

Multistage stochastic optimization and polyhedral geometry

PhD Defense Maël Forcier

advised by Stéphane Gaubert and Vincent Leclère,
supervised by Jean-Philippe Chancelier.

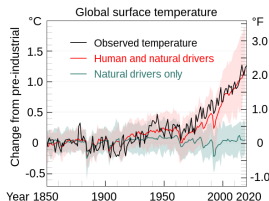
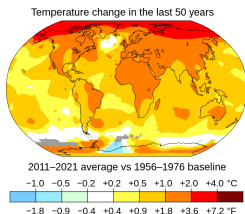
December 14th 2022



Motivating example: hydroelectric energy management



- Need low-carbon energy to stop global warming
- Hydroelectricity is a controllable renewable energy
- 83% of electricity is hydroelectric in Brazil, 17% in France and 92% in Norway



Motivating example: hydroelectric energy management



- u water hustled
- d demand
- c cost of unmet demand
- x_0/x_1 water in the reservoir
- \bar{x} capacity of the reservoir
- w rain and runoff

$$\min_{u, x_1} c(d - u)$$

$$s.t. \quad 0 \leq u \leq d$$

$$x_1 \leq x_0 - u + w$$

$$0 \leq x_1 \leq \bar{x}$$

$$x_0 \text{ fixed}$$

Motivating example: hydroelectric energy management



At step t

- u_t water hustled
- d_t demand
- c_t cost of unmet demand
- x_t water in the reservoir
- \bar{x} capacity of the reservoir
- w_t rain and runoff

$$\begin{aligned} \min_{u_t, x_t} \quad & \sum_{t=1}^T c_t(d_t - u_t) \\ \text{s.t.} \quad & 0 \leq u_t \leq d_t, \quad \forall t \in [T] \\ & x_{t+1} \leq x_t - u_t + w_t, \quad \forall t \in [T] \\ & 0 \leq x_t \leq \bar{x}, \quad \forall t \in [T] \\ & x_0 \text{ fixed} \end{aligned}$$

Motivating example: hydroelectric energy management



At step t

- u_t water hustled
- d_t demand
- c_t cost of unmet demand
- x_t water in the reservoir
- \bar{x} capacity of the reservoir
- w_t rain and runoff

$$\begin{aligned} \min_{u_t, x_t} \quad & \sum_{t=1}^T c_t(d_t - u_t) \\ \text{s.t.} \quad & 0 \leq u_t \leq d_t, \quad \forall t \in [T] \\ & x_{t+1} \leq x_t - u_t + w_t, \quad \forall t \in [T] \\ & 0 \leq x_t \leq \bar{x}, \quad \forall t \in [T] \\ & x_0 \text{ fixed} \end{aligned}$$

General form

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & c^\top x \\ \text{s.t.} \quad & Ax \leq b \end{aligned}$$

Linear Programming and polyhedra

Definition

Polyhedron :

Intersection of finite number of halfspaces

$$\min_{x \in \mathbb{R}^n} c^\top x$$

$$\text{s.t.} \quad Ax \leq b$$

The set $P = \{x \in \mathbb{R}^n \mid Ax \leq b\}$ of admissible solutions is a polyhedron.

$$A = \begin{pmatrix} 1 & 1 \\ & \end{pmatrix} \quad b = \begin{pmatrix} 1 \\ \end{pmatrix}$$

$$x_1 + x_2 \leq 1$$

(1)

(2)

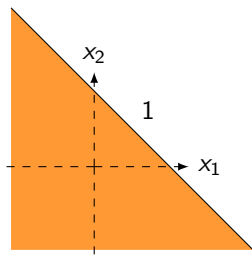
(3)

(4)

(5)

(6)

(7)



Linear Programming and polyhedra

Definition

Polyhedron :

Intersection of finite number of halfspaces

$$\min_{x \in \mathbb{R}^n} c^\top x$$

$$\text{s.t.} \quad Ax \leq b$$

The set $P = \{x \in \mathbb{R}^n \mid Ax \leq b\}$ of admissible solutions is a polyhedron.

$$A = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \quad b = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

$$x_1 + x_2 \leq 1 \quad (1)$$

$$x_1 - x_2 \leq 1 \quad (2)$$

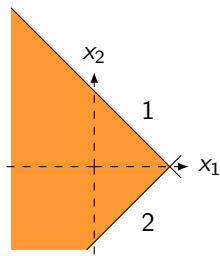
(3)

(4)

(5)

(6)

(7)



Linear Programming and polyhedra

Definition

Polyhedron :

Intersection of finite number of halfspaces

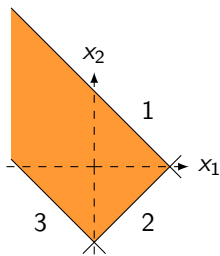
$$\min_{x \in \mathbb{R}^n} c^\top x$$

$$\text{s.t.} \quad Ax \leq b$$

The set $P = \{x \in \mathbb{R}^n \mid Ax \leq b\}$ of admissible solutions is a polyhedron.

$$A = \begin{pmatrix} 1 & 1 \\ 1 & -1 \\ -1 & -1 \end{pmatrix} \quad b = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$$
$$\begin{array}{ll} x_1 + x_2 \leq 1 & (1) \\ x_1 - x_2 \leq 1 & (2) \\ -x_1 - x_2 \leq 1 & (3) \end{array}$$

(4)
(5)
(6)
(7)



Linear Programming and polyhedra

$$\begin{array}{ll}\min_{x \in \mathbb{R}^n} & c^\top x \\ \text{s.t.} & Ax \leq b\end{array}$$

Definition

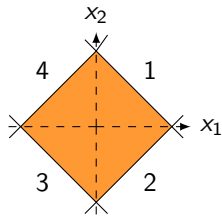
Polyhedron :

Intersection of finite number of halfspaces

The set $P = \{x \in \mathbb{R}^n \mid Ax \leq b\}$ of admissible solutions is a polyhedron.

$$A = \begin{pmatrix} 1 & 1 \\ 1 & -1 \\ -1 & -1 \\ -1 & 1 \end{pmatrix} \quad b = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}$$

$x_1 + x_2 \leq 1$	(1)
$x_1 - x_2 \leq 1$	(2)
$-x_1 - x_2 \leq 1$	(3)
$-x_1 + x_2 \leq 1$	(4)
	(5)
	(6)
	(7)



Linear Programming and polyhedra

$$\min_{x \in \mathbb{R}^n} c^\top x$$

$$\text{s.t.} \quad Ax \leq b$$

Definition

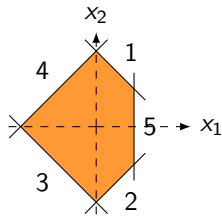
Polyhedron :

Intersection of finite number of halfspaces

The set $P = \{x \in \mathbb{R}^n \mid Ax \leq b\}$ of admissible solutions is a polyhedron.

$$A = \begin{pmatrix} 1 & 1 \\ 1 & -1 \\ -1 & -1 \\ -1 & 1 \\ 1 & 0 \end{pmatrix} \quad b = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 0.5 \end{pmatrix}$$
$$\begin{aligned} x_1 + x_2 &\leq 1 & (1) \\ x_1 - x_2 &\leq 1 & (2) \\ -x_1 - x_2 &\leq 1 & (3) \\ -x_1 + x_2 &\leq 1 & (4) \\ x_1 &\leq 0.5 & (5) \end{aligned}$$

(6)
(7)



Linear Programming and polyhedra

$$\begin{array}{ll}\min_{x \in \mathbb{R}^n} & c^\top x \\ \text{s.t.} & Ax \leq b\end{array}$$

Definition

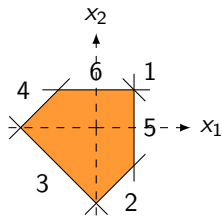
Polyhedron :

Intersection of finite number of halfspaces

The set $P = \{x \in \mathbb{R}^n \mid Ax \leq b\}$ of admissible solutions is a polyhedron.

$$A = \begin{pmatrix} 1 & 1 \\ 1 & -1 \\ -1 & -1 \\ -1 & 1 \\ 1 & 0 \\ 0 & 1 \end{pmatrix} \quad b = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 0.5 \\ 0.5 \end{pmatrix}$$
$$\begin{array}{ll} x_1 + x_2 \leq 1 & (1) \\ x_1 - x_2 \leq 1 & (2) \\ -x_1 - x_2 \leq 1 & (3) \\ -x_1 + x_2 \leq 1 & (4) \\ x_1 \leq 0.5 & (5) \\ x_2 \leq 0.5 & (6) \end{array}$$

(7)



Linear Programming and polyhedra

Definition

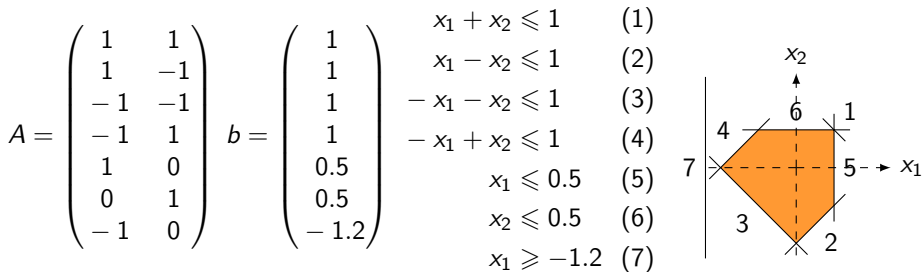
Polyhedron :

Intersection of finite number of halfspaces

$$\min_{x \in \mathbb{R}^n} c^\top x$$

$$\text{s.t.} \quad Ax \leq b$$

The set $P = \{x \in \mathbb{R}^n \mid Ax \leq b\}$ of admissible solutions is a polyhedron.



But renewables are inherently **stochastic** !



Rain, runoff, cost and demand are **random**.

At step t

- u_t water hustled
- d_t demand
- c_t cost of unmet demand
- x_t water in the reservoir
- \bar{x} capacity of the reservoir
- w_t rain and runoff

$$\begin{aligned} \min_{u_t, x_t} \quad & \sum_{t=1}^T c_t(d_t - u_t) \\ \text{s.t.} \quad & 0 \leq u_t \leq d_t, \quad \forall t \in [T] \\ & x_{t+1} \leq x_t - u_t + w_t, \quad \forall t \in [T] \\ & 0 \leq x_t \leq \bar{x}, \quad \forall t \in [T] \\ & x_0 \text{ fixed} \end{aligned}$$

But renewables are inherently **stochastic** !



Rain, runoff, cost and demand are **random**.

At step t

- \mathbf{u}_t water hustled
- \mathbf{d}_t demand
- \mathbf{c}_t cost of unmet demand
- \mathbf{x}_t water in the reservoir
- \bar{x} capacity of the reservoir
- \mathbf{w}_t rain and runoff

$$\min_{\mathbf{u}_t, \mathbf{x}_t} \mathbb{E} \left[\sum_{t=1}^T \mathbf{c}_t (\mathbf{d}_t - \mathbf{u}_t) \right]$$

$$\text{s.t. } 0 \leq \mathbf{u}_t \leq \mathbf{d}_t, \quad \forall t \in [T]$$

$$\mathbf{x}_{t+1} \leq \mathbf{x}_t - \mathbf{u}_t + \mathbf{w}_t, \quad \forall t \in [T]$$

$$0 \leq \mathbf{x}_t \leq \bar{x}, \quad \forall t \in [T]$$

$$\mathbf{x}_0 \equiv x_0 \text{ given}$$

$$\sigma(\mathbf{u}_t) \subset \sigma(\mathbf{c}_\tau, \mathbf{d}_\tau, \mathbf{w}_\tau)_{\tau \leq t}, \quad \forall t \in [T]$$

$$\underbrace{\sigma(\mathbf{x}_t) \subset \sigma(\mathbf{c}_\tau, \mathbf{d}_\tau, \mathbf{w}_\tau)_{\tau \leq t}}_{\text{Measurability constraints}}, \quad \forall t \in [T]$$

Measurability constraints

Multistage stochastic linear programming (MSLP)

$$\begin{aligned}
 \min_{(\mathbf{x}_t)_{t \in [T]}} \quad & \mathbb{E} \left[\sum_{t=1}^T \mathbf{c}_t^\top \mathbf{x}_t \right] \\
 \text{s.t.} \quad & \mathbf{A}_t \mathbf{x}_t + \mathbf{B}_t \mathbf{x}_{t-1} \leq \mathbf{b}_t \quad \forall t \in [T] \\
 & \sigma(\mathbf{x}_t) \subset \sigma(\mathbf{c}_\tau, \mathbf{A}_\tau, \mathbf{B}_\tau, \mathbf{b}_\tau)_{\tau \leq t} \quad \forall t \in [T] \\
 & \mathbf{x}_0 \equiv \mathbf{x}_0 \text{ given}
 \end{aligned}$$

$\xi_t = (\mathbf{c}_t, \mathbf{A}_t, \mathbf{B}_t, \mathbf{b}_t)_{t \in [T]}$ is assumed to be **stagewise independent**.

At each time step: the present noise is revealed then we take a decision.

$$x_0 \rightsquigarrow \xi_1 \rightsquigarrow x_1 \rightsquigarrow \xi_2 \rightsquigarrow \dots \rightsquigarrow x_{T-1} \rightsquigarrow \xi_T \rightsquigarrow x_T$$

Equivalent form

$$\min_{x_1: \mathbf{A}_1 x_1 + \mathbf{B}_1 x_0 \leq \mathbf{b}_1} \mathbf{c}_1^\top x_1 + \mathbb{E} \left[\min_{x_2: \mathbf{A}_2 x_2 + \mathbf{B}_2 x_1 \leq \mathbf{b}_2} \mathbf{c}_2^\top x_2 + \mathbb{E} \left[\dots + \mathbb{E} \left[\min_{x_T: \mathbf{A}_T x_T + \mathbf{B}_T x_{T-1} \leq \mathbf{b}_T} \mathbf{c}_T^\top x_T \right] \right] \right].$$

Multistage stochastic linear programming (MSLP)

$$\begin{aligned} \min_{(\mathbf{x}_t)_{t \in [T]}} \quad & \mathbb{E} \left[\sum_{t=1}^T \mathbf{c}_t^\top \mathbf{x}_t \right] \\ \text{s.t.} \quad & \mathbf{A}_t \mathbf{x}_t + \mathbf{B}_t \mathbf{x}_{t-1} \leq \mathbf{b}_t \quad \forall t \in [T] \\ & \sigma(\mathbf{x}_t) \subset \sigma(\mathbf{c}_\tau, \mathbf{A}_\tau, \mathbf{B}_\tau, \mathbf{b}_\tau)_{\tau \leq t} \quad \forall t \in [T] \\ & \mathbf{x}_0 \equiv \mathbf{x}_0 \text{ given} \end{aligned}$$

$\xi_t = (\mathbf{c}_t, \mathbf{A}_t, \mathbf{B}_t, \mathbf{b}_t)_{t \in [T]}$ is assumed to be **stagewise independent**.

At each time step: the present noise is revealed then we take a decision.

$$x_0 \rightsquigarrow \xi_1 \rightsquigarrow x_1 \rightsquigarrow \xi_2 \rightsquigarrow \cdots \rightsquigarrow x_{T-1} \rightsquigarrow \xi_T \rightsquigarrow x_T$$

Equivalent form

$$\min_{x_1: \mathbf{A}_1 x_1 + \mathbf{B}_1 x_0 \leq \mathbf{b}_1} \mathbf{c}_1^\top x_1 + \mathbb{E} \left[\min_{x_2: \mathbf{A}_2 x_2 + \mathbf{B}_2 x_1 \leq \mathbf{b}_2} \mathbf{c}_2^\top x_2 + \mathbb{E} \left[\cdots + \mathbb{E} \left[\min_{x_T: \mathbf{A}_T x_T + \mathbf{B}_T x_{T-1} \leq \mathbf{b}_T} \mathbf{c}_T^\top x_T \right] \right] \right]$$

Multistage stochastic linear programming (MSLP)

$$\begin{aligned} \min_{(\mathbf{x}_t)_{t \in [T]}} \quad & \mathbb{E} \left[\sum_{t=1}^T \mathbf{c}_t^\top \mathbf{x}_t \right] \\ \text{s.t.} \quad & \mathbf{A}_t \mathbf{x}_t + \mathbf{B}_t \mathbf{x}_{t-1} \leq \mathbf{b}_t \quad \forall t \in [T] \\ & \sigma(\mathbf{x}_t) \subset \sigma(\mathbf{c}_\tau, \mathbf{A}_\tau, \mathbf{B}_\tau, \mathbf{b}_\tau)_{\tau \leq t} \quad \forall t \in [T] \\ & \mathbf{x}_0 \equiv \mathbf{x}_0 \text{ given} \end{aligned}$$

$\xi_t = (\mathbf{c}_t, \mathbf{A}_t, \mathbf{B}_t, \mathbf{b}_t)_{t \in [T]}$ is assumed to be **stagewise independent**.

At each time step: the present noise is revealed then we take a decision.

$$\mathbf{x}_0 \rightsquigarrow \xi_1 \rightsquigarrow \mathbf{x}_1 \rightsquigarrow \xi_2 \rightsquigarrow \cdots \rightsquigarrow \mathbf{x}_{T-1} \rightsquigarrow \xi_T \rightsquigarrow \mathbf{x}_T$$

Equivalent form

$$\min_{\mathbf{x}_1: \mathbf{A}_1 \mathbf{x}_1 + \mathbf{B}_1 \mathbf{x}_0 \leq \mathbf{b}_1} \mathbf{c}_1^\top \mathbf{x}_1 + \mathbb{E} \left[\min_{\mathbf{x}_2: \mathbf{A}_2 \mathbf{x}_2 + \mathbf{B}_2 \mathbf{x}_1 \leq \mathbf{b}_2} \mathbf{c}_2^\top \mathbf{x}_2 + \mathbb{E} \left[\cdots + \mathbb{E} \left[\min_{\mathbf{x}_T: \mathbf{A}_T \mathbf{x}_T + \mathbf{B}_T \mathbf{x}_{T-1} \leq \mathbf{b}_T} \mathbf{c}_T^\top \mathbf{x}_T \right] \right] \right]$$

Multistage stochastic linear programming (MSLP)

$$\begin{aligned} \min_{(\mathbf{x}_t)_{t \in [T]}} \quad & \mathbb{E} \left[\sum_{t=1}^T \mathbf{c}_t^\top \mathbf{x}_t \right] \\ \text{s.t.} \quad & \mathbf{A}_t \mathbf{x}_t + \mathbf{B}_t \mathbf{x}_{t-1} \leq \mathbf{b}_t \quad \forall t \in [T] \\ & \sigma(\mathbf{x}_t) \subset \sigma(\mathbf{c}_\tau, \mathbf{A}_\tau, \mathbf{B}_\tau, \mathbf{b}_\tau)_{\tau \leq t} \quad \forall t \in [T] \\ & \mathbf{x}_0 \equiv \mathbf{x}_0 \text{ given} \end{aligned}$$

$\xi_t = (\mathbf{c}_t, \mathbf{A}_t, \mathbf{B}_t, \mathbf{b}_t)_{t \in [T]}$ is assumed to be **stagewise independent**.

At each time step: the present noise is revealed then we take a decision.

$$\mathbf{x}_0 \rightsquigarrow \xi_1 \rightsquigarrow \mathbf{x}_1 \rightsquigarrow \xi_2 \rightsquigarrow \dots \rightsquigarrow \mathbf{x}_{T-1} \rightsquigarrow \xi_T \rightsquigarrow \mathbf{x}_T$$

Equivalent form

$$\min_{\mathbf{x}_1: \mathbf{A}_1 \mathbf{x}_1 + \mathbf{B}_1 \mathbf{x}_0 \leq \mathbf{b}_1} \mathbf{c}_1^\top \mathbf{x}_1 + \mathbb{E} \left[\min_{\mathbf{x}_2: \mathbf{A}_2 \mathbf{x}_2 + \mathbf{B}_2 \mathbf{x}_1 \leq \mathbf{b}_2} \mathbf{c}_2^\top \mathbf{x}_2 + \mathbb{E} \left[\dots + \mathbb{E} \left[\min_{\mathbf{x}_T: \mathbf{A}_T \mathbf{x}_T + \mathbf{B}_T \mathbf{x}_{T-1} \leq \mathbf{b}_T} \mathbf{c}_T^\top \mathbf{x}_T \right] \right] \right]$$

Multistage stochastic linear programming (MSLP)

$$\begin{aligned} \min_{(\mathbf{x}_t)_{t \in [T]}} \quad & \mathbb{E} \left[\sum_{t=1}^T \mathbf{c}_t^\top \mathbf{x}_t \right] \\ \text{s.t.} \quad & \mathbf{A}_t \mathbf{x}_t + \mathbf{B}_t \mathbf{x}_{t-1} \leq \mathbf{b}_t \quad \forall t \in [T] \\ & \sigma(\mathbf{x}_t) \subset \sigma(\mathbf{c}_\tau, \mathbf{A}_\tau, \mathbf{B}_\tau, \mathbf{b}_\tau)_{\tau \leq t} \quad \forall t \in [T] \\ & \mathbf{x}_0 \equiv \mathbf{x}_0 \text{ given} \end{aligned}$$

$\xi_t = (\mathbf{c}_t, \mathbf{A}_t, \mathbf{B}_t, \mathbf{b}_t)_{t \in [T]}$ is assumed to be **stagewise independent**.

At each time step: the present noise is revealed then we take a decision.

$$\mathbf{x}_0 \rightsquigarrow \xi_1 \rightsquigarrow \mathbf{x}_1 \rightsquigarrow \xi_2 \rightsquigarrow \dots \rightsquigarrow \mathbf{x}_{T-1} \rightsquigarrow \xi_T \rightsquigarrow \mathbf{x}_T$$

Equivalent form

$$\min_{\mathbf{x}_1: \mathbf{A}_1 \mathbf{x}_1 + \mathbf{B}_1 \mathbf{x}_0 \leq \mathbf{b}_1} \mathbf{c}_1^\top \mathbf{x}_1 + \mathbb{E} \left[\min_{\mathbf{x}_2: \mathbf{A}_2 \mathbf{x}_2 + \mathbf{B}_2 \mathbf{x}_1 \leq \mathbf{b}_2} \mathbf{c}_2^\top \mathbf{x}_2 + \mathbb{E} \left[\dots + \mathbb{E} \left[\min_{\mathbf{x}_T: \mathbf{A}_T \mathbf{x}_T + \mathbf{B}_T \mathbf{x}_{T-1} \leq \mathbf{b}_T} \mathbf{c}_T^\top \mathbf{x}_T \right] \right] \right]$$

Multistage stochastic linear programming (MSLP)

$$\begin{aligned}
 \min_{(\mathbf{x}_t)_{t \in [T]}} \quad & \mathbb{E} \left[\sum_{t=1}^T \mathbf{c}_t^\top \mathbf{x}_t \right] \\
 \text{s.t.} \quad & \mathbf{A}_t \mathbf{x}_t + \mathbf{B}_t \mathbf{x}_{t-1} \leq \mathbf{b}_t \quad \forall t \in [T] \\
 & \sigma(\mathbf{x}_t) \subset \sigma(\mathbf{c}_\tau, \mathbf{A}_\tau, \mathbf{B}_\tau, \mathbf{b}_\tau)_{\tau \leq t} \quad \forall t \in [T] \\
 & \mathbf{x}_0 \equiv \mathbf{x}_0 \text{ given}
 \end{aligned}$$

$\xi_t = (\mathbf{c}_t, \mathbf{A}_t, \mathbf{B}_t, \mathbf{b}_t)_{t \in [T]}$ is assumed to be **stagewise independent**.

At each time step: the present noise is revealed then we take a decision.

$$\mathbf{x}_0 \rightsquigarrow \xi_1 \rightsquigarrow \mathbf{x}_1 \rightsquigarrow \xi_2 \rightsquigarrow \dots \rightsquigarrow \mathbf{x}_{T-1} \rightsquigarrow \xi_T \rightsquigarrow \mathbf{x}_T$$

Equivalent form

$$\min_{\mathbf{x}_1: \mathbf{A}_1 \mathbf{x}_1 + \mathbf{B}_1 \mathbf{x}_0 \leq \mathbf{b}_1} \mathbf{c}_1^\top \mathbf{x}_1 + \mathbb{E} \left[\min_{\mathbf{x}_2: \mathbf{A}_2 \mathbf{x}_2 + \mathbf{B}_2 \mathbf{x}_1 \leq \mathbf{b}_2} \mathbf{c}_2^\top \mathbf{x}_2 + \mathbb{E} \left[\dots + \mathbb{E} \left[\min_{\mathbf{x}_T: \mathbf{A}_T \mathbf{x}_T + \mathbf{B}_T \mathbf{x}_{T-1} \leq \mathbf{b}_T} \mathbf{c}_T^\top \mathbf{x}_T \right] \right] \right]$$

Multistage stochastic linear programming (MSLP)

$$\begin{aligned} \min_{(\mathbf{x}_t)_{t \in [T]}} \quad & \mathbb{E} \left[\sum_{t=1}^T \mathbf{c}_t^\top \mathbf{x}_t \right] \\ \text{s.t.} \quad & \mathbf{A}_t \mathbf{x}_t + \mathbf{B}_t \mathbf{x}_{t-1} \leq \mathbf{b}_t \quad \forall t \in [T] \\ & \sigma(\mathbf{x}_t) \subset \sigma(\mathbf{c}_\tau, \mathbf{A}_\tau, \mathbf{B}_\tau, \mathbf{b}_\tau)_{\tau \leq t} \quad \forall t \in [T] \\ & \mathbf{x}_0 \equiv \mathbf{x}_0 \text{ given} \end{aligned}$$

$\xi_t = (\mathbf{c}_t, \mathbf{A}_t, \mathbf{B}_t, \mathbf{b}_t)_{t \in [T]}$ is assumed to be **stagewise independent**.

At each time step: the present noise is revealed then we take a decision.

$$\mathbf{x}_0 \rightsquigarrow \xi_1 \rightsquigarrow \mathbf{x}_1 \rightsquigarrow \xi_2 \rightsquigarrow \cdots \rightsquigarrow \mathbf{x}_{T-1} \rightsquigarrow \xi_T \rightsquigarrow \mathbf{x}_T$$

Equivalent form

$$\min_{\mathbf{x}_1: \mathbf{A}_1 \mathbf{x}_1 + \mathbf{B}_1 \mathbf{x}_0 \leq \mathbf{b}_1} \mathbf{c}_1^\top \mathbf{x}_1 + \mathbb{E} \left[\min_{\mathbf{x}_2: \mathbf{A}_2 \mathbf{x}_2 + \mathbf{B}_2 \mathbf{x}_1 \leq \mathbf{b}_2} \mathbf{c}_2^\top \mathbf{x}_2 + \mathbb{E} \left[\cdots + \mathbb{E} \left[\min_{\mathbf{x}_T: \mathbf{A}_T \mathbf{x}_T + \mathbf{B}_T \mathbf{x}_{T-1} \leq \mathbf{b}_T} \mathbf{c}_T^\top \mathbf{x}_T \right] \right] \right]$$

Multistage stochastic linear programming (MSLP)

$$\begin{aligned} \min_{(\mathbf{x}_t)_{t \in [T]}} \quad & \mathbb{E} \left[\sum_{t=1}^T \mathbf{c}_t^\top \mathbf{x}_t \right] \\ \text{s.t.} \quad & \mathbf{A}_t \mathbf{x}_t + \mathbf{B}_t \mathbf{x}_{t-1} \leq \mathbf{b}_t \quad \forall t \in [T] \\ & \sigma(\mathbf{x}_t) \subset \sigma(\mathbf{c}_\tau, \mathbf{A}_\tau, \mathbf{B}_\tau, \mathbf{b}_\tau)_{\tau \leq t} \quad \forall t \in [T] \\ & \mathbf{x}_0 \equiv \mathbf{x}_0 \text{ given} \end{aligned}$$

$\xi_t = (\mathbf{c}_t, \mathbf{A}_t, \mathbf{B}_t, \mathbf{b}_t)_{t \in [T]}$ is assumed to be **stagewise independent**.

At each time step: the present noise is revealed then we take a decision.

$$\mathbf{x}_0 \rightsquigarrow \xi_1 \rightsquigarrow \mathbf{x}_1 \rightsquigarrow \xi_2 \rightsquigarrow \cdots \rightsquigarrow \mathbf{x}_{T-1} \rightsquigarrow \xi_T \rightsquigarrow \mathbf{x}_T$$

Equivalent form

$$\min_{\mathbf{x}_1: \mathbf{A}_1 \mathbf{x}_1 + \mathbf{B}_1 \mathbf{x}_0 \leq \mathbf{b}_1} \mathbf{c}_1^\top \mathbf{x}_1 + \mathbb{E} \left[\min_{\mathbf{x}_2: \mathbf{A}_2 \mathbf{x}_2 + \mathbf{B}_2 \mathbf{x}_1 \leq \mathbf{b}_2} \mathbf{c}_2^\top \mathbf{x}_2 + \mathbb{E} \left[\cdots + \mathbb{E} \left[\min_{\mathbf{x}_T: \mathbf{A}_T \mathbf{x}_T + \mathbf{B}_T \mathbf{x}_{T-1} \leq \mathbf{b}_T} \mathbf{c}_T^\top \mathbf{x}_T \right] \right] \right]$$

Multistage stochastic linear programming (MSLP)

$$\begin{aligned}
 \min_{(\mathbf{x}_t)_{t \in [T]}} \quad & \mathbb{E} \left[\sum_{t=1}^T \mathbf{c}_t^\top \mathbf{x}_t \right] \\
 \text{s.t.} \quad & \mathbf{A}_t \mathbf{x}_t + \mathbf{B}_t \mathbf{x}_{t-1} \leq \mathbf{b}_t \quad \forall t \in [T] \\
 & \sigma(\mathbf{x}_t) \subset \sigma(\mathbf{c}_\tau, \mathbf{A}_\tau, \mathbf{B}_\tau, \mathbf{b}_\tau)_{\tau \leq t} \quad \forall t \in [T] \\
 & \mathbf{x}_0 \equiv \mathbf{x}_0 \text{ given}
 \end{aligned}$$

$\xi_t = (\mathbf{c}_t, \mathbf{A}_t, \mathbf{B}_t, \mathbf{b}_t)_{t \in [T]}$ is assumed to be **stagewise independent**.

At each time step: the present noise is revealed then we take a decision.

$$x_0 \rightsquigarrow \xi_1 \rightsquigarrow x_1 \rightsquigarrow \xi_2 \rightsquigarrow \dots \rightsquigarrow x_{T-1} \rightsquigarrow \xi_T \rightsquigarrow x_T$$

Equivalent form

$$\min_{x_1: \mathbf{A}_1 x_1 + \mathbf{B}_1 x_0 \leq \mathbf{b}_1} \mathbf{c}_1^\top x_1 + \mathbb{E} \left[\min_{x_2: \mathbf{A}_2 x_2 + \mathbf{B}_2 x_1 \leq \mathbf{b}_2} \mathbf{c}_2^\top x_2 + \mathbb{E} \left[\dots + \mathbb{E} \left[\min_{x_T: \mathbf{A}_T x_T + \mathbf{B}_T x_{T-1} \leq \mathbf{b}_T} \mathbf{c}_T^\top x_T \right] \right] \right].$$

Multistage stochastic linear programming (MSLP)

$$\begin{aligned}
 \min_{(\mathbf{x}_t)_{t \in [T]}} \quad & \mathbb{E} \left[\sum_{t=1}^T \mathbf{c}_t^\top \mathbf{x}_t \right] \\
 \text{s.t.} \quad & \mathbf{A}_t \mathbf{x}_t + \mathbf{B}_t \mathbf{x}_{t-1} \leq \mathbf{b}_t & \forall t \in [T] \\
 & \sigma(\mathbf{x}_t) \subset \sigma(\mathbf{c}_\tau, \mathbf{A}_\tau, \mathbf{B}_\tau, \mathbf{b}_\tau)_{\tau \leq t} & \forall t \in [T] \\
 & \mathbf{x}_0 \equiv \mathbf{x}_0 \text{ given}
 \end{aligned}$$

$\xi_t = (\mathbf{c}_t, \mathbf{A}_t, \mathbf{B}_t, \mathbf{b}_t)_{t \in [T]}$ is assumed to be **stagewise independent**.

At each time step: the present noise is revealed then we take a decision.

$$\mathbf{x}_0 \rightsquigarrow \xi_1 \rightsquigarrow \mathbf{x}_1 \rightsquigarrow \xi_2 \rightsquigarrow \cdots \rightsquigarrow \mathbf{x}_{T-1} \rightsquigarrow \xi_T \rightsquigarrow \mathbf{x}_T$$

Equivalent form

$$\min_{\mathbf{x}_1: \mathbf{A}_1 \mathbf{x}_1 + \mathbf{B}_1 \mathbf{x}_0 \leq \mathbf{b}_1} \mathbf{c}_1^\top \mathbf{x}_1 + \mathbb{E} \left[\min_{\mathbf{x}_2: \mathbf{A}_2 \mathbf{x}_2 + \mathbf{B}_2 \mathbf{x}_1 \leq \mathbf{b}_2} \mathbf{c}_2^\top \mathbf{x}_2 + \mathbb{E} \left[\cdots + \mathbb{E} \left[\min_{\mathbf{x}_T: \mathbf{A}_T \mathbf{x}_T + \mathbf{B}_T \mathbf{x}_{T-1} \leq \mathbf{b}_T} \mathbf{c}_T^\top \mathbf{x}_T \right] \right] \right]$$

Dynamic Programming (Bellman 1966)

$$\min_{x_1: \mathbf{A}_1 x_1 + \mathbf{B}_1 x_0 \leq b_1} \mathbf{c}_1^\top x_1 + \mathbb{E} \left[\min_{x_2: \mathbf{A}_2 x_2 + \mathbf{B}_2 x_1 \leq b_2} \mathbf{c}_2^\top x_2 + \mathbb{E} \left[\cdots + \mathbb{E} \left[\min_{x_T: \mathbf{A}_T x_T + \mathbf{B}_T x_{T-1} \leq b_T} \mathbf{c}_T^\top x_T \right] \right] \right]$$

We set $V_{T+1} \equiv 0$ and $V_t(x_{t-1}) := \mathbb{E} \left[\begin{array}{ll} \min_{x_t \in \mathbb{R}^{n_t}} & \mathbf{c}_t^\top x_t + V_{t+1}(x_t) \\ \text{s.t.} & \mathbf{A}_t x_t + \mathbf{B}_t x_{t-1} \leq b_t \end{array} \right]$

Dynamic Programming (Bellman 1966)

$$\min_{x_1: \mathbf{A}_1 x_1 + \mathbf{B}_1 x_0 \leq b_1} \mathbf{c}_1^\top x_1 + \mathbb{E} \left[\min_{x_2: \mathbf{A}_2 x_2 + \mathbf{B}_2 x_1 \leq b_2} \mathbf{c}_2^\top x_2 + \underbrace{\mathbb{E} \left[\min_{x_T: \mathbf{A}_T x_T + \mathbf{B}_T x_{T-1} \leq b_T} \mathbf{c}_T^\top x_T \right]}_{V_T(x_{T-1})} \right]$$

We set $V_{T+1} \equiv 0$ and $V_t(x_{t-1}) := \mathbb{E} \left[\begin{array}{ll} \min_{x_t \in \mathbb{R}^{n_t}} & \mathbf{c}_t^\top x_t + V_{t+1}(x_t) \\ \text{s.t.} & \mathbf{A}_t x_t + \mathbf{B}_t x_{t-1} \leq b_t \end{array} \right]$

Dynamic Programming (Bellman 1966)

$$\min_{x_1: \mathbf{A}_1 x_1 + \mathbf{B}_1 x_0 \leq b_1} \mathbf{c}_1^\top x_1 + \mathbb{E} \left[\underbrace{\min_{x_2: \mathbf{A}_2 x_2 + \mathbf{B}_2 x_1 \leq b_2} \mathbf{c}_2^\top x_2 + \mathbb{E} \left[\underbrace{\dots + \mathbb{E} \left[\min_{x_T: \mathbf{A}_T x_T + \mathbf{B}_T x_{T-1} \leq b_T} \mathbf{c}_T^\top x_T \right]}_{V_T(x_{T-1})} \right]}_{V_3(x_2)} \right]$$

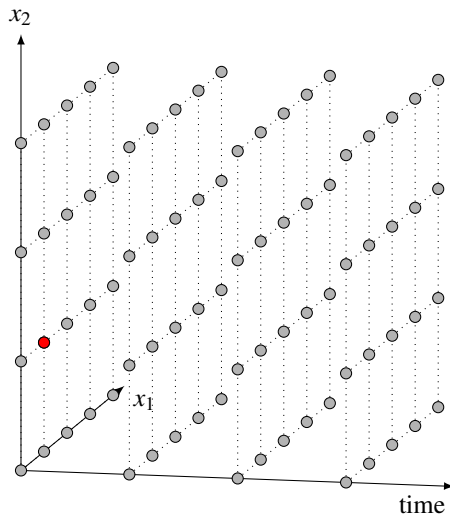
We set $V_{T+1} \equiv 0$ and $V_t(x_{t-1}) := \mathbb{E} \left[\begin{array}{ll} \min_{x_t \in \mathbb{R}^{n_t}} & \mathbf{c}_t^\top x_t + V_{t+1}(x_t) \\ \text{s.t.} & \mathbf{A}_t x_t + \mathbf{B}_t x_{t-1} \leq b_t \end{array} \right]$

Dynamic Programming (Bellman 1966)

$$\min_{x_1: \mathbf{A}_1 x_1 + \mathbf{B}_1 x_0 \leq b_1} \mathbf{c}_1^\top x_1 + \underbrace{\mathbb{E} \left[\underbrace{\min_{x_2: \mathbf{A}_2 x_2 + \mathbf{B}_2 x_1 \leq b_2} \mathbf{c}_2^\top x_2 + \underbrace{\mathbb{E} \left[\dots + \underbrace{\mathbb{E} \left[\min_{x_T: \mathbf{A}_T x_T + \mathbf{B}_T x_{T-1} \leq b_T} \mathbf{c}_T^\top x_T \right]}_{V_T(x_{T-1})} \right]}_{V_3(x_2)} \right]}_{V_2(x_1)} \right]}_{V_2(x_1)}$$

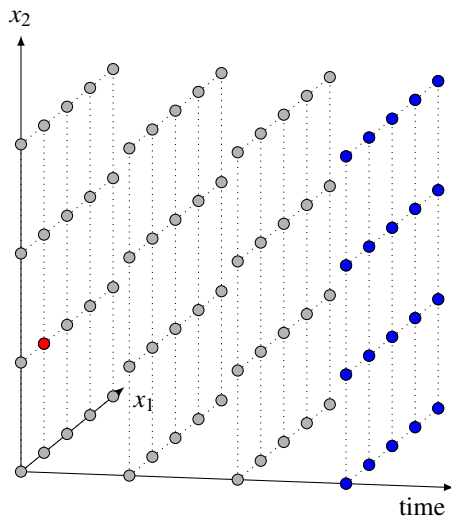
We set $V_{T+1} \equiv 0$ and $V_t(x_{t-1}) := \mathbb{E} \left[\begin{array}{ll} \min_{x_t \in \mathbb{R}^{n_t}} & \mathbf{c}_t^\top x_t + V_{t+1}(x_t) \\ \text{s.t.} & \mathbf{A}_t x_t + \mathbf{B}_t x_{t-1} \leq b_t \end{array} \right]$

Dynamic programming : finite case

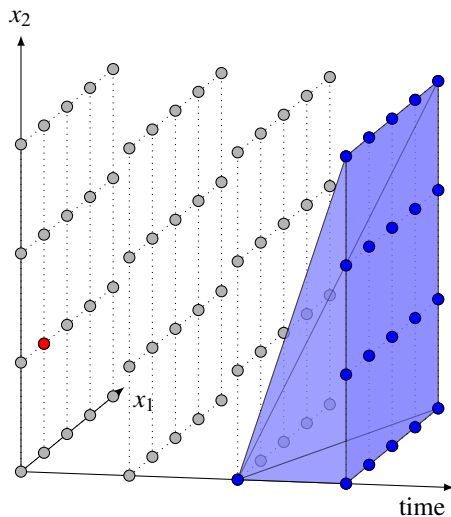


Thank you Vincent for this animation.

