1

An Interior-Point and Decomposition Approach to Multiple Stage Stochastic Programming

Shuzhong Zhang

Let us start our discussion with a famous textbook example: the newsboy problem. The story goes like this. Every morning a newsboy has to decide how many newspapers to buy from a newspaper publisher. Let us assume that the publisher sells the newspaper to the boy at the price of \$2 each paper, and the boy then sells the newspaper along the street at the price of \$5 per copy. In the end of the day, the newsboy may return any unsold copies to the publisher at \$1 for each copy. The profit of the newsboy in this business depends, obviously, on the success of the sales and his initial decision on the order quantity. Unfortunately, the problem is that, as it is always the case, one cannot really predict the future with certainty. To make our analysis simple, let us further assume that there are only two possible scenarios: (1) the newsboy can sell 100 copies a day, or, (2) in the case of a boring day, he can only sell 50 copies. Furthermore, let us assume that the chance for a day with some exciting news is lower. Hence, (1) occurs with probability 0.25, and (2) occurs with probability 0.75. In fact, for a person with mathematical background, it is more convenient to introduce a random variable, ω, to denote the demand for this purpose,

$$\omega = \begin{cases} 100, & \text{with probability } 0.25 \\ 50, & \text{with probability } 0.75. \end{cases}$$

After some struggling, one can figure out that we need to solve the following optimization problem

$$\begin{array}{lllll} (NB) & \text{minimize} & 2x & + & \mathsf{E}\,[-5y_\omega - z_\omega] \\ & \text{subject to} & x \geq 0 & & \\ & & & & y_\omega \leq \omega \\ & & & & y_\omega + z_\omega = x \\ & & & & y_\omega \geq 0,\ z_\omega \geq 0. \end{array}$$

Let us explain what is going on here. In the above model, x stands for the quantity of newspapers to order in the morning, and y_ω is the amount of sold newspapers, which is bounded by the demand ω. Finally, z_ω is the amount of newspapers to be returned to the publisher at the end of the day. A closer look suggests that the objective is actually to maximize the expected profit, which is the revenue minus the costs.

The model looks like a pretty ordinary optimization problem, except that there is a random variable ω in the constraint. The model is known as *two-stage stochastic linear programming*, as it involves decisions at two stages: (1) the decision at the beginning of the day, namely x; (2) the decision to be made during the day, namely y_ω and z_ω, given that x is already decided. Moreover, all the relationships happen to be linear. Clearly, one may consider extended models where more than two stages of decisions have to be made sequentially, depending on the newly arrived information on the uncertain factors. That more general case is naturally termed *multi-stage stochastic linear programming*.

Since ω has only two possibly outcomes in our case, we may write (NB) equivalently as a usual deterministic linear programming problem as follows

$$\begin{array}{lllll} (DNB) & \text{minimize} & 2x & + & 0.25\times[-5y_1-z_1]+0.75\times[-5y_2-z_2] \\ & \text{subject to} & x\geq 0 & & \\ & & & & y_1\leq 100 \\ & & & & y_1+z_1=x \\ & & & & y_1\geq 0,\ z_1\geq 0 \\ & & & & y_2\leq 50 \\ & & & & y_2+z_2=x \\ & & & & y_2\geq 0,\ z_2\geq 0. \end{array}$$

If one uses a linear programming solver, then the solution can be found to be

$$[x^*, y_1^*, z_1^*, y_2^*, z_2^*] = [64.2569, 64.2569, 0, 50, 14.2569].$$

The interpretation is: one should buy 64.2569 copies of the newspaper in the morning from the publisher, and then sell them all if it turns out to be a good day, and sell 50 copies and return 14.2569 copies to the publisher if it is a bad day.

The second stage problem is known as *recourse*. In the above example, it is to decide on y_ω and z_ω, provided that x was already done and the state ω is just observed. Its analog in control theory is called the feedback control: observing the true state of the world and then decide the best action to take. In fact, in the above newsboy's situation, the recourse problem is extremely simple. It can even be explicitly solved and written out as: $y_\omega = \max\{\omega, x\}$ and $z_\omega = \max\{x-\omega, 0\}$. In most other applications, of course, this is not possible.

So, the good news for the newsboy is that the decision problem can be solved by some ordinary LP solver. But why stop? There are so many real

decision problems where the future uncertainty will have a great impact on the quality of the decision we make now. So the idea of the above approach should be extended. This is exactly what happened. But there is an intrinsic difficulty with the extension, known as the dimensionality curse. That is to say, as we introduce reasonably fine (but finite) approximations of the underlying random variables (or stochastic process), and try to go beyond two stages, then the total amount of scenarios will explode very quickly, in fact, exponentially. As a result, even though we end up with a deterministic optimization problem to solve, the size of the problem, for most real applications, will be far too large to be plugged into an existing optimization solver. We know that the situation will not improve if we just wait for more powerful computers to appear, as the problem is structural. In stochastic programming, people therefore look for better solution methods, exploiting the very nature of multiple-stage stochastic programming.

In this chapter we shall introduce a particular method of this type, using the so called homogeneous self-dual embedding technique and the central path following method. The results presented in this chapter are based on the author's earlier papers, [5] and [6], and are made simpler for the expository purpose. In [6], a convex objective function is allowed.

1.1 Two-stage stochastic linear programming

Before we talk about the solution methods, let us first consider the models. We start with two-stage stochastic linear programming. Two stage stochastic linear programming is cast as

$$\begin{array}{lllll} (2SLP) & \text{minimize} & c_0^T x_0 & + & \mathsf{E}\left[c_\omega^T x_\omega\right] \\ & \text{subject to} & W_0 x_0 = h_0 & & \\ & & x_0 \geq 0 & & \\ & & & & B_\omega x_0 + W_\omega x_\omega = h_\omega \\ & & & & x_\omega \geq 0, \end{array}$$

where x_0 is the first stage decision variable, and ω is the uncertain factor (a random variable), and x_ω is the corresponding recourse action.

Clearly, the newsboy problem is just a special case of this model.

Now let us further assume that ω takes discrete values, say, there are in total N different scenarios. Then, we can explicitly write down the deterministic equivalent of $(2SLP)$ just like in the newsboy's case

$$\begin{array}{lllll} (2DSLP) & \text{minimize} & c_0^T x_0 & + & \sum_{n=1}^N \pi_n \left[c_n^T x_n\right] \\ & \text{subject to} & W_0 x_0 = h_0 & & \\ & & x_0 \geq 0 & & \\ & & & & B_n x_0 + W_n x_n = h_n, \\ & & & & x_n \geq 0,\ n = 1, ..., N, \end{array}$$

where π_n is the probability of the event $\omega = n$, $n = 1, ..., N$.

If one examines closely the constraint matrix in $(2DSLP)$, then one can observe an L-shaped block structure. That is to say, most part of the matrix contains only zero elements, and the possible non-zero blocks form a fallen L letter viewed from a far distance. Based on this observation, a famous simplex-type method was introduced by Van Slyke and Wets [23]. For a detailed account of the method, one is referred to, e.g., the recent textbooks on stochastic programming, [9] and [17]. In this chapter however, we will concentrate on a quite different approach, based on the interior point method. No matter what, the key idea here is always to make use of the structure so that a decomposition of some sort can be applied. In fact there are, broadly speaking, two types of decomposition approaches: (1) the scenario-based decomposition, and (2) the recourse-based decomposition. The L-shaped method belongs to the second category. That is to say, we employ an existing method, say the simplex method, to solve $(2DSLP)$, and in the process of implementing the method we make use of the special structure of $(2DSLP)$, and decompose the computation of the data as much as possible. Another good example of this type is Birge and Qi's decomposition algorithm, [10], based on Karmarkar's original method for linear programming. Our method to be introduced is in this category as well.

A typical scenario-based method is based on the so-called Lagrangian multiplier approach, or, the augmented Lagrangian multiplier approach; see [21]. The key observation is that, $(2DSLP)$ can be rewritten as

$$
\begin{array}{lll}
(2DSLP)' & \text{minimize} & \sum_{n=1}^{N} \pi_n \left[c_0^T x_{0n} + c_n^T x_n \right] \\
 & \text{subject to} & W_0 x_{0n} = h_0 \\
 & & x_{0n} \geq 0 \\
 & & B_n x_{0n} + W_n x_n = h_n, \\
 & & x_n \geq 0, \ n = 1, ..., N, \\
 & & x_{0n} - \sum_{n=1}^{N} \pi_n x_{0n} = 0, \ n = 1, ..., N.
\end{array}
$$

The last constraint in $(2DSLP)'$ is called *non-anticipativity*, i.e., one is not allowed, however desirable it may be, to use the second stage information in the first stage decision. This reformulation suggests that, if the optimal Lagrangian multipliers for the last set of constraints would be known, say, to be equal to y_n, $n = 1, ..., N$, then the problem could be solved by simply putting them in the objective, i.e.,

$$
\begin{array}{ll}
\text{minimize} & \sum_{n=1}^{N} \left\{ \pi_n \left[c_0^T x_{0n} + c_n^T x_n \right] + y_n \left[x_{0n} - \sum_{n=1}^{N} \pi_n x_{0n} \right] \right\} \\
\text{subject to} & W_0 x_{0n} = h_0 \\
 & x_{0n} \geq 0 \\
 & B_n x_{0n} + W_n x_n = h_n, \\
 & x_n \geq 0, \ n = 1, ..., N.
\end{array}
$$

The above problem is completely separable in terms of scenarios and hence

can be solved by quickly solving N small size linear programs. This approach also works when one introduces an additional pure quadratic penalty term of the constraints $x_{0n} - \sum_{n=1}^{N} \pi_n x_{0n} = 0$, $n = 1, ..., N$, in the objective, yielding the so-called augmented Lagrangian approach. The remaining task is to come up with a good scheme to update the Lagrangian multipliers y_n in order to accelerate convergence, for which existing theory for the Lagrangian method can be used. We note that the scenario-based approach extends easily to the multiple stage framework, even when the objective is nonlinear. One drawback, however, is the slow convergence. In the bid to a faster convergence speed, we turn to the state-of-the-art interior point method, where the number of iterations is known to be insensitive to the problem dimension. Certainly this will be an extremely important property in the real application.

Before we introduce the new method, let us first consider an application of the two-stage stochastic programming model to motivate the solution method.

1.2 A case study

We consider the following two-period problem. An investor can invest in a bank account, a stock index, and European (exchange listed) options on this index with different maturities. We denote the stock index by S. Current time is denoted by t_0, and the expiration dates of the options by t_1 and t_2 with $t_0 < t_1 < t_2$. At t_0 the investor forms a portfolio consisting of some amount of money invested in the bank account, some in the stock index, and a set of options on the stock index. At time t_1 he/she may revise his/her portfolio, depending on the value of the index at t_1, i.e. he/she can change some of the existing positions in the options and/or buy new options starting from t_1 and maturing at t_2. The investor's goal is to guarantee that the value of the portfolio is always above a given level depending on the index at t_2, and that the expected value of the portfolio is maximized at the horizon of the investment.

