More approximation algorithms for stochastic programming programs

David B. Shmoys

Joint work with Chaitanya Swamy
Cal Tech

Stochastic Optimization

- Way of modeling uncertainty.
- Exact data is unavailable or expensive – data is uncertain, specified by a probability distribution.

  Want to make the best decisions given this uncertainty in the data.

- Dates back to 1950's and the work of Dantzig.
- Applications in logistics, transportation models, financial instruments, network design, production planning, …

Two-Stage Recourse Model

Given : Probability distribution over inputs.

Stage I : Make some advance decisions – plan ahead or hedge against uncertainty.

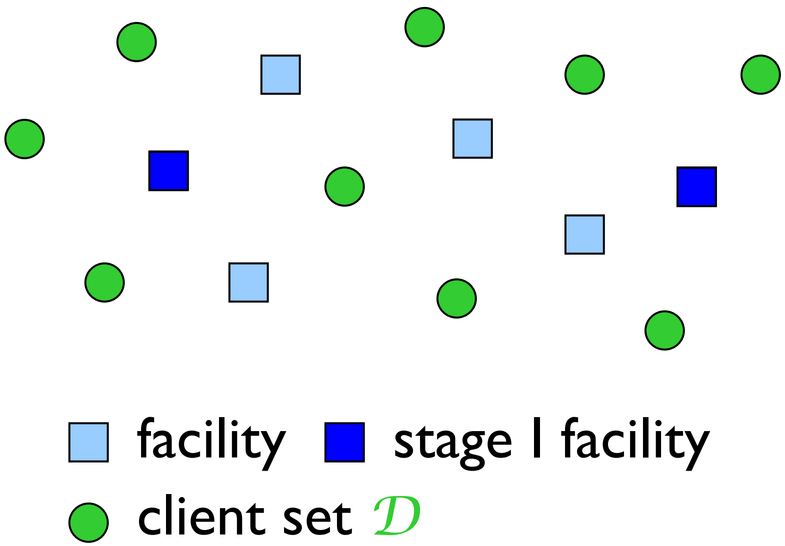
Observe the actual input scenario.

Stage II : Take recourse. Can augment earlier solution paying a recourse cost.

Choose stage I decisions to minimize

(stage I cost) + (expected stage II recourse cost).

2-Stage Stochastic Facility Location

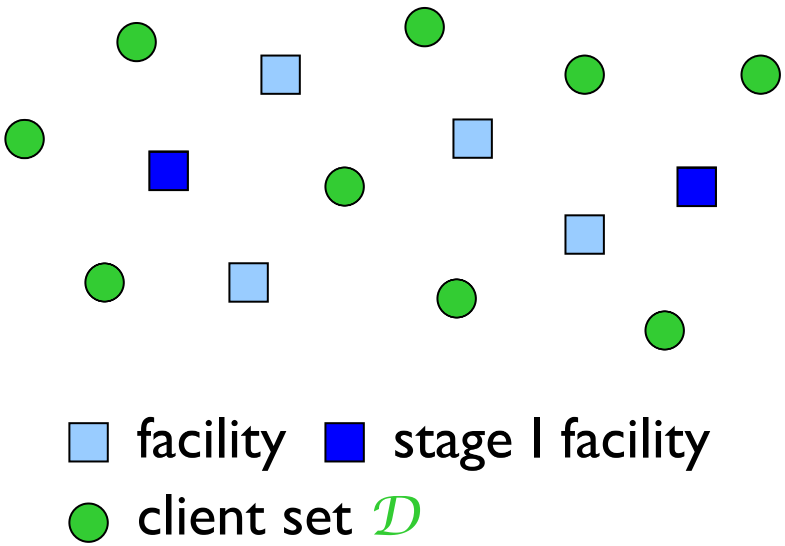


Distribution over clients gives the set of clients to serve.

Stage I: Open some facilities in advance; pay cost f_i for facility i.

Stage I cost = $\sum_{(i \text{ opened})} f_i$.

2-Stage Stochastic Facility Location



Distribution over clients gives the set of clients to serve.

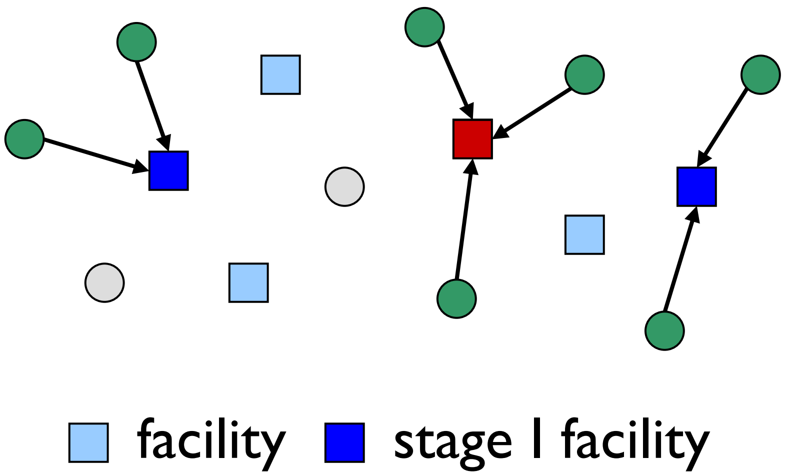
Stage I: Open some facilities in advance; pay cost f_i for facility i.

Stage I cost = $\sum_{(i \text{ opened})} f_i$.

How is the probability distribution on clients specified?

- A short (polynomial) list of possibile scenarios;
- Independent probabilities that each client exists;
- A black box that can be sampled.

2-Stage Stochastic Facility Location



Distribution over clients gives the set of clients to serve.

Stage I: Open some facilities in advance; pay cost f_i for facility i.

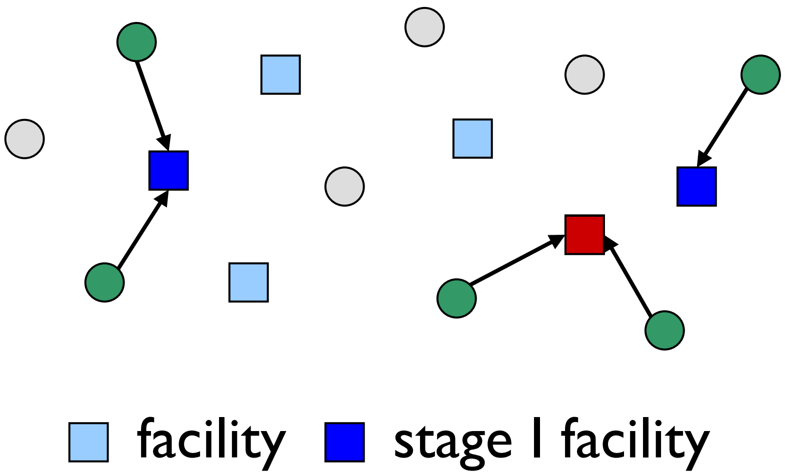
Stage I cost = $\sum_{(i \text{ opened})} f_i$.

