DOCTOR OF SCIENCES

Trust-region algorithms for nonlinear stochastic programming and mixed logit models

Bastin, Fabian

Award date:
2004

Awarding institution:
University of Namur

FACULTES UNIVERSITAIRES NOTRE-DAME DE LA PAIX NAMUR

FACULTE DES SCIENCES

Trust-Region Algorithms for Nonlinear Stochastic Programming and Mixed Logit Models

Dissertation présentée par
Fabian Bastin
pour l'obtention du grade
de Docteur en Sciences

Composition du Jury:

Michel BIERLAIRE
François LOUVEAUX (Co-Promoteur)
John POLAK
Marcel RÉMON
Annick SARTENAER
Philippe L. TOINT (Promoteur)

2004

Imprimé en Belgique

ISBN: 2-87037-446-1
Dépôt légal: D / 2004 / 1881 / 19

Facultés Universitaires Notre-Dame de la Paix
Faculté des Sciences
rue de Bruxelles, 61, B-5000 Namur, Belgium

Facultés Universitaires Notre-Dame de la Paix
Faculté des Sciences
Rue de Bruxelles, 61, B-5000 Namur, Belgium

Algorithmes de région de confiance pour la programmation stochastique non-linéaire et les modèles mixed logit

par Fabian Bastin

Résumé: Ce travail a pour objet l'étude de la programmation stochastique nonlinéaire non-convexe, en particulier dans le contexte des approches de région de confiance. Nous explorons tout d'abord comment exploiter la structure des programmes stochastiques nonlinéaires multi-étapes avec contraintes linéaires, au sein de méthodes de point intérieurs primales-duales. Nous étudions ensuite la consistance des approximations par moyenne d'échantillonnage pour des programmes stochastiques nonlinéaires généraux. Nous développons également un nouvel algorithme pour résoudre le problème approximé au moyen d'une stratégie interne permettant de varier la taille d'échantillonnage utilisée. Nous jugeons finalement de l'efficacité numérique de la méthode proposée pour l'estimation de modèles de choix discrets, plus précisément les modèles mixed logit, à l'aide de notre logiciel AMLET, écrit pour les besoins.

Trust-region algorithms for nonlinear stochastic programming and mixed logit models

by Fabian Bastin

Abstract: This work is concerned with the study of nonlinear nonconvex stochastic programming, in particular in the context of trust-region approaches. We first explore how to exploit the structure of multistage stochastic nonlinear programs with linear constraints, in the framework of primal-dual interior point methods. We next study consistency of sample average approximations (SAA) for general nonlinear stochastic programs. We also develop a new algorithm to solve the SAA problem, using the statistical inference information to reduce numercial costs, by means of an internal variable sample size strategy. We finally assess the numerical efficiency of the proposed method for the estimation of discrete choice models, more precisely mixed logit models, using our software AMLET, written for this purpose.

Dissertation doctorale en Sciences mathématiques (Ph.D. thesis in Mathematics)
Date: 12-03-2004
Département de Mathématique
Promoteur (Advisor): Prof. Ph. L. TOINT
Co-Promoteur (Co-Advisor): Prof. F. LOUVEAUX

Remerciements

Mes premiers remerciements s'adresseront à Philippe Toint, mon promoteur, qui m'a donné la chance de travailler au sein du Groupe de Recherche sur les Transports (GRT), puis au sein de l'Unité d'Analyse Numérique. Je voudrais le remercier pour la confiance qu'il n'a cessé de m'accorder, même dans les moments difficiles, et les encouragements qui m'ont permis de produire la présente thèse. Merci aussi à François Louveaux, mon co-promoteur, pour m'avoir guidé dans mes premiers pas en optimisation stochastique.

Ma reconnaissance s'adresse également au Fonds National de la Recherche Scientifique, qui m'a accordé une bourse d'aspirant et m'a ainsi permis de mener à bien mon travail de thèse.

Merci à Cinzia Cirillo, collègue de bureau et de travail, pour son accent italien d'une part mais surtout pour la précieuse collaboration qui s'est nouée dans l'étude des modèles de choix discrets. Je redoute cependant qu'elle ne soit pas prête d'oublier une marche forcée dans les montagnes bordant Lucerne. Merci à Annick Sartenaer, ainsi qu'à Jie Sun, avec qui j'ai pu explorer les approches de points intérieurs en programmation avec scénarios. Je voudrais les remercier en particulier pour leurs encouragements et leur extrême sympathie à mon égard.

Merci à Marcel Rémon, pour sa permanente disponibilité pour répondre à mes questions, et son engagement de tous les jours pour les autres.

De nombreux autres chercheurs et scientifiques m'ont aussi aidé et soutenu. Je citerai ici Hatem Masri qui m'a de plus offert de découvrir son pays, la Tunisie, et sa famille, dont l'accueil fut des plus chaleureux. Je voudrais également remercier Rudiger Schültz et Michel Bierlaire, pour leurs pertinents conseils, ainsi que Stéphane Hess dont les discussions se sont révélées particulièrement pertinentes.

Merci à mes collègues d'analyse numérique, à savoir Benoît Colson, que j'invite d'ores et déjà à la prochaine démonstration de "comment cracker le PC du secrétariat", Katia Demaseure, dite 4chats, et Caroline Sainvitu, inoubliable en sortie, et dont le prêt de sa machine, "Tore", fut des plus utiles lors des tests numériques. Merci aussi à Jean Tsimenga et Émilie Wanufelle, nouvellement arrivés dans l'équipe.

Merci à Jean-Paul Hubert, du GRT, pour nos nombreux débats durant les dîners, ou devrais-je dire déjeuner pour respecter ses origines parisiennes.

Merci à mes collègues du Département de Mathématique, et en particulier Eric Cornélis, Bernard De Saedeleer, André Fuzfa, Murielle Haguinet, Jacques Henrard, Jean-Paul Rasson.

Merci aux membres du comité du Namur LUG[1], anciens ou actuels, avec qui j'ai passé de nombreux moments pour promouvoir l'idée que l'information est la propriété de tous. Je citerai en particulier Frédéric Burlet, Christophe Chisogne, Nicolas Di Pietro, Vincent Fally, Emmanuel Koch, Rémi Letot, Arnaud Ligot, Frédéric Renaud, Cyril Romain, François Schoubben, Louis Swinnen et Jean-François Wauthy. Merci aussi à Serge Lambert, autre compagnon linuxien, caractérisé par une perpétuelle bonne humeur.

Je voudrais également remercier mes amis tireurs, en particulier Raymonde Albessart, Laurent Bacq, Christian Bungeneers, Raymond Chabot, Michel Chardon, Guy Denotte, Michel Denruyter, Eric Félix, Steve Flammang, Patrick My, Joseph Pirard, Cyril Smidts, Stéphane Thoumsin, Constantin Tzoumacas.

Je voudrais de plus remercier mes parents ainsi que toutes les personnes de ma famille, présents ou disparus, qui ont fait de moi ce que je suis. Merci aussi à tous ceux qui m'ont encouragé à titres divers. Leur liste est trop longue que pour la dresser ici ; je mentionnerai toutefois en particulier Lionel Ileka Ahuré, Bernard Noël, Jean-Philippe Tossut.

Enfin, merci à Xavier Struyven, qui s'est révélé un ami rare, comme il est donné à peu d'avoir. Merci aussi à Charline Marroy, en qui je n'ai toujours pas trouvé les limites de l'affection dont elle est capable, et sans qui je n'aurais pas trouvé la force d'aller jusqu'au bout de ce travail.

À tous, encore merci.

Fabian
Aspirant au FNRS

[1] `http://www.namurlug.org`

Contents

List of Definitions

Introduction

This work is concerned with the study of nonlinear stochastic programming. Mathematical programming is the study of problems where the goal is to find the optimal value of a given mathematical function called the objective function. The optimum may be the minimum or the maximum of the considered function depending on the problem formulation, and has often to satisfy constraints. All parameters present in the program are usually assumed to be perfectly known; however, in many situations, the reality cannot be captured entirely, for instance when decisions have to be taken regarding observations that are not currently available and not perfectly predictable. The aim of stochastic programming is precisely to find an optimum in problems involving uncertain data. In this terminology, stochastic is opposed to deterministic and means that some data are random.

This thesis deals more specifically with nonlinear nonconvex stochastic programming, in particular in the context of trust-region methods, which has proven to be very efficient in nonlinear programming, but has only received little attention in stochastic programming.

Structure of the document and contributions

We divide the presentation in three parts. Stochastic programming is a rather complex subject; this complexity is in our view not inherent to stochastic programming itself, but lies in the mix of different large fields of mathematics: optimization, probability and statistics. Therefore we present in this work a relatively large introduction in Part I, composed of the three first chapters. The coverage of these fields, and more specifically nonlinear and stochastic programming, is nevertheless far from being exhaustive; our intention is primarily to give some intuition about stochastic programming, by comparison to classical nonlinear programming, in particular for nonconvex problems. Chapter 1 opens discussion about how uncertainty can be incorporated in a mathematical program, and exposes the basic mathematical concepts useful for our needs. The second chapter is a quick review of nonlinear programming; covering the essential tools that will be used in the thesis. Finally Chapter 3 presents classical formulations of nonlinear stochastic programs.

Part 2, made of the fourth and the fifth chapters, presents the two classes of stochastic problems considered in the present work. Chapter 4 considers nonlinear stochastic programs with scenarios, and linear and nonnegativity constraints. An interior point trust-region algorithm is proposed along with a new decomposition scheme for the primal-dual systems occurring during the inner iterations. Chapter 5 explores Monte Carlo sampling techniques for nonlinear stochastic programs. We present new consistency results for the solutions obtained when solving the

sample average approximations and present a new trust-region algorithm that takes advantage of the statistical inference to improve its numerical efficiency.

Finally, in Part 3, composed of the two last chapters, we adapt the results of Chapter 5 to the field of discrete choice theory, and more precisely mixed logit models. Chapter 6 introduces discrete choice models, and we study the mixed logit models as an extension of usual stochastic programs. Chapter 7 presents our software AMLET (Another Mixed Logit Estimation Tool), based on the algorithm presented in Chapter 5, and written from scratch. We also discuss the main numerical results obtained with it. Our main contributions are therefore presented in Part 2 and Part 3.[2]

We conclude this introduction by replicating the citation made by Nocedal and Wright [104], in preface of their 1999 book, and borrowed from Fletcher [53], who described the field of optimization as a "fascinating blend of theory and computation, heuristics and rigor". We think indeed that stochastic programming is a nice illustration of these words.

[2]The larger coverage of the Monte Carlo approximations compared to the primal-dual approaches reflects more our current achievements than any judgement concerning the importance of one technique over the other.

Summary of contributions

Our contributions are the study and the design of algorithms for nonlinear stochastic programming, and applications in discrete choice theory, more specifically to mixed logit models. We summarize these contributions below.

Interior-point methods The decomposition scheme for the Newton primal-dual systems related to nonlinear stochastic programs with linear and nonnegativity constraints (Section 4.3.5) and the QR factorization with partial pivoting for elimination of redundant nonanticipativity constraints (Section 4.2.1) are new, and have been presented in the SIAM Conference on Optimization (Toronto, 2002) by myself and the ISMP conference (Copenhagen, 2003), by Annick Sartenaer [11]. The primal-dual interior-point method (Algorithms 4.1 and 4.3) are not new in their essence, but represent one of the first applications of interior-point approaches to nonlinear nonconvex stochastic programming.

Monte Carlo samplings The consistency of solutions obtained when using Monte-Carlo approximations in nonlinear stochastic programming is new and has been studied in [10], where new results have been derived, in particular about conditions ensuring asymptotic second-order criticality.

Mixed logit models The developments obtained in the study of Monte Carlo approximations in nonlinear stochastic programming have been applied to mixed logit models to prove a new result on almost-sure convergence of the Monte Carlo estimators [10], and in the development of a new software, AMLET [8, 9].

Note also that a survey of stochastic programming and related applications can be found in our master's thesis [7], published in 2001.

Part I

An introduction to nonlinear stochastic programming

Chapter 1

Why stochastic programming?

"Nature does not play dice."
Albert Einstein.

1.1 A world involved in optimization

Optimization is a major component of many physical phenomena as well as human life. For instance, numerous natural observations can be explained by the minimization of the involved energy. People also optimize each time they want to maximize or minimize something, such as the travel time, the profit, the number of tasks during a defined laps of time. We then want to make the best decisions with respect to some criteria, taking operational constraints into account. Many day-to-day tasks involve optimization, while most of the time we intuitively perform such optimization activities, by using for instance our past experience of similar situations.

However we cannot rely on such an intuitive behaviour in many complex situations. With the emergence of computational resources, operational research has grown up to be now a major field in applied mathematics. Applications can be found in an impressive number of disciplines, like economics, social management, planning, engineering, biology, physics, and chemistry. They focus on situations where the system can be affected or controlled by outside decisions that should be selected in the best possible manner. To this end, the notion of an optimization problem has proved very useful. The alternatives open to the decision maker can be expressed in terms of a set $\mathcal{F}$ whose elements are called feasible solutions. The goal is then to optimize over $\mathcal{F}$ a certain function f known as the objective function. The objective is a quantitative measure of the performance (or output) of the system under study, and depends on some variables or unknowns. The aim is to assign values to these variables in order to minimize or maximize the objective. In many situations, variables are subject to constraints, which define some restrictions on the acceptable values. Such constraints may arise from the properties of the studied problem (for instance the maximum of revolutions per minutes for a car engine before it breaks), from physical constraints (e.g., available quantity of raw materials), or from operational decisions (e.g. upper limits on investments).

The process of identifying objective, variables, and constraints for a given problem is known as modelling. The construction of an appropriate model is the first step in the optimization

process, and is crucial for the subsequent stages. If the model is too simplistic, it will not give useful insight into the practical problem. If it is too complex, it may be difficult, or impossible, to solve it. Moreover, the model will be expressed in mathematical terms. Therefore some aspects of the problem have to be neglected, but this can lead to hazardous models in some situations.

In particular, it is usually assumed that one has precise information about the objective function f and the constraints. Parameters are assumed to be known exactly, so we suppose implicitly that the underlying aspects are perfectly known. In practice, however, this is not the case for many optimization problems. For instance, measures of physical phenomena are biased with errors. Several sources of errors also exist in the context of industrial production, as the differences between machines and the raw materials (quantity and quality). Other processes are random by definition. For most of the economic problems, demands are uncertain. Farming productions depend of climatic conditions. The question is therefore how we can take account of the involved uncertainty. One possible answer is to incorporate random variables into the model; another one is to use fuzzy numbers. We will focus here on the first approach, which is known as stochastic programming (SP).

1.2 Mathematical and stochastic programming

1.2.1 Formulation of mathematical programs

A mathematical program is an optimization problem aiming at minimizing some objective function taking some possible constraints into account. Unconstrained optimization problems may be written under the following (standard) form

$$\min_{x\in\mathbb{R}^n} f(x),$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is the objective function. Constrained optimization problems require that the variable x belongs to some feasible region $\mathcal{F}$, whose prototype is

$$\mathcal{F} = \{x \in \mathbb{R}^n | x \in X,\ c_i(x) \leq 0,\ i \in \mathcal{I},\ c_i(x) = 0,\ i \in \mathcal{E}\},$$

where $\mathcal{I}$ and $\mathcal{E}$ are (disjoint) index sets, $c_i : \mathbb{R}^n \to \mathbb{R}$ ($i \in \mathcal{I} \cup \mathcal{E}$) and X is a subset of $\mathbb{R}^n$ and belongs to the domain of the functions f and c_i ($i \in \mathcal{E} \cup \mathcal{I}$). X is usually of simple character, say $\mathbb{R}^n_+$ or $\mathbb{R}^n$. This leads to the following program:

$$\min_{x\in X} f(x) \tag{1.1}$$

$$\text{s.t. } c_i(x) = 0,\ i \in \mathcal{E}, \tag{1.2}$$

$$c_i(x) \leq 0,\ i \in \mathcal{I}. \tag{1.3}$$

The relations $c_i(x) \leq 0$ ($i \in \mathcal{I}$) and $c_i(x) = 0$ ($i \in \mathcal{E}$) are called inequality and equality constraints respectively. A point $x \in X$ is said to be feasible if it satisfies all the constraints, in other terms if $x \in \mathcal{F}$. Assuming that the cardinality of the sets $\mathcal{E}$ and $\mathcal{I}$ is given by m_1 and m_2 respectively, we can gather the constraint functions into the mapping

$$\begin{aligned} c :& \mathbb{R}^n \to \mathbb{R}^m \\ & x \rightsquigarrow (c_i(x))_{i\in\mathcal{E}\cup\mathcal{I}}, \end{aligned}$$

where $m = m_1 + m_2$.

It is sometimes convenient to consider strictly feasible points, i.e., points such that the inequality constraints are strictly satisfied. We denote the strictly feasible set by

$$\text{strict}\{\mathcal{F}\} \stackrel{def}{=} \{x \in \mathbb{R}^n | x \in X, c_i(x) < 0,\ i \in \mathcal{I}, c_i(x) = 0,\ i \in \mathcal{E}\}.$$

1.2.2 A quick introduction to stochastic programming

Consider the general mathematical programming problem (1.1)–(1.3). For many optimization problems, the functions f and c_i ($i = 1, \ldots, m$) are not known very accurately and in those cases, it is fruitful to think of the functions c_i as depending on a pair of variables $(x, \xi(\omega))$, where ω is a vector that takes its values in a set Ω, and $\boldsymbol{\xi}$ is a real random vector with the support $\Xi \subset \mathbb{R}^q$. We may think of $\xi(\omega)$ as the environment-determining variable that conditions the system under investigation. A decision x then results in different outcomes

$$\left(f(x, \xi(\omega)), c_1(x, \xi(\omega)), \ldots, c_m(x, \xi(\omega))\right), \tag{1.4}$$

depending on the uncontrollable factors. More formally, the problem can now be reformulated as

$$\text{“}\min_{x \in X}\text{”}\ f(x, \xi(\omega)) \tag{1.5}$$

$$\text{s.t. } c_i(x, \xi(\omega)) = 0,\ i \in \mathcal{E}, \tag{1.6}$$

$$c_i(x, \xi(\omega)) \leq 0,\ i \in \mathcal{I}, \tag{1.7}$$

where $X \subseteq \mathbb{R}^n$. We assume that the probability distribution P is given and is independent of x, and that, for all x, $f(x, \cdot) : \Xi \to \mathbb{R}$ and $c_i(x, \cdot) : \Xi \to \mathbb{R}$ ($i = 1, \ldots, m$) are random variables themselves (see Section 1.4 for an introduction to probability theory).

We seek for some x that is feasible and that minimizes the objective for all or for nearly all possible values of ω in Ω, or in other sense that needs to be specified. Any fixed $x \in X$ may be feasible for some $\omega \in \Omega$, but infeasible for some other $\omega' \in \Omega$. The notion of feasibility therefore needs to be made precise and depends on the problem at hand, in particular whether or not we are able to obtain some information about the value of $\xi(\omega)$, before choosing x. Similarly, what must be understood by optimality depends on the uncertainty involved as well as on the view one may have of the overall objective(s), e.g., avoid a disastrous situation, do well in nearly all cases,... We cannot “solve” (1.5)–(1.6) by finding the optimal solution for every possible value of ω in Ω. We must therefore precise the meanings of “$\min$” as well of the constraints.

Ideally ω, and thus $\xi(\omega)$, would be completely known before we have to choose x. The optimal solution of (1.5)–(1.6) could be then obtained by assigning to the variables ω the known values of these parameters. Even in this case, there arises the question of implementability. Namely, how to design a practical (implementable) decision rule

$$\omega \to x^*(\xi(\omega))$$

such that $x^*(\xi(\omega))$ is feasible for (nearly) all $\omega \in \Omega$, and that is “optimal” in some sense. Ideally, $x(\xi(\omega))$ minimizes $f(\xi(\omega))$ on $S(\xi(\omega))$ for all $\omega \in \Omega$, where $S(\xi(\omega))$ is defined by

$$S(\xi(\omega)) = \{x \in X |\, x \in X,\ c_i(\xi(\omega)) = 0,\ i \in \mathcal{E},\ \text{and } c_i(\xi(\omega)) \leq 0,\ i \in \mathcal{I}\}.$$

For Ermoliev [48], such an ideal decision rule is only rarely simple enough to be implementable, so the notion of optimality must be redefined in order to make the search for such a decision rule meaningful.

More usually, the goal is to take a decision on x, or part of it, before knowing the realization of the random variable $\boldsymbol{\xi}$. When no information is available about ω before the choice of x is made, (1.5)–(1.7) can be analyzed in terms of the vector (1.4), as ω varies in Ω. We could then formulate (1.5)–(1.7) as

$$\begin{aligned} \min_{x \in X} z(x) &= \{ f(x, \xi(\omega)) \,|\, \omega \in \Omega \}, \\ \text{s.t. } & c_i(x) = 0, \; i = 1, \ldots, \mathcal{E}, \\ & c_i(x) \leq 0, \; i = 1, \ldots, \mathcal{I}. \end{aligned}$$

If Ξ is finite, this becomes a multiobjective optimization problem. The reader interested in such problems can refer to, amongst others, Kaisa [79], Coello, Van Veldhuizen and Lamont [28]. The optimality is then most commonly defined with the concept of Pareto optimality. Consider the multiobjective programming problem

$$\begin{aligned} & \min f(x) \stackrel{def}{=} (f_1(x), f_2(x), \ldots, f_k(x))^T \\ & \text{s.t. } x \in S = \{ x \in X | \, c_i(x) \leq 0, \; i \in \mathcal{I}, \text{ and } c_i(x) = 0, \; i \in \mathcal{E} \}. \end{aligned}$$

In our context, $k = \#\Xi$, and each component of $f(x)$ corresponds to some realization ξ.

Since there is no total order in $\mathbb{R}^n$, we must reexamine the notion of minimization. The Pareto optimality is defined as follows.

Definition 1.1: Pareto optimality
x^* is a Pareto-optimal solution if there is no other $x \in X$ such that $f_i(x) \leq f_i(x^*), i = 1, \ldots, k$ and $f_j(x) < f_j(x^*)$ for at least one j.

There still remains the question of how to choose a (unique) decision among the Pareto-optimal points. A popular approach is to proceed by "worst-case analysis". For a given x, one calculates the worst situation that could happen, in terms of all the objectives, and then chooses a solution that minimizes the value of the worst-case loss. This should single out some point that is optimal in a pessimistic minimax sense, but this solution is often not very satisfactory. We have therefore to modify the program (1.5)–(1.7) so that the minimization can be undertaken without knowing the realization of the random variable, while the objective remains unique. This leads to the developments of deterministic equivalents for (1.5)–(1.7), whose formulations influence both methods and interpretation of the problem (see Chapter 3).

1.2.3 Other approaches in uncertainty modelling

Incorporating uncertainty in the optimization problem requires some knowledge about the uncertainty. Until now we have supposed that this knowledge was sufficient to develop random variables, but this is not always the case. Therefore other ways of representing uncertainty in mathematical programming have been considered. To date, we can distinguish two major

approaches: stochastic programming and fuzzy programming. The availability of historical data often determines which approach to use.

Stochastic programming Uncertain parameters are represented as random variables. The challenge is to choose good probabilistic distributions. A good knowledge of the process to optimize, availability of historical data... make this choice easier.

Fuzzy programming If no historical data are available, it is difficult to represent uncertainty with probabilistic distribution. The modeller has only an a priori knowledge. In this case, fuzzy numbers are usually preferred. The shape of the fuzzy numbers, associated to some set of fuzzy rules, then reflects the knowledge of the decision maker.

We emphasize that if historical data are available, one should favour stochastic programming. This approach is indeed less sensitive to the personality of the modeller, and benefits from stronger analytical properties. Fuzzy programming, on its side, supposes that the decision maker has a good idea about how to represent uncertainty, while no formal data can establish the adequate form of the probabilistic distributions. Furthermore a nonlinear fuzzy program often amounts to a deterministic program, which is handled by classical methods (Sakawa [122], Fuller and Carlsson [56]). The question of interest is then how to incorporate the fuzziness. Stochastic programming, on its side, usually leads to specific procedures that take into account the probabilistic aspects. The choice between fuzzy and stochastic programming is thus more dependant on practical knowledge and intuition. In few words, one often prefers stochastic programming for well-known processes or situations where random variables adequately represent the uncertainty present in the problem, and fuzzy programming for unusual, one-shot processes, for which uncertainty cannot be easily quantified. Hybrid methods, introducing fuzziness in stochastic programs, have also been proposed (see for instance Mohans and Nguyen [99], Sinha, Suwarnan and Biswal [77] and in particular Masri [91]), however such approaches are beyond the concern of the present work.

1.3 Basic notions

We devote the remaining of this chapter to an introduction to some basic concepts that will serve as background material in the next chapters. Since we represent in this work uncertainty with random variables, we also review briefly in the next section measure and probability theories. Nonlinear mathematical programming and stochastic program formulations will be studied in the next two chapters.

1.3.1 Vectors and matrices

If x is a vector in $\mathbb{R}^n$, we denote its i-th component by $[x]_i$ ($1 \leq i \leq n$), or simply x_i if no confusion can arise from other suffixes attached to x. We denote by e the unit vector $(1, 1, \ldots, 1)^T$. Similarly, the (i, j)-th component of the matrix $A \in \mathbb{R}^{m \times n}$ will be written as $[a]_{i,j}$ or simply $a_{i,j}$. Note that matrices are traditionally written in upper case, while their components are in lower case. When describing iterative methods, we will use the notation $x_{[k]}$ to represent a

vector x at iteration k ($k \in \mathbb{N}$), or x_k when there is no risk of confusion with the k-th component of x. Finally we denote by $e^{[j]}$ the j-th coordinate vector in $\mathbb{R}^n$.

A symmetric matrix $A \in \mathbb{R}^{n\times n}$ is positive semidefinite if and only if

$$\langle x, Ax\rangle \geq 0 \text{ for all } x \in \mathbb{R}^n,$$

and it is positive definite if and only if the above inequality is a strict one, that is

$$\langle x, Ax\rangle > 0 \text{ for all } x \in \mathbb{R}^n,\ x \neq 0.$$

We denote the smallest and largest eigenvalues of the symmetric matrix A by $\lambda_{\min}[A]$ and $\lambda_{\max}[A]$ respectively. The inertia of a symmetric $n \times n$ matrix A is the triple

$$In(A) \stackrel{def}{=} (n_+, n_-, n_0),$$

where n_+, n_- and n_0 are respectively the number of positive, negative and zero eigenvalues of A.

Consider now the rectangular matrix $A \in \mathbb{R}^{m\times n}$. The null space of A is defined as

$$\mathcal{N}(A) \stackrel{def}{=} \{x \in \mathbb{R}^n \mid Ax = 0\}.$$

Assume that $m \leq n$ and A has full row rank, and let the columns of the n by $n - m$ matrix N be an orthonormal basis for the null space of A (so $AN = 0$ and $N^TN = I$). We say that a symmetric matrix $M \in \mathbb{R}^{n\times n}$ is second-order nonsingular (with respect to A) if the reduced matrix N^TMN is nonsingular, and second-order sufficient (with respect to A) if N^TMN is positive definite (see, for instance, Gould [66]).

1.3.2 Basic topology

We now present basic elements of topology that are useful for our needs. For a larger introduction to topology, we refer to the book of Sutherland [131]. Suppose that $\|\cdot\|$ is a norm on $\mathbb{R}^n$, and consider a subset $\mathcal{C}$ of $\mathbb{R}^n$. We define an open ball of radius ϵ about $x \in \mathcal{C}$ to be the set

$$\mathcal{O}_\epsilon(x) = \{y \mid \|y - x\| < \epsilon\}.$$

The set $\mathcal{C}$ is said to be open in $\mathbb{R}^n$ if for every x in $\mathcal{C}$, there is a scalar $\epsilon(x) > 0$ such that $\mathcal{O}_{\epsilon(x)}(x) \subseteq \mathcal{C}$. The intersection of a finite number of open sets, and the union of an arbitrary number of open sets, is open.

$\mathcal{C}$ is closed in $\mathbb{R}^n$ if $\mathbb{R}^n \setminus \mathcal{C}$ is open. The closed ball of radius ϵ about $x \in \mathcal{C}$ is

$$\mathcal{B}_\epsilon(x) = \{y \mid \|y - x\| \leq \epsilon\}.$$

The union of a finite number of closed sets, and the intersection of an arbitrary number of closed sets, is also closed.

$\mathcal{C}$ is bounded if there is a constant κ for which

$$\|x - y\| \leq \kappa$$

for x, $y \in \mathcal{C}$. Moreover, $\mathcal{C}$ is said to be compact if any arbitrary sequence $\{x_k\}$ in $\mathcal{C}$ has a convergent subsequence whose limit is in $\mathcal{C}$. In $\mathbb{R}^n$, $\mathcal{C}$ is compact if and only if $\mathcal{C}$ is both closed and bounded. Continuous functions reach their infima and suprema on compact sets.

As $\mathcal{C}$ may be neither open nor closed, we may be interested in the smallest closed set that contains $\mathcal{C}$ and the largest open set that is contained in $\mathcal{C}$. The closure of $\mathcal{C}$ is the set of points obtained by extending $\mathcal{C}$ to its boundary, or more formally,

$$\text{cl}\{\mathcal{C}\} \stackrel{def}{=} \{x \,|\, \forall\, \epsilon > 0,\ \mathcal{C} \cap \mathcal{B}_\epsilon(x) \neq \emptyset\}.$$

Similarly, the interior of $\mathcal{C}$ is the set of points whose neighbours are also in $\mathcal{C}$, or, more formally, the set

$$\{x \in \mathcal{C} \,|\, \exists\, \epsilon > 0 \text{ so that } \mathcal{O}_\epsilon(x) \subseteq \mathcal{C}\}.$$

The problem with this definition is that it takes no account of the fact that $\mathcal{C}$ may be of lower dimension than is appropriate for the norm used. A more useful concept is that of relative interior of $\mathcal{C}$. We first need to define the affine hull of a set. $\mathcal{A} \subseteq \mathbb{R}^n$ is an affine set if $x + \theta(y - x) \in \mathcal{C}$ for all x and $y \in \mathcal{C}$ and any scalar θ. The affine hull of a set $\mathcal{C}$, aff$\{\mathcal{C}\}$, is the smallest affine set containing $\mathcal{C}$, that is

$$\text{aff}\{\mathcal{C}\} \stackrel{def}{=} \{x \,|\, x \text{ is a linear combination of vectors in } \mathcal{C}\}.$$

The relative interior of $\mathcal{C}$ is the set of points whose neighbours in aff$\{\mathcal{C}\}$ are also in $\mathcal{C}$:

$$\text{ri}\{\mathcal{C}\} \stackrel{def}{=} \{x \in \mathcal{C} \,|\, \exists\, \epsilon > 0 \text{ so that if } y \in \mathcal{O}_\epsilon(x) \cap \text{aff}\{\mathcal{C}\},\ \text{then } y \in \mathcal{C}\}.$$

We then have that

$$\text{ri}\{\mathcal{C}\} \subseteq \mathcal{C} \subseteq \text{cl}\{\mathcal{C}\},$$

and the relative boundary of $\mathcal{C}$ is defined as

$$\partial\mathcal{C} \stackrel{def}{=} \text{cl}\{\mathcal{C}\} \setminus \text{ri}\{\mathcal{C}\}.$$

1.3.3 Convexity

A subset $\mathcal{S}$ of $\mathbb{R}^n$ is said to be convex if for any points x and y in $\mathcal{S}$ and $\lambda \in [0, 1]$, the point $x + \lambda(y - x)$ is also in $\mathcal{S}$. The point $x + \lambda(y - x)$ is said to be a convex combination of x and y. The intersection of convex sets is itself convex; however the union of convex sets is not necessarily convex.

If $\mathcal{U}$ is any subset of $\mathbb{R}^n$, another useful convex set is the convex hull of $\mathcal{U}$, co$\{\mathcal{U}\}$, which is defined to be the intersection of all convex sets containing $\mathcal{U}$. In other words, co$\{S\}$ is the set of all points that can be expressed as a convex combination of points in $\mathcal{U}$.

The concept of convexity is extended to real functions as follows.

Definition 1.2: Convex function
Given a convex subset $\mathcal{S}$ of $\mathbb{R}^n$, a function $f : \mathcal{S} \to \mathbb{R}$ is convex if

$$f(\lambda x + (1 - \lambda)y) \leq \lambda f(x) + (1 - \lambda) f(y)$$

for all $x \neq y \in \mathcal{S}$ and $\lambda \in [0, 1]$.

A function f will be said concave if $(-f)$ is convex. A convex function is continuous, as stated in the following proposition (Bertsekas [14]).

Proposition 1.1 *If $f : \mathbb{R}^n \to \mathbb{R}$ is convex, then it is continuous. More generally, if $\mathcal{C} \subseteq \mathbb{R}^n$ is convex and $f : \mathcal{C} \to \mathbb{R}$ is convex, then f is continuous over the relative interior of $\mathcal{C}$.*

A (constrained) mathematical program is convex if its objective function is convex, the inequality constraint functions g_i ($i \in \mathcal{I}$) are convex, and the equality constraint functions h_i ($i \in \mathcal{E}$) are affine. It follows that the feasible region of a convex program is a convex set. Convex programming is the part of mathematical programming studies and algorithms dedicated to convex programs.

Convexity is a strong property. A first step to generalize this class of functions, while keeping strong properties, is to use the notion of quasi-convexity. A quasi-convex function can be defined in terms of level sets, as follows.

Definition 1.3: Quasi-convex function
A function $f : \mathcal{S} \to \mathbb{R}$, where $\mathcal{S} \subset \mathbb{R}^n$ is a convex set, is quasi-convex if its lower level sets, $lev_{\leq\alpha} \stackrel{def}{=} \{x \in \mathcal{S} : f(x) \leq \alpha\}$ are convex sets.

Similarly, we can define quasi-concave functions.

Definition 1.4: Quasi-concave function
A function $f : \mathcal{S} \to \mathbb{R}$, where $\mathcal{S} \subset \mathbb{R}^n$ is a convex set, is quasi-concave if its upper level sets, $lev_{\geq\alpha} \stackrel{def}{=} \{x \in \mathcal{S} : f(x) \geq \alpha\}$ are convex sets.

The class of quasi-convex functions contains all convex functions and all monotone functions of a single variable. It is sometimes cumbersome to use the previous definitions. In such cases, we can turn to the following characterizations.

Proposition 1.2 *A function $f : \mathcal{S} \to \mathbb{R}$, where $\mathcal{S} \subset \mathbb{R}^n$ is a convex set, is quasi-convex if and only if $\forall x, y \in \mathcal{S}$ and $\forall \lambda \in [0, 1]$:*

$$f(\lambda x + (1 - \lambda)y) \leq \max\{f(x), f(y)\}.$$

Proposition 1.3 *A function $f : \mathcal{S} \to \mathbb{R}$, where $\mathcal{S} \subset \mathbb{R}^n$ is a convex set, is quasi-concave if and only if $\forall x, y \in \mathcal{S}$ and $\forall \lambda \in [0, 1]$:*

$$f(\lambda x + (1 - \lambda)y) \geq \min\{f(x), f(y)\}.$$

These ideas have been extended in the measure theory area (see Section 1.4). In particular, a probability measure P is said to be quasi-concave if, for any measurable sets U and V, and any

$0 \leq \lambda \leq 1$,

$$P\left[\lambda U + (1-\lambda)V\right] \geq \min\{P[U], P[V]\}.$$

1.3.4 Derivatives

First and second derivatives

We start with a univariate function $f : \mathbb{R} \to \mathbb{R}$. The derivative of ϕ at $x \in \mathbb{R}$ is defined by

$$f'(x) \stackrel{def}{=} \frac{df(x)}{dx} = \lim_{\epsilon \to 0} \frac{f(x+\epsilon) - f(x)}{\epsilon},$$

if the limit exists. If x depends on y, we can compute the derivative of f with respect to y:

$$\frac{df(x(y))}{dy} = \frac{df}{dx}\frac{dx}{dy}.$$

This computation is known as the chain rule.

A function $f : \mathbb{R}^n \to \mathbb{R}$ is said to be differentiable at $x \in \mathbb{R}^n$ if all its first partial derivatives

$$\frac{\partial f(x)}{\partial x_i} \stackrel{def}{=} \lim_{\epsilon \to 0} \frac{f(x + \epsilon e^{[i]}) - f(x)}{\epsilon},\ i = 1, \ldots, n,$$

exist. When this is the case, partial derivatives can be merged in a n-vector called the gradient of f and denoted by

$$\nabla f(x) \stackrel{def}{=} \begin{pmatrix} \frac{\partial f(x)}{\partial x_1} \\ \vdots \\ \frac{\partial f(x)}{\partial x_n} \end{pmatrix}.$$

If f is differentiable at x, f is continuous at x. If moreover its derivatives are continuous at x, f is said to be continuously differentiable at x. More generally we say that f is in C^k if it has continuous k-th-order partial derivatives.

If f is twice continuously differentiable ($f \in C^2$), we define the Hessian matrix (or Hessian for short) of f to be the $n \times n$ matrix-valued function

$$\nabla^2 f(x) \stackrel{def}{=} \begin{pmatrix} \frac{\partial^2 f(x)}{\partial x_1^2} & \frac{\partial^2 f(x)}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f(x)}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f(x)}{\partial x_2 \partial x_1} & \frac{\partial^2 f(x)}{\partial x_2^2} & \cdots & \frac{\partial^2 f(x)}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f(x)}{\partial x_n \partial x_1} & \frac{\partial^2 f(x)}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f(x)}{\partial x_n^2} \end{pmatrix}.$$

The curvature of f at $x \in \mathbb{R}^n$ along a direction $d \in \mathbb{R}^n$ is given by

$$\frac{\langle d, \nabla f(x) d \rangle}{\|d\|^2}.$$

A function $g : \mathbb{R}^n \to \mathbb{R}^m$ is differentiable at $x \in \mathbb{R}^n$ if all partial derivatives

$$\frac{\partial g_j(x)}{\partial x_i} \stackrel{def}{=} \lim_{\epsilon \to 0} \frac{g_j(x + \epsilon e^{[i]}) - g_j(x)}{\epsilon}, \; i = 1, \ldots, n, \; j = 1, \ldots, n.$$

exist. If this is the case, they can be gathered in the m-by-n matrix

$$Jg(x) = \begin{pmatrix} \frac{\partial g_1(x)}{\partial x_1} & \frac{\partial g_1(x)}{\partial x_2} & \cdots & \frac{\partial g_1(x)}{\partial x_n} \\ \frac{\partial g_2(x)}{\partial x_1} & \frac{\partial g_2(x)}{\partial x_2} & \cdots & \frac{\partial g_2(x)}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial g_m(x)}{\partial x_1} & \frac{\partial g_m(x)}{\partial x_2} & \cdots & \frac{\partial g_m(x)}{\partial x_n} \end{pmatrix} = \begin{pmatrix} \nabla g_1^T(x) \\ \nabla g_2^T(x) \\ \vdots \\ \nabla g_m^T(x) \end{pmatrix},$$

called the Jacobian matrix (or Jacobian for short) of g. Note that the second equality shows that the Hessian of $g(x)$ is the Jacobian of $\nabla g(x)$.

Note that when a function depends on several vectors, for instance if we consider a function

$$\begin{aligned} h : \mathbb{R}^n \times \mathbb{R}^q &\to \mathbb{R} \\ (x, y) &\rightsquigarrow h(x, y), \end{aligned}$$

we use a different notation to denote the derivatives of h with respect to the variables x and/or y: $\nabla_x h(x, y) \in \mathbb{R}^n$ and $\nabla^2_{xx} h(x, y) \in \mathbb{R}^{n \times n}$ represent the gradient and the Hessian of h with respect to x, $\nabla_y h(x, y) \in \mathbb{R}^q$ and $\nabla^2_{yy} h(x, y) \in \mathbb{R}^{q \times q}$ denote the gradient and the Hessian of h with respect to y, while $\nabla h(x, y) \in \mathbb{R}^{n+q}$ and $\nabla^2 h(x, y) \in \mathbb{R}^{(n+q)\times(n+q)}$ denote the "complete" first- and second-order derivatives of h, i.e. with respect to both x and y.

Directional derivatives and subgradients

Given a function $f : \mathbb{R}^n \to \mathbb{R}$, a point $x \in \mathbb{R}^n$ and a vector $d \in \mathbb{R}^n$, f is said to be directionally differentiable at $x \in \mathbb{R}$ in the direction d if the limit

$$f'_d(x) \stackrel{def}{=} \lim_{t \searrow 0} \frac{f(x + td) - f(x)}{t}$$

exists. $f'_d(x)$ is the one-side directional derivative of f at x in the direction d. If f is continuously differentiable in some neighbourhood of x,

$$f'_d(x) = \nabla f(x)^T d.$$

The subdifferential $\partial_x f(x)$ of f at x is the set

$$\partial_x f(x) \stackrel{def}{=} \{g \in \mathbb{R}^n \,|\, \langle g, d \rangle \leq f'_d(x) \; \forall d \in \mathbb{R}^n\},$$

while each member of the subdifferential is known as a subgradient. We will denote by f_x any arbitrary subgradient of f in x. If f is convex and $f(x)$ is finite, this derivative always exists and $f'_{-d}(x) \leq f'_d(x)$. The subdifferential can then be rewritten (see for example Rockafellar an Wets [116]) as

$$\partial_x f(x) = \{g \in \mathbb{R}^n \,|\, f(x) + \langle g, d \rangle \leq f(x + d), \; \forall d \in \mathbb{R}^n\}.$$

Moreover, the subdifferential is not empty.

1.3.5 Forcing functions

In our later developments (see in particular Chapter 4), we will also need the concept of forcing function.

Definition 1.5: Forcing function
$\phi(\cdot)$ is a forcing function if it is a continuous function from the set of nonnegative reals $\mathbb{R}_+$ into itself with the property that

$$\phi(x) = 0 \text{ if and only if } x = 0.$$

Examples of forcing functions of x are x itself and x^2. The products or the sums of sets of forcing functions are forcing functions, as are their minimum or maximum.

1.4 Measure and probability theories

Probability theory founds its foundations in measure theory, so it is useful to first review measure theory results that will be relevant for our analysis. The reader can refer amongst others to Ash and Doleans-Dade [5] and Davidson [39] for a more detailed coverage of these subjects.

1.4.1 Measure spaces

First, define the type of objects that could be measured. This is done with the concept of σ-algebras.

Definition 1.6: σ-algebra
Let X be a set. A σ-algebra (also called a σ-field) over X is a collection $\mathcal{X}$ of subsets of X with the following properties:

1. *$\mathcal{X}$ is closed under countable unions. In other terms, if $U_1, U_2, \ldots$ are in $\mathcal{X}$, then their union*
$$\bigcup_{n=1}^{\infty} U_n$$
is also in $\mathcal{X}$.
2. *$\mathcal{X}$ is closed under countable intersections. If $U_1, U_2, \ldots$ are in $\mathcal{X}$, then their intersection*
$$\bigcap_{n=1}^{\infty} U_n$$
is also in $\mathcal{X}$.
3. *$\mathcal{X}$ is closed under complementation: if U is in $\mathcal{X}$, then the complementary set of U, denoted by*
$$U^C \stackrel{def}{=} X \setminus U,$$

is also in $\mathcal{X}$.

There are two trivial σ-algebras: for any set X, the collection $\{\emptyset, X\}$ and the power set $\mathcal{P}(X) \stackrel{def}{=} \{S \subset X\}$. The smallest σ-algebra which contains all elements of some collection $\mathcal{M}$ is called the σ-algebra generated by $\mathcal{M}$ and is sometimes denoted by $\sigma(\mathcal{M})$. If $\mathcal{X}$ and $\mathcal{Y}$ are both σ-fields, and $\mathcal{Y} \subset \mathcal{X}$, then $\mathcal{Y}$ is said to be a sub-field (or a sub-algebra) of $\mathcal{X}$, and $\mathcal{X}$ is said to contain more information or refine $\mathcal{Y}$. If $X = \mathbb{R}$ we will often use the Borel field $\mathcal{B}_{\mathbb{R}}$ (or $\mathcal{B}$ for short), the σ-algebra generated by the collection of closed half-lines with rational endpoint $\mathcal{C} = \{(-\infty, r],\ r \in \mathbb{Q}\}$. A number of different base collections generate $\mathcal{B}_{\mathbb{R}}$, including the collection of closed half-lines with real endpoints $\{(-\infty, x],\ x \in \mathbb{Q}\}$, the open intervals of $\mathbb{R}$, the closed intervals, and the half-open intervals.

Adding a σ-algebra to a set X leads to the notion of measurable space.

Definition 1.7: Measurable space
A measurable space is an ordered pair $(X, \mathcal{X})$, where X is a set, and $\mathcal{X}$ is a σ-algebra on X. The sets in $\mathcal{X}$ are called measurable sets.

Having a measurable space $(X, \mathcal{X})$, we would like to be able to quantify in some manner how large is an element of $\mathcal{X}$. This is achieved by introducing a measure on $\mathcal{X}$.

Definition 1.8: Measure
Let X be a set, and $\mathcal{X}$ a σ-algebra on X. A measure on $\mathcal{X}$ is a map

$$\mu : \mathcal{X} \to [0, \infty]$$

which is countable addititive, in the sense that, if $Y_1, Y_2, Y_3, \ldots$ are all elements of $\mathcal{X}$, and are disjoint, then

$$\mu\left(\bigcup_{n=1}^{\infty} Y_n\right) = \sum_{n=1}^{\infty} \mu\left(Y_n\right).$$

The counting measure is the simplest measure of all; it assigns to any set the cardinality of that set:

$$\mu(S) = \#S.$$

If $\mu(X) < \infty$, the measure is said to be finite, and if $X = \bigcup_j X_j$, where $\{X_j\}$ is a countable collection of $\mathcal{X}$-sets, and $\mu(X_j) < \infty$ for each j, μ is said to be σ-finite.

The next definition that we introduce synthesizes the previous ones.

Definition 1.9: Measure space
A measure space is an ordered triple $(X, \mathcal{X}, \mu)$, where X is a set, $\mathcal{X}$ is a σ-algebra and μ is a measure on $\mathcal{X}$.

1.4.2 Product spaces and product measures

If $(X_1, \mathcal{X}_1)$ and $(X_2, \mathcal{X}_2)$ are two measurable spaces, let

$$X_1 \times X_2 = \{(x_1, x_2) \,|\, x_1 \in X_1,\ x_2 \in X_2\}$$

be the Cartesian product of X_1 and X_2, and define $\mathcal{X}_1 \otimes \mathcal{X}_2 = \sigma(\mathcal{R}_{\mathcal{X}_1\mathcal{X}_2})$, where

$$\mathcal{R}_{\mathcal{X}_1\mathcal{X}_2} = \{(Y_1 \times Y_2),\ Y_1 \in \mathcal{X}_1,\ Y_2 \in \mathcal{X}_2\}.$$

The space $(X_1 \times X_2, \mathcal{X}_1 \otimes \mathcal{X}_2)$ is called a product space, and $(X_1, \mathcal{X}_1)$ and $(X_2, \mathcal{X}_2)$ are the factor spaces, or coordinate spaces, of the product. The elements of the collection $\mathcal{R}_{\mathcal{X}_1\mathcal{X}_2}$ are called the measurable rectangles.

Let μ be a measure on $\mathcal{X}_1$, and ν, a measure on $\mathcal{X}_2$. Therefore $(X_1, \mathcal{X}_1, \mu)$ and $(X_2, \mathcal{X}_2, \nu)$ are two measure spaces. Define the set function

$$\pi : \mathcal{R}_{\mathcal{X}_1\mathcal{X}_2} \mapsto \overline{\mathbb{R}}_+,$$

where $\mathcal{R}_{\mathcal{X}_1\mathcal{X}_2}$ denotes the measurable rectangles of the space $X_1 \times X_2$, by

$$\pi(X_1 \times X_2) = \mu(X_1)\mu(X_2).$$

π is a measure on $\mathcal{R}_{\mathcal{X}_1\mathcal{X}_2}$, called the product measure, and $(X_1 \times X_2, \mathcal{X}_1 \times \mathcal{X}_2, \pi)$ is a measure space. μ and ν are called the marginal measures corresponding to π.

1.4.3 Functions on measure spaces

It is often convenient to be able to make some correspondence between measurable sets of two spaces. This is accomplished by means of measurable functions.

Definition 1.10: Measurable function
Let $(X_1, \mathcal{X}_1)$ and $(X_2, \mathcal{X}_2)$ be measurable spaces. A function

$$f : X_1 \to X_2$$

is called measurable with respect to $\mathcal{X}_1$ and $\mathcal{X}_2$, or $\mathcal{X}_1/\mathcal{X}_2$-measurable, if every $\mathcal{X}_2$-measurable set has an $\mathcal{X}_1$-measurable preimage under f. Formally:

$$\forall A \subset X_2,\ \textit{if } A \in \mathcal{X}_2,\ \textit{then } f^{-1}(A) \in \mathcal{X}_1.$$

When the relevant σ-algebras are understood, we will often simply say that f is measurable, without explicit reference to $\mathcal{X}_1$ and $\mathcal{X}_2$.

This definition depends only on the σ-algebras $\mathcal{X}_1$ and $\mathcal{X}_2$ and has nothing to do with any actual measure on $\mathcal{X}_1$ or $\mathcal{X}_2$.

Consider now measure spaces; it is usually sufficient to consider structures or properties almost everywhere, i.e., holding for every element of $\mathcal{X}$, except possibly for sets of measure zero. This leads to the definition of a function mod zero.

Definition 1.11: Function mod zero
Let $(X, \mathcal{X}, \mu)$ be a measure space, and let $\hat{X}$ be some other set. A function mod zero is a function f defined almost everywhere on X. In other words, there is some subset $X_0 \in \mathcal{X}$, with $\mu(X \setminus X_0) = 0$, such that

$$f : X_0 \to \hat{X}.$$

The integral of a measurable function $f : X \to \overline{\mathbb{R}}$ on a measure space $(X, \mathcal{B}, \mu)$ is written

$$\int_X f d\mu, \text{ or just } \int f d\mu.$$

It is defined via the following steps:

- If $f = \mathbf{1}_A$ is the characteristic function of a set $A \in \mathcal{B}$, i.e.,

$$\mathbf{1}_A(x) = \begin{cases} 1 & \text{if } x \in A, \\ 0 & \text{if } x \in X \setminus A, \end{cases}$$

  then set

$$\int_X \mathbf{1}_A d\mu \stackrel{def}{=} \mu(A).$$

- If f is a simple function, i.e., if f can be written as

$$f = \sum_{k=1}^{n} c_k \mathbf{1}_{A_k},$$

  where $c_k \in \mathbb{R}$ and $A_k \in \mathcal{B}$, $k = 1, \ldots, n$, then

$$\int_X f d\mu \stackrel{def}{=} \sum_{k=1}^{n} c_k \int_X \mathbf{1}_{A_k} d\mu = \sum_{k=1}^{n} c_k \mu(A_k).$$

- If f is a nonnegative measurable function, then

$$\int_X f d\mu \stackrel{def}{=} \sup \left\{ \int_X h d\mu \,|\, h \text{ is simple and } h(x) \leq f(x) \text{ for all } x \in X \right\}.$$

- If $f : X \to \overline{\mathbb{R}}$ is any measurable function, let $f^+ = \max\{f, 0\} \geq 0$ and $f^- = \max\{-f, 0\} \geq 0$. The integral of f is defined as

$$\int f d\mu = \int f^+ d\mu - \int f^- d\mu$$

  so long as at least one of the right-hand side integrals are finite. If both $\int f^+ d\mu = +\infty$ and $\int f^- d\mu = +\infty$, the integral is undefined. A function is said to be integrable only if its integral is both defined and finite. Noting that $|f| = f^+ + f^-$, f is integrable if and only if

$$\int |f| d\mu < \infty.$$

If μ is the Lebesgue measure and X is any interval in $\mathbb{R}^n$ then the integral is called the Lebesgue integral. If the Lebesgue integral of a function f on a set A exists, f is said to be Lebesgue integrable.

Let p be a strictly positive integer, and $(X, \mathcal{X}, \mu)$ a measure space. $L_p(X, \mathcal{X}, \mu)$ is the collection of real-valued functions defined on X such that $|f|^p$ is measurable. We can then define the L_p-norm of the function f as

$$\|f\|_p = \sqrt{\frac{1}{p}} \int |f|^p d\mu.$$

We conclude this section with the Fubini's theorem.

Theorem 1.1 (Fubini's theorem) *Let π be a product measure with σ-finite marginal measures μ and ν; let $f : X \times Y \to \mathbb{R}$ be $(\mathcal{X} \otimes \mathcal{Y})/\mathcal{B}$-measurable with*

$$\int_{X \times Y} |f(x, y)| \, d\pi(x, y) < \infty.$$

Define $f_x : Y \to \mathbb{R}$ by $f_x(y) = f(x, y)$ and let $g(y) = \int_Y f_x d\nu$. Then

(i) f_x is $\mathcal{Y}/\mathcal{B}$-measurable and integrable for $x \in A \subseteq X$, with $\mu(X \setminus A) = 0$;

(ii) g is $\mathcal{X}/\mathcal{B}$-measurable, and integrable on A;

(iii) $\int_{X \times Y} f(x, y) d\pi(x, y) = \int_X \left(\int_Y f(x, y) d\nu(y) \right) d\mu(x)$.

1.4.4 Conditional expectations

Suppose $(X, \mathcal{X}, \mu)$ is some measure space, and $f : X \to C \subset \mathbb{R}$ is $\mathcal{X}$-measurable. If $\mathcal{Y} \subset \mathcal{X}$ is some σ-subalgebra, the function f will not, in general, also be measurable with respect to $\mathcal{Y}$. We can however approximate f by a $\mathcal{Y}$-measurable function, called the conditional expectation of f.

Definition 1.12: Conditional expectation
Let $(X, \mathcal{X}, \mu)$ be a measure space, and let $\mathcal{Y}$ be a σ-subalgebra of $\mathcal{X}$. Let $f \in L_1(X, \mathcal{X}, \mu)$. The conditional expectation of f with respect to $\mathcal{Y}$ is the unique function, written $E_{\mathcal{Y}}[f]$, satisfying:

1. *$E_{\mathcal{Y}}[f]$ is $\mathcal{Y}$-measurable,*
2. *For any subset $U \in \mathcal{Y}$, we have:*

$$\int_U E_{\mathcal{Y}} d\mu = \int_U f d\mu.$$

If f is $\mathcal{Y}$-measurable, we can expect that the conditional expectation of f with respect to Y corresponds in fact to f. This is obtained from the following proposition.

Proposition 1.4 *Let $(X, \mathcal{X}, \mu)$ be a measure space, $f \in L_1(X, \mathcal{X}, \mu)$ and $\mathcal{Y} \subset \mathcal{X}$ be a σ-*

subalgebra. If f is $\mathcal{Y}$-measurable, then

$$E_{\mathcal{Y}}[f] = f.$$

If $\mathcal{Y}_1 \subset \mathcal{Y}_2$, then

$$E_{\mathcal{Y}_1}\left[E_{\mathcal{Y}_2}[f]\right] = E_{\mathcal{Y}_1}[f].$$

1.4.5 Probability spaces and random variables

In stochastic programming, uncertainty is represented in terms of random experiments with outcomes, called random elements, individually denoted by ω. The set of all outcomes is Ω, the sample space. An event A is a subset of Ω, and we represent the collection of random events by $\mathcal{A}$. More formally, $\mathcal{A}$ is a σ-field of subsets of Ω. The event $A \in \mathcal{A}$ is said to have occurred if the outcome of the experiment is an element of A. Finally, for each $A \in \mathcal{A}$, we associate a probability measure $P[A]$ defined as below.

Definition 1.13: Probability measure
A probability measure (p.m.) on a measurable space $(\Omega, \mathcal{A})$ is a set function $P : \mathcal{A} \to [0, 1]$ satisfying the axioms of probability:

(a) $P(A) \geq 0$, *for all* $A \in \mathcal{A}$,

(b) $P(\Omega) = 1$,

(c) countable additivity: for a disjoint collection $\{A_j \in \mathcal{A},\ j \in \mathbb{N}\}$,

$$P\left(\bigcup_j A_j\right) = \sum_j P(A_j).$$

Additional properties of P follow from the axioms.

Proposition 1.5 *If A, B, and $\{A_j,\ j \in \mathbb{N}\}$ are arbitrary $\mathcal{A}$-sets, then*

(i) $P[A] \leq 1$,

(ii) $P[A^c] = 1 - P[A]$,

(iii) $P[\emptyset] = 0$,

(iv) monotonicity: if $A \subseteq B$, then, $P[A] \leq P[B]$,

(v) $P[A \cup B] = P[A] + P[B] - P[A \cap B]$,

(vi) countable subadditivity: $P[\cup_j A_j] \leq \sum_j P[A_j]$,

(vii) continuity: if $A_j \nearrow A$ or $A_j \searrow A$, then $P[A_j] \to P[A]$.

If $A \in \mathcal{A}$ has $P[A] = 1$, then A is said to occur almost surely (a.s.), or with probability

one (w.p. 1). If $A \in \mathcal{A}$ has $P(A) = 0$, then A is said to occur with probability 0 (w.p. 0). An event is a measurable subset of Ω. The triplet $(\Omega, \mathcal{A}, P)$ is called a probability space. It is easy to see that any probability space $(\Omega, \mathcal{A}, P)$ is a measure space, so it is also called a measure probability space. An important situation in programming stochastic is when the elements $\omega \in \Omega$ are used to describe a few states of the world or scenarios (see Section 3.2.4 on page 52). All random elements then jointly depend on these finitely many scenarios; it's a frequent situation in strategic modelling where only a few scenarios are considered in detail. In many cases however, the construction of Ω and $\mathcal{A}$ is extremely complicated, and the knowledge of the random variables is sufficient.

A random variable $\boldsymbol{\xi}$ on a measurable space $(\Omega, \mathcal{A})$, taking its values in a measurable space $(\Omega', \mathcal{A}')$, is a measurable function $\boldsymbol{\xi} : \Omega \to \Omega'$. We will only consider real discrete or continuous random variables. For discrete variables, Ω' is a discrete subset of $\mathbb{R}$, associated to the power set. For continuous variables, Ω' is equal to $\mathbb{R}$ or $\overline{\mathbb{R}} = \mathbb{R} \cup \{+\infty\} \cup \{-\infty\}$, and is associated to the Borel field $\mathcal{B}_{\mathbb{R}}$ or $\mathcal{B}_{\overline{\mathbb{R}}}$. Define a probability measure P on $(\Omega, \mathcal{A})$. We introduce the following notation:

$$P_{\boldsymbol{\xi}}[A'] \stackrel{def}{=} P\left[\boldsymbol{\xi}^{-1}(A')\right].$$

It is easy to check that the function $P_{\boldsymbol{\xi}}$, defined on $(\Omega', \mathcal{A}')$, is a probability measure.

If the random variable is discrete we can also assume that $(\Omega', \mathcal{A}') = (\mathbb{R}, \mathcal{B}_{\mathbb{R}})$. Consider a subset $\mathcal{A}'$ measurable in $(\mathbb{R}, \mathcal{B}_{\mathbb{R}})$; since $\{a_i | a_i \in \mathcal{A}' \cap \Omega'\}$ is measurable, we can define $P_{\boldsymbol{\xi}}[A'] = P_{\boldsymbol{\xi}}[\{a_i | a_i \in \mathcal{A}' \cap \Omega'\}]$. Moreover if the random variable is continuous and $(\Omega', \mathcal{A}') = (\overline{\mathbb{R}}, \mathcal{B}_{\mathbb{R}})$, we could have $P[\boldsymbol{\xi} \in \{-\infty, +\infty\}] > 0$. Assigning a positive probability to infinity does not however lead to meaningful results (Davidson [39], page 117). Random variables must be finite with probability one, so $(\mathbb{R}, \mathcal{B}, P_{\boldsymbol{\xi}})$, the trace of $(\overline{\mathbb{R}}, \mathcal{B}_{\overline{\mathbb{R}}}, P_{\boldsymbol{\xi}})$ on $\mathbb{R}$, is equivalent to it for nearly all purposes.