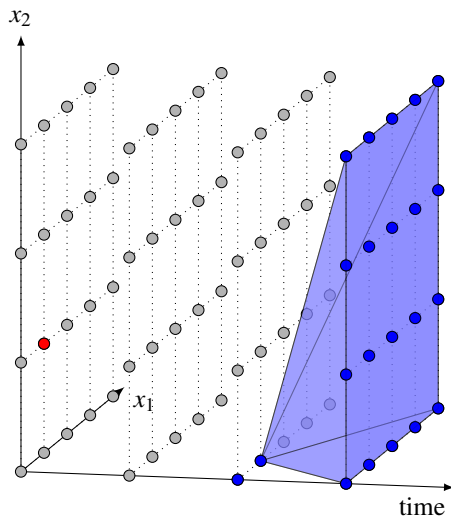
Dynamic programming : finite case



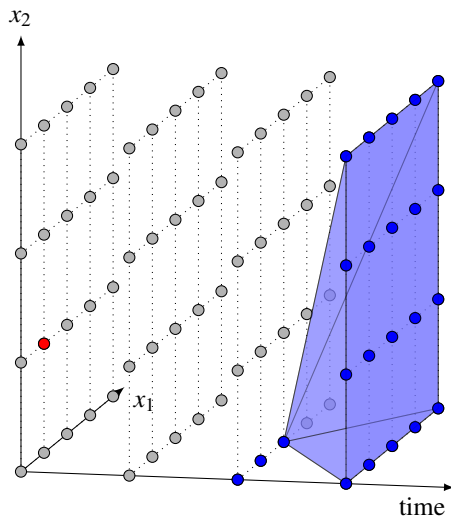
Dynamic programming : finite case



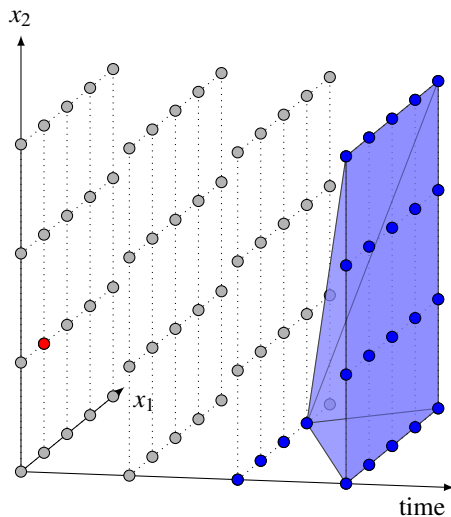
Dynamic programming : finite case



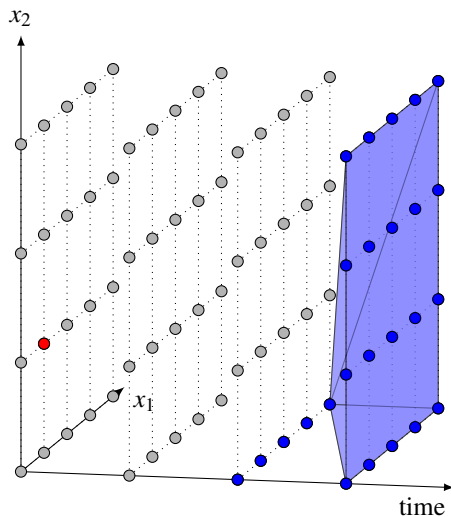
Dynamic programming : finite case



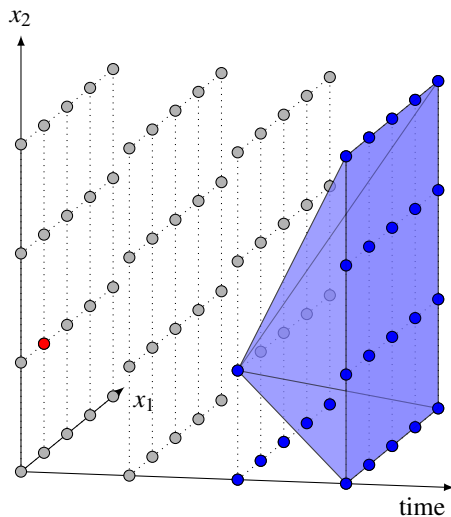
Dynamic programming : finite case



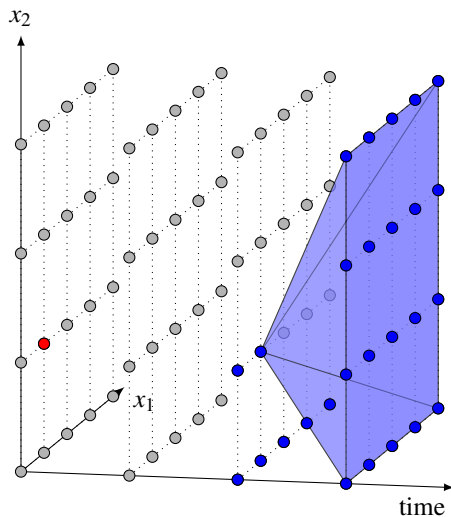
Dynamic programming : finite case



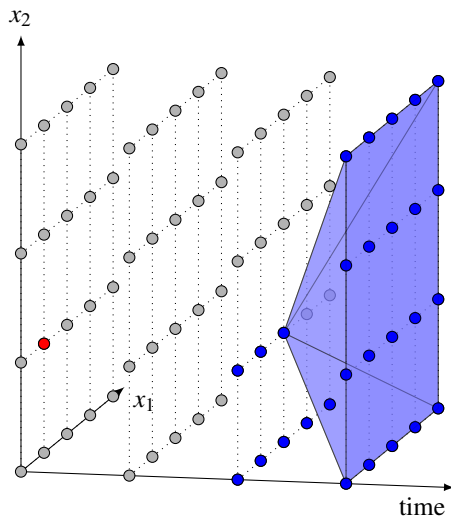
Dynamic programming : finite case



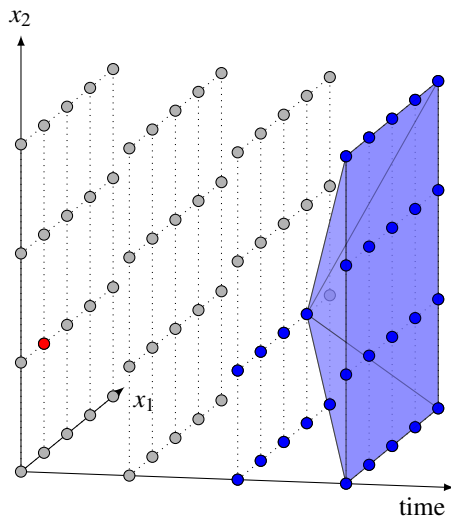
Dynamic programming : finite case



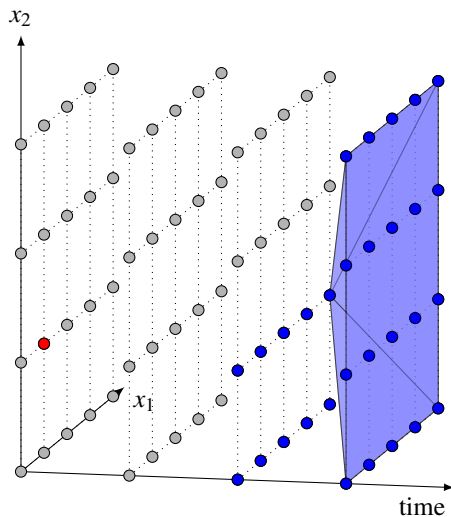
Dynamic programming : finite case



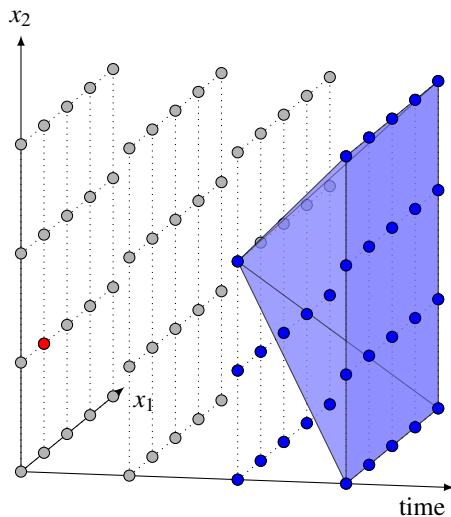
Dynamic programming : finite case



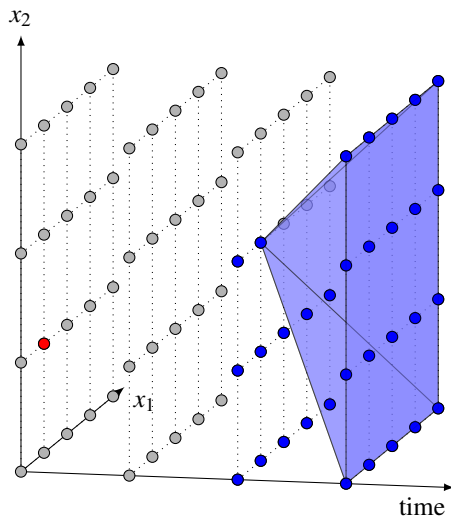
Dynamic programming : finite case



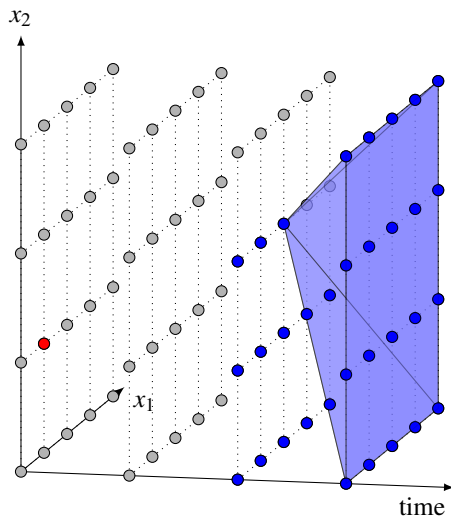
Dynamic programming : finite case



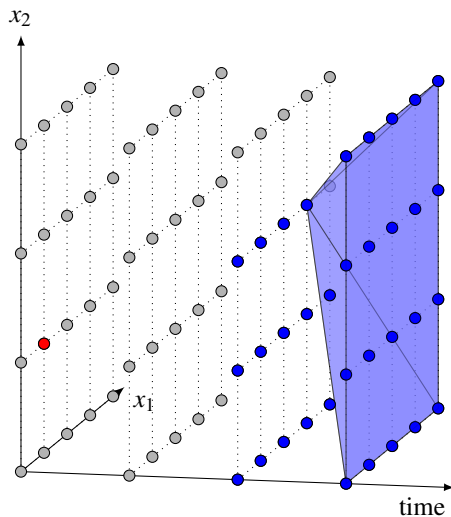
Dynamic programming : finite case



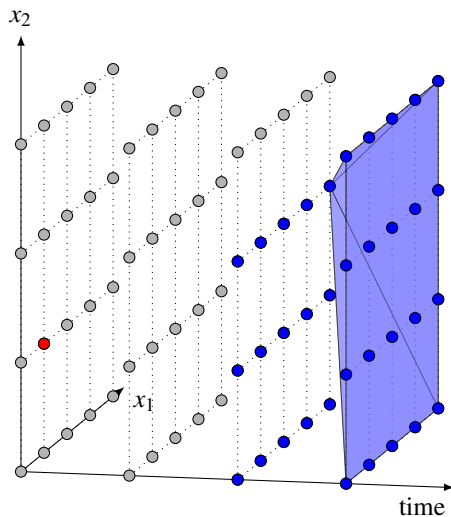
Dynamic programming : finite case



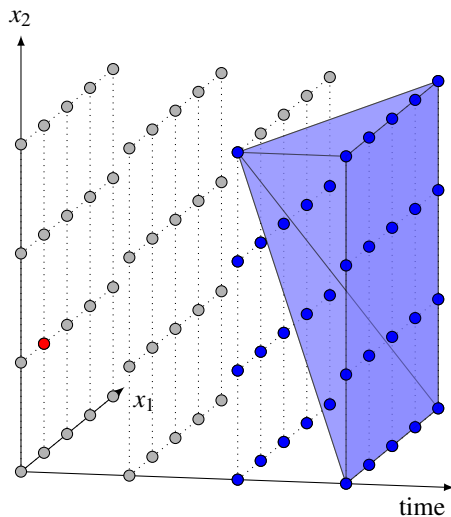
Dynamic programming : finite case



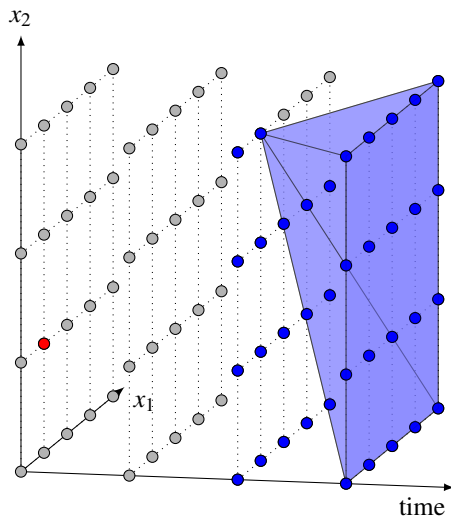
Dynamic programming : finite case



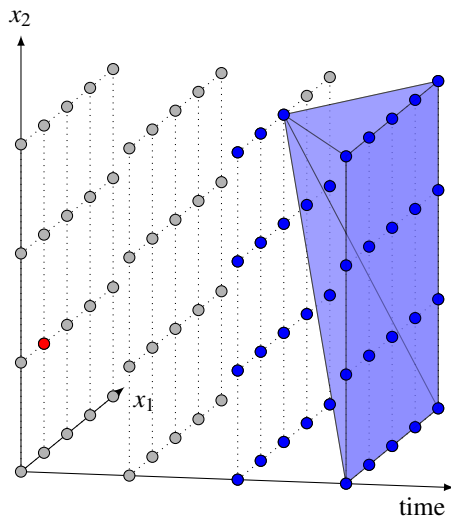
Dynamic programming : finite case



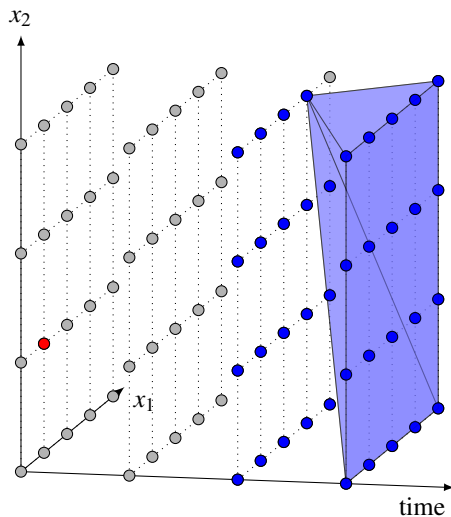
Dynamic programming : finite case



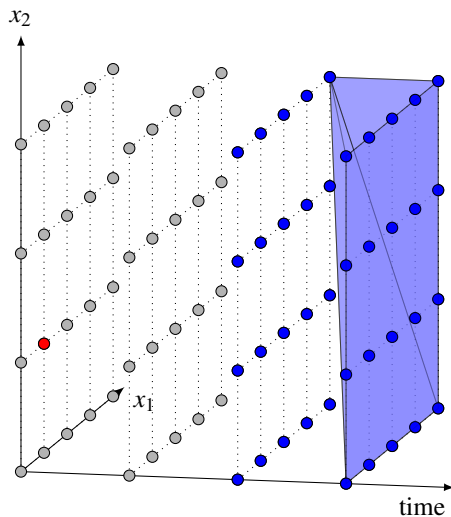
Dynamic programming : finite case



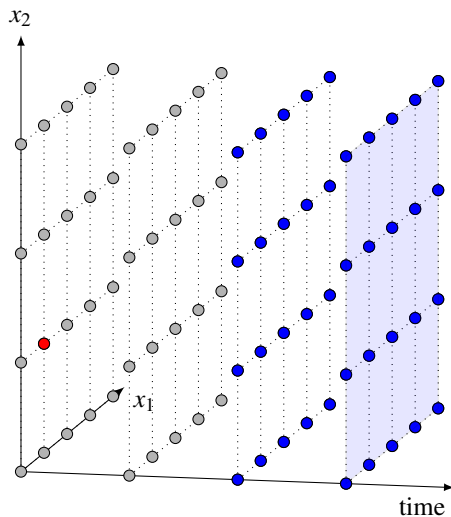
Dynamic programming : finite case



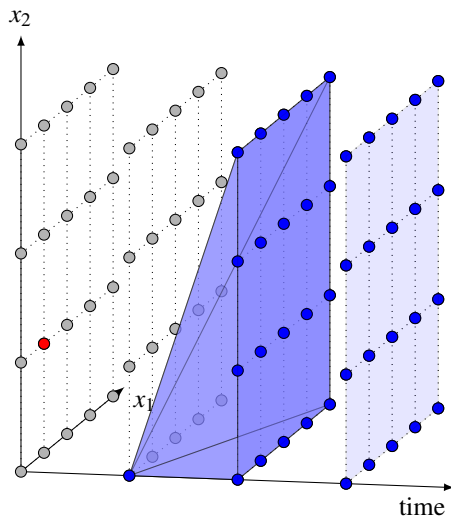
Dynamic programming : finite case



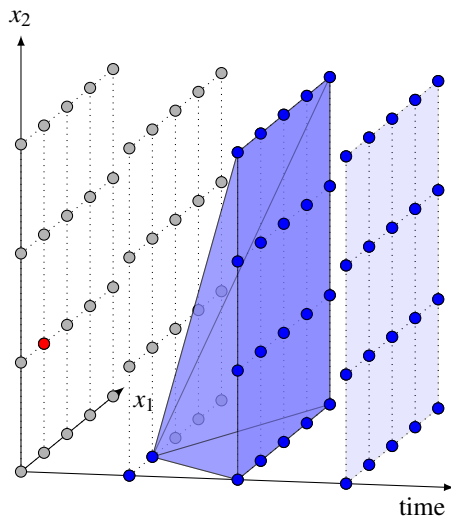
Dynamic programming : finite case



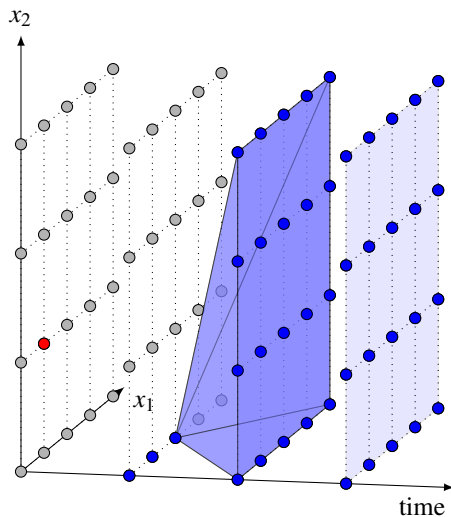
Dynamic programming : finite case



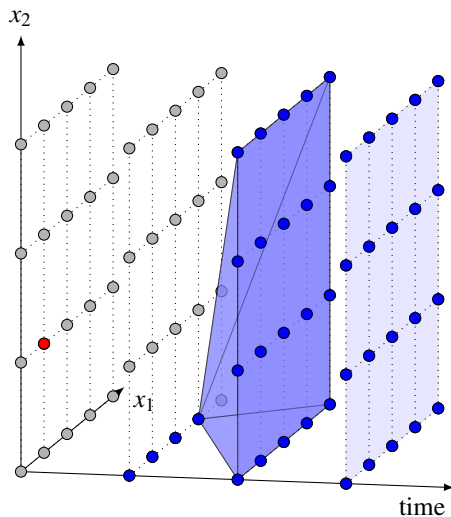
Dynamic programming : finite case



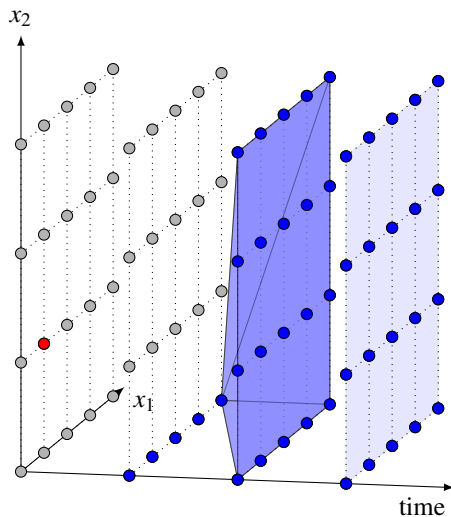
Dynamic programming : finite case



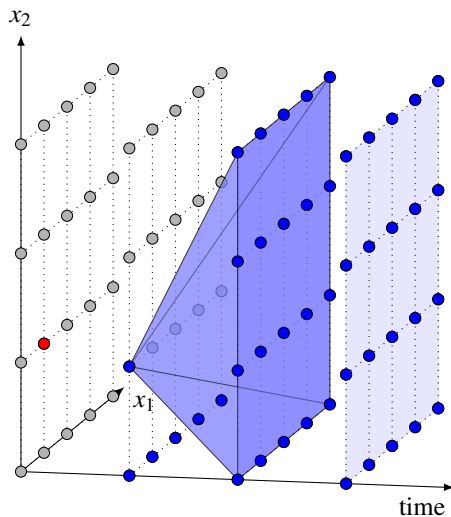
Dynamic programming : finite case



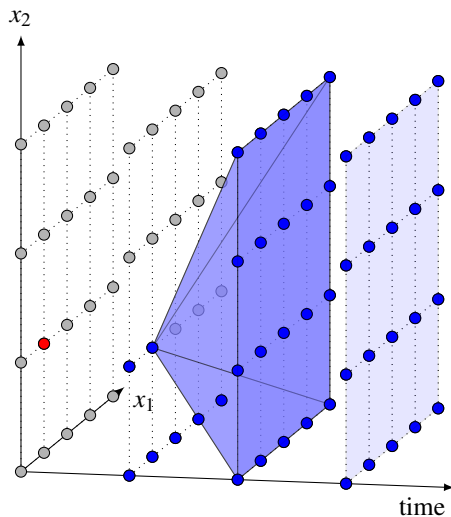
Dynamic programming : finite case



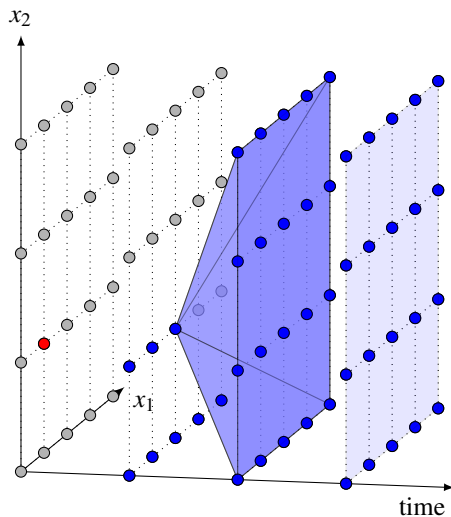
Dynamic programming : finite case



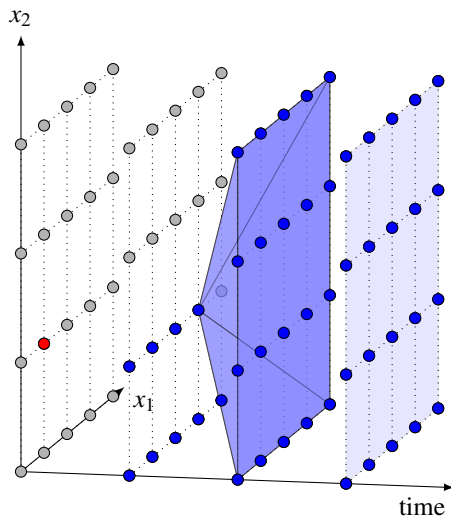
Dynamic programming : finite case



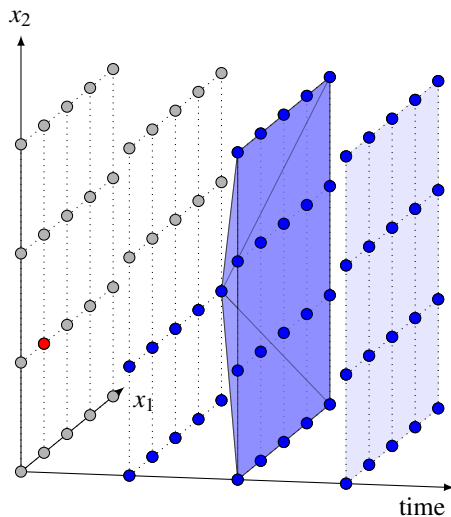
Dynamic programming : finite case



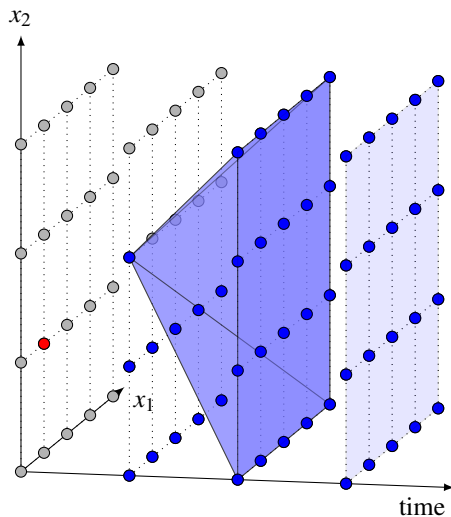
Dynamic programming : finite case



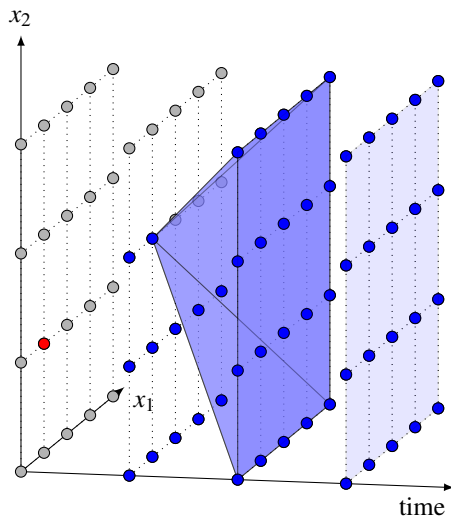
Dynamic programming : finite case



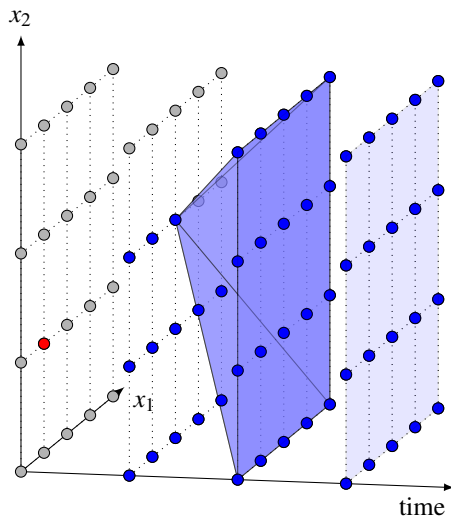
Dynamic programming : finite case



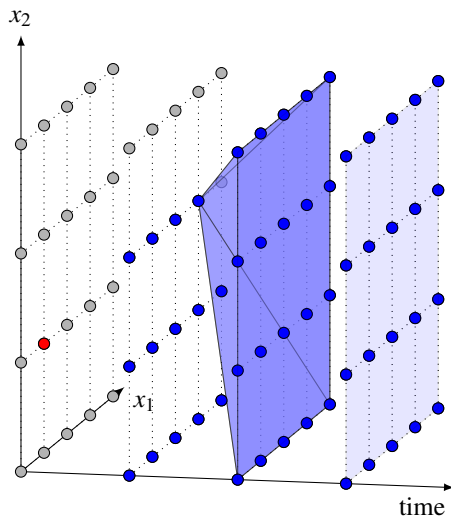
Dynamic programming : finite case



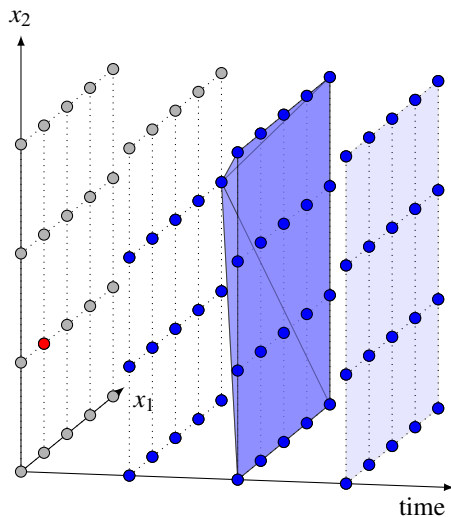
Dynamic programming : finite case



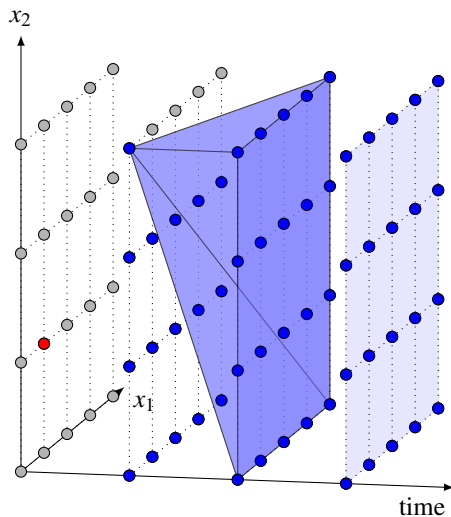
Dynamic programming : finite case



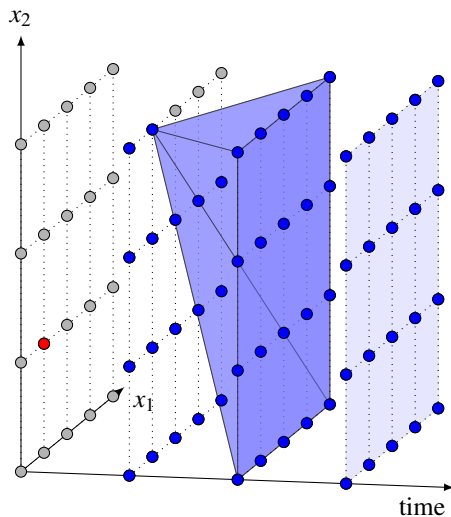
Dynamic programming : finite case



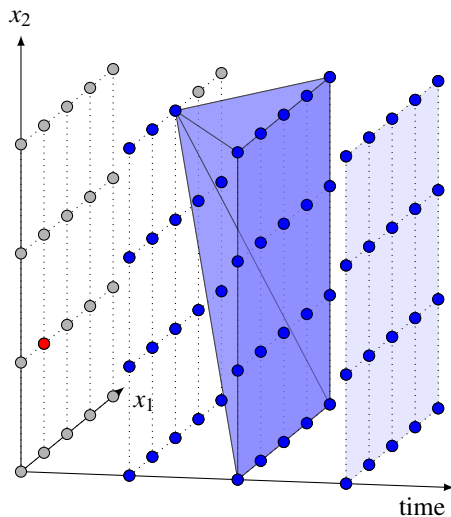
Dynamic programming : finite case



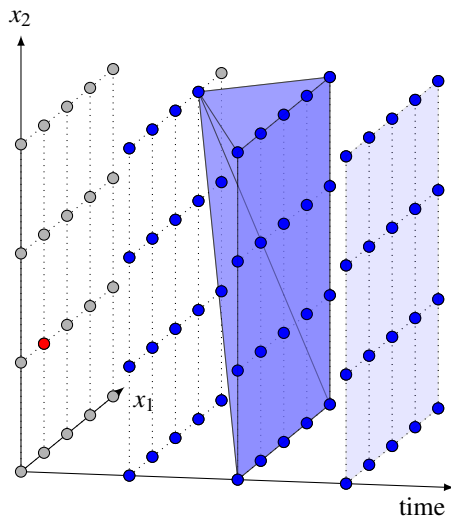
Dynamic programming : finite case



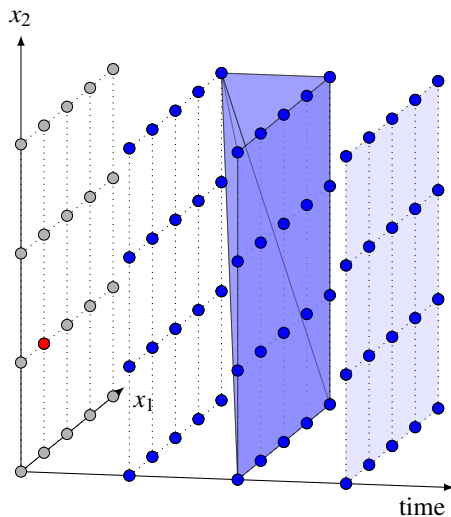
Dynamic programming : finite case



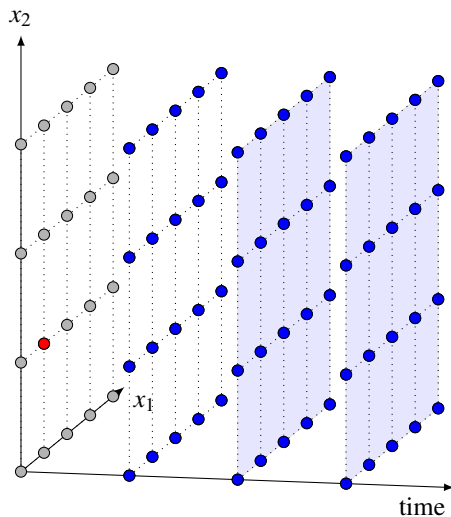
Dynamic programming : finite case



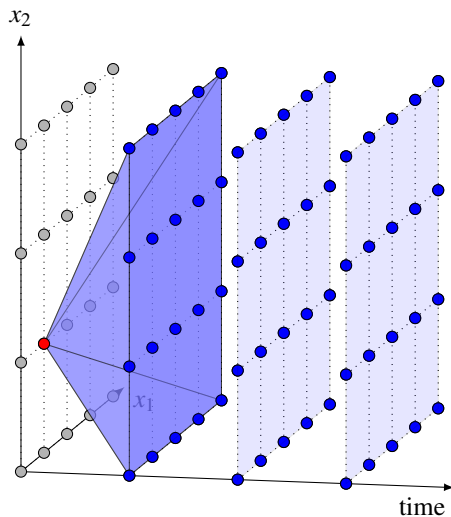
Dynamic programming : finite case



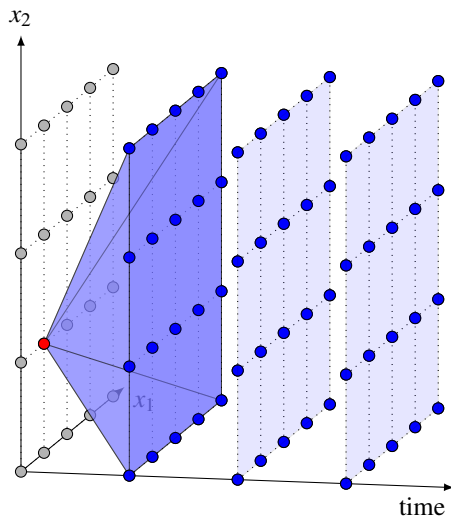
Dynamic programming : finite case



Dynamic programming : finite case

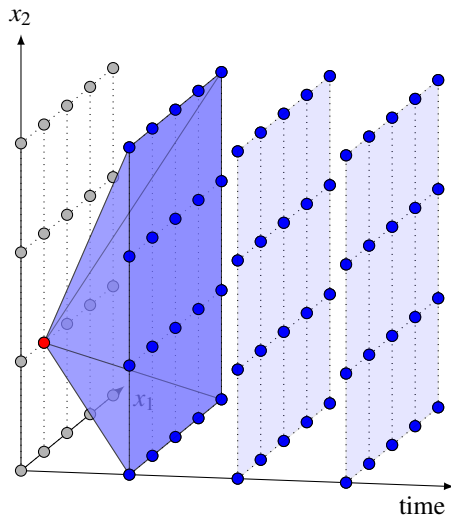


Dynamic programming : finite case



➡ Continuous space : algorithms such as SDDP later discussed.

Dynamic programming : finite case

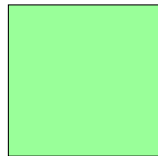


- ➡ Continuous space : algorithms such as SDDP later discussed.
- ➡ How to deal with continuous distributions ?

Quantization of a MSLP

Real problem

$$V_t(x) = \mathbb{E}[\hat{V}_t(x, \xi_t)] = \mathbb{E} \left[\begin{array}{ll} \min_{y \in \mathbb{R}^{n_t}} & \mathbf{c}_t^\top y + V_{t+1}(y) \\ \text{s.t.} & \mathbf{A}_t y + \mathbf{B}_t x \leq \mathbf{b}_t \end{array} \right]$$

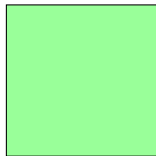


ξ_t continuous

Quantization of a MSLP

Real problem

$$V_t(x) = \mathbb{E}[\hat{V}_t(x, \xi_t)] = \mathbb{E} \left[\begin{array}{ll} \min_{y \in \mathbb{R}^{n_t}} & \mathbf{c}_t^\top y + V_{t+1}(y) \\ \text{s.t.} & \mathbf{A}_t y + \mathbf{B}_t x \leq \mathbf{b}_t \end{array} \right]$$

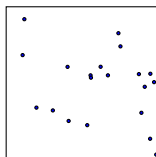


ξ_t continuous

Sample Average Approximation (SAA)

$$V_{t,N}^{SAA}(x) := \frac{1}{N} \sum_{k=1}^N \hat{V}_t(x, \xi^k)$$

ξ^1, \dots, ξ^N drawn by Monte Carlo (ex Shapiro 2011)

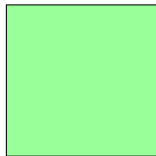


SAA $N = 20$

Quantization of a MSLP

Real problem

$$V_t(x) = \mathbb{E}[\hat{V}_t(x, \xi_t)] = \mathbb{E} \left[\begin{array}{ll} \min_{y \in \mathbb{R}^{n_t}} & \mathbf{c}_t^\top y + V_{t+1}(y) \\ \text{s.t.} & \mathbf{A}_t y + \mathbf{B}_t x \leq \mathbf{b}_t \end{array} \right]$$

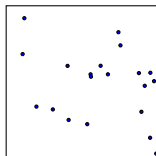


ξ_t continuous

Sample Average Approximation (SAA)

$$V_{t,N}^{SAA}(x) := \frac{1}{N} \sum_{k=1}^N \hat{V}_t(x, \xi^k)$$

ξ^1, \dots, ξ^N drawn by Monte Carlo (ex Shapiro 2011)

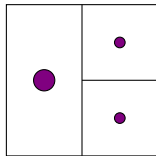


SAA $N = 20$

Partition-based

$$V_{t,P}(x) := \sum_{P \in \mathcal{P}} \check{p}_{t,P} \hat{V}_t(x, \check{\xi}_{t,P})$$

with $\check{p}_{t,P} := \mathbb{P}[\xi_t \in P]$ and $\check{\xi}_{t,P} := \mathbb{E}[\xi_t | \xi_t \in P]$

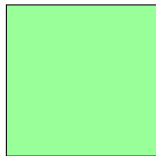


Partition-based

Quantization of a MSLP

Real problem

$$V_t(x) = \mathbb{E}[\hat{V}_t(x, \xi_t)] = \mathbb{E} \left[\begin{array}{ll} \min_{y \in \mathbb{R}^{n_t}} & \mathbf{c}_t^\top y + V_{t+1}(y) \\ \text{s.t.} & \mathbf{A}_t y + \mathbf{B}_t x \leq \mathbf{b}_t \end{array} \right]$$

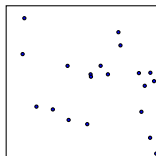


ξ_t continuous

Sample Average Approximation (SAA)

$$V_{t,N}^{SAA}(x) := \frac{1}{N} \sum_{k=1}^N \hat{V}_t(x, \xi^k)$$

ξ^1, \dots, ξ^N drawn by Monte Carlo (ex Shapiro 2011)



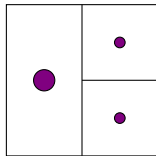
SAA $N = 20$

Partition-based

$$V_{t,\mathcal{P}}(x) := \sum_{P \in \mathcal{P}} \check{p}_{t,P} \hat{V}_t(x, \check{\xi}_{t,P})$$

with $\check{p}_{t,P} := \mathbb{P}[\xi_t \in P]$ and $\check{\xi}_{t,P} := \mathbb{E}[\xi_t | \xi_t \in P]$

If $\xi \mapsto \hat{V}(x, \xi)$ is convex, $V_{t,\mathcal{P}}(x) \leq V_t(x)$ (Jensen, Kuhn) Partition-based



Exact quantization

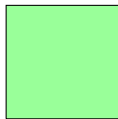
Definition

A MSLP admits a **local exact quantization** at time t on x if there exists a finitely supported $(\check{\xi}_t)_{t \in [T]}$ such that

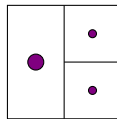
$$V_t(x) = \mathbb{E}[\hat{V}_t(x, \xi_t)] = \mathbb{E}[\hat{V}_t(x, \check{\xi}_t)].$$

We call an exact quantization

- **uniform** if it is locally exact at all $x \in \mathbb{R}^{n_t}$, and all $t \in [T]$.
- **universal** if there exists a partition $\mathcal{P}_{t,x}$ such that the induced quantization is exact at time t on x , for all distributions of $(\xi_\tau)_{\tau \in [T]}$.



ξ_t continuous



$\check{\xi}_t$ quantized

Conditions for the existence of an exact quantization ?

Assume $V_{t+1} \equiv 0$ and denote $V := V_t$, $\hat{V} := \hat{V}_t$ and $\xi := \xi_t$ for now.

$$V(x) = \mathbb{E}[\hat{V}(x, \xi)] = \mathbb{E} \left[\begin{array}{ll} \min_{y \in \mathbb{R}^n} & \mathbf{c}^\top y \\ \text{s.t.} & \mathbf{A}y + \mathbf{B}x \leq \mathbf{b} \end{array} \right]$$

We have an exact quantization if and only if there exists a finitely supported noise $\check{\xi}$ such that

$$\mathbb{E}[\hat{V}(x, \xi)] = \mathbb{E}[\hat{V}(x, \check{\xi})].$$

	\mathbf{A}	(\mathbf{B}, \mathbf{b})	\mathbf{c}
Local	?	?	?
Uniform	?	?	?

A first counter example

	\mathbf{A}	(\mathbf{B}, \mathbf{b})	\mathbf{c}
Local	?	?	?
Uniform	?	?	?

Let $\mathbf{A} = (-\mathbf{u})$, $\mathbf{B} \equiv (0)$, $\mathbf{b} \equiv (-1)$ where $\mathbf{u} \sim \mathcal{U}([1, 2])$.

$$\hat{V}(x, \xi) = \min_{y \in \mathbb{R}} y \quad \text{s.t.} \quad \mathbf{u}y \geq 1 \quad = \frac{1}{\mathbf{u}}$$

By strict convexity, for all partition \mathcal{P}

$$\sum_{P \in \mathcal{P}} \check{p}_P \hat{V}(x, \check{\xi}_P) < V(x) = \mathbb{E} \left[\frac{1}{\mathbf{u}} \right]$$

with $\check{p}_P = \mathbb{P}[\xi \in P]$, $\check{\xi}_P = \mathbb{E}[\xi | \xi \in P]$.

- ➡ There is no partition-based (local, uniform or universal) exact quantization result for \mathbf{A} non-finitely supported.
- ➡ For now on, \mathbf{A} is deterministic: fixed recourse.

A first counter example

	\mathbf{A}	(\mathbf{B}, \mathbf{b})	\mathbf{c}
Local	?	?	?
Uniform	?	?	?

Let $\mathbf{A} = (-\mathbf{u})$, $\mathbf{B} \equiv (0)$, $\mathbf{b} \equiv (-1)$ where $\mathbf{u} \sim \mathcal{U}([1, 2])$.

$$\hat{V}(x, \xi) = \min_{y \in \mathbb{R}} y \quad \text{s.t.} \quad \mathbf{u}y \geq 1 \quad = \frac{1}{\mathbf{u}}$$

By strict convexity, for all partition \mathcal{P}

$$\sum_{P \in \mathcal{P}} \check{p}_P \hat{V}(x, \check{\xi}_P) < V(x) = \mathbb{E} \left[\frac{1}{\mathbf{u}} \right]$$

with $\check{p}_P = \mathbb{P}[\xi \in P]$, $\check{\xi}_P = \mathbb{E}[\xi | \xi \in P]$.

- ➡ There is no partition-based (local, uniform or universal) exact quantization result for \mathbf{A} non-finitely supported.
- ➡ For now on, \mathbf{A} is deterministic: fixed recourse.

A first counter example

	\mathbf{A}	(\mathbf{B}, \mathbf{b})	\mathbf{c}
Local	?	?	?
Uniform	?	?	?

Let $\mathbf{A} = (-\mathbf{u})$, $\mathbf{B} \equiv (0)$, $\mathbf{b} \equiv (-1)$ where $\mathbf{u} \sim \mathcal{U}([1, 2])$.

$$\hat{V}(x, \xi) = \min_{y \in \mathbb{R}} y \quad \text{s.t.} \quad \mathbf{u}y \geq 1 \quad = \frac{1}{\mathbf{u}}$$

By strict convexity, for all partition \mathcal{P}

$$\sum_{P \in \mathcal{P}} \check{p}_P \hat{V}(x, \check{\xi}_P) < V(x) = \mathbb{E} \left[\frac{1}{\mathbf{u}} \right]$$

with $\check{p}_P = \mathbb{P}[\xi \in P]$, $\check{\xi}_P = \mathbb{E}[\xi | \xi \in P]$.

➡ There is no partition-based (local, uniform or universal) exact quantization result for \mathbf{A} non-finitely supported.

➡ For now on, \mathbf{A} is deterministic: fixed recourse.

A first counter example

	\mathbf{A}	(\mathbf{B}, \mathbf{b})	\mathbf{c}
Local	✗	?	?
Uniform	✗	?	?

Let $\mathbf{A} = (-\mathbf{u})$, $\mathbf{B} \equiv (0)$, $\mathbf{b} \equiv (-1)$ where $\mathbf{u} \sim \mathcal{U}([1, 2])$.

$$\hat{V}(x, \xi) = \min_{y \in \mathbb{R}} y \quad \text{s.t.} \quad \mathbf{u}y \geq 1 \quad = \frac{1}{\mathbf{u}}$$

By strict convexity, for all partition \mathcal{P}

$$\sum_{P \in \mathcal{P}} \check{p}_P \hat{V}(x, \check{\xi}_P) < V(x) = \mathbb{E} \left[\frac{1}{\mathbf{u}} \right]$$

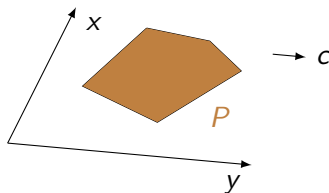
with $\check{p}_P = \mathbb{P}[\xi \in P]$, $\check{\xi}_P = \mathbb{E}[\xi | \xi \in P]$.

- ➡ There is no partition-based (local, uniform or universal) exact quantization result for \mathbf{A} non-finitely supported.
- ➡ For now on, \mathbf{A} is deterministic: fixed recourse.

Uniform exact quantization and polyhedrality

$$\hat{V}(x, \xi) = \min_{y \in \mathbb{R}^m} c^\top y$$

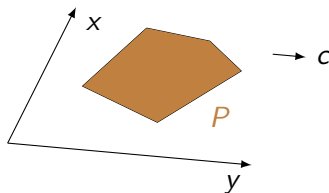
s.t. $Ay + Bx \leq b$



Uniform exact quantization and polyhedrality

$$\hat{V}(x, \xi) = \min_{y \in \mathbb{R}^m} c^\top y$$

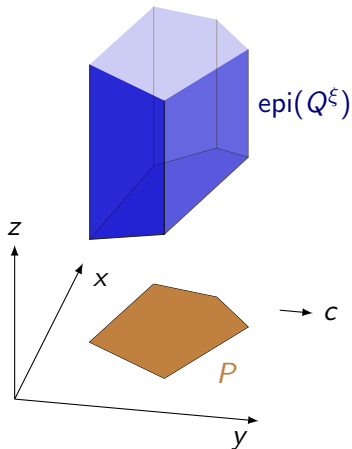
s.t. $(x, y) \in P$



Uniform exact quantization and polyhedrality

$$\begin{aligned}\hat{V}(x, \xi) &= \min_{y \in \mathbb{R}^m} c^\top y \\ &\text{s.t. } (x, y) \in P \\ &= \min_{y \in \mathbb{R}^m} Q^\xi(x, y)\end{aligned}$$

with $Q^\xi(x, y) := c^\top y + \mathbb{I}_{(x, y) \in P}$.

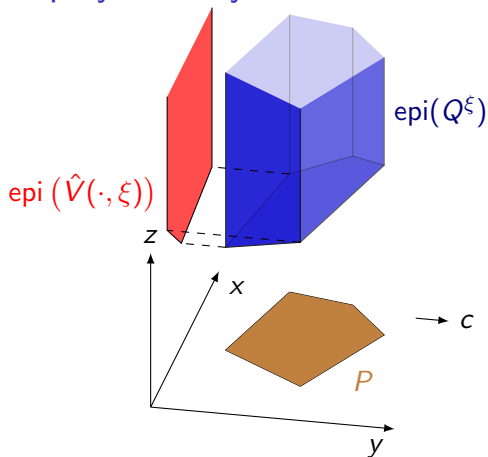


Uniform exact quantization and polyhedrality

$$\begin{aligned}\hat{V}(x, \xi) &= \min_{y \in \mathbb{R}^m} c^\top y \\ \text{s.t. } & (x, y) \in P \\ &= \min_{y \in \mathbb{R}^m} Q^\xi(x, y)\end{aligned}$$

with $Q^\xi(x, y) := c^\top y + \mathbb{I}_{(x, y) \in P}$.

$\hat{V}(\cdot, \xi)$ is polyhedral because
 $\text{epi}(\hat{V}(\cdot, \xi))$ is the projection of
 $\text{epi}(Q^\xi)$.

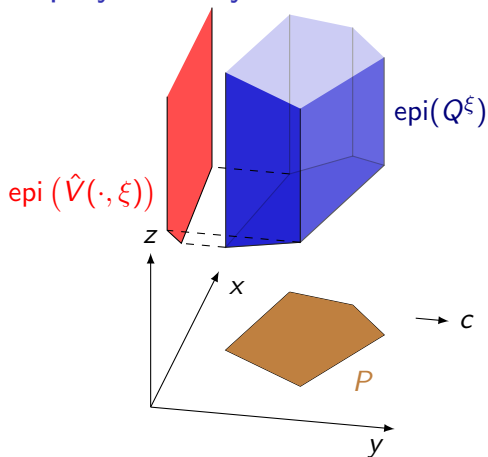


Uniform exact quantization and polyhedrality

$$\begin{aligned}\hat{V}(x, \xi) &= \min_{y \in \mathbb{R}^m} c^\top y \\ \text{s.t. } & (x, y) \in P \\ &= \min_{y \in \mathbb{R}^m} Q^\xi(x, y)\end{aligned}$$

with $Q^\xi(x, y) := c^\top y + \mathbb{I}_{(x, y) \in P}$.

$\hat{V}(\cdot, \xi)$ is polyhedral because
 $\text{epi}(\hat{V}(\cdot, \xi))$ is the projection of
 $\text{epi}(Q^\xi)$.



$$V(x) = \mathbb{E}[\hat{V}(x, \xi)] = \sum_{\xi \in \text{supp}(\xi)} p_\xi \hat{V}(x, \xi)$$

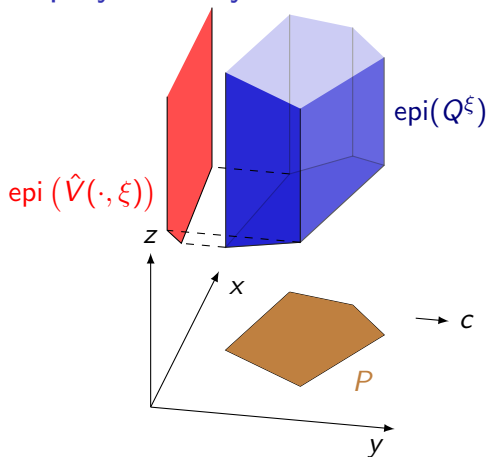
➡ If the noise is finitely supported, then V is polyhedral

Uniform exact quantization and polyhedrality

$$\begin{aligned}\hat{V}(x, \xi) &= \min_{y \in \mathbb{R}^m} c^\top y \\ \text{s.t. } & (x, y) \in P \\ &= \min_{y \in \mathbb{R}^m} Q^\xi(x, y)\end{aligned}$$

with $Q^\xi(x, y) := c^\top y + \mathbb{I}_{(x, y) \in P}$.

$\hat{V}(\cdot, \xi)$ is polyhedral because
 $\text{epi}(\hat{V}(\cdot, \xi))$ is the projection of
 $\text{epi}(Q^\xi)$.



$$V(x) = \mathbb{E}[\hat{V}(x, \xi)] = \sum_{\xi \in \text{supp}(\check{\xi})} p_\xi \hat{V}(x, \xi)$$

- ➡ If the noise is finitely supported, then V is polyhedral
- ➡ Existence of uniform exact quantization implies polyhedrality of V .

Counter examples with stochastic constraints

	A	(B, b)	c
Local	×	?	?
Uniform	×	?	?

u is uniform on $[0, 1]$

Counter examples with stochastic constraints

	A	(B, b)	c
Local	×	?	?
Uniform	×	?	?

$$\begin{aligned} \text{Stochastic } \mathbf{B} \quad & V(x) = \mathbb{E} \left[\begin{array}{l} \min_{y \in \mathbb{R}^m} y \\ \text{s.t. } \mathbf{u}x - y \leq 0 \\ y \geq 1 \end{array} \right] \\ &= \mathbb{E} [\max(\mathbf{u}x, 1)] \\ &= \begin{cases} 1 & \text{if } x \leq 1 \\ \frac{x}{2} + \frac{1}{2x} & \text{if } x \geq 1 \end{cases} \end{aligned}$$

\mathbf{u} is uniform on $[0, 1]$

Counter examples with stochastic constraints

	A	(B, b)	c
Local	×	?	?
Uniform	×	?	?

$$\begin{aligned}
 \text{Stochastic } \mathbf{B} \quad & V(x) = \mathbb{E} \left[\begin{array}{l} \min_{y \in \mathbb{R}^m} y \\ \text{s.t. } \mathbf{u}x - y \leq 0 \\ y \geq 1 \end{array} \right] \\
 &= \mathbb{E} [\max(\mathbf{u}x, 1)] \\
 &= \begin{cases} 1 & \text{if } x \leq 1 \\ \frac{x}{2} + \frac{1}{2x} & \text{if } x \geq 1 \end{cases}
 \end{aligned}$$

$$\begin{aligned}
 \text{Stochastic } \mathbf{b} \quad & V(x) = \mathbb{E} \left[\begin{array}{l} \min_{y \in \mathbb{R}^m} y \\ \text{s.t. } y \geq \mathbf{u} \\ x - y \leq 0 \end{array} \right] \\
 &= \mathbb{E} [\max(x, \mathbf{u})] \\
 &= \begin{cases} \frac{1}{2} & \text{if } x \leq 0 \\ \frac{x^2+1}{2} & \text{if } x \in [0, 1] \\ x & \text{if } x \geq 1 \end{cases}
 \end{aligned}$$

\mathbf{u} is uniform on $[0, 1]$

Counter examples with stochastic constraints

	A	(B, b)	c
Local	×	?	?
Uniform	×	?	?

$$\begin{aligned}
 \text{Stochastic } \mathbf{B} \quad & V(x) = \mathbb{E} \left[\begin{array}{l} \min_{y \in \mathbb{R}^m} y \\ \text{s.t. } \mathbf{u}x - y \leq 0 \\ y \geq 1 \end{array} \right] \\
 &= \mathbb{E} [\max(\mathbf{u}x, 1)] \\
 &= \begin{cases} 1 & \text{if } x \leq 1 \\ \frac{x}{2} + \frac{1}{2x} & \text{if } x \geq 1 \end{cases}
 \end{aligned}$$

$$\begin{aligned}
 \text{Stochastic } \mathbf{b} \quad & V(x) = \mathbb{E} \left[\begin{array}{l} \min_{y \in \mathbb{R}^m} y \\ \text{s.t. } y \geq \mathbf{u} \\ x - y \leq 0 \end{array} \right] \\
 &= \mathbb{E} [\max(x, \mathbf{u})] \\
 &= \begin{cases} \frac{1}{2} & \text{if } x \leq 0 \\ \frac{x^2+1}{2} & \text{if } x \in [0, 1] \\ x & \text{if } x \geq 1 \end{cases}
 \end{aligned}$$

➡ V is not polyhedral \Rightarrow No uniform exact quantization for non-finitely supported \mathbf{B} and \mathbf{b} .

\mathbf{u} is uniform on $[0, 1]$

Counter examples with stochastic constraints

	\mathbf{A}	(\mathbf{B}, \mathbf{b})	\mathbf{c}
Local	×	?	?
Uniform	×	✗	?

$$\begin{aligned}
 \text{Stochastic } \mathbf{B} \quad & V(x) = \mathbb{E} \left[\begin{array}{l} \min_{y \in \mathbb{R}^m} y \\ \text{s.t. } \mathbf{u}x - y \leq 0 \\ y \geq 1 \end{array} \right] \\
 &= \mathbb{E} [\max(\mathbf{u}x, 1)] \\
 &= \begin{cases} 1 & \text{if } x \leq 1 \\ \frac{x}{2} + \frac{1}{2x} & \text{if } x \geq 1 \end{cases}
 \end{aligned}$$

$$\begin{aligned}
 \text{Stochastic } \mathbf{b} \quad & V(x) = \mathbb{E} \left[\begin{array}{l} \min_{y \in \mathbb{R}^m} y \\ \text{s.t. } y \geq \mathbf{u} \\ x - y \leq 0 \end{array} \right] \\
 &= \mathbb{E} [\max(x, \mathbf{u})] \\
 &= \begin{cases} \frac{1}{2} & \text{if } x \leq 0 \\ \frac{x^2+1}{2} & \text{if } x \in [0, 1] \\ x & \text{if } x \geq 1 \end{cases}
 \end{aligned}$$

➡ V is not polyhedral \Rightarrow No **uniform** exact quantization for non-finitely supported \mathbf{B} and \mathbf{b} .

\mathbf{u} is uniform on $[0, 1]$

Remaining cases

$$V(x) = \mathbb{E} \left[\begin{array}{ll} \min_{y \in \mathbb{R}^m} & \mathbf{c}^\top y \\ \text{s.t.} & \mathbf{A}y + \mathbf{B}x \leq \mathbf{b} \end{array} \right]$$

	\mathbf{A}	(\mathbf{B}, \mathbf{b})	\mathbf{c}
Local	×	?	?
Uniform	×	×	?

Remaining cases

$$V(x) = \mathbb{E} \left[\begin{array}{ll} \min_{y \in \mathbb{R}^m} & \mathbf{c}^\top y \\ \text{s.t.} & \mathbf{A}y + \mathbf{B}x \leq \mathbf{b} \end{array} \right]$$

	\mathbf{A}	(\mathbf{B}, \mathbf{b})	\mathbf{c}
Local	×	?	✓
Uniform	×	×	✓

Theorem (FGL 2021)

If \mathbf{A} , \mathbf{B} and \mathbf{b} are deterministic,
then there exists a *universal and uniform* exact quantization.

Remaining cases

$$V(x) = \mathbb{E} \left[\begin{array}{ll} \min_{y \in \mathbb{R}^m} & \mathbf{c}^\top y \\ \text{s.t.} & \mathbf{A}y + \mathbf{B}x \leq \mathbf{b} \end{array} \right]$$

	\mathbf{A}	(\mathbf{B}, \mathbf{b})	\mathbf{c}
Local	×	✓	✓
Uniform	×	×	✓

Theorem (FGL 2021)

If \mathbf{A} , \mathbf{B} and \mathbf{b} are deterministic,
then there exists a *universal and uniform* exact quantization.

Theorem (FL 2022)

If \mathbf{A} is deterministic,
then there exists a *universal and local* exact quantization.

Contents of the manuscript and articles

Chapter 3:



Chapter 4:



M. Forcier, S. Gaubert, V. Leclère

Exact quantization of multistage stochastic linear problems,
arXiv preprint arXiv:2107.09566 (2021),
Best student paper, ECSO-CMS 2022, Venice.

Chapter 5:



M. Forcier, V. Leclère

Generalized adaptive partition-based method for two-stage stochastic linear programs: convergence and generalization,
Operation Research Letters, to appear (2022).

Chapter 6:



M. Forcier, V. Leclère

Convergence of Trajectory Following Dynamic Programming algorithms for multistage stochastic problems without finite support assumptions,
HAL Id : hal-03683697 (2022).

Contents

- 1 Universal Exact Quantization for cost
 - Local in 2-stage
 - Uniform in 2-stage
 - Uniform in multistage
 - Complexity results
- 2 Local and universal exact Quantization for constraints
 - Adapted partitions
 - Adaptive Partition-based Methods
 - Convergence, complexity and numerical results
- 3 Trajectory Following Dynamic Programming
- 4 Conclusion and perspectives

Contents

- 1 Universal Exact Quantization for cost
 - Local in 2-stage
 - Uniform in 2-stage
 - Uniform in multistage
 - Complexity results
- 2 Local and universal exact Quantization for constraints
 - Adapted partitions
 - Adaptive Partition-based Methods
 - Convergence, complexity and numerical results
- 3 Trajectory Following Dynamic Programming
- 4 Conclusion and perspectives

Reformulation of $V(x)$ highlighting the role of the fiber P_x

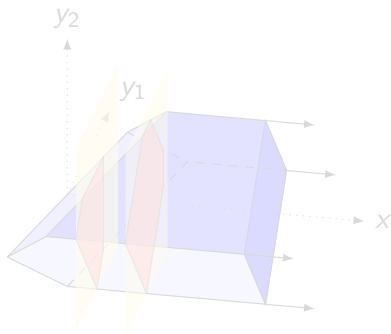
For a given x , (we still assume $V_{t+1} \equiv 0$)

$$V(x) := \mathbb{E} \left[\min_{y \in \mathbb{R}^m} \mathbf{c}^\top y \right. \\ \left. \text{s.t. } Ay + Bx \leq b \right]$$

$$V(x) = \mathbb{E} \left[\min_{y \in P_x} \mathbf{c}^\top y \right] \quad \text{where} \quad P_x := \{y \in \mathbb{R}^m \mid Ay + Bx \leq b\}$$

Illustrative running example:

$$P_x := \{y \in \mathbb{R}^m \mid \|y\|_1 \leq 1, \\ y_1 \leq x, y_2 \leq x\}$$



Reformulation of $V(x)$ highlighting the role of the fiber P_x

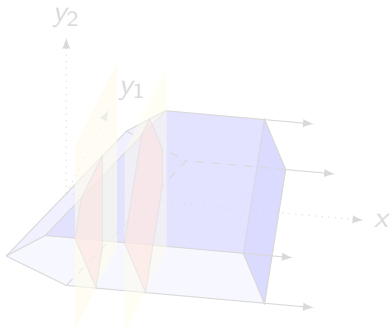
For a given x , (we still assume $V_{t+1} \equiv 0$)

$$V(x) := \mathbb{E} \left[\min_{y \in \mathbb{R}^m} \mathbf{c}^\top y \right. \\ \left. \text{s.t. } Ay + Bx \leq b \right]$$

$$V(x) = \mathbb{E} \left[\min_{y \in P_x} \mathbf{c}^\top y \right] \quad \text{where} \quad P_x := \{y \in \mathbb{R}^m \mid Ay + Bx \leq b\}$$

Illustrative running example:

$$P_x := \{y \in \mathbb{R}^m \mid \|y\|_1 \leq 1, \\ y_1 \leq x, y_2 \leq x\}$$



Reformulation of $V(x)$ highlighting the role of the fiber P_x

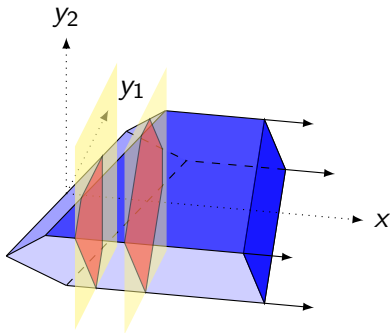
For a given x , (we still assume $V_{t+1} \equiv 0$)

$$V(x) := \mathbb{E} \left[\min_{y \in \mathbb{R}^m} \mathbf{c}^\top y \right. \\ \left. \text{s.t. } Ay + Bx \leq b \right]$$

$$V(x) = \mathbb{E} \left[\min_{y \in P_x} \mathbf{c}^\top y \right] \quad \text{where} \quad P_x := \{y \in \mathbb{R}^m \mid Ay + Bx \leq b\}$$

Illustrative running example:

$$P_x := \{y \in \mathbb{R}^m \mid \|y\|_1 \leq 1, \\ y_1 \leq x, y_2 \leq x\}$$



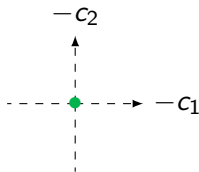
Normal fan $\mathcal{N}(P_x)$

Definition

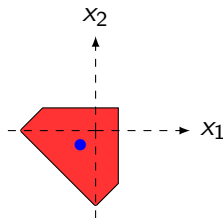
The normal fan of the fiber P_x is

$$\mathcal{N}(P_x) := \{N_{P_x}(y) \mid y \in P_x\}$$

with $N_{P_x}(y) = \{c \mid \forall y' \in P_x, c^\top(y' - y) \leq 0\}$ the normal cone of P_x at y .



$N_{P_x}(y)$ for $x = 0.3$



P_x, y and $N_{P_x}(y)$ for $x = 0.3$

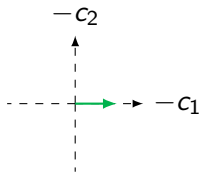
Normal fan $\mathcal{N}(P_x)$

Definition

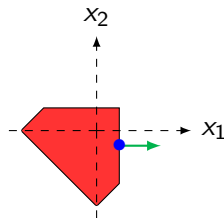
The normal fan of the fiber P_x is

$$\mathcal{N}(P_x) := \{N_{P_x}(y) \mid y \in P_x\}$$

with $N_{P_x}(y) = \{c \mid \forall y' \in P_x, c^\top(y' - y) \leq 0\}$ the normal cone of P_x at y .



$N_{P_x}(y)$ for $x = 0.3$



P_x , y and $N_{P_x}(y)$ for $x = 0.3$

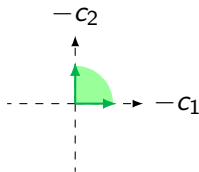
Normal fan $\mathcal{N}(P_x)$

Definition

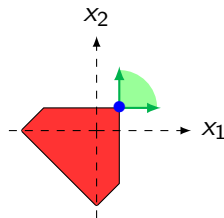
The normal fan of the fiber P_x is

$$\mathcal{N}(P_x) := \{N_{P_x}(y) \mid y \in P_x\}$$

with $N_{P_x}(y) = \{c \mid \forall y' \in P_x, c^\top(y' - y) \leq 0\}$ the normal cone of P_x at y .



