradient, using the technique from [Geiersbach and Wollner, 2019]. To obtain convergence rates for the function values of the last iterate, we adapt the concept of modified Fejér monotonicity [Lin et al., 2018] to the framework of the stochastic APP algorithm.

11.2 Description of the algorithm and examples

In this section, we describe the version of the stochastic APP algorithm that is studied in this chapter and we give some examples of problems that fit in the general framework of Problem (11.1).

11.2.1 Setting of the stochastic APP algorithm

We aim at solving Problem (11.1) that we call the *master problem*. The original idea of the APP, first introduced in [Cohen, 1978] and extended to the stochastic case in [Culioli and Cohen, 1990], is to solve a sequence of auxiliary problems whose solutions converge to the optimal solution of the master problem.

Assume that j^{Δ} is subdifferentiable. At iteration l of the algorithm, a realization w_{l+1} of a random variable $\boldsymbol{W}_{l+1}$ is drawn. The random variables $\boldsymbol{W}_1, \dots, \boldsymbol{W}_{l+1}$ are independent and identically distributed as $\boldsymbol{W}$. Then, the following auxiliary problem is solved:

$$\min_{u \in U^{\mathrm{ad}}} K(u) + \langle \varepsilon_l(g_l + r_l) - \nabla K(u_l)\,, u \rangle + \varepsilon_l j^{\Sigma}(u, w_{l+1})\,, \tag{11.6}$$

where $g_l \in \partial_u j^{\Delta}(u_l, w_{l+1})$ and we allow for an additive error r_l on the gradient. The term r_l represents a numerical error or a bias due to an approximation of the gradient e.g. with a finite difference scheme. The auxiliary problem is characterized by the choice of the auxiliary function K. In the introduction, we have given particular choices for K that lead to well-known algorithms. Depending on the context, the function K allows for an adaptation of the algorithm to the geometric structure of the data or it can provide decomposition properties to the algorithm, see Example 11.2. The stochastic APP algorithm is given in Algorithm 9.

Algorithm 9 Stochastic APP algorithm

1: Choose an initial point $u_0 \in U^{\mathrm{ad}}$, and a positive sequence $\{\varepsilon_l\}_{l \in \mathbb{N}}$.
2: At iteration l, draw a realization w_{l+1} of the random variable $\boldsymbol{W}_{l+1}$.
3: Solve Problem (11.6), denote by u_{l+1} the solution.
4: $l \leftarrow l + 1$ and go back to 2.

Note that no explicit stopping rule is provided in Algorithm 9. It is indeed difficult to know when to stop a stochastic algorithm as its properties are of statistical nature. Nevertheless, stopping rules have been developed in [Wada and Fujisaki, 2015, Yin, 1990] for the Robbins-Monro algorithm. In practice, the stopping criterion may be a maximal number of evaluations imposed by a budget limitation.

11.2.2 Some cases of interest for the stochastic APP

The structure of Problem (11.1) is very general and covers a wide class of problems that arise in machine learning or stochastic optimal control. We give some cases of interest that can be cast in this framework.

Example 11.1. *Regularized risk minimization in machine learning*

Let $(\mathbb{X}, \mathcal{X})$ and $(\mathbb{Y}, \mathcal{Y})$ be two measurable spaces, where $\mathcal{X}$ and $\mathcal{Y}$ denote respectively the σ-fields on $\mathbb{X}$ and $\mathbb{Y}$. Let $X \subset \mathbb{X}$ and $Y \subset \mathbb{Y}$ and assume there is a probability distribution ν on $X \times Y$. Suppose that we have a training set $\{(x_i, y_i)\}_{1 \le i \le N} \in (X \times Y)^N$ which consists in independent and identically distributed samples of a random vector $(\boldsymbol{X}, \boldsymbol{Y})$ following the distribution ν. Consider a convex loss function $\ell : Y \times Y \to \mathbb{R}_+$ and let $\mathbb{U}$ be a space of functions from X to Y. The goal of regularized expected loss minimization is to find a regression function $u^{\sharp} \in U^{\mathrm{ad}}$, where $U^{\mathrm{ad}} \subset \mathbb{U}$, such that:

$$u^{\sharp} \in \operatorname*{arg\,min}_{u \in U^{\mathrm{ad}}} \int_{X \times Y} \ell\Big(y, u(x)\Big) \nu(\mathrm{d}x, \mathrm{d}y) + R(u)\,, \tag{11.7}$$

where R is a regularization term. In practice, as the distribution ν is unknown, we solve an approximate problem, called the regularized empirical risk minimization problem:

$$u^\sharp \in \arg\min_{u \in U^{\mathrm{ad}}} \frac{1}{N} \sum_{i=1}^{N} \ell\Big(y_i, u(x_i)\Big) + R(u) \;, \tag{11.8}$$

Note that Problem (11.8) is in fact exactly of the form of Problem (11.7) if the distribution ν is taken to be the empirical measure $\nu = 1/N \sum_{i=1}^{N} \delta_{(x_i,y_i)}$, where $\delta_{(x_i,y_i)}$ denotes the measure of mass one at (x_i, y_i) and zero elsewhere.

The regularized expected loss minimization Problem (11.7) is of the form of Problem (11.1) with the smooth term $J^\Delta(u) = \int_{X\times Y} \ell\Big(y, u(x)\Big)\nu(\mathrm{d}x, \mathrm{d}y)$ and the possibly non-smooth term $J^\Sigma(u) = R(u)$. △

Example 11.2. *Decomposition aspects of the stochastic APP algorithm*

Let $N > 0$ be a given positive integer. Suppose that $\mathbb{U} = \mathbb{U}_1 \times \ldots \times \mathbb{U}_N$ and $U^{\mathrm{ad}} = U_1^{\mathrm{ad}} \times \ldots \times U_N^{\mathrm{ad}}$ with $U_i^{\mathrm{ad}} \subset \mathbb{U}_i$ for all $i \in \{1, \cdots, N\}$. Moreover, assume that j^Σ is an additive function, that is, $j^\Sigma(u, \boldsymbol{W}) = \sum_{i=1}^{N} j_i^\Sigma(u^i, \boldsymbol{W})$ with $u^i \in \mathbb{U}_i$, whereas j^Δ induces a non-additive coupling. In this case, Problem (11.1) is:

$$\min_{u \in U^{\mathrm{ad}}} J^\Delta(u) + \sum_{i=1}^{N} J_i^\Sigma(u^i) \;.$$

where $J_i^\Sigma(u^i) = \mathbb{E}\left(j_i^\Sigma(u^i, \boldsymbol{W})\right)$. We apply the stochastic APP algorithm with an additive auxiliary function $K(u) = \sum_{i=1}^{N} K_i(u^i)$. Let $\bar{u} \in \mathbb{U}$ be given, a canonical choice for K_i is:

$$K_i(u^i) = J^\Delta\left(\bar{u}^{1:i-1}, u^i, \bar{u}^{i+1:N}\right), \quad i \in \{1, \ldots, N\}$$

where $\bar{u}^{i:j} = (\bar{u}^i, \ldots, \bar{u}^j)$ for $1 \leq i \leq j \leq N$ and $\bar{u}^{1:0}$ denotes the empty vector by convention. Another classical choice is $K = \frac{1}{2}\|\cdot\|^2$. With an additive function K, the auxiliary problem (11.6) can be split into N independent subproblems that can be solved in parallel. At iteration l of the stochastic APP algorithm, the i-th subproblem is:

$$\min_{u^i \in U_i^{\mathrm{ad}}} K_i(u^i) + \left\langle \varepsilon_l(g_l^i + r_l^i) - \nabla K_i(u_l^i) \,, u^i \right\rangle + \varepsilon_l j_i^\Sigma(u^i, w_{l+1}) \;,$$

where $g_l^i \in \partial_{u^i} j^\Delta(u_l, w_{l+1})$ and r_l^i is an additive error on $\partial_{u^i} j^\Delta(u_l, w_{l+1})$. This example shows that the stochastic APP framework encompasses decomposition techniques. △

11.3 Measurability of the iterates of the stochastic APP algorithm

Convergence results for SA algorithms often consist in proving the almost sure convergence of the sequence of iterates $\left\{\boldsymbol{U}_l\right\}_{l\in\mathbb{N}}$ to the optimal value $u^\sharp$. Other results provide non-asymptotic bounds for the expectation of function values $\mathbb{E}\Big(J(\boldsymbol{U}_l) - J(u^\sharp)\Big)$, the probability of large deviation $\mathbb{P}\Big(J(\boldsymbol{U}_l) - J(u^\sharp) > \eta\Big)$ for some $\eta > 0$, or the quadratic mean $\mathbb{E}\left(\left\|\boldsymbol{U}_l - u^\sharp\right\|^2\right)$. In order for these expectations and probabilities to be well-defined, $\boldsymbol{U}_l$ must be a measurable mapping from Ω to $\mathbb{U}$. Hence, the measurability of the iterates is a key theoretical issue. However, it is hardly addressed in the literature. In this section, we prove the measurability of the iterates of the stochastic APP algorithm.

11.3.1 A general measurability result

The aim of this section is to prove that we can obtain the measurability of the iterates of the stochastic APP algorithm. For that purpose, we prove a general measurability result in Theorem 11.24 and obtain the measurability of the iterates of the stochastic APP algorithm as a consequence in Theorem 11.28.

Recall that $(\Omega, \mathcal{A}, \mathbb{P})$ is a probability space and that $(\mathbb{W}, \mathcal{B}(\mathbb{W}))$ is a measurable topological vector space. The Banach space $\mathbb{U}$ is equipped with the Borel σ-field $\mathcal{B}(\mathbb{U})$. The topological dual of $\mathbb{U}$ is denoted by $\mathbb{U}^\star$, and its Borel σ-field is $\mathcal{B}(\mathbb{U}^\star)$. We consider the following problem:

$$\min_{u \in U^{\mathrm{ad}}} \left\{ \Phi(\omega, u) := K(u) + \left\langle \boldsymbol{\varphi}(\omega)\,, u \right\rangle + \varepsilon j^{\Sigma}\Big(u, \boldsymbol{W}(\omega)\Big) \right\}\,, \tag{11.9}$$

where $\varepsilon > 0$ is a given positive real and $\boldsymbol{\varphi} : \Omega \to \mathbb{U}^\star$ is a given measurable function. The goal is to show the existence of a measurable mapping $\widetilde{\boldsymbol{U}}$ such that for all $\omega \in \Omega$, $\widetilde{\boldsymbol{U}}(\omega) \in \arg\min_{u \in U^{\mathrm{ad}}} \Phi(\omega, u)$. The mapping $\omega \mapsto \arg\min_{u \in U^{\mathrm{ad}}} \Phi(\omega, u)$ is a set-valued mapping. We recall some useful results on set-valued mappings in the next section.

11.3.1.1 Some tools from the theory of set-valued mappings

We introduce some tools from the theory of set-valued mappings that are used to state and prove the measurability result of Theorem 11.24. The definitions and propositions are mostly taken from [Castaing and Valadier, 1977, Hess, 1995]. For two sets X, Y, we denote by $\Gamma : X \rightrightarrows Y$ a set-valued mapping Γ from X to Y. This means that for $x \in X$, $\Gamma(x) \subset Y$ or in other words that $\Gamma(x) \in \mathcal{P}(Y)$ where $\mathcal{P}(Y)$ is the power set of Y.

Definition 11.3 (Measure completion)**.** Let $(\Omega, \mathcal{A})$ be a measurable space.

- Let μ be a measure on $(\Omega, \mathcal{A})$. The *μ-completion* of $\mathcal{A}$ is the σ-field $\mathcal{A}_\mu$ generated by $\mathcal{A} \cup \{A' \in \mathcal{P}(\Omega) \,|\, A' \subset A, A \in \mathcal{A} \text{ and } \mu(A) = 0\}$, that is, the union of $\mathcal{A}$ and the μ-negligible sets. The σ-field $\mathcal{A}$ is said to be *complete* for the measure μ if $\mathcal{A} = \mathcal{A}_\mu$.
- The σ-field $\hat{\mathcal{A}}$ of universally measurable sets is defined by $\hat{\mathcal{A}} = \bigcap_\mu \mathcal{A}_\mu$ where μ ranges over the set of positive σ-finite measures on the measurable space $(\Omega, \mathcal{A})$.

Definition 11.4 (Measurable selection)**.** Let $(\Omega, \mathcal{A})$ be a measurable space and $\mathbb{U}$ be a separable metric space. Let $\Gamma : \Omega \rightrightarrows \mathbb{U}$ be a set-valued mapping. A function $\gamma : \Omega \to \mathbb{U}$ is a measurable selection of Γ if $\gamma(\omega) \in \Gamma(\omega)$ for all $\omega \in \Omega$ and γ is measurable.

Definition 11.5 (Measurable mapping)**.** Let $(\Omega, \mathcal{A})$ be a measurable space and $\mathbb{U}$ be a separable metric space. A set-valued mapping $\Gamma : \Omega \rightrightarrows \mathbb{U}$ is Effros-measurable if, for every open set $O \subset \mathbb{U}$, we have:

$$\Gamma^-(O) = \Big\{ \omega \in \Omega,\ \Gamma(\omega) \cap O \neq \emptyset \Big\} \in \mathcal{A}\,.$$

Remark 11.6. The Effros-measurability of a set-valued mapping $\Gamma : \Omega \rightrightarrows \mathbb{U}$ is equivalent to the measurability of Γ viewed as a function from Ω to $\mathcal{P}(\mathbb{U})$. ◊

Proposition 11.7. *[Castaing and Valadier, 1977, Theorem III.9] Assume that $(\Omega, \mathcal{A})$ is a measurable space and let $\mathbb{U}$ be a separable Banach space. Let $\Gamma : \Omega \rightrightarrows \mathbb{U}$ be a non-empty-valued and closed-valued mapping. Then the following statements are equivalent:*

(i) Γ *is Effros-measurable.*

(ii) Γ *admits a Castaing representation: there exists a sequence of measurable functions* $\{\gamma_n\}_{n\in\mathbb{N}}$ *such that for all* $\omega \in \Omega$, $\Gamma(\omega) = \mathrm{cl}\left\{\gamma_n(\omega),\ n \in \mathbb{N}\right\}$ *where* cl *denotes the closure of a set.*

Proposition 11.8. *[Castaing and Valadier, 1977, Proposition III.23: Sainte-Beuve's projection theorem] Let* $(\Omega, \mathcal{A})$ *be a measurable space and* $(\mathbb{U}, \mathcal{B}(\mathbb{U}))$ *be a separable Banach space equipped with its Borel* σ*-field. Let* $G \in \mathcal{A} \otimes \mathcal{B}(\mathbb{U})$. *Denote by* $\mathrm{proj}_\Omega(G)$ *the projection of* G *on* Ω. *Then,* $\mathrm{proj}_\Omega(G) \in \hat{\mathcal{A}}$, *where we recall that* $\hat{\mathcal{A}}$ *is the* σ*-field of universally measurable sets.*

Proposition 11.9. *[Castaing and Valadier, 1977, Proposition III.30] Let* $(\Omega, \mathcal{A}, \mathbb{P})$ *be a measure space where* $\mathcal{A}$ *is a complete* σ*-field, that is,* $\mathcal{A} = \mathcal{A}_{\mathbb{P}}$. *Let* $\mathbb{U}$ *be a separable Banach space. Let* $\Gamma : \Omega \rightrightarrows \mathbb{U}$ *be a non-empty valued and closed-valued mapping. The following statements are equivalent:*

(i) Γ *is Effros-measurable.*

(ii) For every closed set $C \subset \mathbb{U}$, *we have:*

$$\Gamma^-(C) = \{\omega \in \Omega,\ \Gamma(\omega) \cap C \neq \emptyset\} \in \mathcal{A}\ .$$

Remark 11.10. When $\mathbb{U}$ is finite-dimensional, Proposition 11.9 is true in any measurable space $(\Omega, \mathcal{A})$, that is, the completeness assumption of the σ-field $\mathcal{A}$ is not needed [Rockafellar and Wets, 2004, Theorem 14.3]. In the infinite-dimensional setting, (ii) implies (i) is true in any measurable space $(\Omega, \mathcal{A})$ [Castaing and Valadier, 1977, Proposition III.11]. The completeness assumption is only required to prove (i) implies (ii) when $\mathbb{U}$ is infinite-dimensional. Essentially, in the finite-dimensional case, the proof of (i) implies (ii) relies on the fact that $\mathbb{U}$ is locally compact. In the infinite-dimensional case, $\mathbb{U}$ is not locally compact and the proof uses the Sainte Beuve's projection theorem. ♢

Definition 11.11 (Graph and epigraph)**.** Let $(\mathbb{X}, \mathcal{X})$ be a measurable space and $\mathbb{U}$ be a Banach space. Let $h : \mathbb{X} \to \mathbb{R} \cup \{+\infty\}$ be a function and $\Gamma : \mathbb{X} \rightrightarrows \mathbb{U}$ be a set-valued mapping.

- The graph and the epigraph of h are respectively defined by:

$$\mathrm{gph}\, h = \left\{(x, \alpha) \in \mathbb{X} \times \mathbb{R},\ h(x) = \alpha\right\},$$
$$\mathrm{epi}\, h = \left\{(x, \alpha) \in \mathbb{X} \times \mathbb{R},\ h(x) \leq \alpha\right\}.$$

- The graph of Γ is defined by:

$$\mathrm{gph}\, \Gamma = \left\{(x, u) \in \mathbb{X} \times \mathbb{U},\ u \in \Gamma(x)\right\}.$$

Definition 11.12 (Normal integrand)**.** Let $(\Omega, \mathcal{A})$ be a measurable space and $\mathbb{U}$ be a Banach space. A function $f : \Omega \times \mathbb{U} \to \mathbb{R} \cup \{+\infty\}$ is a normal integrand if it satisfies the following conditions:

(i) For all $\omega \in \Omega$, $f(\omega, \cdot)$ is l.s.c.

(ii) The epigraphical mapping $S_f : \Omega \rightrightarrows \mathbb{U} \times \mathbb{R}$ defined by $S_f(\omega) = \operatorname{epi} f(\omega, \cdot)$ is Effros-measurable.

Remark 11.13. The point (i) of Definition 11.12 is equivalent to S_f being closed-valued. Here, we consider the definition of the normal integrand used by [Hess, 1995]. It differs from the definition of [Castaing and Valadier, 1977] where the point (ii) is replaced by the $\mathcal{A} \otimes \mathcal{B}(\mathbb{U})$-measurability of f. We shall see in Proposition 11.18 that the Effros-measurability of the epigraphical mapping S_f implies the $\mathcal{A} \otimes \mathcal{B}(\mathbb{U})$-measurability of f. Note also that if $\mathcal{A}$ is complete for a positive σ-finite measure $\mathbb{P}$, these two definitions are equivalent, see [Castaing and Valadier, 1977, Proposition III.30]. ◊

Definition 11.14 (Carathéodory integrand)**.** Let $(\Omega, \mathcal{A})$ be a measurable space and $\mathbb{U}$ be a separable Banach space. A function $f : \Omega \times \mathbb{U} \to \mathbb{R}$ (finite-valued) is a Carathéodory integrand if it satisfies the following conditions:

(i) For all $u \in \mathbb{U}$, $f(\cdot, u)$ is measurable.

(ii) For all $\omega \in \Omega$, $f(\omega, \cdot)$ is continuous.

Proposition 11.15. *[Hess, 1995, Proposition 2.5] If f is a Carathéodory integrand, then it is a normal integrand.*

Proposition 11.16. *[Castaing and Valadier, 1977, Proposition III.13] Let $(\Omega, \mathcal{A})$ be a measurable space and $(\mathbb{U}, \mathcal{B}(\mathbb{U}))$ be a separable Banach space equipped with its Borel σ-field. If $\Gamma : \Omega \rightrightarrows \mathbb{U}$ is an Effros-measurable, closed-valued mapping, then* $\operatorname{gph} \Gamma \in \mathcal{A} \otimes \mathcal{B}(\mathbb{U})$.

We now recall a technical result on the Borel σ-field of a product space that is used in the proof of subsequent propositions.

Proposition 11.17. *[Bertsekas and Shreve, 1996, Proposition 7.13] Consider a sequence of measurable separable topological spaces $\left\{\left(\mathbb{X}_i, \mathcal{B}(\mathbb{X}_i)\right)\right\}_{i \in \mathbb{N}}$ equipped with their Borel σ-fields. For $n \in \mathbb{N}$, let $\mathbb{Y}_n = \prod_{i=1}^n \mathbb{X}_i$ and let $\mathbb{Y} = \prod_{i \in \mathbb{N}} \mathbb{X}_i$. Then, the Borel σ-field of the product space $\mathbb{Y}_n$ (resp. $\mathbb{Y}$) coincides with the product of the Borel σ-fields of $\{\mathbb{X}_i\}_{i=1}^n$ (resp. $\{\mathbb{X}_i\}_{i \in \mathbb{N}}$), that is:*

$$\mathcal{B}\left(\mathbb{Y}_n\right) = \bigotimes_{i=1}^{n} \mathcal{B}(\mathbb{X}_i) \quad \text{and} \quad \mathcal{B}\left(\mathbb{Y}\right) = \bigotimes_{i \in \mathbb{N}} \mathcal{B}(\mathbb{X}_i) \, .$$

The following proposition shows that a normal integrand $f : \Omega \times \mathbb{U} \to \mathbb{R} \cup \{+\infty\}$, as defined in [Hess, 1995], is jointly $\mathcal{A} \otimes \mathcal{B}(\mathbb{U})$-measurable. This result is given in [Rockafellar and Wets, 2004, Corollary 14.34] when $\mathbb{U} = \mathbb{R}^n$ but is extended here in the Banach case.

Proposition 11.18. *Let $(\Omega, \mathcal{A})$ be a measurable space and $(\mathbb{U}, \mathcal{B}(\mathbb{U}))$ be a separable Banach space equipped with its Borel σ-field. If $f : \Omega \times \mathbb{U} \to \mathbb{R} \cup \{+\infty\}$ is a normal integrand, then f is $\mathcal{A} \otimes \mathcal{B}(\mathbb{U})$-measurable.*

Proof. The function f is a normal integrand so its epigraphical mapping S_f is Effros-measurable and closed-valued. Moreover $\mathbb{U}$ is separable, so by Proposition 11.16, we get that:

$$\operatorname{gph} S_f = \Big\{(\omega, u, \alpha) \in \Omega \times \mathbb{U} \times \mathbb{R}, \; f(\omega, u) \leq \alpha\Big\} \in \mathcal{A} \otimes \mathcal{B}(\mathbb{U} \times \mathbb{R}) \, .$$

Using that $\mathbb{U}$ and $\mathbb{R}$ are separable, we have $\mathcal{B}(\mathbb{U}\times\mathbb{R}) = \mathcal{B}(\mathbb{U})\otimes\mathcal{B}(\mathbb{R})$ by Proposition 11.17. Then, for each $\alpha\in\mathbb{R}$, we get:

$$f^{-1}\Big(]-\infty,\alpha]\Big) = \Big\{(\omega,u)\in\Omega\times\mathbb{U},\ f(\omega,u)\leq\alpha\Big\}\in\mathcal{A}\otimes\mathcal{B}(\mathbb{U})\ .$$

This shows that f is $\mathcal{A}\otimes\mathcal{B}(\mathbb{U})$-measurable. □

The following proposition is an adaptation of [Rockafellar and Wets, 2004, Proposition 14.45(c)] on the composition operations on normal integrands to the Banach case. Note that the separability of $\mathbb{U}$ is a crucial assumption that is used explicitly in the proof of Proposition 11.19 and that appears in most of the results of this part. Essentially, as only a countable union of measurable sets is measurable, countable dense subsets of a separable space are often used in proofs of measurability. Moreover, in the infinite-dimensional setting, we must assume the completeness of the σ-field $\mathcal{A}$ because we appeal to Proposition 11.9 in the proof.

Proposition 11.19. *Let $(\Omega,\mathcal{A},\mathbb{P})$ be a measure space where $\mathcal{A}$ is a complete σ-field, that is, $\mathcal{A}=\mathcal{A}_{\mathbb{P}}$. Let $(\mathbb{W},\mathcal{B}(\mathbb{W}))$ be a topological measurable space and $(\mathbb{U},\mathcal{B}(\mathbb{U}))$ be a separable Banach space equipped with its Borel σ-field. Let $h:\mathbb{U}\times\mathbb{W}\to\mathbb{R}\cup\{+\infty\}$ be l.s.c. and $\boldsymbol{W}:\Omega\to\mathbb{W}$ be a measurable mapping. Then:*

$$f:(\omega,u)\in\Omega\times\mathbb{U}\mapsto h\Big(u,\boldsymbol{W}(\omega)\Big)\in\mathbb{R}\cup\{+\infty\}$$

is a normal integrand.

Proof. We have that h is l.s.c. so $f(\omega,\cdot)=h(\cdot,\boldsymbol{W}(\omega))$ is l.s.c. for all $\omega\in\Omega$. It remains to prove that the epigraphical mapping S_f is Effros-measurable. As h is l.s.c., the set $\operatorname{epi}h$ is closed. Define:

$$G:(\omega,u,\alpha)\in\Omega\times\mathbb{U}\times\mathbb{R}\mapsto(u,\boldsymbol{W}(\omega),\alpha)\in\mathbb{U}\times\mathbb{W}\times\mathbb{R}\ ,$$

Then, let:

$$\begin{aligned}Q(\omega)&=\Big[(\mathbb{U}\times\mathbb{R})\times\operatorname{epi}h\Big]\cap\operatorname{gph}G(\omega,\cdot,\cdot)\ ,\\&=\Big\{\Big((u,\alpha),(u,\boldsymbol{W}(\omega),\alpha)\Big)\ \text{ such that }\ h(u,\boldsymbol{W}(\omega))\leq\alpha,\ (u,\alpha)\in\mathbb{U}\times\mathbb{R}\Big\}\ ,\\&=\Big\{\Big((u,\alpha),(u,\boldsymbol{W}(\omega),\alpha)\Big)\ \text{ such that }\ f(\omega,u)\leq\alpha,\ (u,\alpha)\in\mathbb{U}\times\mathbb{R}\Big\}\ .\end{aligned}$$

Now, define the projection operator P as:

$$\begin{aligned}P:(\mathbb{U}\times\mathbb{R})\times(\mathbb{U}\times\mathbb{W}\times\mathbb{R})&\to(\mathbb{U}\times\mathbb{R})\ ,\\((u,\alpha),(v,w,\beta))&\mapsto(u,\alpha)\end{aligned}$$

so that we have:

$$S_f(\omega)=\Big\{(u,\alpha)\in\mathbb{U}\times\mathbb{R},\ f(\omega,u)\leq\alpha\Big\}=P\Big(Q(\omega)\Big)\ .$$

- Let Γ be the set valued mapping defined by $\Gamma:\omega\in\Omega\mapsto\operatorname{gph}G(\omega,\cdot,\cdot)\in(\mathbb{U}\times\mathbb{R})\times(\mathbb{U}\times\mathbb{W}\times\mathbb{R})$. We show that Γ is Effros-measurable. As $\mathbb{U}$ is separable, there exists a countable dense subset $\{(b_n,r_n),\ n\in\mathbb{N}\}$ of $\mathbb{U}\times\mathbb{R}$. For $n\in\mathbb{N}$, let $\gamma_n(\omega)=\Big((b_n,r_n),G(\omega,b_n,r_n)\Big)$. As $G(\omega,b_n,r_n)=(b_n,\boldsymbol{W}(\omega),r_n)$ and $\boldsymbol{W}$ is measurable, we get that γ_n is measurable. Then, we have $\Gamma(\omega)=\operatorname{cl}\{\gamma_n(\omega),\ n\in\mathbb{N}\}$. Hence, $\{\gamma_n\}_{n\in\mathbb{N}}$ is a Castaing representation of Γ. Moreover, Γ is closed-valued and non-empty valued so by Proposition 11.7, we deduce that Γ is Effros-measurable.