The cumulative distribution function (c.d.f.) of $\boldsymbol{\xi}$ is $F_{\boldsymbol{\xi}} : \overline{\mathbb{R}} \to [0, 1]$, where

$$F_{\boldsymbol{\xi}}(x) = P_{\boldsymbol{\xi}}[\boldsymbol{\xi} \leq x],\ x \in \overline{\mathbb{R}}.$$

We take the domain to be $\overline{\mathbb{R}}$ since it is natural to assign the values 0 and 1 to $F_{\boldsymbol{\xi}}(-\infty)$ and $F_{\boldsymbol{\xi}}(+\infty)$ respectively. If $\boldsymbol{\xi}$ is continuous, $F_{\boldsymbol{\xi}}$ is absolutely continuous and the derivative $f = dF_{\boldsymbol{\xi}}/dx$ exists almost everywhere (with respect to the Lebesgue measure), and is called the probability density function (p.d.f.). According to the Radon-Nikodym theorem, the p.d.f. has the property that for each $E \in \mathcal{B}$,

$$\mu(E) = \int_E f(x)dx.$$

When no confusion is possible, we will write $F(x)$ instead of $F_{\boldsymbol{\xi}}(x)$.

Example 1.1. *For the uniform distribution (also called rectangular distribution) on* $[0, 1]$,

$$F(x) = \begin{cases} 0 & \textit{if } x < 0, \\ x & \textit{if } 0 \leq x \leq 1, \\ 1 & \textit{if } x > 1. \end{cases}$$

The p.d.f. is constant at 1 on the interval, but is undefined at 0 and 1. If $\boldsymbol{\xi}$ *follows an uniform distribution on* $[0, 1]$*, we will write* $\boldsymbol{\xi} \sim U(0, 1)$.

Example 1.2. *The p.d.f. of a normally distributed random variable, with mean μ and standard deviation σ, is*

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}.$$

If $\boldsymbol{\xi}$ follows a normal distribution with mean μ and variance σ^2, we will write $\boldsymbol{\xi} \sim N(\mu, \sigma^2)$.

Example 1.3. *A popular distribution in discrete choice models (see Chapter 6) is the Gumbel distribution, also called the extreme value type I distribution. Its c.d.f. is given by*

$$F(x) = e^{-e^{-\mu(\epsilon-\eta)}},$$

where η is a location parameter and $\mu > 0$ is a scale parameter. The p.d.f. is

$$f(x) = \mu e^{-\mu(\epsilon-\eta)} e^{-e^{-\mu(\epsilon-\eta)}}.$$

The mean of the Gumbel distribution is

$$\eta - \frac{\Gamma'(1)}{\mu},$$

where $\Gamma'(1) \approx -0.57721$ is the first derivative of the gamma function $\Gamma(n)$ with respect to n at $n = 1$. Note that $\gamma \stackrel{def}{=} -\Gamma'(1)$ is also known as the Euler constant. The variance of a Gumbel distributed variable is

$$\frac{\mu^2\pi^2}{6}.$$

The reader interested in properties of major statistical distributions can refer to Evans, Hastings and Peacock [49].

Let μ be a measure on $(\Omega', \mathcal{A}')$. $\boldsymbol{\xi}$ has a probability density if there is a function f such that for all $A' \in \mathcal{A}'$,

$$P_{\boldsymbol{\xi}}(A') = \int_{A'} f(x) d\mu.$$

If $(\Omega, \mathcal{A}') = (\mathbb{R}, \mathcal{B}_{\mathbb{R}})$, μ is the Lebesgue measure and $d\mu$ is then usually replaced by the notation dx. Else, if Ω is a discrete set X, and $\mathcal{A}'$ is the power set, μ is the counting measure.

1.4.6 Multivariate variables

In Euclidian n-space $\mathbb{R}^n$, the n-dimensional Borel field $\mathcal{B}^n$ is $\sigma(\mathcal{R}^n)$, where $\mathcal{R}^n$ denotes the measurable rectangles of $\mathbb{R}^n$, the sets of the form $B_1 \times B_2 \times \ldots \times B_n$ where $B_i \in \mathcal{B}$ for $i = 1, \ldots, n$. In a space $(\Omega, \mathcal{A}, P)$, a random vector $\boldsymbol{\xi} = (\boldsymbol{\xi}_1, \boldsymbol{\xi}_2, \ldots, \boldsymbol{\xi}_n)^T$ is a measurable mapping

$$\boldsymbol{\xi} : \Omega \to \mathbb{R}^n.$$

If μ is the derived measure defined by $\mu(E) = P[A]$ for $E \in \mathcal{B}^n$ and $A \in \mathcal{A}$, such that $A = \xi^{-1}(E)$, the multivariate c.d.f., $F_{\boldsymbol{\xi}} : \overline{\mathbb{R}}^n \to [0, 1]$, is defined for $x = (x_1, \ldots, x_n)^T$ by

$$F_{\boldsymbol{\xi}}(x) = \mu((-\infty, x_1] \times \ldots \times (-\infty, x_n]).$$

If the distribution is continuous with p.d.f. $f(x)$, Fubini's theorem gives

$$F(x) = \int_{(-\infty,x_1]\times\ldots\times(-\infty,x_n]} f(x_1,\ldots,x_n)dx_1\ldots dx_n = \int_{-\infty}^{x_n}\ldots\int_{-\infty}^{x_1} f(x_1,\ldots,x_n)dx_1\ldots dx_n.$$

Example 1.4 (Multivariate normal distribution). *Let $\boldsymbol{\zeta} = (\boldsymbol{\zeta}_1,\ldots,\boldsymbol{\zeta}_n)^T$ be a random vector with p.d.f.*

$$f(\boldsymbol{\zeta}) = \frac{1}{\sqrt{(2\pi)^n|\Sigma|}} e^{-\frac{1}{2}(\boldsymbol{\zeta}-\mu)^T\Sigma^{-1}(\boldsymbol{\zeta}-\mu)},$$

where $\boldsymbol{\zeta} = (\boldsymbol{\zeta}_1,\ldots,\boldsymbol{\zeta}_n)$, μ denotes a vector of means $\mu_1,\ldots,\mu_n$, and Σ is a $n\times n$ symmetric positive define matrix of covariance. The notation $|\cdot|$ designs the determinant operator. $\boldsymbol{\zeta}$ is then said to follow a multivariate normal distribution, and we write $\boldsymbol{\xi} \sim \mathcal{N}(\mu,\Sigma)$. If $\boldsymbol{\xi} \sim \mathcal{N}(\mathbf{0}, I)$, where $\mathbf{0}$ is a column vector of n zeros and I is the $n\times n$ identity matrix, then we can represent $\boldsymbol{\zeta}$ as

$$\boldsymbol{\zeta} = C\boldsymbol{\xi} + \mu,$$

where C is the (unique) Cholesky factor of Σ.

We also define the support of $\boldsymbol{\xi}$ as follows:

Definition 1.14: Support of random variable
Let $\boldsymbol{\xi}$ a random variable. $\Xi \in \mathbb{R}^n$ is the support of $\boldsymbol{\xi}$ if it is the smallest closed subset in $\mathbb{R}^n$ such that $P_{\boldsymbol{\xi}}[\boldsymbol{\xi} \in \Xi] = 1$.

We can associate to the random vector $\boldsymbol{\xi}$ the probability space $(\Xi, \mathcal{F}, P_{\boldsymbol{\xi}})$, with $\mathcal{F} = \{A \,|\, A = B \cap \Xi,\ B \in \mathcal{B}^n\}$.

1.4.7 Independence

A pair of events A, $B \in \mathcal{A}$ is said to be independent if

$$P[A \cap B] = P[A]P[B], \tag{1.8}$$

or, equivalently, if

$$P[B|A] = P[B].$$

If, in a collection of events $\mathcal{C}$, (1.8) holds for every pair of distinct sets A and B from the collection, $\mathcal{C}$ is said to be pairwise independent. In addition, $\mathcal{C}$ is said to be totally independent if

$$P\left[\cap_{A\in\mathcal{J}} A\right] = \prod_{A\in\mathcal{J}} P[A].$$

for every subset $\mathcal{J} \subseteq \mathcal{C}$ containing two or more events.

The extension to random variables is direct. Consider two random variables $\boldsymbol{X}$ and $\boldsymbol{Y}$ on $(\mathbb{R}^2, \mathcal{B}^2, \mu)$. If we are interested only in predicting X, the events of interest are the cylinder sets

in $\mathbb{R}^2$, having the form $B \times \mathbb{R}$, $B \in \mathcal{B}$. The marginal distribution of X is defined by $(\mathbb{R}, \mathcal{B}, \mu_X)$ where

$$\mu_X(A) = \mu(A \times \mathbb{R}) \tag{1.9}$$

for $A \in \mathcal{B}$. $\boldsymbol{X}$ and $\boldsymbol{Y}$ are called independent if and only if

$$\mu(A \times B) = \mu_{\boldsymbol{X}}(A)\mu_{\boldsymbol{Y}}(B)$$

for all pairs of events A, $B \in \mathcal{B}$, where $\mu_{\boldsymbol{X}}$ is defined by (1.9) and $\mu_{\boldsymbol{Y}}$ is defined analogously. Equivalently, μ is the product measure generated by $\mu_{\boldsymbol{X}}$ and $\mu_{\boldsymbol{Y}}$.

Theorem 1.2 *$\boldsymbol{X}$ and $\boldsymbol{Y}$ are independent if and only if for each $x, y \in \mathbb{R}$*

$$F(x, y) = F_{\boldsymbol{X}}(x)F_{\boldsymbol{Y}}(y).$$

If the distribution is continuous the p.d.f. factorizes as

$$f(x, y) = f_{\boldsymbol{X}}(x)f_{\boldsymbol{Y}}(y).$$

Variables $\boldsymbol{\xi}_1, \ldots, \boldsymbol{\xi}_n$ distributed on the space $(\mathbb{R}^n, \mathcal{B}^n, \mu)$ are said to be totally independent if

$$\mu\left(\bigtimes_{i=1}^{n} A_i\right) = \prod_{i=1}^{n} \mu_{X_i}(A_i)$$

for all n-uples of events $A_1, \ldots, A_n \in \mathcal{B}$. Another way to define total independence is in terms of a partitioning of a vector $\boldsymbol{\xi} = (\boldsymbol{\xi}_1, \ldots, \boldsymbol{\xi}_n)^T$ into subvectors $\boldsymbol{\xi}_1$ of dimension j and $\boldsymbol{\xi}_2$ of dimension $n - j$, for j satisfying $0 < j < n$. Under total independence, the measure of $\boldsymbol{\xi}$ is always expressible as the product measure of the two subvectors, under all orderings and partitionings of the elements.

Moreover if the variables $\boldsymbol{\xi}_1, \ldots, \boldsymbol{\xi}_n$ follow the same probability distribution, they are said to be independent and identically distributed (i.i.d.).

1.4.8 Convergence of stochastic sequences

It is not always possible in stochastic programming to use directly a random variable $\boldsymbol{\xi}$. For instance, if $\boldsymbol{\xi}$ is defined on $(\mathbb{R}^n, \mathcal{B}^n, \mu)$, $E[f(\boldsymbol{\xi})]$ involves a multidimensional integral whose solution can be difficult to obtain. Rather than $\boldsymbol{\xi}$, we then use random variables that are close in some sense to $\boldsymbol{\xi}$, but we would like to be sure that refining these approximating random variables leads us towards the true random variables $\boldsymbol{\xi}$. This question can be formally addressed by proving convergence of the sequences of approximating random variables.

Consider the functional expression $\{\boldsymbol{\xi}_n(\omega)\}_{n=1}^{\infty}$ for a random sequence on the space $(\Omega, \mathcal{A}, P)$. When evaluated at a point $\omega \in \Omega$ this denotes a realization of the sequence, the actual collection of real numbers generated when the outcome ω is drawn. If for every $\omega \in \Omega$, $\boldsymbol{\xi}_n(\omega) \to \boldsymbol{\xi}(\omega)$, we say that $\boldsymbol{\xi}_n \to \boldsymbol{\xi}$ surely (or elementwise). It is usually difficult to establish that a stochastic sequence converges surely to a limit. A much more useful notion (because more easily shown), is almost sure convergence, or equivalently, convergence with probability one.

Definition 1.15: Convergence with probability one
A sequence of random variables $\{\boldsymbol{\xi}_n\}_{n=1}^{+\infty}$ converges almost surely to a random variable $\boldsymbol{\xi}$ with probability one if

$$P_{\boldsymbol{\xi}}[\{\omega|\, \boldsymbol{\xi}_n(\omega) \to \boldsymbol{\xi}(\omega) \text{ as } n \to \infty\}] = 1.$$

The notations $\boldsymbol{\xi}_n \xrightarrow{a.s.} \boldsymbol{\xi}$, or $\boldsymbol{\xi}_n \to \boldsymbol{\xi}$ a.s. and a.s.lim $\boldsymbol{\xi}_n = \boldsymbol{\xi}$ are all used to denote almost sure convergence. A weaker kind of convergence is the convergence in probability.

Definition 1.16: Convergence in probability
The sequence $\{\boldsymbol{\xi}_n\}_{n=1}^{+\infty}$ of random variables converges in probability to the random variable $\boldsymbol{\xi}$ if for all $\epsilon > 0$,

$$\lim_{n\to\infty} P_{\boldsymbol{\xi}}[\{\omega\,|\,|\boldsymbol{\xi}_n(\omega) - \boldsymbol{\xi}(\omega)\}| \geq \epsilon] = 0.$$

Almost sure convergence implies convergence in probability; however the converse is not true.

We are sometimes only interested in the marginal cumulative distribution functions, $\{F_n\}_{n=1}^{\infty}$, or equivalently the marginal probability measures $\{\mu_n\}_{n=1}^{\infty}$.

Definition 1.17: Convergence in distribution
Given a sequence of random variables $\{\boldsymbol{\xi}_n\}_{n=1}^{\infty}$ and a random variable $\boldsymbol{\xi}$, $\boldsymbol{\xi}_n$ converges in distribution to $\boldsymbol{\xi}$ if $\boldsymbol{\xi}_n$ has c.d.f. F_n and $\boldsymbol{\xi}$ has c.d.f. ξ, and $\xi_n(x) \to F(x)$ pointwise for each $x \in C$, where $C \subseteq \mathbb{R}$ is the set of points at which F is continuous.

The convergence of the c.d.f.s F_n at points where the limiting function F is continuous, is also called weak convergence. Equivalent notations for weak convergence are $\boldsymbol{\xi}_n \Rightarrow \boldsymbol{\xi}$, and $\boldsymbol{\xi}_n \xrightarrow{D} \boldsymbol{\xi}$. This kind of convergence is sometimes also referred as convergence in probability law, and is denoted by $\boldsymbol{\xi}_n \xrightarrow{L} \boldsymbol{\xi}$. To say that a sequence of random variables converges in distribution means only that the limiting random variable has the given distribution. If both $\boldsymbol{\xi}$ and $\boldsymbol{\zeta}$ follow the distribution specified by F, then $\boldsymbol{\xi}_n \Rightarrow \boldsymbol{\xi}$ and $\boldsymbol{\xi}_n \Rightarrow \boldsymbol{\zeta}$ are equivalent statements.

The corresponding notion of weak convergence for the sequence of measures $\{\mu_n\}$ is given by the following theorem.

Theorem 1.3 *μ_n converges weakly to μ ($\mu_n \Rightarrow \mu$) if and only if $\mu_n(A) \to \mu(A)$ for every $A \in \mathcal{B}$ for which $\mu(\partial A) = 0$.*

1.4.9 The central limit theorem

The central limit theorem plays an important role in statistics, and we will frequently refer to it when studying Monte Carlo samplings (see Chapters 5–7). Basically, it expresses that if a sequence of random variables $\{\boldsymbol{\xi}_t\}_{t=1}^{\infty}$ have means of zero and the partial sums $\sum_{t=1}^{n} \boldsymbol{\xi}_t$, $n = 1, 2, 3, \ldots$, have finite variances s_n^2, with s_n^2 tending to infinity as $n \to \infty$, then, subject to

rather mild additional conditions on the distributions and the sampling process,

$$S_n \overset{def}{=} \frac{1}{s_n} \sum_{t=1}^{n} X_t \Rightarrow N(0,1).$$

Several sets of sufficient conditions have been proposed to ensure such a convergence; we restrict here to the i.i.d. case. We then have the following result.

Theorem 1.4 (Lindeberg-Lévy theorem) *If $\{\boldsymbol{\xi}_t\}_{t=1}^{\infty}$ is an i.i.d. sequence having zero means and variances σ^2,*

$$S_n \overset{def}{=} \frac{1}{\sqrt{n}} \sum_{t=1}^{n} \frac{\boldsymbol{\xi}_t}{\sigma} \Rightarrow N(0,1).$$

The reader can refer for instance to Davidson [39], Chapter 23, for more general results.

Chapter 2

Nonlinear mathematical programming

We consider in this chapter general nonlinear programs. We first characterize a solution of such a program and its related properties, in particular from the optimality viewpoint, which are crucial ingredients for our discussion about consistency of solutions obtained when solving sample average approximations of nonlinear stochastic programs (see Chapter 5). We finally present some classical methods to solve nonlinear programs. The review is far from being exhaustive but focus on trust-region and primal-dual methods since these are the foundations for our algorithmic developments in Part II.

2.1 What is a solution?

Consider the problem

$$\min_{x \in S} f(x),$$

where $f : \mathbb{R}^n \to \mathbb{R}$ and S is a subset of $\mathbb{R}^n$. Ideally we would like to find a global minimizer of f in S, i.e., a point in S where the function reaches its least value. A formal definition is:

Definition 2.1: Global minimizer
A point x^* is a global minimizer of $f : \mathbb{R}^n \to \mathbb{R}$ in S if $f(x^*) \leq f(x)$ for all $x \in S$.

The global minimizer can be difficult to find, especially when the objective function is nonconvex. Most algorithms are able to find only a local minimizer, which is a point that achieves the smallest value of f in its neighbourhood.

Definition 2.2: Local minimizer
A point x^* of $f : \mathbb{R}^n \to \mathbb{R}$ in S is a local minimizer if there is a neighbourhood $\mathcal{V}$ of x^* such that $f(x^*) \leq f(x)$ for $x \in \mathcal{V} \cap S$.

A point that satisfies this definition is sometimes called a weak local minimizer. This terminology distinguishes it from a strict local minimizer, formally defined as follows.

Definition 2.3: Strict local minimizer

A point x^* is a strict local minimizer (also called a strong local minimizer) if there is a neighbourhood $\mathcal{V}$ of x^* such that $f(x^*) < f(x)$ for all $x \in \mathcal{V}$ with $x \neq x^*$.

Strict local minimizers are not always isolated, but all isolated local minimizers are strict.

Example 2.1. *Consider the following function, from Nocedal and Wright [104], page 14:*

$$f(x) = x^4 \cos \frac{1}{x} + 2x^4.$$

$f(x) \in C^2$ and has a strict local minimizer at $x^* = 0$. However there are strict local minimizers at many nearby points x_n, and we can label these points so that $x_n \to 0$, as $n \to 0$.

A strict local minimizer can be seen as a global minimizer is some neighbourhood $\mathcal{V}$ of x^*, however if it is not isolated, it can be very difficult to identify it. This is in particular the case when we use approximations of the objective function (see Chapter 5 on Monte Carlo samplings). Sometimes the problem exhibits properties that are helpful in finding a global minimum. An important special case is that of convex functions, for which every local minimizer is also a global minimizer.

2.2 Optimality conditions

Having defined the notion of solution, there remains the question of how to check that a point is indeed a solution. This is done in practice by testing some optimality conditions for the optimization problem under study. Note that there are many different types of optimality conditions, some of them requiring more of less assumptions than others, and in fact the search for new conditions, e.g., better-suited to some particular classes of problems, remains an active research area.

As said before, finding global optimal solutions is a very difficult task. Consequently, optimality conditions aim at characterizing local solutions. However, they may be extended to cover global solutions in some particular situations, as it is the case for convex programming. In this thesis, we will focus on optimality conditions suitable for general nonlinear programs.

2.2.1 Nonlinear unconstrained programming

Given a function $f : \mathbb{R}^n \to \mathbb{R}$, we consider the problem

$$\min_{x \in \mathbb{R}^n} f(x). \tag{2.1}$$

We start with necessary conditions, where it is assumed that a point x^* is a local minimizer. Properties of $\nabla_x f(x^*)$ and $\nabla^2_{xx} f(x^*)$ are derived, leading to first- and second-order necessary conditions respectively.

Theorem 2.1 (First-order necessary condition) *If x^* is a local minimizer of problem* (2.1) *and f is continuously differentiable in an open neighbourhood of x^*, then $\nabla_x f(x^*) = 0$.*

Note that if $\nabla_x f(x^*) = 0$, x^* is also said to be a stationary point.

Theorem 2.2 (Second-order necessary condition) *If x^* is a local minimizer of problem* (2.1) *and f is twice continuously differentiable in an open neighbourhood of x^*, then $\nabla_x f(x^*) = 0$ and $\nabla^2_{xx} f(x^*)$ is positive semidefinite.*

The next theorem gives sufficient conditions on x^*, that is conditions on $\nabla_x f(x^*)$ and $\nabla_{xx} f(x^*)$ that guarantee that x^* is a (strict) local minimizer of f.

Theorem 2.3 (Second-order sufficient condition) *Assume that $\nabla^2_{xx} f$ is continuous in an open neighbourhood of x^* and that $\nabla_x f(x^*) = 0$ and $\nabla^2_{xx} f(x^*)$ is positive definite. Then x^* is a strict local minimizer of problem* (2.1)*.*

As announced above, optimality conditions are easier in the case of convex programming, and requires fewer assumptions, as shown in the following theorem.

Theorem 2.4 *If f is convex, then any local minimizer x^* of problem* (2.1) *is a global minimizer. If in addition f is differentiable, then any stationary point x^* is a global minimizer of* (2.1)*.*

These theorems show that the gradient gives valuable information when checking optimality, and the numerical cost associated to its evaluation is often manageable. As a result, algorithms for unconstrained optimization usually seek a point x^* at which the gradient of f vanishes. From a practical point of view, a frequently used stopping criterion is that $\|\nabla_x f(x^*)\|$ is less then some small predefined tolerance.

2.2.2 Nonlinear constrained programming

Consider now a constrained problem of the form

$$\min_{x \in X} f(x) \tag{2.2}$$

$$\text{s.t. } c_i(x) = 0,\ i \in \mathcal{E}, \tag{2.3}$$

$$c_i(x) \leq 0,\ i \in \mathcal{I}. \tag{2.4}$$

The Lagrangian (or Lagrange function) for the program (2.2)–(2.4) is defined by

$$\mathcal{L}(x, \lambda) = f(x) + \sum_{i \in \mathcal{E} \cup \mathcal{I}} \lambda_i c_i(x),$$

where λ_i, $i \in \mathcal{E} \cup \mathcal{I}$, are called the Lagrange multipliers. As will be seen throughout the next sections and chapters, the Lagrangian and the Lagrange multipliers are crucial tools for the study and the development of solution approaches in the framework of constrained problems. Note that it is common to refer to x as the primal variables and to the Lagrange multipliers λ_i as the dual variable.

As for unconstrained problems, we start with necessary optimality conditions.

Theorem 2.5 (First-order necessary conditions) *Suppose that x^* is a local solution of* (2.2)–(2.4) *and that a constraint qualification holds at x^*. Then there is a Lagrange multiplier vector λ^*, with components λ_i^*, $i \in \mathcal{E} \cup \mathcal{I}$, such that the following conditions are satisfied at (x^*, λ^*):*

$$\nabla_x \mathcal{L}(x^*, \lambda^*) = 0, \tag{2.5}$$
$$c_i(x^*) = 0, \ \forall i \in \mathcal{E}, \tag{2.6}$$
$$c_i(x^*) \leq 0, \ \forall i \in \mathcal{I}, \tag{2.7}$$
$$\lambda_i^* \geq 0, \ \forall i \in \mathcal{I}, \tag{2.8}$$
$$\lambda_i^* c_i(x^*) = 0, \ \forall i \in \mathcal{E} \cup \mathcal{I}, \tag{2.9}$$

Proof. See Nocedal and Wright [104], Chapter 12. □

The conditions (2.5)–(2.9) are often known as the Karush-Kuhn-Tucker conditions, or KKT conditions for short. These conditions were first derived by Karush in 1939, in his Master's thesis at the University of Chicago [81], and were later re-derived by the Princeton mathematicians Kuhn and Tucker [84], in 1951. A detailed account of the history of the derivation of the KKT conditions and the theory of nonlinear programming can be found in Kjeldsen [82] and references therein.

The constraint qualification assumption aims to ensure that no degenerate behaviour occurs at the value x^*. The most often used is the linear independence constraint qualification, which is based on the active set notion. The active set $\mathcal{A}(x^*)$ at any feasible x^* is the union of the set of indices of equality constraints with the indices of active inequality constraints:

$$\mathcal{A}(x^*) = \{i \in \mathcal{I} \,|\, c_i(x^*) = 0\} \cup \mathcal{E}.$$

Definition 2.4: Linear Independence Qualification Constraint (LICQ)
Given the point x^* and the active set $\mathcal{A}(x^*)$, the linear independence constraint qualification (LICQ) holds if the set of active constraint gradients $\{\nabla_x c_i(x^*), i \in \mathcal{A}(x^*)\}$ is linearly independent.

An useful generalization of the LICQ is the Mangasarian-Fromowitz constraint qualification (MFCQ), which is implied by the LICQ. We refer to Nocedal and Wright, Chapter 12, for a more detailed exposition of optimality conditions, qualification constraints, and their qualification.

An important particular case is again that of convex programming problems: in this framework, every local solution is a global solution and the KKT conditions are both necessary and sufficient provided a constraint qualification holds. A well-known constraint qualification for the convex case is Slater's constraint qualification, which is formulated as follows.

Definition 2.5: Slater's constraint qualification
There exists $\hat{x} \in \mathbb{R}^n$ such that $c_i(\hat{x}) < 0$ for all $i \in \mathcal{I}$ and $c_i(\hat{x}) = 0$ for all $i \in \mathcal{E}$.

We then have the following theorem:

Theorem 2.6 (Convex programming) *If f is convex and the feasible region is convex, any local solution of the constrained problem* (2.2)–(2.4) *is also a global solution. Furthermore, if f and c_i ($i \in \mathcal{E} \cup \mathcal{I}$) are differentiable and if the Slater's constraint qualification holds, the KKT conditions* (2.5)–(2.9) *are necessary and sufficient for x^* (and λ^*) to define a global solution.*

If in addition f is strictly convex, the global solution is unique.

When the problem is nonconvex, the optimality of some point x^* usually requires the knowledge of the second derivatives of f, as in the results below. Let $p = \#\mathcal{I}$ and $m = \#\mathcal{E}$. For given vectors $x \in \mathbb{R}^n$ and $\lambda \in \mathbb{R}^{p+m}$, the set $\mathcal{N}_+(x, \lambda)$ is defined by

$$\mathcal{N}_+(x,\lambda) \stackrel{def}{=} \left\{ w \in \mathbb{R}^n \;\middle|\; \begin{array}{l} \nabla_x c_i(x)^T w = 0 \; \forall i \in \mathcal{E} \cup \{j \in \mathcal{A}(x) \cap \mathcal{I} : \lambda_j > 0\} \\ \text{and } \nabla_x c_i(x)^T w \geq 0 \; \forall i \in \{j \in \mathcal{A}(x) \cap \mathcal{I} : \lambda_j = 0\} \end{array} \right\}.$$

Theorem 2.7 (Second-order necessary conditions) *Assume that x^* is a local solution of problem* (2.2)–(2.4) *and that a constraint qualification holds at x^*. Let λ^* be a Lagrange multiplier vector such that the KKT conditions* (2.5)–(2.9) *are satisfied. Then the curvature of the Lagrangian along directions in $\mathcal{N}_+(x^*, \lambda^*)$ must be nonnegative, that is*

$$w^T \nabla^2_{xx} \mathcal{L}(x^*, \lambda^*) w \geq 0, \text{ for all } w \in \mathcal{N}_+(x^*, \lambda^*). \tag{2.10}$$

A sufficient second-order condition may be derived if we strengthen the inequality (2.10).

Theorem 2.8 (Second-order sufficient conditions) *Assume that for some feasible point $x^* \in \mathbb{R}^n$ there exists a Lagrange vector λ^* such that the KKT conditions* (2.5)–(2.9) *are satisfied. Assume further that*

$$w^T \nabla^2_{xx} \mathcal{L}(x^*, \lambda^*) w > 0, \text{ for all } w \in \mathcal{N}_+(x^*, \lambda^*) \text{ with } w \neq 0.$$

Then x^* is a strict local solution for problem (2.2)–(2.4).

2.2.3 A geometric viewpoint

It is sometimes useful to adopt a geometric, rather than algebraic, viewpoint, while analyzing feasible sets or optimality conditions. One advantage is that the feasible region is then independent of the coordinate system used, and this can sometimes lead to a deeper understanding of the underlying properties of optimization algorithms.

The most important geometric object for nonlinear programming is the cone. A subset C of $\mathbb{R}^n$ is a cone if for all $x \in C$ and $\lambda > 0$, $\lambda x \in C$. Given a cone C, its polar is defined as

$$C^0 \stackrel{def}{=} \{y \in \mathbb{R}^n | \langle y, u \rangle \leq 0, \; \forall u \in C\}.$$

It is trivial that C^0 is also a cone. Moreover $(C^0)^0 = C$ whenever C is a nonempty closed convex cone.

Let X be a closed convex set. Two cones play a special role in the theory of constrained optimization, the normal and tangent cones. The normal cone of X at $x \in X$ is defined to be the

set

$$\mathcal{N}_X(x) \stackrel{def}{=} \{y \in \mathbb{R}^n | \langle y, u - x\rangle \leq 0, \forall u \in X\}.$$

The tangent cone of X at $x \in X$ is the polar of the normal cone at the same point, that is,

$$\mathcal{T}_X(x) \stackrel{def}{=} \mathcal{N}_X(x)^0 = \text{cl}\{\theta(u - x) | \theta \geq 0 \text{ and } u \in X\}.$$

Notice this if x lies in the interior of X, $\mathcal{N}_X(x) = 0$ and $\mathcal{T}_X(x) = \mathbb{R}^n$.

Consider the problem

$$\min_x f(x) \text{ subject to } x \in X, \tag{2.11}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is twice continuously differentiable. The first-order necessary condition for (2.11) can then be expressed in a very simple way.

Theorem 2.9 (First-order conditions) *Suppose that x^* is a local minimizer of f in X. Then*

$$-\nabla_x f(x^*) \in \mathcal{N}_X(x^*).$$

2.3 Methods and algorithms in unconstrained nonlinear programming

We now introduce some methods and algorithms for nonlinear programming, that will be useful for the next chapters. For a more comprehensive review of numerical methods and related properties, we refer to Dennis and Schnabel [42], Minoux [98], Nocedal and Wright [104], Conn, Gould, and Toint [35], but this is not an exhaustive list of references. We study in this section algorithms designed for unconstrained programs, while the next section will be devoted to methods for constrained programming.

We consider mathematical programs of the form

$$\min_{x \in \mathbb{R}^n} f(x).$$

As said in the previous section, we then usually aim at finding points at which the gradient of the objective function vanishes. Therefore a natural approach is to consider the system of n nonlinear equations of n variables given by

$$\nabla_x f(x) = 0, \tag{2.12}$$

and to solve this system. Usually, there is no direct methods except for some particular cases, so we will use iterative procedures to construct a sequence of points converging to a solution of (2.12).

2.3.1 Newton's method

Consider the problem

$$p(x) = 0, \tag{2.13}$$

where $p : \mathbb{R}^n \to \mathbb{R}^n$, $p \in C^1$. Let x_* be a solution of (2.13); x_* will be referred as a root of p. The first-order development of p around x_k is

$$p(x_k + s) = p(x_k) + J(x_k)s + O(s^2), \tag{2.14}$$

where $J(x_k)$ is the Jacobian matrix of $p(\cdot)$, evaluated at x_k. We seek s such that $p(x_k + s) = 0$. It can be approximated by s_k, obtained by solving the following system of equations (known as Newton's equations):

$$J(x_k)s_k = -p(x_k). \tag{2.15}$$

If $J(x_k)$ is invertible, s_k can be computed as

$$s_k = -J(x_k)^{-1}p(x_k).$$

The quantity s_k is then called Newton's direction. Let $x_{k+1} = x_k + s_k$. For a given x_0, this method allows us to construct a sequence of iterates $\{x_k\}_{k=0}^{\infty}$, if $J(x_k)$ is invertible for all k.

The approximation (2.15) is only valid if s_k is small, so the term in $O(s^2)$ in (2.14) can be neglected. Therefore the initial point x_0 must be sufficiently close to x_* to obtain convergence of the Newton's method, which is extremely efficient in this case. The algorithm below summarizes the procedure.

Algorithm 2.1: Newton's method for nonlinear systems

Step 0. Choose x_0 sufficiently closed to x_*.

Step 1. Solve $J(x_k)s_k = -p(x_k)$; let $x_{k+1} = x_k + s_k$.

Step 2. If $p(x_{k+1}) \approx 0$, let $x_* = x_{k+1}$ and stop. Else, let $k = k + 1$; go to step 1.

The application of the Newton's method to the system (2.12) is direct, by setting $p(x) = \nabla_x f(x)$. The equation (2.15) becomes

$$\nabla^2_{xx} f(x_k)s_k = -\nabla_x f(x_k). \tag{2.16}$$

If $\nabla^2_{xx} f(x_k)$ is positive definite, then its inverse exists and s_k is given by

$$s_k = -(\nabla^2_{xx} f(x_k))^{-1}\nabla_x f(x_k). \tag{2.17}$$

We can deduce from (2.16) that

$$\nabla_x f(x_k)^T s_k = -s_k^T \nabla^2_{xx} f(x_k)s_k \leq -\sigma_k \|s_k\|,$$

for some $\sigma_k > 0$. We finally conclude that

$$\nabla_x f(x_k)^T s_k < 0. \tag{2.18}$$

Computing $\nabla^2_{xx} f(x)$ at x_k may be numerically expensive, so it is sometimes preferable to use some approximation H_k of $\nabla^2_{xx} f(x)$. (2.17) then becomes

$$H_k s_k = -\nabla_x f(x_k).$$

It is nevertheless desirable to maintain the property (2.18) since from the Taylor's theorem, for any $d \in \mathbb{R}^n$ and $\epsilon > 0$, we have

$$f(x + \epsilon d) = f(x) + \epsilon \nabla_x f(x)^T d + 0(\epsilon^2).$$

Therefore any d such that $\nabla_x f(x)^T d < 0$ produces a decrease in f if ϵ is sufficiently small; such a d is then called a descent direction. The construction of adequate approximations H_k leads to quasi-Newton methods.

Recall that the condition (2.12) is only a first-order necessary optimality condition; sufficient second-order conditions require that $\nabla^2_{xx} f(x^*)$ is positive definite in addition to $\nabla_x f(x^*) = 0$. This fact and the absence of global convergence led to the development of so-called globalization techniques. The purpose of such approaches is to refine the ideas of Newton's method in order to design algorithms converging to a first-order or second-order critical point from any starting point. These methods can be gathered in two classes, namely linesearch methods and trust-region methods.

2.3.2 Linesearch methods

The linesearch methods are built around the following idea: at each iteration, given an iterate x_k, choose a direction d_k and search along this direction from x_k a new iterate x_{k+1} such that $f(x_{k+1}) \leq f(x_k)$. In theory, once a direction d_k is found, the best decrease in f would be obtained by solving

$$\min_{\alpha > 0} f(x_k + \alpha d_k). \tag{2.19}$$

Usually, the exact minimization of (2.19) is expensive and unnecessary, so instead of solving this problem exactly, it is more efficient to solve (2.19) only approximately. We then seek some α_k such that a sufficient decrease of f is obtained at $x_{k+1} \overset{def}{=} x_k + \alpha_k d_k$. The new point x_{k+1} is then the next iterate and the whole process is repeated.

2.3.3 Trust-region methods

Another broad class of globalization approaches for solving nonlinear unconstrained mathematical programs is the class of trust-region methods. The main idea of a trust-region algorithm is, at a current iterate x_k, to calculate a trial point $x_k + s_k$ by minimizing a model m_k of the objective function at x_k inside a trust region, at each iteration. This region is defined as

$$\mathcal{B}_k = \{x \in \mathbb{R}^m \mid \|x - x_k\|_k \leq \Delta_k\},$$

where Δ_k is called the trust-region radius, and where $\|\cdot\|_k$ is an iteration-dependent norm. A classical choice is the 2-norm, but other norms can be more efficient when the geometry of the

problem is taken into account. The predicted and actual decreases in objective function values are then compared. If the agreement is sufficiently good, the trial point becomes the new iterate and the trust-region radius is (possibly) enlarged. If this agreement is poor, the trust region is shrunk in order to improve the quality of the model. A formal description of the basic trust-region algorithm follows.

Algorithm 2.2: Basic trust-region algorithm (BTR)

Step 0. Initialization. An initial point x_0 and an initial trust-region radius Δ_0 are given. The constants η_1, η_2, γ_1, and γ_2 are also given and satisfy

$$0 < \eta_1 \leq \eta_2 < 1 \text{ and } 0 < \gamma_1 \leq \gamma_2 < 1. \tag{2.20}$$

Compute $f(x_0)$ and set $k = 0$.

Step 1. Model definition. Choose $\| \cdot \|_k$ and define a model m_k in $\mathcal{B}_k$.

Step 2. Step calculation. Compute a step s_k that "sufficiently reduces the model" m_k and such that $x_k + s_k \in \mathcal{B}_k$.

Step 3. Acceptance of the trial point. Compute $f(x_k + s_k)$ and define

$$\rho_k = \frac{f(x_k) - f(x_k + s_k)}{m_k(x_k) - m_k(x_k + s_k)}. \tag{2.21}$$

If $\rho_k \geq \eta_1$, then define $x_{k+1} = x_k + s_k$; otherwise define $x_{k+1} = x_k$.

Step 4. Trust-region radius update.

$$\Delta_{k+1} \in \begin{cases} [\Delta_k, \infty) & \text{if } \rho_k \geq \eta_2, \\ [\gamma_2 \Delta_k, \Delta_k] & \text{if } \rho_k \in [\eta_1, \eta_2), \\ [\gamma_1 \Delta_k, \gamma_2 \Delta_k] & \text{if } \rho_k < \eta_1. \end{cases}$$

Increment k by 1 and go to Step 1.

In this description, reasonable values for the constants of (2.20) are, for instance,

$$\eta_1 = 0.01, \ \eta_2 = 0.9, \ \text{ and } \gamma_1 = \gamma_2 = 0.5,$$

but other values may be selected. The most classical choice for m_k is a quadratic function of the type

$$m_k(x_k + s) = f(x_k) + \nabla_x f(x_k)^T s + \frac{1}{2} s^T H_k s,$$

where H_k is either the Hessian $\nabla^2_{xx} f(x_k)$ or some approximation of it. Note that some authors have suggested using other, more general, models, like the approach of Alexandrov, Dennis, Lewis and Torczon [1].

If $\rho_k \geq \eta_1$ in Step 1, the iteration k is said to be successful since the candidate point $x_k + s_k$ is accepted; otherwise the iteration is declared unsuccessful and the new point is rejected. If $\rho_k \geq \eta_2$, the agreement between the model and the function is particularly good, so the iteration is said to be very successful. This then suggests increasing the trust-region radius, as in Step 4, in order to allow a longer step at the next iteration.

2.4 Methods for nonlinear constrained programming

We return now to the general nonlinear program (1.1)–(1.3), that we restate for clarity:

$$\min_{x \in X} f(x) \tag{2.22}$$

$$\text{s.t. } c_i(x) = 0, \ i \in \mathcal{E}, \tag{2.23}$$

$$c_i(x) \leq 0, \ i \in \mathcal{I}. \tag{2.24}$$

The constraints are then often added to the objective so we can benefit from methods devoted to nonlinear unconstrained programming. Reformulations of (2.22)–(2.24) leads to various classes of methods for solving this program. We will review some of them in the remaining of this chapter, in particular those that will be useful for our needs.

2.4.1 Penalty, barrier and augmented Lagrangian methods

The three types of methods reviewed in this section seek a solution by replacing the original constrained problem (2.22)–(2.24) by a sequence of unconstrained subproblems. These methods described hereafter may be also viewed as attempts to solve two problems simultaneously since their — reformulated — objective function combines both the original objective $f(x)$ and a measure of the constraint violation, called a penalty term that we denote by $\theta(x)$. They are implemented through the use of a so-called penalty parameter ρ, which is used to determine and control the weight of the penalty term in the new objective function, combining $f(x)$ and $\theta(x)$. To achieve this, the value of ρ is made iteration-dependent, and the following problem is solved:

$$\min_s f(x_k + s) + \rho_k \theta(x_k + s), \tag{2.25}$$

where the sequence $\{\rho_k\}$ tends to $+\infty$, so the penalty term plays a prominent role as k increases and the solution is enforced to be feasible. $f(x) + \rho\theta(x)$ is called the penalty function, or also merit function.

Penalty and augmented Lagrangian methods

Let consider the equality-constrained optimization problem

$$\min_x f(x) \tag{2.26}$$

$$\text{s.t. } c_i(x) = 0, \quad i \in \mathcal{E}. \tag{2.27}$$

The ideas is to include the equality constraints (2.27) into a penalty term that measures the constraints violation and is added to the objective (2.26), as in (2.25). Thus the penalty term is nonzero when x is infeasible with respect to the constraints. For this reason, such approaches are also known as exterior penalty methods.

The quadratic penalty method uses

$$Q(x;\mu) \stackrel{def}{=} \frac{1}{2\mu} \sum_{i\in\mathcal{E}} c_i^2(x) \tag{2.28}$$

as a penalty term, where the penalty parameter is given by $\rho = \frac{1}{2\mu} > 0$. At iteration k, we minimize the function

$$f(x_k) + Q(x_k;\mu_k),$$

where μ_k is constructed in such a way that $\mu_k \to 0$ when $k \to \infty$.

When inequality constraints are present, as in (2.2)–(2.4), the quadratic penalty function (2.28) is augmented by a term reflecting their violation and becomes

$$Q(x;\mu) \stackrel{def}{=} \frac{1}{2\mu} \sum_{i\in\mathcal{E}} c_i^2(x) + \frac{1}{2\mu} \sum_{i\in\mathcal{I}} (\max\{c_i(x), 0\})^2.$$

Note however that adding such terms may result in a less smooth penalty function, which is of course less convenient when it is minimized. More generally, a major drawback to penalty methods is that the Hessian $\nabla^2_{xx} Q(x;\mu)$ usually becomes ill-conditioned near the minimizer of $Q(x;\mu)$ when μ tends to 0.

Another type of penalty term is given by

$$E(x;\mu) \stackrel{def}{=} \frac{1}{\mu} \sum_{i\in\mathcal{I}} \max\{0, -c_i(x)\} + \frac{1}{\mu} \sum_{i\in\mathcal{E}} |c_i(x)|.$$

It can be shown that $\phi(x;\mu) \stackrel{def}{=} f(x) + E(x;\mu)$, called the ℓ_1 exact merit function, satisfies the following definition.

Definition 2.6: Exact merit function
A merit function $\psi(x;\mu)$ is said to be exact if there is a positive scalar μ^* such that for any $\mu \in (0, \mu^*]$, any local solution of the nonlinear programming problem is a local minimizer of $\psi(x;\mu)$.

In practice however, choosing the adequate value of μ prior an iteration is difficult when solving most problems. Moreover, exact penalty functions are often nondifferentiable. This is for instance the case for the ℓ_1 exact penalty function since its first derivative is not defined at any x for which $c_i(x) = 0$ ($i \in \mathcal{E} \cup \mathcal{I}$).

A way to alleviate these difficulties is to add the penalty term to the Lagrangian function instead of the objective. The resulting function is called the augmented Lagrangian function, that can be formally formed as follows:

$$\mathcal{L}_A(x,\lambda;\mu) \stackrel{def}{=} f(x) + \sum_{i\in\mathcal{E}} \lambda_i c_i(x) + \frac{1}{2\mu} \sum_{i\in\mathcal{E}} c_i^2(x).$$

An augmented Lagrangian method then performs the minimization of $\mathcal{L}_A(x, \lambda; \mu)$ with respect to x, and uses some rule for updating the estimates of the Lagrange multipliers from iteration to iteration. A very popular implementation of the augmented Lagrangian method is that of Conn, Gould, and Toint [32] in the LANCELOT package.

Barrier methods

We now start by assuming that the problem to be solved involves inequality constraints only:

$$\min_x f(x) \tag{2.29}$$

$$\text{s.t. } c_i(x) \leq 0, \quad i \in \mathcal{I}. \tag{2.30}$$

A possible approach for solving such a problem is based on the use of barrier functions, whose properties are the following:

- it is infinite everywhere except in the strictly feasible region

$$\text{strict}\{\mathcal{F}\} = \{x \in \mathbb{R}^n : c_i(x) < 0 \text{ for all } i \in \mathcal{I}\};$$

- it is smooth inside strict$\{\mathcal{F}\}$;
- its value increases to ∞ as x approaches the boundary of strict$\{\mathcal{F}\}$.

A well-known barrier function is the logarithmic barrier function (also called log-barrier)

$$P(x; \mu) = -\mu \sum_{i \in \mathcal{I}} \log(-c_i(x)),$$

where this time μ is called the barrier parameter. As for penalty methods, the idea is to minimize

$$f(x) + P(x; \mu) \tag{2.31}$$

with decreasing values of μ. This process can be shown to converge to a solution of (2.29)–(2.30) as $\mu \to 0$, under some reasonable assumptions.

The main issue with barrier methods is that the minimizer of (2.31) is more difficult to find when μ is close to 0, which results from the poor scaling of the function $P(x; \mu)$.

For equality constraints, the barrier approach cannot be applied since such constraints are always active. Therefore an alternative strategy must be used. For instance, one may combine the penalty and the barrier methods and add a quadratic penalty term to the barrier function, yielding

$$f(x) - \mu \sum_{i \in \mathcal{I}} \log(-c_i(x)) + \frac{1}{2\mu} \sum_{i \in \mathcal{E}} c_i^2(x).$$

However, as it might be expected, minimizing such an objective function suffers from the numerical difficulties encountered with the use of both penalty and barrier functions.

Despite the difficulties encountered in implementing log-barrier methods effectively, they have regained interest in recent years, mainly because of their connection to primal-dual interior point methods, which have been shown to perform extremely well for large linear programming and convex quadratic programming problems. Interior point methods have also been successfully applied to nonlinear programming. We will explore primal-dual interior points methods more deeply in Section 2.4.2.

2.4.2 Primal-dual interior point methods

Interior point methods are very popular in linear programming (see in particular Wright [142]) and have been successfully extended in the framework of nonlinear (convex and nonconvex) programming. We present in this section the basic concepts to these methods, that will serve as foundations to the primal-dual algorithm for nonlinear stochastic problems with scenarios, presented in Chapter 4. We first discuss the linear programming case and then introduce the nonlinear case, for which we clarify its relationship to the log-barrier approach.

Linear programming

We start by considering the linear program

$$\min_x c^T x, \tag{2.32}$$

$$\text{s.t. } Ax = b, \tag{2.33}$$

$$x \geq 0, \tag{2.34}$$

where c and x are vectors in $\mathbb{R}^n$, b is a vector in $\mathbb{R}^m$, and A is an $m \times n$ matrix. The dual problem for (2.32)–(2.34) is

$$\begin{aligned} &\max_\lambda b^T \lambda, \\ &\text{s.t. } A^T \lambda + s = c, \\ &\quad s \geq 0. \end{aligned}$$

The KKT system for (2.32)–(2.34) can be formulated as follows:

$$F(x, \lambda, s) = \begin{pmatrix} A^T \lambda + s - c \\ Ax - b \\ XSe \end{pmatrix} = 0, \tag{2.35}$$

$$(x, s) \geq 0, \tag{2.36}$$

where $X = \text{diag}(x_1, \ldots, x_n)$ and $S = \text{diag}(s_1, \ldots, s_n)$. Recall that e is the vector whose n components are 1. The constraint $XSe = 0$ is known as the complementarity condition, expressing a true combinatorial requirement: "if a constraint is non-zero then its corresponding dual variable must be zero" and vice-versa. A primal-dual interior point method finds a solution (x^*, λ^*, s^*) to the KKT system (2.35)–(2.36) by applying variants of Newton's method to the

equalities (2.35), while the direction and the step length are chosen such that inequalities (2.36) hold strictly at each iteration, that is

$$x > 0 \quad \text{and} \quad s > 0,$$

hence the qualifier interior point. We see that in primal-dual methods, the Lagrange multipliers are treated as independent variables, with equal status to the primal variables x.

The Newton's system for (2.35) is

$$J(x, \lambda, s) \begin{pmatrix} \Delta x \\ \Delta \lambda \\ \Delta s \end{pmatrix} = -F(x, \lambda, s), \tag{2.37}$$

where $J(x, \lambda, s)$ denotes the Jacobian of $F(x, \lambda, s)$. A vector (x, λ, s) is said to be strictly feasible if it satisfies the two first equalities in (2.35), i.e., if

$$A^T \lambda + s - c = 0 \quad \text{and} \quad Ax = b.$$

(2.37) is then equivalent to

$$\begin{pmatrix} 0 & A^T & I \\ A & 0 & 0 \\ S & 0 & X \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta \lambda \\ \Delta s \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ -XSe \end{pmatrix}.$$

Taking full steps $(\Delta x, \Delta \lambda, \Delta s)$ might yield a violation of (2.36), so a possible strategy is to perform a linesearch along this direction and to update the variables as follows:

$$(x_{k+1}, \lambda_{k+1}, s_{k+1}) = (x_k, \lambda_k, s_k) + \alpha_k (\Delta x_k, \Delta \lambda_k, \Delta s_k), \tag{2.38}$$

with $\alpha_k \in (0, 1]$. In practice, this often produces very small values for the α_k's in order to maintain the positiveness of x and s. Moreover combinatorial conditions as the complementarity condition may be very hard to satisfy, especially for large problems. Primal-dual interior point methods rather direct the step towards the interior of the nonnegative orthant $\{x \geq 0,\ s \geq 0\}$, while preventing x and s from being too close to its boundary. This usually allows to take longer steps than along the pure Newton directions for F, before violating the nonnegativity condition. To do this, we perturb the complementarity condition by introducing a small perturbation $\mu > 0$, i.e., we replace the condition $XSe = 0$ by

$$XSe = \mu e. \tag{2.39}$$

The optimization is then repeated for successively smaller values of μ, as in barrier methods. This implies that $x_i \neq 0$, $s_i \neq 0$ $(i = 1, \ldots, n)$. In other terms the points must be interior since the constraints (2.36) must be strictly satisfied. The KKT system for (2.32)–(2.34) is therefore replaced by the following one:

$$A^T \lambda + s = c, \tag{2.40}$$

$$Ax = b, \tag{2.41}$$

$$XSe = \mu e, \tag{2.42}$$

$$x > 0,\ s > 0. \tag{2.43}$$

The set of points

$$\mathcal{C} = \{(x, \lambda, s) \text{ satisfying system (2.40)–(2.43) with } \mu > 0\}$$

is called the central path. An important result, due to Fiacco and McCornick [51], says that if convergence of the points of the central path $\mathcal{C}$ is observed as μ goes to zero, then they must converge to a primal-dual solution of the linear program (2.32)–(2.34) (see for instance Theorem 3.12 in Forsgren, Gill and Wright [54]). Therefore primal-dual algorithms take Newton steps toward points on $\mathcal{C}$ for which $\mu > 0$, rather than pure Newton steps for F. More precisely, the following system is solved at each iteration

$$\begin{pmatrix} 0 & A^T & I \\ A & 0 & 0 \\ S & 0 & X \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta \lambda \\ \Delta s \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ -XSe + \sigma\tau e \end{pmatrix}, \tag{2.44}$$

where $\mu \stackrel{def}{=} \sigma\tau$, with

$$\sigma \in (0, 1] \quad \text{and} \quad \tau = \frac{1}{n}\sum_{i=1}^{n} x_i s_i = \frac{x^T s}{n}.$$

The parameters σ and τ are called the centring parameter and the duality measure respectively. Once this system is solved, the iterates are updated as in (2.38), where α_k is chosen so that the variables x_{k+1} and s_{k+1} are strictly positive. The performance of the algorithm is nevertheless dependent of the choice of the values for σ and α_k. In particular, the iterates are usually required to lie within some neighbourhood of the central path $\mathcal{C}$, by choosing a judicious α_k in (2.38). In practice, the iterates then follow the central path to a solution of the linear program, and the resulting method is called a path-following method.

So far we have assumed that all iterates are strictly feasible, so it is in particular the case for the starting point (x_0, λ_0, s_0). It may however be difficult to find such an initial point and often, we only assume that x_0 and s_0 are strictly positive. Consequently, we have to modify the system (2.44) in order to ensure that both feasibility and centrality are improved at each iteration. This leads to the following system:

$$\begin{pmatrix} 0 & A^T & I \\ A & 0 & 0 \\ S & 0 & X \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta \lambda \\ \Delta s \end{pmatrix} = - \begin{pmatrix} A^T\lambda + s - c \\ Ax - b \\ XSe - \sigma\tau e \end{pmatrix}.$$

The most popular implementation of primal-dual interior point methods for solving linear programs is that of Mehrotra [97] (see also Lustig, Marsten and Shanno [89]), which is a so-called predictor-corrector algorithm since it adds a corrector step to the direction computed with (2.44) so that it follows more closely a trajectory to the primal-dual solution set. Further, Merhotra's algorithm chooses the value of σ adaptively (instead of assigning it a value before computing the search direction), depending on whether centring is needed or not at the considered iteration.

Nonlinear programming

There exists several extensions of primal-dual interior point methods for solving more general problems than linear programs. For simplicity, we restrict here our attention to inequality-

constrained problems of the form (2.29)–(2.30):

$$\begin{aligned} &\min_x f(x) \\ &\text{s.t. } c_i(x) \leq 0, \quad i \in \mathcal{I}. \end{aligned}$$

We first introduce slack variables in (2.30), which yields the following problem:

$$\min_x f(x) \tag{2.45}$$
$$\text{s.t. } g(x) + s = 0, \tag{2.46}$$
$$s \geq 0. \tag{2.47}$$

The KKT conditions for this problem are

$$\nabla_x f(x) + \sum_{i \in \mathcal{I}} \lambda_i \nabla_x c_i(x) = 0, \tag{2.48}$$
$$c(x) + s = 0, \tag{2.49}$$
$$\lambda_i s_i = 0, \text{ for all } i \in \mathcal{I}, \tag{2.50}$$
$$\lambda \geq 0, \, s \geq 0. \tag{2.51}$$

The bound constraints (2.51) play a vital role: primal-dual points (x, λ, s) that satisfy (2.48)–(2.50) but violate (2.51) typically lie far from a solution of (2.29)–(2.30). As before, we perturb the complementarity condition with the a parameter μ, and consider the system

$$\begin{pmatrix} \nabla_x f(x) + A(x)^T \lambda \\ g(x) + s \\ \Lambda S e - \mu e \end{pmatrix} = 0, \tag{2.52}$$

with $\lambda > 0$, $s > 0$, instead of (2.48)–(2.51). The matrix $A(x)$ denotes the Jacobian of the constraints, $\Lambda = \text{diag}(\lambda_1, \ldots, \lambda_n)$ and $S = \text{diag}(s_1, \ldots, s_n)$. The Newton system associated to (2.52) is then

$$\begin{pmatrix} \nabla^2_{xx} \mathcal{L}(x, \lambda) & A(x)^T & 0 \\ A(x) & 0 & I \\ 0 & S & \Lambda \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta \lambda \\ \Delta s \end{pmatrix} = - \begin{pmatrix} \nabla_x f(x) + A(x)^T \lambda \\ g(x) + s \\ \Lambda S e - \mu e \end{pmatrix}. \tag{2.53}$$

An alternative way to obtain these equations is to solve problem (2.29)–(2.30) by means of barrier functions as in Section 2.4.1. Using a logarithmic barrier function for dealing with the nonnegativity constraints (2.47), we obtain the problem

$$\begin{aligned} &\min_x f(x) - \mu \sum_{i \in \mathcal{I}} \log s_i, \\ &\text{s.t. } g(x) + s = 0. \end{aligned}$$

The Lagrangian associated to this problem is

$$\mathcal{L}(x, \lambda, s) = f(x) - \mu \sum_{i \in \mathcal{I}} \log s_i + \lambda^T (g(x) + s)$$

and it follows that the first-order optimality conditions are

$$\begin{aligned} \nabla_x f(x) + A(x)^T \lambda &= 0, \\ g(x) + s &= 0, \\ \mu S^{-1} e - \Lambda &= 0, \end{aligned}$$

or, if we multiply the third equation by S,

$$\begin{aligned} \nabla_x f(x) + A(x)^T \lambda &= 0, \\ g(x) + s &= 0, \\ \Lambda S e - \mu e &= 0, \end{aligned}$$

which is equivalent to (2.52). Note that, in the log-barrier method, we eliminate s and λ directly from the nonlinear system before applying Newton's method.

For nonlinear functions however, the matrix

$$\begin{pmatrix} \nabla^2_{xx} \mathcal{L}(x, \lambda) & A(x)^T & 0 \\ A(x) & 0 & I \\ 0 & S & \Lambda \end{pmatrix}$$

may be indefinite, so the Newton's system (2.53) may have no solution. Interior point methods for solving nonlinear programs therefore turn to quasi-Newton techniques such that the primal-dual systems to be solved are well behaved. Some recent works are those of Conn, Gould, and Toint [34], Conn, Gould, Orban, and Toint [30], Gay, Overton and Wright [59], Gould, Orban, Sartenaer, and Toint [65], M. Ulbrich, S. Ulbrich and Vicente [136], Vanderbei and Shanno [137], Vicente [138] and Wächter [139]. In Chapter 4, we will also show that the primal-dual interior point methods can also benefit from the structure of nonlinear stochastic programs; in particular we will present a decomposition scheme for the quasi-Newton's systems that we have developed with Sartenaer Jie Sun [11].

Chapter 3

Stochastic program formulations

We have seen In Chapter 1 that stochastic programs are mathematical programs in which uncertainty is introduced by means of random variables. However, until now, we did not have produced a manageable formulation of stochastic programs. It is therefore our purpose in this chapter to review some of the most frequently encountered formulations of stochastic programs and give some of their main properties, in particular the recourse programs. We also briefly introduce other aspects of stochastic programming, as chance-constrained programs, which form an important class in stochastic programming, while keeping this presentation succinct seeing that we will not use these additional developments in the rest of the thesis.

3.1 General formulation

Since we do not know the realization of the random variable $\boldsymbol{\xi}$, an operational decision is to take the best decision in "average". Probability theory gives us an efficient definition of "average" with the concept of expectation. The general mathematical program (1.5)–(1.6) is therefore reformulated as follows:

$$\min_{x \in X} E_{\boldsymbol{\xi}}[f(x, \boldsymbol{\xi})] \tag{3.1}$$

$$\text{s.t. } E_{\boldsymbol{\xi}}[c_i(x, \boldsymbol{\xi})] \leq 0, \ i \in \mathcal{I}, \tag{3.2}$$

$$E_{\boldsymbol{\xi}}[c_i(x, \boldsymbol{\xi})] = 0, \ i \in \mathcal{E}, \tag{3.3}$$

where $\boldsymbol{\xi}$ is a real random vector defining the probability space $(\Xi, \mathcal{F}, P_{\boldsymbol{\xi}})$, where Ξ is the support of $\boldsymbol{\xi}$, $\mathcal{F}$ is a σ-field of subsets of Ξ, and $P_{\boldsymbol{\xi}}$ is the associated probability measure. Most of stochastic programs can be stated in the form (3.1)–(3.3). However in some cases this is not the most efficient formulation, as we will see in the next sections. We introduce two important classes of stochastic programs: recourse programs and chance-constrained programs, with a stress put on recourse programs, that we consider explicitly in Chapter 4. We can also apply consistency results of Chapter 5 to such programs. Note also that some specific problems require more general formulations than (3.1)–(3.3). It is in particular the case with mixed logit models (see Chapter 6), but for simplicity we will only consider the formulation (3.1)–(3.3) in this chapter, which is more standard in stochastic programming literature.