$N_{P_x}(y)$ for $x = 0.3$



P_x, y and $N_{P_x}(y)$ for $x = 0.3$

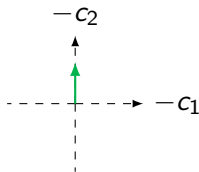
Normal fan $\mathcal{N}(P_x)$

Definition

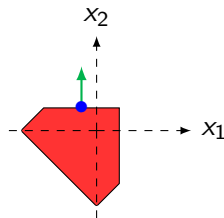
The normal fan of the fiber P_x is

$$\mathcal{N}(P_x) := \{N_{P_x}(y) \mid y \in P_x\}$$

with $N_{P_x}(y) = \{c \mid \forall y' \in P_x, c^\top(y' - y) \leq 0\}$ the normal cone of P_x at y .



$N_{P_x}(y)$ for $x = 0.3$



P_x, y and $N_{P_x}(y)$ for $x = 0.3$

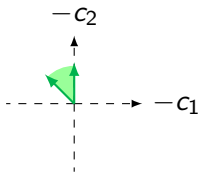
Normal fan $\mathcal{N}(P_x)$

Definition

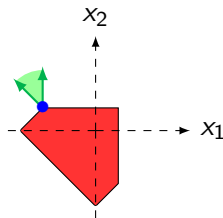
The normal fan of the fiber P_x is

$$\mathcal{N}(P_x) := \{N_{P_x}(y) \mid y \in P_x\}$$

with $N_{P_x}(y) = \{c \mid \forall y' \in P_x, c^\top(y' - y) \leq 0\}$ the normal cone of P_x at y .



$N_{P_x}(y)$ for $x = 0.3$



P_x , y and $N_{P_x}(y)$ for $x = 0.3$

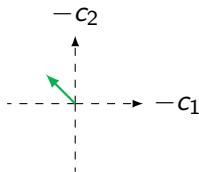
Normal fan $\mathcal{N}(P_x)$

Definition

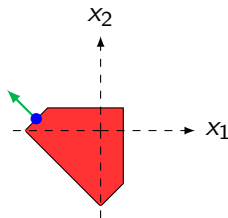
The normal fan of the fiber P_x is

$$\mathcal{N}(P_x) := \{N_{P_x}(y) \mid y \in P_x\}$$

with $N_{P_x}(y) = \{c \mid \forall y' \in P_x, c^\top(y' - y) \leq 0\}$ the normal cone of P_x at y .



$N_{P_x}(y)$ for $x = 0.3$



P_x , y and $N_{P_x}(y)$ for $x = 0.3$

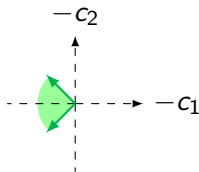
Normal fan $\mathcal{N}(P_x)$

Definition

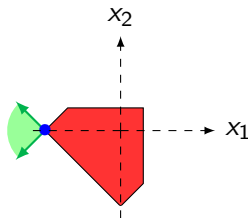
The normal fan of the fiber P_x is

$$\mathcal{N}(P_x) := \{N_{P_x}(y) \mid y \in P_x\}$$

with $N_{P_x}(y) = \{c \mid \forall y' \in P_x, c^\top(y' - y) \leq 0\}$ the normal cone of P_x at y .



$N_{P_x}(y)$ for $x = 0.3$



P_x, y and $N_{P_x}(y)$ for $x = 0.3$

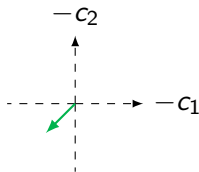
Normal fan $\mathcal{N}(P_x)$

Definition

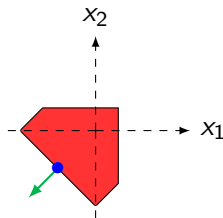
The normal fan of the fiber P_x is

$$\mathcal{N}(P_x) := \{N_{P_x}(y) \mid y \in P_x\}$$

with $N_{P_x}(y) = \{c \mid \forall y' \in P_x, c^\top(y' - y) \leq 0\}$ the normal cone of P_x at y .



$N_{P_x}(y)$ for $x = 0.3$



P_x , y and $N_{P_x}(y)$ for $x = 0.3$

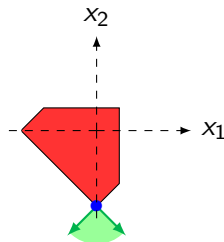
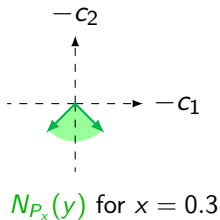
Normal fan $\mathcal{N}(P_x)$

Definition

The normal fan of the fiber P_x is

$$\mathcal{N}(P_x) := \{N_{P_x}(y) \mid y \in P_x\}$$

with $N_{P_x}(y) = \{c \mid \forall y' \in P_x, c^\top(y' - y) \leq 0\}$ the normal cone of P_x at y .



P_x , y and $N_{P_x}(y)$ for $x = 0.3$

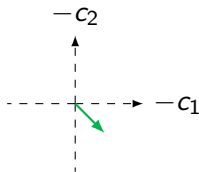
Normal fan $\mathcal{N}(P_x)$

Definition

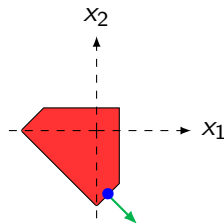
The normal fan of the fiber P_x is

$$\mathcal{N}(P_x) := \{N_{P_x}(y) \mid y \in P_x\}$$

with $N_{P_x}(y) = \{c \mid \forall y' \in P_x, c^\top(y' - y) \leq 0\}$ the normal cone of P_x at y .



$N_{P_x}(y)$ for $x = 0.3$



P_x , y and $N_{P_x}(y)$ for $x = 0.3$

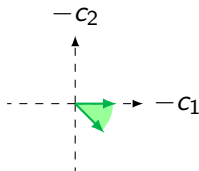
Normal fan $\mathcal{N}(P_x)$

Definition

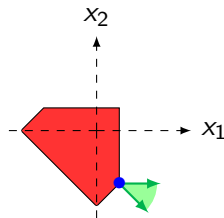
The normal fan of the fiber P_x is

$$\mathcal{N}(P_x) := \{N_{P_x}(y) \mid y \in P_x\}$$

with $N_{P_x}(y) = \{c \mid \forall y' \in P_x, c^\top(y' - y) \leq 0\}$ the normal cone of P_x at y .



$N_{P_x}(y)$ for $x = 0.3$



P_x , y and $N_{P_x}(y)$ for $x = 0.3$

Normal fan $\mathcal{N}(P_x)$

Definition

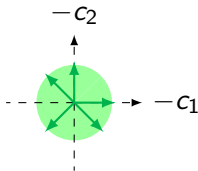
The normal fan of the fiber P_x is

$$\mathcal{N}(P_x) := \{N_{P_x}(y) \mid y \in P_x\}$$

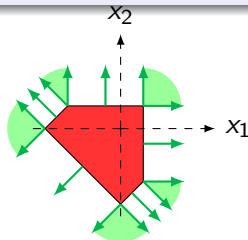
with $N_{P_x}(y) = \{c \mid \forall y' \in P_x, c^\top(y' - y) \leq 0\}$ the normal cone of P_x at y .

Proposition

If P_x is bounded, $\{\text{ri}(N) \mid N \in \mathcal{N}(P_x)\}$ is a partition of \mathbb{R}^m .



$\mathcal{N}(P_x)$ for $x = 0.3$

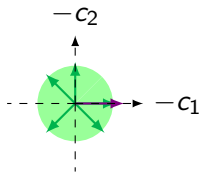


P_x and $\mathcal{N}(P_x)$ for $x = 0.3$

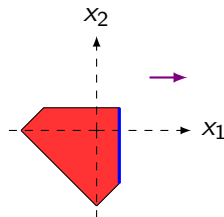
$\mathcal{N}(P_x)$: partition of cost coherent with the min

$$V(x) = \mathbb{E} \left[\min_{y \in P_x} c^\top y \right]$$

For any $N \in \mathcal{N}(P_x)$, $-c \mapsto \arg \min_{y \in P_x} c^\top y$ is constant for all $-c \in \text{ri}(N)$.



Cost $-c$ and $\mathcal{N}(P_x)$ for $x = 0.3$

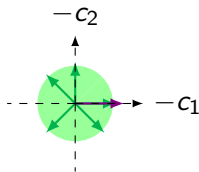


P_x for $x = 0.3$

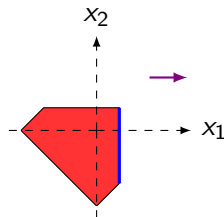
$\mathcal{N}(P_x)$: partition of cost coherent with the min

$$V(x) = \mathbb{E} \left[\min_{y \in P_x} c^\top y \right]$$

For any $N \in \mathcal{N}(P_x)$, $-c \mapsto \arg \min_{y \in P_x} c^\top y$ is constant for all $-c \in \text{ri}(N)$.



Cost $-c$ and $\mathcal{N}(P_x)$ for $x = 0.3$

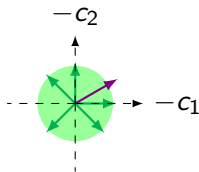


P_x for $x = 0.3$

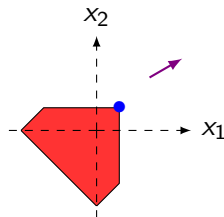
$\mathcal{N}(P_x)$: partition of cost coherent with the min

$$V(x) = \mathbb{E} \left[\min_{y \in P_x} c^\top y \right]$$

For any $N \in \mathcal{N}(P_x)$, $-c \mapsto \arg \min_{y \in P_x} c^\top y$ is constant for all $-c \in \text{ri}(N)$.



Cost $-c$ and $\mathcal{N}(P_x)$ for $x = 0.3$

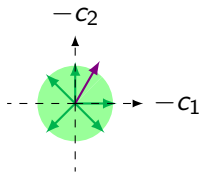


P_x for $x = 0.3$

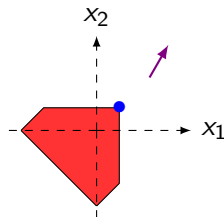
$\mathcal{N}(P_x)$: partition of cost coherent with the min

$$V(x) = \mathbb{E} \left[\min_{y \in P_x} c^\top y \right]$$

For any $N \in \mathcal{N}(P_x)$, $-c \mapsto \arg \min_{y \in P_x} c^\top y$ is constant for all $-c \in \text{ri}(N)$.



Cost $-c$ and $\mathcal{N}(P_x)$ for $x = 0.3$

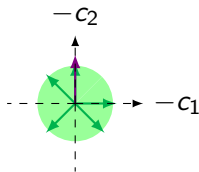


P_x for $x = 0.3$

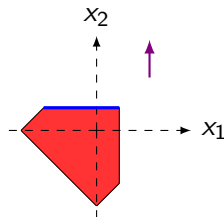
$\mathcal{N}(P_x)$: partition of cost coherent with the min

$$V(x) = \mathbb{E} \left[\min_{y \in P_x} c^\top y \right]$$

For any $N \in \mathcal{N}(P_x)$, $-c \mapsto \arg \min_{y \in P_x} c^\top y$ is constant for all $-c \in \text{ri}(N)$.



Cost $-c$ and $\mathcal{N}(P_x)$ for $x = 0.3$

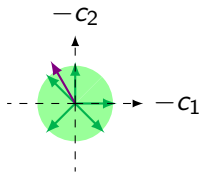


P_x for $x = 0.3$

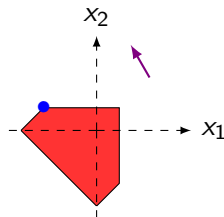
$\mathcal{N}(P_x)$: partition of cost coherent with the min

$$V(x) = \mathbb{E} \left[\min_{y \in P_x} c^\top y \right]$$

For any $N \in \mathcal{N}(P_x)$, $-c \mapsto \arg \min_{y \in P_x} c^\top y$ is constant for all $-c \in \text{ri}(N)$.



Cost $-c$ and $\mathcal{N}(P_x)$ for $x = 0.3$

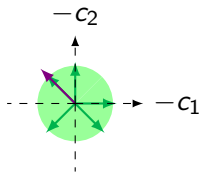


P_x for $x = 0.3$

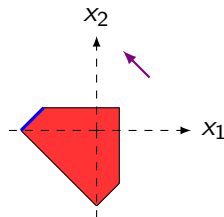
$\mathcal{N}(P_x)$: partition of cost coherent with the min

$$V(x) = \mathbb{E} \left[\min_{y \in P_x} c^\top y \right]$$

For any $N \in \mathcal{N}(P_x)$, $-c \mapsto \arg \min_{y \in P_x} c^\top y$ is constant for all $-c \in \text{ri}(N)$.



Cost $-c$ and $\mathcal{N}(P_x)$ for $x = 0.3$

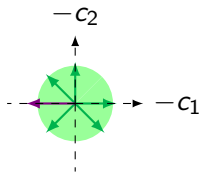


P_x for $x = 0.3$

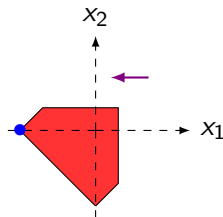
$\mathcal{N}(P_x)$: partition of cost coherent with the min

$$V(x) = \mathbb{E} \left[\min_{y \in P_x} c^\top y \right]$$

For any $N \in \mathcal{N}(P_x)$, $-c \mapsto \arg \min_{y \in P_x} c^\top y$ is constant for all $-c \in \text{ri}(N)$.



Cost $-c$ and $\mathcal{N}(P_x)$ for $x = 0.3$

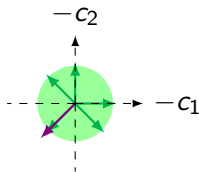


P_x for $x = 0.3$

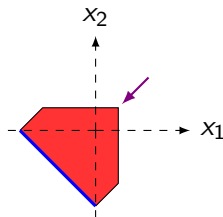
$\mathcal{N}(P_x)$: partition of cost coherent with the min

$$V(x) = \mathbb{E} \left[\min_{y \in P_x} c^\top y \right]$$

For any $N \in \mathcal{N}(P_x)$, $-c \mapsto \arg \min_{y \in P_x} c^\top y$ is constant for all $-c \in \text{ri}(N)$.



Cost $-c$ and $\mathcal{N}(P_x)$ for $x = 0.3$

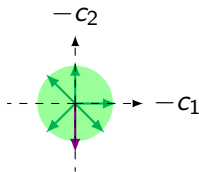


P_x for $x = 0.3$

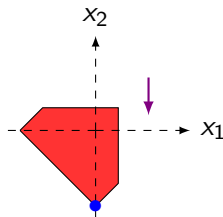
$\mathcal{N}(P_x)$: partition of cost coherent with the min

$$V(x) = \mathbb{E} \left[\min_{y \in P_x} c^\top y \right]$$

For any $N \in \mathcal{N}(P_x)$, $-c \mapsto \arg \min_{y \in P_x} c^\top y$ is constant for all $-c \in \text{ri}(N)$.



Cost $-c$ and $\mathcal{N}(P_x)$ for $x = 0.3$

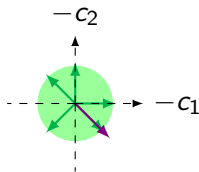


P_x for $x = 0.3$

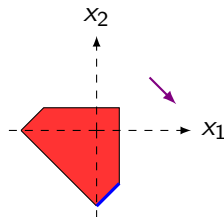
$\mathcal{N}(P_x)$: partition of cost coherent with the min

$$V(x) = \mathbb{E} \left[\min_{y \in P_x} c^\top y \right]$$

For any $N \in \mathcal{N}(P_x)$, $-c \mapsto \arg \min_{y \in P_x} c^\top y$ is constant for all $-c \in \text{ri}(N)$.



Cost $-c$ and $\mathcal{N}(P_x)$ for $x = 0.3$

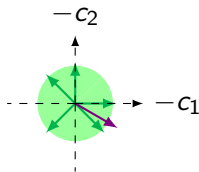


P_x for $x = 0.3$

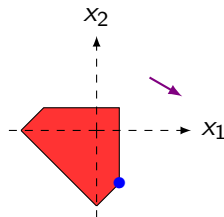
$\mathcal{N}(P_x)$: partition of cost coherent with the min

$$V(x) = \mathbb{E} \left[\min_{y \in P_x} c^\top y \right]$$

For any $N \in \mathcal{N}(P_x)$, $-c \mapsto \arg \min_{y \in P_x} c^\top y$ is constant for all $-c \in \text{ri}(N)$.



Cost $-c$ and $\mathcal{N}(P_x)$ for $x = 0.3$

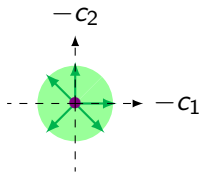


P_x for $x = 0.3$

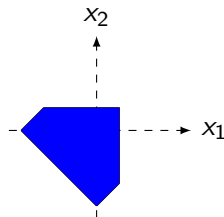
$\mathcal{N}(P_x)$: partition of cost coherent with the min

$$V(x) = \mathbb{E} \left[\min_{y \in P_x} c^\top y \right]$$

For any $N \in \mathcal{N}(P_x)$, $-c \mapsto \arg \min_{y \in P_x} c^\top y$ is constant for all $-c \in \text{ri}(N)$.



Cost $-c$ and $\mathcal{N}(P_x)$ for $x = 0.3$

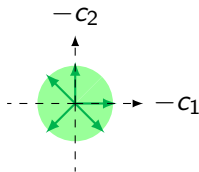


P_x for $x = 0.3$

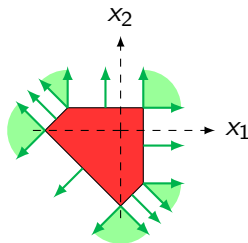
$\mathcal{N}(P_x)$: partition of cost coherent with the min

$$V(x) = \mathbb{E} \left[\min_{y \in P_x} c^\top y \right]$$

For any $N \in \mathcal{N}(P_x)$, $-c \mapsto \arg \min_{y \in P_x} c^\top y$ is constant for all $-c \in \text{ri}(N)$.



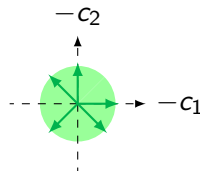
Cost $-c$ and $\mathcal{N}(P_x)$ for $x = 0.3$



P_x for $x = 0.3$

Local and universal exact quantization for \mathbf{c}

$$\begin{aligned} V(x) &= \mathbb{E} \left[\min_{y \in P_x} \mathbf{c}^\top y \right] \\ &= \sum_{N \in \mathcal{N}(P_x)} \mathbb{E} \left[\mathbb{1}_{\mathbf{c} \in -\text{ri } N} \min_{y \in P_x} \mathbf{c}^\top y \right] \end{aligned}$$

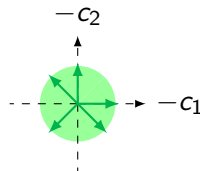


$\mathcal{N}(P_x)$

for $x = 0.3$

Local and universal exact quantization for \mathbf{c}

$$\begin{aligned}
 V(x) &= \mathbb{E} \left[\min_{y \in P_x} \mathbf{c}^\top y \right] \\
 &= \sum_{N \in \mathcal{N}(P_x)} \mathbb{E} \left[\mathbb{1}_{\mathbf{c} \in -\text{ri } N} \min_{y \in P_x} \mathbf{c}^\top y \right] \quad \text{where } y_N(x) \in \arg \min_{y \in P_x} \underbrace{\mathbf{c}^\top}_{\in -\text{ri } N} y. \\
 &= \sum_{N \in \mathcal{N}(P_x)} \mathbb{E} \left[\mathbb{1}_{\mathbf{c} \in -\text{ri } N} \mathbf{c}^\top \right] y_N(x)
 \end{aligned}$$

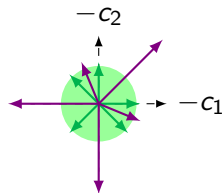


$\mathcal{N}(P_x)$

for $x = 0.3$

Local and universal exact quantization for \mathbf{c}

$$\begin{aligned}
 V(x) &= \mathbb{E} \left[\min_{y \in P_x} \mathbf{c}^\top y \right] \\
 &= \sum_{N \in \mathcal{N}(P_x)} \mathbb{E} \left[\mathbb{1}_{\mathbf{c} \in -\text{ri } N} \min_{y \in P_x} \mathbf{c}^\top y \right] \quad \text{where } y_N(x) \in \arg \min_{y \in P_x} \underbrace{\mathbf{c}^\top}_{\in -\text{ri } N} y. \\
 &= \sum_{N \in \mathcal{N}(P_x)} \mathbb{E} \left[\mathbb{1}_{\mathbf{c} \in -\text{ri } N} \mathbf{c}^\top \right] y_N(x) \\
 &= \sum_{N \in \mathcal{N}(P_x)} p_N \check{\mathbf{c}}_N^\top y_N(x)
 \end{aligned}$$



$\mathcal{N}(P_x)$ and $p_N \check{\mathbf{c}}_N$ for $x = 0.3$

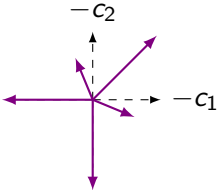
For $N \in \mathcal{N}(P_x)$,

$$p_N := \mathbb{P}[\mathbf{c} \in -\text{ri } N]$$

$$\check{\mathbf{c}}_N := \mathbb{E}[\mathbf{c} \mid \mathbf{c} \in -\text{ri } N]$$

We replace the continuous cost \mathbf{c} ,
by the discrete cost $\check{\mathbf{c}}$.

Local and universal exact quantization for \mathbf{c}

$$\begin{aligned} V(x) &= \mathbb{E} \left[\min_{y \in P_x} \mathbf{c}^\top y \right] \\ &= \sum_{N \in \mathcal{N}(P_x)} \mathbb{E} \left[\mathbb{1}_{\mathbf{c} \in -\text{ri } N} \min_{y \in P_x} \mathbf{c}^\top y \right] \text{ where } y_N(x) \in \arg \min_{y \in P_x} \underbrace{\mathbf{c}^\top}_{\in -\text{ri } N} y. \\ &= \sum_{N \in \mathcal{N}(P_x)} \mathbb{E} \left[\mathbb{1}_{\mathbf{c} \in -\text{ri } N} \mathbf{c}^\top \right] y_N(x) \\ &= \sum_{N \in \mathcal{N}(P_x)} p_N \check{\mathbf{c}}_N^\top y_N(x) \\ &= \sum_{N \in \mathcal{N}(P_x)} p_N \min_{y \in P_x} \check{\mathbf{c}}_N^\top y \end{aligned}$$


$p_N \check{\mathbf{c}}_N$ for $x = 0.3$

For $N \in \mathcal{N}(P_x)$,

$$p_N := \mathbb{P}[\mathbf{c} \in -\text{ri } N]$$

$$\check{\mathbf{c}}_N := \mathbb{E}[\mathbf{c} \mid \mathbf{c} \in -\text{ri } N]$$

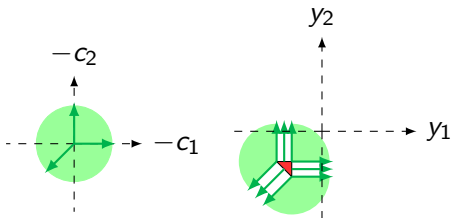
We replace the continuous cost \mathbf{c} ,
by the discrete cost $\check{\mathbf{c}}$.

Contents

- 1 Universal Exact Quantization for cost
 - Local in 2-stage
 - **Uniform in 2-stage**
 - Uniform in multistage
 - Complexity results
- 2 Local and universal exact Quantization for constraints
 - Adapted partitions
 - Adaptive Partition-based Methods
 - Convergence, complexity and numerical results
- 3 Trajectory Following Dynamic Programming
- 4 Conclusion and perspectives

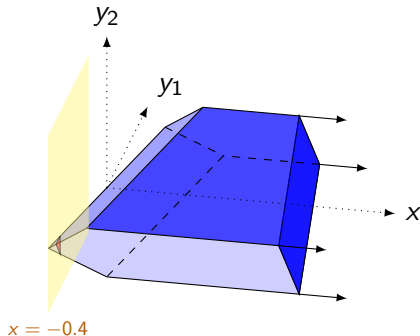
x is no longer fixed but $x \mapsto \mathcal{N}(P_x)$ is piecewise constant.

$$P_x := \{y \mid Ay + Bx \leq b\} \quad \text{and} \quad P := \{(x, y) \mid Ay + Bx \leq b\}$$



$\mathcal{N}(P_x)$

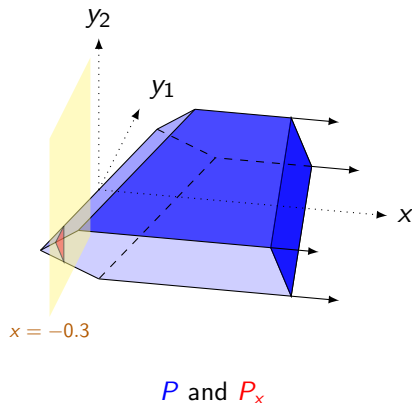
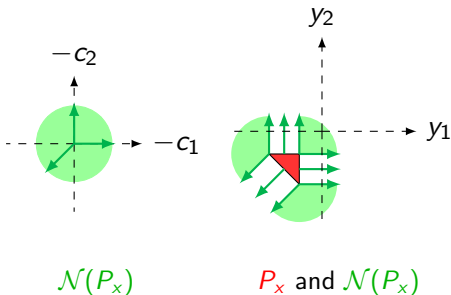
P_x and $\mathcal{N}(P_x)$



P and P_x

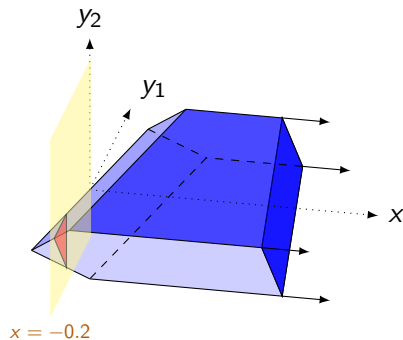
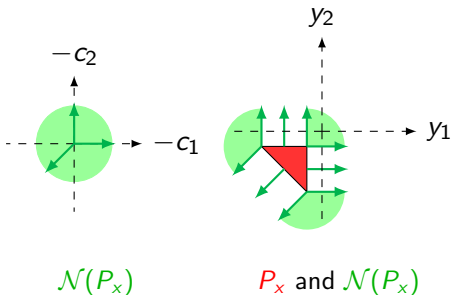
x is no longer fixed but $x \mapsto \mathcal{N}(P_x)$ is piecewise constant.

$$P_x := \{y \mid Ay + Bx \leq b\} \quad \text{and} \quad P := \{(x, y) \mid Ay + Bx \leq b\}$$



x is no longer fixed but $x \mapsto \mathcal{N}(P_x)$ is piecewise constant.

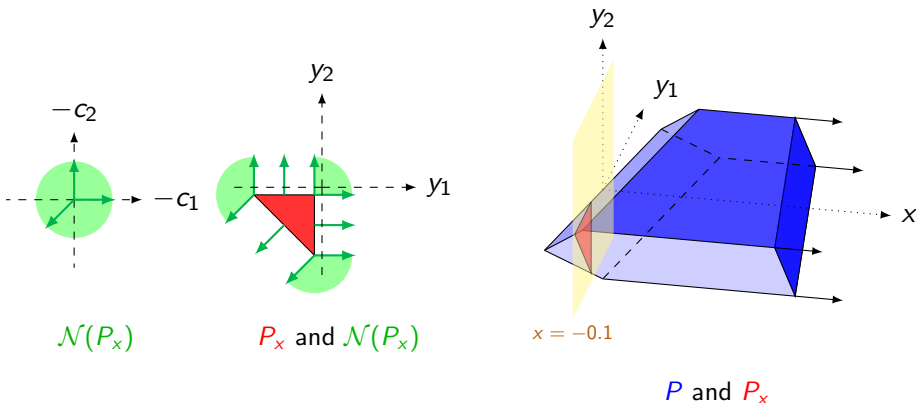
$$P_x := \{y \mid Ay + Bx \leq b\} \quad \text{and} \quad P := \{(x, y) \mid Ay + Bx \leq b\}$$



P and P_x

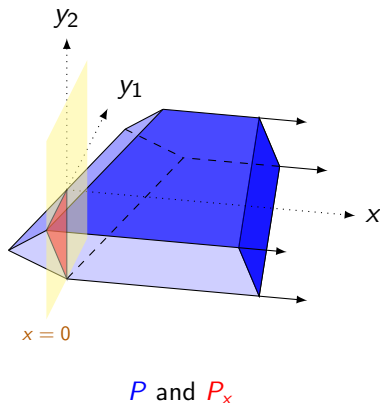
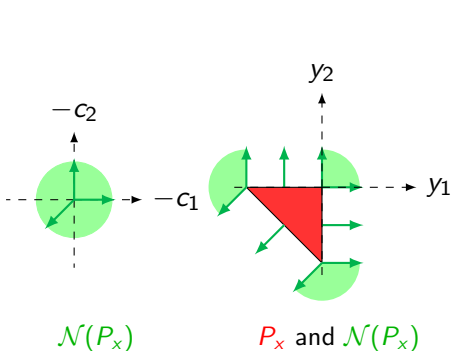
x is no longer fixed but $x \mapsto \mathcal{N}(P_x)$ is piecewise constant.

$$P_x := \{y \mid Ay + Bx \leq b\} \quad \text{and} \quad P := \{(x, y) \mid Ay + Bx \leq b\}$$



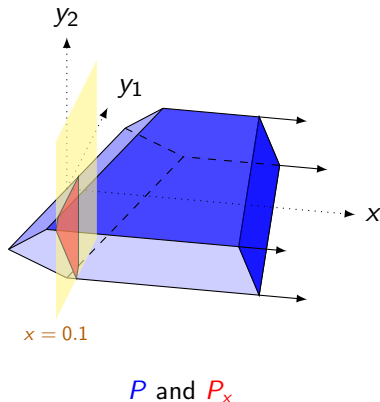
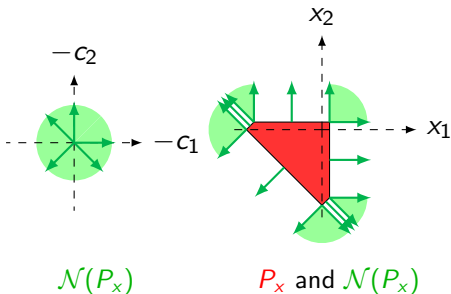
x is no longer fixed but $x \mapsto \mathcal{N}(P_x)$ is piecewise constant.

$$P_x := \{y \mid Ay + Bx \leq b\} \quad \text{and} \quad P := \{(x, y) \mid Ay + Bx \leq b\}$$



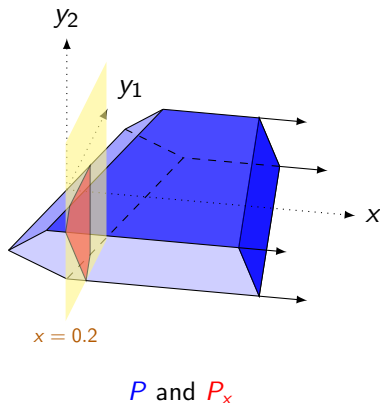
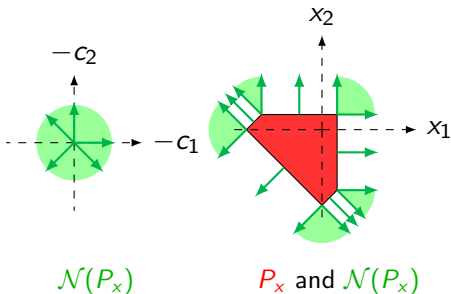
x is no longer fixed but $x \mapsto \mathcal{N}(P_x)$ is piecewise constant.

$$P_x := \{y \mid Ay + Bx \leq b\} \quad \text{and} \quad P := \{(x, y) \mid Ay + Bx \leq b\}$$



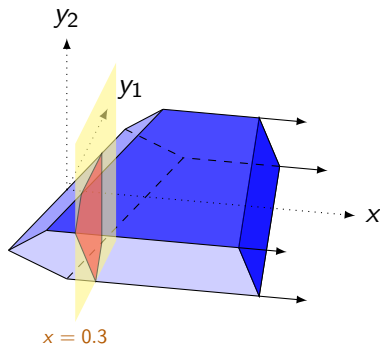
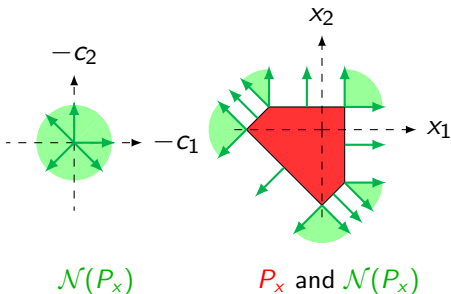
x is no longer fixed but $x \mapsto \mathcal{N}(P_x)$ is piecewise constant.

$$P_x := \{y \mid Ay + Bx \leq b\} \quad \text{and} \quad P := \{(x, y) \mid Ay + Bx \leq b\}$$



x is no longer fixed but $x \mapsto \mathcal{N}(P_x)$ is piecewise constant.

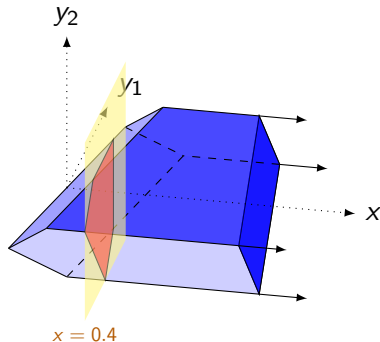
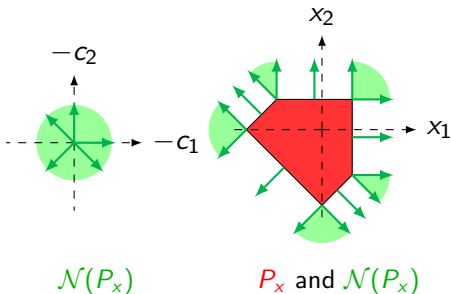
$$P_x := \{y \mid Ay + Bx \leq b\} \quad \text{and} \quad P := \{(x, y) \mid Ay + Bx \leq b\}$$



P and P_x

x is no longer fixed but $x \mapsto \mathcal{N}(P_x)$ is piecewise constant.

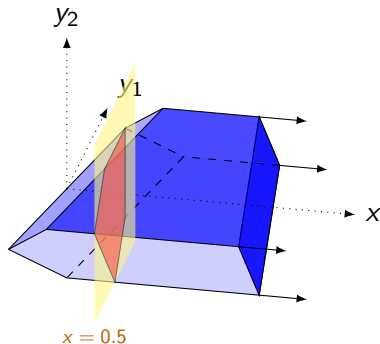
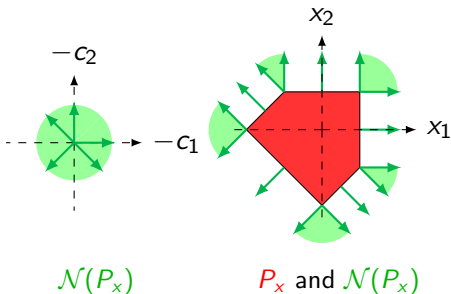
$$P_x := \{y \mid Ay + Bx \leq b\} \quad \text{and} \quad P := \{(x, y) \mid Ay + Bx \leq b\}$$



P and P_x

x is no longer fixed but $x \mapsto \mathcal{N}(P_x)$ is piecewise constant.

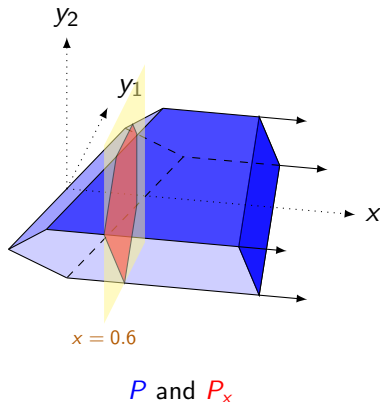
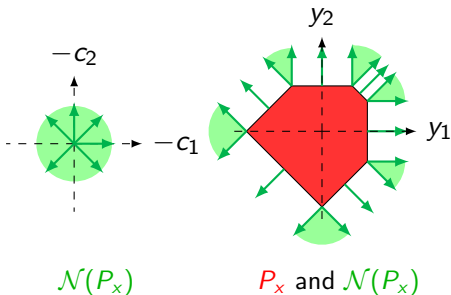
$$P_x := \{y \mid Ay + Bx \leq b\} \quad \text{and} \quad P := \{(x, y) \mid Ay + Bx \leq b\}$$



P and P_x

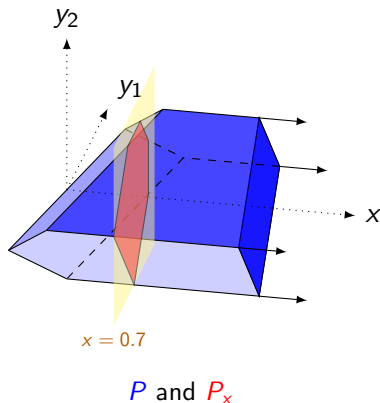
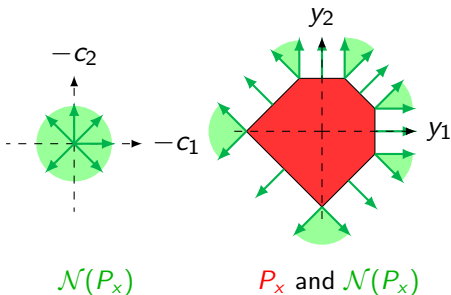
x is no longer fixed but $x \mapsto \mathcal{N}(P_x)$ is piecewise constant.

$$P_x := \{y \mid Ay + Bx \leq b\} \quad \text{and} \quad P := \{(x, y) \mid Ay + Bx \leq b\}$$



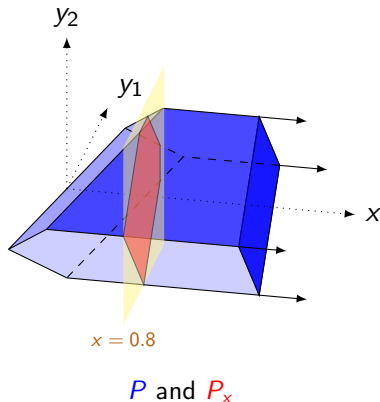
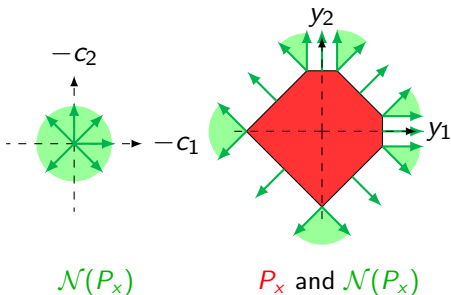
x is no longer fixed but $x \mapsto \mathcal{N}(P_x)$ is piecewise constant.

$$P_x := \{y \mid Ay + Bx \leq b\} \quad \text{and} \quad P := \{(x, y) \mid Ay + Bx \leq b\}$$



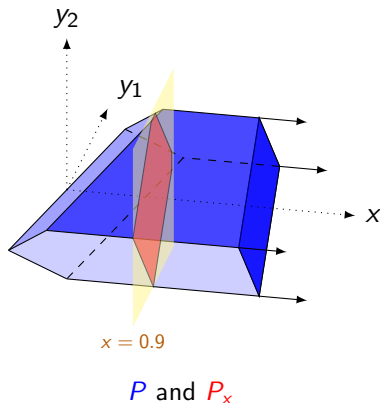
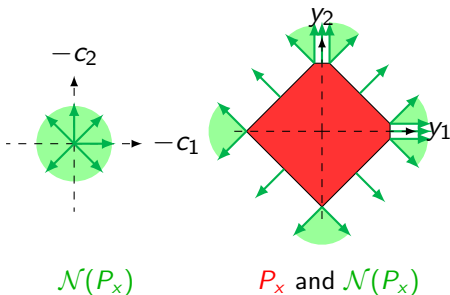
x is no longer fixed but $x \mapsto \mathcal{N}(P_x)$ is piecewise constant.

$$P_x := \{y \mid Ay + Bx \leq b\} \quad \text{and} \quad P := \{(x, y) \mid Ay + Bx \leq b\}$$



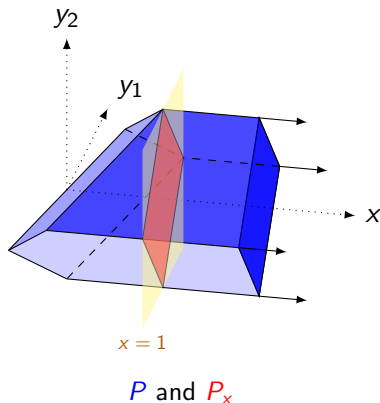
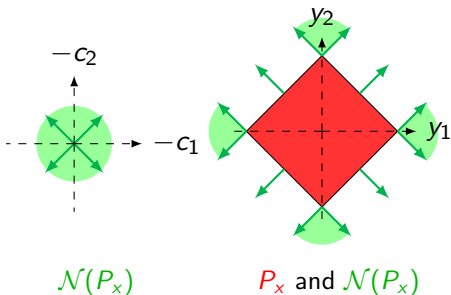
x is no longer fixed but $x \mapsto \mathcal{N}(P_x)$ is piecewise constant.

$$P_x := \{y \mid Ay + Bx \leq b\} \quad \text{and} \quad P := \{(x, y) \mid Ay + Bx \leq b\}$$



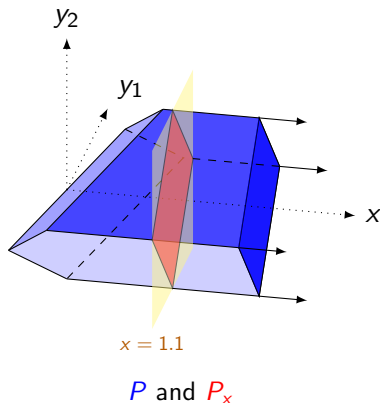
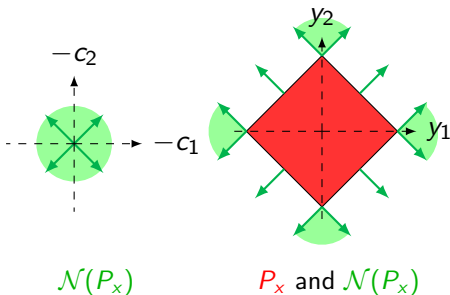
x is no longer fixed but $x \mapsto \mathcal{N}(P_x)$ is piecewise constant.

$$P_x := \{y \mid Ay + Bx \leq b\} \quad \text{and} \quad P := \{(x, y) \mid Ay + Bx \leq b\}$$



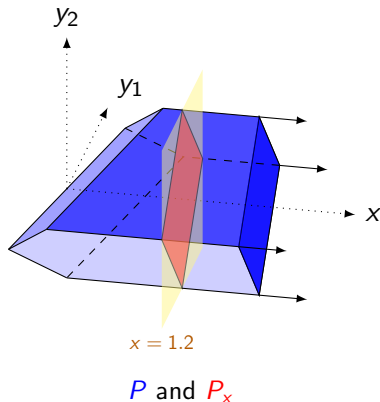
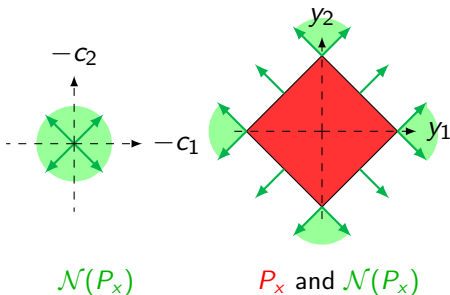
x is no longer fixed but $x \mapsto \mathcal{N}(P_x)$ is piecewise constant.

$$P_x := \{y \mid Ay + Bx \leq b\} \quad \text{and} \quad P := \{(x, y) \mid Ay + Bx \leq b\}$$



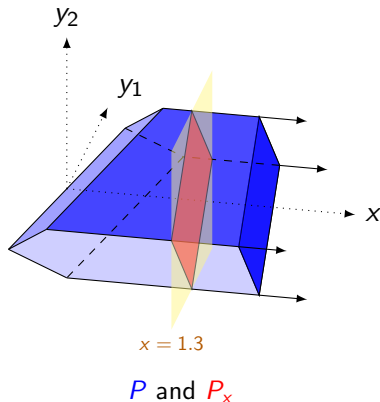
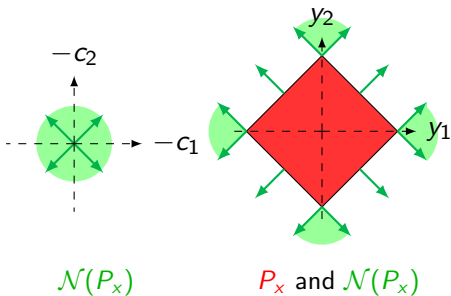
x is no longer fixed but $x \mapsto \mathcal{N}(P_x)$ is piecewise constant.

$$P_x := \{y \mid Ay + Bx \leq b\} \quad \text{and} \quad P := \{(x, y) \mid Ay + Bx \leq b\}$$



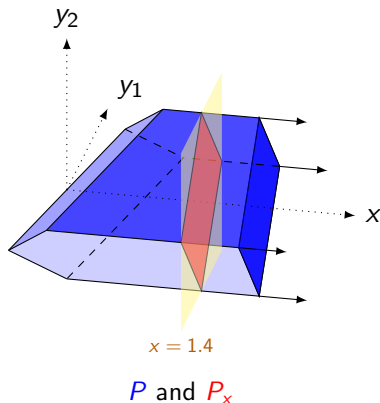
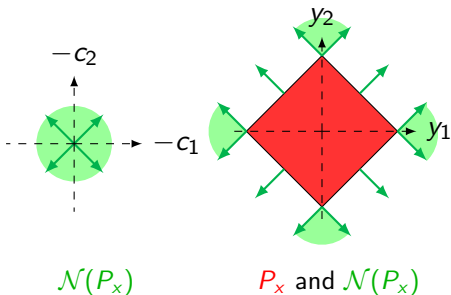
x is no longer fixed but $x \mapsto \mathcal{N}(P_x)$ is piecewise constant.

$$P_x := \{y \mid Ay + Bx \leq b\} \quad \text{and} \quad P := \{(x, y) \mid Ay + Bx \leq b\}$$



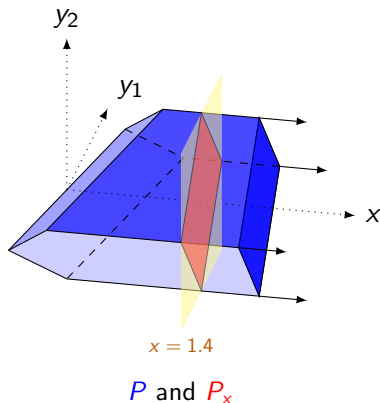
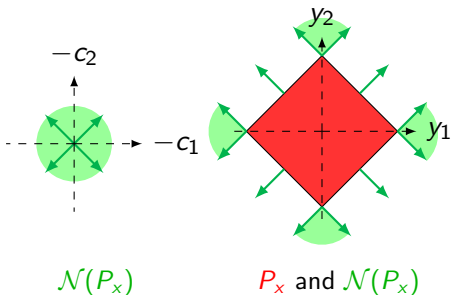
x is no longer fixed but $x \mapsto \mathcal{N}(P_x)$ is piecewise constant.

$$P_x := \{y \mid Ay + Bx \leq b\} \quad \text{and} \quad P := \{(x, y) \mid Ay + Bx \leq b\}$$



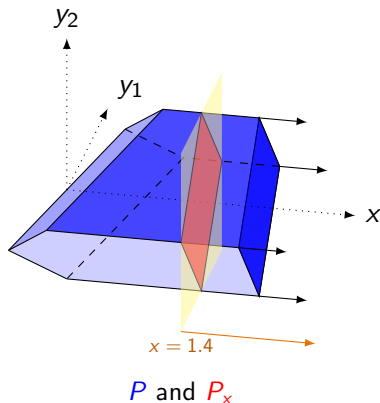
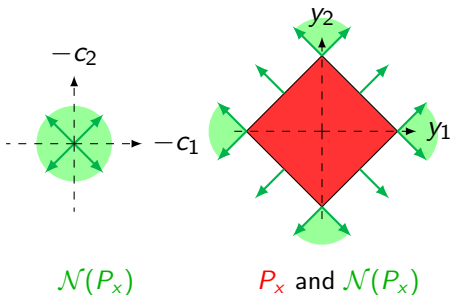
x is no longer fixed but $x \mapsto \mathcal{N}(P_x)$ is piecewise constant.

$$P_x := \{y \mid Ay + Bx \leq b\} \quad \text{and} \quad P := \{(x, y) \mid Ay + Bx \leq b\}$$



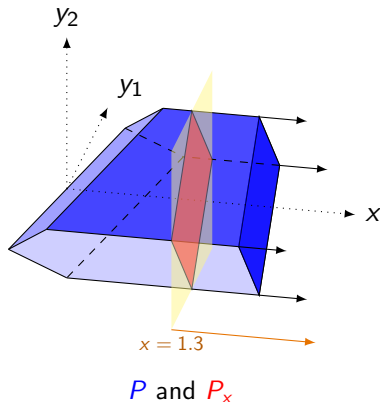
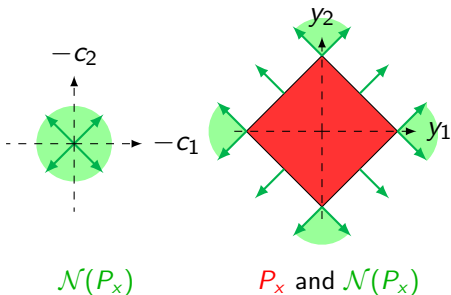
x is no longer fixed but $x \mapsto \mathcal{N}(P_x)$ is piecewise constant.

$$P_x := \{y \mid Ay + Bx \leq b\} \quad \text{and} \quad P := \{(x, y) \mid Ay + Bx \leq b\}$$



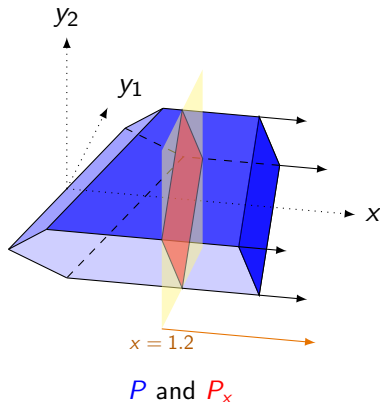
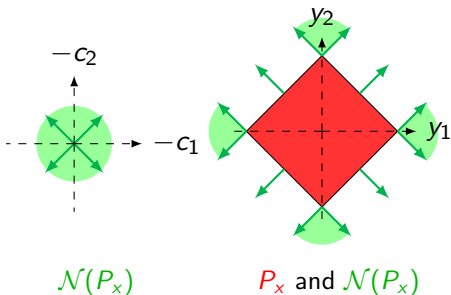
x is no longer fixed but $x \mapsto \mathcal{N}(P_x)$ is piecewise constant.

$$P_x := \{y \mid Ay + Bx \leq b\} \quad \text{and} \quad P := \{(x, y) \mid Ay + Bx \leq b\}$$



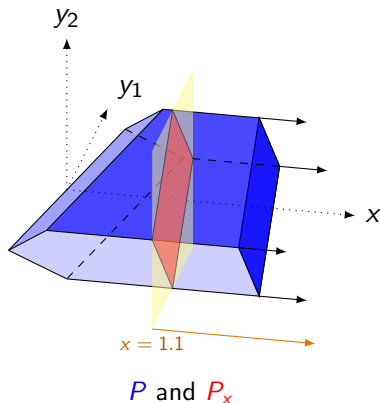
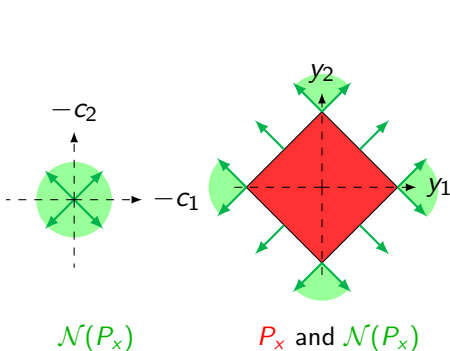
x is no longer fixed but $x \mapsto \mathcal{N}(P_x)$ is piecewise constant.

$$P_x := \{y \mid Ay + Bx \leq b\} \quad \text{and} \quad P := \{(x, y) \mid Ay + Bx \leq b\}$$



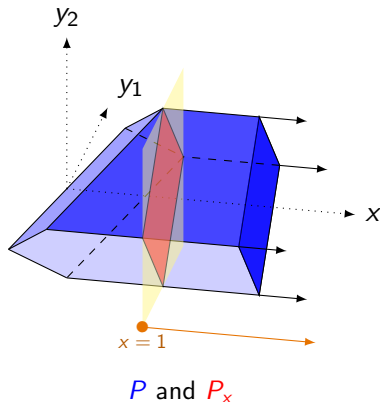
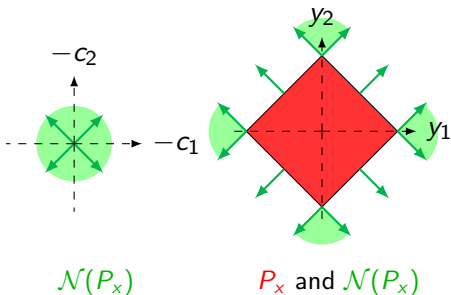
x is no longer fixed but $x \mapsto \mathcal{N}(P_x)$ is piecewise constant.

$$P_x := \{y \mid Ay + Bx \leq b\} \quad \text{and} \quad P := \{(x, y) \mid Ay + Bx \leq b\}$$



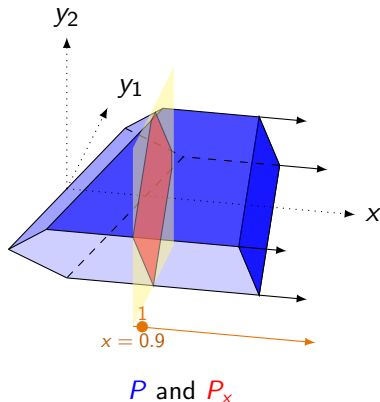
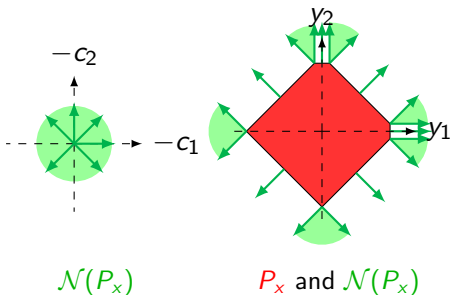
x is no longer fixed but $x \mapsto \mathcal{N}(P_x)$ is piecewise constant.

