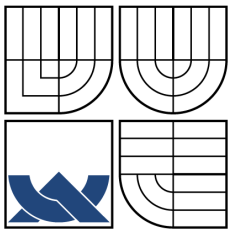
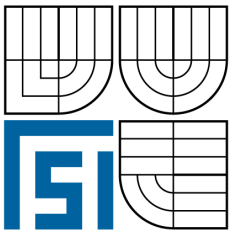


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA STROJNÍHO INŽENÝRSTVÍ
ÚSTAV MATEMATIKY
FACULTY OF MECHANICAL ENGINEERING
INSTITUTE OF MATHEMATICS

# MODELY STOCHASTICKÉHO PROGRAMOVÁNÍ A JEJICH APLIKACE

STOCHASTIC PROGRAMMING MODELS WITH APPLICATIONS

DIPLOMOVÁ PRÁCE
DIPLOMA THESIS

AUTOR PRÁCE
AUTHOR
JAN NOVOTNÝ

VEDOUCÍ PRÁCE
SUPERVISOR
RNDr. PAVEL POPELA, Ph.D.

BRNO 2008

**Abstrakt**
Diplomová práce se zabývá stochastickým programováním a jeho aplikací na problém mísení kameniva z oblasti stavebního inženýrství. Teoretická část práce je věnována odvození základních přístupů stochastického programování, tj. optimalizace se zohledněním náhodných vlivů v modelech. V aplikované části je prezentována tvorba vhodných optimalizačních modelů pro mísení kameniva, jejich implementace a výsledky. Práce zahrnuje původní aplikační výsledky docílené při řešení projektu GA ČR reg. čís. 103/08/1658 Pokročilá optimalizace návrhu složených betonových konstrukcí a teoretické výsledky projektu MŠMT České republiky čís. 1M06047 Centrum pro jakost a spolehlivost výroby.

**Abstract**
The thesis deals with stochastic programming and its application to aggregate blending, an optimization problem within the area of civil engineering. The theoretical part is devoted to the derivation of basic principles of stochastic programming (optimization under uncertainty). The applied part presents a development of suitable mathematical models for aggregate blending, their implementation and results. The thesis contains original results achieved in solution of the project GA ČR reg. n. 103/08/1658 Advanced optimum design of composed concrete structures and it contains theoretical results of the project from MSMT of the Czech Republic no. 1M06047 Centre for Quality and Reliability of Production.

**Klíčová slova**
stochastické programování, dvojstupňové stochastické programování, mísení kameniva

**Keywords**
stochastic programming, two-stage stochastic programming, aggregate blending

NOVOTNÝ, J. *Modely stochastického programování a jejich aplikace.* Brno: Vysoké učení technické v Brně, Fakulta strojnÍho inženýrstvÍ, 2008. 52 s. Vedoucí diplomové práce RNDr. Pavel Popela, Ph.D.

Prohlašuji, že jsem diplomovou práci *Modely stochastického programování a jejich aplikace* vypracoval samostatně pod vedením RNDr. Pavla Popely, Ph.D., s použitím materiálů uvedených v seznamu literatury.

Jan Novotný

Rád bych poděkoval RNDr. Pavlu Popelovi, Ph.D. za vedení mé diplomové práce a za mnoho laskavých a přínosných diskuzí nejen o stochastickém programování. Rád bych též poděkoval prof. Lino Santovi (Department of Statistics and Operations Research, University of Malta) za jeho přátelskou podporu a za otevřený přístup k mnohým matematickým záležitostem.

I would like to thank to RNDr. Pavel Popela, Ph.D. for supervising my thesis and for many kind and helpful discussions on stochastic programming and other topics. I would also like to give my thanks to prof. Lino Sant (Department of Statistics and Operations Research, University of Malta) for his friendly support and his open attitude to many mathematical matters.

Jan Novotný

# Contents

# Chapter 1

# Introduction

Stochastic programming provides mathematical models for decision problems under uncertainty or equivalently for problems involving randomness. Historically, stochastic programming arose from joining the concepts of linear and nonlinear programming with probability and measure theory. Nowadays its models are being used in various areas ranging from economics (e.g. portfolio management) to logistics and engineering (e.g. electric or gas transport network optimization).

This thesis deals with stochastic programming and its application to a problem in civil engineering, namely the optimization of aggregate blending for the production of concrete mixture. In Chapter 2 we start by analyzing the aggregate blending problem and we develop suitable linear programming models. Then we discuss briefly the reasons, why randomness should be considered in the models, and this leads naturally to an exposition of the basic principles of stochastic programming, which is presented in Chapter 3.

The presentation of the theory is original to certain extent, since we have adopted a slightly different approach compared to the standard textbooks (see e.g. [4],[6],[7]). Instead of introducing the randomness to a mathematical model after it has been built, we start our model development including the probability space from the beginning. We also give a correct derivation of the so-called extensive form of stochastic program, which is usually used for implementation. A short discussion is devoted to the probability issues in stochastic programming with respect to the notation used.

We return to the real application in Chapter 4, where we design stochastic programming models for the aggregate blending problem and we discuss their properties.

The overview of our achievements is presented in Chapter 5. We discuss some aspects of our implementation of the models and we also present numerical and graphical results.

# Chapter 2

# Aggregate blending

## 2.1 Introduction, motivation

The problem of optimizing an aggregate mix arises in civil engineering applications like concrete or asphalt mixture design. Various physical properties of the mixture can be of interest, depending on particular application. There is a variety of results and texts on this topic within the area of civil engineering, see for example Svoboda [15].

One of the possible approaches being used consists in formulating the optimal properties of the mixture in terms of its grading curve, which represents the distribution of the size of the particles in the mixture. The justification lies in the fact, that the proportions of particles of different sizes affect the mechanical properties of the aggregate mix in its hardened state and also its workability during the preparation process. The formulation of optimality using the grading curve is suitable for application of mathematical programming.

The aggregate blending problem has been dealt by several authors in various settings, for a brief survey see Bibliography [11], [12], [13], [14], [16], [17] . Generally, models based on linear programming or genetic algorithms are being used.

## 2.2 Grading curves

The desired composition of the mixture is defined in terms of its grading curve, which represents the distribution of the size of particles in the mixture. The idea of the grading curve is very similar to that of the distribution function of a random variable with bounded range in $\mathbb{R}$. We illustrate this with Figure (2.1).

The size of the particles in the mixture is obviously bounded from below by 0 and the upper bound on it also exists (the number of particles in the mixture is finite). Let's denote the upper bound by $b$. The grading curve $f : [0, b] \to [0, 1]$ is then defined by

$$f(x) = \frac{\text{mass of particles with size less than } x}{\text{total mass of the mixture}}.$$

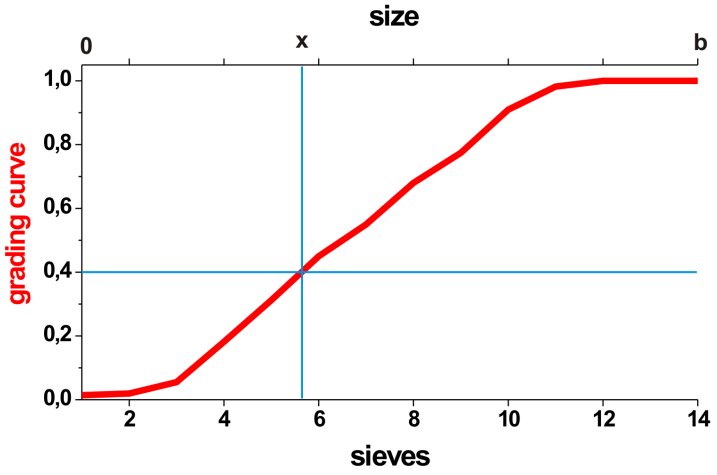



Figure 2.1: The grading curve -
- 40% of the particles are smaller than $x$ and fall through the sieves 6 - 14.

By the size of the particle we mean its characteristic size which may be defined as the diameter of the smallest ball into which the particle fits. Some other characteristics including the particle's volume may be used as well. These definitions are presented just to give an idea about the problem to the reader. Practically they are not important. In practice, the grading curve is estimated by measurement. There is an n-tuple of sieves with increasing size of mesh ranging from $a \geq 0$ to $b$. The single sizes are specified by technical standards (ISO), the worldwide standard of today is 14 sieves with sizes 0.063, 0.125, 0.25, 0.5, 1, 2, 4, 8, 11, 16, 22, 32, 45 and 63 mm.

Let's denote the mesh size of the i-th sieve by $a_i$. The practical grading curve $g$ is then a piecewise linear continuous function determined by the values

$$g(a_i) = \frac{\text{mass of particles that fall through the } i\text{-th sieve}}{\text{total mass of the mixture}}$$

and represents a piecewise linear approximation for the theoretical grading curve $f$ on the interval $[a_1, a_n]$.

If we can guarantee that the particle falls through the i-th sieve if and only if its size is less than the mesh size $a_i$, we obtain the equality $f(a_i) = g(a_i)$ of the theoretical and practical grading curves at the points $a_i$. In practice, this is satisfied only approximately, giving one of the reasons for the development of a stochastic programming model.

In the following, when talking about the grading curve, we will always mean the practical grading curve. Moreover, we will identify the practical grading curve with the vector $\mathbf{g} = (g(a_i))_{i=1}^n$ of its values. We also present a mathematical justification for this (intuitively clear) approach.

Given an interval $[a_1, a_n] \subset \mathbb{R}$ and its partition $p \in \mathbb{R}^n$, $p = (a_1, a_2, \ldots, a_n)$ with $a_i < a_{i+1}$ for $i \in \{1, 2, \ldots, n-1\}$, the set of all continuous piecewise linear functions on $[a_1, a_n]$ with usual addition and multiplication by a scalar is a vector space over $\mathbb{R}$ of dimension $n$, therefore isomorphic to the vector space $\mathbb{R}^n$ over $\mathbb{R}$.

## 2.3 Mixture of ingredients

The ingredients available for the production of the mixture are aggregate mixtures themselves and are characterized by their own grading curves. Suppose we have $m$ ingredients to choose from. To produce one unit of the mixture, we need to decide the mixing ratios, that is, we need to fix an m-tuple $\mathbf{x} = (x_j)_{j=1}^m \in \mathbb{R}^m$ of real numbers where $x_j \geq 0$ for $j \in \{1, \ldots, m\}$ and $\sum_{j=1}^m x_j = 1$. From this it follows that $x_j \leq 1$ for all $j$.

Let the ingredients have theoretical grading curves $g_j : [0, b] \to [0, 1]$. Then $g_j(t)$ denote the mass of particles of size less than $t$ in one mass unit of the i-th ingredient. In one unit of the mixture, the mass of particles of size less than $t$ coming from the i-th ingredient is therefore $x_j g_j(t)$, and the total mass of particles of size less than $t$ is given by $\sum_{j=1}^m x_j g_j(t)$.

We have shown that the grading curve $g$ of the mixture is a convex combination of the grading curves $g_j$ of the ingredients

$$g = \sum_{j=1}^{m} x_j g_j,$$

with mixing ratios $x_j$ as the coefficients of the convex combination. We present Figure (2.2) for illustration.

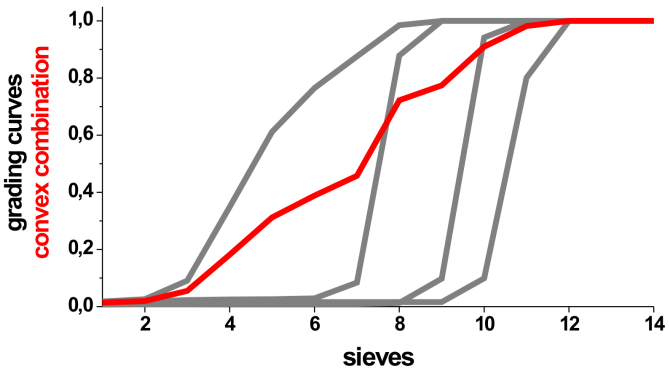



Figure 2.2: Convex combination of grading curves.

When dealing with practical grading curves, we construct an $n$ by $m$ matrix $\mathbf{A}$ with columns as grading curves of the ingredients: $A_{i,j} = g_j(a_i)$. The grading curve $\mathbf{g}$ of the mixture is then given by $\mathbf{Ax} = \mathbf{g}$.

## 2.4 The goal and the objective

Having $m$ different ingredients at disposal, the goal is to produce a mixture of required properties. That is, given grading curves $(\mathbf{g}_j)_{j=1}^m$, the objective is to find the ratios $(x_j)_{j=1}^m$ such that the grading curve $\mathbf{Ax} = \sum_{j=1}^m x_j \mathbf{g}_j$ of the mixture meets the requested grading curve $\mathbf{g}$.

Two different situations may arise. First, the required grading curve $\mathbf{g}$ belongs to the convex hull of the grading curves $\mathbf{g}_j$. Then there are (possibly infinitely many) choices of

$\mathbf{x}$ such that $\mathbf{Ax} = \mathbf{g}$. If the cost of one unit of the j-th ingredient is $c_j$, the optimization model is

$$\begin{array}{lrcl} \min & \mathbf{c}^{\mathrm{T}}\mathbf{x} & & \\ \text{s.t.} & \mathbf{Ax} & = & \mathbf{g}, \\ & \mathbf{1}^{\mathrm{T}}\mathbf{x} & = & 1, \\ & \mathbf{x} & \geq & \mathbf{0}, \end{array}$$

where $\mathbf{1}^{\mathrm{T}}$ denotes the vector $(1, 1, \ldots, 1) \in \mathbb{R}^m$ and the equation $\mathbf{1}^{\mathrm{T}}\mathbf{x} = 1$ stands for $\sum_{j=1}^{m} x_j = 1$. The vector $\mathbf{c} = (c_j)_{j=1}^{m}$ is the vector of cost coefficients. The model seeks for the mixture with minimal cost.

Second, the required grading curve $\mathbf{g}$ does not belong to the convex hull of the grading curves $\mathbf{g}_j$. Since the vectors $\mathbf{g}_j$ are nonnegative for $j \in \{1, \ldots, m\}$ and also $\mathbf{g} \geq \mathbf{0}$, we have that $\mathbf{g}$ belongs to the convex hull of $\mathbf{g}_j$ if and only if $\mathbf{g}$ belongs to the linear span of $\mathbf{g}_j$. This means that we cannot achieve $\mathbf{Ax} = \mathbf{g}$ by any choice of $\mathbf{x} \in \mathbb{R}^n$. But still we would like the values of $\mathbf{Ax}$ to be close to $\mathbf{g}$. Hence it is natural to state the optimization goal as minimizing the distance between $\mathbf{Ax}$ and $\mathbf{g}$. To measure the distance we use the standard metrics given by norms in $\mathbb{R}^n$:

$$\text{the 1-norm:} \qquad ||x||_1 = \sum_{i=1}^{n} |x_i|,$$

$$\text{the } \infty\text{-norm:} \qquad ||x||_\infty = \max_{i=1,\ldots,n} |x_i|.$$

## 2.5 Bounds on the objective

Apart from the primary goal, which is to stay with $\mathbf{Ax}$ as close to $\mathbf{g}$ as possible, there are two more bounds on $\mathbf{Ax}$ imposed. They are the lower bound $\mathbf{l}$ and the upper bound $\mathbf{u}$, within which $\mathbf{Ax}$ is considered to be acceptable.

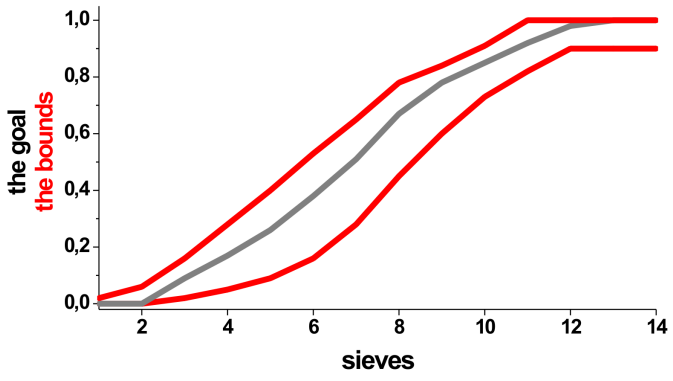



Figure 2.3: Bounds on the grading curve.

This yields two additional constraints for our models, namely

$$\mathbf{Ax} \geq \mathbf{l} \quad \text{and} \quad \mathbf{Ax} \leq \mathbf{u}.$$

In practice, the lower and the upper bound are specified on selected sieves only (and possibly on different sieves for lower and upper bound). We therefore formulate the constraints as

$$\mathbf{A}_l\mathbf{x} \geq \mathbf{l} \quad \text{and} \quad \mathbf{A}_u\mathbf{x} \leq \mathbf{u}.$$

where the matrices $\mathbf{A}_l$ ($\mathbf{A}_u$) are obtained from the matrix $\mathbf{A}$ by skipping the rows of $\mathbf{A}$ which correspond to the sieves without the lower (upper) bound imposed.

