A Gentle? Introduction to Stochastic Programming

Jeff Linderoth

Department of Industrial and Systems Engineering
Lehigh University

COR@L Distinguished Lecture Series :-)
Lehigh University
September 9, 2004

Outline

- What is Stochastic Programming (SP)?
  - There are lots of stochastic programming problems
  - The Canonical Problem

Outline

- What is Stochastic Programming (SP)?
  - There are lots of stochastic programming problems
  - The Canonical Problem
- Solving Stochastic Programs
  - Deterministic equivalents
  - Sampling
  - A decomposition algorithm

Outline

- What is Stochastic Programming (SP)?
  - There are lots of stochastic programming problems
  - The Canonical Problem
- Solving Stochastic Programs
  - Deterministic equivalents
  - Sampling
  - A decomposition algorithm
- Stochastic Integer Programming
  - It's Very Hard

Why Care about Stochastic Programming?

What we anticipate seldom occurs; what we least expected generally happens.

Benjamin Disraeli (1804 - 1881)

Why Care about Stochastic Programming?

> *What we anticipate seldom occurs; what we least expected generally happens.*
>
> Benjamin Disraeli (1804 - 1881)

- Think of Stochastic Programming (SP) as Mathematical Programming (MP) with random parameters

Why Care about Stochastic Programming?

> *What we anticipate seldom occurs; what we least expected generally happens.*
>
> Benjamin Disraeli (1804 - 1881)

- Think of Stochastic Programming (SP) as Mathematical Programming (MP) with random parameters
- This is useful, since we often really don't know the data
  - Customer demands
  - Market actions
  - Insert your own favorite uncertainty here...

Why Care about Stochastic Programming?

What we anticipate seldom occurs; what we least expected generally happens.

Benjamin Disraeli (1804 - 1881)

- Think of Stochastic Programming (SP) as Mathematical Programming (MP) with random parameters
- This is useful, since we often really don't know the data
  - Customer demands
  - Market actions
  - Insert your own favorite uncertainty here...
- SP assumes a probability distribution for the random variable (ω) is known or can be approximated with reasonable accuracy

Mathematical Formulations

A Mathematical Program

$$\min_{x \in X} f(x) \qquad \text{(MP)}$$

$$X \stackrel{\text{def}}{=} \{x \in X_0 \mid g_i(x) \leq 0 \quad \forall i \in M\}$$

Mathematical Formulations

A Mathematical Program

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

$$X \stackrel{\text{def}}{=} \{x \in X_0 \mid g_i(x) \leq 0 \quad \forall i \in M\}$$

A Stochastic Program

$$\min_{x \in X(\omega)} F(x, \omega) \tag{SP}$$

But I Haven't Told you Anything!

How should we deal with the randomness

But I Haven't Told you Anything!

$\boxed{\text{How should we deal with the randomness}}$

- $\min F(x, \hat{\omega})$ Point Estimate

But I Haven't Told you Anything!

How should we deal with the randomness

- $\min F(x, \hat{\omega})$ — Point Estimate
- $\min \mathbb{E}_\omega F(x, \omega)$ — Risk Neutral

But I Haven't Told you Anything!

How should we deal with the randomness

- $\min F(x, \hat{\omega})$ Point Estimate
- $\min \mathbb{E}_\omega F(x, \omega)$ Risk Neutral
- $\min \mathbb{E}_\omega F(x, \omega) - \lambda\rho(F(x, \omega))$ Risk Measures

But I Haven't Told you Anything!

How should we deal with the randomness

- $\min F(x, \hat{\omega})$ — Point Estimate
- $\min \mathbb{E}_\omega F(x, \omega)$ — Risk Neutral
- $\min \mathbb{E}_\omega F(x, \omega) - \lambda \rho(F(x, \omega))$ — Risk Measures
  - $\rho(F(x, \omega)) = \mathsf{Var} F(x, \omega)$ — Markowitz

But I Haven't Told you Anything!

How should we deal with the randomness

- $\min F(x, \hat{\omega})$ — Point Estimate
- $\min \mathbb{E}_{\omega} F(x, \omega)$ — Risk Neutral
- $\min \mathbb{E}_{\omega} F(x, \omega) - \lambda \rho(F(x, \omega))$ — Risk Measures
  - $\rho(F(x, \omega)) = \mathsf{Var} F(x, \omega)$ — Markowitz
  - $\rho(F(x, \omega)) = \mathbb{E}\left[(F(x, \omega) - \mathbb{E}F(x, \omega))_+\right]$ — Semideviation

Coping with Randomness—The Constraints

- $X(\omega) = \{x \in X_0 \mid G_i(x, \hat{\omega}) \leq 0 \quad \forall i \in M\}$
  - Point Estimate

Coping with Randomness—The Constraints

- $X(\omega) = \{x \in X_0 \mid G_i(x, \hat{\omega}) \leq 0 \quad \forall i \in M\}$
  - Point Estimate
- $X(\omega) = \{x \in X_0 \mid G_i(x, \omega) \leq 0 \quad \forall i \in M, \forall \omega \in \Omega\}$
  - For all realizations