$$P_x := \{y \mid Ay + Bx \leq b\} \quad \text{and} \quad P := \{(x, y) \mid Ay + Bx \leq b\}$$



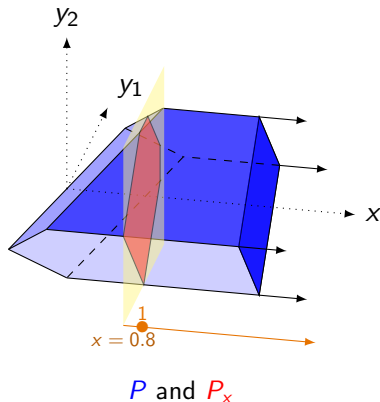
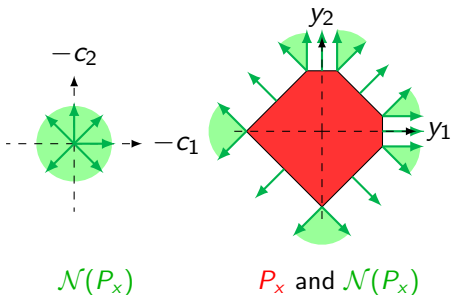
x is no longer fixed but $x \mapsto \mathcal{N}(P_x)$ is piecewise constant.

$$P_x := \{y \mid Ay + Bx \leq b\} \quad \text{and} \quad P := \{(x, y) \mid Ay + Bx \leq b\}$$



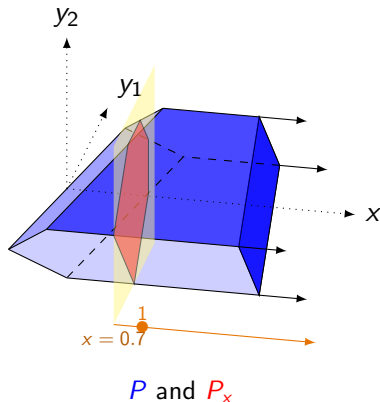
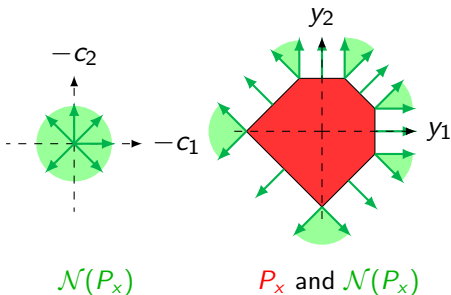
x is no longer fixed but $x \mapsto \mathcal{N}(P_x)$ is piecewise constant.

$$P_x := \{y \mid Ay + Bx \leq b\} \quad \text{and} \quad P := \{(x, y) \mid Ay + Bx \leq b\}$$



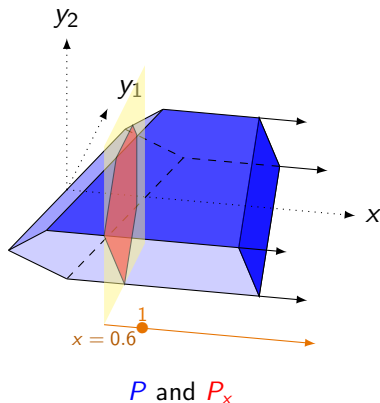
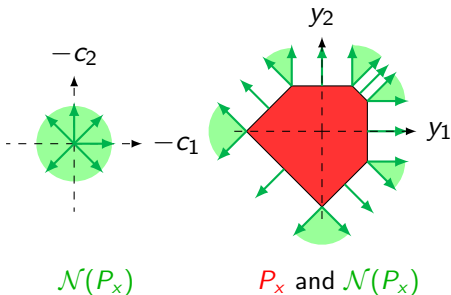
x is no longer fixed but $x \mapsto \mathcal{N}(P_x)$ is piecewise constant.

$$P_x := \{y \mid Ay + Bx \leq b\} \quad \text{and} \quad P := \{(x, y) \mid Ay + Bx \leq b\}$$



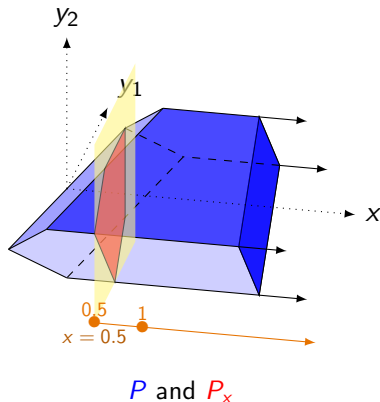
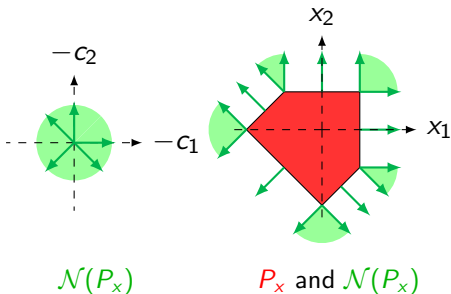
x is no longer fixed but $x \mapsto \mathcal{N}(P_x)$ is piecewise constant.

$$P_x := \{y \mid Ay + Bx \leq b\} \quad \text{and} \quad P := \{(x, y) \mid Ay + Bx \leq b\}$$



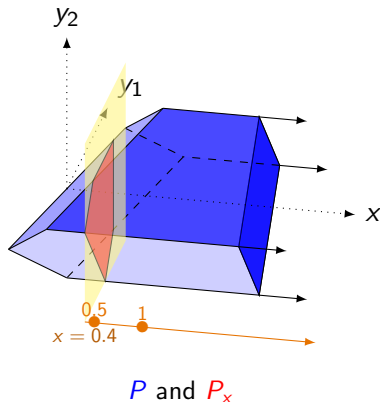
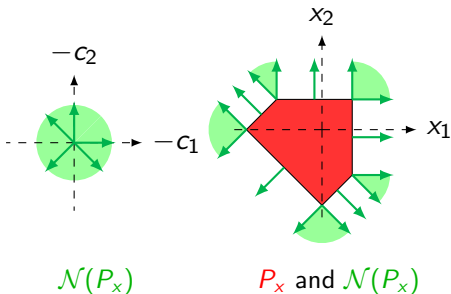
x is no longer fixed but $x \mapsto \mathcal{N}(P_x)$ is piecewise constant.

$$P_x := \{y \mid Ay + Bx \leq b\} \quad \text{and} \quad P := \{(x, y) \mid Ay + Bx \leq b\}$$



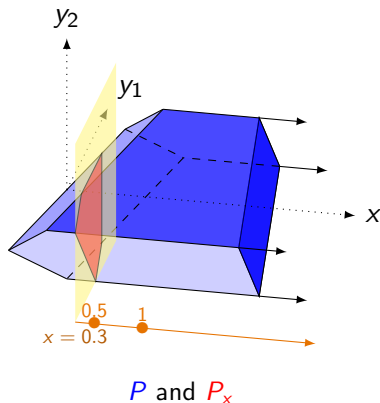
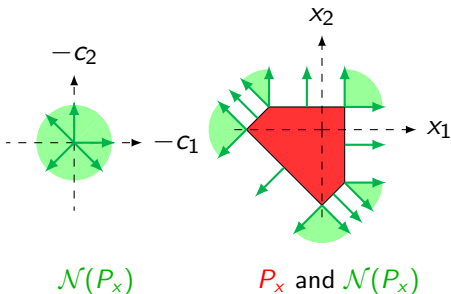
x is no longer fixed but $x \mapsto \mathcal{N}(P_x)$ is piecewise constant.

$$P_x := \{y \mid Ay + Bx \leq b\} \quad \text{and} \quad P := \{(x, y) \mid Ay + Bx \leq b\}$$



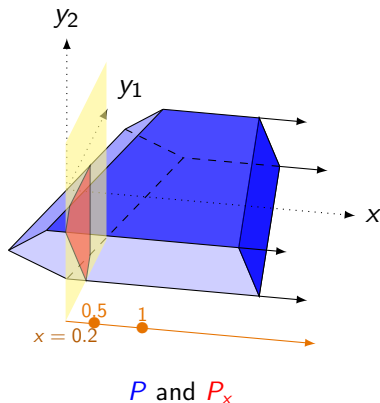
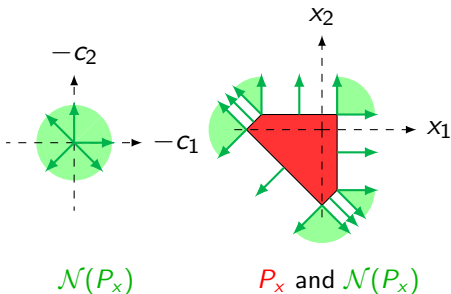
x is no longer fixed but $x \mapsto \mathcal{N}(P_x)$ is piecewise constant.

$$P_x := \{y \mid Ay + Bx \leq b\} \quad \text{and} \quad P := \{(x, y) \mid Ay + Bx \leq b\}$$



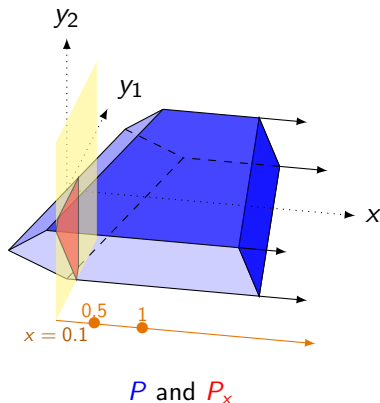
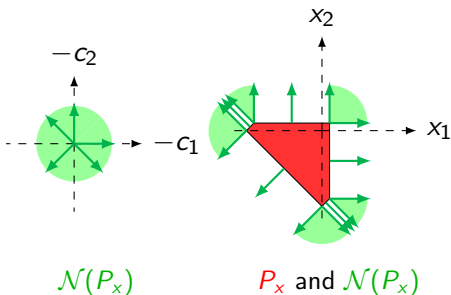
x is no longer fixed but $x \mapsto \mathcal{N}(P_x)$ is piecewise constant.

$$P_x := \{y \mid Ay + Bx \leq b\} \quad \text{and} \quad P := \{(x, y) \mid Ay + Bx \leq b\}$$



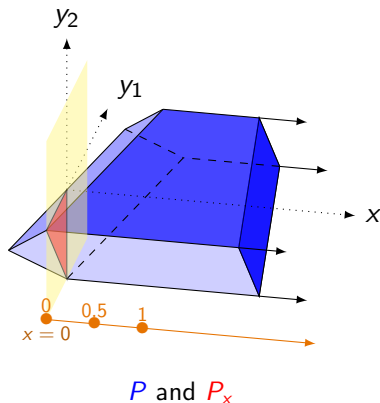
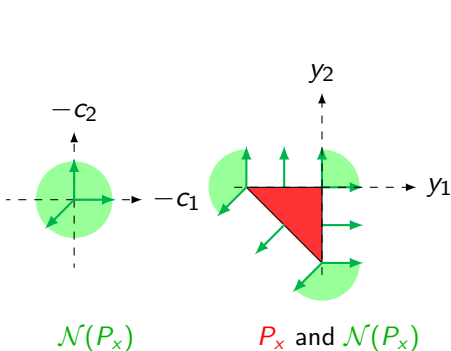
x is no longer fixed but $x \mapsto \mathcal{N}(P_x)$ is piecewise constant.

$$P_x := \{y \mid Ay + Bx \leq b\} \quad \text{and} \quad P := \{(x, y) \mid Ay + Bx \leq b\}$$



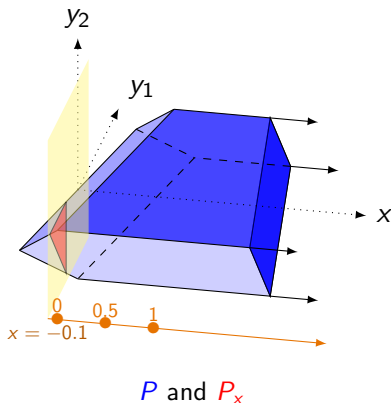
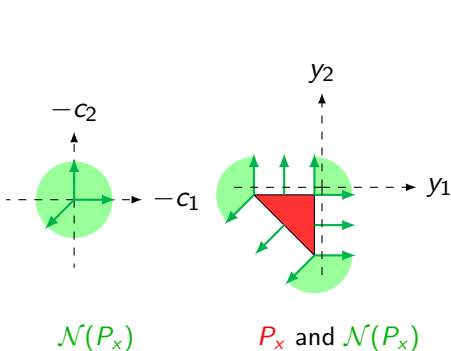
x is no longer fixed but $x \mapsto \mathcal{N}(P_x)$ is piecewise constant.

$$P_x := \{y \mid Ay + Bx \leq b\} \quad \text{and} \quad P := \{(x, y) \mid Ay + Bx \leq b\}$$



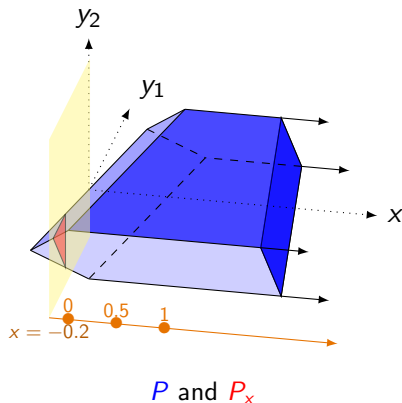
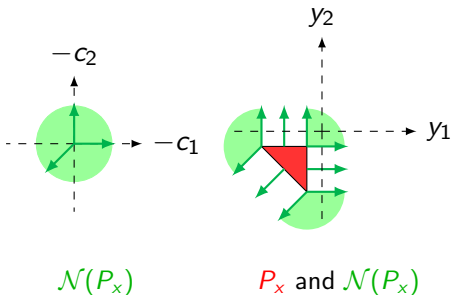
x is no longer fixed but $x \mapsto \mathcal{N}(P_x)$ is piecewise constant.

$$P_x := \{y \mid Ay + Bx \leq b\} \quad \text{and} \quad P := \{(x, y) \mid Ay + Bx \leq b\}$$



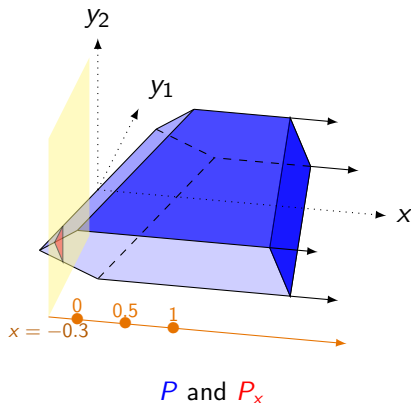
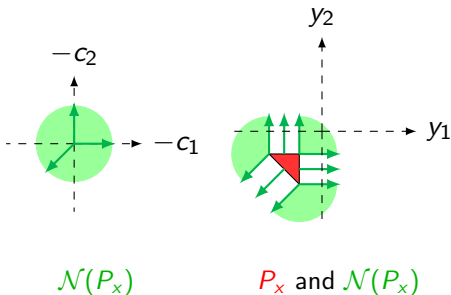
x is no longer fixed but $x \mapsto \mathcal{N}(P_x)$ is piecewise constant.

$$P_x := \{y \mid Ay + Bx \leq b\} \quad \text{and} \quad P := \{(x, y) \mid Ay + Bx \leq b\}$$



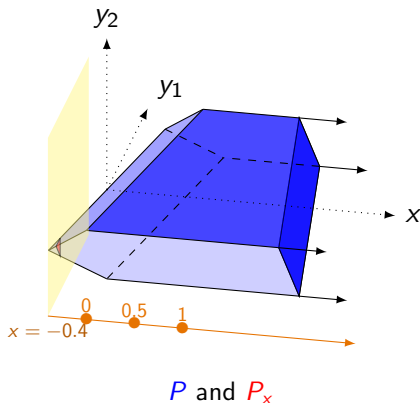
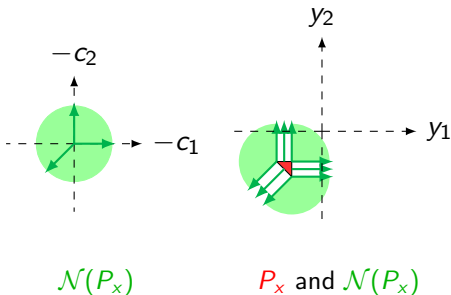
x is no longer fixed but $x \mapsto \mathcal{N}(P_x)$ is piecewise constant.

$$P_x := \{y \mid Ay + Bx \leq b\} \quad \text{and} \quad P := \{(x, y) \mid Ay + Bx \leq b\}$$



x is no longer fixed but $x \mapsto \mathcal{N}(P_x)$ is piecewise constant.

$$P_x := \{y \mid Ay + Bx \leq b\} \quad \text{and} \quad P := \{(x, y) \mid Ay + Bx \leq b\}$$

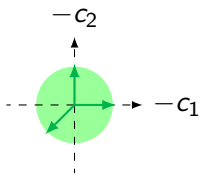
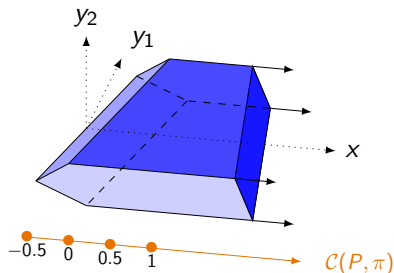


What are the constant regions of $x \mapsto \mathcal{N}(P_x)$?

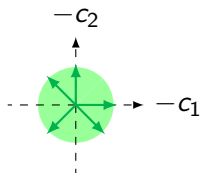
Proposition

There exists a collection $\mathcal{C}(P, \pi)$ called the **chamber complex** whose relative interior of cells are the constant regions of $x \mapsto \mathcal{N}(P_x)$.

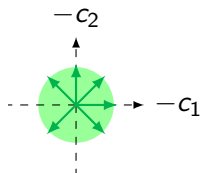
I.e, for $\sigma \in \mathcal{C}(P, \pi)$ and $x, x' \in \text{ri}(\sigma)$, we have $\mathcal{N}(P_x) = \mathcal{N}(P_{x'}) =: \mathcal{N}_\sigma$



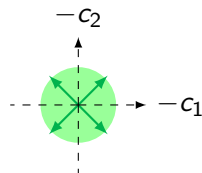
\mathcal{N}_σ for $\sigma = [-0.5, 0]$



\mathcal{N}_σ for $\sigma = [0, 0.5]$



\mathcal{N}_σ for $\sigma = [0.5, 1]$



\mathcal{N}_σ for $\sigma = [1, +\infty)$

Chamber complex

Definition (Billera, Sturmfels 92)

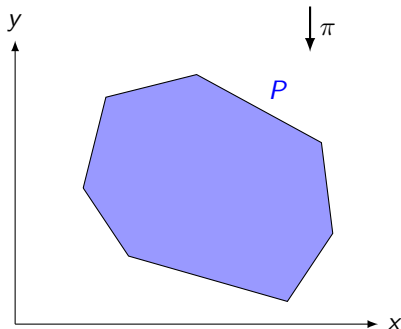
The *chamber complex* $\mathcal{C}(P, \pi)$ of P along π is

$$\mathcal{C}(P, \pi) := \{\sigma_{P, \pi}(x) \mid x \in \pi(P)\}$$

where

$$\sigma_{P, \pi}(x) := \bigcap_{F \in \mathcal{F}(P) \mid x \in \pi(F)} \pi(F)$$

where $\mathcal{F}(P)$ is the set of faces of P and π is the projection $(x, y) \mapsto x$.



Chamber complex

Definition (Billera, Sturmfels 92)

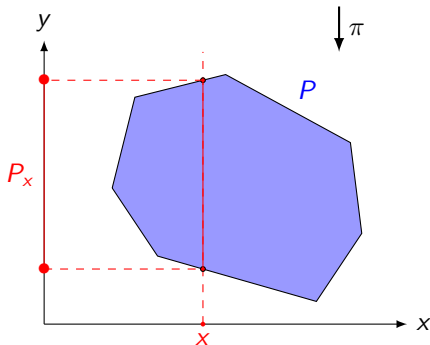
The *chamber complex* $\mathcal{C}(P, \pi)$ of P along π is

$$\mathcal{C}(P, \pi) := \{\sigma_{P, \pi}(x) \mid x \in \pi(P)\}$$

where

$$\sigma_{P, \pi}(x) := \bigcap_{F \in \mathcal{F}(P) \mid x \in \pi(F)} \pi(F)$$

where $\mathcal{F}(P)$ is the set of faces of P and π is the projection $(x, y) \mapsto x$.



Chamber complex

Definition (Billera, Sturmfels 92)

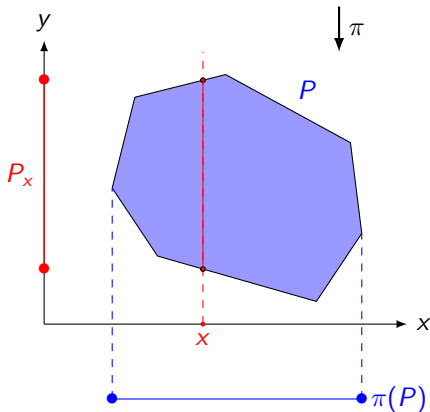
The *chamber complex* $\mathcal{C}(P, \pi)$ of P along π is

$$\mathcal{C}(P, \pi) := \{\sigma_{P, \pi}(x) \mid x \in \pi(P)\}$$

where

$$\sigma_{P, \pi}(x) := \bigcap_{F \in \mathcal{F}(P) \mid x \in \pi(F)} \pi(F)$$

where $\mathcal{F}(P)$ is the set of faces of P and π is the projection $(x, y) \mapsto x$.



Chamber complex

Definition (Billera, Sturmfels 92)

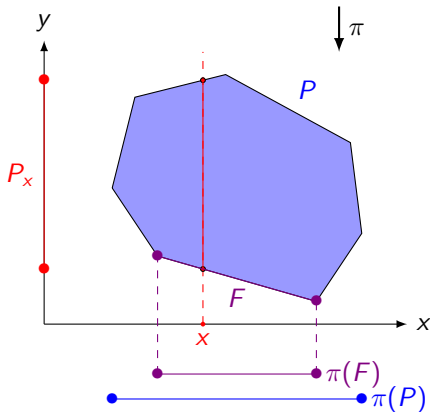
The *chamber complex* $\mathcal{C}(P, \pi)$ of P along π is

$$\mathcal{C}(P, \pi) := \{\sigma_{P, \pi}(x) \mid x \in \pi(P)\}$$

where

$$\sigma_{P, \pi}(x) := \bigcap_{F \in \mathcal{F}(P) \mid x \in \pi(F)} \pi(F)$$

where $\mathcal{F}(P)$ is the set of faces of P and π is the projection $(x, y) \mapsto x$.



Chamber complex

Definition (Billera, Sturmfels 92)

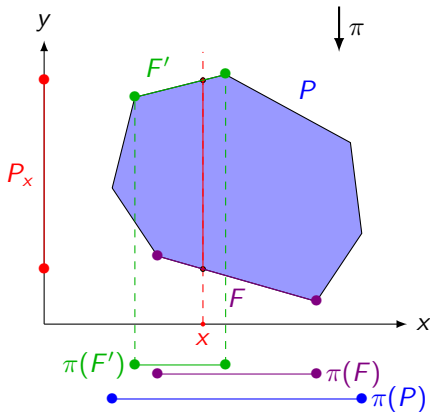
The *chamber complex* $\mathcal{C}(P, \pi)$ of P along π is

$$\mathcal{C}(P, \pi) := \{\sigma_{P, \pi}(x) \mid x \in \pi(P)\}$$

where

$$\sigma_{P, \pi}(x) := \bigcap_{F \in \mathcal{F}(P) \mid x \in \pi(F)} \pi(F)$$

where $\mathcal{F}(P)$ is the set of faces of P and π is the projection $(x, y) \mapsto x$.



Chamber complex

Definition (Billera, Sturmfels 92)

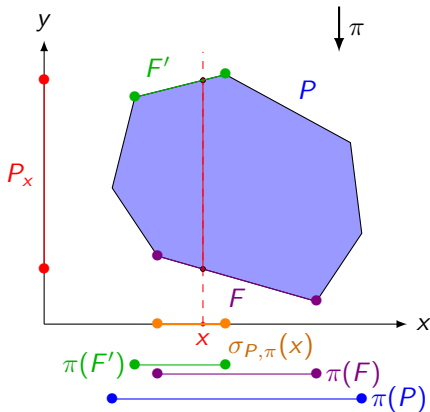
The *chamber complex* $\mathcal{C}(P, \pi)$ of P along π is

$$\mathcal{C}(P, \pi) := \{\sigma_{P, \pi}(x) \mid x \in \pi(P)\}$$

where

$$\sigma_{P, \pi}(x) := \bigcap_{F \in \mathcal{F}(P) \mid x \in \pi(F)} \pi(F)$$

where $\mathcal{F}(P)$ is the set of faces of P and π is the projection $(x, y) \mapsto x$.



Chamber complex

Definition (Billera, Sturmfels 92)

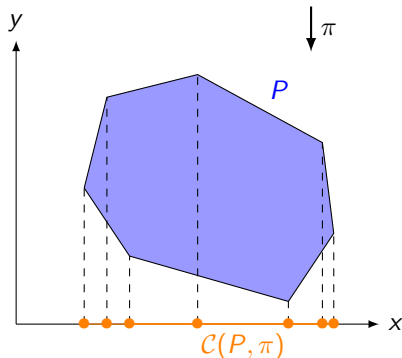
The *chamber complex* $\mathcal{C}(P, \pi)$ of P along π is

$$\mathcal{C}(P, \pi) := \{\sigma_{P, \pi}(x) \mid x \in \pi(P)\}$$

where

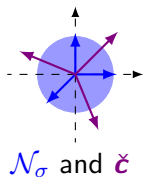
$$\sigma_{P, \pi}(x) := \bigcap_{F \in \mathcal{F}(P) \mid x \in \pi(F)} \pi(F)$$

where $\mathcal{F}(P)$ is the set of faces of P and π is the projection $(x, y) \mapsto x$.



Common Refinement of Normal Fans

We can quantize \mathbf{c} on each chamber.

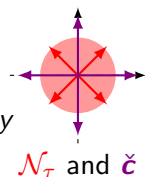


For all $x \in \text{ri}(\sigma)$,

$$V(x) = \sum_{N \in \mathcal{N}_\sigma} p_N \min_{y \in P_x} \check{\mathbf{c}}_N^\top y$$

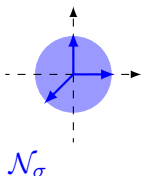
For all $x' \in \text{ri}(\tau)$,

$$V(x') = \sum_{N \in \mathcal{N}_\tau} p_N \min_{y \in P_x} \check{\mathbf{c}}_N^\top y$$



Common Refinement of Normal Fans

We can quantize \mathbf{c} on each chamber.

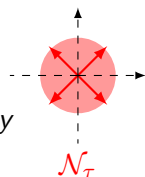


For all $x \in \text{ri}(\sigma)$,

$$V(x) = \sum_{N \in \mathcal{N}_\sigma} p_N \min_{y \in P_x} \check{c}_N^\top y$$

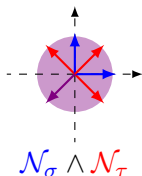
For all $x' \in \text{ri}(\tau)$,

$$V(x') = \sum_{N \in \mathcal{N}_\tau} p_N \min_{y \in P_x} \check{c}_N^\top y$$



We take the *common refinement*:

$$\mathcal{R} := \mathcal{N}_\sigma \wedge \mathcal{N}_\tau = \{N \cap N' \mid N \in \mathcal{N}_\sigma, N' \in \mathcal{N}_\tau\}$$

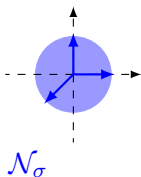


For all $x \in \text{ri}(\sigma) \cup \text{ri}(\tau)$,

$$V(x) = \sum_{N \in \mathcal{N}_\sigma \wedge \mathcal{N}_\tau} p_N \min_{y \in P_x} \check{c}_N^\top y$$

Common Refinement of Normal Fans

We can quantize \mathbf{c} on each chamber.

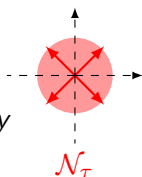


For all $x \in \text{ri}(\sigma)$,

$$V(x) = \sum_{N \in \mathcal{N}_\sigma} p_N \min_{y \in P_x} \check{c}_N^\top y$$

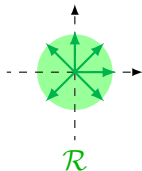
For all $x' \in \text{ri}(\tau)$,

$$V(x') = \sum_{N \in \mathcal{N}_\tau} p_N \min_{y \in P_x} \check{c}_N^\top y$$



We take the *common refinement*:

$$\mathcal{R} := \mathcal{N}_\sigma \wedge \mathcal{N}_\tau = \{N \cap N' \mid N \in \mathcal{N}_\sigma, N' \in \mathcal{N}_\tau\}$$



For all $x \in \text{ri}(\sigma) \cup \text{ri}(\tau)$,

$$V(x) = \sum_{N \in \mathcal{R}} p_N \min_{y \in P_x} \check{c}_N^\top y$$

Uniform exact quantization for \mathcal{C}

Let's sum up:

- local exact quantization at x induced by $\mathcal{N}(P_x)$,
- $x \mapsto \mathcal{N}(P_x)$ is constant on each $\sigma \in \mathcal{C}(P, \pi)$,
- local exact quantization at $\text{ri}(\sigma)$ induced by \mathcal{N}_σ ,
- local exact quantization at $\text{ri}(\sigma) \cup \text{ri}(\tau)$ induced by $\mathcal{N}_\sigma \wedge \mathcal{N}_\tau$.

Uniform exact quantization for \mathbf{c}

Let's sum up:

- local exact quantization at x induced by $\mathcal{N}(P_x)$,
- $x \mapsto \mathcal{N}(P_x)$ is constant on each $\sigma \in \mathcal{C}(P, \pi)$,
- local exact quantization at $\text{ri}(\sigma)$ induced by \mathcal{N}_σ ,
- local exact quantization at $\text{ri}(\sigma) \cup \text{ri}(\tau)$ induced by $\mathcal{N}_\sigma \wedge \mathcal{N}_\tau$.

Theorem (FGL21, Uniform and universal quantization of the cost)

Let $\mathcal{R} = \bigwedge_{\sigma \in \mathcal{C}(P, \pi)} -\mathcal{N}_\sigma$, then **for all** $x \in \mathbb{R}^n$

$$V(x) = \sum_{R \in \mathcal{R}} \check{p}_R \min_{y \in P_x} \check{c}_R^\top y$$

where $\check{p}_R := \mathbb{P}[\mathbf{c} \in \text{ri}(R)]$ and $\check{c}_R := \mathbb{E}[\mathbf{c} \mid \mathbf{c} \in \text{ri}(R)]$

Polyhedral characterization of V

Theorem (FGL 2021)

For all distributions of \mathbf{c} , V is affine on each cell of $\mathcal{C}(P, \pi)$.

Polyhedral characterization of V

Theorem (FGL 2021)

For all distributions of \mathbf{c} , V is affine on each cell of $\mathcal{C}(P, \pi)$.

Theorem (FGL 2021)

Under an affine change of variable, V is the support function of E

$$V(x) = \sigma_E(b - Bx) = \sup_{\lambda \in E} (b - Bx)^\top \lambda$$

Polyhedral characterization of V

Theorem (FGL 2021)

For all distributions of \mathbf{c} , V is affine on each cell of $\mathcal{C}(P, \pi)$.

Theorem (FGL 2021)

Under an affine change of variable, V is the support function of E

$$V(x) = \sigma_E(b - Bx) = \sup_{\lambda \in E} (b - Bx)^\top \lambda$$

*where $E := \mathbb{E}[D_{\mathbf{c}}] = \int D_{\mathbf{c}} \mathbb{P}(d\mathbf{c})$ is the **weighted fiber polyhedron** and $D_{\mathbf{c}} := \{\lambda \mid A^\top \lambda + \mathbf{c} = 0\}$ the dual admissible set.*

The weighted fiber polyhedron is a Minkowski integral with respect to the distribution $d\mathbb{P}(\mathbf{c})$

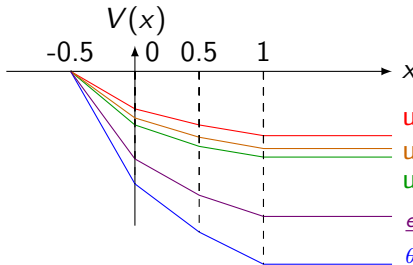
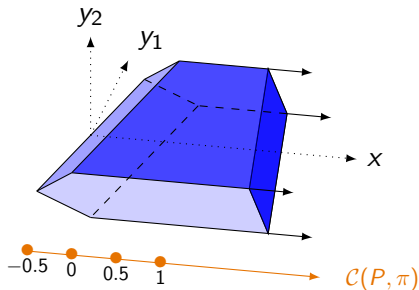
\rightsquigarrow extension of **fiber polytope** (uniform distribution) of



L. Billera, B. Sturmfels, Fiber polytopes, *Annals of Mathematics*, p527–549, 1992.

Explicit computation of the example

$$V(x) = \mathbb{E} \left[\begin{array}{ll} \min_{y \in \mathbb{R}^2} & \mathbf{c}^\top y \\ \text{s.t.} & \|y\|_1 \leq 1 \\ & y_1 \leq x \\ & y_2 \leq x \end{array} \right]$$



Different distributions of \mathbf{c} :

uniform on norm 1 ball

uniform on norm 2 ball

uniform on norm ∞ ball

$$\frac{e^{-\frac{\|\mathbf{c}\|_2^2}{2\gamma^2}}}{2\pi\gamma^2} d\mathbf{c}$$

$$\frac{\theta^2 e^{-\theta\|\mathbf{c}\|_1}}{4} d\mathbf{c}$$

Contents

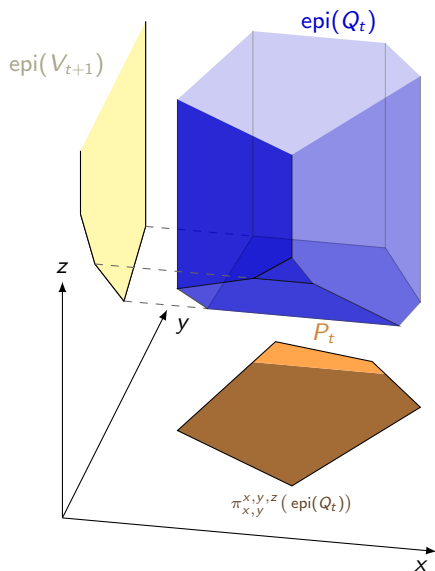
- 1 Universal Exact Quantization for cost
 - Local in 2-stage
 - Uniform in 2-stage
 - **Uniform in multistage**
 - Complexity results
- 2 Local and universal exact Quantization for constraints
 - Adapted partitions
 - Adaptive Partition-based Methods
 - Convergence, complexity and numerical results
- 3 Trajectory Following Dynamic Programming
- 4 Conclusion and perspectives

Multistage uniform and universal exact quantization

$$V_t(x) = \mathbb{E} \left[\min_{y \in \mathbb{R}^{n_t}} \mathbf{c}_t^\top y + V_{t+1}(y) \right]$$

s.t. $(x, y) \in P_t$

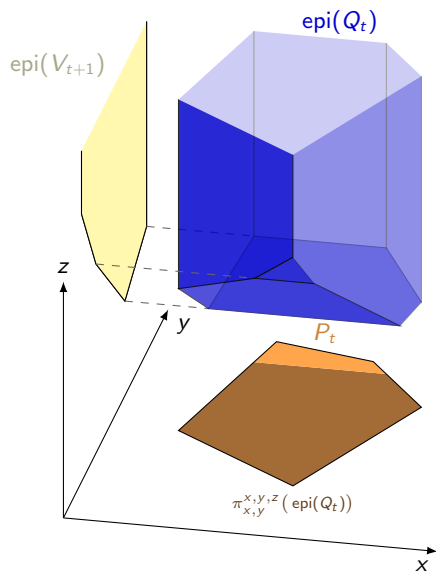
with $Q_t(x, y) := V_{t+1}(y) + \mathbb{I}_{(x, y) \in P_t}$.



Multistage uniform and universal exact quantization

$$V_t(x) = \mathbb{E} \left[\min_{\substack{y \in \mathbb{R}^{n_t} \\ z \in \mathbb{R}}} \mathbf{c}_t^\top y + z \right. \\ \left. \text{s.t. } (x, y, z) \in \text{epi}(Q_t) \right]$$

with $Q_t(x, y) := V_{t+1}(y) + \mathbb{I}_{(x,y) \in P_t}$.

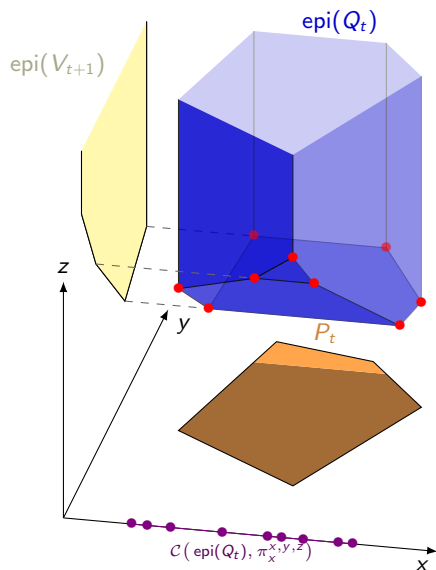


Multistage uniform and universal exact quantization

$$V_t(x) = \mathbb{E} \left[\begin{array}{l} \min_{\substack{y \in \mathbb{R}^{n_t} \\ z \in \mathbb{R}}} \mathbf{c}_t^\top y + z \\ \text{s.t. } (x, y, z) \in \text{epi}(Q_t) \end{array} \right]$$

with $Q_t(x, y) := V_{t+1}(y) + \mathbb{I}_{(x,y) \in P_t}$.

➡ V_t affine, $x \mapsto \mathcal{N}(P_x)$ constant
on $\mathcal{C}(\text{epi}(Q_t), \pi_x^{x,y,z})$



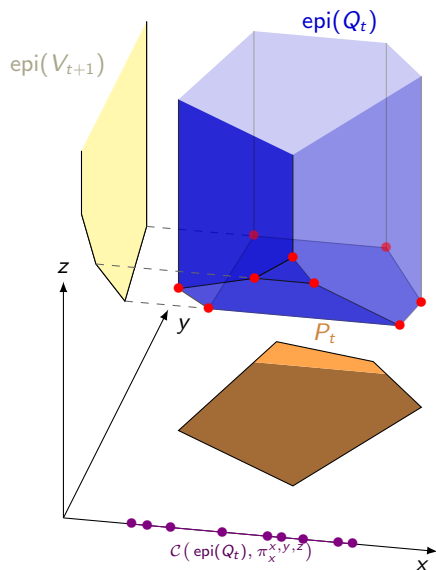
Multistage uniform and universal exact quantization

$$V_t(x) = \mathbb{E} \left[\begin{array}{ll} \min_{\substack{y \in \mathbb{R}^{n_t} \\ z \in \mathbb{R}}} & \mathbf{c}_t^\top y + z \\ \text{s.t.} & (x, y, z) \in \text{epi}(Q_t) \end{array} \right]$$

with $Q_t(x, y) := V_{t+1}(y) + \mathbb{I}_{(x,y) \in P_t}$.

➡ V_t affine, $x \mapsto \mathcal{N}(P_x)$ constant
on $\mathcal{C}(\text{epi}(Q_t), \pi_x^{x,y,z})$

⚠ $\text{epi}(Q_t)$ appears in the constraint
and depends on $\mathbf{c}_{t+1}, \dots, \mathbf{c}_T$!



Multistage uniform and universal exact quantization

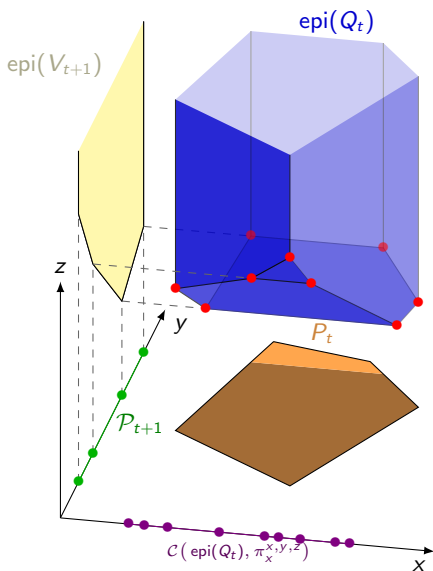
$$V_t(x) = \mathbb{E} \left[\begin{array}{ll} \min_{\substack{y \in \mathbb{R}^{n_t} \\ z \in \mathbb{R}}} & \mathbf{c}_t^\top y + z \\ \text{s.t.} & (x, y, z) \in \text{epi}(Q_t) \end{array} \right]$$

with $Q_t(x, y) := V_{t+1}(y) + \mathbb{I}_{(x,y) \in P_t}$.

➡ V_t affine, $x \mapsto \mathcal{N}(P_x)$ constant
on $\mathcal{C}(\text{epi}(Q_t), \pi_x^{x,y,z})$

⚠ $\text{epi}(Q_t)$ appears in the constraint
and depends on $\mathbf{c}_{t+1}, \dots, \mathbf{c}_T$!

V_{t+1} affine on \mathcal{P}_{t+1} (by assumption)



Multistage uniform and universal exact quantization

$$V_t(x) = \mathbb{E} \left[\begin{array}{ll} \min_{\substack{y \in \mathbb{R}^{n_t} \\ z \in \mathbb{R}}} & \mathbf{c}_t^\top y + z \\ \text{s.t.} & (x, y, z) \in \text{epi}(Q_t) \end{array} \right]$$

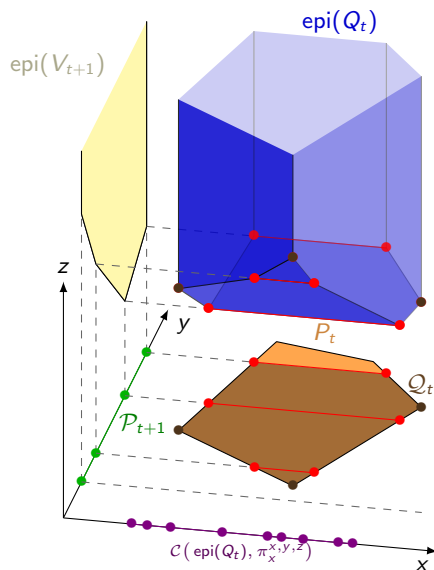
with $Q_t(x, y) := V_{t+1}(y) + \mathbb{I}_{(x,y) \in P_t}$.

➡ V_t affine, $x \mapsto \mathcal{N}(P_x)$ constant
on $\mathcal{C}(\text{epi}(Q_t), \pi_x^{x,y,z})$

⚠ $\text{epi}(Q_t)$ appears in the constraint
and depends on $\mathbf{c}_{t+1}, \dots, \mathbf{c}_T$!

V_{t+1} affine on \mathcal{P}_{t+1} (by assumption)

$$Q_t := (\mathbb{R}^{n_t} \times \mathcal{P}_{t+1}) \wedge \mathcal{F}(P_t)$$



Multistage uniform and universal exact quantization

$$V_t(x) = \mathbb{E} \left[\begin{array}{ll} \min_{\substack{y \in \mathbb{R}^{n_t} \\ z \in \mathbb{R}}} & \mathbf{c}_t^\top y + z \\ \text{s.t.} & (x, y, z) \in \text{epi}(Q_t) \end{array} \right]$$

with $Q_t(x, y) := V_{t+1}(y) + \mathbb{I}_{(x,y) \in P_t}$.

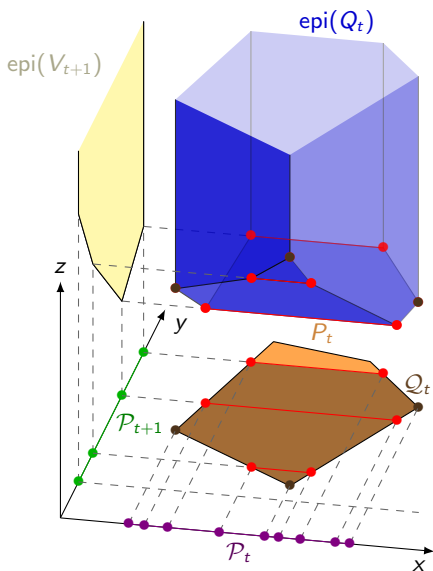
➡ V_t affine, $x \mapsto \mathcal{N}(P_x)$ constant
on $\mathcal{C}(\text{epi}(Q_t), \pi_x^{x,y,z})$

⚠ $\text{epi}(Q_t)$ appears in the constraint
and depends on $\mathbf{c}_{t+1}, \dots, \mathbf{c}_T$!

V_{t+1} affine on \mathcal{P}_{t+1} (by assumption)

$$Q_t := (\mathbb{R}^{n_t} \times \mathcal{P}_{t+1}) \cap \mathcal{F}(P_t)$$

$$\mathcal{P}_t := \mathcal{C}(Q_t, \pi_x^{x,y})$$



Multistage uniform and universal exact quantization

$$V_t(x) = \mathbb{E} \left[\begin{array}{ll} \min_{\substack{y \in \mathbb{R}^{n_t} \\ z \in \mathbb{R}}} & \mathbf{c}_t^\top y + z \\ \text{s.t.} & (x, y, z) \in \text{epi}(Q_t) \end{array} \right]$$

with $Q_t(x, y) := V_{t+1}(y) + \mathbb{I}_{(x,y) \in P_t}$.

➡ V_t affine, $x \mapsto \mathcal{N}(P_x)$ constant on $\mathcal{C}(\text{epi}(Q_t), \pi_x^{x,y,z})$

⚠ $\text{epi}(Q_t)$ appears in the constraint and depends on $\mathbf{c}_{t+1}, \dots, \mathbf{c}_T$!

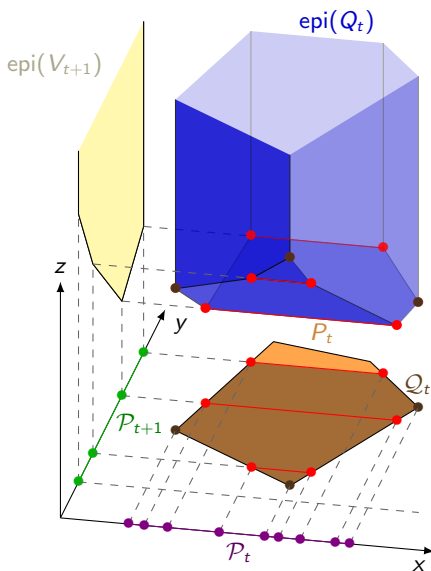
V_{t+1} affine on P_{t+1} (by assumption)

$$Q_t := (\mathbb{R}^{n_t} \times P_{t+1}) \wedge \mathcal{F}(P_t)$$

$$P_t := \mathcal{C}(Q_t, \pi_x^{x,y})$$

[FGL21, Lem. 4.1]: $P_t \preceq \mathcal{C}(\text{epi}(Q_t), \pi_x^{x,y,z})$

➡ V_t affine on P_t , $\mathcal{N}(P_x)$ constant on P_t



Extension to multistage and stochastic constraints

Iterated chamber complexes by backward induction

$$\mathcal{P}_{t,\xi} := \mathcal{C}\left((\mathbb{R}^{n_t} \times \mathcal{P}_{t+1}) \wedge \mathcal{F}(P_t(\xi)), \pi_{x_{t-1}}^{x_{t-1}, x_t}\right)$$
$$\mathcal{P}_t := \bigwedge_{\xi_t \in \text{supp } \xi_t} \mathcal{P}_{t,\xi}$$

Extension to multistage and stochastic constraints

Iterated chamber complexes by backward induction

$$\mathcal{P}_{t,\xi} := \mathcal{C}\left(\left(\mathbb{R}^{n_t} \times \mathcal{P}_{t+1}\right) \wedge \mathcal{F}\left(P_t(\xi)\right), \pi_{x_{t-1}}^{x_{t-1}, x_t}\right)$$
$$\mathcal{P}_t := \bigwedge_{\xi_t \in \text{supp } \xi_t} \mathcal{P}_{t,\xi}$$

Theorem (FGL 21)

All results generalizes to MSLP with finitely supported stochastic constraints.

- ➡ $(V_t)_t$ are affine on *universal* chamber complexes, i.e. independent of the law of $(\mathbf{c}_t)_t$
- ➡ We have an *uniform and universal* exact quantization.

Contents

1 Universal Exact Quantization for cost

- Local in 2-stage
- Uniform in 2-stage
- Uniform in multistage
- Complexity results

2 Local and universal exact Quantization for constraints

- Adapted partitions
- Adaptive Partition-based Methods
- Convergence, complexity and numerical results

3 Trajectory Following Dynamic Programming

4 Conclusion and perspectives

Earlier and new complexity results

Volume of a polytope

$$\text{Vol}(\{z \in \mathbb{R}^d \mid Az \leq b\}) \text{ or}$$
$$\text{Vol}(\text{Conv}(v_1, \dots, v_n))$$

- $\#P$ -complete:
Dyer and Frieze (1988)
- Polynomial for fixed dimension
 d : Lawrence (1991)

Earlier and new complexity results

Volume of a polytope

$$\text{Vol}(\{z \in \mathbb{R}^d \mid Az \leq b\}) \text{ or } \\ \text{Vol}(\text{Conv}(v_1, \dots, v_n))$$

- $\#P$ -complete:
Dyer and Frieze (1988)
- Polynomial for fixed dimension d : Lawrence (1991)

2-stage linear problem

$$\min_{x \in \mathbb{R}^n} c_1^\top x + \mathbb{E} \left[\begin{array}{l} \min_{y \in \mathbb{R}^m} c_2^\top y \\ \text{s.t. } A_2 y + B_2 x \leq b_2 \end{array} \right] \\ \text{s.t. } A_1 x \leq b_1$$

- $\#P$ -hard: Hanasusanto, Kuhn and Wiesemann (2016)
- Polynomial for fixed m ?

Earlier and new complexity results

Volume of a polytope

$$\text{Vol}(\{z \in \mathbb{R}^d \mid Az \leq b\}) \text{ or } \\ \text{Vol}(\text{Conv}(v_1, \dots, v_n))$$

- $\#P$ -complete:
Dyer and Frieze (1988)
- Polynomial for fixed dimension d : Lawrence (1991)

2-stage linear problem

$$\min_{x \in \mathbb{R}^n} c_1^\top x + \mathbb{E} \left[\begin{array}{l} \min_{y \in \mathbb{R}^m} c_2^\top y \\ \text{s.t. } A_2 y + B_2 x \leq b_2 \end{array} \right] \\ \text{s.t. } A_1 x \leq b_1$$

- $\#P$ -hard: Hanasusanto, Kuhn and Wiesemann (2016)
- **Polynomial for fixed m :**
FGL (2021)

Earlier and new complexity results

Volume of a polytope

$$\text{Vol}(\{z \in \mathbb{R}^d \mid Az \leq b\}) \text{ or } \\ \text{Vol}(\text{Conv}(v_1, \dots, v_n))$$

- $\#P$ -complete:
Dyer and Frieze (1988)
- Polynomial for fixed dimension d : Lawrence (1991)

2-stage linear problem

$$\min_{x \in \mathbb{R}^n} c_1^\top x + \mathbb{E} \left[\begin{array}{l} \min_{y \in \mathbb{R}^m} c_2^\top y \\ \text{s.t. } A_2 y + B_2 x \leq b_2 \end{array} \right] \\ \text{s.t. } A_1 x \leq b_1$$

- $\#P$ -hard: Hanasusanto, Kuhn and Wiesemann (2016)
- **Polynomial for fixed m :**
FGL (2021)
 - \rightsquigarrow Exact case
 - \rightsquigarrow Approximated case

Complexity result multistage

Theorem (FGL21: MSLP is polynomial for fixed dimensions)

Assume that T, n_2, \dots, n_T , are fixed.¹

Assume that \mathbf{c} admits a density function with a bounded total variation.

*Then, there exists an algorithm that finds an ε -solution² in **polynomial** time in $\log(\frac{1}{\varepsilon})$ with **probability 1**.*

¹No requirement for the first decision.

²Or asserts that MSLP is unfeasible.

Complexity result multistage

Theorem (FGL21: MSLP is polynomial for fixed dimensions)

Assume that T, n_2, \dots, n_T , are fixed.¹

Assume that \mathbf{c} admits a density function with a bounded total variation.

*Then, there exists an algorithm that finds an ε -solution² in **polynomial** time in $\log(\frac{1}{\varepsilon})$ with **probability 1**.*

➡ Can be adapted to exact complexity when we can compute exactly $\mathbb{E}[\mathbf{c} | \mathbf{c} \in C, (\mathbf{A}_t, \mathbf{B}_t, \mathbf{b}_t) = (A, B, b)]$ and $\mathbb{P}[\mathbf{c} \in C | (\mathbf{A}_t, \mathbf{B}_t, \mathbf{b}_t) = (A, B, b)]$.

¹No requirement for the first decision.

²Or asserts that MSLP is unfeasible.

Complexity result multistage

Theorem (FGL21: MSLP is polynomial for fixed dimensions)

Assume that T, n_2, \dots, n_T , are fixed.¹

Assume that \mathbf{c} admits a density function with a bounded total variation.

Then, there exists an algorithm that finds an ε -solution² in **polynomial** time in $\log(\frac{1}{\varepsilon})$ with **probability 1**.

