

Stochastic Optimization for Robust Planning in Transportation

by

Francisco J. Jauffred

Ingeniero Civil, Universidad Nacional Autonoma de Mexico (1990)
M.S. in Transportation, Massachusetts Institute of Technology (1993)

Submitted to the Department of Civil and Environmental Engineering
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Transportation Science and Logistics

at the

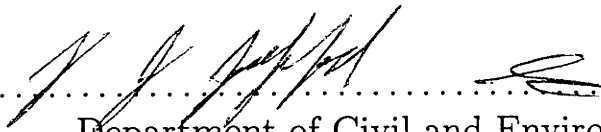
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

December 1997

[February 1998]

© Massachusetts Institute of Technology 1997. All rights reserved.

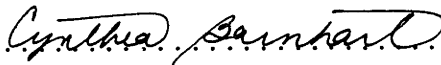
Author



Department of Civil and Environmental Engineering

December 17, 1997

Certified by

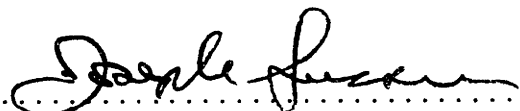


Cynthia Barnhart

Associate Professor of Civil and Environmental Engineering.

Thesis Supervisor

Accepted by



Joseph Sussman

Chairman, Departmental Committee on Graduate Students

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

ARCHIVES

FEB 13 1998

Stochastic Optimization for Robust Planning in Transportation

by

Francisco J. Jauffred

Submitted to the Department of Civil and Environmental Engineering
on December 17, 1997, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Transportation Science and Logistics

Abstract

Many transportation planning problems, such as crew scheduling and traffic assignment, can be modeled using optimization models. Typically these are modeled as deterministic problems, since stochastic models are more difficult to solve. The result is that transportation modelers deal with uncertainties, such as inaccurate forecasting, unplanned events and catastrophic incidents, by performing sensitivity analyses rather than by incorporating randomness into the model structure.

We examine the role of stochastic optimization models in transportation, in particular robust optimization methods in which system reliability and expected costs are balanced. We study stochastic programming duality and find that duality results for deterministic problems apply to stochastic problems. From this, we describe general subgradient methods to solve stochastic convex problems. We use this to solve our *Average Plan Model*. The average plan model, a stochastic extension of current deterministic optimization models, incorporates uncertainty into well known deterministic models. The objective is to find a solution that is *average* in the sense that it is closer to the solution of very high probability events, as opposed to infrequent events. We detail the conditions for optimality of the average plan model and we describe a methodology for its solution.

We demonstrate the applicability of our average plan model and solution methodology by applying them to two transportation problems. The first is a network design problem for the distribution of crops in Mexico and the second is an airplane scheduling problem. We evaluate the solutions provided and contrast them with solutions generated under varying assumptions and policies.

Thesis Supervisor: Cynthia Barnhart

Title: Associate Professor of Civil and Environmental Engineering.

Acknowledgments

In first place I would like to thank Prof. Cynthia Barnhart for her tireless effort to make of this thesis a readable document. I will always cherish my experience as Cindy's student. I hope we will cooperate in the future and she will keep putting periods at the end of my equations. I would like to thank Prof. Jeremy F. Shapiro and Prof. Amadeo Odoni for they work as members of my doctoral committee and their invaluable suggestions and comments.

In the institutional side, I would like to thank the organizations that have supported my research in different ways and made possible for me to get into MIT. First the Mexican Institute of Transportation (Instituto Mexicano del Transporte). In particular the following people: Ing. Luis Enrique Bracamontes, Ing. Daniel Diaz Diaz, Ing. Alfonso Rico and Dr. Octavio Rascon. Thanks are due also to Conacyt, Mexico's equivalent of NSF, and to the University Transportation Center of New England.

In the home end of things, I express my limitless gratitude to my Mother and my sisters Martha, Paola and Luciana for always making me feel close to home, an important part of my life support system. And to my Grandmother for her help in times of hardship.

I thank Chris and Daeki for showing me that there is some sort of life beyond MIT, and that it tastes better with beer (or soju). Another important part of my life support system had been my Mexican friends, neighbors and roommates: Beltran, Alfonso, Gabi Arroyo, Jesus Favela, Memo, Rocio, Paul, Marisol and Agustin. Of course I also remember Sandy and Bailey.

Last but not least, I should mention Tina. Not by chance the actual production of this document coincided with her arrival. It should rightfully be dedicated to her.

Contents

1	Stochastic optimization models in transportation	9
1.1	Computation and mathematical programming	9
1.2	Stochastic models	10
1.2.1	Robust planning	11
1.2.2	Real time operations	11
1.3	The philosophy of the average plan model	12
2	Stochastic Programming and Duality	13
2.1	Introduction	13
2.2	The general stochastic programming problem	13
2.3	The Lagrangian dual of the general stochastic programming problem	16
2.4	Weak duality	19
2.5	Strong duality	19
2.5.1	Discussion	21
2.6	Optimality conditions	22
2.7	Concavity of the Lagrangian and convexity of the feasible region . . .	24
2.8	Chapter 2 summary	25
3	Solution of Stochastic Convex Problems	26
3.1	Introduction	26
3.2	Convex problems in the deterministic variable	26
3.3	The dual of a problem convex in the deterministic variable	28
3.4	Strong duality	29

3.5	Subgradient solution methods	31
3.5.1	A gradient like method	31
3.5.2	Column generation like method	33
3.6	Problems with discrete deterministic variables	34
3.7	Optimization with sampling, stochastic quasigradient methods	36
3.7.1	Stochastic quasigradient method	37
3.8	Chapter 3 summary	38
4	The average plan model for robust planning	40
4.1	Robust planning	40
4.2	The average plan model	41
4.3	The dual of the average plan	42
4.4	Optimality conditions	44
4.5	Dispersion of the solution	45
4.6	The stochastic quasigradient method for the average plan model	45
4.7	The average plan model for multicommodity flows on networks with random capacity and demand	47
4.8	Chapter 4 summary	50
5	The generalized average plan model	52
5.1	Planning models for random scenarios	52
5.2	The generalized average plan model as a quadratic dual	54
5.3	Optimality and strong duality of the quadratic dual	55
5.4	Solution of the generalized average plan model for a fixed value of Q	56
5.4.1	Gradient method	59
5.4.2	Column generation method	60
5.4.3	Stochastic quasigradient method	61
5.5	Applying the generalized average plan model to discrete problems	62
5.5.1	Non-convexity of discrete problems	62
5.5.2	Solving 0-1 problems by thorough exploration of the domain	63
5.5.3	An iteration method	67

5.5.4	Computational issues	68
5.5.5	Saving branch and bound trees	69
5.6	Post-solution analyses	69
5.7	Chapter 5 summary	71
6	Transportation applications of the average plan model	73
6.1	Introduction	73
6.2	A service network design model for the distribution of crops in Mexico	73
6.2.1	Problem background	73
6.2.2	Methodology	74
6.2.3	Results	77
6.3	An airplane scheduling problem	80
6.3.1	Problem background	80
6.3.2	Methodology	81
6.3.3	Results	84
6.4	Chapter 6 summary	86
7	Future Research	87
7.1	Theoretical research issues	87
7.2	Computational issues	88
7.3	Economic issues	88
A	Software implementation	89
A.0.1	mplane.cc	89
A.0.2	plane.h	93
A.0.3	plane.cc	94
A.0.4	vector.h	98
A.0.5	vector.cc	99
A.0.6	array.h	106
A.0.7	array.cc	109
A.0.8	Output for $q = 50$	114

List of Figures

2-1	The stochastic programming problem as a decision tree	15
5-1	A non-convex problem	63
5-2	The same problem with a continuous variable	64
5-3	A plot of the quadratic term	65
6-1	The network model for Mexico's distribution problem	76
6-2	Solution with $q = 50$	78
6-3	Solution with $q = 500$	78
6-4	Deterministic solution of Mexico's logistic problem.	79

List of Tables

- 3.1 Value of the series $\sum_{k=1}^{\infty} \frac{1}{k(n+k)^{1/2}}$ 39

- 6.1 Modeled nodes 75
- 6.2 Shortest path distances in the network 75
- 6.3 Scenarios and associated probabilities 81
- 6.4 Shadow prices of time slots 82
- 6.5 Solution of the average plan model for the flight scheduling problem.
Even numbered flights arrive to airport 1. 85

Chapter 1

Stochastic optimization models in transportation

1.1 Computation and mathematical programming

Transportation systems are growing more complex day to day, and are thus growing beyond the capacity of the human mind alone to operate, even near optimally. It is in this context that the computer becomes the most essential tool for the people that need to plan, to operate and to manage large transportation systems. The properly programmed computer can generate, simulate and evaluate many thousands of alternatives. To increase the utilization (or impact) of modern computers, we need to translate the transportation planning and operations process to simple instructions that accurately reflect its quantitative aspects. That is, we need to formulate a *mathematical model*. The mathematical model relates in quantitative ways the information the planner has about the system (network topology, demand, capacity supply) with the valid decisions he/she can take (new schedule, increase/decrease capacity, build a new link). The expected output of the mathematical model is a possible list of the feasible options and in particular the best option from the point of view of the planner.

1.2 Stochastic models

One important feature to exploit in mathematical and computer modeling of transportation systems is the possibility of planning for uncertain circumstances. Examples of uncertainty in transportation systems are:

- Statistical variations between the actual demand for transportation resources and the forecasted demand;
- Random changes in the capacity of the network links due to weather:
 - In air traffic control the change in weather patterns changes the capacity of airports and airways;
 - Heavy snow or other kind of weather conditions may reduce the capacity of highways and other ground tracks; and
- Random changes in the capacity due to failure of the vehicles.

Although these uncertainties can be forecast on average, their effect is to alter the actual operations from the optimal plan. One rather simple way to do this is to include some *slack* in the data input to the model. For example in scheduling flights for an airline, instead of considering the average fly time between two cities, the planner might consider a larger flight time. This may result in a more reliable plan, but its expected average cost is increased as scheduled flight time increases. This illustrates the inherent trade-off between the expected cost of operation and the reliability of the system.

A more sophisticated approach to capture stochasticity is to use *stochastic programming models* in which the uncertainty is not considered in the model data but in its structure. Stochastic programs include a large number of *scenarios* (even infinity). For each scenario, there are certain corrective actions that are proper only to the scenario, while there are other decisions that are common to all scenarios. This approach divides the decisions in two groups: a first stage in which decisions are taken before knowing the with certainty the details of the scenario and a second stage in

which more information is acquired and the decisions are of a corrective nature.(See [Wet96].)

There are two main applications in transportation science for the stochastic programming approaches: namely robust planning and real time operations.

1.2.1 Robust planning

In robust planning we try to equilibrate the two opposing objectives of minimizing expected cost and maintaining system reliability. In robust planning models, there is a gap between the optimal robust plan and the optimal plan for a particular scenario. The objective is to design the optimal robust plan to make the expected deviation between it and the scenario specific plans small as possible. The robust plan, then, is *central* to all the scenario solutions, but closer to solutions with large probability and far from very low probability scenarios.

1.2.2 Real time operations

An important area of decision making in transportation is real time decision making. In which a manager at given intervals of time decides to move vehicles or to allocate shipments. Although he/she may have a forecast of the state of the network resources in the near future his actions may have effects several periods after they were made so the exact consequences may not be easy to predict. The stochastic optimization model can help to find an optimal decision by modeling, evaluating, generating and simulating alternatives in the short time spans necessary to take a decision. In other hand. the advances in information technology make possible to combine in a single system a plan optimization engine and a system database, that can advice manager in a just in time way.

1.3 The philosophy of the average plan model

This thesis deals with the actual modeling of optimal decisions under uncertainty. It examines and explores general formulation and properties of these models and bridges the gap between the well known and studied deterministic models and their stochastic counterparts. Then the *average plan model* is proposed as one alternative to create stochastic models.

The philosophy of the average plan is to express analytically the trade-off between robustness of a plan and its expected implementation cost. From the point of view of modeling we should think that there are basically two kinds of events that can occur to a system, regular events and catastrophic events.

- **Regular events** are usual, high probability events that the system handles on daily bases. For example in an airport the changes of visibility and cloud ceiling over the day constitute regular events.
- **Catastrophic events** are those that had a very low probability but their cost may be extremely high making a significant impact in the expected operating cost of the system. For example, in an airport a crash landing, a huge winter storm, a terrorist attack.

The average plan model gives the decision maker a decision that balances the costs of these events and finds a plan that is half way between the different conditions in which the system has to operate. Although the distinction between these two kinds of events may be in general subjective or dependent of the decision makers preferences, the average plan model gives a formal mechanism to examine the different possibilities at hand given the importance he is willing to give to catastrophic events.

Chapter 2

Stochastic Programming and Duality

2.1 Introduction

In this chapter we define the mathematical framework of this thesis. First we define the most general stochastic programming problem under study. Then we define its Lagrangian and its Lagrangian dual. It will be shown that if there is strong duality for the second stage deterministic problems, then it exists also for the two stages general problem. From this we can find optimality conditions and the existence of a dual solution

2.2 The general stochastic programming problem

We are trying to solve stochastic programming problems of the form:

$$\begin{aligned} \inf_x E \left[\min_{y(\omega)} f(x, y(\omega), \omega) \right] & \quad (2.1) \\ \text{s.t.} & \\ g(x, y(\omega), \omega) \leq 0 & \quad \text{a.s.} \end{aligned}$$

$$\begin{aligned}
x &\in X \\
y(\omega) &\in Y(\omega) \quad a.s.
\end{aligned}$$

where:

$x \in X$ is an element from a set of *deterministic decisions*;

$\omega \in \Omega$ is a scenario from the scenario set; and

$y(\omega) \in Y(\omega)$ is an element of the set of decisions proper only to scenario ω .

All variables are vectors of the appropriate dimensions.

Problem 2.1 can be seen as a set of different optimization problems tied together by the deterministic variable x . This gives way to the *two stage* interpretation of stochastic programming problems. The deterministic variable x can be interpreted as a *first stage* decision in which only the probabilities of future events are known and it is necessary to optimize the future expected outcome. The stochastic variable $y(\omega)$ can be interpreted as a *second stage* variable in which the random event is known with certainty and it is necessary to take a decision that optimizes the outcome for that particular event. Thus problem 2.1 is a mathematical programming interpretation of a decision tree as depicted in figure 2-1.

As for the technical side of the probability theory, we make the following assumptions:

Assumption 1 There is a probability space $(\Omega, \mathfrak{F}, P)$, where Ω is the scenario set, \mathfrak{F} is a σ -algebra over Ω and P is a probability measure.

Assumption 2 For each first stage decision $x \in X$ and for each $\omega \in \Omega$ with $p(\omega) > 0$, there exists:

$$y^*(\omega) = \arg \min_{\substack{y(\omega) \in Y(\omega) \\ g(x, y(\omega), \omega) \leq 0}} f(x, y(\omega), \omega)$$

such that

$$-\infty < f(x, y(\omega)^*, \omega) < \infty$$

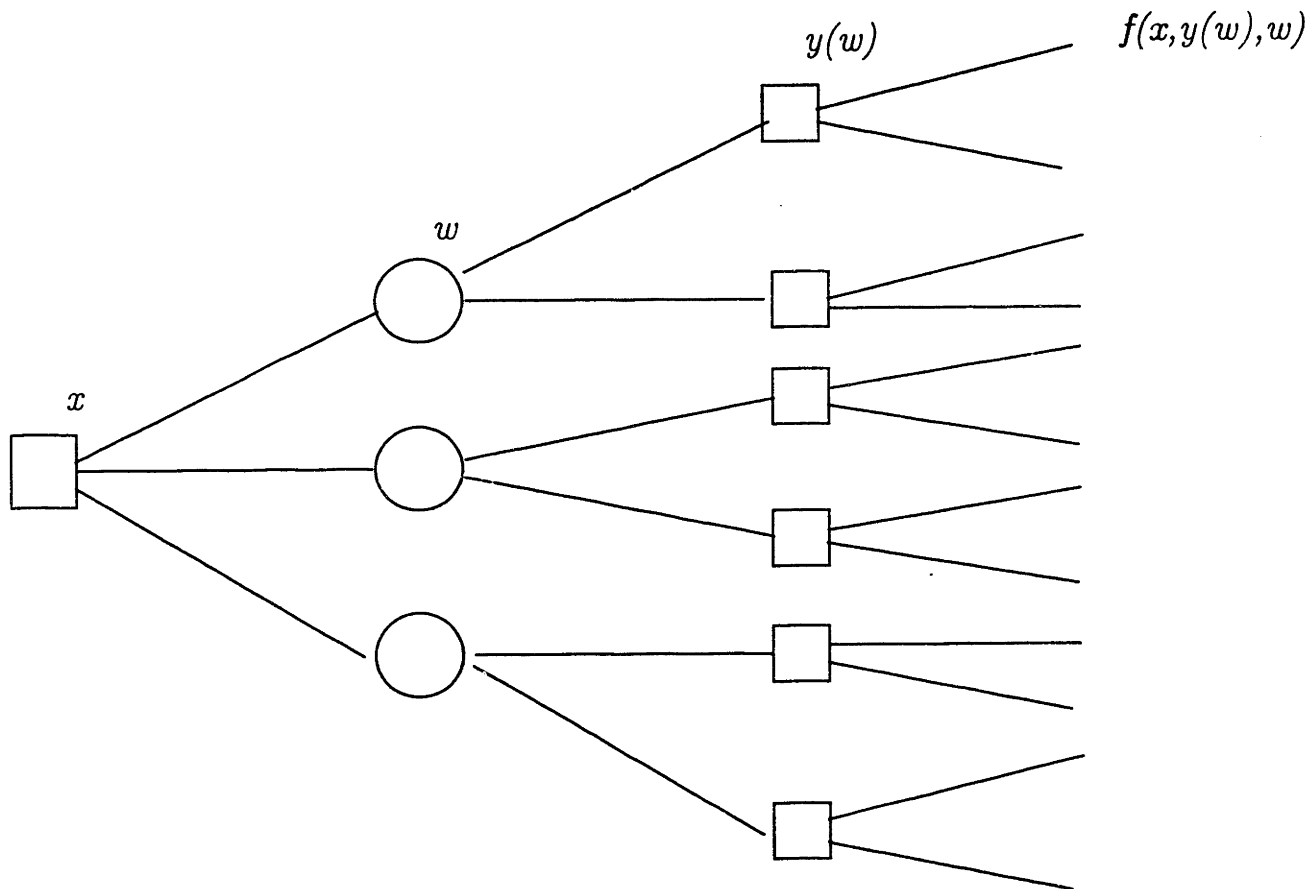


Figure 2-1: The stochastic programming problem as a decision tree

Then we have:

$$\int_{\Omega} |f(x, y(\omega), \omega)| dP(\omega) < \infty$$

so

$$E \left[\min_{\substack{y(\omega) \in Y(\omega) \\ g(x, y(\omega), \omega) \leq 0}} f(x, y(\omega), \omega) \right] \quad (2.2)$$

exists. This means that if there exists a finite optimal solution for all second stage decision problems, then the expected value 2.2 exists and problem 2.1 has a solution.

2.3 The Lagrangian dual of the general stochastic programming problem

We define the Lagrangian for 2.1 as:

$$L(\lambda(\omega)) = \inf_{x \in X} E \left[\min_{y(\omega) \in Y(\omega)} f(x, y(\omega), \omega) + \lambda(\omega) g(x, y(\omega), \omega) \right]. \quad (2.3)$$

Let us also define the dual of 2.1 as:

$$\begin{aligned} & \sup L(\lambda(\omega)) && (2.4) \\ & \text{s.t.} \\ \lambda(\omega) \in & \left\{ \begin{array}{l} \lambda(\omega) \mid \min_{y(\omega) \in Y(\omega)} f(x, y(\omega), \omega) + \lambda(\omega) g(x, y(\omega), \omega) > -\infty, \\ \forall \omega : p(\omega) > 0, \quad \forall x \in X \end{array} \right\} && (2.5) \\ & \text{a.s.} \end{aligned}$$

$$\lambda(\omega) \geq 0 \quad \text{a.s.}$$

where the existence condition 2.5 reflects the need to be sure that the mathematical expectation exists. Then the dual is constrained to the region in which the mathematical expectation in the Lagrangian is well defined. Consider the following example.