When introducing the constraints to our programs (see next section), we do it in the standard form, with a nonnegative surplus vector variable $\mathbf{y}_l$ added to the constraint for the lower bound and a nonnegative slack vector variable $\mathbf{y}_u$ added to the constraint for the upper bound:

$$\begin{array}{rcrcll} \mathbf{A}_l\mathbf{x} & - & \mathbf{y}_l & = & \mathbf{l}, & \mathbf{y}_l \geq \mathbf{0}, \\ \mathbf{A}_u\mathbf{x} & + & \mathbf{y}_u & = & \mathbf{u}, & \mathbf{y}_u \geq \mathbf{0}. \end{array}$$

## 2.6 Model with 1-norm

Minimizing the 1-norm of $\mathbf{Ax} - \mathbf{g}$ leads to the following program:

$$\begin{array}{lrcl} \min & \displaystyle\sum_{i=1}^{n} \left| \sum_{j=1}^{m} a_{i,j}x_j - g_i \right| & & \\ \text{s.t.} & \mathbf{1}^{\mathrm{T}}\mathbf{x} & = & 1, \\ & \mathbf{x} & \geq & \mathbf{0}. \end{array} \tag{2.1}$$

We will reformulate the program to get a linear one. First, we introduce a new variable $\mathbf{y}$ to measure the difference between $\mathbf{Ax}$ and $\mathbf{g}$: $\mathbf{Ax} + \mathbf{y} = \mathbf{g}$. For a fixed $\mathbf{x}$ we have $\mathbf{y} = \mathbf{g}$ - $\mathbf{Ax}$ and some of its components can be negative or positive. That is why we introduce two nonnegative variables $\mathbf{y}^+$ and $\mathbf{y}^-$ by setting $\mathbf{y} = \mathbf{y}^+$ - $\mathbf{y}^-$. We obtain the program:

$$\begin{array}{lrcl} \min & \displaystyle\sum_{i=1}^{n} \left(y_i^+ + y_i^-\right) & & \\ \text{s.t.} & \mathbf{Ax} + \mathbf{y}^+ - \mathbf{y}^- & = & \mathbf{g}, \\ & \mathbf{1}^{\mathrm{T}}\mathbf{x} & = & 1, \\ & \mathbf{x}, \mathbf{y}^+, \mathbf{y}^- & \geq & \mathbf{0}. \end{array} \tag{2.2}$$

For a fixed $\mathbf{x}$, there are infinitely many choices of $\mathbf{y}^+$ and $\mathbf{y}^-$ to make the equation $\mathbf{Ax} + \mathbf{y}^+ - \mathbf{y}^- = \mathbf{g}$ hold. It is easy to see that among all these, the choice

$$\mathbf{y}^+ = \max\left(\mathbf{0}, \mathbf{g} - \mathbf{Ax}\right), \quad \mathbf{y}^- = -\min\left(\mathbf{0}, \mathbf{g} - \mathbf{Ax}\right) \tag{2.3}$$

minimizes the objective $\sum_{i=1}^{n}\left(y_i^+ + y_i^-\right)$. Moreover we have

$$|\mathbf{Ax} - \mathbf{g}| = \left|\mathbf{y}^+ - \mathbf{y}^-\right| = \mathbf{y}^+ + \mathbf{y}^-$$

for every $\mathbf{x}$ and for $\mathbf{y}^+$ and $\mathbf{y}^-$ optimal (given by (2.3))with respect to this $\mathbf{x}$. In particular this holds for $\mathbf{x}$ being the optimal solution of the original program (2.1). With this we

have justified the transition from the nonlinear program (2.1) to the linear one (2.2).

Adding the constraints representing the requested lower and upper bounds on the grading curve and rewriting the objective $\sum_{i=1}^{n}\left(y_i^+ + y_i^-\right)$ as $\mathbf{1}^{\mathrm{T}}\mathbf{y}^+ + \mathbf{1}^{\mathrm{T}}\mathbf{y}^-$, the final model is

$$\begin{aligned} \min \quad & \mathbf{1}^{\mathrm{T}}\mathbf{y}^+ + \mathbf{1}^{\mathrm{T}}\mathbf{y}^- \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} + \mathbf{y}^+ - \mathbf{y}^- = \mathbf{g}, \\ & \mathbf{A}_l\mathbf{x} - \mathbf{y}_l = \mathbf{l}, \\ & \mathbf{A}_u\mathbf{x} + \mathbf{y}_u = \mathbf{u}, \\ & \mathbf{1}^{\mathrm{T}}\mathbf{x} = 1, \\ & \mathbf{x},\, \mathbf{y}^+,\, \mathbf{y}^-,\, \mathbf{y}_l,\, \mathbf{y}_u \geq \mathbf{0}. \end{aligned} \tag{2.4}$$

The simplex table corresponding to this linear program is

| $\mathbf{0}$ | $-\mathbf{1}^{\mathrm{T}}$ | $-\mathbf{1}^{\mathrm{T}}$ | $\mathbf{0}$ | $\mathbf{0}$ | |
|---|---|---|---|---|---|
| $\mathbf{A}$ | $\mathbf{I}$ | $-\mathbf{I}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{g}$ |
| $\mathbf{A}_l$ | $\mathbf{0}$ | $\mathbf{0}$ | $-\mathbf{I}$ | $\mathbf{0}$ | $\mathbf{l}$ |
| $\mathbf{A}_u$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{I}$ | $\mathbf{u}$ |
| $\mathbf{1}^{\mathrm{T}}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{0}$ | 1, |

where $\mathbf{I}$ denotes the identity matrix of appropriate size (possibly different on each position), (0) denotes a matrix or vector of zeros (depending on its position) and the vector of decision variables is

$$\begin{pmatrix} \mathbf{x} \\ \mathbf{y}^+ \\ \mathbf{y}^- \\ \mathbf{y}_l \\ \mathbf{y}_u \end{pmatrix}.$$

Remark: The nonnegativity constraints for all the variables in models (2.2) and (2.4) are formulated in one expression (as the last row of the constraints). Note that this formulation is used as a shortcut. The vector variables appearing in the expression can be of various dimensions and there should be a zero vector $\mathbf{0}$ of corresponding dimension for each of them in the right-hand side. The meaning is however clear.

## 2.7 Model with ∞-norm

Minimizing the ∞-norm of $\mathbf{A}\mathbf{x} - \mathbf{g}$ leads to the following program:

$$\begin{aligned} \min \quad & \max_{i\in\{1,\ldots,n\}} \left| \sum_{j=1}^{m} a_{i,j}x_j - g_i \right| \\ \text{s.t.} \quad & \mathbf{1}^{\mathrm{T}}\mathbf{x} = 1, \\ & \mathbf{x} \geq \mathbf{0}. \end{aligned} \tag{2.5}$$

We will again seek for an equivalent linear program. We introduce a new nonnegative scalar variable $y$ and consider the program

$$\begin{aligned} \min \quad & y \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} + y\mathbf{1} \geq \mathbf{g}, \\ & \mathbf{A}\mathbf{x} - y\mathbf{1} \leq \mathbf{g}, \\ & \mathbf{1}^{\mathrm{T}}\mathbf{x} = 1, \\ & \mathbf{x} \geq \mathbf{0}, \\ & y \geq 0. \end{aligned} \tag{2.6}$$

From the first two constraints it follows that

$$y \geq (\mathbf{g} - \mathbf{A}\mathbf{x})_i = g_i - \sum_{j=1}^{m} a_{i,j}x_j \quad \forall i \in \{1, \ldots, n\}$$

and

$$y \geq (\mathbf{A}\mathbf{x} - \mathbf{g})_i = \sum_{j=1}^{m} a_{i,j}x_j - g_i \quad \forall i \in \{1, \ldots, n\},$$

which yields

$$y \geq \left| \sum_{j=1}^{m} a_{i,j}x_j - g_i \right| \quad \forall i \in \{1, \ldots, n\},$$

and therefore we have

$$y \geq \max_{i \in \{1,\ldots,n\}} \left| \sum_{j=1}^{m} a_{i,j}x_j - g_i \right|.$$

Moreover, for every fixed $\mathbf{x}$, the value $y = \max_{i \in \{1,\ldots,n\}} \left| \sum_{j=1}^{m} a_{i,j}x_j - g_i \right|$ is feasible in this program and obviously minimizes the objective ($\min y$). We have justified the transition from the nonlinear program (2.5) to the linear program (2.6).

To obtain the final version of the program in the standard form, we introduce surplus and slack vector variables $\mathbf{y}_1$, $\mathbf{y}_2$ to the first two constraints and add the constraints for the upper and the lower bound:

$$\begin{aligned} \min \quad & y \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} - \mathbf{y}_1 + y\mathbf{1} = \mathbf{g}, \\ & \mathbf{A}\mathbf{x} + \mathbf{y}_2 - y\mathbf{1} = \mathbf{g}, \\ & \mathbf{A}_l\mathbf{x} - \mathbf{y}_l = \mathbf{l}, \\ & \mathbf{A}_u\mathbf{x} + \mathbf{y}_u = \mathbf{u}, \\ & \mathbf{1}^{\mathrm{T}}\mathbf{x} = 1, \\ & \mathbf{x}, \mathbf{y}_1, \mathbf{y}_2, \mathbf{y}_l, \mathbf{y}_u \geq \mathbf{0}, \\ & y \geq 0. \end{aligned} \tag{2.7}$$

The simplex table corresponding to this linear program is

| | | | | | | |
|---|---|---|---|---|---|---|
| $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{0}$ | $-1$ | |
| $\mathbf{A}$ | $-\mathbf{I}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{1}$ | $\mathbf{g}$ |
| $\mathbf{A}$ | $\mathbf{0}$ | $\mathbf{I}$ | $\mathbf{0}$ | $\mathbf{0}$ | $-\mathbf{1}$ | $\mathbf{g}$ |
| $\mathbf{A}_l$ | $\mathbf{0}$ | $\mathbf{0}$ | $-\mathbf{I}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{l}$ |
| $\mathbf{A}_u$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{I}$ | $\mathbf{0}$ | $\mathbf{u}$ |
| $\mathbf{1}^{\mathrm{T}}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{0}$ | 1, |

and the vector of decision variables is the vector

$$\begin{pmatrix} \mathbf{x} \\ \mathbf{y}_1 \\ \mathbf{y}_2 \\ \mathbf{y}_l \\ \mathbf{y}_u \\ y \end{pmatrix}.$$

## 2.8 Joint objectives

Further modeling possibility is to formulate a joint objective function to optimize a certain combination of cost and quality (measured by the norm of $\mathbf{Ax} - \mathbf{g}$).

The objective function in (2.4) is then modified in the following way:

$$\min \ \alpha \mathbf{c}^{\mathrm{T}}\mathbf{x} + (1-\alpha)\left(\mathbf{1}^{\mathrm{T}}\mathbf{y}^{+} + \mathbf{1}^{\mathrm{T}}\mathbf{y}^{-}\right)$$

and the objective function in (2.7) reads as

$$\min \ \alpha \mathbf{c}^{\mathrm{T}}\mathrm{x} + (1-\alpha)y,$$

where the coefficient $\alpha \in [0,1]$ allows to adjust the relative proportion of the cost vector in the objective.

## 2.9 Further refinements

It may happen that some parts of the grading curve are more important than the others. We would like to include this preference into the optimization criterion. To achieve a minimum distance of the curves on the selected parts with priority, we can use weighed norms:

$$\text{the weighed 1-norm:} \qquad ||y||_1 = \sum_{i=1}^{n} w_i |y_i|,$$

$$\text{the weighed } \infty\text{-norm:} \qquad ||y||_\infty = \max_{i=1,\ldots,n} w_i |y_i|.$$

Here $\mathbf{w} = (w_i)_{i=1}^n$ is a vector of positive coefficients. It is reasonable to demand $\sum_{i=1}^n w_i = n$ or $\sum_{i=1}^n w_i = 1$. Scaled norms are obviously equivalent with respect to minimization.

In model (2.4), the objective will change to

$$\min \ \mathbf{w}^{\mathrm{T}}\mathbf{y}^{+} + \mathbf{w}^{\mathrm{T}}\mathbf{y}^{-}.$$

On the other hand, the weights will enter into the constraints in model (2.7). The first two constraints will be modified to

$$\begin{aligned} \mathbf{Ax} - \mathbf{y}_1 + y\mathbf{w}^{-1} &= \mathbf{g}, \\ \mathbf{Ax} + \mathbf{y}_2 - y\mathbf{w}^{-1} &= \mathbf{g}, \end{aligned}$$

where $\mathbf{w}^{-1}$ stand for the vector of reciprocal values of $\mathbf{w}$.

## 2.10 Reasons for a stochastic model

The input data of the models are the desired grading curve, the bounds and the grading curves of the ingredients. As was mentioned in the introduction, the grading curves of the ingredients are determined by measurement on a sequence of 14 sieves.

It is obvious that one cannot obtain identical results when the measurement is performed repeatedly, and in this case there are two reasons. The first reason is the nature of the measurement itself and it becomes evident when measuring the same collection of particles repeatedly. The second reason, which is of greater practical importance, is the non-homogeneity of the aggregate. Choosing different samples of the aggregate for the measurement and for the mixture certainly gives different results. And differences between two distinct series of production of the aggregate also fall into this category.

For these reasons, it is useful to develop stochastic models for the blending problem.

We return to the models in Chapter 4, after the presentation of stochastic programming that follows in the next chapter.

# Chapter 3

# Stochastic programming

## Notation

We use the following notation convention. A function $f : D \to C$ is always denoted by $f$ (with $D$ denoting the domain and $C$ the codomain of $f$). The symbol $f(x)$ denotes the element of $C$, which is assigned to the element of $D$ denoted by $x$. The function denoted by $f$ is then identically the set $\{\,(x, f(x)) \,|\, x \in D\}$, where $(a, b)$ denotes an ordered pair of elements $a$ and $b$. So, particularly for real functions, $f(x)$ denotes a real number, whereas $f$ denotes the (whole) function.

Although this is a standard mathematical convention, we state it explicitly as an information for the reader, that the text respects this notation and not bearing it in mind can bring difficulties in understanding.

Vector variables are *not* marked as boldface in this chapter. One reason is to avoid too much boldface notation. A more serious justification is that in most cases, the vector nature of the variables is not of importance in this Chapter.

## 3.1 Introduction

Mathematical programming is in its nature motivated by practical problems (consider for example various optimal planning problems, that led to the development of linear programming, see [2],[8] ). Despite of this, it can be treated mathematically in a correct way, completely disregarding the underlying practical motivation.

Our aim is to present the practical motivations as a reasonable justification for several concepts of stochastic programming, and then to treat the concepts purely in a mathematical sense. Conceptually, we will begin with simple considerations and we will see how they lead to generalizations.

The usual way of introducing stochastic programming is to present a deterministic program and then to claim that some of its data (coefficients of the objective function or constraints) are stochastic, therefore random variables (see e.g. [4], [7]). Further the fact is revealed, that having introduced random variables into the program, it is no longer

well-defined (i.e. it does not make sense). In order to cope with this, various reformulations are sought for.

We will adopt a different attitude in this text. We will start with probability space and mathematical program as two basic concepts and we will see how the usual stochastic programming formulations naturally arise from joining them together.

## 3.2 Basic considerations

A mathematical programming problem is usually presented in the general form:

$$\min_{x \in C} f(x).$$

The meaning is this: There is a set $C$ and a function $f : C \to \mathbb{R}$. The task is to find the minimum of the set $\{ f(x) \,|\, x \in C \} \subset \mathbb{R}$, provided that it exists. In the positive case (we denote the minimum by $f^*$), the task is also to determine the set

$$\{ x \,|\, x \in C \text{ and } f(x) = f^* \},$$

the so called set of optimal solutions. We call the member $x \in C$ an optimal solution if $f(x) = f^*$, and we call $f^*$ the value of optimal solution, the optimal value, or simply the optimum. The set of optimal solutions is usually denoted by $C^*$.

If the mathematical program represents a model of a real problem, the set $C$ represents a set of feasible solutions (or possible decisions, equivalently) and it is usually specified by constraints, which have concrete interpretations (e.g. technological constraints). Also the function $f$ (called the objective function) has a practical meaning (e.g. a cost function).

In practice it often happens, that the conditions, under which the constraints and the objective function were constructed, change in the course of time. In that case, if we want to solve the practical problem, we have to change the mathematical program. The change of the conditions may be random to certain extent. Or in order to take a decision concerning future, we are forced to create and solve the program in advance, before the conditions are realized, and we cannot predict them accurately due to some randomness involved. These considerations lead to the idea of joining mathematical programming with probability theory in order to obtain useful models for these problems.

It will prove beneficial to identify a mathematical program with its data, that is with the ordered pair $(C, f)$:

$$\min_{x \in C} f(x) \stackrel{\text{df}}{=} (C, f).$$

This might seem strange at first sight. We do it, apart from other reasons, to be able to speak about programs as mathematical objects and to be able to assign programs to elements of sets. This approach is not frequent, can be however found (in slight modification) for example in Kall [6]. Thus in the following text, the symbol $\min_{x \in C} f(x)$ just denotes the ordered pair $(C, f)$. We can still speak about the optimum (minimum) and about the set of optimal solutions of a given program. To see that this concept is quite natural and common, consider for example an ordered set $(X, \leq)$ and the set of its minimal elements as an analogy.

## 3.3 The first approach - wait-and-see

We try to build a basic probabilistic mathematical program. Let $(\Omega, \mathcal{F}, P)$ be a probability space. To every $\omega \in \Omega$, we assign a set $C_\omega$ and a function $f_\omega : C_\omega \to \mathbb{R}$. For every $\omega \in \Omega$ there are three possibilities. The assigned program

$$\min_{x \in C_\omega} f_\omega(x) \tag{3.1}$$

has an optimal solution, in which case we denote the optimal value by $f^*_\omega$. Or the function $f_\omega$ is unbounded from below on $C_\omega$, in this case we set $f^*_\omega = -\infty$. Or finally, $C_\omega = \emptyset$, in which case we set $f^*_\omega = \infty$.