- Let $C \subset (\mathbb{U} \times \mathbb{R}) \times (\mathbb{U} \times \mathbb{W} \times \mathbb{R})$ be a closed set. We have:

$$\begin{aligned} Q^-(C) &= \left\{\omega \in \Omega, \left[(\mathbb{U} \times \mathbb{R}) \times \operatorname{epi} h\right] \cap \Gamma(\omega) \cap C \neq \emptyset\right\} , \\ &= \Gamma^-\left(C \cap \left[(\mathbb{U} \times \mathbb{R}) \times \operatorname{epi} h\right]\right) . \end{aligned}$$

  As $\operatorname{epi} h$ is closed, the set $C \cap \left[(\mathbb{U} \times \mathbb{R}) \times \operatorname{epi} h\right]$ is closed. By assumption, the σ-field $\mathcal{A}$ is complete and we have shown that Γ is Effros-measurable, therefore by Proposition 11.9, we get that $\Gamma^-\left(C \cap \left[(\mathbb{U} \times \mathbb{R}) \times \operatorname{epi} h\right]\right) = Q^-(C) \in \mathcal{A}$. Hence, Q is Effros-measurable.

- Finally, for every open set $V \subset \mathbb{U} \times \mathbb{R}$, as $S_f(\omega) = P\Big(Q(\omega)\Big)$, we have:

$$S_f^-(V) = \left\{\omega \in \Omega,\ Q(\omega) \cap P^{-1}(V) \neq \emptyset\right\} .$$

  The projection P is continuous so $P^{-1}(V)$ is open. By the Effros-measurability of Q, we get that $S_f^-(V) \in \mathcal{A}$, that is, S_f is Effros-measurable.

This completes the proof. □

We now give the main results that are used to prove the measurability of the iterates of the stochastic APP. The following proposition is a slight extension of [Hess, 1996, Proposition 4.2(c)].

Proposition 11.20. *Let $(\Omega, \mathcal{A})$ be a measurable space and $(\mathbb{U}, \mathcal{B}(\mathbb{U}))$ be a separable Banach space equipped with its Borel σ-field. Let U^{ad} be a closed subset of $\mathbb{U}$. Let $f : \Omega \times \mathbb{U} \to \mathbb{R} \cup \{+\infty\}$ be an $\mathcal{A} \otimes \mathcal{B}(\mathbb{U})$-measurable function. Let $M : \Omega \rightrightarrows \mathbb{U}$ be the argmin set-valued mapping:*

$$M(\omega) = \operatorname*{arg\,min}_{u \in U^{\mathrm{ad}}} f(\omega, u) .$$

Assume that the argmin mapping M is non-empty valued, then M admits an $\hat{\mathcal{A}}$-measurable selection.

Proof. Let $\alpha \in \mathbb{R}$ and $m(\omega) = \min_{u \in U^{\mathrm{ad}}} f(\omega, u)$. The function m is well-defined as M is non-empty valued. Let:

$$H = (\Omega \times U^{\mathrm{ad}}) \cap \{(\omega, u) \in \Omega \times \mathbb{U},\ f(\omega, u) < \alpha\} .$$

We have:

$$\{\omega \in \Omega,\ m(\omega) < \alpha\} = \operatorname{proj}_\Omega(H) ,$$

where $\operatorname{proj}_\Omega(H)$ is the projection of H on Ω. As f is $\mathcal{A} \otimes \mathcal{B}(\mathbb{U})$-measurable and U^{ad} is closed hence measurable, we get that $H \in \mathcal{A} \otimes \mathcal{B}(\mathbb{U})$. From Proposition 11.8, we deduce that $m^{-1}(]-\infty, \alpha[)$ is $\hat{\mathcal{A}}$-measurable so that m is $\hat{\mathcal{A}}$-measurable. As $\mathcal{A} \subset \hat{\mathcal{A}}$, the function f is $\hat{\mathcal{A}} \otimes \mathcal{B}(\mathbb{U})$-measurable. We can write:

$$M(\omega) = \left\{u \in U^{\mathrm{ad}},\ f(\omega, u) = m(\omega)\right\} ,$$

so, $\operatorname{gph} M = \left\{(\omega, u) \in \Omega \times U^{\mathrm{ad}}, f(\omega, u) = m(\omega)\right\}$. Therefore, $\operatorname{gph} M$ is $\hat{\mathcal{A}} \otimes \mathcal{B}(\mathbb{U})$-measurable as the inverse image of $\{0\}$ under the $\hat{\mathcal{A}} \otimes \mathcal{B}(\mathbb{U})$-measurable mapping $(\omega, u) \mapsto f(\omega, u) - m(\omega)$. Let O be an open subset of $\mathbb{U}$. We have:

$$M^-(O) = \operatorname{proj}_\Omega((\Omega \times O) \cap \operatorname{gph} M) .$$

As $(\Omega \times O) \cap \operatorname{gph} M \in \hat{\mathcal{A}} \otimes \mathcal{B}(\mathbb{U})$, by Proposition 11.8, we get that $M^{-}(O) \in \hat{\hat{\mathcal{A}}} = \hat{\mathcal{A}}$. Hence, M is Effros-measurable for the σ-field $\hat{\mathcal{A}}$ and is non-empty-valued by assumption, so by Proposition 11.7, M admits an $\hat{\mathcal{A}}$-measurable selection. □

Corollary 11.21. *Let $(\Omega, \mathcal{A}, \mathbb{P})$ be a probability space, where $\mathcal{A}$ is a complete σ-field, that is, $\mathcal{A} = \mathcal{A}_{\mathbb{P}}$. Let $(\mathbb{U}, \mathcal{B}(\mathbb{U}))$ be a separable Banach space equipped with its Borel σ-field. Let $f : \Omega \times \mathbb{U} \to \mathbb{R} \cup \{+\infty\}$ be an $\mathcal{A} \otimes \mathcal{B}(\mathbb{U})$-measurable function. Suppose that the argmin mapping M is non-empty valued. Then, M admits an $\mathcal{A}$-measurable selection.*

Proof. As $\mathbb{P}$ is a positive σ-finite measure, we have $\hat{\mathcal{A}} = \bigcap_{\mu} \mathcal{A}_{\mu} \subset \mathcal{A}_{\mathbb{P}} = \mathcal{A}$. By Proposition 11.20, M admits an $\hat{\mathcal{A}}$-measurable selection, which is also an $\mathcal{A}$-measurable selection. □

Proposition 11.22. *[Hess, 1995, Theorem 4.6] Let $(\Omega, \mathcal{A})$ be a measurable space, $\mathbb{U}$ be a separable Banach space with separable topological dual $\mathbb{U}^{\star}$. Let $f : \Omega \times \mathbb{U} \to \mathbb{R} \cup \{+\infty\}$ be a normal integrand and assume that $f(\omega, \cdot)$ is proper, convex and subdifferentiable for all $\omega \in \Omega$. Let $\boldsymbol{U} : \Omega \to \mathbb{U}$ be a measurable mapping. Then, the set-valued mapping $D_{\boldsymbol{U}} : \Omega \rightrightarrows \mathbb{U}^{\star}$ such that:*

$$\begin{aligned} D_{\boldsymbol{U}}(\omega) &= \partial_u f\Big(\omega, \boldsymbol{U}(\omega)\Big) \\ &= \Big\{ v \in \mathbb{U}^{\star},\ f(\omega, u) \geq f(\omega, \boldsymbol{U}(\omega)) + \Big\langle v\,, u - \boldsymbol{U}(\omega) \Big\rangle,\ \forall u \in \mathbb{U} \Big\}, \end{aligned}$$

is Effros-measurable.

11.3.1.2 Existence of a measurable selection for the argmin mapping

In this section, we make use of the tools introduced in §11.3.1.1 to prove our main measurability result. We introduce the argmin set-valued mapping $M : \Omega \rightrightarrows \mathbb{U}$ for Problem (11.9):

$$M(\omega) = \operatorname*{arg\,min}_{u \in U^{\mathrm{ad}}} \Big\{ \Phi(\omega, u) := K(u) + \Big\langle \boldsymbol{\varphi}(\omega)\,, u \Big\rangle + \varepsilon j^{\Sigma}\Big(u, \boldsymbol{W}(\omega)\Big) \Big\}\ . \quad (11.10)$$

We consider the following assumptions:

(A1) The space $\mathbb{U}$ is a reflexive, separable Banach space.

(A2) U^{ad} is a non-empty closed convex subset of $\mathbb{U}$.

(A3) $j^{\Sigma} : \mathbb{U} \times \mathbb{W} \to \mathbb{R}$ is jointly l.s.c. and for all $w \in \mathbb{W}$, $j^{\Sigma}(\cdot, w)$ is proper and convex.

(A4) The function $K : \mathbb{U} \to \mathbb{R}$ is proper, convex, l.s.c. and Gateaux-differentiable on an open set containing U^{ad}.

(A5) For all $\omega \in \Omega$, the function $u \mapsto \Phi(\omega, u)$ is coercive.

(A6) The σ-field $\mathcal{A}$ is complete for the measure $\mathbb{P}$, that is, $\mathcal{A} = \mathcal{A}_{\mathbb{P}}$.

(A7) The function $\boldsymbol{W} : \Omega \to \mathbb{W}$ is measurable.

(A8) The function $\boldsymbol{\varphi} : \Omega \to \mathbb{U}^{\star}$ is measurable.

The objective of this part is to prove that M defined in Equation (11.10) admits a measurable selection. We start by a classical theorem from optimization theory giving conditions for the existence and unicity of a minimizer $\Phi(\omega, \cdot)$.

Theorem 11.23. *Let $\omega \in \Omega$. Under Assumptions (A1)-(A5), $M(\omega)$ is non-empty, closed and convex. Moreover, if K is strongly convex, then $M(\omega)$ is a singleton, meaning that $\Phi(\omega, \cdot)$, defined in (11.10), has a unique minimizer.*

Proof. The objective function $\Phi(\omega, \cdot)$ is the sum of three convex, l.s.c. functions, it is then convex and l.s.c. By (A5), the objective function is also coercive. As $\mathbb{U}$ is a reflexive Banach space (A1) and U^{ad} is non-empty, closed and convex (A2), the set of minimizers $M(\omega)$ is non-empty [Brézis, 2005, Corollary III.20]. The convexity of $\Phi(\omega, \cdot)$ ensures that $M(\omega)$ is convex and the lower-semicontinuity of $\Phi(\omega, \cdot)$ ensures that $M(\omega)$ is closed.

If K is strongly convex, then $\Phi(\omega, \cdot)$ is strongly convex, hence the minimizer of $\Phi(\omega, \cdot)$ is unique so $M(\omega)$ is a singleton.[1] □

Theorem 11.24. *Under Assumptions (A1)-(A8), the mapping M defined in Equation (11.10) admits a measurable selection.*

Proof. We start by proving that $\Phi(\omega, u) = K(u) + \left\langle \boldsymbol{\varphi}(\omega)\,, u \right\rangle + \varepsilon j^{\Sigma}\Big(u, \boldsymbol{W}(\omega)\Big)$ is a normal integrand:

- As the function K is l.s.c. (A4), $(\omega, u) \mapsto K(u)$ is a normal integrand. Indeed, its epigraphical mapping $\omega \mapsto \{(u, \alpha) \in \mathbb{U} \times \mathbb{R},\ K(u) \leq \alpha\}$ is a constant function of ω and is then measurable.

- We have that the Banach space $\mathbb{U}$ is separable (A1) and that $\mathcal{A}$ is complete (A6). The space $\mathbb{U}^\star$ equipped with its Borel σ-field $\mathcal{B}(\mathbb{U}^\star)$ is a measurable space. The function $\boldsymbol{\varphi}$ is measurable (A8) and the function $(u, v) \in \mathbb{U} \times \mathbb{U}^\star \mapsto \langle v\,, u \rangle \in \mathbb{R}$ is continuous hence l.s.c. in particular. Then, Proposition 11.19 applies, showing that the function $(\omega, u) \mapsto \left\langle \boldsymbol{\varphi}(\omega)\,, u \right\rangle$ is a normal integrand.

- With the same reasoning, using that $\mathbb{U}$ is separable (A1), $\boldsymbol{W}$ is measurable (A7), $\mathcal{A}$ is complete (A6) and j^{Σ} is l.s.c. (A3), we appeal to Proposition 11.19 with $h = j^{\Sigma}$ to deduce that $(\omega, u) \mapsto j^{\Sigma}\Big(u, \boldsymbol{W}(\omega)\Big)$ is a normal integrand.

The function Φ is then a normal integrand as the sum of three normal integrands. By Proposition 11.18, Φ is then $\mathcal{A} \otimes \mathcal{B}(\mathbb{U})$-measurable. Moreover, the σ-field $\mathcal{A}$ is complete for $\mathbb{P}$ (A6) and $\mathbb{U}$ is separable (A1). In addition, using (A2)-(A5) to apply Theorem 11.23 ensures that M is non-empty valued. Hence, by Corollary 11.21, we conclude that $M : \omega \mapsto \arg\min_{u \in U^{\text{ad}}} \Phi(\omega, u)$ admits a measurable selection. □

Corollary 11.25. *Under Assumptions (A1)-(A8) and if we additionally assume that K is strongly convex, then for all $\omega \in \Omega$, $\Phi(\omega, \cdot)$, defined in (11.10), has a unique minimizer and the mapping:*

$$\widetilde{\boldsymbol{U}}(\omega) = \operatorname*{arg\,min}_{u \in U^{\text{ad}}} \Phi(\omega, u) \in \mathbb{U}$$

is measurable, that is, $\widetilde{\boldsymbol{U}}$ is a random variable.

[1] In the case where K is strongly convex, the coercivity assumption is not needed as it is implied by the strong convexity of $\Phi(\omega, \cdot)$.

11.3.2 Application to the stochastic APP algorithm

We aim at studying the iterations of the stochastic APP in terms of random variables so we consider the argmin set-valued mapping $M : \Omega \rightrightarrows \mathbb{U}$ defined by:

$$M(\omega) = \underset{u \in U^{\mathrm{ad}}}{\arg\min}\, K(u) + \Big\langle \varepsilon(\boldsymbol{G}(\omega) + \boldsymbol{R}(\omega)) - \nabla K(\boldsymbol{U}(\omega)) \,, u \Big\rangle + \varepsilon j^{\Sigma}(u, \boldsymbol{W}(\omega)) \,, \quad (11.11)$$

with $\varepsilon > 0$, $\boldsymbol{U}(\omega) \in U^{\mathrm{ad}}$, $\boldsymbol{W}(\omega) \in \mathbb{W}$, $\boldsymbol{G}(\omega) \in \partial_u j^{\Delta}(\boldsymbol{U}(\omega), \boldsymbol{W}(\omega))$ and $\boldsymbol{R}(\omega) \in \mathbb{U}^{\star}$. An iteration of the stochastic APP algorithm consists in solving Problem (11.11), which is exactly of the form of Problem (11.10) with:

$$\boldsymbol{\varphi}(\omega) = \varepsilon\Big(\boldsymbol{G}(\omega) + \boldsymbol{R}(\omega)\Big) - \nabla K\Big(\boldsymbol{U}(\omega)\Big) \,. \quad (11.12)$$

In addition to (A1)-(A7), we assume now:

(A9) The dual space $\mathbb{U}^{\star}$ is separable.

(A10) The function $j^{\Delta} : \mathbb{U} \times \mathbb{W} \to \mathbb{R}$ that appears in Problem (11.1) is jointly l.s.c. and for all $w \in \mathbb{W}$, $j^{\Delta}(\cdot, w)$ is proper, convex and subdifferentiable on an open set containing U^{ad}.

(A11) The mappings $\boldsymbol{U} : \Omega \to U^{\mathrm{ad}}$ and $\boldsymbol{R} : \Omega \to \mathbb{U}^{\star}$ are measurable.

Assumption (A11) means that the mappings $\boldsymbol{U}$ and $\boldsymbol{R}$ are random variables. We cannot do the same for the mapping $\boldsymbol{G}$ as it must satisfy $\boldsymbol{G}(\omega) \in \partial_u j^{\Delta}(\boldsymbol{U}(\omega), \boldsymbol{W}(\omega))$ for all $\omega \in \Omega$. Hence, we need to ensure that there exists a measurable mapping satisfying this constraint. This is the object of the following proposition.

Proposition 11.26. *Under Assumptions (A1), (A6), (A7), (A9)-(A11), the subgradient mapping $\Gamma : \omega \mapsto \partial_u j^{\Delta}(\boldsymbol{U}(\omega), \boldsymbol{W}(\omega)) \subset \mathbb{U}^{\star}$ admits a measurable selection $\boldsymbol{G} : \Omega \to \mathbb{U}^{\star}$.*

Proof. Let $f(\omega, u) = j^{\Delta}\Big(u, \boldsymbol{W}(\omega)\Big)$ for $\omega \in \Omega,\ u \in \mathbb{U}$.

- Using that $\mathbb{U}$ is separable (A1), $\boldsymbol{W}$ is measurable (A7), $\mathcal{A}$ is complete (A6) and j^{Δ} is l.s.c. (A10), Proposition 11.19 with $h = j^{\Delta}$ shows that f is a normal integrand.
- We have that for all $\omega \in \Omega,\ \Gamma(\omega) = \partial_u f\Big(\omega, \boldsymbol{U}(\omega)\Big)$. With (A10), we get that $f(\omega, \cdot)$ is proper, convex and subdifferentiable for all $\omega \in \Omega$. We have that $\mathbb{U}$ and $\mathbb{U}^{\star}$ are separable (A1), (A9), $\boldsymbol{U}$ is measurable (A11) and f is a normal integrand, so by Proposition 11.22, Γ is Effros-measurable.

Assumption (A10) ensures that Γ is non-empty valued. In addition, Γ is Effros-measurable and closed-valued in $\mathbb{U}^{\star}$ which is separable (A9). By Proposition 11.7, Γ admits a measurable selection. This means that there exists a measurable function $\boldsymbol{G} : \Omega \to \mathbb{U}^{\star}$ such that for all $\omega \in \Omega,\ \boldsymbol{G}(\omega) \in \Gamma(\omega) = \partial_u j^{\Delta}(\boldsymbol{U}(\omega), \boldsymbol{W}(\omega))$. □

In the sequel, $\boldsymbol{G}$ denotes a measurable selection of Γ. In order to apply Theorem 11.24 to prove that the iterates of the stochastic APP algorithm are measurable, we must ensure that Assumption (A8) is satisfied, that is, we must show that the mapping $\boldsymbol{\varphi}$ defined in (11.12) is measurable. We prove in Proposition 11.27 that Assumption (A8) can be deduced from the other assumptions.

Proposition 11.27. *Under Assumptions (A1), (A4), (A7), (A9)-(A11), the function $\boldsymbol{\varphi}$ is measurable.*

Proof. We have already seen in the proof of Theorem 11.24 that $\Lambda : (\omega, u) \mapsto K(u)$ is a normal integrand. Assumption (A4) ensures that $\Lambda(\omega, \cdot)$ is proper, convex and subdifferentiable for all $\omega \in \Omega$. We have that $\mathbb{U}$ and $\mathbb{U}^\star$ are separable (A1), (A9), $\boldsymbol{U}$ is measurable (A11), so $\omega \mapsto \nabla_u \Lambda\Big(\omega, \boldsymbol{U}(\omega)\Big) = \nabla K\Big(\boldsymbol{U}(\omega)\Big)$ is measurable by Proposition 11.22. Finally, $\boldsymbol{R}$ is also measurable (A11), so $\boldsymbol{\varphi}$ is measurable as a sum of measurable functions. □

Putting Theorem 11.24 and Proposition 11.27 together, we have obtained that under Assumptions (A1)-(A7), (A9)-(A11), the mapping $M : \Omega \rightrightarrows \mathbb{U}$ defined in (11.11) admits a measurable selection. We can now give the measurability result for the iterates of the stochastic APP algorithm, which is defined by the following recursion for $\omega \in \Omega$ and $l \in \mathbb{N}$:

$$\begin{aligned} M_0(\omega) &= \{u_0\} \subset U^{\mathrm{ad}} \,, \\ M_{l+1}(\omega) &= \operatorname*{arg\,min}_{u \in U^{\mathrm{ad}}} K(u) + \Big\langle \varepsilon_l\Big(\boldsymbol{G}_l(\omega) + \boldsymbol{R}_l(\omega)\Big) - \nabla K\Big(\boldsymbol{U}_l(\omega)\Big) \,, u \Big\rangle \\ &\qquad + \varepsilon_l j^{\Sigma}\Big(u, \boldsymbol{W}_{l+1}(\omega)\Big) \,, \end{aligned} \tag{11.13}$$

Theorem 11.28. *Under Assumptions (A1)-(A7), (A9)-(A11), for all $l \in \mathbb{N}$, the mapping M_l that defines the l-th iteration of the stochastic APP algorithm (11.13) admits a measurable selection.*

Proof. The mapping M_0 admits a measurable selection defined by $\boldsymbol{U}_0(\omega) = u_0$. Then, by iteratively using the fact that (11.11) admits a measurable selection, we deduce that for all $l \in \mathbb{N}$, M_l admits a measurable selection. □

Corollary 11.29. *Assume that (A1)-(A7), (A9)-(A11) are satisfied and that the auxiliary mapping K is strongly convex. Then, for all $l \in \mathbb{N}$, the unique mapping $\boldsymbol{U}_l$ that defines the l-th iterate of the stochastic APP algorithm is measurable.*

Proof. If K is strongly convex, from Corollary 11.25, we get that M_l is single-valued, so the iterate $\boldsymbol{U}_l$ is uniquely defined. The measurability of $\boldsymbol{U}_l$ follows from Theorem 11.28. This concludes the proof of the measurability of the iterates of the stochastic APP algorithm. □

Remark 11.30. In [Rockafellar and Wets, 2004, Chapter 14], a whole set of measurability results are exposed in the case where $\mathbb{U}$ is finite-dimensional, which allows to avoid some technicalities of the infinite-dimensional case. In particular, the completeness Assumption (A6) is not needed as shown by [Rockafellar and Wets, 2004, Proposition 14.37] which is the analogous of Proposition 11.20 in the finite-dimensional case. ♢

Remark 11.31. In Problem (11.1), when $\mathbb{U}$ is a Hilbert space (and hence $\mathbb{U}^\star = \mathbb{U}$), $U^{\mathrm{ad}} = \mathbb{U}$ and $j^{\Sigma} = 0$, we can use stochastic gradient descent. Then, we have the explicit formula:

$$\boldsymbol{U}_{l+1} = \boldsymbol{U}_l - \varepsilon_l \nabla j^{\Delta}(\boldsymbol{U}_l, \boldsymbol{W}_{l+1}) \,. \tag{11.14}$$

Under Assumptions (A1), (A7), (A10), the measurability of the iterates is directly obtained by induction using the explicit formula (11.14). ♢

11.4 Convergence results and efficiency estimates

In this section, we prove the convergence of the stochastic APP algorithm for solving Problem (11.1) that we recall here:

$$\min_{u\in U^{\mathrm{ad}}} \Big\{ J(u) := J^{\Delta}(u) + J^{\Sigma}(u) \Big\} \text{ where } \begin{cases} J^{\Delta}(u) = \mathbb{E}\left(j^{\Delta}(u, \boldsymbol{W}) \right) , \\ J^{\Sigma}(u) = \mathbb{E}\left(j^{\Sigma}(u, \boldsymbol{W}) \right) . \end{cases}$$

In addition, we give efficiency estimates for the convergence of function values. In Appendix E, we give some technical results that are used in the proofs of this section.

11.4.1 Convergence of the stochastic APP algorithm

We introduce a filtration $\{\mathcal{F}_l\}_{l\in\mathbb{N}}$, where for $l \in \mathbb{N}$, $\mathcal{F}_l$ is the σ-field generated by the random variables $\left(\boldsymbol{W}_1, \ldots, \boldsymbol{W}_l\right)$ that appear in the successive iterations of the stochastic APP algorithm (11.13). Recall that, in (11.13), $\boldsymbol{G}_l \in \partial_u j^{\Delta}(\boldsymbol{U}_l, \boldsymbol{W}_{l+1})$[2] is an unbiased stochastic gradient, whereas the term $\boldsymbol{R}_l$ represents a bias on the gradient.

The convergence results for the iterates and the function values of the stochastic APP algorithm are already proved in [Culioli, 1987, Culioli and Cohen, 1990] in the case where $\mathbb{U}$ is a Hilbert space (possibly infinite-dimensional) and when there is no bias $\boldsymbol{R}_l$. In [Geiersbach and Pflug, 2019], convergence of the projected stochastic gradient descent is proved in a Hilbert space and with a bias $\boldsymbol{R}_l$. For stochastic mirror descent, convergence results and efficiency estimates can be found in [Nemirovski et al., 2009], but no bias is considered. Here, we present convergence results in the Banach case for the stochastic APP algorithm and we allow for a bias $\boldsymbol{R}_l$, hence generalizing the previous results.