Example 1 Consider the linear stochastic programming problem:

$$\begin{aligned} \min_x cx + E \left[\min_{y(\omega)} q(\omega) y(\omega) \right] & \quad (2.6) \\ \text{s.t.} & \\ Ax = b & \\ T(\omega)x + Wy(\omega) = h(\omega) & \quad \text{a.s.} \\ x \geq 0, \quad y(\omega) \geq 0 & \quad \text{a.s.} \end{aligned}$$

This is the well known single recourse stochastic programming problem (see Birge [BL97]). It is a very general version of a two stage stochastic linear program. To show how we obtain its Lagrangian dual we have to rewrite it as:

$$\begin{aligned} \min_{x_0} E \left[\min_{x(\omega), y(\omega)} cx(\omega) + q(\omega) y(\omega) \right] & \quad (2.7) \\ \text{s.t.} & \\ x(\omega) = x_0 & \quad \text{a.s.} \\ Ax(\omega) = b & \quad \text{a.s.} \\ T(\omega)x(\omega) + Wy(\omega) = h(\omega) & \quad \text{a.s.} \\ x(\omega) \geq 0, \quad y(\omega) \geq 0 & \quad \text{a.s.} \end{aligned}$$

Then the Lagrangian is :

$$L(\lambda(\omega), \gamma(\omega), \mu(\omega)) = \inf_{x_0} E \left[\begin{aligned} & \min_{x(\omega) \geq 0, y(\omega) \geq 0} cx(\omega) + q(\omega) y(\omega) + \gamma(\omega) (x_0 - x(\omega)) \\ & \quad + \mu(\omega) (b - Ax(\omega)) \\ & \quad + \lambda(\omega) (h(\omega) - T(\omega)x(\omega) + Wy(\omega)) \end{aligned} \right] \quad (2.8)$$

Take notice that the dual variables $\gamma(\omega)$, $\mu(\omega)$ and $\lambda(\omega)$ are unrestricted in sign.

Changing the order of terms in the Lagrangian 2.8 we obtain:

$$L(\lambda(\omega), \gamma(\omega), \mu(\omega)) = \inf_{x_0} E \left[\begin{array}{l} \min_{\substack{x(\omega) \geq 0, \\ y(\omega) \geq 0}} (c - \gamma(\omega) I - \mu(\omega) A - \lambda(\omega) T(\omega)) x(\omega) \\ + (q(\omega) - \lambda(\omega) W) y(\omega) + \\ \mu(\omega) b + \lambda(\omega) h(\omega) - \gamma(\omega) x_0 \end{array} \right] \quad (2.9)$$

To enforce the existence condition 2.5, we must be sure that what is inside of the expectation operator 2.9 is finite. This implies that:

$$c - \gamma(\omega) I - \mu(\omega) A - \lambda(\omega) T(\omega) \geq 0 \quad a.s. \quad (2.10)$$

$$q(\omega) - \lambda(\omega) W \geq 0 \quad a.s. \quad (2.11)$$

otherwise, if any of the components of these vectors is negative, we can make the corresponding value of $x(\omega)$ or $y(\omega)$ arbitrarily large. Satisfying constraints 2.10 and 2.11, however, causes the corresponding terms in the Lagrangian 2.9 to equal zero. We can also observe that in order to have a finite solution, it is necessary that:

$$E[\gamma(\omega)] = 0.$$

So the dual of 2.1 is:

$$\begin{aligned} & \max E[\mu(\omega) b + \lambda(\omega) h] \\ & \text{s.t.} \\ & \gamma(\omega) I + \mu(\omega) A + \lambda(\omega) T \leq c \quad a.s \\ & \lambda(\omega) W \leq q(\omega) \quad a.s. \\ & E[\gamma(\omega)] = 0. \end{aligned}$$

■

2.4 Weak duality

The weak duality result for the stochastic two stage problem is similar to that for deterministic problems (see [Ber95] and [Sha79]).

Theorem 1 (Weak duality) *The primal problem 2.1 and its dual problem 2.3 are always related by the following inequality:*

$$\sup_{\lambda(\omega) \geq 0} L(\lambda(\omega)) \leq \inf_{x \in X} E \left[\min_{\substack{y(\omega) \in Y(\omega) \\ g(x, y(\omega), \omega) \leq 0}} f(x, y(\omega), \omega) \right]. \quad (2.12)$$

Proof. To prove it, observe that by the definition of the feasible region of the Lagrangian, and since $\lambda(\omega) \geq 0$, we have for all x and $y(\omega)$ feasible in 2.1:

$$f(x, y(\omega), \omega) + \lambda(\omega) g(x, y(\omega), \omega) \leq f(x, y(\omega), \omega) \quad a.s.. \quad (2.13)$$

Since 2.13 holds with probability one, we have:

$$E[f(x, y(\omega), \omega) + \lambda(\omega) g(x, y(\omega), \omega)] \leq E[f(x, y(\omega), \omega)] \\ \forall x \in X, \quad \forall y(\omega) \in Y(\omega) \quad a.s..$$

Then,

$$\inf_{x \in X} E \left[\min_{y(\omega) \in Y(\omega)} f(x, y(\omega), \omega) + \lambda(\omega) g(x, y(\omega), \omega) \right] \leq \inf_{x \in X} E \left[\min_{y(\omega) \in Y(\omega)} f(x, y(\omega), \omega) \right].$$

■

2.5 Strong duality

As in the previous section we can extend the strong duality result for deterministic problems in developing strong duality for problem 2.1.

Theorem 2 (Strong duality) *Let the deterministic problem for a fixed scenario ω :*

$$\begin{aligned} \min_{y(\omega)} f(x, y(\omega), \omega) & \quad (2.14) \\ \text{s.t.} & \\ g(x, y(\omega), \omega) \leq 0 & \quad \text{a.s.} \\ y(\omega) \in Y(\omega) & \quad \text{a.s.} \end{aligned}$$

have an optimal solution and optimal Lagrangian multipliers with no duality gap for all $x \in X$, $X \neq \emptyset$ and all ω such that $p(\omega) > 0$. Then

$$\sup_{\lambda(\omega) \geq 0} L(\lambda(\omega)) = \inf_{x \in X} E \left[\min_{\substack{y(\omega) \in Y(\omega) \\ g(x, y(\omega), \omega) \leq 0}} f(x, y(\omega), \omega) \right] \quad (2.15)$$

holds.

Proof. Let $\mu(x, \omega) \geq 0$ a.s. be the optimal Lagrangian multipliers for problem 2.14. Then we have:

$$\min_{y(\omega) \in Y(\omega)} f(x, y(\omega), \omega) = \min_{y(\omega) \in Y(\omega)} f(x, y(\omega), \omega) + \mu(x, \omega) g(x, y(\omega), \omega)$$

Since this equality holds with probability one and by assumption 2, we can take the mathematical expectation:

$$E \left[\min_{y(\omega) \in Y(\omega)} f(x, y(\omega), \omega) \right] = E \left[\min_{y(\omega) \in Y(\omega)} f(x, y(\omega), \omega) + \mu(x, \omega) g(x, y(\omega), \omega) \right] \quad (2.16)$$

It follows:

$$\inf_{x \in X} E \left[\min_{y(\omega) \in Y(\omega)} f(x, y(\omega), \omega) \right] = \inf_{x \in X} E \left[\min_{y(\omega) \in Y(\omega)} f(x, y(\omega), \omega) + \mu(x, \omega) g(x, y(\omega), \omega) \right],$$

and

$$\begin{aligned} & \inf_{x \in X} E \left[\min_{y(\omega) \in Y(\omega)} f(x, y(\omega), \omega) + \mu(x, \omega) g(x, y(\omega), \omega) \right] \\ \leq & \sup_{\lambda(\omega) \geq 0} \inf_{x \in X} E \left[\min_{y(\omega) \in Y(\omega)} f(x, y(\omega), \omega) + \lambda(\omega) g(x, y(\omega), \omega) \right] \end{aligned}$$

which in combination with 2.12 proves 2.15 . ■

2.5.1 Discussion

There are some important features of this strong duality result worth mentioning. First, the result does not require any particular problem structure (like convexity). Instead, the only necessary condition is that the second stage problem 2.14 have a Lagrangian solution and no duality gap. Although this condition holds always for convex problems, it may also hold in special cases for other kinds of math programs, for example some integer programming problems.

Surprisingly, the requirements for the deterministic variables are even more relaxed. In fact the only requirement is that the set X is not empty. This gives us great liberty in modeling problems of the form of 2.1 since the first stage decision variables are not constrained to any particular structure set. In fact, the first stage decision variables can be integer or continuous.

Corollary 3 (Complementary Slackness) *Let 2.1 be a problem that satisfies the strong duality conditions. Then for an optimal triplet $(x^*, y(\omega)^*, \lambda(\omega)^*)$ for 2.1, we have the following complementary slackness requirement:*

$$E [\lambda(\omega)^* g(x^*, y(\omega)^*, \omega)] = 0. \tag{2.17}$$

Proof: From the strong duality result 2.15 we have:

$$E [f(x^*, y(\omega)^*, \omega) + \lambda(\omega)^* g(x^*, y(\omega)^*, \omega)] = E [f(x^*, y(\omega)^*, \omega)].$$

The result follows immediately.

Corollary 4 (Deterministic multipliers) *Let 2.1 be a problem that satisfies the strong duality conditions and let the triplet $(x^*, y(\omega)^*, \lambda(\omega)^*)$ be the optimal solution to its dual. Let $\mu(x^*, \omega)$ be the optimal Lagrangian multipliers for problem 2.14 then*

$$\lambda(\omega)^* = \mu(x^*, \omega).$$

Proof: Letting $x = x^*$ in the strong duality result, then:

$$E \left[\min_{y(\omega) \in Y(\omega)} f(x^*, y(\omega), \omega) + \lambda(\omega)^* g(x^*, y(\omega), \omega) \right] = E \left[\min_{\substack{y(\omega) \in Y(\omega) \\ g(x^*, y(\omega), \omega) \leq 0}} f(x^*, y(\omega), \omega) \right].$$

By the definition of mathematical expectation, we have with probability one:

$$\min_{y(\omega) \in Y(\omega)} f(x^*, y(\omega), \omega) + \lambda(\omega)^* g(x^*, y(\omega), \omega) = \min_{\substack{y(\omega) \in Y(\omega) \\ g(x^*, y(\omega), \omega) \leq 0}} f(x^*, y(\omega), \omega). \quad (2.18)$$

2.18 is the optimality condition for problem 2.14 with $x = x^*$. So $\lambda(\omega)^* = \mu(x^*, \omega)$

2.6 Optimality conditions

Theorem 5 (Optimality Condition) *Let the triplet $(x^*, y(\omega)^*, \lambda(\omega)^*)$ satisfy the following optimality conditions:*

1. *It is optimal for the dual problem 2.4;*
2. *It satisfies the complementary slackness requirement 2.17; and*
3. *It feasible for the primal problem 2.1.*

Then the triplet is optimal in the primal problem 2.1 .

Proof. We have for any pair $(x, y(\omega))$ feasible in 2.1:

$$\begin{aligned}
E[f(x^*, y(\omega)^*, \omega) + \lambda(\omega)^* g(x^*, y(\omega)^*, \omega)] &\leq E[f(x, y(\omega), \omega) + \lambda(\omega)^* g(x, y(\omega), \omega)] \\
&\leq E[f(x, y(\omega), \omega)]
\end{aligned}$$

and by the second condition :

$$E[f(x^*, y(\omega)^*, \omega)] \leq E[f(x, y(\omega), \omega)].$$

■

Corollary 6 (Necessary and Sufficient Conditions) *A triplet $(x^*, y(\omega)^*, \lambda(\omega)^*)$ satisfies the optimality conditions if and only if there is strong duality.*

Proof. From optimality conditions 1 and 2:

$$E[f(x^*, y(\omega)^*, \omega) + \lambda(\omega)^* g(x^*, y(\omega)^*, \omega)] = E[f(x^*, y(\omega)^*, \omega)]$$

and so strong duality is satisfied. In the reverse part of the proof, if there is strong duality, then conditions 1 and 3 must be satisfied. Condition 2 is guaranteed by complementary slackness. ■

Theorem 7 (Existence) *Let problem 2.1 be such that for all $x \in X$, $X \neq \emptyset$ and for all $\omega \in \Omega$ such that $p(\omega) > 0$, problem 2.14 has an optimal solution and optimal deterministic Lagrangian multipliers. Then there exists a triplet $(x^*, y(\omega)^*, \lambda(\omega)^*)$ that satisfies the optimality conditions and is optimal in the dual problem 2.4.*

Proof. Since the set X is not empty and $f > -\infty$ in the feasible region so the mathematical expectation exists, an optimal x^* exists. By the definition of 2.14, $y(\omega)^*$ and $\mu(x^*, \omega)$ also exist. So, a primal optimal solution exists. But by the Strong Duality Theorem and the Necessity and Sufficiency Corollary, there also exists an optimal dual solution. ■

2.7 Concavity of the Lagrangian and convexity of the feasible region

Theorem 8 *The Lagrangian 2.3 is concave.*

Proof. Let $x, y(\omega)$ be the corresponding values for the Lagrangian at $\lambda(\omega)$, then:

$$\begin{aligned}
 & L(\lambda(\omega)) + E \left[(\lambda(\omega)' - \lambda(\omega)) g(x, y(\omega), \omega) \right] & (2.19) \\
 = & E \left[f(x, y(\omega), \omega) + \lambda(\omega) g(x, y(\omega), \omega) + (\lambda(\omega)' - \lambda(\omega)) g(x, y(\omega), \omega) \right] \\
 = & E \left[f(x, y(\omega), \omega) + \lambda(\omega)' g(x, y(\omega), \omega) \right] \geq L(\lambda(\omega)').
 \end{aligned}$$

Then for $\lambda(\omega)^0$ and $\lambda(\omega)^1$ we obtain:

$$\begin{aligned}
 L\left(\frac{\lambda(\omega)^0 + \lambda(\omega)^1}{2}\right) + E \left[\left(\frac{\lambda(\omega)^1 - \lambda(\omega)^0}{2}\right) g(x, y(\omega), \omega) \right] & \geq L(\lambda(\omega)^1) \\
 L\left(\frac{\lambda(\omega)^0 + \lambda(\omega)^1}{2}\right) + E \left[\left(\frac{\lambda(\omega)^0 - \lambda(\omega)^1}{2}\right) g(x, y(\omega), \omega) \right] & \geq L(\lambda(\omega)^0)
 \end{aligned}$$

adding the two inequalities:

$$L\left(\frac{\lambda(\omega)^0 + \lambda(\omega)^1}{2}\right) \geq \frac{L(\lambda(\omega)^1) + L(\lambda(\omega)^0)}{2}.$$

Thus the Lagrangian is concave. ■

Theorem 9 (Convexity of the feasible region) *The feasible region of the dual 2.5 is convex.*

Proof. It is trivial to see that the constraints $\lambda(\omega) \geq 0$ form a convex set. For constraints 2.5 consider the function:

$$\ell(\lambda(\omega), x, \omega) = \min_{y(\omega) \in Y(\omega)} f(x, y(\omega), \omega) + \lambda(\omega) g(x, y(\omega), \omega) \quad \forall x \in X, \quad a.s..$$

By arguments similar to the last theorem, we find for any two feasible points $\lambda(\omega)^0$ and $\lambda(\omega)^1$:

$$\ell(\alpha\lambda(\omega)^0 + (1 - \alpha)\lambda(\omega)^1, x, \omega) \geq \alpha\ell(\lambda(\omega)^0, x, \omega) + (1 - \alpha)\ell(\lambda(\omega)^1, x, \omega) > -\infty.$$

So clearly the feasible region is convex. ■

2.8 Chapter 2 summary

In this chapter, we extend and apply Lagrangian theory for deterministic optimization to problems with a mixture of random and deterministic variables and deterministic and probabilistic constraints. We call these problems *two stage stochastic programs* because we can separate the problem into a *first stage* containing only deterministic variables and a *second stage* containing only random variables.

We define the Lagrangian for the 2-stage stochastic programming problem and show that weak duality is always satisfied. Further we show that if there is strong duality in the second stage problems for all positive probability scenarios, then strong duality exists for the 2-stage problem. We show that there is a complementary slackness condition that is both necessary and sufficient for optimality. Finally we show that the Lagrangian is always concave in the Lagrangian multipliers.

Chapter 3

Solution of Stochastic Convex Problems

3.1 Introduction

In Chapter 2, we showed some general properties of stochastic programming problems and Lagrangian duals. Particularly interesting is the fact that very little special problem structure is needed to achieve quite strong results. In this chapter we develop solution methods for problems satisfying certain convexity requirements.

Since the stochastic program 2.1 decomposes into independent problems for a given x , it is natural to try to solve the problem by breaking it into scenario-specific subproblems. Some methods have been developed for the *stochastic linear programming* problem, such as the L-shape method and the progressive hedging algorithm (see Birge [BL97], Kall [KW94] and Wets [Wet88]). These methods require the problem to be convex in the first stage variable.

3.2 Convex problems in the deterministic variable

Consider the second stage problem for a fixed deterministic variable $x_0 \in R^n$ and some scenario $\omega \in \Omega$:

$$\begin{aligned}
& \min_{y(\omega) \in Y(\omega)} f(x_0, y(\omega), \omega) & (3.1) \\
& \text{s.t.} \\
& g(x_0, y(\omega), \omega) \leq 0 \quad \text{a.s.} \\
& y(\omega) \in Y(\omega) \quad \text{a.s..}
\end{aligned}$$

Assuming that the optimal Lagrangian multipliers exist for this problem, we define the function:

$$p(x_0, \omega) = \sup_{\mu(x_0, \omega) \geq 0} \min_{y(\omega) \in Y(\omega)} f(x_0, y(\omega), \omega) + \mu(x_0, \omega) g(x_0, y(\omega), \omega) \quad (3.2)$$

We say that problem 2.1 is *convex in the deterministic variable* if there exists a random variable $\gamma(\omega) \in R^n$ such that:

$$\begin{aligned}
p(x_0, \omega) & \leq p(x, \omega) - \gamma(\omega)(x - x_0) \quad \text{a.s.} & (3.3) \\
\forall x_0 & \in X, \forall x \in X \subseteq R^n.
\end{aligned}$$

Clearly, condition 3.3 means that $p(x, \omega)$ is convex in x and $\gamma(\omega)$ is a subgradient with probability one.

Furthermore, it can be seen from 3.3 that:

$$\begin{aligned}
E[p(x_0, \omega)] & \leq E[p(x, \omega)] - E[\gamma(\omega)](x - x_0) & (3.4) \\
\forall x_0 & \in X, \forall x \in X \subseteq R^n,
\end{aligned}$$

since 3.3 holds with probability one.

3.3 The dual of a problem convex in the deterministic variable

Consider problem 2.1 and assume that it is convex in the deterministic variable. It can be written as:

$$\begin{aligned}
 & \min_{x(\omega), y(\omega)} E[f(x(\omega), y(\omega), \omega)] & (3.5) \\
 & \text{s.t.} \\
 & g(x(\omega), y(\omega), \omega) \leq 0 \quad \text{a.s.} \\
 & x(\omega) - x_0 = 0 \quad \text{a.s.} \\
 & x_0 \in R^n, x(\omega) \in X, y(\omega) \in Y(\omega) \quad \text{a.s.}
 \end{aligned}$$

where the deterministic variable has been replaced by a set of random variables $x(\omega)$, all constrained to be equal. Dualizing this constraint we obtain:

$$\begin{aligned}
 L(\gamma(\omega)) &= \min_{x(\omega), y(\omega)} E[f(x(\omega), y(\omega), \omega) + \gamma(\omega)(x_0 - x(\omega))] & (3.6) \\
 & \text{s.t.} \\
 & g(x(\omega), y(\omega), \omega) \leq 0 \quad \text{a.s.} \\
 & x(\omega) \in X, y(\omega) \in Y(\omega) \quad \text{a.s..}
 \end{aligned}$$

We define the following scenario independent problems:

$$\begin{aligned}
 q(\gamma(\omega)) &= \min_{x(\omega), y(\omega)} f(x(\omega), y(\omega), \omega) - \gamma(\omega)x(\omega) & (3.7) \\
 & \text{s.t.} \\
 & g(x(\omega), y(\omega), \omega) \leq 0 \quad \text{a.s.} \\
 & x(\omega) \in X, y(\omega) \in Y(\omega) \quad \text{a.s..}
 \end{aligned}$$

Using 3.7, 3.6 can be written as:

$$L(\gamma(\omega)) = E[q(\gamma(\omega))] + \min_{x_0 \in \mathbb{R}^n} E[\gamma(\omega)] x_0. \quad (3.8)$$

For the Lagrangian to be defined we have to impose the constraint $E[\gamma(\omega)] = 0$. So the dual of 3.8 is:

$$\begin{aligned} & \max_{\gamma(\omega)} E[q(\gamma(\omega))] & (3.9) \\ & \text{s.t.} \\ & E[\gamma(\omega)] = 0. \end{aligned}$$

The main difference between this result and the dual found in 2.4 is that in 2.4, we didn't dualize the first stage variable. So there was strong duality when there was strong duality for all the second stage variables. In 3.9, the first stage variable is dualized, adding a new level of complexity to the problem. In fact, as shown in the next section, we can not ensure that there is strong duality unless the problem is convex in the deterministic variable.