Coping with Randomness—The Constraints

- $X(\omega) = \{x \in X_0 \mid G_i(x, \hat{\omega}) \leq 0 \quad \forall i \in M\}$
  - Point Estimate
- $X(\omega) = \{x \in X_0 \mid G_i(x, \omega) \leq 0 \quad \forall i \in M, \forall \omega \in \Omega\}$
  - For all realizations
- $X(\omega) = \{x \in X_0 \mid G_i(x, \omega) \leq 0 \quad \forall \omega \in \mathcal{U} \subset \Omega\}$
  - Robust Optimization

Coping with Randomness—The Constraints

- $X(\omega) = \{x \in X_0 \mid G_i(x, \hat{\omega}) \leq 0 \quad \forall i \in M\}$
  - Point Estimate
- $X(\omega) = \{x \in X_0 \mid G_i(x, \omega) \leq 0 \quad \forall i \in M, \forall \omega \in \Omega\}$
  - For all realizations
- $X(\omega) = \{x \in X_0 \mid G_i(x, \omega) \leq 0 \quad \forall \omega \in \mathcal{U} \subset \Omega\}$
  - Robust Optimization
- $X(\omega) = \{x \in X_0 \mid \mathbb{P}\{G_i(x, \omega) \leq 0 \ \forall i \in M\} \geq 1 - \alpha\}$
  - Joint Chance Constraints

Coping with Randomness—The Constraints

- $X(\omega) = \{x \in X_0 \mid G_i(x, \hat{\omega}) \leq 0 \quad \forall i \in M\}$
  - Point Estimate
- $X(\omega) = \{x \in X_0 \mid G_i(x, \omega) \leq 0 \quad \forall i \in M, \forall \omega \in \Omega\}$
  - For all realizations
- $X(\omega) = \{x \in X_0 \mid G_i(x, \omega) \leq 0 \quad \forall \omega \in \mathcal{U} \subset \Omega\}$
  - Robust Optimization
- $X(\omega) = \{x \in X_0 \mid \mathbb{P}\{G_i(x, \omega) \leq 0 \ \forall i \in M\} \geq 1 - \alpha\}$
  - Joint Chance Constraints
- $X(\omega) = \{x \in X_0 \mid \mathbb{P}\{G_i(x, \omega) \leq 0\} \geq 1 - \alpha_i \ \forall i \in M\}$
  - Individual Chance Constraints

Things People Want

Continuous Distributions

Things People Want

(Conditional) Value at Risk

Continuous Distributions

Things People Want

Network Problems

(Conditional) Value at Risk

Continuous Distributions

Things People Want

Network Problems

(Conditional) Value at Risk

Scenario Trees

Continuous Distributions

Things People Want

Network Problems

(Conditional) Value at Risk

Scenario Trees

Continuous Distributions

Stochastic Dynamic Programming

Things People Want

Network Problems

(Conditional) Value at Risk

Scenario Trees

Continuous Distributions

Stochastic Dynamic Programming

Robust Optimization

Things People Want

Network Problems

(Conditional) Value at Risk

Scenario Trees

Continuous Distributions

Stochastic Dynamic Programming

(Joint) Chance Constraints

Robust Optimization

Things People Want

Network Problems

(Conditional) Value at Risk

Scenario Trees

Continuous Distributions

Stochastic Dynamic Programming

(Joint) Chance Constraints

Network Problems

Robust Optimization

Things People Want

Network Problems

(Conditional) Value at Risk

Scenario Trees

Continuous Distributions

Stochastic Dynamic Programming

(Joint) Chance Constraints

Robust Optimization

Network Problems

Stochastic Control

Things People Want

Network Problems

(Conditional) Value at Risk

Scenario Trees

Continuous Distributions

Stochastic Dynamic Programming

(Joint) Chance Constraints

Network Problems

Robust Optimization

Stochastic Control

Joint Distributions

Things People Want

- Network Problems
- Nonlinear problems
- (Conditional) Value at Risk
- Scenario Trees
- Continuous Distributions
- Stochastic Dynamic Programming
- (Joint) Chance Constraints
- Network Problems
- Robust Optimization
- Stochastic Control
- Joint Distributions

Things People Want

- Network Problems
- Nonlinear Problems
- Integer problems
- (Conditional) Value at Risk
- Scenario Trees
- Continuous Distributions
- Stochastic Dynamic Programming
- (Joint) Chance Constraints
- Robust Optimization
- Network Problems
- Stochastic Control
- Joint Distributions

Things People Want

Network Problems

Nonlinear Problems

Integer problems

(Conditional) Value at Risk

Scenario Trees

Free Beer

Continuous Distributions

Stochastic Dynamic Programming

(Joint) Chance Constraints

Network Problems

Robust Optimization

Stochastic Control

Joint Distributions

Supporting Stochastic Programs

- I point out all these different flavors of SP to highlight what I think has been one of the hinderances of acceptance of stochastic programming in the broader community

Supporting Stochastic Programs

- I point out all these different flavors of SP to highlight what I think has been one of the hinderances of acceptance of stochastic programming in the broader community

I don't know the key to success, but the key to failure is trying to please everybody.