Assume that the level of the stock index is S_0 at time t_0, S_1 at time t_1, and S_2 at time t_2. Moreover, there are n European puts and calls struck at K_l^j with $l = 1, 2, ..., n$, respectively, where $j = 1, 2$ denotes the expiration of the options t_j. Let $Q^p_{t_i t_j}(S) \in \Re^n$ denote the n-dimensional vector whose l-th component represents the price of buying a put option at time t_i maturing at t_j with strike price K_l, while the stock index at t_i is S. Similarly, denote $Q^c_{t_i t_j}(S) \in \Re^n$ to be the n-dimensional vector which l-th component represent the price of buying a call option at time t_i maturing at t_j with strike price K_l while the stock index at t_i is S. The risk-free interest rate from t_0 to t_1 is denoted by r_1, the risk-free interest rate from t_0 to t_2 is denoted by r_2, and the forward rate from t_1 to t_2 is denoted by f_2. Now, let

$x^p_{t_it_j} \in \Re^n$ denote the amount of put options purchased at time t_i maturing at t_j, and $x^c_{t_it_j} \in \Re^n$ be the amount of call options purchased at time t_i maturing at t_j. Let x^s_0 be the amount invested in the stock index, and x^f_0 be the amount invested at t_0 in the money-market account. Similarly, let x^s_1 be the amount invested in the stock index and x^f_1 be the amount invested in the money-market account at t_1. The decision variables $x^p_{t_0t_j}$ and $x^c_{t_0t_j}$ with $j = 1, 2$, and x^s_0 and x^f_0 denote the first-stage variables. The decision variables $x^p_{t_1t_2}$ and $x^c_{t_1t_2}$, and x^s_1 and x^f_1 denote the second-stage variables. Suppose that the initial budget for the investment is B.

Clearly, the following initial budget equation should hold:

$$B = x^s_0 S_0 + x^f_0 + \sum_{j=1}^{2} \langle x^p_{t_0t_j}, Q^p_{t_0t_j}(S_0)\rangle + \sum_{j=1}^{2} \langle x^c_{t_0t_j}, Q^c_{t_0t_j}(S_0)\rangle, \tag{1.1}$$

where for notational simplicity in this context we used $\langle x, y\rangle$ as the inner product between x and y, i.e., $\langle x, y\rangle = x^T y$.

At t_1 the value of the portfolio is given by:

$$\begin{aligned} V(t_1, S_1; x^s, x^f, x^p, x^c) &= x^s_0 S_1 + x^f_0 \exp(r_1(t_1 - t_0)) \\ &+\langle (\mathcal{K}_1 - S_1 e)^+, x^p_{t_0t_1}\rangle + \langle (S_1 e - \mathcal{K}_1)^+, x^c_{t_0t_1}\rangle \\ &+\langle Q^p_{t_1t_2}(S_1), x^p_{t_0t_2}\rangle + \langle Q^c_{t_1t_2}(S_1), x^c_{t_0t_2}\rangle \end{aligned} \tag{1.2}$$

where $\mathcal{K}_1 = (K^1_1, ..., K^1_n)^T$, and for given $y \in \Re^n$, y^+ denotes the vector

$$(\max\{y_1, 0\}, ..., \max\{y_n, 0\})^T.$$

The second-stage recourse problem is as follows. First, there is an intermediate budget constraint:

$$\begin{aligned} &V(t_1, S_1; x^s, x^f, x^p, x^c) \\ &= \quad x^s_1 S_1 + x^f_1 + \langle Q^p_{t_1t_2}(S_1), x^p_{t_1t_2}\rangle + \langle Q^c_{t_1t_2}(S_1), x^c_{t_1t_2}\rangle. \end{aligned} \tag{1.3}$$

Second, the value of the portfolio at the horizon is given by:

$$\begin{aligned} &V(t_2, S_2; x^s, x^f, x^p, x^c) \\ &= \quad x^s_1 S_2 + x^f_1 \exp(f_2(t_2 - t_1)) \\ &\quad +\langle (\mathcal{K}_2 - S_2 e)^+, x^p_{t_1t_2}\rangle + \langle (S_2 e - \mathcal{K}_2)^+, x^c_{t_1t_2}\rangle. \end{aligned} \tag{1.4}$$

We require the value of the portfolio at the horizon never to be less than $c_0 S_2 + c_1$ with $c_0 \geq 0$ and $c_1 > 0$. Using the piecewise linearity of $V(t_2, S_2; x^p, x^c)$, this yields:

$$V(t_2, K^2_i; x^s, x^f, x^p, x^c) \geq c_0 K^2_i + c_1 \text{ for } i = 1, ..., n \tag{1.5}$$

and

$$V(t_2, 0; x^s, x^f, x^p, x^c) \geq c_1$$

and

$$V'_{S_2}(t_2, S_2; x^s, x^f, x^p, x^c) \mid_{S_2=K_n^2+} \geq c_0.$$

These constraints are all linear in terms of x^s, x^f, x^p and x^c.

Finally, we require the probability that the portfolio value will be above a given threshold value $c_2 > 0$ to be at least λ $(0 < \lambda < 1)$. This, again by piecewise linearity, can be modelled by selecting a given I $(1 \leq I \leq n)$, and adding the following constraints:

$$V(t_2, K_i; x^s, x^f, x^p, x^c) \geq c_2 \text{ for } i = I, I+1, ..., n. \tag{1.6}$$

Similar constraints can be added to the model at t_1.

The expected value of the portfolio at t_2 is given by:

$$\begin{aligned} &\mathsf{E}\left[V(t_2, S_2; x^s, x^f, x^p, x^c)\right] \\ &\quad = \quad x_1^s \mathsf{E}\left[S_2\right] + x_1^f \exp(r(t_2 - t_1)) \\ &\qquad +\langle \mathsf{E}\left[(\mathcal{K}_2 - S_2 e)^+\right], x^p_{t_1 t_2}\rangle + \langle \mathsf{E}\left[(S_2 e - \mathcal{K}_2)^+\right], x^c_{t_1 t_2}\rangle. \end{aligned} \tag{1.7}$$

The optioned portfolio selection problem is now well defined as a two-stage stochastic linear program:

$$\begin{array}{ll} \max & w_1 \mathsf{E}\left[V(t_1, S_1; x^s, x^f, x^p, x^c)\right] + w_2 \mathsf{E}\left[V(t_2, S_2; x^s, x^f, x^p, x^c)\right] \\ \text{s.t.} & (1.1), (1.3), (1.5) \text{ and } (1.6) \end{array}$$

where w_1 and w_2 $(w_2 > w_1)$ are weights for the first and second stage expected values.

Problems of this nature exist widely in the investment world. The above model is discussed in Berkelaar, Dert, Oldenkamp and Zhang [5]; two of the authors work for the ABN-AMRO Bank and manage financial products similar to the above described model.

1.3 Multiple stage stochastic programming

There is of course no need to restrict ourselves to the two stage models only. It is very natural to extend them to *multiple stage stochastic programming*. For simplicity, we consider here the problem with a linear objective.

Let us assume that there are K decision stages. At stage 0, which is the current time, no information about the future is known for certain. However, we can model the future development using a *scenario tree*; see Figure 1.1. Each node in this tree is layered according to the stage, where it may occur. At stage t, we let the set of nodes be $\mathcal{F}_t$. Technically, $\mathcal{F}_t$ is also known as filtration.

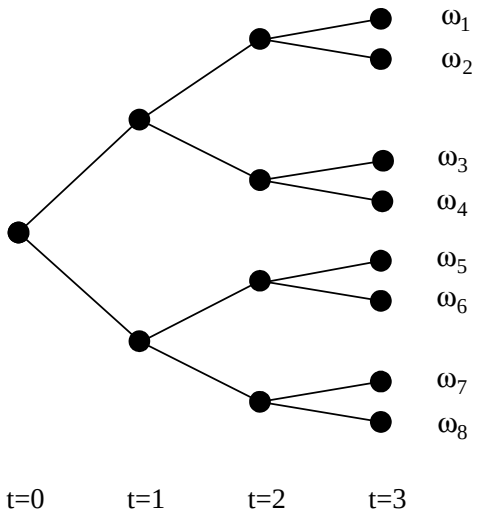

FIGURE 1.1. Scenario Tree

The multi-stage stochastic programming model can thus be described as

$$
\begin{array}{lllll}
(MSLP) & & & & \\
\min & c_0^T x_0 & & +\sum_{t=1}^{K} \mathsf{E}\left[c_t(\omega)^T x_t(\omega)\right] & \\
\text{s.t.} & W_0 x_0 & & & = h_0 \\
& x_0 \geq 0 & & & \\
& & B_t(\omega) x_{t-1}(\omega) & +W_t(\omega) x_t(\omega) & = h_t(\omega), \\
& & x_t(\omega) \geq 0, & \omega \in \mathcal{F}_t,\ t = 1, ..., K. &
\end{array}
$$

More specifically, we used finite scenarios in the tree, and so each node has a predecessor, except for node 0, which is the current stage, and each node has possibly several successors, except for the last layer at final stage K. Variables defined on ω are denoted by subscripts n and t, where $n \in \mathcal{F}_t$. The deterministic equivalent problem can now be cast as

$$
\begin{array}{lllll}
(MDSLP) & & & & \\
\min & c_0^T x_0 & & +\sum_{t=1}^{K} \sum_{n \in \mathcal{F}_t} \pi_{nt} c_{nt}^T x_{nt} & \\
\text{s.t.} & W_0 x_0 & & & = h_0 \\
& & B_{nt} x_{a(n),t-1} & +W_{nt} x_{nt} & = h_{nt}, \\
& x_0 \geq 0, & x_{nt} \geq 0, & n \in \mathcal{F}_t,\ t = 1, ..., K. &
\end{array}
$$

where subscript (n, t) stands for scenario n at stage t, π_{nt} is its probability,

$(a(n), t-1)$ is the ancestor of (n,t), and $(C(n), t+1)$ will be denoted the set of children of (n,t). As a convention we let $n=0$ denote the current state. Therefore, the decision variables are x_0 (the immediate action) and x_{nt} (the recourse action to be taken if scenario n unfolds at stage t).

In our notation, each node on the scenario tree (at different stages) is associated with a unique n. In this sense, the association t in the notation (n,t) is redundant. Nevertheless, we shall keep it in places where confusion is possible.

In general, the above formulation $(MDSLP)$ can be a large size linear program. The number of variables is of the order $O(C^K)$ where C represents the number of children following each scenario at an intermediate stage. For practical purposes we may assume that each of the matrices B_{nt}, and W_{nt} are reasonably sized. For technical reasons we also assume that all the matrices W_{nt} have full row ranks.