3.4 Strong duality

Consider the following problem:

$$\begin{aligned} & \min p(x(\omega), \omega) & (3.10) \\ & \text{s.t.} \\ & x(\omega) - x^k = 0 \quad \text{a.s.} \\ & x(\omega) \in X \quad \text{a.s.} \end{aligned}$$

The Lagrangian of this problem is:

$$\sup_{\gamma(\omega) \in \mathbb{R}^n} \min_{x(\omega) \in X} p(x(\omega), \omega) - \gamma(\omega) (x(\omega) - x^k). \quad (3.11)$$

We can find $\gamma(\omega)$ by using the following lemma.

Lemma 10 (Deterministic convexity) *Let $p(x(\omega), \omega)$ be convex over X with probability one. Let $\gamma(\omega)$ be a subgradient to $p(x(\omega), \omega)$ at $x(\omega) = x^k$ with probability one. Then $\gamma(\omega)$ is the optimal solution to 3.11 and there is no duality gap.*

Proof. We have from the convexity assumption:

$$\min_{x(\omega) \in X} p(x(\omega), \omega) - \gamma(\omega) (x(\omega) - x^k) \leq p(x^k, \omega) \leq p(x(\omega), \omega) - \gamma(\omega) (x(\omega) - x^k) \quad a.s. \quad (3.12)$$

The left hand of the assumption holds from the optimal solution of problem 3.10. The second inequality is the definition of a function convex in the deterministic variable 3.3. ■

This takes us to the issue of the strong duality between the dual problem 3.9 and the primal 2.1. Notice that for convex problems in the deterministic variable, with the optimal solution $(x^*, \gamma(\omega)^*)$ we have:

$$q(\gamma(\omega)^*) = p(x^*, \omega) - \gamma(\omega)^* x^* \quad (3.13)$$

and taking expectations:

$$E[q(\gamma(\omega)^*)] = E[p(x^*, \omega)] - E[\gamma(\omega)^*] x^*. \quad (3.14)$$

So for a random variable $\gamma(\omega)^*$ that is feasible in the dual 3.9, we obtain:

$$E[q(\gamma(\omega)^*)] = E[p(x^*, \omega)] = E \left[\min_{\substack{y(\omega) \in Y(\omega) \\ g(x^*, y(\omega), \omega) \leq 0 \quad a.s.}} f(x^*, y(\omega), \omega) \right] \quad (3.15)$$

because we assume that there is strong duality in the second stage problem.

Theorem 11 (Strong convex duality) *If the primal problem 2.1 is convex in the deterministic variable and the second stage problems have optimal Lagrangian multipliers, then 3.9 is a strong dual.*

Proof: This follows from the identity in equation 3.15.

3.5 Subgradient solution methods

By taking the expectation over the right hand side of inequality 3.12, it can be seen that $-E[\gamma(\omega)]$ is a subgradient to problem 3.10. This is the key to solving problems that are convex in the deterministic variable, since there are methods for convex deterministic problems that are based on using subgradients to point towards the optimal solution. Such methods have been reviewed by Shapiro [Sha79] and Bertsekas [Ber95]. The complicated part is solving for the stochastic variables in the problem. So it is necessary to separate solving for the stochastic variables from solving for the deterministic ones. Algorithms achieve this using a Bender's decomposition approach, as suggested by Shapiro [Sha79]. An important example of this approach is the L-shape method presented by Wets [Wet88] for the stochastic linear programming problem.

In the next sections, two methods using subgradient optimization are presented for problems convex in the deterministic variable.

3.5.1 A gradient like method

The steps of our gradient-like method are:

Step 0 Initialization. Let $k = 0$. Find a feasible deterministic variable $x^k \in X$.

Step 1 Stochastic variables solution. For every scenario ω solve the problem:

$$\begin{aligned} \min f(x(\omega), y(\omega), \omega) \\ \text{s.t.} \end{aligned} \tag{3.16}$$

$$\begin{aligned}
g(x(\omega), y(\omega), \omega) &\leq 0 && \text{a.s.} \\
x(\omega) - x^k &= 0 && \text{a.s.} \\
x(\omega) \in X, y(\omega) \in Y(\omega) &&& \text{a.s.}
\end{aligned}$$

Obtain the subgradient $-\gamma(\omega)^k$, represented by the vector of dual variables associated with the constraints $x(\omega) - x^k = 0$ a.s. . In the special case where the functions f and g have derivatives at x^k (a Hessian exists) then:

$$\gamma(\omega)^k = \nabla_{x(\omega)} f(x^k, y(\omega), \omega) + \lambda(\omega) \nabla_{x(\omega)} g(x^k, y(\omega), \omega) \quad (3.17)$$

is the subgradient.

Step 2 Descent step. Find a new solution:

$$x^{k+1} = x^k + \frac{1}{k} E[\gamma(\omega)^k]. \quad (3.18)$$

Let $k := k + 1$.

Step 3 Stopping rule. If $|E[\gamma(\omega)^k]| \leq \epsilon$, stop since the solution has been found (ϵ is a suitable tolerance), else go to Step 1.

This a common gradient method, as shown in [Ber95]. There are two remarkable things about it. First, all of the stochastic parts of the problem has been concentrated in Step 2 . It assumes that it is possible to solve problem 3.16 for a general random variable $y(\omega)$. This theoretically is possible, although it may seldom be accomplished in practice unless special structure exists. In particular, in the case of discrete scenarios, 3.16 is solved for all scenarios at each iteration step. In the case of continuous distributions, often the distribution are discretized in order to solve 3.16 (See Birge and Louveaux [BL97]).

The second thing to notice in the algorithm is the descent step (equation 3.18) is $\frac{1}{k}$. This step size guarantees convergence when $k \rightarrow \infty$, although it is not unique. Many other step sizes can be selected, as discussed by Bertsekas [Ber95].

3.5.2 Column generation like method

The gradient like method of section 3.5.1 can be redefined so that the descent step uses the knowledge of all previous solutions generated. Such methods, referred to as *Generalized Linear Programming*, have been proposed by Geoffrion [Geo72] Bertsekas [Ber95] and Lasdon [Las70]. Applying these ideas, we replace step 2 in our gradient-like algorithm with a step involving the solution of a linear program. The linear programming algorithm exploits the convexity property 3.3 to obtain a new point x^k . The revised step 2 is:

Step 2 **Descent step.** Solve:

$$\begin{aligned} & \min \theta && (3.19) \\ & \text{s.t.} \\ & E [q (\gamma (\omega)^i)] \leq \theta - E [\gamma (\omega)^i] x \quad i = 1, \dots, k \\ & x \in R^n, \theta \in R \end{aligned}$$

Make $x^{k+1} := x$ and $k := k + 1$.

To understand how this method works, consider that the constraints in the LP are a piecewise linear approximation to the objective function. Observe that each constraint can be rewritten using 3.4 as:

$$E [p (x^i, \omega)] - E [\gamma (\omega)^i] x^i \leq E [p (x, \omega)] - E [\gamma (\omega)^i] x \quad (3.20)$$

So given the assumption that the problem is convex in the deterministic variable, constraints of the form of 3.20 actually are a convex representation of the problem 3.2. The convergence of a method like this has been studied by Shapiro [Sha79] and by Bertsekas [Ber95].

A very interesting interpretation involves the column generation aspect of this method. The column generation approach takes a sample of points x^1, x^2, \dots, x^k from the domain X . For each point, problem 3.2 is solved obtaining the values of the ex-

pected subgradient $E[\gamma(\omega)^i]$ and the expected value of the problem $E[q(\gamma(\omega)^i)] = E[p(x^i, \omega)] - E[\gamma(\omega)^i] x^i$. Then we can write a discretized version of the dual problem 3.9 :

$$\begin{aligned} & \min \sum_{i=1}^k \alpha_i E[q(\gamma(\omega)^i)] & (3.21) \\ & \text{s.t.} \\ & \sum_{i=1}^k \alpha_i E[\gamma(\omega)^i] = 0 \\ & \sum_{i=1}^k \alpha_i = 1 \\ & \alpha_i \geq 0 \quad i = 1, \dots, k \end{aligned}$$

That is in fact the dual of problem 3.20. The idea drawn for this representation is that if a sample large enough from the domain is taken, the optimal or near optimal solution can be found. If the sample is not large enough, a new sample point can be obtained by setting the dual variables of the first set of constraints in 3.21 to the value of the new sample point.

Using column generation jargon, step 2 is the master problem 3.21, and step 1 is the *pricing problem*, that is, the problem of finding a new variable to enter the LP basis. Thus finding the optimal solution to the convex problem is equivalent to solving a LP problem with a huge number of columns, each one representing a feasible point from the domain.

3.6 Problems with discrete deterministic variables

Discrete domain problems are known to be difficult and most of the solution methods fall short because they might require complete enumeration. In the particular framework of this chapter there is not any reason why the column generation method should not work for problems with discrete deterministic variables.

In fact, consider the domain X to be finite, then it is possible to write a discrete dual as 3.21 with all the elements of the domain. Such problems may be large but

with the techniques of column generation it is possible and practical to find solutions.

The really important issue about problems with discrete deterministic variables is that no point in the domain will honor by itself the optimality condition $E[\gamma(\omega)] = 0$. The gradient may not vanish for any of the feasible solutions. But instead what we can find for the optimal solution of 3.21 is a subset of solutions $X_s \subset X$ such $X_s = \{i \mid \alpha_i > 0\}$ that necessarily honors the constraint $\sum_{i \in X_s} \alpha_i E[\gamma(\omega)^i] = 0$. So in fact we can find a subset of solutions for which a convex combination of their subgradients will honor the optimality condition. The optimal solution is then found by enumerating the elements of the set X_s and selecting the one that has the smallest value of $E[q(\gamma(\omega))]$. Notice that by the properties of LP, if the vector $E[\gamma(\omega)]$ is size n , then the maximum number of elements in the set X_s is $n + 1$. Making the enumeration easy and practical, as argued in the following proposition.

Proposition 12 *The optimal integer solution of a discrete problem of the form 3.21 that is the discretized version of the dual 3.9, is always a basic variable in the LP relaxation of 3.21.*

Proof. Let x^*, θ^* be the optimal solution to 3.19. Consider $\gamma(\omega)^0$ to be the optimal solution to the integer version of 3.21 (implying $\alpha_0 = 1$). Let's proceed by contradiction.

If $\gamma(\omega)^0$ is not in the basis, its reduced cost is strictly greater than zero. So:

$$E[q(\gamma(\omega)^0)] + E[\gamma(\omega)^0] x^* < \theta^*. \quad (3.22)$$

Now for any variable $\gamma(\omega)^b$ that was included in the basis, we have:

$$E[q(\gamma(\omega)^b)] + E[\gamma(\omega)^b] x^* = \theta^*. \quad (3.23)$$

Substituting the value of θ^* from 3.23 in 3.22 and recalling that $E[q(\gamma(\omega)^i)] = E[p(x^i, \omega)] - E[\gamma(\omega)^i] x^i$, we can write:

$$E[p(x^0, \omega)] + E[\gamma(\omega)^0] (x^* - x^0) < E[p(x^b, \omega)] + E[\gamma(\omega)^b] (x^* - x^b), \quad (3.24)$$

but this contradicts the fact that x^0 is optimal. ■

3.7 Optimization with sampling, stochastic quasi-gradient methods

In the methods presented in this chapter we have assumed that is possible to solve the scenario problem for every possible scenario. Although this may be possible in particular cases, such as discrete probability distributions, it may be very difficult for more general distributions.

As mentioned above, one way to proceed with general distributions is to divide the scenario space into a finite number of subsets and then use the probability for each such subset to create a discrete distribution version of the original problem. Birge calls these the *stochastic equivalent* [BL97].

Another way to proceed is to try to combine sampling with the methods we have presented so far in order to avoid solving $x(\omega)$ and $y(\omega)$ for the entire scenario space. Instead we obtain at each step a sample of size n (usually about 20) and we use it to produce estimators of the quantities of interest, for example $\hat{\gamma} = E[\gamma(\omega)]$.

The methods that combine gradient and sampling techniques are called *Stochastic quasigradient methods* by Ermoliev [Erm88]. The methods that combine Bender's decomposition and sampling are *Stochastic decomposition methods* by Hige and Sen [HS96].

The convergence of methods of this kind is insured by showing that successive iterations form a *super-martingale*. Although martingale theory is beyond the scope of this thesis, it is the opinion of the author that future work in the area of optimization using sampling techniques should be subject to standards of this theory. Some interesting applications of martingale theory to optimization can be find in Polyak [Pol87] and in Motwani and Raghavan [MR95].

3.7.1 Stochastic quasigradient method

Here we present the stochastic quasigradient version of the gradient method for convex problems in the deterministic variable.

Step 0 Initialization. Find a feasible deterministic solution $x^0 \in X$. Make $k = 0$.

Step 1 Stochastic variables solution. At step k take a sample of size n of scenarios from Ω . For each scenario ω^i , $i = 1, \dots, n$ solve:

$$\begin{aligned} \min f(x^k, y(\omega^i), \omega^i) & \quad (3.25) \\ \text{s.t.} & \\ g(x^k, y(\omega^i), \omega^i) \leq 0 & \\ y(\omega^i) \in Y(\omega^i) & \end{aligned}$$

Obtain the subgradient $-\gamma(\omega^i)^k$.

Step 2 Subgradient estimation. Compute the estimator $\hat{\gamma}^k$ for the subgradient.

$$\hat{\gamma}^k = \frac{1}{n} \sum_{i=1}^n \gamma(\omega^i)^k \quad (3.26)$$

Step 3 Descent step. Find a new solution:

$$x^{k+1} = \arg \min_{x \in X} \left[\left(x^k + \frac{1}{k} \hat{\gamma}^k - x \right)^2 \right] \quad (3.27)$$

Step 4 Stopping rule. If $|\hat{\gamma}^k| \leq \epsilon$ stop the solution has been found. (ϵ is a suitable tolerance), else make $k := k + 1$, $n := n + 1$ and go to step 1.

To understand the way the method works we have to review a theorem cited by Ermoliev [Erm88] based in the research done by Polyak [Pol87]. The theorem is:

Theorem 13 *Assume that:*

1. $E[p(x, \omega)]$ is a convex continuous function.
2. X is a convex compact set.
3. The parameters $\rho_k = \frac{1}{k}$, and the sampling error ε_k are such:

$$\varrho_k \geq 0, \quad \sum_{k=1}^{\infty} \varrho_k = \infty, \quad \sum_{k=1}^{\infty} E \left[\varrho_k |\varepsilon_k| + \varrho_k^2 \|\hat{\gamma}^k\|^2 \right] < \infty \quad (3.28)$$

Then the stochastic quasigradient method converges to the optimal solution.

It is easy to see that the method presented in this section honors these conditions. Conditions 1 and 2 hold trivially as the first part and second part of condition 3. For the third part of condition 3, it can be seen from the central limit theorem of probability that for a large sample of size $n + k$ we would have:

$$E[|\varepsilon_k|] \leq \frac{\sigma}{\sqrt{2\pi\sqrt{n+k}}} \quad (3.29)$$

where σ is the upper bound of the standard deviation of $\{\varepsilon_k\}$. Then the convergence of the first term will depend of the convergence of the series:

$$\sum_{k=1}^{\infty} \frac{1}{k\sqrt{n+k}}. \quad (3.30)$$

The value of this series can be seen in table 3.7.1.

3.8 Chapter 3 summary

In this chapter, we have studied problems which are convex in the deterministic variable for all non-zero probability scenarios. This property makes it possible to find strong duals and the optimality condition $E[\gamma(\omega)] = 0$ makes the expected subgradient equal to zero at the optimal solution. This is a stochastic extension of deterministic convex programs for which the optimality condition is that the gradient

n	s
0	2.612
10	1.356
20	1.111
30	0.981
40	0.894
50	0.831
60	0.782
70	0.743
80	0.709
90	0.681
100	0.657

Table 3.1: Value of the series $\sum_{k=1}^{\infty} \frac{1}{k(n+k)^{1/2}}$

equal zero. So we extend the methods used to solve deterministic problems to solve the first stage of two stage stochastic programs. It is possible to decompose stochastic problems into two stages in order to facilitate their solution. The first method to do this that we examine is a gradient-like method in which new solutions are obtained by following the maximum descent direction. The second method is a column generation-like method in which we enumerate successive feasible solutions until we find one that satisfies the required tolerance. We show that this method can be easily extended to problems with discrete domain for the deterministic variables provided that the convex property holds. Finally, we show that it is possible to combine these methods with random sampling in order to make the computation of the scenario dependent problems easier.

Chapter 4

The average plan model for robust planning

4.1 Robust planning

Although for many reasons most optimization models are solved as deterministic problems, in practice it often is necessary to have solutions that somehow consider the uncertainties encountered operationally. The idea of robust planning is to generate solutions that somehow consider some degree of uncertainty in the planning data. For example, consider the routing and scheduling of a fleet of vehicles. The solution is a set of planned itineraries for all vehicles. Since there might be uncertainties that can lead to disruptions to the plan, the planner could give connection times and travel times larger than that required on average to ensure *robustness* of the solution, on one leg of an itinerary to ensure that a delay in the network will not propagate to the next legs.

There are two important characteristics of a robust plan:

1. The plan is not dynamic. In general, there is not a decision rule that tells us what to do in case a particular scenario is realized. Instead, we expect that our robust plan will be applicable most of the time. If an unforeseen condition occurs, we expect changes to the plan to be minimal.

2. A robust plan is suboptimal in the average scenario. Since the average plan has to be applicable with minimum changes to extreme conditions it probably may be different from the solution using average values for the data.

Thus, the job of the transportation scientist is to find an equilibrium between these two contradictory strategies, of minimizing the solution cost for the common occurrence cases and finding a solution for all cases, including rare ones. The first strategy is risky and the second is conservative.

4.2 The average plan model

We define the average plan model as:

$$\begin{aligned} \min E \left[c(\omega) x(\omega) + \frac{1}{2} (x(\omega) - \bar{x})' Q (x(\omega) - \bar{x}) \right] & \quad (4.1) \\ \text{s.t.} & \\ A(\omega) x(\omega) \geq b(\omega) & \quad \text{a.s.} \end{aligned}$$

where:

$\omega \in \Omega$ is the set of scenarios

\bar{x} is the average plan

Q is a positive definite matrix.

The idea is that if the planner knows with certainty the scenario that will be taking place, he/she will find an optimal plan $x(\omega)$ that solves the scenario LP problem:

$$\begin{aligned} \min c(\omega) x(\omega) & \quad (4.2) \\ \text{s.t.} & \\ A(\omega) x(\omega) \geq b(\omega) & \quad \text{a.s.} \end{aligned}$$

Since the resulting scenario is not known with certainty, we generate an *average*

plan \bar{x} . This plan constrains the solutions of the individual scenarios by means of the quadratic term in the objective function of 4.1. Thus, we have the desired effect: the individual scenario solutions for 4.1 are not optimal for problems 4.2, instead they are likely to be near the average plan \bar{x} .

A better understanding of the model can come from reformulating it by defining the variables:

$$y(\omega) = x(\omega) - \bar{x} \quad (4.3)$$

Then 4.1 is:

$$\begin{aligned} \min E [c(\omega)] \bar{x} + E \left[c(\omega) y(\omega) + \frac{1}{2} y(\omega)' Q y(\omega) \right] \quad (4.4) \\ \text{s.t.} \end{aligned}$$

$$A(\omega) (\bar{x} + y(\omega)) \geq b(\omega) \quad \text{a.s.}$$

$y(\omega)$ represents the difference between the actual solution for scenario ω and the average plan. It also can be interpreted as the actions necessary to *fix* the plan when it is known with certainty how it failed. The objective function minimizes the cost of the average plan plus the cost of fixing it.

4.3 The dual of the average plan

Before examining more deeply the optimality conditions for the average plan, let's study its dual using the theory developed in the last chapter. We are going to focus on problem 4.4. Its Lagrangian is:

$$L(\lambda(\omega)) = \inf_{\bar{x}} E \left[\min_{y(\omega) \in Y(\omega)} c(\omega) \bar{x} + c(\omega) y(\omega) + \frac{1}{2} y(\omega)' Q y(\omega) + \lambda(\omega) (b(\omega) - A(\omega) (\bar{x} + y(\omega))) \right] \quad (4.5)$$

where the average plan has taken the place of the first stage variable and the

correction $y(\omega)$ the place of the second stage.