Actual scenario A = { ● clients to serve}, materializes.

Stage II: Can open more facilities to serve clients in A; pay cost f_i^A to open facility i. Assign clients in A to facilities.

Stage II cost = $\sum_{\substack{i \text{ opened in} \\ \text{scenario } A}} f_i^A$ + (cost of serving clients in A).

2-Stage Stochastic Facility Location



Distribution over clients gives the set of clients to serve.

Stage I: Open some facilities in advance; pay cost f_i for facility i.

Stage I cost = $\sum_{(i \text{ opened})} f_i$.

Actual scenario A = { ● clients to serve}, materializes.

Stage II: Can open more facilities to serve clients in A; pay cost f_i^A to open facility i. Assign clients in A to facilities.

Stage II cost = $\sum_{\substack{i \text{ opened in} \\ \text{scenario } A}} f_i^A$ + (cost of serving clients in A).

Want to decide which facilities to open in stage I.

Goal: Minimize Total Cost =

$$(\text{stage I cost}) + \mathbf{E}_{A \subseteq \mathcal{D}}[\text{stage II cost for } A].$$

We want to prove a worst-case guarantee.

Give an algorithm that "works well" on any instance, and for any probability distribution.

A is an α-approximation algorithm if -

- A runs in polynomial time;
- $A(I) \leq \alpha.OPT(I)$ on all instances I.

α is called the approximation ratio of A.

What is new here?

- Previous “black box” results all assumed that, for each element of the solution (facility opened, edge in Steiner tree) the costs in the two stages are proportional: (stage II cost) = λ(stage I cost).
- Note: λ in this talk is the same as σ in previous one
- We allow independent stage I and stage II costs
- Previous results rely on structure of underlying stochastic LPs; we will provide algorithms to (approximately) solve those LPs

Our Results

- Give the first approximation algorithms for 2-stage discrete stochastic problems
  - black-box model
  - no assumptions on costs.
- Give a fully polynomial randomized approximation scheme for a large class of 2-stage stochastic linear programs (contrast to Kleywegt, Shapiro, & Homem-DeMillo 01, Dyer, Kannan & Stougie 02, Nesterov & Vial 00)
- Give another way to "reduce" stochastic optimization problems to their deterministic versions.

Stochastic Set Cover (SSC)

Universe $U = \{e_1, \ldots, e_n\}$, subsets $S_1, S_2, \ldots, S_m \subseteq U$, set S has weight w_S.

Deterministic problem: Pick a minimum weight collection of sets that covers each element.

Stochastic version: Set of elements to be covered is given by a probability distribution.

- choose some sets initially paying w_S for set S
- subset $A \subseteq U$ to be covered is revealed
- can pick additional sets paying w_S^A for set S.

Minimize (w-cost of sets picked in stage I) +

$\mathbf{E}_{A \subseteq U}$ [w^A-cost of new sets picked for scenario A].

An LP formulation

For simplicity, consider $w_S^A = W_S$ for every scenario A.

p_A : probability of scenario $A \subseteq U$.

x_S : indicates if set S is picked in stage I.

$y_{A,S}$: indicates if set S is picked in scenario A.

Minimize $\sum_S \omega_S x_S + \sum_{A \subseteq U} p_A \sum_S W_S y_{A,S}$

subject to,

$$\sum_{S:e \in S} x_S + \sum_{S:e \in S} y_{A,S} \geq 1 \quad \text{for each } A \subseteq U,\ e \in A$$

$$x_S,\ y_{A,S} \geq 0 \quad \text{for each } S, A.$$

Exponential number of variables and exponential number of constraints.

A Rounding Theorem

Stochastic Problem: LP can be solved in polynomial time.

Example: polynomial scenario setting

Deterministic problem: α-approximation algorithm A with respect to the LP relaxation, $\mathcal{A}(I) \leq \alpha \cdot$LP-OPT(I) for each I.

Example: "the greedy algorithm" for set cover is a log n-approximation algorithm w.r.t. LP relaxation.

Theorem: Can use such an α-approx. algorithm to get a 2α-approximation algorithm for stochastic set cover.

Rounding the LP

Assume LP can be solved in polynomial time.

Suppose we have an α-approximation algorithm wrt. the LP relaxation for the deterministic problem.

Let (x,y) : optimal solution with cost LP-OPT.

$$\sum_{S:e\in S} x_S + \sum_{S:e\in S} y_{A,S} \geq 1 \qquad \text{for each } A \subseteq U,\ e\in A$$

$\Rightarrow$ for every element e, either

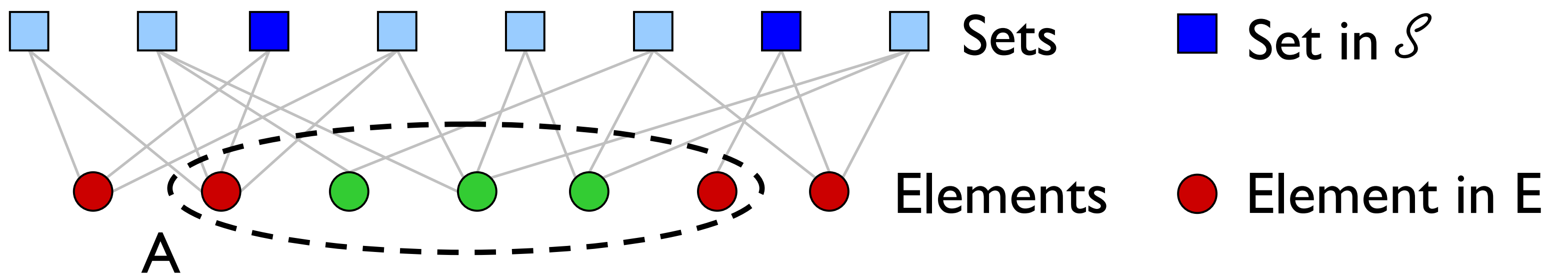
$\sum_{S:e\in S} x_S \geq 1/2$ OR in each scenario $A : e\in A$, $\sum_{S:e\in S} y_{A,S} \geq 1/2$.

Let $E = \{e : \sum_{S:e\in S} x_S \geq 1/2\}$.

So $(2x)$ is a fractional set cover for the set E $\Rightarrow$ can "round" to get an integer set cover $\mathcal{S}$ for E of cost $\sum_{S\in\mathcal{S}} \omega_S \leq \alpha(\sum_S 2\omega_S x_S)$.

$\mathcal{S}$ is the first stage decision.