In the sequel, in addition to (A1)-(A7), (A9)-(A11), we make the following assumptions:

(A12) The functions $j^{\Delta}(\cdot, w) : \mathbb{U} \to \mathbb{R}$ and $j^{\Sigma}(\cdot, w) : \mathbb{U} \to \mathbb{R}$ have linearly bounded subgradient in u, uniformly in $w \in \mathbb{W}$:

$$\exists c_1 > 0,\ \exists c_2 > 0,\ \forall u \in U^{\mathrm{ad}},\ \forall w \in \mathbb{W},\ \forall r \in \partial_u j^{\Delta}(u, w),\ \|r\| \leq c_1\|u\| + c_2 .$$
$$\exists d_1 > 0,\ \exists d_2 > 0,\ \forall u \in U^{\mathrm{ad}},\ \forall w \in \mathbb{W},\ \forall s \in \partial_u j^{\Sigma}(u, w),\ \|s\| \leq d_1\|u\| + d_2 .$$

(A13) The objective function J is coercive[3] on U^{ad}. This assumption is automatically satisfied if U^{ad} is bounded.

(A14) The function K is b-strongly convex[3] with $b > 0$ and ∇K is L_K-Lipschitz continuous[3] with $L_K > 0$.

(A15) The sequence of step sizes $\{\varepsilon_l\}_{l\in\mathbb{N}}$ is such that:

$$\sum_{l\in\mathbb{N}} \varepsilon_l = +\infty , \quad \sum_{l\in\mathbb{N}} \varepsilon_l^2 < +\infty .$$

[2] In this expression, the $\in$ relationship is to be understood ω *by* ω.
[3] Basic notions in convex analysis are defined in Appendix A.

(A16) The sequence of random variables $\left\{\boldsymbol{R}_l\right\}_{l\in\mathbb{N}}$ is $\mathbb{P}$-almost surely ($\mathbb{P}$-a.s.) bounded,[4] each $\boldsymbol{R}_l$ is measurable with respect to $\mathcal{F}_{l+1}$ and we have:

$$\sum_{l\in\mathbb{N}} \varepsilon_l \mathbb{E}\Big(\left\|\boldsymbol{R}_l\right\| \;\Big|\; \mathcal{F}_l \Big) < +\infty \quad \mathbb{P}\text{-a.s.}$$

Assumptions (A1)-(A3), (A10) and (A13) ensure that J is well-defined, convex, l.s.c., coercive and attains its minimum on U^{ad}. Hence, Problem (11.1) has a non-empty set of solutions $U^\sharp$. We denote by $J^\sharp$ the value of J on $U^\sharp$. From (A14), K is b-strongly convex, so by Corollary 11.29, the problem solved at each iteration l of the stochastic APP algorithm admits a unique solution $\boldsymbol{U}_{l+1}$, which is measurable.

We start by a technical lemma where we give an inequality that is satisfied by a Lyapunov function for the stochastic APP algorithm. This inequality will be used for the proof of convergence of the stochastic APP algorithm in Theorem 11.33 but also to derive efficiency estimates in Theorems 11.36 and 11.37.

Lemma 11.32. *Let $v \in U^{\mathrm{ad}}$ and consider the Lyapunov function:*

$$\ell_v(u) = K(v) - K(u) - \left\langle \nabla K(u)\,, v-u \right\rangle, \quad u \in U^{\mathrm{ad}}\,.$$

Let $\{u_l\}_{l\in\mathbb{N}}$ be the sequence of iterates generated by Algorithm 9 corresponding to the realization $\{w_l\}_{l\in\mathbb{N}}$ of the stochastic process $\left\{\boldsymbol{W}_l\right\}_{l\in\mathbb{N}}$. Then, under Assumptions (A10), (A12) and (A14), there exists constants $\alpha,\beta,\gamma,\delta>0$ such that, for all $l\in\mathbb{N}$:

$$\begin{aligned}\ell_v(u_{l+1}) \le \Big(1+\alpha\varepsilon_l^2 + \frac{2}{b}\varepsilon_l\|r_l\|\Big)\ell_v(u_l) + \beta\varepsilon_l^2\ell_v(u_{l+1}) \\ + \Big(\gamma\varepsilon_l^2 + \varepsilon_l\|r_l\| + \delta(\varepsilon_l\|r_l\|)^2\Big) \\ + \varepsilon_l\Big((j^\Delta + j^\Sigma)(v,w_{l+1}) - (j^\Delta + j^\Sigma)(u_l,w_{l+1})\Big)\,, \end{aligned} \tag{11.15}$$

where we recall that $b>0$ is the strong convexity constant of K, ε_l is the step size and r_l is the additive error on the stochastic gradient at iteration l of the stochastic APP algorithm.

Proof. By (A14), K is b-strongly convex implying that:

$$\frac{b}{2}\|u-v\|^2 \le \ell_v(u)\,. \tag{11.16}$$

This shows that ℓ_v is lower bounded and coercive.

Let $l\in\mathbb{N}$, as u_{l+1} is solution of (11.6), it solves the following variational inequality: for all $u\in U^{\mathrm{ad}}$,

$$\left\langle \nabla K(u_{l+1}) - \nabla K(u_l) + \varepsilon_l(g_l+r_l)\,, u-u_{l+1}\right\rangle + \varepsilon_l(j^\Sigma(u,w_{l+1}) - j^\Sigma(u_{l+1},w_{l+1})) \ge 0\,. \tag{11.17}$$

Then, we have:

$$\begin{aligned}\ell_v(u_{l+1}) - \ell_v(u_l) = \underbrace{K(u_l) - K(u_{l+1}) - \left\langle \nabla K(u_l)\,, u_l - u_{l+1}\right\rangle}_{T_1} \\ + \underbrace{\left\langle \nabla K(u_l) - \nabla K(u_{l+1})\,, v-u_{l+1}\right\rangle}_{T_2}\,.\end{aligned}$$

[4] The set $\left\{\omega\in\Omega,\ \{\boldsymbol{R}_l(\omega)\}_{l\in\mathbb{N}} \text{ is unbounded}\right\}$ is negligible.

- By the convexity of K (A14), we get:

$$T_1 \leq 0 \,.$$

- The optimality condition (11.17) at $u = v$ implies:

$$\begin{aligned} T_2 &\leq \varepsilon_l \Big\langle g_l + r_l \,, v - u_{l+1} \Big\rangle + \varepsilon_l \Big(j^\Sigma(v, w_{l+1}) - j^\Sigma(u_{l+1}, w_{l+1}) \Big) \\ &\leq \varepsilon_l \Bigg(\underbrace{\Big\langle g_l \,, v - u_l \Big\rangle + j^\Sigma(v, w_{l+1}) - j^\Sigma(u_l, w_{l+1})}_{T_3} + \underbrace{\Big\langle r_l \,, v - u_l \Big\rangle}_{T_4} \\ &\qquad + \underbrace{\Big\langle g_l + r_l \,, u_l - u_{l+1} \Big\rangle + j^\Sigma(u_l, w_{l+1}) - j^\Sigma(u_{l+1}, w_{l+1})}_{T_5} \Bigg) \,. \end{aligned}$$

  - As $j^\Delta(\cdot, w_{l+1})$ is convex (A10), we get:

$$T_3 \leq \Big(j^\Delta + j^\Sigma \Big)(v, w_{l+1}) - \Big(j^\Delta + j^\Sigma \Big)(u_l, w_{l+1}) \,.$$

  - By Schwarz inequality, using $a \leq a^2 + 1$ for all $a \geq 0$ and the upper bound (11.16), we get:

$$\begin{aligned} T_4 &\leq \|r_l\| \|v - u_l\| \\ &\leq \|r_l\| \Big(\|v - u_l\|^2 + 1 \Big) \\ &\leq \|r_l\| + \frac{2}{b} \ell_v(u_l) \|r_l\| \,. \end{aligned}$$

  - The optimality condition (11.17) at $u = u_l$ and the strong monotonicity of ∇K, that arises from (A14), imply:

$$b \Big\| u_{l+1} - u_l \Big\|^2 \leq \varepsilon_l \Big(\langle g_l + r_l, u_l - u_{l+1} \rangle + j^\Sigma(u_l, w_{l+1}) - j^\Sigma(u_{l+1}, w_{l+1}) \Big) \,, \quad (11.18)$$

  where we recognize T_5 as the right-hand side of (11.18). Using the linearly bounded subgradient property of j^Σ (A12) with the technical result of Proposition E.4, we deduce that:

$$\begin{aligned} \Big| j^\Sigma(u_l, w_{l+1}) - j^\Sigma(u_{l+1}, w_{l+1}) \Big| &\leq \Big(d_1 \max \Big\{ \|u_l\|, \|u_{l+1}\| \Big\} + d_2 \Big) \|u_l - u_{l+1}\| \,, \\ &\leq \Big(d_1 \Big(\|u_l\| + \|u_{l+1}\| \Big) + d_2 \Big) \|u_l - u_{l+1}\| \,. \end{aligned}$$

  With Schwarz inequality on the first term of T_5, we have:

$$T_5 \leq \|g_l + r_l\| \|u_l - u_{l+1}\| + \Big(d_1 \|u_l\| + d_1 \|u_{l+1}\| + d_2 \Big) \|u_l - u_{l+1}\| \,.$$

  By the triangular inequality and Assumption (A12) for j^Δ, we deduce that there exist positive constants e_1, e_2 and e_3 such that:

$$T_5 \leq \Big(e_1 \|u_l\| + e_2 \|u_{l+1}\| + e_3 + \|r_l\| \Big) \|u_{l+1} - u_l\| \,.$$

  By the inequality (11.18), we then get:

$$\Big\| u_{l+1} - u_l \Big\| \leq \frac{\varepsilon_l}{b} \Big(e_1 \|u_l\| + e_2 \|u_{l+1}\| + e_3 + \|r_l\| \Big) \,, \quad (11.19)$$

and therefore by a repeated use of $(a+b)^2 \leq 2(a^2+b^2)$,

$$T_5 \leq \frac{\varepsilon_l}{b}\Big(e_1\|u_l\| + e_2\|u_{l+1}\| + e_3 + \|r_l\|\Big)^2 ,$$
$$\leq \frac{4\varepsilon_l}{b}\Big(e_1^2\,\|u_l\|^2 + e_2^2\,\|u_{l+1}\|^2 + e_3^2 + \|r_l\|^2\Big) .$$

Finally we bound $\|u_l\|$ (resp. $\|u_{l+1}\|$) by $\|u_l - v\| + \|v\|$ (resp. $\|u_{l+1} - v\| + \|v\|$) and we use (11.16) again to deduce that there exist four positive constants α, β, γ and δ such that:

$$T_5 \leq \varepsilon_l\Big(\alpha\ell_v(u_l) + \beta\ell_v(u_{l+1}) + \gamma + \delta\|r_l\|^2\Big) .$$

We collect the bounds we have obtained for T_1, T_3, T_4 and T_5 to get:

$$\ell_v(u_{l+1}) \leq \Big(1 + \alpha\varepsilon_l^2 + \frac{2}{b}\varepsilon_l\|r_l\|\Big)\ell_v(u_l) + \beta\varepsilon_l^2\ell_v(u_{l+1})$$
$$+ \Big(\gamma\varepsilon_l^2 + \varepsilon_l\|r_l\| + \delta(\varepsilon_l\|r_l\|)^2\Big)$$
$$+ \varepsilon_l\Big((j^\Delta + j^\Sigma)(v, w_{l+1}) - (j^\Delta + j^\Sigma)(u_l, w_{l+1})\Big) .$$

□

When no bias is present, $r_l = 0$, we retrieve the same inequality as in the PhD thesis of Culioli [Culioli, 1987, §2.5.1]. In the proofs of the subsequent theorems, Inequality (11.15) will be fundamental to derive boundedness properties or convergence results for the Lyapunov function ℓ_v, using variants of the Robbins-Siegmund theorem.

Now, we give convergence results for the stochastic APP algorithm, in terms of function values as well as for the iterates. The proof is similar to that in [Culioli, 1987, Culioli and Cohen, 1990] (case of a Hilbert space, no bias considered). The assumption that the Banach $\mathbb{U}$ is reflexive (A1) allows for a similar treatment as in the Hilbert case. The additional contribution of the bias is already taken care of by inequality (11.15).

Theorem 11.33. *Under Assumptions (A1)-(A7), (A9)-(A16), we have the following statements:*

- *The sequence of random variables* $\Big\{J(\boldsymbol{U}_l)\Big\}_{l\in\mathbb{N}}$ *converges to* $J^\sharp$ *almost surely.*
- *The sequence of iterates* $\Big\{\boldsymbol{U}_l\Big\}_{l\in\mathbb{N}}$ *of the stochastic APP algorithm is bounded almost surely and every weak cluster point of a bounded realization of this sequence belongs to the optimal set* $U^\sharp$.

Proof. Let $u^\sharp \in U^\sharp$ be a solution of Problem (11.1) and let $\{u_l\}_{l\in\mathbb{N}}$ be the sequence of iterates generated by Algorithm 9 with the realization $\{w_l\}_{l\in\mathbb{N}}$ of the stochastic process $\Big\{\boldsymbol{W}_l\Big\}_{l\in\mathbb{N}}$.

1. **Upper bound on the variation of the Lyapunov function.**

   Lemma 11.32 with $v = u^\sharp$ yields:

$$\ell_{u^\sharp}(u_{l+1}) \leq \Big(1 + \alpha\varepsilon_l^2 + \frac{2}{b}\varepsilon_l\|r_l\|\Big)\ell_{u^\sharp}(u_l) + \beta\varepsilon_l^2\ell_{u^\sharp}(u_{l+1})$$
$$+ \Big(\gamma\varepsilon_l^2 + \varepsilon_l\|r_l\| + \delta(\varepsilon_l\|r_l\|)^2\Big)$$
$$+ \varepsilon_l\Big((j^\Delta + j^\Sigma)(u^\sharp, w_{l+1}) - (j^\Delta + j^\Sigma)(u_l, w_{l+1})\Big) .$$

We write this inequality in terms of random variables and take the conditional expectation on both sides with respect to the σ-field $\mathcal{F}_l$ generated by the random variables $(\boldsymbol{W}_1, \dots, \boldsymbol{W}_l)$. By construction $\boldsymbol{U}_l$ is $\mathcal{F}_l$-measurable, so:

$$\mathbb{E}\Big(\ell_{u^\sharp}\left(\boldsymbol{U}_l\right) \;\Big|\; \mathcal{F}_l\Big) = \ell_{u^\sharp}\left(\boldsymbol{U}_l\right) \; .$$

The random variable $\boldsymbol{W}_{l+1}$ is independent of the past random variables $\left\{\boldsymbol{W}_k\right\}_{k\leq l}$ and therefore of $\boldsymbol{U}_l$, thus we have:

$$\mathbb{E}\Big((j^\Delta + j^\Sigma)(\boldsymbol{U}_l, \boldsymbol{W}_{l+1}) \;\Big|\; \mathcal{F}_l\Big) = (J^\Delta + J^\Sigma)(\boldsymbol{U}_l) = J(\boldsymbol{U}_l) \; .$$

We finally get:

$$\mathbb{E}\Big(\ell_{u^\sharp}\left(\boldsymbol{U}_{l+1}\right) \;\Big|\; \mathcal{F}_l\Big) \leq (1+\boldsymbol{\alpha}_l)\ell_{u^\sharp}\left(\boldsymbol{U}_l\right) + \boldsymbol{\beta}_l\mathbb{E}\Big(\ell_{u^\sharp}\left(\boldsymbol{U}_{l+1}\right) \;\Big|\; \mathcal{F}_l\Big) + \boldsymbol{\gamma}_l \\ - \varepsilon_l\Big(J(\boldsymbol{U}_l) - J(u^\sharp)\Big) \; ,$$

where we have:

$$\begin{aligned}
\boldsymbol{\alpha}_l &= \alpha\varepsilon_l^2 + \frac{2}{b}\varepsilon_l\mathbb{E}\Big(\|\boldsymbol{R}_l\| \;\Big|\; \mathcal{F}_l\Big) \; , \\
\boldsymbol{\beta}_l &= \beta\varepsilon_l^2 \; , \\
\boldsymbol{\gamma}_l &= \gamma\varepsilon_l^2 + \varepsilon_l\mathbb{E}\Big(\|\boldsymbol{R}_l\| \;\Big|\; \mathcal{F}_l\Big) + \delta\left(\varepsilon_l\mathbb{E}\Big(\|\boldsymbol{R}_l\| \;\Big|\; \mathcal{F}_l\Big)\right)^2 \; ,
\end{aligned}$$

By Assumptions (A15) and (A16), $\boldsymbol{\alpha}_l, \boldsymbol{\beta}_l$ and $\boldsymbol{\gamma}_l$ are the terms of convergent series. Recall that $J(\boldsymbol{U}_l) - J(u^\sharp)$ is almost surely nonnegative as $u^\sharp$ is solution of (11.1).

2. **Convergence analysis.** A direct application of Corollary E.3 of Robbins-Siegmund theorem, shows that the sequence of random variables $\left\{\ell_{u^\sharp}\left(\boldsymbol{U}_l\right)\right\}_{l\in\mathbb{N}}$ converges $\mathbb{P}$-a.s. to a random variable $\boldsymbol{\ell}_{u^\sharp}^\infty$ almost surely bounded,

$$\sum_{l=0}^{+\infty} \varepsilon_l\Big(J(\boldsymbol{U}_l) - J(u^\sharp)\Big) < +\infty \;\; \mathbb{P}\text{-a.s.} \; . \tag{11.20}$$

3. **Limits of sequences.** The sequence $\left\{\ell_{u^\sharp}(\boldsymbol{U}_l)\right\}_{l\in\mathbb{N}}$ is $\mathbb{P}$-a.s. bounded, so by the inequality (11.16), we get that the sequence $\left\{\boldsymbol{U}_l\right\}_{l\in\mathbb{N}}$ is also $\mathbb{P}$-a.s. bounded. Assumption (A12) then implies that the sequence $\left\{\boldsymbol{G}_l\right\}_{l\in\mathbb{N}}$ is also $\mathbb{P}$-a.s. bounded. Finally, as the sequence $\left\{\boldsymbol{R}_l\right\}_{l\in\mathbb{N}}$ is assumed to be $\mathbb{P}$-a.s. bounded (A16), we deduce from (11.19) that the sequence $\left\{\|\boldsymbol{U}_{l+1} - \boldsymbol{U}_l\|/\varepsilon_l\right\}_{l\in\mathbb{N}}$ is also $\mathbb{P}$-a.s. bounded. This last property ensures that Assumption (c) of Proposition E.6 is satisfied. Assumption (b) of Proposition E.6 is exactly (11.20) and Assumption (a) is satisfied as we have (A15). On a bounded set containing the sequence $\left\{\boldsymbol{U}_l\right\}_{l\in\mathbb{N}}$, for instance the convex hull of this sequence, the function J is Lipschitz continuous by Corollary E.5. This ensures the continuity assumption required to apply Proposition E.6. We conclude that $\left\{J(\boldsymbol{U}_l)\right\}_{l\in\mathbb{N}}$ converges almost surely to $J(u^\sharp) = J^\sharp$, the optimal value of Problem (11.1).

Let Ω_0 be the negligible subset of Ω on which the sequence $\left\{\ell_{u^\sharp}\left(\boldsymbol{U}_l\right)\right\}_{l\in\mathbb{N}}$ is unbounded and Ω_1 the negligible subset of Ω on which the relation (11.20) is not satisfied. We have $\mathbb{P}\big(\Omega_0 \cup \Omega_1\big) = 0$. Let $\omega \notin \Omega_0 \cup \Omega_1$. The sequence $\{u_l\}_{l\in\mathbb{N}}$ associated to this element ω is bounded and each u_l is in U^{ad}, a closed subset of $\mathbb{U}$. As $\mathbb{U}$ is reflexive (A1), there exists a weakly converging subsequence $\{u_{\xi(l)}\}_{l\in\mathbb{N}}$. Note that $\{\xi(l)\}_{l\in\mathbb{N}}$ depends on ω. Let $\overline{u}$ be the weak limit of the sequence $\{u_{\xi(l)}\}_{l\in\mathbb{N}}$. The function J is l.s.c. and convex, it is then weakly l.s.c. by [Ekeland and Temam, 1976, Corollary 2.2]. Thus we have:

$$J(\overline{u}) \leq \liminf_{l\to+\infty} J(u_{\xi(l)}) = J(u^\sharp) \ .$$

We conclude that $\overline{u} \in U^\sharp$. □

When the differential of K is weakly continuous, we can prove stronger convergence results for the sequence of iterates of the stochastic APP algorithm. These results already appear in [Culioli, 1987] and remain valid for our more general version of the algorithm.

Theorem 11.34. *Consider again (A1)-(A7), (A9)-(A16) and suppose that the differential of K is weakly continuous. Then, the sequence of iterates $\left\{\boldsymbol{U}_l\right\}_{l\in\mathbb{N}}$ converges weakly $\mathbb{P}$-a.s. to a single element of $U^\sharp$. If moreover, the function J^Δ is strongly convex, then, the sequence of iterates $\left\{\boldsymbol{U}_l\right\}_{l\in\mathbb{N}}$ converges strongly $\mathbb{P}$-a.s. to the unique solution $u^\sharp$ of Problem (11.1).*

Proof. Consider the case where the differential of K is weakly continuous. Let $\{u_l\}_{l\in\mathbb{N}}$ be a sequence of iterates generated by the algorithm. Suppose that there exist two subsequences $\{u_{\xi(l)}\}_{l\in\mathbb{N}}$ and $\{u_{\psi(l)}\}_{l\in\mathbb{N}}$ converging weakly respectively to two solutions $\overline{u}_\xi$ and $\overline{u}_\psi$ of the problem, with $\overline{u}_\xi \neq \overline{u}_\psi$. Then we have:

$$\begin{aligned} K(\overline{u}_\psi) - K(u_{\xi(l)}) - \left\langle \nabla K(u_{\xi(l)}) \, , \overline{u}_\psi - u_{\xi(l)} \right\rangle &= K(\overline{u}_\psi) - K(\overline{u}_\xi) - \left\langle \nabla K(u_{\xi(l)}) \, , \overline{u}_\psi - \overline{u}_\xi \right\rangle \\ &\quad + \left(K(\overline{u}_\xi) - K(u_{\xi(l)}) - \left\langle \nabla K(u_{\xi(l)}) \, , \overline{u}_\xi - u_{\xi(l)} \right\rangle \right) . \end{aligned}$$

By the point 2 of the proof of Theorem 11.33,

$$\begin{aligned} \lim_{l\to+\infty} K(\overline{u}_\psi) - K(u_{\xi(l)}) - \left\langle \nabla K(u_{\xi(l)}) \, , \overline{u}_\psi - u_{\xi(l)} \right\rangle &= \lim_{l\to+\infty} \ell_{\overline{u}_\psi}(u_l) = \ell_{\overline{u}_\psi} \ , \\ \lim_{l\to+\infty} K(\overline{u}_\xi) - K(u_{\xi(l)}) - \left\langle \nabla K(u_{\xi(l)}) \, , \overline{u}_\xi - u_{\xi(l)} \right\rangle &= \lim_{l\to+\infty} \ell_{\overline{u}_\xi}(u_l) = \ell_{\overline{u}_\xi} \ , \end{aligned}$$

therefore by weak continuity of the differential of K and strong convexity of K, we get:

$$\begin{aligned} \ell_{\overline{u}_\psi} - \ell_{\overline{u}_\xi} &= \lim_{l\to+\infty} K(\overline{u}_\psi) - K(\overline{u}_\xi) - \left\langle \nabla K(u_{\xi(l)}) \, , \overline{u}_\psi - \overline{u}_\xi \right\rangle , \\ &= K(\overline{u}_\psi) - K(\overline{u}_\xi) - \left\langle \nabla K(\overline{u}_\xi) \, , \overline{u}_\psi - \overline{u}_\xi \right\rangle , \\ &\geq \frac{b}{2} \|\overline{u}_\xi - \overline{u}_\psi\|^2 \ . \end{aligned}$$

Inverting the roles of $\overline{u}_\psi$ and $\overline{u}_\xi$, by a similar calculation as previously we get:

$$\ell_{\overline{u}_\xi} - \ell_{\overline{u}_\psi} \geq \frac{b}{2} \|\overline{u}_\xi - \overline{u}_\psi\|^2 \ ,$$

We then deduce that $\overline{u}_\xi = \overline{u}_\psi$, which contradicts the initial assumption. We conclude that all weakly converging subsequences of the sequence $\{u_l\}_{l\in\mathbb{N}}$ converge to the same limit, hence we have the weak convergence of the whole sequence $\{u_l\}_{l\in\mathbb{N}}$ to a single element of $U^\sharp$.

Now let us consider the case where J^Δ is strongly convex, with constant a. Then, Problem (11.1) admits a unique solution $u^\sharp$ which is characterized by the following variational inequality:

$$\exists r^\sharp \in \partial J^\Delta(u^\sharp),\ \forall u \in U^{\mathrm{ad}},\ \left\langle r^\sharp\,, u - u^\sharp \right\rangle + J^\Sigma(u) - J^\Sigma(u^\sharp) \geq 0\ .$$

The strong convexity assumption on J^Δ yields:

$$\begin{aligned} J(\boldsymbol{U}_l) - J(u^\sharp) &\geq \left\langle r^\sharp\,, \boldsymbol{U}_l - u^\sharp \right\rangle + \frac{a}{2}\|\boldsymbol{U}_l - u^\sharp\|^2 + J^\Sigma(\boldsymbol{U}_l) - J^\Sigma(u^\sharp)\ , \\ &\geq \frac{a}{2}\|\boldsymbol{U}_l - u^\sharp\|^2\ . \end{aligned}$$

As $\left\{J(\boldsymbol{U}_l)\right\}_{l\in\mathbb{N}}$ converges almost surely to $J(u^\sharp)$, we get that $\|\boldsymbol{U}_l - u^\sharp\|$ converges to zero. Thus, we have the strong convergence of the sequence $\left\{\boldsymbol{U}_l\right\}_{l\in\mathbb{N}}$ to the unique solution $u^\sharp$ of the problem. □

11.4.2 Efficiency estimates

In this section, we derive efficiency estimates for the convergence of the expectation of function values. In Theorem 11.36, we consider the expected function value taken for the averaged iterates following the technique of [Polyak and Juditsky, 1992, Ruppert, 1988]. We take a step size ε_l of the order $\mathcal{O}\left(l^{-\theta}\right)$ with $1/2 < \theta < 1$, ensuring the convergence of the algorithm, and leading to a better convergence rate than with a small step size $\varepsilon_l = \mathcal{O}\left(l^{-1}\right)$. The efficiency estimate is obtained using a similar technique as in [Nemirovski et al., 2009] but without requiring the boundedness of U^{ad}. Moreover, we are able to take into account the bias on the gradient with the following assumption, inspired from [Geiersbach and Wollner, 2019]:

(A17) For $l \in \mathbb{N}$, let $Q_l = \operatorname{ess\,sup}_{\omega\in\Omega}\left\|\boldsymbol{R}_l(\omega)\right\|$ be the essential supremum of $\left\|\boldsymbol{R}_l\right\|$ and assume that:

$$\sum_{l\in\mathbb{N}} Q_l\varepsilon_l < \infty\ .$$

We start by a lemma that proves the boundedness of the expectation of the Lyapunov function. This result will be used multiple times in this section.