Arranging terms:

$$L(\lambda(\omega)) = \inf_{\bar{x}} E \left[\begin{array}{l} \lambda(\omega) b(\omega) + (c(\omega) - \lambda(\omega) A(\omega)) \bar{x} + \\ \min_{y(\omega)} (c(\omega) - \lambda(\omega) A(\omega)) y(\omega) + \frac{1}{2} y(\omega)' Q y(\omega) \end{array} \right] \quad (4.6)$$

We see that the optimal correction is:

$$y(\omega) = -Q^{-1} (c(\omega) - \lambda(\omega) A(\omega))' \quad (4.7)$$

So the Lagrangian is:

$$L(\lambda(\omega)) = \inf_{\bar{x}} E \left[\begin{array}{l} \lambda(\omega) b(\omega) + (c(\omega) - \lambda(\omega) A(\omega)) \bar{x} + \\ -\frac{1}{2} (c(\omega) - \lambda(\omega) A(\omega)) Q^{-1} (c(\omega) - \lambda(\omega) A(\omega))' \end{array} \right] \quad (4.8)$$

To ensure that the Lagrangian has a finite solution we should have

$$E[c(\omega) - \lambda(\omega) A(\omega)] = 0. \quad (4.9)$$

Then the dual problem is:

$$\begin{array}{l} \max_{\lambda(\omega)} E \left[\lambda(\omega) b(\omega) - \frac{1}{2} (c(\omega) - \lambda(\omega) A(\omega)) Q^{-1} (c(\omega) - \lambda(\omega) A(\omega))' \right] \\ s.t. \end{array}$$

$$E[\lambda(\omega) A(\omega)] = E[c(\omega)]$$

$$\lambda(\omega) \geq 0 \quad a.s.$$

by virtue of the solution represented by equation 4.7 we see that the conditions of the strong duality theorem are honored so there is no duality gap.

4.4 Optimality conditions

Here we give the optimality theorem for the average plan.

Theorem 14 (Optimality) *The average plan \bar{x}^* is optimal if and only if:*

$$\bar{x}^* = E[x(\omega)^*] \quad (4.11)$$

and

$$x(\omega)^* = \arg \min_{A(\omega)x(\omega) \geq b(\omega)} c(\omega)x(\omega) + \frac{1}{2}(x(\omega) - \bar{x}^*)' Q(x(\omega) - \bar{x}^*). \quad (4.12)$$

Proof: By equation 4.12 we see that x^* is feasible in 4.1. Next, taking the mathematical expectation of equation 4.7 we obtain:

$$E[y(\omega)] = -Q^{-1}E[c(\omega) - \lambda(\omega)A(\omega)] = 0. \quad (4.13)$$

The equality to zero follows from equation 4.9. Condition 4.11 follows from 4.3 and 4.12 implies strong duality since it is a convex problem with no duality gap. Therefore $x(\omega)^*$ is optimal in 4.1 and \bar{x}^* , obtained by equation 4.11, is also optimal in 4.1.

Corollary 15 *Let \bar{x}^* and $y(\omega)^* = x(\omega)^* - \bar{x}^*$ be the optimal solution for 4.1, then we have:*

$$A(\omega)y(\omega)^* \geq -A(\omega)Q^{-1}c(\omega)' \quad a.s. \quad (4.14)$$

with probability one.

Proof: We have with probability one:

$$A(\omega)(\bar{x}^* + y(\omega)^*) \geq b(\omega) \quad a.s. .$$

And the second stage problem:

$$\begin{aligned} \min_{y(\omega)} c(\omega) y(\omega) + \frac{1}{2} y(\omega)' Q y(\omega) & \quad (4.15) \\ \text{s.t.} & \end{aligned}$$

$$A(\omega) y(\omega) \geq b(\omega) - A(\omega) \bar{x}^* \quad \text{a.s. .}$$

There are two cases: in the first, $-A(\omega) Q^{-1} c(\omega)' \geq b(\omega) - A(\omega) \bar{x}^* \quad \text{a.s.}$. So 4.15 has the interior point solution $y(\omega)^* = -Q^{-1} c(\omega)' \quad \text{a.s.}$. Thus $A(\omega) y(\omega)^* = -A(\omega) Q^{-1} c(\omega)' \quad \text{a.s.}$. In the second: $A(\omega) y(\omega) \geq b(\omega) - A(\omega) \bar{x}^* \geq -A(\omega) Q^{-1} c(\omega)' \quad \text{a.s.}$. From both cases, we see that 4.14 is always honored.

4.5 Dispersion of the solution

The quadratic term $E[y(\omega)' Q y(\omega)]$ in the solution shows how far the solution is from the feasibility condition $y(\omega) = 0 \quad \text{a.s.}$. The formal statement of this dependence can be obtained from Markov's inequality [MR95] for a nonnegative random variable $z(\omega)$. Markov's inequality states that for a positive real t it is always true that:

$$P(z(\omega) > t) < \frac{E[z(\omega)]}{t}. \quad (4.16)$$

Substituting the quadratic term for $z(\omega)$ we obtain:

$$P(y(\omega)' Q y(\omega) > t) < \frac{E[y(\omega)' Q y(\omega)]}{t}. \quad (4.17)$$

4.6 The stochastic quasigradient method for the average plan model

In this section we show how to adapt the method of section 3.7 to the average plan model. Some theoretical considerations, for example, the convexity of the average plan, we leave for the next chapter.

First we find the gradient for \bar{x} . It can be seen from the Lagrangian 4.6 that:

$$\nabla_{\bar{x}}L(\bar{x}, \lambda(\omega)) = E[(c(\omega) - \lambda(\omega) A(\omega))], \quad (4.18)$$

but by equation 4.7:

$$\nabla_{\bar{x}}L(\bar{x}, \lambda(\omega)) = E[(c(\omega) - \lambda(\omega) A(\omega))] = -QE[y(\omega)]. \quad (4.19)$$

So the gradient for the average plan is the expected value of the deviations from the plan times the parameter matrix $-Q$.

Now we can present the method.

Step 0 Initialization. Select an initial average plan \bar{x}^0 . Set the counter $k := 0$. Set an initial sample size n .

Step 1 Sampling. Obtain a sample $\omega^i \in \Omega$, $i = 1, \dots, n+k$ from the scenario set. For each sampled scenario, solve the plan deviation problem:

$$\begin{aligned} \min c(\omega^i) y(\omega^i) + \frac{1}{2} y(\omega^i)' Q y(\omega^i) \\ \text{s.t.} \\ A(\omega^i) y(\omega^i) \geq b(\omega^i) - A(\omega^i) \bar{x} \\ y(\omega^i) \in R^n. \end{aligned} \quad (4.20)$$

Step 2 Descent direction estimation. Obtain the average of the plan deviations:

$$\hat{y}^k = \frac{1}{n+k} \sum_{i=1}^{n+k} y(\omega^i). \quad (4.21)$$

Step 3 Plan update Find the average plan:

$$\bar{x}^{k+1} = \bar{x}^k + \frac{1}{k} Q \hat{y}^k. \quad (4.22)$$

Step 4 Stopping rule If $|\hat{y}^k| < \epsilon$, where ϵ is a tolerance, stop; the solution has been

achieved, otherwise go to step 1.

The conditions for the convergence of the method have been discussed in section 3.7. Notice that this method diverges from the method in section 3.7 in step 3 . We do not have to project the new solution to a constraint set since we passed all the constraints in the plan to the second stage problem, making the solution update a very easy step.

4.7 The average plan model for multicommodity flows on networks with random capacity and demand

One of the fundamental problems underlying transportation and logistical analysis is the multicommodity flow problem. It requires the determination of an optimal sharing of network resources between several commodities competing for those resources. A discussion and deterministic formulations of the problem can be found in Barnhart in [Bar93] and in [BHV96], and in Ahuja, Magnanti and Orlin [AMO93].

In many logistical applications involving multicommodity flows, there are uncertainties, the most common ones are in the capacity of the network and in the demand on the nodes. The average plan model 4.1 is well suited to model and solve uncertainties in network models, such as the multicommodity flow problem. In this section, we show that the route generation (column generation) methodology can be applied to the average plan formulation for multicommodity flows.

Let $G = (N, A)$ be a network, where N represents the set of locations and A represents the arc set of transport between the locations. The average plan formulation of a multicommodity flow problem for the scenario set $\Omega \ni \omega$ and K commodities is:

$$\begin{aligned} \min E \left[\sum_{k=1}^K \left(c(\omega)_k x(\omega)_k + \frac{q}{2} (x(\omega)_k - \bar{x}_k)^2 \right) \right] \quad (4.23) \\ \text{s.t.} \end{aligned}$$

$$Nx(\omega)_k = d(\omega)_k \quad k = 1, \dots, K \quad \text{a.s.}$$

$$\begin{aligned} \sum_{k=1}^K x(\omega)_k &\leq v(\omega) \quad a.s. \\ x(\omega)_k &\geq 0 \quad k = 1, \dots, K \quad a.s.. \end{aligned} \quad (4.24)$$

Where:

\bar{x}_k is the vector of the expected arc flows of commodity k ;

$x(\omega)_k$ is the vector of the arc flows of commodity k in scenario ω ;

$v(\omega)$ is the vector of arc capacities in scenario ω ;

$d(\omega)_k$ is the supply/demand at nodes of commodity k in scenario ω ;

$c(\omega)_k$ is the cost of transporting commodity k in scenario ω ; and

N is a full rank node-arc incidence matrix.

In the following, we examine how to solve problem 4.23 for a fixed value of \bar{x}_k . In doing this, we learn how to solve the scenario dependent average plan multicommodity flow model.

In the deterministic case, multicommodity flow problems are commonly solved iteratively. One approach is to relax constraints 4.24, thereby decomposing the resulting problem into several shortest path problems, one for each commodity. We can use a similar approach for model 4.23, the Lagrangian is:

$$\begin{aligned} L(\bar{x}_k, \lambda(\omega)_k) &= E \left[\sum_{k=1}^K \min_{\substack{N x(\omega)_k = d(\omega)_k \\ x(\omega)_k \geq 0}} \left(c(\omega)_k x(\omega)_k + \frac{q}{2} (x(\omega)_k - \bar{x}_k)^2 \right) + \right. \\ &\quad \left. \sum_{k=1}^K \lambda(\omega)_k (x(\omega)_k - v(\omega)) \right] \\ \lambda(\omega)_k &\geq 0 \quad k = 1, \dots, K \quad a.s.. \end{aligned} \quad (4.25)$$

We solve 4.25 by solving $|K| |\Omega|$ independent problems of the form:

$$\min_{\substack{N x(\omega)_k = d(\omega)_k \\ x(\omega)_k \geq 0}} \left(c(\omega)_k + \lambda(\omega)_k \right) x(\omega)_k + \frac{q}{2} (x(\omega)_k - \bar{x}_k)^2. \quad (4.26)$$

In the remaining part of this section we assume that each commodity has a single origin and destination. If $q = 0$, 4.26 is a shortest path problem. For a general $q > 0$, 4.26 is not a shortest path problem, instead, it is a projection problem (see [Ber95]). One way to find the solution for problem 4.26 is to use the convex combinations method presented by Sheffi [She85]. This method is an adaptation of the gradient method to solve network problems.

Assuming that it is possible to enumerate all possible routes between the origin and destination of commodity k in scenario ω a LP can be used to solve problem 4.26. Let $P(\omega)_k$ be a matrix whose columns are the feasible routes for commodity k in scenario ω . Then the flow can be represented as:

$$x(\omega)_k = P(\omega)_k \delta(\omega)_k, \quad (4.27)$$

where $\delta(\omega)_k$ is a vector whose elements are the flows in each route. We obtain the optimal flow by solving:

$$\max \Delta \quad (4.28)$$

s.t.

$$P(\omega)_k' (c(\omega)_k + \lambda(\omega)_k)' - qP(\omega)_k' \bar{x}_k + P(\omega)_k' P(\omega)_k \delta(\omega)_k \geq u\Delta \quad (4.29)$$

$$u'\delta(\omega)_k = \bar{d}(\omega)_k \quad (4.30)$$

$$\delta(\omega)_k \geq 0.$$

Where:

Δ is a scalar representing the marginal cost of used routes;

$\delta(\omega)_k$ are the flows in the feasible routes for commodity k in scenario ω .

$\bar{d}(\omega)_k$ is a scalar with the amount of commodity k in scenario ω ;

u is a vector of ones of dimension equal to the number of paths.

Problem 4.28 is based in the results presented by Bertsekas in [Ber95] and in

[BG92] about optimization over simplexes applied to network flow problems. It reflects the very well known result that commodities use routes with the same marginal cost while ignoring routes with a greater marginal cost. Constraints 4.29 represent the marginal cost of the commodity for each route.

In order to obtain the optimal multipliers $\lambda(\omega)_k$ we need a master program to use with the column generation step. Let $x(\omega)_k^i$ be the routing generated in the i^{th} iteration and $g(\omega)_k^i$ its cost, then the master LP is:

$$\min E \left[\sum_{k=1}^K \sum_{j=1}^i g(\omega)_k^j \quad f(\omega)_k^i \right] \quad (4.31)$$

s.t.

$$\sum_{k=1}^K \sum_{j=1}^i x(\omega)_k^j \quad f(\omega)_k^j \leq v(\omega) \quad a.s. \quad (4.32)$$

$$\sum_{j=1}^i f(\omega)_k^j = 1 \quad a.s.$$

$$f(\omega)_k^j \geq 0 \quad k = 1, \dots, K \quad j = 1, \dots, i \quad a.s..$$

$\lambda(\omega)_k$ is the dual solution for constraints 4.32.

The solution of the average plan multicommodity flow problem 4.23 for a fixed \bar{x}_k iterates 4.28 and 4.31. This shows that the average plan version of the multicommodity flow problem is not intrinsically more difficult than the linear, deterministic version.

We have shown in this section how to implement the scenario dependent problem for multicommodity flows needed in all solution methods of the average plan model presented in this thesis.

4.8 Chapter 4 summary

In this chapter we present the average plan model, that incorporates stochasticity to a plan that in the deterministic case would be represented by a LP. The main idea behind the model is that deviations from the plan are penalized by a quadratic term. We use the Lagrangian methods of chapter 2 to obtain the average plan dual and from

it, we deduce the optimality condition $E [y (\omega)] = 0$, meaning that the expected value of plan deviations should be zero. We adapt and apply the stochastic quasigradient method of chapter 3 to obtain the solution of the average plan model. Finally, we use the theory of chapter 2 to find a column generation solution of the average model for multicommodity flow problems.

Chapter 5

The generalized average plan model

5.1 Planning models for random scenarios

In this chapter we expand the average plan model of chapter 4 to include plans that can not be formulated as linear programming problems in their deterministic form. We show in this chapter that a fairly large class of models such as IP's or NLP's can be cast as average plan models.

The approach followed here is similar to the one proposed by Richetta and Odoni [RO93]. We know that there is a set of scenarios Ω . If we know in advance with certainty which is the scenario ω that will occur, we can determine an optimal plan $x(\omega)$ that is the solution of

$$\begin{aligned} \min f(x(\omega), \omega) & \qquad (5.1) \\ \text{s.t.} & \\ x(\omega) \in X(\omega). & \end{aligned}$$

Nevertheless we don't have certainty about the scenario that will occur, just its probability distribution. The Richetta approach is to find a plan x that fits all scenarios

$\omega \in \Omega$, and solves the problem:

$$\begin{aligned} \min E [f(x, \omega)] & \quad (5.2) \\ \text{s.t.} & \\ x \in \bigcap_{\omega \in \Omega} X(\omega). & \end{aligned}$$

Unfortunately, as mentioned in the previous chapter, there are two important setbacks. First there may not be a plan x that is feasible for all scenarios. Second, if x exists it may be constrained by a very low probability set $X(\omega)$ that may have very large costs $f(x, \omega)$ associated with it, making the plan too costly, although feasible.

Another approach is to allow an *acceptable risk* of selecting an infeasible plan. The planner establishes a level of risk of infeasibility that he is willing to take in order to obtain a more economical plan. This can be modeled as the chance constraint program:

$$\min E [f(x(\omega), \omega)] \quad (5.3)$$

s.t.

$$P(x(\omega) = x) \geq \alpha \quad (5.4)$$

$$x(\omega) \in X(\omega) \quad \text{a.s.}$$

$$x \in R^n.$$

Here constraint 5.4 establishes that the plan x is feasible at least α percent of the time. However with this model it is desirable to find the maximum level α that makes the plan feasible; since it may be infeasible for high values of this constant.

In this chapter we present an alternative model called the *Generalized Average Plan Model* with the goal of overcoming the shortcomings of the previous two models. It is:

$$\min E \left[f(x(\omega), \omega) + \frac{1}{2} (x(\omega) - \bar{x})' Q (x(\omega) - \bar{x}) \right] \quad (5.5)$$

s.t.

$$x(\omega) \in X(\omega) \quad a.s.$$

$$\bar{x} \in R^n.$$

As in chapter 4, the rationale of this model is that for large values of Q (a positive definite matrix), the optimal solution of 5.5 will try to minimize the quadratic term by selecting values of $x(\omega)$ that are close to \bar{x} . A deeper insight to this effect will be presented in the next section.

5.2 The generalized average plan model as a quadratic dual

We can rewrite the planning model 5.2 as:

$$\begin{aligned} & \min E [f(x(\omega), \omega)] && (5.6) \\ & s.t. \\ & x(\omega) = \bar{x} \quad a.s. \\ & x(\omega) \in X(\omega) \quad a.s. \\ & \bar{x} \in R^n \end{aligned}$$

Assume that 5.6 has an optimal solution $x(\omega)^*$, \bar{x}^* , otherwise the value of the objective function will be $+\infty$. Then, we have the following inequality:

$$\min_{x(\omega) \in X(\omega)} f(x(\omega), \omega) + \frac{1}{2} (x(\omega) - \bar{x}^*)' Q (x(\omega) - \bar{x}^*) \leq f(x(\omega)^*, \omega) \quad a.s. \quad (5.7)$$

This holds because $x(\omega)^*$ is a feasible solution to the left hand side of the inequality. Since 5.7 holds in the almost surely sense, it is possible to take the mathematical expectation in both sides of the inequality and optimizing over \bar{x} we obtain:

$$\min_{\bar{x} \in R^n} E \left[\min_{x(\omega) \in X(\omega)} f(x(\omega), \omega) + \frac{1}{2} (x(\omega) - \bar{x})' Q (x(\omega) - \bar{x}) \right] \leq E [f(x(\omega)^*, \omega)]. \quad (5.8)$$

So the average plan is always a lower bound on the optimal value of the planning model represented by 5.6. Now, if we want a strong lower bound we should find a value of the parametric matrix Q that makes the left hand side as big as possible. Consider \mathfrak{S} to be the set of all positive definite matrices of size n (size of the vector of variables). Then the optimal lower bound is:

$$\max_{Q \in \mathfrak{S}} \min_{\bar{x} \in \mathbb{R}^n} E \left[\min_{x(\omega) \in X(\omega)} f(x(\omega), \omega) + \frac{1}{2} (x(\omega) - \bar{x})' Q (x(\omega) - \bar{x}) \right] \leq E[f(x(\omega)^*, \omega)]. \quad (5.9)$$

We conclude with the following definition:

Definition 1 (Quadratic dual) *The quadratic dual of the planning model 5.6 is the optimization problem:*

$$\max_{Q \in \mathfrak{S}} \min_{\bar{x} \in \mathbb{R}^n} E \left[\min_{x(\omega) \in X(\omega)} f(x(\omega), \omega) + \frac{1}{2} (x(\omega) - \bar{x})' Q (x(\omega) - \bar{x}) \right] \quad (5.10)$$

5.3 Optimality and strong duality of the quadratic dual

We have shown that the generalized average plan is a dual for the program 5.6, in a non-Lagrangian sense. Shapiro [Sha79] examines the theory of generalized duals and finds that the optimality conditions are similar to those for Lagrangian duals. Here we examine the optimality and strong duality issue applied to the generalized average plan.

The optimality conditions are established by the following proposition:

Proposition 16 (Optimality conditions for the quadratic dual) *Let $x(\omega)^*$, \bar{x}^* , Q^* be an optimal solution to the quadratic dual 5.10 such that:*

$$E \left[(x(\omega)^* - \bar{x}^*)' Q^* (x(\omega)^* - \bar{x}^*) \right] = 0. \quad (5.11)$$

Then it is optimal to problem 5.6 and there is no duality gap.

