Experiments with Robust Optimization

Daniel Bienstock

Columbia University
New York

ISMP 2006, Rio

Robust Optimization

- Optimization under parameter (data) uncertainty
- Ben-Tal and Nemirovsky, El Ghaoui et al
- Bertsimas et al
- Uncertainty is modeled by assuming that data is not known precisely, and will instead lie in known sets.
- Example: a coefficient a_i is uncertain. We allow $a_i \in [l_i, u_i]$.
- Typically, a **minimization** problem becomes a **min-max** problem.

Robust Optimization

- Optimization under parameter (data) uncertainty
- Ben-Tal and Nemirovsky, El Ghaoui et al
- Bertsimas et al
- Uncertainty is modeled by assuming that data is not known precisely, and will instead lie in known sets.
- Example: a coefficient a_i is uncertain. We allow $a_i \in [l_i, u_i]$.
- Typically, a **minimization** problem becomes a **min-max** problem.

Robust Optimization

- Optimization under parameter (data) uncertainty
- Ben-Tal and Nemirovsky, El Ghaoui et al
- Bertsimas et al
- Uncertainty is modeled by assuming that data is not known precisely, and will instead lie in known sets.
- Example: a coefficient a_i is uncertain. We allow $a_i \in [l_i, u_i]$.
- Typically, a **minimization** problem becomes a **min-max** problem.

Robust Optimization

- Optimization under parameter (data) uncertainty
- Ben-Tal and Nemirovsky, El Ghaoui et al
- Bertsimas et al
- Uncertainty is modeled by assuming that data is not known precisely, and will instead lie in known sets.
- Example: a coefficient a_i is uncertain. We allow $a_i \in [l_i, u_i]$.
- Typically, a **minimization** problem becomes a **min-max** problem.

Robust Optimization

- Optimization under parameter (data) uncertainty
- Ben-Tal and Nemirovsky, El Ghaoui et al
- Bertsimas et al
- Uncertainty is modeled by assuming that data is not known precisely, and will instead lie in known sets.
- Example: a coefficient a_i is uncertain. We allow $a_i \in [l_i, u_i]$.
- Typically, a **minimization** problem becomes a **min-max** problem.

Robust Optimization

- Optimization under parameter (data) uncertainty
- Ben-Tal and Nemirovsky, El Ghaoui et al
- Bertsimas et al
- Uncertainty is modeled by assuming that data is not known precisely, and will instead lie in known sets.
- Example: a coefficient a_i is uncertain. We allow $a_i \in [l_i, u_i]$.
- Typically, a **minimization** problem becomes a **min-max** problem.

Example: Linear Programs with Row-Wise uncertainty

Ben-Tal and Nemirovsky, 1999 (also: Soyster (1973))

$\min c^t x$
Subject to:
$Ax \geq b$ **for all** $A \in \mathcal{U}$
$x \in X$

$\mathcal{U} =$ **uncertainty set**
$\longrightarrow$ **the** i^{th} **row of A belongs to an** ellipsoidal **set** $\mathcal{E}_i$
e.g. $\sum_j \alpha_{ij}^2 (a_{ij} - \bar{a}_{ij})^2 \leq 1$

$\longrightarrow$ can be solved using SOCP techniques

Other forms of optimization under uncertainty

- Stochastic programming
- Adversarial queueing, online optimization
- “Risk-aware” optimization
- Optimization of utility functions as a substitute for handling infeasibilities

Other forms of optimization under uncertainty

- Stochastic programming
- Adversarial queueing, online optimization
- "Risk-aware" optimization
- Optimization of utility functions as a substitute for handling infeasibilities

Other forms of optimization under uncertainty

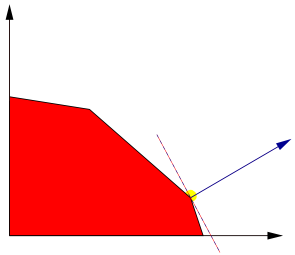
- Stochastic programming
- Adversarial queueing, online optimization
- “Risk-aware” optimization
- Optimization of utility functions as a substitute for handling infeasibilities

Other forms of optimization under uncertainty

- Stochastic programming
- Adversarial queueing, online optimization
- “Risk-aware” optimization
- Optimization of utility functions as a substitute for handling infeasibilities

Scenario I: Stability

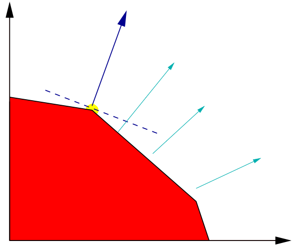
Data is fairly accurate, though possibly noisy – small errors are possible



→ Idiosyncratic decisions and small changes in data could have major impact

Scenario I: Stability

Data is fairly accurate, though possibly noisy – small errors are possible



$\longrightarrow$ Idiosyncratic decisions and small changes in data could have major impact

Scenario II: Hedging

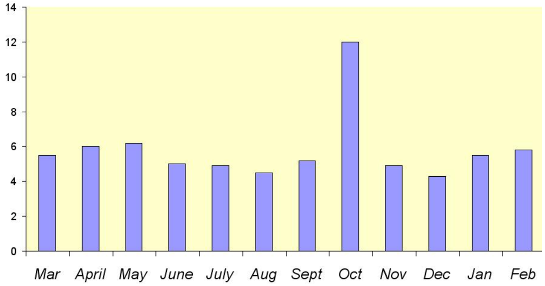
Significant, but within order-of-magnitude, data uncertainty

Example:
A certain parameter, α, is volatile. Its long-term average is **1.5** but it we could expect changes of the order of **.3**.

- Possibly more than just noise
- Could use deviations to our advantage, especially if there are several uncertain parameters that act "correlated"
- Are we guarding against risk or are we hedging?

Scenario III: Insurance

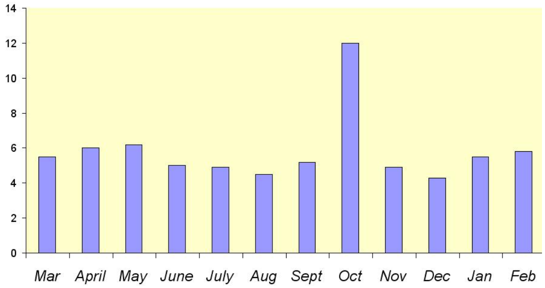
Real world data can exhibit undesirable and unexpected behavior



- Classical goal: how can we protect without becoming too risk averse
- Need to clearly spell out desired tradeoff between risk and performance
- Magnitude and geometry of risk are not the same

Scenario III: Insurance

Real world data can exhibit undesirable and unexpected behavior



- Classical goal: how can we protect without becoming too risk averse
- Need to clearly spell out desired tradeoff between risk and performance
- Magnitude and geometry of risk are not the same

Application: Portfolio Optimization

- $$\min \lambda x^T Qx \; - \; \mu^T x$$

  Subject to:

  $$Ax \geq b$$

- μ = vector of “returns”, Q = “covariance” matrix
- x = vector of “asset weights”
- $Ax \geq b$: general linear constraints
- $\lambda \geq 0$ = “risk-aversion” multiplier

Application: Portfolio Optimization

- $$\min \lambda x^T Q x \; - \; \mu^T x$$

  Subject to:

  $$Ax \geq b$$

- μ = vector of "returns", Q = "covariance" matrix
- x = vector of "asset weights"
- $Ax \geq b$: general linear constraints
- $\lambda \geq 0$ = "risk-aversion" multiplier

Application: Portfolio Optimization

- $$\min \lambda x^T Q x \; - \; \mu^T x$$

  Subject to:

  $$Ax \geq b$$

- μ = vector of "returns", Q = "covariance" matrix
- x = vector of "asset weights"
- $Ax \geq b$: general linear constraints
- $\lambda \geq 0$ = "risk-aversion" multiplier

Application: Portfolio Optimization

- $$\min \lambda x^T Q x \; - \; \mu^T x$$

  Subject to:

  $$Ax \ge b$$

- μ = vector of "returns", Q = "covariance" matrix
- x = vector of "asset weights"
- $Ax \ge b$: general linear constraints
- $\lambda \ge 0$ = "risk-aversion" multiplier

Application: Portfolio Optimization

- $$\min \lambda x^T Q x \ - \ \mu^T x$$

  Subject to:

  $$Ax \geq b$$

- $\mu =$ vector of “returns”, $Q =$ “covariance” matrix
- $x =$ vector of “asset weights”
- $Ax \geq b$: general linear constraints
- $\lambda \geq 0 =$ “risk-aversion” multiplier

Robust Portfolio Optimization

Goldfarb and Iyengar, 2001

$\rightarrow$ $\boldsymbol{Q}$ and μ are uncertain

Robust Problem

$$\min_x \left\{ \max_{Q \in \mathcal{Q}} \lambda x^T Q x \; - \; \min_{\mu \in \mathcal{E}} \mu^T x \right\}$$

Subject to:

$$\sum_j x_j = 1, \quad x \geq 0$$

$\rightarrow$ When $\mathcal{Q}$ is an ellipsoid and $\mathcal{E}$ is a product of intervals the robust problem can be solved as an SOCP

Robust Portfolio Optimization

A different uncertainty model

$\longrightarrow$ Want to model that deviations of the returns μ_j from their nominal values are rare but could be significant