We construct a function $f^* : \Omega \to \overline{\mathbb{R}}$ by assigning the values $f^*_\omega$ to the corresponding events:

$$f^* = \{ (\omega, f^*_\omega) \mid \omega \in \Omega \}.$$

We can ask now if $f^*$ is measurable (that is, whether it is a random variable) and if it is integrable. If both questions are answered affirmatively, we can determine the expected value $\mathbb{E}(f^*)$, for example.

Let's discuss this model from the practical point of view. To every event $\omega \in \Omega$, which represents a certain realization of relevant conditions, we have assigned a certain mathematical program. The reason for introducing this setting is the fact that different events can be assignment different programs. Now, we fix certain $\omega$ and solve the assigned program. We do it for every $\omega \in \Omega$. For this approach to be of practical relevance, the nature of the practical problem must make it possible for us to react to the upcoming events and with each of them to take the corresponding optimal decision. This situation is usually called *wait-and-see*. The value $\mathbb{E}(f^*)$ represents the expected value of optimal solutions.

It can be seen that this approach is nothing more than parametric programming, with $\Omega$ being the set of parameters, equipped with the structure of a probability space. This simple setting does not provide a model for decision under uncertainty, before the realization of the conditions becomes known.

## 3.4 From wait-and-see to here-and now

Suppose we are facing the decision problem under uncertainty. We have to make a decision before the realization of the conditions is known. Having chosen a particular decision, we will have to face different consequences depending on what conditions will come up. What decision is now the optimal one?

The choice of the decision should be based on an analysis of all its possible consequences. We are provided with objective functions $f_\omega$ corresponding to every single $\omega \in \Omega$, but they are (taken separately) of no use for us, since we cannot know in advance, which event will become true. We are lacking a unique objective function $F$, that would somehow aggregate the information provided by the single objective functions $f_\omega$. To set $F$ up, we would like to carry out the following analysis for every decision $x$:

1. We would like to see how it behaves under all possible conditions.
2. Assign to $x$ a value expressing the plausibility of its a behaviour.

We will give the two points a mathematical meaning:

1. Given a certain decision $x$, what are the values $f_\omega(x)$ of the objective functions $f_\omega$ for each possible event $\omega \in \Omega$? We may run into trouble here, because (given $x$) for some $\omega$ this question may not even make sense. The decision $x$ may not belong to $C_\omega$. However, at least for $x \in \bigcap_{\omega\in\Omega} C_\omega$ (if nonempty) this question makes sense.

Thus, fixed $x \in \bigcap_{\omega\in\Omega} C_\omega$, we construct the function $h_x : \Omega \to \mathbb{R}$ defined by

$$h_x(\omega) = f_\omega(x).$$

2. If for every $x \in \bigcap_{\omega\in\Omega} C_\omega$ the function $h_x$ is measurable and integrable (or even square integrable), we can use the functionals $\mathbb{E}$ or $\mathbb{V}\mathrm{ar}$ to assign the value $\mathbb{E}(h_x)$ or $\mathbb{V}\mathrm{ar}(h_x)$ to $x$, that is, to define a function

$$F : \bigcap_{\omega\in\Omega} C_\omega \to \mathbb{R}$$

by

$$F(x) = \mathbb{E}(h_x), \quad \text{or by} \quad F(x) = \mathbb{V}\mathrm{ar}(h_x).$$

We can obviously use some general functional $H$ defined on $\mathbb{R}^\Omega$ to assign the value $H(h_x)$ to $x$. If we use the expected value $\mathbb{E}$, the resulting value tells us how good (or bad) the decision is in average, and this might seem as a reasonable objective. If we are afraid of decisions that would bring a big variability of outcomes, we can use the functional $\mathbb{V}\mathrm{ar}$.

The final step of solving this model lies in finding the minimum of $F$ on its domain.

## 3.5 The second approach - here-and-now

If we consider a model, where for all events $\omega$ the set $C_\omega$ is the same (let's denote it by $C$), or more specially, if $C_\omega = \mathbb{R}^n$ for all $\omega$ (this represents an unconstrained problem), the approach described in the previous section can be well used. We call it the *here-and-now* approach. Using the functional $\mathbb{E}$, the mathematical program

$$\min_{x\in C} F(x),$$

where $F : C \to \mathbb{R}$ is given by

$$F(x) = \mathbb{E}(h_x) = \mathbb{E}(\ \{\,(\omega, f_\omega(x))\,|\,\omega \in \Omega\}\ ),$$

is usually called the *expected objective reformulation* (EO reformulation).

In stochastic programming textbooks (see [4], [7], [9]), its usual statement is

$$\min_{x\in C} \mathbb{E}\left(f(x,\omega)\right).$$

This notation is incosistent with our notation. The symbol $f(x,\omega)$ in this notation refers to our value $f_\omega(x)$, to our function $f_\omega$ (for fixed $\omega$) and to our function $h_x$ (for fixed $x$). The notation itself doesn't present a problem when one can distinguish the real meaning in all cases.

## 3.6 Two-stage models

Further step in our development of stochastic programming lies naturally in joining the here-and-now and wait-and-see approaches. Suppose the feature of the decision problem is this: We have to take a certain decision in advance (that is a here-and-now decision, called also a first stage decision) and then, after the realization of relevant conditions, take a second stage decision (which is a wait-and-see one). In order to provide a proper formulation for two-stage models, we need to return to wait-and-see models and discuss their possible formulations.

### 3.6.1 Wait-and-see formulations

We have discussed the motivation for wait-and-see model in Section (3.3). For a given $\omega \in \Omega$, we have assigned a set $C_\omega$ and a function $f_\omega$ with $C_\omega$ as its domain. The function $f^* : \Omega \to \overline{\mathbb{R}}$ assigns the optimal value of the program

$$\min_{x \in C_\omega} f_\omega(x) \tag{3.2}$$

to $\omega$, provided the optimum exists. Otherwise $f^*(\omega) = \infty$ or -$\infty$ (see Section (3.3)). If $f^*$ is measurable and integrable, we determine $\mathbb{E}(f^*)$.

The procedure just described is still not a mathematical program in the sense of Section (3.2), we haven't obtained the value $\mathbb{E}(f^*)$ as a minimum of any objective function $f$ on any underlying set $C$. In the sequel, we will search for such a program $(C, f)$.

If we already had $(C, f)$ at hand, we would expect its optimal solution to represent some optimal decision. But what is an optimal decision in this case? There is in fact a collection of optimal decisions, each of them corresponding to the program (3.2) for different $\omega$. Solving the programs (3.2) separately is sometimes called *scenario analysis*, with scenario being a synonym for a random event (i.e. $\omega$).

We can try to construct the program $(C, f)$ in such a way, that its optimal decision consists of the "vector" of optimal solutions of the single programs (3.2). This leads us to the specification of the underlying set $C$. We will first present the construction on a simple exapmle.

**Example**: Let $\Omega = \{1, 2\}$, $\mathcal{F} = \mathcal{P}(\Omega)$ (the power set of $\Omega$), and $P$ be some probability measure on $\mathcal{F}$. Let

$$\begin{array}{llll} C_1 = [0,1] \subset \mathbb{R} & \text{and} & f_1(z) = 3z + 2 & \text{for } z \in C_1, \\ C_2 = [3,8] \subset \mathbb{R} & \text{and} & f_2(z) = -z & \text{for } z \in C_2. \end{array}$$

Then $f_1^* = 2$ at $z_1^* = 0$ and $f_2^* = -8$ at $z_2^* = 8$. The optimal solution $x^*$ of the program $(C, f)$ (which is still left to be defined), should be the ordered pair $x^* = (0, 8)$. This leads to the idea of defining the underlying set $C$ as some subset of $\mathbb{R}^2$, namely the Cartesian product $[0, 1] \times [3, 8]$. Then it is natural to define the objective function $f$ on $C$ as

$$f(x, y) = f_1(x) + f_2(y). \tag{3.3}$$

We emphasize here our notation convention, due to which + in (3.3) means addition of numbers (as opposed to addition of functions; clearly $f_1$ and $f_2$ are defined on different domains). Thus our program reads as

$$\begin{aligned} \min \quad & 3x - y + 2 \\ \text{s.t.} \quad & (x, y) \in [0, 1] \times [3, 8]. \end{aligned} \tag{3.4}$$

What we have done here is that we have "grouped" together the two programs, creating one which is "separable". In the following, we will give the terms grouped and separable a precise meaning. The optimal solution $x^*$ of (3.4) is $x^* = (0, 8)$ with the optimal value $f_{opt} = f_1^* + f_2^* = 2 - 8 = -6$.
Let's turn our attention to the probability now. Let the probability measure in this example be defined by $P(\{1\}) = 0.8$ and $P(\{2\}) = 0.2$. We have $f^* = \{(1, 2), (2, -8)\}$ and $\mathbb{E}(f^*) = 0.8 \times 2 + 0.2 \times (-8) = 0$. We see that $\mathbb{E}(f^*) \neq f_{opt}$. To fix this inconvenience, we redefine $f$ in (3.3) as

$$f(x, y) = P(\{1\})f_1(x) + P(\{2\})f_2(y),$$

which modifies the objective in (3.4) to

$$\min \quad 2.4x - 0.2y + 1.6.$$

The set of optimal solutions is left unchanged, and we get $f_{opt} = P(\{1\})f_1^* + P(\{2\})f_2^* = = \mathbb{E}(f^*)$. ⌟

We will now formulate the sketched procedure precisely. We will start with the notion of separability which will prove very useful in later formulations. The usual definition of separability is this:

**Definition 1'** ***(Separable function)***: Let $f : \mathbb{R}^n \to \mathbb{R}$ be a function and $J$ be an index set: $J = \{1, \ldots, n\}$. If there exist functions $f_i : \mathbb{R} \to \mathbb{R}$ for each $i \in J$ such that the identity $f(x_1, \ldots, x_n) = \sum_{i=1}^n f_i(x_i)$ holds for each $x = (x_1, \ldots, x_n) \in \mathbb{R}^n$, we say that $f$ is separable.

The definition is motivated by the fact, that we would like to call the function $f : \mathbb{R}^3 \to \mathbb{R}$ given by $f(x, y, z) = x - \sin(y) + 5z^3$ separable, and other "similar" functions as well. Proper specification of this "similarity" is the main burden of the definition.

For our purposes, we will generalize the domain. Instead of Cartesian product of n-tuple of sets of real numbers, we will take a Cartesian product of n-tuple of some general sets, each of them possibly different.

**Definition 1** ***(Separable function)***: Let $J = \{1, \ldots, n\}$ be an index set and let $S_i$ denote some set for each $i \in J$. Let $f : \prod_{i=1}^n S_i \to \mathbb{R}$ be a function. If there exist functions $f_i : S_i \to \mathbb{R}$ for each $i \in J$ such that the identity $f((x_1, \ldots, x_n)) = \sum_{i=1}^n f_i(x_i)$ holds for each $x = (x_1, \ldots, x_n) \in \prod_{i=1}^n S_i$, we say that $f$ is separable.

A special case of this definition might be to take $S_i = \mathbb{R}^{m_i}$. Then $\prod_{i=1}^n S_i = \mathbb{R}^{m_1 + \ldots + m_n}$. We can call a function defined by $f(x, y, z) = (x - y)^2 + z$ separable (bearing in mind

that we separate the domain $\mathbb{R}^3$ into $\mathbb{R}^2 \times \mathbb{R}$). This is usually called *block-separability.*

**Definition 2** ***(Separable program)***: We say that a mathematical program $(C, f)$ is separable, if $C$ is a Cartesian product: $C = \prod_{i=1}^n C_i$, and $f : C \to \mathbb{R}$ is a separable function.

**Theorem 1** ***(Equivalence of programs)***: Let $(C, f)$ be a separable program with $C = \prod_{i=1}^n C_i$. Let's denote the optimum (provided it exists) of $(C, f)$ by $f^*$ and the optima of the programs $(C_i, f_i)$ by $f_i^*$ (the functions $f_i$ are given as in Definition 1). Let's denote the sets of optimal solutions of the corresponding programs by $C^*$ and $C_i^*$. Then $f^* = \sum_{i=1}^n f_i^*$ and $C^* = \prod_{i=1}^n C_i^*$, provided $C^*$ or $\prod_{i=1}^n C_i^*$ is nonempty.

**Proof:** Suppose that $\prod_{i=1}^n C_i^*$ is nonempty. For each $i$, we construct a function $g_i : C \to \mathbb{R}$ by letting $g_i(x_1, \ldots, x_n) = f_i(x_i)$. Then the programs $(C, g_i)$ and $(C_i, f_i)$ have the same optima $f_i^*$ and the set $G_i^*$ of optimal solutions of $(C, g_i)$ is equal to $C_1 \times \ldots \times C_i^* \times \ldots \times C_n$. Further, the identity $f = \sum_{i=1}^n g_i$ holds. Therefore

$$\min \{ f(x) \mid x \in C \} = \min \Big\{ \sum_{i=1}^n g_i(x) \mid x \in C \Big\}$$

and

$$\min \Big\{ \sum_{i=1}^n g_i(x) \mid x \in C \Big\} \geq \sum_{i=1}^n \min \{ g_i(x) \mid x \in C \} = \sum_{i=1}^n f_i^*$$

On the other hand, for $x \in \prod_{i=1}^n C_i^*$ we have $f(x) = \sum_{i=1}^n g_i(x) = \sum_{i=1}^n f_i^*$, and therefore $\min \{ f(x) \mid x \in C \} \leq \sum_{i=1}^n f_i^*$. So the first equality is established, and moreover for all $x \in \prod_{i=1}^n C_i^*$ we have $f(x) = f^*$ and therefore $\prod_{i=1}^n C_i^* \subset C^*$ (which means that $C^*$ is nonempty).
Now let $C^*$ be nonempty and let $x \in C^*$. Since $f(x) = \sum_{i=1}^n g_i(x)$, we claim that $g_i(x) = \min \{g_i(y) \mid y \in C\}$. Suppose it is not true. Then there exists $y \in C$ with $g_i(y) < g_i(x)$. Since $g_i(x_1, \ldots, x_n) = f_i(x_i)$, the i-th projection $y_i$ of the member $y$ must be different from $x_i$. Moreover, at the point $z = (x_1, \ldots, x_{i-1}, y_i, x_{i+1}, \ldots, x_n)$ we have $g_i(z) = g_i(y) < g_i(x)$. For $j \neq i$, we have $g_j(z) = g_j(x)$. Therefore $f(z) = \sum_{j=1}^n g_j(z) = \sum_{j \neq i} g_j(x) + g_i(z) < \sum_{j=1}^n g_j(x) = f(x)$, contradicting the optimality of $f$ at $x$. This statement about optimality of $g_i$ holds for each $i \in J$, and therefore at $x$ we have $f^* = \sum_{i=1}^n f_i^*$. Moreover, since $g_i$ is minimal in $x$, $x_i$ belongs to $C_i^*$, and we obtain that $x$ belongs to $\prod_{i=1}^n C_i^*$. This holds for arbitrary $x \in C$, so we have $C \subset \prod_{i=1}^n C_i^*$. This finishes the proof. ⌟

Especially note, that the proof is valid for arbitrary sets $C_i$, there is no need to require them to be subsets of any linear space nor $\mathbb{R}^n$. We are now ready to generalize the approach given in the previous example.

**Definition 3** ***(Product program)*** : Let $J = \{1, \ldots, n\}$ be an index set and let a program $(C_i, f_i)_{i=1}^n$ be assigned to every $i \in J$. Set $C = \prod_{i=1}^n C_i$ and define the function $f : C \to \mathbb{R}$ by $f(x_1, \ldots, x_n) = \sum_{i=1}^n f_i(x_i)$. The program $(C, f)$ is said to be the *product program* assigned to the $n$-tuple of programs $(C_i, f_i)_{i=1}^n$.

It is obvious from the definition that the product program $(C, f)$ is separable and therefore by Theorem 1 we know that the sets of optimal solutions are in exact correspondence. We are also provided with a relation linking the optimal values.

We come to the main goal of this section, that is to the formulation of wait-and-see stochastic program with finite probability space:

**Definition 4** ***(Wait-and-see program)***: Let $\Omega$ be finite with $\mathrm{card}(\Omega) = n$ and let's denote the members of $\Omega$ by $\omega_i$, $i \in \{1, \ldots, n\}$. Let $\mathcal{F} = \mathcal{P}(\Omega)$ and let the probability measure $P : \mathcal{F} \to \mathbb{R}$ be given by $P(\{\omega_i\}) = p_i$, where $p_i > 0$, $\sum_{i=1}^{n} p_i = 1$. Let there be a program $(C_{\omega_i}, f_{\omega_i})$ assigned to every $\omega_i \in \Omega$. We call the product program $(C, f)$ assigned to $(C_{\omega_i}, p_i f_{\omega_i})_{i=1}^{n}$ the wait-and-see program with finite probability space.

**Theorem 2** ***(Optimum of wait-and-see)***: The optimum $f_{opt}$ of the wait-and-see program $(C, f)$ as given in Definition 4 is equal to $\mathbb{E}(f^*)$, where the random variable $f^* : \Omega \to \mathbb{R}$ is given by $f^*(\omega_i) = f_i^*$, provided the programs $(C_{\omega_i}, f_{\omega_i})_{i=1}^{n}$ have optimal solutions (with the optimal values denoted by $f_i^*$).