Rounding (contd.)



Consider any scenario A. Elements in $A \cap E$ are covered.

For every $e \in$ A¥E, it must be that $\sum_{S:e \in S} y_{A,S} \geq ½$.

So $(2y^A)$ is a fractional set cover for A¥E $\Rightarrow$ can round to get a set cover of W-cost $\leq \alpha(\sum_S 2W_S y_{A,S})$.

Using this to augment $\mathcal{S}$ in scenario A, expected cost

$$\leq \sum_{S \in \mathcal{S}} \omega_S + 2\alpha \cdot \sum_{A \subseteq U} p_A \left(\sum_S W_S y_{A,S}\right) \leq 2\alpha \cdot \text{LP-OPT}.$$

A Rounding Theorem

Stochastic Problem: LP can be solved in polynomial time.

Example: polynomial scenario setting

Deterministic problem: α-approximation algorithm A with respect to the LP relaxation, $\mathcal{A}(I) \leq \alpha \cdot$LP-OPT(I) for each I.

Example: "the greedy algorithm" for set cover is a log n-approximation algorithm w.r.t. LP relaxation.

Theorem: Can use such an α-approx. algorithm to get a 2α-approximation algorithm for stochastic set cover.

A Rounding Technique

Assume LP can be solved in polynomial time.

Suppose we have an α-approximation algorithm w.r.t. the LP relaxation for the deterministic problem.

Let (x,y) : optimal solution with cost OPT.

$$\sum_{S:e\in S} x_S + \sum_{S:e\in S} y_{A,S} \geq 1 \qquad \text{for each } A \subseteq U,\ e\in A$$

$\Rightarrow$ for every element e, either

$\sum_{S:e\in S} x_S \geq ½$ OR in each scenario $A : e\in A$, $\sum_{S:e\in S} y_{A,S} \geq ½$.

Let $E = \{e : \sum_{S:e\in S} x_S \geq ½\}$.

So $(2x)$ is a fractional set cover for the set $E \Rightarrow$ can “round” to get an integer set cover $\mathcal{S}$ of cost $\sum_{S\in\mathcal{S}} \omega_S \leq \alpha(\sum_S 2\omega_S x_S)$.

$\mathcal{S}$ is the first stage decision.

A Compact Formulation

p_A : probability of scenario $A \subseteq U$.

x_S : indicates if set S is picked in stage I.

Minimize $h(x) = \sum_S \omega_S x_S + f(x)$ s.t. $x_S \geq 0$ for each S

where, $f(x) = \sum_{A \subseteq U} p_A f_A(x)$

and $f_A(x) =$ min. $\sum_S W_S y_{A,S}$

s.t. $\sum_{S: e \in S} y_{A,S} \geq 1 - \sum_{S: e \in S} x_S$ for each $e \in A$

$y_{A,S} \geq 0$ for each S.

Equivalent to earlier LP.

Each $f_A(x)$ is convex, so $f(x)$ and $h(x)$ are convex functions.

The Algorithm

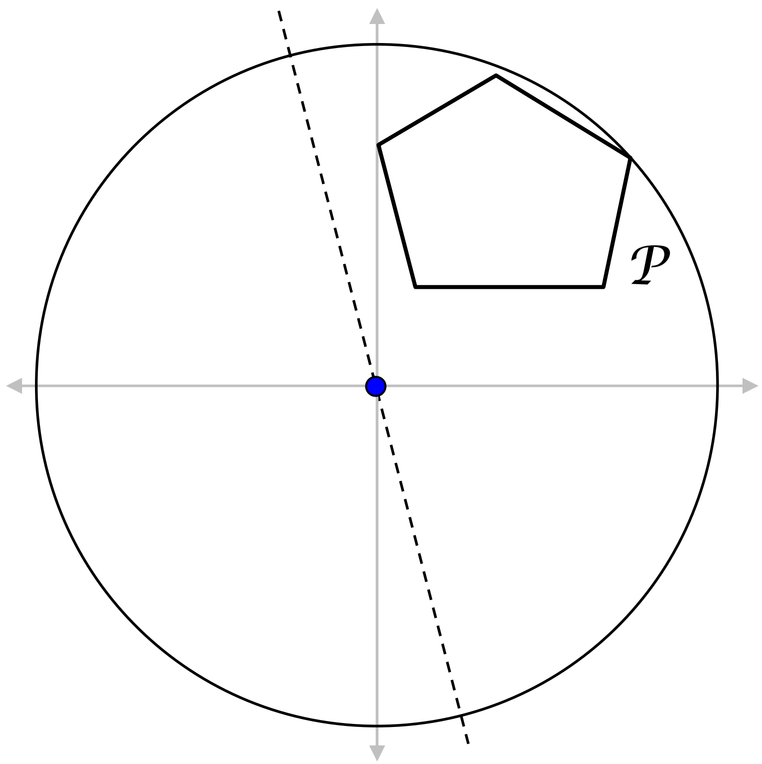
1. Get a $(1+\varepsilon)$-optimal solution (x) to compact convex program using the ellipsoid method.

2. Round (x) using a $\log n$-approx. algorithm for the deterministic problem to decide which sets to pick in stage I.

Obtain a $(2\log n+\varepsilon)$-approximation algorithm for the stochastic set cover problem.

The Ellipsoid Method

Min $c \cdot x$ subject to $x \in \mathcal{P}$.



Ellipsoid ≡ squashed sphere

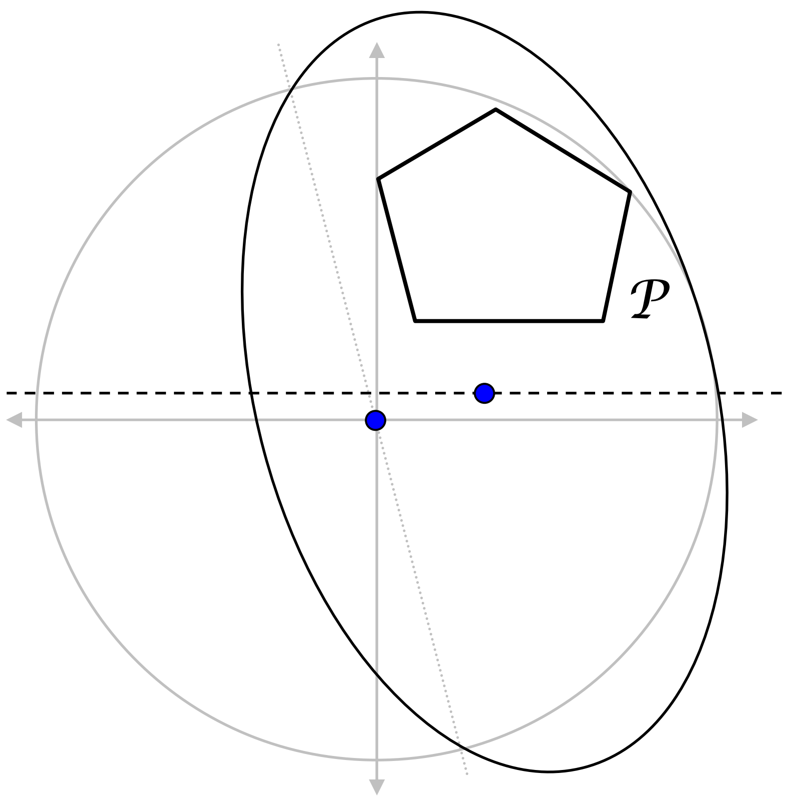
Start with ball containing polytope $\mathcal{P}$.

y_i = center of current ellipsoid.