A simple example

- Parameters: $0 \leq \gamma \leq 1$, integer $N \geq 0$,
  for each asset j:
  $\bar{\mu}_j =$ expected return, $0 \leq \delta_j$ small (possibly zero)
- **Well-behaved asset j:** $\bar{\mu}_j - \delta_j \leq \mu_j \leq \bar{\mu}_j + \delta_j$
- **Misbehaving asset j:** $(1 - \gamma)\bar{\mu}_j \leq \mu_j \leq \bar{\mu}_j$
- At most N assets misbehave

Robust Portfolio Optimization

A different uncertainty model

$\longrightarrow$ Want to model that deviations of the returns μ_j from their nominal values are rare but could be significant

A simple example

- Parameters: $0 \le \gamma \le 1$, integer $N \ge 0$,
  for each asset j:
  $\bar{\mu}_j$ = expected return, $0 \le \delta_j$ small (possibly zero)
- **Well-behaved asset j:** $\bar{\mu}_j - \delta_j \le \mu_j \le \bar{\mu}_j + \delta_j$
- **Misbehaving asset j:** $(1-\gamma)\bar{\mu}_j \le \mu_j \le \bar{\mu}_j$
- At most N assets misbehave

Robust Portfolio Optimization

A different uncertainty model

$\longrightarrow$ Want to model that deviations of the returns μ_j from their nominal values are rare but could be significant

A simple example

- Parameters: $0 \leq \gamma \leq 1$, integer $N \geq 0$,
  for each asset j:
  $\bar{\mu}_j =$ expected return, $0 \leq \delta_j$ small (possibly zero)
- **Well-behaved asset j:** $\bar{\mu}_j - \delta_j \ \leq \ \mu_j \ \leq \ \bar{\mu}_j + \delta_j$
- **Misbehaving asset j:** $(1 - \gamma)\bar{\mu}_j \ \leq \ \mu_j \ \leq \ \bar{\mu}_j$
- At most N assets misbehave

Robust Portfolio Optimization

A different uncertainty model

$\longrightarrow$ Want to model that deviations of the returns μ_j from their nominal values are rare but could be significant

A simple example

- Parameters: $0 \leq \gamma \leq 1$, integer $N \geq 0$,
  for each asset j:
  $\bar{\mu}_j =$ expected return, $0 \leq \delta_j$ small (possibly zero)
- **Well-behaved asset j:** $\bar{\mu}_j - \delta_j \ \leq \ \mu_j \ \leq \ \bar{\mu}_j + \delta_j$
- **Misbehaving asset j:** $(1-\gamma)\bar{\mu}_j \ \leq \ \mu_j \ \leq \ \bar{\mu}_j$
- At most N assets misbehave

A more comprehensive setting

- Parameters: $\mathbf{0} \le \gamma_{\mathbf{1}} \le \gamma_{\mathbf{2}} \le \ldots \le \gamma_{\boldsymbol{K}} \le \mathbf{1}$,
  integers $\mathbf{0} \le \boldsymbol{n_i} \le \boldsymbol{N_i}$, $\mathbf{1} \le \boldsymbol{i} \le \boldsymbol{K}$

  for each asset j: $\bar{\mu}_j =$ expected return

- between $\boldsymbol{n_i}$ and $\boldsymbol{N_i}$ assets j satisfy:

$$(\mathbf{1} - \gamma_i)\bar{\mu}_j \le \mu_j \le (\mathbf{1} - \gamma_{i-1})\bar{\mu}_j, \text{ for each } \boldsymbol{i} \ge \mathbf{1} \quad (\gamma_0 = \mathbf{0})$$

A more comprehensive setting

- Parameters: $0 \leq \gamma_1 \leq \gamma_2 \leq \ldots \leq \gamma_K \leq 1$, integers $0 \leq n_i \leq N_i$, $1 \leq i \leq K$

  for each asset j: $\bar{\mu}_j =$ expected return

- between n_i and N_i assets j satisfy:

$$(1 - \gamma_i)\bar{\mu}_j \leq \mu_j \leq (1 - \gamma_{i-1})\bar{\mu}_j, \text{ for each } i \geq 1 \quad (\gamma_0 = 0)$$

A more comprehensive setting

Alternative

- Parameters: $\mathbf{0} \leq \gamma_1 \leq \gamma_2 \leq \ldots \leq \gamma_K \leq \mathbf{1}$,
  integers $\mathbf{0} \leq \boldsymbol{n_i} \leq \boldsymbol{N_i}$, $\mathbf{1} \leq \boldsymbol{i} \leq \boldsymbol{K}$

  for each asset j: $\bar{\mu}_j$ = expected return, $\bar{\delta}_j$ = "standard deviation" of return

- between $\boldsymbol{n_i}$ and $\boldsymbol{N_i}$ assets j satisfy:

$$\bar{\mu}_j - \gamma_i \delta_j \leq \mu_j \leq \bar{\mu}_j - \gamma_{i-1} \delta_j$$

A more comprehensive setting

Alternative

- Parameters: $0 \leq \gamma_1 \leq \gamma_2 \leq \ldots \leq \gamma_K \leq 1$,
  integers $0 \leq n_i \leq N_i$, $1 \leq i \leq K$

  for each asset j: $\bar{\mu}_j$ = expected return, $\bar{\delta}_j$ = "standard deviation" of return

- between n_i and N_i assets j satisfy:

$$\bar{\mu}_j - \gamma_i \delta_j \;\leq\; \mu_j \;\leq\; \bar{\mu}_j - \gamma_{i-1} \delta_j$$

A more comprehensive setting

- Parameters: $\mathbf{0} \leq \gamma_1 \leq \gamma_2 \leq \ldots \leq \gamma_K \leq \mathbf{1}$,
  integers $\mathbf{0} \leq n_i \leq N_i$, $\mathbf{1} \leq i \leq K$
  for each asset j: $\bar{\mu}_j =$ expected return

- between n_i and N_i assets j satisfy:
  $(1 - \gamma_i)\bar{\mu}_j \leq \mu_j \leq (1 - \gamma_{i-1})\bar{\mu}_j$

- $\sum_j \mu_j \geq \Gamma \sum_j \bar{\mu}_j$; $\Gamma > 0$ a parameter

- (R. Tütüncü) For $1 \leq h \leq H$,

  - a **set** ("tier") T_h of assets, and a parameter $\Gamma_h > 0$

  for each h, $\sum_{j \in T_h} \mu_j \geq \Gamma_h \sum_{j \in S_h} \bar{\mu}_j$

Note: only downwards changes are modeled

A more comprehensive setting

- Parameters: $\mathbf{0} \leq \gamma_1 \leq \gamma_2 \leq \ldots \leq \gamma_K \leq \mathbf{1}$,
  integers $\mathbf{0} \leq n_i \leq N_i$, $\mathbf{1} \leq i \leq K$
  for each asset j: $\bar{\mu}_j =$ expected return
- between n_i and N_i assets j satisfy:
  $(1 - \gamma_i)\bar{\mu}_j \leq \mu_j \leq (1 - \gamma_{i-1})\bar{\mu}_j$
- $\sum_j \mu_j \geq \Gamma \sum_j \bar{\mu}_j$; $\Gamma > 0$ a parameter
- (R. Tütüncü) For $1 \leq h \leq H$,
  - a **set** ("tier") T_h of assets, and a parameter $\Gamma_h > 0$

  for each h, $\sum_{j \in T_h} \mu_j \geq \Gamma_h \sum_{j \in S_h} \bar{\mu}_j$

Note: only downwards changes are modeled

A more comprehensive setting

- Parameters: $\mathbf{0} \leq \gamma_1 \leq \gamma_2 \leq \ldots \leq \gamma_K \leq \mathbf{1}$,
  integers $\mathbf{0} \leq n_i \leq N_i$, $\mathbf{1} \leq i \leq K$
  for each asset j: $\bar{\mu}_j =$ expected return

- between n_i and N_i assets j satisfy:
  $(1 - \gamma_i)\bar{\mu}_j \leq \mu_j \leq (1 - \gamma_{i-1})\bar{\mu}_j$

- $\sum_j \mu_j \geq \Gamma \sum_j \bar{\mu}_j$; $\Gamma > 0$ a parameter

- (R. Tütüncü) For $1 \leq h \leq H$,
  - a **set** (“tier”) T_h of assets, and a parameter $\Gamma_h > 0$

  for each h, $\sum_{j \in T_h} \mu_j \geq \Gamma_h \sum_{j \in S_h} \bar{\mu}_j$

Note: only downwards changes are modeled

A more comprehensive setting

- Parameters: $\mathbf{0} \leq \gamma_1 \leq \gamma_2 \leq \ldots \leq \gamma_K \leq \mathbf{1}$,
  integers $\mathbf{0} \leq \boldsymbol{n_i} \leq \boldsymbol{N_i}$, $\mathbf{1} \leq \boldsymbol{i} \leq \boldsymbol{K}$
  for each asset j: $\bar{\mu}_j =$ expected return
- between $\boldsymbol{n_i}$ and $\boldsymbol{N_i}$ assets j satisfy:
  $(1 - \gamma_i)\bar{\mu}_j \leq \mu_j \leq (1 - \gamma_{i-1})\bar{\mu}_j$
- $\sum_j \mu_j \geq \Gamma \sum_j \bar{\mu}_j$; $\Gamma > 0$ a parameter
- (R. Tütüncü) For $\mathbf{1} \leq \boldsymbol{h} \leq \boldsymbol{H}$,
  - a **set** (“tier”) $\boldsymbol{T_h}$ of assets, and a parameter $\Gamma_h > 0$

for each $\boldsymbol{h}$, $\sum_{j \in T_h} \mu_j \geq \Gamma_h \sum_{j \in S_h} \bar{\mu}_j$