Lemma 11.35. *Under Assumptions (A10), (A12), (A14), (A15) and (A17), the sequence of expectations of the Lyapunov function* $\left\{\mathbb{E}\left(\ell_{u^\sharp}\left(\boldsymbol{U}_l\right)\right)\right\}_{l\in\mathbb{N}}$ *is bounded.*

Proof. We start from Lemma 11.32 with $v = u^\sharp$ where we use $\|r_l\| \leq Q_l$ and then take the full expectation. This yields:

$$\mathbb{E}\left(\ell_{u^\sharp}\left(\boldsymbol{U}_{l+1}\right)\right) \leq (1+\alpha_l)\mathbb{E}\left(\ell_{u^\sharp}\left(\boldsymbol{U}_l\right)\right) + \beta_l\mathbb{E}\left(\ell_{u^\sharp}\left(\boldsymbol{U}_{l+1}\right)\right) + \gamma_l - \varepsilon_l\mathbb{E}\left(J\left(\boldsymbol{U}_l\right) - J\left(u^\sharp\right)\right)\ , \tag{11.21}$$

where:

$$\alpha_l = \alpha\varepsilon_l^2 + \frac{2}{b}\varepsilon_l Q_l, \quad \beta_l = \beta\varepsilon_l^2, \quad \gamma_l = (\gamma + \delta Q_l^2)\varepsilon_l^2 + Q_l\varepsilon_l\ ,$$

From (A15) and (A17), α_l, β_l and γ_l are the terms of convergent series. Using a deterministic version of Corollary E.3, we get that the sequence $\left\{\mathbb{E}\left(\ell_{u^\sharp}\left(\boldsymbol{U}_l\right)\right)\right\}_{l\in\mathbb{N}}$ converges and is therefore bounded. □

Theorem 11.36. *Suppose that Assumptions (A1)-(A7), (A9)-(A17) are satisfied. Let $n \in \mathbb{N}$ and let $\left\{\boldsymbol{U}_l\right\}_{l\in\mathbb{N}}$ be the sequence of iterates of the stochastic APP algorithm. Define the averaged iterate as:*

$$\widetilde{\boldsymbol{U}}_i^n = \sum_{l=i}^{n} \eta_l^i \boldsymbol{U}_l \quad \text{with} \quad \eta_l^i = \frac{\varepsilon_l}{\sum_{p=i}^{n} \varepsilon_p} \, .$$

Suppose that for all $l \in \mathbb{N}$, $\varepsilon_l = cl^{-\theta}$ with $1/2 < \theta < 1$ and a constant $c > 0$. Then for any minimizer $u^\sharp$ of J, we have:

$$\mathbb{E}\left(J\left(\widetilde{\boldsymbol{U}}_1^n\right) - J\left(u^\sharp\right)\right) = \mathcal{O}\left(n^{\theta-1}\right) \, .$$

In particular, the rate of convergence can be arbitrarily close to the order $n^{-1/2}$ if θ is chosen to be arbitrarily close to $1/2$.

Proof. From Lemma 11.35, we get that inequality (11.21) is satisfied and the sequence $\left\{\mathbb{E}\left(\ell_{u^\sharp}\left(\boldsymbol{U}_l\right)\right)\right\}_{l\in\mathbb{N}}$ is bounded. Then, there exists a constant $M \geq 0$ such that $\mathbb{E}\left(\ell_{u^\sharp}\left(\boldsymbol{U}_l\right)\right) \leq M$ for all $l \in \mathbb{N}$. Summing (11.21) over $i \leq l \leq n$ and using $\mathbb{E}\left(\ell_{u^\sharp}\left(\boldsymbol{U}_l\right)\right) \leq M$, we get:

$$\sum_{l=i}^{n} \varepsilon_l \mathbb{E}\left(J\left(\boldsymbol{U}_l\right) - J\left(u^\sharp\right)\right) \leq \sum_{l=i}^{n} \left(M(\alpha+\beta) + \gamma + \delta Q_l^2\right)\varepsilon_l^2 + \left(\frac{2}{b}M + 1\right) Q_l \varepsilon_l \, .$$

In the sequel, let $R = M(\alpha+\beta)+\gamma$ and $S = \frac{2}{b}M + 1$. By convexity of J, we get:

$$\mathbb{E}\left(J\left(\widetilde{\boldsymbol{U}}_i^n\right) - J(u^\sharp)\right) \leq \frac{\sum_{l=i}^{n}\left(R + \delta Q_l^2\right)\varepsilon_l^2 + S Q_l \varepsilon_l}{\sum_{l=i}^{n} \varepsilon_l} \, .$$

We have $\varepsilon_l = cl^{-\theta}$ with $1/2 < \theta < 1$ and:

$$\sum_{l=1}^{n} l^{-\theta} \geq \frac{(n+1)^{1-\theta} - 1}{1-\theta} \geq \widetilde{C}_\theta n^{1-\theta} \, ,$$

for some $\widetilde{C}_\theta > 0$. Moreover, from (A15) and (A17), ε_l^2, $Q_l\varepsilon_l$ and $Q_l^2\varepsilon_l^2$ are the terms of convergent series. Thus, there exists a constant $C_\theta > 0$ such that:

$$\mathbb{E}\left(J\left(\widetilde{\boldsymbol{U}}_1^n\right) - J(u^\sharp)\right) \leq \frac{C_\theta}{n^{1-\theta}} \, ,$$

which gives the desired rate of convergence. □

Theorem 11.36 proves a convergence rate of order $\mathcal{O}\left(n^{\theta-1}\right)$ for the stochastic APP algorithm without assuming strong convexity of the objective. This rate appears for stochastic gradient descent in [Bach and Moulines, 2011] where it is stated that the combination of large step sizes of order $\mathcal{O}\left(n^{-\theta}\right)$ with $1/2 < \theta < 1$, together with averaging lead to the best convergence behavior. A similar rate is also given for stochastic proximal gradient in [Rosasco et al., 2019].

In the following theorem, we show that this rate also holds when we consider the expected function value taken at the last iterate $\boldsymbol{U}_n$ instead of the averaged iterate $\widetilde{\boldsymbol{U}}_1^n$. Using the concept of modified Fejér monotone sequences, the authors of [Lin et al., 2018] have been able to give convergence rates of the expected function value of the last iterate for many algorithms, such as the projected subgradient method or the proximal gradient algorithm. The idea of modified Fejér sequence is adapted to the stochastic case in [Rosasco et al., 2019, Theorem 3.1]. We further adapt this concept for the stochastic APP algorithm which allows to derive a convergence rate for the expected function value of the last iterate.

Theorem 11.37. *Suppose that Assumptions (A1)-(A7), (A9)-(A17) are satisfied. Let $n \in \mathbb{N}$ and let $\left\{\boldsymbol{U}_l\right\}_{l\in\mathbb{N}}$ be the sequence of the iterates of the stochastic APP algorithm. Suppose that for all $l \in \mathbb{N}$, $\varepsilon_l = cl^{-\theta}$ with $1/2 < \theta < 1$ and a constant $c > 0$ and that $Q_l \leq ql^{-\nu}$ for $\nu > 1 - \theta$ and a constant $q > 0$. Then, for any minimizer $u^\sharp$ of J we have:*

$$\mathbb{E}\left(J\left(\boldsymbol{U}_n\right) - J\left(u^\sharp\right)\right) = \mathcal{O}\left(n^{\theta-1}\right) .$$

In particular, the rate of convergence can be arbitrarily close to the order $n^{-1/2}$ if θ is chosen to be arbitrarily close to $1/2$.

Proof. Let $a_l = \mathbb{E}\left(J\left(\boldsymbol{U}_l\right) - J\left(u^\sharp\right)\right)$, from Lemma E.1, we can write:

$$\varepsilon_n a_n = \frac{1}{n}\sum_{l=1}^{n}\varepsilon_l a_l + \sum_{i=1}^{n-1}\frac{1}{i(i+1)}\left(\sum_{l=n-i+1}^{n}\varepsilon_l a_l - i\varepsilon_{n-i}a_{n-i}\right) .$$

We have:

$$\begin{aligned}\frac{1}{i(i+1)}\left(\sum_{l=n-i+1}^{n}\varepsilon_l a_l - i\varepsilon_{n-i}a_{n-i}\right) &= \frac{1}{i(i+1)}\sum_{l=n-i+1}^{n}\varepsilon_l\mathbb{E}\left(J\left(\boldsymbol{U}_l\right) - J\left(\boldsymbol{U}_{n-i}\right)\right) \\ &\quad + \frac{1}{i+1}\left(\frac{1}{i}\sum_{l=n-i+1}^{n}\varepsilon_l - \varepsilon_{n-i}\right)a_{n-i} .\end{aligned}$$

By choice of ε_l, the sequence $\{\varepsilon_l\}_{l\in\mathbb{N}}$ is decreasing so,

$$\frac{1}{i}\sum_{l=n-i+1}^{n}\varepsilon_l - \varepsilon_{n-i} \leq 0 .$$

Moreover, by optimality of $u^\sharp$, we have $a_{n-i} \geq 0$ so,

$$\varepsilon_n a_n \leq \frac{1}{n}\sum_{l=1}^{n}\varepsilon_l a_l + \sum_{i=1}^{n-1}\frac{1}{i(i+1)}\sum_{l=n-i+1}^{n}\varepsilon_l\mathbb{E}\left(J\left(\boldsymbol{U}_l\right) - J\left(\boldsymbol{U}_{n-i}\right)\right) .$$

Again by optimality of $u^\sharp$, we have $\mathbb{E}\left(J\left(\boldsymbol{U}_l\right) - J\left(\boldsymbol{U}_{n-i}\right)\right) \leq \mathbb{E}\left(J\left(\boldsymbol{U}_l\right) - J\left(u^\sharp\right)\right) = a_l$. This yields:

$$\varepsilon_n a_n \leq \frac{1}{n}\sum_{l=1}^{n}\varepsilon_l a_l + \sum_{i=1}^{n-1}\frac{1}{i(i+1)}\sum_{l=n-i+1}^{n}\varepsilon_l a_l . \tag{11.22}$$

From Lemma 11.35, inequality (11.21) is satisfied and there exists a constant $M \geq 0$ such that $\mathbb{E}\left(\ell_{u^\sharp}\left(\boldsymbol{U}_l\right)\right) \leq M$ for all $l \in \mathbb{N}$. Using this bound into (11.21) and summing over $j \leq l \leq n$, we get:

$$\sum_{l=j}^{n} \varepsilon_l a_l \leq \sum_{l=j}^{n} M(\alpha_l + \beta_l) + \gamma_l \ . \tag{11.23}$$

Taking $j = 1$ or $j = n - i + 1$ in (11.23) allows to bound both terms in (11.22). Define:

$$\bar{\alpha}_l = \alpha\varepsilon_l^2 + \frac{2}{b}\varepsilon_l q l^{-\nu}, \quad \bar{\gamma}_l = \left(\gamma + \delta q^2 l^{-2\nu}\right)\varepsilon_l^2 + \varepsilon_l q l^{-\nu} \ ,$$

As $Q_l \leq q l^{-\nu}$, we have $\alpha_l \leq \bar{\alpha}_l$ and $\gamma_l \leq \bar{\gamma}_l$. We let $\xi_l = M(\bar{\alpha}_l + \beta_l) + \bar{\gamma}_l$, so that we have:

$$\sum_{l=1}^{n} \varepsilon_l a_l \leq \sum_{l=1}^{n} \xi_l \quad \text{and} \quad \sum_{i=1}^{n-1} \frac{1}{i(i+1)} \sum_{l=n-i+1}^{n} \varepsilon_l a_l \leq \sum_{i=1}^{n-1} \frac{1}{i(i+1)} \sum_{l=n-i+1}^{n} \xi_l \ . \tag{11.24}$$

Exchanging the order in the sum yields:

$$\sum_{i=1}^{n-1} \frac{1}{i(i+1)} \sum_{l=n-i+1}^{n} \xi_l = \sum_{l=2}^{n} \sum_{i=n-l+1}^{n-1} \left(\frac{1}{i} - \frac{1}{i+1}\right)\xi_l = \sum_{l=2}^{n} \frac{1}{n-l+1}\xi_l - \frac{1}{n}\sum_{l=2}^{n} \xi_l \ . \tag{11.25}$$

Plugging (11.24) and (11.25) into (11.22), we get:

$$\varepsilon_n a_n \leq \frac{1}{n}\sum_{l=1}^{n} \xi_l + \sum_{l=2}^{n} \frac{1}{n-l+1}\xi_l - \frac{1}{n}\sum_{l=2}^{n} \xi_l = \sum_{l=1}^{n} \frac{1}{n-l+1}\xi_l \ .$$

From the assumptions on ε_l, $\{\xi_l\}_{l\in\mathbb{N}}$ is non-increasing. Thus,

$$\begin{aligned}\sum_{l=1}^{n} \frac{1}{n-l+1}\xi_l &\leq \xi_{\lfloor \frac{n}{2}+1 \rfloor} \sum_{n/2+1 \leq l \leq n} \frac{1}{n-l+1} + \frac{2}{n} \sum_{1 \leq l < n/2+1} \xi_l \ , \\ &\leq \xi_{\lfloor \frac{n}{2}+1 \rfloor} \left(\log\left(\frac{n}{2}\right) + 1\right) + \frac{2}{n}\sum_{l=1}^{n} \xi_l \ .\end{aligned}$$

Hence,

$$a_n \leq \frac{\xi_{\lfloor \frac{n}{2}+1 \rfloor}}{\varepsilon_n}\left(\log\left(\frac{n}{2}\right) + 1\right) + \frac{2}{n\varepsilon_n}\sum_{l=1}^{n} \xi_l \ .$$

Recall that,

$$\xi_l = \bar{M}(\bar{\alpha}_l + \beta_l) + \bar{\gamma}_l = \left(\bar{M}(\alpha+\beta) + \gamma + \delta q^2 l^{-2\nu}\right) c^2 l^{-2\theta} + \left(\frac{2}{b}\bar{M} + 1\right) c q l^{-(\nu+\theta)} \leq \xi l^{-\mu} \ ,$$

for $\mu = \min\{2\theta, \nu + \theta\}$ and some constant $\xi > 0$ so that,

$$a_n \leq 2^\mu \frac{\xi}{c} n^{\theta-\mu}\left(\log\left(\frac{n}{2}\right) + 1\right) + 2\frac{\xi}{c} n^{\theta-1} \sum_{l=1}^{n} l^{-\mu} \ .$$

As $\theta > 1/2$ and $\nu > 1 - \theta$, we have $\mu > 1$ so,

$$\sum_{l=1}^{n} l^{-\mu} \leq \frac{\mu}{\mu - 1} \ .$$

Thus, noting that $\theta - \mu < \theta - 1$, we have:

$$a_n \leq 2^\mu \frac{\xi}{c} n^{\theta-\mu}\left(\log\left(\frac{n}{2}\right) + 1\right) + \frac{2\mu\xi}{c(\mu-1)} n^{\theta-1} = \mathcal{O}\left(n^{\theta-1}\right) \ .$$

This concludes the proof. □

Remark 11.38. The inequality (11.21) (which holds in fact for any $u \in U^{\mathrm{ad}}$ in place of $u^\sharp$) is the counterpart of modified Fejér monotonicity [Lin et al., 2018]. The main differences are that (11.21) involves a Bregman divergence instead of the Euclidean distance. Moreover, there are coefficients $\alpha_l, \beta_l > 0$ that slightly degrade the inequality compared to what we obtain with Fejér monotone sequences where $\alpha_l = \beta_l = 0$. The summability of α_l and β_l in addition with the boundedness of the expectation of the Bregman divergence $\left\{\mathbb{E}\left(\ell_{u^\sharp}\left(\boldsymbol{U}_l\right)\right)\right\}_{l\in\mathbb{N}}$ allow us to proceed in the same way as in [Lin et al., 2018, Rosasco et al., 2019] to get the convergence rate of Theorem 11.37. ◊

11.5 Conclusion

We have studied the stochastic APP algorithm in a Banach case. This framework generalizes many stochastic optimization algorithms. We have proved the measurability of the iterates of the algorithm, hence filling a theoretical gap to ensure that the quantities we manipulate when deriving efficiency estimates are well-defined. We have shown the convergence of the stochastic APP algorithm in the case where a bias on the gradient is considered. Finally, efficiency estimates are derived while taking the bias into account. Assuming a sufficiently fast decay of this bias, we get a convergence rate for the expectation of the function values that is similar to that of well-known stochastic optimization algorithms when no bias is present, such as stochastic gradient descent [Bach and Moulines, 2011], stochastic mirror descent [Nemirovski et al., 2009] or the stochastic proximal gradient algorithm [Rosasco et al., 2019].

General conclusion and perspectives

This thesis is driven by the industrial issue of maintenance scheduling optimization for components of hydroelectric power plants that share a common stock of spare parts.

- In the first part, we consider blackbox optimization algorithms that can be easily coupled to VME, the software used at EDF for the evaluation of the cost of maintenance strategies. We present an in-depth review of a kriging-based algorithm, EGO, and of a direct search method, MADS, both from a theoretical and a practical point of view. To guide the choice of the solver for the maximization of the EI within EGO, we carry out a benchmark that provides a quantitative assessment of a large number of solvers for this particular task. Then, we propose a variant of EGO, called EGO-FSSF, where we use a sequential initial design with metamodel validation instead of a fixed-size initial design. Thus, the evaluation budget allocated to the initial design step can be adapted to the difficulty of the optimization problem. The algorithms EGO, EGO-FSSF and MADS are launched on the COCO benchmark for a comparaison of their performance on a large variety of blackbox optimization problems. MADS reveals to be the most competitive algorithm in general, and EGO-FSSF is slightly better than EGO. We choose to plug MADS and EGO-FSSF on VME in order to solve small industrial maintenance optimization cases, ranging from 2 to 10 components, and considering only periodic maintenance strategies. MADS is more and more efficient relatively to EGO as the number of components increases. Moreover, MADS is much faster than EGO. These small industrial cases show the limitations of the blackbox approach: MADS may be trapped in a local optimum that is not of good quality as the exploration of a high-dimensional space is difficult to achieve. However, we mention that on small systems and for common maintenance operations, such as lubrication of the components, looking for periodic strategies is sufficient. In this case, blackbox methods are adapted for an industrial application.

- In the second part, we tackle the optimal maintenance scheduling problem for large-scale systems while considering more general maintenance strategies, where one maintenance decision can be taken each year for each component. To do so, we go out from the blackbox framework and develop an analytical model of the industrial system of interest. Then, we formulate an explicit stochastic optimal control problem that reflects the industrial needs. To solve the large-scale maintenance optimization problem, we design a decomposition by prediction of the problem, called *decomposition by component*. We review the principles of the APP, which is the general framework that is at the heart of the decomposition methodology. The APP turns the resolution of a large-scale optimization prob-

lem into the iterative resolution of a sequence of subproblems of smaller size. The decomposition by prediction is implemented through a fixed-point algorithm that is first tested on synthetic cases. These synthetic cases have been designed so as to have a similar structure as the industrial problem and so that the application of the fixed-point algorithm is straightforward. The numerical experiments show that the decomposition by prediction is very efficient on the synthetic cases. The APP cannot be applied directly for the industrial problem because some integer variables are present in the model. Therefore, we design a relaxation of the system. After a careful tuning of the algorithm, the decomposition methodology is applied on a large-scale maintenance optimization problem with 80 components. The decomposition leads to a gain of 11% over the current reference algorithm, which represents more than $1,4$M€. The decomposition manages to design an efficient maintenance strategy by exploiting the fact that the components are new at the beginning of the time horizon and that the discount rate makes failures not too penalizing at the end of the horizon.

- Finally, in the third part, we focus on theoretical aspects regarding the stochastic APP algorithm in a Banach space. We prove the measurability of the iterates, which is essential to ensure that the convergence results or the efficiency estimates, given in terms of random variables, are well-defined. Then, we extend convergence results from the Hilbert case to the Banach case. Finally, we derive efficiency estimates for the function value taken at the averaged sequence of iterates and also at the last iterate.

This work opens up several research perspectives:

- Regarding blackbox optimization algorithms, we could consider to combine EGO and MADS to make the most of these algorithms that have complementary strengths: an efficient exploratory behavior for EGO and a guarantee of convergence towards a local minimum for MADS. We suggest several ideas:

  1. EGO could be used within the search step of MADS. This would give MADS a more exploratory behavior, hence improving the weakness that has been pointed out in the numerical experiments of Sections 6.3 and 10.8. This idea has been explored by [Talgorn et al., 2015].
  2. The numerical results of Section 6.3 suggest that EGO is more efficient than MADS in the first iterations whereas MADS manages to steadily improve the value of the objective function. We can then design an algorithm where a first batch of iterations is done with EGO before moving on to MADS. This idea is similar to that of adding a local optimizer at the end of EGO (§3.3.4), except that using MADS still preserves an exploratory behavior in the algorithm.
  3. On the other way around, in [Munoz Zuniga and Sinoquet, 2020], the authors use MADS as the solver for EI maximization within EGO.

- From an operational perspective, it may be worth to apply the decomposition methodology on large systems with a search space restricted to periodic maintenance strategies. The computation time will be greatly reduced compared to the problem tackled in Chapter 10 and the returned strategy may still be efficient.

- When modeling the industrial system in Chapter 8, we have considered that no cost is incurred by the stock and that the spare part management strategy is fixed. It is possible to consider a more complete model, that considers a holding cost for the spare parts and a control for the dates of order of the spare parts. In Chapter 8, we have also assumed that there is no dependence between the failures of the components, and we can lift this restriction in a more realistic model. Introducing a control on the stock or a dependence between the components lead respectively to an increase of the dimension of the search space and to a more complicated dynamics, which makes the optimization problem more challenging. A balance must be found between the operational needs for a realistic model and the difficulty of the optimization.

- In the thesis, we optimize the expected LCC but a decision maker often looks at the robustness of the maintenance strategies. In Section 10.8, we compute the distribution of the LCC of the optimal maintenance strategy *a posteriori* in order to assess its robustness. An interesting direction for future research would be to perform the optimization with respect to a risk criterion on the LCC, for example the conditional value at risk. We can look at [Ruszczyński and Shapiro, 2006] for a reference on the optimization of risk measures.