If y_i is infeasible, use violated inequality to chop off infeasible half-ellipsoid.

The Ellipsoid Method

Min $c \cdot x$ subject to $x \in \mathcal{P}$.



Ellipsoid ≡ squashed sphere

Start with ball containing polytope $\mathcal{P}$.

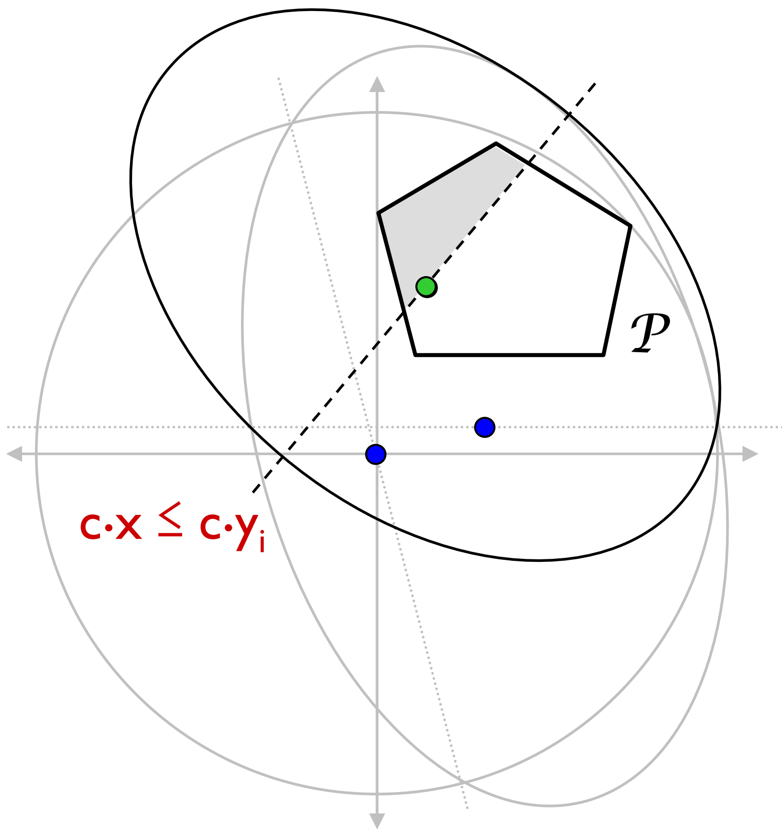
y_i = center of current ellipsoid.

If y_i is infeasible, use violated inequality to chop off infeasible half-ellipsoid.

New ellipsoid = min. volume ellipsoid containing "unchopped" half-ellipsoid.

The Ellipsoid Method

Min $c \cdot x$ subject to $x \in \mathcal{P}$.



Ellipsoid ≡ squashed sphere

Start with ball containing polytope $\mathcal{P}$.

y_i = center of current ellipsoid.

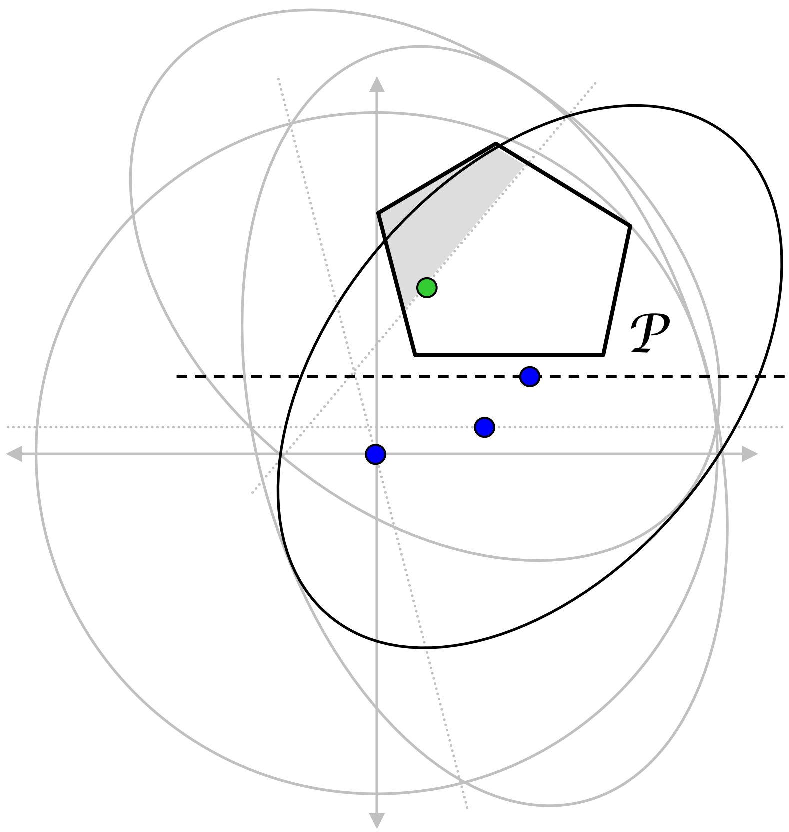
If y_i is infeasible, use violated inequality to chop off infeasible half-ellipsoid.

If $y_i \in \mathcal{P}$, use objective function cut $c \cdot x \leq c \cdot y_i$ to chop off polytope, half-ellipsoid.

New ellipsoid = min. volume ellipsoid containing "unchopped" half-ellipsoid.

The Ellipsoid Method

Min $c \cdot x$ subject to $x \in \mathcal{P}$.



Ellipsoid ≡ squashed sphere

Start with ball containing polytope $\mathcal{P}$.

y_i = center of current ellipsoid.