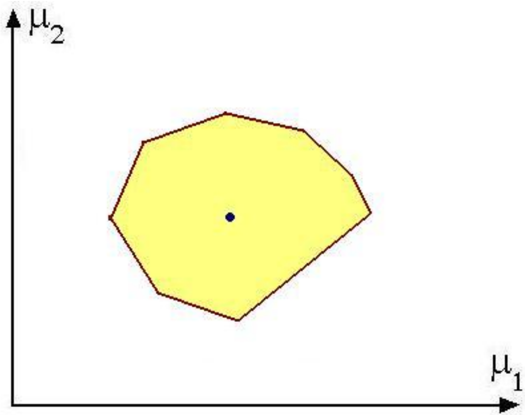
Note: only downwards changes are modeled

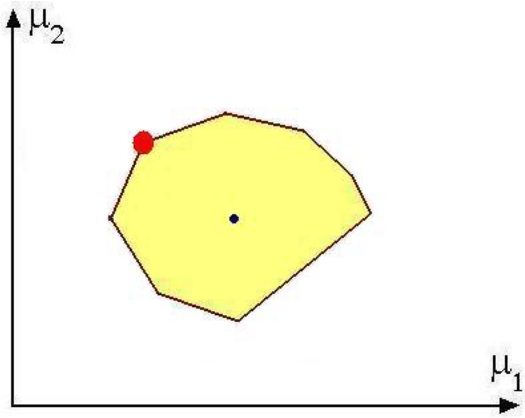
A more comprehensive setting

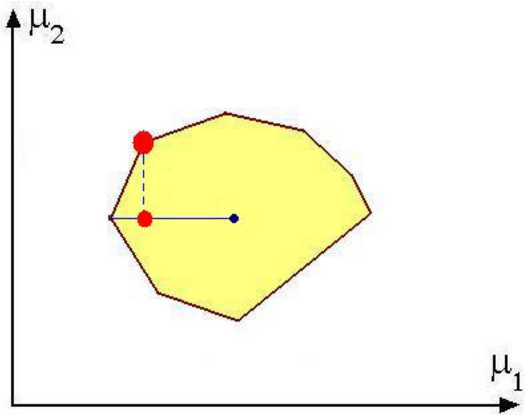
- Parameters: $0 \leq \gamma_1 \leq \gamma_2 \leq \ldots \leq \gamma_K \leq 1$,
  integers $0 \leq n_i \leq N_i$, $1 \leq i \leq K$
  for each asset j: $\bar{\mu}_j =$ expected return
- between n_i and N_i assets j satisfy:
  $(1 - \gamma_i)\bar{\mu}_j \leq \mu_j \leq (1 - \gamma_{i-1})\bar{\mu}_j$
- $\sum_j \mu_j \geq \Gamma \sum_j \bar{\mu}_j$; $\Gamma > 0$ a parameter
- (R. Tütüncü) For $1 \leq h \leq H$,
  - a **set** (“tier”) T_h of assets, and a parameter $\Gamma_h > 0$

  for each h, $\sum_{j \in T_h} \mu_j \geq \Gamma_h \sum_{j \in S_h} \bar{\mu}_j$

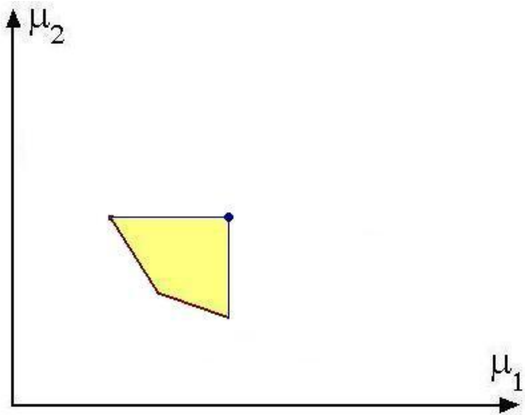
Note: only downwards changes are modeled


μ_2
μ_1


μ_2
μ_1



μ_2
μ_1



μ_2
μ_1

General methodology:

Benders' decomposition (= cutting-plane algorithm)

Generic problem: $\min_{x \in X} \max_{d \in \mathcal{D}} f(x, d)$

$\longrightarrow$ Maintain a **finite subset** $\tilde{D}$ of $\mathcal{D}$ (a **"model"**)

GAME

1. **Implementor:** solve $\min_{x \in X} \max_{d \in \tilde{D}} f(x, d)$, with solution x^*
2. **Adversary:** solve $\max_{d \in \mathcal{D}} f(x^*, d)$, with solution $\tilde{d}$
3. **Add** $\tilde{d}$ to $\tilde{D}$, and go to 1.

General methodology:

Benders' decomposition (= cutting-plane algorithm)

Generic problem: $\min_{x \in X} \max_{d \in \mathcal{D}} f(x, d)$

$\longrightarrow$ Maintain a **finite subset** $\tilde{D}$ of $\mathcal{D}$ (a **"model"**)

GAME

1. **Implementor:** solve $\min_{x \in X} \max_{d \in \tilde{D}} f(x, d)$, with solution x^*
2. **Adversary:** solve $\max_{d \in \mathcal{D}} f(x^*, d)$, with solution $\tilde{d}$
3. **Add** $\tilde{d}$ to $\tilde{D}$, and go to 1.

General methodology:

Benders' decomposition (= cutting-plane algorithm)

Generic problem: $\min_{x \in X} \max_{d \in \mathcal{D}} f(x, d)$

$\longrightarrow$ Maintain a **finite subset** $\tilde{D}$ of $\mathcal{D}$ (a **"model"**)

GAME

1. **Implementor:** solve $\min_{x \in X} \max_{d \in \tilde{D}} f(x, d)$, with solution x^*
2. **Adversary:** solve $\max_{d \in \mathcal{D}} f(x^*, d)$, with solution $\tilde{d}$
3. **Add** $\tilde{d}$ to $\tilde{D}$, and go to 1.

General methodology:

Benders' decomposition (= cutting-plane algorithm)

Generic problem: $\min_{x \in X} \max_{d \in \mathcal{D}} f(x, d)$

$\longrightarrow$ Maintain a **finite subset** $\tilde{D}$ of $\mathcal{D}$ (a **"model"**)

GAME

1. **Implementor:** solve $\min_{x \in X} \max_{d \in \tilde{D}} f(x, d)$, with solution x^*
2. **Adversary:** solve $\max_{d \in \mathcal{D}} f(x^*, d)$, with solution $\tilde{d}$
3. **Add** $\tilde{d}$ to $\tilde{D}$, and go to 1.

General methodology:

Benders' decomposition (= cutting-plane algorithm)

Generic problem: $\min_{x \in X} \max_{d \in \mathcal{D}} f(x, d)$

$\longrightarrow$ Maintain a **finite subset** $\tilde{D}$ of $\mathcal{D}$ (a **"model"**)

GAME

1. **Implementor:** solve $\min_{x \in X} \max_{d \in \tilde{D}} f(x, d)$, with solution x^*
2. **Adversary:** solve $\max_{d \in \mathcal{D}} f(x^*, d)$, with solution $\tilde{d}$
3. **Add** $\tilde{d}$ to $\tilde{D}$, and go to 1.

Why this approach

- Decoupling of implementor and adversary yields considerably simpler, and smaller, problems
- Decoupling allows us to use more sophisticated uncertainty models
- If number of iterations is small, implementor's problem is a small "convex" problem
- Most progress will be achieved in initial iterations – permits "soft" termination criteria

Why this approach

- Decoupling of implementor and adversary yields considerably simpler, and smaller, problems
- Decoupling allows us to use more sophisticated uncertainty models
- If number of iterations is small, implementor's problem is a small "convex" problem
- Most progress will be achieved in initial iterations – permits "soft" termination criteria

Why this approach

- Decoupling of implementor and adversary yields considerably simpler, and smaller, problems
- Decoupling allows us to use more sophisticated uncertainty models
- If number of iterations is small, implementor's problem is a small "convex" problem
- Most progress will be achieved in initial iterations – permits "soft" termination criteria

Why this approach

- Decoupling of implementor and adversary yields considerably simpler, and smaller, problems
- Decoupling allows us to use more sophisticated uncertainty models
- If number of iterations is small, implementor's problem is a small "convex" problem
- Most progress will be achieved in initial iterations – permits "soft" termination criteria

Why this approach

- Decoupling of implementor and adversary yields considerably simpler, and smaller, problems
- Decoupling allows us to use more sophisticated uncertainty models
- If number of iterations is small, implementor's problem is a small "convex" problem
- Most progress will be achieved in initial iterations – permits "soft" termination criteria