Bill Cosby (1937 -)

Supporting Stochastic Programs

- I point out all these different flavors of SP to highlight what I think has been one of the hinderances of acceptance of stochastic programming in the broader community

  I don't know the key to success, but the key to failure is trying to please everybody.

  Bill Cosby (1937 -)

- An Anecdote. ISMP XVIII, Copenhagen, 2003.
  - Irv Lustig, "Optimization Envanglist", ILOG

The Canonical Problem—Multistage Linear Recourse

- I will focus on (multistage) linear, recourse problems.

The Canonical Problem—Multistage Linear Recourse

- I will focus on (multistage) linear, recourse problems.
  - I know most about these.

The Canonical Problem—Multistage Linear Recourse

- I will focus on (multistage) linear, recourse problems.
  - I know most about these.
  - Maybe we can look at other types of stochastic programs next.

The Canonical Problem—Multistage Linear Recourse

- I will focus on (multistage) linear, recourse problems.
  - I know most about these.
  - Maybe we can look at other types of stochastic programs next.
  - I think this is very likely the most useful "stochastic program".

The Canonical Problem—Multistage Linear Recourse

- I will focus on (multistage) linear, recourse problems.
  - I know most about these.
  - Maybe we can look at other types of stochastic programs next.
  - I think this is very likely the most useful "stochastic program".
- Typically, we must make decision x before ω is known. But we have some recourse once we know ω.

The Canonical Problem—Multistage Linear Recourse

- I will focus on (multistage) linear, recourse problems.
  - I know most about these.
  - Maybe we can look at other types of stochastic programs next.
  - I think this is very likely the most useful "stochastic program".
- Typically, we must make decision x before ω is known. But we have some recourse once we know ω.

---

1. We make a decision now (first-period decision)

The Canonical Problem—Multistage Linear Recourse

- I will focus on (multistage) linear, recourse problems.
  - I know most about these.
  - Maybe we can look at other types of stochastic programs next.
  - I think this is very likely the most useful "stochastic program".
- Typically, we must make decision x before ω is known. But we have some recourse once we know ω.

---

1. We make a decision now (first-period decision)
2. Nature makes a random decision ("stuff" happens)

The Canonical Problem—Multistage Linear Recourse

- I will focus on (multistage) linear, recourse problems.
  - I know most about these.
  - Maybe we can look at other types of stochastic programs next.
  - I think this is very likely the most useful "stochastic program".
- Typically, we must make decision x before ω is known. But we have some recourse once we know ω.

---

1. We make a decision now (first-period decision)
2. Nature makes a random decision ("stuff" happens)
3. We make a second period decision that attempts to repair the havoc wrought by nature in (2). (recourse)

The Canonical Problem—Multistage Linear Recourse

- I will focus on (multistage) linear, recourse problems.
  - I know most about these.
  - Maybe we can look at other types of stochastic programs next.
  - I think this is very likely the most useful “stochastic program”.
- Typically, we must make decision x before ω is known. But we have some recourse once we know ω.

---

1. We make a decision now (first-period decision)
2. Nature makes a random decision (“stuff” happens)
3. We make a second period decision that attempts to repair the havoc wrought by nature in (2). (recourse)

- Let’s do a simple model...

Random Linear Programming

- Everyone's Favorite Problem. The Linear Program.

$$\min_{x \in X}\{c^T x \mid Ax = b\}$$

  - $X = \{x \in \Re^n : l \leq x \leq u\}$

Random Linear Programming

- Everyone's Favorite Problem. The Linear Program.

$$\min_{x \in X} \{c^T x \mid Ax = b\}$$

  - $X = \{x \in \Re^n : l \leq x \leq u\}$

- What if some parameters are random?

$$\min_{x \in X} \{c^T x \mid Ax = b, T(\omega)x = h(\omega)\}$$

The Recourse Game

- Again, we are interesting in solving decision problems where the decision x must be made before the realization of ω is known.

The Recourse Game

- Again, we are interesting in solving decision problems where the decision x must be made before the realization of ω is known.
- We do, however, know the distribution of ω on Ω.

The Recourse Game

- Again, we are interesting in solving decision problems where the decision x must be made before the realization of ω is known.
- We do, however, know the distribution of ω on Ω.
- In recourse models, the random constraints are modeled as "soft" constraints. Possible violation is accepted, but the cost of violations will influence the choice of x.

The Recourse Game

- Again, we are interesting in solving decision problems where the decision x must be made before the realization of ω is known.
- We do, however, know the distribution of ω on Ω.
- In recourse models, the random constraints are modeled as "soft" constraints. Possible violation is accepted, but the cost of violations will influence the choice of x.
- In fact, a *second-stage* linear program is introduced that will describe how the violated random constraints are dealt with.