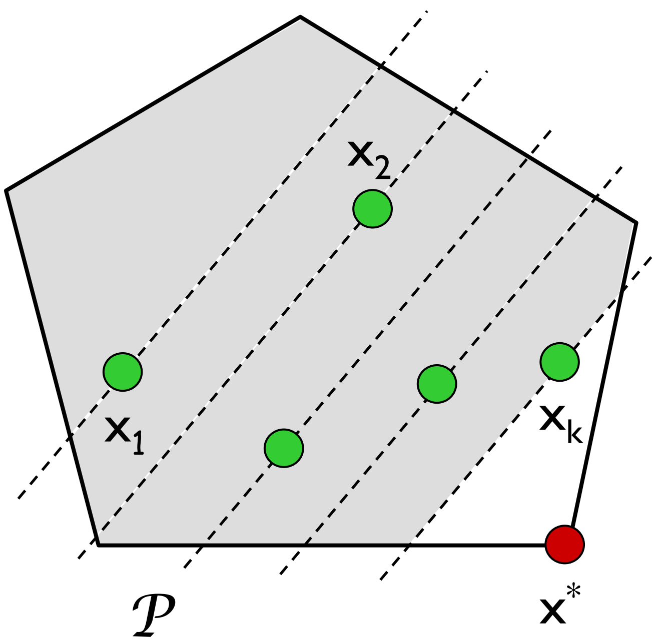
If y_i is infeasible, use violated inequality to chop off infeasible half-ellipsoid.

If $y_i \in \mathcal{P}$, use objective function cut $c \cdot x \leq c \cdot y_i$ to chop off polytope, half-ellipsoid.

New ellipsoid = min. volume ellipsoid containing "unchopped" half-ellipsoid.

The Ellipsoid Method

Min $c \cdot x$ subject to $x \in \mathcal{P}$.



Ellipsoid $\equiv$ squashed sphere

Start with ball containing polytope $\mathcal{P}$.

y_i = center of current ellipsoid.

If y_i is infeasible, use violated inequality to chop off infeasible half-ellipsoid.

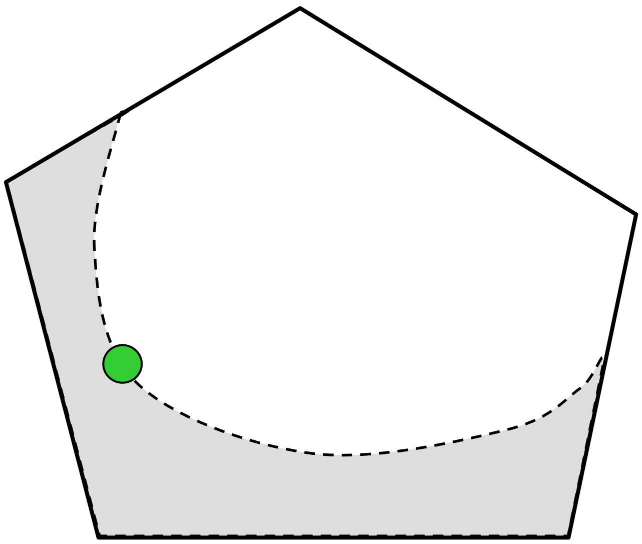
If $y_i \in \mathcal{P}$, use objective function cut $c \cdot x \leq c \cdot y_i$ to chop off polytope, half-ellipsoid.

New ellipsoid = min. volume ellipsoid containing "unchopped" half-ellipsoid.

$x_1, x_2, \ldots, x_k$: points lying in $\mathcal{P}$. $c \cdot x_k$ is a close to optimal value.

Ellipsoid for Convex Optimization

Min $h(x)$ subject to $x \in \mathcal{P}$.



Start with ball containing polytope $\mathcal{P}$.

y_i = center of current ellipsoid.

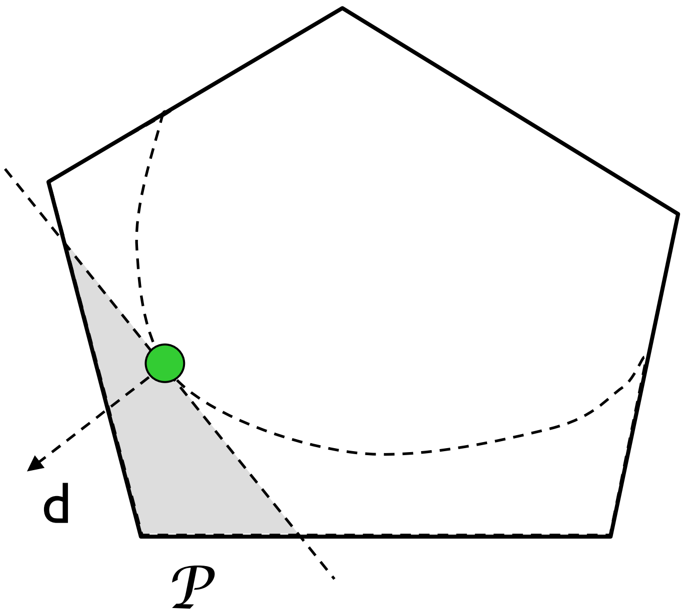
If y_i is infeasible, use violated inequality.

If $y_i \in \mathcal{P}$ – how to make progress?

add inequality $h(x) \leq h(y_i)$? Separation becomes difficult.

Ellipsoid for Convex Optimization

Min $h(x)$ subject to $x \in \mathcal{P}$.



Start with ball containing polytope $\mathcal{P}$.

y_i = center of current ellipsoid.

If y_i is infeasible, use violated inequality.

If $y_i \in \mathcal{P}$ – how to make progress?

add inequality $h(x) \leq h(y_i)$? Separation becomes difficult.

Let d = subgradient at y_i.

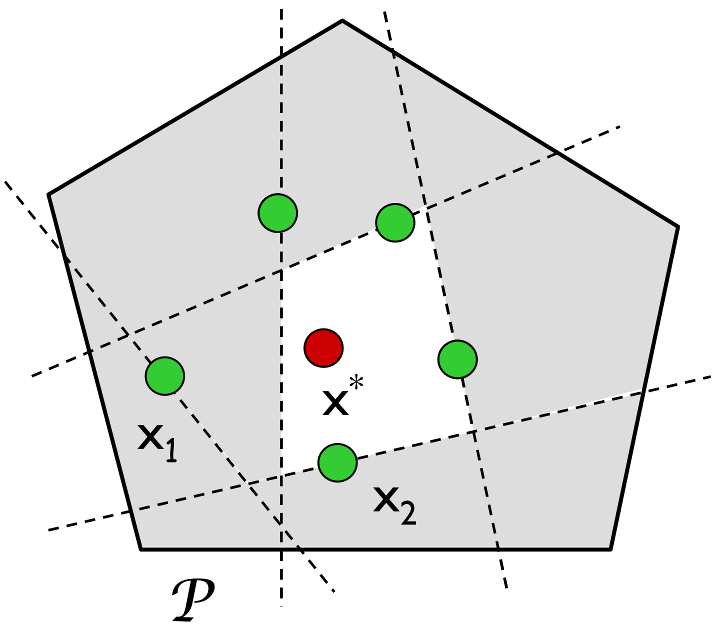
use subgradient cut $d \cdot (x - y_i) \leq 0$.