Implementor's problem

A convex quadratic program

At iteration m, solve

$$\min \ \lambda x^T Q x \ - \ r$$

Subject to:

$$Ax \geq b$$

$$r \leq \mu_{(i)}^T x, \quad i = 1, \ldots, m$$

Here, $\mu_{(1)}, \ldots, \mu_{(m)}$ are given return vectors

Adversarial problem: A mixed-integer program

$x^* =$ given asset weights

$$\min \ \sum_j x_j^* \mu_j$$

Subject to:

$$\bar{\mu}_j (1 - \sum_i \gamma_{i-1} y_{ij}) \leq \mu_j \leq \bar{\mu}_j (1 - \sum_i \gamma_i y_{ij}) \quad \forall i \geq 1$$

$\sum_i y_{ij} \leq 1, \quad \forall j$ (each asset in at most one segment)

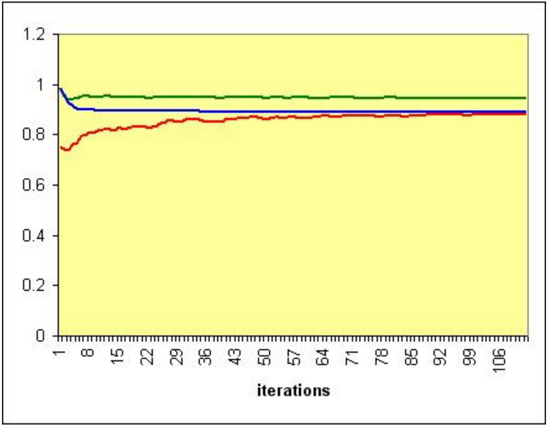
$n_i \leq \sum_j y_{ij} \leq N_i, \quad 1 \leq i \leq K$ (segment cardinalities)

$\sum_{j \in T_h} \mu_j \geq \Gamma_h \sum_{j \in T_h} \bar{\mu}_j, \quad 1 \leq h \leq H$ (tier ineqs.)

μ_j free, $y_{ij} = 0$ or 1, all i, j

2464 assets, 152 factors; total CPU time = 93.27 sec.

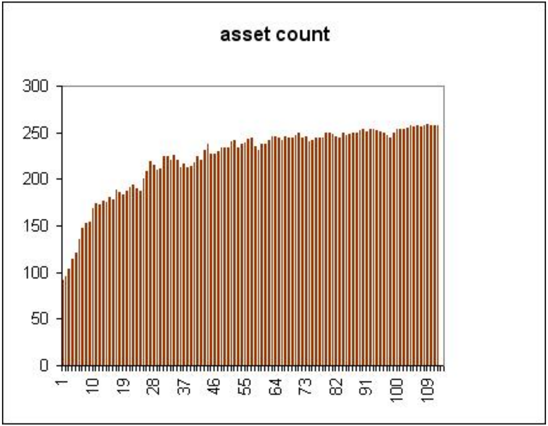
4 segments: (400, 0.01), (50, 0.05), (30, 0.2), (10, 0.3)
3 tiers: the three top deciles lose at most **10%** each



green = nominal return, blue = estimate, red = adversarial

Same run

2464 assets, 152 factors; total CPU time = 93.27 sec.
4 segments: (400, 0.01), (50, 0.05), (30, 0.20), (10, 0.30)
3 tiers: the three top deciles lose at most **10%** each



Summary

	columns	rows	iterations	time (sec.)	imp. time	adv. time
1	500	20	47	1.85	1.34	0.46
2	500	20	3	0.09	0.01	0.03
3	703	108	1	0.29	0.13	0.04
4	499	140	3	3.12	2.65	0.05
5	499	20	19	0.42	0.21	0.17
6	1338	81	7	0.45	0.17	0.08
7	2019	140	8	41.53	39.6	0.36
8	2443	153	2	12.32	9.91	0.07
9	2464	153	111	100.81	60.93	36.78

Why the adversarial problem is “easy”

($x^* =$ given asset weights)

$$\min \ \sum_j x_j^* \mu_j$$

Subject to:

$$\bar{\mu}_j (1 - \sum_i \gamma_{i-1} y_{ij}) \leq \mu_j \leq \bar{\mu}_j (1 - \sum_i \gamma_i y_{ij})$$

$\sum_i y_{ij} \leq 1, \ \forall j$ (each asset in at most one segment)

$n_i \leq \sum_j y_{ij} \leq N_i, \ \forall i$ (segment cardinalities)

$\sum_{j \in T_h} \mu_j \geq \Gamma_h \, (\sum_{j \in T_h} \bar{\mu}_j), \ \forall h$ (tier inequalities)

μ_j free, $y_{ij} = 0$ or 1, all i, j

Why the adversarial problem is “easy”

(K = no. of segments, H = no. of tiers)

Theorem. For every fixed K and H, and for every $\epsilon > 0$, there is an algorithm that finds a solution to the adversarial problem with optimality relative error $\leq \epsilon$, in time polynomial in ϵ^{-1} and n (= no. of assets).

The simplest case

$$\max \ \sum_j x_j^* \delta_j$$

Subject to:

$$\sum_j \delta_j \ \leq \ \Gamma$$

$$0 \ \leq \ \delta_j \ \leq \ u_j y_j, \ \ y_j \ = \ 0 \text{ or } 1, \ \ \text{all j}$$

$$\sum_j y_j \ \leq \ N$$

$\cdots$ a *cardinality constrained knapsack problem*
B. (1995), DeFarias and Nemhauser (2004)

What is the impact of the uncertainty model

All runs on the same data set with 1338 columns and 81 rows

- 1 segment: $(200, 0.5)$
  robust random return = **4.57**, **157** assets
- 2 segments: $(200, 0.25)$, $(100, 0.5)$
  robust random return = **4.57**, **186** assets
- 2 segments: $(200, 0.2)$, $(100, 0.6)$
  robust random return = **3.25**, **213** assets
- 2 segments: $(200, 0.1)$, $(100, 0.8)$
  robust random return = **1.50**, **256** assets
- 1 segment: $(100, 1.0)$
  robust random return = **1.24**, **281** assets

Ambiguous chance-constrained models

1. The implementor chooses a vector x^* of assets
2. The adversary chooses a *probability distribution* P for the returns vector
3. A random returns vector μ is drawn from P

$\rightarrow$ Implementor wants to choose x^* so as to minimize **value-at-risk** (conditional value at risk, etc.)

Erdogan and Iyengar (2004), Calafiore and Campi (2004)

$\rightarrow$ We want to model *correlated* errors in the returns

Ambiguous chance-constrained models

1. The implementor chooses a vector x^* of assets
2. The adversary chooses a *probability distribution* P for the returns vector
3. A random returns vector μ is drawn from P

$\longrightarrow$ Implementor wants to choose x^* so as to minimize **value-at-risk** (conditional value at risk, etc.)

Erdogan and Iyengar (2004), Calafiore and Campi (2004)

$\longrightarrow$ We want to model *correlated* errors in the returns

Ambiguous chance-constrained models

1. The implementor chooses a vector x^* of assets
2. The adversary chooses a *probability distribution* P for the returns vector
3. A random returns vector μ is drawn from P

$\longrightarrow$ Implementor wants to choose x^* so as to minimize **value-at-risk** (conditional value at risk, etc.)

Erdogan and Iyengar (2004), Calafiore and Campi (2004)

$\longrightarrow$ We want to model *correlated* errors in the returns

Ambiguous chance-constrained models

1. The implementor chooses a vector x^* of assets
2. The adversary chooses a *probability distribution* P for the returns vector
3. A random returns vector μ is drawn from P

$\longrightarrow$ Implementor wants to choose x^* so as to minimize **value-at-risk** (conditional value at risk, etc.)

Erdogan and Iyengar (2004), Calafiore and Campi (2004)

$\longrightarrow$ We want to model *correlated* errors in the returns

Ambiguous chance-constrained models

1. The implementor chooses a vector x^* of assets
2. The adversary chooses a *probability distribution* P for the returns vector
3. A random returns vector μ is drawn from P

$\longrightarrow$ Implementor wants to choose x^* so as to minimize **value-at-risk** (conditional value at risk, etc.)

Erdogan and Iyengar (2004), Calafiore and Campi (2004)

$\longrightarrow$ We want to model *correlated* errors in the returns

Ambiguous chance-constrained models

1. The implementor chooses a vector $\boldsymbol{x}^*$ of assets
2. The adversary chooses a *probability distribution* P for the returns vector
3. A random returns vector μ is drawn from P

$\longrightarrow$ Implementor wants to choose x^* so as to minimize **value-at-risk** (conditional value at risk, etc.)

Erdogan and Iyengar (2004), Calafiore and Campi (2004)

$\longrightarrow$ We want to model *correlated* errors in the returns

Uncertainty set

Given a vector x^* of assets, the adversary

1. Chooses a vector $w \in R^n$ (n = no. of assets) with $0 \leq w_j \leq 1$ for all j.
2. Chooses a random variable $0 \leq \delta \leq 1$

$\rightarrow$ Random return: $\mu_j = \bar{\mu}_j (1 - \delta w_j)$ ($\bar{\mu}$ = nominal returns).