Penalizing Shortfall with $LP(\omega)$

- In the simplest case, we may just count penalize deviation in the constraints by penalty coefficient vectors q_+ and q_-

minimize

$$c^T x + q_+^T s(\omega) + q_-^T t(\omega)$$

subject to

$$\begin{aligned} Ax &= b \\ T(\omega)x + s(\omega) - t(\omega) &= h(\omega) \\ x &\in X \end{aligned}$$

The New Optimization Problem

- So then, a reasonable problem to solve (to deal with the randomness) is...

minimize

$$c^T x + \mathbb{E}_\omega \left[q_+^T s(\omega) + q_-^T t(\omega) \right]$$

subject to

$$\begin{array}{rcll} Ax & = & b & \\ T(\omega)x + s(\omega) - t(\omega) & = & h(\omega) & \forall \omega \in \Omega \\ x & \in & X & \end{array}$$

Recourse

- In general, we can *react* in an intelligent (or optimal) way.

Recourse

- In general, we can *react* in an intelligent (or optimal) way.
- We have some *recourse!*

Recourse

- In general, we can *react* in an intelligent (or optimal) way.
- We have some *recourse!*
- A recourse structure is provided by three items

Recourse

- In general, we can *react* in an intelligent (or optimal) way.
- We have some *recourse!*
- A recourse structure is provided by three items
  - A set $Y \in \Re^p$ that describes the feasible set of recourse actions.

Recourse

- In general, we can *react* in an intelligent (or optimal) way.
- We have some *recourse!*
- A recourse structure is provided by three items
  - A set $Y \in \Re^p$ that describes the feasible set of recourse actions.
    - $Y = \{y \in \Re^p : y \geq 0\}$

Recourse

- In general, we can *react* in an intelligent (or optimal) way.
- We have some *recourse!*
- A recourse structure is provided by three items
  - A set $Y \in \Re^p$ that describes the feasible set of recourse actions.
    - $Y = \{y \in \Re^p : y \geq 0\}$
  - q : a vector of recourse costs.

Recourse

- In general, we can *react* in an intelligent (or optimal) way.
- We have some *recourse!*
- A recourse structure is provided by three items
  - A set $Y \in \Re^p$ that describes the feasible set of recourse actions.
    - $Y = \{y \in \Re^p : y \geq 0\}$
  - q : a vector of recourse costs.
  - W : a $m \times p$ matrix, called the *recourse matrix*

A Recourse Formulation

minimize

$$c^T x + \mathbb{E}_\omega \left[q^T y \right]$$

subject to

$$\begin{array}{rcll} Ax & = & b & \\ T(\omega)x + Wy(\omega) & = & h(\omega) & \quad \forall \omega \in \Omega \\ x & \in & X & \\ y(\omega) & \in & Y & \end{array}$$

Writing With the y's

$$\min_{x \in \Re^n, y(\omega) \in \Re^p} \mathbb{E}_\omega \left[c^T x + q^T y(\omega) \right]$$

subject to

$$\begin{array}{lllll} Ax & & = & b & \text{First Stage Constraints} \\ T(\omega)x & + \quad W y(\omega) & = & h(\omega) \quad \forall \omega \in \Omega & \text{Second Stage Constraints} \\ x \in X & \quad y(\omega) \in Y & & & \end{array}$$

- Imagine the case where $\Omega = \{\omega_1, \omega_2, \dots \omega_S\} \subseteq \Re^r$.
- $\mathrm{P}(\omega = \omega_s) = p_s, \forall s = 1, 2, \dots, S$
- $T_s \equiv T(\omega_s), h_s = h(\omega_s)$

Deterministic Equivalent

- We can then write the deterministic equivalent as:

$$
\begin{array}{ccccccccccc}
c^T x & + & p_1 q^T y_1 & + & p_2 q^T y_2 & + & \cdots & + & p_s q^T y_s & & \\
\text{s.t.} & & & & & & & & & & \\
Ax & & & & & & & & & = & b \\
T_1 x & + & W y_1 & & & & & & & = & h_1 \\
T_2 x & & & + & W y_2 & & & & & = & h_2 \\
\vdots & & & + & & & \ddots & & & & \\
T_S x & & & & & & & + & W y_s & = & h_s \\
x \in X & & y_1 \in Y & & y_2 \in Y & & & & y_s \in Y & &
\end{array}
$$

About the DE

- $y_s \equiv y(\omega_s)$ is the recourse action to take if scenario ω_s occurs.

About the DE

- $y_s \equiv y(\omega_s)$ is the recourse action to take if scenario ω_s occurs.
- Pro: It's a linear program.

About the DE

- $y_s \equiv y(\omega_s)$ is the recourse action to take if scenario ω_s occurs.
- Pro: It's a linear program.
- Con: It's a big linear program.

About the DE

- $y_s \equiv y(\omega_s)$ is the recourse action to take if scenario ω_s occurs.
- Pro: It's a linear program.
- Con: It's a big linear program.
- How BIG is it?