3.2 Recourse programs

3.2.1 Decisions and stages

A basic, and often used, problem in stochastic programming is the situation where decisions have to be taken without a complete knowledge of the problem. Further decisions or recourse actions can be taken after the uncertainty is disclosed. The uncertainty is here described with random variables whose probability measures are available; the particular values $\xi = \xi(\omega)$ that the various random variables will take are thus only known at some point in the decisions process. The set of decisions is then divided into two groups.

First-stage decisions A number of decisions have to be taken before the experiment. All these decisions are called first-stage decisions and the period during which these decisions are taken is called the first stage.

Second-stage decisions A number of additional decisions, also called recourse actions, can be taken after the experiment. They are called second-stage decisions and the corresponding period, second-stage.

Some illustrations of these concepts can be found in Birge and Louveaux [21], Chapter 1.

We denote the first-stage decisions by the vector x, and the second-stage decisions by the vector y. These latter are sometimes written $y(\xi)$ or even $y(\xi, x)$, to stress that y differs according to the outcome of the random experiment and of the first-stage decision. The sequence of events and decisions is thus summarized as

$$\boxed{x \rightarrow \xi \rightarrow y(\xi, x)}$$

3.2.2 Two-stage stochastic programming with recourse

We seek therefore to take first-stage decisions that are in average optimal, with the possibility to take some recourse decisions to face the additional knowledge that we will obtain after disclosure of the uncertainty. This suggests defining an objective function and constraints associated to the first-stage variables, while for the second-stage decisions we consider an additional objective and constraints that depend on the realization of the random variables. We then combine the two stages by adding the expectation of the second-stage objective to the first-stage objective. The resulting program is called the two-stage stochastic (nonlinear) program with (additive) recourse.

Definition 3.1: Two-Stage Stochastic Program with Recourse (TSSPR)

$$\begin{aligned} \min_x z(x) &= f_1(x) + \mathcal{Q}(x) \\ \textit{s.t. } & c_{1,i}(x) \leq 0,\ i = 1, \ldots, \overline{m}_1, \\ & c_{1,i}(x) = 0, i = \overline{m}_1 + 1, \ldots, m_1, \end{aligned} \tag{3.4}$$

where $\mathcal{Q}(x) = E_{\boldsymbol{\xi}}[Q(x,\xi)]$, *and*

$$\begin{aligned} Q(x,\xi) &= \min_y f_2(y(\xi),\xi) \\ \text{s.t. } & t_{2,i}(x,\xi) + c_{2,i}(y(\xi),\xi) \le 0,\ i = 1,\dots,\overline{m}_2, \\ & t_{2,i}(x,\xi) + c_{2,i}(y(\xi),\xi) = 0,\ i = \overline{m}_2 + 1,\dots,m_2. \end{aligned} \tag{3.5}$$

For simplicity, we will usually omit the term stochastic as well as the nonlinear qualification, while most authors consider only linear programs when speaking of TSSPR. For the linear case, the reader can refer to the monograph of Frauendorfer [55]. We suppose here that all functions $f_2(\cdot,\xi),\ t_{2,i}(\cdot,\xi)$ and $c_{2,i}(\cdot,\xi)$ are continuous for any fixed ξ, and measurable in ξ for any fixed first argument. Given this assumption, $Q(x,\xi)$ is measurable. The difficulty inherent in stochastic programming clearly lies in the computational burden of computing $\mathcal{Q}(x)$ for all x.

Note that (3.4)–(3.5) can be expressed in a more general way. First the first-stage term is sometimes considered to be random too; second we can express first-stage constraints in the form $x \in X$. Therefore (3.4) can be replaced by

$$\min_{x\in X} z = f_1(x,\xi) + \mathcal{Q}(x) \tag{3.6}$$

(see for instance Kall and Wallace [80], page 26). Furthermore, x acts separately in the constraints of the recourse problem (3.5). This is useful to develop optimality conditions that are separable between the first- and second-period variables. Others formulations allow for nonseparable constraints and dependence of the second-stage objective on x.

Feasible sets

It is often convenient to define the feasible sets associated to the different stages of the stochastic recourse program. First let K_1 be the set determined by the fixed constraints, namely those that do not depend on the particular realization of the random vector:

$$K_1 \overset{def}{=} \{x \,|\, c_{1,i}(x) \le 0,\ i = 1,\dots,\overline{m}_1; c_{1,i}(x) = 0,\ i = \overline{m}_1 + 1,\dots,m_1\}.$$

K_1 is called the first-stage feasible set. Similarly, the second-stage feasible set is given by

$$K_2 \overset{def}{=} \{x \,|\, \mathcal{Q}(x) < \infty\}.$$

The TSSPR (3.4)–(3.5) can then be reformulated as

$$\begin{aligned} &\inf z(x) = f_1(x) + \mathcal{Q}(x) \\ &\text{s.t. } x \in K_1 \cap K_2. \end{aligned}$$

We also introduce the elementary feasible sets, associated to the realizations of the random vector, defined as

$$K_2(\xi) \overset{def}{=} \left\{ x \;\middle|\; \begin{array}{rl} \exists\, y(\xi) \text{ such that} & t_{2,i}(x,\xi) + c_{2,i}(y(\xi),\xi) \le 0,\ i = 1,\dots,\overline{m}_2 \\ \text{and} & t_{2,i}(x,\xi) + c_{2,i}(y(\xi),\xi) = 0,\ i = \overline{m}_2 + 1,\dots,m_2 \end{array} \right\}.$$

The set

$$K_2^P = \bigcap_{\xi \in \Xi} K_2(\xi)$$

is said to define the possibility interpretation of second-stage feasible set. If $\boldsymbol{\xi}$ is a continuous random variable, we may have that K_2 differs from K_2^P (see Birge and Louveaux [21], page 125, for an example). The set K_2 contains decisions that are feasible almost surely, while K_2^P is the set of decisions that are feasible surely.

Special cases: relatively complete, complete, and simple recourse

The feasible sets and objective functions can have special properties that are particularly useful for computations. We say that the stochastic program (3.4)–(3.5) has a relatively complete recourse if $K_1 \subset K_2$. In other words, every solution x that satisfies the first-period constraints has a feasible completion in the second stage. Relatively complete recourse is very useful in practice and in many of the theoretical results. Unfortunately, its identification can be difficult since it requires some knowledge of the sets K_1 and K_2.

Assume now that the recourse objective is linear, i.e.,

$$\mathcal{Q}(x) = E_{\boldsymbol{\xi}}[Q(x, \boldsymbol{\xi})]$$

where

$$Q(x, \xi) = \min_y \{\langle q(\xi), Ty \rangle \mid W(\xi)y = h(\xi) - T(\xi)x,\ y \geq 0\},$$

where $W(\xi)$ is a matrix of size $m_2 \times n_2$ and is called the recourse matrix. For each ξ, $T(\xi)$ is $m_2 \times n_2$, $q(\xi) \in \mathbb{R}^{n_2}$ and $h(\xi) \in \mathbb{R}^{m_2}$. Assume furthermore that $W(\xi)$ is deterministic, i.e., $W(\xi) = W$, for all $\xi \in \Xi$. The recourse is then said to be fixed. If furthermore there exists $y \geq 0$ such that $Wy = t$, for all $t \in \mathbb{R}^{m_2}$, the stochastic program has a complete recourse. Moreover if $W = [I, -I]$, the TSSPR is said to have a simple recourse, and $\boldsymbol{y}$ is divided correspondingly as $(\boldsymbol{y}^+, \boldsymbol{y}^-)$ and $\boldsymbol{q}$ as $(\boldsymbol{q}^+, \boldsymbol{q}^-)$. Note that, in this case, the optimal values of $y_i^+(\xi)$ and $y_i^-(\xi)$, $i = 1, \ldots, m_2$, are determined purely by the sign of $h_i(\xi) - T_i(\xi)x$ provided that $\boldsymbol{q}_i^+ + \boldsymbol{q}_i^- \geq 0$ with probability one. We then have the following result.

Theorem 3.1 *Consider a feasible two-stage stochastic program with simple recourse, such that $\boldsymbol{\xi}$ has finite second moments (i.e. the variance-covariance matrix $C = E_{\boldsymbol{\xi}}[(\boldsymbol{\xi} - E_{\boldsymbol{\xi}}[\boldsymbol{\xi}])(\boldsymbol{\xi} - E_{\boldsymbol{\xi}}[\boldsymbol{\xi}])^T]$ is finite). Then $\mathcal{Q}(x)$ is finite if and only if $\boldsymbol{q}_i^+ + \boldsymbol{q}_i^- \geq 0$, $i = 1, \ldots, m_2$, almost surely.*

Proof. See Birge and Louveaux [21], page 92. □

The simple recourse benefits from strong theoretical results, especially for linear stochastic programs. Since linear stochastic programming is not the main concern of this work, we refer the reader to Birge and Louveaux [21] for more results.

Differentiability of the stochastic program

Many theoretical results and algorithms in the nonlinear optimization field assume that the objective and constraints are differentiable, and often their derivatives too. It is therefore interesting to study the smoothness properties of a stochastic program.

ϕ is partially differentiable at some point $(\hat{x}, \hat{y})$ with respect to x, if there exists a function called the partial derivative and denoted by $\frac{\partial \phi(x,y)}{\partial x}$, such that

$$\frac{\phi(\hat{x}+h,\hat{y})-\phi(\hat{x},\hat{y})}{h} = \frac{\partial \phi(\hat{x},\hat{y})}{\partial x} + \frac{r(\hat{x},\hat{y};h)}{h},$$

where the residuum r satisfies

$$\frac{r(\hat{x},\hat{y};h)}{h} \to 0$$

when $h \to 0$. Similarly, the recourse function is partially differentiable with respect to x_j in $(\hat{x}, \hat{\xi})$ if there is a function $\frac{\partial Q(x,\xi)}{\partial x_j}$ such that

$$\frac{Q(\hat{x}+he_j,\hat{\xi})-Q(\hat{x},\hat{\xi})}{h} = \frac{\partial Q(\hat{x},\hat{\xi})}{\partial x_j} + \frac{\rho_j(\hat{x},\hat{\xi};h)}{h}$$

with

$$\frac{\rho_j(\hat{x},\hat{\xi};h)}{h} \to 0,$$

as $h \to 0$. This yields the following result.

Theorem 3.2 *If $Q(x,\xi)$ is partially differentiable with respect to x_j at some $\hat{x}$ almost surely, if its partial derivative $\frac{\partial Q(\hat{x},\xi)}{\partial x_j}$ is integrable and if the residuum satisfies*

$$\frac{1}{h}\int_{\Xi} \rho_j(\hat{x},\hat{\xi};h)\,dP_{\boldsymbol{\xi}} \to 0 \text{ when } h \to 0, \tag{3.7}$$

then $\frac{\partial \mathcal{Q}(\hat{x},\xi)}{\partial x_j}$ exists and

$$\frac{\partial \mathcal{Q}(\hat{x})}{\partial x_j} = \int_{\Xi} \frac{\partial Q(\hat{x},\xi)}{\partial x_j} dP_{\boldsymbol{\xi}}.$$

Proof. Provided $Q(x,\xi)$ is partially differentiable at $\hat{x}$ for all $\xi \in \Xi$, we get

$$\begin{aligned}\frac{\mathcal{Q}(\hat{x}+he_j)-\mathcal{Q}(\hat{x})}{h} &= \int_{\Xi} \frac{Q(\hat{x}+he_j,\xi)-Q(\hat{x},\xi)}{h} dP \\ &= \int_{\Xi\setminus A} \frac{\partial Q(\hat{x},\xi)}{\partial x_j} dP + \int_{\Xi\setminus A} \frac{\rho_j(\hat{x},\xi;h)}{h} dP\end{aligned}$$

where $A \in \mathcal{F}$ and $P_{\boldsymbol{\xi}}[A] = 0$. □

While theoretically important, this theorem has little practical interest. It is often possible to guarantee that the recourse function is partially differentiable almost surely and the partial derivative is integrable, but the requirement (3.7) can be difficult to check, and in fact a two-stage stochastic program with recourse is often nonsmooth. To illustrate this difficulty, consider the classical two-stage linear problem:

$$\min_x c^T x + \mathcal{Q}(x), \tag{3.8}$$

$$\text{s.t. } Ax = b,\ x \geq 0, \tag{3.9}$$

where

$$\mathcal{Q}(x) = E_{\boldsymbol{\xi}}[Q(x, \boldsymbol{\xi})]$$

and

$$Q(x, \boldsymbol{\xi}) = \min_y \left\{ q(\boldsymbol{\xi})^T y \,|\, Wy = h(\boldsymbol{\xi}) - T(\boldsymbol{\xi})x,\ y \geq 0 \right\}. \tag{3.10}$$

If $\boldsymbol{\xi}$ is a continuous random variable, it is possible to show that conditions of Theorem 3.2 are satisfied, and therefore $\mathcal{Q}(x)$ is differentiable (see for instance Kall and Wallace [80], Section 1.4). However, if the support of $\boldsymbol{\xi}$ is discrete, $\mathcal{Q}(x)$ is no more everywhere differentiable This comes from the fact that, for some realization ξ, $Q(x, \xi)$ is a piecewise linear convex function in x (see Theorem 3.5, page 89, in Birge and Louveaux [21], and Example 5.1.1, page 159). Therefore, there exists a finite set of points where $Q(x, \xi)$ is not differentiable. The measure of this set is strictly positive if the distribution of $\boldsymbol{\xi}$ is discrete, and is equal to zero if the distribution is continuous. Thus $\mathcal{Q}(x)$ is differentiable when we face continuous random parameters, even if $Q(x, \boldsymbol{\xi})$ is not everywhere (but almost everywhere) differentiable with respect to x.

Since nondifferentiable problems are very difficult to manage, especially for nonlinear non-convex functions, a formulation that restores the differentiability property is appealing. Assume therefore that $\boldsymbol{\xi}$ is a discrete random variable with S possible realizations, that we denote by $s = 1, \ldots, S$ for simplicity. Let $p^{(s)}$ be the probability that $\xi = s$ ($s = 1, \ldots, S$). Practical methods for two-stage programs are then usually based on the extensive form, also called dual decomposition structure, of the stochastic program (3.8)–(3.10):

$$\begin{aligned}
&\min_x c^T x + \sum_{s=1}^{S} p^{(s)} q^{(s)T} y^{(s)} \\
&\text{s.t. } Ax = b, \\
&\quad T^{(s)} x + W y^{(s)} = h^{(s)},\ s = 1, \ldots, S, \\
&\quad x \geq 0,\ y^{(s)} \geq 0,\ s = 1, \ldots, S.
\end{aligned}$$

This form is obtained by defining second-stage vectors, one for each realization of $\boldsymbol{\xi}$, while maintaining only one variable for the first-stage. The resulting problem is then differentiable, and has a formulation similar to classical mathematical programs. A well-known algorithm for two-stage linear programs is the L-Shaped method (see again Birge and Louveaux [21], Chapter 5). More generally, if Ξ is finite, we will work with the extensive form of the stochastic program and make smoothness assumptions with respect to x and $y^{(s)}$, $s = 1, \ldots, S$.

Convexity and optimality

Most of the known properties in nonlinear stochastic programming are based on the assumption that the problem is convex. This is in particular the case for optimality properties. It is therefore useful to give some conditions that guarantee that a stochastic program is indeed convex, and that it is well-defined. Since we are primarily interested in nonconvex programs, we only state here state here the main results; more details can be found for instance in Birge and Louveaux [21], Section 3.4.

Return to the general form (3.6), namely

$$\min_{x \in X} z = f_1(x, \xi) + \mathcal{Q}(x).$$

We can state the following property.

Proposition 3.1 *If $f_1(x, \xi)$ and $Q(x, \xi)$ are convex in x for all $\xi \in \Xi$, and if X is a convex set, then (3.6) is a convex program.*

Consider now the formulation (3.4)–(3.5), that we restate for clarity:

$$\begin{aligned} \min_x z(x) &= f_1(x) + \mathcal{Q}(x) \\ \text{s.t. } & c_{1,i}(x) \leq 0,\ i = 1, \ldots, \overline{m}_1, \\ & c_{1,i}(x) = 0, i = \overline{m}_1 + 1, \ldots, m_1, \end{aligned} \tag{3.11}$$

where $\mathcal{Q}(x) = E_{\boldsymbol{\xi}}\left[Q(x, \xi)\right]$, and

$$\begin{aligned} Q(x, \xi) &= \min_y f_2(y(\xi), \xi) \\ \text{s.t. } & t_{2,i}(x, \xi) + c_{2,i}(y(\xi), \xi) \leq 0,\ i = 1, \ldots, \overline{m}_2, \\ & t_{2,i}(x, \xi) + c_{2,i}(y(\xi), \xi) = 0,\ i = \overline{m}_2 + 1, \ldots, m_2. \end{aligned} \tag{3.12}$$

From the previous proposition, this problem is convex if we assume that

$$\begin{gathered} f_1,\ c_{1,i}\ (i = 1, \ldots, \overline{m}_1) \text{ are convex on } \mathbb{R}^{n_1}, \\ c_{1,i}\ (i = \overline{m}_1 + 1, \ldots, m_1) \text{ are affine on } \mathbb{R}^{n_1}, \\ f_2(\cdot, \xi),\ c_{2,i}(\cdot, \xi)\ (i = 1, \ldots, \overline{m}_2) \text{ are convex on } \mathbb{R}^{n_2},\ \forall \xi \in \Xi, \\ c_{2,i}(\cdot, \xi)\ (i = \overline{m}_2 + 1, \ldots, m_2) \text{ are affine on } \mathbb{R}^{n_2},\ \forall \xi \in \Xi, \\ t_{2,i}(\cdot, \xi),\ (i = 1, \ldots, \overline{m}_2) \text{ are convex on } \mathbb{R}^{n_1},\ \forall \xi \in \Xi, \\ t_{2,i}(\cdot, \xi)\ (i = \overline{m}_2 + 1, \ldots, m_2) \text{ are affine on } \mathbb{R}^{n_1}\ \forall \xi \in \Xi. \end{gathered}$$

We also assume often that (3.12) satisfies the Slater's constraint qualification (see Definition 2.5) for the second-stage problem, ensuring that the second-stage problem is well defined and that a solution can be obtained by solving the associated KKT system.

A.1 *If $\mathcal{Q}(x) < \infty$, for almost all $\xi \in \Xi$, there exists some $y(\xi)$ such that $t_{2,i}(x, \xi) + c_i(y(\xi), \xi) < 0$ for $i = 1, \ldots, \overline{m}_2$ and $t_{2,i}(x, \xi) + c_{2,i}(y(\xi), \xi) = 0$ for $i = \overline{m_2} + 1, \ldots, m_2$.*

We can also obtain continuity of the recourse function if we assume that the recourse feasible region is bounded, as stated in the following theorem.

Theorem 3.3 *If the recourse feasible region is bounded for any $x \in \mathbb{R}^{n_1}$, then the function $Q(x, \xi)$ is lower semicontinuous in x for all $\xi \in \Xi$.*

Proof. See Birge and Louveaux [21], page 124. □

Under these assumptions, if for all $\xi \in \Xi$, $Q(x, \xi)$ is convex in x, $Q(x, \xi)$ is a closed convex function in x and the feasible set $K_2 = \{x \mid \mathcal{Q}(x) < \infty\}$ is closed and convex. The following result ensures the existence of a solution.

Theorem 3.4 *Assume that the TSSPR is convex, and that f_1 is continuous, $c_{1,i}$ and $c_{2,i}$ are continuous for each i. Suppose also that the recourse feasible region is bounded for any $x \in \mathbb{R}^{n_1}$, K_1 is bounded and $K_1 \cap K_2 \neq \emptyset$. Then* (3.4)–(3.5) *has a finite optimal solution and the infimum of $f_1(x) + \mathcal{Q}(x)$ is attained.*

From the general optimality conditions in nonlinear convex programming (see e.g. Bertsekas [14]), we can also state the following result:

Theorem 3.5 (Optimality conditions for TSSPR) *Consider a convex two-stage stochastic with recourse of the form* (3.11)-(3.12)*. Suppose that there exists some x^* such that $x^* \in ri(Dom(f_1(x))$ and $x^* \in ri(Dom(\mathcal{Q}(x)))$. Suppose also that x^* satisfies the Slater's constraint qualification for the first stage, i.e., $c_{1,i}(x^*) < 0$ for all $i = 1, \ldots, \overline{m}_1$ and $c_{1,i}(x^*) = 0$ for all $i = \overline{m}_1 + 1, \ldots, \overline{m}_1$. Then x^* is optimal in (3.4) if and only if $x^* \in K_1$ and there exists $\lambda_i^* \geq 0,\ i = 1, \ldots, \overline{m}_1$, λ_i^*, $i = \overline{m}_1 + 1, \ldots, m_1$, such that $\lambda_i^* c_{1,i}(x^*) = 0, i = 1, \ldots, \overline{m_1}$, and*

$$0 \in \partial_x f_1(x^*) + \partial_x \mathcal{Q}(x^*) + \sum_{i=1}^{m_1} \lambda_i^* \partial_x c_{1,i}(x^*). \tag{3.13}$$

For practical purposes, we can use the decomposition of $\partial\mathcal{Q}(x)$ into subgradients of the $Q(x, \xi)$ as below:

$$\partial\mathcal{Q}(x) = E_{\boldsymbol{\xi}} \left[\partial Q(x, \xi)\right] + \mathcal{N}_{K_2}(x). \tag{3.14}$$

If we have relatively complete recourse, we can remove the normal cone term $\mathcal{N}_{K_2}(x)$ in (3.14). We then have indeed that

$$\mathcal{N}_{K_2}(x) \subseteq \mathcal{N}_{K_1}(x),$$

and if $\eta \in \mathcal{N}_{K_1}(x)$, η has the form

$$\sum_{i=1}^{m_1} \lambda_i \partial_x c_{1,i}(x), \quad \lambda_i \geq 0,\ i = 1, \ldots, \overline{m_1},$$

so the condition (3.13) can be rewritten as

$$0 \in \partial_x f_1(x^*) + E_{\boldsymbol{\xi}}\left[\partial Q(x^*, \boldsymbol{\xi})\right] + \sum_{i=1}^{m_1} \lambda_i^* \partial_x c_{1,i}(x^*).$$

3.2.3 Multistage stochastic programming

In the previous section, we have studied problems for which only one recourse action is undertaken. However many operational and planning problems involve sequences of decisions that respond to realizations of outcomes that are not known a priori and evolve over time. Practical applications include amongst others portfolio problems (Klassen, Shapiro and Spitz [83]), optimal power dispatch (e.g. Dentecheva, Gollmer, Möller, Römisch and Schultz [43], Gröwe, Römisch and Schultz [67], Römisch and Schultz [117], Bastin [7]), forest planning (Gassmann [58]), agricultural management (Rosa and Yates [119]). It is therefore useful to borrow some terminology to discrete time stochastic process theory (McFadden [94]). The random parameter is described as a data process:

$$\xi \overset{def}{=} \{\xi_t : t = 0, \dots\}.$$

For simplicity, we introduce the history process

$$\overline{\xi}_t \overset{def}{=} \{\xi_0, \dots, \xi_t\}.$$

Let $T+1$ be the number of stages, also called the horizon. Instead of the two decisions x and y, we have now $T+1$ sequential decisions $x_0, x_1, \dots, x_T$, to be taken at the subsequent stages $t = 0, 1, \dots, T$. The term stages can, but need not, be interpreted as time period. More generally, the decisions are a process denoted by

$$\boldsymbol{x} \overset{def}{=} \{x_t : t = 0, \dots\},$$

such that x is a measurable function $x : \xi \to x(\xi)$. We then define the decision process, similarly to the history process, as

$$\overline{x}_t \overset{def}{=} \{x_0, \dots, x_t\}.$$

At stage t, we know $\overline{\xi}_t$ as well as $\overline{x}_{t-1}$. We have to decide on x_t such that the constraints are satisfied. Assume that on the first step, the objective is deterministic (i.e. there is no random variable $\boldsymbol{\xi}_0$). The program can then be expressed as

$$\min_{x_0} f_0(x_0) + E_{\boldsymbol{\xi}_1}\Big[\min_{x_1} f_1\left(\overline{\boldsymbol{\xi}}_1, x_0, x_1(\boldsymbol{\xi}_1)\right) + E_{\boldsymbol{\xi}_2|\boldsymbol{\xi}_1}\Big[\min_{x_2} f_2\left(\overline{\boldsymbol{\xi}}_2, \overline{x}_1, x_2(\boldsymbol{\xi}_2)\right) + \ldots + E_{\boldsymbol{\xi}_T|\boldsymbol{\xi}_1,\dots,\boldsymbol{\xi}_{T-1}}\Big[\min_{x_T} f_T\left(\overline{\boldsymbol{\xi}}_T, \overline{x}_{T-1}, x_T(\boldsymbol{\xi}_T)\right)\Big]\Big]\Big] \quad (3.15)$$

such that

$$\begin{array}{lll} c_{i,0}(x_0) \le 0, & i = 1, \dots, m_0, & \\ c_{t,i}(\overline{\xi}_t, \overline{x}_{t-1}(\overline{\xi}_{t-1}), x_t(\xi_t)) \le 0 \text{ a.s.}, & i = 1, \dots, m_t, \ t = 1, \dots, T, & (3.16) \\ x_t(\xi_t) \in X_t, & t = 0, \dots, T. & \end{array}$$

From the additive property of the expectation, we can rewrite (3.15)–(3.16) as

$$\min_{x_0 \in X_0} f_0(x_0) + \sum_{t=1}^{T} E_{\boldsymbol{\xi}_1,\ldots,\boldsymbol{\xi}_t} \left[Q_t \left(\overline{\boldsymbol{\xi}}_t, \overline{x}_{t-1}, x_t(\boldsymbol{\xi}_t) \right) \right], \tag{3.17}$$

where

$$X_0 = \{c_{i,0}(x) \leq 0,\ i = 1, \ldots, m_0\},$$
$$Q_t = \inf_{x_t(\xi_t) \in X_t} \{ f_t(\overline{\xi}_t, \overline{x}_{t-1}, x_t(\xi_t)) \mid c_{t,i}(\overline{\xi}_t, \overline{x}_{t-1}(\overline{\xi}_{t-1}), x_t(\xi_t)) \leq 0 \text{ a.s.},\ i = 1, \ldots m_t \}.$$

3.2.4 Scenario approach

The expectations in (3.17) involve multidimensional integration of implicitly defined functions. To avoid this difficulty, the probability spaces are generally assumed to be discrete. This allows the objective to be written as a finite sum and the constraints to be enforced by replicating them for each element in Ξ_t. Assume thus that at each period t, $\boldsymbol{\xi}_t$ has a discrete probability distribution over Ξ. We can consider the set of possible future sequences of outcomes, which is finite; such a sequence is called a scenario, which we identify with the superscript (s).

Definition 3.2: Scenario
A scenario s is a sequence of outcomes ξ_t, $t = 0, \ldots, T$, denoted by

$$s = \begin{pmatrix} \xi_0^{(s)} \\ \xi_1^{(s)} \\ \vdots \\ \xi_T^{(s)} \end{pmatrix}.$$

The number of scenarios is denoted by S.

The scenario concept allows us to represent the possible outcomes by means of a graph called tree of scenarios because of its aspect. An example of such a graph is presented in Figure 3.1. The root of the tree correspond to the first-stage decision, made before any realization of the random parameters, while each realization of $\boldsymbol{\xi}_{t+1}$, conditionally to $\overline{\boldsymbol{\xi}}_t$, is associated with a node.

It is sometimes convenient to develop some terminology associated to such a tree. Let V be a tree of scenarios and v be one of its nodes, also denoted by v_t to express that v appears at stage t ($t \in \{0, \ldots, T\}$). We adopt the following definitions, borrowed from graph theory. $D(v)$ is the set of descendants of v in V, and $A(v)$, the set of its ancestors. We let $\mathcal{S}(v)$ and $\mathcal{P}(v)$ be respectively the set of direct successors of v and its predecessor. The root of the tree is noted r and the set of the terminal nodes, also called leaves, L. $d(v)$ represent the depth of v in V, with $d(r) = 0$ (so $d(v_t) = t$). The partial subtree of V rooted in some node v, containing all descendants of v in V down to level k, is designed by $V_k(v)$, and $L_k(v)$ is the set of its terminal nodes.

Each node v is associated to a (conditional) probability $\pi(v)$ to be chosen at the stage $d(v)$, if its predecessor has been selected at the stage $d(v) - 1$, with $\pi(r) = 1$. If the $\boldsymbol{\xi}_t$'s, $t = 0, \ldots, T$,

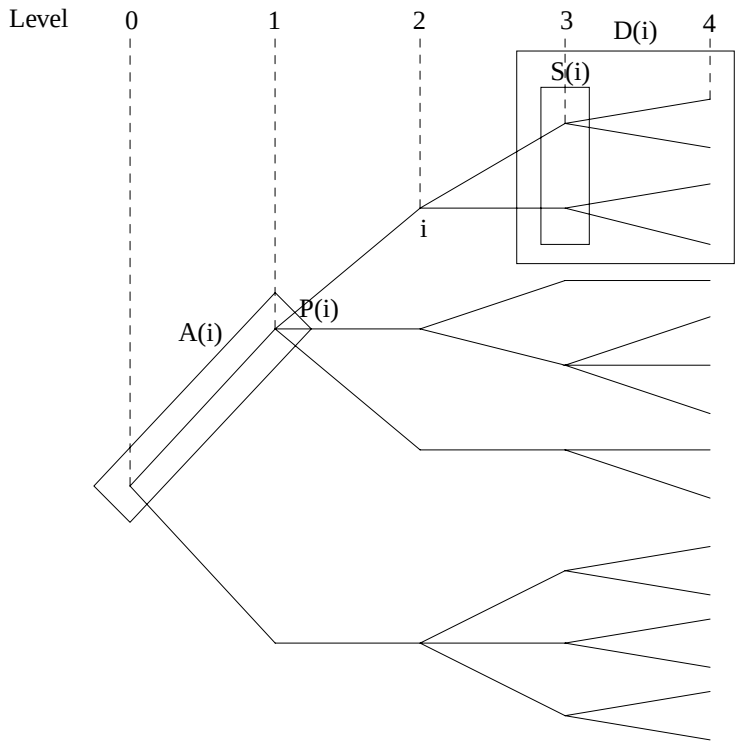


Figure 3.1: Tree of scenarios

are independent, it follows that the probability of scenario s is

$$p^{(s)} = \prod_{t=0}^{T} \pi\left(v_t^{(s)}\right).$$

The (unconditional) probability of the path leading to a node v

$$\mu(v) = \pi(v) \prod_{w \in A(v)} \pi(w),$$

Let $x(v)$ be the decision vector associated to node $v \in V$, and $\overline{x}(v) = \{x(w) \mid w \in A(v) \cup v\}$ be the decisions made at previous stages. If the objective and constraints are separable between the different stages, the multistage stochastic program (with recourse) can be decomposed following the nodes, and expressed as:

$$\sum_{v \in V} \mu(v) f_v(\overline{\xi}_{d(v)}, \overline{x}_v, x_v), \tag{3.18}$$

such that

$$\begin{aligned} c_i(\overline{\xi}_{d(v)}, \overline{x}_v, x_v) &= 0,\ i \in \mathcal{E}_v, \quad \text{for all } v \in V, \\ c_i(\overline{\xi}_{d(v)}, \overline{x}_v, x_v) &\leq 0,\ i \in \mathcal{I}_v, \quad \text{for all } v \in V. \end{aligned} \tag{3.19}$$

Note that the nonanticipativity constraints are implicitly included in (3.18)–(3.19).

3.2.5 Split-variable formulation

The stochastic program is often rewritten with the split-variable formulation, introduced by Rockafellar and Wets [115] in late 80s, and used for instance in Nielsen [103] and Ruszczyński [121]. The split-variable formulation exhibits the similarities of subproblems corresponding to the different scenarios by associating with each scenario s ($s = 1, \ldots, S$) and each stage t ($t = 0, \ldots, T$) a set of decision variables, $x_t^{(s)} \in \mathbb{R}^{n_t}$, which replace the stochastic variables x_t.

In order to make the presentation clearer, we return briefly to a two-stage stochastic program with recourse, of the form (3.4)-(3.5). Assume furthermore that Ξ is finite and write it as $\Xi = \{1, \ldots, S\}$, with $P_{\boldsymbol{\xi}}[\xi = s] = p_s$, $s = 1, \ldots, S$. Then (3.4)-(3.5) can be rewritten as

$$\min_{(x^{(1)},\ldots,x^{(S)})} \sum_{s=1}^{S} p_s \left(f_1\left(x^{(s)}\right) + Q\left(x^{(s)}, s\right)\right) \tag{3.20}$$

$$\text{s.t. } c_{1,i}\left(x^{(s)}\right) \leq 0,\ i = 1, \ldots, \overline{m}_1,\ s = 1, \ldots, S, \tag{3.21}$$

$$c_{1,i}\left(x^{(s)}\right) = 0,\ i = \overline{m}_1 + 1, \ldots, m_1,\ s = 1, \ldots, S, \tag{3.22}$$

$$x^{(s)} - z = 0,\ s = 1, \ldots, S, \tag{3.23}$$

where

$$Q(x^{(s)}, s) = \min f_2\left(y^{(s)}, s\right) \tag{3.24}$$

$$\text{s.t. } t_{2,i}\left(x^{(s)}\right) + c_{2,i}\left(y^{(s)}, s\right) \leq 0,\ i = 1, \ldots, \overline{m}_2, \tag{3.25}$$

$$t_{2,i}\left(x^{(s)}\right) + c_{2,i}\left(y^{(s)}, s\right) = 0,\ i = \overline{m}_2 + 1, \ldots, m_2. \tag{3.26}$$

(3.23) reflects the nonanticipativity conditions linking the scenarios together. If not for the common constraints $x^{(s)} - z = 0$, each scenario could be solved independently without any trouble. These constraints are referred as the nonanticipativity constraints.

If Ξ is infinite, the objective in (3.20) has to be rewritten as

$$\min_{x(\xi),\ y(\xi)} \int_{\Xi} f_1(x(\xi)) + f_2(y(\xi), \xi) d\mu \tag{3.27}$$

$$\text{s.t. } c_{1,i}(x(\xi)) \leq 0 \text{ a.s.},\ i = 1, \ldots, \overline{m}_1 \tag{3.28}$$

$$c_{1,i}(x(\xi)) = 0 \text{ a.s.},\ i = \overline{m}_1 + 1, \ldots, m_1, \tag{3.29}$$

$$E_{\boldsymbol{\xi}}[x(\boldsymbol{\xi})] - x(\xi) = 0,\ \text{a.s.} \tag{3.30}$$

$$t_{2,i}(x, \xi) + c_{2,i}(y(\xi), \xi) \leq 0 \text{ a.s.},\ i = 1, \ldots, \overline{m}_2, \tag{3.31}$$

$$t_{2,i}(x, \xi) + c_{2,i}(y(\xi), \xi) = 0 \text{ a.s.},\ i = \overline{m}_2 + 1, \ldots, m_2. \tag{3.32}$$

The nonanticipativity constraints (3.30) then imply that almost all $x(\xi)$ values are the same. In order to be able to solve (3.27)–(3.32) we can discretize this program by using Monte Carlo approximations. To date such discretization process is subject to extensive research in stochastic programming (see Chapter 5 for Monte Carlo samplings in nonlinear stochastic programming and Shapiro [128] for a review of currently known properties in the context of multistage problems).

We return to the finite case and, for a multistage stochastic program, denote $x^{(s)}$ the vector of decisions at the successive stages, associated to the scenario s, $s = 1, \ldots, S$:

$$x^{(s)} = \begin{pmatrix} x_0^{(s)} \\ x_1^{(s)} \\ \vdots \\ x_T^{(s)} \end{pmatrix}.$$

Let p_s be the probability of the scenario s, $s = 1, \ldots, S$. We then aim at solving the program

$$\min_{x_0} \sum_{s=1}^{S} p_s \left(f_0(x_0^{(s)}) + \sum_{t=1}^{T} f_t(\overline{x}_{t-1}^{(s)}, x_t^{(s)}, s) \right) \tag{3.33}$$

subject to

$$c_{0,i}(x_0^{(s)}) \leq 0, \quad i = 1, \ldots, m_0, \ s = 1, \ldots, S, \tag{3.34}$$

$$c_{t,i}\left(\overline{x}_{t-1}^{(s)}, x_t^{(s)}, s\right) \leq 0, \quad i = 1, \ldots, m_t, \ t = 1, \ldots, T, \ s = 1, \ldots, S, \tag{3.35}$$

$$x_t^{(s)} \in X_t, \quad t = 0, \ldots, T, \ s = 1, \ldots, S, \tag{3.36}$$

$$x_t^{(s)} \text{ is nonanticipative}, \quad t = 0, \ldots, T, \ s = 1, \ldots, S. \tag{3.37}$$

The nonanticipativity constraints can be expressed in various ways. For instance let $\mathcal{C}_t$ be the set of scenarios with the same history at stage t. The condition (3.37) can be rewritten as

$$\overline{x}_t^{(s)} = \overline{x}_t^{(u)}, \text{ for any } s, u \in \mathcal{C}_t. \tag{3.38}$$

In order to make (3.38) clear, let $k^{(s)}$, $s = 1, \ldots, S-1$, be the stage number up to which scenarios s and $s+1$ share the same history. This number may possibly take values from 0 (only initial decisions $x_0^{(s)}$ and $x_0^{(s+1)}$ are the same in scenario s and scenario $s+1$) to $T-1$ (scenarios s and $s+1$ have the same history up to last but one stage). The nonanticipativity constraints can then be expressed in a mathematical way as follows:

$$x_k^{(s)} = x_k^{(s+1)}, \ s = 1, \ldots, S-1, \ k = 0, \ldots, k^{(s)}. \tag{3.39}$$

We can summarize (3.39) by a constraint of the form

$$Nx = 0,$$

where N is a full-row rank matrix called the nonanticipativity matrix: writing

$$I^{(s)} = \begin{pmatrix} I_0 & & \\ & \ddots & \\ & & I_{k^{(s)}} \end{pmatrix}, \ s = 1, \ldots, S-1,$$

constraints (3.39) can be organized to give the linear system

$$\begin{pmatrix} I^{(1)} & 0 & -I^{(1)} & 0 & & & & & \\ & & I^{(2)} & 0 & -I^{(2)} & 0 & & & \\ & & & & \ddots & \ddots & & & \\ & & & & & I^{(S-1)} & 0 & -I^{(S-1)} & 0 \end{pmatrix} \begin{pmatrix} x_0^{(1)} \\ \vdots \\ x_T^{(1)} \\ x_0^{(2)} \\ \vdots \\ x_T^{(2)} \\ \vdots \\ x_0^{(S)} \\ \vdots \\ x_T^{(S)} \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 0 \\ \vdots \\ 0 \\ \vdots \\ 0 \\ \vdots \\ 0 \end{pmatrix},$$

where the sizes of the matrices $I^{(s)}$, $s = 1, \ldots, S$, are $k^{(s)}$, $s = 1, \ldots, S$. The left hand-side matrix is clearly of full row rank.

Example 3.1. *Consider a 3-stage ($T = 2$), 6-scenario problem ($S = 6$), with the following nonanticipativity constraints:*

$$x_0^{(1)} = x_0^{(2)} = x_0^{(3)} = x_0^{(4)} = x_0^{(5)} = x_0^{(6)}, \tag{3.40}$$
$$x_1^{(1)} = x_1^{(2)} = x_1^{(3)}, \tag{3.41}$$
$$x_1^{(5)} = x_1^{(6)}. \tag{3.42}$$

In this case, we have $k^{(1)} = 1$, $k^{(2)} = 1$, $k^{(3)} = 0$, $k^{(4)} = 0$ and $k^{(5)} = 1$. The corresponding tree of scenarios is illustrated in Figure 3.2.

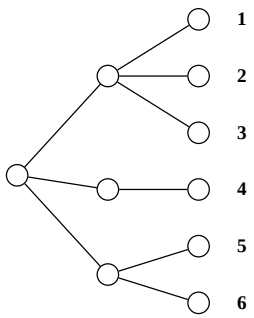


Figure 3.2: Tree of scenarios.

The constraints matrix N corresponding to (3.40)–(3.42) *can be expressed as follows:*

$$
\begin{array}{c}
\\ n_0 \\ n_1 \\ n_0 \\ n_1 \\ n_0 \\ n_0 \\ n_0 \\ n_1
\end{array}
\begin{array}{c}
\begin{array}{cccccccccccccccccc}
n_0 & n_1 & n_2 & n_0 & n_1 & n_2 & n_0 & n_1 & n_2 & n_0 & n_1 & n_2 & n_0 & n_1 & n_2 & n_0 & n_1 & n_2
\end{array} \\
\left(\begin{array}{cccccccccccccccccc}
I_0 & 0 & 0 & -I_0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & I_1 & 0 & 0 & -I_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & I_0 & 0 & 0 & -I_0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & I_1 & 0 & 0 & -I_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & I_0 & 0 & 0 & -I_0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & I_0 & 0 & 0 & -I_0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & I_0 & 0 & 0 & -I_0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & I_1 & 0 & 0 & -I_1 & 0
\end{array}\right)
\end{array},
$$

where $I_i, i = 0, \ldots, T-1$, is the identity matrix of dimension $n_i \times n_i$, and n_i is the size of vector $x_i^{(s)}$ ($s \in \{1, \ldots, S\}$). In others terms, n_i is the number of variables that belong to the i^{th} stage, for one scenario. This matrix forms the nonanticipativity matrix N for the particular example (3.40)–(3.42).

3.2.6 Extensive form versus split-variable formulation

The main drawback to scenario formulation is the multiplication of the number of variables. We could therefore prefer to use extensive forms, which limit the multiplication of variables, while ensuring smoothness properties (see Section 3.2.2). Consider a tree of scenarios V. A convenient expression is then the formulation (3.18)–(3.19), where a decision vector x_v is associated at each node v of the tree of scenarios:

$$\sum_{v \in V} \mu(v) f_v(\overline{\xi}_{d(v)}, \overline{x}_v, x_v),$$

such that

$$
\begin{aligned}
c_i(\overline{\xi}_{d(v)}, \overline{x}_v, x_v) &= 0, \ i \in \mathcal{E}_v, \quad \text{for all } v \in V, \\
c_i(\overline{\xi}_{d(v)}, \overline{x}_v, x_v) &\leq 0, \ i \in \mathcal{I}_v, \quad \text{for all } v \in V.
\end{aligned}
$$

The split-variable formulation then consists to define a decision vector $x_{d(v)}^{(s)}$ associated to each scenario visiting the node v, instead of one vector x_v, and to link the variables $x_{d(v)}^{(s)}$ with explicit nonanticipativity constraints. The procedure is summarized in Figure 3.3.

The split-variable formulation is attractive for any algorithmic framework where the nonanticipativity constraints can be temporarily ignored, so the S scenario-based subproblems appearing in (3.33)–(3.37) become independent and can be solved in parallel, otherwise the extensive form is preferable, specially when the problem presents an exploitable structure (see for instance Steinbach [130]). Consider for instance a two-stage problem where we have one first-stage variable and one second-stage variable, and S scenarios. The extensive form will therefore present $S + 1$ variables, while the split-variable formulation will have $2S$ variables. In an ideal context, we could then solve S subproblems in parallel, whose solutions have to be recombined to take into account the nonanticipativity constraints, while the number of variables has only been multiplied by approximately 2. One of the most popular algorithm for split-variables formulations of convex stochastic programs is the progressive hedging algorithm, proposed by

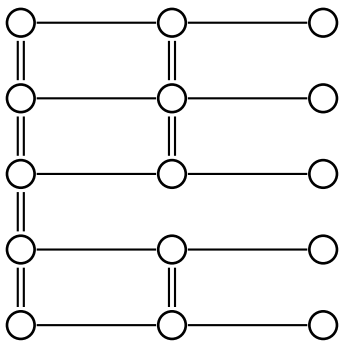


Figure 3.3: Variables splitting

Rockafellar and Wets [115]. Other methods have also been developed, for instance by Rosa and Ruszczyński [118]. These authors use an augmented Lagrangian strategy and propose a nodal decomposition based on the extensive form and a scenario decomposition based on the split-variable formulation. They then report better results with the scenario decomposition; in particular, the scenario decomposition is shown to less depend on the considered application that the nodal decomposition.

3.3 Other developments in stochastic programming

Other formulations are popular in stochastic programming, that we briefly mention here, while our subsequent developments will not rely on such approaches. In particular, chance-constrained programming, also called probabilistic programming, is another way to give some meaning to the minimum operator in the problem (1.5). In this case, the constraints are assumed to hold with some probabilities, and are called probabilistic constraints. They can be expressed as follows:

$$P_{\boldsymbol{\xi}}[\{\xi \mid \phi_i(x,\xi) \leq 0\}] \geq \alpha_i,\ i = 1,\ldots,m. \tag{3.43}$$

where $\alpha_i \in [0,1]$ and $\phi_i : \mathbb{R}^n \times \Xi \to \mathbb{R}$ $(i = 1,\ldots,m)$. We can also require that all constraints jointly hold with some probability α, i.e.

$$P_{\boldsymbol{\xi}}[\{\xi \mid \phi_i(x,\xi) \leq 0,\ i = 1,\ldots,m\}] \geq \alpha,$$

where $\phi(x,\xi) = (\phi_1(x,\xi),\ldots,\phi_m(x,\xi))^T$. In this case, the problem is said to have joint probabilistic constraints, while if the constraints are expressed as in (3.43), the constraints are called single (or separate) probabilistic constraints.

The chance constrained programming was first developed by Charnes and Cooper [25] in 1959, and have received a lot of attention since, especially by Prekopa [108, 109, 110, 111]. Chance-constrained programs can be derived from the general program (3.1)–(3.3), as shown for instance in Kall and Wallace [80], Section 1.3. For more properties of chance-constrained programs, we refer to our master's thesis [7] and the books of Birge and Louveaux [21] and Kall and Wallace [80]; some applications can also be found in Dempster [41].

We conclude this chapter by remarking that, due to the complexity of stochastic programs, we could be inclined to solve simpler problems when we are confronted by real-world applications. In Chapter 6, Section 6.2.5, we will see that multinomial logit models can be viewed as mixed logit models where the random variables are replaced by their expectation. More generally, the technique resulting from the replacement of all random variables by they expected values is know as the "expected value method" in stochastic optimization, and the related optimization problem is called the expected value problem or mean value problem. Such simplifications, while producing more manageable problems, can lead to poor solutions. The pertinence of the expected value method and that of the stochastic programming approach are usually evaluated by means of two theoretical concepts: the expected value of perfect information and the value of stochastic information. We refer the interested reader to Birge and Louveaux [21], Chapter 4, for more details on these questions.

Part II

Trust-region methods for nonlinear stochastic programming

Chapter 4

Interior point methods for scenario formulations

Having reviewed several stochastic programming formulations in the previous chapter, we now focus on a special case of the multistage stochastic program (3.15)–(3.16), where the objective is assumed to be nonlinear. As shown before, the scenarios approach suggests developing special techniques that allow parallelization of the algorithmic procedure. In particular, decomposition techniques have been previously successfully applied in linear stochastic programming (see for instance Berkelaar and al. [13] for two-stage programming, and Liu and Sun [86] for multistage programming), and split-variable formulation has led to popular algorithms as the progressive hedging algorithms (Rockafellar and Wets [115]) in convex stochastic programming. The nonconvex case has however received much less attention. One of the few attempts can be found in Liu and Zhao [87] who propose a sequential quadratic programming approach.

In this chapter, we examine how to benefit from the structure of nonlinear stochastic programs with scenarios in the context of interior-point techniques (see Section 2.4.2), in particular in the computation of a quasi-Newton step for the constrained barrier subproblem. This step is combined with the projected constrained steepest descent direction to form a dogleg path, that allows to approximately minimize some model of the barrier subproblem, similarly to the method proposed by Zhang and Xu [143]. The agreement between the model reduction and the function reduction is controlled by a trust-region mechanism, as proposed in Conn, Gould, Orban, and Toint [30]. The primal-dual systems involved in the quasi-Newton steps computations are solved with a new decomposition technique based on scenario analysis and split-variable formulation, that can also be employed for the computation of a feasible starting point. This decomposition technique has been developed conjointly with Annick Sartenaer (University of Namur) and Jie Sun (University of Singapore).

4.1 Problem formulation

We consider the multistage program (3.15)–(3.16) with $(T+1)$-stage, where only linear equality and nonnegativity constraints are present:

$$\min_{y_0} q_0(y_0) + \sum_{t=1}^{T} E_{\boldsymbol{\xi}_1,\ldots,\boldsymbol{\xi}_t}[Q_t(y_0, y_1, \ldots, y_{t-1}, \boldsymbol{\xi}_1, \ldots, \boldsymbol{\xi}_t)], \tag{4.1}$$

$$\text{s.t. } \hat{A}y_0 = \hat{b},\ y_0 \geq 0, \tag{4.2}$$

where, for $t = 1, \ldots, T$,

$$Q_t(y_0, y_1, \ldots, y_{t-1}, \boldsymbol{\xi}_1, \ldots, \boldsymbol{\xi}_t) = \min_{y_t} q_t(y_0, \ldots, y_{t-1}, y_t, \boldsymbol{\xi}_1, \ldots, \boldsymbol{\xi}_t) \tag{4.3}$$

$$\text{s.t. } \sum_{k=0}^{t} U_{t,k}(\boldsymbol{\xi}_1, \ldots, \boldsymbol{\xi}_t)\, y_k = h_t(\boldsymbol{\xi}_1, \ldots, \boldsymbol{\xi}_t),\ y_t \geq 0, \tag{4.4}$$

and $y_t \in \mathbb{R}^{n_t}$, $t = 0, \ldots, T$.

In the above equations (4.1)–(4.4), $\boldsymbol{\xi}_t$, $t = 1, \ldots, T$, is a random vector associated with stage t, while $\xi_t \in \mathbb{R}^{m_t}$ is a realization of $\boldsymbol{\xi}_t$. The functions q_t, $t = 0, \ldots, T$, are real-valued smooth nonlinear functions. $\hat{A}$ is a fixed matrix defining linear constraints for the first-stage decision y_0, with right-hand side denoted by $\hat{b}$. $U_{t,k}(\boldsymbol{\xi}_1, \ldots, \boldsymbol{\xi}_t)$, $t = 1, \ldots, T$, $k = 0, \ldots, t$, are random matrices and $h_t(\boldsymbol{\xi}_1, \ldots, \boldsymbol{\xi}_t)$, $t = 1, \ldots, T$, are random vectors, all of them being decided by the realizations $\xi_1, \ldots, \xi_t$. The decision at stage t, $t = 0, \ldots, T$, is represented by the vector y_t.

Let $\boldsymbol{\xi} = (\boldsymbol{\xi}_1; \boldsymbol{\xi}_2; \ldots; \boldsymbol{\xi}_T)$ and $(\Xi, \mathcal{A}, P)$ be the associated probability space, and suppose that we have S scenarios $\xi^{(s)} = \left(\xi_1^{(s)}; \xi_2^{(s)}; \ldots; \xi_T^{(s)}\right)$, $s = 1, \ldots, S$, with a fixed known probability distribution $\left\{(\xi^{(s)}, p^{(s)}), s = 1, \ldots, S\right\}$. We choose to use the split-variable formulation (see Section 3.2.5): the vectors of variables y_t, $t = 0, \ldots, T$, are duplicated for each scenario, yielding the vectors $y_t^{(s)}$, $t = 0, \ldots, T$, $s = 1, \ldots, S$. We introduce the following notation, for $s = 1, \ldots, S$:

$$z^{(s)} = \left(y_0^{(s)}, \ldots, y_T^{(s)}\right)^T \in \mathbb{R}^n,\ n = \sum_{i=0}^{T} n_i,$$

$$f^{(s)}\left(z^{(s)}\right) = q_0\left(y_0^{(s)}\right) + \sum_{t=1}^{T} q_t\left(y_0^{(s)}, y_1^{(s)}, \ldots, y_t^{(s)}, \xi_1^{(s)}, \ldots, \xi_t^{(s)}\right),$$

$$B^{(s)} = \begin{pmatrix} \hat{A} & & & \\ U_{1,0}\left(\xi_1^{(s)}\right) & U_{1,1}\left(\xi_1^{(s)}\right) & & \\ \vdots & & \ddots & \\ U_{T,0}\left(\xi_1^{(s)}, \ldots, \xi_T^{(s)}\right) & U_{T,1}\left(\xi_1^{(s)}, \ldots, \xi_T^{(s)}\right) & \cdots & U_{T,T}\left(\xi_1^{(s)}, \ldots, \xi_T^{(s)}\right) \end{pmatrix},$$

and

$$b^{(s)} = \begin{pmatrix} \hat{b} \\ h_1\left(\xi_1^{(s)}\right) \\ \vdots \\ h_T\left(\xi_1^{(s)}, \ldots, \xi_T^{(s)}\right) \end{pmatrix}.$$

The vector $z^{(s)}$ corresponds to the decisions associated with the s^{th}-scenario. The function $f^{(s)}$ is the general objective for scenario s, taking all the stages into account. The matrix $B^{(s)}$ gathers the left-hand side coefficients of the linear constraints in (4.2) and (4.4), while $b^{(s)}$ collects the right-hand side coefficients of those constraints, still for scenario s.

The stochastic program (4.1)–(4.4) can then be reformulated as follows:

$$\min_z \sum_{s=1}^{S} p^{(s)} f^{(s)}\left(z^{(s)}\right) \tag{4.5}$$

$$\text{s.t. } B^{(s)} z^{(s)} = b^{(s)},\ s = 1, \ldots, S, \tag{4.6}$$

$$z^{(s)} \geq 0,\ s = 1, \ldots, S, \tag{4.7}$$

$$Nz = 0, \tag{4.8}$$

where $z = \left(z^{(1)}; z^{(2)}; \ldots; z^{(S)}\right) \in \mathbb{R}^{nS}$, and N in (4.8) is the banded block nonanticipativity matrix, as expressed in Section 3.2.5.

4.2 Notation and preliminary assumptions

Let us introduce the following notation:

$$f(z) = \sum_{s=1}^{S} p^{(s)} f^{(s)}\left(z^{(s)}\right).$$

Due to the separability nature of $f(z)$, its gradient, denoted by $g(z)$, can be expressed as follows:

$$g(z) \stackrel{def}{=} \nabla_z f(z) = \begin{pmatrix} p^{(1)} g^{(1)}\left(z^{(1)}\right) \\ \vdots \\ p^{(S)} g^{(S)}\left(z^{(S)}\right) \end{pmatrix},$$

where $g^{(i)}(z^{(i)}) \stackrel{def}{=} \nabla_{z^{(i)}} f^{(i)}\left(z^{(i)}\right)$, $i = 1, \ldots, S$. Similarly, its Hessian can be written as

$$\nabla^2_{zz} f(z) = \begin{pmatrix} p^{(1)} \nabla^2_{z^{(1)} z^{(1)}} f^{(1)}\left(z^{(1)}\right) & & \\ & \ddots & \\ & & p^{(S)} \nabla^2_{z^{(S)} z^{(S)}} f^{(S)}\left(z^{(S)}\right) \end{pmatrix}.$$

The constraints (4.6) can also be rewritten as

$$Bz = b,$$

where

$$B = \begin{pmatrix} B^{(1)} & & & \\ & B^{(2)} & & \\ & & \ddots & \\ & & & B^{(S)} \end{pmatrix} \quad \text{and} \quad b = \begin{pmatrix} b^{(1)} \\ b^{(2)} \\ \vdots \\ b^{(S)} \end{pmatrix}.$$

Using these notations we can rewrite problem (4.5)–(4.8) as the general nonlinear program with linear and nonnegativity constraints

$$\min f(z) \tag{4.9}$$
$$\text{s.t. } Bz = b, \tag{4.10}$$
$$Nz = 0, \tag{4.11}$$
$$z \geq 0. \tag{4.12}$$

We now state our first assumptions on the problem. Let $\mathcal{P} = \{z | z \geq 0\}$ be the set of points satisfying the nonnegativity constraints, $\mathcal{L} = \{z | Bz = b,\ Nz = 0\}$ be the set of points satisfying the linear equality constraints as well as the nonanticipativity constraints, so that the intersection $\mathcal{F} \stackrel{\text{def}}{=} \mathcal{P} \cap \mathcal{L}$ is the feasible set of problem (4.5)–(4.8). We denote $\mathcal{N}(A)$ the null space of the matrix A, as in Chapter 1. Recall also that the relative boundary of a set X is denoted by ∂X, so $\partial P = \{z \,|\, \exists i \text{ such that } [z]_i = 0\}$ and that e is the vector with all components equal to one. We make the following assumptions:

A.2 *The function $f(\cdot)$ is twice continuously differentiable in its argument over some open set containing $\mathcal{F}$.*

A.3 *The matrix B has full row rank.*

A.4 *The log-barrier function $f(z) - \mu\langle e, log(z)\rangle$ is bounded below on $\mathcal{F}$ for every $\mu > 0$.*

A.5 *There exists an $z_0 \in \mathbb{R}^{nS}$, such that $z_0 > 0$, $Nz_0 = 0$ and $Bz_0 = b$.*

Assumption A.2 simply ensures that $f(z)$ is well behaved in the region of interest, while A.3 states that there is no redundant or incompatible constraints in B. Assumption A.4 is intended to rule out functions which grow more slowly at infinity, if included in the feasible set, than the log function. For such functions, the logarithmic barrier approach is unlikely to succeed as the global minimizer of the barrier function is unbounded. Finally Assumption A.5 specifies that the feasible set $\mathcal{F}$ is nonempty and that at least one feasible point belongs to the relative interior of $\mathcal{P}$, which is a natural assumption in interior-point methods.

4.2.1 Preprocessing: full row rank reduction

Assumption A.3 does not ensure that the coefficient matrix $\begin{pmatrix} B^T & N^T \end{pmatrix}^T$ of the linear constraints (4.10)–(4.11) is of full row rank. Such a property is however desirable when designing

an interior point method, but this requirement is not necessarily fulfilled, even if both B and N are of full row rank, as illustrated by the following example, due to Liu and Sun [86].

Example 4.1. *Consider a simple 3-stage stochastic program with linear constraints*

$$\begin{aligned} [y_0]_1 - [y_0]_2 &= 1, \\ 2[y_0]_1 + [y_0]_2 &= \xi_1, \\ 2[y_1]_1 - [y_2]_1 &= \xi_2. \end{aligned}$$

We then have

$$B^{(s)} = \begin{pmatrix} 1 & -1 & 0 & 0 \\ 2 & 1 & -1 & 0 \\ 0 & 0 & 2 & -1 \end{pmatrix}, \quad s = 1, \ldots, S.$$

Suppose that $\boldsymbol{\xi}_1$ and $\boldsymbol{\xi}_2$ have two realizations respectively, thus $S = 4$. The matrix N is therefore defined by

$$N = \begin{pmatrix} 1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & & & & & & & & \\ 0 & 1 & 0 & 0 & 0 & -1 & 0 & 0 & & & & & & & & \\ 0 & 0 & 1 & 0 & 0 & 0 & -1 & 0 & & & & & & & & \\ & & & & 1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & & & & \\ & & & & 0 & 1 & 0 & 0 & 0 & -1 & 0 & 0 & & & & \\ & & & & & & & & 1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ & & & & & & & & 0 & 1 & 0 & 0 & 0 & -1 & 0 & 0 \\ & & & & & & & & 0 & 0 & 1 & 0 & 0 & 0 & -1 & 0 \end{pmatrix}.$$

Hence N is a 8×16 matrix and B is a 12×16 matrix. Thus, $\begin{pmatrix} B^T & N^T \end{pmatrix}^T$ is not of full row rank.