References

[Abramowitz and Stegun, 1964] Abramowitz, M. and Stegun, A. (1964). *Handbook of Mathematical Functions With Formulas, Graphs, and Mathematical Tables*. (p. 35)

[Abramson et al., 2009] Abramson, M. A., Audet, C., Dennis Jr, J. E., and Le Digabel, S. (2009). OrthoMADS: A deterministic MADS instance with orthogonal directions. *SIAM Journal on Optimization*, 20(2):948–966. (p. 57, 58)

[Abtini, 2018] Abtini, M. (2018). *Plans prédictifs à taille fixe et séquentiels pour le krigeage*. PhD thesis, Université de Lyon. (p. 31)

[Alarie et al., 2019] Alarie, S., Audet, C., Bouchet, P.-Y., and Digabel, S. L. (2019). Optimization of noisy blackboxes with adaptive precision. *arXiv:1911.05846 [math]*. (p. 6, 16, 72)

[Almakhlafi and Knowles, 2012] Almakhlafi, A. and Knowles, J. (2012). Benchmarks for maintenance scheduling problems in power generation. In *2012 IEEE Congress on Evolutionary Computation*, pages 1–8, Brisbane, Australia. (p. 7, 16)

[Alrabghi and Tiwari, 2015] Alrabghi, A. and Tiwari, A. (2015). State of the art in simulation-based optimisation for maintenance systems. *Computers & Industrial Engineering*, 82:167–182. (p. 16)

[Alrabghi et al., 2013] Alrabghi, A., Tiwari, A., and Alabdulkarim, A. (2013). Simulation based optimization of joint maintenance and inventory for multi-components manufacturing systems. In *Proceedings of the 2013 Winter Simulation Conference*, pages 1109–1119. (p. 6)

[Aoudjit, 2010] Aoudjit, H. (2010). *Planification de la maintenance d'un parc de turbines-alternateurs par programmation mathématique*. PhD thesis, Université de Montréal. (p. 72)

[Arrow and Hurwicz, 1960] Arrow, K. J. and Hurwicz, L. (1960). *Decentralization and Computation in Resource Allocation*. Stanford University, Department of Economics. (p. 93)

[Atchade et al., 2017] Atchade, Y. F., Fort, G., and Moulines, E. (2017). On Perturbed Proximal Gradient Algorithms. *Journal of Machine Learning Research*, 18:1–33. (p. 152)

[Audet, 2004] Audet, C. (2004). Convergence Results for Generalized Pattern Search Algorithms are Tight. *Optimization and Engineering*, 5(2):101–122. (p. 54)

[Audet et al., 2008a] Audet, C., Béchard, V., and Digabel, S. L. (2008a). Nonsmooth optimization through Mesh Adaptive Direct Search and Variable Neighborhood Search. *Journal of Global Optimization*, 41(2):299–318. (p. 87)

[Audet et al., 2008b] Audet, C., Dennis, J. E., and Le Digabel, S. (2008b). Parallel Space Decomposition of the Mesh Adaptive Direct Search Algorithm. *SIAM Journal on Optimization*, 19(3):1150–1170. (p. 87, 143)

[Audet and Dennis, 2002] Audet, C. and Dennis, Jr, J. E. (2002). Analysis of Generalized Pattern Searches. *SIAM Journal on Optimization*, 13(3):889–903. (p. 48, 50, 52, 54)

[Audet and Dennis, 2006] Audet, C. and Dennis, Jr, J. E. (2006). Mesh Adaptive Direct Search Algorithms for Constrained Optimization. *SIAM Journal on Optimization*, 17(1):188–217. (p. 8, 18, 22, 53, 54, 56, 57, 115, 134)

[Audet and Hare, 2017] Audet, C. and Hare, W. (2017). *Derivative-Free and Blackbox Optimization*. Springer Series in Operations Research and Financial Engineering. Springer International Publishing, Cham. (p. 21)

[Bach and Moulines, 2011] Bach, F. and Moulines, E. (2011). Non-Asymptotic Analysis of Stochastic Approximation Algorithms for Machine Learning. In *Advances in Neural Information Processing Systems*, pages 451–459. (p. 151, 172, 175)

[Bachoc, 2013] Bachoc, F. (2013). Cross Validation and Maximum Likelihood estimations of hyper-parameters of Gaussian processes with model misspecification. *Computational Statistics & Data Analysis*, 66:55–69. (p. 39)

[Barty et al., 2010] Barty, K., Carpentier, P., and Girardeau, P. (2010). Decomposition of large-scale stochastic optimal control problems. *RAIRO - Operations Research*, 44(3):167–183. (p. 93)

[Baudin et al., 2017] Baudin, M., Dutfoy, A., Iooss, B., and Popelin, A.-L. (2017). Open TURNS: An industrial software for uncertainty quantification in simulation. In *Handbook of uncertainty quantification*, page 46. (p. 36, 66, 76)

[Bauschke and Combettes, 2011] Bauschke, H. H. and Combettes, P. L. (2011). *Convex analysis and monotone operator theory in Hilbert spaces*. CMS books in mathematics. Springer, New York. (p. 151, 191)

[Benveniste et al., 2012] Benveniste, A., Metivier, M., Priouret, P., and Wilson, S. S. (2012). *Adaptive algorithms and stochastic approximations*. Number 22 in Stochastic modelling and applied probability. Springer-Verl, Berlin. (p. 151)

[Bertsekas and Shreve, 1996] Bertsekas, D. P. and Shreve, S. E. (1996). *Stochastic Optimal Control: The Discrete-Time Case*. Athena Scientific. (p. 89, 158)

[Billingsley, 1995] Billingsley, P. (1995). *Probability and measure*. Wiley series in probability and mathematical statistics. Wiley, New York, 3rd edition. (p. 151, 193)

[Bittar et al., 2020] Bittar, T., Carpentier, P., Chancelier, J.-P., and Lonchampt, J. (2020). A Decomposition Method by Interaction Prediction for the Optimization of Maintenance Scheduling. *arXiv:2002.10719 [math]*. (Submitted to Annals of Operations Research). (p. 91)

[Bittar et al., 2021] Bittar, T., Carpentier, P., Chancelier, J.-P., and Lonchampt, J. (2021). The stochastic Auxiliary Problem Principle in Banach spaces: measurability and convergence. *arXiv:2101.08073 [math]*. (Submitted to SIAM Journal on Optimization). (p. 149)

[Booker et al., 1999] Booker, A. J., Dennis, J. E., Frank, P. D., Serafini, D. B., Torczon, V., and Trosset, M. W. (1999). A rigorous framework for optimization of expensive functions by surrogates. *Structural optimization*, 17(1):1–13. (p. 50)

[Bregman, 1967] Bregman, L. M. (1967). The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming. *USSR Computational Mathematics and Mathematical Physics*, 7(3):200–217. (p. 152)

[Brochu et al., 2010] Brochu, E., Cora, V. M., and de Freitas, N. (2010). A Tutorial on Bayesian Optimization of Expensive Cost Functions, with Application to Active User Modeling and Hierarchical Reinforcement Learning. *arXiv:1012.2599 [cs]*. (p. 65)

[Brézis, 2005] Brézis, H. (2005). *Analyse fonctionnelle: Théorie et applications*. Mathématiques appliquées pour la maîtrise. Dunod. (p. 162)

[Bubeck, 2015] Bubeck, S. (2015). Convex Optimization: Algorithms and Complexity. *Foundations and Trends® in Machine Learning*, 8(3-4):231–357. (p. 152)

[Bull, 2011] Bull, A. D. (2011). Convergence rates of efficient global optimization algorithms. *Journal of Machine Learning Research*, 12(10):2879–2904. (p. 44)

[Campolongo et al., 2007] Campolongo, F., Cariboni, J., and Saltelli, A. (2007). An effective screening design for sensitivity analysis of large models. *Environmental Modelling & Software*, 22(10):1509–1518. (p. 141, 142)

[Carpentier and Cohen, 2017] Carpentier, P. and Cohen, G. (2017). *Décomposition-coordination en optimisation déterministe et stochastique*, volume 81 of *Mathématiques et Applications*. Springer Berlin Heidelberg. (p. 93, 94, 95, 96, 149)

[Carpentier et al., 2015] Carpentier, P., Cohen, G., Chancelier, J.-P., and De Lara, M. (2015). *Stochastic Multi-Stage Optimization*. Number 75 in Probability Theory and Stochastic Modelling. Springer. (p. 89)

[Castaing and Valadier, 1977] Castaing, C. and Valadier, M. (1977). *Convex Analysis and Measurable Multifunctions*, volume 580 of *Lecture Notes in Mathematics*. Springer Berlin Heidelberg. (p. 153, 156, 157, 158)

[Cho and Parlar, 1991] Cho, D. I. and Parlar, M. (1991). A survey of maintenance models for multi-unit systems. *European Journal of Operational Research*, 51(1):1–23. (p. 6, 16)

[Clarke, 1990] Clarke, F. H. (1990). *Optimization and nonsmooth analysis*. Number 5 in Classics in Applied Mathematics. Society for Industrial and Applied Mathematics. (p. 53)

[Cohen, 1978] Cohen, G. (1978). Optimization by decomposition and coordination: A unified approach. *IEEE Transactions on Automatic Control*, 23(2):222–232. (p. 93, 154)

[Cohen, 1980] Cohen, G. (1980). Auxiliary problem principle and decomposition of optimization problems. *Journal of Optimization Theory and Applications*, 32(3):277–305. (p. 90, 97)

[Conn and Le Digabel, 2013] Conn, A. R. and Le Digabel, S. (2013). Use of quadratic models with mesh-adaptive direct search for constrained black box optimization. *Optimization Methods and Software*, 28(1):139–158. (p. 77, 86)

[Conn et al., 2009] Conn, A. R., Scheinberg, K., and Vicente, L. N. (2009). *Introduction to Derivative-Free Optimization*. Society for Industrial and Applied Mathematics. (p. 22)

[Coope and Price, 2000] Coope, I. D. and Price, C. J. (2000). Frame based methods for unconstrained optimization. *Journal of Optimization Theory and Applications*, 107(2):261–274. (p. 55)

[Cox and John, 1992] Cox, D. D. and John, S. (1992). A statistical method for global optimization. In *Proceedings of the 1992 IEEE International Conference on Systems, Man, and Cybernetics*, volume 2, pages 1241–1246. (p. 42)

[Culioli, 1987] Culioli, J.-C. (1987). *Algorithmes de décomposition/coordination en optimisation stochastique*. PhD thesis, Ecole Nationale Supérieure des Mines de Paris. (p. 165, 168, 170)

[Culioli and Cohen, 1990] Culioli, J.-C. and Cohen, G. (1990). Decomposition/Coordination Algorithms in Stochastic Optimization. *SIAM Journal on Control and Optimization*, 28(6):1372–1403. (p. 93, 149, 151, 153, 154, 165, 168)

[Damblin et al., 2013] Damblin, G., Couplet, M., and Iooss, B. (2013). Numerical studies of space-filling designs: optimization of Latin Hypercube Samples and subprojection properties. *Journal of Simulation*, 7(4):276–289. (p. 30, 31, 138)

[Davidon, 1991] Davidon, W. C. (1991). Variable Metric Method for Minimization. *SIAM Journal on Optimization*, 1(1):1–17. (p. 47, 52)

[Dekker, 1996] Dekker, R. (1996). Applications of maintenance optimization models: a review and analysis. *Reliability Engineering & System Safety*, 51(3):229–240. (p. 6, 16)

[Demgne, 2015] Demgne, J. (2015). *Modélisation d'actifs industriels pour l'optimisation robuste des stratégies de maintenance*. PhD thesis, Université de Pau et des Pays de l'Adour. (p. 7, 17)

[Ding and Kamaruddin, 2015] Ding, S.-H. and Kamaruddin, S. (2015). Maintenance policy optimization—literature review and directions. *The International Journal of Advanced Manufacturing Technology*, 76(5-8):1263–1283. (p. 6, 16)

[Eaton, 1983] Eaton, M. L. (1983). Chapter 3: The normal distribution on a vector space. In *Multivariate Statistics*, volume 53 of *Lecture Notes–Monograph Series*, pages 103–131. Institute of Mathematical Statistics. (p. 26)

[Ekeland and Temam, 1976] Ekeland, I. and Temam, R. (1976). *Convex Analysis and Variational Problems*. North-holland edition. (p. 170, 191)

[Fattahi et al., 2014] Fattahi, M., Mahootchi, M., Mosadegh, H., and Fallahi, F. (2014). A new approach for maintenance scheduling of generating units in electrical power systems based on their operational hours. *Computers & Operations Research*, 50:61–79. (p. 7, 16)

[Fermi et al., 1954] Fermi, E., Metropolis, N., and Alei, E. F. (1954). Phase Shift Analysis of the Scattering of Negative Pions by Hydrogen. *Physical Review Journals Archive*, 95(6):1581–1585. (p. 47, 52)

[Froger et al., 2016] Froger, A., Gendreau, M., Mendoza, J. E., Pinson, E., and Rousseau, L.-M. (2016). Maintenance Scheduling in the Electricity Industry: A Literature Review. *European Journal of Operational Research*, 251(3):695–706. (p. 6, 7, 16)

[Geiersbach and Pflug, 2019] Geiersbach, C. and Pflug, G. C. (2019). Projected Stochastic Gradients for Convex Constrained Problems in Hilbert Spaces. *SIAM Journal on Optimization*, 29(3):2079–2099. (p. 151, 153, 165)

[Geiersbach and Wollner, 2019] Geiersbach, C. and Wollner, W. (2019). A Stochastic Gradient Method with Mesh Refinement for PDE Constrained Optimization under Uncertainty. *arXiv:1905.08650 [cs, math]*. arXiv: 1905.08650. (p. 153, 171)

[Ginsbourger et al., 2008] Ginsbourger, D., Riche, R. L., and Carraro, L. (2008). A Multi-points Criterion for Deterministic Parallel Global Optimization based on Gaussian Processes. Technical report. (p. 87)

[Grigoriev et al., 2006] Grigoriev, A., van de Klundert, J., and Spieksma, F. C. (2006). Modeling and solving the periodic maintenance problem. *European Journal of Operational Research*, 172(3):783–797. (p. 7, 16)

[Halton, 1960] Halton, J. H. (1960). On the efficiency of certain quasi-random sequences of points in evaluating multi-dimensional integrals. *Numerische Mathematik*, 2(1):84–90. (p. 31, 57)

[Hansen et al., 2010] Hansen, N., Auger, A., Ros, R., Finck, S., and Pošík, P. (2010). Comparing results of 31 algorithms from the black-box optimization benchmarking BBOB-2009. In *Proceedings of the 12th annual conference companion on Genetic and evolutionary computation*, pages 1689–1696. ACM. (p. 78)

[Hansen et al., 2021] Hansen, N., Auger, A., Ros, R., Mersmann, O., Tušar, T., and Brockhoff, D. (2021). COCO: A Platform for Comparing Continuous Optimizers in a Black-Box Setting. *Optimization Methods and Software*, 36(1):114–144. (p. 8, 18, 65, 72)

[Hansen et al., 2009] Hansen, N., Finck, S., Ros, R., and Auger, A. (2009). Real-parameter black-box optimization benchmarking 2009: Noiseless functions definitions. Working Paper 2009/20, INRIA. (p. 73, 74)

[Hansen and Ostermeier, 1996] Hansen, N. and Ostermeier, A. (1996). Adapting arbitrary normal mutation distributions in evolution strategies: the covariance matrix adaptation. In *Proceedings of IEEE International Conference on Evolutionary Computation*, pages 312–317, Nagoya, Japan. IEEE. (p. 44, 65)

[Hansen et al., 2016] Hansen, N., Tusar, T., Mersmann, O., Auger, A., and Brockhoff, D. (2016). COCO: The Experimental Procedure. *arXiv:1603.08776 [cs]*. arXiv: 1603.08776. (p. 77)

[Hastie et al., 2001] Hastie, T., Tibshirani, R., and Friedman, J. (2001). *The elements of statistical learning*, volume 1. Springer series in statistics New York. (p. 39)

[Hernández-Lobato et al., 2014] Hernández-Lobato, J. M., Hoffman, M. W., and Ghahramani, Z. (2014). Predictive Entropy Search for Efficient Global Optimization of Black-box Functions. In *Proceedings of the 27th International Conference on Neural Information Processing Systems*, volume 1, pages 918–926. (p. 43)

[Hess, 1995] Hess, C. (1995). On the Measurability of the Conjugate and the Subdifferential of a Normal Integrand. *Journal of Convex Analysis*, 2(1-2):153–165. (p. 153, 156, 158, 161)

[Hess, 1996] Hess, C. (1996). Epi-convergence of sequences of normal integrands and strong consistency of the maximum likelihood estimator. *The Annals of Statistics*, 24(3):1298–1315. (p. 160)

[Holland, 1975] Holland, J. H. (1975). *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. University of Michigan Press, Ann Arbor, MI, USA. (p. 22)

[Hooke and Jeeves, 1961] Hooke, R. and Jeeves, T. A. (1961). "Direct Search" Solution of Numerical and Statistical Problems. *Journal of the ACM*, 8(2):212–229. (p. 47)

[Householder, 1958] Householder, A. S. (1958). Unitary Triangularization of a Nonsymmetric Matrix. *Journal of the ACM*, 5(4):339–342. (p. 57)

[Iooss et al., 2010] Iooss, B., Boussouf, L., Feuillard, V., and Marrel, A. (2010). Numerical studies of the metamodel fitting and validation processes. *International Journal on Advances in Systems and Measurements*, 3(1&2):11. (p. 38, 39)

[Iooss and Marrel, 2019] Iooss, B. and Marrel, A. (2019). Advanced Methodology for Uncertainty Propagation in Computer Experiments with Large Number of Inputs. *Nuclear Technology*, 205(12):1588–1606. (p. 24)

[Johnson et al., 1990] Johnson, M., Moore, L., and Ylvisaker, D. (1990). Minimax and maximin distance designs. *Journal of Statistical Planning and Inference*, 26(2):131–148. (p. 31)

[Johnson, 2009] Johnson, S. G. (2009). The NLopt nonlinear-optimization package. http://github.com/stevengj/nlopt. (p. 66)

[Jones, 2001] Jones, D. R. (2001). A taxonomy of global optimization methods based on response surfaces. *Journal of global optimization*, 21(4):345–383. (p. 21, 41, 62)

[Jones et al., 1993] Jones, D. R., Perttunen, C. D., and Stuckman, B. E. (1993). Lipschitzian optimization without the Lipschitz constant. *Journal of Optimization Theory and Applications*, 79(1):157–181. (p. 22, 65)

[Jones et al., 1998] Jones, D. R., Schonlau, M., and Welch, W. J. (1998). Efficient Global Optimization of Expensive Black-Box Functions. *Journal of Global optimization*, 13(4):455–492. (p. 8, 18, 24, 41, 42, 61, 62)

[Joseph et al., 2015] Joseph, V. R., Gul, E., and Ba, S. (2015). Maximum projection designs for computer experiments. *Biometrika*, 102(2):371–380. (p. 31)

[Karimi et al., 2019] Karimi, B., Miasojedow, B., Moulines, E., and Wai, H.-T. (2019). Non-asymptotic Analysis of Biased Stochastic Approximation Scheme. In *Proceedings of the Thirty-Second Conference on Learning Theory*, volume 99, pages 1944–1974. (p. 151)

[Kennedy and Eberhart, 1995] Kennedy, J. and Eberhart, R. (1995). Particle Swarm Optimization. In *Proceedings of ICNN'95*, volume 4, pages 1942–1948, Perth, WA, Australia. (p. 22)

[Kiefer and Wolfowitz, 1952] Kiefer, J. and Wolfowitz, J. (1952). Stochastic Estimation of the Maximum of a Regression Function. *The Annals of Mathematical Statistics*, 23(3):462–466. (p. 151)

[King, 2009] King, D. E. (2009). Dlib-ml: A Machine Learning Toolkit. *The Journal of Machine Learning Research*, 10:1755–1758. (p. 66)

[Kucherenko and Sytsko, 2005] Kucherenko, S. and Sytsko, Y. (2005). Application of Deterministic Low-Discrepancy Sequences in Global Optimization. *Computational Optimization and Applications*, 30(3):297–318. (p. 67, 77)

[Kushner and Yin, 1997] Kushner, H. J. and Yin, G. (1997). *Stochastic approximation algorithms and applications*. Number 35 in Applications of Mathematics. (p. 151)

[Lasdon and Schoeffler, 1965] Lasdon, L. S. and Schoeffler, J. D. (1965). A multi-level technique for optimization. *Joint Automatic Control Conference*, 3:85–92. (p. 93)

[Le Digabel, 2011] Le Digabel, S. (2011). Algorithm 909: NOMAD: Nonlinear Optimization with the MADS Algorithm. *ACM Transactions on Mathematical Software*, 37(4):1–15. (p. 76)

[Le Gratiet, 2013] Le Gratiet, L. (2013). *Multi-fidelity Gaussian process regression for computer experiments*. PhD Thesis, Université Paris-Diderot-Paris VII. (p. 28)

[Lewis and Torczon, 1999] Lewis, R. M. and Torczon, V. (1999). Pattern Search Algorithms for Bound Constrained Minimization. *SIAM Journal on Optimization*, 9(4):1082–1099. (p. 52)

[Lewis and Torczon, 2000] Lewis, R. M. and Torczon, V. (2000). Pattern Search Methods for Linearly Constrained Minimization. *SIAM Journal on Optimization*, 10(3):917–941. (p. 52)

[Lin et al., 2018] Lin, J., Rosasco, L., Villa, S., and Zhou, D.-X. (2018). Modified Fejér sequences and applications. *Computational Optimization and Applications*, 71(1):95–113. (p. 153, 173, 175)

[Loeppky et al., 2009] Loeppky, J. L., Sacks, J., and Welch, W. J. (2009). Choosing the Sample Size of a Computer Experiment: A Practical Guide. *Technometrics*, 51(4):366–376. (p. 61, 62)

[Lusby et al., 2013] Lusby, R., Muller, L. F., and Petersen, B. (2013). A solution approach based on Benders decomposition for the preventive maintenance scheduling problem of a stochastic large-scale energy system. *Journal of Scheduling*, 16(6):605–628. (p. 6, 16)

[Marrel et al., 2008] Marrel, A., Iooss, B., Van Dorpe, F., and Volkova, E. (2008). An efficient methodology for modeling complex computer codes with Gaussian processes. *Computational Statistics & Data Analysis*, 52(10):4731–4744. (p. 24)

[Martin et al., 2019] Martin, M., Nobile, F., and Tsilifis, P. (2019). A Multilevel Stochastic Gradient method for PDE-constrained Optimal Control Problems with uncertain parameters. *arXiv:1912.11900 [math]*. (p. 151)

[Matheron, 1962] Matheron, G. (1962). *Traité de géostatistique appliquée*, volume 1 of *Mémoires du BRGM*. Éditions Technip, Paris. (p. 24)

[McKay et al., 1979] McKay, M. D., Beckman, R. J., and Conover, W. J. (1979). A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code. *Technometrics*, 21(2):239–245. (p. 31)

[Meeker and Escobar, 2014] Meeker, W. Q. and Escobar, L. A. (2014). *Statistical methods for reliability data*. John Wiley & Sons. (p. 103)

[Mesarović et al., 1970] Mesarović, M. D., Macko, D., and Takahara, Y. (1970). *Theory of Hierarchical, Multilevel Systems*, volume 68 of *Mathematics in Science and Engineering*. Academic Press. (p. 93, 95)

[Metropolis et al., 1953] Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., and Teller, E. (1953). Equation of State Calculations by Fast Computing Machines. *Journal of Chemical Physics*, 21:1087–1092. (p. 22)

[Mohammadi, 2016] Mohammadi, H. (2016). *Kriging-based black-box global optimization: analysis and new algorithms*. PhD thesis, Université de Lyon. (p. 8, 18, 36, 44, 62, 65, 77, 80, 81)

[Morris, 1991] Morris, M. D. (1991). Factorial Sampling Plans for Preliminary Computational Experiments. *Technometrics*, 33(2):161–174. (p. 140)

[Munoz Zuniga and Sinoquet, 2020] Munoz Zuniga, M. and Sinoquet, D. (2020). Global optimization for mixed categorical-continuous variables based on Gaussian process models with a randomized categorical space exploration step. *INFOR: Information Systems and Operational Research*, pages 1–32. (p. 65, 87, 177)

[Nash, 2000] Nash, S. G. (2000). A survey of truncated-Newton methods. *Journal of Computational and Applied Mathematics*, 124:45–59. (p. 66)

[Nelder and Mead, 1965] Nelder, J. A. and Mead, R. (1965). A simplex method for function minimization. *The computer journal*, 7(4):308–313. (p. 47)

[Nemirovski et al., 2009] Nemirovski, A., Juditsky, A., Lan, G., and Shapiro, A. (2009). Robust Stochastic Approximation Approach to Stochastic Programming. *SIAM Journal on Optimization*, 19(4):1574–1609. (p. 90, 149, 151, 152, 153, 165, 171, 175)

[Nemirovski and Yudin, 1983] Nemirovski, A. and Yudin, D. B. (1983). *Problem Complexity and Method Efficiency in Optimization*. A Wiley-Interscience publication. Wiley. (p. 152)

[Nicolai and Dekker, 2008] Nicolai, R. P. and Dekker, R. (2008). Optimal Maintenance of Multi-component Systems: A Review. In *Complex System Maintenance Handbook*, pages 263–286. Springer London. (p. 6, 16)

[Nocedal, 1980] Nocedal, J. (1980). Updating Quasi-Newton Matrices With Limited Storage. *Mathematics of Computation*, 35(151):773–782. (p. 44, 77)

[Olde Keizer et al., 2017] Olde Keizer, M. C., Teunter, R. H., and Veldman, J. (2017). Joint condition-based maintenance and inventory optimization for systems with multiple components. *European Journal of Operational Research*, 257(1):209–222. (p. 6, 16)

[Parikh and Boyd, 2014] Parikh, N. and Boyd, S. P. (2014). Proximal Algorithms. *Foundations and Trends® in Optimization*, 1(3):127–239. (p. 152)

[Park and Baek, 2001] Park, J.-S. and Baek, J. (2001). Efficient computation of maximum likelihood estimators in a spatial linear model with power exponential covariogram. *Computers & Geosciences*, 27(1):1–7. (p. 35)

[Penrose, 1955] Penrose, R. (1955). A generalized inverse for matrices. *Mathematical Proceedings of the Cambridge Philosophical Society*, 51(3):406–413. (p. 36)

[Picheny et al., 2013] Picheny, V., Wagner, T., and Ginsbourger, D. (2013). A benchmark of kriging-based infill criteria for noisy optimization. *Structural and Multidisciplinary Optimization*, 48(3):607–626. (p. 43, 80)

[Polak and Wetter, 2006] Polak, E. and Wetter, M. (2006). Precision Control for Generalized Pattern Search Algorithms with Adaptive Precision Function Evaluations. *SIAM Journal on Optimization*, 16(3):650–669. (p. 6, 16)

[Polyak and Juditsky, 1992] Polyak, B. T. and Juditsky, A. B. (1992). Acceleration of Stochastic Approximation by Averaging. *SIAM Journal on Control and Optimization*, 30(4):838–855. (p. 151, 171)

[Powell, 2002] Powell, M. (2002). UOBYQA: unconstrained optimization by quadratic approximation. *Mathematical Programming*, 92(3):555–582. (p. 21)

[Powell, 1994] Powell, M. J. D. (1994). A Direct Search Optimization Method That Models the Objective and Constraint Functions by Linear Interpolation. In Gomez, S. and Hennart, J.-P., editors, *Advances in Optimization and Numerical Analysis*, pages 51–67. Springer Netherlands, Dordrecht. (p. 47, 66)

[Rasmussen and Williams, 2006] Rasmussen, C. E. and Williams, C. K. I. (2006). *Gaussian processes for machine learning*. Adaptive computation and machine learning. MIT Press, Cambridge, Mass. (p. 24, 33, 35)

[Rehbach et al., 2020] Rehbach, F., Zaefferer, M., Naujoks, B., and Bartz-Beielstein, T. (2020). Expected improvement versus predicted value in surrogate-based optimization. In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference*, pages 868–876, Cancún Mexico. ACM. (p. 43, 81)

[Rios and Sahinidis, 2013] Rios, L. M. and Sahinidis, N. V. (2013). Derivative-free optimization: a review of algorithms and comparison of software implementations. *Journal of Global Optimization*, 56(3):1247–1293. (p. 21)

[Robbins and Monro, 1951] Robbins, H. and Monro, S. (1951). A Stochastic Approximation Method. *The Annals of Mathematical Statistics*, 22(3):400–407. (p. 89, 148, 151)

[Robbins and Siegmund, 1971] Robbins, H. and Siegmund, D. (1971). A Convergence Theorem for Non Negative Almost Supermartingales and Some Applications. In *Optimizing Methods in Statistics*, pages 233–257. Springer, New York, NY. (p. 206)

[Rockafellar and Wets, 2004] Rockafellar, R. T. and Wets, R. J.-B. (2004). *Variational analysis*. Number 317 in Grundlehren der mathematischen Wissenschaften. Springer, Berlin. (p. 153, 157, 158, 159, 164)

[Rosasco et al., 2019] Rosasco, L., Villa, S., and Vũ, B. C. (2019). Convergence of Stochastic Proximal Gradient Algorithm. *Applied Mathematics & Optimization*, pages 1–27. (p. 172, 173, 175)

[Roustant et al., 2012] Roustant, O., Ginsbourger, D., and Deville, Y. (2012). DiceKriging, DiceOptim: Two R packages for the analysis of computer experiments by kriging-based metamodeling and optimization. *Journal of Statistical Software*, 51(1). (p. 33, 65)