About the DE

- $y_s \equiv y(\omega_s)$ is the recourse action to take if scenario ω_s occurs.
- Pro: It's a linear program.
- Con: It's a big linear program.
- How BIG is it?
- Imagine the following (real) problem. A Telecom company wants to expand its network in a way in which to meet an unknown (random) demand.

About the DE

- $y_s \equiv y(\omega_s)$ is the recourse action to take if scenario ω_s occurs.
- Pro: It's a linear program.
- Con: It's a big linear program.
- How BIG is it?
- Imagine the following (real) problem. A Telecom company wants to expand its network in a way in which to meet an unknown (random) demand.
- There are 86 unknown demands. Each demand is independent and may take on one of five values.

About the DE

- $y_s \equiv y(\omega_s)$ is the recourse action to take if scenario ω_s occurs.
- Pro: It's a linear program.
- Con: It's a big linear program.
- How BIG is it?
- Imagine the following (real) problem. A Telecom company wants to expand its network in a way in which to meet an unknown (random) demand.
- There are 86 unknown demands. Each demand is independent and may take on one of five values.
- $S = |\Omega| = \Pi_{k=1}^{86}(5) = 5^{86} = 4.77 \times 10^{72}$

About the DE

- $y_s \equiv y(\omega_s)$ is the recourse action to take if scenario ω_s occurs.
- Pro: It's a linear program.
- Con: It's a big linear program.
- How BIG is it?
- Imagine the following (real) problem. A Telecom company wants to expand its network in a way in which to meet an unknown (random) demand.
- There are 86 unknown demands. Each demand is independent and may take on one of five values.
- $S = |\Omega| = \Pi_{k=1}^{86}(5) = 5^{86} = 4.77 \times 10^{72}$
  - The number of subatomic particles in the universe.

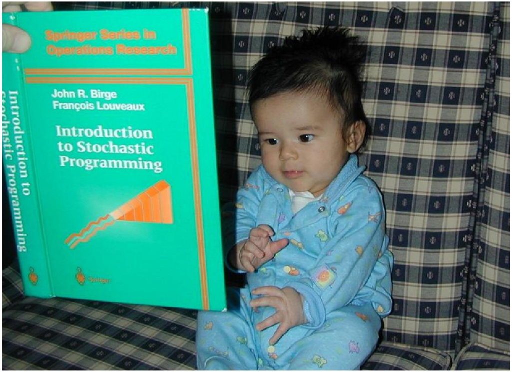
Why Don't More People Use Stochastic Programming

Why Don't More People Use Stochastic Programming

They don't start their training early enough!

Why Don't More People Use Stochastic Programming

They don't start their training early enough!



Why Don't More People Use Stochastic Programming

- Because they can't "solve" them? Try Sampling!

Why Don't More People Use Stochastic Programming

- Because they can't "solve" them? Try Sampling!

$$\min_{x \in X} \{ f(x) \stackrel{\text{def}}{=} \mathbb{E}_P F(x, \omega) \equiv \int_\Omega F(x, \omega) dP(\omega) \}$$

Why Don't More People Use Stochastic Programming

- Because they can't "solve" them? Try Sampling!

$$\min_{x \in X} \{ f(x) \stackrel{\text{def}}{=} \mathbb{E}_P F(x, \omega) \equiv \int_{\Omega} F(x, \omega) dP(\omega) \}$$

- Draw $\omega^1, \omega^2, \ldots \omega^N$ from P
- Sample Average Approximation (SAA):

$$\widehat{f}_N(x) \equiv N^{-1} \sum_{j=1}^{N} g(x, \omega^j)$$

Why Don't More People Use Stochastic Programming

- Because they can't "solve" them? Try Sampling!

$$\min_{x \in X}\{f(x) \stackrel{\text{def}}{=} \mathbb{E}_P F(x,\omega) \equiv \int_\Omega F(x,\omega) dP(\omega)\}$$

- Draw $\omega^1, \omega^2, \ldots \omega^N$ from P
- Sample Average Approximation (SAA):

$$\widehat{f}_N(x) \equiv N^{-1} \sum_{j=1}^{N} g(x, \omega^j)$$

- $\widehat{f}_N(x)$ is an unbiased estimator of $f(x)$ ($\mathbb{E}[\widehat{f}_N(x)] = f(x)$).

Why Don't More People Use Stochastic Programming

- Because they can't "solve" them? Try Sampling!

$$\min_{x\in X}\{f(x) \stackrel{\text{def}}{=} \mathbb{E}_P F(x,\omega) \equiv \int_\Omega F(x,\omega)dP(\omega)\}$$

- Draw $\omega^1, \omega^2, \ldots \omega^N$ from P
- Sample Average Approximation (SAA):

$$\widehat{f}_N(x) \equiv N^{-1} \sum_{j=1}^{N} g(x, \omega^j)$$

- $\widehat{f}_N(x)$ is an unbiased estimator of $f(x)$ ($\mathbb{E}[\widehat{f}_N(x)] = f(x)$).
- Minimize the SAA: $\min_{x\in X}\{\widehat{f}_N(x)\}$

Sampling is Good!