Proof. Since Q^* is positive definite, condition 5.11 implies:

$$(x(\omega)^* - \bar{x}^*)' Q^* (x(\omega)^* - \bar{x}^*) = 0 \quad a.s. \quad (5.12)$$

and thus:

$$x(\omega)^* = \bar{x}^* \quad a.s. \quad (5.13)$$

This implies that $x(\omega)^*$, \bar{x}^* is feasible in 5.6. Substituting condition 5.11 in inequality 5.8, we find:

$$E[f(x(\omega)^*, \omega)] \leq \min_{\substack{x(\omega) = \bar{x} \\ x(\omega) \in X(\omega), \\ \bar{x} \in R^n}} \min_{a.s.} E[f(x(\omega), \omega)] \quad (5.14)$$

So $x(\omega)^*$, \bar{x}^* is optimal in 5.6 and therefore there is no duality gap. ■

5.4 Solution of the generalized average plan model for a fixed value of Q

We are interested in solving the generalized average plan model 5.5 for a fixed value of Q to obtain some feasible plans for different scenarios. In order to obtain this solution we are going to apply the theory developed in chapter 3 for stochastic programs that are convex in the deterministic variable. In this case, the first stage deterministic variable is the average plan \bar{x} and the second stage variables are the plans for the specific scenarios $x(\omega)$. First we have to show that the generalized average plan model is convex in the deterministic variable as defined by equation 3.3.

Theorem 17 (Subgradients for the average plan) *Let $f(x(\omega), \omega)$ be convex on $X(\omega)$ with probability one, and $X(\omega)$ compact with probability one. Then the function*

$$p(\bar{x}, \omega) = \min_{x(\omega) \in X(\omega)} f(x(\omega), \omega) + \frac{1}{2} (x(\omega) - \bar{x})' Q (x(\omega) - \bar{x}) \quad (5.15)$$

is convex in the deterministic variable \bar{x} and

$$\gamma(\omega) = -(x(\omega) - \bar{x})' Q \quad (5.16)$$

is always a subgradient.

Proof. Consider an average plan \bar{x}^0 and its corresponding second stage decision $x(\omega)^0$. Then

$$p(\bar{x}^0, \omega) = f(x(\omega)^0, \omega) + \frac{1}{2} (x(\omega)^0 - \bar{x}^0)' Q (x(\omega)^0 - \bar{x}^0). \quad (5.17)$$

Let $\varepsilon(\omega)^0$ be a subgradient in the almost surely sense for $f(x(\omega), \omega)$ at $x(\omega)^0$ then $\delta(\omega)^0 = \varepsilon(\omega)^0 + (x(\omega)^0 - \bar{x}^0)' Q$ is a subgradient for the function $f(x(\omega)^0, \omega) + \frac{1}{2} (x(\omega)^0 - \bar{x}^0)' Q (x(\omega)^0 - \bar{x}^0)$ respect to the variable $x(\omega)$ since the second term is the subgradient for the quadratic term. Furthermore since $x(\omega)^0$ is optimal for \bar{x}^0 in the scenario ω we have

$$\begin{aligned} \delta(\omega)^0 (x(\omega) - x(\omega)^0) &\geq 0 \\ \forall x(\omega) \in X(\omega) &\quad a.s. \end{aligned} \quad (5.18)$$

To show convexity on the average plan, we analyze the value of $p(\bar{x}^0, \omega) + \gamma(\omega)^0 (\bar{x} - \bar{x}^0)$. Using 5.18 we have

$$p(\bar{x}^0, \omega) + \gamma(\omega)^0 (\bar{x} - \bar{x}^0) \leq p(\bar{x}^0, \omega) + \delta(\omega)^0 (x(\omega) - x(\omega)^0) + \gamma(\omega)^0 (\bar{x} - \bar{x}^0). \quad (5.19)$$

Developing the right hand side:

$$\begin{aligned} &f(x(\omega)^0, \omega) + \frac{1}{2} (x(\omega)^0 - \bar{x}^0)' Q (x(\omega)^0 - \bar{x}^0) \\ &+ \left[\varepsilon(\omega)^0 + (x(\omega)^0 - \bar{x}^0)' Q \right] (x(\omega) - x(\omega)^0) \\ &- (x(\omega)^0 - \bar{x}^0)' Q (\bar{x} - \bar{x}^0). \end{aligned} \quad (5.20)$$

Regrouping terms we have:

$$\begin{aligned}
& f(x(\omega)^0, \omega) + \varepsilon(\omega)^0 (x(\omega) - x(\omega)^0) \\
& + \frac{1}{2} (x(\omega)^0 - \bar{x}^0)' Q (x(\omega)^0 - \bar{x}^0) \\
& + (x(\omega)^0 - \bar{x}^0)' Q [(x(\omega) - \bar{x}) - (x(\omega)^0 - \bar{x}^0)].
\end{aligned} \tag{5.21}$$

We can add and subtract $\frac{1}{2} [(x(\omega) - \bar{x}) - (x(\omega)^0 - \bar{x}^0)]' Q [(x(\omega) - \bar{x}) - (x(\omega)^0 - \bar{x}^0)]$ to this expression and rearrange the squared expressions to obtain:

$$\begin{aligned}
& f(x(\omega)^0, \omega) + \varepsilon(\omega)^0 (x(\omega) - x(\omega)^0) \\
& + \frac{1}{2} (x(\omega) - \bar{x})' Q (x(\omega) - \bar{x}) \\
& - \frac{1}{2} [(x(\omega) - \bar{x}) - (x(\omega)^0 - \bar{x}^0)]' Q [(x(\omega) - \bar{x}) - (x(\omega)^0 - \bar{x}^0)].
\end{aligned} \tag{5.22}$$

But this is always less or equal to:

$$f(x(\omega), \omega) + \frac{1}{2} (x(\omega) - \bar{x})' Q (x(\omega) - \bar{x}) \tag{5.23}$$

because $\varepsilon(\omega)^0$ is a subgradient to $f(x(\omega), \omega)$ and the fourth term of 5.22 is always negative. Selecting \bar{x}^1 and its corresponding second stage solution $x(\omega)^1$ we obtain :

$$p(\bar{x}^0, \omega) + \gamma(\omega)^0 (\bar{x}^1 - \bar{x}^0) \leq p(\bar{x}^1, \omega) \quad a.s.. \tag{5.24}$$

So the average plan model is convex in the deterministic variable. ■

Now that we have found the expression for the subgradient of function 5.15, we can find the optimality conditions for the generalized average plan model. The argument, as in chapter 3, is to find a first stage variable such that the expected subgradient is zero, as is shown in the following theorem.

Theorem 18 (Optimality conditions) *Consider the generalized average plan model 5.5, if $f(x(\omega), \omega)$ has a subgradient for all $x(\omega) \in X(\omega)$ with probability one, then*

\bar{x}^* and $x(\omega)^*$ are optimal in 5.5 if and only if:

$$\bar{x}^* = E[x(\omega)^*]. \quad (5.25)$$

Proof. We can write the generalized average plan model as:

$$\min_{\bar{x} \in R^n} E \left[\min_{x(\omega) \in X(\omega)} \underset{a.s.}{f(x(\omega), \omega)} + \frac{1}{2} (x(\omega) - \bar{x})' Q (x(\omega) - \bar{x}) \right] = \min_{\bar{x} \in R^n} E[p(\bar{x}, \omega)]. \quad (5.26)$$

Since $f(x(\omega), \omega)$ is convex over the dominion, inequality 5.24 is valid and then

$$E[p(\bar{x}^*, \omega)] + E[\gamma(\omega)^*](\bar{x} - \bar{x}^*) \leq E[p(\bar{x}, \omega)] \quad (5.27)$$

is also valid. But at \bar{x}^* , $E[\gamma(\omega)^*] = E[x(\omega)^* - \bar{x}^*]Q = 0$ by condition 5.25 and by 5.16. Thus \bar{x}^* and $x(\omega)^*$ are optimal. ■

As shown, the optimality condition 5.25 is similar to condition 4.15 of chapter 4. This is not surprising since the linear average plan model 4.1 is a special case of the generalized average plan model 5.5.

5.4.1 Gradient method

Now that we have found the subgradient and the optimality conditions for the generalized average plan model we can apply the methods obtained in section 3.5. As was shown there, we can separate the second stage variable solution (meaning the scenario dependent plan) from the solution of the average plan. Then we can calculate a descent direction and update the first stage variable. The algorithm is as follows:

Step 0 Initialization. Let $k = 0$. Select an initial average plan \bar{x}^0 .

Step 1 Scenario solution. For every scenario $\omega \in \Omega$ solve:

$$x(\omega)^k = \arg \min_{x(\omega) \in X(\omega)} f(x(\omega), \omega) + \frac{1}{2} (x(\omega) - \bar{x}^k)' Q (x(\omega) - \bar{x}^k). \quad (5.28)$$

Step 2 Average plan update. Find the new average plan:

$$\bar{x}^{k+1} = \bar{x}^k + \frac{1}{k} Q \left(E \left[x(\omega)^k \right] - \bar{x}^k \right). \quad (5.29)$$

Step 3 Stopping rule. If $\left| \left(E \left[x(\omega)^k \right] - \bar{x}^k \right) \right| < \epsilon$, where ϵ is a tolerance, stop since the solution has been found, else go to step 1.

Since this algorithm is the same as the one of section 3.5.1, it's convergence is guaranteed by similar arguments.

5.4.2 Column generation method

As in section 3.5, we use the column generation method as an alternative to gradient methods and as a tool to methodically enumerate possible solution plans. In this case we define column i as:

$$E \left[\gamma(\omega)^i \right] = Q \left(E \left[x(\omega)^i - \bar{x}^i \right] \right). \quad (5.30)$$

We also define the value of the function $q \left(\gamma(\omega)^i \right)$ as:

$$q \left(\gamma(\omega)^i \right) = p \left(\bar{x}^i, \omega \right) - \gamma(\omega)^i \bar{x}^i. \quad (5.31)$$

With these two definitions we can write the column generation method for the average plan model.

Step 0 Initialization. Let $k = 0$. Select an initial average plan \bar{x}^0 .

Step 1 Pricing problem. For all scenarios $\omega \in \Omega$, solve 5.28. Obtain the new column using 5.30 and its associated cost $E \left[q \left(\gamma(\omega)^i \right) \right]$ using 5.31.

Step 2 Master problem. Solve for the variables α^i , $i = 1, \dots, k$ from the linear programming problem:

$$\min \sum_{i=1}^k E \left[q \left(\gamma(\omega)^i \right) \right] \alpha^i \quad (5.32)$$

$$\begin{aligned}
& \text{s.t.} \\
& \sum_{i=1}^k E \left[\gamma(\omega)^i \right] \alpha^i = 0 \\
& \sum_{i=1}^k \alpha^i = 1 \\
& \alpha^i \geq 0, \quad i = 1, \dots, k.
\end{aligned} \tag{5.33}$$

The dual variable associated with the constraint 5.33 is the new plan \bar{x}^{k+1} .

Step 3 Stopping rule. If $\left| E \left[\gamma(\omega)^i \right] \right| < \epsilon$ where ϵ is a tolerance, stop since the solution has been found, else go to step 1.

5.4.3 Stochastic quasigradient method

Following section 3.7, we can write a solution method for the generalized average plan model based on using statistical samples to avoid the work of enumerating all scenarios.

Step 0 Initialization. Set $k = 0$ and select an initial sample size n . Select an initial average plan \bar{x}^0 .

Step 1 Scenario sampling. Take a sample of size n of scenarios from Ω . For each sampled scenario ω^i solve the problem:

$$x(\omega^i)^k = \arg \min_{x(\omega^i) \in X(\omega^i)} f(x(\omega^i), \omega^i) + \frac{1}{2} (x(\omega^i) - \bar{x}^k)' Q (x(\omega^i) - \bar{x}^k). \tag{5.34}$$

Step 2 Solution update. Obtain the new average plan using:

$$\bar{x}^{k+1} = \bar{x}^k + \frac{Q}{k} \left(\frac{1}{n} \sum_{i=1}^n x(\omega^i) - \bar{x}^k \right). \tag{5.35}$$

Step 3 Stopping rule. If $\left| \frac{1}{n} \sum_{i=1}^n x(\omega^i) - \bar{x}^k \right| < \epsilon$ where ϵ is a tolerance, stop since the solution has been found, else let $k := k + 1$, $n = n + 1$ and go to step 1.

5.5 Applying the generalized average plan model to discrete problems

Many transportation problems, such as crew scheduling vehicle routing and network design, are well represented by integer programming models. These models are in general non-convex and therefore simple gradient methods may not converge to the global optimum. Instead it is necessary to do a more thorough exploration of the decision space to obtain a global optimum or at least a good local optimum. Furthermore, the average plan \bar{x} as defined by 5.5 is not itself guaranteed to be a feasible plan since it may not be an element of the feasible decision space. Nevertheless the interpretation of the value \bar{x} , although problem dependent, may be interesting and will give insight into the problem. A more detailed discussion follows.

5.5.1 Non-convexity of discrete problems

Consider the following small example. Considering a scenario ω^0 , we have the second stage problem:

$$p(\bar{x}, \omega^0) = \min_{x(\omega^0) \in \{0,1\}} x(\omega^0) + \frac{4}{2} (x(\omega^0) - \bar{x})^2. \quad (5.36)$$

A plot of this function can be seen in figure 5-1. As shown, there is obvious non-convexity at $\bar{x} = \frac{3}{4}$. The reason for this is that at this point there is no subgradient $\delta(\omega^0) = \varepsilon(\omega^0) + 4(x(\omega^0) - \frac{3}{4})$ such that inequality 5.18 is satisfied. For example, for $x(\omega^0) = 0$ we should have $\varepsilon(\omega^0) + 4(x(\omega^0) - \frac{3}{4}) \geq 0$, therefore $\varepsilon(\omega^0) \geq 3$. But such a subgradient doesn't exist for $f(x(\omega^0), \omega^0) = x(\omega^0)$. In the same way, we can see that for $x(\omega^0) = 1$, we should have $\varepsilon(\omega^0) \leq -1$ but this subgradient doesn't exist either. This is not the case for a problem with continuous variables such (see Figure 5-2) as:

$$p(\bar{x}, \omega^0) = \min_{x(\omega^0) \in [0,1]} x(\omega^0) + \frac{4}{2} (x(\omega^0) - \bar{x})^2. \quad (5.37)$$

The reason for this phenomenon is the behavior of the quadratic term when the

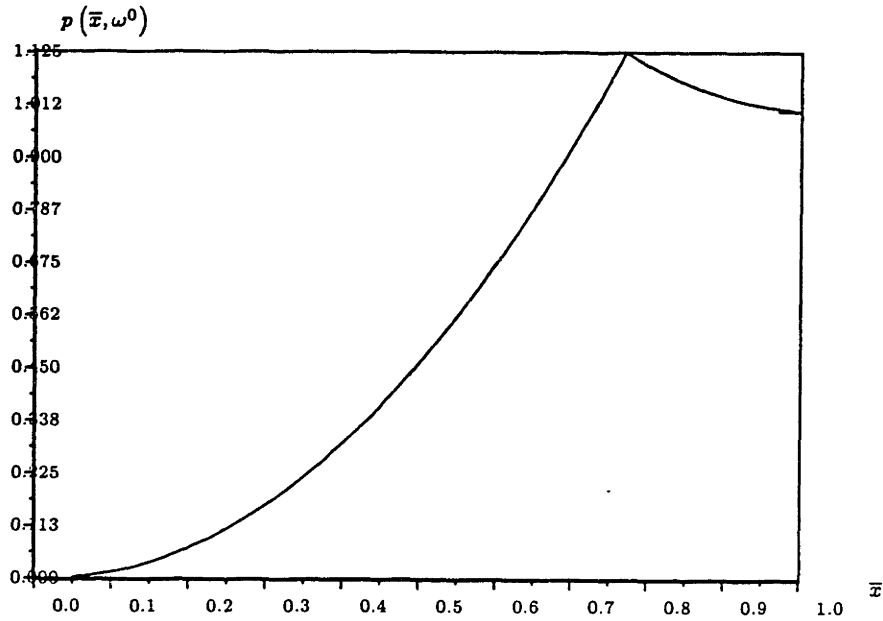


Figure 5-1: A non-convex problem

second stage variable is discrete. Consider the function

$$\min_{x(\omega) \in \{0,1\}} \frac{1}{2} (x(\omega) - \bar{x})^2, \quad (5.38)$$

that is plotted in figure 5-3. There is obvious non-convexity at $\bar{x} = \frac{1}{2}$.

5.5.2 Solving 0-1 problems by thorough exploration of the domain

In this subsection, we will concentrate on models in which the second stage solutions are constrained to have binary values, this is, $X(\omega) \subset \{0,1\}^n$. Their average plan solution has a very interesting interpretation. By the optimality condition 5.25, we

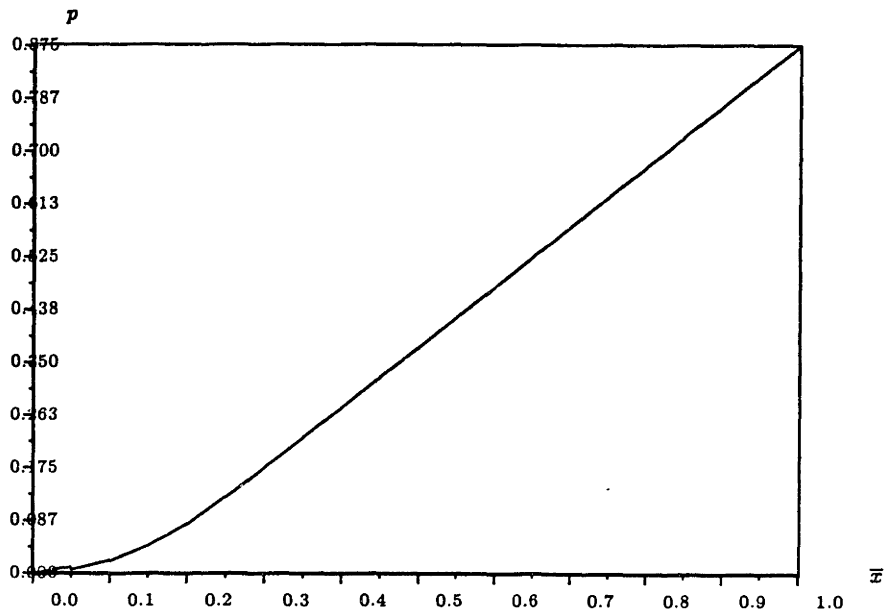


Figure 5-2: The same problem with a continuous variable

have:

$$\bar{x} = E[x(\omega)] = \int_{\Omega} x(\omega) dP(\omega) = \begin{bmatrix} P(x_1(\omega) = 1) \\ P(x_2(\omega) = 1) \\ \dots \\ P(x_n(\omega) = 1) \end{bmatrix}. \quad (5.39)$$

That means that the average plan represents the probability that each individual variable equals one, when the scenario ω occurs. In the case that the problem under study is a network design problem the average plan solution will represent the probability that a given link will be included under the actual scenario in which the network will operate.

The lack of convexity of these kinds of problems makes them particularly challenging, because we can use our subgradient to guide us to a nearby local optima but we don't have any information about where the global optimum is. Neither do we have a lower bounding technique that would allow us to enumerate efficiently the solutions.

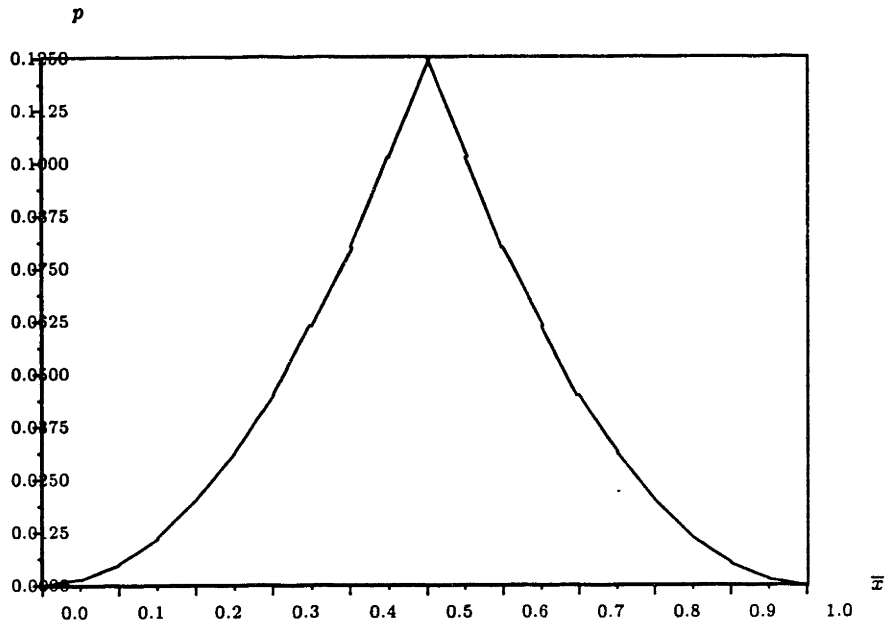


Figure 5-3: A plot of the quadratic term

The resulting approach is to enumerate a list of candidate solutions, evaluate them, refine them, and select the best one. Fortunately we can do evaluation and refinement in a fairly efficient way. The algorithm is as follows:

Step 0 Initialization. Make a list of candidate solutions $\bar{x}^k \in [0, 1]^n$, $k = 1, \dots, K$.