Under Assumption A.5, the problem is consistent. Therefore, if $\begin{pmatrix} B^T & N^T \end{pmatrix}^T$ is not of full row rank, some of the constraints in problem (4.5)–(4.8) are redundant and can be suppressed. Under Assumption A.3, B is of full row rank so we can proceed by eliminating only some of the nonanticipativity constraints. Liu and Sun [86] have developed a procedure that can be applied scenario by scenario. We propose here another approach, that we have developed for our specific case; we have also tested it by modifying the routines `dgeqr` and `dgeqpf` of CLAPACK, version 3.0 (Anderson et al. [3]). While it is numerically more expensive, it gives sufficient information to construct a projection matrix to the null space of the constraints by exploiting the information collected during the preprocessing (we will make use of this projection matrix later

on). The matrix $\begin{pmatrix} B^T & N^T \end{pmatrix}^T$ can be expanded as follows:

$$\begin{pmatrix}
B^{(1)} & & & & & & & & \\
& B^{(2)} & & & & & & & \\
& & B^{(3)} & & & & & & \\
& & & \ddots & & & & & \\
& & & & B^{(S-1)} & & & & \\
& & & & & & B^{(S)} & & \\
\begin{bmatrix} I^{(1)} & 0 \end{bmatrix} & \begin{bmatrix} -I^{(1)} & 0 \end{bmatrix} & & & & & & & \\
& \begin{bmatrix} I^{(2)} & 0 \end{bmatrix} & \begin{bmatrix} -I^{(2)} & 0 \end{bmatrix} & & & & & & \\
& & & \ddots & & & & & \\
& & & & \begin{bmatrix} I^{(S-1)} & 0 \end{bmatrix} & \begin{bmatrix} -I^{(S-1)} & 0 \end{bmatrix} & & &
\end{pmatrix}.$$

If we transpose the previous matrix, and permute nonanticipativity columns, we can consider the following matrix:

$$C^T = \left(\begin{array}{c|c|c|c|c|c|c}
B^{(1)^T} & I^{(1)} & & & & & \\
& 0 & & & & & \\
B^{(2)^T} & -I^{(1)} & & & & & \\
& 0 & & & & & \\
\hline
 & & B^{(3)^T} & I^{(2)} & & & \\
 & & & 0 & & & \\
 & & & -I^{(2)} & & & \\
 & & & 0 & & & \\
\hline
 & & & & \ddots & & \\
\hline
 & & & & B^{(S-1)^T} & I^{(S-2)} & \\
 & & & & & 0 & \\
 & & & & & -I^{(S-2)} & \\
 & & & & & 0 & \\
\hline
 & & & & & & B^{(S)^T} \\
 & & & & & & -I^{(S-1)} \\
 & & & & & & 0
\end{array}\right). \tag{4.13}$$

Let n_C denote the number of rows in C or, equivalently, the number of columns in C^T. Matrix (4.13) allows us to identify nonanticipativity constraints linking scenarios i and $i+1$ by considering only corresponding columns during a QR factorization of C^T. In order to identify redundant nonanticipativity constraints, we use a QR factorization with partial pivoting. We compute the factorization by generating Householder reflectors, columns by columns. Assume for some j that we have computed Householder matrices $H_1, \ldots, H_{j-1}$ and permutations $\Pi_1, \ldots, \Pi_{j-1}$ such that

$$\left(H_{j-1} \ldots H_1\right) C^T \left(\Pi_1 \ldots \Pi_{j-1}\right) = \begin{pmatrix} R_{11}^{j-1} & R_{12}^{j-1} \\ 0 & R_{22}^{j-1} \end{pmatrix},$$

where R_{11}^{j-1} is a nonsingular and upper triangular matrix. Consider the following column partitioning of R_{22}^{j-1}:

$$R_{22}^{j-1} = \left(r_j^{j-1}, \ldots, r_{n_C}^{j-1}\right).$$

Suppose furthermore that column j is associated to a nonanticipativity constraint linking scenarios i and $i+1$ and denote j_i the last column index corresponding to nonanticipativity constraints

linking scenarios i and $i+1$. Let $p \geq j$ be the smallest index such that

$$\left\|r_p^{j-1}\right\|_2 = \max\left\{\left\|r_j^{j-1}\right\|_2, \ldots, \left\|r_{j_i}^{j-1}\right\|_2\right\}.$$

If this maximum is zero, this implies that columns j to j_i are linearly dependent of column 1 to $j-1$, and therefore can be suppressed. In this case we apply the permutation $\Pi = \Pi_1 \ldots \Pi_{j-1}$ to C^T and delete columns j to j_i. Due to numerical errors, we consider that constraints j to j_i are redundant if

$$\|r_p^{j-1}\|_2 \leq \max\left\{\|c_i\|_2, i = 1, \ldots, n_C\right\} \epsilon_m,$$

where ϵ_m is the machine precision and c_i is the i^{th} row of C, $i = 1, \ldots, n_C$.

Otherwise, let Π_j be the nS-by-nS identity matrix with columns p and j interchanged. We determine then a Householder matrix H_j such that if $R^j = H_j R^{j-1} \Pi_j$, then $R^j\,(j+1:m, j) = 0$.

Let C' be the matrix resulting from C, after permutations and redundant constraints deletion. Therefore C' is of full row rank. We obtain finally the factorization

$$C'^T = QR, \tag{4.14}$$

or, in others terms, since $Q^T = Q$, we have

$$C' = R^T Q^T = LQ,$$

where L is lower triangular.

While this process is expensive, the resulting factorization can be exploited to compute a projection on the null space of the constraints. Assume that C has rank r. Then the (linearly independent) r first columns of Q determine a basis of $ran(C)$, while the remaining columns give a basis of its null space. Indeed since C'^T is of full column rank, we know that its number of columns is equal to r and that a basis of $ran\left(C'^T\right)$ is obtain with the r first columns of Q. The other columns are a basis for its null space. This comes from the fact that each column of C'^T is a linear combination of the first r columns of Q (see for example Golub and Van Loan [61], Section 5.2), and the same is true for C'.

Let

$$Q = \begin{pmatrix} D & K \end{pmatrix}, \tag{4.15}$$

where D contains the r first columns of Q and K, the remaining ones. Define

$$P = KK^T;$$

the matrix P is a projection onto $\mathcal{N}\left(\begin{pmatrix} B^T & N^T \end{pmatrix}^T\right)$.

If Assumption A.5 does not hold, problem (4.5)–(4.8) can be infeasible, even if for each scenario the associated subproblem is feasible. It is indeed possible that there exists some solution that fulfils the constraints $B^{(s)} z^{(s)} = b^{(s)}$, $s = 1, \ldots, S$, but is not nonanticipative. Unfortunately, the preprocessing suppresses the incompatible nonanticipative constraints, leading to a feasible problem. Feasibility of the original problem can be tested by checking that the starting point respects the suppressed constraints (see Section 4.4.1).

4.3 The algorithm

Our algorithm is basically a sequential minimization of a logarithmic barrier function subject to linear constraints as proposed in Conn, Gould, Orban, and Toint [30] for nonlinear problems with linear constraints. Recall that we want to solve the program (4.9)–(4.12):

$$\begin{aligned} \min\ & f(z) \\ \text{s.t. } & Bz = b, \\ & Nz = 0, \\ & z \geq 0. \end{aligned}$$

We propose to (approximately) solve

$$\begin{aligned} \min\ & \phi(z, \mu_k) \\ \text{s.t. } & Bz = b, \\ & Nz = 0, \end{aligned} \tag{4.16}$$

where

$$\phi(z, \mu_k) = f(z) - \mu_k \langle e, \log(z) \rangle,$$

for a sequence of barrier parameters $\mu_k > 0$, $k = 1, 2, \ldots$, satisfying

$$\lim_{k \to \infty} \mu_k = 0. \tag{4.17}$$

An approximate minimizer of problem (4.16), z_{k+1}, defines an outer iterate, and the associated adjustment of the barrier parameter and other tolerances defines the outer iteration. Outer iterations will be indexed by the subscript $k \geq 0$, while the subscript j will index the inner iteration. Each outer iterate z_{k+1} is computed by using an appropriate inner iteration algorithm to approximately solve (4.16), with a corresponding sequence of inner iterates $\{z_{k,j}\}$.

We start by explaining the inner iteration and the outer iteration. Next we give details about the step computation and show how the structure of the problem can be exploited for the problem in interest.

4.3.1 The inner iteration

We first examine the inner iteration, whose aim is to approximately solve (4.16) for a given value $\mu_k > 0$. For this purpose we apply a standard Newton-like trust-region method (see Section 2.3.3) with the restriction that the iterates have to lie in the null space of $\begin{pmatrix} B^T & N^T \end{pmatrix}^T$. The trust region is now defined as

$$\mathcal{B}_{k,j} \stackrel{def}{=} \{z_{k,j} + \Delta z_{k,j} \in \mathbb{R}^{nS} \mid B\Delta z_{k,j} = 0,\ N\Delta z_{k,j} = 0 \text{ and } \|\Delta z_{k,j}\|_{k,j} \leq \Delta_{k,j}\},$$

where $\|\cdot\|_{k,j}$ is the norm used at the iteration (k, j) and $\Delta_{k,j}$ is the trust-region radius at this iteration.

We consider here a quadratic model: at iteration (k, j), we attempt to decrease the value of the function

$$m_{k,j}(z_{k,j}+\Delta z_{k,j}) = \phi(z_{k,j},\mu_k)+\langle g_{k,j}-\mu_k Z_{k,j}^{-1}e, \Delta z_{k,j}\rangle+\frac{1}{2}\langle \Delta z_{k,j}, (H_{k,j}+Z_{k,j}^{-1}V_{k,j})\Delta z_{k,j}\rangle, \tag{4.18}$$

where $g_{k,j} = \nabla_z f(z_{k,j})$, $H_{k,j}$ is an approximation of $\nabla^2_{zz} f(z_{k,j})$ and $V_{k,j}$ is some bounded positive diagonal matrix of entries $[v_{k,j}]_i$, $i = 1, \ldots, nS$. This model can be seen as a quadratic model of the log-barrier function, with a modification of the barrier model's Hessian whose goal is to benefit from a primal-dual approach when solving problem (4.16) (see Conn, Gould, Orban, and Toint [30]). Note also that the approximation can be made scenario per scenario, with an approximation $H_{k,j}^{(s)}$ for $\nabla^2_{z^{(s)}z^{(s)}} f^{(s)}\left(z_{k,j}^{(s)}\right)$, $s = 1, \ldots, S$. $H_{k,j}$ is now defined as a block diagonal matrix gathering these approximations.

Let

$$D_{k,j} = Z_{k,j}^{-1}V_{k,j}, \quad G_{k,j} = H_{k,j} + D_{k,j},$$

and

$$D_{k,j}^{(s)} = \left(Z_{k,j}^{(s)}\right)^{-1} V_{k,j}^{(s)}, \quad G_{k,j}^{(s)} = H_{k,j}^{(s)} + D_{k,j}^{(s)}, \quad s = 1, \ldots, S.$$

The model can then be rewritten as

$$m_{k,j}(z_{k,j} + \Delta z_{k,j}) = \phi(z_{k,j},\mu_k) + \langle g_{k,j} - \mu_k Z_{k,j}^{-1}e, \Delta z_{k,j}\rangle + \frac{1}{2}\langle \Delta z_{k,j}, G_{k,j}\Delta z_{k,j}\rangle, \tag{4.19}$$

The general framework of the primal-dual inner algorithm is similar to the BTR Algorithm 2.2 on page 33, but now restricts the iterates to $\mathcal{N}\left(B^T \quad N^T\right)^T$.

Algorithm 4.1: Inner iteration

Step 0. Initialization. An initial point $z_{k,0} \in \text{strict}\{\mathcal{P}\} \cap \mathcal{L}$, a vector $v_{k,0} > 0$ of dual variables associated to the positivity constraints, and an initial trust-region radius $\Delta_{k,0}$ are given. The constants $\eta_1, \eta_2, \gamma_1, \gamma_2$ are also given, as in Algorithm 2.2. Finally, a $\zeta_k \in (0, 1)$ is also given. Compute $f(z_{k,0})$ (if not already known) and set $j = 0$.

Step 1. Model definition. Define a model $m_{k,j}$ of $\phi(z_{k,j} + \Delta z_{k,j}, \mu_k)$ in $\mathcal{B}_k$, which is of the form (4.19).

Step 2. Step calculation. Define $d_{k,j} = \text{dist}(z_{k,j}, \partial\mathcal{P})$. Compute a step $\Delta z_{k,j}$ such that

$$z_{k,j} + \Delta z_{k,j} \in \mathcal{B}_{k,j} \cap \mathcal{F} \text{ and dist}\,(z_{k,j} + s_{k,j}, \partial\mathcal{P}) \geq \zeta_k d_{k,j},$$

and such that it sufficiently reduces the model $m_{k,j}$.

Step 3. Acceptance of the trial point. Compute $\phi(z_{k,j} + \Delta z_{k,j}, \mu_k)$ and

$$\rho_{k,j} = \frac{\phi(z_{k,j},\mu_k) - \phi(z_{k,j} + \Delta z_{k,j},\mu_k)}{m_{k,j}(z_{k,j}) - m_{k,j}(z_{k,j} + \Delta z_{k,j})}.$$

Then if $\rho_{k,j} \geq \eta_1$, define $z_{k,j+1} = z_{k,j} + \Delta z_{k,j}$; otherwise $z_{k,j+1} = z_{k,j}$.

Step 4. Trust-region update. Identical to Step 4 of Algorithm 2.2.

Step 5. Update the dual variables. Define $v_{k,j+1} > 0$. Increment j by one and go to Step 1.

The starting point $(z_{k,0}, v_{k,0})$ is given by the final point obtained at the previous outer iteration if $k > 0$ (see Algorithm 4.3). The computation of an initial point $(z_{0,0}, v_{0,0})$ is discussed in Section 4.4.1.

It is important to notice that we are prepared to solve in Step 2 the trust-region subproblem

$$\begin{aligned} \min\ & m_{k,j}(z_{k,j} + \Delta z_{k,j}) \\ \text{s.t. } & B\Delta z_{k,j} = 0, \\ & N\Delta z_{k,j} = 0, \\ & z_{k,j} + \Delta z_{k,j} > 0, \\ \text{and } & \|\Delta z_{k,j}\|_{k,j} \leq \Delta_{k,j}, \end{aligned}$$

only approximately, in that we merely aim at improving $m_{k,j}(z_{k,j} + s)$ while satisfying the remaining constraints. The computation of the constrained Newton direction can be efficiently operated as we will show below. However, if this direction is especially interesting to accelerate convergence as we approach the solution, it does not guarantee global convergence of the algorithm. This can be obtained by first computing the projected constrained Cauchy point. The step is therefore computed in our algorithm by forming a dogleg path based on the constrained projected Cauchy point and a quasi-Newton step. We first introduce the latter because it allows a better insight into the model choice. Note that we choose to compute the step along the dogleg path instead of other paths as the projected conjugate gradient path (see for instance Conn, Gould, and Toint [35], Section 5.4) in order to privilege exploitation of the scenario structure. For simplicity, we will use the 2-norm at each iteration, so for convenience, we will omit the index k, j in $\|\cdot\|_{k,j}$ from now on.

Quasi-Newton step

We first rewrite problem (4.9)–(4.12) into an equivalent form. This will allow us to develop a linear system involving a symmetric matrix, as we show below. Writing problem (4.9)–(4.12) as

$$\begin{aligned} \min\ & f(z) && (4.20) \\ \text{s.t. } & -Bz = -b, && (4.21) \\ & -Nz = 0, && (4.22) \\ & z \geq 0, && (4.23) \end{aligned}$$

the Karush-Kuhn-Tucker conditions for this problem can then be expressed as:

$$g(z) + B^T u - v + N^T w = 0, \tag{4.24}$$
$$Bz = b, \tag{4.25}$$
$$Nz = 0, \tag{4.26}$$
$$(z, v) \geq 0, \tag{4.27}$$
$$v_i z_i = 0,\ i = 1, \ldots, nS, \tag{4.28}$$

where v is the vector of dual variables (Lagrange multipliers) for the bound constraints (4.23) and u and w are the vectors of Lagrange multipliers associated with the equality constraints (4.21) and (4.22), respectively. Equation (4.28) is the problem's complementarity condition (see Section 2.4.2), so we perturb it with the barrier parameter $\mu > 0$, and replace condition (4.28) by

$$ZVe = \mu e,$$

where $Z = \text{diag}(z_1, \ldots, z_{nS})$ and $V = \text{diag}(v_1, \ldots, v_{nS})$. The system (4.24)–(4.28) can then be restated as follows:

$$F(z, u, v, w) = \begin{pmatrix} g(z) + B^T u - v + N^T w \\ Bz - b \\ ZVe - \mu e \\ Nz \end{pmatrix} = 0, \tag{4.29}$$
$$(z, v) > 0. \tag{4.30}$$

Let $J(z, u, v, w)$ be the Jacobian of $F(z, u, v, w)$. Ignoring the nonnegativity constraints, the Newton's equation for this system at some inner iterate $(z_{k,j}, u_{k,j}, v_{k,j}, w_{k,j})$ and for some value μ_k of the barrier parameter is

$$J(z_{k,j}, u_{k,j}, v_{k,j}, w_{k,j}) \begin{pmatrix} \Delta z_{k,j} \\ \Delta u_{k,j} \\ \Delta v_{k,j} \\ \Delta w_{k,j} \end{pmatrix} = - \begin{pmatrix} g(z_{k,j}) + B^T u_{k,j} - v_{k,j} + N^T w_{k,j} \\ Bz_{k,j} - b \\ Z_{k,j} V_{k,j} e - \mu_k e \\ Nz_{k,j} \end{pmatrix}, \tag{4.31}$$

where

$$J = \begin{pmatrix} \nabla_{zz} f(z_{k,j}) & B^T & -I & N^T \\ B & 0 & 0 & 0 \\ V_{k,j} & 0 & Z_{k,j} & 0 \\ N & 0 & 0 & 0 \end{pmatrix}.$$

The primal-dual Newton's system corresponding to our problem can therefore be formulated as

$$\begin{pmatrix} H_{k,j} & B^T & -I & N^T \\ B & 0 & 0 & 0 \\ V_{k,j} & 0 & Z_{k,j} & 0 \\ N & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} \Delta z_{k,j} \\ \Delta u_{k,j} \\ \Delta v_{k,j} \\ \Delta w_{k,j} \end{pmatrix} = - \begin{pmatrix} g(z_{k,j}) + B^T u_{k,j} - v_{k,j} + N^T w_{k,j} \\ Bz_{k,j} - b \\ Z_{k,j} V_{k,j} e - \mu_k e \\ Nz_{k,j} \end{pmatrix}, \tag{4.32}$$

where we have replaced $\nabla_{zz} f(z_{k,j})$ by its approximation $H_{k,j}$. We immediately see that $H_{k,j}$ must be chosen so that the left-hand side matrix is nonsingular.

Note also that the introduction of the parameter μ_k can also be motivated by considering the barrier problem (4.16), as discussed in Section 2.4.2, where only inequality constraints were considered. Eliminating $\Delta v_{k,j}$ and writing $u_{k,j+1} = u_{k,j} + \Delta u_{k,j}$, $w_{k,j+1} = w_{k,j} + \Delta w_{k,j}$, we obtain the following system

$$\begin{pmatrix} H_{k,j} + D_{k,j} & B^T & N^T \\ B & 0 & 0 \\ N & 0 & 0 \end{pmatrix} \begin{pmatrix} \Delta z_{k,j} \\ u_{k,j+1} \\ w_{k,j+1} \end{pmatrix} = - \begin{pmatrix} g(z_{k,j}) - \mu_k Z_{k,j}^{-1} e \\ B z_{k,j} - b \\ N z_{k,j} \end{pmatrix} \tag{4.33}$$

and

$$\Delta v_{k,j} = \mu_k Z_{k,j}^{-1} e - v_{k,j} - D_{k,j} \Delta z_{k,j}.$$

The first equation of the system (4.33) can be rewritten as

$$G_{k,j} \Delta z_{k,j} + B^T u_{k,j+1} + N^T w_{k,j+1} = -\nabla_z \phi(z_{k,j}, \mu_k).$$

Moreover, equations (4.33) are precisely the first-order optimality conditions for the problem of minimizing the model (4.19), subject to the constraints

$$\begin{aligned} B(z_{k,j} + \Delta z_{k,j}) &= b, \\ N(z_{k,j} + \Delta z_{k,j}) &= 0. \end{aligned}$$

Hence $\Delta z_{k,j}$ may be interpreted as a constrained quasi-Newton step for the problem (4.16).

We now rewrite the system (4.32) to obtain a symmetric left-hand side matrix: premultiplying each side of (4.32) by

$$\begin{pmatrix} I & 0 & 0 & 0 \\ 0 & I & 0 & 0 \\ 0 & 0 & -V_{k,j}^{-1} & 0 \\ 0 & 0 & 0 & I \end{pmatrix},$$

the primal-dual system becomes

$$\begin{pmatrix} H_{k,j} & B^T & -I & N^T \\ B & 0 & 0 & 0 \\ -I & 0 & -D_{k,j}^{-1} & 0 \\ N & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} \Delta z_{k,j} \\ \Delta u_{k,j} \\ \Delta v_{k,j} \\ \Delta w_{k,j} \end{pmatrix} = - \begin{pmatrix} g(z_{k,j}) + B^T u_{k,j} - v_{k,j} + N^T w_{k,j} \\ B z_{k,j} - b \\ -Z_{k,j} e + \mu_k V_{k,j}^{-1} e \\ N z_{k,j} \end{pmatrix}. \tag{4.34}$$

The right-hand side of this system can be simplified if we require that each iterate is strictly feasible, or in other terms that constraints (4.10) and (4.11) are satisfied. We immediately see that if $z_{k,j}$ is strictly feasible, $z_{k,j+1}$ will also be strictly feasible if the step $\Delta z_{k,j}$ belongs to the null space of $\begin{pmatrix} B^T & N^T \end{pmatrix}^T$, i.e.

$$\begin{aligned} B \Delta z_{k,j} &= 0, \\ N \Delta z_{k,j} &= 0. \end{aligned}$$

In practice, the cost involved to produce such a starting point is balanced by the subsequent algorithmic simplifications (Conn, Gould, Orban, and Toint [30]). We discuss the search of such a point in Section 4.4.

Assume that $z_{k,j}$ is strictly feasible. The system (4.34) can then be rewritten as

$$\begin{pmatrix} H_{k,j} & B^T & -I & N^T \\ B & 0 & 0 & 0 \\ -I & 0 & -D_{k,j}^{-1} & 0 \\ N & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} \Delta z_{k,j} \\ \Delta u_{k,j} \\ \Delta v_{k,j} \\ \Delta w_{k,j} \end{pmatrix} = - \begin{pmatrix} g(z_{k,j}) + B^T u_{k,j} - v_{k,j} + N^T w_{k,j} \\ 0 \\ -Z_{k,j}e + \mu V_{k,j}^{-1} e \\ 0 \end{pmatrix}. \quad (4.35)$$

For reference, we will denote by $\Delta z_{k,j}^{CN}$ the quasi-Newton step obtained by solving the linear system (4.35).

Computing the step from (4.35) requires some precautions when the objective function is nonlinear nonconvex, since the coefficient matrix can then be singular. More significantly (4.35) is inappropriate if $G_{k,j}$ is not second-order sufficient with respect to $\begin{pmatrix} B^T & N^T \end{pmatrix}^T$, as $\Delta z_{k,j}$ at best defines a saddle point for the model. Forcing iterates to belong to a trust region is one possibility to circumvent these limitations (see Conn, Gould, Orban, and Toint [30]).

Restricting the directions in $\mathcal{N}\left(\begin{pmatrix} B^T & N^T \end{pmatrix}^T\right)$, the model (4.19) can be reformulated as

$$m_{k,j}(z_{k,j}+P\Delta z_{k,j}) = \phi(z_{k,j},\mu_k)+\langle \nabla_z \phi(z_{k,j},\mu_k), P\Delta z_{k,j}\rangle + \frac{1}{2}\langle K^T \Delta z_{k,j}, K^T G_{k,j} K K^T \Delta z_{k,j}\rangle.$$

If the approximation $G_{k,j}$ is second-order sufficient with respect to $\begin{pmatrix} B^T & N^T \end{pmatrix}^T$ then $K^T G_{k,j} K$ will be positive definite (Gould [66]). This assumption is usually made, at least asymptotically, to ensure optimality (see for instance Conn, Gould, and Toint [34]). We will also discuss this assumption in Section 4.3.2, describing the dogleg path approach used in our method. The computation of the quasi-Newton step will be addressed in Section 4.3.5.

The projected constrained Cauchy point

We now describe how to obtain the projected Cauchy point, in order to ensure global convergence (to a first-order critical point). Assume that we have a basis K of the null space of $\begin{pmatrix} B^T & N^T \end{pmatrix}^T$. Define

$$P = KK^T.$$

This matrix P is an orthogonal projective matrix onto $\mathcal{N}\left(\begin{pmatrix} B^T & N^T \end{pmatrix}^T\right)$ (see Section 4.2.1 for a way to obtain such a matrix). We can thus project the gradient of the model by applying P to it, as it is done in some indefinite dogleg methods (Zhang and Xu [143]). The projected gradient is still a descent direction since

$$\begin{aligned} \langle -KK^T \nabla_z \phi(z_{k,j},\mu_k), \nabla_z \phi(z_{k,j},\mu_k)\rangle &= -\langle K^T \nabla_z \phi(z_{k,j},\mu_k), K^T \nabla_z \phi(z_{k,j},\mu_k)\rangle \\ &= -\|K^T \nabla_z \phi(z_{k,j},\mu_k)\|^2. \end{aligned}$$

The projected constrained Cauchy point

$$z_{k,j}^{\mathrm{CC}} = z_{k,j} - t_{k,j}^{CC} P \nabla_z \phi\left(z_{k,j},\mu_k\right) \quad (4.36)$$

is now defined as the solution of the problem

$$\min_{t\geq 0} m_{k,j}\left(z_{k,j} - tP\nabla_z\phi\left(z_{k,j},\mu_k\right)\right), \tag{4.37}$$

such that

$$z_{k,j} - tP\nabla_z\phi\left(z_{k,j},\mu_k\right) \in \mathcal{B}_{k,j} \text{ and } t\left\|P\nabla_z\phi\left(z_{k,j},\mu_k\right)\right\| \leq (1-\zeta_k)\, d_{k,j}. \tag{4.38}$$

The left part of (4.38) ensures that the candidate point belongs to the trust region while the right part guarantees that we stop before we get too close to the boundary of the positive orthant ∂P.

The constrained projected Cauchy step $\Delta z_{k,j}^{CC}$ is therefore defined as

$$\Delta z_{k,j}^{CC} = -t_{k,j}^{CC} P\nabla_z\phi\left(z_{k,j},\mu_k\right). \tag{4.39}$$

Computation of the projected constrained Cauchy point

The computation of the constrained Cauchy point involves a minimization process in one dimension, so the main cost is in the computation of the projected gradient of the log-barrier model. The gradient is

$$g\left(z_{k,j}\right) - \mu_k Z_{k,j}^{-1} e,$$

which can easily be calculated scenario by scenario since

$$\nabla_z\phi\left(z_{k,j},\mu_k\right) = \begin{pmatrix} g^{(1)}\left(z_{k,j}^{(1)}\right) - \mu_k {Z_{k,j}^{(1)}}^{-1} e \\ \vdots \\ g^{(S)}\left(z_{k,j}^{(S)}\right) - \mu_k {Z_{k,j}^{(S)}}^{-1} e \end{pmatrix}.$$

We have that

$$\begin{aligned} m_{k,j}\left(z_{k,j} - tP\nabla_z\phi\left(z_{k,j},\mu_k\right)\right) = \phi\left(z_{k,j},\mu_k\right) - t\left\|P\nabla_z\phi\left(z_{k,j},\mu_k\right)\right\|^2 \\ + \frac{1}{2}t^2\left\langle P\nabla_z\phi\left(z_{k,j},\mu_k\right), G_{k,j}P\nabla_z\phi\left(z_{k,j},\mu_k\right)\right\rangle. \end{aligned}$$

If the curvature of the model is negative in the direction $-P\nabla_z\phi\left(z_{k,j},\mu_k\right)$, then, from Step 2 of Algorithm 4.1, the optimal step length is given by

$$t_k^{CC} = \min\left[\frac{\Delta_{k,j}}{\|P\nabla_z\phi(z_{k,j},\mu_k)\|}, \frac{(1-\zeta_k)d_{k,j}}{\|P\nabla_z\phi(z_{k,j},\mu_k)\|}\right].$$

Otherwise, it is defined as

$$t_k^{CC} = \min\left[\frac{\Delta_k}{\|P\nabla_z\phi(z_{k,j},\mu_k)\|}, \frac{(1-\zeta_k)d_{k,j}}{\|P\nabla_z\phi(z_{k,j},\mu_k)\|}, \frac{\|P\nabla_z\phi(z_{k,j},\mu_k)\|^2}{\langle P\nabla_z\phi(z_{k,j},\mu_k), G_{k,j}P\nabla_z\phi(z_{k,j},\mu_k)\rangle}\right]. \tag{4.40}$$

The main cost is the application of the projection operator P since its size is $nS \times nS$. Fortunately, as it is simply a matrix-vector product, we can easily parallelize it in order to work

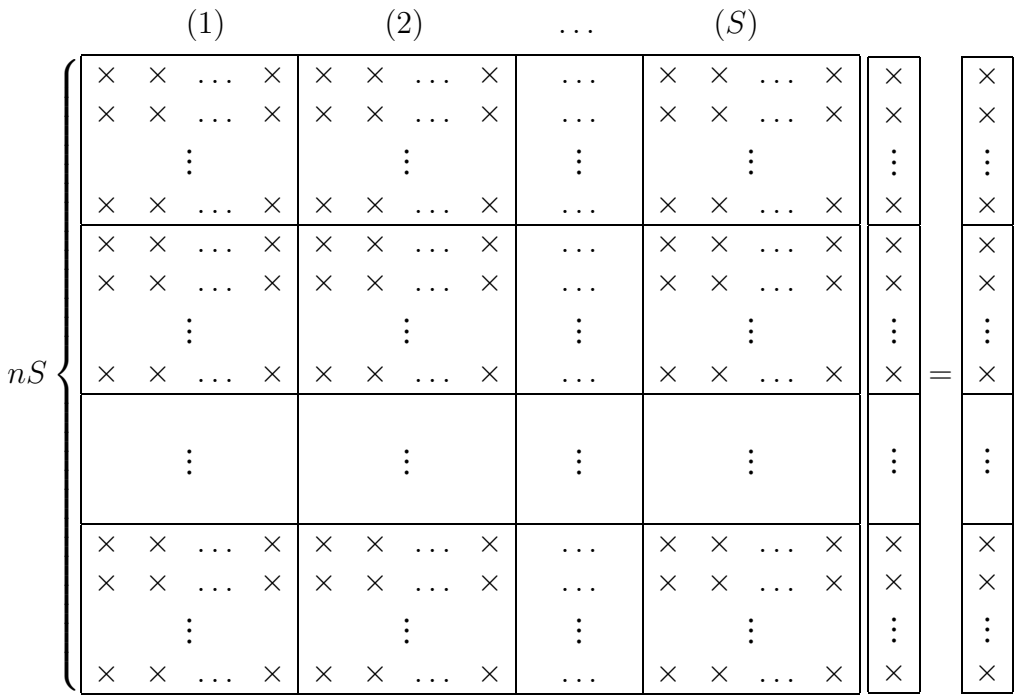


Figure 4.1: Projection application

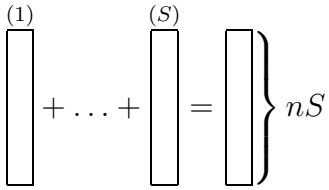


Figure 4.2: Combination of the partial results for the projection

scenario per scenario. Considering Figure 4.1, we see that we can split the projection matrix into S parts. We can therefore take the n first columns and apply them to the gradient of the first scenario objective. We repeat the process with the n next columns and the gradient of the second scenario. The same procedure is applied for each scenario. Finally, the projected gradient is computed by summing the S intermediate results, as illustrated in Figure 4.2.

The projected constrained Cauchy step ensures that a positive decrease of the model $m_{k,j}$ is obtained at each iteration (k, j), as stated in the following theorem.

Theorem 4.1 *If the model is of the form* (4.19) *and if we define the projected constrained Cauchy point by* (4.37)–(4.38)*, we have that*

$$m_{k,j}(z_{k,j}) - m_{k,j}(z_{k,j} + \Delta z_{k,j}^{CC}) \geq \frac{1}{2}\|P\nabla_z\phi(z_{k,j}, \mu_k)\| \min\left\{\frac{\|P\nabla_z\phi(z_{k,j}, \mu_k)\|}{\beta_{k,j}}, \Delta_{k,j}, (1-\zeta_k)d_{k,j}\right\},$$

where $\beta_{k,j} = 1 + \max_{x \in \mathcal{B}_k} \|\nabla_{zz} m_k(z)\|$.

Proof. The proof is similar to that of Theorem 13.3.1 in Conn, Gould, and Toint [35]. □

4.3.2 Dogleg path

Let define

$$\Delta z_{k,j}^{CP} = -\frac{\|P\nabla_z \phi(z_{k,j}, \mu_k)\|^2}{\langle P\nabla_z \phi(z_{k,j}, \mu_k), G_{k,j} P\nabla_z \phi(z_{k,j}, \mu_k)\rangle} P\nabla_z \phi(z_{k,j}, \mu_k).$$

If the model curvature is positive in the direction $\Delta z_{k,j}^{CP}$, then from (4.39) and (4.40), the projected constrained Cauchy step $\Delta z_{k,j}^{CC}$ is on the segment $z_{k,j} + \alpha \Delta z_{k,j}^{CP}$, $0 \leq \alpha \leq 1$. We combine $\Delta z_{k,j}^{CP}$ with the quasi-Newton step, denoted as previously by $\Delta z_{k,j}^{CN}$, to form a single dogleg path:

$$\Delta z_{k,j}(\alpha) = \begin{cases} \alpha \Delta z_{k,j}^{CP}, & \text{for } 0 \leq \alpha \leq 1, \\ \Delta z_{k,j}^{CP} + (\alpha - 1)(\Delta z_{k,j}^{CN} - \Delta z_{k,j}^{CP}) & 1 \leq \alpha \leq 2. \end{cases} \tag{4.41}$$

For notational convenience, we also define

$$p_{k,j}(\Delta z_{k,j}) = m_{k,j}(z_{k,j} + \Delta z_{k,j}) = \phi(z_{k,j}, \mu_k) + \langle \nabla_z \phi(z_{k,j}, \mu_k), \Delta z_{k,j}\rangle + \frac{1}{2}\langle \Delta z_{k,j}, G_{k,j} \Delta z_{k,j}\rangle, \tag{4.42}$$

and

$$h_{k,j} = P\nabla_z \phi\left(z_{k,j}, \mu_k\right). \tag{4.43}$$

Assuming that $h_{k,j} \neq 0$, we have

$$\Delta z_{k,j}^{CP} = -\frac{\|h_{k,j}\|^2}{\langle h_{k,j}, G_{k,j} h_{k,j}\rangle} h_{k,j}. \tag{4.44}$$

In order to ensure that (4.41) is actually a dogleg path for our problem, we have to check that the two following conditions are satisfied:

(R1) $\|\Delta z_{k,j}(\alpha)\|$ is monotonically increasing with α, $0 \leq \alpha \leq 2$;

(R2) $p_{k,j}(\Delta z_{k,j}(\alpha))$ decreases monotically as α increases.

Note that **(R1)** implies that when the path goes out of the trust region, it cannot reenter into it. Moreover since $\Delta z_{k,j}^{CP}$ and $\Delta z_{k,j}^{CN}$ belong to the null space of $\begin{pmatrix} B^T & N^T \end{pmatrix}^T$, this path is entirely in $\mathcal{N}\left(\begin{pmatrix} B^T & N^T \end{pmatrix}^T\right)$, ensuring that the step is in $\mathcal{N}\left(\begin{pmatrix} B^T & N^T \end{pmatrix}^T\right)$, and the iterate $z_{k,j+1}$ satisfies the constraints (4.21) and (4.22).

Lemma 4.1 *If $G_{k,j}$ is second-order sufficient with respect to $\begin{pmatrix} B^T & N^T \end{pmatrix}^T$,*

$$\langle h_{k,j}, \Delta z_{k,j}^{CN}\rangle \leq -\frac{\|h_{k,j}\|^4}{\langle h_{k,j}, G_{k,j} h_{k,j}\rangle} < 0. \tag{4.45}$$

Proof. Starting from the system (4.33), we have

$$G_{k,j}\Delta z_{k,j}^{CN} + B^T u_{k,j+1} + N^T w_{k,j+1} = -\nabla_z \phi(z_{k,j}, \mu_k). \tag{4.46}$$

Premultiplying each part of (4.46) by the projection operator P on $\mathcal{N}\left(\begin{pmatrix} B^T & N^T \end{pmatrix}^T\right)$, we obtain

$$PG_{k,j}\Delta z_{k,j}^{CN} = -h_{k,j}, \tag{4.47}$$

and from (4.43),

$$h_{k,j} = Ph_{k,j}. \tag{4.48}$$

Since $\Delta z_{k,j}^{CN}$ and $h_{k,j}$ belong to the null space of $\begin{pmatrix} B^T & N^T \end{pmatrix}^T$, we have from (4.47) that

$$\begin{aligned}
&\langle h_{k,j}, \Delta z_{k,j}^{CN}\rangle \langle h_{k,j}, G_{k,j}h_{k,j}\rangle + \|h_{k,j}\|^4 \\
&\quad = -\langle PG_{k,j}\Delta z_{k,j}^{CN}, \Delta z_{k,j}^{CN}\rangle \langle h_{k,j}P^T, G_{k,j}Ph_{k,j}\rangle + \langle h_{k,j}, PG_{k,j}\Delta z_{k,j}^{CN}\rangle^2 \\
&\quad = -\langle \Delta z_{k,j}^{CN}, PGP\Delta z_{k,j}^{CN}\rangle \langle h_{k,j}, PG_{k,j}Ph_{k,j}\rangle + \langle h_{k,j}, PG_{k,j}P\Delta z_{k,j}^{CN}\rangle^2.
\end{aligned}$$

Developing $P = KK^T$, we can write

$$\begin{aligned}
&\langle h_{k,j}, \Delta z_{k,j}^{CN}\rangle \langle h_{k,j}, G_{k,j}h_{k,j}\rangle + \|h_{k,j}\|^4 \\
&\quad = -\langle K^T\Delta z_{k,j}^{CN}, K^TG_{k,j}KK^T\Delta z_{k,j}^{CN}\rangle \langle K^Th_{k,j}, K^TG_{k,j}KK^Th_{k,j}\rangle \\
&\qquad + \langle K^Th_{k,j}, K^TG_{k,j}KK^T\Delta z_{k,j}^{CN}\rangle^2.
\end{aligned} \tag{4.49}$$

Since $G_{k,j}$ is second-order sufficient with respect to $\begin{pmatrix} B^T & N^T \end{pmatrix}^T$, $K^TG_{k,j}K$ is positive definite and we can introduce the scalar product

$$\langle x, y\rangle_{K^TG_{k,j}K} = \langle x, K^TG_{k,j}Ky\rangle,$$

and its associated matrix norm

$$\|x\|_{K^TG_{k,j}K} = \sqrt{\langle x, K^TG_{k,j}Kx\rangle}.$$

We can thus rewrite (4.49) as

$$\begin{aligned}
&\langle h_{k,j}, \Delta z_{k,j}^{CN}\rangle \langle h_{k,j}, G_{k,j}h_{k,j}\rangle + \|h_{k,j}\|^4 \\
&\quad = -\langle K^T\Delta z_{k,j}^{CN}, K^T\Delta z_{k,j}^{CN}\rangle_{K^TG_{k,j}K} \langle K^Th_{k,j}, K^Th_{k,j}\rangle_{K^TG_{k,j}K} \\
&\qquad + \langle K^Th_{k,j}, K^T\Delta z_{k,j}^{CN}\rangle^2_{K^TG_{k,j}K}.
\end{aligned} \tag{4.50}$$

From the Cauchy-Schwartz inequality, we have

$$\langle K^Th_{k,j}, K^T\Delta z_{k,j}^{CN}\rangle^2_{K^TG_{k,j}K} \leq \langle K^T\Delta z_{k,j}^{CN}, K^T\Delta z_{k,j}^{CN}\rangle_{K^TG_{k,j}K} \langle K^Th_{k,j}, K^Th_{k,j}\rangle_{K^TG_{k,j}K}.$$

Therefore, from (4.50),

$$\langle h_{k,j}, \Delta z_{k,j}^{CN}\rangle \langle h_{k,j}, G_{k,j}h_{k,j}\rangle + \|h_{k,j}\|^4 \leq 0,$$

which yields the desired conclusion. □

We now prove that $\Delta z_{k,j}(\alpha)$ $(0 \leq \alpha \leq 2)$ is a dogleg path, as desired.

Theorem 4.2 *If $G_{k,j}$ is second-order sufficient with respect to $\begin{pmatrix} B^T & N^T \end{pmatrix}^T$, then $\Delta z_{k,j}(\alpha)$ $(0 \leq \alpha \leq 2)$ is a dogleg path in $\mathcal{N}\left(\begin{pmatrix} B^T & N^T \end{pmatrix}^T\right)$.*

Proof. From (4.41), $\|\Delta z_{k,j}(\alpha)\|$ is increasing with α, for $\alpha \in [0,1]$. All we have to prove is that it is also increasing for $\alpha \in [1,2]$. From Lemma 4.1 and (4.44), we have that

$$\left\|\Delta z_{k,j}^{CP}\right\| \|h_{k,j}\| = \frac{\|h_{k,j}\|^2}{\langle h_{k,j}, G_{k,j} h_{k,j}\rangle}\|h_{k,j}\|^2 \leq \left|\langle h_{k,j}, \Delta z_{k,j}^{CN}\rangle\right| \leq \left\|\Delta z_{k,j}^{CN}\right\| \|h_{k,j}\|,$$

where we have again used the Cauchy-Schwartz inequality to obtain the last inequality. Therefore we have

$$\left\|\Delta z_{k,j}^{CP}\right\| \leq \left\|\Delta z_{k,j}^{CN}\right\|.$$

In other words, the projected Cauchy step is shorter than the quasi-Newton step. Moreover, from Lemma 4.1, we also have that

$$\begin{aligned}\langle \Delta z_{k,j}^{CP}, \Delta z_{k,j}^{CN} - \Delta z_{k,j}^{CP}\rangle &= -\frac{\|h_{k,j}\|^2}{\langle h_{k,j}, G_{k,j} h_{k,j}\rangle}\langle h_{k,j}, \Delta z_{k,j}^{CN}\rangle - \frac{\|h_{k,j}\|^6}{\langle h_{k,j}, G_{k,j} h_{k,j}\rangle^2} \\ &= -\frac{\|h_{k,j}\|^2}{\langle h_{k,j}, G_{k,j} h_{k,j}\rangle}\left(\langle h_{k,j}, \Delta z_{k,j}^{CN}\rangle + \frac{\|h_{k,j}\|^4}{\langle h_{k,j}, G_{k,j} h_{k,j}\rangle}\right) \geq 0.\end{aligned}$$

Therefore $\|\Delta z_{k,j}(\alpha)\|$ is increasing with α, for $\alpha \in [1,2]$, and the path satisfies **(R1)**.

We now proved that **(R2)** is also valid. Using $P\Delta z_{k,j}(\alpha) = \Delta z_{k,j}(\alpha)$ and $P = P^T$, we can rewrite (4.42) as follows:

$$p_{k,j}(\Delta z_{k,j}(\alpha)) = \phi(z_{k,j}, \mu_k) + \langle P\nabla_z \phi(z_{k,j}, \mu_k), \Delta z_{k,j}(\alpha)\rangle + \frac{1}{2}\langle \Delta z_{k,j}(\alpha), G_{k,j}\Delta z_{k,j}(\alpha)\rangle.$$

The value of $m(z_{k,j} + \Delta z_{k,j}(\alpha))$ decreases along the first piece of the path, where α varies from 0 to 1. Consider a point on the second piece of the path. For $1 \leq \alpha \leq 2$,

$$\Delta z_{k,j}(\alpha) = \Delta z_{k,j}^{CP} + (\alpha - 1)\left(\Delta z_{k,j}^{CN} - \Delta z_{k,j}^{CP}\right),$$

so we have

$$\frac{d}{d\alpha} p_{k,j}(\Delta z_{k,j}(\alpha)) = \left\langle \nabla_{\Delta z} p_{k,j}(\Delta z_{k,j}(\alpha)), \Delta z_{k,j}^{CN} - \Delta z_{k,j}^{CP}\right\rangle. \tag{4.51}$$

Developing $\nabla_{\Delta z} p_{k,j}(\Delta z_{k,j}(\alpha))$, we obtain from (4.51) that

$$\frac{d}{d\alpha} p_{k,j}(\Delta z_{k,j}(\alpha)) = \left\langle h_{k,j}^T + G_{k,j}\left(\Delta z_{k,j}^{CP} + (\alpha - 1)\left(\Delta z_{k,j}^{CN} - \Delta z_{k,j}^{CP}\right)\right), \left(\Delta z_{k,j}^{CN} - \Delta z_{k,j}^{CP}\right)\right\rangle$$

and thus,

$$\begin{aligned}\frac{d}{d\alpha} p_{k,j}(\Delta z_{k,j}(\alpha)) = &\left\langle h_{k,j}, \Delta z_{k,j}^{CN} - \Delta z_{k,j}^{CP}\right\rangle + (2 - \alpha)\left\langle \Delta z_{k,j}^{CP}, G_{k,j}\left(\Delta z_{k,j}^{CN} - \Delta z_{k,j}^{CP}\right)\right\rangle + \\ &(\alpha - 1)\left\langle \Delta z_{k,j}^{CN}, G_{k,j}\left(\Delta z_{k,j}^{CN} - \Delta z_{k,j}^{CP}\right)\right\rangle.\end{aligned} \tag{4.52}$$

Note that, from (4.44), we have

$$\left\langle \Delta z_{k,j}^{CP}, G_{k,j}\Delta z_{k,j}^{CP}\right\rangle = \frac{\|h_{k,j}\|^4}{\langle h_{k,j}, G_{k,j}h_{k,j}\rangle},$$

and, from (4.47),

$$\begin{aligned}\left\langle \Delta z_{k,j}^{CP}, G_{k,j}\Delta z_{k,j}^{CN}\right\rangle &= -\left\langle h_{k,j}, \Delta z_{k,j}^{CP}\right\rangle,\\ \left\langle \Delta z_{k,j}^{CN}, G_{k,j}\Delta z_{k,j}^{CN}\right\rangle &= -\left\langle h_{k,j}, \Delta z_{k,j}^{CN}\right\rangle.\end{aligned}$$

Thus, (4.52) can be rewritten as

$$\begin{aligned}\frac{d}{d\alpha}p_{k,j}(\Delta z_{k,j}(\alpha)) = \left\langle h_{k,j}, \Delta z_{k,j}^{CN} - \Delta z_{k,j}^{CP}\right\rangle + (2-\alpha)\left(-\left\langle h_{k,j}, \Delta z_{k,j}^{CP}\right\rangle + \frac{\|h_{k,j}\|^4}{\langle h_{k,j}, G_{k,j}h_{k,j}\rangle}\right) +\\ (\alpha-1)\left\langle h_{k,j}, \Delta z_{k,j}^{CP} - \Delta z_{k,j}^{CN}\right\rangle,\end{aligned}$$

leading to

$$\frac{d}{d\alpha}p_{k,j}(\Delta z_{k,j}(\alpha)) = (2-\alpha)\left(\left\langle h_{k,j}, \Delta z_{k,j}^{CN}\right\rangle + \frac{\|h_{k,j}\|^4}{\langle h_{k,j}, G_{k,j}h_{k,j}\rangle}\right) \leq 0.$$

Therefore the value of $m(z_{k,j} + \Delta z_{k,j}(\alpha))$ is also monotically decreasing along the second piece of the path and **(R2)** is satisfied. □

We can summarize the computation of the step length and direction as described in Algorithm 4.2.

Algorithm 4.2: Step calculation

1. Compute the constrained quasi-Newton step by solving (4.35).
2. If the quasi-Newton step is within the trust region and $\left\|\Delta z_{k,j}^{CN}\right\| \leq (1-\zeta_k)d_{k,j}$, then set $\Delta z_{k,j} = \Delta z_{k,j}^{CN}$.
3. Else compute the constrained Cauchy step. If $\Delta z_{k,j}^{CC}$ is not equal to $\Delta z_{k,j}^{CP}$, set $\Delta z_{k,j} = \Delta z_{k,j}^{CC}$. Otherwise set $\Delta z_{k,j} = \Delta z_{k,j}^{CP} + (\alpha-1)\left(\Delta z_{k,j}^{CN} - \Delta z_{k,j}^{CC}\right)$, where α is chosen such that $\|\Delta z_{k,j}\| = \min\left[\Delta_{k,j}, (1-\zeta_k)d_{k,j}\right]$.

Note that it is possible to relax the second-order sufficiency condition in Theorem 4.2, as done for instance in Zhang and Xu [143]. However, if $G_{k,j}$ is not second-order sufficient, the dogleg path is then usually based on some modifications of the primal-dual system (4.35) for which a new decomposition scheme has to be defined. The second-order sufficiency assumption is therefore useful to keep the method simple, and as said before, is not too restrictive since it is often required asymptotically to prove second-order convergence.

4.3.3 Updating the dual variables

We now indicate how the dual variables $v_{k,j+1}$ may be updated in practice in Step 5 of the primal-dual inner Algorithm 4.1. Our approach follows the proposition of Conn, Gould, Orban, and Toint [30]. A simple idea is to use the value predicted in the middle part of the quasi-Newton system (4.35):

$$\overline{v}_{k,j+1} = v_{k,j} + \Delta v_{k,j} = \mu_k Z_{k,j}^{-1} e - Z_{k,j}^{-1} V_{k,j} \Delta z_{k,j}. \tag{4.53}$$

However, there is no guarantee that the choice $\overline{v}_{k,j+1}$ maintains feasibility ($v_{k,j+1} \geq 0$). Moreover, the dual variables have to satisfy some technical conditions to ensure optimality (see again Conn, Gould, Orban, and Toint [30]). It can be shown that an adequate update is achieved by projecting (componentwise) $\overline{v}_{k,j+1}$ into the interval

$$\mathcal{K} = \left[\kappa_1 \min\{e, v_{k,j}, \mu_k Z_{k,j+1}^{-1} e\}, \max\{\kappa_2 e, v_{k,j}, \kappa_2 \mu_k^{-1} e, \kappa_2 \mu_k Z_{k,j+1}^{-1} e\}\right],$$

where κ_1 and κ_2 are constants such that

$$0 < \kappa_1 < 1 < \kappa_2.$$

This is to say that

$$v_{k,j+1} = \begin{cases} P_{\mathcal{K}}[\overline{v}_{k,j+1}] & \text{if } z_{k,j+1} = z_{k,j} + \Delta z_{k,j}, \\ v_{k,j} & \text{if } z_{k,j+1} = z_{k,j}, \end{cases}$$

where $P_{\mathcal{K}}[x]$ is the componentwise projection of the vector x onto the interval $\mathcal{K}$. Convenient values for the constants are $\kappa_1 = 0.5$ and $\kappa_2 = 10^{20}$.

4.3.4 The outer iteration

The outer iteration that controls the global optimization process is described in Algorithm 4.3. It is similar to the outer iteration of the primal-dual algorithm proposed in Conn, Gould, and Toint [35].

Algorithm 4.3: Outer iteration

Step 0. Initialization. An initial feasible point $z_0 > 0$, a vector of initial dual variables $v_0^T > 0$ and an initial barrier parameter μ_0 are given. The forcing functions $\epsilon^D(\mu)$, $\epsilon^E(\mu)$ and $\epsilon^C(\mu)$ are also given. Set $k = 0$.

Step 1. Inner minimization. Choose a value $\zeta_k \in (0, 1)$. Approximately minimize the log-barrier function $\phi(z, \mu_k) = f(z) - \mu_k \langle e, \log(z) \rangle$ using Algorithm 4.1. Stop this algorithm as soon as an iterate $(z_{k,j}, v_{k,j}) = (z_{k+1}, v_{k+1})$ is found for which

$$\left\| \nabla_{z^{(s)}} f^{(s)} \left(z_{k+1}^{(s)} \right) - v_{k+1}^{(s)} \right\| \leq \epsilon^D(\mu_k), \ s = 1, \ldots, S,$$
$$\| Z_{k+1} V_{k+1} - \mu_k I \| \leq \epsilon^C(\mu_k),$$
$$\lambda_{min} \left[G_{k+1}^{(s)} + \left(Z_{k+1}^{(s)} \right)^{-1} V_{k+1}^{(s)} \right] \geq -\epsilon^E(\mu_k), \ s = 1, \ldots, S,$$

and

$$z_{k+1} > 0 \quad \text{and} \quad v_{k+1} > 0.$$

Step 2. Update the barrier parameter. Choose $\mu_{k+1} > 0$ in such a way to ensure that

$$\lim_{k \to \infty} \mu_k = 0.$$

Increment k by 1 and return to Step 1.

The convergence of the algorithm can be proved as in Conn, Gould, and Toint [35], Chapter 13 (see in particular Section 13.9 and the discussion in the beginning of Section 13.10). A more general version can be found in Conn, Gould, Orban, and Toint [30] and Gould, Orban, Sartenaer, and Toint [65]. The practical aspects for Algorithm 4.3 will be discussed in Section 4.4.

4.3.5 Quasi-Newton step computation

We now focus on the practical details when computing the quasi-Newton step during an inner iteration, and show how to benefit from the structure of the nonlinear stochastic program (4.5)–(4.8). To achieve this, we propose a new decomposition scheme that exhibits strong parallelism properties, by treating separately the nonanticipativity constraints.

Decomposition of the primal-dual system

The primal-dual system (4.35) could be split into S smaller systems, one per scenario, if there were no nonanticipativity constraints. This suggests decomposing it in separable systems, when possible. In this section, we show that the solution of the system (4.35) can be replaced by the solutions of three systems, two of them being separable with respect to the scenarios. The third system explicitly takes the nonanticipativity constraints into account, and has a sparse structure that can be exploited. For simplicity, we will omit the subscripts k, j in the developments below.

Consider the first system,

$$\begin{pmatrix} H & B^T & -I \\ B & 0 & 0 \\ -I & 0 & -D^{-1} \end{pmatrix} \begin{pmatrix} \overline{\Delta z} \\ \overline{\Delta u} \\ \overline{\Delta v} \end{pmatrix} = \begin{pmatrix} g(z) + B^T u - v \\ 0 \\ -Ze + \mu V^{-1} e \end{pmatrix}, \tag{4.54}$$

with solutions $\overline{\Delta z}$, $\overline{\Delta u}$, $\overline{\Delta v}$. Eliminating $\overline{\Delta v}$ in this system, we obtain

$$\begin{pmatrix} G & B^T \\ B & 0 \end{pmatrix} \begin{pmatrix} \overline{\Delta z} \\ \overline{\Delta u} \end{pmatrix} = \begin{pmatrix} g(z) + B^T u - \mu Z^{-1} e \\ 0 \end{pmatrix}, \tag{4.55}$$

where

$$\overline{\Delta v} = -D\overline{\Delta z} + v - \mu Z^{-1} e.$$

In order to have a well-defined system (4.55), we need to impose that the matrix

$$J = \begin{pmatrix} G & B^T \\ B & 0 \end{pmatrix}$$

is nonsingular. Let A be constructed so that $BA = 0$ and rank $\begin{pmatrix} B^T & A \end{pmatrix} = nS$. From the Sylvester's law of inertia, we have that J is nonsingular if and only if $A^T G A$ is nonsingular (Gould [66]). In other terms, J is nonsingular if and only if G is second-order nonsingular with respect to B. We therefore make the following assumption from now on:

A.6 *G is second-order nonsingular with respect to B.*

Using the solution of the system (4.54), the system (4.35) can now be rewritten as

$$\begin{pmatrix} H & B^T & -I & N^T \\ B & 0 & 0 & 0 \\ -I & 0 & -D^{-1} & 0 \\ N & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} \Delta z + \overline{\Delta z} \\ \Delta u + \overline{\Delta u} \\ \Delta v + \overline{\Delta v} \\ \Delta w + w \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ N\overline{\Delta z} \end{pmatrix}.$$

Defining $\tilde{\Delta} z = \Delta z + \overline{\Delta z}$, $\tilde{\Delta} u = \Delta u + \overline{\Delta u}$, $\tilde{\Delta} v = \Delta v + \overline{\Delta v}$ and $\tilde{\Delta} w = \Delta w + w$, we obtain the following set of equations:

$$H\tilde{\Delta} z + B^T \tilde{\Delta} u - \tilde{\Delta} v + N^T \tilde{\Delta} w = 0, \tag{4.56}$$

$$B\tilde{\Delta} z = 0, \tag{4.57}$$

$$-\tilde{\Delta} z - D^{-1}\tilde{\Delta} v = 0, \tag{4.58}$$

$$N\tilde{\Delta} z = N\overline{\Delta z}. \tag{4.59}$$

The second system of interest, that we now construct, will allow us to determine Δw, so that we will be able to eliminate this variable in the third step of the decomposition. Putting (4.58) in (4.56), we have

$$-HD^{-1}\tilde{\Delta} v + B^T \tilde{\Delta} u - \tilde{\Delta} v + N^T \tilde{\Delta} w = 0,$$

or, equivalently,

$$-\left(I + HD^{-1}\right)\tilde{\Delta} v + B^T \tilde{\Delta} u + N^T \tilde{\Delta} w = 0. \tag{4.60}$$

Hence (4.60) can be rewritten as

$$-GD^{-1}\tilde{\Delta} v + B^T \tilde{\Delta} u + N^T \tilde{\Delta} w = 0. \tag{4.61}$$

In order to derive $\tilde{\Delta} v$ from (4.60), we must impose that G is invertible; in other terms we require the following assumption to hold:

A.7 *G is nonsingular.*

Note that Assumption A.7 does not imply that Assumption A.6 is satisfied, as shown in the example below.

Example 4.2. *Consider the matrix A of dimensions $n \times n$ and the full row rank matrix B, of dimensions $m \times n$ ($m < n$):*

$$A = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}, \qquad B = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}.$$

Then we have that

$$BAB^T = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix},$$

which is clearly singular.