- For two-stage stochastic recourse problems, some *very interesting* recent theory of Shapiro and Homem-de-Mello has shown that you need shockingly few scenarios in order for the solution of the sample average approximation to be a very good solution to the true problem
- This theory has been backed up with computational experience.
  - For a problem with 10^{81} scenarios, a 100 scenario sample was sufficient.
  - A different instance with 10^{70} scenarios required around a 5000 scenario sample

Solving "Medium Sized" Problems

- Formulate as "two-level" problem

$$\min_{x \in \mathbb{R}^n_+ : Ax = b} \left\{ c^T x + \mathbb{E}_\omega \left[\min_{y \in Y} \{ q^T y : Wy = h(\omega) - T(\omega)x \} \right] \right\}$$

- Second stage value function, or Cost-to-go function $v : \Re^m \mapsto \Re$.
  - $v(z) \equiv \min_{y \in Y} \{ q^T y : Wy = z \}$,
- Expected Value Function, or Expected cost-to-go function $\mathcal{Q} : \Re^n \mapsto \Re$.
  - $\mathcal{Q}(x) \equiv \mathbb{E}_\omega [v(h(\omega) - T(\omega)x)]$
  - For any policy $x \in \Re^n$, it describes the expected cost of the recourse.

The SP Problem

- Using these definitions, we can write our recourse problem in terms only of the x variables:

$$\min_{x \in X} \{c^T x + \mathcal{Q}(x) : Ax = b\}$$

- This is a (nonlinear) programming problem in $\Re^n$.
- The ease of solving such a problem depends on the properties of $\mathcal{Q}(x)$.
- $\mathcal{Q}(x)$ is...
  - Convex...
  - Continuous...
  - Non-Differentiable

Important (and well-known) Facts

- $\mathcal{Q}(x)$ is a piecewise linear convex function of x.
- If π_i is an optimal dual solution to the linear program corresponding to ith scenario, then $T_i^T \pi_i \in \partial \mathcal{Q}(\hat{x})$
- $g(\hat{x}) \stackrel{\text{def}}{=} \sum_{i \in S} p_i T_i^T \pi_i \in \partial \mathcal{Q}(\hat{x})$.
- Evaluation of $\mathcal{Q}(\hat{x})$ is separable
  - We can solve linear programs corresponding to each $\mathcal{Q}(\hat{x})$ independently – in parallel!

Best-Known Solution Procedure

Best-Known Solution Procedure

M
E
T
H O D

L-shaped method

- Represent $\mathcal{Q}(x)$ by an artificial variable θ and find supporting planes for θ (from subgradients of $\mathcal{Q}(x^k)$).
  - $\theta \geq g(x^k)^T x + (\mathcal{Q}(x^k) - g^T x^k)$ (*)

L-shaped method

- Represent $\mathcal{Q}(x)$ by an artificial variable θ and find supporting planes for θ (from subgradients of $\mathcal{Q}(x^k)$).
  - $\theta \geq g(x^k)^T x + (\mathcal{Q}(x^k) - g^T x^k)$ (*)

---

1. Solve the **master problem** with the current θ_j-approximations to $\mathcal{Q}(x)$ for x^k.

L-shaped method

- Represent $\mathcal{Q}(x)$ by an artificial variable θ and find supporting planes for θ (from subgradients of $\mathcal{Q}(x^k)$).
  - $\theta \geq g(x^k)^T x + (\mathcal{Q}(x^k) - g^T x^k)$ (*)

---

1. Solve the **master problem** with the current θ_j-approximations to $\mathcal{Q}(x)$ for x^k.
2. Solve the **subproblems**, evaluating $\mathcal{Q}(x^k)$ and obtaining a subgradient $g(x^k)$. Add inequalities (*) to the master problem

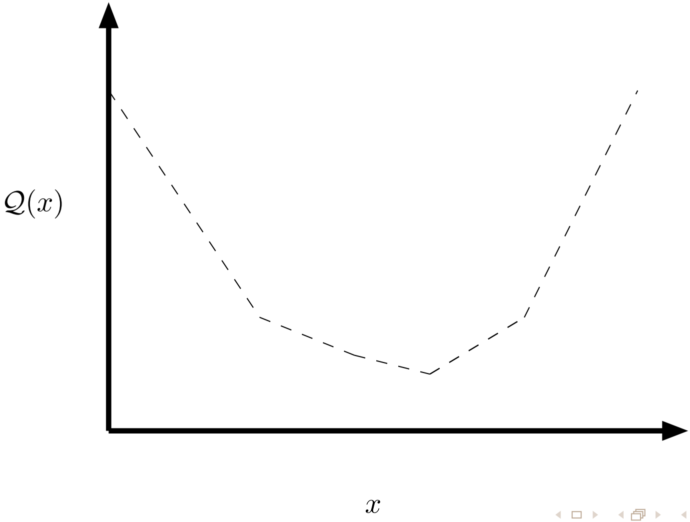
L-shaped method

- Represent $\mathcal{Q}(x)$ by an artificial variable θ and find supporting planes for θ (from subgradients of $\mathcal{Q}(x^k)$).
  - $\theta \geq g(x^k)^T x + (\mathcal{Q}(x^k) - g^T x^k)$ (*)