Step 1 First evaluation. For each member of the list solve the second stage problem:

$$x(\omega)^k = \arg \min_{x(\omega) \in X(\omega)} f(x(\omega), \omega) + \frac{1}{2} (x(\omega) - \bar{x}^k)' Q (x(\omega) - \bar{x}^k). \quad (5.40)$$

Step 2 Refinement. Obtain for all k , the refined average plan

$$\bar{x}_R^k = E [x(\omega)^k]. \quad (5.41)$$

Step 3 Second evaluation. Solve for all k , the problem:

$$z^k = E \left[f \left(x(\omega)^k, \omega \right) + \frac{1}{2} \left(x(\omega)^k - \bar{x}_R^k \right)' Q \left(x(\omega)^k - \bar{x}_R^k \right) \right]. \quad (5.42)$$

Step 4 Plan selection. Select:

$$z^* = \min_{k=1, \dots, K} z^k. \quad (5.43)$$

The corresponding refined plan \bar{x}_R^* is the solution.

The idea behind this algorithm is that even if we start with a value \bar{x}^k that is not a local optimum for the discrete average plan 5.5, we can find an approximation to the corresponding local optimum in a single step. Because once we have fixed the second stage variable $x(\omega)^k$ we only need to evaluate the optimality condition to obtain the local optimal average plan. We can show it in the following lemma.

Lemma 19 *Let $\bar{x}^k \in [0, 1]^n$ be any average plan, then \bar{x}_R^k obtained by using 5.40 and 5.41 is the corresponding local optimum.*

Proof. From 5.40 we can obtain the second stage solution $x(\omega)^k$: Then we obtain the local optimum by solving:

$$\min_{\bar{x}_R^k \in R^n} E \left[f \left(x(\omega)^k, \omega \right) + \frac{1}{2} \left(x(\omega)^k - \bar{x}_R^k \right)' Q \left(x(\omega)^k - \bar{x}_R^k \right) \right]. \quad (5.44)$$

But since \bar{x}_R^k is an unconstrained vector of real numbers we can obtain it just by finding the derivative and making it equal to zero:

$$E \left[\left(x(\omega)^k - \bar{x}_R^k \right)' Q \right] = 0, \quad (5.45)$$

thereby obtaining 5.41. ■

An heuristic that can be used to avoid the process of generating several trial solutions and then evaluating them, is to use instead a relaxed version of the discrete

problem:

$$\begin{aligned} & \min E \left[f(x(\omega), \omega) + \frac{1}{2} (x(\omega) - \bar{x})' Q (x(\omega) - \bar{x}) \right] & (5.46) \\ & \text{s.t.} \\ & x(\omega) \in \text{conv}(X(\omega)) \quad \text{a.s.} \\ & \bar{x} \in R^n \quad , \end{aligned}$$

in which the convex hull of the feasible second stage solution is used instead of the discrete set. This problem can be solved using any of the algorithms of section 5.4. Then we can use the refinement algorithm of this section to obtain the corresponding local optimum. Since the initial solution was the global optimum for the relaxed problem, there is at least some hope that the refined solution will also be the global optimum.

5.5.3 An iteration method

The previous method is a way to explore quickly a great number of starting points and evaluate them. If we commit to a particular starting point and we want to find the corresponding local optima we have to use an iteration method that will take us there. The following method works with problems with discrete variables as well as with continuous variables and mixed discrete-continuous variables.

Step 0 Initialization. Let $k = 0$. Select an initial average plan \bar{x}^0 .

Step 1 Scenario solution. For every scenario $\omega \in \Omega$ solve:

$$x(\omega)^k = \arg \min_{x(\omega) \in X(\omega)} f(x(\omega), \omega) + \frac{1}{2} (x(\omega) - \bar{x}^k)' Q (x(\omega) - \bar{x}^k). \quad (5.47)$$

Step 2 Solution update. Do:

$$\bar{x}^{k+1} = E [x(\omega)^k]. \quad (5.48)$$

Step 3 Stopping rule. If $\left|(\bar{x}^{k+1} - \bar{x}^k)\right| < \epsilon$, where ϵ is a tolerance, stop since the solution has been found, else go to step 1.

To show how the algorithm works, define:

$$\tilde{p}(x(\omega), \bar{x}) = E \left[f(x(\omega), \omega) + \frac{1}{2} (x(\omega) - \bar{x})' Q (x(\omega) - \bar{x}) \right]. \quad (5.49)$$

Then observe that we have the following inequalities:

$$\tilde{p}(x(\omega)^{k-1}, \bar{x}^k) \geq \tilde{p}(x(\omega)^k, \bar{x}^k) \geq \tilde{p}(x(\omega)^k, \bar{x}^{k+1}). \quad (5.50)$$

The first inequality corresponds to step 1 in the algorithm and holds because we are minimizing over the stochastic variable $x(\omega)$. The second inequality holds because evaluating the optimality conditions 5.48 is equivalent to minimizing over the deterministic variable \bar{x} , as shown in lemma 21. The inequalities 5.50 show that successive iterations of the algorithms decrease the objective function. The method is similar to *coordinate descent* presented by Bertsekas [Ber95]. Although this method works for problems with all continuous variables, it is not recommended because these problems may have regions in which the objective function is linear (as an example see figure 5-2). A subgradient method may be quicker in these situations.

5.5.4 Computational issues

Solving the non-linear integer program 5.40 can be quite challenging. Fortunately, for binary decision variables, it is possible to express the quadratic term as a linear function when the matrix Q is diagonal. If $f(x(\omega), \omega) = c(\omega)x(\omega)$, then the linearization of the quadratic term makes 5.40 a linear integer program for all scenarios. Consider the quadratic term corresponding to the i^{th} element:

$$\frac{q_i}{2} (x(\omega)_i - \bar{x}_i)^2. \quad (5.51)$$

Instead we can write this term as:

$$\frac{q_i}{2} [(1 - \bar{x}_i)^2 - \bar{x}_i^2] x(\omega)_i + \frac{q_i}{2} \bar{x}_i^2. \quad (5.52)$$

We can see that for $x(\omega) \in \{0, 1\}$ the values of 5.51 are the same as those of 5.52, but 5.52 is linear in the second stage variables.

5.5.5 Saving branch and bound trees

In the solution methods presented above, we found that in the second stage we have to solve an integer or mixed integer program for each positive probability scenario. This is very time consuming. Instead, we propose at each iteration of the method, to save the branch and bound trees for all scenarios. Observe that at the end of any second stage scenario solution all the leaves of the B&B tree have been either fathomed and the search stopped on that particular node or they contain a discrete feasible solution. Observe that in the following iteration the costs in the objective function of the second stage problems are updated using 5.52. Now we use our saved branch and bound trees in the following way. First we evaluate the leaves that contain a feasible solution using the updated objective function, to obtain a new lower bound for the tree. Next we examine the previously fathomed nodes using the new lower bound to see which ones should still be fathomed and which may be opened to further exploration.

We consider that this method may help considerably to reduce the time and make more efficient the solution of the discrete second stage problems.

5.6 Post-solution analyses

We have seen so far how to solve (at least approximately) the generalized average plan model 5.5 for continuous and discrete variables. In some cases, we may find an average plan \bar{x} that is feasible for all strictly positive probability scenarios. Then this is the plan we should follow. However, the average plan model is designed precisely

to cover the cases in which no such solution exists. What we have to do in this case is to select one of the second stage plans to be our master plan. We know by the formulation of the average plan that the second stage solutions are reasonably close to each other. We propose the following heuristic to select a second stage plan.

Step 0 Scenario sampling. Make an unbiased sample of scenarios $\omega^m \in \Omega$, $m = 1, \dots, M$.

Step 1 Scenario evaluation. Using the optimal average plan solution \bar{x}^* obtain the scenario dependent plan.

$$x(\omega^m) = \arg \min_{x(\omega^m) \in X(\omega^m)} f(x(\omega^m), \omega^m) + \frac{1}{2} (x(\omega^m) - \bar{x}^*)' Q (x(\omega^m) - \bar{x}^*) \quad (5.53)$$

$m = 1, \dots, M$

Step 2 Scenario evaluation. For all the scenarios sampled, obtain the following value:

$$z^m = \sum_{n=1}^M (x(\omega^m) - x(\omega^n))' Q (x(\omega^m) - x(\omega^n)) \quad (5.54)$$

$m = 1, \dots, M$

Step 3 Plan selection. Find:

$$\min_{m=1, \dots, M} z^m. \quad (5.55)$$

The corresponding value of $x(\omega^m)$ is the desired plan.

The rationale behind this algorithm is that we are trying to find the plan from which a deviation would be the least costly. We are pricing the deviations by using the quadratic term, so 5.54 and 5.55 help us to find the sample plan that will have the minimum correction cost.

Another way to find an *acting or master plan* is to find the plan that is closest to the master plan \bar{x}^* . We do this by solving the problem:

$$\min_{m=1, \dots, M} (x(\omega^m) - \bar{x}^*)' Q (x(\omega^m) - \bar{x}^*). \quad (5.56)$$

In fact this approach is equivalent to the one obtained using equations 5.54 and 5.55. Observe that:

$$\begin{aligned} & E [(x(\omega) - x(\omega^m)) Q (x(\omega) - x(\omega^m))] \\ &= E [((x(\omega) - \bar{x}^*) - (x(\omega^m) - \bar{x}^*)) Q ((x(\omega) - \bar{x}^*) - (x(\omega^m) - \bar{x}^*))]. \end{aligned} \tag{5.57}$$

Developing the right hand side, we have:

$$\begin{aligned} & E [(x(\omega) - x(\omega^m)) Q (x(\omega) - x(\omega^m))] \\ &= E [(x(\omega) - \bar{x}^*) Q (x(\omega) - \bar{x}^*)] + 2 E [(x(\omega) - \bar{x}^*)] Q (x(\omega^m) - \bar{x}^*) \\ & \quad + (x(\omega^m) - \bar{x}^*) Q (x(\omega^m) - \bar{x}^*). \end{aligned} \tag{5.58}$$

But the second term of the right hand side goes to zero because of the optimality condition 5.25. It shows that finding the acting plan $x(\omega^m)$ that minimizes the correction costs with respect to the plans of other scenarios is the same as that which minimizes the correction cost with respect to the optimal average plan.

5.7 Chapter 5 summary

In this chapter, we extend the average plan model 4.1 of chapter 4 to a generalized model 5.5 that is not limited to a particular shape of the objective function or to a set of linear constraints. We show that the generalized average plan model is a relaxation of the planning model represented by 5.6, and a non-Lagrangian dual to this problem in which the unknown dual variable is the positive matrix Q . In some cases, it will be possible to find a Q that makes it possible to use the average plan 5.5 to obtain a solution to 5.6, but in general it will not be possible since no such solution exists. Instead, fixing Q will help us to find reasonable plans.

Then we study how to solve the average plan 5.5 when there is convexity of the deterministic variables \bar{x} . We prove that a solution exists when the function $f(x(\omega), \omega)$ is convex in the almost surely sense. Furthermore, we find an explicit

expression for the subgradient 5.16 and optimality conditions 5.25 in this case. This allows us to apply the algorithms of chapter 3 to solve for the average plan.

When we study how the average plan behaves for discrete problems, we find that these are not convex in the deterministic variables and therefore, we cannot find a global optimum except with complete exploration of the decision space. Instead, we find local minima and we propose a heuristic to find reasonable solutions. Finally, we propose a heuristic to select a particular plan to follow based on choosing the one with least expected correction cost.

Chapter 6

Transportation applications of the average plan model

6.1 Introduction

The purpose of this chapter is to provide proof of concept of the generalized average plan model. Our approach is to examine certain transportation problems that have been solved in a deterministic manner and alter them to include randomness, thereby applying and illustrating the theory of chapters 4 and 5.

6.2 A service network design model for the distribution of crops in Mexico

6.2.1 Problem background

Mexico's surface transportation network (railway and highways) has two characteristics:

1. It spans very difficult terrain due to geographical accidents,
2. Only important cities and ports are connected by high capacity roads and/or railway tracks.

The consequence of these two points is that the surface transportation network is not redundant. Meaning that there are very few routes between any two points in the network, sometimes just one. Furthermore, these roads link only the big cities and major ports.

There are well defined crop producing regions and consumption centers in Mexico. In particular most of the production is done in the North-West region. The crop demand that cannot be supported by internal production is supplied by importation of crops through the Gulf's ports (Veracruz and Tampico). The primary consumption centers are the cities (Mexico city, Guadalajara and Monterrey).

[Jau89] studies the problem of routing railcars at the Mexican National Railway company to service crops. The transport requirements goal is to find routings for loaded and unloaded cars over the railroad network, based on demands for transport of crops.

In this thesis, we solve the same problem but consider different demand scenarios to find new routings under stochastic conditions. In the original problem, routes were constructed by sending loaded cars to their destination using the shortest paths, and the service network was completed by returning empty cars to loading points by using a network flow model. Demand data represents supply and demand forecasts for crops, for the year 1991.

For purpose of our case study, we have simplified the 1989 network by using node consolidation (nodes represent regions rather than individual cities). The nodes in our network and the corresponding demands (-) and supplies (+) are presented in Table 6.1. The corresponding network is shown in figure 6.2.1. Table 6.2 shows the distances between any two cities using the shortest paths in the network. The cost of creating a route between any two cities was calculated using the empirical formula $.5d^8$, where d is the length of the shortest path between these two cities.

6.2.2 Methodology

We solve this problem using the algorithm of section 5.5.3. The second stage scenario problem is the *Service Network Design Model* studied by Barnhart and Kim [Kim97].

Number	Name	Average cargo (Loads/Week)
1	North-West Region	122.15
2	Torreon	-97.38
3	Monterrey	236.94
4	Guadalajara	-44.13
5	Aguascalientes	0
6	San Luis Potosi	0
7	Tampico	18
8	Mexico City	-492.98
9	Veracruz	257.4

Table 6.1: Modeled nodes

Nodes	1	2	3	4	5	6	7	8	9
1	0	8.8	12.42	11.6	13.6	13.6	17.6	15.2	19.5
2	8.8	0	3.62	7.1	5.1	6.78	8.92	10.7	13.71
3	12.42	3.62	0	7.37	7.05	5.37	5.3	9.56	10.09
4	11.6	7.1	7.37	0	2	2	6	3.6	7.9
5	13.6	5.1	7.05	2	0	1.68	5.68	5.6	9.9
6	13.6	6.78	5.37	2	1.68	0	4	4.19	8.49
7	17.6	8.92	5.3	6	5.68	4	0	8.19	4.79
8	15.2	10.7	9.56	3.6	5.6	4.19	8.19	0	4.3
9	19.5	13.71	10.09	7.9	9.9	8.49	4.79	4.3	0

Table 6.2: Shortest path distances in the network

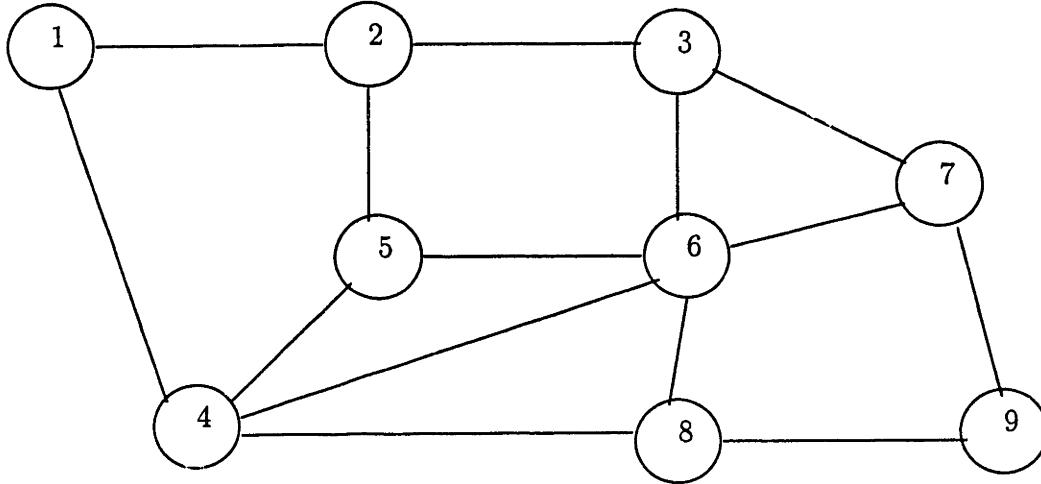


Figure 6-1: The network model for Mexico's distribution problem

This is similar to the network design problem studied by Magnanti [MW84] with the following special characteristics:

- The links between nodes are vehicles moving between them. The capacity of the link will be a function of the type of vehicle used and the frequency of travel. In this particular application we only consider the aggregate capacity of the vehicles in the link.
- To model vehicle tours it is necessary to add conservation of flow constraints for the vehicles to the model.

The model for the second stage scenario dependent problem is:

$$\min \sum_{(i,j)} \left(f_{(i,j)} + q \left(1 - \bar{x}_{(i,j)}^k \right)^2 + q \left(\bar{x}_{(i,j)}^k \right)^2 \right) x_{(i,j)}(\omega) + \quad (6.1)$$

$$\sum_{(i,j)} c_{(i,j)} y_{(i,j)}(\omega)$$

s.t.

$$Ny(\omega) = d(\omega) \quad (6.2)$$

$$Nx(\omega) = 0 \quad (6.3)$$

$$\begin{aligned}
Ux_{(i,j)}(\omega) &\geq y_{(i,j)}(\omega) \quad \forall (i,j) \\
x_{(i,j)}(\omega) &\in \{0,1\} \quad \forall (i,j).
\end{aligned}
\tag{6.4}$$

where:

$x_{(i,j)}(\omega)$ is a binary vector representing the assignment of a fixed number of vehicles to arc (i,j) of the network in scenario ω ;

$y_{(i,j)}(\omega)$ is the number of loads of crops on arc (i,j) in scenario ω ;

N is the node-arc incidence matrix representing the network;

$d(\omega)$ is the demand in scenario ω ;

U is the aggregate capacity of the stream of vehicles in the arc;

$f_{(i,j)}$ is the cost of assigning a stream of vehicles to arc (i,j) , and

$c_{(i,j)}$ is the cost per load of transporting crops over arc (i,j) .

Constraints 6.2 enforce conservation of flow for the crops, constraints 6.3 guarantee conservation of flow for the vehicles and constraints 6.4 allow crops to flow only on arcs where vehicles have been assigned.

We linearize the quadratic term in the objective function using the method of equation 5.52. For the solution of this problem, we generate 5 random scenarios using a normal distribution with mean given by the values of Table 6.1 and a diagonal covariance matrix with 10 loads/week in all non-zero elements.

6.2.3 Results

We solve the problem using $q = 50$ and $q = 500$. The corresponding average plan routings of vehicles are shown in Figures 6-2 and 6-3.