Generate new min. volume ellipsoid.

$d \in \Re^n$ is a subgradient of $h(.)$ at u, if for every v, $h(v)-h(u) \geq d \cdot (v-u)$.

Ellipsoid for Convex Optimization

Min $h(x)$ subject to $x \in \mathcal{P}$.



Start with ball containing polytope $\mathcal{P}$.

y_i = center of current ellipsoid.

If y_i is infeasible, use violated inequality.

If $y_i \in \mathcal{P}$ – how to make progress?

add inequality $h(x) \leq h(y_i)$? Separation becomes difficult.

Let d = subgradient at y_i.

use subgradient cut $d \cdot (x - y_i) \leq 0$.

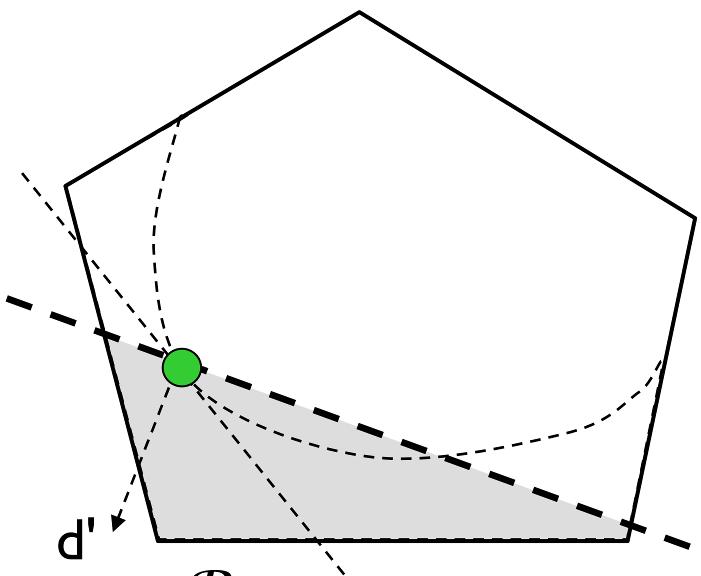
Generate new min. volume ellipsoid.

$d \in \mathfrak{R}^n$ is a subgradient of $h(.)$ at u, if for every v, $h(v)-h(u) \geq d \cdot (v-u)$.

$x_1, x_2, \ldots, x_k$: points in $\mathcal{P}$. Can show, $\min_{i=1\ldots k} h(x_i) \leq OPT+\rho$.

Ellipsoid for Convex Optimization

Min $h(x)$ subject to $x \in \mathcal{P}$.



Start with ball containing polytope $\mathcal{P}$.

y_i = center of current ellipsoid.

If y_i is infeasible, use violated inequality.

If $y_i \in \mathcal{P}$ – how to make progress?

add inequality $h(x) \leq h(y_i)$? Separation becomes difficult.

subgradient is difficult to compute.

Let d' = ε-subgradient at y_i.

use ε-subgradient cut $d' \cdot (x - y_i) \leq 0$.

$d' \in \Re^n$ is a ε-subgradient of $h(.)$ at u, if $\forall v \in \mathcal{P}$, $h(v) - h(u) \geq d' \cdot (v-u) - \varepsilon \cdot h(u)$.

$x_1, x_2, \ldots, x_k$: points in $\mathcal{P}$. Can show, $\min_{i=1\ldots k} h(x_i) \leq OPT/(1-\varepsilon) + \rho$.

Subgradients and ε-subgradients

Vector d is a subgradient of $h(.)$ at u,

if for every v, $h(v) - h(u) \geq d \cdot (v-u)$.

Vector d' is an ε-subgradient of $h(.)$ at u,

if for every $v \in \mathcal{P}$, $h(v) - h(u) \geq d' \cdot (v-u) - \varepsilon \cdot h(u)$.

$\mathcal{P} = \{ x : 0 \leq x_S \leq 1$ for each set $S \}$.

$h(x) = \Sigma_S \, \omega_S x_S + \Sigma_{A \subseteq U} \, p_A \, f_A(x) = \omega \cdot x + \Sigma_{A \subseteq U} \, p_A \, f_A(x)$

Lemma: Let d be a subgradient at u, and d' be a vector such that $d_S - \varepsilon\omega_S \leq d'_S \leq d_S$ for each set S. Then, d' is an ε-subgradient at point u.

Getting a "nice" subgradient

$h(x) = \omega \cdot x + \sum_{A \subseteq U} p_A f_A(x)$

$f_A(x) =$ min. $\sum_S W_S y_{A,S}$

s.t. $\sum_{S: e \in S} y_{A,S} \geq 1 - \sum_{S: e \in S} x_S \quad \forall e \in A$

$y_{A,S} \geq 0 \quad \forall S$

Getting a "nice" subgradient

$h(x) = \omega \cdot x + \sum_{A \subseteq U} p_A f_A(x)$

$f_A(x) =$ min. $\sum_S W_S y_{A,S}$ $=$ max. $\sum_{e \in A} (1 - \sum_{S: e \in S} x_S) z_{A,e}$

s.t. $\sum_{S: e \in S} y_{A,S} \geq 1 - \sum_{S: e \in S} x_S$ $\quad \forall e \in A$ s.t. $\sum_{e \in A \cap S} z_{A,e} \leq W_S$ $\quad \forall S$

$y_{A,S} \geq 0 \quad \forall S$ $\qquad z_{A,e} \geq 0 \quad \forall e \in A$

Getting a "nice" subgradient

$h(x) = \omega\cdot x + \sum_{A\subseteq U} p_A f_A(x)$

$f_A(x) =$ min. $\sum_S W_S y_{A,S}$ = max. $\sum_e (1 - \sum_{S:e\in S} x_S)\, z_{A,e}$

s.t. $\sum_{S:e\in S} y_{A,S} \geq 1 - \sum_{S:e\in S} x_S \quad \forall e\in A$ s.t. $\sum_{e\in S} z_{A,e} \leq W_S \quad \forall S$

$y_{A,S} \geq 0 \quad \forall S$ $z_{A,e} = 0 \;\; \forall e\notin A, \;\; z_{A,e} \geq 0 \;\; \forall e$

Consider point $u \in \Re^n$. Let $z_A \equiv$ optimal dual solution for A at u.

Lemma: For any point $v \in \Re^n$, we have $h(v) - h(u) \geq d\cdot(v-u)$ where $d_S = \omega_S - \sum_{A\subseteq U} p_A \sum_{e\in S} z_{A,e}$.