This problem, $(MDSLP)$, has the following dual problem

$$\begin{aligned}
\max \quad & h_0^T y_0 + \sum_{t=1}^{K} \sum_{n \in \mathcal{F}_t} h_{nt}^T y_{nt} \\
\text{s.t.} \quad & W_0^T y_0 + \sum_{m \in C(0)} B_{m1}^T y_{m1} + s_0 = c_0 \\
& W_{nt}^T y_{nt} + \sum_{m \in C(n)} B_{m,t+1}^T y_{m,t+1} + s_{nt} = \pi_{nt} c_{nt},\ t = 1, ..., K-1,\ n \in \mathcal{F}_t \\
& W_{nK}^T y_{nK} + s_{nK} = \pi_{nK} c_{nK},\ n \in \mathcal{F}_K \\
& x_0 \geq 0,\ x_{nt} \geq 0,\ s_0 \geq 0,\ s_{nt} \geq 0,\ n \in \mathcal{F}_t,\ t = 1, ..., K.
\end{aligned}$$

1.4 An interior point method

Before we offer our decomposition solution method for solving $(MDSLP)$, let us first introduce a particular type of interior point method, known as the *homogeneous self-dual and path-following* algorithm. To start with, consider a standard linear program

$$\begin{array}{lll} (P) & \text{minimize} & c^T x \\ & \text{subject to} & Ax = b \\ & & x \geq 0, \end{array}$$

where $A \in \Re^{m \times n}$, $b \in \Re^m$ and $c \in \Re^n$. Moreover, without loss of generality, let us assume that the rank of A is m. The dual of (P) is

$$\begin{array}{lll} (D) & \text{maximize} & b^T y \\ & \text{subject to} & A^T y + s = c \\ & & s \geq 0. \end{array}$$

The famous analytic central path is defined to be

$$\{(x(\mu), y(\mu), s(\mu)) \mid Ax(\mu) = b,\ x(\mu) > 0,$$

$$A^T y(\mu) + s(\mu) = c,\ s(\mu) > 0,\ x_i(\mu)s_i(\mu) = \mu,\ i = 1, 2, ..., n\}.$$

It is well known that if (P) and (D) both have interior solutions, i.e.

$$\{x \mid Ax = b,\ x > 0\} \neq \emptyset \text{ and } \left\{(y, s) \mid A^T y + s = c,\ s > 0\right\} \neq \emptyset, \quad (1.8)$$

then the analytic central path exists, and vice versa.

The limit, $\lim_{\mu \downarrow 0}(x(\mu), y(\mu), s(\mu)) =: (x(0), y(0), s(0))$, exists, and it is the analytic center of the optimal set. Hence it is optimal. For detailed proofs of those nowadays well known facts, one is referred to the book of Roos, Terlaky and Vial [22].

Most interior point methods, and certainly by definition the so-called path-following methods, are based on tracing the analytic central path till close to $(x(0), y(0), s(0))$.

All is well, except for the fact that one may not know whether the interior point condition (1.8) is satisfied, and if yes, how to get hold of one initial interior point solution. This difficulty was resolved beautifully by Ye, Todd and Mizuno in [27], and later was simplified by Xu, Hung and Ye [24]. A detailed description of an efficient numerical implementation of the method was documented in [1]. The technique is called *homogeneous self-dual embedding*. In fact, the iterates produced by the algorithm in [27] will be identical to that produced by the one in [24], provided that they have identical parameters and initial points. Hence, in a sense, these two algorithms are simply identical for linear programming, and below we shall choose to present the method based on [24].

The idea of dealing with homogeneous self-dual systems can be traced back to Goldman and Tucker [14]. In [14] the following system is considered:

$$\begin{array}{cccccc} & Ax & -b\tau & = & 0 \\ -A^T y & & +c\tau & \geq & 0 \\ b^T y & -c^T x & & \geq & 0 \\ & x \geq 0, & \tau \geq 0. & & \end{array}$$

Clearly, this system is homogeneous and has a skew-symmetric constraint matrix leading to the notion of self-duality. For convenience, additional variables are introduced to replace the inequality constraints, yielding

$$(H) \left\{ \begin{array}{ccccccc} & Ax & & -b\tau & & = & 0 \\ -A^T y & & -s & +c\tau & & = & 0 \\ b^T y & -c^T x & & & -\kappa & = & 0 \\ & x \geq 0, & s \geq 0, & \tau \geq 0, & \kappa \geq 0. & & \end{array} \right.$$

If system (H) has a solution $(y^*, x^*, s^*, \tau^*, \kappa^*)$ such that $\tau^* > 0$ and $\kappa^* = 0$, then an optimal solution to (P) is simply x^*/τ^* and an optimal solution to (D) is $(y^*/\tau^*, s^*/\tau^*)$.

However, (H) also contains trivial solutions such as

$$(y, x, s, \tau, \kappa) = (0, 0, 0, 0, 0),$$

from which no information concerning solutions for (P) and (D) can be deduced. To avoid trivial solutions, we note the following fundamental result concerning (H) due to Goldman and Tucker [14].

Theorem 1.1 *There exists a solution $(y^*, x^*, s^*, \tau^*, \kappa^*)$ for (H) such that*

$$x^* + s^* > 0 \text{ and } \tau^* + \kappa^* > 0.$$

It is elementary to check that any solution (y, x, s, τ, κ) to (H) necessarily satisfies

$$x^T s + \tau\kappa = 0.$$

That is why the Goldman-Tucker type solution is called a strictly complementary solution, since it implies that either x_i^* or s_i^* is zero (and not both) for all i, and either τ^* or κ^* is zero (and not both). Based on a strictly complementary solution for (H), solutions for the original linear programming problems (P) and (D) can easily be found, as the next lemma demonstrates.

Lemma 1.2 *Let $(y^*, x^*, s^*, \tau^*, \kappa^*)$ be as in Theorem 1.1. If $\tau^* > 0$, then x^*/τ^* is an optimal solution to (P) and $(y^*/\tau^*, s^*/\tau^*)$ is an optimal solution to (D). If $\tau^* = 0$, then $\kappa^* > 0$, i.e. $b^T y^* - c^T x^* > 0$. In this case, if $b^T y^* > 0$, then (P) is infeasible, and if $c^T x^* < 0$, then (D) is infeasible.*

One may argue, however, that (H) is nothing but a reformulation of the problem, as finding the desirable $(y^*, x^*, s^*, \tau^*, \kappa^*)$ as stipulated in Lemma 1.2 is equally difficult as solving (P) and (D).

This view proves to be a little narrow minded. Let us see why.

Take any vectors $0 < x^0 \in \Re^n$, $y^0 \in \Re^m$, $0 < s^0 \in \Re^n$, $\tau^0 > 0$ and $\kappa^0 > 0$, such that $x_i^0 s_i^0 = 1$, $i = 1, ..., n$, and $\tau^0\kappa^0 = 1$. In practice, one may simply let $x^0 = e$, $y^0 = 0$ and $s^0 = e$, where e is the all-one vector, and $\tau^0 = \kappa^0 = 1$. Denote

$$\begin{cases} r_p^0 &= Ax^0 - b\tau^0 \\ r_d^0 &= -A^T y^0 + c\tau^0 - s^0 \\ r_g^0 &= b^T y^0 - c^T x^0 - \kappa^0. \end{cases} \tag{1.9}$$

Now we introduce a parameterized system of nonlinear equations

$$(H_\mu) \begin{cases} Ax - b\tau &= \mu r_p^0 \\ -A^T y - s + c\tau &= \mu r_d^0 \\ b^T y - c^T x - \kappa &= \mu r_g^0 \\ x_i s_i &= \mu,\ i = 1, ..., n, \\ \tau\kappa &= \mu. \end{cases}$$

Obviously, (H_1) has a solution, e.g.,

$$(y, x, s, \tau, \kappa) = (y^0, x^0, s^0, \tau^0, \kappa^0).$$

Now we argue that (H_μ) always has a unique solution

$$(y(\mu), x(\mu), s(\mu), \tau(\mu), \kappa(\mu))$$

for any fixed $\mu > 0$.

To see this we consider a linear programming problem

$$\begin{array}{ll} \text{minimize} & -\mu(r_p^0)^T y - \mu(r_d^0)^T x - \mu r_g^0 \tau \\ \text{subject to} & Ax - b\tau = \mu r_p^0 \\ & -A^T y + c\tau \geq \mu r_d^0 \\ & b^T y - c^T x \geq \mu r_g^0 \\ & x \geq 0,\ \tau \geq 0. \end{array}$$

This problem has an interior feasible solution, e.g.

$$(y, x, \tau) = \mu(y^0, x^0, \tau^0).$$

Moreover, it is self-dual, i.e., its dual problem is the same as itself. Hence, the dual also has an interior feasible solution, and therefore the primal-dual analytic central path exists. The point on the central path with parameter μ, after introducing the slack variables s and κ for the second and third constraints, is the unique solution for (H_μ). This leads naturally to the idea of following the central path by reducing μ, starting from $\mu = 1$. As is standard nowadays in primal-dual interior point methods, we solve the system (H_μ) by applying Newton's method. In particular, we use the predictor-corrector approach proposed by Mizuno, Todd and Ye in [19]. To facilitate the method, we introduce a concept called the neighborhood of the central path with size $0 < \beta < 1$

$$\mathcal{N}(\beta) = \left\{ \left[\begin{array}{c} x \\ s \\ \tau \\ \kappa \end{array} \right] \;\middle|\; \delta(x, s, \tau, \kappa) := \left\| e - \frac{n+1}{x^T s + \tau\kappa} \left[\begin{array}{c} Xs \\ \tau\kappa \end{array} \right] \right\| \leq \beta \right\} \quad (1.10)$$

where X stands for the diagonal matrix whose elements are taken from the vector x. This notation is standard in interior point methods and will be used throughout the chapter

We now wish to update (x, s, τ, κ) in such a way that it is in a good position, i.e., the iterate stays inside a certain neighborhood of the central path, and at the same time, the duality gap, namely the value of $x^T s + \tau\kappa$, is reduced. These two wishes, however, are often contradicting to each other. A compromise is proposed in the predictor-corrector approach, in which we simply separate these two tasks by using two neighborhoods, one is bigger than the other, say one is $\mathcal{N}(1/4)$ and the other one is $\mathcal{N}(1/2)$. Once we are inside the smaller neighborhood, then we consider ourselves well positioned, and can afford to sacrifice the centrality to a certain degree while concentrating on reducing the duality gap at the step. This step is termed

predictor. As soon as we are on the boundary of the big neighborhood, this is the sign of alarming and we try to move back into the smaller neighborhood without damaging the duality gap. This is the so called *corrector* step.