Definition: Given reals ν and $0 \leq \theta \leq 1$ the *value-at-risk* of x^* is the real $\rho \geq 0$ such that

$$\text{Prob}(\nu - \mu^T x^* \geq \rho) \geq \theta$$

$\rightarrow$ The adversary wants to maximize VAR

Uncertainty set

Given a vector x^* of assets, the adversary

1. Chooses a vector $w \in R^n$ (n = no. of assets) with $0 \le w_j \le 1$ for all j.
2. Chooses a random variable $0 \le \delta \le 1$

$\rightarrow$ Random return: $\mu_j = \bar{\mu}_j (1 - \delta w_j)$ ($\bar{\mu}$ = nominal returns).

Definition: Given reals ν and $0 \le \theta \le 1$ the *value-at-risk* of x^* is the real $\rho \ge 0$ such that

$$\text{Prob}(\nu - \mu^T x^* \ge \rho) \ge \theta$$

$\rightarrow$ The adversary wants to maximize VAR

Uncertainty set

Given a vector x^* of assets, the adversary

1. Chooses a vector $w \in R^n$ (n = no. of assets) with $0 \leq w_j \leq 1$ for all j.
2. Chooses a random variable $0 \leq \delta \leq 1$

$\rightarrow$ Random return: $\mu_j = \bar{\mu}_j (1 - \delta w_j)$ ($\bar{\mu}$ = nominal returns).

Definition: Given reals ν and $0 \leq \theta \leq 1$ the *value-at-risk* of x^* is the real $\rho \geq 0$ such that

$$\text{Prob}(\nu - \mu^T x^* \geq \rho) \geq \theta$$

$\rightarrow$ The adversary wants to maximize VAR

Uncertainty set

Given a vector x^* of assets, the adversary

1. Chooses a vector $w \in R^n$ (n = no. of assets) with $0 \leq w_j \leq 1$ for all j.
2. Chooses a random variable $0 \leq \delta \leq 1$

$\longrightarrow$ Random return: $\mu_j = \bar{\mu}_j \, (1 - \delta w_j)$ ($\bar{\mu}$ = nominal returns).

Definition: Given reals ν and $0 \leq \theta \leq 1$ the *value-at-risk* of x^* is the real $\rho \geq 0$ such that

$$\text{Prob}(\nu - \mu^T x^* \geq \rho) \geq \theta$$

$\longrightarrow$ The adversary wants to maximize VAR

Uncertainty set

Given a vector x^* of assets, the adversary

1. Chooses a vector $w \in R^n$ (n = no. of assets) with $0 \leq w_j \leq 1$ for all j.
2. Chooses a random variable $0 \leq \delta \leq 1$

$\rightarrow$ Random return: $\mu_j = \bar{\mu}_j (1 - \delta w_j)$ ($\bar{\mu}$ = nominal returns).

Definition: Given reals ν and $0 \leq \theta \leq 1$ the *value-at-risk* of x^* is the real $\rho \geq 0$ such that

$$\text{Prob}(\nu - \mu^T x^* \geq \rho) \geq \theta$$

$\rightarrow$ The adversary wants to maximize VAR

Uncertainty set

Given a vector x^* of assets, the adversary

1. Chooses a vector $w \in R^n$ (n = no. of assets) with $0 \le w_j \le 1$ for all j.
2. Chooses a random variable $0 \le \delta \le 1$

$\longrightarrow$ Random return: $\mu_j = \bar{\mu}_j (1 - \delta w_j)$ ($\bar{\mu}$ = nominal returns).

Definition: Given reals ν and $0 \le \theta \le 1$ the *value-at-risk* of x^* is the real $\rho \ge 0$ such that

$$\text{Prob}(\nu - \mu^T x^* \ge \rho) \ge \theta$$

$\longrightarrow$ The adversary wants to maximize VAR

OR ...

Given a vector x^* of assets, the adversary

1. Chooses a vector $w \in R^n$ (n = no. of assets) with $0 \leq w_j \leq W$ for all j.
2. Chooses a random variable $0 \leq \delta \leq 1$

$\rightarrow$ Random return: $\mu_j = \bar{\mu}_j - \delta w_j$ ($\bar{\mu}$ = nominal returns).

Definition: Given reals ν and $0 \leq \theta \leq 1$ the *value-at-risk* of x^* is the real $\rho \geq 0$ such that

$$\text{Prob}(\nu - \mu^T x^* \geq \rho) \geq \theta$$

$\rightarrow$ The adversary wants to maximize VAR

OR ...

Given a vector x^* of assets, the adversary

1. Chooses a vector $w \in R^n$ (n = no. of assets) with $0 \leq w_j \leq W$ for all j.
2. Chooses a random variable $0 \leq \delta \leq 1$

$\longrightarrow$ Random return: $\mu_j = \bar{\mu}_j - \delta w_j$ ($\bar{\mu}$ = nominal returns).

Definition: Given reals ν and $0 \leq \theta \leq 1$ the *value-at-risk* of x^* is the real $\rho \geq 0$ such that

$$\text{Prob}(\nu - \mu^T x^* \geq \rho) \geq \theta$$

$\longrightarrow$ The adversary wants to maximize VAR

The classical factor model for returns

$$\mu = \bar{\mu} + \boldsymbol{V}^{\boldsymbol{T}}\boldsymbol{f} + \epsilon$$

where

- $\bar{\mu}$ = expected return,
- $\boldsymbol{V}$ = "factor exposure matrix",
- $\boldsymbol{f}$ = a bounded random variable,
- ϵ = residual errors

V is $r \times n$ with $r << n$.

$\rightarrow$ Random return$_j$ $= \bar{\mu}_j(1 - \delta w_j)$ where $0 \leq w_j \leq 1 \;\; \forall j$, and $0 \leq \delta \leq 1$ is a random variable.

A *discrete distribution:*

- We are given **fixed** values $0 = \delta_0 \leq \delta_2 \leq \ldots \leq \delta_K = 1$ example: $\delta_i = \frac{i}{K}$
- Adversary *chooses* $\pi_i = \text{Prob}(\delta = \delta_i)$, $0 \leq i \leq K$
- The π_i are *constrained*: we have fixed bounds, $\pi_i^l \leq \pi_i \leq \pi_i^u$ (and possibly other constraints)
- Tier constraints: for sets ("tiers") T_h of assets, $1 \leq h \leq H$, we require:

  $E(\delta \sum_{j \in T_h} w_j) \leq \Gamma_h$ (given)

  or, $(\sum_i \delta_i \pi_i) \sum_{j \in T_h} w_j \leq \Gamma_h$
- Cardinality constraint: $w_j > 0$ for at most N indices j

$\rightarrow$ Random return$_j$ $= \bar{\mu}_j(1 - \delta w_j)$ where $0 \leq w_j \leq 1 \ \forall j$, and $0 \leq \delta \leq 1$ is a random variable.

A *discrete distribution:*

- We are given **fixed** values $0 = \delta_0 \leq \delta_2 \leq \ldots \leq \delta_K = 1$
  example: $\delta_i = \frac{i}{K}$
- Adversary *chooses* $\pi_i = \text{Prob}(\delta = \delta_i)$, $0 \leq i \leq K$
- The π_i are *constrained*: we have fixed bounds, $\pi_i^l \leq \pi_i \leq \pi_i^u$
  (and possibly other constraints)
- Tier constraints: for sets ("tiers") T_h of assets, $1 \leq h \leq H$, we require:
  $E(\delta \sum_{j \in T_h} w_j) \leq \Gamma_h$ (given)

  or, $(\sum_i \delta_i \pi_i) \sum_{j \in T_h} w_j \leq \Gamma_h$
- Cardinality constraint: $w_j > 0$ for at most N indices j

$\rightarrow$ Random return$_j$ $= \bar{\mu}_j(1 - \delta w_j)$ where $0 \le w_j \le 1 \;\; \forall j$, and $0 \le \delta \le 1$ is a random variable.

A *discrete distribution:*

- We are given **fixed** values $0 = \delta_0 \le \delta_2 \le \ldots \le \delta_K = 1$ example: $\delta_i = \frac{i}{K}$
- Adversary *chooses* $\pi_i = \text{Prob}(\delta = \delta_i)$, $0 \le i \le K$
- The π_i are *constrained*: we have fixed bounds, $\pi_i^l \le \pi_i \le \pi_i^u$ (and possibly other constraints)
- Tier constraints: for sets ("tiers") T_h of assets, $1 \le h \le H$, we require:

  $E(\delta \sum_{j \in T_h} w_j) \le \Gamma_h$ (given)

  or, $(\sum_i \delta_i \pi_i) \sum_{j \in T_h} w_j \le \Gamma_h$
- Cardinality constraint: $w_j > 0$ for at most N indices j

$\rightarrow$ Random return$_j$ $= \bar{\mu}_j(1 - \delta w_j)$ where $0 \le w_j \le 1 \ \ \forall j$, and $0 \le \delta \le 1$ is a random variable.

A *discrete distribution:*

- We are given **fixed** values $0 = \delta_0 \le \delta_2 \le \ldots \le \delta_K = 1$ example: $\delta_i = \frac{i}{K}$
- Adversary *chooses* $\pi_i = \text{Prob}(\delta = \delta_i)$, $0 \le i \le K$
- The π_i are *constrained*: we have fixed bounds, $\pi_i^l \le \pi_i \le \pi_i^u$ (and possibly other constraints)
- Tier constraints: for sets ("tiers") T_h of assets, $1 \le h \le H$, we require:
  $E(\delta \sum_{j \in T_h} w_j) \le \Gamma_h$ (given)

  or, $(\sum_i \delta_i \pi_i) \sum_{j \in T_h} w_j \le \Gamma_h$
- Cardinality constraint: $w_j > 0$ for at most N indices j

$\rightarrow$ Random return$_j$ $= \bar{\mu}_j(1 - \delta w_j)$ where $0 \le w_j \le 1 \;\; \forall j$, and $0 \le \delta \le 1$ is a random variable.