[Ruppert, 1988] Ruppert, D. (1988). Efficient estimations from a slowly convergent Robbins-Monro process. Technical report, Cornell University Operations Research and Industrial Engineering. (p. 171)

[Ruszczyński and Shapiro, 2006] Ruszczyński, A. and Shapiro, A. (2006). Optimization of Risk Measures. In Calafiore, G. and Dabbene, F., editors, *Probabilistic and Randomized Methods for Design under Uncertainty*, pages 119–157. Springer-Verlag, London. (p. 178)

[Sacks et al., 1989] Sacks, J., Welch, W. J., Mitchell, T. J., and Wynn, H. P. (1989). Design and Analysis of Computer Experiments. *Statistical Science*, 4(4):409–423. (p. 24)

[Sekhon and Mebane, 1998] Sekhon, J. S. and Mebane, W. R. (1998). Genetic optimization using derivatives. *Political Analysis*, 7:187–210. (p. 65)

[Shang and Apley, 2020] Shang, B. and Apley, D. W. (2020). Fully-sequential space-filling design algorithms for computer experiments. *Journal of Quality Technology*, pages 1–24. (p. 62)

[Shapiro et al., 2009] Shapiro, A., Dentcheva, D., and Ruszczyński, A. (2009). *Lectures on Stochastic Programming: Modeling and Theory*. Society for Industrial and Applied Mathematics. (p. 148)

[Sobol, 1967] Sobol, I. (1967). On the distribution of points in a cube and the approximate evaluation of integrals. *USSR Computational Mathematics and Mathematical Physics*, 7(4):86–112. (p. 31)

[Spall, 2003] Spall, J. C. (2003). *Introduction to Stochastic Search and Optimization: Estimation, Simulation, and Control*. John Wiley & Sons, Ltd. (p. 5, 15)

[Spendley et al., 1962] Spendley, W., Hext, G. R., and Himsworth, F. R. (1962). Sequential Application of Simplex Designs in Optimisation and Evolutionary Operation. *Technometrics*, 4(4):441–461. (p. 47)

[Srinivas et al., 2010] Srinivas, N., Krause, A., Kakade, S., and Seeger, M. (2010). Gaussian Process Optimization in the Bandit Setting: No Regret and Experimental Design. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, pages 1015–1022, Haifa, Israel. (p. 42)

[Stein, 1999] Stein, M. L. (1999). *Interpolation of Spatial Data*. Springer Series in Statistics. Springer New York. (p. 27, 35, 36, 37, 38)

[Suresh and Kumarappan, 2013] Suresh, K. and Kumarappan, N. (2013). Hybrid improved binary particle swarm optimization approach for generation maintenance scheduling problem. *Swarm and Evolutionary Computation*, 9:69–89. (p. 7, 16)

[Sánchez-Silva et al., 2016] Sánchez-Silva, M., Frangopol, D. M., Padgett, J., and Soliman, M. (2016). Maintenance and Operation of Infrastructure Systems: Review. *Journal of Structural Engineering*, 142(9):F4016004. (p. 6, 16)

[Takahara, 1964] Takahara, Y. (1964). Multilevel Approach to Dynamic Optimization. Technical Report SRC-50- C-64-18, Case Western Reserve University, Systems Research Center. (p. 93)

[Talgorn et al., 2015] Talgorn, B., Le Digabel, S., and Kokkolaras, M. (2015). Statistical Surrogate Formulations for Simulation-Based Design Optimization. *Journal of Mechanical Design*, 137(2). (p. 43, 177)

[Torczon, 1989] Torczon, V. (1989). *Multi-directional search: a direct search algorithm for parallel machines*. PhD thesis, Rice University, Houston, Texas. (p. 47)

[Torczon, 1997] Torczon, V. (1997). On the convergence of pattern search algorithms. *SIAM Journal on optimization*, 7(1):1–25. (p. 22, 47, 48, 52)

[Torczon and Trosset, 1998] Torczon, V. and Trosset, M. W. (1998). From Evolutionary Operation to Parallel Direct Search: Pattern Search Algorithms for Numerical Optimization. *Computing Science and Statistics*, 29:396–401. (p. 47)

[Tuo and Wang, 2019] Tuo, R. and Wang, W. (2019). Kriging prediction with isotropic Matern correlations: robustness and experimental design. *arXiv:1911.05570 [math, stat]*. (p. 37, 38)

[Vazquez and Bect, 2010] Vazquez, E. and Bect, J. (2010). Convergence properties of the expected improvement algorithm with fixed mean and covariance functions. *Journal of Statistical Planning and Inference*, 140(11):3088–3095. (p. 43, 44)

[Villemonteix et al., 2009] Villemonteix, J., Vazquez, E., and Walter, E. (2009). An informational approach to the global optimization of expensive-to-evaluate functions. *Journal of Global Optimization*, 44(4):509–534. (p. 43)

[Wada and Fujisaki, 2015] Wada, T. and Fujisaki, Y. (2015). A stopping rule for stochastic approximation. *Automatica*, 60:1–6. (p. 154)

[Wang et al., 2020] Wang, W., Tuo, R., and Wu, C. F. J. (2020). On Prediction Properties of Kriging: Uniform Error Bounds and Robustness. *Journal of the American Statistical Association*, 115(530):920–930. (p. 31)

[Woods, 1985] Woods, D. J. (1985). *An Interactive Approach ror Solving Multi-Objective Optimization Problems*. PhD thesis, Rice University, Houston, Texas. (p. 47)

[Wright, 1996] Wright, M. H. (1996). Direct search methods: Once scorned, now respectable. In *Numerical analysis: Proceedings of the 1995 Dundee Biennial Conference in Numerical Analysis*, pages 191–208. Addison-Wesley, Harlow, United Kingdom. (p. 47)

[Yarotsky, 2013] Yarotsky, D. (2013). Examples of inconsistency in optimization by expected improvement. *Journal of Global Optimization*, 56(4):1773–1790. (p. 44)

[Yin, 1990] Yin, G. (1990). A stopping rule for the Robbins-Monro method. *Journal of Optimization Theory and Applications*, 67(1):151–173. (p. 154)

A Basic definitions in analysis

We recall some basic notions in analysis that are used within the manuscript. The definitions presented in this appendix are taken from [Ekeland and Temam, 1976] and [Bauschke and Combettes, 2011].

Definition A.1. (Topological dual) Let $\mathbb{U}$ be a real vector space. The topological dual of $\mathbb{U}$, denoted by $\mathbb{U}^\star$ is the set of all continuous linear functionals *i.e.* the linear maps from $\mathbb{U}$ to $\mathbb{R}$.

In the following definitions, the space $\mathbb{U}$ is a Banach space. We denote by $\langle \cdot\,, \cdot \rangle$ the duality pairing between $\mathbb{U}$ and its topological dual $\mathbb{U}^\star$, by $\|\cdot\|$ the norm on $\mathbb{U}$ and by $\|\cdot\|_\star$ the norm on $\mathbb{U}^\star$.

Definition A.2. (Proper function) A function $J : \mathbb{U} \to \overline{\mathbb{R}} = \mathbb{R} \cup \{-\infty, +\infty\}$ is said to be *proper* if it nowhere takes the value $-\infty$ and is not identically equal to $+\infty$.

Definition A.3. (Convexity) A function $J : \mathbb{U} \to \mathbb{R}$ is said to be:

- *convex*, if for all $\alpha \in [0,1]$ and all $u, v \in \mathbb{U}$, we have:

$$J(\alpha u + (1-\alpha)v) \leq \alpha J(u) + (1-\alpha)J(v) \ ,$$

- *strictly convex*, if for all $\alpha \in]0,1[$ and all $u, v \in \mathbb{U}$ with $u \neq v$, we have:

$$J(\alpha u + (1-\alpha)v) < \alpha J(u) + (1-\alpha)J(v) \ ,$$

- *a-strongly convex*, with $a > 0$, if all $\alpha \in [0,1]$ and all $u, v \in \mathbb{U}$, we have:

$$J(\alpha u + (1-\alpha)v) \leq \alpha J(u) + (1-\alpha)J(v) - \alpha(1-\alpha)\frac{a}{2}\|u-v\|^2 \ ,$$

Definition A.4. (Gateaux-differentiability) Let $J : \mathbb{U} \to \mathbb{R}$ be a function. The *directional derivative* of J at $u \in \mathbb{U}$ in the direction $d \in \mathbb{U}$ is:

$$DJ(u;d) = \lim_{\varepsilon \to 0^+} \frac{J(u+\varepsilon d) - J(u)}{\varepsilon} \ ,$$

provided that the limit exists. The function J is said to be *Gateaux-differentiable* at u if it admits directional derivatives in all directions d and if $DJ(u;\cdot)$ is linear and

continuous on $\mathbb{U}$. The Gateaux-derivative of J at u is then the element of $\mathbb{U}^\star$ denoted by $J'(u)$ such that for all $d \in \mathbb{U}$:

$$DJ(u;d) = \langle J'(u)\,, d\rangle \ .$$

Finally, the function J is said to be Gateaux-differentiable if it is Gateaux-differentiable at every $u \in \mathbb{U}$.

Definition A.5. (Subdifferential) Let $J : \mathbb{U} \to \mathbb{R} \cup \{+\infty\}$ be a proper, convex, function. The *subdifferential* of J at $u \in \mathbb{U}$ is the set:

$$\partial J(u) = \{s \in \mathbb{U}^\star,\ \forall v \in \mathbb{U},\ J(v) \geq J(u) + \langle s\,, v-u\rangle\}\ .$$

If $\partial J(u) \neq \emptyset$, we say that J is *subdifferentiable* at u. If J is subdifferentiable for all $u \in \mathbb{U}$, we say that J is subdifferentiable.

Proposition A.6. *Let $J : \mathbb{U} \to \mathbb{R}$ be a Gateaux-differentiable function. Then, there is an equivalence between J being a-strongly convex for some $a > 0$ and the following inequality:*

$$\forall (u,v) \in \mathbb{U}^2,\ J(v) \geq J(u) + \langle J'(u)\,, v-u\rangle + \frac{a}{2}\left\|v-u\right\|^2 \ .$$

Definition A.7. (Lower-semicontinuity) A function $J : \mathbb{U} \to \mathbb{R}$ is said to be *lower semi-continuous* (l.s.c.) if for all $u \in \mathbb{U}$:

$$\liminf_{v \to u} J(v) \geq J(u)\ .$$

Definition A.8. (Coercivity) A function $J : \mathbb{U} \to \mathbb{R}$ is *coercive* on $U^{\mathrm{ad}} \subset \mathbb{U}$ if:

$$\lim_{\substack{\|u\| \to +\infty \\ u \in U^{\mathrm{ad}}}} J(u) = +\infty\ .$$

Note that if U^{ad} is bounded, J is automatically coercive.

Definition A.9. (Lipschitz-continuity) An operator $\Psi : \mathbb{U} \to \mathbb{U}^\star$ is *A-Lipschitz continuous*, with $A > 0$, if for all $u, v \in \mathbb{U}$, we have:

$$\left\|\Psi(u) - \Psi(v)\right\|_\star \leq A\left\|u-v\right\| \ .$$

Definition A.10. (Monotonicity) An operator $\Psi : \mathbb{U} \to \mathbb{U}^\star$ is said to be:

- *monotone*, if for all $u, v \in \mathbb{U}$ we have:

$$\langle \Psi(u) - \Psi(v)\,, u-v\rangle \geq 0\ ,$$

- *a-strongly monotone*, with $a > 0$, if for all $u, v \in \mathbb{U}$ we have:

$$\langle \Psi(u) - \Psi(v)\,, u-v\rangle \geq a\left\|u-v\right\|^2 \ .$$

Remark A.11. Definitions A.9 and A.10 are often used in the case where $\Psi : \mathbb{U} \to \mathbb{U}^\star$ is the derivative of a convex function. ♢

B Basic notions from measure theory

We recall some definitions from measure theory that are used within the manuscript. Most of these definitions come into play in Part III where measurability is at the heart of the matter. We can refer to [Billingsley, 1995] for a more detailed exposition of measure theory.

Definition B.1. (σ-field, measurable space) Let Ω be an arbitrary non-empty set. A family $\mathcal{A}$ of subsets of Ω is called a *σ-field* (or *σ-algebra*) on Ω if it satisfies the following properties:

- $\Omega \in \mathcal{A}$,
- if $A \in \mathcal{A}$, then $A^{\mathrm{c}} \in \mathcal{A}$,
- for any countable family $\{A_n\}_{n\in\mathbb{N}}$ of elements of $\mathcal{A}$, we have $\bigcup_{n\in\mathbb{N}} A_n \in \mathcal{A}$.

The pair $(\Omega, \mathcal{A})$ is called a *measurable space* and the elements of $\mathcal{A}$ are called the *measurable sets*.

Definition B.2. (Measure, probability space) Let $(\Omega, \mathcal{A})$ be a measurable space. A *measure* on $(\Omega, \mathcal{A})$ is a function $\mu : \mathcal{A} \to [0, +\infty]$ such that:

- $\mu(\emptyset) = 0$,
- μ is σ-additive: for every countable sequence of pairwise disjoint sets $A_1, \ldots, A_n \subset \mathcal{A}$, we have:

$$\mu\left(\bigcup_{i=1}^{n} A_i\right) = \sum_{i=1}^{n} \mu(A_i) \ .$$

The triple $(\Omega, \mathcal{A}, \mu)$ is called a *measure space*. If moreover $\mu(\Omega) = 1$, we say that $(\Omega, \mathcal{A}, \mu)$ is a *probability space*.

Definition B.3. (Measurable function) Let $(E, \mathcal{E})$ and $(F, \mathcal{F})$ be two measurable spaces. A function $f : (E, \mathcal{E}) \to (F, \mathcal{F})$ is *measurable* if for all $B \in \mathcal{F}$, we have $f^{-1}(B) \in \mathcal{E}$.

Definition B.4. (Random variable) Let $(\Omega, \mathcal{A}, \mu)$ be a probability space and $(E, \mathcal{E})$ be a measurable space. An $(E, \mathcal{E})$-valued *random variable* $\boldsymbol{X} : (\Omega, \mathcal{A}, \mu) \to (E, \mathcal{E})$ is a measurable function from $(\Omega, \mathcal{A}, \mu)$ to $(E, \mathcal{E})$.

Definition B.5. (Topology, topological space) Let E be an arbitrary set. A family τ of subsets of E is called a *topology* on E if it satisfies the following properties:

- $\emptyset \in \tau$ and $E \in \tau$,
- for any finite family $\{A_1, \ldots, A_n\}$ of elements of τ, we have $\bigcap_{i=1}^{n} A_i \in \tau$,
- for any arbitrary family $\{A_i\}_{i \in I}$ of elements of τ, where I is an index set (that may be uncountable), we have $\bigcup_{i \in I} A_i \in \tau$.

The elements of τ are called the *open sets* in E. The pair (E, τ) is called a *topological space*.

Definition B.6. (Borel σ-field) Let (E, τ) be a topological space. The *Borel σ-field* on E, denoted by $\mathcal{B}(E)$, is the smallest σ-field that contains all open sets of E.

When we consider real-valued random variables, we assume that $\mathbb{R}$ is equipped with its Borel σ-field $\mathcal{B}(\mathbb{R})$. The two following definitions are useful to characterize the Weibull distribution that appears in Chapters 6, 8 and 10.

Definition B.7. (Cumulative distribution function) Let $(\Omega, \mathcal{A}, \mathbb{P})$ be a probability space and $\boldsymbol{X}$ be a real-valued random variable defined on Ω. The *cumulative distribution function* of $\boldsymbol{X}$ is the function $F_{\boldsymbol{X}} : \mathbb{R} \to [0, 1]$ such that:

$$F_{\boldsymbol{X}}(x) = \mathbb{P}\left(\{\omega \in \Omega,\ \boldsymbol{X}(\omega) \leq x\}\right) = \mathbb{P}\left(\boldsymbol{X}^{-1}\left((-\infty, x]\right)\right) = \mathbb{P}(\boldsymbol{X} \leq x) \ .$$

Definition B.8. (Weibull distribution) Let $(\Omega, \mathcal{A}, \mathbb{P})$ be a probability space and $\boldsymbol{X}$ be a real-valued random variable defined on Ω. We say that $\boldsymbol{X}$ has a *Weibull distribution* of parameters $\beta > 0$ and $\lambda > 0$, denoted by $\mathrm{Weib}(\beta, \lambda)$, if its cumulative distribution function $F_{\boldsymbol{X}}$ is given by:

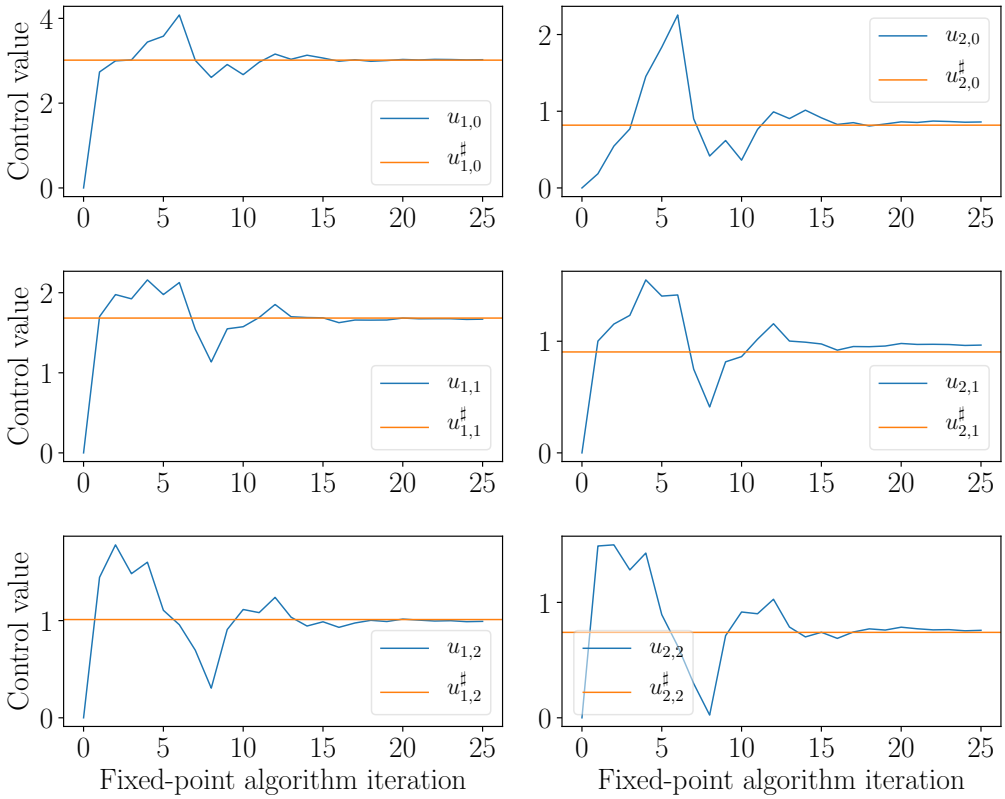
$$F_{\boldsymbol{X}}(x) = \begin{cases} 1 - e^{-(x/\lambda)^{\beta}} & \text{if } x \geq 0 \ , \\ 0 & \text{if } x < 0 \ . \end{cases}$$

C Full convergence plots of the APP on the synthetic cases

We give the full convergence plots of the family of controls $u = \{u_{i,t}\}_{(i,t)\in I\times\mathbb{T}_{-1}}$ for the application of the APP to the synthetic problems of Chapter 9.

C.1 For the deterministic synthetic test case

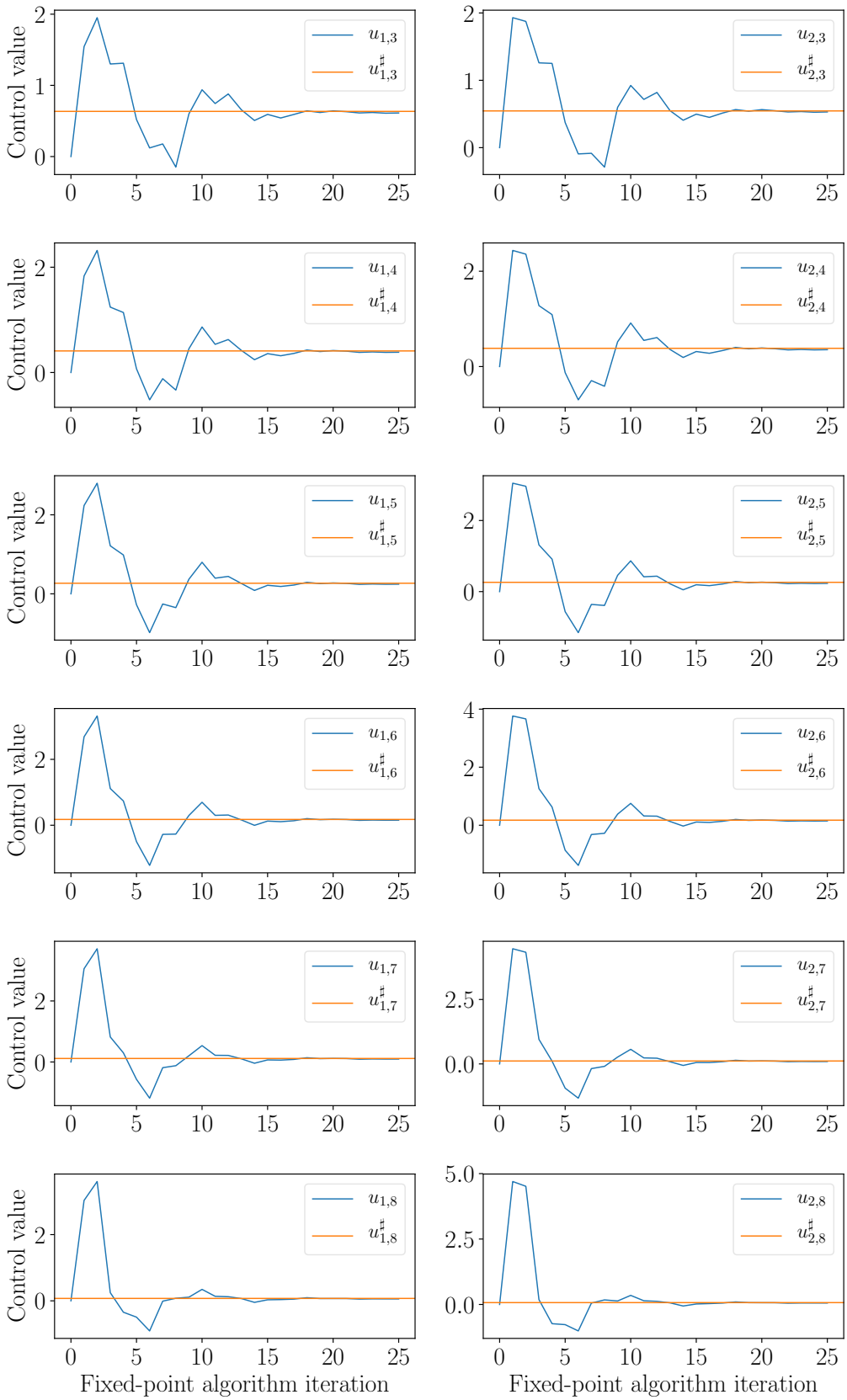
We give the convergence plots for the deterministic test case tackled in Section 9.2.



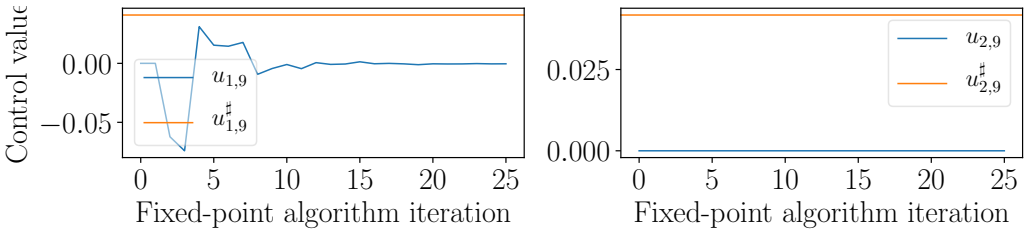
(a) Convergence of the whole family of controls $u = \{u_{i,t}\}_{(i,t)\in I\times\mathbb{T}_{-1}}$ during the fixed-point algorithm (first part).

C.2 For the stochastic synthetic test case

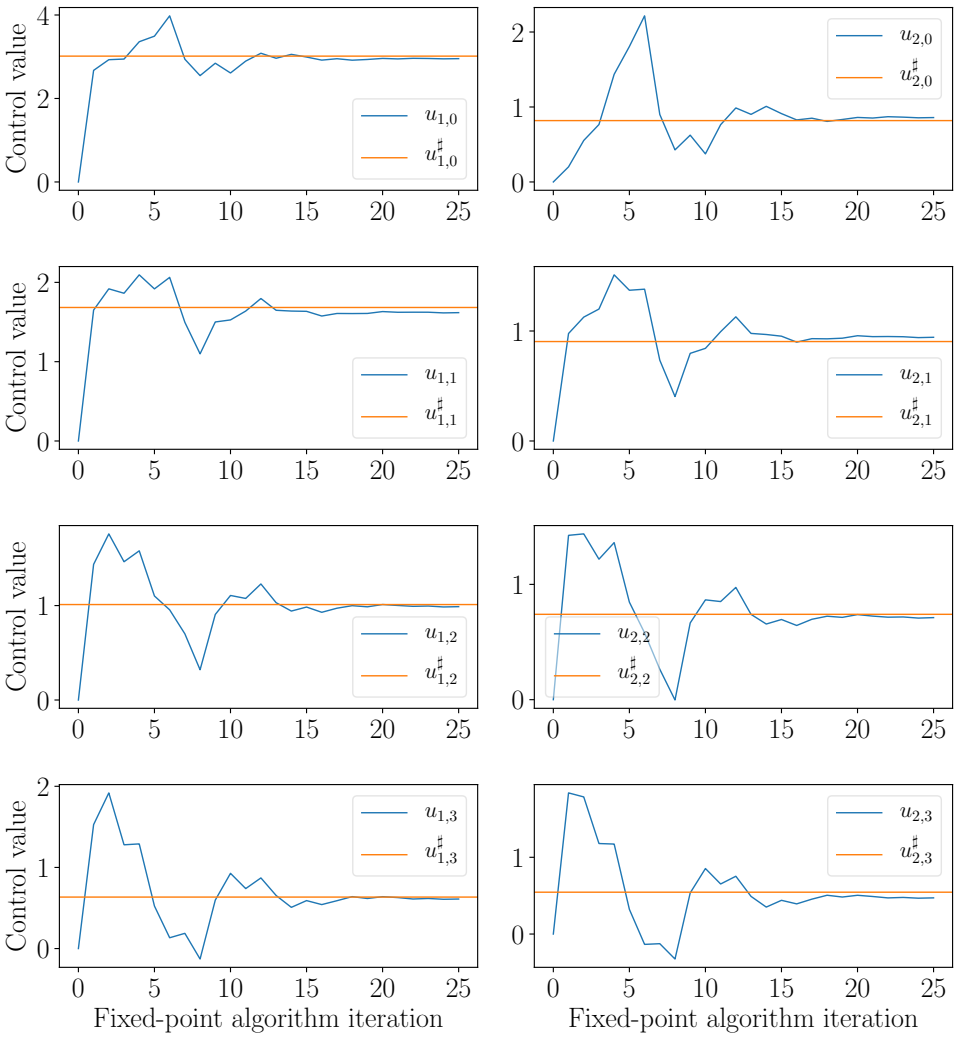
We give the convergence plots for the stochastic test case tackled in Section 9.3.