**Proof:** If the programs $(C_{\omega_i}, f_{\omega_i})_{i=1}^{n}$ have optimal solutions, the Cartesian product $\prod_{i=1}^{n} C_{\omega_i}^*$ of sets $C_{\omega_i}^*$ of their optimal solutions is nonempty. The programs $(C_{\omega_i}, p_i f_{\omega_i})$ are obtained from the original programs $(C_{\omega_i}, f_{\omega_i})$ by replacing the original objective functions $f_{\omega_i}$ with the objective functions $p_i f_{\omega_i}$ defined by $(p_i f_{\omega_i})(x) = p_i f_{\omega_i}(x)$ on $C_{\omega_i}$. Clearly, the sets $C_{\omega_i}^*$ of optimal solutions are left unchanged by this scaling of objective functions, since $p_i > 0$. The optimal values $f_i^*$ are scaled to $p_i f_i^*$. By Theorem 1 we then have $f_{opt} = \sum_{i=1}^{n} p_i f_i^* = \mathbb{E}(f^*)$. (The function $f^*$ is measurable because $\mathcal{F} = \mathcal{P}(\Omega)$). ⌟

So the optimum of the wait-and-see program $(C, f)$ is given by the expected value of optimal solutions of the original $n$-tuple of programs. That is exactly what we wanted to get.

We have constructed the wait-and-see program for the expected value in objective. The main use of this formulation takes place in two-stage stochastic programming models (which use the expectation). This formulation of the wait-and-see program is sometimes called the ***extensive form of stochastic program*** (see Birge and Louveaux [4]) and allows the L-shaped decomposition algorithm to be applied.

We have managed to get the formulation of the wait-and-see model as a mathematical program in case of finite probability space. We will see now, what difficulties arise when we turn our attention to the infinite case. There are no problems with infinite Cartesian products of sets. Recall that the general Cartesian product is defined as

$$\prod_{i \in J} C_i = \left\{ t : J \to \bigcup_{i \in J} C_i \,|\, t(i) \in C_i \ \forall i \in J \right\}, \tag{3.5}$$

that is the set of all mappings $t$ from $J$ to $\bigcup_{i \in J} C_i$, for which $t(i)$ is a member of $C_i$.

The definition of separable function (see Definition 1) uses addition, which is undefined for infinite cases. We can try to formulate directly the separable objective function $f$

of the wait-and-see program. For simplicity of notation, we identify $\Omega$ with $J$, so that instead of indexing by $\omega_i$ we can index by $i$. In the finite case we have

$$f(x_1, \ldots, x_n) = \sum_{i=1}^{n} p_i f_i(x_i)$$

for every $x = (x_1, \ldots, x_n) \in \prod_{i \in J} C_i$. Thus generally, we would like to have

$$f(t) = \int_J h_t \, \mathrm{d}P,$$

where the function $h_t : J \to \mathbf{R}$ is given by

$$h_t(i) = f_i\left(t(i)\right)$$

for every $t \in \prod_{i \in J} C_i$ as defined by (3.5).

The problem is that the function $h$ is not likely to be measurable nor integrable. Even if we claim that the sets $C_i$ belong to a common measurable space and the mapping $t$ is measurable, with hope to obtain the measurability of $h$ by the theorem about measurability of composed mappings, we don't succeed. The function $h$ is not defined by composition of $t$ with another single mapping.

There is a result by Kall (see [6]), that shows the measurability of one concrete mapping $h$ in a special case. It is the case when the programs $(C_i, f_i)$ are linear programs with $C_i \subset \mathbb{R}^k$ for each $i \in J$ and the mapping $h$ is the "optimal mapping" defined by $h(i) = f_i^*$. Note that it is exactly the mapping $f^*$ and its expected value $\mathbb{E}(f^*)$, that were of main interest in this section. Kall shows that under further assumptions on the data of the linear programs, $f^*$ is also integrable.

To summarize, we managed to assign a mathematical program to the wait-and-see problem in the finite case. In the infinite case, the wait-and-see problem is left without a mathematical program, but under suitable conditions on the data, we have at least the mapping $f^*$ measurable and integrable. We will shortly return to the measurability of $f^*$ in Section 3.10.

### 3.6.2 Formulation of two-stage models

As mentioned at the beginning of this section, in two-stage models some of the decisions are supposed to be here-and-now decisions, while the remaining ones are wait-and-see decisions. This means that the sets $C_\omega$ are somehow divided with respect to the interpretation of their members (decisions). The first idea would be to take $C_\omega = X \times D_\omega$ with $X$ the set of here-and-now decisions. This setting would however not be rich enough - we wouldn't be able to model situations in which the set of second stage decisions possibly varies with the first stage decision taken, which is quite frequent in practise. Thus we set

$$C_\omega = \bigcup_{x \in X} \{\, x\} \times D_{x,\omega},$$

where $D_{x,\omega}$ is the set of second stage decisions corresponding to the event $\omega$ provided that first stage decision $x \in X$ has been taken. When referring to the values of our objective functions $f_\omega : C_\omega \to \mathbb{R}$, we will use the notation $f_\omega(x, y)$, with $x \in X$ and $y \in D_{x,\omega}$.

For each $x \in X$ and every $\omega \in \Omega$, using the objective function $f_\omega$, we define the second-stage objective functions $f_{x,\omega} : D_{x,\omega} \to \mathbb{R}$ by

$$f_{x,\omega}(y) = f_\omega(x, y) \tag{3.6}$$

and we consider the programs $(D_{x,\omega}, f_{x,\omega})$. This family of programs represents a wait-and-see model discussed in the previous subsection. In the two-stage setting, we have such a family assigned to every $x \in X$. For fixed $x$, we proceed as in the previous subsection. We denote the optimum of $(D_{x,\omega}, f_{x,\omega})$ (or $\pm\infty$) by $f^*_{x,\omega}$ and then we construct a function $Q_x : \Omega \to \mathbb{R}$ by

$$Q_x(\omega) = f^*_{x,\omega},$$

which we assume to be an integrable random variable. This allows us to define an objective function $Q : X \to \mathbb{R}$ as

$$Q(x) = \mathbb{E}(Q_x). \tag{3.7}$$

The two-stage program we sought for is then the program $(X, Q)$, minimizing the function $Q$ over the set of first stage decisions $X$. Having found the optimal solution $x^*$, the set of optimal second-stage decisions for each event $\omega$ is clearly the set $D^*_{x^*,\omega}$.

The practical interpretation of the two-stage model is the following. We rate the first-stage decisions $x$ by taking into account the various situations that may arise when the uncertain conditions are realized, provided we have taken the decision $x$. Each of the situations represents a decision problem, which we solve optimally. We have the expected value of the optima as the criterion for choosing $x$.

We will now present how the general linear two-stage model, as stated in various texts ([4],[7],[9]), fits into our formulation.

**The linear two-stage model**:

$$\min \quad c^{\mathrm{T}}x + \mathbb{E}\Big( \min\left(q^{\mathrm{T}}(\omega)y(\omega)\right) \Big) \tag{3.8}$$

$$\begin{aligned} \text{s.t.} \qquad Ax &= b, \\ W(\omega)y(\omega) &= h(\omega) - T(\omega)x, \\ x &\geq 0, \\ y(\omega) &\geq 0 \quad \forall \omega \in \Omega. \end{aligned}$$

Here $A \in \mathbb{R}^{m\times n}$, $b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$, $W(\omega) \in \mathbb{R}^{r\times s}$, $h(\omega) \in \mathbb{R}^r$, $T(\omega) \in \mathbb{R}^{r\times n}$ and $q(\omega) \in \mathbb{R}^s$, for every $\omega \in \Omega$. The variables of the model are $x$ (a variable of $\mathbb{R}^n$) and $y(\omega)$ (variables of $\mathbb{R}^s$ for each $\omega \in \Omega$).

Observe that the notation in (3.8) is not compatible with our notation convention. The reasoning is the same as in Section 3.5. The set $X$ of first stage decisions is defined by

$$X = \{x \in \mathbb{R}^n | Ax = b,\ x \geq 0\}.$$

The sets $D_{x,\omega}$ of second stage decisions (with $x$ and $\omega$ fixed) are specified by

$$D_{x,\omega} = \{y \in \mathbb{R}^s \,|\, W(\omega)y = h(\omega) - T(\omega)x,\ y \geq 0\}\,. \tag{3.9}$$

The second-stage objective functions $f_{x,\omega}$ are given by the expression

$$f_{x,\omega}(y) = c^{\mathrm{T}}x + q^{\mathrm{T}}(\omega)y \tag{3.10}$$

on $D_{x,\omega}$ (with $x$ and $\omega$ fixed). The original objective functions $f_\omega$ are given by the same expression on $\bigcup_{x\in X}\{x\} \times D_{x,\omega}$, with $x$ and $\omega$ as variables (not fixed).

**Important remark:** Note the use of $y$ instead of $y(\omega)$ in (3.9). At this point it is revealed, that although the traditional notation $y(\omega)$ suggests viewing $y$ as a random variable, the meaning is definitely different. The notation just means that the programs $(D_{x,\omega}, f_{x,\omega})$ are required to be solved separately, and the way of communicating this consists in indexing the variables of the domains (by $\omega$).

The property of indexing can be well seen in the case of finite probability space, when we can formulate the wait-and-see model of the second stage as a mathematical program. Then the entire two-stage model can be formulated as a mathematical program $(C, f)$ in the extensive form. The underlying set $C$ is specified by

$$C = \bigcup_{x\in X} \{x\} \times D_{x,\omega_1} \times \ldots \times D_{x,\omega_n}$$

and the objective function $f$ on $C$ is defined as

$$f(x, y_1, \ldots, y_n) = \sum_{i=1}^{n} p_i f_{x,\omega_i}(y_i) = \sum_{i=1}^{n} p_i f_{\omega_i}(x, y_i). \tag{3.11}$$

The second equality in (3.11) follows from the definition of $f_{x,\omega_i}(y_i)$ given by(3.6). The aforementioned indexing property for finite distribution is obvious here.

In case when $f_{\omega_i}(x, y_i) = F(x) + G_{\omega_i}(y_i)$, we have

$$f(x, y_1, \ldots, y_n) = F(x) + \sum_{i=1}^{n} p_i G_{\omega_i}(y_i),$$

since $\sum_{i=1}^{n} p_i = 1$. Here $F(x)$ corresponds to $c^{\mathrm{T}}x$ and $G_{\omega_i}(y_i)$ corresponds to $q(\omega)^T y(\omega)$ in the usual formulation (3.8) of the two-stage model.

There is also another usual form of statement of the linear two-stage model, the so called *nested form.* We will state it for completeness and relate it to our development of the two-stage model.

**The nested form of linear two-stage model**:

$$\begin{aligned} \min \quad & c^{\mathrm{T}}x + \mathbb{E}(Q(x,\omega)) \\ \text{s.t.} \quad & Ax = b, \\ & x \geq 0, \end{aligned} \tag{3.12}$$

where

$$Q(x,\omega) = \min\Big\{q^{\mathrm{T}}(\omega)y \,|\, W(\omega)y = h(\omega) - T(\omega)x, y \geq 0\Big\}.$$

Note that this statement corresponds to our two-stage model $(X, Q)$. The expression $\mathbb{E}(Q(x,\omega))$ should be understood as defining the function $Q$ on $X$ introduced in (3.7). Altough the notation in (3.12) is again not consistent with the ours, it is not misleading and we will use it in subsequent statements of two-stage programs.

The usual formulation of two-stage problem, which basically defines all the sets $D_{x,\omega}$ at once using (3.9), brings some difficulties. It may happen that the expression (3.9) defines an empty set for some of the first-stage decisions. Various properties of the second-stage data $W(\omega)$, $T(\omega)$, $h(\omega)$ and $q(\omega)$ are usually required in order to make sure this does not happen. With various requirements on the data, we speak about *complete, relatively complete* or *simple recourse* (see e.g. [4], [7]).

## 3.7 Considerations regarding feasibility

In this section we turn our attention to the general here-and-now model as introduced in Section 3.3. From our discussion of feasibility, we will arrive at probabilistically constrained programs and we will also derive independently the two-stage model.

Recall that using the here-and-now approach, we were able to deal with unconstrained problems or problems where all the programs were defined on a common set $C$. If it is not the case, we can perform the here-and-now style analysis only for the decisions that are always feasible, belonging to $\bigcap C_\omega$. In some applications, certain decision might still seem plausible even if it is infeasible in some cases, given their probability is small enough. Or the infeasibility may seem to be compensated by extremely good behaviour in the feasible cases. For these reasons, it is useful to extend the objective functions $f_\omega$ outside the sets $C_\omega$. We will assume in this section that $C_\omega \subset \mathbb{R}^n$ for all $\omega \in \Omega$.

We say that $\overline{f}_\omega : \mathbb{R}^n \to \mathbb{R}$ is an extension of $f_\omega : C_\omega \to \mathbb{R}$, if we have $\overline{f}_\omega(x) = f_\omega(x)$ for all $x \in C_\omega$.

It will be useful to discuss possible forms of extension in more detail. From a mathematical point of view, the extension can be arbitrary. From a modelling perspective, we would like the values $\overline{f}_\omega(x)$ for $x \in \mathbb{R}^n \setminus C_\omega$ to reflect somehow the infeasibility of $x$, and possibly to give a rough information about the distance of $x$ from $C_\omega$. The distance might be viewed as a measure of infeasibility and might be useful from the modelling point of view.

At this point, we will take a look at the usual way of specification of $C_\omega$ and $f_\omega$. The mathematical program assigned to $\omega$ might be a linear program or a general nonlinear program. In the nonlinear case, the set $C_\omega$ is given by a system of inequalities $(g_{\omega,i} \leq 0)_{i=1}^m$, where $g_{\omega,i} : \mathbb{R}^n \to \mathbb{R}$ are given functions. The precise meaning is this:

$$C_\omega = \bigcap_{i=1}^{m} \{x \,|\, x \in \mathbb{R}^n \text{ and } g_{\omega,i}(x) \leq 0\}\,. \tag{3.13}$$

The objective function $f_\omega$ is usually given by an expression, which makes sense for all $x \in \mathbb{R}^n$. So it is natural to have the objective function $f_\omega : \mathbb{R}^n \to \mathbb{R}$, instead of $f_\omega : C_\omega \to \mathbb{R}$, given by the same expression.

Our $f_\omega$ is extended now, but without including the information about feasibility. This information is contained in the values of the functions $g_{\omega,i}$ at $x$. Consider a suitable loss function $t_\omega : \mathbb{R}^{m+1} \to \mathbb{R}$ and define $\overline{f}_\omega$ by composition as

$$\overline{f}_\omega(x) = t_\omega\Big(\, g_{\omega,1}(x), \ldots, g_{\omega,m}(x),\; f_\omega(x)\Big) \quad \forall x \in \mathbb{R}^n.$$

If desired, the function $t$ can be defined in such a way, that $\overline{f}_\omega$ is indeed an extension of $f_\omega$. It is particularly easy to see in case of linear programming model. The linear program reads as

$$\min\Big\{\, c_\omega^{\mathrm{T}} x \,|\, A_\omega x = b_\omega \text{ and } x \geq 0\Big\}$$

and the functions $g_{\omega,i}$ are linear and can be represented as

$$g_{\omega,i}(\mathbf{x}) = \Big(A_\omega^{\mathrm{T}} x - b_\omega\Big)_i,$$

where the index $i$ refers to the i-th component of the right-hand side vector. If we set, for example, $t_\omega(x_1, \ldots, x_m) = \sum_{i=1}^{m+1} x_i$, the composed mapping $\overline{f}_\omega$ will coincide with $f_\omega$ on $C_\omega$, because all the functions $g_{\omega,i}$ have zero value on $C_\omega$.

## 3.8 Probabilistic constraints

The concept of probabilistic constraints naturally follows from the considerations regarding feasibility presented in the previous section. Recall that for each $\omega$, we have a collection of $m$ functions $g_{\omega,i}$, each of them defined on the whole $\mathbb{R}^n$. The set $C_\omega$ assigned to $\omega$ is uniquely specified by (3.13).

It is important to note that every $\omega$ is assigned the same number $m$ of constraints. The applications of stochastic programming often arise from deterministic (linear or nonlinear) problems by introducing the randomness to the coefficients of the constrains. In this situation, although the constraints are different for each $\omega$, their number stays the same in all cases.

In the very general situation we obviously don't need to limit ourselves to this setting. Generally, if there exists an $\omega^*$ with maximal number of constraints, we can add the appropriate number of constraints to the remaining members of $\Omega$. These added constraints

can be represented by constant functions (with constant value 0 or -1, for example) on the whole $\mathbb{R}^n$. This guarantees that their adding doesn't affect the original feasible regions $C_\omega$.

Our effort up to now has been to extend the objective functions $f_\omega$ to $\mathbb{R}^n$, in order to be able to evaluate every possible decision, no matter if it is feasible for certain $\omega$ or not. Having done this, we can solve the unconstrained problem, using for example the expected objective approach (see section (3.5)).

Now, we present a possible way of restriction of the domain $\mathbb{R}^n$. For every fixed $x \in \mathbb{R}^n$ and every $i \in \{1, \dots, m\}$ (recall the fixed number of constraints) we define a function $G_{x,i} : \Omega \to \mathbb{R}$ by

$$G_{x,i}(\omega) = g_{\omega,i}(x). \tag{3.14}$$

Provided all the functions $G_{x,i}$ are measurable, we have obtained a collection of $m$ random variables at every point $x$. We will assume the measurability of the functions $G_{x,i}$ in the sequel.