A *discrete distribution:*

- We are given **fixed** values $0 = \delta_0 \le \delta_2 \le \ldots \le \delta_K = 1$ example: $\delta_i = \frac{i}{K}$
- Adversary *chooses* $\pi_i = \text{Prob}(\delta = \delta_i)$, $0 \le i \le K$
- The π_i are *constrained*: we have fixed bounds, $\pi_i^l \le \pi_i \le \pi_i^u$ (and possibly other constraints)
- Tier constraints: for sets ("tiers") T_h of assets, $1 \le h \le H$, we require:

  $E(\delta \sum_{j \in T_h} w_j) \le \Gamma_h$ (given)

  or, $(\sum_i \delta_i \pi_i) \sum_{j \in T_h} w_j \le \Gamma_h$
- Cardinality constraint: $w_j > 0$ for at most N indices j

→ Random return$_j$ $= \bar{\mu}_j(1 - \delta w_j)$ where $0 \leq w_j \leq 1 \ \forall j$, and $0 \leq \delta \leq 1$ is a random variable.

A *discrete distribution:*

- We are given **fixed** values $0 = \delta_0 \leq \delta_2 \leq \ldots \leq \delta_K = 1$
  example: $\delta_i = \frac{i}{K}$
- Adversary *chooses* $\pi_i = \text{Prob}(\delta = \delta_i)$, $0 \leq i \leq K$
- The π_i are *constrained*: we have fixed bounds, $\pi_i^l \leq \pi_i \leq \pi_i^u$ (and possibly other constraints)
- Tier constraints: for sets (“tiers”) T_h of assets, $1 \leq h \leq H$, we require:
  $E(\delta \sum_{j \in T_h} w_j) \leq \Gamma_h$ (given)

  or, $(\sum_i \delta_i \pi_i) \sum_{j \in T_h} w_j \leq \Gamma_h$

- Cardinality constraint: $w_j > 0$ for at most N indices j

$\rightarrow$ Random return$_j$ $= \bar{\mu}_j(1 - \delta w_j)$ where $0 \le w_j \le 1 \;\; \forall j$, and $0 \le \delta \le 1$ is a random variable.

A *discrete distribution:*

- We are given **fixed** values $0 = \delta_0 \le \delta_2 \le \ldots \le \delta_K = 1$
  example: $\delta_i = \frac{i}{K}$
- Adversary *chooses* $\pi_i = \text{Prob}(\delta = \delta_i)$, $0 \le i \le K$
- The π_i are *constrained*: we have fixed bounds, $\pi_i^l \le \pi_i \le \pi_i^u$
  (and possibly other constraints)
- Tier constraints: for sets ("tiers") T_h of assets, $1 \le h \le H$, we require:

  $E(\delta \sum_{j \in T_h} w_j) \le \Gamma_h$ (given)

  or, $(\sum_i \delta_i \pi_i) \sum_{j \in T_h} w_j \le \Gamma_h$
- Cardinality constraint: $w_j > 0$ for at most N indices j

$\longrightarrow$ Random return$_j$ $= \bar{\mu}_j(1 - \delta w_j)$ where $0 \leq w_j \leq 1 \;\; \forall j$, and $0 \leq \delta \leq 1$ is a random variable.

A *discrete distribution:*

- We are given **fixed** values $0 = \delta_0 \leq \delta_2 \leq \ldots \leq \delta_K = 1$ example: $\delta_i = \frac{i}{K}$
- Adversary *chooses* $\pi_i = \text{Prob}(\delta = \delta_i)$, $0 \leq i \leq K$
- The π_i are *constrained*: we have fixed bounds, $\pi_i^l \leq \pi_i \leq \pi_i^u$ (and possibly other constraints)
- Tier constraints: for sets ("tiers") T_h of assets, $1 \leq h \leq H$, we require:

  $E(\delta \sum_{j \in T_h} w_j) \leq \Gamma_h$ (given)

  or, $(\sum_i \delta_i \pi_i) \sum_{j \in T_h} w_j \leq \Gamma_h$
- Cardinality constraint: $w_j > 0$ for at most N indices j

The adversarial problem is "easy"

K = no. of points in discrete distribution, H = no. of tiers

Theorem

- Without the cardinality constraint, for each fixed K and H the adversarial problem can be solved as a polynomial number of linear programs.
- With the cardinality constraint, for each fixed K and H the adversarial problem can be solved as a polynomial number of knapsack problems.

The adversarial problem is “easy”

K = no. of points in discrete distribution, H = no. of tiers

Theorem

- Without the cardinality constraint, for each fixed K and H the adversarial problem can be solved as a polynomial number of linear programs.
- With the cardinality constraint, for each fixed K and H the adversarial problem can be solved as a polynomial number of knapsack problems.

The adversarial problem is “easy”

K = no. of points in discrete distribution, H = no. of tiers

Theorem

- Without the cardinality constraint, for each fixed K and H the adversarial problem can be solved as a polynomial number of linear programs.
- With the cardinality constraint, for each fixed K and H the adversarial problem can be solved as a polynomial number of knapsack problems.

Adversarial problem as an MIP

Recall: random return$_j$ $\mu_j = \bar{\mu}_j(1 - \delta w_j)$
where $\delta = \delta_i$ (given) with probability π_i (chosen by adversary),
$0 \leq \delta_0 \leq \delta_1 \leq \ldots \leq \delta_K = 1$ and $0 \leq w$

$\min_{\pi,w,V} \ \min_{1 \leq i \leq k} V_i$

Subject to

$0 \leq w_j \leq 1$, all j, $\pi_i^l \leq \pi_i \leq \pi_i^u$, all i,
$\sum_i \pi_i = 1$,

$V_i = \sum_j \bar{\mu}_j(1 - \delta_i w_j)x_j^*$, if $\pi_i + \pi_{i+1} + \ldots + \pi_K \geq 1 - \theta$
$V_i = M$ (large), otherwise

$(\sum_i \delta_i \pi_i) \sum_{j \in T_h} w_j \leq \Gamma_h$, for each tier h

Adversarial problem as an MIP

Recall: random return$_j$ $\mu_j = \bar{\mu}_j(1 - \delta w_j)$
where $\delta = \delta_i$ (given) with probability π_i (chosen by adversary),
$0 \le \delta_0 \le \delta_1 \le \ldots \le \delta_K = 1$ and $0 \le w$

$$\min_{\pi,w,V} \ \min_{1 \le i \le k} V_i$$

Subject to

$$0 \le w_j \le 1, \ \text{all j}, \ \pi_i^l \le \pi_i \le \pi_i^u, \ \text{all i},$$
$$\sum_i \pi_i = 1,$$

$$V_i = \sum_j \bar{\mu}_j(1 - \delta_i w_j)x_j^*, \quad \text{if } \pi_i + \pi_{i+1} + \ldots + \pi_K \ge 1 - \theta$$
$$V_i = M \ (large), \quad \text{otherwise}$$

$$\left(\sum_i \delta_i \pi_i\right) \sum_{j \in T_h} w_j \le \Gamma_h, \quad \text{for each tier h}$$

Adversarial problem as an MIP

Recall: random return$_j$ $\mu_j = \bar{\mu}_j(1 - \delta w_j)$
where $\delta = \delta_i$ (given) with probability π_i (chosen by adversary),
$0 \leq \delta_0 \leq \delta_1 \leq \ldots \leq \delta_K = 1$ and $0 \leq w$

$$\min_{\pi, w, V} \ \min_{1 \leq i \leq k} V_i$$

Subject to

$0 \leq w_j \leq 1$, all j, $\pi_i^l \leq \pi_i \leq \pi_i^u$, all i,
$\sum_i \pi_i = 1$,

$V_i = \sum_j \bar{\mu}_j (1 - \delta_i w_j) x_j^*$, if $\pi_i + \pi_{i+1} + \ldots + \pi_K \geq 1 - \theta$
$V_i = M$ (*large*), otherwise

$(\sum_i \delta_i \pi_i) \sum_{j \in T_h} w_j \leq \Gamma_h$, for each tier h

Adversarial problem as an MIP

Recall: random return$_j$ $\mu_j = \bar{\mu}_j(1 - \delta w_j)$
where $\delta = \delta_i$ (given) with probability π_i (chosen by adversary),
$0 \le \delta_0 \le \delta_1 \le \ldots \le \delta_K = 1$ and $0 \le w$

$$\min_{\pi, w, V} \ \min_{1 \le i \le k} V_i$$

Subject to

$$0 \le w_j \le 1, \ \text{all j}, \ \pi_i^l \le \pi_i \le \pi_i^u, \ \text{all i},$$
$$\sum_i \pi_i = 1,$$

$$V_i = \sum_j \bar{\mu}_j(1 - \delta_i w_j)x_j^*, \quad \text{if } \pi_i + \pi_{i+1} + \ldots + \pi_K \ge 1 - \theta$$
$$V_i = M \ (large), \quad \text{otherwise}$$

$$\left(\sum_i \delta_i \pi_i\right) \sum_{j \in T_h} w_j \le \Gamma_h, \quad \text{for each tier h}$$