➡ Can be adapted to exact complexity when we can compute exactly $\mathbb{E}[\mathbf{c} | \mathbf{c} \in C, (\mathbf{A}_t, \mathbf{B}_t, \mathbf{b}_t) = (A, B, b)]$ and $\mathbb{P}[\mathbf{c} \in C | (\mathbf{A}_t, \mathbf{B}_t, \mathbf{b}_t) = (A, B, b)]$.

Proof based on ellipsoid (Grötschel, Lovász, Schrijver)
and upper bound theorems (McMullen, Stanley)



¹No requirement for the first decision.

²Or asserts that MSLP is unfeasible.

Complexity result multistage

Theorem (FGL21: MSLP is polynomial for fixed dimensions)

Assume that T, n_2, \dots, n_T , are fixed.¹

Assume that \mathbf{c} admits a density function with a bounded total variation.

Then, there exists an algorithm that finds an ε -solution² in **polynomial** time in $\log(\frac{1}{\varepsilon})$ with **probability 1**.

➡ Can be adapted to exact complexity when we can compute exactly $\mathbb{E}[\mathbf{c} | \mathbf{c} \in C, (\mathbf{A}_t, \mathbf{B}_t, \mathbf{b}_t) = (A, B, b)]$ and $\mathbb{P}[\mathbf{c} \in C | (\mathbf{A}_t, \mathbf{B}_t, \mathbf{b}_t) = (A, B, b)]$.

Proof based on ellipsoid (Grötschel, Lovász, Schrijver)
and upper bound theorems (McMullen, Stanley)



By SAA, we can solve MSLP, up to precision ε , in **pseudo-polynomial** time, i.e. polynomial in $\frac{1}{\varepsilon}$, with **probability $1 - \alpha$** , when T, n_1, \dots, n_T are fixed.

¹No requirement for the first decision.

²Or asserts that MSLP is unfeasible.

Contents

- 1 Universal Exact Quantization for cost
 - Local in 2-stage
 - Uniform in 2-stage
 - Uniform in multistage
 - Complexity results
- 2 Local and universal exact Quantization for constraints
 - Adapted partitions
 - Adaptive Partition-based Methods
 - Convergence, complexity and numerical results
- 3 Trajectory Following Dynamic Programming
- 4 Conclusion and perspectives

Local exact quantization for constraints ?

Back to the 2-stage problem

	A	(B, b)	c
Local	×	?	✓
Uniform	×	×	✓

Duality result

$$V(x) = \mathbb{E}[V(x, \xi)] = \mathbb{E} \left[\begin{array}{ll} \min_{y \in \mathbb{R}^n} & c^\top y \\ \text{s.t.} & Ay + Bx \leq b \end{array} \right] = \mathbb{E} \left[\begin{array}{ll} \max_{\lambda \in \mathbb{R}^\ell} & (Bx - b)^\top \lambda \\ \text{s.t.} & A^\top \lambda + c = 0 \end{array} \right]$$

➡ Back to the case with random cost

⚠ The new cost depends on x : only local exact quantization.

Local exact quantization for constraints ?

Back to the 2-stage problem

	A	(B, b)	c
Local	×	?	✓
Uniform	×	×	✓

Duality result

$$V(x) = \mathbb{E}[V(x, \xi)] = \mathbb{E} \left[\begin{array}{ll} \min_{y \in \mathbb{R}^n} & c^\top y \\ \text{s.t.} & Ay + Bx \leq b \end{array} \right] = \mathbb{E} \left[\begin{array}{ll} \max_{\lambda \in \mathbb{R}^\ell} & (Bx - b)^\top \lambda \\ \text{s.t.} & A^\top \lambda + c = 0 \end{array} \right]$$

➡ Back to the case with random cost

⚠ The new cost depends on x : only local exact quantization.

Local exact quantization for constraints ?

Back to the 2-stage problem

	A	(B, b)	c
Local	×	?	✓
Uniform	×	×	✓

Duality result

$$V(x) = \mathbb{E}[V(x, \xi)] = \mathbb{E} \left[\begin{array}{ll} \min_{y \in \mathbb{R}^n} & c^\top y \\ \text{s.t.} & Ay + Bx \leq b \end{array} \right] = \mathbb{E} \left[\begin{array}{ll} \max_{\lambda \in \mathbb{R}^\ell} & (Bx - b)^\top \lambda \\ \text{s.t.} & A^\top \lambda + c = 0 \end{array} \right]$$

➡ Back to the case with random cost

⚠ The new cost depends on x : only local exact quantization.

Local exact quantization for constraints ?

Back to the 2-stage problem

	A	(B, b)	c
Local	×	?	✓
Uniform	×	×	✓

Duality result

$$V(x) = \mathbb{E}[V(x, \xi)] = \mathbb{E} \left[\begin{array}{ll} \min_{y \in \mathbb{R}^n} & c^\top y \\ \text{s.t.} & Ay + Bx \leq b \end{array} \right] = \mathbb{E} \left[\begin{array}{ll} \max_{\lambda \in \mathbb{R}^\ell} & (Bx - b)^\top \lambda \\ \text{s.t.} & A^\top \lambda + c = 0 \end{array} \right]$$

➡ Back to the case with random cost

⚠ The new cost depends on x : only local exact quantization.

Local exact quantization for constraints

Random cost

Recall that for a fixed x ,

$$\begin{aligned} V(x) &= \mathbb{E} \left[\min_{y \in P_x} \mathbf{c}^\top y \right] \\ &= \sum_{N \in \mathcal{N}(P_x)} p_N \min_{y \in P_x} \check{\mathbf{c}}_N^\top y \end{aligned}$$

where,

$$\begin{aligned} p_N &:= \mathbb{P}[\mathbf{c} \in -\text{ri } N] \\ \check{\mathbf{c}}_N &:= \mathbb{E}[\mathbf{c} \mid \mathbf{c} \in -\text{ri } N] \end{aligned}$$

$$P_x := \{y \in \mathbb{R}^m \mid Ay + Bx \leq b\}$$

Random constraints

Similarly, for a given c and x ,

$$\begin{aligned} V(x) &= \mathbb{E} \left[\max_{\lambda \in D_c} (\mathbf{b} - \mathbf{B}x)^\top \lambda \right] \\ &= \sum_{N \in \mathcal{N}(D_c)} p_{N,x} \max_{\lambda \in D_c} \psi_{N,x}^\top \lambda \end{aligned}$$

where,

$$\begin{aligned} p_{N,x} &:= \mathbb{P}[\mathbf{b} - \mathbf{B}x \in \text{ri } N] \\ \psi_{N,x} &:= \mathbb{E}[\mathbf{b} - \mathbf{B}x \mid \mathbf{b} - \mathbf{B}x \in \text{ri } N] \end{aligned}$$

$$D_c := \{\lambda \in \mathbb{R}^l \mid A^\top \lambda + c = 0\}$$

Local exact quantization for constraints

Random cost

Recall that for a fixed x ,

$$\begin{aligned} V(x) &= \mathbb{E} \left[\min_{y \in P_x} \mathbf{c}^\top y \right] \\ &= \sum_{N \in \mathcal{N}(P_x)} p_N \min_{y \in P_x} \check{\mathbf{c}}_N^\top y \end{aligned}$$

where,

$$\begin{aligned} p_N &:= \mathbb{P}[\mathbf{c} \in -\text{ri } N] \\ \check{\mathbf{c}}_N &:= \mathbb{E}[\mathbf{c} \mid \mathbf{c} \in -\text{ri } N] \end{aligned}$$

$$P_x := \{y \in \mathbb{R}^m \mid Ay + Bx \leq b\}$$

Random constraints

Similarly, for a given c and x ,

$$\begin{aligned} V(x) &= \mathbb{E} \left[\max_{\lambda \in D_c} (\mathbf{b} - \mathbf{B}x)^\top \lambda \right] \\ &= \sum_{N \in \mathcal{N}(D_c)} p_{N,x} \max_{\lambda \in D_c} \psi_{N,x}^\top \lambda \end{aligned}$$

where,

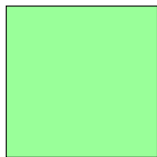
$$\begin{aligned} p_{N,x} &:= \mathbb{P}[\mathbf{b} - \mathbf{B}x \in \text{ri } N] \\ \psi_{N,x} &:= \mathbb{E}[\mathbf{b} - \mathbf{B}x \mid \mathbf{b} - \mathbf{B}x \in \text{ri } N] \end{aligned}$$

$$D_c := \{\lambda \in \mathbb{R}^l \mid A^\top \lambda + c = 0\}$$

Contents

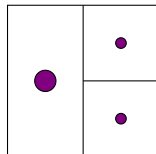
- 1 Universal Exact Quantization for cost
 - Local in 2-stage
 - Uniform in 2-stage
 - Uniform in multistage
 - Complexity results
- 2 Local and universal exact Quantization for constraints
 - **Adapted partitions**
 - Adaptive Partition-based Methods
 - Convergence, complexity and numerical results
- 3 Trajectory Following Dynamic Programming
- 4 Conclusion and perspectives

Partitioned cost-to-go functions (recalls)



ξ_t continuous

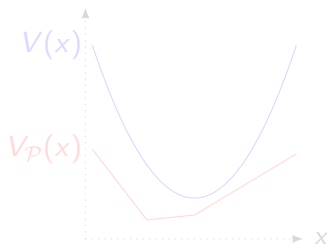
$$V(x) = \mathbb{E} \left[\hat{V}(x, \xi) \right]$$



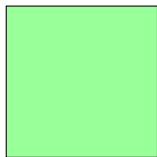
ξ_t partitioned

$$V_{\mathcal{P}}(x) = \sum_{P \in \mathcal{P}} \mathbb{P}[P] \hat{V}(x, \mathbb{E}[\xi|P])$$

- $\hat{V}(x, \cdot)$ is convex
 $\Rightarrow V_{\mathcal{P}} \leq V$.
- $\hat{V}(\cdot, \mathbb{E}[\xi|P])$ is polyhedral
 $\Rightarrow V_{\mathcal{P}}$ is polyhedral.

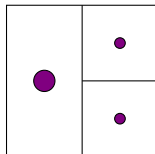


Partitioned cost-to-go functions (recalls)



ξ_t continuous

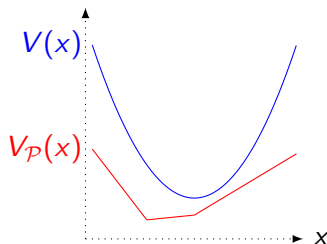
$$V(x) = \mathbb{E} \left[\hat{V}(x, \xi) \right]$$



$\check{\xi}_t$ partitioned

$$V_{\mathcal{P}}(x) = \sum_{P \in \mathcal{P}} \mathbb{P}[P] \hat{V}(x, \mathbb{E}[\xi|P])$$

- $\hat{V}(x, \cdot)$ is convex
 $\Rightarrow V_{\mathcal{P}} \leq V$.
- $\hat{V}(\cdot, \mathbb{E}[\xi|P])$ is polyhedral
 $\Rightarrow V_{\mathcal{P}}$ is polyhedral.

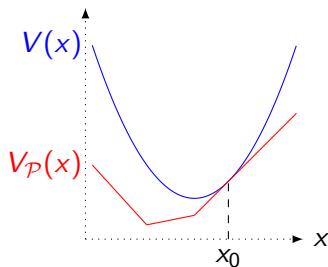


Adapted partition

Definition

A partition \mathcal{P} is *adapted* to x_0 if

$$V_{\mathcal{P}}(x_0) = V(x_0) := \mathbb{E}[\hat{V}(x_0, \xi)]$$



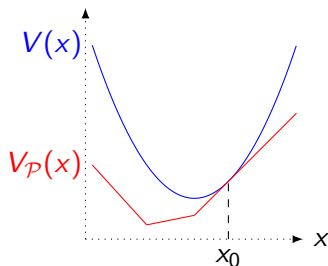
¹Can be extended to generic random \mathbf{c} and finitely supported \mathbf{A}

Adapted partition

Definition

A partition \mathcal{P} is *adapted* to x_0 if

$$V_{\mathcal{P}}(x_0) = V(x_0) := \mathbb{E}[\hat{V}(x_0, \xi)]$$



Consider $x \in \mathbb{R}^n$ and $N \in \mathcal{N}(D_q)$ a normal cone of D_q . We define

$$E_{N,x} := \{\xi \in \Xi \mid b - Bx \in \text{ri } N\}$$

Theorem (FL 2021)

$\mathcal{R}_x := \{E_{N,x} \mid N \in \mathcal{N}(D_q)\}$ is adapted to x i.e. $V_{\mathcal{R}_x}(x) = V(x)$

In particular: if only B and b are stochastic,

then there exists a *universal and local* exact quantization¹.

Bonus: necessary and sufficient condition for a partition to be adapted

¹Can be extended to generic random c and finitely supported A

Contents

- 1 Universal Exact Quantization for cost
 - Local in 2-stage
 - Uniform in 2-stage
 - Uniform in multistage
 - Complexity results
- 2 Local and universal exact Quantization for constraints
 - Adapted partitions
 - **Adaptive Partition-based Methods**
 - Convergence, complexity and numerical results
- 3 Trajectory Following Dynamic Programming
- 4 Conclusion and perspectives

General framework for Adaptive Partition-based Methods

```
 $\mathcal{P}^0 \leftarrow \{\Xi\} ;$   
for  $k = 1 \dots \infty$  do  
    | Let  $x^k$  be an optimal solution  $\min_{x \in X} c_1^\top x + V_{\mathcal{P}^{k-1}}(x) ;$   
    | Let  $\mathcal{P}_{x^k}$  a partition adapted to  $x^k ;$   
    |  $\mathcal{P}^k \leftarrow \mathcal{P}^{k-1} \wedge \mathcal{P}_{x^k} ;$   
end
```

Algorithm 1: General framework for APM.

$$\min_{x \in X} c_1^\top x + V_{\mathcal{P}}(x)$$

is equivalent to

$$\begin{aligned} \min_{x \in X, (y_P)_{P \in \mathcal{P}}} \quad & c_1^\top x + \sum_{P \in \mathcal{P}} \mathbb{P}[P] c_2^\top y_P \\ & Ay_P + \mathbb{E}[\mathbf{B}|P]x \leq \mathbb{E}[\mathbf{b}|P] \quad , \forall P \in \mathcal{P} \end{aligned}$$

General framework for Adaptive Partition-based Methods

```
 $\mathcal{P}^0 \leftarrow \{\Xi\} ;$   
for  $k = 1 \dots \infty$  do  
    | Let  $x^k$  be an optimal solution  $\min_{x \in X} c_1^\top x + V_{\mathcal{P}^{k-1}}(x) ;$   
    | Let  $\mathcal{P}_{x^k}$  a partition adapted to  $x^k ;$   
    |  $\mathcal{P}^k \leftarrow \mathcal{P}^{k-1} \wedge \mathcal{P}_{x^k} ;$   
end
```

Algorithm 1: General framework for APM.

$$\min_{x \in X} c_1^\top x + V_{\mathcal{P}}(x)$$

is equivalent to

$$\begin{aligned} \min_{x \in X, (y_P)_{P \in \mathcal{P}}} \quad & c_1^\top x + \sum_{P \in \mathcal{P}} \mathbb{P}[P] c_2^\top y_P \\ & Ay_P + \mathbb{E}[\mathbf{B}|P]x \leq \mathbb{E}[\mathbf{b}|P] \quad , \forall P \in \mathcal{P} \end{aligned}$$

A (partial) comparison between partition based results

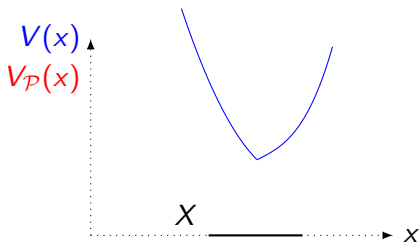
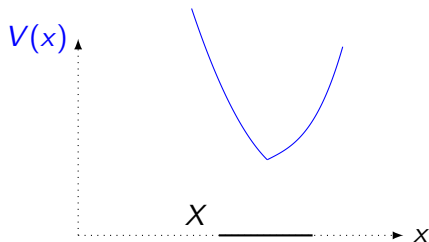
Paper	Song, Luedtke (2015)	Ramirez-Pico, Moreno (2020)	FL (2021)
Non-finite supp(ξ)	✗	✓	✓
Explicit oracle	✓	✗	✓
Proof of convergence	✓	✗	✓
Complexity result	✗	✗	✓
Fast iteration	✓	✗	✗

Contents

- 1 Universal Exact Quantization for cost
 - Local in 2-stage
 - Uniform in 2-stage
 - Uniform in multistage
 - Complexity results
- 2 Local and universal exact Quantization for constraints
 - Adapted partitions
 - Adaptive Partition-based Methods
 - Convergence, complexity and numerical results
- 3 Trajectory Following Dynamic Programming
- 4 Conclusion and perspectives

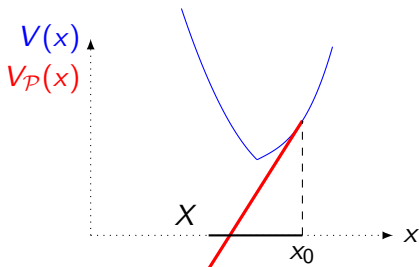
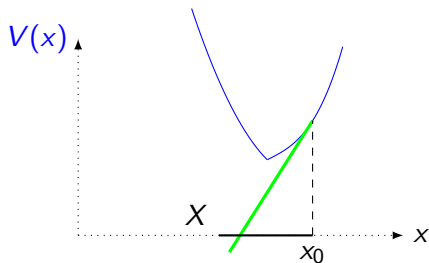
Link with Benders decomposition and L-shaped

Partition based method can be seen as a tangent cone method: a cutting plane method where we add all active cuts instead of a single one.



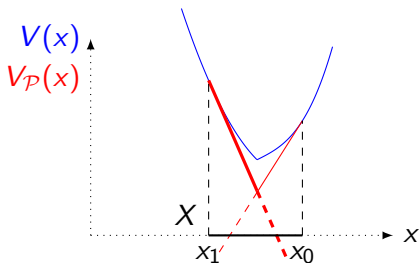
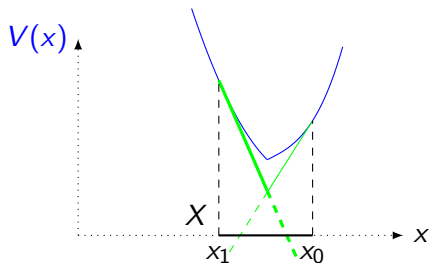
Link with Benders decomposition and L-shaped

Partition based method can be seen as a tangent cone method: a cutting plane method where we add all active cuts instead of a single one.



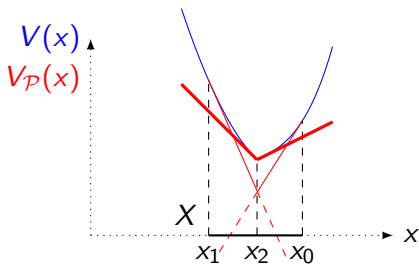
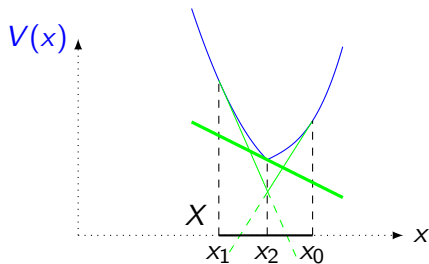
Link with Benders decomposition and L-shaped

Partition based method can be seen as a tangent cone method: a cutting plane method where we add all active cuts instead of a single one.



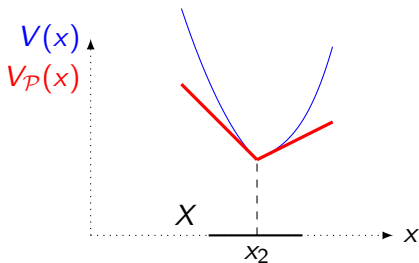
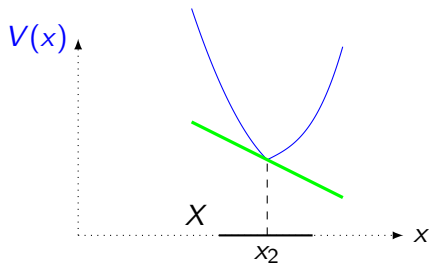
Link with Benders decomposition and L-shaped

Partition based method can be seen as a tangent cone method: a cutting plane method where we add all active cuts instead of a single one.



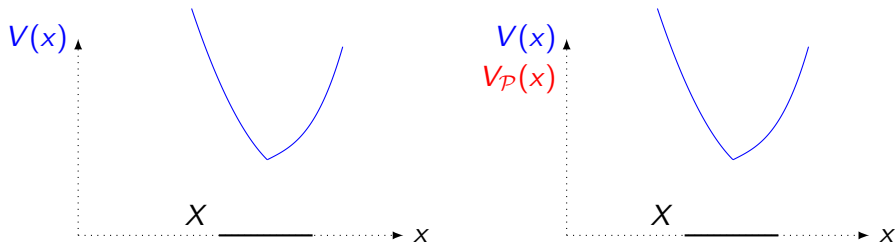
Link with Benders decomposition and L-shaped

Partition based method can be seen as a tangent cone method: a cutting plane method where we add all active cuts instead of a single one.



Link with Benders decomposition and L-shaped

Partition based method can be seen as a tangent cone method: a cutting plane method where we add all active cuts instead of a single one.



Theorem (Convergence and complexity results)

If $X \cap \text{dom}(V) \subset \mathbb{R}^+$ is contained in a ball of diameter $M \in \mathbb{R}^+$ and $x \rightarrow c_1^\top x + V(x)$ is Lipschitz with constant L then the partition based method finds an ε -solution in at most $(\frac{LM}{\varepsilon} + 1)^n$ iterations.

Numerical Results - ProdMix

k	x_k	z_L^k	z_U^k	Gap	$ \mathcal{P}_k^{\max} $
1	(1333.33, 66.67)	-18666.67	-16939.71	9.3%	4
2	(1441.41, 59.57)	-17873.01	-17383.73	2.7%	9
3	(1399.05, 57.91)	-17789.88	-17659.19	0.74%	16
4	(1379.98, 56.64)	-17744.67	-17708.00	0.20%	25
5	(1371.36, 55.71)	-17718.96	-17709.05	0.056%	36
6	(1375.55, 56.21)	-17713.74	-17711.37	0.013%	49

Table: Results for problem Prod-Mix

To compare our approach with SAA, we solved the same problem 100 times, each with 10 000 scenarios randomly drawn, yielding a 95% confidence interval centered in -17711 , with radius 2.2.

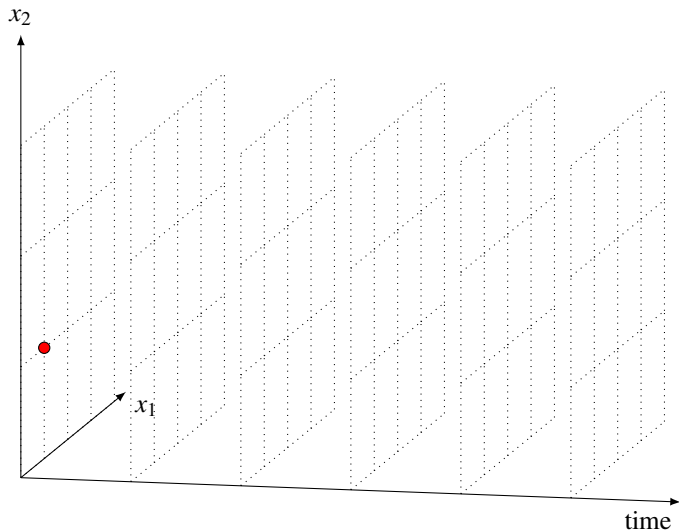
Contents

- 1 Universal Exact Quantization for cost
 - Local in 2-stage
 - Uniform in 2-stage
 - Uniform in multistage
 - Complexity results
- 2 Local and universal exact Quantization for constraints
 - Adapted partitions
 - Adaptive Partition-based Methods
 - Convergence, complexity and numerical results
- 3 Trajectory Following Dynamic Programming
- 4 Conclusion and perspectives

History of stochastic dual dynamic programming (SDDP)

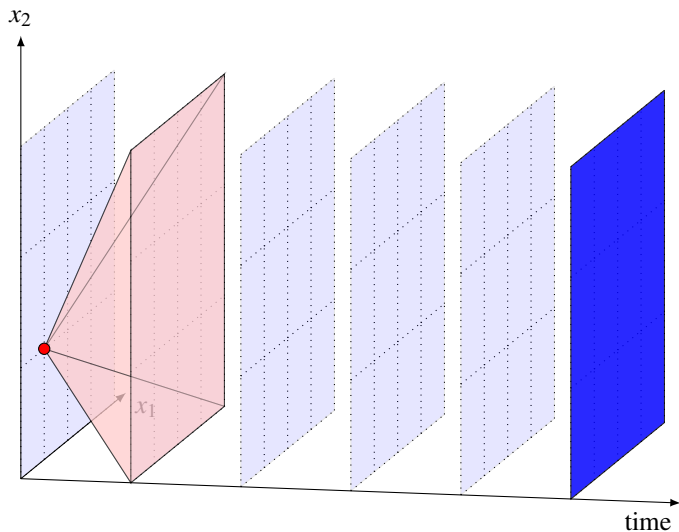
- Designed by Pereira and Pinto in 1991, used to manage brazilian hydroelectricity network
 - Proof of asymptotic convergence in the linear case (Philpott and Guan 2008) and in the convex case (Girardeau, Leclère, Philpott 2015)
 - Complexity proof (Lan 2020, Zhang and Sun 2022)
 - Plenty of variants: trajectory following dynamic programming algorithms
- ➡ All with finitely supported distribution

Trajectory Following Dynamic Programming



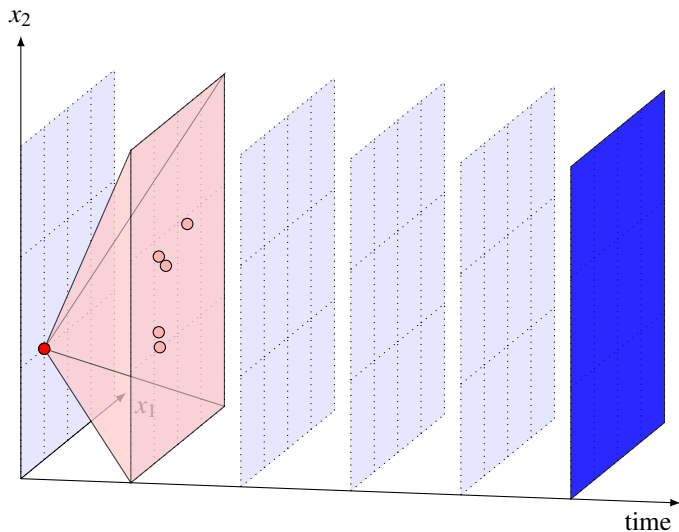
Thanks again Vincent !

Trajectory Following Dynamic Programming



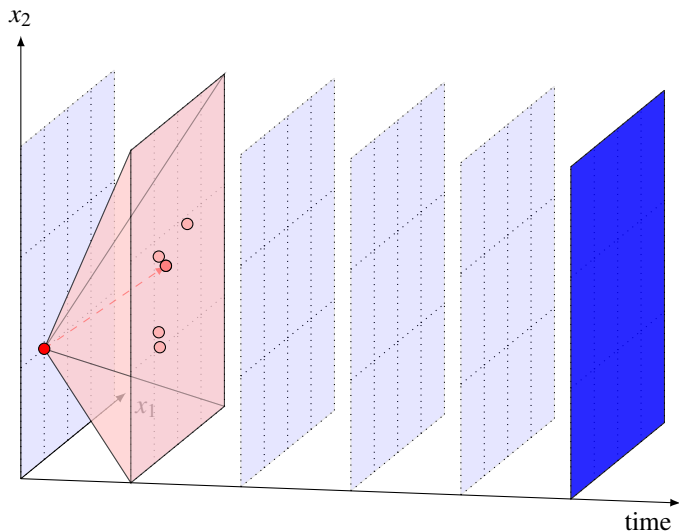
First forward pass : computing trajectory

Trajectory Following Dynamic Programming



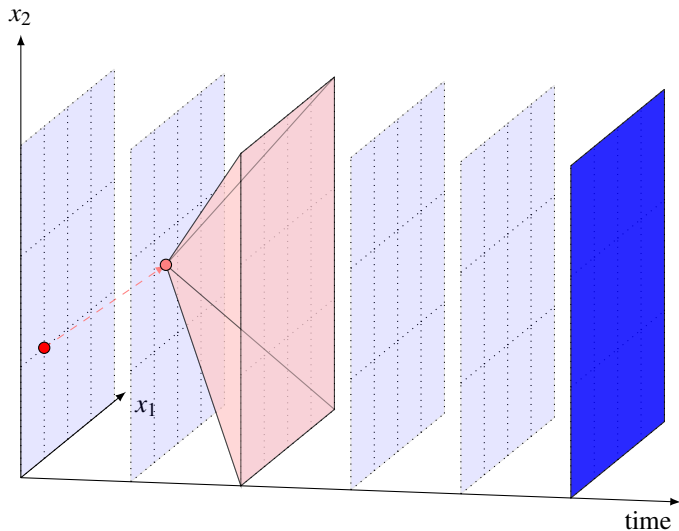
First forward pass : computing trajectory

Trajectory Following Dynamic Programming



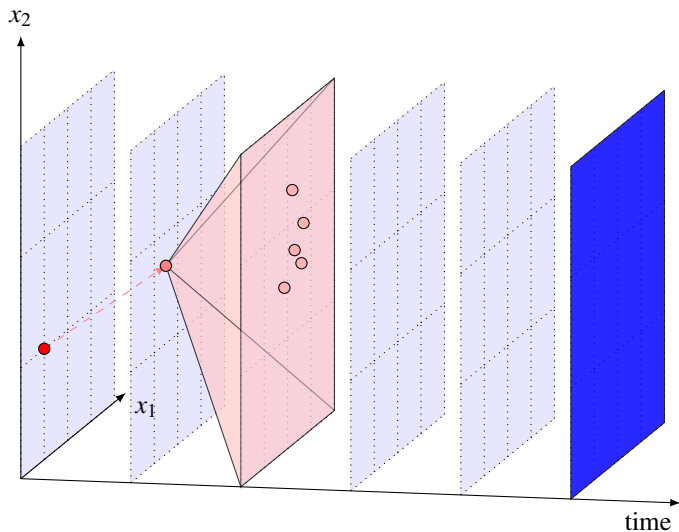
First forward pass : computing trajectory

Trajectory Following Dynamic Programming



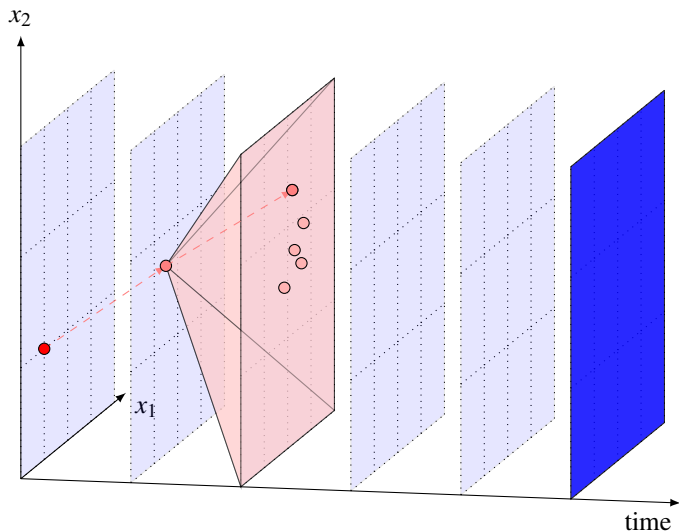
First forward pass : computing trajectory

Trajectory Following Dynamic Programming



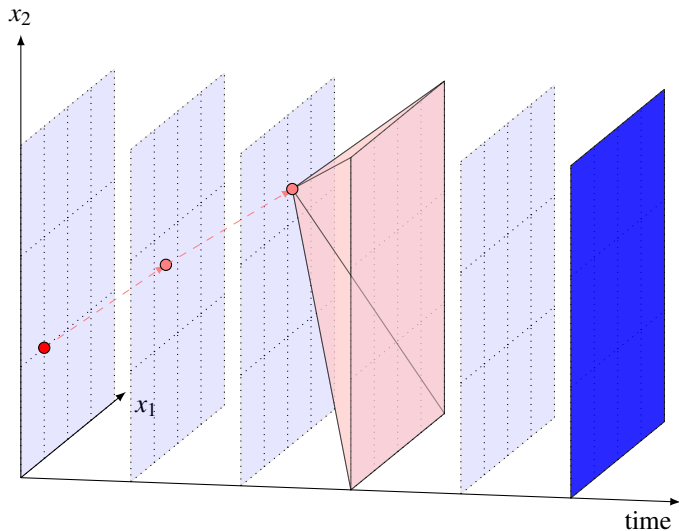
First forward pass : computing trajectory

Trajectory Following Dynamic Programming



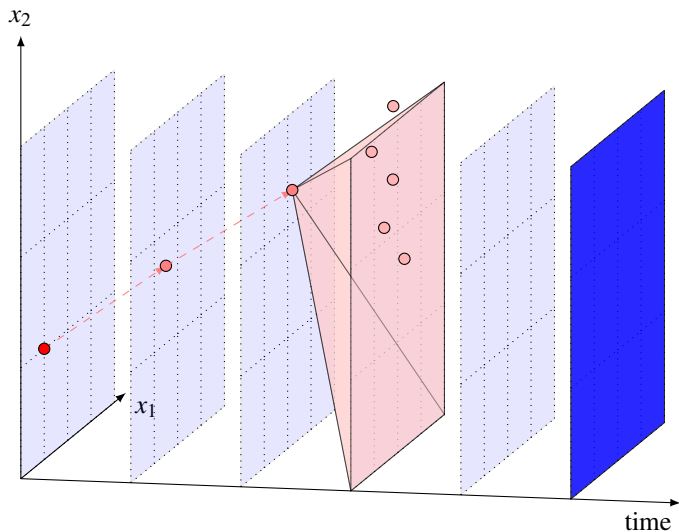
First forward pass : computing trajectory

Trajectory Following Dynamic Programming



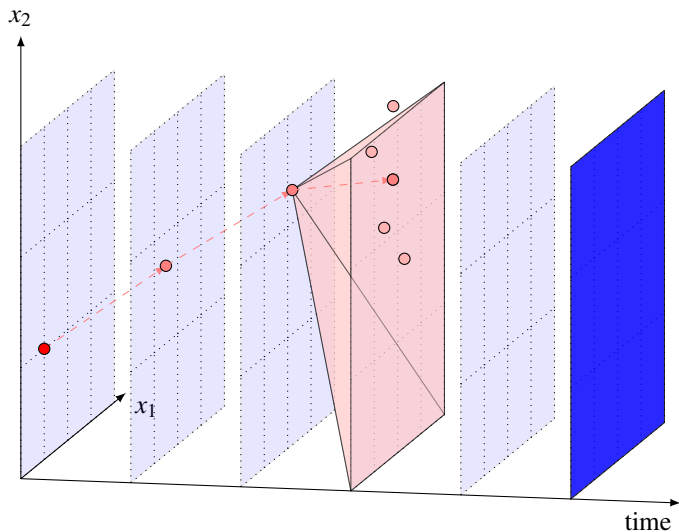
First forward pass : computing trajectory

Trajectory Following Dynamic Programming



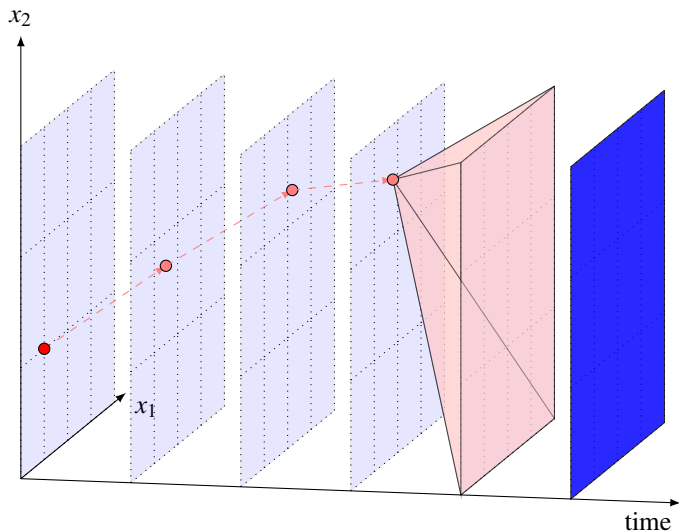
First forward pass : computing trajectory

Trajectory Following Dynamic Programming



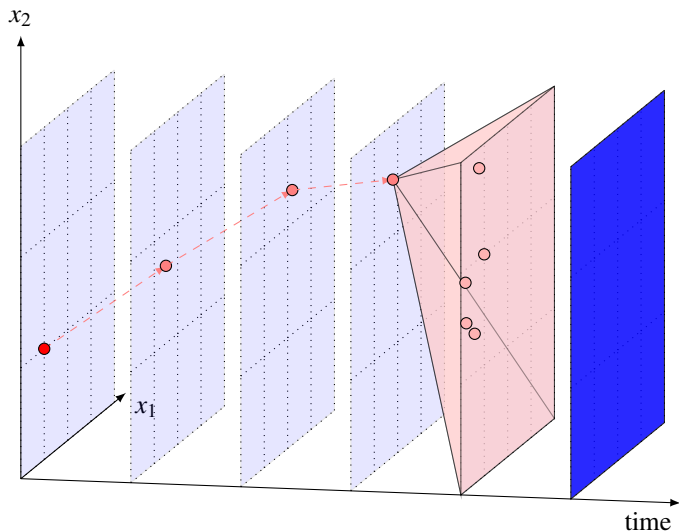
First forward pass : computing trajectory

Trajectory Following Dynamic Programming



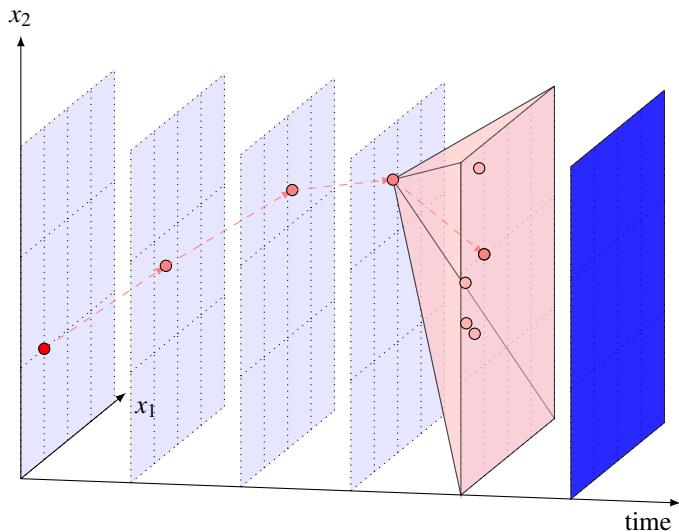
First forward pass : computing trajectory

Trajectory Following Dynamic Programming



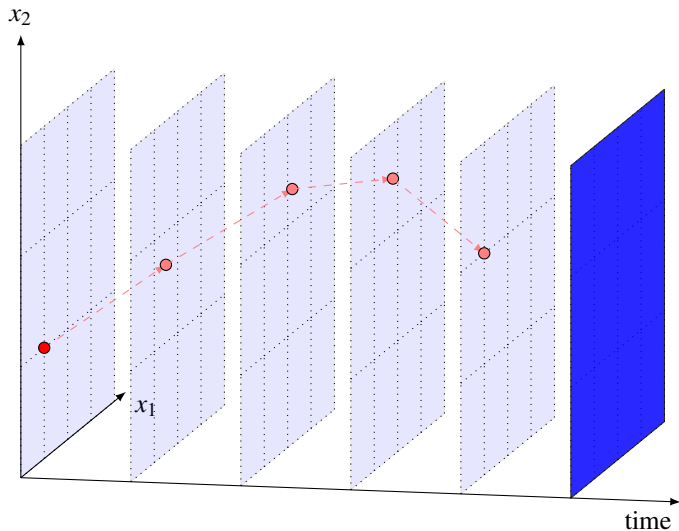
First forward pass : computing trajectory

Trajectory Following Dynamic Programming



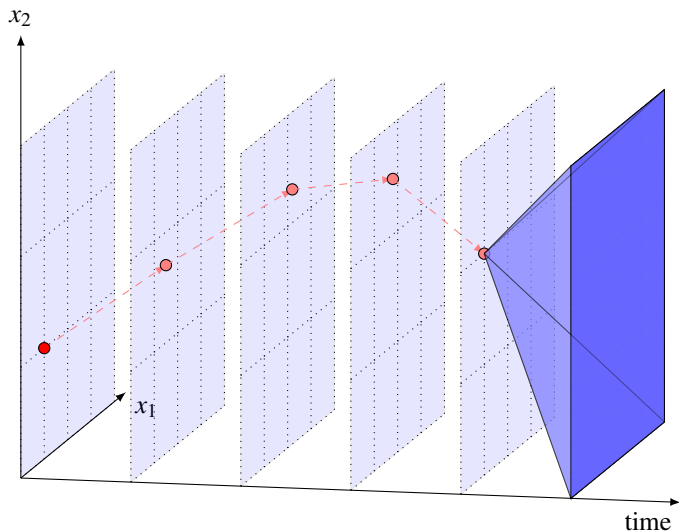
First forward pass : computing trajectory

Trajectory Following Dynamic Programming



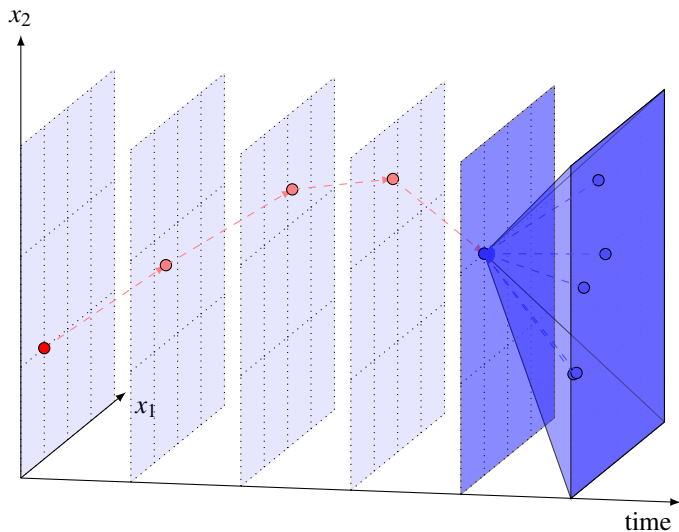
First forward pass : computing trajectory

Trajectory Following Dynamic Programming



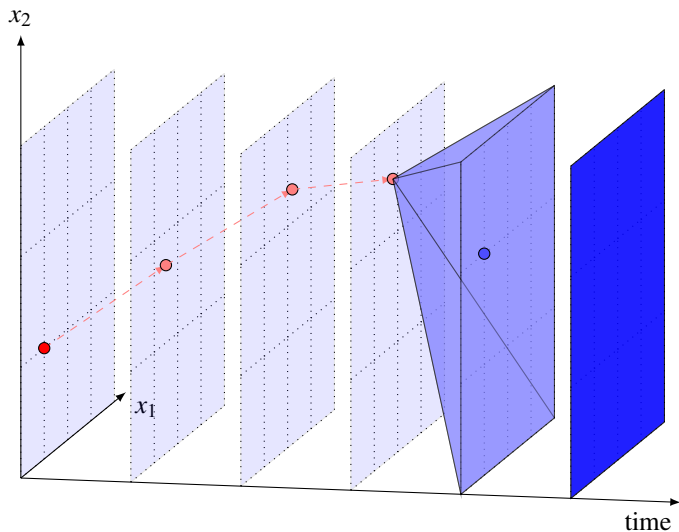
First backward pass : refining approximation (adding cuts)

Trajectory Following Dynamic Programming



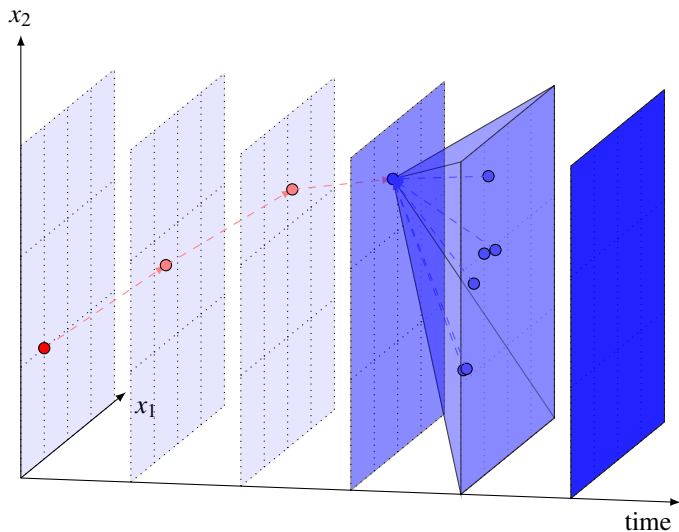
First backward pass : refining approximation (adding cuts)

Trajectory Following Dynamic Programming



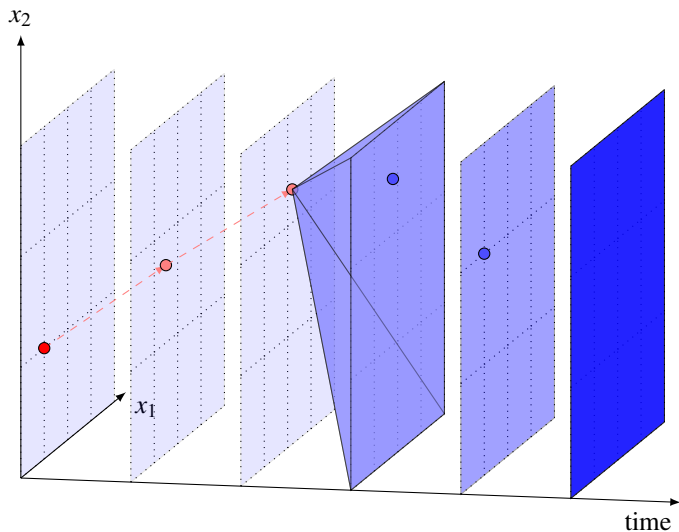
First backward pass : refining approximation (adding cuts)

Trajectory Following Dynamic Programming



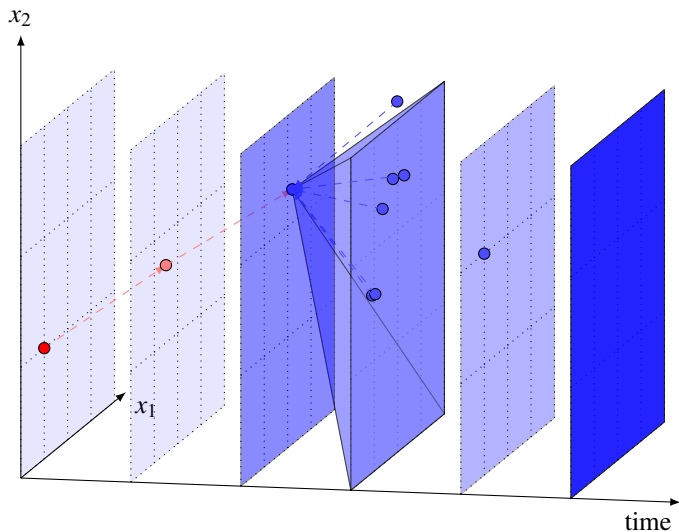
First backward pass : refining approximation (adding cuts)

Trajectory Following Dynamic Programming



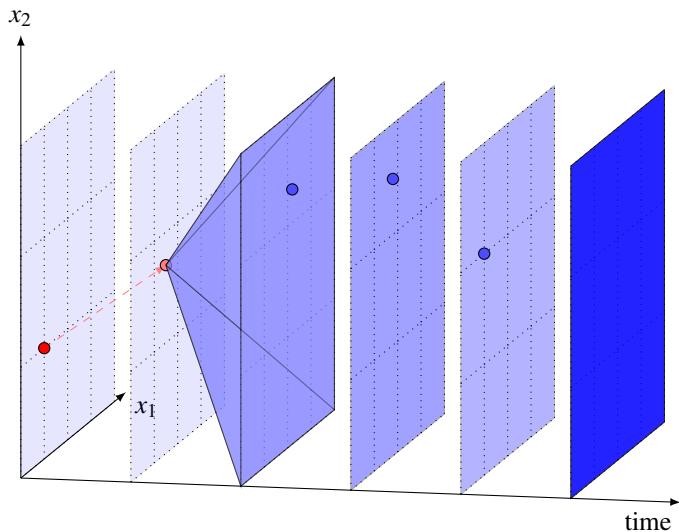
First backward pass : refining approximation (adding cuts)

Trajectory Following Dynamic Programming



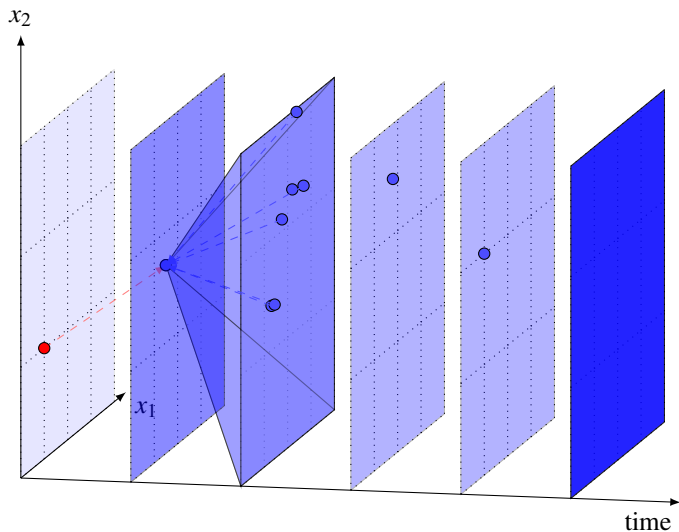
First backward pass : refining approximation (adding cuts)

Trajectory Following Dynamic Programming



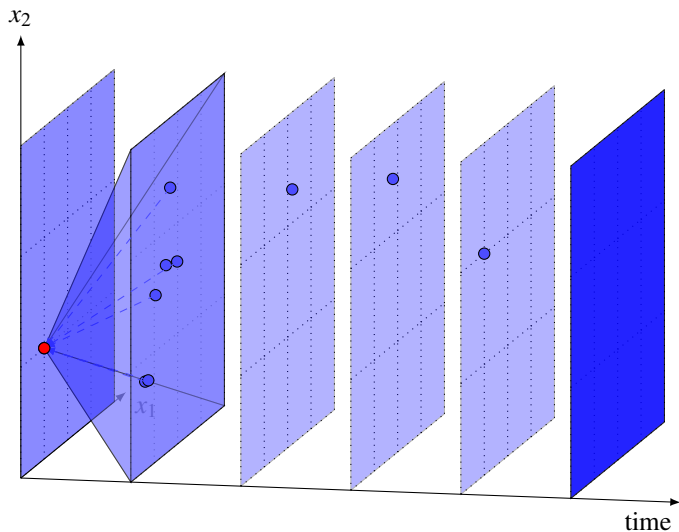
First backward pass : refining approximation (adding cuts)

Trajectory Following Dynamic Programming



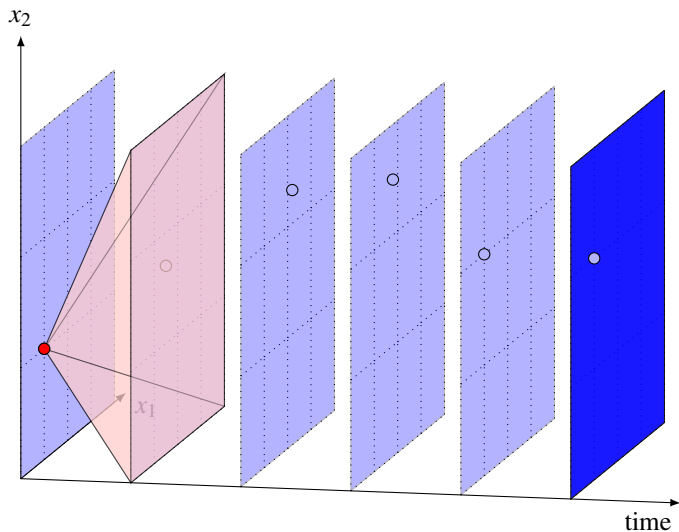
First backward pass : refining approximation (adding cuts)

Trajectory Following Dynamic Programming



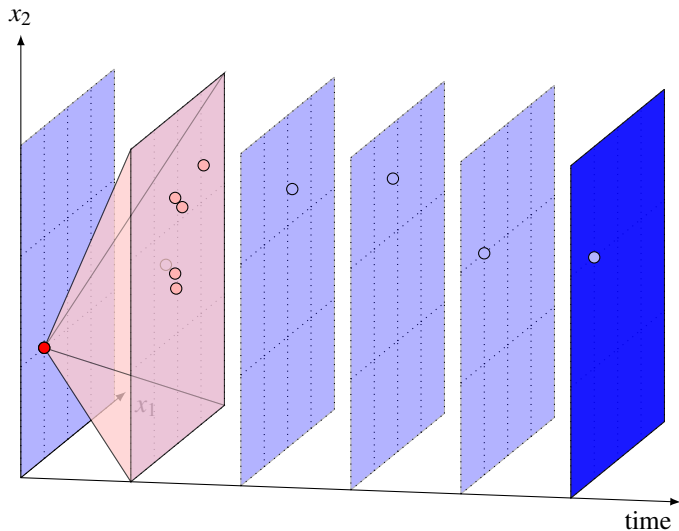
First backward pass : refining approximation (adding cuts)