With the value of $q = 500$ we have a single solution that is feasible and is the same for the 5 scenarios. The objective function expected value is 171298.3 Load-sKm/Week. In contrast, when $q = 50$, one solution (shown in figure 6-2) is selected

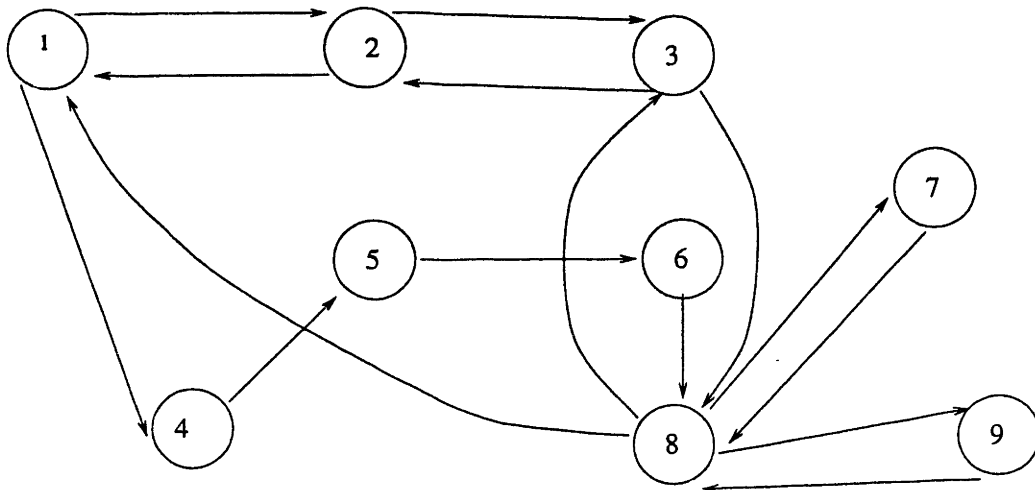


Figure 6-2: Solution with $q = 50$

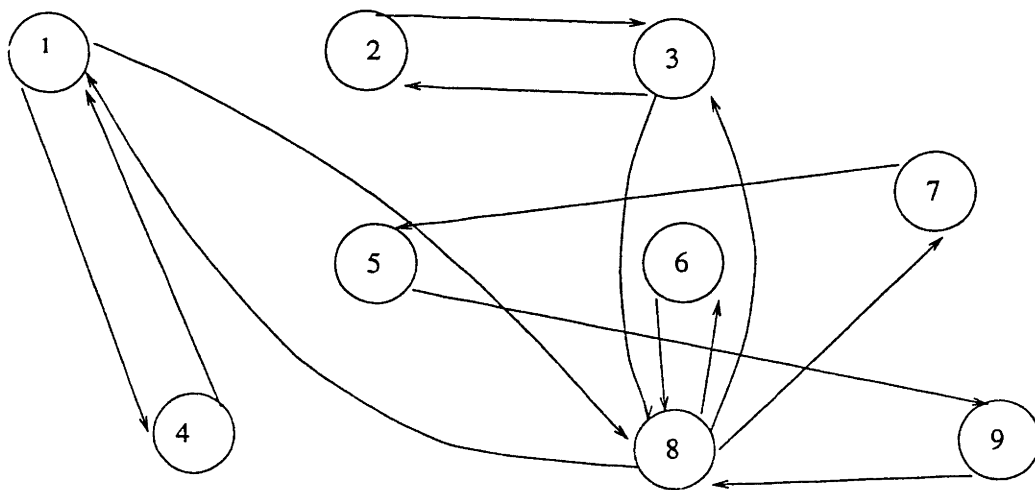


Figure 6-3: Solution with $q = 500$

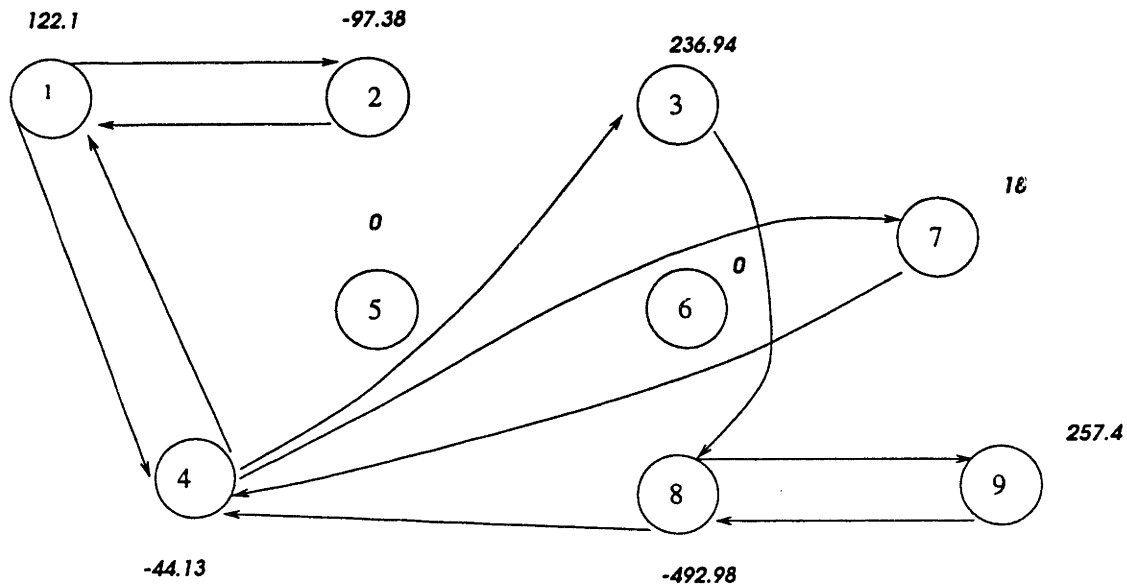


Figure 6-4: Deterministic solution of Mexico's logistic problem.

for only 4 of the 5 scenarios. In the solution for the 5th scenario, we found two additional links between nodes 3 and 6. The expected value of the objective function is 125861.9 Loads*Km/Week.

There are a few points worth mentioning about the solution:

- First, it is interesting to contrast the difference between the solution obtained using the average plan model and the solution of the deterministic model of the problem using average values for the demand. The deterministic solution is shown in Figure 6-4 . The most striking difference in the routing is that nodes 5 and 6 are ignored in the deterministic routing because their expected demand is zero. This doesn't happen in a stochastic routing because these nodes can serve as supply or demand points and hence, the demand for those nodes will be different from zero for some scenarios. Thus the average plan will route a stream of vehicles through them and the deterministic plan, which does not, is infeasible for of the 5 scenarios.
- As mention in section 5.5, problems with discrete variables as this one may have

several local optima, so the solutions presented in Figures 6-2 and 6-3 may not be the absolute minima. Nevertheless, both present the desirable characteristics expected from the average plan. In the solution obtain for $q = 50$ we obtain $P(\bar{x} = x(\omega)) = 0.8$ while for $q = 500$ we obtain $P(\bar{x} = x(\omega)) = 1$.

- The numerical example presented here uses five scenarios obtained from sampling a multidimensional normal distribution. It is easy to understand the results obtained in the case of $q = 50$ by thinking about the shape of the distribution. There are a few scenarios that are in the tail of the distribution while most of them are in the main body of the distribution. The scenarios in the main body are closer to each other and the solution for them would be similar. In this example, 4 of 5 scenarios have the same solution. While for the scenarios in the tail, the solution is different.

6.3 An airplane scheduling problem

6.3.1 Problem background

During the last decade, the Federal Aviation Administration (FAA) has been scheduling daily ground delays for commercial flights. The rationale behind this policy is that it is cheaper and safer to make airplanes wait on the ground of the departing airports than to circle in the air above the arrival airport. This problem, called *the ground holding problem*, has been extensively researched by Odoni and Richetta[RO93], and by Vranas, Bertsimas and Odoni [VBO94] just to mention a few.

In particular Vranas et al. [VBO94] have developed a deterministic model to study the network effects of such imposed ground delays. In their model, the main set of constraints reflect airport landing capacities in the network, throughout the day.

In our case study, we do not solve the network wide problem under uncertainty. Instead we solve the problem of finding the schedule (itinerary) of flights of a single aircraft moving over congested networks. We consider the shadow price of landing at a given airport at a given time rather than the capacity of the airport. Our intent

Scenario	Airport 1	Airport 2	Probability
1	Good	Good	$\frac{1}{3}$
2	Good	Bad	$\frac{1}{6}$
3	Bad	Good	$\frac{1}{6}$
4	Bad	Bad	$\frac{1}{3}$

Table 6.3: Scenarios and associated probabilities

is for this work to serve as a prototype for the column generation step of a larger problem that would solve the ground holding problem under uncertainty.

An interesting interpretation for the average plan model in this case, is that an airline does not design a schedule for each possible climatic condition, but rather it designs a schedule that works on average. The property of the average plan to tie schedules for different scenarios in a flexible way can be exploited in this problem, since we are not looking for schedules that are optimal all the time.

6.3.2 Methodology

For this case study, we use the data from Vranas' first example containing two airports (JFK and BOS) and 64 time slots of 15 minute durations for 16 hours of airport operation. For each airport we consider two possible states: "Good" in which the same landing capacities as in Vranas' data are used and "Bad" in which we reduce by 1 airplane/time period the capacity of airport 2 (BOS) in slots 12 to 49. For the "Bad" scenario in airport 1 (JFK), we reduce the capacity by 1 or 2 airplanes/time period in slots 13 to 51. Combining the states, we create four scenarios, shown in Table 6.3. We assign a higher probability to scenarios 1 and 4 because we consider the states of both airports to be positively correlated. Using these scenarios, we solved deterministically the *ground holding problem* for each scenario and obtain the corresponding shadow prices for the capacity of each airport in each time period, for each scenario, as shown in Table 6.4.

We are trying to schedule 20 flights of the same airplane between the two airports.

Time Slots	Airport 1				Airport 2			
	Scenario 1	2	3	4	Scenario 1	2	3	4
1	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0
5	50	50	50	50	0	0	0	0
6	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0
10	200	300	200	300	200	200	200	200
11	150	250	150	250	150	150	150	150
12	100	200	100	200	100	100	100	100
13	50	150	50	150	50	50	50	50
14	0	100	0	100	0	0	0	0
15	0	50	0	50	0	0	0	0
16	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0
20	300	450050	375050	450050	200	200	300	450050
21	250	450000	375000	450000	150	150	250	450000
22	200	449950	374950	449950	100	100	200	449950
23	150	449900	374900	449900	50	50	150	449900
24	100	449850	374850	449850	0	0	100	449850
25	50	100	50	100	0	0	50	100
26	0	50	0	50	0	0	0	50
27	0	0	0	0	0	0	0	0
28	0	0	0	0	0	0	0	0
29	0	0	0	0	0	0	0	0
30	375050	450050	375050	450050	200	200	1.3499e+06	1.3499e+06
31	375000	450000	375000	450000	150	150	1.34985e+06	1.34985e+06
32	374950	449950	374950	449950	100	100	1.3498e+06	1.3498e+06
33	374900	449900	374900	449900	50	50	1.34975e+06	1.34975e+06
34	374850	449850	374850	449850	0	0	1.3497e+06	1.3497e+06
35	50	100	50	100	200	450	1.34965e+06	1.34965e+06
36	0	50	0	50	150	400	1.3496e+06	1.3496e+06
37	0	0	0	0	100	350	1.34955e+06	1.34955e+06
38	0	0	0	0	50	300	1.3495e+06	1.3495e+06
39	0	0	0	0	0	250	1.34945e+06	1.34945e+06
40	300	450050	300	450050	200	200	524950	524950
41	250	450000	250	450000	150	150	524900	524900
42	200	449950	200	449950	100	100	524850	524850
43	150	449900	150	449900	50	50	524800	524800
44	100	449850	100	449850	0	0	524750	524750
45	50	100	50	100	0	0	50	50
46	0	50	0	50	0	0	0	0
47	0	0	0	0	0	0	0	0
48	0	0	0	0	0	0	0	0
49	0	0	0	0	0	0	0	0
50	375050	375050	375050	375050	200	200	250	200
51	375000	375000	375000	375000	150	150	200	150
52	374950	374950	374950	374950	100	100	150	100
53	374900	374900	374900	374900	50	50	100	50
54	374850	374850	374850	374850	0	0	50	0
55	50	50	50	50	0	0	0	0
56	0	0	0	0	0	0	0	0
57	0	0	0	0	0	0	0	0
58	0	0	0	0	0	0	0	0
59	0	0	0	0	0	0	0	0
60	200	200	200	200	0	0	0	0
61	150	150	150	150	0	0	0	0
62	100	100	100	100	0	0	0	0
63	50	50	50	50	0	0	0	0
64	0	0	0	0	0	0	0	0

Table 6.4: Shadow prices of time slots

The mathematical model of the problem for scenario ω is:

$$\min \sum_{i=1}^{20} \left[C_g (r_i(\omega) - t_i(\omega)) + \lambda_i (r_i(\omega) + f) + \frac{q}{2} (t_i(\omega) - \bar{t})^2 \right] \quad (6.5)$$

s.t.

$$r_i(\omega) \geq t_i(\omega) \quad i = 1, \dots, 20 \quad (6.6)$$

$$t_{i+1}(\omega) \geq r_i(\omega) + f + u \quad i = 1, \dots, 19 \quad (6.7)$$

$$t_0(\omega) \geq 1 \quad .$$

Where:

$t_i(\omega)$ is the scheduled departure time for flight i in scenario ω ;

$r_i(\omega)$ is the actual departure time of flight i in scenario ω ;

f is the fly time (considered 1 time period);

u is the turn around time (considered 1 time period);

$\lambda_i(\tau)$ is the shadow cost of flight i landing in time period τ ;

C_g is the ground holding cost (50 cost units/time period).

The objective function has three terms. The first is the cost of the plane waiting on the ground. The second is the shadow cost of constraint associated with the plane landing at a time equal to actual departure time the flying time. The last is the quadratic term of the average plan model. Constraint 6.6 tells us that the actual departure time should be greater than the scheduled departure time. Constraints 6.7 force the next scheduled departure time to be at or after the time at which the airplane is ready to depart on its next flight.

Due to the problem's structure, we solve it as a dynamic program. In general, the ability to solve this problem as dynamic program should be exploited for the purpose of using this model as the column generation step of a network-wide model

with several aircraft. The dynamic programming recursion equations are:

$$\begin{aligned}
 J_i(t_i(\omega), \omega) &= \min_{r_i(\omega) \geq t_i(\omega)} C_g(r_i(\omega) - t_i(\omega)) + \lambda_i(r_i(\omega) + f) \\
 &\quad + \frac{q}{2}(t_i(\omega) - \bar{t})^2 + \min_{t_{i+1}(\omega) \geq r_i(\omega) + f + u} J_i(t_{i+1}(\omega), \omega) \quad (6.8) \\
 i &= 1, \dots, 19
 \end{aligned}$$

$$J_{20}(t_{20}(\omega), \omega) = \min_{r_{20}(\omega) \geq t_{20}(\omega)} C_g(r_{20}(\omega) - t_{20}(\omega)) + \lambda_{20}(r_{20}(\omega) + f). \quad (6.9)$$

These equations are easy to implement and simplify considerably the calculations necessary to solve the scenario specific problem.

Although in reality this is not a discrete variable problem, we have discretized it by dividing the time horizon into periods. Consequently, we solve 6.5 using the algorithm of section 5.5.3 .

6.3.3 Results

We implement the dynamic programming approach of equations 6.8, in a program shown in the appendix. We solve it using values of $q = 50$ and $q = 100$. The resulting average plan schedules are shown in Table 6.5, with the schedule depicting the departure times of each flight. As expected the objective function value for $q = 100$ is much larger than the value for $q = 50$. For $q = 100$, all scenario solutions are equal meaning that for this value of q the constraint $t(\omega) = \bar{t}$ is honored with probability one. Observe that in this solution, the flights are evenly spaced by two time periods, this spacing represents the flying time plus the turn around time. This means that the airline is willing to "pay" whatever is necessary to land their airplanes at the desired time. This disregard for the landing cost causes the schedule to be the same in all scenarios. In the $q = 50$ case, there are again several flights spaced by two time units. However periods 17 and 38 are followed by longer gaps in time before the next flight. This coincides with sharp increases in the landing shadow costs.

This example illustrates how costs increase as the quadratic parameter q increases. Also it shows that the quadratic term does not necessarily increase the numerical

Flight	Schedule $q = 50$				Schedule $q = 100$
	Scenario 1	Scenario 2	Scenario 3	Scenario 4	
1	1	1	1	1	1
2	3	3	3	3	3
3	5	5	5	5	5
4	7	7	7	7	7
5	9	9	9	9	9
6	11	11	11	11	11
7	13	13	13	13	13
8	15	15	15	15	15
9	17	17	17	17	17
10	20	22	22	22	19
11	22	24	24	24	21
12	24	26	26	26	23
13	26	28	28	28	25
14	28	30	30	30	27
15	32	32	33	33	29
16	34	34	35	35	31
17	36	36	37	37	33
18	38	38	39	39	35
19	43	43	44	44	37
20	46	46	46	46	39
Objective function value	110503				358950

Table 6.5: Solution of the average plan model for the flight scheduling problem. Even numbered flights arrive to airport 1.

complexity of the problem, as can be seen from dynamic equations 6.8. Notice, that the solution to this problem is sensitive to the starting value \bar{t}^0 of the average plan in the iteration process and that with other starting points, other solutions may be found.

6.4 Chapter 6 summary

In this chapter we illustrate the application of the average plan model using two case studies. In the first, we consider a logistics model that uses the service network design problem as the second stage scenario dependent problem. In the second we solve a subproblem of the average plan version of the ground holding problem by solving a schedule construction problem. The scenario dependent problem in this case can be solved using dynamic programming making it numerically easy. Both problems illustrate the properties of the average plan model described in chapter 5.

Chapter 7

Future Research

We examine here some topics of future research, and break these topics into three categories: theoretical, computational and economic research issues.

7.1 Theoretical research issues

- It is necessary to formally connect the chance constraint formulation 5.3 with the average plan model 5.5 and show in which cases the two are equivalent.
- It is necessary to develop methods like the stochastic quasigradient algorithm of section 3.7.1 for problems with discrete variables. This could be used to interface with a simulator and manage large numbers of scenarios for problems with discrete variables.
- It is important to explore methods to find the global optimum of the average plan model with discrete variables. Such methods could be based in branch and bound or in heuristic explorations of the domain.
- In the average plan model, we relax the constraint $\bar{x} = x(\omega)$ *a.s.* by means of the quadratic term. Nevertheless other functions may be used to relax this constraint and produce meaningful duals. These other functions should be explored.

- Connections with other models for robust planning such as Yu's [YK97], should be explored.

7.2 Computational issues

- Many researchers are already studying how to parallelize stochastic programming problems (see [NZ96]). In the case of the average plan model, it is pretty obvious that the second stage scenario dependent problems can be solved in parallel. It would be useful to study formal algorithms to parallelize this solution. Furthermore, it may be possible to distribute different functions to several processors in a distributed computing environment.
- It is necessary to explore how to connect the average plan to existing systems such as GIS and optimization solvers in order to turn it into a practical tool for a wide variety of planning problems.

7.3 Economic issues

- It is necessary to investigate the economic interpretation of the quadratic term.

Appendix A

Software implementation

In this appendix we include the software corresponding to the case study of section 6.3

A.0.1 mplane.cc

```
//Implementation of the average plan model  
//applied to the airplane scheduling problem.  
//It uses an object-oriented design to  
//solve the scenario related problem.
```

```
#include <iostream.h>  
#include <.stream.h>  
#include <stdio.h>  
#include "plane.h"
```

```
int nscenarios = 4, nperiods = 64, nflights = 20, tut = 1, ftime = 1;  
double cg = 50., q = 50;  
double pe[] =  
{.33, .17, .17, .33};
```

```
void input_ca (double *, double *, char *);
```

```
main ()
```

10

```

{
    char name[15];
    int iter = 0;
    vector tb (nflights), te (nflights);

    //CConstructors
    double **a = new double *[nscenarios];
    double **b = new double *[nscenarios];
    plane **p = new plane *[nscenarios];
    vector **tw = new vector *[nscenarios];
    double *cst = new double[nscenarios];

    for (int i = 0; i < nscenarios; i++)
    {
        a[i] = new double[nperiods];
        b[i] = new double[nperiods];

        sprintf (name, "scenario%d.dua", i);

        input_ca (a[i], b[i], name);

        p[i] = new plane (nperiods, nflights, ftime, tut, cg, q, a[i], b[i], tb);

        tw[i] = new vector (nperiods);
    }

    //EEngine

    do
    {
        cout << "Iteration " << iter++ << endl;

        tb = te;

        te.zero ();
    }
}

```

```

for (int j = 0; j < nscenarios; j++)
{

    p[j]->new_t (tb);

    for (int k = nflights - 1; k >= 0; k--)
    {
        p[j]->iteration (k);
    }

    //ccout<<"Esc:"<<j<<" "<<*(p[j]->J[0])<<endl;

    *(tw[j]) = p[j]->schedule (cst[j]);

    //ccout<<cst[j]<<endl;

    te += *(tw[j]) * (pe[j]);
}

cout << "E[tw]=" << te << endl << endl;

}
while (te != tb);

//OOutput
for (int j = 0; j < nscenarios; j++)
{
    cout << "Scenario " << j << " =" << *(tw[j]) << endl;
}

cout << "tb=" << tb << endl << endl;

//PPos-solution analysis

```

```

vector tx (nscenarios), tk (nperiods);

for (int k = 0; k < nscenarios; k++)
{
    tk = *tw[k] - tb;
    tx[k] = (tk.module ()) * q;
}

int pi;
double cpi = tx.min (pi);
100

cout << "Recommended scenario:" << pi << endl;
cout << "Schedule:  " << *tw[pi] << endl;

double obj = 0., rel = 0.;

for (int j = 0; j < nscenarios; j++)
{
    obj += cst[j] - tx[j];
110

    if (*tw[pi] == *tw[j])
    {
        rel += pe[j];
    }
}

cout << "Objective value=" << obj << endl;
cout << "Reliability=" << rel << endl << endl;

//ddestructors
120

for (int i = 0; i < nscenarios; i++)
{
    delete a[i], b[i];
    delete p[i];
    //    delete tw[i];
}