Approximation

$\left(\sum_i \delta_i \pi_i\right) \sum_{j \in T_h} w_j \leq \Gamma_h,$ for each tier h $(*)$

Let $N > 0$ be an integer. For $1 \leq k \leq N$, write

$\frac{k}{N} \sum_{j \in T_h} w_j \leq \Gamma_h + M\,(1 - z_{hk}),$ where

$z_{hk} = 1$ if $\frac{k-1}{N} < \sum_i \delta_i \pi_i \leq \frac{k}{N}$

$z_{hk} = 0$ otherwise

$\sum_k z_{hk} = 1$

and M is large

Lemma. Under reasonable conditions, replacing $(*)$ with this system changes the value of the problem by at most a factor of $(1 + \frac{1}{N})$

Approximation

$$\left(\sum_i \delta_i \pi_i\right) \sum_{j \in T_h} w_j \leq \Gamma_h, \quad \text{for each tier h} \qquad (*)$$

Let $N > 0$ be an integer. For $1 \leq k \leq N$, write

$\frac{k}{N} \sum_{j \in T_h} w_j \leq \Gamma_h + M\,(1 - z_{hk}),$ where

$z_{hk} = 1$ if $\frac{k-1}{N} < \sum_i \delta_i \pi_i \leq \frac{k}{N}$

$z_{hk} = 0$ otherwise

$\sum_k z_{hk} = 1$

and M is large

Lemma. Under reasonable conditions, replacing $(*)$ with this system changes the value of the problem by at most a factor of $(1 + \frac{1}{N})$

Approximation

$$\left(\sum_i \delta_i \pi_i\right) \sum_{j \in T_h} w_j \leq \Gamma_h, \quad \text{for each tier h} \qquad (*)$$

Let $N > 0$ be an integer. For $1 \leq k \leq N$, write

$\frac{k}{N} \sum_{j \in T_h} w_j \leq \Gamma_h + M\,(1 - z_{hk}),$ where

$z_{hk} = 1$ if $\frac{k-1}{N} < \sum_i \delta_i \pi_i \leq \frac{k}{N}$

$z_{hk} = 0$ otherwise

$\sum_k z_{hk} = 1$

and M is large

Lemma. Under reasonable conditions, replacing $(*)$ with this system changes the value of the problem by at most a factor of $(1 + \frac{1}{N})$

Implementor's problem

Find a near-optimal solution with minimum value-at-risk

Nominal problem:

$$v^* = \min_x \lambda x^T Q x - \mu^T x$$

Subject to:

$$Ax \geq b$$

Implementor's problem

Find a near-optimal solution with minimum value-at-risk

Nominal problem:

$$v^* = \min_x \lambda x^T Q x - \mu^T x$$

Subject to:

$$Ax \geq b$$

Implementor's problem

Find a near-optimal solution with minimum value-at-risk

Given asset weights $\boldsymbol{x}$, we have:

- value-at-risk $\geq \rho$, if the adversary can produce a return vector μ with

  $$\text{Prob}(\nu - \mu^T x \geq \rho) \geq \theta$$

where ν is a fixed reference value.

Implementor's problem

Find a near-optimal solution with minimum value-at-risk

Implementor's problem at iteration r:

$$\min \ V$$

Subject to:

$$\lambda x^T Q x \ - \ \mu^T x \ \leq \ (1+\epsilon) v^*$$

$$Ax \geq b$$

$$V \ \geq \ \nu \ - \ \sum_j \bar{\mu}_j \left(1 - \delta_{i(t)} w_j^{(t)}\right) x_j, \qquad t = 1, 2, \ldots, r-1$$

Here, $\delta_{i(t)}$ and $w^{(t)}$ are the adversary's output at iteration $t < r$.

Implementor's problem

Find a near-optimal solution with minimum value-at-risk

Implementor's problem at iteration r:

$$\min\ V$$

Subject to:

$$\lambda x^T Q x \ - \ \mu^T x \ \leq \ (1+\epsilon) v^*$$

$$Ax \geq b$$

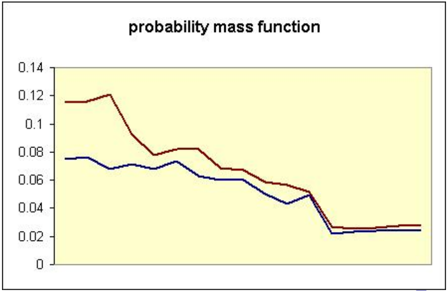
$$V \ \geq \ \nu \ - \ \sum_j \bar{\mu}_j \left(1 - \delta_{i(t)} w_j^{(t)}\right) x_j, \qquad t = 1, 2, \ldots, r-1$$

Here, $\delta_{i(t)}$ and $w^{(t)}$ are the adversary's output at iteration $t < r$.

First set of experiments

1338 assets, 41 factors, 81 rows

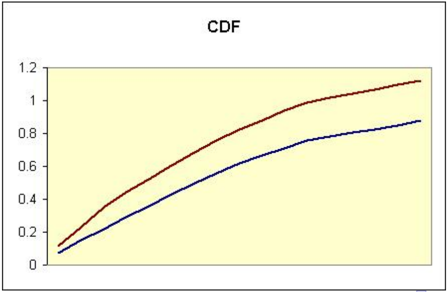
- problem: find a "near optimal" solution with minimum value-at-risk, for a given threshold probability θ
- experiment: investigate different values of θ
- "near optimal": want solutions that are at most 1% more expensive than optimal
- random variable δ:



First set of experiments

1338 assets, 41 factors, 81 rows

- problem: find a "near optimal" solution with minimum value-at-risk, for a given threshold probability θ
- experiment: investigate different values of θ
- "near optimal": want solutions that are at most 1% more expensive than optimal
- random variable δ:

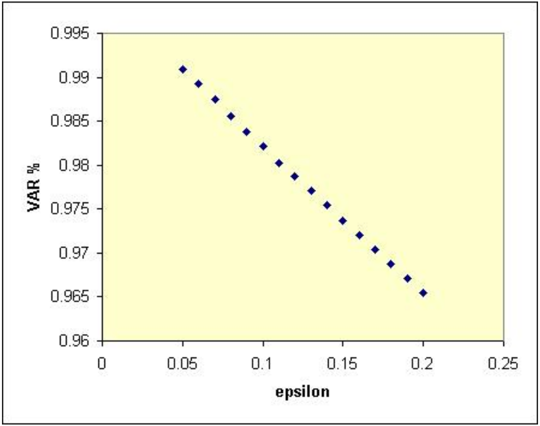


1338 assets, 41 factors, 81 rows, $\leq 1\%$ suboptimality

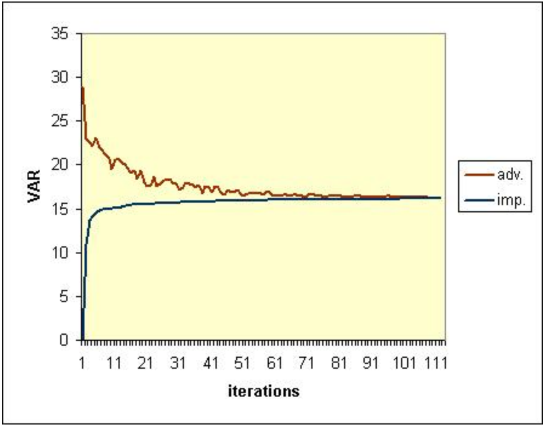
θ	time (sec.)	iters.	VAR	VAR as %
.89	1.18	2	3.43131	71.50
.90	1.42	3	3.74498	78.04
.91	1.42	3	3.74498	78.04
.92	3.47	11	4.05669	84.53
.93	6.35	29	4.05721	84.54
.94	6.37	20	4.05721	84.54
.95	26.59	51	4.35481	90.74
.96	26.25	51	4.35481	90.74
.97	26.20	51	4.35481	90.74
.98	33.07	58	4.63938	96.67
.99	33.11	58	4.63938	96.67

Second set of experiments

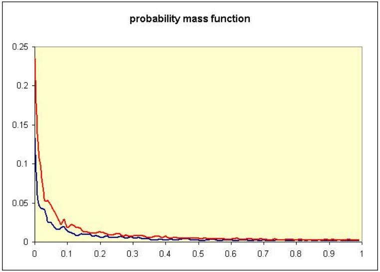
Fix $\theta = 0.90$ but vary suboptimality criterion



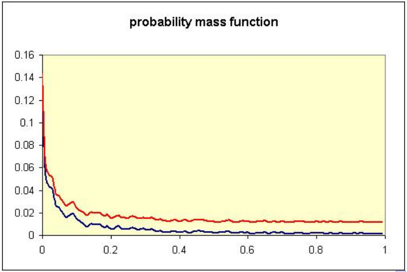
Typical convergence behavior



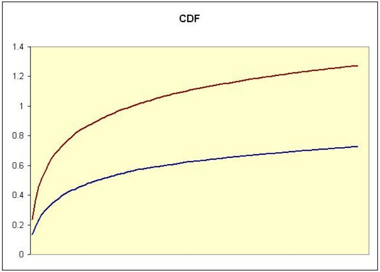
- Heavy tail, proportional error (100 points):