Specifically, the predictor step is to take a Newton direction, $(d_x, d_s, d_\tau, d_\kappa)$, based on linearizing (H_μ) while targeting at $\mu = 0$. The corresponding system of linear equations is

$$(Predictor) \left\{ \begin{array}{rcl} Ad_x - bd_\tau & = & -r_p \\ -A^T d_y - d_s + cd_\tau & = & -r_d \\ b^T d_y - c^T d_x - d_\kappa & = & -r_g \\ s_i d_{x_i} + x_i d_{s_i} & = & -x_i s_i,\ i = 1, ..., n, \\ \kappa d_\tau + \tau d_\kappa & = & -\tau\kappa, \end{array} \right.$$

where the new residuals are defined similarly as in (1.9)

$$\left\{ \begin{array}{rcl} r_p & = & Ax - b\tau \\ r_d & = & -A^T y + c\tau - s \\ r_g & = & b^T y - c^T x - \kappa. \end{array} \right.$$

The corrector step aims at reaching the solution of (H_μ) with $\mu = (x^T s + \tau\kappa)/(n+1)$. Hence, the corresponding system of Newton equations is

$$(Corrector) \left\{ \begin{array}{rcl} Ad_x - bd_\tau & = & 0 \\ -A^T d_y - d_s + cd_\tau & = & 0 \\ b^T d_y - c^T d_x - d_\kappa & = & 0 \\ s_i d_{x_i} + x_i d_{s_i} & = & \mu - x_i s_i,\ i = 1, ..., n, \\ \kappa d_\tau + \tau d_\kappa & = & \mu - \tau\kappa. \end{array} \right.$$

The key to note here is that, if one follows either the predictor step, or the corrector step, then it always holds that

$$\left\{ \begin{array}{rcl} r_p & = & \mu r_{p_0} \\ r_d & = & \mu r_{d_0} \\ r_g & = & \mu r_{g_0}. \end{array} \right.$$

In other words, the direction of residuals will not change, only the norm will be reduced, proportional to μ. In general, we need to solve the following linear equations

$$(Newton) \left\{ \begin{array}{rcl} Ad_x - bd_\tau & = & -\rho r_p \\ -A^T d_y - d_s + cd_\tau & = & -\rho r_d \\ b^T d_y - c^T d_x - d_\kappa & = & -\rho r_g \\ s_i d_{x_i} + x_i d_{s_i} & = & (1-\rho)\mu - x_i s_i,\ i = 1, ..., n, \\ \kappa d_\tau + \tau d_\kappa & = & (1-\rho)\mu - \tau\kappa, \end{array} \right. \tag{1.11}$$

where $\rho = 1$ yields the predictor direction, and $\rho = 0$ yields the corrector direction.

Let us solve the linear system (1.11) and update the iterates based on the Newton directions with a steplength $t > 0$, i.e. we let

$$\begin{cases} x(t) & := & x + td_x > 0 \\ y(t) & := & y + td_y \\ s(t) & := & s + td_s > 0 \\ \tau(t) & := & \tau + td_\tau > 0 \\ \kappa(t) & := & \kappa + td_\kappa > 0. \end{cases}$$

Next we study what happens to the new iterate $(x(t), y(t), s(t), \tau(t), \kappa(t))$. First let us study the new duality gap

$$\begin{aligned} x(t)^T s(t) + \tau(t)\kappa(t) &= (x + td_x)^T (s + td_s) + (\tau + td_\tau)(\kappa + td_\kappa) \\ &= x^T s + \tau\kappa + t(x^T d_s + s^T d_x + \tau d_\kappa + \kappa d_\tau) \\ &\quad + t^2(d_x^T d_s + d_\tau d_\kappa) \\ &= (1 - t\rho)(n+1)\mu + t^2(d_x^T d_s + d_\tau d_\kappa) \qquad (1.12) \end{aligned}$$

where we used the fourth and the fifth equations in (1.11). It remains to estimate the second term on the right hand side of (1.12). For this purpose we note from the second equation in (1.11) that

$$d_s = -A^T d_y + cd_\tau + \rho r_d$$

and therefore

$$\begin{aligned} d_x^T d_s &= -(Ad_x)^T d_y + c^T d_x d_\tau + \rho r_d^T d_x \\ &= -(bd_\tau - \rho r_d)^T d_y + c^T d_x d_\tau + \rho r_d^T d_x \\ &= \rho(r_p^T d_y + r_d^T d_x) + (c^T x - b^T y)d_\tau. \qquad (1.13) \end{aligned}$$

Hence,

$$\begin{aligned} & d_x^T d_s + d_\tau d_\kappa \\ &= \rho(r_p^T d_y + r_d^T d_x) + (c^T x - b^T y + d_\kappa)d_\tau \\ &= \rho(r_p^T d_y + r_d^T d_x + r_g d_\tau) \\ &= \rho\left[(Ax - b\tau)^T d_y + (-A^T y + c\tau - s)^T d_x + (b^T y - c^T x - \kappa)d_\tau\right] \\ &= \rho(x^T A^T d_y - \tau b^T d_y - y^T A d_x + \tau c^T d_x \\ &\quad - s^T d_x + b^T y d_\tau - c^T x d_\tau - \kappa d_\tau) \\ &= \rho[x^T(-d_s + cd_\tau + \rho r_d) - \tau b^T d_y - y^T(bd_\tau - \rho r_p) \\ &\quad + \tau c^T d_x - s^T d_x + b^T y d_\tau - c^T x d_\tau - \kappa d_\tau] \\ &= \rho\left[-x^T d_s - s^T d_x + \rho x^T r_d + \rho y^T r_p + \tau(c^T d_x - b^T d_y) - \kappa d_\tau\right] \\ &= \rho\left[\rho(x^T r_d + y^T r_p + \tau r_g) - (x^T d_s + s^T d_x + \kappa d_\tau + \tau d_\kappa)\right]. \qquad (1.14) \end{aligned}$$

On one hand, adding up the terms in the fourth and the fifth equations of (1.11) we obtain

$$\begin{aligned} & x^T d_s + s^T d_x + \kappa d_\tau + \tau d_\kappa \\ &= (1 - \rho)(n+1)\mu - (n+1)\mu = -\rho(n+1)\mu. \qquad (1.15) \end{aligned}$$

On the other hand,

$$\begin{aligned} & x^T r_d + y^T r_p + \tau r_g \\ &= x^T(-A^T y + \tau c - s) + y^T(Ax - \tau b) + \tau(b^T y - c^T x - \kappa) \\ &= -x^T s - \tau\kappa \\ &= -(n+1)\mu. \end{aligned} \tag{1.16}$$

Substituting (1.15) and (1.16) into (1.14) yields

$$d_x^T d_s + d_\tau d_\kappa = 0. \tag{1.17}$$

Now we obtain from (1.12) that

$$x(t)^T s(t) + \tau(t)\kappa(t) = (1 - t\rho)(n+1)\mu. \tag{1.18}$$

Having established how the duality gap, viz. the value of $x(t)^T s(t) + \tau(t)\kappa(t)$, will be improved, next we wish to see how the centering measure,

$$\delta(x(t), s(t), \tau(t), \kappa(t)) = \left\| e - \frac{n+1}{x(t)^T s(t) + \tau(t)\kappa(t)} \begin{bmatrix} X(t)s(t) \\ \tau(t)\kappa(t) \end{bmatrix} \right\|$$

(see (1.10)), is affected by such a move.

Using (1.18), and the fourth and fifth equations in (1.11) it follows that

$$\begin{aligned} & (1 - t\rho)\delta(x(t), s(t), \tau(t), \kappa(t)) \\ &= \left\| e - \frac{1}{\mu} \begin{bmatrix} Xs \\ \tau\kappa \end{bmatrix} - t\left((1-\rho)e - \frac{1}{\mu} \begin{bmatrix} Xs \\ \tau\kappa \end{bmatrix} \right) - \frac{t^2}{\mu} \begin{bmatrix} d_x \circ d_s \\ d_\tau \circ d_\kappa \end{bmatrix} \right\| \end{aligned} \tag{1.19}$$

where $\circ$ stands for the componentwise product of two vectors.

In order to further study the change in the centrality measure, it is crucial to estimate the term $\left\| \begin{bmatrix} d_x \circ d_s \\ d_\tau \circ d_\kappa \end{bmatrix} \right\|$. The last two equations in (1.11) can be rewritten as

$$\begin{aligned} (XS)^{0.5}\left[X^{-0.5}S^{0.5}d_x + X^{0.5}S^{-0.5}d_s\right] &= (1-\rho)\mu e - Xs \\ (\tau\kappa)^{0.5}\left[\tau^{-0.5}\kappa^{0.5}d_\tau + \tau^{0.5}\kappa^{-0.5}d_\kappa\right] &= (1-\rho)\mu - \tau\kappa. \end{aligned}$$

Let

$$\begin{bmatrix} \bar{d}_x \\ \bar{d}_\tau \end{bmatrix} := \begin{bmatrix} X^{-0.5}S^{0.5}d_x \\ \tau^{-0.5}\kappa^{0.5}d_\tau \end{bmatrix} \text{ and } \begin{bmatrix} \bar{d}_s \\ \bar{d}_\kappa \end{bmatrix} := \begin{bmatrix} X^{0.5}S^{-0.5}d_s \\ \tau^{0.5}\kappa^{-0.5}d_\kappa \end{bmatrix}.$$

We then have

$$\begin{bmatrix} \bar{d}_x + \bar{d}_s \\ \bar{d}_\tau + \bar{d}_\kappa \end{bmatrix} = \begin{bmatrix} (XS)^{-0.5} & 0 \\ 0 & (\tau\kappa)^{-0.5} \end{bmatrix} \begin{bmatrix} (1-\rho)\mu e - Xs \\ (1-\rho)\mu - \tau\kappa \end{bmatrix}. \tag{1.20}$$

Moreover,

$$(\bar{d}_x)_i(\bar{d}_s)_i = (d_x)_i(d_s)_i \text{ for } i = 1, ..., n, \text{ and } \bar{d}_\tau \bar{d}_\kappa = d_\tau d_\kappa.$$

Hence

$$\bar{d}_x^T \bar{d}_s + \bar{d}_\tau \bar{d}_\kappa = d_x^T d_s + d_\tau d_\kappa = 0.$$

Therefore

$$\begin{aligned}
\left\| \begin{bmatrix} d_x \circ d_s \\ d_\tau \circ d_\kappa \end{bmatrix} \right\| &\leq \left\| \begin{bmatrix} \bar{d}_x \circ \bar{d}_s \\ \bar{d}_\tau \circ \bar{d}_\kappa \end{bmatrix} \right\| \\
&\leq \left\| \begin{bmatrix} \bar{d}_x \circ \bar{d}_s \\ \bar{d}_\tau \circ \bar{d}_\kappa \end{bmatrix} \right\|_1 \\
&\leq \left\| \begin{bmatrix} \bar{d}_x \\ \bar{d}_\tau \end{bmatrix} \right\| \cdot \left\| \begin{bmatrix} \bar{d}_s \\ \bar{d}_\kappa \end{bmatrix} \right\| \\
&\leq \left(\left\| \begin{bmatrix} \bar{d}_x \\ \bar{d}_\tau \end{bmatrix} \right\|^2 + \left\| \begin{bmatrix} \bar{d}_s \\ \bar{d}_\kappa \end{bmatrix} \right\|^2 \right) /2 \\
&= \left\| \begin{bmatrix} \bar{d}_x + \bar{d}_s \\ \bar{d}_\tau + \bar{d}_\kappa \end{bmatrix} \right\|^2 /2 \qquad (1.21)
\end{aligned}$$

where in the third inequality follows from the Cauchy-Schwarz inequality, and the last equality follows from the orthogonality.