Under Assumption A.7, we obtain from (4.61) that

$$\tilde{\Delta v} = DG^{-1}B^T\tilde{\Delta u} + DG^{-1}N^T\tilde{\Delta w}. \tag{4.62}$$

From (4.57) and (4.58), we can write

$$BD^{-1}\tilde{\Delta v} = 0. \tag{4.63}$$

Premultiplying each side of (4.62) by BD^{-1}, we have from (4.63) that

$$BG^{-1}B^T\tilde{\Delta u} + BG^{-1}N^T\tilde{\Delta w} = 0.$$

By Assumptions A.6 and A.7, $BG^{-1}B^T$ is nonsingular. Indeed, we have that (Gould [66])

$$In(A) = In(G) + In(-BG^{-1}B^T),$$

where $In(A)$ is the inertia of A (see Section 1.3.1). Therefore, under Assumption A.7, A is nonsingular if and only if $BG^{-1}B^T$ is invertible. Hence,

$$\tilde{\Delta u} = -\left(BG^{-1}B^T\right)^{-1} BG^{-1}N^T\tilde{\Delta w}. \tag{4.64}$$

From (4.58) and (4.59), we have

$$ND^{-1}\tilde{\Delta v} = -N\overline{\Delta z}. \tag{4.65}$$

Putting (4.62) in (4.65), we obtain

$$-N\overline{\Delta z} = NG^{-1}B^T\tilde{\Delta u} + NG^{-1}N^T\tilde{\Delta w},$$

or, with (4.64),

$$N\left(G^{-1} - G^{-1}B^T\left(BG^{-1}B^T\right)^{-1}BG^{-1}\right)N^T\tilde{\Delta w} = -N\overline{\Delta z}. \tag{4.66}$$

In order to ensure that this system has a unique solution, we need the following assumption:

A.8 *G is second-order nonsingular with respect to $\begin{pmatrix} B^T & N^T \end{pmatrix}^T$.*

This assumption is trivially satisfied if G is second-order sufficient with respect to $\begin{pmatrix} B^T & N^T \end{pmatrix}^T$. We can now write the proposition below.

Theorem 4.3 *Under Assumptions A.6–A.8, the matrix*

$$M = N\left(G^{-1} - G^{-1}B^T\left(BG^{-1}B^T\right)^{-1}BG^{-1}\right)N^T$$

is nonsingular.

Proof. Let

$$J = \begin{pmatrix} G & B^T \\ B & 0 \end{pmatrix}, \quad L = \begin{pmatrix} J & \begin{pmatrix} N^T \\ 0 \end{pmatrix} \\ \begin{pmatrix} N & 0 \end{pmatrix} & 0 \end{pmatrix}.$$

We can factorize L as follows:

$$\begin{pmatrix} J & \begin{pmatrix} N^T \\ 0 \end{pmatrix} \\ \begin{pmatrix} N & 0 \end{pmatrix} & 0 \end{pmatrix} = \begin{pmatrix} I & 0 \\ \begin{pmatrix} N & 0 \end{pmatrix} J^{-1} & I \end{pmatrix} \begin{pmatrix} J & 0 \\ 0 & -\begin{pmatrix} N & 0 \end{pmatrix} J^{-1} \begin{pmatrix} N^T \\ 0 \end{pmatrix} \end{pmatrix} \begin{pmatrix} I & J^{-1}\begin{pmatrix} N^T \\ 0 \end{pmatrix} \\ 0 & I \end{pmatrix}.$$

From Assumption A.8, L is nonsingular, so, from the law of inertia, the matrix

$$-\begin{pmatrix} N & 0 \end{pmatrix} J^{-1} \begin{pmatrix} N^T \\ 0 \end{pmatrix} \tag{4.67}$$

is also nonsingular. Note that (4.67) is nothing else than the Schur complement of J in L.

Since the inverse of J is

$$\begin{pmatrix} G^{-1} - G^{-1}B^T\left(BG^{-1}B^T\right)^{-1}BG^{-1} & G^{-1}B^T\left(BG^{-1}B^T\right)^{-1} \\ \left(BG^{-1}B^T\right)^{-1}BG^{-1} & -\left(BG^{-1}B^T\right)^{-1} \end{pmatrix},$$

we have

$$\begin{pmatrix} N & 0 \end{pmatrix} J^{-1} \begin{pmatrix} N^T \\ 0 \end{pmatrix} = N[G^{-1} - G^{-1}B^T(BG^{-1}B^T)^{-1}BG^{-1}]N^T,$$

Therefore, the matrix

$$N[G^{-1} - G^{-1}B^T(BG^{-1}B^T)^{-1}BG^{-1}]N^T$$

is nonsingular. □

The last system to solve in order to obtain the solution of (4.35) then becomes:

$$\begin{pmatrix} H & B^T & -I \\ B & 0 & 0 \\ -I & 0 & -D^{-1} \end{pmatrix} \begin{pmatrix} \Delta z \\ \Delta u \\ \Delta v \end{pmatrix} = -\begin{pmatrix} g(z) + B^T u - v + N^T(w + \Delta w) \\ 0 \\ -Ze + \mu V^{-1}e \end{pmatrix}. \tag{4.68}$$

Solution of the second system of the decomposition

As said before, the second system, while large, has an exploitable sparse structure, as shown in the following theorem.

Theorem 4.4 *Suppose that N is selected such that $\begin{pmatrix} B^T & N^T \end{pmatrix}^T$ has full row rank. Then the coefficient of the linear equation* (4.66) *is a symmetric tri-block-diagonal matrix, or a block-diagonal matrix whose blocks are symmetric tri-block diagonal submatrices.*

Proof. The proof is similar to Liu and Sun [86]. We consider three cases.

1. The matrix N has the same form than

$$\begin{pmatrix} N^{(1)} & -N^{(1)} & & & \\ & N^{(2)} & -N^{(2)} & & \\ & & \ddots & \ddots & \\ & & & N^{(S-1)} & -N^{(S-1)} \end{pmatrix},$$

with all $N^{(i)}$ ($i = 1, \ldots, S-1$) having n columns. Let

$$F = G^{-1} - G^{-1}B^T(BG^{-1}B^T)^{-1}BG^{-1}.$$

P can be decomposed as

$$F = \text{diag}(F^{(1)}, \ldots, F^{(S)}),$$

where, for $s = 1, \ldots, S$,

$$F^{(s)} = {G^{(s)}}^{-1} - {G^{(s)}}^{-1}{B^{(s)}}^T(B^{(s)}{G^{(s)}}^{-1}{B^{(s)}}^T)^{-1}B^{(s)}{G^{(s)}}^{-1}.$$

Let $C^{(i)} = F^{(i)} + F^{(i+1)}$. It is then easy to check that

$$NFN^T = \begin{pmatrix} N^{(1)}C^{(1)}{N^{(1)}}^T & -N^{(1)}F^{(2)}{N^{(2)}}^T & & & \\ -N^{(2)}F^{(2)}{N^{(1)}}^T & N^{(2)}C^{(2)}{N^{(2)}}^T & -N^{(2)}F^{(3)}N_3^T & & \\ & \ddots & \ddots & \ddots & \\ & & -N^{(S-1)}F^{(S-1)}{N^{(S-2)}}^T & N^{(S-1)}C^{(S-1)}{N^{(S-1)}}^T \end{pmatrix}, \tag{4.69}$$

which is a symmetric tri-block-diagonal matrix.

2. The matrix N is partitioned into the form

$$\begin{pmatrix} N^{(1)} & -N^{(1)} & & & & & & & \\ & \ddots & \ddots & & & & & & \\ & & N^{(i-1)} & -N^{(i-1)} & & & & & \\ & & & & N^{(i+1)} & -N^{(i+1)} & & & \\ & & & & & \ddots & \ddots & & \\ & & & & & & N^{(S-1)} & -N^{(S-1)} \end{pmatrix}.$$

In this case, we partition F into diag$(\overline{F}_1, \overline{F}_2)$ with $\overline{F}_1 = \text{diag}(F^{(1)}, \ldots, F^{(i)})$ and $\overline{F}_2 = \text{diag}(F^{(i+1)}, \ldots, F^{(S)})$. Then NFN^T is a block-diagonal matrix with two blocks being tri-block-diagonal submatrices respectively.

3. The matrix N has the form

$$\begin{pmatrix} N^{(1)} & -N^{(1)} & & & & & & & & \\ & \ddots & \ddots & & & & & & & \\ & & N^{(i-1)} & -N^{(i-1)} & & & & & & \\ & & & & 0 \cdots 0 & N^{(j+1)} & -N^{(j+1)} & & & \\ & & & & & & \ddots & \ddots & & \\ & & & & & & & N^{(S-1)} & -N^{(S-1)} \end{pmatrix}.$$

By permuting the positions of columns of N, we may have $N' = (\overline{N}, 0)$, where $\overline{N}$ has the same form as IN case 2. Correspondingly, F is changed as $F' = \text{diag}(\overline{F}_1, \overline{F}_2, \overline{F}_3)$ where we have

$$\overline{F}_1 = \text{diag}(F^{(1)}, \ldots, F^{(i)}),\ \overline{F}_2 = \text{diag}(F^{(j+1)}, \ldots, F^{(S)}),$$

and

$$\overline{F}_3 = \text{diag}(F^{(i+1)}, \ldots, F^{(j)}).$$

Then

$$NFN^T = N'F'N'^T = \overline{N}\text{diag}(\overline{F}_1, \overline{F}_2)\overline{N}^T.$$

Thus, solution to this system reduces to the solution of case 2.

□

Therefore, without loss of generality, we have to consider the solution of a linear system involving a tri-block-diagonal system of the form (4.69).

Applying Proposition 4.3 to the subsystem defined by considering the k first scenarios, $k = 1, \ldots, S$, every principal block submatrix is nonsingular. Therefore NFN^T has a block LU factorization (see for instance Golub and Van Loan [61]). We can then use a block LDL^T factorization, where L is a block unit lower matrix and D is a block diagonal matrix. The factorization can be expressed as follows. Consider the tri-block diagonal matrix

$$R = \begin{pmatrix} H_1 & U_1 & & & \\ U_1^T & H_2 & U_2 & & \\ & U_2^T & H_3 & & \\ & & \ddots & \ddots & \\ & & & U_{j-1}^T & H_j \end{pmatrix}.$$

Then

$$R = \begin{pmatrix} I_1 & & & \\ U_1^T Q_1^{-1} & I_2 & & \\ & \ddots & \ddots & \\ & & U_{j-1}^T Q_{j-1} & I_j \end{pmatrix} \begin{pmatrix} Q_1 & & & \\ & Q_2 & & \\ & & \ddots & \\ & & & Q_J \end{pmatrix} \begin{pmatrix} I_1 & Q_1^{-1}U_1 & & \\ & I_2 & Q_2^{-2}U_2 & \\ & & \ddots & \ddots \\ & & & I_J \end{pmatrix},$$

where $Q_1 = H_1$, $Q_i = H_i - U_{i-1}^T Q_{i-1}^{-1} U_{i-1}$, $i = 2, \ldots, J$.

However, in the nonlinear case, the LDL^T can be numerically instable, so it is preferable to use other space approaches, as the multifrontal method implemented in MA27 from the Harwell Subroutine Library (HSL) when solving the linear system (4.66). Note that MA27 can also be used when solving systems (4.54) and (4.68).

Solution of the primal-dual subsystem

We now summarize our decomposition strategy.

Algorithm 4.4: Primal-dual system solution

Step 1. Solve the system (4.54) in parallel.

Step 2. Solve the system (4.66) in order to get $\tilde{\Delta w}$, and compute Δw by

$$\Delta w = \tilde{\Delta w} - w.$$

Step 3. Solve the system (4.68) in parallel.

4.4 Practical aspects

We conclude the algorithm presentation by considering some practical aspects for its implementation. First of all we will examine how to compute a suitable starting point. After this, we will discuss how to initialize and update quantities involved in the algorithm.

4.4.1 Starting point

Another benefit of the QR-factorization as described in Section 4.2.1 is that we can then easily compute a feasible point with respect to the equality constraints. Taking the factorization (4.14), and the decomposition of Q as in (4.15), the point

$$\overline{z} = DR^{-T}b \tag{4.70}$$

is feasible (see Nocedal and Wright [104]). However, such a point do not necessarily satisfy nonnegativity constraints (4.30). One possibility is to move the point along the null space of the equality constraints, but it is not trivial to determine it.

We adopt here another strategy, as in Conn, Gould, Orban, and Toint [30]. Considering problem (4.5)–(4.8), where the objective function is set to zero, we obtain:

$$\min 0 \tag{4.71}$$

$$\text{s.t. } Bz = b, \tag{4.72}$$

$$Nz = 0, \tag{4.73}$$

$$z \geq 0. \tag{4.74}$$

Every solution of this problem is feasible for the preprocessed problem (4.5)–(4.8), and the original one as soon as Assumption A.5 on feasibility is fulfilled.

Problem (4.71)–(4.74) is linear, so we can use any linear interior point method. However, it is more efficient to use some centred damped Newton methods. These leads the iterates towards the central path, that plays a vital role in interior point methods (Nocedal and Wright [104]). Conn, Gould, Orban, and Toint [30] have adapted the method proposed by Zhang [144]. Here we use the algorithm of González, Tapia and Potra [64]. Both converge to the analytic centre once a feasible point has been found. However, in the event that the size of the iterate exceeds some prescribed upper bound, the last point with a norm smaller than this bound is taken for the initial point.

Note that the primal-dual system associated to (4.71)–(4.74) is

$$\begin{pmatrix} 0 & B^T & -I & N^T \\ B & 0 & 0 & 0 \\ -I & 0 & D_{k,j}^{-1} & 0 \\ N & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} \Delta z_{k,j} \\ \Delta u_{k,j} \\ \Delta v_{k,j} \\ \Delta w_{k,j} \end{pmatrix} = - \begin{pmatrix} B^T u_{k,j} - v_{k,j} + N^T w_{k,j} \\ B z_{k,j} - b \\ -Z_{k,j} e + \mu V_{k,j} e \\ N z_{k,j} \end{pmatrix}. \tag{4.75}$$

Since the objective function is constant, the Hessian is zero and $G_k = D_k$. Therefore G_k is positive definite and Assumptions A.6 and A.7 are trivially satisfied. In the implementation of the algorithm proposed by González, Tapia and Potra, we can again benefit from the previously described decomposition.

4.4.2 Initialization and updating strategies

While the implementation of the method is not yet completed, we propose in this section some possible practical choices, based on the papers of Conn, Gould, and Toint [34] and Conn, Gould, Orban, and Toint [30].

The initial dual variables v_0 are simply those calculated at the analytic centre z_0, while the initial value of the barrier parameter is the smallest power of 10 larger than $\langle z_0, v_0 \rangle / n$. The barrier parameter can be updated as follows

$$\mu_{k+1} = \min \left(0.1\mu_k, \mu_k^{1.5} \right).$$

Such a scheme allows a slower reduction of μ_k as it approaches zero, while satisfying (4.17). The forcing functions which control the inner-iteration convergence are defined to be

$$\epsilon^C(\mu) = \epsilon^D(\mu) = \mu^{1.01} = -\epsilon^E(\mu).$$

Values $\eta_1 = 0.01$, $\eta_2 = 0.9$ are used to accept and reject steps in the inner-iteration, and the trust-region radius can be updated according to the usual rule

$$\Delta_{k,j+1} = \begin{cases} \min \left[10^{20}, \max \left(2\|\Delta z_{k,j}\|, \Delta_{k,j} \right) \right] & \text{if } \rho_{k,j} \geq \eta_2, \\ \Delta_{k,j} & \text{if } \rho_{k,j} \in [\eta_1, \eta_2), \\ \frac{1}{2}\Delta_{k,j} & \text{if } \rho_{k,j} < \eta_1; \end{cases}$$

the initial radius for each inner iteration is $\Delta_{k,0} = 1000\mu_k$.

A key point of the algorithm is the computation of some approximation $H_{k,j}$ of the Hessian $\nabla^2_{zz} f(z)$. This approximation has to respect some properties, as the second-order sufficiency of $G_{k,j}$ with respect to $\begin{pmatrix} B^T & N^T \end{pmatrix}^T$. A possible approach is to first set $H^{(s)}_{k,j} = \nabla^2_{z^{(s)}z^{(s)}} f^{(s)}\left(z^{(s)}\right)$, $s = 1, \ldots, S$. The multifrontal scheme used in MA27 (see, Duff and Reid [46]) can cope with large, sparse systems and reports the inertia of the coefficient matrix. If, for some s, MA27 reports that $G^{(s)}_{k,j}$ is not second-order sufficient or if $G^{(s)}_{k,j}$ is singular, we use the naive expedient of replacing $G^{(s)}_{k,j}$ by $G^{(s)}_{k,j} = G^{(s)}_{k,j} + \left\| G^{(s)}_{k,j} \right\| I$. More sophisticated strategies can be considered, and are subject to investigations.

4.5 Conclusion

We have constructed in this chapter a feasible primal-dual interior point algorithm. As in classical interior point methods, the main numerical costs lie in the computation of steps of the inner iterations. Such a step can be obtained along a dogleg path, whose calculation can be organized to exhibit strong parallelism properties when applied to a stochastic nonlinear non-convex multistage stochastic program, in particular when solving the linear system giving the quasi-Newton direction, for which a new decomposition scheme is proposed.

We use a well-studied trust-region algorithm, developed by Conn, Gould, Orban, and Toint [30], as the general framework for our method, so the global convergence is ensured. We have also shown that the decomposition scheme could be exploited when computing a starting feasible point. More numerical experimentation is nevertheless required to assess the gains in computation times that can be achieved by the parallelization.

Chapter 5

Monte Carlo samplings

Monte Carlo methods are well-known tools in stochastic programming for the case where the random variables are either discrete with a large number of possible realizations, or continuous. Usually the optimization process is assumed to produce a global minimum in all the feasible set (see for instance Shapiro [126]), a first-order critical point (Gürkan, Özge and Robinson [68, 69], Shapiro [127]) or a solution in a complete local minimizing set with respect to some nonempty open bounded set (Robinson [114]).

In this chapter we will examine nonlinear nonconvex programs for which a global minimizer is very difficult to obtain, so our hope is limited in finding second-order critical points. We introduce our approach by reexamining consistency results when only first-order critical points are considered. We next consider second-order criticality and show that when the sample size tends to infinity, approximating local solutions may have limit points that are not (local) solutions of the true problem. This leads us to develop new conditions under which second-order properties are preserved for limit points. Such a task is indeed valuable since there may be more than one solution in the nonconvex case, and they often do not share the same constraint qualification properties; moreover this avoids the need to consider the complete set of minimizers of the true problem by focusing only on the limit points of the computed approximating solutions.

5.1 True and SAA problems

We consider the problem

$$\min_{z \in S} g(z) = E_{\boldsymbol{\xi}}[G(z, \boldsymbol{\xi})], \tag{5.1}$$

where $z \in \mathbb{R}^m$ is a vector of decision variables, S is a compact subset of $\mathbb{R}^m$ representing feasible solutions of the above problem, $\boldsymbol{\xi}$ is a random vector defined on the probability space $(\Xi, \mathcal{F}, P)$ and $G : \mathbb{R}^m \times \Xi \to \mathbb{R}$ is a real valued function. We assume that, for every $z \in S$, the expected value function $g(z)$ is well defined, i.e. that the function $G(z, \cdot)$ is $\mathcal{F}$-measurable and $P_{\boldsymbol{\xi}}$-integrable. For simplicity, we restrict ourselves in a first step to the case where the set S is deterministic and convex.

If the distribution function of $\boldsymbol{\xi}$ is continuous or discrete with a large number of possible realizations, $g(z)$ is usually very difficult to evaluate. Therefore solving problem (5.1) becomes difficult and we have to turn to approximations such as Monte Carlo methods (see Shapiro [126, 127]

for a review). In these methods, the original problem (5.1) is replaced by successive approximations obtained by generating samples $\xi_1, \ldots, \xi_N$. The approximation for a sample of size N is

$$\min_{z \in S} \hat{g}_N(z) = \frac{1}{N} \sum_{i=1}^{N} G(z, \xi_i). \tag{5.2}$$

We refer to (5.1) and (5.2) as the true (or expected value) and the sample average approximation (SAA) problems, respectively.

5.2 First-order convergence

We now investigate the convergence of the solutions and optimal values of the sequence of SAA problems (5.2) to a solution and optimal value of (5.1) for N approaching infinity. We first introduce the basic concepts used in this chapter by reviewing first-order convergence.

5.2.1 Deterministic and convex constraints

Let z_N^* be a first-order critical point of $\hat{g}_N(\cdot)$ as defined in (5.2) and consider the sequence

$$\{z_N^*\}_{N=1}^{\infty},$$

constructed by taking larger and larger samplings. Since S is a compact set, this sequence has some limit point z^*. Without loss of generality, we identify the sequence $\{z_N^*\}_{N=1}^{\infty}$ with one of its subsequences converging to z^* from now on. Our first aim is to show that, under reasonable assumptions, z^* is a first-order critical point for the true problem (5.1).

We now state our assumptions.

A.9 *The random draws are independently and identically distributed.*

A.10 *For $P_{\boldsymbol{\xi}}$-almost every ξ, the function $G(\cdot, \xi)$ is continuously differentiable on S.*

A.11 *The family $G(z, \xi)$, $z \in S$, is dominated by a $P_{\boldsymbol{\xi}}$-integrable function $K(\xi)$, i.e. $E_{\boldsymbol{\xi}}[K(\boldsymbol{\xi})]$ is finite and $|G(z, \xi)| \leq K(\xi)$ for all $z \in S$ and $P_{\boldsymbol{\xi}}$-almost every ξ.*

A.10 obvioulsy implies that $G(\cdot, \xi)$ is continuous almost surely. This fact and **A.11** are typical assumptions of stochastic programming theory (see Rubinstein and Shapiro [120], for instance). The stronger form of **A.10** is justified by our interest in first-order optimality conditions, which are expressed in terms of the objective function's gradient.

It is important to note (see [120] again) that **A.9**–**A.11** together imply that there exists a uniform law of large numbers (ULLN) on S, for the approximation $\hat{g}_N(z)$ of $g(z)$, that is

$$\sup_{z \in S} |\hat{g}_N(z) - g(z)| \xrightarrow{a.s.} 0 \quad \text{as } N \to \infty.$$

They also imply that $g(z)$ is then continuous on S.

The ULLN property corresponds to the stochastic version of the uniform convergence of a sequence of functions. Therefore, we have the following lemma

Lemma 5.1 *Assume that **A.9–A.11** hold. Then*

$$\hat{g}_N(z_N^*) \xrightarrow{a.s.} g(z^*). \tag{5.3}$$

Furthermore, if $f(\cdot)$ is a continuous function defined on some convex domain that includes $\hat{g}_N(z_N^*)$ ($N = 1, \ldots, \infty$) and $g(z^*)$, then

$$f(\hat{g}_N(z_N^*)) \xrightarrow{a.s.} f(g(z^*)). \tag{5.4}$$

If $\hat{h}_N(\cdot)$ ($N = 1, \ldots, \infty$), and $h(\cdot)$ are functions such that

$$\hat{h}_N(z_N^*) \xrightarrow{a.s.} h(z^*),$$

then, for any real scalar a,

$$a\hat{h}_N(z_N^*) + \hat{g}_N(z_N^*) \xrightarrow{a.s.} ah(z^*) + g(z^*), \tag{5.5}$$

and

$$\hat{h}_N(z_N^*)\hat{g}_N(z_N^*) \xrightarrow{a.s.} h(z^*)g(z^*). \tag{5.6}$$

Proof. We first prove (5.3), or, more precisely, that

$$\forall \epsilon > 0, \exists N_1 \text{ s.t. } \forall N > N_1, |\hat{g}_N(z_N^*) - g(z^*)| < \epsilon \quad \text{a.s.} \tag{5.7}$$

The triangle inequality gives that

$$|\hat{g}_N(z_N^*) - g(z^*)| \leq |\hat{g}_N(z_N^*) - g(z_N^*)| + |g(z_N^*) - g(z^*)|\,. \tag{5.8}$$

From the ULLN property, we deduce that $\sup_{z\in S} |\hat{g}_N(z) - g(z)| \xrightarrow{a.s} 0$. Therefore, there exists some $N_1 > 0$ such that for all $N > N_1$,

$$|\hat{g}_N(z_N^*) - g(z_N^*)| \leq \sup_{z\in S} |\hat{g}_N(z) - g(z)| < \frac{\epsilon}{2} \quad \text{a.s..} \tag{5.9}$$

Moreover, the continuity of $g(z)$ on S and the fact that z^* is the limit point of $\{z_N^*\}_{N=1}^{\infty}$, together imply that for N large enough, say $N > N_2 > 0$,

$$|g(z_N^*) - g(z^*)| < \frac{\epsilon}{2}. \tag{5.10}$$

Combining (5.8), (5.9) and (5.10), for N larger than $\max(N_1, N_2)$, we have that $|\hat{g}_N(z_N^*) - g(z^*)| < \epsilon$ a.s., which proves (5.7).

We now prove (5.4). Let $\epsilon > 0$. Then, by continuity,

$$\exists\, \delta_\epsilon \text{ s.t. } \forall x, y \in dom(f),\ |x - y| < \delta_\epsilon \Rightarrow |f(x) - f(y)| < \epsilon.$$

But, since $\hat{g}_N(z_N^*)$ and $g(z^*)$ both belong to the domain of f, we have that

$$\exists\, N_\epsilon \text{ s.t. } \forall N > N_\epsilon,\ |\hat{g}_N(z_N^*) - g(z^*)| < \delta_\epsilon, \quad \text{a.s.}$$

Combining these two inequalities, we obtain that

$$\exists\, N_\epsilon \text{ s.t. } \forall N > N_\epsilon,\ |f(\hat{g}_N(z_N^*)) - f(g(z^*))| < \epsilon, \quad \text{a.s.},$$

which is (5.4).

Finally, we prove (5.5) and (5.6). The result (5.5) is immediate; (5.6) follows easily from the following relations:

$$\begin{aligned}
&|h(z_N^*)g(z_N^*) - h(z^*)g(z^*)| \\
&\quad = |(h(z_N^*) - h(z^*))(g(z_N^*) - g(z^*)) + g(z^*)(h(z_N^*) - h(z^*)) + h(z^*)(g(z_N^*) - g(z^*))| \\
&\quad \leq |h(z_N^*) - h(z^*)|\,|g(z_N^*) - g(z^*)| + |g(z^*)|\,|h(z_N^*) - h(z^*)| + |h(z^*)|\,|g(z_N^*) - g(z^*)|.
\end{aligned}$$

□

As first-order conditions are generally expressed in terms of the objective function's gradient, we need a further assumption on this gradient.

A.12 *The gradient components $\frac{\partial}{\partial z_l} G(z, \xi)$ $(l = 1, \ldots, m)$ are dominated by a $P_{\boldsymbol{\xi}}$-integrable function.*

This new assumption allows us to apply the results of Rubinstein and Shapiro [120], page 71, and deduce that the expected value function $g(z)$ is continuous and differentiable at y almost surely, and that expectation can be interchanged in the expression of the gradient, that is

$$\nabla_z g(y) = E\left[\nabla_z G(y, \xi)\right].$$

We then have that

$$E\left[\nabla \hat{g}_N(z^*)\right] = \frac{1}{N}\sum_{i=1}^{N} E\left[\nabla_z G(z^*, \xi^i)\right] = \frac{1}{N}\sum_{i=1}^{N} \nabla_z E\left[G(z^*, \xi^i)\right] = \nabla g(z^*).$$

i.e. $\nabla \hat{g}_N(z^*)$ is an unbiased estimator of $\nabla g(z^*)$. We may now verify that z^* is a first-order critical point.

Theorem 5.1 *Assume that **A.9**–**A.12** hold. Then z^* is a first-order critical point of* (5.1) *almost surely.*

Proof. Since z_N^* is a first-order critical point of $\hat{g}_N(z)$, from Theorem 2.9 on page 30, we have that

$$-\nabla_x \hat{g}_N(z_n^*) \in \mathcal{N}_S(z_N^*),$$

where $\mathcal{N}_S(z_N^*)$ is the normal cone to S at z_N^*. Since S is convex, this implies (see Rockafellar and Wets [116], Section 6.9) that each z_N^* satisfies the following first-order conditions for (5.2), i.e.

$$\langle \nabla_z \hat{g}_N(z_N^*), u - z_N^* \rangle \geq 0, \text{ for all } u \in S. \tag{5.11}$$

From Lemma 5.1, applied to the derivatives, we have $\nabla_z \hat{g}_N(z_N^*) \xrightarrow{a.s.} \nabla_z g(z^*)$. Therefore, taking the limit as N tends to infinity in (5.11), we obtain that

$$\langle \nabla_z g(z^*), u - z^* \rangle \geq 0, \text{ for all } u \in S, \quad \text{a.s..}$$

In other terms, $-\nabla_z g(z^*) \in \mathcal{N}_S(z^*)$ almost surely. Thus z^* is a first-order critical point for (5.1) almost surely, as desired. □

The previous result can also be derived from stochastic variational inequalities, as presented in Shapiro [127]. Consider a mapping $\Phi : \mathbb{R}^n \times \Xi \to \mathbb{R}^n$ and a multifunction $\Gamma : \mathbb{R}^n \rightrightarrows \mathbb{R}^n$. Suppose that the expectation $\phi(z) := E_{\boldsymbol{\xi}}[\Phi(z, \boldsymbol{\xi})]$ is well defined. We refer now to

$$\phi(z) \in \Gamma(z) \tag{5.12}$$

as the true, or expected value, generalized equation and say that a point $z^* \in \mathbb{R}^n$ is a solution of (5.12) if $\phi(z^*) \in \Gamma(z^*)$. If $\xi_1, \ldots, \xi_N$ is a random sample, we refer to

$$\hat{\phi}_N(z) \in \Gamma(z) \tag{5.13}$$

as the SAA generalized equation, where $\hat{\phi}_N(z) = N^{-1} \sum_{i=1}^N \Phi(z, \xi_i)$. We denote by S^* and S_N^* the sets of (all) solutions of the true (5.12) and SAA (5.13) generalized equations, respectively.

We then have the following result (Shapiro [127]):

Theorem 5.2 *Let S be a compact subset of $\mathbb{R}^n$ such that $S^* \subset S$. Assume that*

(a) the multifunction $\Gamma(z)$ is closed, that is if $z_k \to z$, $y_k \in \Gamma(z_k)$ and $y_k \to y$, then $y \in \Gamma(z)$,

(b) the mapping $\phi(z)$ is continuous on S,

(c) almost surely for N large enough the set S_N^* is nonempty and $S_N^* \subset S$,

(d) $\hat{\phi}_N(z)$ converges to $\phi(z)$ almost surely uniformly on S as $N \to \infty$.

Then $d(S_N^*, S^*) \to 0$ almost surely as $N \to \infty$.

Consider the particular case $\Gamma(\cdot) = \mathcal{N}_S(\cdot)$. Then $\phi(z^*) \in \Gamma(z^*)$ if and only if

$$\langle \phi(z^*), u - z^* \rangle \leq 0, \ \forall u \in S.$$

Following Shapiro, we refer to such variational inequalities as stochastic variational inequalities. The assumption $\Gamma(z)$ closed always holds for variational inequalities. Take $\Phi(z, \xi) =$

$-\nabla_z G(z,\xi)$ and let S^* and S_N^* represent the set of first-order critical points of the true (5.12) and SAA (5.13) generalized equations, respectively. Then under **A.9**–**A.12**, we have $\phi(z) = -\nabla_z g(z)$, and $\phi(z)$ is continuous on S. Assumption (d) of Theorem 5.2 corresponds to the ULLN, while **A.10** and S compact ensure that Assumption (c) is fulfilled. Therefore, we see that Theorem 5.1 is a particular case of Theorem 5.2.

5.2.2 Stochastic constraints

It is also possible to prove first-order convergence when S is nonconvex or non-deterministic under stronger assumptions. We suppose that the feasible set can be described by equality and inequality constraints. The original problem is then stated as follows

$$\begin{aligned} &\min_{z\in V} g(z) = E_{\boldsymbol{\xi}}[G(z,\boldsymbol{\xi})], \\ &\text{s.t. } c_j(z) \le 0,\ j = 1,\dots,k, \\ &\qquad c_j(z) = 0,\ j = k+1,\dots,M, \end{aligned} \tag{5.14}$$

where V is a compact subset of $\mathbb{R}^n$. The corresponding SAA problem is now defined as

$$\begin{aligned} &\min_{z\in V} \hat{g}_N(z), \\ &\text{s.t. } \hat{c}_{jN}(z) \le 0,\ j = 1,\dots,k, \\ &\qquad \hat{c}_{jN}(z) = 0,\ j = k+1,\dots,M. \end{aligned} \tag{5.15}$$

Here, for every $j = 1,\dots,M$, $\{\hat{c}_{jN}(\cdot)\}$ is a sequence of real-valued (random) functions converging asymptotically in N, to the corresponding function $c_j(\cdot)$. We will assume that the functions $c_j(\cdot)$ can be represented in the form of expected values:

$$c_j(z) = E_{\boldsymbol{\xi}}[H_j(z,\boldsymbol{\xi})],\ j = 1,\dots,M.$$

The expected value functions $c_j(\cdot)$ ($j = 1,\dots,M$) can be estimated by the corresponding sample mean functions

$$\hat{c}_{jN}(z) = \frac{1}{N}\sum_{i=1}^{N} H_j(z,\xi_i).$$

For simplicity, we will consider the more general parametric mathematical programming problem

$$\begin{aligned} &\min_{z\in V} \hat{g}(z,\boldsymbol{\epsilon}), \\ &\text{s.t. } \hat{c}_j(z,\boldsymbol{\epsilon}) \le 0,\ j = 1,\dots,k, \\ &\qquad \hat{c}_j(z,\boldsymbol{\epsilon}) = 0,\ j = k+1,\dots,M, \end{aligned} \tag{5.16}$$

where $\boldsymbol{\epsilon}$ is a vector of parameters giving perturbations of the program (5.16); $g(\cdot)$, $\hat{g}(\cdot,\boldsymbol{\epsilon})$, $c_j(\cdot)$, $\hat{c}_j(\cdot,\boldsymbol{\epsilon})$ are assumed to be twice continuously differentiable.

We will assume that the perturbation is of the form

$$\boldsymbol{\epsilon} = \begin{pmatrix} \boldsymbol{\epsilon}_g & \boldsymbol{\epsilon}_{c_1} & \dots & \boldsymbol{\epsilon}_{c_M} & \boldsymbol{\epsilon}_{\nabla g} & \boldsymbol{\epsilon}_{\nabla c_1} & \dots & \boldsymbol{\epsilon}_{\nabla c_M} \end{pmatrix}^T,$$

and

$$\begin{aligned}\hat{g}(z,\boldsymbol{\epsilon}) &= g(z) + \boldsymbol{\epsilon}_g,\\ \hat{c}_j(z,\boldsymbol{\epsilon}) &= c_j(z) + \boldsymbol{\epsilon}_{c_j},\ j = 1,\dots,M,\\ \nabla_z\hat{g}(z,\boldsymbol{\epsilon}) &= \nabla_z g(z) + \boldsymbol{\epsilon}_{\nabla g},\\ \nabla_z\hat{c}_j(z,\boldsymbol{\epsilon}) &= \nabla_z c_j(z) + \boldsymbol{\epsilon}_{\nabla c_j},\ j = 1,\dots,M.\end{aligned}$$

In the Monte Carlo context, we can define $\boldsymbol{\epsilon}_N$ as

$$\boldsymbol{\epsilon}_N = \begin{pmatrix} \hat{g}_N(z) - g(z) \\ \hat{c}_{jN}(z) - c_j(z),\ j = 1,\dots,M \\ \nabla_z\hat{g}_N(z) - \nabla_z g(z) \\ \nabla_z\hat{c}_{jN}(z) - \nabla_z c_j(z),\ j = 1,\dots,M \end{pmatrix}.$$

We will assume that $\boldsymbol{\epsilon}_N$ converges uniformly on V to 0 almost surely as N tends to infinity. In other terms, we assume that the ULLN holds for the objective and the constraints, as well as for the corresponding derivatives.

The Lagrangian functions associated to (5.14) and (5.16) are respectively

$$\mathcal{L}(z,\lambda) = g(z) + \sum_{j=1}^{M} \lambda_j c_j(z)$$

and

$$L(z,\lambda,\boldsymbol{\epsilon}) = \hat{g}(z,\boldsymbol{\epsilon}) + \sum_{j=1}^{M} \lambda_j \hat{c}_j(z,\boldsymbol{\epsilon}).$$

Let $z^*(\boldsymbol{\epsilon})$ denotes a first-order critical point for program (5.16). Therefore there exists Lagrange multipliers $\lambda^*(\boldsymbol{\epsilon})$ such that $(z^*(\boldsymbol{\epsilon}), \lambda^*(\boldsymbol{\epsilon}))$ satisfy the Karush-Kuhn-Tucker (KKT) conditions; in other terms $(z^*(\boldsymbol{\epsilon}), \lambda^*(\boldsymbol{\epsilon}))$ is solution of the system

$$\begin{aligned}\nabla_z L(z,\lambda,\boldsymbol{\epsilon}) &= 0,\\ \lambda_j\hat{c}_j(z,\boldsymbol{\epsilon}) &= 0, \quad j = 1,\dots,M,\\ \hat{c}_j(z,\boldsymbol{\epsilon}) &= 0, \quad j = k+1,\dots,M,\\ \hat{c}_j(z,\boldsymbol{\epsilon}) &\le 0, \quad j = 1,\dots,k,\\ \lambda_j(\boldsymbol{\epsilon}) &\ge 0, \quad j = 1,\dots,k.\end{aligned}$$

As before, since V is compact, $z^*(\epsilon_N)$ has some limit point z^* as $N \to \infty$, and without loss of generality, we can assume that $z^*(\epsilon_N) \to z^*$ as $N \to \infty$. We can now prove first-order convergence for the general case.

Theorem 5.3 *Assume that*

(a) $\boldsymbol{\epsilon}_N \xrightarrow{a.s.} 0$ *uniformly on V, as* $N \to \infty$,

(b) $\lambda^*(\boldsymbol{\epsilon}_N)$ *has some limit point* λ^* *almost surely as* $N \to \infty$,

Then z^* *is a first-order critical point for* (5.14).

Proof. Consider the sequence $\{(z^*(\boldsymbol{\epsilon}_N), \lambda^*(\boldsymbol{\epsilon}_N))\}$, $N = 1, \ldots, \infty$. From (b), there exists a subsequence converging almost surely to (z^*, λ^*). Our assumption (a) and Lemma 5.1 imply that (z^*, λ^*) satisfies the KKT conditions for the true problem. □

Note that assumption (b) always holds if the multipliers remain bounded. If this stronger assumption is made, it is also possible to use Theorem 5.2 to show that z^* is first-order critical, as in Shapiro [127]. Let $\mu := (z, \lambda) \in \mathbb{R}^{m+M}$ and $\mathcal{K} := \mathbb{R}^m \times \mathbb{R}^k_+ \times \mathbb{R}^{M-k} \subset \mathbb{R}^{m+M}$. Define

$$\phi(\mu) = (\nabla_z \mathcal{L}(z, \lambda), c_{k+1}(z), \ldots, c_M(z)),$$

and

$$\hat{\phi}_N(\mu) = (\nabla_z L(z, \lambda, \boldsymbol{\epsilon}_N), \hat{c}_{k+1}(z, \boldsymbol{\epsilon}_N), \ldots, \hat{c}_M(z, \boldsymbol{\epsilon}_N)).$$

The variational inequality $\phi(\mu) \in \mathcal{N}_K(\mu)$ represents the KKT optimality conditions for the true optimization problem.

5.3 Second-order convergence

5.3.1 Deterministic constraints

We now show that we may verify that z^* is a local minimizer, if we further strengthen our assumptions. We first consider the case where S is deterministic and assume that z^*_N is a local minimum of $\hat{g}_N(x)$. This is to say that

$$\exists\ \delta_N \text{ s.t. } \forall z \in B(z^*_N, \delta_N) \cap S,\ \hat{g}_N(z^*_N) \leq \hat{g}_n(z),$$

where $B(x, d)$ is the open ball centred at x and of radius d. In order to show that z^* is a local minimum of $g(\cdot)$, we must therefore have that the neighbourhood in which z^*_N is a local minimum does not shrink to a singleton when $N \to \infty$. We express this requirement by the following technical assumption.

A.13 *There exists $a > 0$ and $N_a > 0$ such that, for all $N \geq N_a$,*

$$\exists \delta_N \geq a \text{ s.t. } \forall x \in B(z^*_N, \delta_N) \cap S, \hat{g}_N(z^*_N) \leq \hat{g}_n(z).$$

This allows us to write a basic second-order convergence theorem.

Theorem 5.4 *Assume that **A.9–A.13** hold. Then z^* is a local minimum of $g(\cdot)$ almost surely.*

Proof. Let z' be a minimizer of g in $\mathcal{K} = \overline{B}\left(z^*, \frac{a}{2}\right) \cap S$, where a is defined as in **A.13**. Then for N sufficiently large, we have that,

$$\overline{B}\left(z^*, \frac{a}{2}\right) \subset B(z^*_N, \epsilon_N) \text{ and } z^*_N \in \overline{B}\left(z^*, \frac{a}{2}\right). \tag{5.17}$$

Since z^* is the limit point of $\{z_N^*\}_{N=0}^{\infty}$, the second part of (5.17) is satisfied for N sufficiently large. Consider now $z \in \overline{B}(z^*, \frac{a}{2})$. We have that, $|z - z_N^*| \leq |z - z^*| + |z^* - z_N^*|$. Thus

$$|z - z_N^*| \leq \frac{a}{2} + \frac{a}{2} = a \leq \delta_N.$$

Therefore $z \in \overline{B}(z_N^*, \epsilon_N)$, so the first part of (5.17) is also satisfied when N is sufficiently large. We now verify that

$$|\hat{g}_N(z_N^*) - g(z')| \xrightarrow{a.s.} 0. \tag{5.18}$$

Assume first that $\hat{g}_N(z_N^*) \leq g(z')$, then

$$|\hat{g}_N(z_N^*) - g(z')| = g(z') - \hat{g}_N(z_N^*)$$

Since z' minimizes $g(\cdot)$ in $\mathcal{K}$ and, from (5.17), $z_N^* \in K$, we have that

$$g(z') - \hat{g}_N(z_N^*) \leq g(z_N^*) - \hat{g}_N(z_N^*) \leq \sup_{z \in S} |\hat{g}_N(z) - g(z)|.$$

But the ULLN property implies that

$$\sup_{z \in S} |\hat{g}_N(z) - g(z)| \xrightarrow{a.s.} 0,$$

and therefore

$$|g(z') - \hat{g}_N(z_N^*)| \leq 0 \text{ a.s.} \tag{5.19}$$

Assume now that $\hat{g}_N(z_N^*) \geq g(z')$. Since $z' \in K$, from the first part of (5.17), $\hat{g}_N(z_N^*) \leq \hat{g}_N(z')$. Therefore we can write

$$|\hat{g}_N(z_N^*) - g(z')| \leq \hat{g}_N(z') - g(z') \leq \sup_{z \in S} |\hat{g}_N(z) - g(z)|.$$

and we deduce that

$$|g(z') - \hat{g}_N(z_N^*)| \geq 0 \quad \text{a.s.} \tag{5.20}$$

Combining (5.19) and (5.20), we obtain (5.18). From Lemma 5.1, $\hat{g}_N(z_N^*) \xrightarrow{a.s} g(z^*)$, so we have that $g(z') = g(z^*)$. In other terms, z^* is a local solution of (5.1) almost surely. □

Classical results, where global minimizers are considered, express that dist(z_N^*, S^*) converges almost surely to 0 as N tends to infinity, where S^* is the set of minimizers of the true problem (see for instance Theorem 5.2). Robinson [114] shows that, under mild regularity conditions, if the true problem has a complete local minimizing (CLM) set with respect to a nonempty open bounded set $\mathcal{G}$, then for large N, the approximating problem has almost surely a CLM set with respect to $\mathcal{G}$ such that the distance between the CLM set associated to true problem and that of the approximating problem tends to 0 as N tends to infinity. Moreover, the approximating infimum over the closure of $\mathcal{G}$ converges to a finite minimum for the true problem over the closure of $\mathcal{G}$. While this proves the existence of solutions for the approximating problem, Assumption **A.13** reflects that if z_N^* is a local minimizer of the approximating problem, without any other restriction, its distance to the set of true local minimizers can remain strictly positive with a probability that do not converge to zero as N tends to infinity.

Example 5.1. *Consider the problem*

$$\min_{z\in[-1,1]} z^3 - \frac{z}{2}E_{\boldsymbol{\xi}}[\boldsymbol{\xi}], \tag{5.21}$$

where $\Xi = \{-1, 1\}$ *and* $P_{\boldsymbol{\xi}}[\xi = -1] = P_{\boldsymbol{\xi}}[\xi = 1] = 0.5$*, so* $E_{\boldsymbol{\xi}}[\boldsymbol{\xi}] = 0$*. Therefore* (5.21) *has only one local minimizer, which is also global, at* $z^* = -1$*. The SAA problem is then*

$$\min_{z\in[-1,1]} z^3 - \frac{z}{2N}\sum_{i=1}^{N}\xi_i.$$

This last problem has two (isolated) local minimizers,

$$\left\{-1, \sqrt{\frac{\sum_{i=1}^{N}\xi_i}{6N}}\right\},$$

when $\sum_{i=1}^{N}\xi_i > 0$*. When* $N \to \infty$*, we have that* $P_{\boldsymbol{\xi}}\left[\sum_{i=1}^{N}\xi_i > 0\right] \to 0.5$*, and*

$$\sqrt{\frac{\sum_{i=1}^{N}\xi_i}{6N}} \xrightarrow{a.s.} 0,$$

since, from the strong law of large numbers, $\frac{1}{N}\sum_{i=1}^{N}\xi_i \to E_{\boldsymbol{\xi}}[\boldsymbol{\xi}] = 0$ *almost surely as* $N \to \infty$*. But 0 is a saddle point of the true problem* (5.21)*, not a minimizer, even locally, and the distance to* $S^* = \{-1\}$ *is then equal to 1. Note that in this example the ULLN holds for the objective function as well as for all its derivatives.*

It can be shown that in a neighbourhood of a local solution of the true solution, under some mild regularity the SAA has almost surely a solution when N is sufficiently large (Shapiro [127]). However, care must be taken when solving the SAA problem for N fixed since, as illustrated by the previous example, we can find approximating local minimizers that are not close to a true local minimizer.

5.3.2 Stochastic constraints

Assumption **A.13** is somewhat artificial and it is thus of interest to search for more elegant conditions. While our arguments will be similar to those presented in perturbation analysis, as for instance in Rubinstein and Shapiro [120], it is important to note that perturbation analysis assumes the existence of a solution of the true problem and then studies the existence and behaviour of solutions of the perturbed problem in a neighbourhood of this original solution. At variance with this approach, we focus here on conditions under which the limit point of a sequence of approximating solutions is a solution of the true problem. The difference will be more formally illustrated at the end of this section where we compare our developments to some sensitivity analysis results.

We consider the case where the feasible set is described by a set of equality and inequality constraints, as in (5.14). Without loss of generality, we can assume that $z^*(\boldsymbol{\epsilon}_N) \to z^*$ almost surely, as N tends to infinity. If there is a unique Lagrange multipliers vector $\lambda^* \equiv \lambda^*(0)$ associated to z^* that satisfies the KKT conditions, then under some conditions $\lambda^*(\boldsymbol{\epsilon}_N) \to \lambda^*$ almost surely as $N \to \infty$.

Lemma 5.2 *Assume that the Jacobian of*

$$\begin{aligned} \nabla_z \mathcal{L}(z, \lambda) &= 0, \\ \lambda_j c_j(z) &= 0, \quad j = 1, \ldots, M, \\ c_j(z) &= 0, \quad j = k+1, \ldots, M, \end{aligned} \tag{5.22}$$

is well defined at (z^*, λ^*) and nonsingular. Then $\lambda^*(\boldsymbol{\epsilon}_N)$ converges almost surely to λ^* as N tends to infinity.

Proof. Consider the linear system

$$\begin{aligned} \nabla_z L(z, \lambda, \boldsymbol{\epsilon}) &= 0, \\ \lambda_j \hat{c}_j(z, \boldsymbol{\epsilon}) &= 0, \quad j = 1, \ldots, M, \\ c_j(z, \boldsymbol{\epsilon}) &= 0, \quad j = k+1, \ldots, M. \end{aligned} \tag{5.23}$$

From our initial assumptions, this system is continuously differentiable in all the arguments. Moreover, the Jacobian of the left-hand side with respect to (z, λ) is invertible at (z^*, λ^*).

From the implicit function theorem (see for instance Fiacco [50], page 36), in a neighbourhood $\mathcal{B}$ of $(z^*, \lambda^*, 0)$, there exists a unique continuously differentiable function $\gamma(\boldsymbol{\epsilon}) = (z^*(\boldsymbol{\epsilon}), \lambda^*(\boldsymbol{\epsilon}))$ satisfying (5.23) with $(z^*(0), \lambda^*(0)) = (z^*, \lambda^*)$. Since $z^*(\boldsymbol{\epsilon}_N) \to z^*$ almost surely, $z^*(\boldsymbol{\epsilon}_N)$ belongs almost surely to $\mathcal{B}$ for N large enough. But $(z^*(\boldsymbol{\epsilon}_N), \lambda^*(\boldsymbol{\epsilon}_N))$ satisfies (5.23), so from the unicity and continuity of $\gamma(\boldsymbol{\epsilon})$, $\lambda^*(\boldsymbol{\epsilon}_N)$ tends to λ^* almost surely as N tends to infinity. □

Therefore, $\nabla L(z^*(\boldsymbol{\epsilon}_N), \lambda^*(\boldsymbol{\epsilon}_N))$ converges almost surely to $\nabla \mathcal{L}(z^*, \lambda^*)$, when N tends to infinity. The unicity of λ^* can be ensured with a suitable constraint qualification, as the linear independence constrained qualification, which is particularly convenient for our discussion.

Definition 5.1: Strict complementarity
Given z^* and a vector λ^* satisfying the KKT conditions, we say that the strict complementarity condition holds if exactly one of λ_j^* and $c_j(z^*)$ is zero for each index $j = 1, \ldots, k$. In other words, we have that $\lambda_j^* > 0$ for each $j \in \{1, \ldots, k\} \cap \mathcal{A}(z^*)$.

Assume that the strict complementarity condition and the LICQ hold at (z^*, λ^*) for the program (5.16). If $\lambda^*(\boldsymbol{\epsilon}_N) \to \lambda^*$ almost surely, we obtain that almost surely $\lambda_N(\boldsymbol{\epsilon}_N)$, $j \in J(z^*)$, are strictly positive and hence the corresponding constraints are active at $z^*(\boldsymbol{\epsilon}_N)$ for all N large enough. Moreover, since $\hat{c}_j(z^*(\boldsymbol{\epsilon}_N), \boldsymbol{\epsilon}_N) \to c(z^*)$ almost surely, we have that for all N large enough, almost surely, $\mathcal{A}(z^*(\boldsymbol{\epsilon}_N)) = \mathcal{A}(z^*)$ and strict complementarity condition holds at $z^*(\boldsymbol{\epsilon}_N)$, associated to the Lagrangian multipliers $\lambda^*(\boldsymbol{\epsilon}_N)$, for the program (5.15).

Theorem 5.5 (Second-order convergence) *Assume that λ^* is the unique vector of Lagrangian multipliers associated to the program* (5.14) *at z^*, and that*

(a) $\boldsymbol{\epsilon}_N \xrightarrow{a.s.} 0$ *uniformly on V, as* $N \to \infty$,

(b) $z^*(\boldsymbol{\epsilon}_N) \xrightarrow{a.s.} z^*$, $\lambda^*(\boldsymbol{\epsilon}_N) \xrightarrow{a.s.} \lambda^*$ *as* $N \to \infty$,

(c) $\nabla^2_{zz}\hat{g}(z^*(\boldsymbol{\epsilon}_N), \boldsymbol{\epsilon}_N) \xrightarrow{a.s.} \nabla^2_{zz} g(z^*)$ *as* $N \to \infty$,

(d) $\nabla^2_{zz}\hat{c}_j(z^*(\boldsymbol{\epsilon}_N), \boldsymbol{\epsilon}_N) \xrightarrow{a.s.} \nabla^2_{zz} c_j(z^*)$ $(j = 1, \ldots, M)$ *as* $N \to \infty$.

Suppose also that the strict complementarity condition and the LICQ hold at (z^*, λ^*) for (5.14). *Then, almost surely,*

(i) the LICQ holds at $(z^*(\boldsymbol{\epsilon}_N), \lambda^*(\boldsymbol{\epsilon}_N))$,

(ii) (z^*, λ^*) satisfy the second-order necessary condition for (5.14)*:*

$$w^T \nabla^2_{zz}\mathcal{L}(z^*, \lambda^*)w \geq 0, \text{ for all } w \in \mathit{Null}[\nabla_z \hat{c}_j(z^*)^T]_{j\in\mathcal{A}(z^*)}, \ \|w\| = 1. \tag{5.24}$$

If furthermore there exists some $\alpha > 0$ such that for all N large enough

$$w^T \nabla^2_{zz} L_N(z^*_N, \lambda^*_N)w > \alpha, \text{ for all } w \in \mathit{Null}[\nabla_z \hat{c}_j(z^*(\boldsymbol{\epsilon}_N))^T]_{j\in\mathcal{A}(z^*(\boldsymbol{\epsilon}_N))}, \ \|w\| = 1,$$

then z^*, associated to λ^*, satisfies the second-order sufficient conditions for program (5.14)*:*

$$\textit{(iii)}\ w^T \nabla^2_{zz}\mathcal{L}(z^*, \lambda^*)w > 0, \text{ for all } w \in \mathit{Null}[\nabla_z c_j(z^*)^T]_{j\in\mathcal{A}(z^*)}, \ \|w\| = 1. \tag{5.25}$$

In other terms v^* is an isolated local minimizer of (5.14).

Proof. In order to show (i), consider the matrix formed by the gradients of active constraints at z^* for (5.14):

$$\{\nabla_z c_j(z^*)\}_{j\in\mathcal{A}(z^*)}. \tag{5.26}$$

From the strict complementarity conditions and convergence of Lagrange multipliers, the active set of program (5.15) at z^*_N is asymptotically the same as the active set of program (5.14) at z^*, almost surely. Since $\boldsymbol{\epsilon}_N \to 0$ uniformly on V, almost surely, we have that the matrix formed by the active constraints of the perturbed problem,

$$\{\nabla_z \hat{c}_j(z^*(\boldsymbol{\epsilon}_N), \boldsymbol{\epsilon}_N)\}_{j\in\mathcal{A}(z^*(\boldsymbol{\epsilon}_N))} \tag{5.27}$$

converges to (5.26) almost surely as N tends to infinity:

$$\{\nabla_z \hat{c}_j(z^*(\boldsymbol{\epsilon}_N), \boldsymbol{\epsilon}_N)\}_{j\in\mathcal{A}(z^*(\boldsymbol{\epsilon}_N))} \xrightarrow{a.s.} \{\nabla_z c_j(z^*)\}_{j\in\mathcal{A}(z^*)}. \tag{5.28}$$

The LICQ amounts to say that any square submatrix of the Jacobian of (5.26) is nonsingular. From (5.28), the same is true almost surely for (5.27) for N large enough. Thus the LICQ holds almost surely for the approximating problems when N is sufficiently large.

We now show (ii). From (5.28), we may associate a basis K_N with the null space of (5.27) such that

$$K_N \xrightarrow{a.s.} K, \tag{5.29}$$

where K is a basis of Null$[\nabla_z c_j(z)^T]_{j\in\mathcal{A}(z^*)}$ (see Gill and al [60]).

$(z^*(\boldsymbol{\epsilon}_N), \lambda^*(\boldsymbol{\epsilon}_N))$ satisfy the second-order necessary conditions, that can be expressed as, from the strict complementarity conditions and LICQ,

$$K_N^T \nabla^2_{zz} L(z^*(\boldsymbol{\epsilon}_N), \lambda^*(\boldsymbol{\epsilon}_N), \boldsymbol{\epsilon}_N) K_N \text{ is semi-positive definite.}$$

From (5.29) and Assumptions (a)–(d), we have that

$$K_N^T \nabla^2_{zz} L(z^*(\boldsymbol{\epsilon}_N), \lambda^*(\boldsymbol{\epsilon}_N), \boldsymbol{\epsilon}_N) K_N \xrightarrow{a.s.} K^T \nabla^2_{zz} \mathcal{L}(z^*, \lambda^*) K.$$

Therefore we have (5.24) and (5.25). □

Note that LICQ and strict complementarity conditions imply that the minimizer is isolated while the second-order sufficient condition is usually used to characterize strict local minimizers. In other terms, there exists a neighbourhood $\mathcal{N}_S$ of z^* such that z^* is the only local minimizer in $\mathcal{N}_S$. Recall that isolated local minimizers are also strict local minimizers but that the inverse is not always true (Nocedal and Wright [104], page 14). If z^* is a strict but not isolated local minimizer, every neighbourhood of z^* contains other local minimizers than z^*, which are candidates to be limit points of the sequences of solutions of the SAA problems (5.2), as N tends to infinity, so z^* can be difficult to identify.

Recall that we have proved the convergence of the Lagrange multipliers by assuming that the Jacobian of (5.22) is nonsingular. The same assumption can be used to replace the non-degeneracy hypothesis, as shown in the following corollary.

Corollary 5.6
Assume that λ^* is the unique vector of Lagrangian multipliers associated to the rogram (5.14) *at z^*, and that*

(a) $\boldsymbol{\epsilon}_N \xrightarrow{a.s.} 0$ *uniformly on V as $N \to \infty$,*

(b) $z^*(\boldsymbol{\epsilon}_N) \xrightarrow{a.s.} z^*$, *as $N \to \infty$,*

(c) the Jacobian of (5.22) *is nonsingular,*

(d) $\nabla^2_{zz} \hat{g}(z^*(\boldsymbol{\epsilon}_N), \boldsymbol{\epsilon}_N) \xrightarrow{a.s.} \nabla^2_{zz} g(z^*)$ *as $N \to \infty$,*

(e) $\nabla^2_{zz} \hat{c}_j(z^*(\boldsymbol{\epsilon}_N), \boldsymbol{\epsilon}_N) \xrightarrow{a.s.} \nabla^2_{zz} c_j(z^*)$ $(j = 1, \ldots, M)$ *as $N \to \infty$.*

Suppose also that the strict complementarity condition and the LICQ hold at (z^*, λ^*) for (5.14). *Then z^*, associated to λ^*, satisfies the second-order sufficient conditions for program* (5.14)*:*

$$w^T \nabla^2_{zz} \mathcal{L}(z^*, \lambda^*) w > 0, \text{ for all } w \in \text{Null}[\nabla_z c_j(z^*)^T]_{j\in\mathcal{A}(z^*)},\ \|w\| = 1.$$

Proof. From Theorem 5.5, z^* associated to λ^* satisfies the second-order necessary conditions for program (5.14). But (c) implies that

$$w^T \nabla^2_{zz} \mathcal{L}(z^*, \lambda^*) w \neq 0, \text{ for all } w \in \text{Null}[\nabla_z c_j(z^*)^T]_{j\in\mathcal{A}(z^*)},\ \|w\| = 1.$$

Therefore (z^*, λ^*) satisfies the second-order sufficient property for program (5.14). □

The converse of Theorem 5.5 can be obtained from classical results of perturbation analysis (Fiacco [50], Theorem 3.2.2), that we restate for completeness. More developments in the context of stochastic programming can be found in Rubinstein and Shapiro [120] and Shapiro [125].

Theorem 5.7 *Suppose that the following assumptions hold:*

(a) the functions defining (5.16) *are twice continuously differentiable in z and their gradients with respect to z and the constraints are once continuously differentiable in ϵ in a neighbourhood of $(z^*, 0)$,*

(b) the second-order sufficient conditions for a local minimum of (5.16) *hold at z^*, with associated Lagrange multipliers λ^*,*

(c) the LICQ holds at $(z^*, 0)$,

(d) the strict complementarity condition holds at $(z^*, 0)$,

then

(i) z^* is a local isolated minimum of (5.16) *with $\epsilon = 0$ and the associated Lagrange multipliers λ^* are unique,*

(ii) for ϵ in a neighbourhood of 0, there exists a unique, once continuously differentiable vector function $\gamma(\epsilon) = (x(\epsilon), \lambda(\epsilon))^T$ satisfying the second-order sufficient conditions for a local minimum of problem (5.16) *such that $\gamma(0) = (z^*, \lambda^*)^T$, and hence $z(\epsilon)$ is a local isolated minimizer of problem* (5.16) *with associated unique Lagrange multipliers $\lambda(\epsilon)$, and*

(iii) for ϵ near 0, the set of active constraints is unchanged, strict complementarity conditions hold, and the LICQ holds at $z^*(\epsilon)$.