Trajectory Following Dynamic Programming



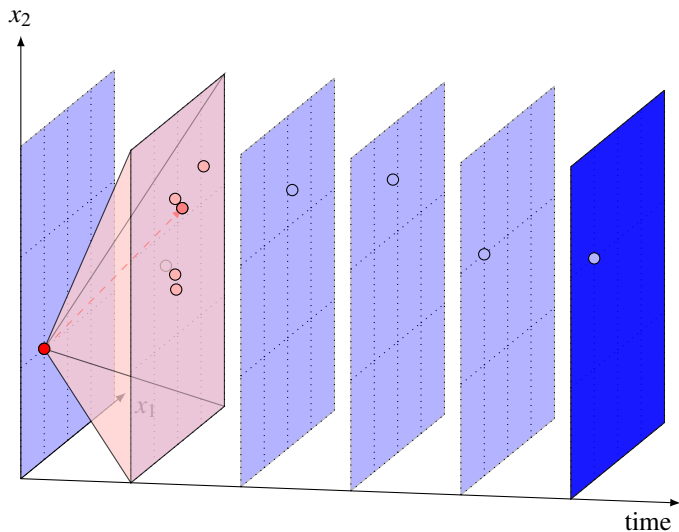
second forward pass : computing trajectory

Trajectory Following Dynamic Programming



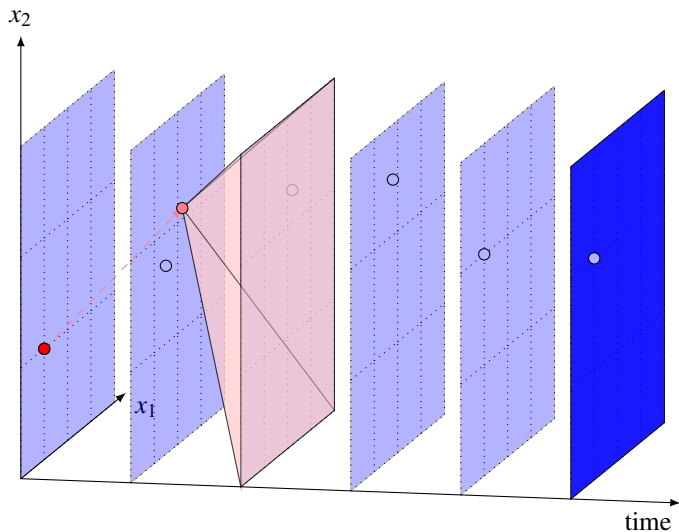
second forward pass : computing trajectory

Trajectory Following Dynamic Programming



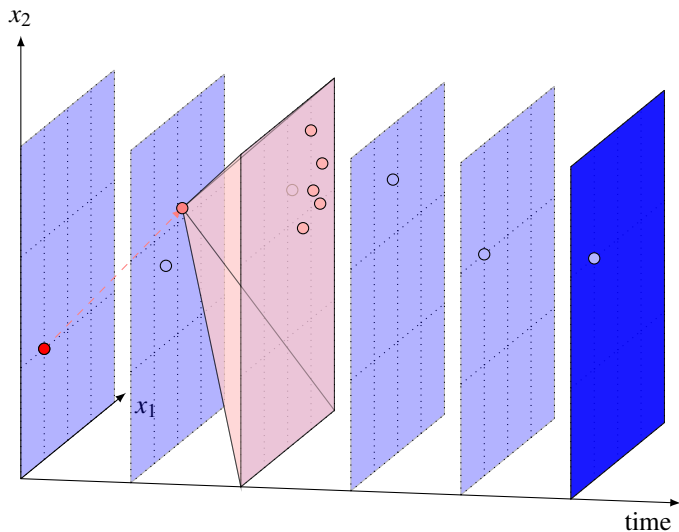
second forward pass : computing trajectory

Trajectory Following Dynamic Programming



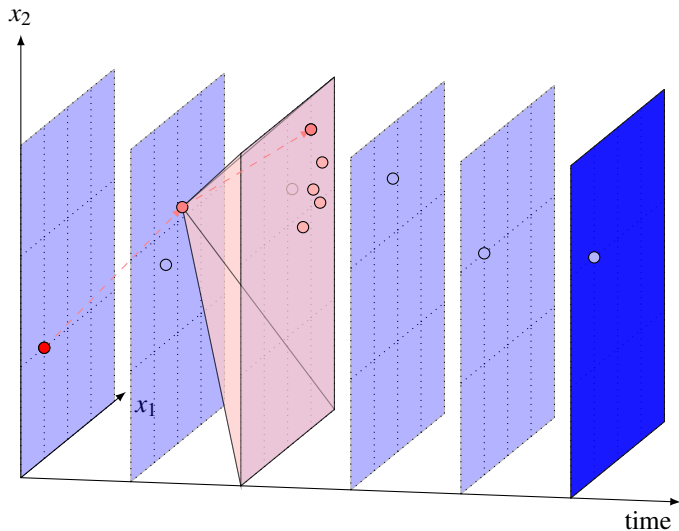
second forward pass : computing trajectory

Trajectory Following Dynamic Programming



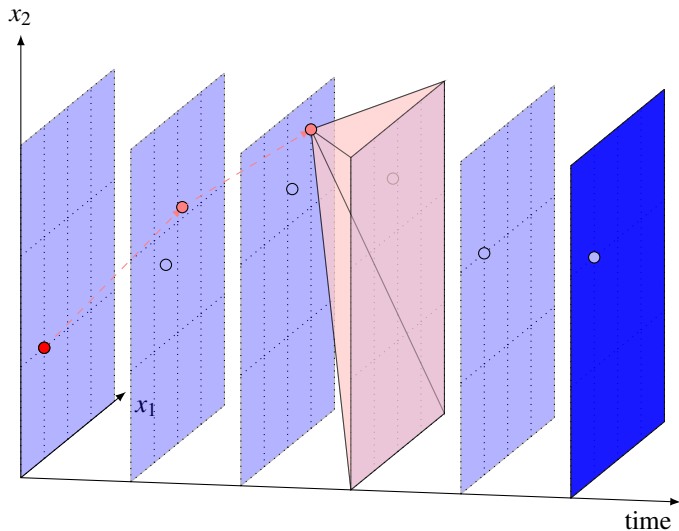
second forward pass : computing trajectory

Trajectory Following Dynamic Programming



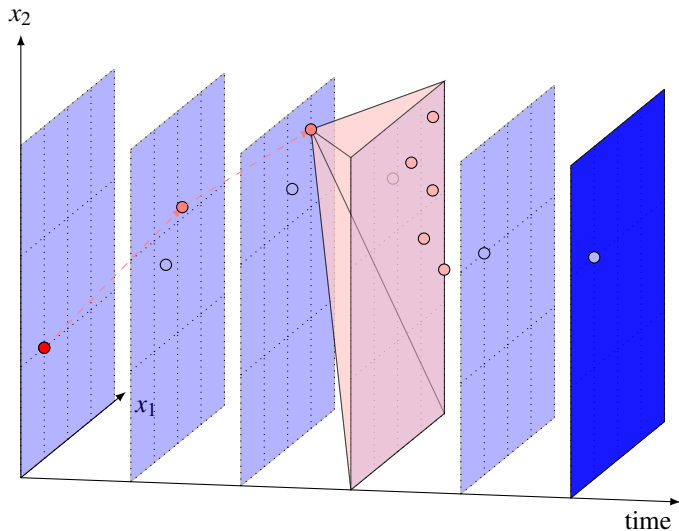
second forward pass : computing trajectory

Trajectory Following Dynamic Programming



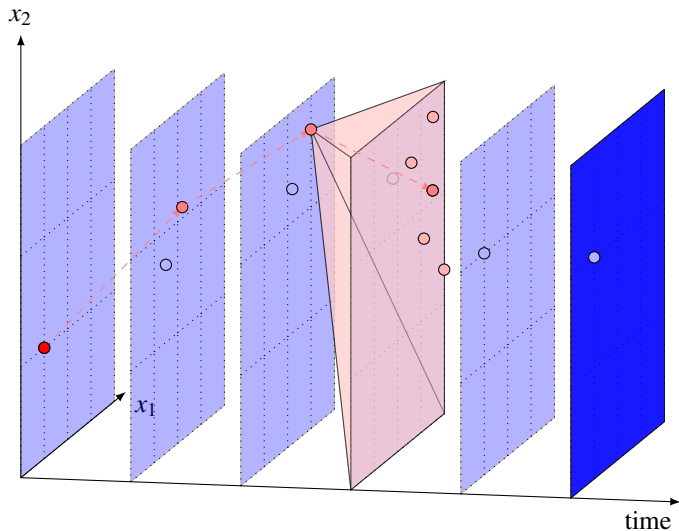
second forward pass : computing trajectory

Trajectory Following Dynamic Programming



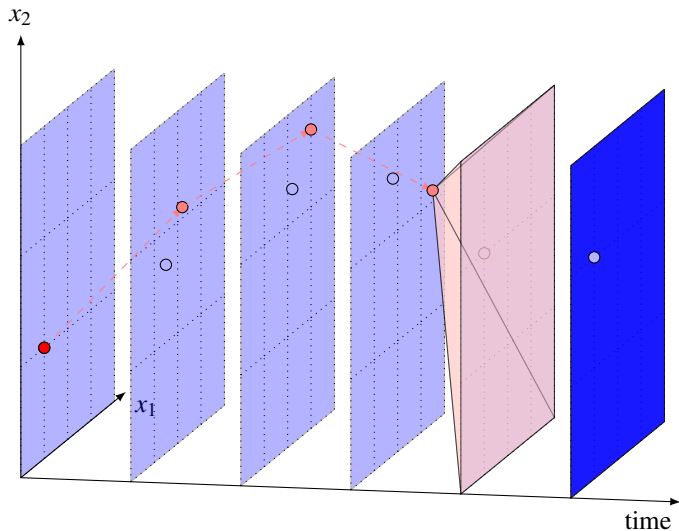
second forward pass : computing trajectory

Trajectory Following Dynamic Programming



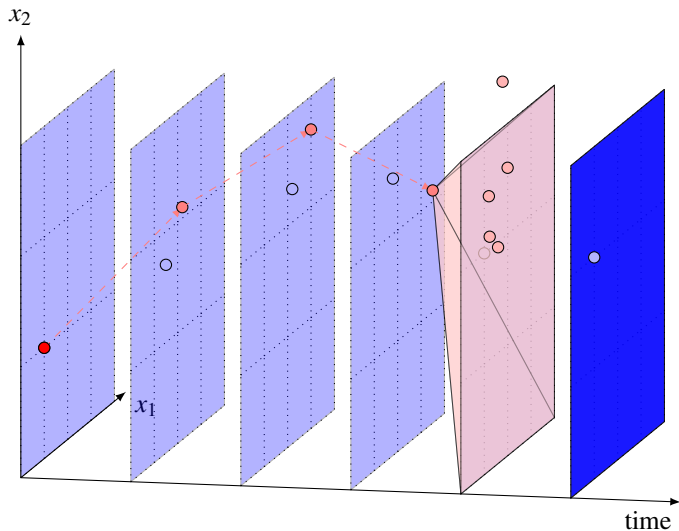
second forward pass : computing trajectory

Trajectory Following Dynamic Programming



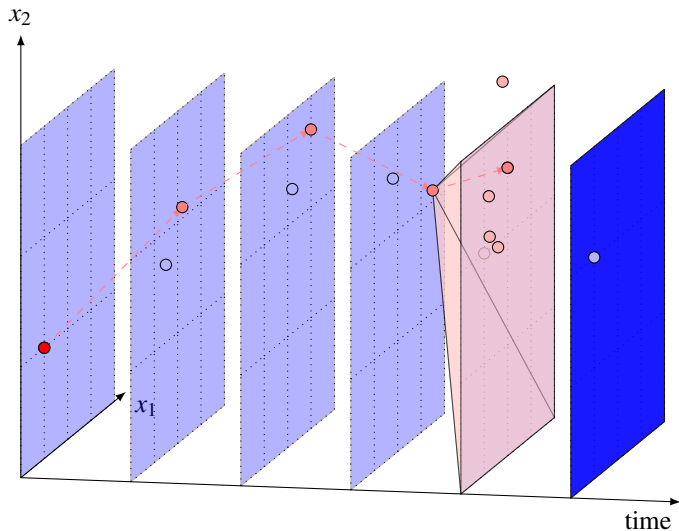
second forward pass : computing trajectory

Trajectory Following Dynamic Programming



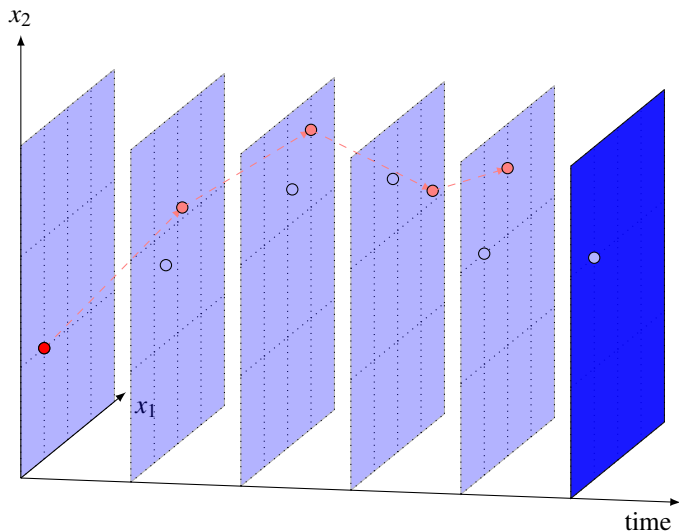
second forward pass : computing trajectory

Trajectory Following Dynamic Programming



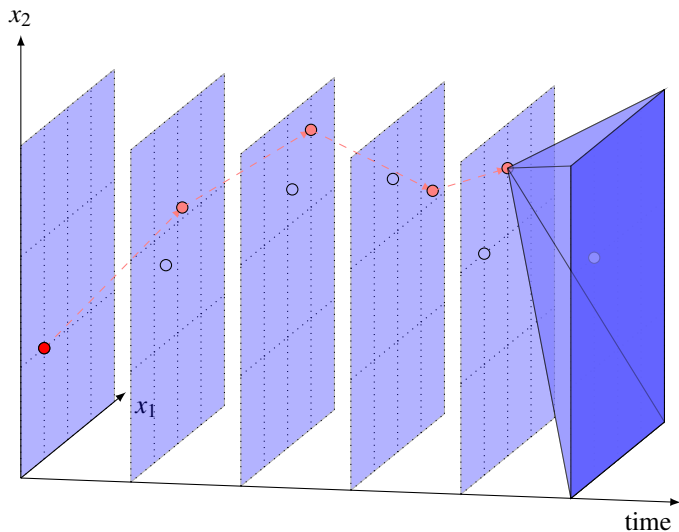
second forward pass : computing trajectory

Trajectory Following Dynamic Programming



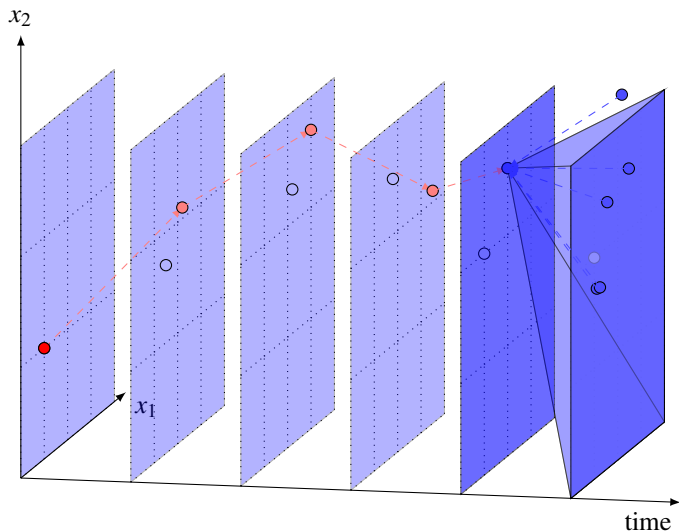
second backward pass : refining approximation (adding cuts)

Trajectory Following Dynamic Programming



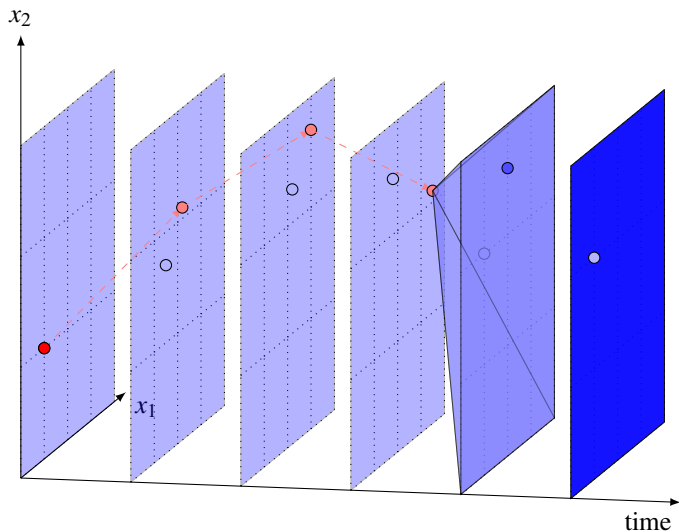
second backward pass : refining approximation (adding cuts)

Trajectory Following Dynamic Programming



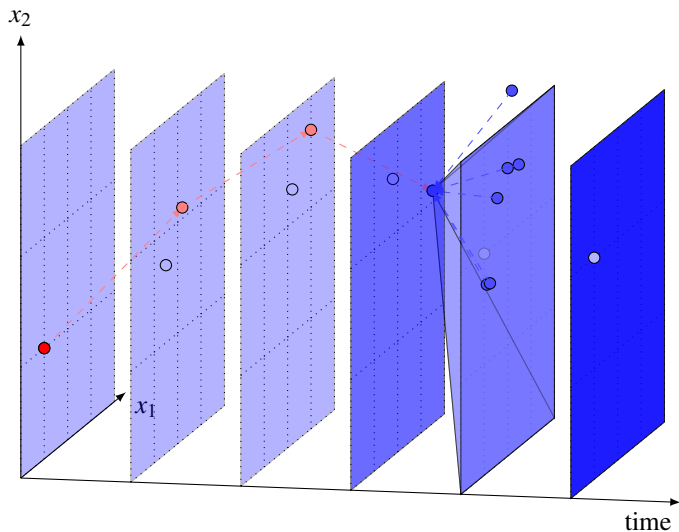
second backward pass : refining approximation (adding cuts)

Trajectory Following Dynamic Programming



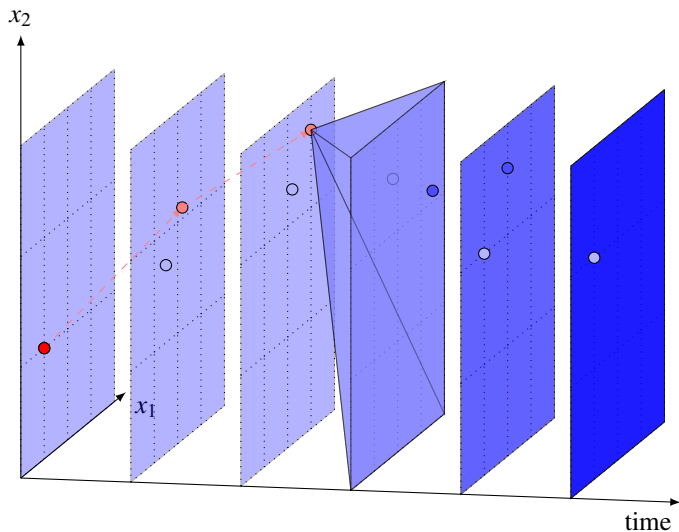
second backward pass : refining approximation (adding cuts)

Trajectory Following Dynamic Programming



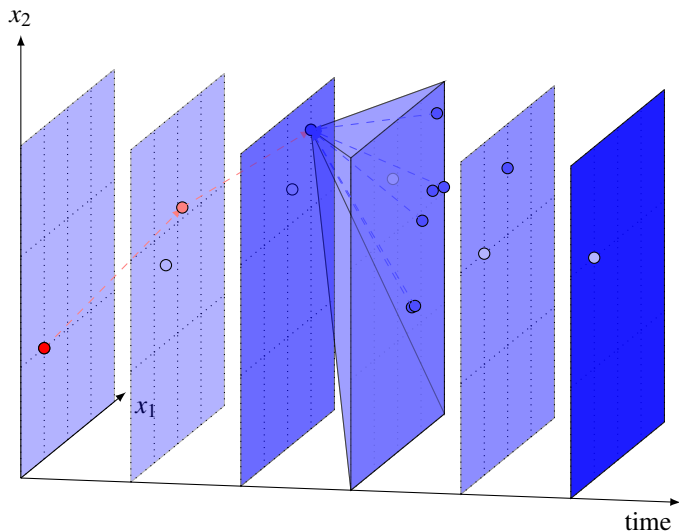
second backward pass : refining approximation (adding cuts)

Trajectory Following Dynamic Programming



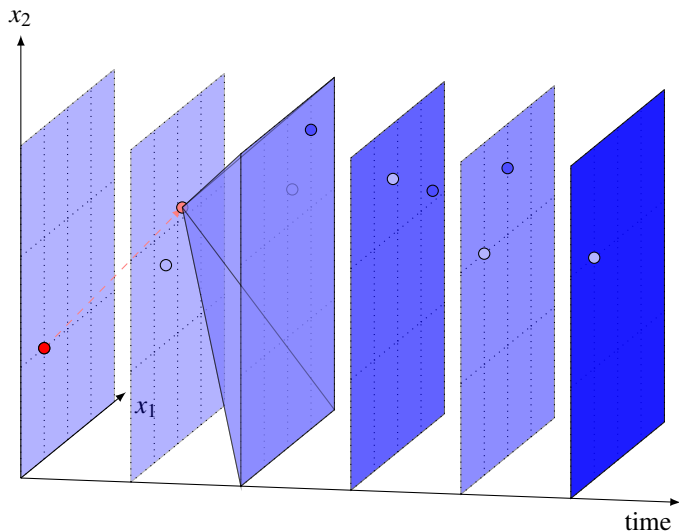
second backward pass : refining approximation (adding cuts)

Trajectory Following Dynamic Programming



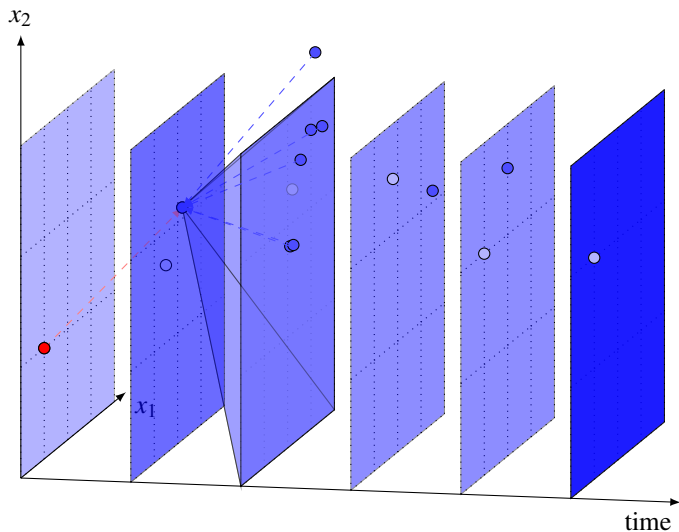
second backward pass : refining approximation (adding cuts)

Trajectory Following Dynamic Programming



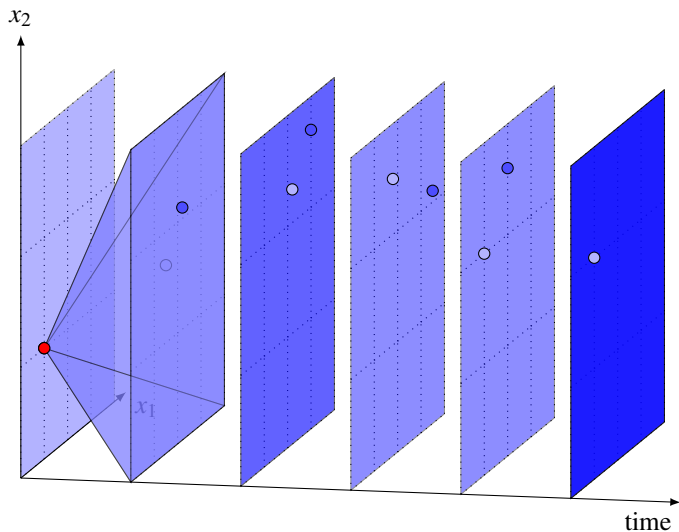
second backward pass : refining approximation (adding cuts)

Trajectory Following Dynamic Programming



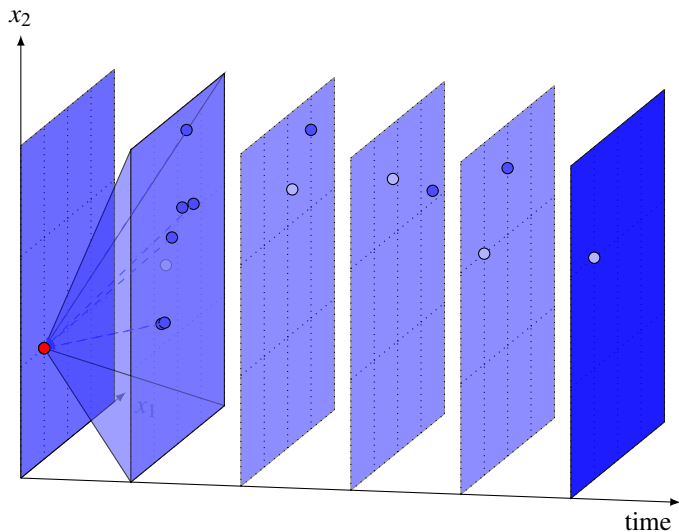
second backward pass : refining approximation (adding cuts)

Trajectory Following Dynamic Programming



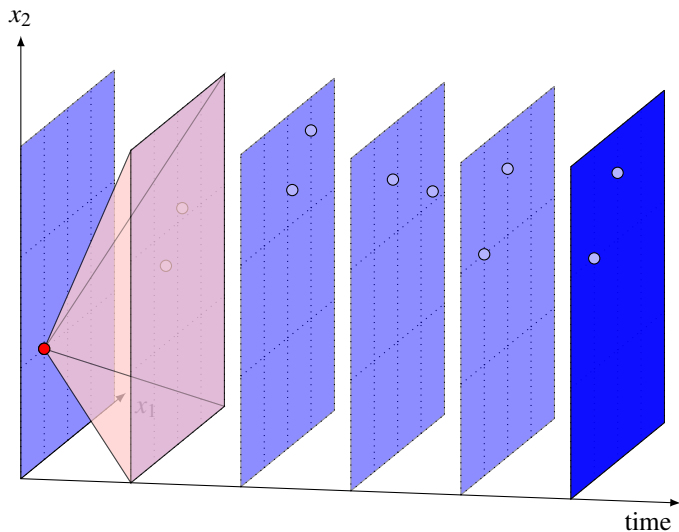
second backward pass : refining approximation (adding cuts)

Trajectory Following Dynamic Programming



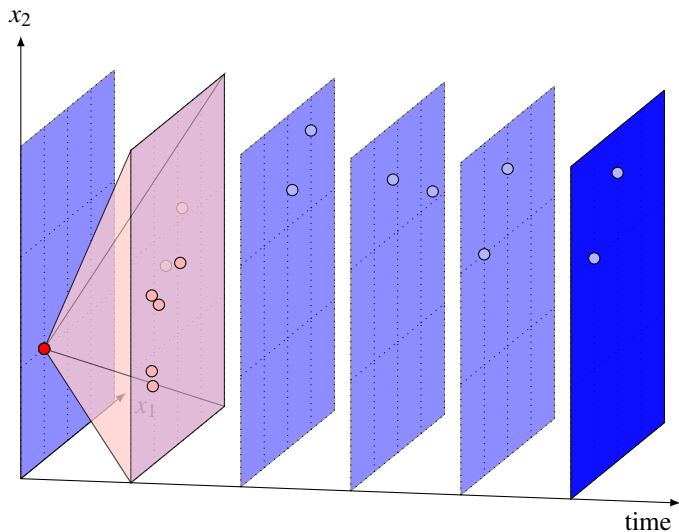
second backward pass : refining approximation (adding cuts)

Trajectory Following Dynamic Programming



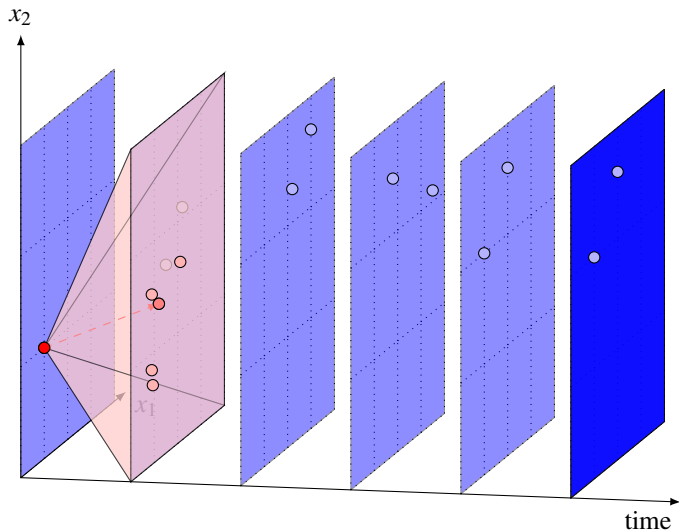
third forward pass : computing trajectory

Trajectory Following Dynamic Programming



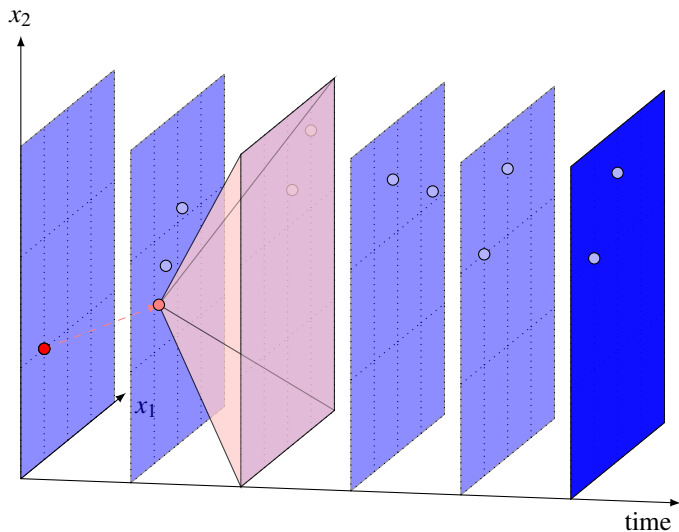
third forward pass : computing trajectory

Trajectory Following Dynamic Programming



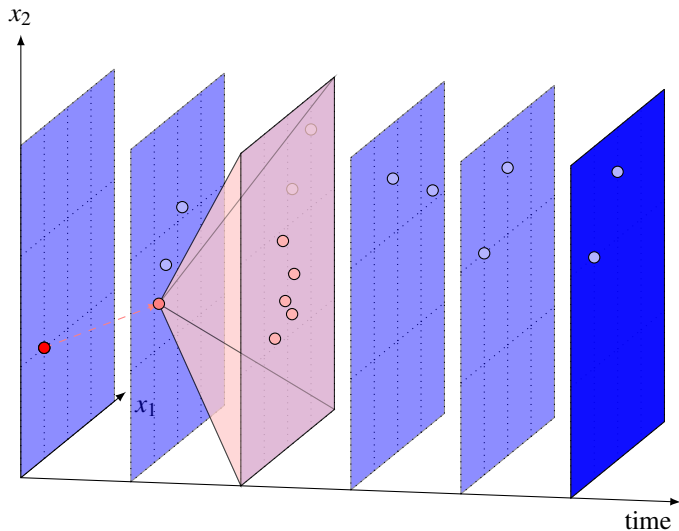
third forward pass : computing trajectory

Trajectory Following Dynamic Programming



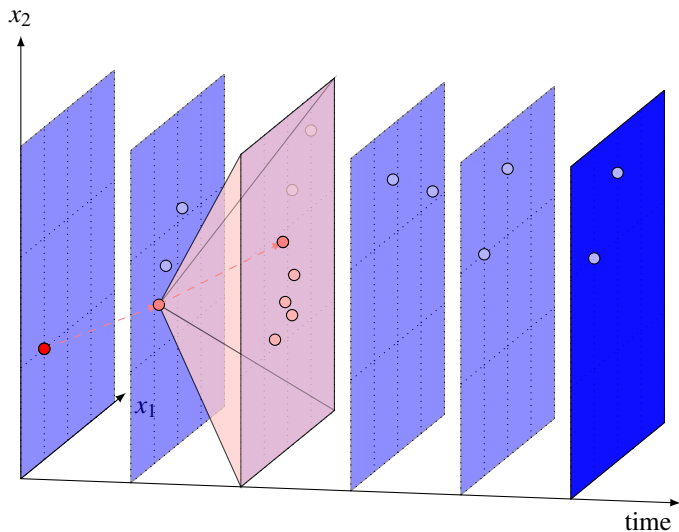
third forward pass : computing trajectory

Trajectory Following Dynamic Programming



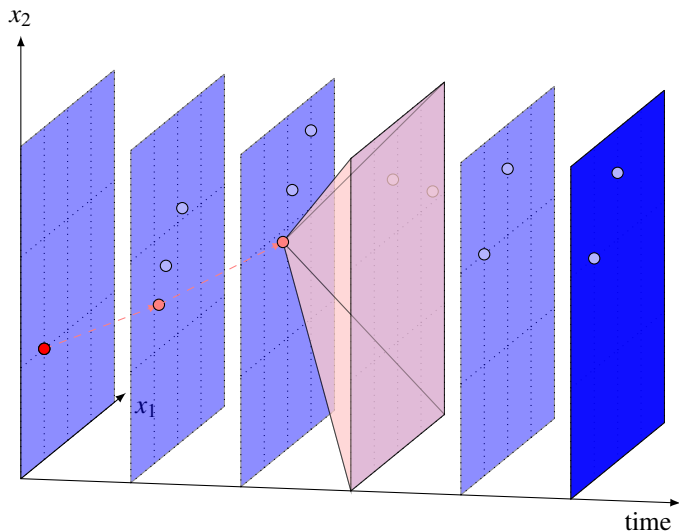
third forward pass : computing trajectory

Trajectory Following Dynamic Programming



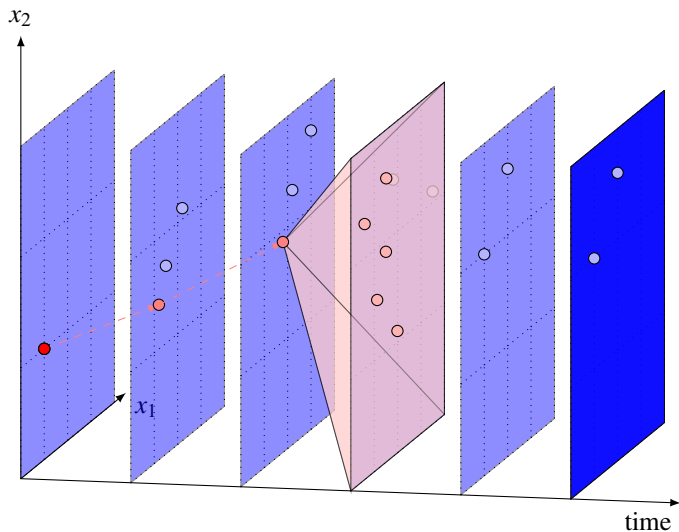
third forward pass : computing trajectory

Trajectory Following Dynamic Programming



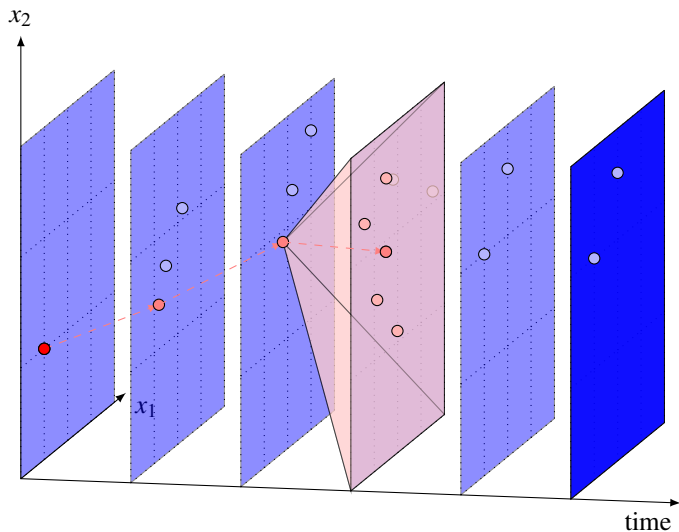
third forward pass : computing trajectory

Trajectory Following Dynamic Programming



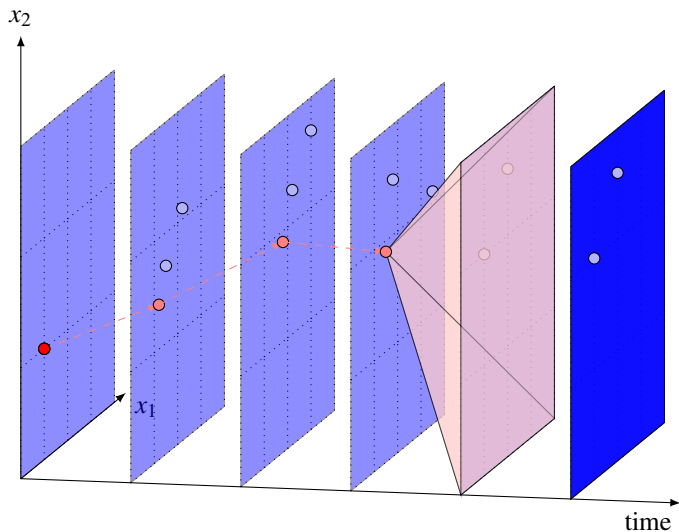
third forward pass : computing trajectory

Trajectory Following Dynamic Programming



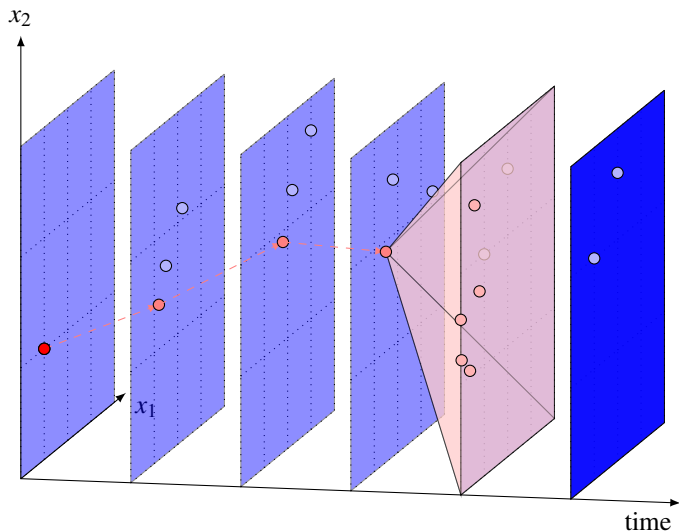
third forward pass : computing trajectory

Trajectory Following Dynamic Programming



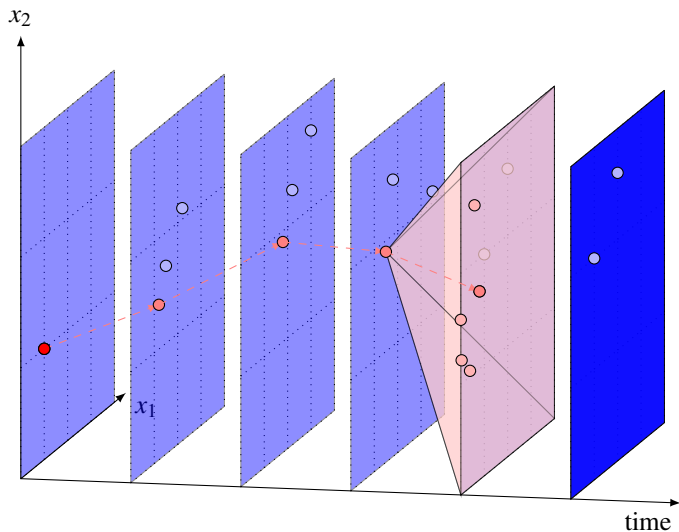
third forward pass : computing trajectory

Trajectory Following Dynamic Programming



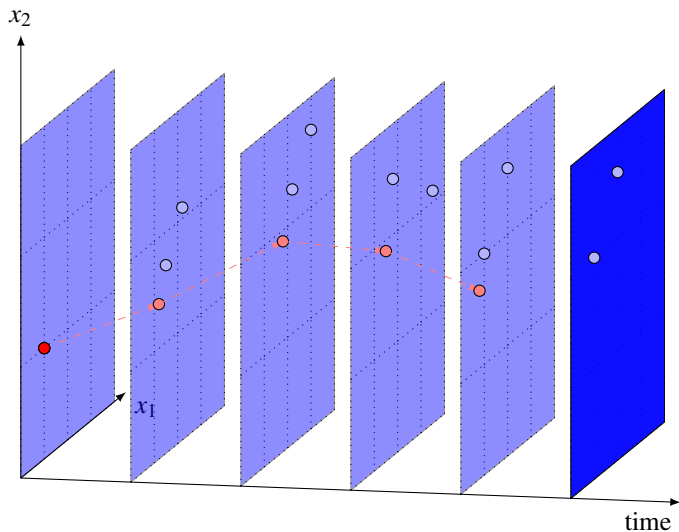
third forward pass : computing trajectory

Trajectory Following Dynamic Programming



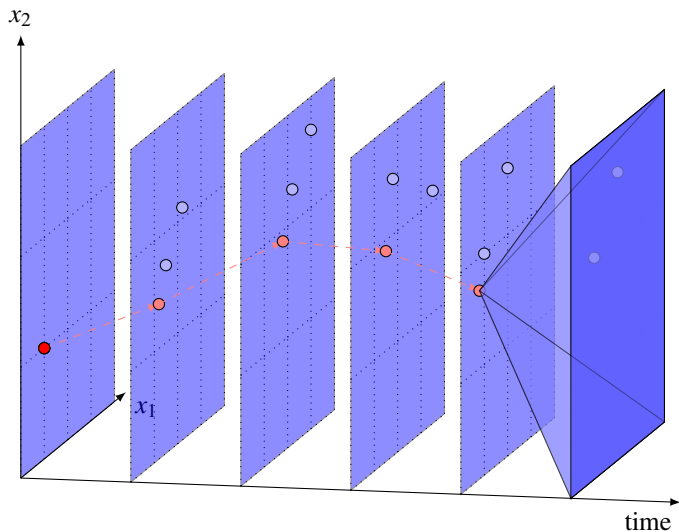
third forward pass : computing trajectory

Trajectory Following Dynamic Programming



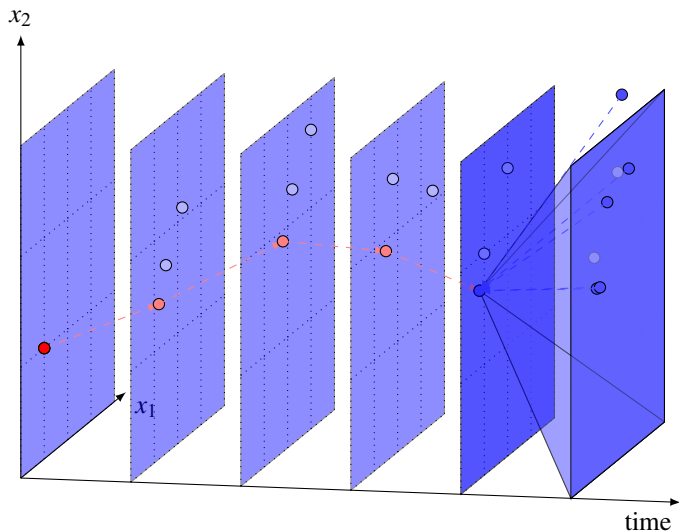
third backward pass : refining approximation (adding cuts)

Trajectory Following Dynamic Programming



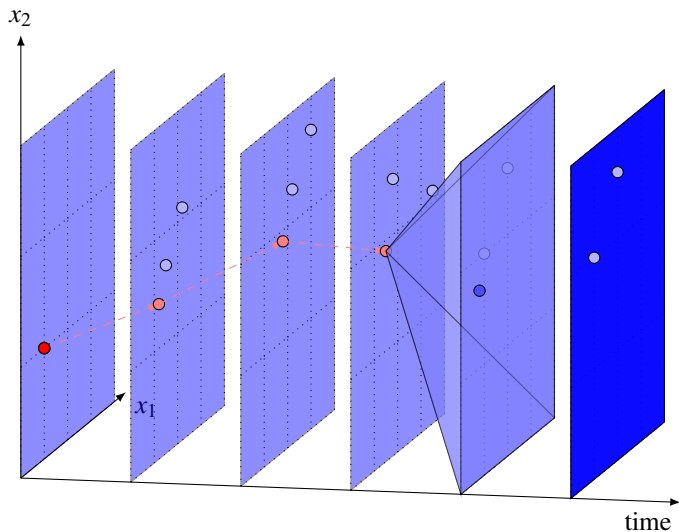
third backward pass : refining approximation (adding cuts)

Trajectory Following Dynamic Programming



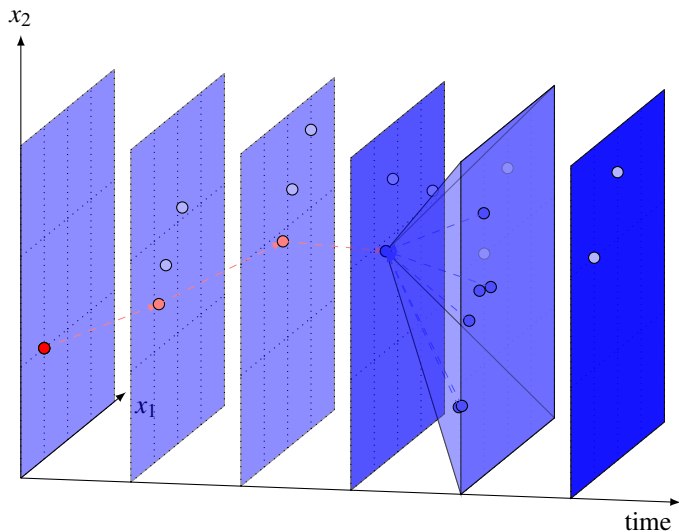
third backward pass : refining approximation (adding cuts)

Trajectory Following Dynamic Programming



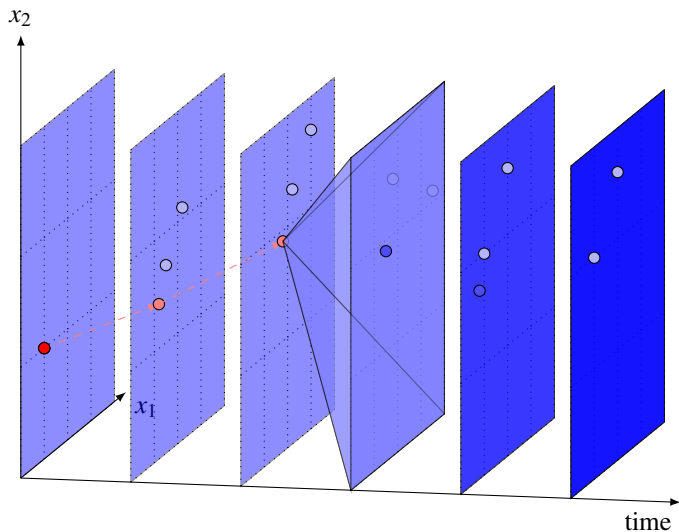
third backward pass : refining approximation (adding cuts)

Trajectory Following Dynamic Programming



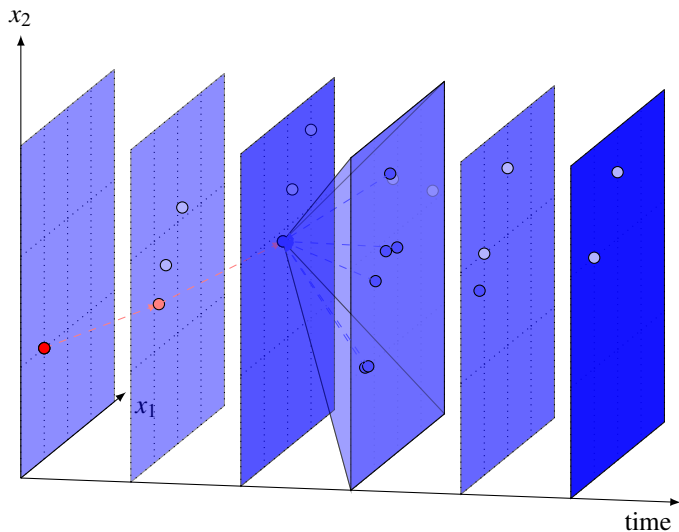
third backward pass : refining approximation (adding cuts)

Trajectory Following Dynamic Programming



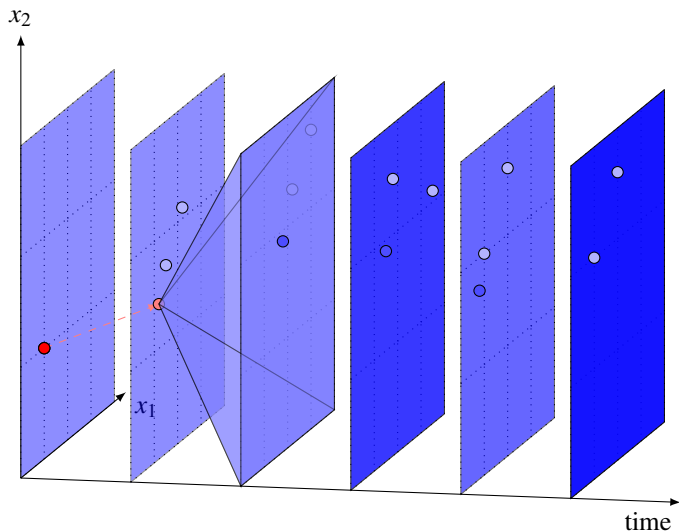
third backward pass : refining approximation (adding cuts)

Trajectory Following Dynamic Programming



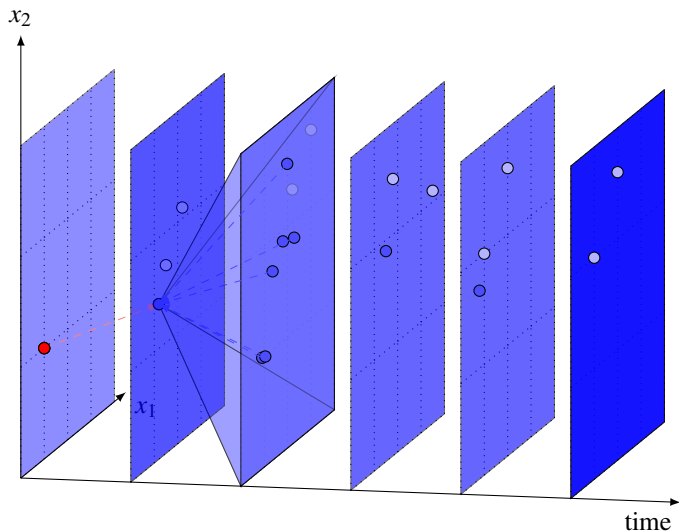
third backward pass : refining approximation (adding cuts)

Trajectory Following Dynamic Programming



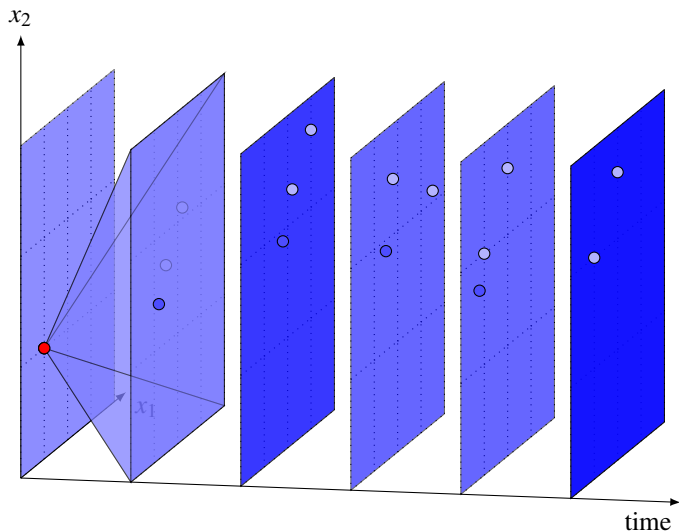
third backward pass : refining approximation (adding cuts)

Trajectory Following Dynamic Programming



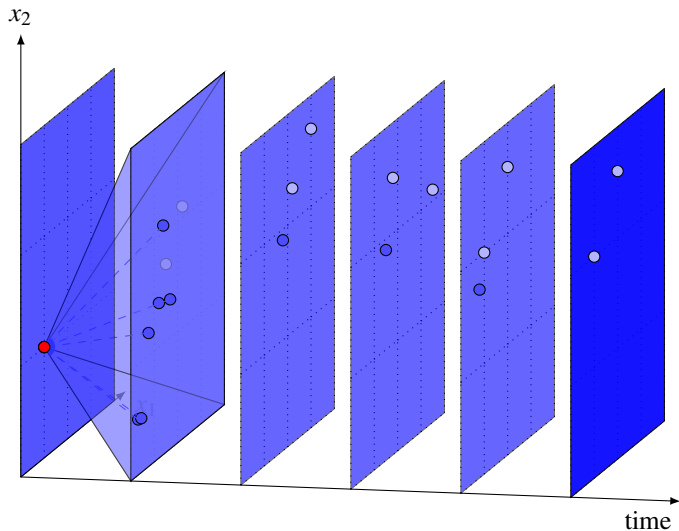
third backward pass : refining approximation (adding cuts)

Trajectory Following Dynamic Programming



third backward pass : refining approximation (adding cuts)

Trajectory Following Dynamic Programming



And so on...