Since $(x, s, \tau, \kappa) \in \mathcal{N}(\beta)$, we have $x_i s_i \geq (1-\beta)\mu$, $i = 1, ..., n$, and $\tau\kappa \geq (1-\beta)\mu$. This yields, using (1.21), that

$$\left\| \begin{bmatrix} d_x \circ d_s \\ d_\tau \circ d_\kappa \end{bmatrix} \right\| \leq \frac{1}{(1-\beta)\mu} \left\| \begin{bmatrix} (1-\rho)\mu e - Xs \\ (1-\rho)\mu - \tau\kappa \end{bmatrix} \right\|^2 /2. \qquad (1.22)$$

Using (1.22), we obtain from (1.19) that

$$\begin{aligned}
&(1-t\rho)\delta(x(t), s(t), \tau(t), \kappa(t)) \\
&\leq (1-t)\delta(x, s, \tau, \kappa) + t\rho\sqrt{n+1} + \frac{t^2}{2(1-\beta)\mu^2} \left\| \begin{bmatrix} (1-\rho)\mu e - Xs \\ (1-\rho)\mu - \tau\kappa \end{bmatrix} \right\|^2. \qquad (1.23)
\end{aligned}$$

Now we are in the position to present the pure form of predictor-corrector homogeneous self-dual interior point algorithm.

Algorithm PCHSD

Step 0 Let $k = 0$.

Step 1 If $k = 0$ or k is even, go to Step 2; otherwise go to Step 3.

Step 2 Let $\rho = 0$. Solve the linear system (1.11) to get the predictor search directions. Let

$$\begin{cases} x^{k+1} & := & x^k + t^* d_x > 0 \\ y^{k+1} & := & y^k + t^* d_y \\ s^{k+1} & := & s^k + t^* d_s > 0 \\ \tau^{k+1} & := & \tau^k + t^* d_\tau > 0 \\ \kappa^{k+1} & := & \kappa^k + t^* d_\kappa > 0, \end{cases}$$

where $t^* > 0$ is chosen to be the maximum step length such that

$$(x^k + t^* d_x, s^k + t^* d_s, \tau^k + t^* d_\tau, \kappa^k + t^* d_\kappa) \in \mathcal{N}(1/2).$$

Let $k := k + 1$ and return to Step 1.

Step 3 Let $\rho = 1$. Solve the linear system (1.11) to get the corrector search directions. Let

$$\begin{cases} x^{k+1} & := & x^k + d_x > 0 \\ y^{k+1} & := & y^k + d_y \\ s^{k+1} & := & s^k + d_s > 0 \\ \tau^{k+1} & := & \tau^k + d_\tau > 0 \\ \kappa^{k+1} & := & \kappa^k + d_\kappa > 0. \end{cases}$$

Let $k := k + 1$ and return to Step 1.

Now we show that each corrector step brings the iterate from $\mathcal{N}(1/2)$ to $\mathcal{N}(1/4)$.

Lemma 1.3 *If* $(x^k, s^k, \tau^k, \kappa^k) \in \mathcal{N}(1/2)$ *and* $\rho = 0$, *then*

$$(x^{k+1}, s^{k+1}, \tau^{k+1}, \kappa^{k+1}) \in \mathcal{N}(1/4).$$

Proof. In the corrector step, we use unitary Newton step, i.e. $t = 1$. Moreover, $\beta = 1/2$, and so by (1.23) we get

$$\delta((x^{k+1}, s^{k+1}, \tau^{k+1}, \kappa^{k+1})) \leq \frac{1}{2(1-\beta)} \delta((x^k, s^k, \tau^k, \kappa^k))^2 \leq 1/4.$$

□

Next we shall prove that the predictor step guarantees a steplength at least of order $1/\sqrt{n+1}$.

Lemma 1.4 *If* $(x^k, s^k, \tau^k, \kappa^k) \in \mathcal{N}(1/4)$ *and* $\rho = 1$, *then* $t^* \geq \frac{1}{8\sqrt{n+1}}$.

Proof. We let $t = \frac{1}{8\sqrt{n+1}}$. Since $(x^k, s^k, \tau^k, \kappa^k) \in \mathcal{N}(1/4)$, it follows that $x_i^k s_i^k \leq (1 + 1/4)\mu^k$, $i = 1, ..., n$, and $\tau^k \kappa^k \leq (1 + 1/4)\mu^k$. By (1.23) we get

$$\begin{aligned} (1 - t\rho)\delta(x(t), s(t), \tau(t), \kappa(t)) &\leq (1-t)\delta(x^k, s^k, \tau^k, \kappa^k) \\ &\quad + t/\sqrt{n+1} + \frac{t^2}{2(1-1/4)}(1 + 1/4)(n+1) \\ &\leq (1-t)\delta(x^k, s^k, \tau^k, \kappa^k) + 0.15. \end{aligned}$$

Because $t \leq 1/8$, we have $1 - t \geq 7/8$, and so

$$\delta(x(t), s(t), \tau(t), \kappa(t)) \leq 1/4 + \frac{0.15}{7/8} \leq 1/2.$$

This shows that the maximum steplength must be at least $\frac{1}{8\sqrt{n+1}}$.
□

Theorem 1.5 *Algorithm PCHSD requires to take at most $O(\sqrt{n}L)$ number of steps to reach a solution with duality gap no more than 2^{-L}, where $L \geq \log(n+1)$.*

Proof. The duality gap is reduced at the predictor step, while the corrector step will not change the duality gap. Algorithm PCHSD takes a predictor step and a corrector step alternatively. According to (1.18), the rate of duality gap reduction at a predictor step is simply $1 - t \geq 1 - \frac{1}{8\sqrt{n+1}}$. Note that the initial duality gap is $n + 1$. The theorem thus follows.
□

A careful analysis shows that the algorithm actually converges quadratically; see [26]. We remark that, despite of its simple form, Algorithm PCHSD works extremely well in practice. In particular, one can observe that it requires only a very few number (almost insensitive to the dimension n) of iterations to reach a highly accurate solution. Since the deterministic equivalence of multiple stage stochastic linear programming can be cast as a large size, however well structured, linear program. Therefore it is natural to consider applying Algorithm PCHSD to solve it. The key to success, of course, lies in the speed of solving the direction finding subproblem (1.11). This becomes the topic of next section.

1.5 Finding search directions

As we have mentioned in the previous section, it is crucial to be able to efficiently compute the search directions, based on the Newton equation (1.11), when the homogeneous self-dual embedding technique is applied to the deterministic-equivalent of multistage stochastic programming problem. To be precise, the linear system of concern is as follows, which we will

denote as $(\bar{S})$ for later reference,

$$
\left\{
\begin{array}{lll}
W_0 d_{x_0} & -h_0 d_\tau = & -\rho r_{p_0} \\
B_{nt} d_{x_{a(n),t-1}} + W_{nt} d_{x_{nt}} & -h_{nt} d_\tau = & -\rho r_{p_{nt}}, \\
 & & t = 1, ..., K,\ n \in \mathcal{F}_t \\
-W_{nt}^T d_{y_{nt}} - \sum_{m \in C(n)} B_{m,t+1}^T d_{y_{m,t+1}} - d_{s_{nt}} & +\pi_{nt} c_{nt} d_\tau = & -\rho r_{d_{nt}}, \\
 & & t = 0, ..., K-1,\ n \in \mathcal{F}_t \\
-W_{nK}^T d_{y_{nK}} - d_{s_{nK}} & +\pi_{nK} c_{nK} d_\tau = & -\rho r_{d_{nK}}, \\
 & & n \in \mathcal{F}_K \\
S_0 d_{x_0} + X_0 d_{s_0} & = & (1-\rho)\mu e \\
 & & -X_0 s_0 \\
S_{nt} d_{x_{nt}} + X_{nt} d_{s_{nt}} & = & (1-\rho)\mu e \\
 & & -X_{nt} s_{nt} \\
 & & t = 1, ..., K,\ n \in \mathcal{F}_t \\
 & \kappa d_\tau + \tau d_\kappa = & (1-\rho)\mu \\
 & & -\tau\kappa \\
\sum_{t=0}^{K} \sum_{n \in \mathcal{F}_t} \left[h_{nt}^T d_{y_{nt}} - \pi_{nt} c_{nt}^T d_{x_{nt}} \right] & -d_\kappa = & -\rho r_g
\end{array}
\right.
$$

where

$$
\begin{array}{lll}
r_{p_0} & = & W_0 x_0 - \tau h_0 \\
r_{p_{nt}} & = & B_{nt} x_{a(n),t-1} + W_{nt} x_{nt} - \tau h_{nt},\ t = 1, ..., K,\ n \in \mathcal{F}_t \\
r_{d_{nt}} & = & -W_{nt}^T y_{nt} - \sum_{m \in C(n)} B_{m,t+1}^T y_{m,t+1} - s_{nt} + \pi_{nt} c_{nt}, \\
 & & t = 0, ..., K-1,\ n \in \mathcal{F}_t \\
r_{d_{nK}} & = & -W_{nK}^T d_{y_{nK}} - s_{nK} + \pi_{nK} c_{nK},\ n \in \mathcal{F}_K \\
r_g & = & \sum_{t=0}^{K} \sum_{n \in \mathcal{F}_t} \left[h_{nt}^T y_{nt} - \pi_{nt} c_{nt}^T x_{nt} \right] - \kappa.
\end{array}
$$

Consider a given $m \in \mathcal{F}_K$. From the sixth equation in $(\bar{S})$ we obtain

$$
d_{s_{mK}} = -X_{mK}^{-1} S_{mK} d_{x_{mK}} + X_{mK}^{-1}((1-\rho)\mu e - X_{mK}^{-1} s_{mK}). \tag{1.24}
$$

Substituting (1.24) into the fourth equation in $(\bar{S})$ yields

$$
\begin{array}{l}
-W_{mK}^T d_{y_{mK}} + X_{mK}^{-1} S_{mK} d_{x_{mK}} - X_{mK}^{-1}((1-\rho)\mu e - X_{mK}^{-1} s_{mK}) \\
+\pi_{mK} c_{mK} d_\tau = (1-\rho) r_{d_{mK}},
\end{array}
$$

and therefore

$$\begin{aligned} d_{x_{mK}} &= (X_{mK}^{-1}S_{mK})^{-1}W_{mK}^T d_{y_{mK}} - \pi_{mK}(X_{mK}^{-1}S_{mK})^{-1}c_{mK}d_\tau \\ &\quad +(X_{mK}^{-1}S_{mK})^{-1}\left[-\rho r_{d_{mK}} + X_{mK}^{-1}((1-\rho)\mu e - X_{mK}^{-1}s_{mK})\right]. \end{aligned}$$

Let

$$M_{mK} := X_{mK}^{-1}S_{mK}. \tag{1.25}$$

We have

$$\begin{aligned} d_{x_{mK}} &= M_{mK}^{-1}W_{mK}^T d_{y_{mK}} - \pi_{mK}M_{mK}^{-1}c_{mK}d_\tau \\ &\quad +M_{mK}^{-1}\left[-\rho r_{d_{mK}} + X_{mK}^{-1}((1-\rho)\mu e - X_{mK}^{-1}s_{mK})\right]. \end{aligned} \tag{1.26}$$