The second-order sufficiency property is now taken as an assumption, so that z^* is in fact assumed to be a local solution. More general results of perturbation analysis can be obtained by using epi-continuity arguments and the concept of complete local minimizing set (Robinson [113, 114]).

5.4 Asymptotic analysis of the optimal value

In this section we briefly discuss the asymptotic behaviour of the optimal values $\hat{g}(z_N^*)$ of the sample average approximations. Our differentiability conditions allow us to extend some results given in Rubinstein and Shapiro [120], at the price of stronger assumption.

We consider here the case where S is deterministic. We will assume that z^* can be seen as an unique global minimizer in some neighbourhood with the following assumption:

A.14 *z^* is an isolated local minimizer of $g(z)$ as defined in* (5.1).

Consider the sample average approximation (5.2) of the true problem (5.1), and as before, without loss of generality, assume that z_N^* converges to z^* almost surely as N tends to infinity. We have the following theorem.

Theorem 5.8 *Assume that **A.9** and **A.14** hold, and that $z_N^* \to z^*$ almost surely. Let $\mathcal{N}$ be a neighbourhood of z^* where z^* is an isolated minimizer of $g(\cdot)$. If*

(a) for every $z \in S$, the function $G(z, \cdot)$ is $\mathcal{F}$-measurable;

(b) for some $z \in S \cap \mathcal{N}$, the expectation $E_{\boldsymbol{\xi}}[G(z, \boldsymbol{\xi})^2]$ is finite;

(c) there exists an $\mathcal{F}$-measurable function $K : \Xi \to \mathbb{R}$ such that $E_{\boldsymbol{\xi}}[K(\boldsymbol{\xi})^2]$ is finite and

$$|g(z_1, \xi) - g(z_2, \xi)| \leq K(\xi)\|z_1 - z_2\|$$

for all $z_1, z_2 \in S \cap \mathcal{N}$ and almost every $\xi \in \Xi$;

(d) for all N sufficiently large, $\hat{g}_N(z_N^*) \leq \hat{g}_N(z)$ for all $z \in \mathcal{N}$ if $z_N^* \in \mathcal{N}$,

then we have that

$$\sqrt{N}(\hat{g}_N(z_N^*) - g(z^*)) \Rightarrow N(0, \sigma^2)$$

almost surely, where

$$\sigma^2 = E_{\boldsymbol{\xi}}[G(z^*, \boldsymbol{\xi})^2] - g(z^*)^2.$$

Proof. Since $z_N^* \to z^*$ almost surely as $N \to \infty$, there exists some N_0 such that for all $N \geq N_0$, z_N^* belongs to $\mathcal{N}$ almost surely. Define $S' = S \cap \mathcal{N}$. z^* can be seen as the unique global minimizer of $g(z)$ over S' and almost surely z_N^* is also a global minimizer of $\hat{g}_N(z)$ over S'. The result then follows directly from Theorem 6.4.3 of Rubinstein and Shapiro [120]. □

Theorem 5.7 is stronger than the classical central limit theorem in the sense that the approximating and the true objectives are not evaluated at the same point. When global solutions are found more detailed results can be derived (Rubistein and Shapiro [120], Section 6.5). They are still valid if for N large enough Assumption (d) of Theorem 5.7 is satisfied. Further research would include reexamination of this assumption and investigation of the complete local minimizers set concept, introduced by Robinson [113], to develop a better framework for local optimization.

5.5 A trust-region algorithm with dynamic accuracy

The SAA problem (5.2)

$$\min_{z \in S} \hat{g}_N(z) = \frac{1}{N} \sum_{i=1}^{N} G(z, \xi_i),$$

can still be expensive to solve, even on modern computers. This is for instance the case for mixed logit models that we present in the next chapters. However, for a fixed level of approximation, defined by the number of random draws, or, in other terms, the sample size, it is possible to

accelerate the first iterates of the used optimization procedure by considering subsets of the sampling instead of its entirety. More precisely, we generate a sample set prior the optimization process, with $N_{\max}$ i.i.d. random draws. At iteration k, we will use a subset of this sample set, by using N_k of the $N_{\max}$ random draws. For simplicity, we then use the N_k first random draws. This therefore implies that $\hat{g}_N$ is a well defined smooth function for each choice of N.

A first question is to measure the approximation accuracy. Let α_δ be the quantile of a $N(0,1)$ associated to some level of signification δ, i.e. $P_{\boldsymbol{\xi}}[-\alpha_\delta \leq X \leq \alpha_\delta] = \delta$, where $X \sim N(0,1)$. We work here pointwise, so we can benefit from the central limit theorem:

$$g(z) - \hat{g}_N(z) \Rightarrow N\left(0, \frac{\sigma^2(z)}{N}\right),$$

where $\sigma^2(z)$ is the variance of g at z. Therefore, the confidence interval for $g(z)$ around $\hat{g}_N(z)$ is

$$[\hat{g}_N(z) - \epsilon_N^\delta(z),\ \hat{g}_N(z) + \epsilon_N^\delta(z)],$$

where $\epsilon_N^\delta(z)$ is given by

$$\epsilon_\delta^N(z) = \alpha_\delta \frac{\sigma(z)}{\sqrt{N}}. \tag{5.30}$$

Typically, one chooses $\alpha_{0.9} \approx 1.64$ or $\alpha_{0.95} \approx 1.96$. In practice, we do not know $\sigma^2(z)$, so we approximate it by its estimator

$$\hat{\sigma}_N^2(z) = \frac{1}{N-1} \sum_{i=1}^{N} (G(z, \xi_i) - \hat{g}_N(z))^2.$$

The variable sample size procedure can be external to the algorithm, or internal. An external approach consists to apply the optimization algorithm repeatedly with samplings of increasing sizes. The algorithm can be expressed more formally as follows.

Step 0. Set $k = 0$, $N_{\max}$ and N_0, with $0 < N_0 \leq N_{\max}$. Define some feasible point $\tilde{z}$.

Step 1. Solve (approximately) $\hat{g}_{N_k}$ with $\tilde{z}$ as the starting point and let $z^*_{N_k}$ be the found solution.

Step 2. If $N_k = N_{\max}$, stop. Otherwise, set N_{k+1} such that $N_k < N_{k+1} < N_{\max}$, and $\tilde{z} = z^*_{N_k}$. Increment k by 1 and go to Step 1.

The major difficulty in this procedure is to quantify the word "approximately" in Step 1. If no care is taken, the resulting algorithm may in fact consume more time than the direct minimization of $\hat{g}_{N_{\max}}$.

On the other hand, the internal approach consists to vary the number of used random draws inside the optimization algorithm. We present here the approach developed for unconstrained optimization by Bastin, Cirillo and Toint [8, 9] in the context of mixed logit models estimation. The proposed algorithm is of the trust-region type (see Section 2.3.3 on page 32), where, for simplicity, we assume that the 2-norm is used at each iteration. Basically if the model approximates the objective function well compared to the accuracy of the objective function itself (which is dependent on the Monte Carlo sample size), we surmise that we could work with a less precise

approximation and therefore reduce the sample size. On the other hand, if the model adequation is poor compared to the precision of the objective function, we increase the sample size in an attempt to correct this deficiency. A formal description of the algorithm follows.

Algorithm 5.1: Basic trust-region algorithm with dynamic accuracy (BTRDA)

Step 0. Initialization. An initial point z_0 and in initial trust-region radius Δ_0 are given. The constants η_1, η_2, γ_1, and γ_2 are also given as in Algorithm 2.2. Set a minimum number of draws $N_{\min} = N^0_{\min}$ and a sample size N_0 satisfying $\|\nabla_\theta \hat{g}_{N_0}(z_0)\| \neq 0$ if $\epsilon_\delta^{N_0}(z_{k+1}) \neq 0$, except if $N_0 = N_{\max}$. Compute $\hat{g}_{N_0}(z_0)$ and set $k = 0$, $t = 0$.

Step 1. Stopping test. Stop if $\|\nabla_\theta \hat{g}_{N_k}(z_k)\| = 0$ and either $N_k = N_{\max}$, or $\epsilon_\delta^{N_k}(z_k) = 0$. Otherwise go to Step 2.

Step 2. Model definition. Define a model $m_k^{N_k}$ of $\hat{g}_{N_k}(\theta)$ in $\mathcal{B}_k$. Compute a new adequate sample size N^+ (see Algorithm 5.2). Set $N^- = N_k$.

Step 3. Step calculation. Compute a step s_k that sufficiently decreases the model $m_k^{N_k}$ and such that $z_k + s_k \in \mathcal{B}_k$. Set

$$\Delta m_k^{N_k} = m_k^{N_k}(z_k) - m_k^{N_k}(z_k + s_k).$$

Step 4. Comparison of decreases. Compute $\hat{g}_{N^+}(z_k + s_k)$ and define

$$\rho_k = \frac{\hat{g}_{N_k}(z_k) - \hat{g}_{N^+}(z_k + s_k)}{\Delta m_k^{N_k}}. \tag{5.31}$$

Step 5. Sample size update. If $\rho_k < \eta_1$ and $N_k \neq N^+$, modify N^- or the candidate sample size N^+ to take into account variance differences (see Algorithm 5.3). Recompute ρ_k.

Step 6. Acceptance of the trial point. If $\rho_k < \eta_1$, define $z_{k+1} = z_k$, $N_{k+1} = N^-$. Otherwise define $z_{k+1} = z_k + s_k$ and set $N_{k+1} = N^+$; increment t by one.

If $N_{k+1} \neq N^{\max}$, $\|\nabla_\theta \hat{g}_{N_{k+1}}(z_{k+1})\| = 0$, and $\epsilon_\delta^{N_{k+1}}(z_{k+1}) \neq 0$, increase N_{k+1} to some size less or equal to $N_{\max}$ such that $\|\nabla_\theta \hat{g}_{N_{k+1}}(z_{k+1})\| \neq 0$ if $N_{k+1} \neq N_{\max}$, and compute $\hat{g}_{N_{k+1}}(z_{k+1})$.

If $N_k = N_{k+1}$ or if sufficient decrease has been observed since the last evaluation of $\hat{g}_{N_{k+1}}$, set $N^{k+1}_{\min} = N^k_{\min}$. Otherwise define $N^{k+1}_{\min} > N^k_{\min}$ (see Algorithm 5.4).

Step 7. Trust-region radius update. Identical to Step 4 of Algorithm 2.2.

Recall that the iteration k is said to be successful if $\rho_k \geq \eta_1$ and very successful if $\rho_k \geq \eta_2$. In this algorithm, the variable t is used to record the number of successful iterations. Note also that the BTR and the BTRDA algorithms coincide if we fix N_k to $N_{\max}$ for all $k \geq 0$.

It is important to keep in mind that at each iteration, we work with a fixed subset of the $N_{\max}$

random draws, and that this subset is the same at each iteration using N_k random draws. A different strategy would be to allow the use of newly generated random draws at each iteration. Such an approach has some advantages, at least theoretically:

- the random draws have not to be kept in memory (or possibly in a file),
- when the iterates are close enough and the same number of draws is used, the variation of the objective function could inform us on the quality of the approximation,
- it is not always possible to reuse the same sample (if for instance the random variable corresponds to a realization of a physical phenomena[1]),
- if the algorithm converges to a second-order critical point, the evaluation of the objective with different sets of random draws may help to detect degenerate cases as in Example 5.1 on page 102.

However, the noise in the objective function makes difficult the identification of an adequate step if we use different random samples, since this step can be smaller than the noise, in particular when we are close to the solution. In order to avoid failure of the optimization procedure, we could then increase the number of random draws, possibly beyond $N_{\max}$, in such a way that the reduction is always (sufficiently) greater than the accuracy. This is for instance the underlying idea of the trust-region algorithm using dynamic accuracy presented in Conn, Gould and Toint [35], Section 10.6, but the strategy developed in Algorithm 5.1 is then no longer applicable since it allows steps smaller than the approximation accuracy. Moreover, the required sample size can then be very large and numerically unmanageable, for instance if the neighbourhood of the current iterate is nearly flat, and we cannot define some maximum sample size $N_{\max}$ as in Algorithm 5.1. Finally, the cost of the sample generation represents another overhead of the optimization process.

5.5.1 The variable sample size strategy

A crucial ingredient to make our algorithm efficient is to design a technique which adapts the number of draws used to the optimality level of the successive iterates. We now outline the proposed design.

Prior to the optimization, the user chooses a maximum sample size $N_{\max}$. A minimum sample size $N^0_{\min}$ is defined to allow estimation of the accuracy; in our tests, we have used $N^0_{\min} = 36$, which has revealed to be a convenient value. We also define $N_0 = \max\{N^0_{\min}, 0.1N_{\max}\}$ if $\|\nabla_\theta \hat{g}_{N_0}(z_0)\| = 0$ and $\epsilon_\delta^{N_0}(z_0) \neq 0$, $N_0 = N_{\max}$ otherwise. The choice of N^+ in Step 3 of Algorithm 2.2 is described below.

Algorithm 5.2: Candidate sample size selection
Define some constants ν_1 and χ_1 such that $\nu_1, \chi_1 \in (0, 1)$. Use (5.30) to estimate the size

[1] Note that if the evaluation of the objective involves some physical measurement, the derivatives are also often difficult to obtain, suggesting the use of derivative-free strategies (see for instance Colson [29] for derivative-free trust-region methods).

needed to obtain a precision equal to the model decrease, that is

$$N^s = \max\left\{ N^k_{\min}, \left\lceil \frac{\alpha_\delta^2 \hat{\sigma}_N(z)}{(\Delta m_k^{N_k})^2} \right\rceil \right\}.$$

Compute the ratio between the model improvement and the estimated accuracy,

$$\tau_1^k = \frac{\Delta m_k^{N_k}}{\epsilon_\delta^{N_k}(z_k)},$$

and the ratio between the current sample size and the suggested sample size for the next iteration:

$$\tau_2^k = \frac{N_k}{\min\{N_{\max}, N^s\}}.$$

Then define

$$N' = \begin{cases} \min\left\{\lceil \chi_1 N_{\max} \rceil, \lceil N^s \rceil\right\} & \text{if } \tau_1^k \geq 1, \\ \min\left\{\lceil \chi_1 N_{\max} \rceil, \lceil \tau_1^k N^s \rceil\right\} & \text{if } \tau_1^k < 1 \text{ and } \tau_1^k \geq \tau_2^k, \\ \lceil \chi_1 N_{\max} \rceil & \text{if } \nu_1 \leq \tau_1^k < 1 \text{ and } \tau_1^k < \tau_2^k, \\ N_{\max} & \text{if } \tau_1^k < \nu_1 \text{ and } \tau_1^k < \tau_2^k. \end{cases}$$

Set $N^+ = \max\{N', N^k_{\min}\}$.

A possible value for χ_1 is 0.5.

If $\tau_1^k \geq 1$, the model decrease is greater or equal to the estimated accuracy, and we then reduce the sample size to the minimum between N^s and $\lceil \chi_1 N_{\max} \rceil$. The idea to use $\lceil \chi_1 N_{\max} \rceil$ comes from the practical observation that enforcing such a decrease in the proposed sample sizes provides better numerical performance.

If $\tau_1^k < 1$ the improvement is smaller than the precision. However, since the sample has been generated before the optimization process, a sufficient improvement during several consecutive iterations may lead to a significant improvement compared to the approximation accuracy, while keeping the computational costs lower than if $N_{\max}$ draws were used. We then consider two cases.

- If $\tau_1^k \geq \tau_2^k$, the ratio between the current sample size and the potential next one is lower than the ratio between the model decrease and the estimated error. If the sample size increases, the error decreases for a similar $\Delta m_j^{N_j}$ ($j \geq k$), and therefore τ_1^k increases. We capitalize on τ_1^k by computing a sample size lower than N^s, such that an increase of order $\epsilon_\delta^{N_k}(z_k)$ would be reached in approximately $\lceil \tau_1^k \rceil$ iterations if τ_1^j is similar τ_1^k for j close to k. We therefore propose to use the minimum between $\lceil \chi_1 N_{\max} \rceil$ and $\lceil \tau_1^k N^s \rceil$ as the new sample size.
- If $\tau_1^k < \tau_2^k$, it may nevertheless be cheaper to continue to work with a smaller sample size, defined again as $\lceil \chi_1 N_{\max} \rceil$. This is why we choose to use this smaller sample size as long as τ_1^k is superior to some threshold ν_1, set to 0.2 in our simulations. Below to

this threshold, we consider that the decrease is too small compared to the log-likelihood accuracy, and we possibly increase the sample size.

If N^+ is not equal to N_k, the computation of

$$\hat{g}_{N_k}(z_k) - \hat{g}_{N^+}(z_k + s_k)$$

in Algorithm 2.2 is affected by the change in approximation variance. This can lead to a small or negative ratio ρ_k, even when the model $m_k^{N_k}$ gives a good prediction for the sample size N^k. In particular, $\hat{g}_{N^+}(\theta)$ can be superior to $\hat{g}_{N_k}(z_k)$ for all θ in a neighbourhood of z_k. It is therefore important to avoid such cases, which motivates the possible redefinition of ρ_k as described in the algorithm below.

Algorithm 5.3: Sample size revision when $\rho_k < \eta_1$ and $N_k \neq N^+$.

If $\rho_k < \eta_1$, compare N^+ and N_k. If $N^+ > N_k$, compute $\hat{g}_{N^+}(z_k)$, $\Delta m_k^{N^+}$ and $\epsilon_\delta^{N^+}(z_k)$, else if $N^+ < N_k$ compute $\hat{g}_{N_k}(z_k + s_k)$. Set N^- to $\max\{N_k, N^+\}$, and redefine

$$\rho_k = \frac{\hat{g}_{N^-}(z_k + s_k) - \hat{g}_{N^-}(z_k)}{\Delta m_k^{N^-}}.$$

While we expect to benefit from smaller sample sizes when we are far from the solution, we ought to be sure that we use a sample size equal to $N_{\max}$ during the final iterations, in order to benefit from the desired accuracy. For this purpose, we increase the minimum sample size when the variable sample size strategy does not provide sufficient numerical gains. This is done as described below.

We first define two $N_{\max}$-dimensional vectors v and l, and, at iteration $k = 0$, set $v(N_0) = \hat{g}_{N_0}(z_0)$, $l(N_0) = 0$, and for $i = 1, \ldots, N_{\max}$, $i \neq N_0$[2], set $v(i) = +\infty$, $l(i) = -1$. At the beginning of iteration k, $v(i) = \hat{g}_i(z_{h(i)})$, where $h(i)$ corresponds to the index of the last iteration for which $N_{h(i)} = i$, and $N_{h(i)-1} \neq N_{h(i)}$ if $h(i) > 0$, or $+\infty$ if the size i has not been used yet. $l(i)$ contains the number of successful iterations until iteration $h(i)$ (included), or -1 if the size i has not been used. Recall also that t contains the total number of successful iterations encountered until the current iteration k (included).

Algorithm 5.4: Minimum sample size update when $N_k \neq N_{k+1}$.
Let $\gamma_3 \in (0, 1]$ be a constant. If

$$v(N_{k+1}) - \hat{g}_{N_{k+1}}(z_{k+1}) \geq \gamma_3 \nu_1 (t - l(N_{k+1})) \epsilon_\delta^{N_{k+1}}(z_{k+1}), \tag{5.32}$$

set $N_{\min}^{k+1} = N_{\min}^k$. Otherwise increase the minimum sample size: set

$$N_{\min}^{k+1} \in \{N_{k+1} + 1, \ldots, N_{\max}\}.$$

Set $l(N_{k+1}) = t$ and $v(N_{k+1}) = \hat{g}_{N_{k+1}}(z_{k+1})$.

[2]We are in fact only interested in $i \geq N_{\min}^0$, since the sample size used at iteration k ($k \geq 0$) has to be greater or equal to $N_{\min}^0$, but allow i to start from 1 for notational convenience.

A convenient value for γ_3 is 0.5. Note that $N_{\min}^{k+1} > N^k$ if (5.32) is not satisfied. Moreover, we have that if $N_k \neq N_{k+1}$, $t - l(N_{k+1}) \geq 1$. This is clearly true if $l(N_{k+1}) = -1$, so without loss of generality, we assume that $l(N_{k+1}) \geq 0$. At the beginning of iteration k, we have $l(N_i) \leq t$, $i = 1, \ldots, N_{\max}$. If $\rho_k \geq \eta_1$, t is incremented by 1 in Step 6 of Algorithm 2.2, so $l(N_{k+1}) < t$ in Algorithm 5.4. If $\rho_k < \eta_1$, from Algorithm 5.3, $N_k < N_{k+1}$ since reductions of sample sizes can only occur at successful iterations. This also implies that $l(N_{k+1}) < l(N_k) \leq t$.

Finally, we note that, if $N_k \neq N_{\max}$, we cannot exclude the pathological case in which z_k is a first-order critical point for $\hat{g}_{N_k}$. If $\epsilon_\delta^{N_k}(z_k) \neq 0$, the algorithm does not stop, but since the model is quadratic, no decrease is achieved if H_k is positive definite. The algorithm would then break down. In order to avoid this situation, we therefore force an increase of N_{k+1} in Step 6 when this situation occurs. In practice however, the gradient norm usually changes slowly in the vicinity of such a critical point, and a small gradient typically leads to a small model decrease, which itself then causes the sample size to increase and $N_{\max}$ is reached before our safeguard is activated.

5.6 Convergence to solutions of the SAA problem

We now consider the formal convergence properties of the BTRDA algorithm for the solution of the SAA problem, and show that they can be derived from results known for general trust-region methods.

5.6.1 Convergence of the sample size

We start by investigating properties of our variable sample size technique and prove the crucial property that N_k converges to $N_{\max}$ as $k \to \infty$, under some regularity assumptions that we now make explicit.

A.15 *For $P_{\boldsymbol{\xi}}$-almost every ξ, the function $G(\cdot, \xi)$ is twice continously differentiable on S.*

A.16 *For all N such that $N_{\min}^0 \leq N \leq N_{\max}$, $\hat{g}_N$ is bounded below on S almost surely.*

A.17 *The Hessian of each SAA objective is uniformly bounded, that is there exists a positive constant κ_1 such that for all z and $N = N_{\min}^0, \ldots, N_{\max}$,*

$$\|\nabla_{zz} \hat{g}_N(z)\| \leq \kappa_1.$$

A.18 *For all k, $m_k^{N_k}$ is twice differentiable on $\mathcal{B}_k$.*

A.19 *For all k, $m_k^{N_k}(z_k) = \hat{g}_{N_k}(z_k)$ and $\nabla_z m_k^{N_k}(z_k) = \nabla_z \hat{g}_{N_k}(z_k)$.*

A.20 *The Hessian of the model remains bounded within the trust-region, that is there exists a positive constant κ_2 such that for all $z \in \mathcal{B}_k$,*

$$\left\| \nabla_{zz} m_k^{N_k}(z) \right\| \leq \kappa_2.$$

Assumption A.15 ensures that, almost surely, $\hat{g}_N$ is twice continuously differentiable on S. These assumptions allows us to show the announced result.

Theorem 5.9 *Suppose that Assumptions A.9, A.15–A.20 hold and that we have*

$$\exists \kappa > 0 \text{ such that } \epsilon_\delta^{N_k}(z_k) \geq \kappa, \tag{5.33}$$

for all k sufficiently large. Then, almost surely, either the algorithm converges in a finite number number of iterations with a final number of random draws equal to $N_{\max}$, or the number of iterations is infinite and there exists some j such that for all iterations i, $i \geq j$, N_i is equal to $N_{\max}$.

Proof. Consider first the finite case. From the stopping criteria in Step 1 of Algorithm 2.2, we cannot stop with a sample size less than $N_{\max}$ as long as (5.33) is fulfilled, so the result is immediate.

Consider now the infinite case. We first prove that the sample size cannot stay fixed at a value $N_1 < N_{\max}$, after what we show that the maximum sample size must be reached and that the sample size is equal to $N_{\max}$ for k large enough.

Assume, for the purpose of obtaining a contradiction, that

$$\exists k_1 \text{ such that } \forall k \geq k_1, N_k = N_{k_1} < N_{\max}. \tag{5.34}$$

For a fixed sample size, Algorithm 2.2 corresponds to the basic trust-region algorithm (Conn, Gould and Toint [35], Chapter 6). Assume first that there are only finitely many successful iterations. Let s be the index of the last successful iterations. Then $z_k = z_{s+1}$ for all $k > s$. From Assumptions A.15–A.20 and our model choice, we can apply Theorem 6.4.4 in Conn, Gould and Toint [35] to deduce that $\|\nabla_z \hat{g}_{N_{s+1}}\| = 0$ almost surely. From Steps 0 and 6 of Algorithm 2.2, either (5.33) is violated or $N_{s+1} = N_{\max}$, and the algorithm stops, violating our assumption that the number of iterations is infinite.

We may therefore assume, without loss of generality, that there is an infinite number of successful iterations. However, from Algorithms 5.2 and 5.3, and (5.33), a necessary condition for $N^+ < N_{\max}$ at iteration k is

$$\Delta m_k^{N_k} \geq \nu_1 \kappa,$$

when $\tau_1^k \geq \nu_1$, or

$$\Delta m_k^{N_k} \geq \kappa \frac{N_{\min}^k}{N_{\max}},$$

when $\tau_1^k \geq \tau_2^k$. Assume that the iteration is successful. Then $N^+ = N_{k+1} = N_{k_1}$ for k large enough and we have from (5.34) that

$$\hat{g}_{N_{k_1}}(z_k) - \hat{g}_{N_{k_1}}(z_{k+1}) \geq \eta_1 \Delta m_k^{N_{k_1}} \geq \eta_1 \min\left\{\nu_1\kappa, \kappa\frac{N_{\min}^0}{N_{\max}}\right\}.$$

Since there is an infinite number of successful iterations, $\hat{g}_{N_{k_1}}(z_k)$ converges to infinity, as $k \to \infty$, but this contradicts the assumption that $\hat{g}_{N_{k_1}}$ is bounded below. We have therefore that

$$\text{if } N_{k_1} < N_{\max}, \text{ then there exists } k_2 > k_1 \text{ such that } N_{k_2} \neq N_{k_1}. \tag{5.35}$$

Assume now by contradiction that

$$\forall k, \text{ there exists } j \geq k \text{ such that } N_j < N_{\max}. \tag{5.36}$$

From Algorithm 5.4, $N_{\min}^k$ increases monotically and is bounded above by $N_{\max}$. Therefore there exists some $N^\# = \lim_{k\to\infty} N_{\min}^k$, with $N^\# \leq N_{\max}$. Since $N_{\min}^k$ is finite for all k, $N^\#$ is reached at some iteration $k_\#$ and $N_{\min}^k = N^{k_\#} < N_{\max}$ for all $k \geq k_\#$. From (5.35) and (5.36), there exists an infinite subsequence of iterations such that $N_{k+1} \neq N_k$. Let $m \geq k_\#$ be the index of such an iteration. From Algorithm 5.4 and (5.33) we have that

$$v(N_{m+1}) - \hat{g}_{N_{m+1}}(N_{m+1}) \geq \gamma_3\nu_1(t - l(N_{m+1}))\epsilon_\delta^{N_{m+1}}(z_m) \geq \gamma_3\nu_1\kappa, \tag{5.37}$$

otherwise we would have $N_{\min}^{m+1} > N_{\min}^m$. However each SAA objective is bounded below from Assumption A.16, and there is a finite number of possible sample sizes. Therefore, (5.37) can only be satisfied for a finite number of iterations, so we obtain a contradiction if (5.36) is satisfied. Consequently $N_k = N_{\max}$ for all k large enough. □

5.6.2 First-order optimality

Having proved that the sample size must be equal to $N_{\max}$ for k large enough, we now prove first-order convergence of the proposed algorithm by applying convergence results known for trust-region methods. For this purpose, we impose a sufficient decrease of the model at each iteration:

A.21 *For all k*

$$m_k^{N_k}(z_k) - m_k^{N_k}(z_k + s_k) \geq \kappa_3 \|\nabla_z \hat{g}_{N_k}(z_k)\| \min\left\{\frac{\|\nabla_z \hat{g}_{N_k}(z_k)\|}{\zeta_k}, \Delta_k\right\},$$

for some constant $\kappa_3 \in (0,1)$ and $\zeta_k = 1 + \max_{x\in\mathcal{B}_k} \left\|\nabla_{xx} m_k^{N_k}(z_k)\right\|$.

Assumption A.21, a classic in trust-region method analysis, is fulfilled as soon as the step ensures a model decrease at least as much as that obtained at the approximate Cauchy point (Conn, Gould and Toint [35], page 131). We then obtain our first convergence result.

Theorem 5.10 (First-order convergence)
Suppose that Assumptions A.9, A.15–A.21 hold and that

$$\exists \kappa > 0 \text{ such that } \epsilon_\delta^{N_k}(z_k) \geq \kappa,$$

for all k sufficiently large. Then, almost surely, either the algorithm converges in a finite number of iterations to a first-order critical point of $\hat{g}_{N_{\max}}$, or the number of iterations is infinite and

$$\lim_{k \to \infty} \|\nabla_z \hat{g}_{N_k}(z_k)\| = 0,$$

with $N_k = N_{\max}$ for all k sufficiently large.

Proof. From Theorem 1, we know that $N_k = N_{\max}$ for all k sufficiently large. The first-order convergence then results from the Theorem 6.4.4 in Conn, Gould and Toint [35] in the finite case, and Theorem 6.4.6 in the infinite case. □

From (5.30), we see that $\epsilon_\delta^N(z)$ is equal to 0 if and only if $\sigma(z) = 0$. This means however that each ξ_i, $i = 1, \ldots, N$, are almost surely equal, or in other terms, that almost surely, the SAA is equal to $g(z)$, independently of the number of random draws.

5.6.3 Second-order optimality

We conclude our convergence analysis by briefly indicating that, under some additional assumptions, any limit point of the sequence of iterates may be proved to be second-order critical. We first slightly strengthen the conditions governing the trust-region update, imposing that the radius actually increases at very successful iterations:

A.22 *If $\rho_k \geq \eta_2$ and $\Delta_k \leq 10^{20}$, then $\Delta_{k+1} \in [\gamma_4 \Delta_k, \gamma_5 \Delta_k]$ for some $\gamma_5 \geq \gamma_4 > 1$.*

We also require that the Hessian of the model and that of the simulated log-likelihood asymptotically coincide whenever a first-order limit point is approached.

A.23 *We assume that*

$$\lim_{k \to \infty} \|\nabla_{zz} \hat{g}_{N_k}(z)_k - \nabla_{zz} m_k^{N_k}(z_k)\| = 0 \text{ whenever } \lim_{k \to \infty} \|\nabla_z m_k^{N_k}(z_k)\| = 0.$$

Second-order convergence is then ensured if the step uses negative curvature of the model when present. This is expressed formally by the following theorem, where $\lambda_{\min}[A]$ denotes the smallest eigenvalue of the matrix A.

Theorem 5.11 (Second-order convergence) *Suppose that Assumptions A.9, A.15–A.23 hold and that*

$$\exists \kappa > 0 \text{ such that } \epsilon_\delta^{N_k}(z_k) \geq \kappa,$$

for all k sufficiently large. Let k_1 be such that $N_k = N_{\max}$ for all $k \geq k_1$. Assume furthermore that for all $k \geq k_1$, if $\tau_k = \lambda_{\min}\left[\nabla_{zz} m_k^{N_k}(z_k)\right] < 0$, then

$$m_k^{N_k}(z_k) - m_k^{N_k}(z_k + s_k) \geq \pi_2 |\tau_k| \min\{\tau_k^2, \Delta_k^2\},$$

for some constant $\pi_2 \in \left(0, \frac{1}{2}\right)$. Then, almost surely, any limit point of the sequence of iterates is second-order critical.

Proof. Directly follows from Theorem 6.6.8 of Conn, Gould and Toint [35]. □

Note also that the existence of a limit point is ensured if, as is nearly always the case, all iterates lie within a closed, bounded domain $\mathcal{C} \subseteq \mathbb{R}^m$.

Part III

Application to discrete choice theory

Chapter 6

Mixed logit models

In this chapter we adapt the consistency results developed in Sections 5.2 and 5.3 to the specific case of (possibly) constrained parameter estimation in mixed logit models. Mixed logit modelling belongs to the family of discrete choice models; this is one of the most used tools to estimate disaggregate individual preferences. Discrete choice problems have been of interest to researchers for many years in a variety of disciplines (Ben-Akiva and Lerman [12]). Examples of the many possible applications can be found in mathematical psychology (Luce [88]), in marketing (see for instance McFadden and Train [95] and Allenby and Rossi [2]), in econometric studies (e.g. McFadden [93]) and in transportation (see for instance Sheffi [129], Chapter 10) The first generation models (logit and nested logit), developed in the last 30 years contain a number of important limitations, so new models have been proposed, as the generalized extreme value and mixed logit models. Mixed logit models, that consitute the main concern of this chapter, are gaining more and more popularity within the practitioners community. Unfortunately, those specifications have no closed form expression and the solution is approximated through Monte Carlo simulations. Monte Carlo methods have been extensively used in the area of stochastic programming to treat problems incorporating uncertainty. We dedicate this chapter to a brief review of discrete models and to the application of nonconvex stochastic programming to mixed logit models. In particular, we apply our consistency results presented in Chapter 5 to mixed logit models. We also produce estimates of the simulation bias and variance.

6.1 An introduction to discrete choice models

Discrete choice analysis attempts to provide an operational description of how individuals perform a selection amongst a finite (discrete) set of alternatives. The purpose of the first three sections of this chapter is to give some principles of individual choice theories that are useful to understand the formulation and the empirical usages of discrete choice models. For a more exhaustive introduction to discrete choice theory the reader can refer to Bierlaire [19], Ortúzar and Willumsen [105], Chapter 4, and to the books by Anderson, De Palma and Thisse [4], and Train [135].

Following the framework given by Ben Akiva and Lerman [12], we view the choice as the outcome of a sequential decision making process, which includes the following steps:

1. definition of the choice problem;
2. generation of alternatives;
3. evaluation of the attributes of the alternatives;
4. choice and implementation.

We briefly describe the elements of this decisional process below.

6.1.1 Decision-maker

Choice models are referred to as disaggregated models. It means that the decision-maker is assumed to be an individual, while in aggregate modelling one observation is the average of (sometimes) hundreds of individual observations. The concept of individual may be extended, depending on the particular application; in particular the decision-maker may be a group of persons (for instance a household); the internal decisions within the group are then ignored and we consider only the decisions of the group as a whole. We will refer to decision-maker and individual interchangeably and denote by I the population size (the number of individuals).

Because of its disaggregate nature, the model has to include the characteristics, or attributes, of the individual. The analyst has to identify the attributes that are likely to explain the choice of the individual. There is no automatic process to perform this identification. The knowledge of the actual application and the data availability play an important role in this process. We will not discuss these aspects here, while this step is crucial for the practitioner. We refer to Ben-Akiva and Lerman [12] for a coverage of such questions.

6.1.2 Alternatives

The decision-maker is assumed to make a choice among a set of alternatives. The set containing these alternatives is called the choice set, and while this set can be continuous, we will consider here only discrete choice sets. A discrete choice set, that we will denote by $\mathcal{A}$, contains a finite number of alternatives that can be explicitly listed. The corresponding choice models are called discrete choice models. Two concepts of choice set are considered: the universal choice set and the reduced choice set. The universal choice set contains all potential alternatives in the context of the application. The reduced choice set is the subset of the universal choice set considered by a particular individual. Alternatives in the universal choice set that are not available to the individual under consideration are excluded. The awareness of the availability of the alternative by the decision-maker should be considered as well (see Swait [132] for more details on choice set generation). The set of alternatives available for individual i ($i = 1, \ldots, I$) will be represented by $\mathcal{A}(i) \subset \mathcal{A}$. In the following, choice set will refer to the reduced choice set, except explicitly mentioned.

6.1.3 Attributes

Each alternative in the choice set is characterized by a set of attributes that affect the choice of the individual. Some attributes may be generic to all alternatives, and some may be specific to

an alternative. An attribute is not necessarily a directly observed quantity. It can be any function of the available data. The definition of attributes as a function of available data depends on the problem. Several definitions must usually be tested to identify the most appropriate.

6.1.4 Decision rule and utilities definition

In order to model the individual choices, we have to define the rules used by the decision-maker. Different sets of assumptions can be considered, leading to different family of models, as the neoclassical economic theory, the Luce model and random utility models (see Bierlaire [19]). We will focus here on random utility models since they constitute the most common framework for generating discrete-choice models. In random utility theory each alternative has some probability to be chosen by an individual (with a null probability if it is not available to the particular individual). Such a probability is modelled as a function of the socio-economic characteristics of the individual and the relative attractiveness of the alternative. Random utility models assume that the decision-maker belongs to a given homogeneous population, acts rationally and has a perfect discrimination capability. The analyst however has incomplete information and, therefore, uncertainty must be taken into account. Manski [90] identifies four different sources of uncertainty: unobserved alternative attributes, unobserved individual attributes, measurement errors and proxy, or instrumental, variables.

For each individual i, each available alternative $A_j \in \mathcal{A}(i)$ ($j = 1, \ldots, |\mathcal{A}(i)|$) has an associated utility U_{ij}, modelled as a random variable to reflect this uncertainty. The utility is typically split into two components,

$$U_{ij} = V_{ij} + \boldsymbol{\epsilon}_{ij}. \tag{6.1}$$

In this description, $V_{ij} = V_{ij}(\beta_j, x_{ij})$ is a function of some model parameters β_j to be estimated and of x_{ij}, a vector containing all attributes, both of individual i and attributes A_j, while $\boldsymbol{\epsilon}_{ij}$ is a random term reflecting the unobserved part of the utility, reflecting the idiosyncrasies and particular tastes of each individual, together with any measurement or observational errors made by by the modeller. A popular and simple expression for V_{ij} ($j = 1, \ldots, |\mathcal{A}(i)|$) is the linear utility

$$V_{ij}(\beta_j, x_{ij}) = \beta_j^T x_{ij} = \sum_{k=1}^{K_j} \beta_j^k x_{ij}^k,$$

where K_j is the number of observed attributes for alternative A_j ($j = 1, \ldots, |\mathcal{A}(i)|$). The parameter vectors β_j ($j = 1, \ldots, |\mathcal{A}(i)|$) are assumed to be constant for all individuals but may vary across alternatives. The linear assumption simplifies the formulation and the estimation of the model, and nonlinear effects can still be captured in the attributes definitions, as a function of available data. For instance, instead of considering travel time as an attribute, the logarithm of the travel time can be considered.

A model parameter is called generic if it is involved in all alternatives, and has the same value for all of them. Otherwise it is said to be (alternative) specific. Since we can decompose a specific parameter in several parameters taking the same value for a subset of alternatives, and associated to null observations for others, we may assume, without loss of generality, that all

parameters are generic. In order to simplify the notation, we will hence omit the subscript j for parameters vectors.

6.2 Random utility models specification

The derivation of random utility models is based on a specification of the utility as defined by (6.1). The theory then assumes that individual i selects the alternative that maximizes his/her utility. In other terms, he/she chooses A_j if and only if

$$U_{ij} \geq U_{il},\ \forall A_l \in \mathcal{A}(i).$$

Thus the probability of choosing alternative A_j is given by

$$P_{ij} = P\left[\epsilon_{il} \leq \epsilon_{ij} + (V_{ij} - V_{il}), \forall A_l \in \mathcal{A}(i)\right]. \tag{6.2}$$

Different assumptions about the random terms $\boldsymbol{\epsilon}_{ij}$ and the deterministic term V_{ij} will produce specific models. Without loss of generality, it can be assumed that the residuals $\boldsymbol{\epsilon}_{ij}$ are random variables with zero mean and a certain probability distribution to be specified. The zero mean assumption is valid if the deterministic part of the utility function of each alternative includes an alternative specific constant (ASC) (Bierlaire [19]). In practice, it is impossible to estimate the value of all ASCs from observed data. From (6.2), we see that the probability of choosing some alternative is not modified is an arbitrary constant κ is added to all utilities. Only the difference between the ASCs can be identified.

It is common practice to constrain one ASC in the model to zero. While the choice of the particular alternative whose ASC is constrained is purely arbitrary from a modelling viewpoint, its influences the estimation process (see Bierlaire, Lotan and Toint [20], who also propose a different technique of ASC specification, optimal from an estimation perspective).

The scale of the utility may also be arbitrarily specified. Indeed, for any $\alpha \in \mathbb{R}$, $\alpha > 0$, we have from (6.2) that

$$P_{ij} = P[\alpha(\epsilon_{il} - \epsilon_{ij}) \leq \alpha(V_{ij} - V_{il}),\ \forall A_l \in \mathcal{A}(i)].$$

The model is said binary if only two alternatives are available. The arbitrary decision about α is then equivalent to assuming a particular variance v of the distribution of the error term. Indeed if

$$Var[\alpha(\epsilon_{il} - \epsilon_{ij})] = v^2,$$

we have also

$$\alpha = \frac{v}{\sqrt{Var[\epsilon_{il} - \epsilon_{ij}]}}.$$

This equivalence can be extended to the case where more than two alternatives are considered, if the error terms are i.i.d. across the alternatives. However the i.i.d. assumption implies that the alternatives should be, in fact, independent. Mixed-mode options, for example car-rail combinations, will usually violate this condition. We will examine this more deeply in the context of multinomial logit models in Section 6.2.2.

The actual form of the distribution of the residuals $\boldsymbol{\epsilon}_{ij}$ leads to different families of models; we will present in the next subsections the most commonly used ones. Other families of models exist (see Ben Akiva and Lerman [12], Chapter 5).

6.2.1 Probit

The multinomial probit, or normal probability unit, model is derived from the assumption that residuals $\boldsymbol{\epsilon}_i = (\boldsymbol{\epsilon}_{i1}, \ldots, \boldsymbol{\epsilon}_{iJ})^T$ is a multivariate normal distributed with a vector of means 9 and a $J \times J$ variance-covariance matrix $\Sigma_{\boldsymbol{\epsilon}}$. The probit model is motivated by the central limit theorem, assuming that the error terms are the sum of independent unobserved quantities. Unfortunately, the probability function (6.2) has no closed analytical form, which limits practical use of this model. A comprehensive development of probit models can be found in Daganzo [37].

6.2.2 Logit

If we now assume that the residuals $\boldsymbol{\epsilon}_{ij}$ are independent and identically Gumbel distributed with mean 0 and scale factor μ, the probability that the individual i chooses the alternative $A_j \in \mathcal{A}(i)$ can be expressed as

$$\frac{e^{\mu V_{ij}}}{\sum_{m=1}^{|\mathcal{A}(i)|} e^{\mu V_{im}}}. \tag{6.3}$$

This is the multinomial logit model (for derivation of (6.3), see for instance Domencich and McFadden [44], Chapter 4). It is common practice to arbitrary define $\mu = 1$.

If $\boldsymbol{\xi}_1$ and $\boldsymbol{\xi}_2$ are independent Gumbel distributed with parameters (η_1, μ) and (η_2, μ), respectively, then $\boldsymbol{\xi}_1 - \boldsymbol{\xi}_2$ is logistically distributed with mean $\eta_2 - \eta_1$ and scale parameter μ. The c.d.f. of a logistic random variable is

$$\frac{1}{1 + e^{-\mu(x-a)}},$$

where a is the mean and $\mu > 0$ is the scale parameter. The terminology logit models comes from this property. In particular, if there are only two alternatives, (6.2) becomes

$$P_{i1} = P[V_{i1} - V_{i2} \geq \epsilon_{i2} - \epsilon_{i1}].$$

The logistic distribution can be seen as an approximation to the normal distribution, as shown in Figure 6.1. In this figure, we have represented a normal of mean 0 and variance 1, a Gumbel of mean $ln(ln2)$, and scale parameter equal to $\pi/\sqrt{6}$, and a logistic of mean 0 and scale parameter or $\pi/\sqrt{3}$. Then each distribution has a median equal to 0 and a variance equal to 1.

IIA property

An important property of the multinomial logit model is the independence from irrelevant alternatives (IIA). It expresses that, if some alternatives are removed from or added to a choice set, the relative choice probabilities in the reduced choice set then remain unchanged. That is, for any choice sets $\mathcal{S} \subseteq \mathcal{T} \subseteq \mathcal{A}$, for any alternatives A_1 and A_2 in $\mathcal{S}$, we have

$$\frac{P_{\mathcal{S}}[A_1]}{P_{\mathcal{S}}[A_2]} = \frac{P_{\mathcal{T}}[A_1]}{P_{\mathcal{T}}[A_2]}.$$

A more formal description of this property and associated difficulties can be found for instance in Ben-Akiva and Lerman [12], who show, in particular, that its validity depends on the structure

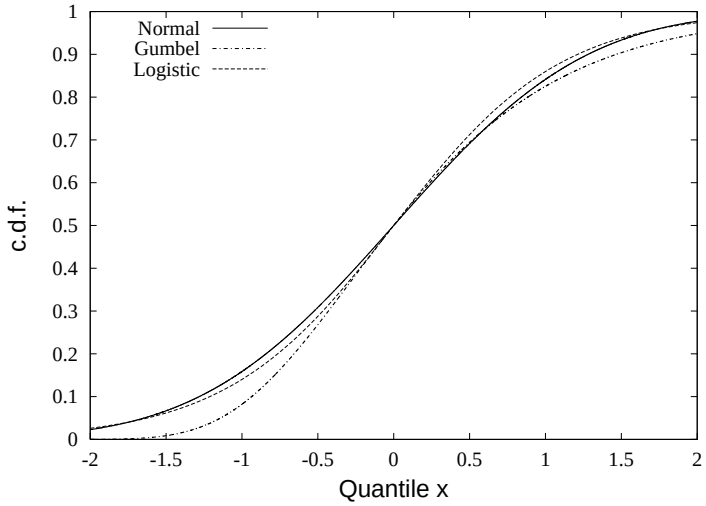


Figure 6.1: comparison between normal, Gumbel and logistic distributions

of the choice set, and also that it may be unrealistic if the alternatives are not distinct for the individual.

Several extensions of the multinomial logit model have been proposed and allow to partially avoid the IID assumption, including nested and mixed logit models (or error component structure models) (see Bhat and Koppelman [18] for a review of these developments).

6.2.3 Nested logit models

The nested logit model, first derived by Ben-Akiva [12], is an extension of the multinomial logit model designed to capture correlations among alternatives. It is based on the partitioning of the choice set $\mathcal{A}$ into n disjoint nests $\mathcal{A}_k$:

$$\mathcal{A} = \bigcup_{k=1}^{n} \mathcal{A}_k, \quad \text{and} \quad \mathcal{A}_k \cap \mathcal{A}_l, \ \forall k \neq l.$$

The utility function of each alternative is composed of a term specific to the alternative, and a term associated with the nest. If $j \in \mathcal{A}_k$, we have

$$U_j = V_j + \boldsymbol{\epsilon}_j + V_{\mathcal{A}_k} + \boldsymbol{\epsilon}_{\mathcal{V}_k},$$

where $V_{\mathcal{A}_k}$ is the component of the utility which is common to all alternatives in the nest $\mathcal{A}_k$. The error terms $\boldsymbol{\epsilon}_j$ and $\boldsymbol{\epsilon}_{\mathcal{A}_k}$ are assumed to be independent, with the error term $\boldsymbol{\epsilon}_k$ independent and identically Gumbel distributed, with scale parameter σ_k. The distribution of $\boldsymbol{\epsilon}_{\mathcal{A}_k}$ is such that the random variable $\max_{l \in \mathcal{A}_k} U_l$ is Gumbel distributed with scale parameter μ.

A pseudo-utility $V'_{\mathcal{A}_k}$ is associated to the nest $\mathcal{A}_k$. $V'_{\mathcal{A}_k}$ is called the composite utility, the expected maximum utility, the inclusive value or the accessibility in the literature. The composite

utility for nest $\mathcal{A}_k$ is defined as

$$V'_{\mathcal{A}_k} = V_{\mathcal{A}_k} + \frac{1}{\sigma_k} \ln \sum_{j \in \mathcal{A}_k} e^{\sigma_k V_j},$$

Since $A_j \in \mathcal{A}_k$, the probability that A_j is chosen is

$$P[A_j] = P[\mathcal{A}_k] P[A_j \,|\, \mathcal{A}_k],$$

where

$$P[\mathcal{A}_k] = \frac{e^{\mu V'_{\mathcal{A}_k}}}{\sum_{s=1}^{n} e^{\mu V'_{\mathcal{A}_s}}} \text{ and } P[\mathcal{A}_j \,|\, \mathcal{A}_k] = \frac{e^{\sigma_k V_j}}{\sum_{a \in \mathcal{A}_k} e^{\sigma_k V_a}},$$

The parameters μ and σ_k reflect the correlation among alternatives in the nest $\mathcal{A}_k$. Indeed, if $j, l \in \mathcal{A}_k$, we have (see Ben-Akiva and Lerman [12], page 289)

$$\frac{\mu}{\sigma_k} = \sqrt{1 - \text{corr}(U_j, U_l)}.$$

This implies that $\frac{\mu}{\sigma_k} \in [0, 1]$. If $\frac{\mu}{\sigma_k} = 1$ for all k, the alternatives are uncorrelated within the nests and the nested logit is equivalent to the multinomial logit. Note that the parameters μ and σ_k are closely related in the model, and only their ratio is meaningful. A common practice is to arbitrarily constrain one of them to a value, usually 1. A model where the scale parameter μ is constrained to 1 is said to be "normalized from the top". A model where one of the parameters σ_k is constrained to 1 is said to be "normalized from the bottom".

A direct extension of the nested logit model consists in partitionning some or all nests into sub-nets, which can, in turn, be divided into sub-nets. Because of the complexity of these models, their structure is usually represented as a tree (Daly [38]). The number of potential correlation structures can however be very large, and no technique currently exists to identify the most appropriate one from the data. Moreover, no correlation across nests can be captured by the nested logit model.

6.2.4 Generalized extreme value model

In a generalized extreme value (GEV) model, the probability of individual i choosing alternative A_j within $\mathcal{A}(i) = \{A_1, \ldots, A_n\}$ is given by

$$P_{ij} = \frac{e^{V_{ij}} \frac{\partial G}{\partial x_j}(e^{V_{i1}}, \ldots, e^{V_{iA_n}})}{\mu G(e^{V_{i1}}, \ldots, e^{V_{iA_n}})}. \tag{6.4}$$

where $G : \mathbb{R}^n_+ \to \mathbb{R}$ is a differentiable function with the following properties:

1. $G(x) \geq 0$ for all $x \in \mathbb{R}^n_+$,
2. G is homogeneous of degree $\mu > 0$, that is $G(\alpha x) = \alpha^\mu G(x)$, for all $x \in \mathbb{R}^n_+$,
3. $\lim_{x_l \to +\infty} G(x_1, \ldots, x_l, \ldots, x_n) = +\infty$ for all l such that $1 \leq l \leq n$, and

4. the k-th partial derivative with respect to k distinct x_j is nonnegative if k is odd, and nonpositive if k is even, that is $\forall j_1, \ldots, j_k$ such that $1 \leq j_l \leq n$ if $1 \leq l \leq k$ and $j_l \neq j_m$ if $1 \leq l, m \leq k$, $l \neq m$, we have

$$\forall x \in \mathbb{R}^n_+, \ \frac{\partial^k G}{\partial x_{j_1} \ldots \partial x_{j_k}}(x) \begin{cases} \geq 0 & \text{if } k \text{ is odd,} \\ \leq 0 & \text{if } k \text{ is even.} \end{cases}$$

McFadden [92] showed that the choice model defined by equation (6.4) is consistent with random utility maximization.

The multinomial logit and the nested logit model are both specific generalized extreme value models. The multinomial logit model can be derived from

$$G(x) = \sum_{j=1}^{n} x_j^{\mu},$$

and the nested logit model, from

$$G(x) = \sum_{k=1}^{n} \left(\sum_{j \in \mathcal{A}_k} e^{\sigma_k x_i} \right)^{\mu/\sigma_k}.$$

6.2.5 Mixed logit models

Decomposition (6.1) assumes that each individuals calmly weighs all the elements of interest (with no randomness) and selects the most convenient option, but only some of the above elements are observed so we need residuals to explain what otherwise would amount to non-rational behaviour. Such assumptions are clearly strong, and suggest allowing the model parameters to vary within the population. More precisely, we will assume that each parameter vector $\beta(i)$ ($i = 1, \ldots, I$) is a realization of a random vector $\boldsymbol{\beta}$. Furthermore, $\boldsymbol{\beta}$ is itself assumed to be derived from a random vector $\boldsymbol{\gamma}$ and a parameters vector θ, which we express as

$$\boldsymbol{\beta} = h(\boldsymbol{\gamma}, \theta). \tag{6.5}$$

$\boldsymbol{\gamma}$ typically specifies the random nature of the model while θ quantifies the population characteristics for the model. Moreover the use of random coefficients relaxes the IID assumption for residuals and overcomes the rigid interalternative substitution pattern of the multinomial logit models.

Usually, $\boldsymbol{\beta}$ follows itself a probability distribution, and θ specifies the parameters of this distribution. Therefore we have that $f(\boldsymbol{\beta}|\theta) = f(h(\boldsymbol{\gamma}, \theta))$, where f designs the underlying distribution function. We will nevertheless use the notation (6.5) in order to emphasize that the random part can be expressed by a non-parametric vector, as in (5.1). For example, assume that $\boldsymbol{\beta}$ is a K-dimensional vector of independent normal variables whose k-th component is $N(\mu_k, \sigma_k^2)$. We may then choose $\boldsymbol{\gamma} = (\boldsymbol{\gamma}_1, \boldsymbol{\gamma}_2, \ldots, \boldsymbol{\gamma}_K)$, with $\boldsymbol{\gamma}_k \sim N(0, 1)$ and let the vector θ specify the means and standard deviations of the $\boldsymbol{\beta}_k$, $\theta = (\mu_1, \sigma_1, \mu_2, \sigma_2, \ldots, \mu_K, \sigma_K)$. Therefore, (6.5) can be written in this case as $\boldsymbol{\beta} = (\mu_1 + \sigma_1 \boldsymbol{\gamma}_1, \mu_2 + \sigma_2 \boldsymbol{\gamma}_2, \ldots, \mu_K + \sigma_K \boldsymbol{\gamma}_K)$.

If we knew the realization $\gamma(i)$, and thus the value $\beta(i) = h(\gamma(i), \theta)$, for some individual i, the conditional probability that he/she chooses alternative j would then be given by the standard logit formula

$$L_{ij}(\gamma, \theta) = \frac{e^{V_{ij}(\beta(i), x_{ij})}}{\sum_{m=1}^{|\mathcal{A}(i)|} e^{V_{im}(\beta(i), x_{im})}}. \tag{6.6}$$

However, since $\boldsymbol{\beta}$ is random, we need to calculate the associated unconditional probability, which is obtained by integrating (6.6) over $\boldsymbol{\gamma}$:

$$P_{ij}(\theta) = E_P\left[L_{ij}(\boldsymbol{\gamma}, \theta)\right] = \int L_{ij}(\gamma, \theta) P(d\gamma) = \int L_{ij}(\gamma, \theta) f(\gamma) d\gamma, \tag{6.7}$$

where P is the probability measure associated to $\boldsymbol{\gamma}$ and $f(\cdot)$ is its distribution function.

The unknown values of θ are estimated by maximizing the log-likelihood function, i.e. by solving the program

$$\max_{\theta} LL(\theta) = \max_{\theta} \frac{1}{I} \sum_{i=1}^{I} \ln P_{ij_i}(\theta), \tag{6.8}$$

where j_i is the alternative choice made by the individual i. This involves the computation of $P_{ij_i}(\theta)$ of (6.7) for each individual i ($i = 1, \ldots, I$), which is impractical since it requires the evaluation of one multidimensional integral per individual. Therefore, we use a Monte-Carlo estimate of $P_{ij_i}(\theta)$ obtained by sampling over γ, and given by

$$SP_{ij_i}^{R}(\theta) = \frac{1}{R} \sum_{r=1}^{R} L_{ij_i}(\gamma_r, \theta), \tag{6.9}$$

where R is the number of random draws γ_r, taken from the distribution function of $\boldsymbol{\gamma}$. As a result, θ is now computed as the solution of the simulated log-likelihood problem

$$\max_{\theta} SLL^{R}(\theta) = \max_{\theta} \frac{1}{I} \sum_{i=1}^{I} \ln SP_{ij_i}^{R}(\theta). \tag{6.10}$$

However, since I can be large (typically in the thousands), the evaluation of $SLL^{R}(\theta)$ may remain very expensive, even on modern computers, as pointed out by Hensher and Greene [72].

We finally notice that the mixed logit problem (6.7)–(6.8) can be viewed as a generalization of the stochastic programming problem (5.1), that we restate for clarity:

$$\min_{z \in S} g(z) = E_{\boldsymbol{\xi}}[G(z, \boldsymbol{\xi})],$$

where $z \in \mathbb{R}^m$ is a vector of decision variables, S is a compact subset of $\mathbb{R}^m$ representing feasible solutions of the above problem, $\boldsymbol{\xi}$ is a random vector defined on the probability space $(\Xi, \mathcal{F}, P)$ and $G : \mathbb{R}^m \times \Xi \to \mathbb{R}$ is a real valued function. Indeed, we may write (6.7)–(6.8) as

$$\min_{\theta} g(\theta) = \min_{\theta} -LL(\theta) = \min_{\theta} -\frac{1}{I} \sum_{i=1}^{I} \ln E_P\left[L_{ij_i}(\gamma, \theta)\right]. \tag{6.11}$$

The associated sample average approximation problem is then written as

$$\min_{\theta} \hat{g}_R(\theta) = \min_{\theta} -SLL^R(\theta) = -\frac{1}{I} \min_{\theta} \sum_{i=1}^{I} \ln SP^R_{ij_i}(\theta). \tag{6.12}$$

We will denote by θ^* a solution of (6.11) and by θ^*_R a solution of (6.12). The generalization is minor since it only consists in optimizing a sum of logarithms of expectations, instead of a single expectation.

Note also that the multinomial logit model can then be viewed as the mean value problem if we use the error component formulation for mixed logit models, that is we express the parameters vector for each individual, $\beta(i)$ $(i = 1, \ldots, I)$, as follows:

$$\beta(i) = b + \eta(i), \tag{6.13}$$

where $b = E[\boldsymbol{\beta}]$ is the parameters mean value over the population and $\eta(i)$ is the individual deviation from the mean, representing personal preferences of each individual. We can then link (6.13) to (6.5) by defining

$$\boldsymbol{\beta} = h(\boldsymbol{\gamma}, \theta) = \theta_1 + \theta_2 \boldsymbol{\gamma},$$

where $E[\boldsymbol{\gamma}] = 0$, $K = K_1 + K_2$, $\theta_1 \in \mathbb{R}^{K_1}$, $\theta_2 \in \mathbb{R}^{K_2}$ and $\theta = (\theta_1, \theta_2)$. Replacing $\boldsymbol{\gamma}$ by its expectation and ignoring θ_2 (which can now take any value in $\mathbb{R}^{K_2}$), we obtain the multinomial logit corresponding to the original mixed logit formulation. The discussion about the value of the stochastic solution in Birge and Louveaux [21], Section 4.5, then illustrates that the multinomial logit solution can be poor compared to the mixed logit solution.

Unfortunately the mixed logit is numerically very expensive to solve, even when Monte-Carlo approximations are used; the choice of an adequate optimization procedure is therefore crucial. We will explore this aspect in Chapter 7. In the remaining of this chapter we will focus on properties of the problem (6.12). As in Chapter 5, we will study the adequation between problem (6.11) and problem (6.12) when R increases. We will also derive some useful statistical estimators that about the approximating log-likelihood. These will serve as foundations for the algorithm with variable sample size presented in Chapter 7.