$\Rightarrow$ d is a subgradient of h(.) at point u.

Computing an ε-Subgradient

Given point $u \in \Re^n$. $z_A \equiv$ optimal dual solution for A at u.

Subgradient at u: $d_S = \omega_S - \sum_{A \subseteq U} p_A \sum_{e \in S} z_{A,e}$.

Want: d' such that $d_S - \varepsilon\omega_S \leq d'_S \leq d_S$ for each S.

For each S, $-W_S \leq d_S \leq \omega_S$. Let $\lambda = \max_S W_S / \omega_S$.

Sample once from black box to get random scenario A.

Compute X with $X_S = \omega_S - \sum_{e \in S} z_{A,e}$.

$\mathbf{E}[X_S] = d_S$ and $\mathbf{Var}[X_S] \leq W_S^2$.

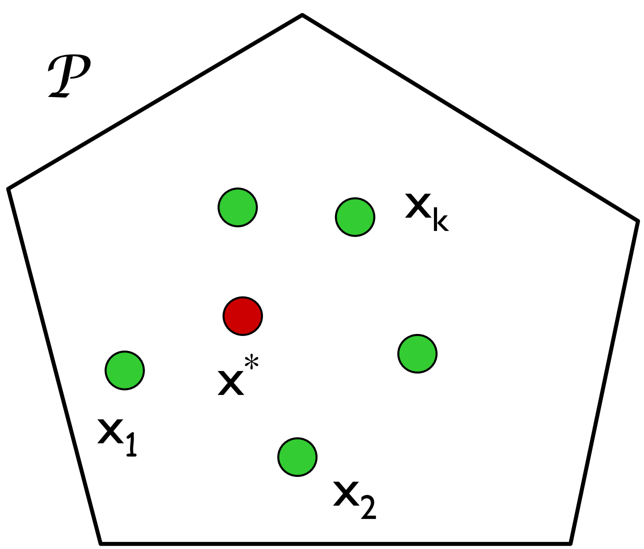
Sample $O(\lambda^2/\varepsilon^2 \cdot \log(n/\delta))$ times to compute d' such that

$\mathbf{Pr}[\forall S, d_S - \varepsilon\omega_S \leq d'_S \leq d_S] \geq 1-\delta$.

$\Rightarrow$ d' is an ε-subgradient at u with probability $\geq 1-\delta$.

Putting it all together

Min $h(x)$ subject to $x \in \mathcal{P}$.



✓ Can compute ε-subgradients.

Run ellipsoid algorithm.

Given y_i = center of current ellipsoid.

If y_i is infeasible, use violated inequality as a cut.

If $y_i \in \mathcal{P}$ use ε-subgradient cut.

Continue with smaller ellipsoid.

Generate points $x_1, x_2, \ldots, x_k$ in $\mathcal{P}$. Return $\bar{x} = \text{argmin}_{i=1\ldots k}\, h(x_i)$.

Get that $h(\bar{x}) \leq OPT/(1-\varepsilon) + \rho$.

Finally,

Get solution x with $h(x)$ close to OPT.

Sample initially to detect if $OPT = \Omega(1/\lambda)$ – this allows one to get a $(1+\varepsilon)\cdot OPT$ guarantee.

Theorem: Compact convex program can be solved to within a factor of $(1 + \varepsilon)$ in polynomial time, with high probability.

Gives a $(2\log n+\varepsilon)$-approximation algorithm for the stochastic set cover problem.

A Solvable Class of Stochastic LPs

Minimize $h(x) = w.x + \sum_{A \subseteq U} p_A f_A(x)$

s.t. $x \in \Re^n,\ x \geq 0,\ x \in \mathcal{P}$

where $f_A(x) = \text{min. } w^A.y_A + c^A.r_A$

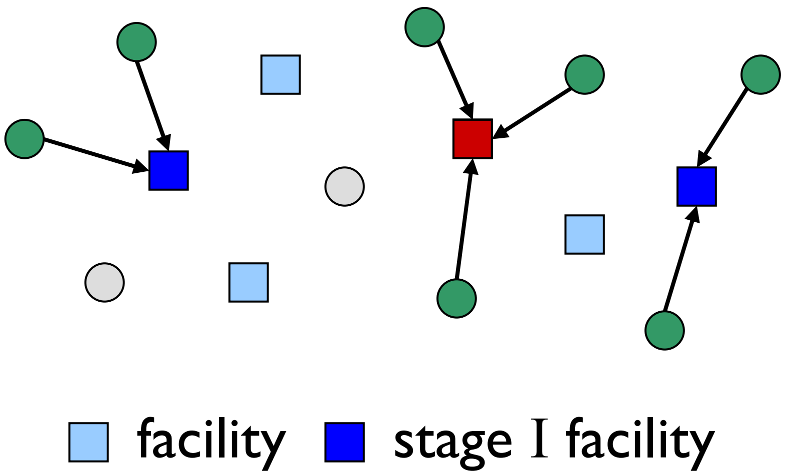
s.t. $Br_A \geq j^A$

$Dr_A + Ty_A \geq \ell^A - Tx$

$y_A \in \Re^n,\ r_A \in \Re^m,\ y_A \geq 0,\ r_A \geq 0.$

Theorem: Can get a $(1+\varepsilon)$-optimal solution for this class of stochastic programs in polynomial time.

2-Stage Stochastic Facility Location



Distribution over clients gives the set of clients to serve.

Stage I: Open some facilities in advance; pay cost f_i for facility i.

Stage I cost = $\sum_{(i \text{ opened})} f_i$.

Actual scenario A = { ● clients to serve}, materializes.

Stage II: Can open more facilities to serve clients in A; pay cost f_i^A to open facility i. Assign clients in A to facilities.

Stage II cost = $\sum_{\substack{i \text{ opened in} \\ \text{scenario } A}} f_i^A$ + (cost of serving clients in A).

A Convex Program

p_A : probability of scenario $A \subseteq \mathcal{D}$.

y_i : indicates if facility i is opened in stage I.

$y_{A,i}$: indicates if facility i is opened in scenario A.

$x_{A,ij}$: whether client j is assigned to facility i in scenario A.

Minimize $h(y) = \sum_i f_i y_i + g(y)$ s.t. $y_i \geq 0$ for each i

(SUFL-P)

where, $g(y) = \sum_{A \subseteq \mathcal{D}} p_A g_A(y)$

and $g_A(y) = \min. \sum_i F_i y_{A,i} + \sum_{j,i} c_{ij} x_{A,ij}$

s.t. $\sum_i x_{A,ij} \geq 1$ for each $j \in A$

$x_{A,ij} \leq y_i + y_{A,i}$ for each i,j

$x_{A,ij}, y_{A,i} \geq 0$ for each i,j.