Now consider an $n \in \mathcal{F}_{K-1}$ such that $m \in C(n)$, i.e., $n = a(m)$. Using (1.26) and the second equation in $(\bar{S})$ we get

$$\begin{aligned} & B_{mK}d_{x_{n,K-1}} + W_{mK}M_{mK}^{-1}W_{mK}^T d_{y_{mK}} - W_{mK}M_{mK}^{-1}W_{mK}^T q_{mK}d_\tau \\ = \;& W_{mK}M_{mK}^{-1}W_{mK}^T v_{mK} \end{aligned}$$

and consequently

$$d_{y_{mK}} = -(W_{mK}M_{mK}^{-1}W_{mK}^T)^{-1}B_{mK}d_{x_{n,K-1}} + q_{mK}d_\tau + v_{mK}, \tag{1.27}$$

where

$$q_{mK} := (W_{mK}M_{mK}^{-1}W_{mK}^T)^{-1}\left[h_{mK} + \pi_{mK}W_{mK}M_{mK}^{-1}c_{mK}\right] \tag{1.28}$$

and

$$\begin{aligned} v_{mK} &:= (W_{mK}M_{mK}^{-1}W_{mK}^T)^{-1}\{-\rho r_{p_{mK}} - W_{mK}M_{mK}^{-1}[-\rho r_{d_{mK}} \\ &\quad +X_{mK}^{-1}((1-\rho)\mu e - X_{mK}s_{mK})]\}. \end{aligned} \tag{1.29}$$

Now we substitute (1.27) back into the third equation in $(\bar{S})$. Further using the sixth equation in $(\bar{S})$ yields

$$\begin{aligned} & -W_{n,K-1}^T d_{y_{n,K-1}} + \\ & \quad \left[X_{n,K-1}^{-1}S_{n,K-1} + \sum_{m\in C(n)} B_{mK}^T(W_{mK}M_{mK}^{-1}W_{mK}^T)^{-1}B_{mK}\right] d_{x_{n,K-1}} \\ & \quad + \left[\pi_{n,K-1}c_{n,K-1} - \sum_{m\in C(n)} B_{mK}^T q_{mK}\right] d_\tau \\ & \quad = -\rho r_{d_{n,K-1}} + \sum_{m\in C(n)} B_{mK}^T v_{mK} + X_{n,K-1}^{-1}((1-\rho)\mu e - X_{n,K-1}s_{n,K-1}). \end{aligned}$$

Letting

$$M_{n,K-1} := X_{n,K-1}^{-1} S_{n,K-1} + \sum_{m \in C(n)} B_{mK}^T (W_{mK} M_{mK}^{-1} W_{mK}^T)^{-1} B_{mK} \tag{1.30}$$

we may rewrite the above expression as

$$\begin{aligned} d_{x_{n,K-1}} = {} & M_{n,K-1}^{-1} W_{n,K-1}^T d_{y_{n,K-1}} - \\ & M_{n,K-1}^{-1} \left[\pi_{n,K-1} c_{n,K-1} - \sum_{m \in C(n)} B_{mK}^T q_{mK} \right] d_\tau + M_{n,K-1}^{-1} \cdot \\ & \left[-\rho r_{d_{n,K-1}} + \sum_{m \in C(n)} B_{mK}^T v_{mK} + X_{n,K-1}^{-1} ((1-\rho)\mu e - X_{n,K-1} s_{n,K-1}) \right]. \end{aligned} \tag{1.31}$$

We trace one more stage back, and let k be at stage $K-2$ such that $n \in C(k)$, i.e., $k = a(n)$. Similar as before, we substitute (1.31) into the second equation in $(\bar{S})$, which is

$$B_{n,K-1} d_{x_{k,K-2}} + W_{n,K-1} d_{x_{n,K-1}} - h_{n,K-1} d_\tau = -\rho r_{p_{n,K-1}},$$

obtaining

$$d_{y_{n,K-1}} = -(W_{n,K-1} M_{n,K-1}^{-1} W_{n,K-1}^T)^{-1} B_{n,K-1} d_{x_{k,K-2}} + q_{n,K-1} d_\tau + v_{n,K-1}, \tag{1.32}$$

where

$$\begin{aligned} q_{n,K-1} := {} & (W_{n,K-1} M_{n,K-1}^{-1} W_{n,K-1}^T)^{-1} \cdot \{ h_{n,K-1} + \\ & W_{n,K-1} M_{n,K-1}^{-1} [\pi_{n,K-1} c_{n,K-1} - \sum_{m \in C(n)} B_{mK}^T q_{mK}] \} \end{aligned} \tag{1.33}$$

and

$$\begin{aligned} v_{n,K-1} := {} & -(W_{n,K-1} M_{n,K-1}^{-1} W_{n,K-1}^T)^{-1} \cdot \{ \rho r_{p_{n,K-1}} \\ & + W_{n,K-1} M_{n,K-1}^{-1} [-\rho r_{d_{n,K-1}} \\ & + X_{n,K-1}^{-1} ((1-\rho)\mu e - X_{n,K-1} s_{n,K-1}) + \sum_{m \in C(n)} B_{mK}^T v_{mK}] \}. \end{aligned} \tag{1.34}$$

Once again, we substitute (1.32) back into the third equation in $(\bar{S})$ and use the sixth equation at the same time. This yields an equation from which

we can solve out $d_{x_{k,K-2}}$ as follows

$$\begin{aligned} d_{x_{k,K-2}} &= M_{k,K-2}^{-1} W_{k,K-2}^T d_{y_{k,K-2}} - \\ & M_{k,K-2}^{-1}[\pi_{k,K-2} c_{k,K-2} - \sum_{n\in C(k)} B_{k,K-1}^T q_{k,K-1}] d_\tau + \\ & M_{k,K-2}^{-1}[-\rho r_{d_{k,K-2}} + \sum_{n\in C(k)} B_{n,K-1}^T v_{n,K-1} + \\ & X_{k,K-2}^{-1}((1-\rho)\mu e - X_{k,K-2} s_{k,K-2})]. \end{aligned} \tag{1.35}$$

Repeating this procedure, we get the following recursive formula

$$d_{y_{mt}} = -(W_{mt} M_{mt}^{-1} W_{mt}^T)^{-1} B_{mt} d_{x_{n,t-1}} + q_{mt} d_\tau + v_{mt}, \tag{1.36}$$

where $n = a(m)$, and

$$\begin{aligned} d_{x_{mt}} &= M_{mt}^{-1} W_{mt}^T d_{y_{mt}} - M_{mt}^{-1}[\pi_{mt} c_{mt} - \sum_{k\in C(m)} B_{k,t+1}^T q_{k,t+1}] d_\tau + M_{mt}^{-1} \cdot \\ & [-\rho r_{d_{mt}} + \sum_{k\in C(m)} B_{k,t+1}^T v_{k,t+1} + X_{mt}^{-1}((1-\rho)\mu e - X_{mt} s_{mt})], \end{aligned} \tag{1.37}$$

for $t = K, K-1, ..., 1$ and $m \in \mathcal{F}_t$. As a notational convention, we assume that $C(m) = \emptyset$ for $m \in \mathcal{F}_K$. The matrices M_{mt} and vectors q_{mt} and v_{mt} are generated according to the following recursive formula

$$(R_1) \begin{cases} M_{n,t-1} &= X_{n,t-1}^{-1} S_{n,t-1} + \sum_{m\in C(n)} B_{mt}^T (W_{mt} M_{mt}^{-1} W_{mt}^T)^{-1} B_{mt} \\ q_{n,t-1} &= (W_{n,t-1} M_{n,t-1}^{-1} W_{n,t-1}^T)^{-1} \cdot \\ & \{h_{n,t-1} + W_{n,t-1} M_{n,t-1}^{-1}[\pi_{n,t-1} c_{n,t-1} - \sum_{m\in C(n)} B_{mt}^T q_{mt}]\} \\ v_{n,t-1} &= -(W_{n,t-1} M_{n,t-1}^{-1} W_{n,t-1}^T)^{-1} \cdot \\ & \{\rho r_{p_{n,t-1}} + W_{n,t-1} M_{n,t-1}^{-1}[-\rho r_{d_{n,t-1}} + \\ & X_{n,t-1}^{-1}((1-\rho)\mu e - X_{n,t-1} s_{n,t-1}) + \sum_{m\in C(n)} B_{mt}^T v_{mt}]\}. \end{cases}$$

with $m \in C(n)$ and $m \in \mathcal{F}_t$; see (1.30), (1.33) and (1.34). The terminal values are given by the following formula

$$(T) \begin{cases} M_{mK} &= X_{mK}^{-1} S_{mK} \\ q_{mK} &= (W_{mK} M_{mK}^{-1} W_{mK}^T)^{-1} \left[h_{mK} + \pi_{mK} W_{mK} M_{mK}^{-1} c_{mK}\right] \\ v_{mK} &= (W_{mK} M_{mK}^{-1} W_{mK}^T)^{-1} \cdot \{-\rho r_{p_{mK}} - W_{mK} M_{mK}^{-1}[-\rho r_{d_{mK}} + \\ & X_{mK}^{-1}((1-\rho)\mu e - X_{mK} s_{mK})]\} \end{cases}$$

where $m \in \mathcal{F}_K$; see (1.25), (1.28) and (1.29).

To further simplify the notation let us rewrite (1.37) as

$$d_{x_{mt}} = M_{mt}^{-1} W_{mt}^T d_{y_{mt}} - p_{mt} d_\tau + u_{mt} \tag{1.38}$$

with

$$\begin{aligned} p_{mt} &= M_{mt}^{-1}[\pi_{mt} c_{mt} - \sum_{k \in C(m)} B_{k,t+1}^T q_{k,t+1}] && (1.39) \\ u_{mt} &= M_{mt}^{-1}[-\rho r_{d_{mt}} + \sum_{k \in C(m)} B_{k,t+1}^T v_{k,t+1} + \\ &\quad X_{mt}^{-1}((1-\rho)\mu e - X_{mt} s_{mt})]. && (1.40) \end{aligned}$$

In particular we have

$$d_{x_0} = M_0^{-1} W_0^T d_{y_0} - p_0 d_\tau + u_0. \tag{1.41}$$

Substituting this into the first equation in $(\bar{S})$ we have

$$W_0(M_0^{-1} W_0^T d_{y_{mt}} - p_0 d_\tau + u_0) - h_0 d_\tau = -\rho r_{p_0}.$$

Equivalently,

$$d_{y_0} = (W_0 M_0^{-1} W_0^T)^{-1}(W_0 p_0 + h_0) d_\tau + (W_0 M_0^{-1} W_0^T)^{-1}(-\rho r_{p_0} - W_0 u_0).$$

We denote

$$\begin{aligned} \alpha_0 &= (W_0 M_0^{-1} W_0^T)^{-1}(W_0 p_0 + h_0) && (1.42) \\ \beta_0 &= (W_0 M_0^{-1} W_0^T)^{-1}(-\rho r_{p_0} - W_0 u_0) && (1.43) \end{aligned}$$

and so

$$d_{y_0} = \alpha_0 d_\tau + \beta_0. \tag{1.44}$$

Substituting (1.44) into (1.41) we get

$$\begin{aligned} d_{x_0} &= M_0^{-1} W_0^T d_{y_0} - p_0 d_\tau + u_0 \\ &= M_0^{-1} W_0^T (\alpha_0 d_\tau + \beta_0) - p_0 d_\tau + u_0 \\ &=: \psi_0 d_\tau + \phi_0 && (1.45) \end{aligned}$$

with

$$\psi_0 = M_0^{-1} W_0^T \alpha_0 - p_0 \tag{1.46}$$

and

$$\phi_0 = M_0^{-1} W_0^T \beta_0 + u_0. \tag{1.47}$$

Now we wish to find a general expression for $d_{x_{mt}}$ and $d_{y_{mt}}$ in terms of d_τ. Let us write them as

$$\begin{aligned} d_{y_{mt}} &= \alpha_{mt} d_\tau + \beta_{mt} && (1.48) \\ d_{x_{mt}} &= \psi_{mt} d_\tau + \phi_{mt} && (1.49) \end{aligned}$$

where $m \in \mathcal{F}_t$ and $t = 0, 1, ..., K$.