Consider a particular random variable, say $G_{x^*,1}$ and the set $S_{x^*} = \{\omega | G_{x^*,1}(\omega) \leq 0\}$. Since $G_{x^*,1}$ is measurable, the set $S_{x^*} \subset \Omega$ belongs to the $\sigma$-algebra $\mathcal{F}$ and has assigned a probability $P(S_{x^*})$. If all the functions $g_{\omega,1}$ are less or equal than 0 at $x^*$, meaning that the first constraint is satisfied for all $\omega$ at $x^*$, then $G_{x^*,1} \leq 0$ for every $\omega$ and therefore $S_x = \Omega$ and $P(S_x) = 1$. On the other hand, if for some $\omega$ the first constraint at $x^*$ is not satisfied ($g_{\omega,1} > 0$), we can expect the the probability of $S_{x^*}$ to be less than 1. If the first constraint is violated at $x^*$ for each $\omega$, then obviously $S_{x^*} = \emptyset$ and $P(S_{x^*}) = 0$.

Let's build a new function, that assigns the probability $P(S_x)$ to every $x \in \mathbb{R}^n$:

$$H_1 : \mathbb{R}^n \to [0, 1], \qquad H_1(x) = P\Big( \{\omega | G_{x,1}(\omega) \leq 0\} \Big). \tag{3.15}$$

The function $H_1$ classifies the decisions $x$ according to the probability of the set of events, for which the first constraint is satisfied at $x$. So, for example the set $\{x | H_1(x) > 0.95\}$ has a natural interpretation as the set of decisions, for which the first constraint is satisfied with probability greater than 0.95.

We will now use a similar construction taking the m-tuple of functions $(g_{\omega,i})_{i=1}^m$ at once. For a fixed $x$, find the set

$$S_x = \{ \omega \,|\, G_{x,i}(\omega) \leq 0 \text{ for } i \in \{1, \dots, m\}\} =$$
$$= \{\omega \,|\, G_{x,1}(\omega) \leq 0\} \cap \{\omega \,|\, G_{x,2}(\omega) \leq 0\} \cap \dots \cap \{\omega \,|\, G_{x,m}(\omega) \leq 0\}\,.$$

The set $S_x$ belongs to $\mathcal{F}$, being an intersection of sets belonging to $\mathcal{F}$. We define a function $H : \mathbb{R}^n \to [0, 1]$ by

$$H(x) = P(S_x) = P\Big( \{ \omega \,|\, G_{x,i}(\omega) \leq 0 \text{ for } i \in \{1, \dots, m\}\} \Big). \tag{3.16}$$

For a finite probability space, we have $H(x) = 1$ if and only if all the constraints are satisfied at $x$ for all $\omega$, that is if and only if $x \in C_\omega$, meaning that $x$ is always feasible. We can see that the following identity holds:

$$H(x) = P(\{\omega \,|\, x \in C_\omega\}). \tag{3.17}$$

Note that the expression (3.17) itself cannot be used to define the function $H$. For general sets $C_\omega$ we cannot decide whether $\{\omega \,|\, x \in C_\omega\} \in \mathcal{F}$. We are able to show or state it with help of the functions $g_{\omega,i}$ specifying $C_\omega$, and definig then $H$ by (3.16).

**Example**: The program

$$\begin{array}{lrcl} \min & f(x) & & \\ \text{s.t.} & H(x) & \geq & 0.9, \\ & x & \geq & 0, \end{array}$$

minimizes the objective function on a set of nonnegative decisions, for which the probability of being feasible is greater or equal 0.9. ⌟

The construction we have described is usually presented in a less clear and more intuitive way:

$$\begin{aligned} H(x) &= P\Big( \{\omega \,|\, G_{x,i}(\omega) \leq 0 \text{ for } i \in \{1,\dots,m\}\} \Big) = \\ &= P\Big( \{G_{x,i} \leq 0 \text{ for } i \in \{1,\dots,m\}\} \Big) = \\ &= P\Big( \{g_{\omega,i}(x) \leq 0 \text{ for } i \in \{1,\dots,m\}\} \Big), \end{aligned}$$

and for example the equation $H(x) \geq 0.9$ is written as

$$P\begin{pmatrix} g_1(\omega,x) & \leq 0 \\ g_2(\omega,x) & \leq 0 \\ \vdots & \\ g_m(\omega,x) & \leq 0 \end{pmatrix} \geq 0.9.$$

This notation looks like a recipe saying "just add probability". We will summarize our results and join them with the ideas from the previous section.

Given programs $(C_\omega, f_\omega)$, with $C_\omega \subset \mathbb{R}^n$, we use the extensions to define an objective function $f$ over $\mathbb{R}^n$. If the sets $C_\omega$ are given by $m$ constraints $g_{\omega,i} \leq 0$ as in (3.13) and the functions $G_{x,i}$ given by (3.14) are measurable, we can use the function $H$ defined in (3.16) to additionally restrict the domain of $f$ to a reasonable subset $C$ of $\mathbb{R}^n$. The program $(C, f)$ then reads as

$$\begin{array}{lrcl} \min & f(x) & & \\ \text{s.t.} & H(x) & \geq & \alpha, \end{array}$$

where $\alpha \in [0,1]$, and is called the program with ***joint probabilistic constraints***.

If we use the functions $H_i$ defined as in (3.15) instead of $H$, we obtain the program

$$\begin{array}{lrcl} \min & f(x) & & \\ \text{s.t.} & H_1(x) & \geq & \alpha_1, \\ & H_2(x) & \geq & \alpha_2, \\ & \vdots & & \\ & H_m(x) & \geq & \alpha_m, \end{array}$$

which is said to be with ***separate probabilistic constraints***. Obviously, we can produce more general settings by coupling certain subsets of the constraints together.

## 3.9 Two-stage models revisited

One of the possible ways of obtaining a two-stage model is this: We have originally a deterministic mathematical program $(C, f)$ with $C \subset \mathbb{R}^n$. Then we claim that some of its data are random, and we in fact make a transition to a collection of models $(C_\omega, f_\omega)$ for $\omega \in \Omega$. We then observe that the sets $C_\omega$ vary with $\omega$ and therefore there are decisions $x$ that are feasible for some $\omega$ while infeasible for others. To cope with this, we follow the procedure discussed in Section 3.7 about feasibility. We redefine the objective functions $f_\omega$, which we assume to be defined over $\mathbb{R}^n$ in a special way.

The value of the redefined function will be obtained through a solution of another mathematical program. To every $x \in \mathbb{R}^n$ and $\omega \in \Omega$ we assign a program $(D_{x,\omega}, g_{x,\omega})$ and define the extension $\overline{f}_\omega$ as

$$\overline{f}_\omega(x) = f_\omega(x) + Q(x, \omega),$$

where $Q(x, \omega)$ is the optimal value of $(D_{x,\omega}, g_{x,\omega})$. Further, we may apply the here-and-now approach and solve the program

$$\begin{array}{ll} \min & \mathbb{E}(\overline{f}_\omega(x)) \\ \text{s.t.} & x \in X \end{array} \tag{3.18}$$

where the set $X$ can be specified by probability constraints, by nonnegativity constraints or simply $X = \mathbb{R}^n$, the domain of the extended functions. For the correct explanation of the shortcut notation of (3.18) and for the integrability assumptions, please refer to Section 3.5 about here-and-now formulations.

We will illustrate the sketched approach. Suppose the original problem is a linear one:

$$\begin{array}{lrcl} \min & c^{\mathrm{T}}x & & \\ \text{s.t.} & Bx & = & r, \\ & x & \geq & 0. \end{array}$$

Thus we have $(C, f)$ with $C = \{x \in \mathbb{R}^n \,|\, Bx = r, \;\; x \geq 0\}$ and $f : \mathbb{R}^n \to \mathbb{R}$ given by $f(x) = c^{\mathrm{T}}x$.

Now suppose that some components of $B$ and $r$ are stochastic. We split the equations $Bx = r$ into the deterministic ones and the stochastic ones obtaining:

$$\begin{array}{lrcl} \min & c^{\mathrm{T}}x & & \\ \text{s.t.} & Ax & = & b, \\ & T(\omega)x & = & h(\omega), \\ & x & \geq & 0. \end{array}$$

Thus we have $(C_\omega, f_\omega)$ with $C_\omega = \{x \in \mathbb{R}^n \,|\, Ax = b,\; T(\omega)x = h(\omega),\; x \geq 0\}$ and $f_\omega = f$ for every $\omega \in \Omega$. The extensions $\overline{f}_\omega$ of the objective functions $f_\omega$ will be given by the program

$$\min \left\{ q^{\mathrm{T}}(\omega)y \,|\, W(\omega)y = h(\omega) - T(\omega)x, \;\; y \geq 0 \right\} \tag{3.19}$$

as

$$\overline{f}_\omega(x) = c^{\mathrm{T}}x + Q(x, \omega),$$

where $Q(x,\omega)$ is its optimum (or $\pm\infty$ in infeasible / unbounded case) and we have

$$\mathbb{E}(\overline{f}_{\omega}(x)) = c^{\mathrm{T}}x + \mathbb{E}(Q(x,\omega)).$$

If we denote $\mathbb{E}(Q(x,\omega))$ by $Q(x)$, the final two-stage program $(X, f)$ reads as

$$\begin{array}{lrcl} \min & c^{\mathrm{T}}x & + & Q(x) \\ \text{s.t.} & Ax & = & b \\ & x & \geq & 0, \end{array} \tag{3.20}$$

where the set $\{x \in \mathbb{R}^m \mid Ax = b,\ x \geq 0\}$ corresponds to $X$ in (3.18).
We have obtained the nested form of two-stage stochastic program (see 3.12).

We also see that the constraint $W(\omega)y = h(\omega) - T(\omega)x$ in (3.19) measures the violation of the original constraint $T(\omega)x = h(\omega)$. A recourse action $y$ compensates the violation by $W(\omega)y$ at a second-stage cost $q^{\mathrm{T}}(\omega)y$. The matrix $W(\omega)$ is usually called the recourse matrix.

## 3.10 Random variables and elements of probability space

The aim of this section is to discuss the randomness of the data of the models and the notation, which is found in several textbooks of stochastic programming (see [4], [7], [9]). This notation, involving expressions like $W(\xi)$, $W(\omega)$, $W(\xi(\omega))$ and $W(\xi_s)$ may be well clear to an expert in stochastic programming (or in statistics) but at the same time it may be confusing for a beginner in this area. That is why we decided to give some remarks on the topic.

Notation: Let $(\Omega_1, \mathcal{F}_1, P_1)$ be a probability space and let $(\Omega_2, \mathcal{F}_2, \mu)$ be an arbitrary measurable spaces. We call a measurable mapping $f$ from $\Omega_1$ to $\Omega_2$ a random variable. Particularly, we call random vectors random variables, too. To stress the measurability property of $f$, we say that the mapping is $\mathcal{F}_1 - \mathcal{F}_2$ measurable (meaning that $f^{-1}(\mathcal{F}_2) \subset \mathcal{F}_1$).

We will be illustrating our discussion on the second-stage program for the two-stage model (see 3.19).

### 3.10.1 Data as mappings from $\Omega$

In the two-stage problem, the second-stage data $q, W, T, h$ that define the underlying set $D_{x,\omega}$ and the objective function $f_{x,\omega}$ vary with $\omega \in \Omega$. Thus we have

$$\begin{array}{rl} q: & \Omega \to \mathbb{R}^k, \\ T: & \Omega \to \mathbb{R}^{m\times n}, \\ W: & \Omega \to \mathbb{R}^{m\times k}, \\ h: & \Omega \to \mathbb{R}^m. \end{array}$$

Very generally, there is no need to claim that the mappings $q, T, W, h$ are random variables, since the only mapping that is needed to be measurable is the mapping $Q_x : \Omega \to \overline{\mathbb{R}}$

given by $Q_x(\omega) = Q(x, \omega)$ for every fixed $x$. It can be proved however, that the measurability of $q, T, W, h$ is sufficient for $Q_x$ to be measurable. So in the sequel we will assume $q, T, W, h$ to be measurable.

The main direction of attack in showing the measurability of $Q_x$ is by the following Theorem 3, for which we need some preliminary considerations. Consider the linear program:

$$\begin{array}{lrcl} \min & c^{\mathrm{T}}x & & \\ \text{s.t.} & Ax & = & b, \\ & x & \geq & 0, \end{array} \tag{3.21}$$

where $A$ is a real $m \times n$ matrix and $c$ and $b$ are real vectors of dimensions $n$ and $m$, respectively. The program may have optimal solution, be unbounded or infeasible depending on its data $(A, b, c) \in \mathbb{R}^{m \times n + m + n}$.

We define the mapping $\gamma : \mathbb{R}^{m \times n + m + n} \to \overline{\mathbb{R}}$ as follows. For $(A, b, c) \in \mathbb{R}^{m \times n + m + n}$ for which the objective of (3.21) is unbounded, we set $\gamma(A, b, c) = -\infty$. For $(A, b, c)$ for which (3.21) is infeasible we set $\gamma(A, b, c) = +\infty$. Otherwise we let $\gamma(A, b, c)$ denote the optimal value of (3.21).

**Theorem 3** ***(Measurability of $\gamma$)***: The mapping $\gamma$ (as defined above) is a Borel measurable extended real-valued function.

Remark: We have the codomain $\overline{\mathbb{R}}$ of $\gamma$ equipped with the extended Borel $\sigma$-algebra $\overline{\mathbb{B}}$ defined as

$$\overline{\mathbb{B}} = \mathbb{B} \cup \{a \cup \{+\infty\} \,|\, a \in \mathbb{B}\} \cup \{a \cup \{-\infty\} \,|\, a \in \mathbb{B}\} \cup \{a \cup \{+\infty, -\infty\} \,|\, a \in \mathbb{B}\},$$

where $\mathbb{B}$ is the usual Borel $\sigma$-algebra on $\mathbb{R}$. For details, see Bauer [1]. The proof of this theorem can be found in Kall [6].

Now, we will see how this implies the measurability of the mapping $Q_x$. For a fixed $x$ and $\omega$, the second stage program reads as

$$\min \left\{ q(\omega)^T x \,|\, W(\omega) y = h(\omega) - T(\omega) x, y \geq 0 \right\}. \tag{3.22}$$

Provided that the mappings $q, W, T, h$ are measurable, the mapping $\xi : \Omega \to \mathbb{R}^{m \times k + m + k}$ obtained by "piecing these mappings together according to (3.22)":

$$\xi(\omega) = (W(\omega), h(\omega) - T(\omega)x, q(\omega))$$

is $\mathcal{F} - \mathbb{B}^{m \times k + m + k}$ measurable.

To see this, recall that a vector mapping is measurable if an only if its components (obtained through its composition with projections) are measurable. The measurability of $W$ and $q$ was supposed, and $h - Tx$ is measurable as a linear combination of measurable functions (recall that $x$ is fixed).

We have designed the mapping $\xi$ in such a way that we can obtain $Q_x$ as a composition of $\xi$ and $\gamma$ from Theorem 3: $Q_x = \gamma \circ \xi$. As a composition of two measurable mappings, $Q_x$ is $(\mathcal{F} - \overline{\mathbb{R}})$ measurable, which is its desired property.

### 3.10.2 Data as elements of real space

By real space we mean $\mathbb{R}^n$ for some $n \in \mathbb{N}$. In order to simplify further notation, we set $p = m \times k + m \times n + m + k$, so that we can abbreviate $\mathbb{R}^{m\times k+m\times n+m+k}$ by $\mathbb{R}^p$.

Let's define the mapping $\xi : \Omega \to \mathbb{R}^p$ by

$$\xi(\omega) = (W(\omega), T(\omega), h(\omega), q(\omega))$$

By the measurability assumption on the data, the mapping $\xi$ is measurable, and therefore it induces a probability measure $P_I$ on $\mathbb{R}^p$ given by

$$P_I(B) = P\Big( \{\omega \,|\, \xi(\omega) \in B\} \Big)$$

for every set $B \in \mathbb{B}^p$. The index $I$ reminds, that $P_I$ is sometimes called an image measure (see Bauer [1]).

We can discard the original probability space $(\Omega, \mathcal{F}, P)$ and make a transition to the new one $(\mathbb{R}^p, \mathbb{B}^p, P_I)$.

In this setting, the data $q, W, T, h$ are no longer random variables. In fact, we have a mathematical program $(C_z, f_z)$ assigned for every $z \in \mathbb{R}^p$. The program

$$\min \big\{ q^T x \,|\, Wy = h - Tx, y \geq 0 \big\}$$

is assigned to the vector $z = (q, W, T, h)$. So $q, W, T$ and $h$ are just variables (vector and matrix variables).

The major difference lies in this observation: Consider $\Omega$ to be finite. Then we have a finite number of programs $(C_\omega, f_\omega)$ with fixed $(q(\omega), W(\omega), T(\omega), h(\omega)) \in \mathbb{R}^p$ for each of them. After the transition, we have a program for every possible (fixed) $(q, W, T, h) \in \mathbb{R}^p$.

If we allow ourselves to write $\omega$ instead of $z$, we see that we arrive back at our previous notation. Some authors use indexing by $\xi$ instead. In here it should be noted that having discarded the original probability space, $\xi$ cannot denote a mapping any more. Instead, we have that $\xi$ denotes a variable in $\mathbb{R}^p$ and $\xi = (q, W, T, h)$. This is not an equality of fixed vectors, it is merely a statement of the notation convention (all the symbols included are variables).

### 3.10.3 Data as functions of real space

Another alternative is to keep the previous setting and introduce projection mappings.