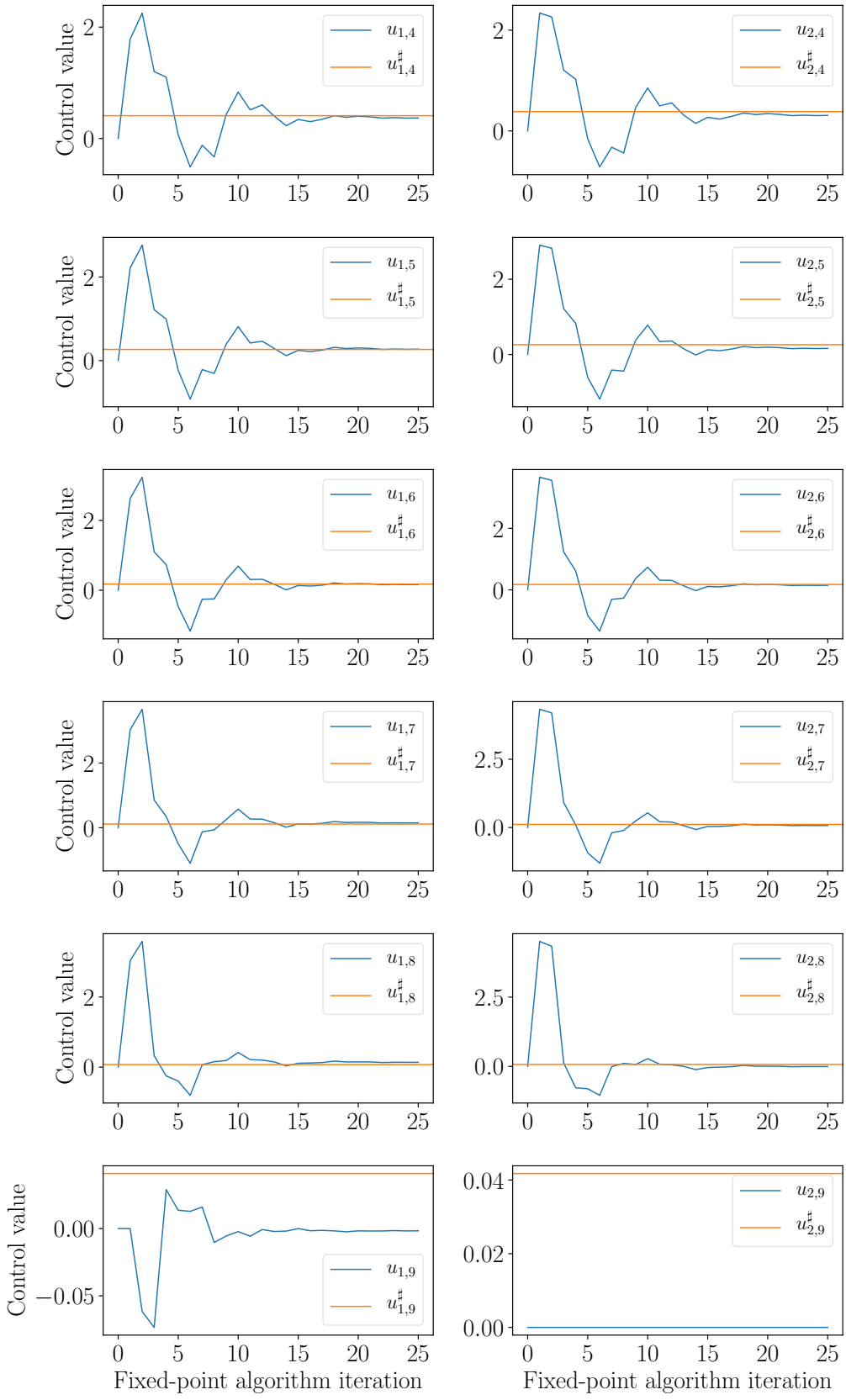
(b) Convergence of the whole family of controls $u = \{u_{i,t}\}_{(i,t)\in I\times\mathbb{T}_{-1}}$ during the fixed-point algorithm (second part).



(c) Convergence of the whole family of controls $u = \{u_{i,t}\}_{(i,t)\in I\times\mathbb{T}_{-1}}$ during the fixed-point algorithm (third part).



(a) Convergence of the whole family of controls $u = \{u_{i,t}\}_{(i,t)\in I\times\mathbb{T}_{-1}}$ during the fixed-point algorithm in the stochastic case (first part).



(b) Convergence of the whole family of controls $u = \{u_{i,t}\}_{(i,t)\in I\times\mathbb{T}_{-1}}$ during the fixed-point algorithm in the stochastic case (second part).

D Explicit expressions for the implementation of the decomposition by component

We give the explicit expression of the dynamics of the industrial system of described in Chapter 8 (Section D.1) and of its relaxed version defined in Chapter 10 (Section D.2). We also give the backward recursion for the optimal multipliers in Section D.3. Finally, in Section D.4, we give some practical details for the computation of the derivative of the relaxed indicator function introduced in Definition 10.2. With these explicit expressions, we have all the tools for the practical implementation of the fixed-point algorithm 8.

D.1 Explicit expression of the dynamics of the industrial system

We give an explicit expression for the dynamics f_i of component $i \in I$ that appears in (8.5). We can write:

$$\boldsymbol{X}_{i,t+1} = \begin{pmatrix} \boldsymbol{E}_{i,t+1} \\ \boldsymbol{A}_{i,t+1} \\ \boldsymbol{P}_{i,t+1} \end{pmatrix} = \begin{pmatrix} f_{i,\boldsymbol{E}}(\boldsymbol{X}_{1:i,t}, \boldsymbol{S}_t, u_{i,t}, \boldsymbol{W}_{i,t+1}) \\ f_{i,\boldsymbol{A}}(\boldsymbol{X}_{1:i,t}, \boldsymbol{S}_t, u_{i,t}, \boldsymbol{W}_{i,t+1}) \\ f_{i,\boldsymbol{P}}(\boldsymbol{X}_{1:i,t}, \boldsymbol{S}_t, u_{i,t}, \boldsymbol{W}_{i,t+1}) \end{pmatrix} ,$$

so that $f_i = (f_{i,\boldsymbol{E}}, f_{i,\boldsymbol{A}}, f_{i,\boldsymbol{P}})$. We give an explicit formula for $f_{i,\boldsymbol{E}}$, $f_{i,\boldsymbol{A}}$ and $f_{i,\boldsymbol{P}}$.

D.1.1 Dynamics of the regime $\boldsymbol{E}_{i,t}$

Using Figure 8.2, we can write:

$$\begin{aligned} \boldsymbol{E}_{i,t+1} &= f_{i,\boldsymbol{E}}(\boldsymbol{X}_{1:i,t}, \boldsymbol{S}_t, u_{i,t}, \boldsymbol{W}_{i,t+1}) \\ &= \mathbf{1}_{\mathbb{R}_+}(\boldsymbol{S}_t - \sum_{j=1}^{i} \mathbf{1}_{\{0\}}(\boldsymbol{E}_{j,t}))\mathbf{1}_{\{0\}}(\boldsymbol{E}_{i,t}) \\ &\quad + \Big(\mathbf{1}_{\mathbb{R}_+}(u_{i,t} - \nu) + \mathbf{1}_{\mathbb{R}_+}(\boldsymbol{W}_{i,t+1} - p_i(\boldsymbol{A}_{i,t}))\mathbf{1}_{\mathbb{R}_+^*}(\nu - u_{i,t})\Big)\mathbf{1}_{\{1\}}(\boldsymbol{E}_{i,t}) \, . \end{aligned} \tag{D.1}$$

The first part of (D.1) means that if the component is broken at t and we have enough spares to repair it, it is then functioning at $t+1$. The second part means that if the component is functioning at t and we do a PM, it is still functioning at $t+1$. Finally, if we do not do a PM, the regime depends on the occurrence of a failure between t and $t+1$.

D.1.2 Dynamics of the age $\boldsymbol{A}_{i,t}$

Again, using Figure 8.2, we can write:

$$\begin{aligned}\boldsymbol{A}_{i,t+1} &= f_{i,\boldsymbol{A}}(\boldsymbol{X}_{1:i,t}, \boldsymbol{S}_t, u_{i,t}, \boldsymbol{W}_{i,t+1}) \\ &= (\boldsymbol{A}_{i,t}+1)\Bigg[\mathbf{1}_{\mathbb{R}_+^*}(\sum_{j=1}^{i}\mathbf{1}_{\{0\}}(\boldsymbol{E}_{j,t}) - \boldsymbol{S}_t)\mathbf{1}_{\{0\}}(\boldsymbol{E}_{i,t}) + \Big((1-u_{i,t})\mathbf{1}_{\mathbb{R}_+}(u_{i,t}-\nu) \\ &\qquad + \mathbf{1}_{\mathbb{R}_+}(\boldsymbol{W}_{i,t+1} - p_i(\boldsymbol{A}_{i,t}))\mathbf{1}_{\mathbb{R}_+^*}(\nu - u_{i,t})\Big)\mathbf{1}_{\{1\}}(\boldsymbol{E}_{i,t})\Bigg] .\end{aligned} \tag{D.2}$$

If the component is broken at t, it stays broken if there are not enough spares in the stock. In this case the time $\boldsymbol{A}_{i,t}$ increases by 1. If the component is healthy at t, it ages if no PM is done and no failure occurs. In the case of a PM the component is rejuvenated. If there is a failure, we have $\boldsymbol{A}_{i,t+1} = 0$.

D.1.3 Dynamics of the vector of times since last failures $\boldsymbol{P}_{i,t}$

The expression of the dynamics of $\boldsymbol{P}_{i,t} = (\boldsymbol{P}_{i,t}^1, \dots, \boldsymbol{P}_{i,t}^D)$ is more complex. We write:

$$\boldsymbol{P}_{i,t+1} = \begin{pmatrix}\boldsymbol{P}_{i,t+1}^1 \\ \vdots \\ \boldsymbol{P}_{i,t+1}^D\end{pmatrix} = \begin{pmatrix}f_{i,\boldsymbol{P}}^1(\boldsymbol{X}_{1:i,t}, \boldsymbol{S}_t, u_{i,t}, \boldsymbol{W}_{i,t+1}) \\ \vdots \\ f_{i,\boldsymbol{P}}^D(\boldsymbol{X}_{1:i,t}, \boldsymbol{S}_t, u_{i,t}, \boldsymbol{W}_{i,t+1})\end{pmatrix} = f_{i,\boldsymbol{P}}(\boldsymbol{X}_{1:i,t}, \boldsymbol{S}_t, u_{i,t}, \boldsymbol{W}_{i,t+1}) ,$$

so that $f_{i,\boldsymbol{P}} = (f_{i,\boldsymbol{P}}^1, \dots, f_{i,\boldsymbol{P}}^D)$. We give the expression of $f_{i,\boldsymbol{P}}^d$ for $d \in \{1, \dots, D\}$:

$$\begin{aligned}\boldsymbol{P}_{i,t+1}^d &= \Big((\boldsymbol{P}_{i,t}^d+1)\mathbf{1}_{\mathbb{R}_+}(\boldsymbol{P}_{i,t}^d) + \delta\mathbf{1}_{\{\delta\}}(\boldsymbol{P}_{i,t}^d)\Big)\Big(1 - \mathbf{1}_{\{1\}}(\boldsymbol{E}_{i,t})\mathbf{1}_{\{0\}}(\boldsymbol{E}_{i,t+1})\Big) \\ &\quad + \Big((\boldsymbol{P}_{i,t}^d+1)\mathbf{1}_{\mathbb{R}_+}(\boldsymbol{P}_{i,t}^d)\mathbf{1}_{\{\delta\}}(\boldsymbol{P}_{i,t}^D) + \delta\mathbf{1}_{\{\delta\}}(\boldsymbol{P}_{i,t}^{d-1})\mathbf{1}_{[2,D]}(d) \\ &\quad + (\boldsymbol{P}_{i,t}^{d+1}+1)\mathbf{1}_{\mathbb{R}_+}(\boldsymbol{P}_{i,t}^D)\mathbf{1}_{[1,D-1]}(d)\Big)\mathbf{1}_{\{1\}}(\boldsymbol{E}_{i,t})\mathbf{1}_{\{0\}}(\boldsymbol{E}_{i,t+1}) .\end{aligned} \tag{D.3}$$

The first line represents the case where there is no failure. Then $\boldsymbol{P}_{i,t}$ increases by one if it is different from δ, otherwise it keeps the value δ. When there is a failure, if component i has undergone fewer than D failures, the evolution of $\boldsymbol{P}_{i,t}$ is described by (8.3a). This case is represented by the second line of (D.3). When the component has already undergone D failures, the evolution of $\boldsymbol{P}_{i,t}$ is described by (8.4a). This case is represented by the third line of (D.3). Note that this expression of $\boldsymbol{P}_{i,t+1}^d$ depends on $\boldsymbol{E}_{i,t+1}$. It is possible to express $\boldsymbol{P}_{i,t+1}^d$ only with variables describing component i at time t, this can be done by replacing $\boldsymbol{E}_{i,t+1}$ by its expression (D.1).

D.1.4 Dynamics of the stock $\boldsymbol{S}_t$

We recall the explicit dynamics of the stock that is already given in (8.6):

$$\boldsymbol{S}_{t+1} = \boldsymbol{S}_t + \sum_{i=1}^{n}\sum_{d=1}^{D}\mathbf{1}_{\{D-1\}}(\boldsymbol{P}_{i,t}^d) - \min\left\{\boldsymbol{S}_t,\ \sum_{i=1}^{n}\mathbf{1}_{\{0\}}(\boldsymbol{E}_{i,t})\right\} . \tag{D.4}$$

The dynamics of the whole system has now been explicitly described.

D.2 Explicit expression of the dynamics of the relaxed system

The expression of the relaxed dynamics of parameter $\alpha > 0$ is obtained from Equations (D.1), (D.2), (D.3) and (D.4) by replacing the indicator function with its relaxed version.

We do not always substitute directly the indicator with its relaxation. The dynamics often involves conditions on complementary events. For example, the condition *if the component is broken* is represented by $\mathbf{1}_{\{0\}}(\boldsymbol{E}_{i,t})$. On the other hand, the condition *if the component is healthy* is represented by $\mathbf{1}_{\{1\}}(\boldsymbol{E}_{i,t})$. For the original dynamics, as $\boldsymbol{E}_{i,t} \in \{0, 1\}$, we always have:

$$\mathbf{1}_{\{0\}}(\boldsymbol{E}_{i,t}) + \mathbf{1}_{\{1\}}(\boldsymbol{E}_{i,t}) = 1 \ .$$

This relation is not true anymore using directly the relaxed version of the indicator with the relaxed variables. Take for example $\alpha = 2$, and suppose $\boldsymbol{E}_{i,t} = \frac{1}{2}$, then:

$$\mathbf{1}^{\alpha}_{\{0\}}(\boldsymbol{E}_{i,t}) + \mathbf{1}^{\alpha}_{\{1\}}(\boldsymbol{E}_{i,t}) = 0 \ .$$

If we replace directly all indicator functions by their relaxation, the consequence would be in this case that $\boldsymbol{E}_{i,t+1} = 0$ no matter the control $u_{i,t}$. This means that even if we do a PM with $u_{i,t} = 1$, the component is down at $t + 1$. This does not represent the dynamics of the system as we would expect. To design a coherent relaxed dynamics, complementary conditions $\mathbf{1}_{\mathcal{A}}$ and $\mathbf{1}_{\mathcal{A}^c}$ are represented using the relaxed version $\mathbf{1}^{\alpha}_{\mathcal{A}}$ of the indicator function for the first condition and the function $1 - \mathbf{1}^{\alpha}_{\mathcal{A}}$ for the complementary condition.

D.2.1 Relaxed dynamics of the regime $\boldsymbol{E}_{i,t}$

The relaxed dynamics of the regime of parameter $\alpha > 0$ is given by:

$$\begin{aligned}
\boldsymbol{E}_{i,t+1} &= f^{\alpha}_{i,\boldsymbol{E}}(\boldsymbol{X}_{1:i,t}, \boldsymbol{S}_t, u_{i,t}, \boldsymbol{W}_{i,t+1}) \\
&= \mathbf{1}^{\alpha}_{\mathbb{R}_+}(\boldsymbol{S}_t - \sum_{j=1}^{i} \mathbf{1}^{\alpha}_{\{0\}}(\boldsymbol{E}_{j,t}))\mathbf{1}^{\alpha}_{\{0\}}(\boldsymbol{E}_{i,t}) + \Big(\mathbf{1}^{\alpha}_{\mathbb{R}_+}(u_{i,t} - \nu) \\
&\qquad + \mathbf{1}^{\alpha}_{\mathbb{R}_+}(\boldsymbol{W}_{i,t+1} - p_i(\boldsymbol{A}_{i,t}))(1 - \mathbf{1}^{\alpha}_{\mathbb{R}_+}(u_{i,t} - \nu))\Big)\Big(1 - \mathbf{1}^{\alpha}_{\{0\}}(\boldsymbol{E}_{i,t})\Big) \ .
\end{aligned}$$

We use the relaxed version of the indicator for $\mathbf{1}_{\{0\}}(\boldsymbol{E}_{i,t})$ and $\mathbf{1}_{\mathbb{R}_+}(u_{i,t} - \nu)$. We relax $\mathbf{1}_{\{1\}}(\boldsymbol{E}_{i,t})$ and $\mathbf{1}_{\mathbb{R}^*_+}(\nu - u_{i,t})$ as $1 - \mathbf{1}^{\alpha}_{\{0\}}(\boldsymbol{E}_{i,t})$ and $1 - \mathbf{1}^{\alpha}_{\mathbb{R}_+}(u_{i,t} - \nu)$ respectively.

D.2.2 Relaxed dynamics of the age $\boldsymbol{A}_{i,t}$

The relaxed dynamics of the age of parameter $\alpha > 0$ is given by:

$$\begin{aligned}
\boldsymbol{A}_{i,t+1} &= f^{\alpha}_{i,\boldsymbol{A}}(\boldsymbol{X}_{1:i,t}, \boldsymbol{S}_t, u_{i,t}, \boldsymbol{W}_{i,t+1}) \\
&= (\boldsymbol{A}_{i,t} + 1)\Bigg[\mathbf{1}^{\alpha}_{\mathbb{R}^*_+}(\sum_{j=1}^{i} \mathbf{1}^{\alpha}_{\{0\}}(\boldsymbol{E}_{j,t}) - \boldsymbol{S}_t)\mathbf{1}^{\alpha}_{\{0\}}(\boldsymbol{E}_{i,t}) + \Big((1 - u_{i,t})\mathbf{1}^{\alpha}_{\mathbb{R}_+}(u_{i,t} - \nu) \\
&\qquad + \mathbf{1}^{\alpha}_{\mathbb{R}_+}(\boldsymbol{W}_{i,t+1} - p_i(\boldsymbol{A}_{i,t}))(1 - \mathbf{1}^{\alpha}_{\mathbb{R}_+}(u_{i,t} - \nu))\Big)\Big(1 - \mathbf{1}^{\alpha}_{\{0\}}(\boldsymbol{E}_{i,t})\Big)\Bigg] \ .
\end{aligned}$$

D.2.3 Relaxed dynamics of the vector of last failures $\boldsymbol{P}_{i,t}$

The relaxed dynamics of parameter $\alpha > 0$ of the d-th element of the vector of last failures is given by:

$$\begin{aligned}\boldsymbol{P}^d_{i,t+1} &= f^{d,\alpha}_{i,\boldsymbol{P}}(\boldsymbol{X}_{1:i,t}, \boldsymbol{S}_t, u_{i,t}, \boldsymbol{W}_{i,t+1}) \\ &= \Big((\boldsymbol{P}^d_{i,t}+1)(1-\mathbf{1}^\alpha_{\{\delta\}}(\boldsymbol{P}^d_{i,t})) + \delta \mathbf{1}^\alpha_{\{\delta\}}(\boldsymbol{P}^d_{i,t})\Big)\Big(1-\mathbf{1}^\alpha_{\{1\}}(\boldsymbol{E}_{i,t})\mathbf{1}^\alpha_{\{0\}}(\boldsymbol{E}_{i,t+1})\Big) \\ &\quad + \Big((\boldsymbol{P}^d_{i,t}+1)(1-\mathbf{1}^\alpha_{\{\delta\}}(\boldsymbol{P}^d_{i,t}))\mathbf{1}^\alpha_{\{\delta\}}(\boldsymbol{P}^D_{i,t}) + \delta \mathbf{1}^\alpha_{\{\delta\}}(\boldsymbol{P}^{d-1}_{i,t})\mathbf{1}_{[2,D]}(d) \\ &\quad\quad + (\boldsymbol{P}^{d+1}_{i,t}+1)(1-\mathbf{1}^\alpha_{\{\delta\}}(\boldsymbol{P}^D_{i,t}))\mathbf{1}_{[1,D-1]}(d)\Big)\mathbf{1}^\alpha_{\{1\}}(\boldsymbol{E}_{i,t})\mathbf{1}^\alpha_{\{0\}}(\boldsymbol{E}_{i,t+1}) \ .\end{aligned}$$

We do not relax $\mathbf{1}_{[2,D]}(d)$ and $\mathbf{1}_{[1,D-1]}(d)$. The reason is that these indicator functions do not arise from a discontinuity in the original dynamics. They are just used to take into account in the same equation the cases of $\boldsymbol{P}^1_{i,t}$ and $\boldsymbol{P}^D_{i,t}$ that have a slightly different expression than $\boldsymbol{P}^d_{i,t}$ for $1 < d < D$.

D.2.4 Relaxed dynamics of the stock $\boldsymbol{S}_t$

The relaxed dynamics of the stock of parameter $\alpha > 0$ is given by:

$$\boldsymbol{S}_{t+1} = \boldsymbol{S}_t + \sum_{i=1}^{n}\sum_{d=1}^{D} \mathbf{1}^\alpha_{\{D-1\}}(\boldsymbol{P}^d_{i,t}) - \min\left\{\boldsymbol{S}_t,\ \sum_{i=1}^{n}\mathbf{1}^\alpha_{\{0\}}(\boldsymbol{E}_{i,t})\right\} \ .$$

D.3 Computation of optimal multipliers

At iteration l of the APP fixed-point algorithm, the subproblem on component $i \in I$ is solved with MADS. MADS directly solves the constrained problem and outputs a primal solution $(\boldsymbol{X}^{l+1}_i, u^{l+1}_i)$. Finding the primal solution $\boldsymbol{S}^{l+1}$ of the subproblem on the stock just requires a simulation of the dynamics. For each subproblem, we also have to compute optimal multipliers $\boldsymbol{\Lambda}^{l+1}_1, \ldots, \boldsymbol{\Lambda}^{l+1}_n, \boldsymbol{\Lambda}^{l+1}_{\boldsymbol{S}}$ to update the coordination term at the end of each iteration.

Suppose that the optimal solution and optimal multiplier of the auxiliary problem (10.3) are uniquely defined. As we know the primal solution, we can compute the optimal multiplier using the stationarity of the Lagrangian. In the following calculation, we use the relaxed cost and dynamics to be able to compute the different gradients that appear, however for the sake of readability we drop the superscript α. The Lagrangian L of the auxiliary problem (10.3) is:

$$\begin{aligned}L(\boldsymbol{X}, \boldsymbol{S}, u, \boldsymbol{\Lambda}) = \mathbb{E}\Big(&\sum_{i=1}^{n}\Big(j_i(\boldsymbol{X}_i, u_i) + j^F(\overline{\boldsymbol{X}}_{1:i-1}, \boldsymbol{X}_i, \overline{\boldsymbol{X}}_{i+1:n})\Big) \\ &+ \frac{\gamma_x}{2}\left\|\boldsymbol{X}-\overline{\boldsymbol{X}}\right\|^2 + \frac{\gamma_s}{2}\left\|\boldsymbol{S}-\overline{\boldsymbol{S}}\right\|^2 + \frac{\gamma_u}{2}\left\|u-\overline{u}\right\|^2 \\ &+ \left\langle \overline{\boldsymbol{\Lambda}}\ , (\Theta'(\overline{\boldsymbol{X}}, \overline{\boldsymbol{S}}, \overline{u}, \boldsymbol{W}) - \Phi'(\overline{\boldsymbol{X}}, \overline{\boldsymbol{S}}, \overline{u}, \boldsymbol{W}))\cdot(\boldsymbol{X}, \boldsymbol{S}, u)\right\rangle \\ &+ \left\langle \boldsymbol{\Lambda}\ , \Phi(\boldsymbol{X}, \boldsymbol{S}, u, \boldsymbol{W})\right\rangle\Big) \ .\end{aligned}$$

At the saddle point $(\boldsymbol{X}^\sharp, \boldsymbol{S}^\sharp, u^\sharp, \boldsymbol{\Lambda}^\sharp)$ of L we have:

$$\nabla L(\boldsymbol{X}^\sharp, \boldsymbol{S}^\sharp, u^\sharp, \boldsymbol{\Lambda}^\sharp) = 0 \ . \tag{D.5}$$

Recall that for $i \in \{1, \dots, n, \boldsymbol{S}\}$, we have

$$\boldsymbol{\Lambda}_i^{l+1} = (\boldsymbol{\Lambda}_{i,0}^{l+1}, \dots, \boldsymbol{\Lambda}_{i,T}^{l+1}) \, .$$

Using (D.5) and knowing the solution $(\boldsymbol{X}^\sharp, \boldsymbol{S}^\sharp, u^\sharp)$ of the auxiliary problem, we can update the multiplier $\boldsymbol{\Lambda}^\sharp$ with a backward recursion.