Contributions on SDDP and its variants

- ➡ New framework called Trajectory Following Dynamic Programming (TFDP) encompassing at least 14 variants of SDDP
- ➡ Complexity proofs, new for most of those variants
- ➡ Do not require finite support assumption
- ➡ Allow approximation error
- ➡ Adapt to robust and risk averse cases

Some TFDP algorithms

Algorithm's name	Node selection: Choice ξ_t^k	\mathcal{F}_t	\underline{V}_t^k	\overline{V}_t^k	Hypothesis	Complexity known
SDDP	Random sampling	Exact	Benders cuts	V_t	Convex	✓
EDDP	Explorative	Exact	Benders cuts	V_t	Convex	✓
APSDDP	Random sampling	Exact	Adaptive partition	V_t	Linear	✗
SDDiP	Random sampling	Exact	Lagrangian or integer cuts	V_t	Mixed Integer Linear	✗
MIDAS	Random sampling	Exact	Step cuts	V_t	Monotonic Mixed Integer	✗
SLDP	Random sampling	Exact	Reverse norm cuts	V_t	Non-Convex	✗
BDZ17	Problem child	Exact	Benders cuts	Epigraph as convex hull	Convex	✗
BDZ18	Problem child	Exact	Benders \times Epigraph	Hypograph \times Benders	Convex-Concave	✗
RDDP	Deterministic	Exact	Benders cuts	Epigraph as convex hull	Robust	✗
ISDDP	Random sampling	Inexact	Inexact Lagrangian cuts	V_t	Convex	✗
TDP	Problem child	Exact	Benders cuts	Min of quadratic	Convex	✗
ZS19	Random or Problem	Regularized	Generalized conjugacy cuts	Norm cuts	Mixed Integer Convex	✓
NDDP	Random or Problem	Regularized	Benders cuts	Norm cuts	Distributionally Robust	✓
DSDDP	Random sampling	Exact	Benders cuts	Fenchel transform	Linear	✗

Contents

- 1 Universal Exact Quantization for cost
 - Local in 2-stage
 - Uniform in 2-stage
 - Uniform in multistage
 - Complexity results
- 2 Local and universal exact Quantization for constraints
 - Adapted partitions
 - Adaptive Partition-based Methods
 - Convergence, complexity and numerical results
- 3 Trajectory Following Dynamic Programming
- 4 Conclusion and perspectives

Conclusion

	A	(B, b)	c
Local	×	✓	✓
Uniform	×	×	✓

- Links with fundamental polyhedral geometry, regular subdivisions and fiber polytope (Chap. 3 and 4).
- *Uniform and universal* exact quantization for c in MSLP (Chap.4).
 - ➡ Polynomial time complexity results.
- *Local* exact quantization for B and b .
 - ➡ Adaptive Partition-based Methods (APM) for general distribution: solves small 2SLP with high precision (Chap. 5).
- Extension of Stochastic Dual Dynamic Programming algorithms and more generally all Trajectory Following Dynamic Programming algorithm for non finitely supported distribution (Chap. 6).

Conclusion

	A	(B, b)	c
Local	×	✓	✓
Uniform	×	×	✓

- Links with fundamental polyhedral geometry, regular subdivisions and fiber polytope (Chap. 3 and 4).
- *Uniform and universal* exact quantization for c in MSLP (Chap.4).
 - ➡ Polynomial time complexity results.
- *Local* exact quantization for B and b .
 - ➡ Adaptive Partition-based Methods (APM) for general distribution: solves small 2SLP with high precision (Chap. 5).
- Extension of Stochastic Dual Dynamic Programming algorithms and more generally all Trajectory Following Dynamic Programming algorithm for non finitely supported distribution (Chap. 6).

Conclusion

	A	(B, b)	c
Local	×	✓	✓
Uniform	×	×	✓

- Links with fundamental polyhedral geometry, regular subdivisions and fiber polytope (Chap. 3 and 4).
- *Uniform and universal* exact quantization for c in MSLP (Chap.4).
 - ➡ Polynomial time complexity results.
- *Local* exact quantization for B and b .
 - ➡ Adaptive Partition-based Methods (APM) for general distribution: solves small 2SLP with high precision (Chap. 5).
- Extension of Stochastic Dual Dynamic Programming algorithms and more generally all Trajectory Following Dynamic Programming algorithm for non finitely supported distribution (Chap. 6).

Conclusion

	A	(B, b)	c
Local	×	✓	✓
Uniform	×	×	✓

- Links with fundamental polyhedral geometry, regular subdivisions and fiber polytope (Chap. 3 and 4).
- *Uniform and universal* exact quantization for c in MSLP (Chap.4).
 - ➡ Polynomial time complexity results.
- *Local* exact quantization for B and b .
 - ➡ Adaptive Partition-based Methods (APM) for general distribution: solves small 2SLP with high precision (Chap. 5).
- Extension of Stochastic Dual Dynamic Programming algorithms and more generally all Trajectory Following Dynamic Programming algorithm for non finitely supported distribution (Chap. 6).

Perspectives (Chap. 7)

- Higher order simplex algorithm on the chamber complex for 2SLP,
- 2-time scale MSLP, nested fiber polyhedra and convex bodies,
- Reintroduce approximation or sampling,
- Exact quantization for stochastic integer linear, problems
- Understanding the complexity of MSLP.

Perspectives (Chap. 7)

- Higher order simplex algorithm on the chamber complex for 2SLP,
- 2-time scale MSLP, nested fiber polyhedra and convex bodies,
- Reintroduce approximation or sampling,
- Exact quantization for stochastic integer linear, problems
- Understanding the complexity of MSLP.

Perspectives (Chap. 7)

- Higher order simplex algorithm on the chamber complex for 2SLP,
- 2-time scale MSLP, nested fiber polyhedra and convex bodies,
- Reintroduce approximation or sampling,
- Exact quantization for stochastic integer linear, problems
- Understanding the complexity of MSLP.

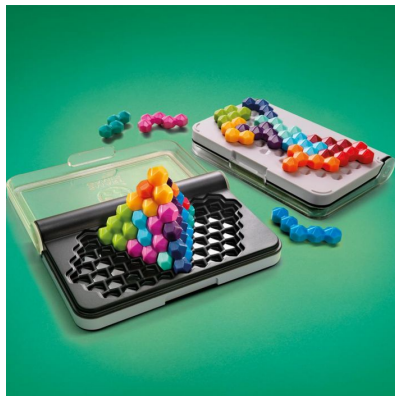
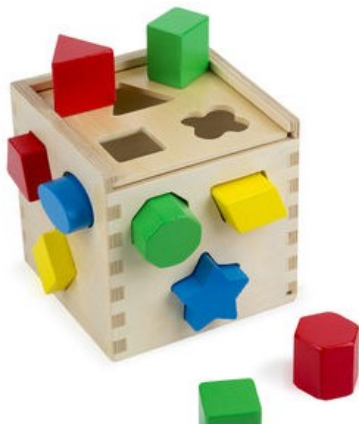
Perspectives (Chap. 7)

- Higher order simplex algorithm on the chamber complex for 2SLP,
- 2-time scale MSLP, nested fiber polyhedra and convex bodies,
- Reintroduce approximation or sampling,
- Exact quantization for stochastic integer linear, problems
- Understanding the complexity of MSLP.

Perspectives (Chap. 7)

- Higher order simplex algorithm on the chamber complex for 2SLP,
- 2-time scale MSLP, nested fiber polyhedra and convex bodies,
- Reintroduce approximation or sampling,
- Exact quantization for stochastic integer linear, problems
- Understanding the complexity of MSLP.

Thank you for listening ! Any question ?



Contents

5 Explicit formulas for general distributions

6 Details on GAPM

- Recalls on APM
- A novel APM algorithm
- Extension of GAPM to general costs

7 Nested fiber polyhedra

8 Polyhedral toolbox for stochastic optimizers

- Active constraints
- Link with regular subdivisions
- Correspondences for parametric linear programming
- Correspondences for 2SLP

Explicit formulas for usual distributions

in the exact case, we need to compute the quantized probabilities $\check{p}_S = \mathbb{P}[\mathbf{c} \in S]$ and the quantized cost $\check{c}_S = \mathbb{E}[\xi \mid \mathbf{c} \in S]$ when S is a polyhedron.

Explicit formulas, valid for S full dimensional **simplex** or **simplicial cone**, which can be used through triangulation.

Distribution	Uniform on polytope	Exponential	Gaussian
$d\mathbb{P}(\xi)$	$\frac{\mathbb{1}_{\xi \in Q}}{\text{Vol}_d(Q)} \mathcal{L}_{\text{Aff}(Q)}(d\xi)$	$\frac{e^{\theta^\top \xi} \mathbb{1}_{\xi \in K}}{\Phi_K(\theta)} \mathcal{L}_{\text{Aff}(K)}(d\xi)$	$\frac{e^{-\frac{1}{2} \xi^\top M^{-2} \xi}}{(2\pi)^{\frac{m}{2}} \det M} d\xi$
Support	Polytope : Q	Cone : K	\mathbb{R}^m
\check{p}_S	$\frac{\text{Vol}_d(S)}{\text{Vol}_d(Q)}$	$\frac{ \det(\text{Ray}(S)) }{\Phi_K(\theta)} \prod_{r \in \text{Ray}(S)} \frac{1}{-r^\top \theta}$	$\text{Ang}(M^{-1}S)$
\check{c}_S	$\frac{1}{d} \sum_{v \in \text{Vert}(S)} v$	$\left(\sum_{r \in \text{Ray}(S)} \frac{-r_i}{r^\top \theta} \right)_{i \in [m]}$	$\frac{\sqrt{2} \Gamma(\frac{m+1}{2})}{\Gamma(\frac{m}{2})} M \text{Ctr}(S \cap \mathbb{S}_{m-1})$

Explicit formulas for usual distributions

in the exact case, we need to compute the quantized probabilities $\check{p}_S = \mathbb{P}[\mathbf{c} \in S]$ and the quantized cost $\check{c}_S = \mathbb{E}[\xi \mid \mathbf{c} \in S]$ when S is a polyhedron.

Explicit formulas, valid for S full dimensional **simplex** or **simplicial cone**, which can be used through triangulation.

Distribution	Uniform on polytope	Exponential	Gaussian
$d\mathbb{P}(\xi)$	$\frac{\mathbb{1}_{\xi \in Q}}{\text{Vol}_d(Q)} \mathcal{L}_{\text{Aff}(Q)}(d\xi)$	$\frac{e^{\theta^\top \xi} \mathbb{1}_{\xi \in K}}{\Phi_K(\theta)} \mathcal{L}_{\text{Aff}(K)}(d\xi)$	$\frac{e^{-\frac{1}{2} \xi^\top M^{-2} \xi}}{(2\pi)^{\frac{m}{2}} \det M} d\xi$
Support	Polytope : Q	Cone : K	\mathbb{R}^m
\check{p}_S	$\frac{\text{Vol}_d(S)}{\text{Vol}_d(Q)}$	$\frac{ \det(\text{Ray}(S)) }{\Phi_K(\theta)} \prod_{r \in \text{Ray}(S)} \frac{1}{-r^\top \theta}$	$\text{Ang}(M^{-1}S)$
\check{c}_S	$\frac{1}{d} \sum_{v \in \text{Vert}(S)} v$	$\left(\sum_{r \in \text{Ray}(S)} \frac{-r_i}{r^\top \theta} \right)_{i \in [m]}$	$\frac{\sqrt{2} \Gamma(\frac{m+1}{2})}{\Gamma(\frac{m}{2})} M \text{Ctr}(S \cap \mathbb{S}_{m-1})$

Contents

5 Explicit formulas for general distributions

6 Details on GAPM

- Recalls on APM
- A novel APM algorithm
- Extension of GAPM to general costs

7 Nested fiber polyhedra

8 Polyhedral toolbox for stochastic optimizers

- Active constraints
- Link with regular subdivisions
- Correspondences for parametric linear programming
- Correspondences for 2SLP

Contents

5 Explicit formulas for general distributions

6 Details on GAPM

- Recalls on APM
- A novel APM algorithm
- Extension of GAPM to general costs

7 Nested fiber polyhedra

8 Polyhedral toolbox for stochastic optimizers

- Active constraints
- Link with regular subdivisions
- Correspondences for parametric linear programming
- Correspondences for 2SLP

2 stage stochastic linear programming (2SLP)

$$\begin{aligned} \min_{x \in \mathbb{R}_+^n} \quad & c^\top x + \mathbb{E}[Q(x, \xi)] \\ \text{s.t.} \quad & Ax = b \end{aligned}$$

where $\xi = (T, h)$ is random whereas q and W are deterministic¹

$$\begin{aligned} Q(x, \xi) &:= \min_{y \in \mathbb{R}_+^m} q^\top y &= \max_{\lambda \in \mathbb{R}^n} (h - Tx)^\top \lambda \\ \text{s.t.} \quad & Tx + Wy = h &\text{s.t.} \quad W^\top \lambda \leq q \end{aligned}$$

We define

$$X := \{x \in \mathbb{R}_+^n \mid Ax = b\} \qquad D := \{\lambda \in \mathbb{R}^\ell \mid W^\top \lambda \leq q\}$$

¹Can be extended to generic random q , and finitely supported W

2 stage stochastic linear programming (2SLP)

$$\begin{aligned} \min_{x \in \mathbb{R}_+^n} \quad & c^\top x + \mathbb{E}[Q(x, \xi)] \\ \text{s.t.} \quad & Ax = b \end{aligned}$$

where $\xi = (\mathbf{T}, \mathbf{h})$ is random whereas q and W are deterministic¹

$$\begin{aligned} Q(x, \xi) &:= \min_{y \in \mathbb{R}_+^m} q^\top y &= \max_{\lambda \in \mathbb{R}^n} (\mathbf{h} - \mathbf{T}x)^\top \lambda \\ \text{s.t.} \quad & \mathbf{T}x + Wy = \mathbf{h} &\text{s.t.} \quad W^\top \lambda \leq q \end{aligned}$$

We define

$$X := \{x \in \mathbb{R}_+^n \mid Ax = b\} \qquad D := \{\lambda \in \mathbb{R}^\ell \mid W^\top \lambda \leq q\}$$

¹Can be extended to generic random q , and finitely supported W

2 stage stochastic linear programming (2SLP)

$$\begin{aligned} \min_{x \in \mathbb{R}_+^n} \quad & c^\top x + \mathbb{E}[Q(x, \xi)] \\ \text{s.t.} \quad & Ax = b \end{aligned}$$

where $\xi = (\mathbf{T}, \mathbf{h})$ is random whereas q and W are deterministic¹

$$\begin{aligned} Q(x, \xi) &:= \min_{y \in \mathbb{R}_+^m} q^\top y &= \max_{\lambda \in \mathbb{R}^n} (\mathbf{h} - \mathbf{T}x)^\top \lambda \\ \text{s.t.} \quad & \mathbf{T}x + Wy = \mathbf{h} &\text{s.t.} \quad W^\top \lambda \leq q \end{aligned}$$

We define

$$\mathbf{X} := \{x \in \mathbb{R}_+^n \mid Ax = b\} \qquad \mathbf{D} := \{\lambda \in \mathbb{R}^\ell \mid W^\top \lambda \leq q\}$$

¹Can be extended to generic random q , and finitely supported W

2 stage stochastic linear programming (2SLP)

$$\begin{aligned} \min_{x \in \mathbb{R}_+^n} \quad & c^\top x + \mathbb{E}[Q(x, \xi)] \\ \text{s.t.} \quad & Ax = b \end{aligned}$$

where $\xi = (\mathbf{T}, \mathbf{h})$ is random whereas q and W are deterministic¹

$$\begin{aligned} Q(x, \xi) &:= \min_{y \in \mathbb{R}_+^m} q^\top y &= \max_{\lambda \in \mathbb{R}^n} (\mathbf{h} - \mathbf{T}x)^\top \lambda \\ \text{s.t.} \quad & \mathbf{T}x + Wy = \mathbf{h} &\text{s.t.} \quad W^\top \lambda \leq q \end{aligned}$$

We define

$$\mathbf{X} := \{x \in \mathbb{R}_+^n \mid Ax = b\} \qquad \mathbf{D} := \{\lambda \in \mathbb{R}^\ell \mid W^\top \lambda \leq q\}$$

¹Can be extended to generic random q , and finitely supported W

2 stage stochastic linear programming (2SLP)

$$\min_{x \in X} \quad c^\top x + \mathbb{E}[Q(x, \xi)]$$

where $\xi = (\mathbf{T}, \mathbf{h})$ is random whereas q and W are deterministic¹

$$Q(x, \xi) := \min_{y \in \mathbb{R}_+^m} q^\top y \quad = \max_{\lambda \in D} (\mathbf{h} - \mathbf{T}x)^\top \lambda$$

s.t. $\mathbf{T}x + Wy = \mathbf{h}$

We define

$$X := \{x \in \mathbb{R}_+^n \mid Ax = b\} \quad D := \{\lambda \in \mathbb{R}^\ell \mid W^\top \lambda \leq q\}$$

¹Can be extended to generic random q , and finitely supported W

2 stage stochastic linear programming (2SLP)

$$\min_{x \in X} \quad c^\top x + \mathbb{E}[Q(x, \xi)]$$

where $\xi = (T, h)$ is random whereas q and W are deterministic¹

$$Q(x, \xi) := \min_{y \in \mathbb{R}_+^m} q^\top y \quad = \max_{\lambda \in D} (h - Tx)^\top \lambda$$

s.t. $Tx + Wy = h$

We define

$$X := \{x \in \mathbb{R}_+^n \mid Ax = b\} \quad D := \{\lambda \in \mathbb{R}^\ell \mid W^\top \lambda \leq q\}$$

No direct formula to compute $V(x) := \mathbb{E}[Q(x, \xi)]$ even for fixed x .

¹Can be extended to generic random q , and finitely supported W

2 stage stochastic linear programming (2SLP)

$$\min_{x \in X} \quad c^\top x + \mathbb{E}[Q(x, \xi)]$$

where $\xi = (\mathbf{T}, \mathbf{h})$ is random whereas q and W are deterministic¹

$$Q(x, \xi) := \min_{y \in \mathbb{R}_+^m} q^\top y \quad = \max_{\lambda \in D} (\mathbf{h} - \mathbf{T}x)^\top \lambda$$

s.t. $\mathbf{T}x + Wy = \mathbf{h}$

We define

$$X := \{x \in \mathbb{R}_+^n \mid Ax = b\} \quad D := \{\lambda \in \mathbb{R}^\ell \mid W^\top \lambda \leq q\}$$

No direct formula to compute $V(x) := \mathbb{E}[Q(x, \xi)]$ even for fixed x .

\rightsquigarrow need to discretize ξ

¹Can be extended to generic random q , and finitely supported W

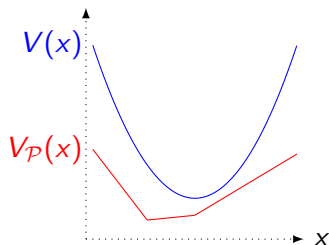
Properties of partitioned cost-to-go

Recall that

$$V(x) = \mathbb{E}[Q(x, \xi)]$$

$$V_{\mathcal{P}}(x) = \sum_{P \in \mathcal{P}} \mathbb{P}[P] Q(x, \mathbb{E}[\xi|P])$$

- $Q(x, \cdot)$ is convex $\rightsquigarrow V_{\mathcal{P}} \leq V$.
- $Q(\cdot, \mathbb{E}[\xi|P])$ is polyhedral $\rightsquigarrow V_{\mathcal{P}}$ is polyhedral.



Finally,

$$\min_{x \in X} c^T x + V_{\mathcal{P}}(x) \quad (2SLP_{\mathcal{P}})$$

is equivalent to

$$\begin{aligned} \min_{x \in X, (y_P)_{P \in \mathcal{P}}} \quad & c^T x + \sum_{P \in \mathcal{P}} \mathbb{P}[P] q^T y_P \\ \text{s.t.} \quad & \mathbb{E}[T|P]x + W y_P \leq \mathbb{E}[h|P] \quad \forall P \in \mathcal{P} \end{aligned}$$

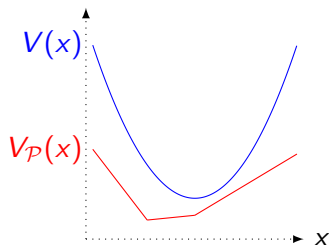
Properties of partitioned cost-to-go

Recall that

$$V(x) = \mathbb{E}[Q(x, \xi)]$$

$$V_{\mathcal{P}}(x) = \sum_{P \in \mathcal{P}} \mathbb{P}[P] Q(x, \mathbb{E}[\xi|P])$$

- $Q(x, \cdot)$ is convex $\rightsquigarrow V_{\mathcal{P}} \leq V$.
- $Q(\cdot, \mathbb{E}[\xi|P])$ is polyhedral $\rightsquigarrow V_{\mathcal{P}}$ is polyhedral.



Finally,

$$\min_{x \in X} c^{\top} x + V_{\mathcal{P}}(x) \quad (2SLP_{\mathcal{P}})$$

is equivalent to

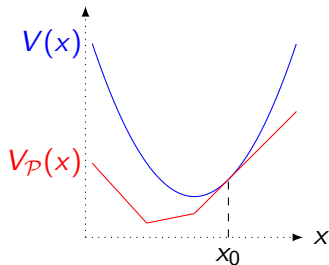
$$\begin{aligned} \min_{x \in X, (y_P)_{P \in \mathcal{P}}} \quad & c^{\top} x + \sum_{P \in \mathcal{P}} \mathbb{P}[P] q^{\top} y_P \\ & \mathbb{E}[\mathbf{T}|P]x + W y_P \leq \mathbb{E}[\mathbf{h}|P] \quad \forall P \in \mathcal{P} \end{aligned}$$

Adapted partition

Definition

We say that a partition \mathcal{P} is *adapted* to x_0 if

$$V_{\mathcal{P}}(x_0) = V(x_0) := \mathbb{E}[Q(x_0, \xi)]$$

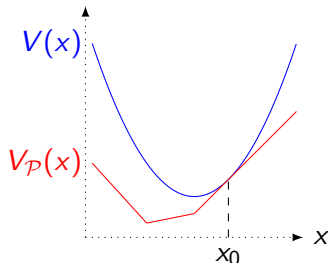


Adapted partition

Definition

We say that a partition \mathcal{P} is *adapted* to x_0 if

$$V_{\mathcal{P}}(x_0) = V(x_0) := \mathbb{E}[Q(x_0, \xi)]$$



Definition

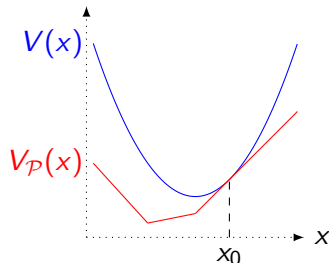
An *partition oracle* is a function taking a first stage decision x^k as argument and returning an partition of Ξ .

Adapted partition

Definition

We say that a partition \mathcal{P} is *adapted* to x_0 if

$$V_{\mathcal{P}}(x_0) = V(x_0) := \mathbb{E}[Q(x_0, \xi)]$$



Definition

An *partition oracle* is a function taking a first stage decision x^k as argument and returning an partition of Ξ .

Definition

An *adapted partition oracle* is a function taking a first stage decision x^k as argument and returning an *adapted to x^k* partition of Ξ .

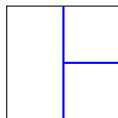
Refinement

\mathcal{R} **refines** \mathcal{P} ($\mathcal{R} \preceq \mathcal{P}$) if

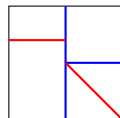
$$\forall R \in \mathcal{R}, \exists P \in \mathcal{P}, R \subset P$$

[$\mathcal{R} \preceq_{\mathbb{P}} \mathcal{P}$ if \mathcal{R} refines \mathcal{P} up to \mathbb{P} -null sets.]

Then, $\mathcal{R} \preceq_{\mathbb{P}} \mathcal{P} \Rightarrow V_{\mathcal{R}} \geq V_{\mathcal{P}}$



\mathcal{P}



\mathcal{R}

Refinement

\mathcal{R} **refines** \mathcal{P} ($\mathcal{R} \preceq \mathcal{P}$) if

$$\forall R \in \mathcal{R}, \exists P \in \mathcal{P}, R \subset P$$

[$\mathcal{R} \preceq_{\mathbb{P}} \mathcal{P}$ if \mathcal{R} refines \mathcal{P} up to \mathbb{P} -null sets.]

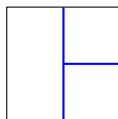
$$\text{Then, } \mathcal{R} \preceq_{\mathbb{P}} \mathcal{P} \Rightarrow V_{\mathcal{R}} \geq V_{\mathcal{P}}$$

The **common refinement** of \mathcal{P} and \mathcal{P}' is

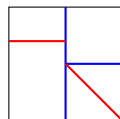
$$\mathcal{P} \wedge \mathcal{P}' := \{P \cap P' \mid P \in \mathcal{P}, P' \in \mathcal{P}'\}$$

Since $\mathcal{P} \wedge \mathcal{P}'$ refines \mathcal{P} and \mathcal{P}'

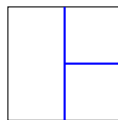
$$\max(V_{\mathcal{P}}, V_{\mathcal{P}'}) \leq V_{\mathcal{P} \wedge \mathcal{P}'}$$



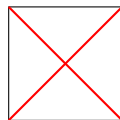
\mathcal{P}



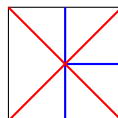
\mathcal{R}



\mathcal{P}



\mathcal{P}'



$\mathcal{P} \wedge \mathcal{P}'$

General framework for APM

```
 $k \leftarrow 0, z_U^0 \leftarrow +\infty, z_L^0 \leftarrow -\infty, \mathcal{P}^0 \leftarrow \{\Xi\} ;$   
while  $z_U^k - z_L^k > \varepsilon$  do  
     $k \leftarrow k + 1;$   
    Solve  $z_L^k \leftarrow \min_{x \in X} c^\top x + V_{\mathcal{P}^{k-1}}(x) ;$   
    and let  $x^k$  be an optimal solution ;  
    Call an adapted partition oracle on  $x^k$  yielding  $\mathcal{P}_{x^k} ;$   
     $\mathcal{P}^k \leftarrow \mathcal{P}^{k-1} \wedge \mathcal{P}_{x^k} ;$   
     $z_U^k \leftarrow \min \left( z_U^{k-1}, c^\top x^k + V_{\mathcal{P}^k}(x^k) \right) ;$   
end
```

Algorithm 1: Generic framework for APM.

Previous APM methods

Lemma (Song & Luedtke, 2015)

Let \mathcal{P} a partition of Ξ . \mathcal{P} is adapted at x iff for all set of scenarios $P \in \mathcal{P}$, there exists a common optimal multiplier λ_P , i.e.

$$\forall P \in \mathcal{P}, \quad \exists \lambda_P \in D, \quad \forall \xi_k \in P, \quad \lambda_P \in \operatorname{argmax}_{\lambda \in D} (h^k - T^k x)^\top \lambda$$

Previous APM methods

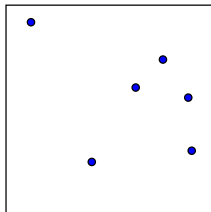
Lemma (Song & Luedtke, 2015)

Let \mathcal{P} a partition of Ξ . \mathcal{P} is adapted at x iff for all set of scenarios $P \in \mathcal{P}$, there exists a common optimal multiplier λ_P , i.e.

$$\forall P \in \mathcal{P}, \quad \exists \lambda_P \in D, \quad \forall \xi_k \in P, \quad \lambda_P \in \operatorname{argmax}_{\lambda \in D} (h^k - T^k x)^\top \lambda$$

Idea

- Sample a large number of scenario
- without loss of precision aggregate scenarios



Previous APM methods

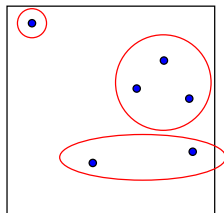
Lemma (Song & Luedtke, 2015)

Let \mathcal{P} a partition of Ξ . \mathcal{P} is adapted at x iff for all set of scenarios $P \in \mathcal{P}$, there exists a common optimal multiplier λ_P , i.e.

$$\forall P \in \mathcal{P}, \quad \exists \lambda_P \in D, \quad \forall \xi_k \in P, \quad \lambda_P \in \operatorname{argmax}_{\lambda \in D} (h^k - T^k x)^\top \lambda$$

Idea

- Sample a large number of scenario
- without loss of precision aggregate scenarios



Previous APM methods

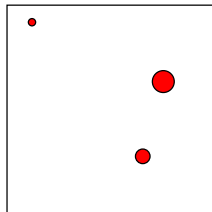
Lemma (Song & Luedtke, 2015)

Let \mathcal{P} a partition of Ξ . \mathcal{P} is adapted at x iff for all set of scenarios $P \in \mathcal{P}$, there exists a common optimal multiplier λ_P , i.e.

$$\forall P \in \mathcal{P}, \quad \exists \lambda_P \in D, \quad \forall \xi_k \in P, \quad \lambda_P \in \operatorname{argmax}_{\lambda \in D} (h^k - T^k x)^\top \lambda$$

Idea

- Sample a large number of scenario
- without loss of precision aggregate scenarios



Previous APM methods

Lemma (Song & Luedtke, 2015)

Let \mathcal{P} a partition of Ξ . \mathcal{P} is adapted at x iff for all set of scenarios $P \in \mathcal{P}$, there exists a common optimal multiplier λ_P , i.e.

$$\forall P \in \mathcal{P}, \quad \exists \lambda_P \in D, \quad \forall \xi_k \in P, \quad \lambda_P \in \operatorname{argmax}_{\lambda \in D} (h^k - T^k x)^\top \lambda$$

Lemma (Ramirez-Pico & Moreno, 2020)

Let \mathcal{P} a partition of Ξ . If there exists $\lambda(\xi)$ such that, for all $P \in \mathcal{P}$,

$$\begin{aligned} \mathbb{E}[h|P]^\top \mathbb{E}[\lambda(\xi)|P] &= \mathbb{E}[h^\top \lambda(\xi)|P] \\ x^\top \mathbb{E}[T|P]^\top \mathbb{E}[\lambda(\xi)|P] &= x^\top \mathbb{E}[T^\top \lambda(\xi)|P] \end{aligned}$$

then \mathcal{P} is an adapted partition.

A (partial) comparison between partition based results

Paper	Song, Luedtke (2015)	Ramirez-Pico, Moreno (2020)	F., Leclère (2021)
Non-finite $\text{supp}(\xi)$	×	✓	✓
Explicit oracle	✓	×	✓
Proof of convergence	✓	×	✓
Complexity result	×	×	✓
Fast iteration	✓	×	×

Contents

5 Explicit formulas for general distributions

6 Details on GAPM

- Recalls on APM
- A novel APM algorithm
- Extension of GAPM to general costs

7 Nested fiber polyhedra

8 Polyhedral toolbox for stochastic optimizers

- Active constraints
- Link with regular subdivisions
- Correspondences for parametric linear programming
- Correspondences for 2SLP

Local exact quantization and adapted partition

Local exact quantization

random cost

Recall that for a fixed x ,

$$\begin{aligned}\mathbb{E} \left[\min_{y \in P_x} \mathbf{c}^\top y \right] \\ = \sum_{N \in \mathcal{N}(P_x)} p_N \min_{y \in P_x} \check{\mathbf{c}}_N^\top y\end{aligned}$$

where,

$$\begin{aligned}p_N &:= \mathbb{P}[\mathbf{c} \in -\text{ri } N] \\ \check{\mathbf{c}}_N &:= \mathbb{E}[\mathbf{c} \mid \mathbf{c} \in -\text{ri } N]\end{aligned}$$

$$P_x := \{y \in \mathbb{R}^m \mid Ay + Bx \leq b\}$$

GAPM

random constraints

Similarly, for a given q , and all x ,

$$\begin{aligned}V(x) &:= \mathbb{E}[Q(x, \boldsymbol{\xi})] \\ &= \mathbb{E} \left[\max_{\lambda \in D_q} (\mathbf{h} - \mathbf{T}x)^\top \lambda \right] \\ &= \sum_{N \in \mathcal{N}(D_q)} p_N \max_{\lambda \in D_q} \psi_{N,x}^\top \lambda\end{aligned}$$

where,

$$\begin{aligned}p_N &:= \mathbb{P}[\mathbf{h} - \mathbf{T}x \in \text{ri } N] \\ \psi_{N,x} &:= \mathbb{E}[\mathbf{h} - \mathbf{T}x \mid \mathbf{h} - \mathbf{T}x \in \text{ri } N] \\ D_q &:= \{\lambda \in \mathbb{R}^l \mid W^\top \lambda \leq q\}\end{aligned}$$

An explicit adapted partition

Consider $x \in \mathbb{R}^n$ and $N \in \mathcal{N}(D_q)$ a normal cone of D_q . We define

$$E_{N,x} := \{\xi \in \Xi \mid h - Tx \in \text{ri } N\}$$

Theorem (FL 2021)

$\mathcal{R}_x := \{E_{N,x} \mid N \in \mathcal{N}(D_q)\}$ is an adapted partition to x
i.e. $V_{\mathcal{R}_x}(x) = V(x)$

Proof:

$$\begin{aligned} V(x) &:= \mathbb{E}[Q(x, \xi)] \\ &= \sum_{N \in \mathcal{N}(D)} \mathbb{P}[h - Tx \in \text{ri } N] \min_{\lambda \in D} \mathbb{E}[h - Tx \mid h - Tx \in \text{ri } N]^\top \lambda \\ &= \sum_{N \in \mathcal{N}(D)} \mathbb{P}[\xi \in E_{N,x}] Q(\mathbb{E}[\xi \mid \xi \in E_{N,x}], x) = V_{\mathcal{R}_x}(x) \end{aligned}$$

An explicit adapted partition

Consider $x \in \mathbb{R}^n$ and $N \in \mathcal{N}(D_q)$ a normal cone of D_q . We define

$$E_{N,x} := \{\xi \in \Xi \mid h - Tx \in \text{ri } N\}$$

Theorem (FL 2021)

$\mathcal{R}_x := \{E_{N,x} \mid N \in \mathcal{N}(D_q)\}$ is an adapted partition to x
i.e. $V_{\mathcal{R}_x}(x) = V(x)$

Proof:

$$\begin{aligned} V(x) &:= \mathbb{E}[Q(x, \xi)] \\ &= \sum_{N \in \mathcal{N}(D)} \mathbb{P}[h - Tx \in \text{ri } N] \min_{\lambda \in D} \mathbb{E}[h - Tx \mid h - Tx \in \text{ri } N]^\top \lambda \\ &= \sum_{N \in \mathcal{N}(D)} \mathbb{P}[\xi \in E_{N,x}] Q(\mathbb{E}[\xi \mid \xi \in E_{N,x}], x) = V_{\mathcal{R}_x}(x) \end{aligned}$$

An explicit adapted partition

Consider $x \in \mathbb{R}^n$ and $N \in \mathcal{N}(D_q)$ a normal cone of D_q . We define

$$E_{N,x} := \{\xi \in \Xi \mid h - Tx \in \text{ri } N\}$$

Theorem (FL 2021)

$\mathcal{R}_x := \{E_{N,x} \mid N \in \mathcal{N}(D_q)\}$ is an adapted partition to x
i.e. $V_{\mathcal{R}_x}(x) = V(x)$

Proof:

$$\begin{aligned} V(x) &:= \mathbb{E}[Q(x, \xi)] \\ &= \sum_{N \in \mathcal{N}(D)} \mathbb{P}[h - Tx \in \text{ri } N] \min_{\lambda \in D} \mathbb{E}[h - Tx \mid h - Tx \in \text{ri } N]^\top \lambda \\ &= \sum_{N \in \mathcal{N}(D)} \mathbb{P}[\xi \in E_{N,x}] Q(\mathbb{E}[\xi \mid \xi \in E_{N,x}], x) = V_{\mathcal{R}_x}(x) \end{aligned}$$

➡ Is it the coarsest one ?

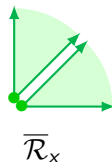
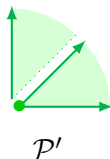
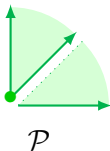
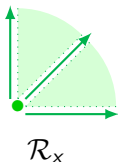
Conditions for a partition to be adapted

Theorem (FL 2021)

For $x \in \mathbb{R}^n$ and \mathcal{P} a partition of Ξ , there exists $\overline{\mathcal{R}}_x \succcurlyeq_{\mathbb{P}} \mathcal{R}_x$ such that

$$\mathcal{P} \preccurlyeq_{\mathbb{P}} \overline{\mathcal{R}}_x \iff V_{\mathcal{P}}(x) = V(x).$$

- If ξ admits a density, $\mathcal{R}_x =_{\mathbb{P}} \overline{\mathcal{R}}_x$.
- An oracle is adapted if and only if it returns a partition \mathcal{P} refining $\overline{\mathcal{R}}_x$.



$$E_{N,x} := \{\xi \in \Xi \mid h - Tx \in \text{ri}(N)\}$$

$$\mathcal{R}_x := \{E_{N,x} \mid N \in \mathcal{N}(D_q)\}$$

$$\overline{E}_{N,x} := \{\xi \in \Xi \mid h - Tx \in N\}$$

$$\overline{\mathcal{R}}_x := \{\overline{E}_{N,x} \mid N \in \mathcal{N}(D_q)^{\max}\}.$$

Subgradient of partition function

Recall that if $\mathcal{P} \preceq_{\mathbb{P}} \mathcal{R}_x$ then

$$V_{\mathcal{R}_x}(x) = V_{\mathcal{P}}(x) = V(x)$$

$$V_{\mathcal{R}_x}(\cdot) \leq V_{\mathcal{P}}(\cdot) \leq V(\cdot)$$

Lemma

Let $x \in \text{dom}(V)$ and \mathcal{P} be a refinement of \mathcal{R}_x , i.e. $\mathcal{P} \preceq \mathcal{R}_x$, then

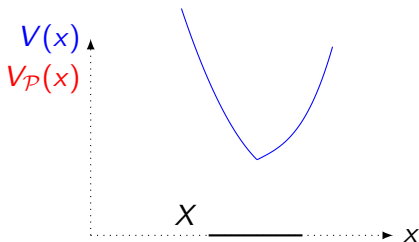
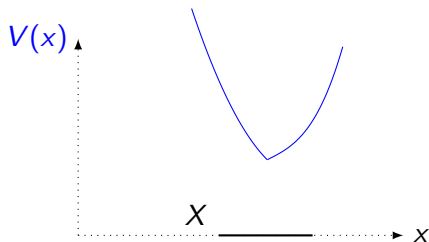
$$\partial V_{\mathcal{R}_x}(x) \subset \partial V_{\mathcal{P}}(x) \subset \partial V(x)$$

Furthermore, if $x \in \text{ri dom}(V)$,

$$\partial V_{\mathcal{R}_x}(x) = \partial V_{\mathcal{P}}(x) = \partial V(x)$$

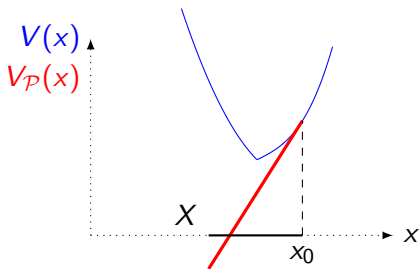
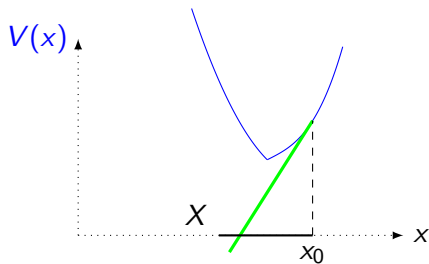
Link with Benders decomposition and L-shaped

Partition based method can be seen as a tangent cone method: a cutting plane method where we add all active cuts instead of a single one.



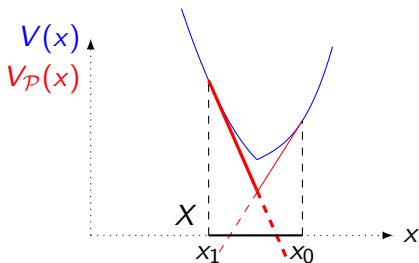
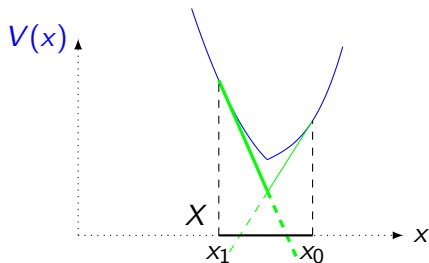
Link with Benders decomposition and L-shaped

Partition based method can be seen as a tangent cone method: a cutting plane method where we add all active cuts instead of a single one.



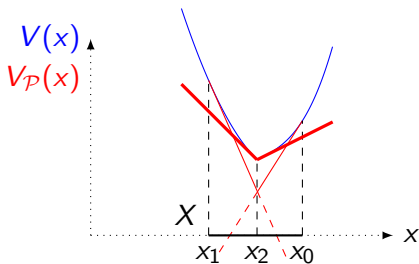
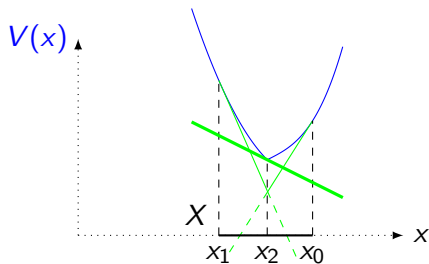
Link with Benders decomposition and L-shaped

Partition based method can be seen as a tangent cone method: a cutting plane method where we add all active cuts instead of a single one.



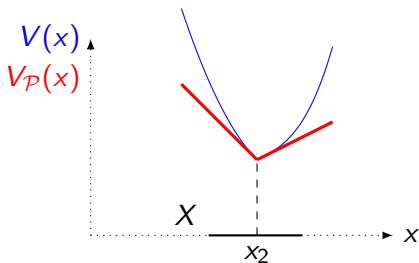
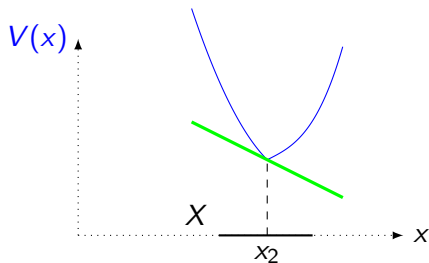
Link with Benders decomposition and L-shaped

Partition based method can be seen as a tangent cone method: a cutting plane method where we add all active cuts instead of a single one.



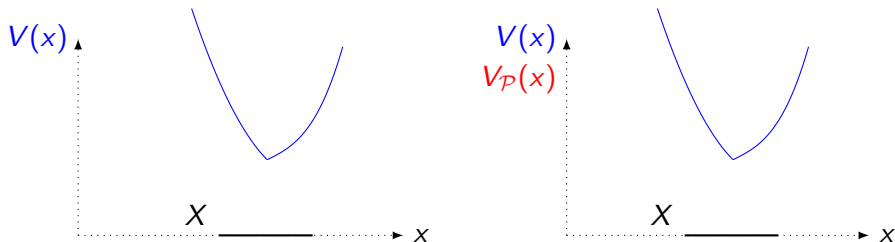
Link with Benders decomposition and L-shaped

Partition based method can be seen as a tangent cone method: a cutting plane method where we add all active cuts instead of a single one.



Link with Benders decomposition and L-shaped

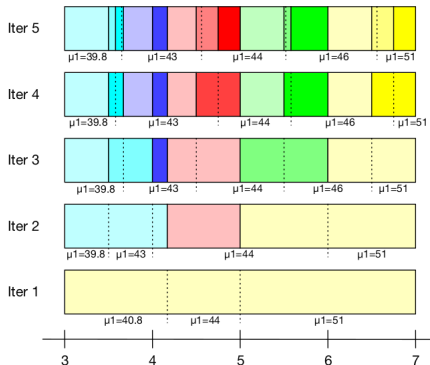
Partition based method can be seen as a tangent cone method: a cutting plane method where we add all active cuts instead of a single one.



Theorem (Convergence and complexity results)

If $X \cap \text{dom}(V) \subset \mathbb{R}^+$ is contained in a ball of diameter $M \in \mathbb{R}^+$ and $x \rightarrow c^\top x + V(x)$ is Lipschitz with constant L then the partition based method finds an ε -solution in at most $(\frac{LM}{\varepsilon} + 1)^n$ iterations.

Numerical Results - LandS



Iter	x_1	x_2	x_3	x_4
1	0.833	3.000	4.167	4.000
2	2.500	3.000	3.500	3.000
3	1.833	4.000	3.667	2.500
4	2.000	4.167	3.583	2.250
5	1.917	4.083	3.625	2.375
6	1.875	4.042	3.646	2.438

Iter	LB	UB	Gap
1	378.667	382.711	1.0567%
2	380.122	381.100	0.2567%
3	380.601	380.844	0.0640%
4	380.842	380.893	0.0007%
5	380.843	380.856	0.0004%
6	380.844	380.847	0.0002%

Results given by GAPM for LandS problem²

²illustration from Ramirez-Pico and Moreno

Numerical Results - ProdMix

k	x_k	z_L^k	z_U^k	Gap	$ \mathcal{P}_k^{\max} $
1	(1333.33, 66.67)	-18666.67	-16939.71	9.3%	4
2	(1441.41, 59.57)	-17873.01	-17383.73	2.7%	9
3	(1399.05, 57.91)	-17789.88	-17659.19	0.74%	16
4	(1379.98, 56.64)	-17744.67	-17708.00	0.20%	25
5	(1371.36, 55.71)	-17718.96	-17709.05	0.056%	36
6	(1375.55, 56.21)	-17713.74	-17711.37	0.013%	49

Table: Results for problem Prod-Mix

To compare our approach with SAA, we solved the same problem 100 times, each with 10 000 scenarios randomly drawn, yielding a 95% confidence interval centered in -17711 , with radius 2.2.

Contents

5 Explicit formulas for general distributions

6 Details on GAPM

- Recalls on APM
- A novel APM algorithm
- Extension of GAPM to general costs

7 Nested fiber polyhedra

8 Polyhedral toolbox for stochastic optimizers

- Active constraints
- Link with regular subdivisions
- Correspondences for parametric linear programming
- Correspondences for 2SLP

Synthesis of local and uniform quantization results

	\mathbf{W}	(\mathbf{T}, \mathbf{h})	\mathbf{q}
Local	\emptyset	\mathcal{R}_x	$\mathcal{N}(P_x)$
Uniform	\emptyset	\emptyset	$\bigwedge_{\sigma \in \mathcal{C}(P, \pi)} \mathcal{N}_\sigma$

Stochastic cost and recourse

- We have shown a local exact quantization result for random \mathbf{T}, \mathbf{h} , and deterministic \mathbf{q}, \mathbf{W} .
- If \mathbf{q} and \mathbf{W} are finitely supported random variable:
 - 1 compute an exact quantization \mathcal{N}_ξ for every element of the support;
 - 2 take the common refinement.

We have seen that we can deal with non-finitely supported \mathbf{q} through the chamber complexes.

➡ Can we do the same here ?

Stochastic cost and recourse

- We have shown a local exact quantization result for random \mathbf{T}, \mathbf{h} , and deterministic \mathbf{q}, \mathbf{W} .
- If \mathbf{q} and \mathbf{W} are finitely supported random variable:
 - 1 compute an exact quantization \mathcal{N}_ξ for every element of the support;
 - 2 take the common refinement.

We have seen that we can deal with non-finitely supported \mathbf{q} through the chamber complexes.

➡ Can we do the same here ?

Adapted partition for general q

We define coupling constraint and fiber for the dual.

$$\begin{aligned} D_q &:= \{\lambda \in \mathbb{R}^\ell \mid W^\top \lambda \leq q\} \\ \Delta &:= \{(\lambda, q) \in \mathbb{R}^\ell \times \mathbb{R}^m \mid W^\top \lambda \leq q\} \\ \mathcal{R}_{x,q} &:= \{E_{N,x} \mid N \in \mathcal{N}(D_q)\} \end{aligned}$$

Recall that $q \mapsto \mathcal{N}(D_q)$ is piecewise constant on $\mathcal{C}(\Delta, \pi_\lambda^{\lambda,q})$ and so is $\mathcal{R}_{x,q}$.
⇒ we can take the common refinement of a finite number of $\mathcal{R}_{x,q}$!!

More precisely:

- The chamber complex $\mathcal{C}(\Delta, \pi_\lambda^{\lambda,q}) = \Sigma\text{-fan}(W)^3$.
 - For $S \in \Sigma\text{-fan}(W)$ define $\mathcal{R}_{x,S} := \mathcal{R}_{x,q}$ for any $q \in \text{ri}(S)$.
- ⇒ $\{\text{ri}(S) \times R \mid S \in \Sigma\text{-fan}(W), R \in \mathcal{R}_{x,S}\}$ is an adapted partition to x .

³The well studied secondary fan of W

Adapted partition for general q

We define coupling constraint and fiber for the dual.

$$\begin{aligned} D_q &:= \{\lambda \in \mathbb{R}^\ell \mid W^\top \lambda \leq q\} \\ \Delta &:= \{(\lambda, q) \in \mathbb{R}^\ell \times \mathbb{R}^m \mid W^\top \lambda \leq q\} \\ \mathcal{R}_{x,q} &:= \{E_{N,x} \mid N \in \mathcal{N}(D_q)\} \end{aligned}$$

Recall that $q \mapsto \mathcal{N}(D_q)$ is piecewise constant on $\mathcal{C}(\Delta, \pi_\lambda^{\lambda,q})$ and so is $\mathcal{R}_{x,q}$.
➡ we can take the common refinement of a finite number of $\mathcal{R}_{x,q}$!!

More precisely:

- The chamber complex $\mathcal{C}(\Delta, \pi_\lambda^{\lambda,q}) = \Sigma\text{-fan}(W)^3$.
 - For $S \in \Sigma\text{-fan}(W)$ define $\mathcal{R}_{x,S} := \mathcal{R}_{x,q}$ for any $q \in \text{ri}(S)$.
- ➡ $\{\text{ri}(S) \times R \mid S \in \Sigma\text{-fan}(W), R \in \mathcal{R}_{x,S}\}$ is an adapted partition to x .

³The well studied secondary fan of W

Adapted partition for general q

We define coupling constraint and fiber for the dual.

$$\begin{aligned} D_q &:= \{ \lambda \in \mathbb{R}^\ell \mid W^\top \lambda \leq q \} \\ \Delta &:= \{ (\lambda, q) \in \mathbb{R}^\ell \times \mathbb{R}^m \mid W^\top \lambda \leq q \} \\ \mathcal{R}_{x,q} &:= \{ E_{N,x} \mid N \in \mathcal{N}(D_q) \} \end{aligned}$$

Recall that $q \mapsto \mathcal{N}(D_q)$ is piecewise constant on $\mathcal{C}(\Delta, \pi_\lambda^{\lambda,q})$ and so is $\mathcal{R}_{x,q}$.
➡ we can take the common refinement of a finite number of $\mathcal{R}_{x,q}$!!

More precisely:

- The chamber complex $\mathcal{C}(\Delta, \pi_\lambda^{\lambda,q}) = \Sigma\text{-fan}(W)^3$.
 - For $S \in \Sigma\text{-fan}(W)$ define $\mathcal{R}_{x,S} := \mathcal{R}_{x,q}$ for any $q \in \text{ri}(S)$.
- ➡ $\{ \text{ri}(S) \times R \mid S \in \Sigma\text{-fan}(W), R \in \mathcal{R}_{x,S} \}$ is an adapted partition to x .