**Example**: Let $t = (x_1, x_2, x_3) \in \mathbb{R}^3$ and define $z : \mathbb{R}^3 \to \mathbb{R}$ by $z(t) = x_3$. ⌟

In the sense of the previous example, we can have $q, W, T, h$ as appropriate projection mappings from $\mathbb{R}^p$ to $\mathbb{R}^k$, $\mathbb{R}^{m\times k}$, $\mathbb{R}^{m\times n}$ and $\mathbb{R}^m$, respectively.

Since $(\mathbb{R}^p, \mathbb{B}^p, P_I)$ is our new probability space and projection mappings are continuous, $q, W, T$ and $h$ are random variables. Then the notation again reads as

$$\min \left\{ q(\omega)^T x \,|\, W(\omega)y = h(\omega) - T(\omega)x, y \geq 0 \right\}$$

or

$$\min \left\{ q(\xi)^T x \,|\, W(\xi)y = h(\xi) - T(\xi)x, y \geq 0 \right\},$$

depending on our choice of the symbol ($\omega$ or $\xi$) to denote the member of $\Omega = \mathbb{R}^p$. (Especially note that still in here $\xi$ is not a random variable.)

### 3.10.4 The composed case

If we wish so, we can keep the original space $(\Omega, \mathcal{F}, P)$, have $\xi$ as a random variable $\xi : \Omega \to \mathbb{R}^p$ and $q, W, T, h$ as the appropriate projection mappings. The notation then reads as:

$$\min \left\{ q(\xi(\omega))^T x \,|\, W(\xi(\omega))y = h(\xi(\omega)) - T(\xi(\omega))x, y \geq 0 \right\} \tag{3.23}$$

and is meaningful.

### 3.10.5 The general case

It is also possible to keep the previous setting, but instead of having $q, W, T, h$ as projection mappings, to have them as some general (though concrete) mappings.

**Example**: Frequently (Birge and Louveaux [4], Kall and Wallace [7]) $h$ is required to be a linear mapping from $\mathbb{R}^r$ to $\mathbb{R}^k$ defined as

$$h = h_0 + \sum_{i=1}^{r} \alpha_i h_i,$$

where $h_i \in \mathbb{R}^k$ for $i \in \{0, \ldots, r\}$ and $(\alpha_i)_{i=1}^r \in \mathbb{R}^r$ is the vector variable of the domain of $h$. ⌟

This resembles a situation in parametric programming, where some of the program data are treated as linear functions of a parameter space. Note that in the context of parametric programming, the definition of linear mapping allows for an additive constant (as is the situation with linear functions in real analysis).

We see that for this approach, it is no longer necessary to keep the codomain of $\xi$ as $\mathbb{R}^p$. Instead, we can just have $\xi : \Omega \to \mathbb{R}^r$ for a suitable $r$. And $q, W, T, h$ are (possibly linear) mappings from $\mathbb{R}^r$ to the corresponding codomains. The notation reads again as in (3.23).

If we now discard the original $\Omega$ as discussed before, the notation reads as

$$\min \left\{ q(\xi)^T x \,|\, W(\xi)y = h(\xi) - T(\xi)x, y \geq 0 \right\},$$

where $\xi$ is not a mapping but a variable of the parameter space $\mathbb{R}^r$.

### 3.10.6 Expected value reformulation

One of the reasons of discussing these concepts lies in the use of the so called Expected value reformulation. Roughly speaking, one of the intuitive approaches to modelling under uncertainty is to substitute the random data by their expectations.

In our very general case, $\Omega$ is some general set and we have a family of programs $(C_\omega, f_\omega)$ assigned to every $\omega \in \Omega$. We don't use any random variables and therefore we cannot apply this intuitive approach. Its analogy (loosely speaking) would be to solve the program corresponding to $\omega$ with the biggest probability in the discrete case (this looks more like a "mode" reformulation).

If we however use the concept of random variables in the programs:

$$\min \left\{ q(\xi(\omega))^T x \,|\, W(\xi(\omega))y = h(\xi(\omega)) - T(\xi(\omega))x, y \geq 0 \right\}, \tag{3.24}$$

we can solve the program

$$\min \left\{ q(\mathbb{E}(\xi))^T x \,|\, W(\mathbb{E}(\xi))y = h(\mathbb{E}(\xi)) - T(\mathbb{E}(\xi))x, y \geq 0 \right\} \tag{3.25}$$

instead, since it is well defined. Note as an interesting observation, that this program possibly wasn't originally assigned to any of the events $\omega \in \Omega$ (it is particularly easy to see for finite $\Omega$).

In case when we discard the original $\Omega$, the notation reads as

$$\min \left\{ q(\xi)^T x \,|\, W(\xi)y = h(\xi) - T(\xi)x, y \geq 0 \right\} \tag{3.26}$$

and the expected value program is written as

$$\min \left\{ q(\mathbb{E}_\xi(\xi))^T x \,|\, W(\mathbb{E}_\xi(\xi))y = h(\mathbb{E}_\xi(\xi)) - T(\mathbb{E}_\xi(\xi))x, y \geq 0 \right\}. \tag{3.27}$$

Here $\mathbb{E}_\xi$ denotes the expected value as a functional on the new probability space, that is with respect to the image measure $P_I$.

It is interesting to compare the two cases (3.25) and (3.27). In the first one, $\xi$ stands for a given random variable $\xi : \mathcal{F} \to \mathbb{R}^r$. In the second one, it is just a symbol - a variable. For $\xi$ to be possible to figure in the domain of $\mathbb{E}_\xi$, it must however denote a random variable. In this case, the random variable is the identity mapping $id : \mathbb{R}^r \to \mathbb{R}^r$. (Compare to the same situation in real analysis where $x$ is a variable and at the same time denotes an identity function on $\mathbb{R}$.)

So the expected value program would be better stated as

$$\min \left\{ q(\mathbb{E}_\xi(id))^T x \,|\, W(\mathbb{E}_\xi(id))y = h(\mathbb{E}_\xi(id)) - T(\mathbb{E}_\xi(id))x, y \geq 0 \right\} \tag{3.28}$$

to avoid confusion. It would be also incorrect to view $\xi$ as the original random variable from the discarded probability space, since then we would have to use $\mathbb{E}$ instead of $\mathbb{E}_\xi$, obtaining the case (3.25).

### 3.10.7 Summary

We have seen many alternatives of correct treatment of the subject. In textbooks, all of them are generally used next to each other, leaving the reader to choose or recognize the proper or suitable one. From what has been showed, it follows that the situation is quite clear in its nature. In spite of this, the diversity may present and frequently presents a source of confusion for readers who are not accustomed to the field. For this reason we have presented the above discussion.

## 3.11 Further properties of two-stage models

At the end of our presentation of stochastic programming, we will give a brief informative overview of further properties of linear two-stage programs.

The two-stage program as stated in its nested form (3.12) minimizes the sum of the first stage objective function $c^T x$ and the recourse function $Q$, which is given by expectation as $Q(x) = \mathbb{E}(Q(x,\omega))$ for $x$ fixed. $Q(x,\omega)$ can be elegantly defined as

$$Q(x,\omega) = \inf\left\{q(\omega)^T x \,|\, W(\omega)y = h(\omega) - T(\omega)x, y \geq 0\right\}. \tag{3.29}$$

The infimum always exists ($\inf \emptyset = \infty$), and we possibly have $Q(x,\omega) = \pm\infty$. The set of $x$ for which $Q(x)$ is well defined and the properties of $Q$ are of main interest.

First, the function $Q_x$, that maps $\omega$ to $Q(x,\omega)$ with $x$ fixed, must be measurable. We have shown this in Section 3.10.1, under the assumption of measurability of $q, W, T$ and $h$. Second, the integral $\mathbb{E}(Q_x)$ must exist and the case when $\mathbb{E}(Q_x) < \infty$ is of interest. A necessary condition for $\mathbb{E}(Q_x) < \infty$ to hold is

$$P\Big(\{\omega \,|\, Q(x,\omega) < \infty\}\Big) = 1. \tag{3.30}$$

In case of fixed recourse, when $W$ is constant with respect to $\omega$, Kall [6] shows, that the square integrability of $q, T$ and $h$ implies $\mathbb{E}(Q_x) < \infty$ on the set $K$ of those $x$, for which (3.30) holds. It can be shown that $K$ is a convex set, and $Q$ is a convex function on $K$. Under further assumptions on the probability measure $P$ and the integrability of $Q_x$, the Lipschitz continuity and differentiability of $Q$ on $K$ can be proved.

This means that, in principle, numerical algorithms of nonlinear optimization can be used to solve the two-stage program. The evaluation of $Q$ and its gradient is however computationally too demanding in practice. Decomposition algorithms for discrete distributions are being used instead (see [2],[4]).

The requirement $Q(x,\omega) < \infty$ is equivalent to

$$\{y \in \mathbb{R}^m \,|\, W(\omega)y = h(\omega) - T(\omega)x, y \geq 0\} \neq \emptyset,$$

which holds if and only if $h(\omega) - T(\omega)x$ belongs to pos$W$, the positive hull of the columns of the recourse matrix $W$. This is surely guaranteed if $W = [I, -I]$ (with $I$ denoting the identity matrix), in which case we speak about complete simple recourse. Generally, it is guaranteed if pos$W = \mathbb{R}^k$ (with $k$ as the dimension of the column space of $W$). Then we speak about complete recourse.

# Chapter 4

# Stochastic model of aggregate blending

In the development of stochastic programming models for the blending problem, we start with the deterministic models of Chapter 2. In order to design a suitable stochastic model, we need to introduce the randomness, formulate the objective and choose one of the approaches discussed in Chapter 3.

## 4.1 The randomness and the objective

We begin with analysis of the randomness in the problem. There are $m$ ingredients (aggregates), each of them available in a sufficient amount. Their grading curves form the matrix $\mathbf{A}$. We select the mixing ratios $\mathbf{x} = (x_j)_{j=1}^m$ and we extract $x_j$ units of the $j$-th ingredient out of its available amount. Then we perfectly mix the extracted amounts and measure the grading curve of the obtained mixture. Every time we repeat this procedure, we get a slightly different result. Suppressing the factor of the measurement, we see that the main reason is the non-homogeneity of the ingredient combined with the extraction.

The grading curve of the mixture that we expect to obtain is $\mathbf{Ax}$. Because of the non-homogeneity, the grading curves of the extracted amounts differ from $\mathbf{A}$. We can try to model them as random variables, denoting them by $\mathbf{A}(\omega)$. The resulting grading curve of the mixture is then $\mathbf{A}(\omega)\mathbf{x}$, depending on $\omega$. As a reasonable probability space, we can choose $\Omega = \mathbb{R}^{n\times m}$, equipped with the Borel $\sigma$-algebra $\mathbb{B}^{n\times m}$ and some probability measure $P$. We can fix the random variable $\mathbf{A}$ to be the identity mapping on $\mathbb{R}^{n\times m}$. The probability measure $P$ then represents also the distribution of $\mathbf{A}$, and it is the only part of the model that is left to be specified. We will discuss it briefly in section (4.7).

When choosing the appropriate stochastic model, we note that the decision on $\mathbf{x}$ is made before performing the mix. It is a here and now decision. Thus we choose a here-and-now or a two-stage model. As we shall see, the choice of two-stage models is natural in our case.

The objective is, analogously to the deterministic case, based on the distance of $\mathbf{A}(\omega)\mathbf{x}$ from $\mathbf{g}$ determined as $||\mathbf{A}(\omega)\mathbf{x} - \mathbf{g}||_p$ with $p = 1$ or $p = \infty$.

## 4.2 Here-and-now models

In the deterministic case, our basic models (before finding their linear equivalents and introducing the constraints for the lower and the upper bound) were

$$\begin{aligned} \min \quad & ||\mathbf{A}\mathbf{x} - \mathbf{g}||_p \\ \text{s.t.} \quad & \mathbf{1}^{\mathrm{T}}\mathbf{x} = 1, \\ & \mathbf{x} \geq \mathbf{0}. \end{aligned} \tag{4.1}$$

Compare this model to (2.1) for $p = 1$ and to (2.5) for $p = \infty$.

After introducing the randomness to the model by setting $\mathbf{A} = \mathbf{A}(\omega)$ in (4.1), we get a family of programs $(X, f_\omega)$ (for the notation refer to Chapter 3), where the set $X$ is given as

$$X = \left\{\mathbf{x} \in \mathbb{R}^m \,|\, \mathbf{1}^{\mathrm{T}}\mathbf{x} = 1, \mathbf{x} \geq \mathbf{0}\right\}$$

and the objective function $f_\omega : X \to \mathbb{R}$ is defined as

$$f_\omega = ||\mathbf{A}(\omega)\mathbf{x} - \mathbf{g}||_p. \tag{4.2}$$

We are in position to apply the expected objective here-and-now model as introduced in Section (3.5).

**The here-and-now blending model**:

$$\begin{aligned} \min \quad & \mathbb{E}\Big(||\mathbf{A}(\omega)\mathbf{x} - \mathbf{g}||_p\Big) \\ \text{s.t.} \quad & \mathbf{1}^{\mathrm{T}}\mathbf{x} = 1, \\ & \mathbf{x} \geq \mathbf{0}. \end{aligned} \tag{4.3}$$

The model optimizes the expected quality of the mixture. When adding the constraints for the lower and the upper bound into (4.1), the situation changes. The model reads as

$$\begin{aligned} \min \quad & ||\mathbf{A}\mathbf{x} - \mathbf{g}||_p \\ \text{s.t.} \quad & \mathbf{A}_l(\omega)\mathbf{x} \geq \mathbf{l}, \\ & \mathbf{A}_u(\omega)\mathbf{x} \leq \mathbf{u}, \\ & \mathbf{1}^{\mathrm{T}}\mathbf{x} = 1, \\ & \mathbf{x} \geq \mathbf{0}. \end{aligned} \tag{4.4}$$

When setting $\mathbf{A} = \mathbf{A}(\omega)$, $\mathbf{A}_l = \mathbf{A}_l(\omega)$ and $\mathbf{A}_u = \mathbf{A}_u(\omega)$ in (4.4), we get a family of programs $(C_\omega, f_\omega)$ with

$$C_\omega = \left\{\mathbf{x} \in \mathbb{R}^m \,|\, \mathbf{1}^{\mathrm{T}}\mathbf{x} = 1,\, \mathbf{A}_u(\omega)\mathbf{x} \leq \mathbf{u},\, \mathbf{A}_l(\omega)\mathbf{x} \geq \mathbf{l},\, \mathbf{x} \geq \mathbf{0}\right\}$$

and with $f_\omega$ as in (4.2). Recall that the matrices $\mathbf{A}_u(\omega)$ and $\mathbf{A}_l(\omega)$ represent a selection of certain rows of the matrix $\mathbf{A}(\omega)$ (the same selection for each $\omega$).

We cannot apply the here-and-now model directly because the sets $C_\omega$ vary with $\omega$. The solution is to apply a two-stage model or a model with probabilistic constraints.

## 4.3 Two-stage model 1

The easiest way of formulating a two-stage model is to start with the deterministic program (2.4). As was shown in Section 2.6, it is a linear program equivalent to the program (4.1) with $p = 1$. We restate it at this point with aim to provide its direct comparison with the two-stage program build on its base.

**The original model**:

$$\begin{aligned} \min \quad & \mathbf{1}^{\mathrm{T}}\mathbf{y}^{+} + \mathbf{1}^{\mathrm{T}}\mathbf{y}^{-} && (4.5)\\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} + \mathbf{y}^{+} - \mathbf{y}^{-} = \mathbf{g}, \\ & \mathbf{A}_l\mathbf{x} - \mathbf{y}_l = \mathbf{l}, \\ & \mathbf{A}_u\mathbf{x} + \mathbf{y}_u = \mathbf{u}, \\ & \mathbf{1}^{\mathrm{T}}\mathbf{x} = 1, \\ & \mathbf{x},\, \mathbf{y}^{+},\, \mathbf{y}^{-},\, \mathbf{y}_l,\, \mathbf{y}_u \geq \mathbf{0}. \end{aligned}$$

**Two-stage model 1**:

$$\begin{aligned} \min \ & \mathbb{E}\left(Q(\mathbf{x}, \omega)\right) && (4.6)\\ \text{s.t.} \quad & \mathbf{1}^{\mathrm{T}}\mathbf{x} = 1, \\ & \mathbf{x} \geq \mathbf{0}, \end{aligned}$$

$$\begin{aligned} Q(\mathbf{x}, \omega) = \min \ & \mathbf{1}^{\mathrm{T}}\mathbf{y}^{+} + \mathbf{1}^{\mathrm{T}}\mathbf{y}^{-} && (4.7)\\ \text{s.t.} \quad & \mathbf{A}(\omega)\mathbf{x} + \mathbf{y}^{+} - \mathbf{y}^{-} = \mathbf{g}, \\ & \mathbf{A}_l(\omega)\mathbf{x} - \mathbf{y}_l = \mathbf{l}, \\ & \mathbf{A}_u(\omega)\mathbf{x} + \mathbf{y}_u = \mathbf{u}, \\ & \mathbf{y}^{+},\, \mathbf{y}^{-},\, \mathbf{y}_l,\, \mathbf{y}_u \geq \mathbf{0}. \end{aligned}$$

Note than in every second stage program (4.7), $\mathbf{x}$ and $\omega$ are fixed. When we relate (4.7) to the general form of second-stage program

$$\min \left\{ \mathbf{q}^{\mathrm{T}}(\omega)\mathbf{y} \,|\, \mathbf{W}(\omega)\mathbf{y} + \mathbf{T}(\omega)\mathbf{x} = \mathbf{h}(\omega), \quad \mathbf{y} \geq \mathbf{0} \right\}, \tag{4.8}$$

we see that

$$\mathbf{T}(\omega) = \begin{pmatrix} \mathbf{A}(\omega) \\ \mathbf{A}_l(\omega) \\ \mathbf{A}_u(\omega) \end{pmatrix}, \quad \mathbf{W}(\omega) = \begin{pmatrix} \mathbf{I} & -\mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & -\mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} \end{pmatrix}, \quad \mathbf{h}(\omega) = \begin{pmatrix} \mathbf{g} \\ \mathbf{l} \\ \mathbf{u} \end{pmatrix},$$

and

$$\mathbf{q}(\omega) = \begin{pmatrix} \mathbf{1} \\ \mathbf{1} \\ \mathbf{0} \\ \mathbf{0} \end{pmatrix}, \quad \mathbf{y} = \begin{pmatrix} \mathbf{y}^{+} \\ \mathbf{y}^{-} \\ \mathbf{y}^{l} \\ \mathbf{y}^{u} \end{pmatrix},$$

where $\mathbf{y}$ is the vector of second stage variables.