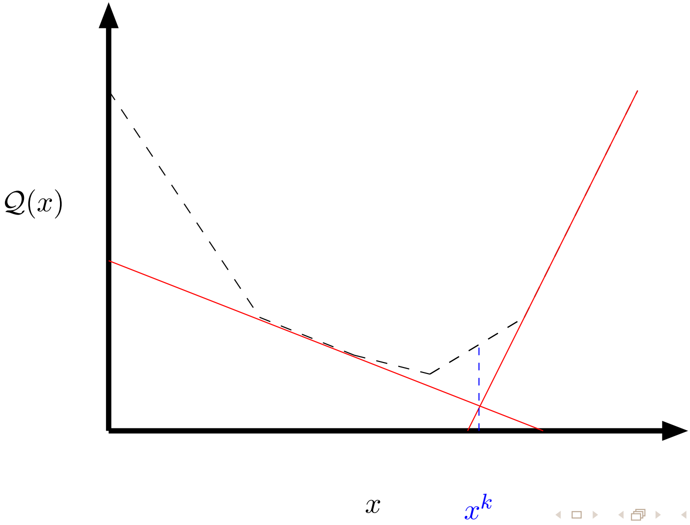
---

1. Solve the **master problem** with the current θ_j-approximations to $\mathcal{Q}(x)$ for x^k.
2. Solve the **subproblems**, evaluating $\mathcal{Q}(x^k)$ and obtaining a subgradient $g(x^k)$. Add inequalities (*) to the master problem
3. k = k+1. Goto 1.

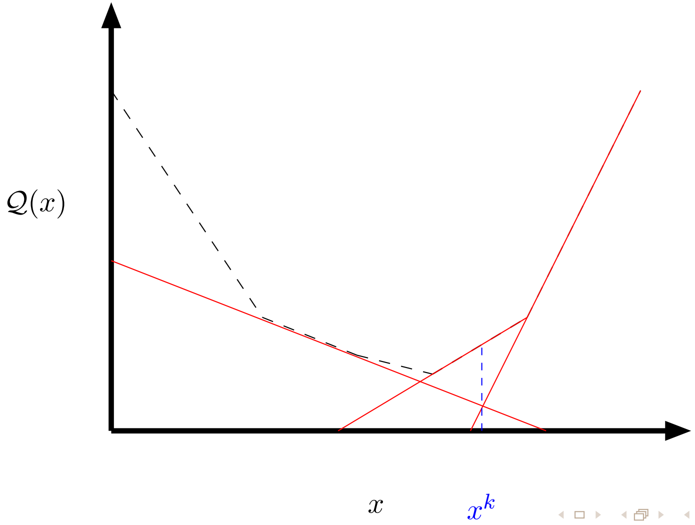
Worth 1000 Words



Worth 1000 Words



Worth 1000 Words

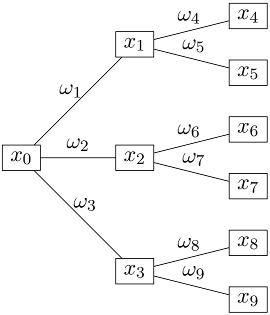


Even Harder—Multistage Problems

- Multistage problems are defined by a sequence of decision, event, decision, event, . . ., decision.

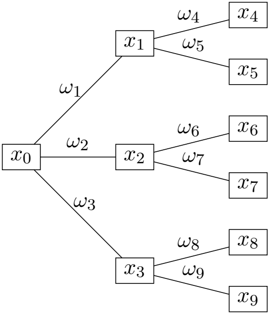
Even Harder—Multistage Problems

- Multistage problems are defined by a sequence of decision, event, decision, event, . . ., decision.



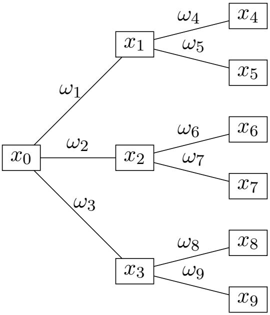
Even Harder—Multistage Problems

- Multistage problems are defined by a sequence of decision, event, decision, event, . . ., decision.
- Multistage problems are even bigger (scenarios grow again at a rate exponential in the number of stages)



Even Harder—Multistage Problems

- Multistage problems are defined by a sequence of decision, event, decision, event, . . ., decision.
- Multistage problems are even bigger (scenarios grow again at a rate exponential in the number of stages)
- We have to keep track of the random event "structure"—the scenario tree—and its relationship to the decisions that we make



Existing(?) Tools

Name	Author(s)	Comment
Gams	Dirkse, Gams	Commercial
XPRESS-SP	Verma, Dash Opt.	Commercial, Beta
SPiNE	Valente, CARISMA	
STRUMS	Fourer and Lopes	Prototype(?)
SUTIL	Czyzyk and Linderoth	C++ classes
SLPLib	Felt, Sarich, Ariyawansa	Open Source C Routines
COIN-Smi, SP/OSL	COIN, IBM	C++ methods

- I am happy to show off the XPRESS-SP tool if anyone is interested.