A forward recursive formula for α_{mt}, β_{mt}, ψ_{mt} and ϕ_{mt} can be found, based on (1.38) and (1.36), as follows

$$(R_2)\begin{cases} \alpha_{mt} &= -(W_{mt}M_{mt}^{-1}W_{mt}^T)^{-1}B_{mt}\psi_{n,t-1} + q_{mt} \\ \beta_{mt} &= -(W_{mt}M_{mt}^{-1}W_{mt}^T)^{-1}B_{mt}\phi_{n,t-1} + v_{mt} \\ \psi_{mt} &= M_{mt}^{-1}W_{mt}^T\alpha_{mt} - p_{mt} \\ \phi_{mt} &= M_{mt}^{-1}W_{mt}^T\beta_{mt} + u_{mt} \end{cases}$$

where $m \in \mathcal{F}_t$, $n \in \mathcal{F}_{t-1}$ and $n = a(m)$. Putting together (1.42), (1.43), (1.46) and (1.47), we have the following initial values for the recursion

$$(I)\begin{cases} \alpha_0 &= (W_0M_0^{-1}W_0^T)^{-1}(W_0p_0 + h_0) \\ \beta_0 &= (W_0M_0^{-1}W_0^T)^{-1}(-\rho r_{p_0} - W_0u_0) \\ \psi_0 &= M_0^{-1}W_0^T\alpha_0 - p_0 \\ \phi_0 &= M_0^{-1}W_0^T\beta_0 + u_0. \end{cases}$$

Now, we eliminate d_κ using the seventh and the eighth equations in $(\bar{S})$. Then we substitute $d_{x_{nt}}$ and $d_{y_{nt}}$, according to (1.48) and (1.49), into the resulting equation. This yields

$$\sum_{t=0}^{K}\sum_{n\in\mathcal{F}_t}[h_{nt}^T(\alpha_{nt}d_\tau + \beta_{nt}) - \pi_{nt}c_{nt}^T(\psi_{nt}d_\tau + \phi_{nt})]$$
$$-(-\tau^{-1}\kappa d_\tau + \tau^{-1}((1-\rho)\mu - \tau\kappa)) = -\rho r_g$$

and so

$$d_\tau = \frac{-\rho r_g + \tau^{-1}((1-\rho)\mu - \tau\kappa) - \sum_{t=0}^{K}\sum_{n\in\mathcal{F}_t}\left[h_{nt}^T\beta_{nt} - \pi_{nt}c_{nt}^T\phi_{nt}\right]}{\sum_{t=0}^{K}\sum_{n\in\mathcal{F}_t}\left[h_{nt}^T\alpha_{nt} - \pi_{nt}c_{nt}^T\psi_{nt}\right] + \tau^{-1}\kappa}. \quad (1.50)$$

All other search directions can be solved using (1.50). In particular, from the seventh equation in $(\bar{S})$ we have

$$d_\kappa = -\tau^{-1}\kappa d_\tau + \tau^{-1}((1-\rho)\mu - \tau\kappa) \quad (1.51)$$

and from (1.48) and (1.49) we get the values for $d_{y_{mt}}$ and $d_{x_{mt}}$. Furthermore, the value for $d_{s_{mt}}$ can be obtained from the sixth equation in $(\bar{S})$, i.e.

$$d_{s_{mt}} = -X_{mt}^{-1}S_{mt}d_{x_{mt}} + X_{mt}^{-1}\left((1-\rho)\mu e - X_{mt}s_{mt}\right). \quad (1.52)$$

To summarize, the search directions implied by the Newton equation $(\bar{S})$ can be solved in the following way. First, we generate the working matrices and vectors M_{mt}, q_{mt}, v_{mt}, p_{mt} and u_{mt} according to the recursive scheme (R_1) with the terminal values determined by (T). This works in a backward manner. Then, we generate another set of working vectors α_{mt}, β_{mt}, ψ_{mt} and ϕ_{mt} using the recursive scheme (R_2) with the initial values given by (I). Finally, the search directions are computed by the formulae (1.50), (1.51), (1.48), (1.49) and (1.52). Clearly, the time and space complexity of such a decomposition scheme is linear with respect to the total number of scenarios.

The crux of this computational scheme is to generate M_{mt}, q_{mt}, v_{mt}, p_{mt} and u_{mt}. This, however, can be accelerated if we have a parallel computational environment, because the computation of the quantities $(M_{n,t-1}, q_{n,t-1}, v_{n,t-1}, p_{n,t-1}, u_{n,t-1})$ from $(M_{mt}, q_{mt}, v_{mt}, p_{mt}, u_{mt})$ can be done independently. The same can be said about the computation of α_{mt}, β_{mt}, ψ_{mt} and ϕ_{mt}. In the computation, one actually only needs to store M_{mt}, q_{mt}, v_{mt}. All the other quantities can easily be derived from this information.

Summarizing, we have the following conclusion.

Theorem 1.6 *Let the total number of scenarios in $(MDSLP)$ be $N := \sum_{t=1}^{K} |\mathcal{F}_t|$. Suppose that that matrices W_{nt} and B_{nt} in $(MDSLP)$ are of size s. Then, solving the Newton equation $(\bar{S})$ requires no more than $O(s^3 N)$ elementary operations according to the decomposition method described above.*

1.6 Model diagnosis

Any optimization model is only an approximation of the reality. This means in particular that if a model returns with an undesirable solution, or declares that no solution exists at all, then it might be: 1) the model is not appropriate; or 2) the model is good, but some 'soft constraints' must be relaxed in order to produce a sensible solution. In either case, it is of crucial importance to understand why the model does not have an optimal solution. For stochastic programming, testing the model is especially relevant, because very often the model is only a rough approximation of the real situation in the presence of uncertainty.

A major advantage of using a self-dual embedded model is that if the primal or the dual problem is infeasible, then, instead of getting insignificant output, such as "Infeasible Model", we always get a Farkas type certificate. As we will see in this section, this information is useful in order to analyze the cause of the infeasibility. To see this, consider a linear programming

problem given as

$$\begin{array}{ll} \max & b^T y \\ \text{s.t.} & A^T y \leq c. \end{array}$$

Suppose that the problem is infeasible. Then by the Farkas lemma and a well known result of Goldman and Tucker [14], there exists $x^* \geq 0$ with maximal cardinality of positive components, such that $Ax^* = 0$ and $c^T x^* < 0$. Let the positive support of x^* be I, i.e., $I = \{i \mid x_i^* > 0\}$. Now we claim that if we remove the set of constraints indexed by I, then the remaining problem must be feasible. The proof is as follows. Let the complement set of I be J. Further let us assume, for the sake of obtaining a contradiction, that the system $A_J^T y \leq c_J$ is still infeasible. Then, by applying the Farkas lemma once more, we conclude that there is $u \geq 0$ such that $A_J u = 0$ and $c_J^T u < 0$. As a by-product, we also know $u \neq 0$. Let

$$\bar{x}_i = \begin{cases} u_i, & \text{if } i \in J \\ 0, & \text{if } i \notin J \end{cases}$$

This implies that $x^* + \bar{x} \geq 0$, $A(x^* + \bar{x}) = 0$, and $c^T(x^* + \bar{x}) < 0$. Moreover, $x^* + \bar{x}$ has a positive support set which is larger in cardinality than that of x^*. This yields a contradiction. As a result, the claimed fact is proven.

One implication of the above result is the following. Suppose that we have a Farkas type infeasibility certificate with maximal support. Then, its positive support part corresponds to the set of constraints that are causing the infeasibility. In other words, this is the set of scenarios that requires scrutiny. Numerically, if all the data are properly scaled, then we may interpret a higher-valued dual multiplier (Farkas type certificate) to correspond to the scenario that is likely to have more responsibilities for causing the infeasibility. Certainly, due to the primal-dual symmetricity, exactly the same thing can be said about a linear program in the primal form

$$\begin{array}{ll} \min & c^T x \\ \text{s.t.} & Ax = b \\ & x \geq 0. \end{array}$$

In this case, if the problem is infeasible then the corresponding Farkas type certificate is $-A^T y^* \geq 0$ such that $b^T y^* > 0$.

Let us consider one example to illustrate this point. Suppose that there is a risky asset, say a common stock, whose value may go up, or down, or remain unchanged in the next period. To be specific, assume that with 40% chance it will go up 10%, with 30% chance it will go down 4%, and with 30% chance it will be unchanged. Let us concentrate on a two-period situation. A riskless investment will always yield a 2% return rate in the same period, no matter what happens. Suppose we have a notional 1$ to invest. Now we wish to find an investment plan, allocating our wealth and possibly updating our portfolio at the end of period 1, in such a way that our wealth will never go below value g, and at the same time, the expected

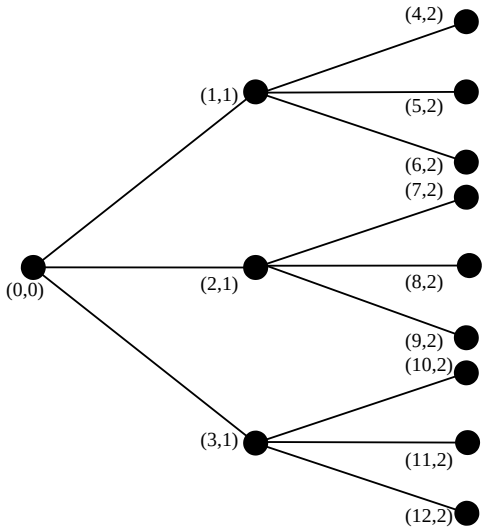

FIGURE 1.2. Scenario Tree: Example

return rate is maximized at the investment horizon, i.e., at the end of period 2.