6.3 Discrete choice models estimation

Having defined the form of the choice probabilities, we now face the problem of estimating the parameters vector β in the alternatives utilities. This is usually done by means of the maximum likelihood (ML) method. Assume that we have a sample of I individuals from an homogeneous population. If this population is large, we can assume that the observations in the sample are independent, so we can define the likelihood function as the product of the model probabilities that each individual chooses the option they actually selected:

$$L(\beta) = \prod_{i=1}^{I} P_{ij_i}(\beta). \tag{6.14}$$

If $\beta_* = \text{argmax} L(\beta)$, then β^* corresponds to the parameters vector which has the greatest probability of having generated the observed sample. However, in practice, I is large, so evaluating (6.14) is numerically instable since $0 \leq P_{ij_i} \leq 1$ ($i = 1, \ldots, I$), and, more importantly, the maximization of a product is often less stable than the maximization of a sum. To avoid these difficulties, it is preferable to consider the logarithm of the likelihood. We then aim at solving the program

$$\max_{\beta} LL(\beta) = \sum_{i=1}^{I} \ln P_{ij_i}(\beta). \tag{6.15}$$

From a theoretical viewpoint, any solution of (6.15) is a maximizer of (6.14) since the logarithm operator is monotically increasing. The log-likelihood function requires nevertheless that $P_{ij_i} < i$ ($i = 1, \ldots, I$) to be well-defined. This condition is fulfilled with usual random utility models. (6.15) is then a unconstrained concave maximization problem that can be easily solved with classical methods.

6.4 Application of stochastic programming of mixed logit

6.4.1 Convergence of SAA estimators

We now apply our consistency results obtained in Sections 5.2 and 5.3 of the previous chapter to the framework of mixed logit models. This is possible because we have already seen in Section 6.2.5 that the mixed logit problem is a generalization of the stochastic program (5.1).

We first specify that the i.i.d. Assumption A.9 is now to be understood as the requirement that different samples used to compute the choice probabilities are identically distributed and independent both for each individual and across them. Next note that the assumption, in the stochastic programming case, that the feasible set S is compact ensures that the solutions of problem (5.2) remain in a bounded domain of $\mathbb{R}^m$, but that our formulation of the mixed logit problem does not include any such safeguard. We therefore complete our assumptions on mixed logit models by introducing the following one.

A.24 *The solution θ_R^* of the simulated mixed logit problem* (6.12) *remains in some convex compact set S for all R sufficiently large.*

The set S can be explicitly expressed as convex constraints (bounds are typical) on the problem or be implicit for an unconstrained problem. In the latter case, A.24 indicates that the solutions are uniformly bounded for sufficiently large sample sizes. Such an assumption is reasonable to avoid pathological cases where some components of θ_R^* converge towards plus or minus infinity. As for the stochastic programming case, this assumption implies that the sequence $\{\theta_R^*\}$ has limit points, and, again as above, we identify it, without loss of generality, to one of its convergent subsequences and assume that $\theta_R^* \to \theta^*$ as $R \to \infty$.

For obtaining convergence to first-order critical points, we also need to ensure A.10 and A.11 by imposing suitable conditions on the problem's components $E_P[L_{ij_i}(\boldsymbol{\gamma}, \theta)]$.

A.10ml *The utilities $V_{ij}(\gamma,\cdot)$ $(i=1,\ldots,I,\ j=1,\ldots,J)$ are continuously differentiable for almost every γ.*

That A.10ml implies A.10 immediately results from the property of the logit formula, which ensures that

$$\frac{\partial}{\partial\theta_t}L_{ij_i}(\gamma,\theta)=L_{ij_i}(\gamma,\theta)\sum_{s\neq j_i}L_{is}(\gamma,\theta)\frac{\partial}{\partial\theta_t}\left(V_{ij_i}(\gamma,\theta,x_{ij_i})-V_{is}(\gamma,\theta,x_{is})\right). \tag{6.16}$$

A.11 is automatically satisfied since $|SP_{ij_i}(\gamma,\theta)|\leq 1$ for all θ and 1 is obviously P-integrable with unit expectation. We obtain from A.24, A.10ml and Lemma 5.1 that, for all individuals i $(i=1,\ldots,I)$,

$$SP^R_{ij_i}(\theta^*_R)\xrightarrow{a.s.}P^R_{ij_i}(\theta^*) \text{ and } SLL^R(\theta^*_R)\xrightarrow{a.s.}LL(\theta^*).$$

We now examine the derivatives of the true and SAA problems. For $t=1,\ldots,m$, $i=1,\ldots,I$, we have

$$\frac{\partial}{\partial\theta_t}LL(\theta)=\frac{1}{N}\sum_{n=1}^{N}\frac{1}{E_P\left[L_{ij_i}(\boldsymbol{\gamma},\theta)\right]}\frac{\partial}{\partial\theta_t}E_P\left[L_{ij_i}(\boldsymbol{\gamma},\theta)\right],$$

and

$$\frac{\partial}{\partial\theta_t}SLL(\theta)=\frac{1}{N}\sum_{n=1}^{N}\frac{1}{SP^R_{ij_i}(\theta)}\frac{1}{R}\sum_{r=1}^{R}\frac{\partial}{\partial\theta_t}L_{ij_i}(\gamma_r,\theta).$$

A.12 now becomes

A.12ml *For $t=1,\ldots,m$, $\frac{\partial}{\partial\theta_t}L_{ij_i}(\gamma,\theta)$ $(i=1,\ldots,I)$ is dominated by a P-integrable function.*

From (6.16), we see that it is in particular true if the following holds.

A.12ml' *For $t=1,\ldots,m$, $\frac{\partial}{\partial\theta_t}V_{ij}(\gamma,\theta,x_{ij})$ $(i=1,\ldots,I,\ j=1,\ldots,J)$ is dominated by a P-integrable function.*

If the utilities are linear in θ, as is often the case in applications, the derivatives are independent of θ. Then all we have to assume is that the expectation of the absolute partial derivatives is finite, which is usually not restrictive. If the utilities are nonlinear, we observe that (A.12ml)' is satisfied if, for $t=1,\ldots,m$, $i=1,\ldots,I$, $j=1,\ldots,J$, $E_P[K(\boldsymbol{\gamma})]$ is finite, where $K(\gamma)=\max_\theta\left|\frac{\partial}{\partial\theta_t}V_{ij}(\gamma,\theta,x_{ij})\right|$. Under A.10ml, and the assumption that $\theta\in S$, where S is compact, $K(\gamma)$ is finite for almost every γ, and its expectation is usually finite.

We may now apply Lemma 5.1 and deduce that

$$\nabla_\theta SLL^R(\theta^*_R)\xrightarrow{a.s.}\nabla_\theta LL(\theta^*),$$

as $R \to \infty$. Using the same arguments as in Theorem 5.1, we then obtain the following result.

Theorem 6.1 (First-order convergence for mixed logit) *Assume that A.9, A.24, A.10ml and A.12ml hold. Then θ^* is a first-order critical point of problem* (6.11) *almost surely.*

We have therefore proved convergence of the simulated estimators to the true maximum likelihood estimators almost surely, allowing the inclusion of convex constraints on θ. Classical results (see Chapter 10 of Train [135]) shows convergence in distribution and in probability asymptotically when the population size increases. The asymptotic behaviour is briefly discussed in section 6.4.3.

The extension of Theorem 5.5 establishing second-order convergence to the mixed logit problem is immediate, as well as Theorem 5.4, as soon as the corresponding assumptions are fulfilled.

6.4.2 Estimation of the simulation's variance and bias

We now further investigate the question of estimating the error made by using the SAA problem (6.12) instead of the true problem (6.11) as a function of the sampling size R. Due to the stochastic nature of the approximation, the size of the error can only been assessed by providing a (hopefully high) probability that it is within some confidence interval asymptotically centred at zero and of radius ϵ. In practice, we first fix some probability level $\alpha > 0$ and determine the value of ϵ such that, for given θ,

$$P\left[|LL(\theta) - SLL^R(\theta)| \leq \epsilon\right] \geq \alpha.$$

Developing this expression we have that $\left|LL(\theta) - SLL^R(\theta)\right|$ is smaller than ϵ if and only if

$$\left|\frac{1}{I}\sum_{i=1}^{I} \ln P_{ij_i}(\theta) - \frac{1}{I}\sum_{i=1}^{I} \ln SP^R_{ij_i}(\theta)\right| \leq \epsilon.$$

Consider now individual i. We are interested in the asymptotic behaviour of

$$\ln P_{ij_i}(\theta) - \ln SP^R_{ij_i}(\theta)$$

for a given θ (such as the solution of the SAA problem). Since the logarithm is continuously differentiable on $\mathbb{R}^+_0$ and since $E_P\left[L_{ij_i}\left(\boldsymbol{\gamma}, \theta\right)^2\right]$ is finite, we can use the delta method (see for instance Borovkov [22], page 44, for the one-dimensional case or Rubinstein and Shapiro [120] section 6.3, for the multi-dimensional case) to conclude that

$$\sqrt{R}\left(\ln P_{ij_i}(\theta) - \ln SP^R_{ij_i}(\theta)\right) \Rightarrow \frac{d}{dP_{ij_i}} \ln P_{ij_i}(\theta) N(0, \sigma^2_{ij_i}(\theta)),$$

where $\sigma^2_{ij_i}(\theta)$ is the variance of $P_{ij_i}(\theta, \gamma)$. In other terms, we have that

$$\ln P_{ij_i}(\theta) - \ln SP^R_{ij_i}(\theta) \Rightarrow \frac{1}{P_{ij_i}(\theta)\sqrt{R}} N\left(0, \sigma^2_{ij_i}(\theta)\right).$$

As samples are independent between individuals, so are the normal distributions in this last limit, and we thus have that

$$LL(\theta) - SLL^R(\theta) \Rightarrow \frac{1}{I} N\left(0, \sum_{i=1}^{I} \frac{\sigma_{ij_i}^2(\theta)}{R\left(P_{ij_i}\right)^2}\right) \tag{6.17}$$

The associated asymptotic value of the confidence interval radius ϵ is then given by

$$\epsilon_\delta^R(\theta) = \alpha_\delta \frac{1}{I} \sqrt{\sum_{i=1}^{I} \frac{\sigma_{ij_i}^2(\theta)}{R\left(P_{ij_i}(\theta)\right)^2}}. \tag{6.18}$$

Recall that α_δ is the quantile of a $N(0,1)$ associated to some level of signification δ. In practice we evaluate this accuracy $\epsilon_\delta^R(\theta)$ by taking the SAA estimators $\sigma_{ij_i}^R(\theta)$ and $P_{ij_i}^R(\theta)$.

Equation (6.18) gives us important information on the quality of the approximation. The accuracy can be improved if we take a bigger sampling size R, but, as in other basic Monte-Carlo methods, the convergence is only in $O\left(\sqrt{R}\right)$ (Fishman [52], page 8). However the population size also has an influence on the quality of the approximation. First of all, we note that

$$0 \leq \epsilon_\delta^R(\theta) \leq \alpha_\delta \frac{1}{I} \sum_{i=1}^{I} \sqrt{\frac{\sigma_{ij_i}^2(\theta)}{R\left(P_{ij_i}(\theta)\right)^2}}.$$

If the total population is assumed to be infinite, then we may consider a population of size I as a independent and identically distributed sample within it. We then obtain from the strong law of large numbers that, almost surely,

$$0 \leq \epsilon_\delta^R(\theta) \leq \frac{\alpha_\delta}{\sqrt{R}} E_I\left[\frac{\sigma_{ij_i}(\theta)}{P_{ij_i}(\theta)}\right].$$

In other terms, for a fixed sampling size R and a fixed θ, if the expectation of individual errors is finite, $\epsilon_\delta^R(\delta)$ converges to some real value which is less than this expectation. Assume indeed that there exists some κ such that for all θ in S, and for each individual i ($i = 1, \ldots, I$),

$$\frac{\sigma_{ij_i}(\theta)}{P_{ij_i}(\theta)} \leq \kappa.$$

Then, from (6.18),

$$\epsilon_\delta^R(\theta) \leq \alpha_\delta \frac{\kappa}{\sqrt{IR}}.$$

This suggests that the error decreases as the population size increases. However, we must remember that $E\left[SLL^R(\theta)\right] \neq LL(\theta)$, because of the logarithmic operator, and our confidence interval is thus centred at zero only asymptotically. However, since (6.17) implies that $LL(\theta) - SLL^R(\theta) \xrightarrow{p} 0$, when R tends to ∞ for a fixed population size I, we deduce that the estimator is consistent. To estimate the bias for a given finite R, we first compute the Taylor development of $\ln SP_{ij_i}^R$ around the true value P_{ij_i}, for some individual i:

$$\ln SP_{ij_i}^R(\theta) = \ln P_{ij_i}(\theta) + \frac{1}{P_{ij_i}(\theta)} h_{ij_i} - \frac{1}{2\left(P_{ij_i}(\theta)\right)^2} h_{ij_i}^2 + O\left(h_{ij_i}^3\right),$$

where $h_{ij_i} = SP^R_{ij_i}(\theta) - P_{ij_i}(\theta)$. Therefore, since $E\left[h_{ij_i}\right] = 0$,

$$E\left[\ln SP^R_{ij_i}(\theta)\right] - \ln P_{ij_i}(\theta) = -\frac{1}{2\left(P_{ij_i}(\theta)\right)^2} E\left[h^2_{ij_i}\right] + E\left[O\left(h^3_{ij_i}\right)\right].$$

Because A.9, we obtain then that

$$E\left[h^2_{ij_i}\right] = \frac{1}{R}\sigma^2_{ij_i}(\theta).$$

Averaging now over the individuals, and neglecting the terms of order three and above, we obtain that the simulation bias B can be approximated by

$$B^R(\theta) := E[SLL^R(\theta)] - LL(\theta) = -\frac{1}{2IR}\sum_{i=1}^{I}\frac{\sigma^2_{ij_i}(\theta)}{\left(P_{ij_i}(\theta)\right)^2} \leq 0, \tag{6.19}$$

which can be easily computed from the estimated error as

$$B^R(\theta) = -\frac{I}{2\alpha^2_\delta}\left(\epsilon^R_\delta(\theta)\right)^2. \tag{6.20}$$

Thus, (6.19) implies that, up to second order,

$$\max_\theta E[SLL^R(\theta)] \leq \max_\theta LL(\theta).$$

It is interesting to note from (6.18) that the confidence interval radius $\epsilon^R_\delta(\theta)$ is small whenever the standard deviations are themselves small compared to the probability choices. Moreover, (6.20) shows that the simulation bias decreases faster than the error. This suggests that the number of random draws is heavily related to the nature of the model: as expected, more variation of model parameters between the individuals imposes larger samples. The choice of a uniformly satisfying sample size across different models thus appears doubtful. This observation seems to support, for the case of the objective function value, the practical conclusions of Section 4.3 of Hensher and Greene [72].

Recall that the bias results from the use of the logarithm operator, so a formulation that transforms the likelihood function (6.14) to a form that is more manageable for the optimization, while avoiding the introduction of a bias, could be desirable. Nevertheless, in addition to the production of a more stable mathematical program, the ability of the logarithm operator to transform a product into a sum is particularly interesting for the simulation error estimation, which is obtained in our case as a confidence interval radius. Such an approach requires the knowledge of the (asymptotic) probability distribution of the simulated objective. Summing normal distributions, as in our estimation of the simulated log-likelihood accuracy, results in another normal distribution, but other transformations can lead to distributions that are difficult to identify. This is in particular the case for the original product formulation (6.14). The SAA problem would then be a product of simulated, normally distributed (with different means and variances), probabilities, but the resulting distribution does not correspond to a simple probability distribution. Techniques exist to reduce the bias in a function, but this often leads to increases of bias

in the derivatives (see for instance Cox and Hinkley [36]), so a point that satisfies optimality conditions of such an approximated problem is not necessarily better than a solution of the SAA problem (6.15). Careful investigation, both theoretically and numerically, would therefore be useful for a better understanding of the bias influence on the solution and the bias reduction techniques that could be used.

Finally, note also that if we make the additional assumption that the SAA problems are solved globally instead of locally, we obtain that

$$\max_{\theta \in S} E[SLL^R(\theta)] \leq E\left[\max_{\theta \in S} SLL^R(\theta)\right].$$

Therefore the maximization procedure itself can produce another bias opposed to the bias of simulation. As a consequence, the solutions of successive SAA problems do not necessarily increase monotonically when R grows, which makes bias tests based on this increase questionable.

These estimates provides information on the quality of the successive average approximation which can be used to improve efficiency of numerical estimation procedures, as done in the software AMLET (Bastin, Cirillo, and Toint [8, 9]), described in the next chapter.

6.4.3 Asymptotic behaviour for increasing population sizes

We finally devote a last paragraph to the extension of the results obtained by Hajivassiliou and McFadden [70] and published in Train [135]. In particular we study the consistency and efficiency of the SAA problem when the population size becomes infinite. Our results apply to the constrained case and the convergence results hold almost everywhere, instead of in distribution.

From the strong law of large numbers,

$$B^R(\theta) \xrightarrow{a.s.} -\frac{1}{2R} E_I\left[\frac{\sigma^2_{ij_i}(\theta)}{(P_{ij_i})^2}\right].$$

Therefore the problem is consistent if and only if R tends to infinity when N tends to infinity, as reported by Hajivassiliou and McFadden [70] and Train [135], page 288. Taking the Taylor expansion around the true parameters, that solve $ELL(\theta) := E_I\left[\ln P_{ij_i}(\theta)\right]$, the expectation of the logarithm of the probability choice for all individuals i, these authors conclude that

- if R is fixed, the SAA problem is inconsistent;
- if R rises slower than $\sqrt{N}$, the SAA problem is consistent but not asymptotically normal;
- if R rises faster than $\sqrt{N}$, the SAA problem is consistent, asymptotically normal and efficient, equivalent to the true problem.

Note that these results are obtained using convergence in distribution of the solutions of the SAA problems. We provide, in the next theorem, results of the same type. They are now expressed almost surely, at the expense of not being directly computable.

Proposition 6.1 *Assume that a ULLN holds for the approximation $LL(\theta)$ of $ELL(\theta)$ and another ULLN holds for the approximation $SLL^R(\theta)$ of $LL(\theta)$. Suppose furthermore that*

$SLL^R(\cdot,\gamma)$ is continuous on S for almost every γ, that $LL(\theta)$ is continuous on S for almost every i, and that $ELL(\theta)$ is continuous on S. Then

$$\sup_{\theta\in S}\left|SLL^R(\theta)-ELL(\theta)\right| \xrightarrow{a.s.} 0$$

as I tends to infinity and R tends to infinity sufficiently fast compared to I.

Proof. Let $\delta>0$ be a small constant. From the ULLN assumption for $LL(\theta)$, we have that, for I sufficiently large,

$$\sup_{\theta}\left|E_I\left[\ln P_{ij_i}(\theta)\right]-\frac{1}{I}\sum_{i=1}^{I}\ln P_{ij_i}(\theta)\right|<\frac{\delta}{2} \quad \text{a.s.}$$

For such an I, we have, from the ULLN assumption for $SLL^R(\theta)$, that for R sufficiently high,

$$\sup_{\theta}\left|\frac{1}{I}\sum_{i=1}^{I}\ln P_{ij_i}(\theta)-\sum_{i=1}^{I}\ln\frac{1}{R}\sum_{r=1}^{R}SP^R_{ij_i}(\theta)\right|<\frac{\delta}{2} \quad \text{a.s.}$$

Combining these two inequalities with the triangular inequality

$$\sup_{\theta}|SLL^R(\theta)-ELL(\theta)|\le\sup_{\theta}|SLL^R(\theta)-LL(\theta)|+\sup_{\theta}|LL(\theta)-ELL(\theta)|,$$

we obtain that

$$\exists I_\delta \text{ s.t. } \forall I\ge I_\delta\ \exists R_I \text{ s.t. } \forall R\ge R_I, \sup_{\theta}|SLL^R(\theta)-ELL(\theta)|<\delta \quad \text{a.s.}$$

Now define some sequence $\{\delta_n\}_{n=1}^{\infty}$ converging to zero, and let $\{I_{\delta_n}\}$ be the corresponding population sizes as given by this last bound. If the population size I grows faster than I_{δ_n} and R faster than R_I, we see that

$$\sup_{\theta}\left|SLL^R(\theta)-LL(\theta)\right|\to 0, \quad \text{a.s.}, \tag{6.21}$$

which implies the desired result in this case. If, on the other hand, I grows slower than I_{δ_n}, we identify an increasing subsequence of population sizes $\{I_n\}\subseteq\{I\}$ that grows faster than I_{δ_n}. For population sizes I' between I_n and I_{n+1}, (6.21) holds if we require $R_{I'}$ to be equal or larger than $R_{I_{n+1}}$. As a consequence, we obtain that (6.21) holds irrespective of the speed of growth of $\{I\}$ provided R grows sufficiently fast. □

Let $\{\theta^*_{I,R}\}$ be the sequence of SAA solutions for I tending to infinity, and R tending to infinity sufficiently fast compared to I. Let θ^* be a limit point of this sequence and assume (without loss of generality) that $\{\theta^*_{I,R}\}$ converges to θ^*. Then, under the assumptions of the previous proposition, we obtain from Lemma 5.1 that

$$SLL^R\left(\theta^*_{I,R}\right)\xrightarrow{a.s.}ELL(\theta^*).$$

We may finally re-apply our convergence analysis to this framework, and obtain, under assumptions similar to those used above (we now need domination by functions that are $(I\times P)$-integrable), that, almost surely, θ^* is a first (second)-order critical point if the $\theta^*_{I,R}$ are first (second)-order critical points.

6.4.4 Applications of mixed logit models

We conclude this chapter by mentioning some of the numerous applications of mixed logit models. To our knowledge, the mixed logit approach appeared in 1977, in a analysis of the demand for different types of automobiles in the United States, made by the Electric Power Research Institute [47] (EPRI), where the parameters to estimate were independent and lognormally distributed. However mixed logit models have became popular only since the last ten years among researchers and practitioners in economics and transportation (see, for instance, Bhat and Castelar [17], Brownstone, Bunch and Train [23], Cirillo and Axhausen [26], Hensher and Greene [72], Hensher and Sullivan [73], Hess and Polak [74]), sociology (Montmarquette, Cannings, and Mahseredjian [100]),.... The developments in science computing during the last decade made them indeed numerically manageable even with personal computers, while in the algorithmic achievements made the estimation process more efficient. We will in particular study the application of the BTRDA Algorithm 5.1 to mixed logit estimation in the next chapter.

Chapter 7

AMLET

In the previous chapter we have established that the sample average approximation of the maximum likelihood program occurring in mixed logit models estimation is a manageable way to approximate the maximum likelihood estimators. But, even in this form, evaluation costs can be prohibitive due the required sample sizes, as mentioned for instance by Hensher and Greene [72]. The current research approach has thus shifted, in order to reduce computational time and simulation error, to quasi-Monte-Carlo approaches instead of pure Monte-Carlo methods. However these authors underline the need for speed in practice, in order to be able to explore alternative model specifications. As a consequence, current research has turned to the cheaper quasi-Monte Carlo approaches, based on low discrepancy sequences, which has been shown to produce more accurate integration approximations when the number of draws is fixed, for instance in the study of physics problem (Morokoff and Caflish [102]). Bhat [15] and Train [134] for instance advocate using Halton sequences (Halton [71]) for mixed logit models and find they perform much better than pure random draws in simulation estimation. Garrido [57] explores the use of Sobol sequences, while Sándor and Train [123] compare randomized Halton draws and (t, m, s)-nets.

This trend is not without drawbacks. For instance, Bhat [15] pointed out that the coverage of the integration domain by Halton sequences rapidly deteriorates for high integration dimensions and consequently proposed a heuristic based on the use of scrambled Halton sequences. He also randomized these sequences in order to allow the computation of the simulation variance of the model parameters. However Hess, Polak and Daly [75] have shown that scrambled Halton methods are very sensitive to the number of draws, and can behave poorly when this number increases. Recently Hess, Train and Polak [76] have proposed the use of randomly shifted uniform vectors and have reported better performance than with any of the Halton based approached. By contrast, the dimensionality problem is irrelevant in pure Monte Carlo methods, and while computational experiments show that for low dimensional integration quasi-Monte Carlo techniques outperform Monte Carlo integration, the advantage is less clear in high-dimension (Deák [40], Morokoff and Caflish [102]). The same is reported for estimation of mixed logit models, where Monte Carlo methods are again competitive when high-dimensional problems are considered (Hess, Train and Polak [76]). Moreover for quasi-Monte Carlo procedures, the quality of the results can only be estimated in practice, by repeating the calibration process on randomized samples and by varying the number of random draws, while Monte-Carlo techniques can benefit

from classical statistical inference. This suggests adapting the BTRDA Algorithm 5.1, presented in Chapter 5 to the mixed logit estimation problem and we show that this technique results in an algorithm that is numerically competitive with existing tools for mixed logit models, while giving more information to the practitioner. This underlines the importance of the choice of optimization algorithm when looking for numerical performances and shows that exploitation of statistical inference is valuable.

7.1 The BTRDA algorithm for mixed logit models

Algorithm 5.1 can be directly applied when solving problem (6.12),

$$\min_{\theta} -SLL^R(\theta) = \min_{\theta} -\frac{1}{I}\sum_{i=1}^{I} \ln SP^R_{ij_i}(\theta).$$

However, this problem differs from the stochastic problem (5.2)

$$\min_{z\in S} \hat{g}_N(z) = \frac{1}{N}\sum_{i=1}^{N} G(z,\xi_i),$$

by the presence of a bias, that we should take into account when varying the sample sizes in Step 5, in order to make the method efficient. Moreover, we prefer to consider here the maximization formulation (6.10),

$$\max_{\theta} SLL^R(\theta) = \max_{\theta} \frac{1}{I}\sum_{i=1}^{I} \ln SP^R_{ij_i}(\theta), \tag{7.1}$$

which is more popular in the discrete choice literature. For clarity, we therefore rewrite the BTRDA Algorithm 5.1, where we use the notations relative to the mixed logit problem (7.1), and adapt the terminology to reflect that we face a maximization, not minimization, problem. The sample size update in Step 5 also considers the simulation bias, and refers to Algorithm 7.2 instead of Algorithm 5.3. We finally explicit in Algorithm 7.3 the practical procedure used when updating the minimum sample size, on the basis of Algorithm 5.4.

Algorithm 7.1: Trust-region maximization algorithm for mixed logit estimation

Step 0. Initialization. An initial point θ_0 and in initial trust-region radius Δ_0 are given. The constants η_1, η_2, γ_1, and γ_2 are also given as in Algorithm 2.2.

Set a minimum number of draws $R_{\min} = R^0_{\min}$ and a sample size R_0 satisfying $\|\nabla_\theta SLL^{R_0}(\theta_0)\| \neq 0$ if $\epsilon_\delta^{R_0}(\theta_{k+1}) \neq 0$, except if $R_0 = R_{\max}$. Compute $SLL^{R_0}(\theta_0)$ and set $k = 0$, $t = 0$.

Step 1. Stopping test. Stop if $\|\nabla_\theta SLL^{R_k}(\theta_k)\| = 0$ and either $R_k = R_{\max}$, or $\epsilon_\delta^{R_k}(\theta_k) = 0$. Otherwise go to Step 2.

Step 2. Model definition. Define a model $m_k^{R_k}$ of $SLL^{R_k}(\theta)$ in $\mathcal{B}_k$. Compute a new adequate sample size R^+ (see Algorithm 5.2). Set $R^- = R_k$.

Step 3. Step calculation. Compute a step s_k that sufficiently increases the model $m_k^{R_k}$ and such that $\theta_k + s_k \in \mathcal{B}_k$. Set

$$\Delta m_k^{R_k} = m_k^{R_k}(\theta_k + s_k) - m_k^{R_k}(\theta_k).$$

Step 4. Comparison of increases. Compute $SLL^{R^+}(\theta_k + s_k)$ and define

$$\rho_k = \frac{SLL^{R^+}(\theta_k + s_k) - SLL^{R_k}(\theta_k)}{\Delta m_k^{R_k}}. \tag{7.2}$$

Step 5. Sample size update. If $\rho_k < \eta_1$ and $R_k \neq R^+$, modify R^- or the candidate sample size R^+ to take bias and variance differences into account (see Algorithm 7.2). Recompute ρ_k.

Step 6. Acceptance of the trial point. If $\rho_k < \eta_1$, define $\theta_{k+1} = \theta_k$, $R_{k+1} = R^-$. Otherwise define $\theta_{k+1} = \theta_k + s_k$ and set $R_{k+1} = R^+$; increment t by one.

If $R_{k+1} \neq R^{\max}$, $\|\nabla_\theta SLL^{R_{k+1}}(\theta_{k+1})\| = 0$, and $\epsilon_\delta^{R_{k+1}}(\theta_{k+1}) \neq 0$, increase R_{k+1} to some size less or equal to $R_{\max}$ such that $\|\nabla_\theta SLL^{R_{k+1}}(\theta_{k+1})\| \neq 0$ if $R_{k+1} \neq R_{\max}$, and compute $SLL^{R_{k+1}}(\theta_{k+1})$.

If $R_k = R_{k+1}$ or if sufficient decrease has been observed since the last evaluation of $SLL^{R_{k+1}}$, set $R_{\min}^{k+1} = R_{\min}^k$. Otherwise define $R_{\min}^{k+1} > R_{\min}^k$ (see Algorithm 7.3).

Step 7. Trust-region radius update. Identical to Step 4 of Algorithm 2.2.

In our implementation, we have set $\eta_1 = 0.01$, $\eta_2 = 0.75$, $\gamma_1 = 0.5$ and $\gamma_2 = 0.5$. Note that t is again used as a successful iterations counter. We develop our other practical choices in the remaining of this section.

7.1.1 Model choice and trial step computation

We use a quadratic model, defined as

$$m_k^R(\theta_k + s) = m_k^R(\theta_k) + \langle g_k^R, s\rangle + \frac{1}{2}\langle s, H_k s\rangle,$$

where

$$m_k^R(\theta_k) = SLL^R(\theta_k) \text{ and } g_k^R = \nabla_\theta SLL^R(\theta_k) \tag{7.3}$$

and where H_k is a symmetric approximation to $\nabla^2_{\theta\theta} SLL^R(\theta_k)$. In our implementation we use the symmetric rank-one (SR1) quasi-Newton update to obtain such an approximation, as described in Nocedal and Wright [104], page 204. Despite being numerically cheap, SR1 is very efficient in the context of trust-region methods (see Conn, Gould and Toint [31, 33] or Byrd, Fayez Khalfan

and Schnabel [24]). In particular, it significantly outperformed the BFGS update, another well-known method, during our preliminary tests (see Section 7.3.2 below).

Another benefit of the SR1 update is that it generates good Hessian approximations for general nonlinear functions, under some reasonable conditions. Using the final SR1 Hessian approximation in the computation of t-statistics, we observed that these statistics are usually similar to those obtained with the true Hessian, but not always. The evaluation of the true Hessian is a very expensive task, especially when the number of parameters is high, while the approximate Hessian is directly available. The potential time saving when using the SR1 approximation could therefore be important: we believe that investigating more precisely when the Hessian approximation is sufficient for computing the t-statistics is a valuable direction for further research.

The computation of the step s_k is performed using the Steihaug-Toint method (see for instance Conn, Gould and Toint [35], Section 7.5.1, or Nocedal and Wright [104], page 75).

7.1.2 The variable sample size strategy

Algorithm 5.2 is based on the possibility to estimate the error made by using the sample average approximation, using statistical inference on the approximating objective. Recall that, from the delta method, we have

$$LL(\theta) - SLL^R(\theta) \Rightarrow \frac{1}{I} N\left(0, \sum_{i=1}^{I} \frac{\sigma_{ij_i}^2(\theta)}{R\left(P_{ij_i}\right)^2}\right) \tag{7.4}$$

so the approximation error can be estimated by

$$\epsilon_\delta^R(\theta) = \alpha_\delta \frac{1}{I} \sqrt{\sum_{i=1}^{I} \frac{\sigma_{ij_i}^2(\theta)}{R\left(P_{ij_i}(\theta)\right)^2}}.$$

Therefore, in Algorithm 5.2 on page 110, the suggested sample size N^s, based on the approximation error and used, is now given by

$$R^s = \max\left\{ R_{\min}^k, \left\lceil \frac{\alpha_\delta^2}{(I\Delta m_k^{R_k})^2} \sum_{i=1}^{I} \frac{\left(\sigma_{ij_i}^{R_k}(\theta)\right)^2}{\left(P_{ij_i}^{R_k}(\theta)\right)^2} \right\rceil \right\}.$$

In our tests, we set $\chi_1 = 0.5$.

The sample size revision of Algorithm 5.3 has to be slightly modified in order to take account of the bias of simulation along with the simulation variance. The sample size revision is now performed with the algorithm below.

Algorithm 7.2: Sample size revision when $\rho_k < \eta_1$ and $R_k \neq R^+$.

If $R^+ < R_k$ set

$$R^b = \left\lceil \frac{1}{2\Delta m_k^{R_k} I} \sum_{i=1}^{I} \frac{\left(\sigma_{ij_i}^{R_k}(\theta)\right)^2}{\left(P_{ij_i}^{R_k}(\theta)\right)^2} \right\rceil.$$

If $R^+ < R^b < R_k$, set $R^+ = R^b$ and recompute ρ_k from (7.2).

If the (possibly recomputed) $\rho_k < \eta_1$, compare R^+ and R_k. If $R^+ > R_k$, compute $SLL^{R^+}(\theta_k)$, $\Delta m_k^{R^+}$ and $\epsilon_\delta^{R^+}(\theta_k)$, else if $R^+ < R_k$ compute $SLL^{R_k}(\theta_k + s_k)$. Set R^- to $\max\{R_k, R^+\}$, and redefine

$$\rho_k = \frac{SLL^{R^-}(\theta_k + s_k) - SLL^{R^-}(\theta_k)}{\Delta m_k^{R^-}}.$$

When the number of draws increases ($R^+ > R_k$), the bias decreases in absolute value, but the objective function can still increase due to the refinement of the sample average approximations. Therefore, we force the algorithm to evaluate $SLL^{R^+}(\theta_k)$ in order to avoid the accuracy difference effect. The case $R^+ < R_k$ is more subtle since the absolute value of the bias then increases, so the objective function is usually lower for a fixed θ. If ρ_k is low, we try to circumvent the bias effect by testing another sample size R^b, that corresponds to the sample size giving a bias equal to the predicted increase, using the estimation (6.20).

We also slightly modify Algorithm 5.4 that updates the minimum sample size, in order to better take the bias of simulation into account. As before we define two $R_{\max}$-dimensional vectors v and l, and, at iteration $k = 0$, we set $v(N_0) = \hat{g}_{N_0}(z_0)$, $l(N_0) = 0$, and for $i = 1, \ldots, N_{\max}$, $i \neq N_0$, $v(i) = -\infty$ (instead of $+\infty$ since we face now to an optimization process), $l(i) = -1$. At the beginning of iteration k, $v(i) = SLL^i\left(\theta_{h(i)}\right)$ where $h(i)$ corresponds to the index of the last iteration for which $R_{h(i)} = i$, and $R_{h(i)-1} \neq R_{h(i)}$ if $h(i) > 0$, or $+\infty$ if the size i has not been used yet. $l(i)$ contains the number of successful iterations until iteration $h(i)$ (included), or -1 if the size i has not been used. Algorithm 5.4 is now reformulated as below.

Algorithm 7.3: Minimum sample size update when $R_k \neq R_{k+1}$.
Let $\gamma_3 \in (0, 1]$ be a constant. If

$$SLL^{R_{k+1}}(\theta_{k+1}) - v(R_{k+1}) \geq \gamma_3 \nu_1 (t - l(R_{k+1})) \epsilon_\delta^{R_{k+1}}(\theta_{k+1}), \tag{7.5}$$

set $R_{\min}^{k+1} = R_{\min}^k$. Otherwise increase the minimum sample size: if $R_k < R_{k+1}$, set

$$R_{\min}^{k+1} = \min\left\{\left\lceil \frac{R_k + R_{k+1}}{2} \right\rceil, R_{\max}\right\},$$

else

$$R_{\min}^{k+1} \in \{R_{k+1} + 1, \ldots, R_{\max}\}.$$

Set $l(R_{k+1}) = t$ and $v(R_{k+1}) = SLL^{R_{k+1}}(\theta_{k+1})$.

Note that we apply a different strategy if the sample size decreases or increases. In the first case, bias difference and loss of precision can explain a decrease or a small increase of the SAA objective, but it is numerically cheaper to continue to use sample sizes as small as possible; in our implementation we then set $R_{\min}^{k+1} = R_{k+1} + 1$. In the second case, we try to avoid poor increases of the SAA objective for large sample sizes since the associated numerical cost is then

important, and we then use a more conservative approach.

The constant γ_3 is set to 0.5 in our tests. As before, we take safeguards to avoid the pathological case in which θ_k is a first-order critical point for SLL^{R_k}. In practice, we have chosen to set $R_{k+1} = R_{\max}$ if $\|\nabla_\theta SLL^{R_k}(\theta_k)\| \leq tol$ and $\epsilon_\delta^{R_k} \geq tol$, where tol is a predefined tolerance (we used 10^{-6}), but this feature was never triggered in our experiments. Indeed, the gradient norm usually changes slowly in the vicinity of such a critical point, and a small gradient typically leads to a small model increase, which itself then causes the sample size to increase and $R_{\max}$ is always reached before our safeguard is activated.

7.1.3 Stopping tests

The presence of statistical error requires that the classical stopping tests for unconstrained optimization, which involve the gradient norm and sometimes the difference between successive iterates or function values, must be considered with caution. In particular they usually lead to final iterations that produce insignificant objective decreases compared to the approximation's accuracy. Numerical simulations revealed however that the algorithm can reach an adequate accuracy for a subset of the parameters but then produce small improvements at the maximum sample size during a few iterations, after what good improvements are again obtained, and the desired accuracy achieved on the remaining parameters. This is in particular true for parameters that are hard to estimate, such as small standard deviations since they produce small variations of the simulated likelihood function. It is therefore important not to stop the algorithm prematurely.

In our implementation, we choose to stop the iterative process if

$$\|\nabla_z SLL^R(z)\| \leq \max(\mu_1 \epsilon_\delta^R(z), tol),$$

where $0 < \mu_1 < 1$ and $\epsilon_\delta^R(z)$ is the estimated log-likelihood accuracy, and either the maximum sample size $N_{\max}$ is used or, in order to consider the multinomial logit case, the estimated log-likelihood accuracy is sufficiently small. The value $\mu_1 = 0.2$ has revealed to be a good compromise, for a signification level δ set to 0.9 in the accuracy estimator. We also stop the algorithm if a (user preset) maximum number of iterations has been reached without convergence, or if the norm of computed step falls under a user-defined significativity threshold (we used 10^{-6}).

7.1.4 Convergence

The convergence properties developed in Section 5.6 for the BTRDA Algorithm 5.1 are still valid. We nevertheless discuss briefly some of the assumptions and show that most of them are implied by our implementation choices. Recall also that the problem is now a maximization one, while in Chapter 5, we consider minimization problems. Therefore, we have to consider the satisfaction of assumptions by the opposite of the simulated log-likelihood objectives and the opposite of the models $m_k^{R_k}$, $k \geq 0$.

First note that Assumption A.15 is fulfilled with the following one:

A.13ml *The utilities $V_{ij}(\gamma, \cdot)$ $(i = 1, \ldots, I, j = 1, \ldots, J)$ are twice continuously differentiable for almost every γ.*

Assumption A.16 is immediately satisfied since the approximating objective is bounded above by zero. We obtain indeed from (6.9)

$$SP_{ij_i}^R(\theta) = \frac{1}{R}\sum_{r=1}^{R} L_{ij_i}(\gamma_r, \theta),$$

and the logit formula (6.6) that:

$$SLL^R(z) = \frac{1}{I}\sum_{i=1}^{I} \ln SP_{ij_i}^R \leq \frac{1}{I}\sum_{i=1}^{I} \ln 1 = 0, \tag{7.6}$$

for $R = R_{\min}^0, \ldots, R_{\max}$.

Our choice to use a quadratic model, as described in Section 7.1.1, guarantees satisfaction of Assumptions A.18 and A.19. Moreover, from Theorem 5.9 on page 114, R_k is constant for k large enough. The SR1 approximation then satisfies A.23 under reasonable conditions (see for instance Nocedal and Wright [104], page 207).

Assumption A.21, a classic in trust-region method analysis, is fulfilled by our choice of the Steihaug-Toint step since it ensures a model decrease at least as much as at that obtained at the approximate Cauchy point (Conn, Gould and Toint [35], page 131).

The condition (5.33) can now be expressed as

$$\exists \kappa > 0 \text{ such that } \epsilon_\delta^{R_k}(z_k) \geq \kappa, \tag{7.7}$$

for all k sufficiently large. From (6.18), we see that $\epsilon_\delta^R(z)$ is equal to 0 if and only if $\sigma_{ij_i}(z) = 0$, for $i = 1, \ldots, I$, and $j = 1, \ldots, R$. We then have a multinomial logit model, instead of a mixed logit, and the simulated likelihood function value is then independent of the sampling. Consequently, the fact that (7.7) is not satisfied is merely an indication that the multinomial logit solution is a limit point of the iterates, and that the mixed logit formulation is probably inappropriate. The algorithm could then be terminated with a sample size less than its maximum, as described in Step 1 of Algorithm 2.2. However, the maximum sample size is always reached in our numerical experimentations, even when testing multinomial logit models. This is explained by the fact that we approximate the $\sigma_{ij_i}(z)$ by $\sigma_{ij_i}^R(z)$ and that small standard deviations are not easy to recover since their influence in the model is weak. Therefore, the error term remains positive and $R_{\max}$ is always reached in the final iterations.

7.2 AMLET

Algorithm 7.1 has been incorporated in the software AMLET (for Another Mixed Logit Estimation Tool), that has been developed in order to validate the proposed methodology. AMLET is entirely written in C and runs in a Linux environment. For efficiency purposes, the basic linear algebra computations are undertaken with the libraries ATLAS (Whaley, Petitet and Dongarra [141]) and CLAPACK (Anderson et al. [3]). We have also implemented two random generators for the Monte Carlo simulations: the standard minimial generator (Park and Miller [107])

and L'Ecuyer [85]. The implementation follows the guidelines proposed by Press et al. [112]. The two methods has given good, similar, results during our tests; the experiments that we report in the next sections have used the standard minimal generator.

The package allows the user to solve mixed or multinomial logit models from existing data, or to set up simulated data corresponding to a user-defined model structure. AMLET computes parameters estimators, classical tests for goodness of fit (as described in Ben-Akiva and Lerman [12] and Ortúzar and Willumsen [105]), and some specific information such as estimations of the simulation bias and log-likelihood accuracy. An example of output is given in Appendix A.2.

7.3 Numerical assessment of AMLET

In this section, we compare results obtained with AMLET on synthetic and real data to those obtained using Gauss 5.1 and the MaxLik module (Schoenberg [124]), in which we have used Halton sequences and the code written by Train [134] (we consider here the version without panels). All tests reported were obtained using AMLET under the gcc 2.95 compiler with optimisation on a Pentium IV 2Ghz. Gauss, on the other hand, was installed on a Pentium III 600Mhz, with 256 Mo of memory. In order to correct for the slower computer, we have computed corrected timings for Gauss (in brackets in the tables below) by multiplying the Pentium III times by execution time ratio (time on Pentium IV divided by time on Pentium III, approximately 0.37) for the solution of a test problem described in Section 7.3.5 with 1000 Monte Carlo draws. For future experimentations, we could also use Ox (Doornik [45]), whose command line versions are free for academic research. For the estimation of the standard deviation and bias, we use a level of signification $\delta = 0.9$. The tests cover five different questions:

- the validity of the bias and accuracy estimators,
- comparisons of different algorithmic options in the optimization process,
- the performance and robustness of the package on simulated data,
- comparisons with the Gauss package using Halton sequences, and
- the performance of the package on real data.

In experiments using simulated data, we always use linear utility functions. Furthermore, the attribute values are drawn from a standard univariate normal distribution $N(0, 1)$ and the coefficient of each independent variable is also drawn from an univariate normal distribution $N(0.5, 1)$. The error term is then generated from an extreme value (Gumbel) distribution, ensuring that the conditional choice probability follows the logit formula (6.6). This allows us to compute the utility of each alternative, including observed and unobserved terms. The individual choice is then identified for each observation as the alternative with the highest utility. The optimization starting point is defined by setting all initial parameter values to 0.1.

7.3.1 Validity of bias and accuracy estimators

Since the bias and accuracy estimates (6.20) and (6.18) are only valid asymptotically as R increases, it is useful to assess them numerically for practical purposes, in particular when the simulation bias is significant. In order to assess these formulae, we consider a synthetic population of 5000 individuals facing 5 alternatives (one of which is the null alternative) for which the utilities involve 5 parameters. We vary the sample sizes using 500, 1000, 2000, 3000, 4000, 5000 random draws per individual. For each fixed number of random draws, we minimize 36 different SAA approximations, resulting in 36 slightly different solutions (with nonnegative components). We then compute the mean of these 36 optimal values to obtain a "mean optimal value", which we will denote by $\overline{SLL^R_*}$, and which can be viewed as an estimator of $E[SLL^R_*]$ with

$$SLL^R_* = \max_{\theta \geq 0} SLL^R(\theta).$$

(The variance of this estimator is equal to σ^2/N, where σ^2 is the variance of SLL^R_* and $N = 36$.) For each of the 36 solutions, we furthermore estimate the standard deviation due to the sampling effect by recomputing the log-likelihood at the solution with 36 new samples; the mean of these newly estimated log-likelihood values is an estimator of $E[SLL^R]$ at the considered solution, that we will denote by $\hat{E}[SLL^R]$. We finally computed the average of these 36 $\hat{E}[SLL^R]$, denoted by $\overline{\hat{E}[SLL^R]}$. We also estimate the model with a sample of 12000 random draws per individual and use the estimated log-likelihood value as reference. The optimal simulated log-likelihood value is then -1.44167087, for an estimated accuracy equal to $2.3434.10^{-4}$ and a bias of $-5.0742.10^{-5}$. The size limit of 12000 draws has been imposed by memory limitations of the used computer, with respect to the current implementation of AMLET. Results are reported in Table 7.1.

We observe that estimated and numerical standard deviations are similar, suggesting that the approximation (6.18) is adequate, even when the simulation bias is of the same order (for instance for the case of 500 draws). The evolution of the estimated bias reflects well that of $\overline{\hat{E}[SLL^R]}$, but the parallelism is less clear between the bias and the mean optimal value $\overline{SLL^R_*}$ (note the decrease in this quantity when the number of draws changes from 2000 to 3000, while $\overline{\hat{E}[SLL^R]}$ increases). This can be explained by the standard error of the mean estimator since, for instance, the confidence interval radius at level 0.95 for $\overline{SLL^R_*}$ can be estimated to be 0.000091 and 0.000070 for the cases of 3000 and 5000 random draws, respectively. The parallelism between bias and $\overline{\hat{E}[SLL^R]}$ is particularly remarkable since this last value is, in our results, approximated from the mean over 36 different (but close) values of θ (resulting in 1296 different samplings), instead of being recomputed for a single θ with a sufficiently high number of samplings. If all our θ values were identical (and in the neighbourhood of the calculated 36 values), the associated confidence interval radii at level 0.95 would be approximately 0.000015 and 0.000012 for 3000 and 5000 random draws respectively, and the influence of bias could then be detected. We finally note that the estimated bias is greater in absolute value than the difference between the mean optimal value and the reference value obtained with 12000 draws. This difference can however be explained by the bias present in the reference value, combined with the simulation errors. In practice, the simulation bias is therefore difficult to quantify when only the approximating optimal values are available, since it is masked by the variance of the simulated log-likelihood

Number of draws	**500**	**1000**	**2000**	**3000**	**4000**	**5000**
Mean optimization time (s)	73	161	324	471	574	781
Mean optimal value	-1.44273663	-1.44210903	-1.44178060	-1.44181801	-1.44173967	-1.44167739
Mean optimal value difference	-	0.00062760	0.00032843	-0.00003741	0.00014864	-0.00000802
Estimated standard deviation	0.00069299	0.00049190	0.00034825	0.00028452	0.00024658	0.00022058
Numerical standard deviation	0.00069968	0.00047864	0.00034943	0.00028007	0.00024944	0.00021436
Estimated accuracy	0.00113987	0.00080911	0.00057281	0.00046799	0.00040559	0.00036282
Estimated bias	-0.00120060	-0.00060492	-0.00030319	-0.00020239	-0.00015201	-0.00012162
Bias difference	-	0.00059568	0.00030174	0.00010080	0.0005039	0.00003038
Average $\hat{E}[SLL^R]$	-1.44273888	-1.44214435	-1.44183149	-1.44173967	-1.44168272	-1.44165735
Average $\hat{E}[SLL^R]$ difference	-	0.00059453	0.00031286	0.00009182	0.00005695	0.00002537
Mean difference with reference value	-0.00106576	-0.00043816	-0.00010973	-0.000147146	0.0000014914	-0.000011856

Table 7.1: Validation of error and bias estimation

values.

Table 6.18 also exhibits the slow convergence in $O(\sqrt{R})$ of Monte Carlo approximations. However we observe that the bias decreases in absolute value faster than the error, this last reduction being in $O(R)$, as predicted by (6.20). It is also interesting to note that none of the 216 numerical optimization processes required unexpectedly large computing times, but some variation remains that depends on the generated samplings.

7.3.2 Algorithmic options for optimization

Solving (7.1) requires the use of numerical optimization procedures, amongst which one of the most popular for calibrating discrete choices models is the BFGS linesearch method (see Train [135], pages 225–226). This method is based on using a quasi-Newton direction where the Hessian of the log-likelihood function is approximated by the well-known BFGS secant formula. A suitable step length is then computed along this direction to yield the final step. This method is acknowledged to be efficient whenever the function to optimize is concave. As this is not the case for mixed logit models, it is thus interesting to evaluate the relevance of trust-region methods, the contending methodology described in Section 7.1 which specialized in nonconcave problems and uses the SR1 secant formula.

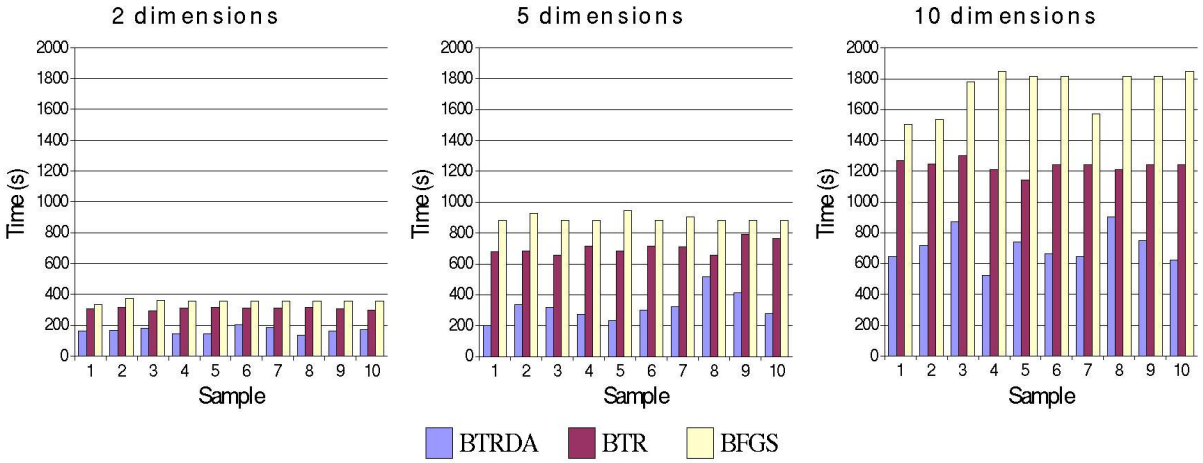
For comparison purposes, we therefore coded a BFGS algorithm, as described by Nocedal and Wright [104], using the efficient linesearch technique by Moré and Thuente [101] and associated code. We then simulated three populations of 5000 individuals facing five alternatives (one of which is the null alternative), with associated utilities involving respectively two, five and ten parameters. We used 2000 random draws and compared the BFGS, BTR and BTRDA algorithms for solving the SAA problem (7.1). For each dimension, the procedure was repeated for 10 different samplings.

The three algorithms give similar optimal log-likelihood values for a particular sampling, that are indistinguishable in view of the simulation accuracy. Figure 7.1 summarizes the computing times for the three methods.

Computing the geometric mean of the ratios between mean optimization times (see Appendix A.1), we obtain that BTR typically requires 79% of the time required by the BFGS algorithm. BTRDA however requires on average only 52% of the time needed by BTR, and 41% of the BFGS time. Moreover, for each simulation, BTRDA is the fastest method, followed by BTR. These results therefore comfort our choice of a trust-region approach combined with the variable sample size strategy.

7.3.3 Performance and robustness on simulated data

We next turn to experiments involving more complex simulated data. Two simulated experiments varying the number of observations and alternatives provide the framework of our evaluation. Both test the methodology on four separate integration dimensions (2, 5, 10 and 15). For each of the simulated cases, we then estimated the model 10 times and computed the average optimization time, accuracy (6.18) and simulation bias (6.20). For each parameter, we also computed the standard error (calculated as the square root of the average of the squared standard error from individual runs, as in Sandór and Train [123]) and the relative error, defined as the



	2 dimensions	5 dimensions	10 dimensions
BTRDA	$167s$	$319s$	$709s$
BTR	$310s$	$706s$	$1235s$
BFGS	$357s$	$894s$	$1736s$

Figure 7.1: Comparison of CPU times for the optimization algorithms

standard error divided by the mean value of the parameter. We finally average the relative errors over the parameters to obtain an indicator of the stability of the parameters when the sampling changes.

We first vary the number of observations over 2000, 5000, 7500 and 10000, while the number of choice alternatives is kept fixed to 5. We limit the sample size to 1000, mainly due to memory limitations.

The results presented in Figure 7.2 then show that the optimization time increases with the number of observations at an approximately linear rate while remaining manageable. The estimated error (Figure 7.3) decreases with the number of observations, while at the same time the bias remains of the same order. The ratio between the bias and the accuracy therefore increases, which makes the application of the delta method as in (7.4) more questionable as the number of observations grows for fixed sample size (as stated from the theoretical point of view by Hajivassiliou and McFadden [70]). The number of dimensions also produces growth in the absolute value of the estimated accuracy and bias.

We next vary the number of alternatives in the choice model amongst 2, 3, 5, and 10 (one of which being the null alternative); the number of simulated individuals remains fixed to 5000, and the maximum sample size is set to 2000.

From Figure 7.4, we see that the optimization time decreases from 2 to 5 alternatives and then increases from 5 to 10 alternatives. Note also that the average relative error tends to increase with the number of parameters. A possible interpretation of these phenomena is that having more alternatives makes the discrimination of individual choices easier, leading to a sharper maximum of the simulated likelihood function.

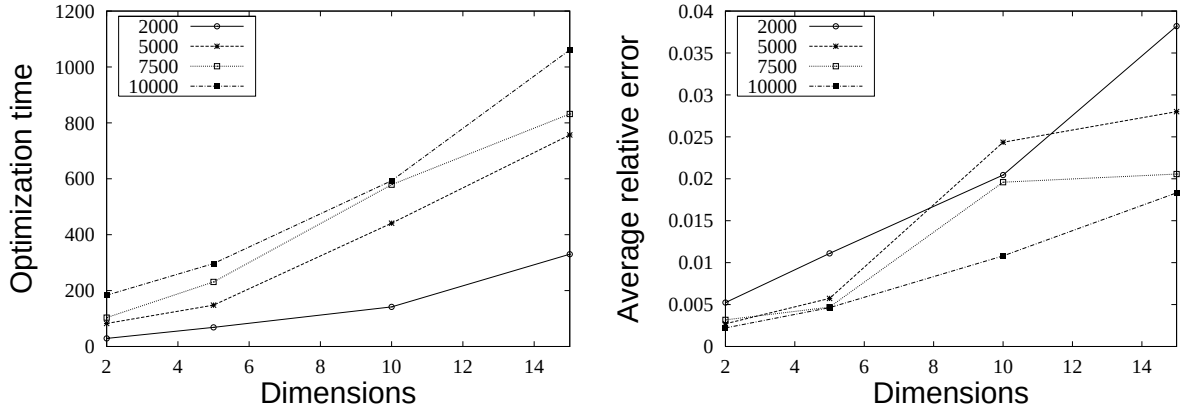


Figure 7.2: Evolution of time and average relative error with the number of observations

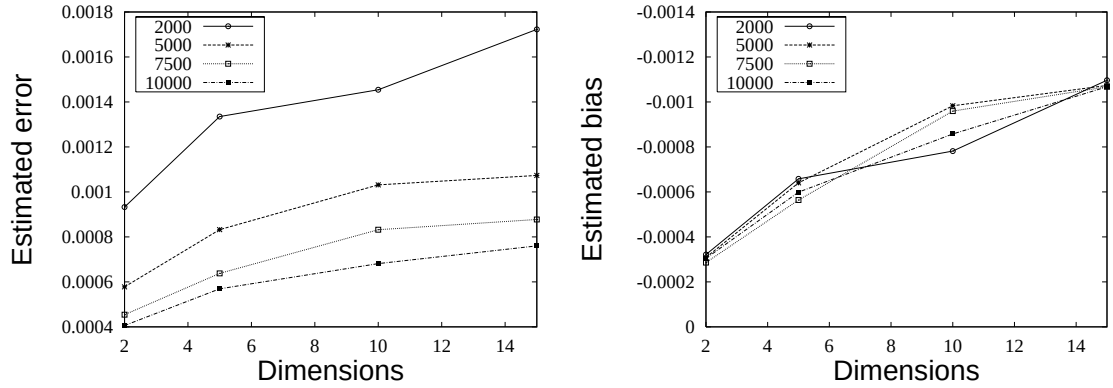


Figure 7.3: Evolution of estimated error and bias with the number of observations

The estimated simulation error and bias (in absolute value) grow with the number of alternatives (Figure 7.5), while the average relative error decreases when the number of alternatives increases. This suggests that the stability of the objective function is far from providing a complete view of the quality of the derived estimations, as already suspected by Bhat [16]. An important topic of research therefore remains to improve error estimates on the parameters when using Monte Carlo approximations.

7.3.4 Comparison with Halton sequences

A major drawback to Monte Carlo methods is their slow convergence rate as a function of the sample size, as shown in Table 7.1. Quasi-Monte Carlo methods try to speed up this convergence by picking up points that are more uniformly distributed in the integration domain. One popular such technique is the generation of Halton sequences, which are used for instance by Train [134] in his Gauss package.