Stochastic MIP

- Recall that if Ω was finite, we could write the (deterministic equivalent) of a stochastic LP

Stochastic MIP

- Recall that if Ω was finite, we could write the (deterministic equivalent) of a stochastic LP
  - Just a large scale LP

Stochastic MIP

- Recall that if Ω was finite, we could write the (deterministic equivalent) of a stochastic LP
  - Just a large scale LP
- We can do the same for stochastic MIP

Stochastic MIP

- Recall that if Ω was finite, we could write the (deterministic equivalent) of a stochastic LP
  - Just a large scale LP
- We can do the same for stochastic MIP
  - Just a large-scale IP

Stochastic MIP

- Recall that if Ω was finite, we could write the (deterministic equivalent) of a stochastic LP
  - Just a large scale LP
- We can do the same for stochastic MIP
  - Just a large-scale IP
  - But a large-scale IP with a very weak linear programming relaxation $\Rightarrow$ not likely to be solved by "off-the-shelf" software like cplex.

Nasty, Nasty, Functions

- If you didn't fall asleep during the mathy part, recall that our L-Shaped method for stochastic LP was based on knowing "nice" properties of the second stage value function ($v(z)$) or the Expected Value Function $\mathcal{Q}(x)$.
- For IP...

$$v(z) = \min_{y \in \mathbb{Z}^n_+} \{q^T y | Wy = z\}$$

- Here are two properties...
  - $v(z)$ is lower semicontinuous on $\Re^m$
  - The discontinuity points of v are contained in a countable union of hyperplanes in $\Re^m$
- These are not very powerful properties

Algorithms for Stochastic IP

- Integer L-Shaped method

Algorithms for Stochastic IP

- Integer L-Shaped method
  - A cutting-plane based method

Algorithms for Stochastic IP

- Integer L-Shaped method
  - A cutting-plane based method
- Dual Decomposition Method

Algorithms for Stochastic IP

- Integer L-Shaped method
  - A cutting-plane based method
- Dual Decomposition Method
  - IMO, "The way to go". Based on a Lagrangian relaxation of nonanticipativy

Algorithms for Stochastic IP

- Integer L-Shaped method
  - A cutting-plane based method
- Dual Decomposition Method
  - IMO, “The way to go”. Based on a Lagrangian relaxation of nonanticipativy
- Stochastic Branch-and-Bound

Algorithms for Stochastic IP

- Integer L-Shaped method
  - A cutting-plane based method
- Dual Decomposition Method
  - IMO, "The way to go". Based on a Lagrangian relaxation of nonanticipativy
- Stochastic Branch-and-Bound
  - Uses Monte-Carlo based bounds

Algorithms for Stochastic IP

- Integer L-Shaped method
  - A cutting-plane based method
- Dual Decomposition Method
  - IMO, "The way to go". Based on a Lagrangian relaxation of nonanticipativy
- Stochastic Branch-and-Bound
  - Uses Monte-Carlo based bounds
- Structured Enumeration

Algorithms for Stochastic IP

- Integer L-Shaped method
  - A cutting-plane based method
- Dual Decomposition Method
  - IMO, "The way to go". Based on a Lagrangian relaxation of nonanticipativy
- Stochastic Branch-and-Bound
  - Uses Monte-Carlo based bounds
- Structured Enumeration
  - Based on strange mathematical entities like *test sets and* Groebner Bases

Conclusions

- You cannot condense stochastic programming into a one-hour course

Conclusions

- You cannot condense stochastic programming into a one-hour course
- Jeff should not wait until the last minute to prepare his slides

Conclusions

- You cannot condense stochastic programming into a one-hour course
- Jeff should not wait until the last minute to prepare his slides

---

- Stochastic Programming can be useful

Conclusions

- You cannot condense stochastic programming into a one-hour course
- Jeff should not wait until the last minute to prepare his slides

---

- Stochastic Programming can be useful
- Stochastic Programming is hard

Conclusions

- You cannot condense stochastic programming into a one-hour course
- Jeff should not wait until the last minute to prepare his slides

---

- Stochastic Programming can be useful
- Stochastic Programming is hard

The Major Conclusion

Stochastic Programming is worthwhile to study a bit more!

Your Next Mission...

- Stochastic Integer Programming is going to be our next topic
- Suvrajeet Sen from NSF will come speak in Friday Seminar on 9/17
- We're going to read a survey paper for next week.

Your Next Mission...

- Stochastic Integer Programming is going to be our next topic
- Suvrajeet Sen from NSF will come speak in Friday Seminar on 9/17
- We're going to read a survey paper for next week.

Rüdiger Schultz, "Stochastic programming with integer variables," *Mathematical Programming, Series B*, Vol. 97, 285-309, 2003.