```

```
}
```

```
delete a, b; 130
```

```
delete p;
```

```
delete tw;
```

```
delete cst;
```

```
}
```

```
void
```

```
input_ca (double *a, double *b, char *s)
```

```
{
```

```
int airport, time0;
```

```
double cgg, ca; 140
```

```
ifstream scenario (s, ios::in);
```

```
for (int i = 0; i < nperiods; i++)
```

```
{
```

```
scenario >> airport >> time0 >> cgg >> ca;
```

```
a[i] = -ca;
```

```
}
```

```
for (int i = 0; i < nperiods; i++) 150
```

```
{
```

```
scenario >> airport >> time0 >> cgg >> ca;
```

```
b[i] = -ca;
```

```
}
```

```
scenario.close ();
```

```
}
```

A.0.2 plane.h

```
#ifndef PLANE
```

```
#define PLANE
```

```

#include <iostream.h>
#include "vector.h"

#define infinity 1E20

class plane
{
    public:

    vector ** J, **ac, *tb;

    int nperiods, nflights, ftime, tut;
    double cg, q;

    plane (int np, int nf, int ft, int tu, double cgg, double qq, double *a, double *b, vector & t0);

    ~plane ();

    void iteration (int k);

    vector schedule (double &val);

    void new_t (vector & t0);
};

#endif

```

10

20

30

A.0.3 plane.cc

```

#include "plane.h"

void plane::
iteration (int k)

```

```

{
  int t, p, tmax, g;
  vector tg (nperiods);
  int arp = k % 2;

  for (t = 0; t < nperiods; t++)
  {
    tg.zero ();

    for (g = 0; g < nperiods - t; g++)
    {
      tmax = t + g + ftime;

      if (tmax < nperiods)
      {
        tg[g] = cg * g + (*ac[arp])[tmax] + q * (t - (*tb)[k]) * (t - (*tb)[k]) / 2;
      }
      else
      {
        tg[g] = infinity;
      }

      tmax += tut;

      if (tmax < nperiods)
      {
        tg[g] += (*J[k + 1]).min (p, tmax, nperiods - 1);
      }
      else
      {
        tg[g] = infinity;
      }

      //ccout<<"g="<<g<<" t="<<t<<" tmax="<<tmax<<endl;
    }
  }

```

```

    //ccout<<"tg="<<tg<<endl;

    (*J[k])[t] = tg.min (p, 0, nperiods - t - 1);
}
//ccout<<"k="<<k<<"*J[k]<<endl;
}

```

```

plane::~plane () 50
{
    for (int i = 0; i <= nflights; i++)
    {
        delete J[i];
    }

    delete J;

    delete ac[0];
    delete ac[1]; 60
    delete ac;
    delete tb;
}

```

```

plane::plane (int np, int nf, int ft, int tu, double cgg, double qq, double *a, double *b, vector & t0)
{
    nperiods = np;
    nflights = nf;
    ftime = ft;
    tut = tu; 70
    cg = cgg;
    q = qq;

    J = new vector *[nflights + 1];

    for (int i = 0; i <= nflights; i++)

```



```
{  
  J[i] = new vector (nperiods);  
}
```

80

```
ac = new vector *[2];
```

```
ac[1] = new vector (nperiods, a);
```

```
ac[0] = new vector (nperiods, b);
```

```
tb = new vector (t0);
```

```
}
```

90

```
vector plane::
```

```
schedule (double &val)
```

```
{
```

```
  vector t (nflights);
```

```
  int td, ta;
```

```
  double cost;
```

```
  ta = 0;
```

100

```
  for (int k = 0; k < nflights; k++)
```

```
  {
```

```
    cost = (*J[k]).min (td, ta, nperiods - 1);
```

```
    if (k == 0)
```

```
      val = cost;
```

```
    t[k] = td;
```

110

```
    ta = td + ftime + tut;
```

```

    }

    //ccout<<"*"<<t<<endl;
    return (t);
}

```

```

void plane::
new_t (vector & t0)
{
    *tb = t0;
}

```

120

A.0.4 vector.h

```

// class to manipulate vector
//iit uses loop unrolling, with 2 elements

```

```

#ifndef VECTOR
#define VECTOR

```

```

#include <iostream.h>
#include <math.h>
#include "array.h"

```

10

```

class vector:public array
{

```

```

    friend ostream & operator << (ostream & os, vector & v);

```

```

    public:
    vector (int n0):array (n0)
    {
    }
}

```

20

```
vector (int n0, double *b):array (n0, b)
{
};
```

```
vector (vector & v):array (v)
{
};
```

```
vector operator + (vector v);
```

30

```
vector operator - (vector v);
```

```
vector operator *(double k);
```

```
double operator *(vector v);
```

```
vector operator / (double k);
```

```
void operator += (vector v);
```

40

```
void operator -= (vector v);
```

```
void operator *= (double k);
```

```
void operator /= (double k);
```

```
double sum (void);
```

```
double module (void);
```

50

```
};
```

```
#endif
```

A.0.5 vector.cc

```

#include "vector.h"

vector vector::operator + (vector v)
{
    vector u (n);

    if (!same_size (v))
    {
        cout << "Vector addition tried in members of different dimensions" << endl;
        exit (1);
    }

    for (int i = 0; i < n - 1; i += 2)
    {
        u.a[i] = a[i] + v.a[i];
        u.a[i + 1] = a[i + 1] + v.a[i + 1];
    }

    if (odd)
        u.a[n - 1] = a[n - 1] + v.a[n - 1];

    return (u);
};

vector vector::operator - (vector v)
{
    vector u (n);

    if (!same_size (v))
    {
        cout << "Vector difference tried in members of different dimensions" << endl;
        exit (1);
    }

    for (int i = 0; i < n - 1; i += 2)
    {

```

```

    u.a[i] = a[i] - v.a[i];
    u.a[i + 1] = a[i + 1] - v.a[i + 1];
}

```

40

```

if (odd)
    u.a[n - 1] = a[n - 1] - v.a[n - 1];

```

```

return (u);
};

```

```

vector vector::operator * (double k)

```

```

{
    vector u (n);

```

50

```

    for (int i = 0; i < n - 1; i += 2)

```

```

    {
        u.a[i] = k * a[i];
        u.a[i + 1] = k * a[i + 1];
    }

```

```

if (odd)

```

```

    u.a[n - 1] = k * a[n - 1];

```

60

```

return (u);
}

```

```

vector vector::operator / (double k)

```

```

{
    vector u (n);

```

```

if (k == 0.)

```

```

    {
        cout << "Tried to divide a vector by zero" << endl;
        exit (1);
    }

```

70

```

for (int i = 0; i < n - 1; i += 2)
{
    u.a[i] = a[i] / k;
    u.a[i + 1] = a[i + 1] / k;
}

if (odd)
    u.a[n - 1] = a[n - 1] / k;

return (u);
}

double vector::operator * (vector v)
{
    double s = 0.;

    if (!same_size (v))
    {
        cout << "Dot product tried in vectors of different dimension" << endl;
        exit (1);
    }

    for (int i = 0; i < n - 1; i += 2)
    {
        s += a[i] * v.a[i] + a[i + 1] * v.a[i + 1];
    }

    if (odd)
        s += a[n - 1] * v.a[n - 1];

    return (s);
}

```

```

void vector::operator += (vector v)
{
    if (!same_size (v))
    {
        cout << "Vector += tried in members of different dimensions" << endl;
        exit (1);
    }

    for (int i = 0; i < n - 1; i += 2)
    {
        a[i] += v.a[i];
        a[i + 1] += v.a[i + 1];
    }

    if (odd)
        a[n - 1] += v.a[n - 1];
}

void vector::operator -= (vector v)
{
    if (!same_size (v))
    {
        cout << "Vector -= tried in members of different dimensions" << endl;
        exit (1);
    }

    for (int i = 0; i < n - 1; i += 2)
    {
        a[i] -= v.a[i];
        a[i + 1] -= v.a[i + 1];
    }

    if (odd)
        a[n - 1] -= v.a[n - 1];
}

```

```
}
```

```
void vector::operator *= (double k)
```

```
{
```

```
    for (int i = 0; i < n - 1; i += 2)
```

150

```
    {
```

```
        a[i] *= k;
```

```
        a[i + 1] *= k;
```

```
    }
```

```
    if (odd)
```

```
        a[n - 1] *= k;
```

```
}
```

160

```
void vector::operator /= (double k)
```

```
{
```

```
    if (k == 0.)
```

```
    {
```

```
        cout << "/=0 tried";
```

```
        exit (1);
```

```
    }
```

```
    for (int i = 0; i < n - 1; i += 2)
```

170

```
    {
```

```
        a[i] /= k;
```

```
        a[i + 1] /= k;
```

```
    }
```

```
    if (odd)
```

```
        a[n - 1] /= k;
```

```
}
```

```
ostream & operator << (ostream & os, vector & v)
```

180


```

{

int n = v.size ();

os << "(";

for (int i = 0; i < n; i++)
{
    os << v[i];

if (i != n - 1)
    os << ",";
else
    os << ")";
}

return (os);

}

```

190

```

double vector::
sum (void)
{
    double s = 0.;

for (int i = 0; i < n - 1; i += 2)
{
    s += a[i] + a[i + 1];
}

if (odd)
{
    s += a[n - 1];
}

return (s);

```

200

210

```
}
```

```
double vector::
```

```
module (void)
```

220

```
{
```

```
    double s = 0.;
```

```
    for (int i = 0; i < n - 1; i += 2)
```

```
    {
```

```
        s += a[i] * a[i] + a[i + 1] * a[i + 1];
```

```
    }
```

```
    if (odd)
```

```
    {
```

```
        s += a[n - 1] * a[n - 1];
```

```
    }
```

230

```
    return (s);
```

```
}
```

A.0.6 array.h

```
// Class to manipulate arrays of real numbers
```

```
#ifndef ARRAY
```

```
#define ARRAY
```

```
typedef unsigned short logical;
```

```
#define TRUE !0
```

```
#define FALSE 0
```

```
class array
```

10

```
{
```

```
    friend ostream & operator << (ostream & os, array & q);
```

```
    friend logical operator == (array & a, array & b);
```

```
    friend logical operator != (array & a, array & b);
```

protected:

double *a;

int n;

logical odd;

20

void isodd (void)

{

 odd = FALSE;

if (n % 2)

 odd = TRUE;

}

public:

array (int n0)

30

{

if (n0 <= 0)

 {

 cout << "Error trying to create array with nonpositive elements" << endl;

 exit (1);

 }

 n = n0;

 a = new double[n];

 zero ();

 isodd ();

40

};

array (int n0, double *b)

{

if (n0 <= 0)

 {

 cout << "Error trying to create array with nonpositive elements" << endl;

 exit (1);

 }

 n = n0;

50

```

a = new double[n];
for (int i = 0; i < n; i++)
    a[i] = b[i];
isodd ();
};

```

```

array (const array & b)
{
    n = b.n;
    a = new double[n];
    for (int i = 0; i < n - 1; i += 2)
        {
            a[i] = b.a[i];
            a[i + 1] = b.a[i + 1];
        }
    odd = b.odd;
    if (odd)
        a[n - 1] = b.a[n - 1];
};

```

60

70

```

~array ()
{
    delete a;
};

```

```

void zero (void)
{
    for (int i = 0; i < n; i++)
        a[i] = 0.;
}

```

80

```

double &operator[] (int j);

```

```

void operator = (array b);

```

```

double min (int &p);

```

```
double min (int &p, int first, int last);
```

```
double max (int &p);
```

90

```
double max (int &p, int first, int last);
```

```
int size (void)
```

```
{  
    return (n);  
}
```

```
logical same_size (array & u)
```

```
{  
    if (n == u.n)  
    {  
        return (TRUE);  
    }  
    return (FALSE);  
}  
};
```

100

```
#endif
```

A.0.7 array.cc

```
#include <iostream.h>
```

```
#include "array.h"
```

```
double array::
```

```
min (int &p)
```

```
{  
    double temp;
```

```
    temp = a[0];
```

```
    p = 0;
```

10

```

for (int i = 1; i < n; i++)
{
    if (temp > a[i])
    {
        temp = a[i];
        p = i;
    }
}
return (temp);
};

```

20

```

double array::
min (int &p, int first, int last)
{
    double t;

    if (first < 0 || first > n - 1 || last < 0 || last > n - 1 || last < first)
    {
        cout << "Bad parameters asked to min function in array" << endl;
        exit (1);
    }
}

```

30

```

t = a[first];
p = first;

for (int i = first; i <= last; i++)
{
    if (t > a[i])
    {
        t = a[i];
        p = i;
    }
}
return (t);
};

```

40

```

double array::
max (int &p) 50
{
    double temp = a[0];
    p = 0;

    for (int i = 1; i < n; i++)
    {
        if (temp < a[i])
        {
            temp = a[i];
            p = i; 60
        }
    }
    return (temp);
};

double &array::operator[] (int j)
{
    if (j >= n || j < 0)
    {
        cout << "Error addressing array element " << j; 70
        exit (0);
    }

    return a[j];
};

double array::
max (int &p, int first, int last)
{
    double t; 80

    if (first < 0 || first > n - 1 || last < 0 || last > n - 1 || last < first)

```

```

    {
        cout << "Bad parameters asked to max function in array" << endl;
        exit (1);
    }

    t = a[first];
    p = first;

    for (int i = first; i <= last; i++)
    {
        if (t < a[i])
        {
            t = a[i];
            p = i;
        }
    }
    return (t);
};

```

90

```

ostream & operator << (ostream & os, array & q)
{
    os << "(";
    for (int i = 0; i < q.n; i++)
    {
        os << q.a[i];
        if (i < q.n - 1)
            os << ", ";
    }
    os << ")";

    return (os);
};

```

100

```

logical operator == (array & a, array & b)
{
    logical p = TRUE;

```

110


```
int i = 0;
```

120

```
if (a.n != b.n)  
return (FALSE);
```

```
while (p == TRUE && i < a.n)
```

```
{  
if (a[i] != b[i])  
p = FALSE;  
i++;  
}  
return (p);  
};
```

130

```
logical operator != (array & a, array & b)
```

```
{  
logical q;  
  
q = a == b;
```

```
return (!q);  
};
```

140

```
void array::operator = (array b)
```

```
{  
n = b.n;  
a = new double[n];  
for (int i = 0; i < n - 1; i += 2)  
{  
a[i] = b.a[i];  
a[i + 1] = b.a[i + 1];  
}  
odd = b.odd;  
if (odd)
```

150

```
a[n - 1] = b.a[n - 1];  
};
```

A.0.8 Output for $q = 50$

Iteration 0

E[tw]=(0,2,4,6,8,10,12,14,16,18,20,22,24,26,28,30,32,34,36,38)

Iteration 1

E[tw]=(0,2,4,6,8,10,12,14,16,19,21,23,25,27,29,31,33,35,37,39)

Iteration 2

E[tw]=(0,2,4,6,8,10,12,14,16,19.67,21.67,23.67,25.67,27.67,30,32,34,36,38,40)

Iteration 3

E[tw]=(0,2,4,6,8,10,12,14,16,20.34,22.34,24.34,26.34,28.34,31,33,35,37,39,41)

Iteration 4

E[tw]=(0,2,4,6,8,10,12,14,16,20.34,22.34,24.34,26.34,28.34,31.5,33.5,35.5,37.5,40,42)

Iteration 5

E[tw]=(0,2,4,6,8,10,12,14,16,20.34,22.34,24.34,26.34,28.34,31.5,33.5,35.5,37.5,41,43)

Iteration 6

E[tw]=(0,2,4,6,8,10,12,14,16,20.34,22.34,24.34,26.34,28.34,31.5,33.5,35.5,37.5,42,44)

Iteration 7

E[tw]=(0,2,4,6,8,10,12,14,16,20.34,22.34,24.34,26.34,28.34,31.5,33.5,35.5,37.5,42.5,44.67)

Iteration 8

E[tw]=(0,2,4,6,8,10,12,14,16,20.34,22.34,24.34,26.34,28.34,31.5,33.5,35.5,37.5,42.5,45)

Iteration 9

E[tw]=(0,2,4,6,8,10,12,14,16,20.34,22.34,24.34,26.34,28.34,31.5,33.5,35.5,37.5,42.5,45)

Scenario 0 =(0,2,4,6,8,10,12,14,16,19,21,23,25,27,31,33,35,37,42,45)

Scenario 1 =(0,2,4,6,8,10,12,14,16,21,23,25,27,29,31,33,35,37,42,45)

Scenario 2 =(0,2,4,6,8,10,12,14,16,21,23,25,27,29,32,34,36,38,43,45)

Scenario 3 =(0,2,4,6,8,10,12,14,16,21,23,25,27,29,32,34,36,38,43,45)

tb=(0,2,4,6,8,10,12,14,16,20.34,22.34,24.34,26.34,28.34,31.5,33.5,35.5,37.5,42.5,45)

Recommended scenario:1

Schedule: (0,2,4,6,8,10,12,14,16,21,23,25,27,29,31,33,35,37,42,45)

Objective value=110503

Reliability=0.17

Bibliography

- [AMO93] R. Ahuja, T. Magnanti, and J. Orlin. *Network Flows*. Prentice-Hall, 1993.
- [Bar93] C. Barnhart. Dual-ascent methods for large-scale multicommodity flow problems. *Naval Research Logistics*, pages 305–324, 1993.
- [Ber95] D.P. Bertsekas. *Nonlinear Programming*. Athena Scientific, 1995.
- [BG92] D. P. Bertsekas and R. G. Gallager. *Data Networks*. Prentice-Hall, 1992.
- [BHV96] C. Barnhart, C. Hane, and P. Vance. Integer multicommodity flow problems. Technical report, LSOG Working paper, MIT, 1996.
- [BL97] J. R. Birge and F. Louveaux. *Introduction to Stochastic Programming*. Springer-Verlag, 1997.
- [Erm88] Yu. Ermoliev. *Stochastic Quasigradient Methods, in Numerical Techniques for Stochastic Optimization*, chapter 6, pages 141–185. Springer-Verlag, 1988.
- [Geo72] A.M. Geoffrion. *Perspectives in optimization: a collection of expository papers*. Addison-Wesley Pub. Co., 1972.
- [HS96] J. Higle and Sen S. *Stochastic Decomposition: a statistical method for large scale stochastic linear programming*. Kluwer Academic Publishers, 1996.
- [Jau89] F.J. Jauffred. Modelo para la asignacion de flujos de carros de ferrocarril. Bachelors Thesis submitted to the School of Engineering of the National Autonomous University of Mexico, 1989.

- [Kim97] D. Kim. *Large scale transportation network design: models, algorithms and applications*. PhD thesis, Massachusetts Institute of Technology, 1997.
- [KW94] P. Kall and S. Wallace. *Stochastic Programming*. John Wiley and Sons, 1994.
- [Las70] L. S. Lasdon. *Optimization Theory for Large Systems*. Macmillan Publishing CO., INC., 1970.
- [MR95] R. Motwani and P. Raghavan. *Randomized Algorithms*. Cambridge University Press, 1995.
- [MW84] T. L. Magnanti and R. T. Wong. Network design and transportation planning: Models and algorithms. *Transportation Science*, pages 1–55, 1984.
- [NZ96] S. S. Nielsen and S. A. Zenios. Solving multistage stochastic programs on massively parallel computers. *Mathematical programming*, pages 227–250, 1996.
- [Pol87] B. T. Polyak. *Introduction to Optimization*. Optimization Software, Inc., 1987.
- [RO93] O. Richetta and A.R. Odoni. Solving optimally the static ground holding policy problem in air traffic control. *Transportation Science*, pages 228–238, 1993.
- [Sha79] J.F. Shapiro. *Mathematical Programming: Structures and Algorithms*. John Wiley and Sons, 1979.
- [She85] Y. Sheffi. *Urban Transportation Networks*. Prentice-Hall, 1985.
- [VBO94] P.B. Vranas, D. Bertsimas, and A. Odoni. The multiperiod ground holding problem in air traffic control. *Operations Research*, pages 249–261, 1994.
- [Wet88] R. Wets. *Large Scale Linear Programming Techniques, in Numerical Techniques for Stochastic Optimization*, chapter 3, pages 65–94. Springer-Verlag, 1988.

- [Wet96] R.J. Wets. Challenges in stochastic programming. *Mathematical Programming*, pages 115–136, 1996.
- [YK97] G. Yu and P. Kouvelis. *Robust Discrete Optimization and its Applications*. Kluwer Academic Publishers, 1997.

THESIS PROCESSING SLIP

FIXED FIELD: ill. _____ name _____

index _____ biblio _____

► COPIES: Archives Aero Dewey Eng Hum
Lindgren Music Rotch Science

TITLE VARIES: ► _____

NAME VARIES: ► Javier _____

IMPRINT: (COPYRIGHT) _____

► COLLATION: 118 l _____

► ADD. DEGREE: _____ ► DEPT.: _____

SUPERVISORS: _____

NOTES:

cat'r:

date:

► DEPT: C.E.

page:
► <u>F49</u>

► YEAR: 1998 ► DEGREE: Ph.D.

► NAME: JAUFFRED, Francisco J.