Moral of the Story

- Even though the Stochastic LP relaxation has an exponential number of variables and constraints, we can still obtain near-optimal solutions to fractional first-stage decisions
- Fractional first-stage decisions are sufficient to decouple the two stages near-optimally
- Many applications: multicommodity flows, vertex cover, facility location, …
- But we still have to solve convex program with many, many samples (not just λ)!

Sample Average Approximation

Sample Average Approximation (SAA) method:

- Sample initially N times from scenario distribution
- Solve 2-stage problem estimating p_A with frequency of occurrence of scenario A

How large should N be?

Kleywegt, Shapiro & Homem De-Mello (KSH01):

- bound N by variance of a certain quantity – need not be polynomially bounded even for our class of programs.

SwamyS:

- show using ε-subgradients that for our class, N can be poly-bounded.

Nemirovskii & Shapiro:

- show that for SSC with non-scenario dependent costs, KSH01 gives polynomial bound on N for (preprocessing + SAA) algorithm.

Sample Average Approximation

Sample Average Approximation (SAA) method:

- Sample N times from distribution
- Estimate p_A by q_A = frequency of occurrence of scenario A

(P) $\min_{x\in\mathcal{P}} \left(h(x) = \omega\cdot x + \sum_{A\subseteq U} p_A f_A(x)\right)$

(SAA-P) $\min_{x\in\mathcal{P}} \left(h'(x) = \omega\cdot x + \sum_{A\subseteq U} q_A f_A(x)\right)$

To show: With poly-bounded N, if $\bar{x}$ solves (SAA-P) then $h(\bar{x}) \approx$ OPT.

Let $z_A \equiv$ optimal dual solution for scenario A at point $u \in \Re^m$.

$\Rightarrow d_u$ with $d_{u,S} = \omega_S - \sum_{A\subseteq U} q_A \sum_{e\in S} z_{A,e}$ is a subgradient of $h'(.)$ at u.

Lemma: With high probability, for "many" points u in $\mathcal{P}$,

d_u is a subgradient of $h'(.)$ at u,

d_u is an approximate subgradient of $h(.)$ at u.

Establishes "closeness" of $h(.)$ and $h'(.)$ and suffices to prove result.

Intuition: Can run ellipsoid on both (P) and (SAA-P) using the same vector d_u at feasible point u.

Multi-stage Problems

Given : Distribution over inputs.

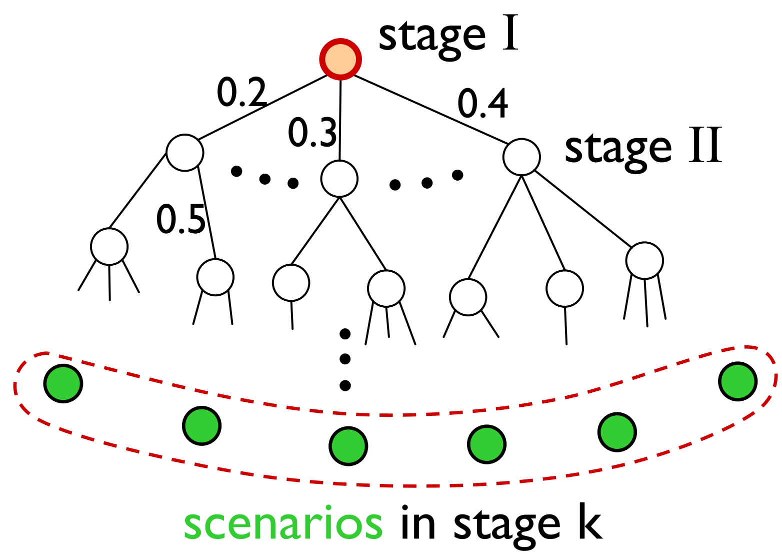
Stage I : Make some advance decisions – hedge against uncertainty.

Uncertainty evolves in various stages.

Learn new information in each stage.

Can take recourse actions in each stage – can augment earlier solution paying a recourse cost.

k-stage problem
≡ k decision points



Multi-stage Problems

Given : Distribution over inputs.

Stage I : Make some advance decisions – hedge against uncertainty.

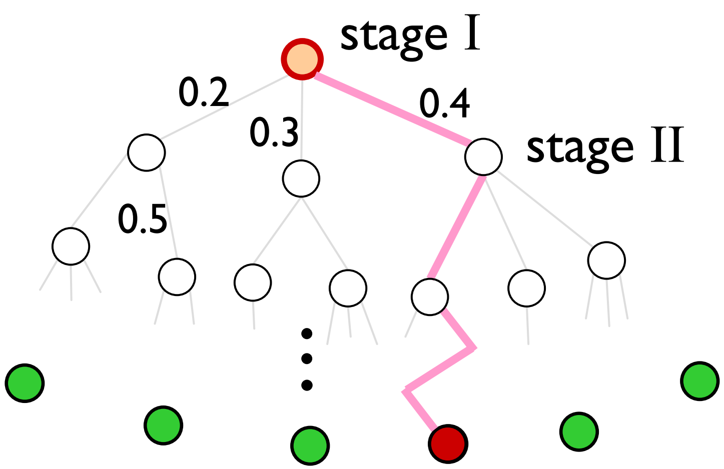
Uncertainty evolves in various stages.

Learn new information in each stage.

Can take recourse actions in each stage – can augment earlier solution paying a recourse cost.

k-stage problem

≡ k decision points



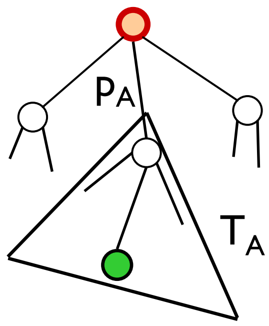
scenarios in stage k

Choose stage I decisions to minimize

expected total cost =

(stage I cost) + $\mathbf{E}_{\text{all scenarios}}$ [cost of stages 2 … k].

Solving k-stage LPs



Consider 3-stage SSC.

How to compute an ε-subgradient?

Want to get d' that is component-wise close to subgradient d where $d_S = \omega_S - \sum_A p_A$(dual solution to T_A).

Problem: To compute d (even in expectation) need to solve the dual of a 2-stage LP – dual has exponential size!

Fix:

- Formulate a new compact non-linear dual of polynomial size.
- Dual has a 2-stage primal LP embedded inside – solve this using earlier algorithm.

Recursively apply this idea to solve k-stage stochastic LPs.

This is just the beginning!

- Multi-stage problems with a variable number of stages
- [Dean, Goemans, Vondrak 04] Stochastic knapsack problem – in each stage decide whether to pack next item
- [Levi, Pal, Roundy, Shmoys 05] Stochastic inventory control problems – in each stage react to updated forecast of future demand
- Stochastic Dynamic Programming ???

Thank You.