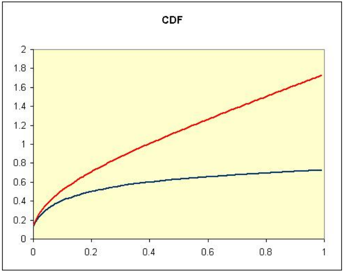
- Heavy tail, constant error (100 points):



- Heavy tail, proportional error (100 points):



- Heavy tail, constant error (100 points):



	Proportional		Constant	
θ	Time	Its	Time	Its
.85	4.79	8	11.84	11
.86	2.32	3	8.27	8
.87	6.40	10	9.55	11
.88	4.34	4	18.10	24
.89	8.00	14	5.85	6
.90	2.58	4	13.54	20
.91	4.79	9	16.31	23
.92	7.99	15	13.13	22
.93	13.43	27	22.47	40
.94	10.04	15	21.99	40
.95	9.59	16	11.90	25
.96	6.63	17	29.89	54
.97	48.43	110	16.45	35
.98	20.25	53	20.25	45
.99	22.02	52	21.89	47

A difficult case

- 2464 columns, 152 factors, 3 tiers
- time = 6191 seconds
- 258 iterations
- implementor time = 6123 seconds,
  adversarial time = 20 seconds

A difficult case

- 2464 columns, 152 factors, 3 tiers
- time = 6191 seconds
- 258 iterations
- implementor time = 6123 seconds,
  adversarial time = 20 seconds

A difficult case

- 2464 columns, 152 factors, 3 tiers
- time = 6191 seconds
- 258 iterations
- implementor time = 6123 seconds,
  adversarial time = 20 seconds

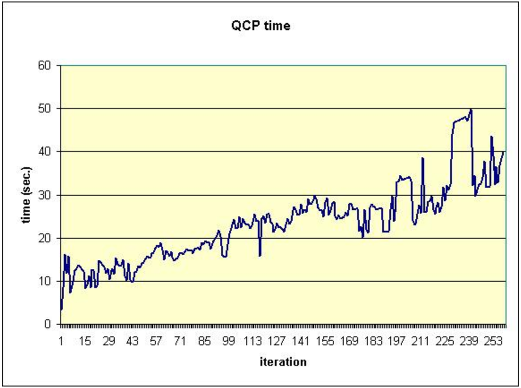
A difficult case

- 2464 columns, 152 factors, 3 tiers
- time = 6191 seconds
- 258 iterations
- implementor time = 6123 seconds, adversarial time = 20 seconds

A difficult case

- 2464 columns, 152 factors, 3 tiers
- time = 6191 seconds
- 258 iterations
- implementor time = 6123 seconds,
  adversarial time = 20 seconds

Implementor runtime



Implementor's problem at iteration r

min V

Subject to:

$$\lambda x^T Q x \; - \; \mu^T x \;\; \leq \;\; (1 + \epsilon) v^*$$

$$Ax \geq b$$

$$V \;\geq\; \nu \;-\; \sum_j \bar{\mu}_j \left(1 - \delta_{i(t)} w_j^{(t)}\right) x_j, \qquad t = 1, 2, \ldots, r-1$$

Here, $\delta_{i(t)}$ and $w^{(t)}$ are the adversary's output at iteration $t < r$.

Implementor's problem at iteration r

Approximate version

min V

Subject to:

$$2\lambda\, x_{(k)}^T Q x \;-\; \lambda\, x_{(k)}^T Q x_{(k)} \;-\; \mu^T x \;\leq\; (1+\epsilon)v^*, \quad \forall k < r$$

$$Ax \geq b$$

$$V \;\geq\; \nu \;-\; \sum_j \bar{\mu}_j \left(1 - \delta_{i(k)} w_j^{(k)}\right) x_j, \quad \forall\, k < r$$

Here, $\delta_{i(k)}$ and $w^{(k)}$ are the adversary's output at iteration $k < r$, and $x_{(k)}$ is the implementor's output at iteration k.

Does it work?

- Before: 258 iterations, **6191** seconds
- Linearized: 1776 iterations, **3969** seconds

Does it work?

- Before: 258 iterations, **6191** seconds
- Linearized: 1776 iterations, **3969** seconds

Averaging

- $x_{(k)}$ is the implementor's output at iteration k.
- Define $y_{(1)} = x_{(1)}$
- For $k > 1$, $y_{(k)} = \lambda x_{(k)} + (1 - \lambda) y_{(k-1)}$, $0 \leq \lambda \leq 1$
- Input $y_{(k)}$ to the adversary
- Old ideas, also Nesterov, Nemirovsky (2003)

Averaging

- $x_{(k)}$ is the implementor's output at iteration k.
- Define $y_{(1)} = x_{(1)}$
- For $k > 1$, $y_{(k)} = \lambda x_{(k)} + (1 - \lambda) y_{(k-1)}$, $0 \leq \lambda \leq 1$
- Input $y_{(k)}$ to the adversary
- Old ideas, also Nesterov, Nemirovsky (2003)

Averaging

- $x_{(k)}$ is the implementor's output at iteration k.
- Define $y_{(1)} = x_{(1)}$
- For $k > 1$, $y_{(k)} = \lambda x_{(k)} + (1 - \lambda) y_{(k-1)}$, $0 \leq \lambda \leq 1$
- Input $y_{(k)}$ to the adversary
- Old ideas, also Nesterov, Nemirovsky (2003)

Averaging

- $x_{(k)}$ is the implementor's output at iteration k.
- Define $y_{(1)} = x_{(1)}$
- For $k > 1$, $y_{(k)} = \lambda\, x_{(k)} + (1 - \lambda)\, y_{(k-1)}$, $0 \le \lambda \le 1$
- Input $y_{(k)}$ to the adversary
- Old ideas, also Nesterov, Nemirovsky (2003)

Averaging

- $\mathbf{x}_{(k)}$ is the implementor's output at iteration k.
- Define $\mathbf{y}_{(1)} = \mathbf{x}_{(1)}$
- For $k > 1$, $\mathbf{y}_{(k)} = \lambda\, \mathbf{x}_{(k)} + (1 - \lambda)\, \mathbf{y}_{(k-1)}$,
  $0 \leq \lambda \leq 1$
- Input $\mathbf{y}_{(k)}$ to the adversary
- Old ideas, also Nesterov, Nemirovsky (2003)

Does it work?

- Default: 258 iterations, **6191** seconds
- Linearized: 1776 iterations, **3969** seconds
- Averaging plus Linearized: 860 iterations, **530** seconds

Does it work?

- Default: 258 iterations, **6191** seconds
- Linearized: 1776 iterations, **3969** seconds
- Averaging plus Linearized: 860 iterations, **530** seconds

Does it work?

- Default: 258 iterations, **6191** seconds
- Linearized: 1776 iterations, **3969** seconds
- Averaging plus Linearized: 860 iterations, **530** seconds

Other robust models

- Min-max expected *loss* with orthogonal missing factor

  Random return = $\bar{\mu} \bullet (\mathbf{1} - \delta \boldsymbol{w})$ where $-\mathbf{1} \leq \boldsymbol{w_j} \leq \mathbf{1} \;\; \forall \boldsymbol{j}$, and $\mathbf{0} \leq \delta \leq \mathbf{1}$ is a random variable.

  normalization constraints, e.g. $\sum_j w_j = 0$

- errors in covariance matrix Q

  robust problem: $\rightarrow \min_x \; \max_{Q \in \mathcal{Q}} \; \lambda x^T Q x - \mu^T x$

Other robust models

- Min-max expected *loss* with orthogonal missing factor

  Random return = $\bar{\mu} \bullet (\mathbf{1} - \delta \boldsymbol{w})$ where $-\mathbf{1} \leq \boldsymbol{w_j} \leq \mathbf{1} \ \forall \boldsymbol{j}$, and $\mathbf{0} \leq \delta \leq \mathbf{1}$ is a random variable.

  normalization constraints, e.g. $\sum_j \boldsymbol{w_j} = \mathbf{0}$

- errors in covariance matrix $\boldsymbol{Q}$

  robust problem: $\rightarrow \min_x \ \max_{Q \in \mathcal{Q}} \ \lambda x^T Q x \ - \ \mu^T x$

Other robust models

- Min-max expected *loss* with orthogonal missing factor

  Random return = $\bar{\mu} \bullet (\mathbf{1} - \delta \boldsymbol{w})$ where $-\mathbf{1} \leq \boldsymbol{w_j} \leq \mathbf{1} \;\; \forall \boldsymbol{j}$, and $\mathbf{0} \leq \delta \leq \mathbf{1}$ is a random variable.

  normalization constraints, e.g. $\sum_{\boldsymbol{j}} \boldsymbol{w_j} = \mathbf{0}$

- errors in covariance matrix $\boldsymbol{Q}$

  robust problem: $\rightarrow \mathbf{min}_{\boldsymbol{x}} \;\; \mathbf{max}_{\boldsymbol{Q} \in \mathcal{Q}} \;\; \lambda \boldsymbol{x}^{\boldsymbol{T}} \boldsymbol{Q} \boldsymbol{x} \; - \; \mu^{\boldsymbol{T}} \boldsymbol{x}$

On-going work: a provably good version of Benders' algorithm