In the unidimensional case, one chooses a prime number b ($b \geq 2$), and expands the se-

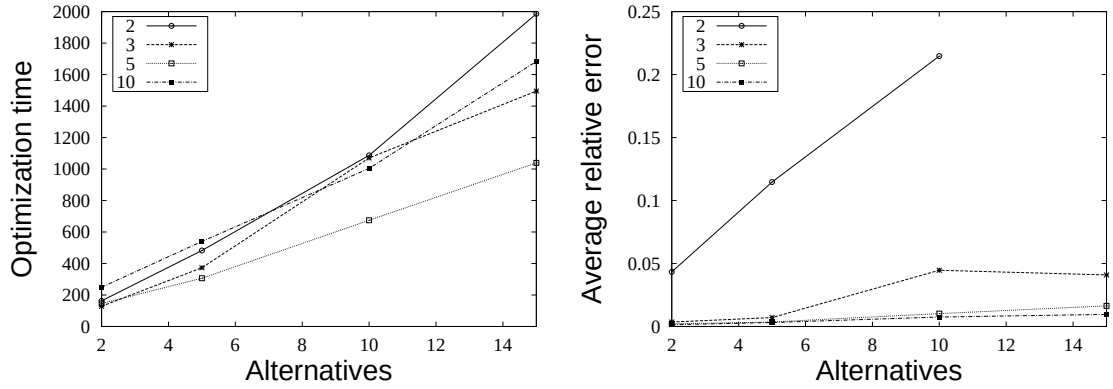


Figure 7.4: Evolution of optimization time and average relative error with the number of alternatives

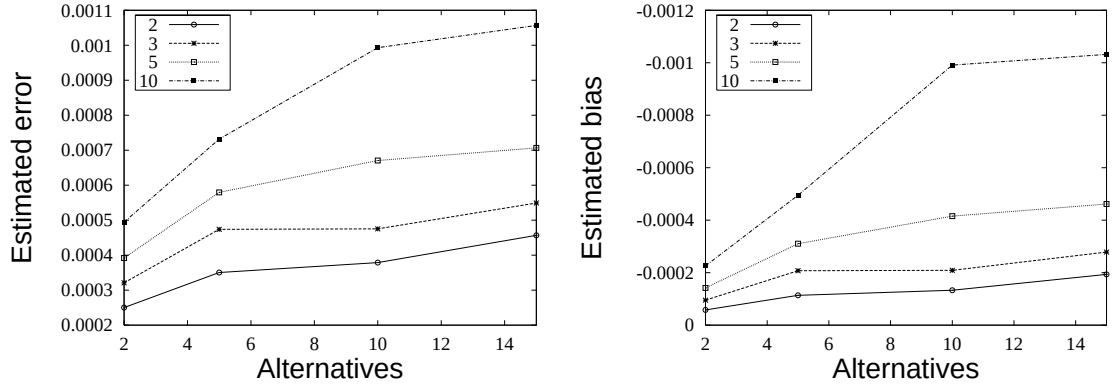


Figure 7.5: Evolution of estimated error and bias with the number of alternatives

quences of integers $1, 2, \ldots$ in terms of the base b. For each integer g, we have

$$g = \sum_{l=0}^{L} \alpha_l(g) b^l, \tag{7.8}$$

where $0 \leq \alpha_l(g) \leq b - 1$ and $b^L \leq g < b^{L+1}$. The order of the digits $\alpha_l(g)$ is then reversed and a radix point placed in front of the sequence, yielding

$$\phi_b(g) = 0.\alpha_0(g)\alpha_1(g) \ldots \alpha_L(g).$$

For example, the Halton sequence associated to base 3 is

$$\frac{1}{3}, \frac{2}{3}, \frac{1}{9}, \frac{4}{9}, \frac{7}{9}, \frac{2}{9}, \frac{5}{9}, \frac{8}{9}, \ldots$$

To obtain a sequence of n-tuples in n-space, we build a Halton sequence for each component with a different prime base b. Typically, the first n primes are used.

When the number of random parameters is small (typically less than six), approximation of the log-likelihood function based on Halton sequences usually gives the same results as pure Monte Carlo sampling, but uses less draws. This is reflected in Table 7.2, where we consider a simulated population of size 2000 with a model that has five parameters and five alternatives. Results for AMLET are averaged over ten simulations. We see also that AMLET is still more than competitive with Gauss in terms of computing time.

Variable	Gauss		AMLET	
	125 Halt.	250 Halt.	1000 MC	2000 MC
P1 mean	0.4588	0.4587	0.457262	0.459014
P1 std. dev.	1.0854	1.0837	1.06473	1.06628
P2 mean	0.4489	0.4530	0.447979	0.451381
P2 std. dev.	1.1620	1.1622	1.17201	1.18576
P3 mean	0.5704	0.5697	0.567744	0.570876
P3 std. dev.	1.1212	1.1044	1.09374	1.109960
P4 mean	0.6165	0.6212	0.613718	0.617642
P4 std. dev.	1.3386	1.3677	1.33244	1. 34054
P5 mean	0.6548	0.6565	0.645966	0.648561
P5 std. dev.	1.2558	1.2360	1.21282	1.22422
Log-likelihood	-1.44770	-1.44784	-1.44835	-1.44813
Bias	NA	NA	-0.000826983	-0.000415893
Accuracy	NA	NA	0.00149579	0.00106076
Optimization time (s)	959 (355)	1649 (611)	75	149

Table 7.2: Halton and Monte Carlo samplings for 5 random parameters

When the number of random parameters increases, results deteriorate as illustrated in Table 7.3, where ten parameters and 5 alternatives are considered for a simulated population of size 5000. Here 250 Halton draws are needed to obtain similar results to those found by AMLET with 1000 random draws. However these results can be further refined with AMLET by taking 2000 or 3000 random draws[1], while using substantially more Halton draws become computationnaly intractable. Computation times clearly favour AMLET since the optimization time used by this package is smaller than that used by Gauss with Halton draws, even when 3000 random draws are compared to 125 Halton draws.

Another experiment, reported in Table 7.4, considers a case where only 2000 individuals are observed, and indicates, surprisingly, that results obtained with Gauss deteriorate when we take 250 Halton draws instead of 125 Halton draws. This problem might reflect the fact that Halton sequences are often less suitable for high-dimensional problems because the sequences associated to successive prime numbers exhibits correlations as the prime numbers increase and the space is not covered as uniformly in this case (see Figure 7.6, where 250 Halton draws are represented). This difficulty can possibly be addressed by considering scrambled Halton

[1]In Euclidian distance, any of the 10 solutions found by AMLET with 2000 random draws is closer to the mean solution obtained with 2000 or 3000 draws than the Halton solution, which is close to the mean of the solutions obtained with 1000 random draws.

Variable	Gauss		AMLET		
	125 Halt.	250 Halt.	1000 MC	2000 MC	3000 MC
P1 mean	0.4022	0.4371	0.438871	0.452648	0.452363
P1 std. dev.	0.9575	1.0546	1.07166	1.10767	1.10093
P2 mean	0.4237	0.4586	0.461841	0.477684	0.476018
P2 std. dev.	0.8423	0.8646	0.891689	0.927504	0.929728
P3 mean	0.3903	0.4305	0.429788	0.443977	0.444003
P3 std. dev.	0.7959	0.8812	0.884533	0.936943	0.933369
P4 mean	0.4700	0.5129	0.517726	0.534389	0.534119
P4 std. dev.	0.6744	0.7205	0.745905	0.775687	0.779972
P5 mean	0.4808	0.5308	0.532857	0.551394	0.551530
P5 std. dev.	0.7027	0.8312	0.849630	0.886852	0.886516
P6 mean	0.3782	0.4100	0.413706	0.426934	0.425777
P6 std. dev.	0.8297	0.9163	0.950053	0.989986	0.984977
P7 mean	0.3920	0.4337	0.436586	0.451637	0.450216
P7 std. dev.	0.9116	1.0398	1.03758	1.08211	1.07664
P8 mean	0.4895	0.5310	0.535650	0.553087	0.552532
P8 std. dev.	0.9198	1.0022	1.02415	1.05907	1.06029
P9 mean	0.4441	0.4768	0.482356	0.499464	0.498142
P9 std. dev.	0.9048	0.9694	0.979119	1.02628	1.02319
P10 mean	0.3702	0.4149	0.413833	0.427490	0.425641
P10 std. dev.	0.6250	0.8410528	0.810528	0.837880	0.839898
Log-likelihood	-1.44244	-1.43990	-1.44132	-1.44086	-1.44076
Accuracy	NA	NA	0.0010331	0.0007410	0.0006046
Bias	NA	NA	-0.00098627	-0.0005073	-0.0003378
Time (s)	5608 (2077)	11480 (4252)	417	868	1281

Table 7.3: Halton and Monte Carlo sampling for 10 random parameters (5000 individuals)

sequences (Bhat [15]), in which more general permutations are applied to the order of the digits in (7.8). As an alternative, Hess, Train and Polak [76] also propose the use of randomly shifted uniform vectors. We plan to pursue the research described here by comparing our results with those obtained with such techniques.

7.3.5 Performance on a real data set

We finally test our algorithm on a real data set obtained from the six-week travel diary Mobidrive (Axhausen, Zimmerman, Schönfelder, Rindsfúser and Haupt [6]) collected in the spring and fall 1999 in Karlsruhe and Halle (Germany). The mixed logit models aims at explaining individual modal choice across five alternatives (car driver, car passenger, public transport, walk and bike). The framework applied considers the daily activity chain, in that the individual pattern is divided into tours, which are themselves defined as the sequence of trips starting and ending at home or at work, both being considered as fixed locations. Details and motivation for the model structure can be found in Cirillo and Toint [27].

In this data set, we restricted our attention to the observations from Karlsruhe because level of service variables (i.e. time and cost for various modes) were available for this location only. The sample then includes approximately 160 households and 360 individuals. After data cleaning,

Variable	Gauss		AMLET	
	125 Halt.	250 Halt.	1000 MC	2000 MC
P1 mean	0.4432	0.4072	0.429672	0.443550
P1 std. dev.	1.0336	0.9438	0.988477	1.02280
P2 mean	0.4837	0.4409	0.462633	0.477523
P2 std. dev.	0.5929	0.5264	0.528293	0.573141
P3 mean	0.4652	0.4438	0.460063	0.474699
P3 std. dev.	0.6226	0.6013	0.606384	0.628658
P4 mean	0.3878	0.3593	0.376619	0.390012
P4 std. dev.	1.0374	0.9715	1.01980	1.07015
P5 mean	0.4576	0.4118	0.434051	0.447357
P5 std. dev.	0.8978	0.7725	0.834783	0.882849
P6 mean	0.4219	0.3920	0.407852	0.420751
P6 std. dev.	1.0214	0.8921	0.946411	0.986813
P7 mean	0.4977	0.4645	0.487255	0.499920
P7 std. dev.	-0.6172	0.5375	0.633064	0.654150
P8 mean	0.4339	0.4008	0.418941	0.431671
P8 std. dev.	0.9481	0.7917	0.898677	0.925221
P9 mean	0.4786	0.4442	0.470780	0.485152
P9 std. dev.	1.0760	0.9738	1.02480	1.05215
P10 mean	0.5014	0.4561	0.482453	0.495858
P10 std. dev.	0.8562	0.7598	0.830905	0.853791
Log-likelihood	-1.43714	-1.43785	-1.43767	-1.43676
Bias	NA	NA	-0.0009082	-0.0004681
Accuracy	NA	NA	0.001567	0.001125
Time (s)	2092 (775)	2911 (1078)	179	400

Table 7.4: Halton and Monte Carlo sampling for 10 random parameters (2000 individuals)

5799 tours were retained for the parameter estimation procedure. As several tours are performed by the same individuals, the data therefore contains significant correlations. At this stage of development of AMLET, this cannot yet be taken into account but we hope to include panel analysis facilities in the package soon. For further details on mixed logit on the Mobidrive panel data set, see Cirillo and Axhausen [26].

Our model contains 14 variables, four of which are alternative specific constants (car driver being the base). We estimate household location characteristics (urban/suburban location), individual socio-demographic variables (female and working part time, being married with children, annual car mileage), tour variables (number of stops), pattern variables (time budget) and level of service variables (time and cost). We specify a mixed logit model with fixed coefficients except for time, cost and time budget, which are expected to vary considerably across observations, and are assumed to be normally distributed. The value of time is 9.55 DM (about 4.9 Euros), which is comparable to that used in other European studies (see TRACE [133]).

The model has been estimated both with Gauss and AMLET, using the same starting point. Results are summarized in Table 7.5, where only one simulation is used for each column, the four

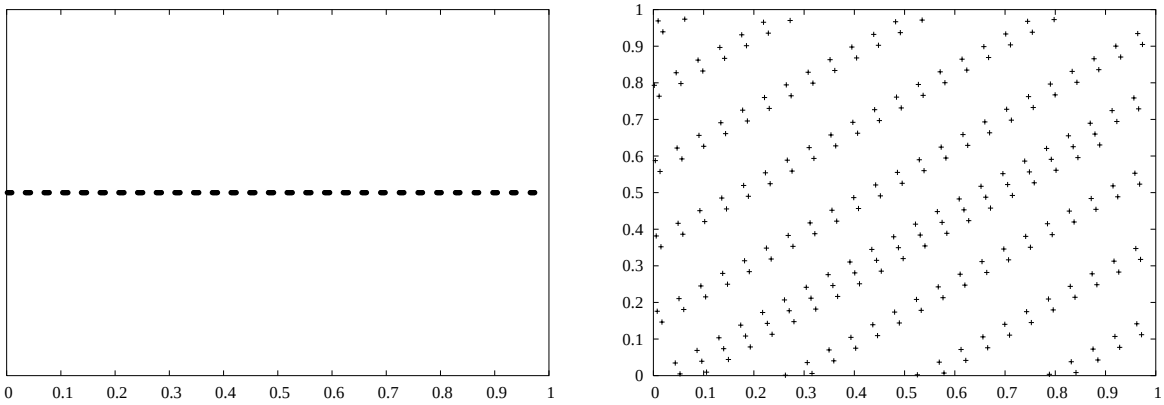


Figure 7.6: Difficulties with standard Halton draws

first parameters being the alternative specific constants. The alternatives associated to other non-generic parameters are indicated between brackets, next to the parameter name. The following abbreviations have been used: CD for car driver, CP for car passenger, PT for public transport, W for walk and B for bike.

AMLET results are similar to those obtained by Gauss, supporting the observation that good results can be obtained with a small number of Halton draws, at least when the integration dimension is low. However, the optimization time of AMLET is very competitive with that of Gauss, since they are similar for 2000 random draws and 125 Halton draws, and AMLET is clearly faster for a sample size of 1000. We have also computed an average solution with AMLET, by running ten simulations. The average optimization time is then 900 seconds with 1000 random draws and 1585 seconds with 2000 random draws. If we use the BTR algorithm, AMLET takes in average 1758 and 3431 seconds with 1000 and 2000 random draws, respectively, so we observe a speed-up factor of approximately 2 when using our sample size strategy.

The crucially beneficial effect of the variable sample size strategy is illustrated in Figure 7.7, giving the evolution of the sample size R_k with the iteration index k. The left graph corresponds to a maximum sample size of 1000 while the right graph has been obtained with a maximum of 2000 random draws. Furthermore, Figure 7.8 shows that the sample size increases towards its maximum value only when the objective function's value is near to its maximum, and is very small when we are far from the solution. The graphs correspond again to 1000 (left) and 2000 (right) random draws.

7.3.6 Discussion

Due to the complexity of the objective function in mixed logit models, the choice of the optimization procedure is of crucial importance. First of all, the speed of convergence can be dramatically increased if the available information is exploited. In our case, we see that the estimation of the standard deviation allows us to speed up the initial iterations by using smaller

Variable \ Software	Gauss (125 Halton)		AMLET (1000 Monte Carlo)		AMLET (2000 Monte Carlo)	
	Coefficient	t-statistic	Coefficient	t-statistic	Coefficient	t-statistic
Car Passenger (CD)	-1.4511	17.678	-1.44109	17.7127	-1.44937	17.7045
Public Transport (PT)	-0.9355	6.869	-0.934355	6.92116	-0.935232	6.86949
Walk (W)	0.1081	0.731	0.103739	0.708912	0.106154	0.718496
Bike (B)	-0.6355	4.614	-0.633507	4.65294	-0.636258	4.62297
Urban household location (PT)	0.5609	5.133	0.552985	5.1011	0.557515	5.11427
Suburban household location (W, B)	-0.3451	3.942	-0.346059	4.0146	-0.343104	3.9256
Full-time worker (PT)	0.2690	2.890	0.272323	2.94153	0.270477	2.90782
Female and part-time (CP)	0.9133	8.711	0.901266	8.6397	0.911615	8.70669
Married with children (CD)	0.9716	11.574	0.963306	11.5908	0.971295	11.5835
Annual mileage (CD)	0.0518	12.096	0.0511034	12.0115	0.051793	12.0896
Number of stop (CD)	0.1349	3.030	0.13049	2.96802	0.133826	3.01395
Time mean	-0.0268	9.868	-0.0261824	9.65188	-0.0268215	9.89273
Time std. dev.	0.0205	6.024	0.0197855	5.79468	-0.0206027	6.0841
Cost mean	-0.1683	11.356	-0.166767	11.2892	-0.168155	11.3176
Cost std. dev.	-0.0452	2.958	0.0450306	2.60626	-0.045562	2.84257
Time budget mean (CD, CP)	-0.1249	7.773	-0.12259	7.71894	-0.124719	7.77774
Time budget std. dev. (CD, CP)	-0.1136	4.932	0.103746	4.27117	0.112292	4.88769
Log-likelihood	-1.16489		-1.165337		-1.164849	
Bias	Not available		-0.0000824754		-0.0000452832	
Accuracy	Not available		0.000277414		0.000205558	
Optimization time (s)	6585 (2439)		871		1652	

Table 7.5: Tests on real data set

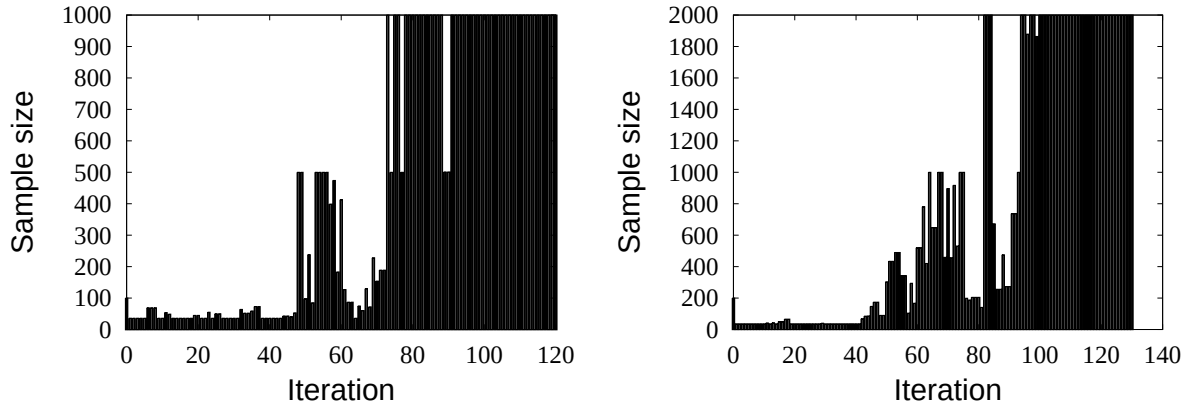


Figure 7.7: Variation of sample sizes with iterations

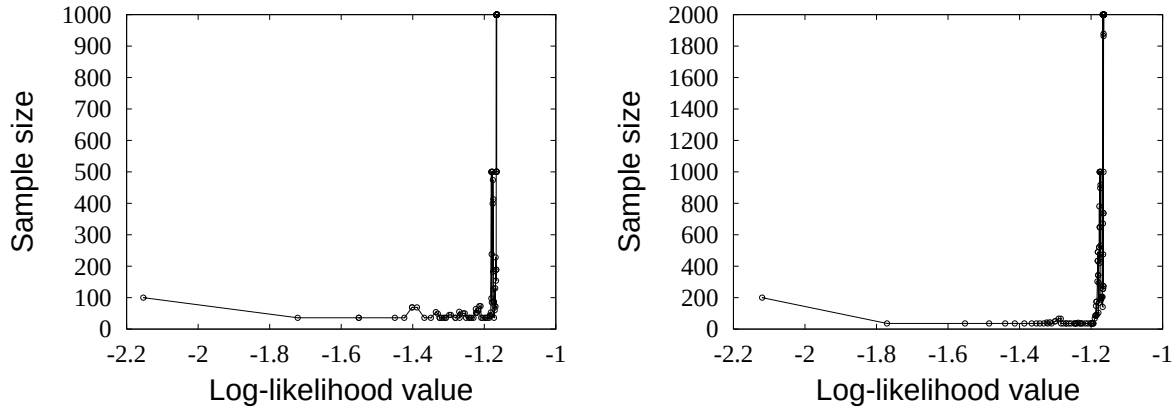


Figure 7.8: Variation of sample sizes with log-likelihood value

samples, and often to stop earlier. Secondly, important savings can be achieved by taking the problem properties into account in order to avoid unnecessary computations. For instance, we evaluate the function and its gradient analytically at the same time, instead of successively.

Quasi-Monte Carlo methods produce the same accuracy with less random draws, at least in low and medium integration dimensions, but this accuracy is difficult to quantify in practice, while this is easy for Monte Carlo approaches. As our algorithm exploits this information, its application to quasi-Monte Carlo techniques is not as direct as with pure random draws. Moreover usual problems in high-dimensional integration with quasi-Monte Carlo methods, such as correlations, do not occur in pure Monte Carlo procedures. Consequently, the latter are often more robust, both theoretically and numerically. Our procedure can therefore be seen as a compromise between speed and the exploitation of theoretical information. More research is however needed to apply the same philosophy to quasi-Monte Carlo sequences.

A Appendix

A.1 Performance comparisons and use of means

When comparing two algorithms, we would like to express the relative with some index. In order to achieve this goal we typically take a set of running tests on the machine and compute some sort of mean. There are a number of different ways to define a mean value; among them the arithmetic mean, the geometric mean, and the harmonic mean. Jacob and Mudge [78] illustrates that these different means are not equal, leading to different conclusions.

Consider n numbers $a_1, \ldots, a_n$. The arithmetic mean is defined as

$$AM(a_1, \ldots, a_n) = \frac{\sum_{i=1}^{n} a_i}{n},$$

while the geometric mean is

$$GM(a_1, \ldots, a_n) = \sqrt[n]{\prod_{i=1}^{n} a_i}.$$

Finally, the harmonic mean is defined as

$$HM(a_1, \ldots, a_n) = \frac{n}{\sum_{i=1}^{n} \frac{1}{a_i}}.$$

Note that the harmonic mean is the inverse of the arithmetic mean:

$$AM(a_1, \ldots, a_n) = \frac{1}{HM\left(\frac{1}{a_1}, \ldots, \frac{1}{a_n}\right)}. \tag{7.9}$$

When comparing running of two algorithms we can also compute the ratio of arithmetic means:

$$RAM = \frac{\sum_{i=1}^{n} t_i}{\sum_{i=1}^{n} t'_i},$$

where t_i and t'_i are the times taken respectively by the first and the second algorithm, on test problems i, $i = 1, \ldots, n$.

Consider for instance that we wish to evaluate two algorithms, and that we have two problems tests. The first algorithm take 3 and 300 seconds for the first and the second problem, respectively, while the second method consume 12 and 600 seconds. Consider the ratios of the running times:

$$test\ 1 : \frac{3}{12} \quad test\ 2 : \frac{300}{600}.$$

Then $AM = 3/8$, so the first algorithm is 8/3 times as fast as the second (reference) algorithm. On the other hand, $HM = 1/3$, so the first algorithm is three times faster than the second. On the other hand $GM = \sqrt{2}/4$. Finally, $RAM = 303/612 \approx 0.495$. Each performance measure gives a different answer, so which one have we to use? Each one gives in fact an answer to a different question.

Let a_i, $i = 1, \ldots, n$, be running times obtained when using a first algorithm, and a'_i, $i = 1, \ldots, n$, the corresponding running times for a second algorithm. The arithmetic mean of the set of ratios is then equal to

$$\frac{1}{n} \sum_{i=1}^{n} \frac{a_i}{a'_i},$$

it is a weighted average where the weights are the inverses of the running times of the second algorithm. By (7.9) the arithmetic and geometric means differ only by the choice of the reference algorithms. The geometric mean has the advantage that it is irrelevant to the choice of the reference algorithm, but suffers from less intuitive interpretation.

The ratio of arithmetic means of running times provides an answer to the question "how much time will be saved by running an algorithm compared the reference one". For this we have to test typical programs that we will have to solve, and to weight them with their frequency when computing the average. While this approach is intuitively appealing, the construction of a test set is not an easy task, and depends on the final user. Since in our context the applications can vary a lot between users, the ratio of arithmetic means is not of huge interest. The choice of an index of comparison is therefore more trickier that it could appear at a first sight. We privilege here the geometric mean for its irrelevance to the choice of the reference algorithm.

A.2 Example of AMLET's output

We reproduce below one output obtained during our tests on the Mobidrive data set. The maximum number of random draws per individual is fixed to 2000. Note that for convenience of presentation, we have omitted the variance-covariance and correlation matrices in the listing below, while they are available in AMLET's output.

```
Time spent: 1712.730 seconds
Number of observations: 5799

Method used for the optimization: mcbtrda
Time consumed during the optimization process: 1693 seconds
Number of iterations: 139
Function evaluations: 141
Seed used for the random generator: 7104900
Final number of random draws: 2000

Log-likelihood value at zero: -1.609437912
Log-likelihood value for specific constants: -1.264117075
Log-likelihood value: -1.164563086
Gradient norm: 1.183717455e-05
Accuracy of the likelihood function evaluation: 0.000210895
Bias of simulation: -4.76652e-05

-------------------------------------------------------------
          Estimation  Standard deviation   t-statistic
-------------------------------------------------------------
   0        -1.45687      0.00676958         17.7068    *
```

```
   1       -0.934566       0.0187581         6.82364    *
   2         0.11002       0.0221068         0.73996
   3       -0.635721       0.0191581         4.59293    *
   4        0.565902       0.0120608         5.15292    *
   5       -0.347151      0.00774181         3.94545    *
   6        0.267496      0.00873135         2.86271    *
   7        0.918413        0.011402         8.60098    *
   8        0.973428      0.00713988         11.5202    *
   9       0.0520725     1.84648e-05         12.1182    *
  10        0.136972      0.00200753         3.05703    *
  11      -0.0271638     7.42529e-06         9.96859    *
  12       0.0209935      1.1595e-05         6.16526    *
  13       -0.169869     0.000219132         11.4753    *
  14       0.0466559     0.000228449         3.08682    *
  15       -0.125882     0.000262423         7.77072    *
  16        0.116828     0.000495892         5.24632    *
---------------------------------------------------------


rho-squared: 0.276416
adjusted rho-squared: 0.274595

Likelihood ratio test
---------------------

Chi-square value for l(0): 5159.66
Level of rejection: 1
```

Conclusions and further research perspectives

The research work described in this thesis is concerned with the study of nonlinear stochastic programming, more particularly with the design of algorithms for solving classes of nonlinear stochastic programs. In order to efficiently solve a stochastic program, we have to take its structure into account, and to adapt algorithms in consequence. In this thesis, we have considered two classes of problems: nonlinear stochastic problems with a split-variable formulation and general nonlinear programs involving continuous variables or discrete variables with a large number of possible realizations, which are approximately solved using Monte Carlo samplings. For both of them, we have designed trust-region based algorithms that take their properties into account.

Interior point methods for scenario formulations

We have first described a primal-dual interior point algorithm where the structure of the primal-dual systems is exploited to favour parallelization of the implementation. We have showed that, under some assumptions, the decomposition scheme is well-defined, while the global structure of the algorithm, based on existing primal-dual trust-region methods, and the use of a dogleg-path for the model minimization, ensures that the algorithm is convergent.

More experimentation is however needed to numerically confirm our conclusions, in particular in order to assess the gains in computation times obtained with the parallelization. Moreover, recent developments in linear and convex stochastic programming are based on the exploitation of the sparse structure related to the formulation in extension (see for instance Gondzio and Grothey [62], Gondzio and Sarkissian [63], Steinbach [130]). This offers the benefit to avoid the variables multiplication, but restricts the possibilities of parallelization. It would therefore be interesting to explore adaptation of such techniques to the nonconvex case, and to compare the numerical performances of the different approaches.

Monte Carlo samplings

A second part of this work is the study of Monte Carlo techniques for stochastic nonlinear nonconvex programs, both on a theoretical level and on a practical one.

Consistency of Monte Carlo approximating solutions

We have first studied the consistency of solutions obtained when solving sample average approximations of the true problems, for an increasing number of random draws. Classical results are limited to first-order critical or global solutions. We have developed new conditions ensuring that second-order critical solutions of the SAA problems converge almost surely to second-order critical solutions of the true problem.

A trust-region algorithm with dynamic accuracy

When the number of draws is fixed, statistical inference allows us to evaluate the accuracy of the approximation. We have capitalized on this accuracy estimation to design a trust-region method that allows the use of subsets of the initial set of random draws when possible, leading to reductions of the numerical cost associated to the objective evaluation. The resulting algorithm, that we refer as BTRDA, for basic trust-region with dynamic accuracy, is nonmontone in the sense that it allows increases of the approximated objective value. However, it is proved to converge to a solution of the original SAA problem under reasonable conditions.

Application to discrete choice theory

Our Monte-Carlo developments have then been applied to the mixed logit models estimation problem. Mixed logit problems are currently very popular among practitioners in discrete choice theory, but are numerically difficult to solve since they involve random parameters, which are usually assumed to be continuous, leading to choice probabilities that are multidimensional integrals. We have shown that the mixed logit problem can be seen as a generalization of usual stochastic programs and that the consistency results obtained for SAA solutions can then be easily extended. Moreover, the BTRDA algorithm has been adapted to take the additional difficulties (in particular the simulation bias) into account, and implemented in our software AMLET (Another Mixed Logit Estimation Tool). Numerical experimentations exhibit very favourable results, in comparison to standard nonlinear programming methods and to existing tools for solving mixed logit problems.

Research perspectives

The BTRDA algorithm is currently for unconstrained problems only. A natural research direction is to adapt it to constrained problems, in particular if these constraints involve some randomness. More generally, the BTRDA algorithm could be adapted to larger classes of problems, as soon as the objective function can be approximated by numerically cheaper approximations, whose accuracy can be estimated. It is even possible to adapt the BTRDA methodology to the previously presented primal-dual interior approach as soon as the objectives related to the different scenarios are of similar order, since the relative weights can then be estimated by the scenarios probabilities. However, care must be taken in practice since scenarios with low probabilities often correspond to situations where a nonanticipative solution leads to a poor value of the related objective.

Practitioners in discrete choice theory usually solve the mixed logit estimation problem using quasi-Monte Carlo approximations instead of Monte Carlo ones, in order to construct numerically cheaper problems. However, the accuracy estimation is then a difficult task, so the BTRDA algorithm cannot be directly applied. The potential benefit of using quasi-Monte Carlo suggests however adapting the methodology to such techniques, for instance the Latin hypercube sampling method (McKay, Beckman and Conover [96], Owen [106]), and the randomized Halton sequences (see in particular Wang and Hickernell [140], that develop associated error estimation procedures).

Finally, the variable sample size strategy, while numerically efficient during our tests with AMLET, can very probably be improved, in order to obtain further numerical gains. Moreover, more numerical tests, both on discrete choice models and on other problems, should be useful in assessment and refinement of the proposed method.

Final conclusions

Nonlinear stochastic programming requires the design of methodologies that take account of the properties of the considered problems, as done in this work. Note however that the two proposed algorithms address different difficulties. In the split-variable formulation, the major computational burden is the number of variables that is often very large. On the other hand, the sampling average approximation can require a very large number of random draws to deliver an adequate accuracy, independently of the number of variables. Combining the two approaches could nevertheless result in a method that will address these two major difficulties in stochastic programming.

We therefore consider the present work as a first step in dealing with nonlinear nonconvex stochastic programs, suggesting many possible future research directions we are interested in, rather than a completed exploration of the subject.

Main notations and abbreviations

General		
	$\mathbb{R}$	set of real numbers
	$\overline{\mathbb{R}}$	set of extended real numbers: $\mathbb{R} \cup \{-\infty, +\infty\}$
	$\mathbb{R}_+$	set of real positive numbers
	$\mathbb{N}$	set of natural numbers
	$\emptyset$	empty set
	$e^{[j]}$	j-th coordinate vector
	Π	projection operator
	$\#\mathcal{S}$	cardinality of $\mathcal{S}$
	$\mathcal{N}_X(x)$	normal cone of X at $x \in X$
	$\mathcal{T}_X(x)$	tangent cone of X at $x \in X$
Mathematical programming		
	$\mathcal{E}$	set of equality constraints
	$\mathcal{I}$	set of inequality constraints
	x^*	optimal solution
	$\mathcal{F}$	feasible set
	strict$\{\mathcal{F}\}$	strictly feasible set
Stochastic programming		
	H	number of stages (horizon) in multistage problems
	s	scenario
	S	number of scenarios
Probability theory		
	ω	random event ($\omega \in \Omega$)
	Ω	set of all random events
	A	event (measurable subset of Ω)
	$\mathcal{A}$	collection of subsets of Ω
	$\boldsymbol{\xi}$	random vector (possibly indexed by time, $\boldsymbol{\xi}^t$)
	ξ	realization of $\boldsymbol{\xi}$
	Ξ	support of $\boldsymbol{\xi}$

$P[\cdot]$	probability of events
$E[\boldsymbol{\xi}]$, $\overline{\xi}$	expectation of $\boldsymbol{\xi}$
$\sigma(\cdot)$	standard deviation
a.s.	almost surely
i.i.d.	independent and identically distributed
$\xrightarrow{a.s.}$	almost sure convergence
$\xrightarrow{p}$	convergence in probability
$\Rightarrow$, $\xrightarrow{D}$	convergence in distribution
$N(\mu, \sigma^2)$	normal distribution with mean μ and variance σ^2
$\Phi(x)$	cumulative distribution function of the standard normal random variable

Main mathematical notations

AMLET	Another Mixed Logit Estimation Tool
BTR	basic trust-region algorithm
IIA	independence from irrelevant alternatives
KKT	Karush-Kuhn-Tucker
LICQ	Liner Independence Constraint Qualification
LL	log-likelihood
MC	Monte Carlo
ML	maximum likelihood
SAA	sample average approximation
SLL	simulated log-likelihood
SP	stochastic programming
TSSPR	two-stage stochastic programming with recourse

Main abbreviations

Bibliography

[1] Natalia M. Alexandrov, John E. Dennis Jr, Robert M. Lewis, and Virginia Torczon. A trust region framework for managing the use of approximation models. *Structural Optimization*, 15(1):16–23, 1998.

[2] Greg M. Allenby and Peter E. Rossi. Marketing models of consumer heterogeneity. *Journal of Econometrics*, 89:57–78, 1999.

[3] E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. Sorensen. *LAPACK Users' Guide*. SIAM, Philadelphia, Pennsylvania, USA, third edition, 1999.

[4] Simon P. Anderson, Andre De Palma, and Jacques-Francois Thisse. *Discrete Choice Theory of Product Differentiation*. MIT Press, Cambridge, Massachusetts, USA, 1992.

[5] Robert B. Ash and Catherine A. Doleans-Dade. *Probability and Measure Theory*. Harcourt/Academic Press, second edition, 1999.

[6] Kay W. Axhausen, Andrea Zimmerman, Stefan Schönfelder, Guido Rindsfüser, and Thomas Haupt. Observing the rythms of daily life: A six week travel diary. *Transportation*, 29(2):95–124, 2002.

[7] Fabian Bastin. Nonlinear stochastic programming. Master's thesis, Department of Mathematics, University of Namur, Namur, Belgium, September 2001.

[8] Fabian Bastin, Cinzia Cirillo, and Philippe L. Toint. Numerical experiments with AMLET, a new Monte-Carlo algorithm for estimating mixed logit models. Paper presented at the 10th International Conference on Travel Behaviour Research, 2003.

[9] Fabian Bastin, Cinzia Cirillo, and Philippe L. Toint. A new efficient Monte Carlo algorithm for estimating mixed-logit models. Technical report, Department of Mathematics, University of Namur, Namur, Belgium, Forthcoming.

[10] Fabian Bastin, Cinzia Cirillo, and Philippe L. Toint. Convergence theory for nonconvex stochastic programming with an application to mixed logit. *Mathematical Programming, Series B*, Submitted.

[11] Fabian Bastin, Annick Sartenaer, and Jie Sun. On interior point methods using a suitable decomposition for multistage nonlinear stochastic programming. In Preparation.

[12] Moshe Ben-Akiva and Steven R. Lerman. *Discrete Choice Analysis: Theory and Application to Travel Demand*. The MIT Press, 1985.

[13] Arjan Berkelaar, Dert Cees, Bart Oldenkamp, and Shuzhong Zhang. A primal-dual decomposition interior point approach to two-stage stochastic linear programming. Technical report, Erasmus University Rotterdam, The Netherlands, 1999. Econometric Institute Report EI-9918/A.

[14] Dimitri P. Bertsekas. *Convexity, Duality, and Lagrange Multipliers*. Lecture Notes, Massachusetts Institute of Technology, 2001.

[15] Chandra R. Bhat. Quasi-random maximum simulated likelihood estimation of the mixed multinomial logit model. *Transportation Research*, 35B(7):677–693, August 2001.

[16] Chandra R. Bhat. Simulation estimation of mixed discrete choice models using randomized and scrambled Halton sequences. *Transportation Research B*, 37(3):837–855, 2003.

[17] Chandra R. Bhat and Saul Castelar. A unified mixed logit framework for modelling revealed and stated preferences: formulation and application to congestion pricing analysis in the San Francisco bay area. *Transportation Research B*, 36(3):593–616, 2002.

[18] Chandra R. Bhat and Frank S. Koppelman. Activity-based modeling of travel demand. In Randolph W. Hall, editor, *Handbook of Transportation Science*, pages 35–61, Norwell, USA, 1999. Kluwer Academic Publisher.

[19] Michel Bierlaire. Discrete choice models. In Martine Labbé, Gilbert Laporte, Katalin Tanczos, and Philippe L. Toint, editors, *Operations Research and Decision Aid Methodologies in Traffic and Transportation Management*, volume F. 166 of *NATO ASI Series*, pages 203–227, Berlin, Germany, 1998. Springer-Verlag.

[20] Michel Bierlaire, T. Lotan, and Philippe L. Toint. On the overspecification of multinomial and nested logit models due to alternative specific constants. *Transportation Science*, 31(4):363–371, 1997.

[21] John R. Birge and François Louveaux. *Introduction to Stochastic Programming*. Springer-Verlag, 1997.

[22] Alexander Borovkov. *Statistique mathématique*. Mir, 1987.

[23] David Brownstone, David S. Bunch, and Kenneth Train. Joint mixed logit models of stated and revealed preferences for alternative-fuel vehicles. *Transportation Research B*, 34(5):315–338, 2000.

[24] Richard H. Byrd, Humaid Fayez Khalfan, and Robert B. Schnabel. Analysis of a symmetric rank-one trust region method. *SIAM Journal on Optimization*, 6(4):1025–1039, 1996.

[25] Abraham Charnes and William W. Cooper. Chance-constrained programming. *Management Science*, 5:73–79, 1959.

[26] Cinzia Cirillo and Kay W. Axhausen. Mode choice of complex tour. In *Proceedings of the European Transport Conference*, Cambridge, UK, 2002.

[27] Cinzia Cirillo and Philippe L. Toint. An activity based approach to the Belgian national travel survey. Technical Report 2001/07, Transportation Research Group, Department of Mathematics, University of Namur, 2001.

[28] Carlos A. Coello Coello, David A. Van Veldhuizen, and Gary B. Lamont. *Evolutionary Algorithms for Solving Multi-Objective Problems*. Kluwer Academic Publishers, Boston, USA, 2002.

[29] Benoît Colson. *Trust-Region Algorithms for Derivative-Free Optimization and Nonlinear Bilevel Programming*. PhD thesis, University of Namur, Namur, Belgium, July 2003.

[30] Andrew R. Conn, Nicholas I. M. Gould, Dominique Orban, and Philippe L. Toint. A Primal-Dual Trust-Region Algorithm for Non-convex Nonlinear Programming. *Mathematical Programming*, 87(2):215–249, 2000.

[31] Andrew R. Conn, Nicholas I. M. Gould, and Philippe L. Toint. Convergence of quasi-Newton matrices generated by the symmetric rank one update. *Mathematical Programming*, 50(2):177–196, 1991.

[32] Andrew R. Conn, Nicholas I. M. Gould, and Philippe L. Toint. LANCELOT*: a Fortran package for large-scale nonlinear optimization (Release A)*. Springer-Verlag, Heidelberg, Berlin, New-York, 1992.

[33] Andrew R. Conn, Nicholas I. M. Gould, and Philippe L. Toint. Numerical experiments with the LANCELOT package (Release A) for large-scale nonlinear optimization. *Mathematical Programming, Series A*, 73(1):73–110, 1996.

[34] Andrew R. Conn, Nicholas I. M. Gould, and Philippe L. Toint. A primal-dual algorithm for minimizing a non-convex function subject to bound and linear equality constraints. In DG. Di Pillo and F. Giannessi, editors, *Nonlinear Optimization and Applications 2*. Kluwer Academic Publishers, 1999.

[35] Andrew R. Conn, Nicholas I. M. Gould, and Philippe L. Toint. *Trust-Region Methods*. SIAM, Philadelphia, USA, 2000.

[36] David R. Cox and David V. Hinkley. *Theoretical Statistics*. Chapman & Hall, London, England, 1974.

[37] Carlos F. Daganzo. *Multinomial Probit: The theory and its application to demand forecasting*. Academic Press, New York, USA, 1979.

[38] Andrew Daly. Estimating "tree" logit models. *Transportation Research B*, 21(4):251–268, 1987.

[39] James Davidson. *Stochastic Limit Theory*. Oxford University Press, Oxford, England, 1994.

[40] István Deák. Multidimensional integration and stochastic programming. In Y. Ermoliev and R. J.-B. Wets, editors, *Numerical Techniques for Stochastic Optimization*, pages 187–200. Springer Verlag, 1988.

[41] M. A. H. Dempster, editor. *Stochastic Programming*. Academic Press, London, England, 1980.

[42] John E. Dennis Jr and Robert B. Schnabel. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. Prentice-Hall, Englewood Cliffs, New Jersey, USA, 1983.

[43] Darinka Dentcheva, Ralf Gollmer, Andris Möller, Werner Römisch, and Rüdiger Schultz. Solving the unit commitment problem in power generation by primal and dual methods. In M. Brøns, M. P. Bendsøe, and M. P. Sørensen, editors, *Progress in Industrial Mathematics at ECMI 96*, pages 332–339. Teubner, Stuttgart, Germany, 1997.

[44] Thomas A. Domencich and Daniel L. McFadden. *Urban Travel Demand: A Behavioral Analysis*. North-Holland, Amsterdam, The Netherlands, 1975. Reprinted 1996.

[45] Jurgen A. Doornik. *Ox: An Object-Oriented Matrix Language*. Timberlake Consultants Press, London, England, 4th edition, 2001.

[46] Iain S. Duff and John K. Reid. The multifrontal solution of indefinite sparse symmetric linear equations. *ACM Transactions on Mathematical Software*, 9(3):302–325, 1993.

[47] Electric Power Research Institute. Methodology for predicting the demand for new electricity-using goods. Final Report EA-593, Project 488-1, Electric Power Research Institute, Palo Alto, California, USA, 1977.

[48] Yuri Ermoliev and Roger J-B Wets. Stochastic programming, an introduction. In Y. Ermoliev and R. J.-B. Wets, editors, *Numerical Techniques for Stochastic Optimization*, pages 1–32, Berlin, 1988. Springer-Verlag.

[49] Merran Evans, Nicholas Hastings, and Brian Peacock. *Statistical distributions*. John Wiley & Sons, 2nd edition, 1993.

[50] Anthony V. Fiacco. *Introduction to Sensitivity and Stability Analysis in Nonlinear Programming*. Academic, New York, USA, 1983.

[51] Anthony V. Fiacco and G. P. McCormick. *Nonlinear Programming: Sequential Unconstrained Minimization Techniques*. John Wiley & Sons, New York, NY, USA, 1968. Reprinted by SIAM Publications in 1990.

[52] George S. Fishman. *Monte Carlo: Concepts, Algorithms and Applications*. Springer Verlag, New York, USA, 1996.

[53] R. Fletcher. *Practical Methods of Optimization*. John Wiley & Sons, New York, USA, second edition, 1987.

[54] A. Forsgren, Philip E. Gill, and Margaret H. Wright. Interior methods for nonlinear optimization. *SIAM Review*, 44(4), 2002.

[55] Karl Frauendorfer. *Stochastic Two-Stage Programming*, volume 392 of *Lecture Notes in Economics and Mathematical Systems*. Springer-Verlag, 1992.

[56] Robert Fuller and Christer Carlsson. Fuzzy multiple criteria decision making; recent developments. *Fuzzy Sets and Systems*, 78(2):139–153, 1996.

[57] Rodrigo A. Garrido. Estimation performance of low discrepancy sequences in stated preferences. Paper presented at the 10th International Conference on Travel Behaviour Research, 2003.

[58] Horand I. Gassmann. Optimal harvest of a forest in the presence of uncertainty. *Canadian Journal of Forest Research*, 19:1267–1274, 1990.

[59] David M. Gay, Michael L. Overton, and Margaret H. Wright. A primal-dual interior point method for nonconvex nonlinear programming. In Y. Yuan, editor, *Advances in Nonlinear Programming*, pages 31–56. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1998.

[60] Philip E. Gill, Walter Murray, Michael Saunders, G. W. Stewart, and Margaret H. Wright. Properties of a representation of a basis for the null space. *Mathematical Programming*, 33(2):172–186, 1985.

[61] Gene H. Golub and Charles F. Van Loan. *Matrix Computations*. The Johns Hopkins University Press, third edition, 1996.

[62] Jacek Gondzio and Andreas Grothey. Reoptimization with the primal-dual interior point method. *SIAM Journal on Optimization*, 13(3):842–864, 2003.

[63] Jacek Gondzio and Robert Sarkissian. Parallel interior point solver for structured quadratic programs: Application to financial planning problems. Technical Report MS-03-001, School of Mathematics, The University of Edinburgh, United Kingdom, 2003.

[64] María D. González-Lima, Richard A. Tapia, and Florian A. Potra. On effectively computing the analytic center of the solution set by primal-dual interior-point methods. *SIAM Journal on Optimization*, 8(1):1–25, February 1998.

[65] Nicholas I. M. Gould, Dominique Orban, Annick Sartenaer, and Philippe L. Toint. Superlinear convergence of primal-dual interior point algorithms for nonlinear programming. *SIAM Journal on Optimization*, 11(4):974–1002, 2001.

[66] Nick Gould. On practical conditions for the existence and uniqueness of solutions to the general equality quadratic programming problem. *Mathematical Programming*, pages 90–95, 1985.

[67] Nicole Gröwe, Werner Römisch, and Rüdiger Schultz. A simple recourse model for power dispatch under uncertain demand. *Annals of Operations Research*, 59:135–164, 1995.

[68] Gül Gürkan, A. Yonca Özge, and Stephen M. Robinson. Sample-path solution of stochastic variational inequalities. *Mathematical Programming*, 84(2):313–333, 1999.

[69] Gül Gürkan, A. Yonca Özge, and Stephen M. Robinson. Solving stochastic optimization problems with stochastic constraints: an application in network design. In D.T. Sturrock P. A. Farrington, H. B. Nembhard and G. W. Evans, editors, *Proceedings of the 1999 Winter Simulation Conference*, pages 471–478, USA, 1999.

[70] Vassilis A. Hajivassiliou and Daniel L. McFadden. The method of simulated scores for the estimation of LDV models. *Econometrica*, 66(4):863–896, 1998.

[71] John H. Halton. On the efficiency of certain quasi-random sequences of points in evaluating multi-dimensional integrals. *Numerische Mathematik*, 2:84–90, 1960.

[72] David A. Hensher and William H. Greene. The mixed logit model: The state of practice. *Transportation*, 30(2):133–176, 2003.

[73] David A. Hensher and Charles Sullivan. Willingness to pay for road curviness and road type for long-distance travel in New Zealand. *Transportation Research D*, 8(2):139–155, 2003.

[74] Stéphane Hess and John Polak. Mixed logit estimation of parking type choice. *Submitted for presentation at the 83rd Transportation Research Board Annual Meeting*, 2004.

[75] Stéphane Hess, John Polak, and Andrew Daly. On the performance of shuffled Halton sequences in the estimation of discrete choice models. In *Proceedings of European Transport Conference*, Strasbourg, France, 2003. PTRC.

[76] Stéphane Hess, Kenneth Train, and John Polak. On the use of randomly shifted uniform vectors in the estimation of a mixed logit model for vehicle choice. *Transportation Research B*, Submitted.

[77] Suwarna Hulsurkar, Mahendra P. Biswal, and Surabhi B. Sinha. Fuzzy programming approach to multi-objective stochastic linear programming problems. *Fuzzy Sets and Systems*, 88(2):173–181, 1997.

[78] Bruce Jacob and Trevor Mudge. Notes on calculating computer performance. Technical Report CSE-TR-231-95, Advanced Computer Architecture Lab, EECS, University of Michigan, Michigan, USA, 1995.

[79] Miettinen Kaisa. *Nonlinear Multiobjective Optimization*, volume 12 of *International Series in Operations Research & Management Science*. Kluwer Academic Publishers, Boston, USA, 1999.

[80] Peter Kall and Stein W. Wallace. *Stochastic Programming*. John Wiley & Sons, 1994.

[81] William Karush. Minima of functions of several variables with inequalities as side conditions. Master's thesis, Department of Mathematics, University of Chicago, Illinois, USA, 1939.

[82] Tinne H. Kjeldsen. A contextualized historical analysis of the Kuhn-Tucker theorem in nonlinear programming: the impact of World War II. *Historia Mathematica*, 27(4):331–361, 2000.

[83] Pieter Klaassen, J. F. Shapiro, and D. E. Spitz. Sequential decision models for selecting currency options. Technical report, International Financial Services Research Center, Massachusetts Institute of Technology, Cambridge, MA, July 1990. IFSRC Report No. 133-90.

[84] Harold W. Kuhn and Albert W. Tucker. Nonlinear programming. In J. Neyman, editor, *Proceeding of the Second Berkeley Symposium on Mathematical Statistics and Probability*, pages 481–492, Berkeley, California, USA, 1951. University of California Press.

[85] Pierre L'Ecuyer. Efficient and portable combined random number generators. *Communications of the ACM*, 31(6):742–774, 1988.

[86] Xinwei Liu and Jie Sun. A new decomposition technique in solving multistage stochastic linear programs by infeasible interior point methods. *Journal of Global Optimization*, to appear.

[87] Xinwei Liu and Gongyun Zhao. A decomposition method based on SQP for a class of multistage nonlinear stochastic programs. Technical report, National University of Singapore, Department of Mathematics, 1999. revised 2000.

[88] R. Duncan Luce. *Individual Choice Behavior: A Theoretical analysis*. Wiley, New York, USA, 1959.

[89] Irvin J. Lustig, Roy E. Marsten, and David F. Shanno. On implementing Mehrotra's predictor-corrector interior-point method for linear programming. *SIAM Journal on Optimization*, 2(3):435–449, 1992.

[90] Charles F. Manski. The structure of random utility models. *Theory and Decision*, 8:229–254, 1977.

[91] Hatem Masri. *Stochastic Programming with Partial Information on Probability Distribution*. PhD thesis, Institut Supérieur de Gestion, Université de Tunis, Tunis, Tunisia, 2003.

[92] Daniel L. McFadden. Modelling the choice of residential location. In A. Karlquist et al., editor, *Spatial Interaction Theory and Residential Location*, pages 75–96. North Holland, Amsterdam, The Netherlands, 1978.

[93] Daniel L. McFadden. Econometric models of probabilistic choice. In C. F. Manski and D. L. McFadden, editors, *Structural Analysis of Discrete Data with Econometric Applications*, pages 198–272. MIT Press, Cambridge, Massachusetts, USA, 1981.

[94] Daniel L. McFadden. *Statistical Tools for Economists*. Lecture Notes, Department of Economics, University of California, Berkeley, USA, 2000.

[95] Daniel L. McFadden and Kenneth Train. Consumers' evaluation of new products: learning from self and others. *Journal of Political Economy*, 104(4):683–703, 1996.

[96] M. D. McKay, R. J. Beckman, and W. J. Conover. 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, 1979.

[97] Sanjay Merhotra. On the implementation of a primal-dual interior-point method. *SIAM Journal on Optimization*, 2:575–601, 1992.

[98] Michel Minoux. *Programmation Mathématique, Théorie et algorithmes*, volume 1 of *Collection Technique et Scientifique des Télécommunications*. Dunod, Paris, France, 1983.

[99] Chander Mohan and Hai Thanh NguyenMohan. An interactive satisficing method for solving multiobjective mixed fuzzy-stochastic programming problems. *Fuzzy Sets and Systems*, 117(1):61–79, 2001.

[100] Claude Montmarquette, Kathy Cannings, and Sophie Mahseredjian. How do young people choose college majors? *Economics of Education Review*, 21(6):543–556, 2002.

[101] Jorge J. Moré and David J. Thuente. Line search algorithms with guaranteed sufficient decrease. *ACM Transactions on Mathematical Software*, 20(3):286–307, 1994.

[102] William J. Morokoff and Russel R. Caflish. Quasi-Monte Carlo integration. *Journal of Computational Physics*, 122(2):218–230, 1995.

[103] Soren S. Nielsen. GAMS Summer School Notes: Mathematical Modeling And Optimization with Applications in Finance, 1999.

[104] Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer, New York, USA, 1999.

[105] Juan de Dios Ortúzar and Luis G. Willumsen. *Modelling Transport*. John Wiley & Sons, 3rd edition, 2001.

[106] Art B. Owen. Monte Carlo variance of scrambled net quadrature. *SIAM Journal of Numerical Analysis*, 34(5):1884–1910, 1997.

[107] Stephen K. Park and Keith W. Miller. Random number generators: good ones are hard to find. *Communications of the ACM*, 31(10):1192–1201, 1988.

[108] Andras Prékopa. Programming under probabilistic constraints with a random technology matrix. *Math. Operationsforsch. Statist,Ser. Optim.*, 5:109–116, 1974.

[109] Andras Prékopa. Logarithmic concave measures and related topics. In M. A. H. Dempster, editor, *Stochastic Programming*, pages 63–81. Academic Press, London, England, 1980.

[110] Andras Prékopa. Numerical solution of probabilistic constrained programming problems. In Y. Ermoliev and R. J-B. Wets, editors, *Numerical Techniques for Stochastic Optimization*, pages 123–139. Springer-Verlag, Berlin, Germany, 1988.

[111] Andras Prékopa. Sharp bounds on probabilities using linear programming. *Operations Research*, 38:227–239, 1990.

[112] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes in C*. Cambridge University Press, Cambridge, USA, second edition, 1992.

[113] Stephen M. Robinson. Local epi-continuity and local optimization. *Mathematical Programming*, 37(2):208–222, 1987.

[114] Stephen M. Robinson. Analysis of sample-path optimization. *Mathematics of Operations Research*, 21(3):513–528, 1996.

[115] R. Tyrrell Rockafellar and Roger J.-B. Wets. Scenarios and policy aggregation in optimization under uncertainty. *Mathematics of Operations Research*, 16(1):119–147, 1991.

[116] R. Tyrrell Rockafellar and Roger J.-B. Wets. *Variational Analysis*. Springer Verlag, Heidelberg, Berlin, New York, 1998.

[117] Werner Römisch and Rüdiger Schultz. Decomposition of a Multi-Stage Stochastic Program for Power Dispatch. *ZAMM - Zeitschrift für Angewandte Mathematik und Mechanik*, 76, Suppl. 3:29–32, 1996.

[118] Charles H. Rosa and Andrzej Ruszczyński. On augmented Lagrangian decomposition methods for multistage stochastic programs. *Annals of Operations Research*, (64):289–309, 1996.

[119] Charles H. Rosa and David N. Yates. Addressing the issue of uncertainty within the Egyptian agricultural sector. Technical Report WP-94-97, International Institute for Applied Systems Analysis, Laxenburg, Austria, 1994.

[120] Reuven Y. Rubinstein and Alexander Shapiro. *Discrete Event Systems*. John Wiley & Sons, Chichester, England, 1993.

[121] Andrzej Ruszczyński. Parallel decomposition of multistage stochastic programming problems. *Mathematical programming*, 58:201–228, 1993.

[122] Masatoshi Sakawa. *Fuzzy Sets and Interactive Multiobjective Optimization*. Plenion Press, New York, 1993.

[123] Zsolt Sándor and Kenneth Train. Quasi-random simulation of discrete choice models. *Transportation Research B*, Forthcoming.

[124] Ronald Schoenberg. Optimization with the quasi-Newton method. Aptech Systems, Inc., Maple Valley, WA, USA, 2001.

[125] Alexander Shapiro. Probabilistic constrained optimization: Methodology and applications. In S. Uryasev, editor, *Statistical inference of stochastic optimization problems*, pages 282–304. Kluwer Academic Publishers, 2000.

[126] Alexander Shapiro. Stochastic programming by Monte Carlo simulation methods. *SPEPS*, 2000.

[127] Alexander Shapiro. Monte Carlo sampling methods. In A. Shapiro and A. Ruszczyński, editors, *Stochastic Programming*, volume 10 of *Handbooks in Operations Research and Management Science*, pages 353–425. Elsevier, 2003.

[128] Alexander Shapiro and Andrzej Ruszczyński, editors. *Stochastic Programming*, volume 10 of *Handbooks in Operations Research and Management Science*. Elsevier, 2003.

[129] Yosef Sheffi. *Urban Transportation Networks*. Prentice-Hall, Englewood Cliffs, New Jersey, USA, 1985.

[130] Marc C. Steinbach. Hierarchical sparsity in multistage convex stochastic programs. In S. P. Uryasev and P. M. Pardalos, editors, *Stochastic Optimization: Algorithms and Applications*, pages 385–410. Kluwer Academic Publishers, 2001.

[131] Wilson A. Sutherland. *Introduction to Metric and Topological Spaces*. Oxford University Press, Oxford, England, 1975.

[132] Joffre Swait. *Probabilistic choice set formulation in transportation demand models.* PhD thesis, Department of Civil and Environmental Engineering, Massachusetts Institute of Technology, Cambridge, Massachussetts, USA, 1984.

[133] TRACE, Costs of private road travel and their effects on demand, including short and long term elasticities. Final report to the European Commission, HCG, Den Haag, The Netherlands, 1999.

[134] Kenneth Train. Halton sequences for mixed logit. Working paper No. E00-278, Department of Economics, University of California, Berkeley, 1999.

[135] Kenneth Train. *Discrete Choice Methods with Simulation*. Cambridge University Press, New York, USA, 2003.

[136] M. Ulbrich, S. Ulbrich, and Luís N. Vicente. A globally convergent primal-dual interior-point filter method for nonlinear programming. Technical Report TR00-12, Department of Computational and Applied Mathematics, Rice University, Houston, Texas, USA, 2000. Revised February 2002.

[137] Robert J. Vanderbei and David F. Shanno. An interior point algorithm for nonconvex nonlinear programming. Technical Report SOR-97-21, Statistics and Operations Research, Princeton University, Princeton, New Jersey, USA, 1997.

[138] Luís N. Vicente. *Trust-Region Interior-Point algorithms for a Class of Nonlinear Programming Problems*. PhD thesis, Department of Computational and Applied Mathematics, Rice University, Houston, Texas, USA, March 1996.

[139] Andreas Wächter. *An Interior Point Algorithm for Large-Scale Nonlinear Optimization with Applications in Process Engineering*. PhD thesis, Carnegie Mellon University, January 2002.

[140] Xiaoqun Wang and Fred J. Hickernell. Randomized Halton sequences. *Math. Comput. Modelling*, 32:887–899, 2000.

[141] R. Clint Whaley, Antoine Petitet, and Jack J. Dongarra. Automated empirical optimization of software and the ATLAS project. *Parallel Computing*, 27(1–2):3–35, 2001.

[142] Stephen J. Wright. *Primal-Dual Interior-Points Methods*. SIAM Publications, Philadelphia, Pa, USA, 1997.

[143] Jianzhong Zhang and Chengxian Xu. A projected indefinite dogleg-path method for equality constrained optimization. *BIT*, 39(3):555–578, 1999.

[144] Yin Zhang. On the convergence of a class of infeasible interior-point methods for the horizontal linear complementarity problem. *SIAM Journal on Optimization*, 4(1):208–227, 1994.

Index