The original deterministic program (4.5) is in fact a "deterministic two-stage program" with $\mathbf{x}$ as first stage variables and $\mathbf{y}$ as the second stage variables. We can get it back

easily from the stochastic two-stage model (4.6) in case of a single-element $\Omega$. Since we have identified $\Omega$ with $\mathbb{R}^{m\times n}$, this situation rather corresponds to having $P$ as the Dirac $\delta$-distribution: $P(B) = \delta(\{\mathbf{y}\})\delta(B)$ for $B \in \mathbb{B}^{m\times n}$ ($\delta$ is the characteristic function of given set, $\mathbf{y} \in \mathbb{R}^{m\times n}$ fixed). This is clearly of no practical relevance. On the other hand, solving (4.6) in the EV reformulation as introduced in Section 3.10.6, we solve exactly the program (4.5) with expected values of the (matrix) random variables $\mathbf{A}, \mathbf{A}_l, \mathbf{A}_u$ at positions of the original deterministic data. If we look at the original deterministic data as point estimates (determined by averaging a finite number of measurements), we see that the EV reformulation and the deterministic program are very closely related.

Notice that the recourse matrix $\mathbf{W}(\omega)$ (constant with respect to $\omega$) of the second stage programs (4.7) doesn't guarantee their feasibility in general. We may have $\mathbf{h} - \mathbf{T}\mathbf{x} \notin \mathrm{pos}\mathbf{W}$. This is not a surprise. When analyzing the feasibility of the original deterministic program (4.5), we can see that for certain data $\mathbf{A}_l$, $\mathbf{A}_u$ and $\mathbf{l}$, $\mathbf{u}$, the program may not be feasible. Practically, it may be impossible to fit the mixture's grading curve between the lower and the upper bound with the ingredients available. In the deterministic case, this simply means that we have to choose different ingredients. For the stochastic program, some cases may be feasible while other may not.

Suppose we have a finite probability space. Then we can solve the two-stage model (4.6) in the extensive form (see Section 3.6). If one of the second-stage programs is not feasible, then the whole program is not feasible as well. So we see that the model as stated requires a solution which always respects the bounds (for a finite $\Omega$). We can however adopt a different attitude leading to a modification of (4.6).

## 4.4 Two-stage model 2

Suppose the production works as follows. When the producer realizes that the grading curve of the mixture violates the bounds, he performs a recourse action, modifying the mixture by adding other ingredients in order to force the grading curve between the bounds. This leads naturally to a stochastic multistage program and opens a possible direction of investigation. In our treatment, we will comfort ourselves with a simpler concept. Instead of incorporating the recourse action into the model, we observe that there is a penalty cost related to it. We will include only the penalty for the violation to the model. If no recourse action is available, the penalty is just the cost of the wasted ingredients.

We will see how the penalty concept relates to the original objective:

$$\begin{array}{ccc} \text{original constraint} & \text{relaxed constraint} & \text{penalty} \\ \mathbf{A}\mathbf{x} = \mathbf{g} & \mathbf{A}\mathbf{x} + \mathbf{y}^+ - \mathbf{y}^- = \mathbf{g} & \mathbf{1}^{\mathrm{T}}\mathbf{y}^+ + \mathbf{1}^{\mathrm{T}}\mathbf{y}^-, \\ \mathbf{A}_l\mathbf{x} \geq \mathbf{l} & \mathbf{A}_l\mathbf{x} + \mathbf{y}_l^+ - \mathbf{y}_l^- = \mathbf{l} & r\mathbf{1}^{\mathrm{T}}\mathbf{y}_l^+, \\ \mathbf{A}_u\mathbf{x} \leq \mathbf{u} & \mathbf{A}_u\mathbf{x} + \mathbf{y}_u^+ - \mathbf{y}_u^- = \mathbf{u} & r\mathbf{1}^{\mathrm{T}}\mathbf{y}_u^-. \end{array} \tag{4.9}$$

The original 1-norm objective may be viewed as a penalty objective for violation of the first constraint in (4.9). The penalties for the bound constraints are associated only with

the appropriate variables ($\mathbf{y}_u^-$, $\mathbf{y}_l^+$), which measure the distance in the direction of violation (up for the upper bound and down for the lower bound). The variables $\mathbf{y}_u^+$ and $\mathbf{y}_l^-$ are introduced in order to obtain a standard form of linear program. The penalty cost as stated is a linear function of the violation scaled by a factor $r > 0$ (other choices possible).

**Two-stage model 2**:

$$\begin{aligned} &\min\ \mathbb{E}\left(Q(\mathbf{x},\omega)\right) && (4.10)\\ &\text{s.t.}\ \ \mathbf{1}^{\mathrm{T}}\mathbf{x} = 1,\\ &\qquad\quad \mathbf{x} \geq \mathbf{0},\\ Q(\mathbf{x},\omega) = \min\ &\mathbf{1}^{\mathrm{T}}\mathbf{y}^+ + \mathbf{1}^{\mathrm{T}}\mathbf{y}^- + r\mathbf{1}^{\mathrm{T}}\mathbf{y}_l^+ + r\mathbf{1}^{\mathrm{T}}\mathbf{y}_u^- && (4.11)\\ \text{s.t.}\ \ &\mathbf{A}(\omega)\mathbf{x} + \mathbf{y}^+ - \mathbf{y}^- = \mathbf{g},\\ &\mathbf{A}_l(\omega)\mathbf{x} + \mathbf{y}_l^+ - \mathbf{y}_l^- = \mathbf{l},\\ &\mathbf{A}_u(\omega)\mathbf{x} + \mathbf{y}_u^+ - \mathbf{y}_u^- = \mathbf{u},\\ &\mathbf{y}^+, \mathbf{y}^-, \mathbf{y}_l^+, \mathbf{y}_l^-, \mathbf{y}_u^+, \mathbf{y}_u^- \geq \mathbf{0}. \end{aligned}$$

Looking at the recourse matrix $\mathbf{W}$ of this model, we see that

$$\mathbf{W} = \begin{pmatrix} \mathbf{I} & -\mathbf{I} \end{pmatrix}, \tag{4.12}$$

provided the vector $\mathbf{y}$ of second stage variables is ordered as

$$\mathbf{y}^T = \left(\mathbf{y}^{+\mathrm{T}}, \mathbf{y}_l^{+\mathrm{T}}, \mathbf{y}_u^{+\mathrm{T}}, \mathbf{y}^{-\mathrm{T}}, \mathbf{y}_l^{-\mathrm{T}}, \mathbf{y}_u^{-\mathrm{T}}\right)$$

and the matrix $\mathbf{I}$ ($-\mathbf{I}$) corresponds to the three first (last) components of $\mathbf{y}$. This means that the model is with *complete simple recourse* and the second stage program is always feasible.

## 4.5 Two-stage model 3

Another possibility of getting a two-stage program with simple recourse from (4.6) is to require the constraints $\mathbf{A}_l\mathbf{x} \geq \mathbf{l}$ and $\mathbf{A}_u\mathbf{x} \geq \mathbf{u}$ to be satisfied $P$ almost surely (with probability one). For discussion on probabilistic constraints refer to Section 3.8. We obtain the following model.

**Two-stage model 3**:

$$\begin{aligned} &\min\ \mathbb{E}\left(Q(\mathbf{x},\omega)\right)\\ &\text{s.t.}\ \ P\begin{pmatrix} \mathbf{A}_l(\omega)\mathbf{x} \geq \mathbf{l} \\ \mathbf{A}_u(\omega)\mathbf{x} \leq \mathbf{u} \end{pmatrix} = 1,\\ &\qquad\quad \mathbf{1}^{\mathrm{T}}\mathbf{x} = 1,\\ &\qquad\quad \mathbf{x} \geq \mathbf{0}, \end{aligned}$$

$$\begin{aligned} Q(\mathbf{x},\omega) = \ \min\ &\mathbf{1}^{\mathrm{T}}\mathbf{y}^+ + \mathbf{1}^{\mathrm{T}}\mathbf{y}^-\\ \text{s.t.}\ \ &\mathbf{A}(\omega)\mathbf{x} + \mathbf{y}^+ - \mathbf{y}^- = \mathbf{g},\\ &\mathbf{y}^+, \mathbf{y}^- \geq \mathbf{0}. \end{aligned}$$

The recourse variables are only $\mathbf{y}^+$ and $\mathbf{y}^-$ in this case.

## 4.6 Models with $\infty$-norm

The stochastic models we presented up to now used the 1-norm in the objective. We can develop similar models with the $\infty$-norm on the base of the deterministic program (2.7). The only difference compared to the previous models is the following. We substitute the constraint

$$\mathbf{A}(\omega)\mathbf{x} + \mathbf{y}^+ - \mathbf{y}^- = \mathbf{g}$$

by the two constraints

$$\begin{array}{lllll} \mathbf{A}(\omega)\mathbf{x} & +\mathbf{y}^+ & & -y\mathbf{1} & = \mathbf{g}, \\ \mathbf{A}(\omega)\mathbf{x} & & -\mathbf{y}^- & +y\mathbf{1} & = \mathbf{g}. \end{array}$$

In Two-stage model 1 (4.6), the recourse matrix $\mathbf{W}$ changes accordingly to

$$\mathbf{W} = \begin{pmatrix} \mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{0} & -\mathbf{1} \\ \mathbf{0} & -\mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{1} \\ \mathbf{0} & \mathbf{0} & -\mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} \end{pmatrix},$$

and in Two-stage model 2 (4.10) to

$$\mathbf{W} = \begin{pmatrix} \mathbf{I} & \mathbf{0} & -\mathbf{1} \\ \mathbf{0} & -\mathbf{I} & \mathbf{1} \end{pmatrix}.$$

Compared to (4.12), we see that we don't have complete simple recourse any more. It can be shown however, that the recourse is complete. The equation $\mathbf{I}\mathbf{y} - \mathbf{1}z = \mathbf{g}$ has a nonnegative solution $(\mathbf{y}, z)$ for arbitrary right-hand side $\mathbf{g} = (g_i)_{i=1}^k$. It is sufficient to set $z \geq |\min_i g_i|$. Then $\mathbf{y} = \mathbf{g} + \mathbf{1}z$ and $\mathbf{y} \geq \mathbf{0}$. The same holds for the other part of $\mathbf{W}$.

## 4.7 Distribution properties

The probability distribution of the stochastic entries of the models is an important feature of the problem. As was discussed before, the stochasticity is involved due to non-homogeneity of the ingredients and due to the nature of the measurement.

Basically there are three possibilities of assessment of the distribution. First, to perform a physical analysis. Determine the distribution by considerations on the physical and mechanical properties of the ingredients and on the mechanism of the measurement of grading curves. Second, to perform a simulation based on the above considerations. The third one is a practical approach. The producer performs repeated measurement on the ingredients to get an estimate of the distribution, which is then entered to the model. The range of types of ingredients that can be directly dealt with this approach is not limited, which might generally not be true for the approaches using physical analysis.

We proposed the following simulation. The grading curve of the mixture or of the ingredient is in fact a distribution function of the size of its particles (see Figure 2.1). Extracting some amount of the ingredient then can be modeled as a random sample from this distribution. We implemented a short code in Matlab (see Appendix A) and we present graphical results in Figure (4.1).

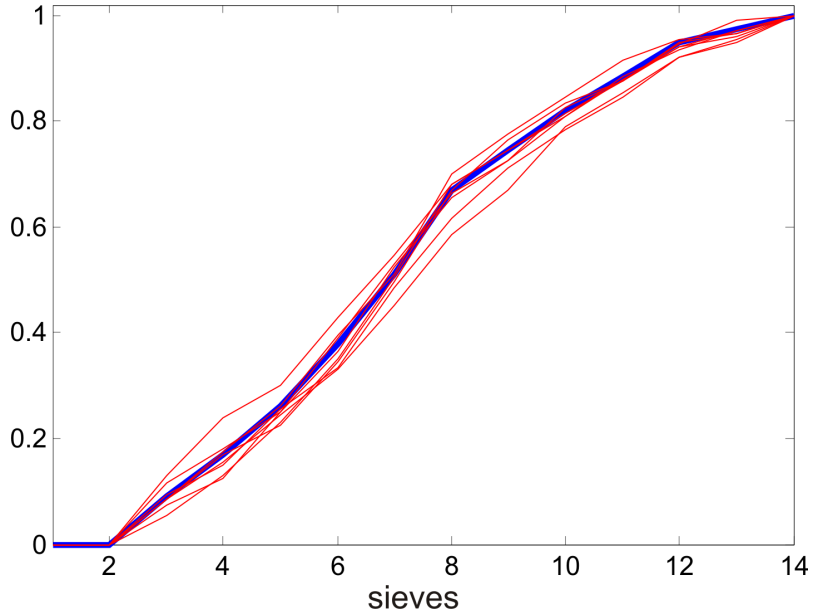



Figure 4.1: Simulation of grading curves.

The figure shows distribution of 8 samples (red curves) of size 200 from the distribution given by the blue distribution function. The simulation provides a useful tool for generation of reasonable random data for testing of the stochastic models.

# Chapter 5

# Implementation and results

## 5.1 Deterministic models

We have implemented the deterministic models (2.4) and (2.7). As a linear solver, we have implemented the Simplex algorithm in C++ using Microsoft Visual Studio 2005. The following remarks on the implementation are addressed to readers familiar with the Simplex method.

Since the simplex tables of our models are relatively small (at most 50 rows and 200 columns) and we couldn't take advantage of sparseness, we decided to perform the operations with the entire simplex tableau (in contrast to the usual revised simplex method - see [2]).

To maintain numerical accuracy, we periodically recompute the basis inverse using the LU decomposition. The main difference of our implementation compared to the usual "by hand" operations on the tableau consists in the way of computation of the vector of reduced costs. We compute it at each step as $\mathbf{c}_B^{\mathrm{T}}\mathbf{B}^{-1}\mathbf{N} - \mathbf{c}_N^{\mathrm{T}}$ from the original data $\mathbf{c}^{\mathrm{T}} = (\mathbf{c}_B^{\mathrm{T}}, \mathbf{c}_N^{\mathrm{T}})$ using the actual basis inverse $\mathbf{B}^{-1}$. This guarantees that the numerical accuracy is the accuracy of the revised simplex method, which computes the reduced costs in the same way. We also use row scaling of the orginal data to enhance the accuracy.

The implemented method is the two-phase primal simplex method. The situation when artificial variables remain in basis at zero level after the first stage is also resolved by the algorithm. For the choice of the pivot column, Dantzig's rule, Bland's (cycle preventing) rule and the Steepest edge rule are available. The Steepest edge rule speeds up the convergence and its use is conditioned by updating the entire simplex tableau at each step, which is our case.

## 5.2 Stochastic models

We have implemented the Two-stage model 2 (4.10) in the modeling language GAMS for both the 1-norm and the $\infty$-norm objective. Our implementation uses the extensive form of the stochastic program and is capable to deal with discrete distributions of the data

(or equivalently to solve the model in case of finite probability space).

The running time for a model with 1000 scenarios (card($\Omega$) = 1000) is about 2 minutes on Pentium CoreDuo 1.86 Ghz.