Clearly, in this case the scenario tree involves 13 nodes (see Figure 1.2). We denote the current stage to be (0,0), and denote the set of scenarios at the first stage (up till down) be (1,1), (2,1) and (3,1), each followed by three children. In particular, (1,1) is followed by (4,2), (5,2) and (6,2), (2,1) is followed by (7,2), (8,2) and (9,2), and (3,1) is followed by (10,2), (11,2) and (12,2); see Figure 1.2.

Let the uncertain return rate of the stock (over one period) be ω. In our model, $\omega_u = 0.1$ (up), $\omega_e = 0$ (equal), and $\omega_d = -0.04$ (down). The riskless rate over one period is $r = 0.02$. Let x^s_{nt} be the amount invested in the stock, and x^b_{nt} be the amount invested in the riskless asset, at stage t if scenario n unfolds. Then, the constraints in the model are:

$$\begin{aligned}
&x^s_0 + x^b_0 = 1,\\
&(1+\omega)x^s_{a(n),0} + (1+r)x^b_{a(n),0} = x^s_{n1} + x^b_{n1}, \text{ for } n = 1,2,3\\
&x^s_{n2} + x^b_{n2} \geq g, \text{ for } n = 4,5,...,12.
\end{aligned}$$

We further assume that no short selling is allowed. In our notation, this is equivalent to the following constraint matrices

$$\begin{aligned}
&W_0 = [1,1],\ h_0 = 1,\\
&B_{n1} = -[1+\omega, 1+r],\ W_{n1} = [1,1],\ h_{n1} = 0,\\
&B_{n2} = \begin{bmatrix} -(1+\omega), & -(1+r) \\ 0, & 0 \end{bmatrix},\ W_{n2} = \begin{bmatrix} 1, & 1, & 0 \\ 1, & 1, & -1 \end{bmatrix},\ h_{n2} = \begin{bmatrix} 0 \\ g \end{bmatrix}.
\end{aligned}$$

Note here that we have introduced 9 slack variables for the guaranteed return constraints: $x^s_{n2} + x^b_{n2} \geq g$, $n = 4,5,...,12$.

Now we run our algorithm on this model. When the investor invests his wealth in the riskless asset over the two-periods in our model, his final wealth will be 1.0404 ($1 * 1.02 * 1.02$). A higher guaranteed return is not possible due to the uncertainty of the risky asset. Hence, for $g \leq 1.0404$, the model is solvable. For instance, if $g = 1$, then $x_0^s = 0.6601$ and $x_0^b = 0.3399$, with an expected return rate at the horizon being 5.03%. In comparison, the riskless return rate at the horizon is 4.04%. As soon as $g > 1.0404$, the model becomes demanding a guaranteed return rate higher than the riskless rate. This is certainly impossible in the world where no arbitrage opportunity exists, and the model should return as being infeasible. Indeed, this turns out to be the case. Set for instance $g = 1.05$. The model is infeasible. The point here is, our algorithm also provides a dual certificate

$$
\begin{aligned}
&-A^T y^* = \\
&[0.3548, 0.0068, 0.1983, 0.0138, 0.2513, 0.0120, 0.2972, 0.0077, \\
&0.0248, 0.0248, 1.7230, 0.0243, 0.0243, 2.8122, 0.0242, 0.0242, 4.4356, \\
&0.0240, 0.0240, 2.1821, 0.0238, 0.0238, 3.7830, 0.0239, 0.0239, 5.6368, \\
&0.0239, 0.0239, 2.5476, 0.0237, 0.0237, 4.5298, 0.0237, 0.0237, 6.7116].
\end{aligned}
$$

If we examine the values in the certificate carefully, then we will find that there are 9 numbers that are significantly higher than 1, and they correspond to the constraints:

$$x_{n2}^s + x_{n2}^b \geq g, \text{ for } n = 4, 5, ..., 12.$$

As we know, these constraints are indeed the ones that are causing the infeasibility.

Similar analysis applies to the situation when the problem has an unbounded optimum value. In portfolio applications, this phenomenon is known as the existence of an arbitrage opportunity. In particular, the primal Farkas type certificate plays the role of displaying one such arbitrage opportunity. In this case, the meaning of the solution produced by our algorithm is quite obvious.

1.7 Notes

There is a large body of literature on multistage stochastic programming. Below we shall point out a few representative references related to our discussions in this chapter for the benefit of the interested reader.

Along the direction of (parallel) implementations of the Benders type decomposition, one may consult [7, 13] and [15].

For the augmented Lagrangian approach, one may further read [4, 20] and [21].

As for the decomposition type approach based on the interior point methods, one is referred to [8, 11, 18, 28] for serial implementations, and [12, 16, 25] for parallel implementations. Finally, for the cutting plane type approach combined with the interior point methodology, see [2] and [3].

Acknowledgments: This project has been supported by the Foundation 'Vereniging Trustfonds Erasmus Universiteit Rotterdam' in The Netherlands, and RGC Earmarked Grant CUHK 4233/01E.

1.8 References

[1] E.D. Andersen, and K.D. Andersen, *The MOSEK interior point optimizer for linear programming: an implementation of the homogeneous algorithm*, High Performance Optimization Techniques, pp. 197-232, eds. J.B.G. Frenk, K. Roos, T. Terlaky and S. Zhang, Kluwer Academic Publishers, 1999.

[2] K.A. Ariyawansa, and P.L. Jiang, Polynomial cutting plane algorithms for two-stage stochastic linear programs based on ellipsoids, volumetric centers and analytic centers, *Working Paper*, Department of Pure and Applied Mathematics, Washington State University, Pullman, USA, 1996.

[3] O. Bahn, O. du Merle, J.-L. Goffin, and J.P. Vial, *A cutting plane method from analytic centers for stochastic programming*, Mathematical Programming 69, 45-73, 1995.

[4] A.J. Berger, J.M. Mulvey and A. Ruszczynski, *An extension of the DQA algorithm to convex stochastic programs*, SIAM Journal on Optimization 4, 735-753, 1994.

[5] A. Berkelaar, C. Dert, C. Oldenkamp, and S. Zhang, A Primal-Dual Decomposition-Based Interior Point Approach to Two-Stage Stochastic Linear Programming, Report 9918/A, Econometric Institute, Erasmus University Rotterdam, 1999. (To appear in *Operations Research*).

[6] A. Berkelaar, R. Kouwenberg, and S. Zhang, A Primal-Dual Decomposition Algorithm for Multistage Stochastic Convex Programming, Technical Report SEEM2000-07, Department of Systems Engineering & Engineering Management, The Chinese University of Hong Kong, 2000. (Submitted for publication).

[7] J.R. Birge, C.J. Donohue, D.F. Holmes, and O.G. Svintsitski, *A parallel implementation of the nested decomposition algorithm for multistage stochastic linear programs*, Mathematical Programming 75, 327-352, 1996.

[8] J.R. Birge, and D.F. Holmes, *Efficient solution of two-stage stochastic linear programs using interior point methods*, Computational Optimization and Applications 1, 245-276, 1992.

[9] J.R. Birge, and F. Louveaux, *Introduction to Stochastic Programming*, Springer, New York, 1997.

[10] J.R. Birge, and L. Qi, *Computing block-angular Karmarkar projections with applications to stochastic programming*, Management Science 34, 1472-1479, 1988.

[11] I.C. Choi, and D. Goldfarb, *Exploiting special structure in a primal-dual path-following algorithm*, Mathematical Programming 58, 33-52, 1993.

[12] J. Czyzyk, R. Fourer and S. Mehrotra, *Using a massively parallel processor to solve large sparse linear programs by an interior-point method*, SIAM Journal on Scientific Computing 19, 553-565, 1998.

[13] E. Fragnière, J. Gondzio, and J.-P. Vial, Building and Solving Large-scale Stochastic Programs on an Affordable Distributed Computing System, forthcoming in Annals of Operations Research, 1998.

[14] A.J. Goldman and A.W. Tucker, Polyhedral convex cones, in H.W. Kuhn and A.W. Tucker eds., *Linear Inequalities and Related Systems*, Princeton University Press, New Jersey, 19-40, 1956.

[15] J. Gondzio, and R. Kouwenberg, High Performance Computing for Asset Liability Management, forthcoming in Operations Research, 2000.

[16] E.R. Jessup, D. Yang, and S.A. Zenios, *Parallel factorization of structured matrices arising in stochastic programming*, SIAM Journal on Optimization 4, 833-846, 1994.

[17] P. Kall, and S.W. Wallace, Stochastic Programming, John Wiley and Sons, Chichester, 1994.

[18] I.J. Lustig, J.M. Mulvey and T.J. Carpenter, *The formulation of stochastic programs for interior point methods*, Operations Research 39, 757-770, 1991.

[19] S. Mizuno, M.J. Todd, and Y. Ye, *On adaptive-step primal-dual interior-point algorithms for linear programming*, Mathematics of Operations Research 18, 964-981, 1993.

[20] J.M. Mulvey, and A. Ruszczynski, *A new scenario decomposition method for large-scale stochastic optimization*, Operations Research 43, 477-490, 1995.

[21] R.T. Rockafellar, and R.J.-B. Wets, *Scenarios and policy aggregation in optimization under uncertainty*, Mathematics of Operations Research 16, 119-147, 1991.

[22] C. Roos, T. Terlaky, and J.-Ph. Vial, *Theory and Algorithms for Linear Optimization*, John Wiley & Sons, chapter 19, 1997.

[23] R. Van Slyke, and R.J.-B. Wets, *L-shaped linear programs with applications to optimal control and stochastic linear programs*, SIAM Journal on Applied Mathematics 17, 638-663, 1969.

[24] X. Xu, P.F. Hung, and Y. Ye, *A simplified homogeneous self-dual linear programming algorithm and its implementation*, Annals of Operations Research 62, 151-171, 1996.

[25] D. Yang, and S.A. Zenios, *A Scalable parallel interior point algorithm for stochastic linear programming and robust optimization*, Computational Optimization and Applications 7, 143-158, 1997.

[26] Y. Ye, and K. Anstreicher, *On quadratic and* $O(\sqrt{n}L)$ *convergence of a predictor-corrector algorithm for LCP*, Mathematical Programming 62, 537-551, (1993).

[27] Y. Ye, M.J. Todd, and S. Mizuno, *An* $\mathcal{O}(\sqrt{n}L)$*-iteration homogeneous and self-dual linear programming algorithm*, Mathematics of Operations Research 19, 53-67, 1994.

[28] G. Zhao, *A log-barrier method with Benders decomposition for solving two-stage stochastic linear programs*, Mathematical Programming 90, 507-536, 2001.

An interior-point and decomposition approach to multiple stage stochastic programming. Publication. Publication.Â Zhang, S. (2002). An interior-point and decomposition approach to multiple stage stochastic programming (No. EI 2002-35). Econometric Institute Research Papers. In this paper we discuss a particular decomposition approach to solve multi-stage stochastic linear programming, based on a high performance interior point method. Numerical experiences suggest that the method requires only a few number of iteration steps to reach a high precision solution, almost regardless the size of the problem. Attention is thus paid to solving a large size Newton equation needed for obtaining a search direction at each iteration by decomposition. It turns out that it is possible to reduce the computational e#ort to a minimum level, i.e., the number of elementary operatio