Proposition D.1. *Let $i \in I$. For the dynamics of component i in the auxiliary problem (10.3), the optimal multiplier $\boldsymbol{\Lambda}_i^\sharp = (\boldsymbol{\Lambda}_{i,0}^\sharp, \dots, \boldsymbol{\Lambda}_{i,T}^\sharp)$ can be computed with the following backward recursion for $t \in \mathbb{T}$:*

$$\begin{aligned}
\boldsymbol{\Lambda}_{i,T}^\sharp = & -\nabla_{\boldsymbol{X}_{i,T}} j_{i,T}(\boldsymbol{X}_{i,T}^\sharp) - \nabla_{\boldsymbol{X}_{i,T}} j_T^F(\overline{\boldsymbol{X}}_{1:i-1,T}, \boldsymbol{X}_{i,T}^\sharp, \overline{\boldsymbol{X}}_{i+1:n,T}) - \gamma_x(\boldsymbol{X}_{i,T}^\sharp - \overline{\boldsymbol{X}}_{i,T}) \, , \\
\boldsymbol{\Lambda}_{i,t}^\sharp = & -\nabla_{\boldsymbol{X}_{i,t}} j_{i,t}(\boldsymbol{X}_{i,t}^\sharp, u_{i,t}^\sharp) - \nabla_{\boldsymbol{X}_{i,t}} j_t^F(\overline{\boldsymbol{X}}_{1:i-1,t}, \boldsymbol{X}_{i,t}^\sharp, \overline{\boldsymbol{X}}_{i+1:n,t}) \\
& - \gamma_x(\boldsymbol{X}_{i,t}^\sharp - \overline{\boldsymbol{X}}_{i,t}) - \sum_{j=i+1}^{n} \partial_{\boldsymbol{X}_{i,t}} \Theta_{j,t+1}(\overline{\boldsymbol{X}}_{1:j}, \overline{\boldsymbol{S}}, \overline{u}_j, \boldsymbol{W}_j)^\top \cdot \overline{\boldsymbol{\Lambda}}_{j,t+1} \\
& - \partial_{\boldsymbol{X}_{i,t}} \Theta_{\boldsymbol{S},t+1}(\overline{\boldsymbol{X}}_{1:n}, \overline{\boldsymbol{S}})^\top \cdot \overline{\boldsymbol{\Lambda}}_{\boldsymbol{S},t+1} - \partial_{\boldsymbol{X}_{i,t}} \Phi_{i,t+1}(\boldsymbol{X}_i^\sharp, u_i^\sharp, \boldsymbol{W}_i)^\top \cdot \boldsymbol{\Lambda}_{i,t+1}^\sharp \, .
\end{aligned}$$

Proof. The gradient of the Lagrangian L with respect to $\boldsymbol{X}_{i,t}$, $t \in \mathbb{T}$ is given by:

$$\begin{aligned}
\nabla_{\boldsymbol{X}_{i,T}} L(\boldsymbol{X}^\sharp, \boldsymbol{S}^\sharp, u^\sharp, \boldsymbol{\Lambda}^\sharp) = & \nabla_{\boldsymbol{X}_{i,T}} j_{i,T}(\boldsymbol{X}_{i,T}^\sharp) + \nabla_{\boldsymbol{X}_{i,T}} j_T^F(\overline{\boldsymbol{X}}_{1:i-1,T}, \boldsymbol{X}_{i,T}^\sharp, \overline{\boldsymbol{X}}_{i+1:n,T}) \\
& + \gamma_x(\boldsymbol{X}_{i,T}^\sharp - \overline{\boldsymbol{X}}_{i,T}) + \partial_{\boldsymbol{X}_{i,T}} \Phi_{i,T}(\boldsymbol{X}_i^\sharp, u_i^\sharp, \boldsymbol{W}_i)^\top \cdot \boldsymbol{\Lambda}_{i,T}^\sharp \, , \\
\nabla_{\boldsymbol{X}_{i,t}} L(\boldsymbol{X}^\sharp, \boldsymbol{S}^\sharp, u^\sharp, \boldsymbol{\Lambda}^\sharp) = & \nabla_{\boldsymbol{X}_{i,t}} j_{i,t}(\boldsymbol{X}_{i,t}^\sharp, u_{i,t}^\sharp) + \nabla_{\boldsymbol{X}_{i,t}} j_t^F(\overline{\boldsymbol{X}}_{1:i-1,t}, \boldsymbol{X}_{i,t}^\sharp, \overline{\boldsymbol{X}}_{i+1:n,t}) \\
& + \gamma_x(\boldsymbol{X}_{i,t}^\sharp - \overline{\boldsymbol{X}}_{i,t}) + \partial_{\boldsymbol{X}_{i,t}} \Theta_{\boldsymbol{S},t+1}(\overline{\boldsymbol{X}}_{1:n}, \overline{\boldsymbol{S}})^\top \cdot \overline{\boldsymbol{\Lambda}}_{\boldsymbol{S},t+1} \\
& + \sum_{j=i+1}^{n} \partial_{\boldsymbol{X}_{i,t}} \Theta_{j,t+1}(\overline{\boldsymbol{X}}_{1:j}, \overline{\boldsymbol{S}}, \overline{u}_j, \boldsymbol{W}_j)^\top \cdot \overline{\boldsymbol{\Lambda}}_{j,t+1} \\
& + \partial_{\boldsymbol{X}_{i,t}} \Phi_{i,t}(\boldsymbol{X}_i^\sharp, u_i^\sharp, \boldsymbol{W}_i)^\top \cdot \boldsymbol{\Lambda}_{i,t}^\sharp \\
& + \partial_{\boldsymbol{X}_{i,t}} \Phi_{i,t+1}(\boldsymbol{X}_i^\sharp, u_i^\sharp, \boldsymbol{W}_i)^\top \cdot \boldsymbol{\Lambda}_{i,t+1}^\sharp \, .
\end{aligned}$$

Using:

$$\begin{aligned}
\nabla_{\boldsymbol{X}_{i,t}} L(\boldsymbol{X}^\sharp, \boldsymbol{S}^\sharp, u^\sharp, \boldsymbol{\Lambda}^\sharp) = 0, \quad t \in \mathbb{T} \, , \\
\partial_{\boldsymbol{X}_{i,t}} \Phi_{i,t}(\boldsymbol{X}_i^\sharp, u_i^\sharp, \boldsymbol{W}_i) = \mathrm{I}, \quad t \in \mathbb{T} \, ,
\end{aligned}$$

where I is the identity matrix of appropriate size, we get the desired result.

Proposition D.2. *The optimal multiplier $\boldsymbol{\Lambda}_{\boldsymbol{S}}^\sharp = (\boldsymbol{\Lambda}_{\boldsymbol{S},0}^\sharp, \dots, \boldsymbol{\Lambda}_{\boldsymbol{S},T}^\sharp)$ associated to the dynamics of the stock in the auxiliary problem (10.3) can be computed with the following backward recursion for $t \in \mathbb{T}$:*

$$\begin{aligned}
\boldsymbol{\Lambda}_{\boldsymbol{S},T}^\sharp = & -\gamma_s(\boldsymbol{S}_T^\sharp - \overline{\boldsymbol{S}}_T) \, , \\
\boldsymbol{\Lambda}_{\boldsymbol{S},t}^\sharp = & -\gamma_s(\boldsymbol{S}_t^\sharp - \overline{\boldsymbol{S}}_t) - \sum_{i=1}^{n} \partial_{\boldsymbol{S}_t} \Theta_{i,t+1}(\overline{\boldsymbol{X}}_{1:i}, \overline{\boldsymbol{S}}, \overline{u}_i, \boldsymbol{W}_i)^\top \cdot \overline{\boldsymbol{\Lambda}}_{i,t+1} \\
& - \partial_{\boldsymbol{S}_t} \Phi_{\boldsymbol{S},t+1}(\boldsymbol{S}^\sharp)^\top \cdot \boldsymbol{\Lambda}_{\boldsymbol{S},t+1}^\sharp \, .
\end{aligned} \tag{D.6}$$

Proof. The gradient of the Lagrangian L with respect to $\boldsymbol{S}_t,\ t \in \mathbb{T}$ is given by:

$$\nabla_{\boldsymbol{S}_T} L(\boldsymbol{X}^\sharp, \boldsymbol{S}^\sharp, u^\sharp, \boldsymbol{\Lambda}^\sharp) = \gamma(\boldsymbol{S}_T^\sharp - \overline{\boldsymbol{S}}_T) + \partial_{\boldsymbol{S}_T} \Phi_{\boldsymbol{S},T}(\boldsymbol{S}^\sharp)^\top \cdot \boldsymbol{\Lambda}_{\boldsymbol{S},T}^\sharp \,,$$

$$\begin{aligned}\nabla_{\boldsymbol{S}_t} L(\boldsymbol{X}^\sharp, \boldsymbol{S}^\sharp, u^\sharp, \boldsymbol{\Lambda}^\sharp) = \gamma(\boldsymbol{S}_t^\sharp - \overline{\boldsymbol{S}}_t) + \sum_{i=1}^n \partial_{\boldsymbol{S}_t} \Theta_{i,t+1}(\overline{\boldsymbol{X}}_{1:i}, \overline{\boldsymbol{S}}, \overline{u}_i, \boldsymbol{W}_i)^\top \cdot \overline{\boldsymbol{\Lambda}}_{i,t+1} \\ + \partial_{\boldsymbol{S}_t} \Phi_{\boldsymbol{S},t+1}(\boldsymbol{S}^\sharp)^\top \cdot \boldsymbol{\Lambda}_{\boldsymbol{S},t+1}^\sharp + \partial_{\boldsymbol{S}_t} \Phi_{\boldsymbol{S},t}(\boldsymbol{S}^\sharp)^\top \cdot \boldsymbol{\Lambda}_{\boldsymbol{S},t}^\sharp \,.\end{aligned}$$

Using:

$$\begin{aligned}\nabla_{\boldsymbol{S}_t} L(\boldsymbol{X}^\sharp, \boldsymbol{S}^\sharp, u^\sharp, \boldsymbol{\Lambda}^\sharp) = 0, \quad t \in \mathbb{T} \,, \\ \partial_{\boldsymbol{S}_t} \Phi_{\boldsymbol{S},t}(\boldsymbol{S}^\sharp) = 1, \quad t \in \mathbb{T} \,,\end{aligned}$$

we get the backward recursion (D.6).

D.4 Derivative of the relaxed indicator function

We give some details about the derivative of the relaxed indicator function. The relaxed indicator function $\mathbf{1}_{\mathcal{A}}^\alpha$, where $\alpha > 0$, appears in the dynamics and cost with three main cases for the set $\mathcal{A} \subset \mathbb{R}$. Note that $\mathbf{1}_{\mathcal{A}}^\alpha$ is not differentiable at $x \in \mathbb{R}$ if $d(\mathcal{A}, x) = \frac{1}{2\alpha}$. At such point, the derivative is taken to be 0. The following situations occur:

1. $\mathcal{A}$ is a singleton $\{a\}$, then for $x \in \mathbb{R}$:

$$\mathbf{1}_{\{a\}}^\alpha(x) = \begin{cases} 1 - 2\alpha|x-a| & \text{if } |x-a| \le \dfrac{1}{2\alpha} \,, \\ 0 & \text{if } |x-a| > \dfrac{1}{2\alpha} \,. \end{cases}$$

   Hence the derivative $\mathbf{1}_{\{a\}}^{\prime\alpha}$ is given by:

$$\mathbf{1}_{\{a\}}^{\prime\alpha}(x) = \begin{cases} 2\alpha & \text{if } a - \dfrac{1}{2\alpha} < x < a \,, \\ -2\alpha & \text{if } a < x < a + \dfrac{1}{2\alpha} \,, \\ 0 & \text{otherwise} \ . \end{cases}$$

2. $\mathcal{A} = \mathbb{R}_+$ then for $x \in \mathbb{R}$ we have:

$$\mathbf{1}_{\mathbb{R}_+}^\alpha(x) = \begin{cases} 2\alpha x + 1 & \text{if } -\dfrac{1}{2\alpha} < x < 0 \,, \\ 1 & \text{if } x \ge 0 \,, \\ 0 & \text{if } x \le -\dfrac{1}{2\alpha} \,. \end{cases} \qquad \mathbf{1}_{\mathbb{R}_+}^{\prime\alpha}(x) = \begin{cases} 2\alpha & \text{if } -\dfrac{1}{2\alpha} < x < 0 \,, \\ 0 & \text{otherwise} \,. \end{cases}$$

3. $\mathcal{A} = \mathbb{R}_+^*$: if we strictly apply Definition 10.2, we would have $\mathbf{1}_{\mathbb{R}_+^*}^\alpha = \mathbf{1}_{\mathbb{R}_+}^\alpha$. However with this definition we would not have pointwise convergence of $\mathbf{1}_{\mathbb{R}_+^*}^\alpha$ towards $\mathbf{1}_{\mathbb{R}_+^*}$ as α goes to 0. Indeed, for all $\alpha > 0$ we would have $\mathbf{1}_{\mathbb{R}_+^*}^\alpha(0) = 1$ but $\mathbf{1}_{\mathbb{R}_+^*}(0) = 0$. To overcome this issue we define $\mathbf{1}_{\mathbb{R}_+^*}^\alpha$ as follows:

$$\mathbf{1}_{\mathbb{R}_+^*}^\alpha(x) = \begin{cases} 2\alpha x & \text{if } 0 < x < \dfrac{1}{2\alpha} \,, \\ 1 & \text{if } x \ge \dfrac{1}{2\alpha} \,, \\ 0 & \text{if } x \le 0 \,. \end{cases} \qquad \mathbf{1}_{\mathbb{R}_+^*}^{\prime\alpha}(x) = \begin{cases} 2\alpha & \text{if } 0 < x < \dfrac{1}{2\alpha} \,, \\ 0 & \text{otherwise} \,. \end{cases}$$

Using these formulas for the derivative of the relaxed indicator function and the explicit expressions of the relaxed cost function and relaxed dynamics given in §10.4.3 and Appendix D.2 respectively, all the gradients that appear either in the objective function of the subproblems or in the backward recursion for the multiplier update can be computed and the fixed-point algorithm can be implemented in practice.

E Technical lemmas for the stochastic APP

We give some technical results that are used in the proof of convergence of the stochastic APP algorithm (Theorem 11.33) and for the derivation of efficiency estimates in §11.4.2.

Lemma E.1. *Let $\{a_i\}_{i\in\mathbb{N}}$ be a sequence in $\mathbb{R}$. Let $n \in \mathbb{N}$ and for $i \in \{1,\dots,n-1\}$, let $s_i = \sum_{l=n-i}^{n} a_l$. Then,*

$$a_n = \frac{s_{n-1}}{n} + \sum_{i=1}^{n-1} \frac{1}{i(i+1)}(s_{i-1} - i a_{n-i}) \ .$$

Proof. We have $s_i = s_{i-1} + a_{n-i}$, so:

$$\begin{aligned}\frac{1}{i}s_{i-1} - \frac{1}{i+1}s_i &= \frac{1}{i(i+1)}\left((i+1)s_{i-1} - i s_i\right) \\ &= \frac{1}{i(i+1)}\left(s_{i-1} - i a_{n-i}\right) \ .\end{aligned}$$

Summing over $1 \le i \le n-1$, we get:

$$a_n - \frac{s_{n-1}}{n} = \sum_{i=1}^{n-1} \frac{1}{i(i+1)}\left(s_{i-1} - i a_{n-i}\right) \ ,$$

giving the desired result. □

Theorem E.2. *[Robbins and Siegmund, 1971] Consider four sequences of nonnegative random variables $\{\boldsymbol{\Lambda}_l\}_{l\in\mathbb{N}}$, $\{\boldsymbol{\alpha}_l\}_{l\in\mathbb{N}}$, $\{\boldsymbol{\beta}_l\}_{l\in\mathbb{N}}$ and $\{\boldsymbol{\eta}_l\}_{l\in\mathbb{N}}$, that are all adapted to a given filtration $\{\mathcal{F}_l\}_{l\in\mathbb{N}}$. Moreover, suppose that:*

$$\mathbb{E}\Big(\boldsymbol{\Lambda}_{l+1} \ \Big|\ \mathcal{F}_l\Big) \le (1+\boldsymbol{\alpha}_l)\boldsymbol{\Lambda}_l + \boldsymbol{\beta}_l - \boldsymbol{\eta}_l, \ \forall l \in \mathbb{N} \ ,$$

and that we have:

$$\sum_{l\in\mathbb{N}} \boldsymbol{\alpha}_l < +\infty \ , \ \ \sum_{l\in\mathbb{N}} \boldsymbol{\beta}_l < +\infty \ , \ \ \mathbb{P}\text{-}a.s. \ .$$

Then, the sequence of random variables $\{\boldsymbol{\Lambda}_l\}_{l\in\mathbb{N}}$ converges almost surely to $\boldsymbol{\Lambda}^\infty$, an almost surely bounded random variable[1], and we have in addition that:

$$\sum_{l\in\mathbb{N}} \boldsymbol{\eta}_l < +\infty \ , \ \ \mathbb{P}\text{-}a.s. \ .$$

[1] A random variable $\boldsymbol{X}$ is bounded $\mathbb{P}$-a.s. if it is such that: $\mathbb{P}\big(\{\omega \in \Omega \mid \boldsymbol{X}(\omega) = +\infty\}\big) = 0$.

An extension of Robbins-Siegmund theorem is given by the following corollary.

Corollary E.3. *Consider five sequences of nonnegative random variables* $\{\boldsymbol{\Lambda}_l\}_{l\in\mathbb{N}}$, $\{\boldsymbol{\alpha}_l\}_{l\in\mathbb{N}}$, $\{\boldsymbol{\beta}_l\}_{l\in\mathbb{N}}$, $\{\boldsymbol{\gamma}_l\}_{l\in\mathbb{N}}$, *and* $\{\boldsymbol{\eta}_l\}_{l\in\mathbb{N}}$, *that are all adapted to a given filtration* $\{\mathcal{F}_l\}_{l\in\mathbb{N}}$. *Moreover suppose that:*

$$\mathbb{E}\Big(\boldsymbol{\Lambda}_{l+1} \;\Big|\; \mathcal{F}_l\Big) \leq \Big(1+\boldsymbol{\alpha}_l\Big)\boldsymbol{\Lambda}_l + \boldsymbol{\beta}_l\mathbb{E}\Big(\boldsymbol{\Lambda}_{l+1} \;\Big|\; \mathcal{F}_l\Big) + \boldsymbol{\gamma}_l - \boldsymbol{\eta}_l \;,$$

and that we have:

$$\sum_{l\in\mathbb{N}} \boldsymbol{\alpha}_l < +\infty \;,\;\; \sum_{l\in\mathbb{N}} \boldsymbol{\beta}_l < +\infty \;,\;\; \sum_{l\in\mathbb{N}} \boldsymbol{\gamma}_l < +\infty \;,\;\; \mathbb{P}\text{-}p.s. \;.$$

Then, the sequence of random variables $\{\boldsymbol{\Lambda}_l\}_{l\in\mathbb{N}}$ *converges almost surely to* $\boldsymbol{\Lambda}^\infty$, *an almost surely bounded random variable and we have in addition that:*

$$\sum_{l\in\mathbb{N}} \boldsymbol{\eta}_l < +\infty \;,\;\; \mathbb{P}\text{-}p.s. \;.$$

Proof. Consider a realization of the different sequences satisfying the assumptions of the corollary, and define three sequences $\{\widetilde{\alpha}_l\}_{l\in\mathbb{N}}$, $\{\widetilde{\gamma}_l\}_{l\in\mathbb{N}}$ and $\{\widetilde{\eta}_l\}_{l\in\mathbb{N}}$ such that:

$$1+\widetilde{\alpha}_l = \frac{1+\alpha_l}{1-\beta_l} \;,\;\; \widetilde{\gamma}_l = \frac{\gamma_l}{1-\beta_l} \;,\;\; \widetilde{\eta}_l = \frac{\eta_l}{1-\beta_l} \;.$$

As the sequence $\{\beta_l\}_{l\in\mathbb{N}}$ converges to zero, we have that $\beta_l \leq 1/2$ for l large enough. For such l, we get:

$$\frac{1}{1-\beta_l} \leq 1+2\beta_l \quad \text{and} \quad 1 \leq \frac{1}{1-\beta_l} \leq 2 \;.$$

Then, we deduce that $\widetilde{\alpha}_l \leq 2(\alpha_l+\beta_l)$, $\widetilde{\gamma}_l \leq 2\gamma_l$ and $\widetilde{\eta}_l \geq \eta_l$. The conclusions of the corollary are then obtained by applying Theorem E.2 directly. □

Proposition E.4. *Consider a function* $J : \mathbb{U} \to \mathbb{R}$ *that is subdifferentiable on a nonempty, closed, convex subset* U^{ad} *of* $\mathbb{U}$, *with linearly bounded subgradient. Then, there exist* $c_1 > 0$ *and* $c_2 > 0$ *such that:*

$$\forall (u,v) \in U^{\mathrm{ad}} \times U^{\mathrm{ad}} \;,\;\; \Big|J(u)-J(v)\Big| \leq \Big(c_1 \max\Big\{\|u\|,\|v\|\Big\} + c_2\Big)\,\|u-v\| \;.$$

Proof. Let $(u,v) \in U^{\mathrm{ad}} \times U^{\mathrm{ad}}$. By the definition of subdifferentiability,

$$\begin{aligned} &\forall r \in \partial J(u) \;,\;\; J(v) \geq J(u) + \langle r \,, v-u\rangle \;, \\ &\forall s \in \partial J(v) \;,\;\; J(u) \geq J(v) + \langle s \,, u-v\rangle \;, \end{aligned}$$

from which we get:

$$\langle s \,, u-v\rangle \leq J(u)-J(v) \leq \langle r \,, u-v\rangle \;,$$

and therefore:

$$\Big|J(u)-J(v)\Big| \leq \max\Big\{\langle r \,, u-v\rangle, \langle s \,, v-u\rangle\Big\} \;.$$

Using Schwarz inequality and the linearly bounded subgradient assumption we have:

$$\begin{aligned} \Big|J(u)-J(v)\Big| &\leq \max\Big\{\|r\|\|u-v\|, \|s\|\|v-u\|\Big\} \;, \\ &\leq \Big(c_1 \max\Big\{\|u\|,\|v\|\Big\} + c_2\Big)\,\|u-v\| \;. \end{aligned}$$

for some $c_1 > 0$ and $c_2 > 0$, which gives the desired result. □

Corollary E.5. *A function $J : \mathbb{U} \to \mathbb{R}$ that satisfies the assumptions of Proposition E.4 is Lipschitz continuous on every bounded subset that is contained in U^{ad}.*

Proposition E.6. *Let $J : \mathbb{U} \to \mathbb{R}$ be a Lipschitz continuous function with constant $L > 0$. Let $\{u_l\}_{l\in\mathbb{N}}$ be a sequence of elements in $\mathbb{U}$ and let $\{\varepsilon_l\}_{l\in\mathbb{N}}$ be a real positive sequence such that:*

(a) $\sum_{l\in\mathbb{N}} \varepsilon_l = +\infty$,

(b) $\exists \mu \in \mathbb{R},\ \sum_{l\in\mathbb{N}} \varepsilon_l \left|J(u_l) - \mu\right| < +\infty$,

(c) $\exists \delta > 0,\ \forall l \in \mathbb{N},\ \|u_{l+1} - u_l\| \leq \delta\varepsilon_l$.

Then, the sequence $\left\{J(u_l)\right\}_{l\in\mathbb{N}}$ converges to μ.

Proof. For $\alpha > 0$, define:

$$N_\alpha = \left\{l \in \mathbb{N},\ \left|J(u_l) - \mu\right| \leq \alpha\right\}, \quad N_\alpha^{\complement} = \mathbb{N} \setminus N_\alpha \ .$$

(i) From Assumption (b), we have:

$$+\infty > \sum_{l\in\mathbb{N}} \varepsilon_l \Big|J(u_l) - \mu\Big| \geq \sum_{l\in N_\alpha^{\complement}} \varepsilon_l \Big|J(u_l) - \mu\Big| \geq \alpha \sum_{l\in N_\alpha^{\complement}} \varepsilon_l \ ,$$

from which we get that:

$$\forall \beta > 0 \ , \ \exists n_\beta \in \mathbb{N} \text{ such that } \sum_{l\geq n_\beta, l\in N_\alpha^{\complement}} \varepsilon_l \leq \beta \ .$$

(ii) From Assumption (a), we have:

$$+\infty = \sum_{l\in\mathbb{N}} \varepsilon_l = \sum_{l\in N_\alpha} \varepsilon_l + \sum_{l\in N_\alpha^{\complement}} \varepsilon_l \ ,$$

but we have just proved that the last sum in the above equality is finite, hence the first sum of the right hand side is infinite, which implies that N_α is infinite.

Let $\epsilon > 0$, choose $\alpha = \epsilon/2$ and $\beta = \epsilon/(2L\delta)$ (where L is the Lipschitz constant of J). Let n_β be the integer defined in (i). For $l \geq n_\beta$, there are two possible cases:

- $l \in N_\alpha$: then, by definition of N_α:
$$\Big|J(u_l) - \mu\Big| \leq \alpha < \epsilon \ ,$$
- $l \notin N_\alpha$: let m be the smallest element of N_α such that $m \geq l$, this element exists by (ii). Using the fact that J is Lipschitz continuous jointly with Assumption (c) and the point (i), it comes:
$$\begin{aligned}
\Big|J(u_l) - \mu\Big| &\leq \Big|J(u_l) - J(u_m)\Big| + \Big|J(u_m) - \mu\Big| \\
&\leq L\|u_l - u_m\| + \alpha \\
&\leq L\delta\left(\sum_{k=l}^{m-1} \varepsilon_k\right) + \alpha \\
&\leq L\delta\left(\sum_{k\geq n_\beta, k\in N_\alpha^{\complement}} \varepsilon_k\right) + \alpha \\
&\leq \epsilon \ ,
\end{aligned}$$

so, we get $\Big|J(u_l) - \mu\Big| \leq \epsilon$ for all $l \geq n_\beta$, giving the desired result. □