## 5.3 Results

We present representative numerical and graphical results. Table (5.1) shows results of mixture optimization with given ingredients. The grading curves of the ingredients are contained in the first four numerical rows. The ingredients themselves are denoted by their technical specifications (HDK16-22K8, ...) and the sieves by standard designations (S00-063, ...). The optimal solution for given bounds and given goal curve is presented. Moreover, we give a comparison to an expert solution, which was provided by civil engineers. The mixing ratios (both optimal and expert values) are further presented, together with the value of the optimization criterion (the min max criterion in this case). Due to engineering conventions, the values are given in percents, so that the range of the grading curves is the interval $[0, 100]$ instead of $[0, 1]$.

| | S00-063 | S00-125 | S00-25 | S00-5 | S01 | S02 | S04 | S08 | S11 |
|---|---|---|---|---|---|---|---|---|---|
| HDK16-22K8 | 1.00 | 1.20 | 1.40 | 1.50 | 1.60 | 1.60 | 1.60 | 1.60 | 1.60 |
| HDK8-16K7 | 0.70 | 0.80 | 0.80 | 0.90 | 0.90 | 0.90 | 0.90 | 1.30 | 9.80 |
| HTK4-8M6 | 1.30 | 1.70 | 2.30 | 2.50 | 2.60 | 2.90 | 8.40 | 87.90 | 100.00 |
| DTK0-4O11 | 1.80 | 2.60 | 9.10 | 34.90 | 61.20 | 76.50 | 87.60 | 98.50 | 100.00 |
| lower | - | - | 2.00 | 5.00 | 9.00 | 16.00 | 28.00 | 45.00 | - |
| upper | 2.00 | 6.00 | 16.00 | 28.00 | 40.00 | 53.00 | 65.00 | 78.00 | - |
| goal | 0.00 | 0.00 | 9.00 | 17.00 | 26.00 | 38.00 | 51.00 | 67.00 | - |
| optimal | 1.43 | 1.96 | 5.35 | 18.17 | 31.20 | 38.84 | 45.77 | 72.21 | 77.43 |
| expert | 1.35 | 1.83 | 4.95 | 16.88 | 29.01 | 36.10 | 42.03 | 59.05 | 63.43 |

| | S16 | S22 | S32 | S45 | S63 | optim ratios | expert ratios |
|---|---|---|---|---|---|---|---|
| HDK16-22K8 | 9.90 | 80.10 | 100.00 | 100.00 | 100.00 | 0.090 | 0.170 |
| HDK8-16K7 | 94.20 | 100.00 | 100.00 | 100.00 | 100.00 | 0.152 | 0.220 |
| HTK4-8M6 | 100.00 | 100.00 | 100.00 | 100.00 | 100.00 | 0.264 | 0.150 |
| DTK0-4O11 | 100.00 | 100.00 | 100.00 | 100.00 | 100.00 | 0.494 | 0.460 |
| lower | 73.00 | - | 90.00 | - | - | | |
| upper | 91.00 | - | - | - | - | | |
| goal | - | - | - | - | - | | |
| optimal | 91.00 | 98.21 | 100.00 | 100.00 | 100.00 | $f^* = 5.21$ | |
| expert | 83.41 | 96.62 | 100.00 | 100.00 | 100.00 | | $f_{exp} = 8.97$ |

Table 5.1: Numerical results for $\infty$-norm.

The optimal value (the minimum absolute distance from the goal) is $f^* = 5.21$ compared to the expert solution value $f_{exp} = 8.97$.

Table (5.2) shows the results obtained with the 1-norm criterion (only a part of the input data from Table (5.1) is repeated).

| | S00-063 | S00-125 | S00-25 | S00-5 | S01 | S02 | S04 | S08 | S11 |
|---|---|---|---|---|---|---|---|---|---|
| lower | - | - | 2.00 | 5.00 | 9.00 | 16.00 | 28.00 | 45.00 | - |
| upper | 2.00 | 6.00 | 16.00 | 28.00 | 40.00 | 53.00 | 65.00 | 78.00 | - |
| goal | 0.00 | 0.00 | 9.00 | 17.00 | 26.00 | 38.00 | 51.00 | 67.00 | - |
| optimal | 1.39 | 1.90 | 5.19 | 17.75 | 30.51 | 37.98 | 44.54 | 66.99 | 72.14 |
| expert | 1.35 | 1.83 | 4.95 | 16.88 | 29.01 | 36.10 | 42.03 | 59.05 | 63.43 |

| | S16 | S22 | S32 | S45 | S63 | criterion |
|---|---|---|---|---|---|---|
| lower | 73.00 | - | 90.00 | - | - | |
| upper | 91.00 | - | - | - | - | |
| goal | - | - | - | - | - | |
| optimal | 91.00 | 98.29 | 100.00 | 100.00 | 100.00 | $f^* = 18.82$ |
| expert | 83.41 | 96.62 | 100.00 | 100.00 | 100.00 | $f_{exp} = 29.19$ |

Table 5.2: Numerical results for 1-norm.

The vector of the optimal mixing ratios is $\mathbf{x}^* = (0.086, 0.215, 0.215, 0.484)$ in this case, the expert ratios are as in Table (5.1). The minimal sum of absolute differences is $f^* = 18.82$, whereas the value of this criterion in the expert case is $f_{exp} = 29.19$.

We present a graphical illustration of the data of Table (5.1):

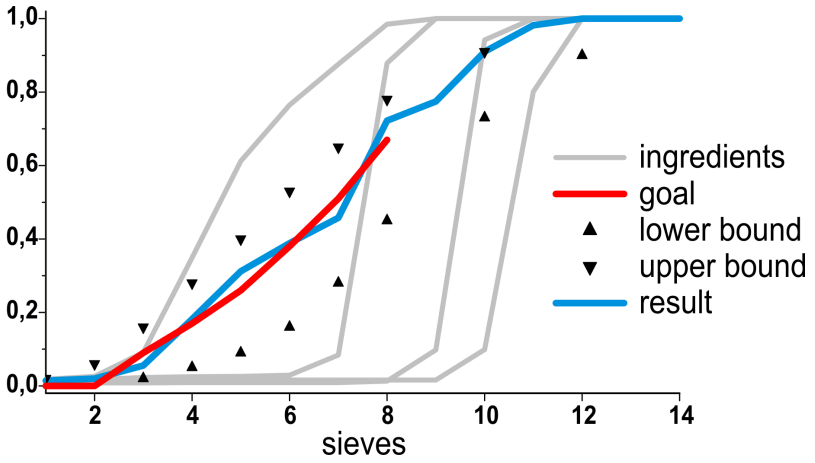



Figure 5.1: The result of the optimization.

The blue curve represents the grading curve of the optimal mixture, fitted to the requested grading curve (the red curve) and respecting the imposed bounds indicated by the black triangles. As mentioned in Chapter 2, the goal and the bounds are given on selected sieves only.

## 5.4 Expert modification

From the discussions with civil engineering professionals, it turned out that the notion of optimality is sometimes judged also from different points of view, reflecting the expert attitudes and the experience of the engineers. A desirable property of the mixture was formulated in terms of minimizing the variance of absolute differences between the requested grading curve and the grading curve of the mixture.

We have formulated a linear model that incorporates this requirement. We present its core part, the whole model formulation being an analogy of formulations in Chapter 4. The constrains concerning the distance from the goal and the bounds are

$$\begin{aligned} \mathbf{A}\mathbf{x} + \mathbf{y}^{+} - \mathbf{y}^{-} &= \mathbf{g}, \\ \mathbf{A}_l\mathbf{x} - \mathbf{y}_l &\geq \mathbf{l}, \\ \mathbf{A}_u\mathbf{x} + \mathbf{y}_u &\leq \mathbf{u}, \\ ny_a - \mathbf{1}^{\mathrm{T}}\mathbf{y}^{+} - \mathbf{1}^{\mathrm{T}}\mathbf{y}^{-} &= 0, \\ \mathbf{y}^{+} - \mathbf{1}y_a &= \mathbf{y}^{++} - \mathbf{y}^{+-}, \\ \mathbf{y}^{-} - \mathbf{1}y_a &= \mathbf{y}^{-+} - \mathbf{y}^{--}, \end{aligned}$$

where $n$ denotes the dimension of the second-stage decision vector $\mathbf{y}$, and the scalar variable $y_a$ represents the average absolute deviation. The variables $\mathbf{y}^{++}$, $\mathbf{y}^{+-}$, $\mathbf{y}^{-+}$, $\mathbf{y}^{--}$ then measure the absolute difference between the deviations $\mathbf{y}^{+}$, $\mathbf{y}^{-}$ and the average absolute deviation $y_a$.

The term to be added to the objective function for minimization is given by

$$\mathbf{1}^{\mathrm{T}}\left(\mathbf{y}^{++} + \mathbf{y}^{+-} + \mathbf{y}^{-+} + \mathbf{y}^{--}\right).$$

# Chapter 6

# Conclusion

We have dealt with stochastic programming as a suitable tool for mathematical modeling of problems involving uncertainty. In Chapter 3 we presented a theoretical development of stochastic programming. The usual approach of many textbooks (e.g. [4], [7], [9]) is to introduce the randomness to a deterministic program and then to seek for its reformulations. We presented a development that involves the probability space from the beginning. The leading idea was to simplify the initial considerations, abstracting from particular forms of the programs and treating them as sets and objective functions assigned to elements of probability space. We managed to obtain the usual concepts of stochastic programming with this approach.

We applied stochastic programming to a civil engineering problem, namely the optimization of aggregate blending. We first developed deterministic models of the blending problem in Chapter 2, then we discussed the stochastic features and we returned to formulation of suitable stochastic models in Chapter 4. We provided here-and-now and two-stage stochastic models with various objectives.

We have implemented the deterministic models in C++ together with our own linear solver. We have used the modeling language GAMS to implement the stochastic models and we also performed simulations in Matlab to get a reasonable random data for our models. We presented numerical and graphical results in Chapter 5.

The deterministic models with various objectives, corresponding to the expected value reformulations of the stochastic models, have been, together with our solver, incorporated into a specialized software. The software was developed by the company Computer MCL Brno, spol. s r.o., for the civil engineering enterprise Českomoravský beton, a.s. Our models are now being used for the optimization of concrete mixtures.

# Bibliography


[1] BAUER, H.: *Probability Theory and Elements of Measure Theory*, Second English Edition, Academic Press, A Subsidiary of Harcourt Brace Jovanovich, Publishers, 1981

[2] BAZARAA, M. S., JARVIS, J. J., SHERALI, H. D.: *Linear Programming and Network Flows*, John Wiley & Sons, Inc., 1990

[3] BAZARAA, M. S., SHERALI H. D., SHETTY, C. M.: *Nonlinear Programming - Theory and Algorithms*, Second Edition, John Wiley & Sons, Inc., New York, 1993

[4] BIRGE, J. R., LOUVEAUX, F.: *Introduction to Stochastic Programming*, Springer-Verlag New York, 1997

[5] CAPINSKY, M., KOPP, E.: *Measure, Integral and Probability*, Second Edition, Springer-Verlag London, 2004

[6] KALL, P.: *Stochastic Linear Programming*, Springer-Verlag Berlin Heidelberg New York 1976

[7] KALL, P., WALLACE, S. W.: *Stochastic Programming* , John Wiley & Sons, Chichester, 1994

[8] PLESNÍK, J., DUPAČOVÁ, J., VLACH, M.: *Lineárne Progarmovanie*, ALFA, Bratislava, 1990

[9] RUSZCZYNSKY, A., SHAPIRO, A.: *Stochastic Programming*, Handbooks in Operations Research and Management Science, Volume 10, ELSEVIER 2003

[10] TAYLOR, H. M., KARLIN, S.: *An Introduction to Stochastic Modelling*, Third Edition, Academic Press, San Diego, 1998

[11] EASA, S. M., CAN, E. K.: *Optimization Model for Aggregate Blending*, J. Construction Engineering and Management, ASCE, 111, 216-231, 1985

[12] MARKS W., POTREBOWSKI J.: *Multicriteria Optimization of Structual Concrete Mixes*, Arch.Civil Engineering 38(4), pp.77-01, 1992

[13] NEUMANN, D. L.: *Mathematical Method for Blending Aggregates*, Journal of Construction Division, ASCE, 90, 1-13., 1964

[14] SHILSTONE, Sr. J.M.: *Concrete Mixture Optimization*, ACI Concrete International 12.6: 3339., 1990

[15] SVOBODA, L.: *Design of Aggregate Mix*, CTU Rep., Proceedings of Workshop 2002, vol.6, pp. 606-607, Prague 2002, ISBN 80-01-02511-X.

[16] TOKLU, Y.C.: *Aggregate Blending Problem - An Arena of Applications of Optimization Methods*, ICCCBE-IX The 9th International Conference on Computing in Civil and Building Engineering, Taipei, Taiwan; 3-5 April 2002

[17] TOKLU, Y.C.: *Aggregate Blending Using Genetic Algorithms*, Computer-Aided Civil and Infrastructure Engineering, , Vol. 20, No:6, November 2005, pp. 450-460.

# List of symbols

s.t. such that
⌟ end of proof or example
$\in$ element of, $\subset$ subset (proper or not)
$\mathbb{R}$ real numbers, $\mathbb{R}^n$ n-dimensional real numbers, $\overline{\mathbb{R}}$ extended real numbers
$\leq$ less or equal in $\mathbb{R}$ or $\mathbb{R}^n$ (componentwise), $\geq$ greater or equal
$\circ$ composition of mappings
$\rightarrow$ maps to
$|x|$ absolute value
$||\mathbf{x}||_p$ p-norm of $\mathbf{x}$ in $\mathbb{R}^n$
$\infty$, $-\infty$ plus infinity, minus infinity
$\Omega$, $\Omega_1$, $\Omega_2$ sets of events
$\omega$ random event
$\xi$ random vector
$\mathcal{F}$ sigma-algebra
$\mathbb{B}$ Borel sigma-algebra in $\mathbb{R}$, $\mathbb{B}^n$ Borel sigma-algebra in $\mathbb{R}^n$
$\overline{\mathbb{B}}$ extended Borel sigma-algebra
$P$ probability measure, $P_I$ image probability measure
$\mathbb{E}$ expected value functional, $\mathbb{V}ar$ variance functional
$\mu$ measure
$\mathbf{I}$ identity matrix
$id$ identity mapping
$W, \mathbf{W}$ recourse matrix
$T, \mathbf{T}$ technology matrix
$q, \mathbf{q}$ recourse cost vector
$h, \mathbf{h}$ vector in second-stage constraints
$Q$ recourse function
$\mathbf{g}$ grading curve as vector; the goal
$g$ grading curve as function; nonlinear function (in constraints)
$\mathbf{A}$ matrix of grading curves, $\mathbf{A}_l$, $\mathbf{A}_u$ row selections from $A$
$\mathbf{l}$ lower bound (vector), $\mathbf{u}$ upper bound (vector)
$\overline{f}$ extension of $f$
$C$ set of decisions
$f$ objective function
$\gamma$ mapping from program data to its optimum

# Appendix A

# Simulation of grading curves

```
%... the distribution function (piecewise linear):
g = [0 0 9  17 26 38 51  67 74.5  82 88.5 95 97.5 100 ]/100;

N = 8;    %... number of performed simulations
M = 300;  %... sample size

for i=1:N
  for j=1:14
    gsim(i,j)=0;  %... contains the simulated curves
  end
end

for ns = 1:N
 for k = 1:M
  x = rand();

  y = 14;            %... 14 sieves (world-wide standard)
  for i=1:14
    if x <=g(15-i)   %... falling throught the 15-i th sieve
        y = 15-i-1;
    end
  end

  for j=y+1:14;
   gsim(ns,j)=gsim(ns,j)+1;  %... updating the simulated curve
  end
 end
gsim(ns,:) = gsim(ns,:) / gsim(ns,14);   %... limit to 1
end

hold off
plot(1:14,g,'b-','LineWidth',3);
hold on
plot(1:14,gsim,'r-');
axis([1 14 0 1.02]);
```

# Appendix B

# GAMS implementation of Two-stage model 2

The presented model uses the 1-norm in the objective.

```
$eolcom //

Scalar num_scenarios /1000/;

Set s /1*1000/;  //scenarios

Scalar bigM / 100 /;

Set i /1*14/;  //number of sieves
Set j /1*4/;   //number of ingredients

Set setL(i);
Set setC(i);
Set setU(i);

Positive variable x(j);

Positive variables Yplus(s,setC)
                   Yminus(s,setC)
                   Lv(s,setL)
                   Uv(s,setU);

Parameters boundL(i)
           boundU(i)
           boundC(i);

Parameter A(s,i,j);

$include "dataSSBeng.gms"; //input file with data

Variable z; //objective variable

Equations EQobjective
```

```
          EQgoal(s,setC)
          EQupper_bound(s,setU)
          EQlower_bound(s,setL)
          suma1;

EQobjective   ..   z =e= sum(s,1/num_scenarios*( sum(setC,Yplus(s,setC)) +
                                                 sum(setC,Yminus(s,setC))+
                                                 bigM*sum(setL,Lv(s,setL)) +
                                                 bigM*sum(setU,Uv(s,setU)) ) );
EQgoal(s,setC)    .. sum(j,A(s,setC,j)*x(j)) - boundC(setC) =e= Yplus(s,setC) -
                                                                Yminus(s,setC);
EQlower_bound(s,setL) .. sum(j,A(s,setL,j)*x(j)) - Lv(s,setL) =g= boundL(setL);
EQupper_bound(s,setU) .. sum(j,A(s,setU,j)*x(j)) - Uv(s,setU) =l= boundU(setU);
suma1                 .. sum(j,x(j)) =e= 1;


Model angelina /all/;

Solve angelina minimizing z using lp;

Display x.l, z.l, A;
```

# Appendix C

# Excerpt from C++ code

The entire code (solver + model) has approximately 2000 lines.

```
void UrciSloupec(int pravidlo, double** &AA, double* &cc,
        int* &basis, bool& opt, int& s, int m, int n){

  const double epsCol = 1e-14;
  int j,k;
  double res,maxi;

    s=0; opt=false;

  switch(pravidlo){
    case 0: //klasicky Dantzig - hleda maximalni cc
      maxi=epsCol;
      for(j=1;j<=n;j++){
        if((basis[j]==0) && (cc[j]>=maxi)){
         maxi=cc[j];
         s=j;
        }
      } break;

     case 1: //steepest edge
      maxi=0;
      for (j=1; j<=n; j++){
        if ( (basis[j]==0) && (cc[j]>=epsCol)){
          res=1;
          for(k=1; k<=m; k++){ res=res+AA[k][j]*AA[k][j]; }
          if ((cc[j]/res) > maxi){
            maxi=cc[j]/res;
            s=j;
          }
        }
      } break;

     case 2: //Blandovo pravidlo
      j=0;
      while( j<=n && s==0 ){
```

```
        j++;
        if( (basis[j]==0) && (cc[j]>epsCol) ){ s=j; }
       } break;
  }
  if(s==0){ opt=true; }
}
```