³The well studied secondary fan of W

Contents

5 Explicit formulas for general distributions

6 Details on GAPM

- Recalls on APM
- A novel APM algorithm
- Extension of GAPM to general costs

7 Nested fiber polyhedra

8 Polyhedral toolbox for stochastic optimizers

- Active constraints
- Link with regular subdivisions
- Correspondences for parametric linear programming
- Correspondences for 2SLP

Dual problem

$$V(x) := \mathbb{E} \left[\begin{array}{ll} \inf_y & \mathbf{c}^\top y \\ \text{s.t.} & Ax + By \leq b \end{array} \right] = \mathbb{E} \left[\inf_{y \in P_x} \mathbf{c}^\top y \right]$$

where $P_x = \{y \mid Ax + By \leq b\}$

$$V(x) := \mathbb{E} \left[\begin{array}{ll} \sup_{\mu} & (Ax - b)^\top \mu \\ \text{s.t.} & B^\top \mu + \mathbf{c} = 0 \\ & \mu \geq 0 \end{array} \right] = \mathbb{E} \left[\sup_{\mu \in D_c} (Ax - b)^\top \mu \right]$$

where $D_c = \{\mu \mid B^\top \mu + \mathbf{c} = 0, \mu \geq 0\}$

Fiber Polyhedron

Minkowski sum :

$$E + F = \{x + x' \mid x \in E, x' \in F\}$$

Definition

The *fiber polyhedron* E of the bundle $(D_c)_{c \in \text{supp}(\mathbf{c})}$ is the Minkowsky integral of all the fiber at c when c varies according to its probability distribution:

$$E := \int D_c \mathbb{P}(dc) = \left\{ \int \mu(c) \mathbb{P}(dc) \mid \mu(c) \in D_c \text{ a.s., } \mu \in L_\infty(\mathbb{R}^m, \mathbb{R}^l) \right\}$$

$$\begin{aligned} V(x) &= \mathbb{E} \left[\sup_{\mu \in D_c} (Ax - b)^\top \mu \right] \\ &= \begin{cases} \sup_{\mu(\cdot)} & (Ax - b)^\top \mathbb{E}[\mu(\mathbf{c})] \\ \text{s.t.} & \mu(\mathbf{c}) \in D_c \text{ a.s.} \end{cases} \end{aligned}$$

Fiber Polyhedron

Minkowski sum :

$$E + F = \{x + x' \mid x \in E, x' \in F\}$$

Definition

The *fiber polyhedron* E of the bundle $(D_c)_{c \in \text{supp}(\mathbf{c})}$ is the Minkowsky integral of all the fiber at c when c varies according to its probability distribution:

$$E := \int D_c \mathbb{P}(dc) = \left\{ \int \mu(c) \mathbb{P}(dc) \mid \mu(c) \in D_c \text{ a.s., } \mu \in L_\infty(\mathbb{R}^m, \mathbb{R}^l) \right\}$$

$$\begin{aligned} V(x) &= \mathbb{E} \left[\sup_{\mu \in D_c} (Ax - b)^\top \mu \right] \\ &= \begin{cases} \sup_{\mu(\cdot)} & (Ax - b)^\top \mathbb{E}[\mu(\mathbf{c})] \\ \text{s.t.} & \mu(\mathbf{c}) \in D_c \text{ a.s.} \end{cases} \end{aligned}$$

Fiber Polyhedron

Minkowski sum :

$$E + F = \{x + x' \mid x \in E, x' \in F\}$$

Definition

The *fiber polyhedron* E of the bundle $(D_c)_{c \in \text{supp}(\mathbf{c})}$ is the Minkowsky integral of all the fiber at c when c varies according to its probability distribution:

$$E := \int D_c \mathbb{P}(dc) = \left\{ \int \mu(c) \mathbb{P}(dc) \mid \mu(c) \in D_c \text{ a.s., } \mu \in L_\infty(\mathbb{R}^m, \mathbb{R}^l) \right\}$$

$$\begin{aligned} V(x) &= \mathbb{E} \left[\sup_{\mu \in D_c} (Ax - b)^\top \mu \right] \\ &= \begin{cases} \sup_{\mu(\cdot)} & (Ax - b)^\top \mathbb{E}[\mu(\mathbf{c})] \\ \text{s.t.} & \mathbb{E}[\mu(\mathbf{c})] \in E \end{cases} \end{aligned}$$

Fiber Polyhedron

Minkowski sum :

$$E + F = \{x + x' \mid x \in E, x' \in F\}$$

Definition

The *fiber polyhedron* E of the bundle $(D_c)_{c \in \text{supp}(\mathbf{c})}$ is the Minkowsky integral of all the fiber at c when c varies according to its probability distribution:

$$E := \int D_c \mathbb{P}(dc) = \left\{ \int \mu(c) \mathbb{P}(dc) \mid \mu(c) \in D_c \text{ a.s., } \mu \in L_\infty(\mathbb{R}^m, \mathbb{R}^l) \right\}$$

$$\begin{aligned} V(x) &= \mathbb{E} \left[\sup_{\mu \in D_c} (Ax - b)^\top \mu \right] \\ &= \begin{cases} \sup_{\mu(\cdot)} & (Ax - b)^\top \mathbb{E}[\mu(\mathbf{c})] \\ \text{s.t.} & \mathbb{E}[\mu(\mathbf{c})] \in E \end{cases} \\ &= \sup (Ax - b)^\top \lambda \end{aligned}$$

The Fiber Polyhedron is a finite Minkowski sum

Theorem

There exists a chamber complex \mathcal{R} depending on A such that

$$E = \int D_{\mathbf{c}} \mathbb{P}(d\mathbf{c}) = \sum_{R \in \mathcal{R}} \check{p}_R D_{\check{\mathbf{c}}_R}$$

where $\check{p}_R := \mathbb{P}[\mathbf{c} \in \text{ri}(R)]$ and $\check{\mathbf{c}}_R := \mathbb{E}[\mathbf{c} \mid \mathbf{c} \in \text{ri}(R)]$.

Alternative proof of the quantization result

$$V(x) = \sigma_E(Ax - b) = \sum_{R \in \mathcal{R}} \check{p}_R \sigma_{D_{\check{\mathbf{c}}_R}}(Ax - b) = \sum_{R \in \mathcal{R}} \check{p}_R \min_{y \in P_x} \check{\mathbf{c}}_R^\top y$$

The Fiber Polyhedron is a finite Minkowski sum

Theorem

There exists a chamber complex \mathcal{R} depending on A such that

$$E = \int D_{\mathbf{c}} \mathbb{P}(d\mathbf{c}) = \sum_{R \in \mathcal{R}} \check{p}_R D_{\check{\mathbf{c}}_R}$$

where $\check{p}_R := \mathbb{P}[\mathbf{c} \in \text{ri}(R)]$ and $\check{\mathbf{c}}_R := \mathbb{E}[\mathbf{c} \mid \mathbf{c} \in \text{ri}(R)]$.

Alternative proof of the quantization result

$$V(x) = \sigma_E(Ax - b) = \sum_{R \in \mathcal{R}} \check{p}_R \sigma_{D_{\check{\mathbf{c}}_R}}(Ax - b) = \sum_{R \in \mathcal{R}} \check{p}_R \min_{y \in P_x} \check{\mathbf{c}}_R^\top y$$

Nested Fiber Polyhedra for Multistage

$$V_t(x_{t-1}) = \mathbb{E} \left[\begin{array}{l} \min_{x_t \in \mathbb{R}^{n_t}} \mathbf{c}_t^\top x_t + V_{t+1}(x_t) \\ \text{s.t. } A_t x_t + B_t x_{t-1} \leq b_t \end{array} \right]$$

Definition

We define by induction the following nested fiber polyhedra

$$D_{t,c_t} := \{\mu_t \mid \mu_t \geq 0, A_t^\top \mu_t + c_t = 0\} \quad \forall t \in [T]$$

$$F_{T,c_T} := D_{T,c_T}$$

$$E_t := \mathbb{E}[F_{t,c_t}] \quad \forall t \in [T]$$

$$F_{t,c_t} := \{(\mu_t, \lambda_{[t+1:T]}) \mid \mu_t \in D_{t,c_t + B_{t+1}^\top \lambda_{t+1}}, \lambda_{[t+1:T]} \in E_{t+1}\} \quad \forall t \in [T-1]$$

$$V_t(x_{t-1}) = \sigma_{E_t}(B_t x_{t-1} - b_t, -b_{[t+1:T]})$$

Nested Fiber Polyhedra for Multistage

$$V_t(x_{t-1}) = \mathbb{E} \left[\begin{array}{l} \min_{x_t \in \mathbb{R}^{n_t}} \mathbf{c}_t^\top x_t + V_{t+1}(x_t) \\ \text{s.t. } A_t x_t + B_t x_{t-1} \leq b_t \end{array} \right]$$

Definition

We define by induction the following nested fiber polyhedra

$$D_{t, \mathbf{c}_t} := \{\mu_t \mid \mu_t \geq 0, A_t^\top \mu_t + \mathbf{c}_t = 0\} \quad \forall t \in [T]$$

$$F_{T, \mathbf{c}_T} := D_{T, \mathbf{c}_T}$$

$$E_t := \mathbb{E}[F_{t, \mathbf{c}_t}] \quad \forall t \in [T]$$

$$F_{t, \mathbf{c}_t} := \{(\mu_t, \lambda_{[t+1:T]}) \mid \mu_t \in D_{t, \mathbf{c}_t + B_{t+1}^\top \lambda_{t+1}}, \lambda_{[t+1:T]} \in E_{t+1}\} \quad \forall t \in [T-1]$$

$$V_t(x_{t-1}) = \sigma_{E_t}(B_t x_{t-1} - b_t, -b_{[t+1:T]})$$

2-time scale MSLP reduced to Quadratic Problem

We consider a MSLP where the dynamic depends on parameters p we have to optimize

$$\begin{aligned} \min_{p \in \mathbb{R}^m, (\mathbf{x}_t) \in \mathbb{R}^{nt}} \quad & q^\top p + \mathbb{E} \left[\sum_{t=1}^T \mathbf{c}_t^\top \mathbf{x}_t \right] \\ \text{s.t.} \quad & Dp \leq d \\ & A_t \mathbf{x}_t + B_t \mathbf{x}_{t-1} + C_t p \leq h_t \quad \text{a.s.} \quad \forall t \in [T] \\ & \mathbf{x}_t \prec \sigma(\mathbf{c}_1, \dots, \mathbf{c}_t) \quad \forall t \in [T] \end{aligned}$$

If we know the fiber polyhedron, it reduces to a finite dimensional quadratic problem

$$\begin{aligned} \min_{p \in \mathbb{R}^m} \quad & q^\top p + \sup_{(\lambda_t)_{t \in [T]}} \sum_{t=1}^T (C_t p - h_t)^\top \lambda_t \\ \text{s.t.} \quad & Dp \leq d \\ & (\lambda_1, \dots, \lambda_T) \in E_1 \end{aligned}$$

Contents

5 Explicit formulas for general distributions

6 Details on GAPM

- Recalls on APM
- A novel APM algorithm
- Extension of GAPM to general costs

7 Nested fiber polyhedra

8 Polyhedral toolbox for stochastic optimizers

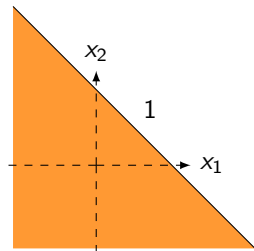
- Active constraints
- Link with regular subdivisions
- Correspondences for parametric linear programming
- Correspondences for 2SLP

Linear Programming and polyhedra

$$\begin{array}{ll}\min_{x \in \mathbb{R}^n} & c^\top x \\ \text{s.t.} & Ax \leq b\end{array}$$

Example: $P = \{x \in \mathbb{R}^n \mid Ax \leq b\}$

$$A = \begin{pmatrix} 1 & 1 \end{pmatrix} \quad b = \begin{pmatrix} 1 \end{pmatrix} \quad x_1 + x_2 \leq 1$$

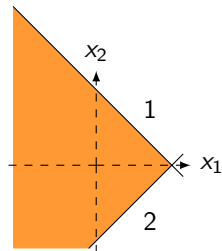


Linear Programming and polyhedra

$$\begin{array}{ll}\min_{x \in \mathbb{R}^n} & c^\top x \\ \text{s.t.} & Ax \leq b\end{array}$$

Example: $P = \{x \in \mathbb{R}^n \mid Ax \leq b\}$

$$A = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \quad b = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$
$$\begin{array}{l} x_1 + x_2 \leq 1 \\ x_1 - x_2 \leq 1 \end{array}$$

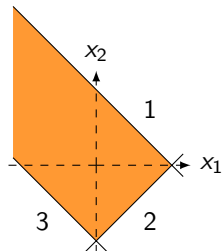


Linear Programming and polyhedra

$$\begin{array}{ll}\min_{x \in \mathbb{R}^n} & c^\top x \\ \text{s.t.} & Ax \leq b\end{array}$$

Example: $P = \{x \in \mathbb{R}^n \mid Ax \leq b\}$

$$A = \begin{pmatrix} 1 & 1 \\ 1 & -1 \\ -1 & -1 \end{pmatrix} \quad b = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \quad \begin{array}{l} x_1 + x_2 \leq 1 \\ x_1 - x_2 \leq 1 \\ -x_1 - x_2 \leq 1 \end{array}$$

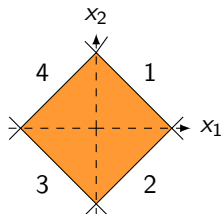


Linear Programming and polyhedra

$$\begin{array}{ll}\min_{x \in \mathbb{R}^n} & c^\top x \\ \text{s.t.} & Ax \leq b\end{array}$$

Example: $P = \{x \in \mathbb{R}^n \mid Ax \leq b\}$

$$A = \begin{pmatrix} 1 & 1 \\ 1 & -1 \\ -1 & -1 \\ -1 & 1 \end{pmatrix} \quad b = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} \quad \begin{array}{l} x_1 + x_2 \leq 1 \\ x_1 - x_2 \leq 1 \\ -x_1 - x_2 \leq 1 \\ -x_1 + x_2 \leq 1 \end{array}$$

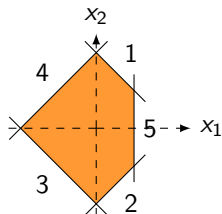


Linear Programming and polyhedra

$$\begin{array}{ll}\min_{x \in \mathbb{R}^n} & c^\top x \\ \text{s.t.} & Ax \leq b\end{array}$$

Example: $P = \{x \in \mathbb{R}^n \mid Ax \leq b\}$

$$A = \begin{pmatrix} 1 & 1 \\ 1 & -1 \\ -1 & -1 \\ -1 & 1 \end{pmatrix} \quad b = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 0.5 \end{pmatrix}$$
$$\begin{array}{l} x_1 + x_2 \leq 1 \\ x_1 - x_2 \leq 1 \\ -x_1 - x_2 \leq 1 \\ -x_1 + x_2 \leq 1 \\ x_1 \leq 0.5 \end{array}$$

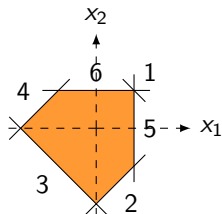


Linear Programming and polyhedra

$$\begin{array}{ll}\min_{x \in \mathbb{R}^n} & c^\top x \\ \text{s.t.} & Ax \leq b\end{array}$$

Example: $P = \{x \in \mathbb{R}^n \mid Ax \leq b\}$

$$A = \begin{pmatrix} 1 & 1 \\ 1 & -1 \\ -1 & -1 \\ -1 & 1 \\ 1 & 0 \\ 0 & 1 \end{pmatrix} \quad b = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 0.5 \\ 0.5 \end{pmatrix}$$
$$\begin{array}{l} x_1 + x_2 \leq 1 \\ x_1 - x_2 \leq 1 \\ -x_1 - x_2 \leq 1 \\ -x_1 + x_2 \leq 1 \\ x_1 \leq 0.5 \\ x_2 \leq 0.5 \end{array}$$

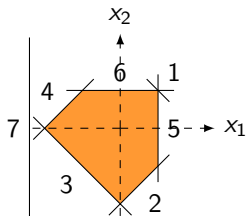


Linear Programming and polyhedra

$$\begin{array}{ll}\min_{x \in \mathbb{R}^n} & c^\top x \\ \text{s.t.} & Ax \leq b\end{array}$$

Example: $P = \{x \in \mathbb{R}^n \mid Ax \leq b\}$

$$A = \begin{pmatrix} 1 & 1 \\ & 1 \\ & 1 \\ & 1 \\ 0.5 & \\ 0.5 & \\ -1.2 & \end{pmatrix} \quad b = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 0.5 \\ 0.5 \\ -1.2 \end{pmatrix}$$
$$\begin{array}{l} x_1 + x_2 \leq 1 \\ x_1 - x_2 \leq 1 \\ -x_1 - x_2 \leq 1 \\ -x_1 + x_2 \leq 1 \\ x_1 \leq 0.5 \\ x_2 \leq 0.5 \\ x_1 \geq -1.2 \end{array}$$



Contents

5 Explicit formulas for general distributions

6 Details on GAPM

- Recalls on APM
- A novel APM algorithm
- Extension of GAPM to general costs

7 Nested fiber polyhedra

8 Polyhedral toolbox for stochastic optimizers

- **Active constraints**
- Link with regular subdivisions
- Correspondences for parametric linear programming
- Correspondences for 2SLP

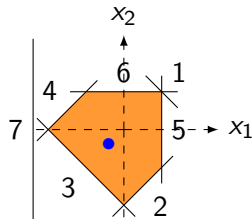
Active constraints

Definition

We denote by $\mathcal{I}(A, b)$, the collection of sets of active constraints as :

$$\mathcal{I}(A, b) = \{I_{A,b}(x) \mid Ax \leq b\}$$

with $I_{A,b}(x) := \{i \in [q] \mid A_i x = b_i\}$



$$I_{A,b}(x) = \emptyset$$

To ease the notation, we write:

$$\mathcal{I}(A, b) = \{\emptyset, \quad \quad \quad \}$$

$$P = \{x \in \mathbb{R}^n \mid Ax \leq b\}$$

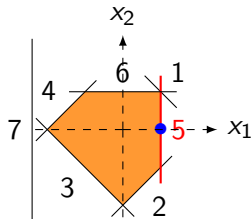
Active constraints

Definition

We denote by $\mathcal{I}(A, b)$, the collection of sets of active constraints as :

$$\mathcal{I}(A, b) = \{I_{A,b}(x) \mid Ax \leq b\}$$

with $I_{A,b}(x) := \{i \in [q] \mid A_i x = b_i\}$



$$I_{A,b}(x) = \{5\}$$

To ease the notation, we write:

$$\mathcal{I}(A, b) = \{\emptyset, 5, \quad \quad \quad \}$$

$$P = \{x \in \mathbb{R}^n \mid Ax \leq b\}$$

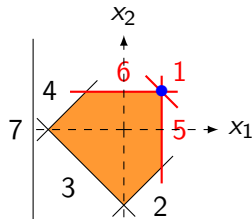
Active constraints

Definition

We denote by $\mathcal{I}(A, b)$, the collection of sets of active constraints as :

$$\mathcal{I}(A, b) = \{I_{A,b}(x) \mid Ax \leq b\}$$

with $I_{A,b}(x) := \{i \in [q] \mid A_i x = b_i\}$



$$I_{A,b}(x) = \{1, 5, 6\}$$

To ease the notation, we write:

$$\mathcal{I}(A, b) = \{\emptyset, 5, 156, \quad \quad \quad \}$$

$$P = \{x \in \mathbb{R}^n \mid Ax \leq b\}$$

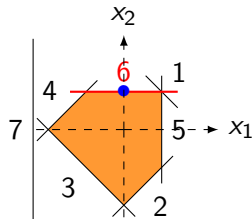
Active constraints

Definition

We denote by $\mathcal{I}(A, b)$, the collection of sets of active constraints as :

$$\mathcal{I}(A, b) = \{I_{A,b}(x) \mid Ax \leq b\}$$

with $I_{A,b}(x) := \{i \in [q] \mid A_i x = b_i\}$



$$I_{A,b}(x) = \{6\}$$

To ease the notation, we write:

$$\mathcal{I}(A, b) = \{\emptyset, 5, 156, 6, \quad \quad \quad \}$$

$$P = \{x \in \mathbb{R}^n \mid Ax \leq b\}$$

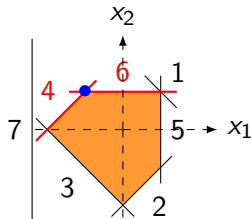
Active constraints

Definition

We denote by $\mathcal{I}(A, b)$, the collection of sets of active constraints as :

$$\mathcal{I}(A, b) = \{I_{A,b}(x) \mid Ax \leq b\}$$

with $I_{A,b}(x) := \{i \in [q] \mid A_i x = b_i\}$



$$I_{A,b}(x) = \{4, 6\}$$

To ease the notation, we write:

$$\mathcal{I}(A, b) = \{\emptyset, 5, 156, 6, 46, \quad \}$$

$$P = \{x \in \mathbb{R}^n \mid Ax \leq b\}$$

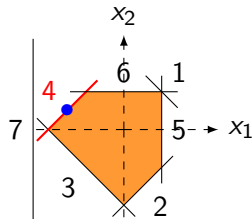
Active constraints

Definition

We denote by $\mathcal{I}(A, b)$, the collection of sets of active constraints as :

$$\mathcal{I}(A, b) = \{I_{A,b}(x) \mid Ax \leq b\}$$

with $I_{A,b}(x) := \{i \in [q] \mid A_i x = b_i\}$



$$I_{A,b}(x) = \{4\}$$

To ease the notation, we write:

$$\mathcal{I}(A, b) = \{\emptyset, 5, 156, 6, 46, 4, \quad \}$$

$$P = \{x \in \mathbb{R}^n \mid Ax \leq b\}$$

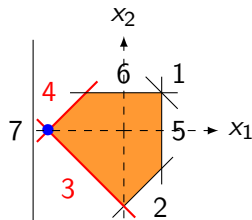
Active constraints

Definition

We denote by $\mathcal{I}(A, b)$, the collection of sets of active constraints as :

$$\mathcal{I}(A, b) = \{I_{A,b}(x) \mid Ax \leq b\}$$

with $I_{A,b}(x) := \{i \in [q] \mid A_i x = b_i\}$



$$I_{A,b}(x) = \{3, 4\}$$

To ease the notation, we write:

$$\mathcal{I}(A, b) = \{\emptyset, 5, 156, 6, 46, 4, 34, \quad \}$$

$$P = \{x \in \mathbb{R}^n \mid Ax \leq b\}$$

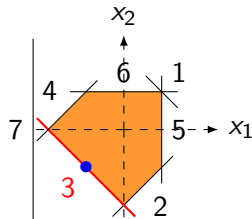
Active constraints

Definition

We denote by $\mathcal{I}(A, b)$, the collection of sets of active constraints as :

$$\mathcal{I}(A, b) = \{I_{A,b}(x) \mid Ax \leq b\}$$

with $I_{A,b}(x) := \{i \in [q] \mid A_i x = b_i\}$



$$I_{A,b}(x) = \{3\}$$

To ease the notation, we write:

$$\mathcal{I}(A, b) = \{\emptyset, 5, 156, 6, 46, 4, 34, 3, \quad \}$$

$$P = \{x \in \mathbb{R}^n \mid Ax \leq b\}$$

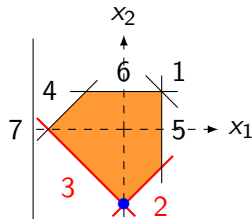
Active constraints

Definition

We denote by $\mathcal{I}(A, b)$, the collection of sets of active constraints as :

$$\mathcal{I}(A, b) = \{I_{A,b}(x) \mid Ax \leq b\}$$

with $I_{A,b}(x) := \{i \in [q] \mid A_i x = b_i\}$



$$I_{A,b}(x) = \{2, 3\}$$

To ease the notation, we write:

$$\mathcal{I}(A, b) = \{\emptyset, 5, 156, 6, 46, 4, 34, 3, 23, \quad \}$$

$$P = \{x \in \mathbb{R}^n \mid Ax \leq b\}$$

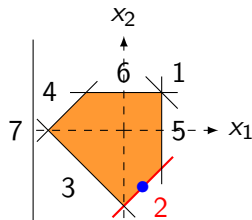
Active constraints

Definition

We denote by $\mathcal{I}(A, b)$, the collection of sets of active constraints as :

$$\mathcal{I}(A, b) = \{I_{A,b}(x) \mid Ax \leq b\}$$

with $I_{A,b}(x) := \{i \in [q] \mid A_i x = b_i\}$



$$I_{A,b}(x) = \{2\}$$

To ease the notation, we write:

$$\mathcal{I}(A, b) = \{\emptyset, 5, 156, 6, 46, 4, 34, 3, 23, 2, \quad \}$$

$$P = \{x \in \mathbb{R}^n \mid Ax \leq b\}$$

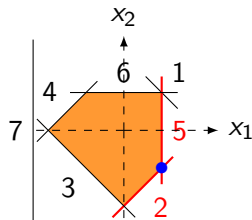
Active constraints

Definition

We denote by $\mathcal{I}(A, b)$, the collection of sets of active constraints as :

$$\mathcal{I}(A, b) = \{I_{A,b}(x) \mid Ax \leq b\}$$

with $I_{A,b}(x) := \{i \in [q] \mid A_i x = b_i\}$



$$I_{A,b}(x) = \{2, 5\}$$

To ease the notation, we write:

$$\mathcal{I}(A, b) = \{\emptyset, 5, 156, 6, 46, 4, 34, 3, 23, 2, 25\}$$

$$P = \{x \in \mathbb{R}^n \mid Ax \leq b\}$$

Faces

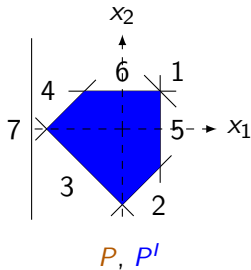
Definition

Let $I \in \mathcal{I}(A, b)$, we denote by P^I the face of P such that:

$$P^I = \{x \in P \mid A_I x = b_I\}$$

We have $\dim(P^I) = n - \text{rg}(A_I)$

Example for $I = \emptyset$



Faces

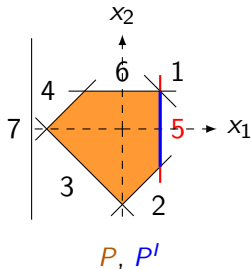
Definition

Let $I \in \mathcal{I}(A, b)$, we denote by P^I the face of P such that:

$$P^I = \{x \in P \mid A_I x = b_I\}$$

We have $\dim(P^I) = n - \text{rg}(A_I)$

Example for $I = \{5\}$



Faces

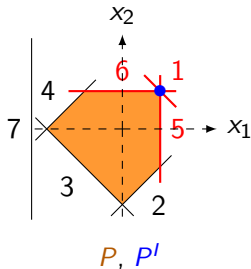
Definition

Let $I \in \mathcal{I}(A, b)$, we denote by P^I the face of P such that:

$$P^I = \{x \in P \mid A_I x = b_I\}$$

We have $\dim(P^I) = n - \text{rg}(A_I)$

Example for $I = \{1, 5, 6\}$



Faces

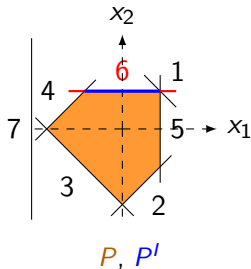
Definition

Let $I \in \mathcal{I}(A, b)$, we denote by P^I the face of P such that:

$$P^I = \{x \in P \mid A_I x = b_I\}$$

We have $\dim(P^I) = n - \text{rg}(A_I)$

Example for $I = \{6\}$



Faces

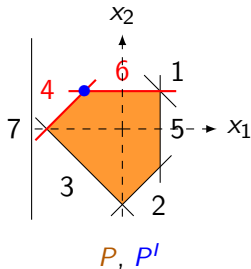
Definition

Let $I \in \mathcal{I}(A, b)$, we denote by P^I the face of P such that:

$$P^I = \{x \in P \mid A_I x = b_I\}$$

We have $\dim(P^I) = n - \text{rg}(A_I)$

Example for $I = \{4, 6\}$



Faces

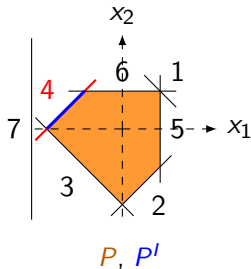
Definition

Let $I \in \mathcal{I}(A, b)$, we denote by P^I the face of P such that:

$$P^I = \{x \in P \mid A_I x = b_I\}$$

We have $\dim(P^I) = n - \text{rg}(A_I)$

Example for $I = \{4\}$



Faces

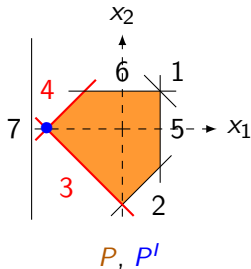
Definition

Let $I \in \mathcal{I}(A, b)$, we denote by P^I the face of P such that:

$$P^I = \{x \in P \mid A_I x = b_I\}$$

We have $\dim(P^I) = n - \text{rg}(A_I)$

Example for $I = \{3, 4\}$



Faces

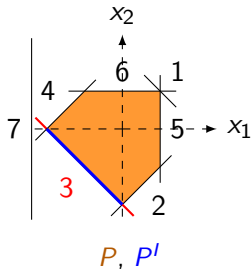
Definition

Let $I \in \mathcal{I}(A, b)$, we denote by P^I the face of P such that:

$$P^I = \{x \in P \mid A_I x = b_I\}$$

We have $\dim(P^I) = n - \text{rg}(A_I)$

Example for $I = \{3\}$



Faces

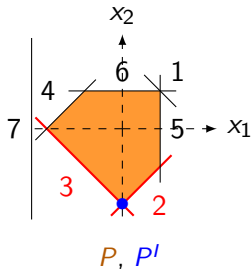
Definition

Let $I \in \mathcal{I}(A, b)$, we denote by P^I the face of P such that:

$$P^I = \{x \in P \mid A_I x = b_I\}$$

We have $\dim(P^I) = n - \text{rg}(A_I)$

Example for $I = \{2, 3\}$



Faces

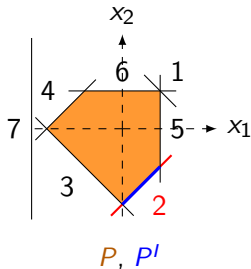
Definition

Let $I \in \mathcal{I}(A, b)$, we denote by P^I the face of P such that:

$$P^I = \{x \in P \mid A_I x = b_I\}$$

We have $\dim(P^I) = n - \text{rg}(A_I)$

Example for $I = \{2\}$



Faces

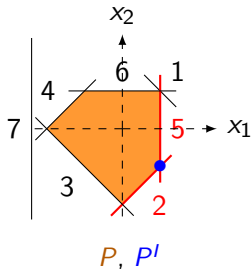
Definition

Let $I \in \mathcal{I}(A, b)$, we denote by P^I the face of P such that:

$$P^I = \{x \in P \mid A_I x = b_I\}$$

We have $\dim(P^I) = n - \text{rg}(A_I)$

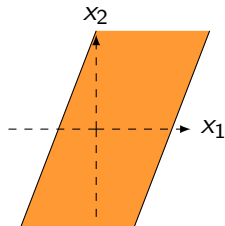
Example for $I = \{2, 5\}$



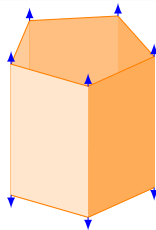
Lineality space, vertices and bases

Definition (Lineality space)

$$\text{Lin}(C) := \{u \in C \mid \forall t \in \mathbb{R}, \forall x \in c, x + tu \in C\}.$$



If
 $P = \{x \in \mathbb{R}^n \mid Ax \leq b\},$
then $\text{Lin}(P) = \text{Ker}(A)$



Definition (Bases and vertices)

A basis B is a subset of $[p]$ such that $A_B = (A_{i,j})_{i \in B, 1 \leq j \leq n}$ is invertible.
A vertex of P is a face of dimension 0. $\text{Vert}(P)$ is the set of vertices.

$\text{Vert}(P) \neq \emptyset \Leftrightarrow A$ admits at least one basis $\Leftrightarrow \text{rg}(A) = n \Leftrightarrow \text{Lin}(P) = \{0\}$

We make this assumption without loss of generality.

Contents

- 5 Explicit formulas for general distributions
- 6 Details on GAPM
 - Recalls on APM
 - A novel APM algorithm
 - Extension of GAPM to general costs
- 7 Nested fiber polyhedra
- 8 Polyhedral toolbox for stochastic optimizers**
 - Active constraints
 - Link with regular subdivisions**
 - Correspondences for parametric linear programming
 - Correspondences for 2SLP

Link with regular subdivisions

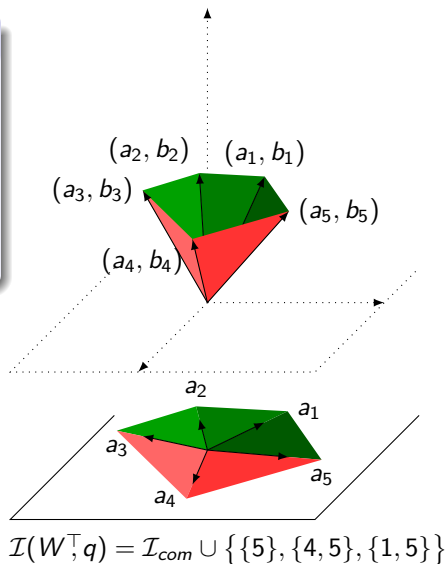
Definition (DLRS10)

$$\mathcal{S}(A^\top, b) := \{I_F \mid F \in \mathcal{F}_{\text{low}}(LC_{A^\top, b})\}$$

$$LC_{A^\top, b} := \text{Cone} \left(\left(\begin{pmatrix} a_i \\ b_i \end{pmatrix} \right)_{i \in [q]} \right)$$

$$I_F := \{i \in [q] \mid (a_i, b_i) \in F\}.$$

$$\mathcal{S}(A^\top, b) = \mathcal{I}(A, b)$$



Link with regular subdivisions

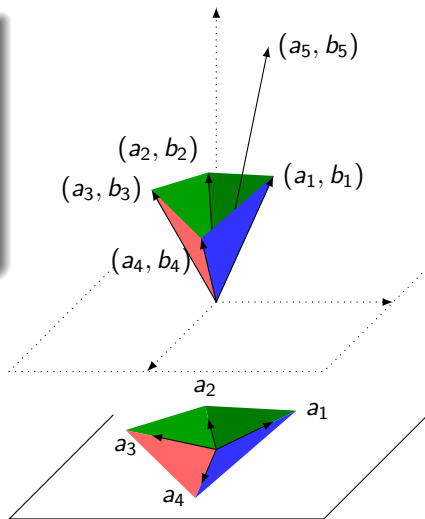
Definition (DLRS10)

$$\mathcal{S}(A^\top, b) := \{I_F \mid F \in \mathcal{F}_{\text{low}}(LC_{A^\top, b})\}$$

$$LC_{A^\top, b} := \text{Cone} \left(\left(\begin{pmatrix} a_i \\ b_i \end{pmatrix} \right)_{i \in [q]} \right)$$

$$I_F := \{i \in [q] \mid (a_i, b_i) \in F\}.$$

$$\mathcal{S}(A^\top, b) = \mathcal{I}(A, b)$$



$$\mathcal{I}(W^\top, q) = \mathcal{I}_{\text{com}} \cup \{\{1, 4\}\}$$

Link with regular subdivisions

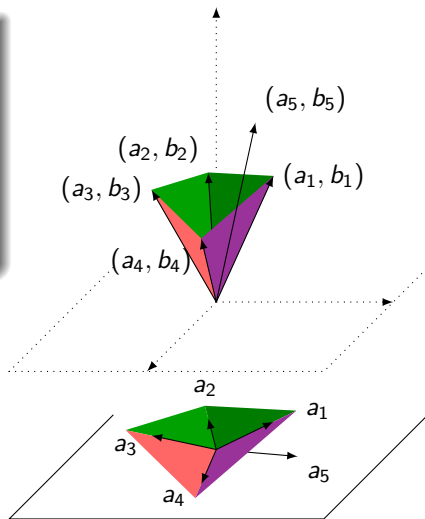
Definition (DLRS10)

$$\mathcal{S}(A^\top, b) := \{I_F \mid F \in \mathcal{F}_{\text{low}}(LC_{A^\top, b})\}$$

$$LC_{A^\top, b} := \text{Cone} \left(\left(\begin{pmatrix} a_i \\ b_i \end{pmatrix} \right)_{i \in [q]} \right)$$

$$I_F := \{i \in [q] \mid (a_i, b_i) \in F\}.$$

$$\mathcal{S}(A^\top, b) = \mathcal{I}(A, b)$$



$$\mathcal{I}(W^\top, q) = \mathcal{I}_{\text{com}} \cup \{\{1, 4, 5\}\}$$

Contents

5 Explicit formulas for general distributions

6 Details on GAPM

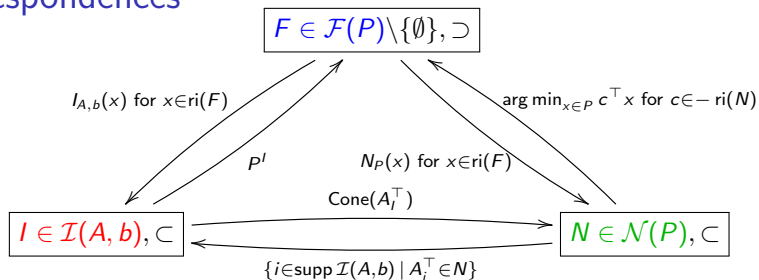
- Recalls on APM
- A novel APM algorithm
- Extension of GAPM to general costs

7 Nested fiber polyhedra

8 Polyhedral toolbox for stochastic optimizers

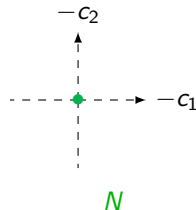
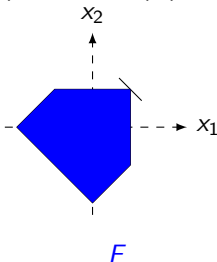
- Active constraints
- Link with regular subdivisions
- Correspondences for parametric linear programming
- Correspondences for 2SLP

Correspondences

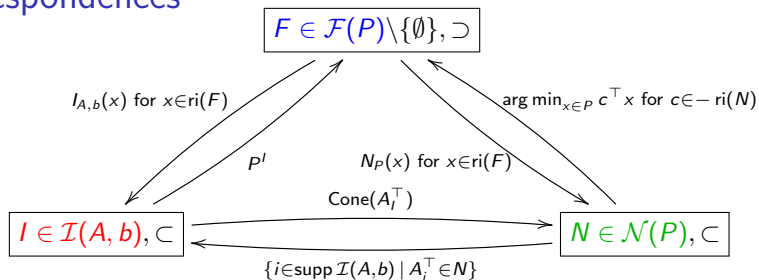


$$\text{rg}(A_I) = n - \dim(F) = \dim(N)$$

$$I = \emptyset$$

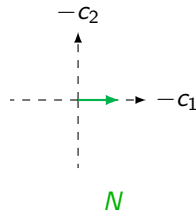
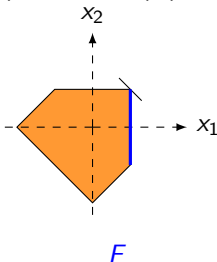


Correspondences

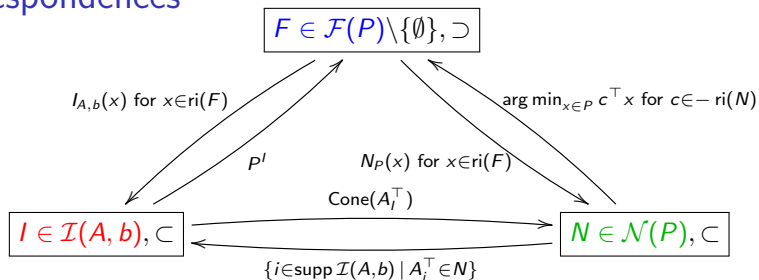


$$\text{rg}(A_I) = n - \dim(F) = \dim(N)$$

$$I = \{5\}$$

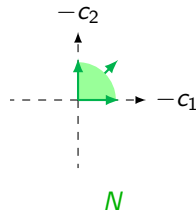
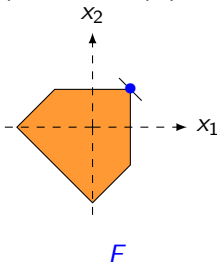


Correspondences

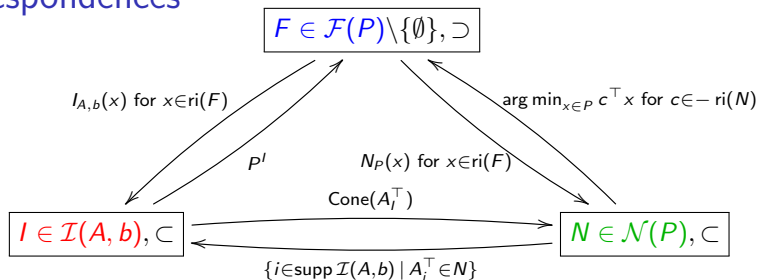


$$\text{rg}(A_I) = n - \dim(F) = \dim(N)$$

$$I = \{1, 5, 6\}$$

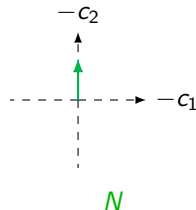
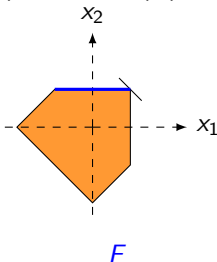


Correspondences

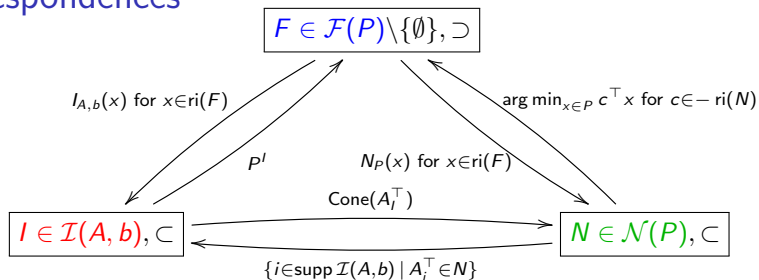


$$\text{rg}(A_I) = n - \dim(F) = \dim(N)$$

$$I = \{6\}$$

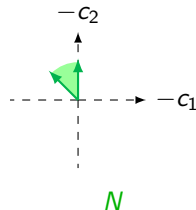
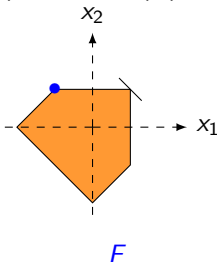


Correspondences

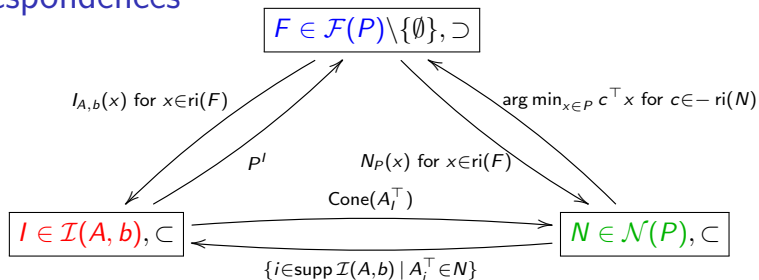


$$\text{rg}(A_I) = n - \dim(F) = \dim(N)$$

$$I = \{4, 6\}$$

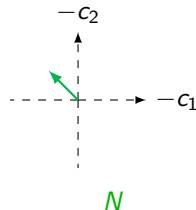
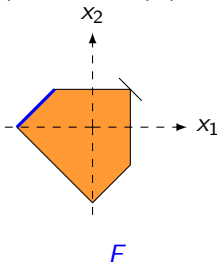


Correspondences

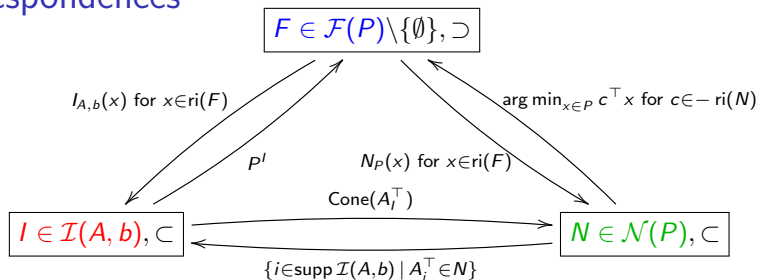


$$\text{rg}(A_I) = n - \dim(F) = \dim(N)$$

$$I = \{4\}$$

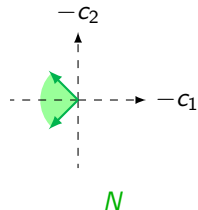
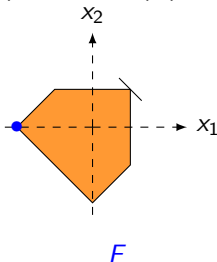


Correspondences

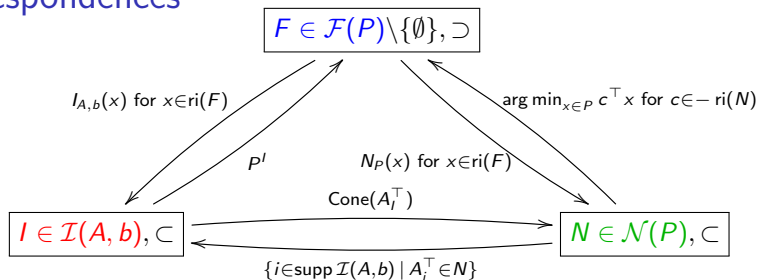


$$\text{rg}(A_I) = n - \dim(F) = \dim(N)$$

$$I = \{3, 4\}$$

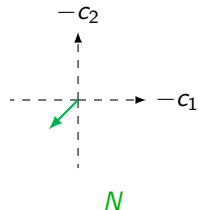
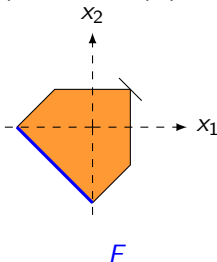


Correspondences

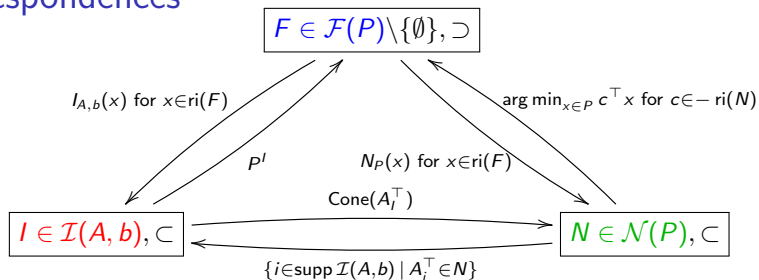


$$\text{rg}(A_I) = n - \dim(F) = \dim(N)$$

$$I = \{3\}$$

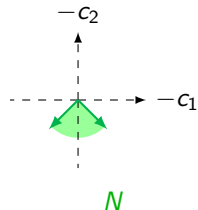
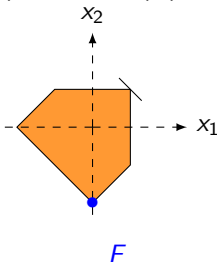


Correspondences

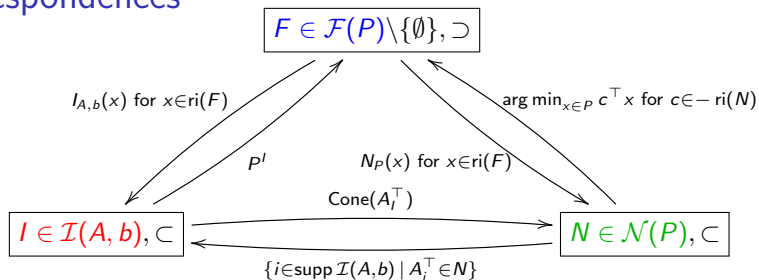


$$\text{rg}(A_I) = n - \dim(F) = \dim(N)$$

$$I = \{2, 3\}$$

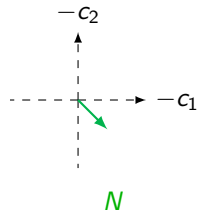
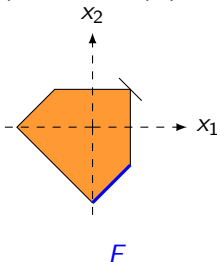


Correspondences

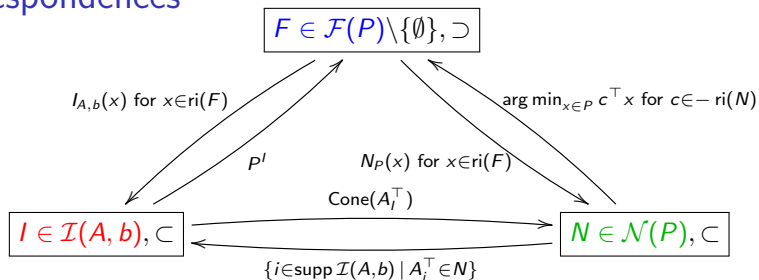


$$\text{rg}(A_I) = n - \dim(F) = \dim(N)$$

$$I = \{2\}$$

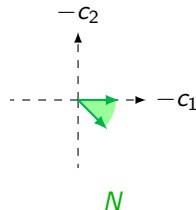
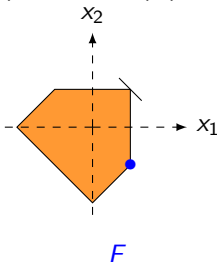


Correspondences

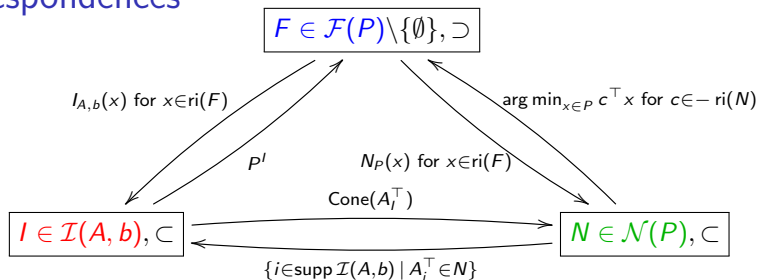


$$\text{rg}(A_I) = n - \dim(F) = \dim(N)$$

$$I = \{2, 5\}$$

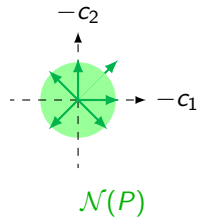
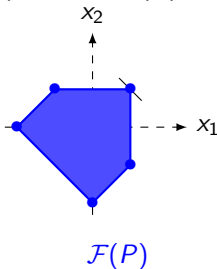


Correspondences

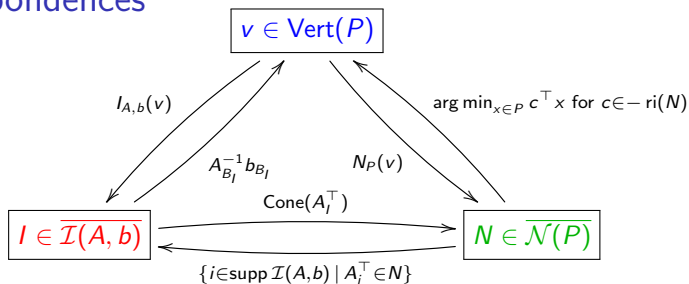


$$\text{rg}(A_I) = n - \dim(F) = \dim(N)$$

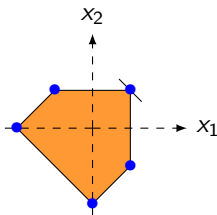
$$\mathcal{I}(A, b) = \{\emptyset, 5, 156, 6, 46, 4, 34, 3, 23, 2, 25\}$$



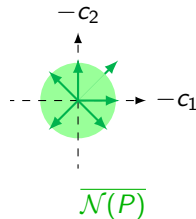
Correspondences



$$\overline{\mathcal{I}(A, b)} = \{156, 46, 34, 23, 25\}$$



$\text{Vert}(P)$



Contents

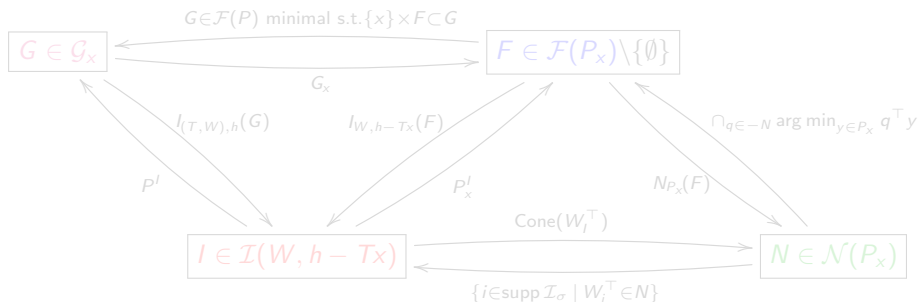
- 5 Explicit formulas for general distributions
- 6 Details on GAPM
 - Recalls on APM
 - A novel APM algorithm
 - Extension of GAPM to general costs
- 7 Nested fiber polyhedra
- 8 Polyhedral toolbox for stochastic optimizers**
 - Active constraints
 - Link with regular subdivisions
 - Correspondences for parametric linear programming
 - Correspondences for 2SLP**

Proof of normal equivalence

$$\mathcal{G}_x := \{G \in \mathcal{F}(P) \mid x \in \text{ri}(\pi(G))\}$$

Let $\sigma \in \mathcal{C}(P, \pi)$, for all $x, x' \in \text{ri}(\sigma)$, we have

$$\mathcal{G}_\sigma := \mathcal{G}_x = \mathcal{G}_{x'}$$



By the correspondences,

$$\mathcal{I}_\sigma := \mathcal{I}(W, h - Tx) = \mathcal{I}(W, h - Tx')$$

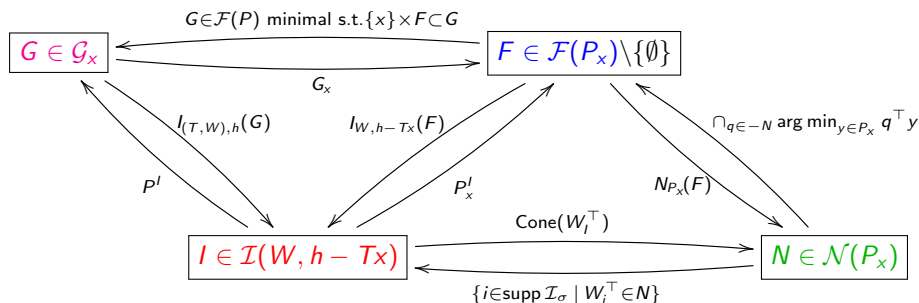
$$\mathcal{N}_\sigma := \mathcal{N}(P_x) = \mathcal{N}(P_{x'})$$

Proof of normal equivalence

$$\mathcal{G}_x := \{G \in \mathcal{F}(P) \mid x \in \text{ri}(\pi(G))\}$$

Let $\sigma \in \mathcal{C}(P, \pi)$, for all $x, x' \in \text{ri}(\sigma)$, we have

$$\mathcal{G}_\sigma := \mathcal{G}_x = \mathcal{G}_{x'}$$

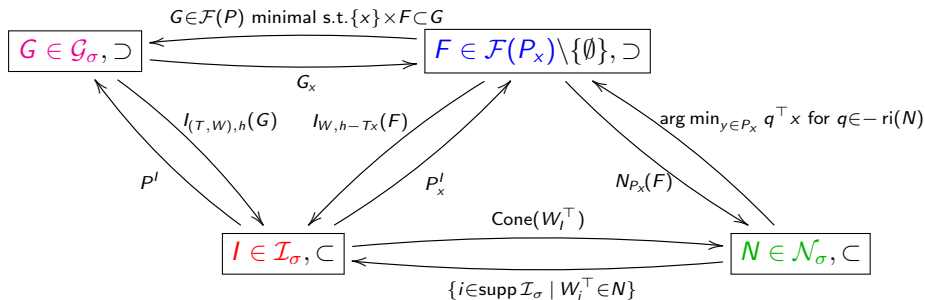


By the correspondences,

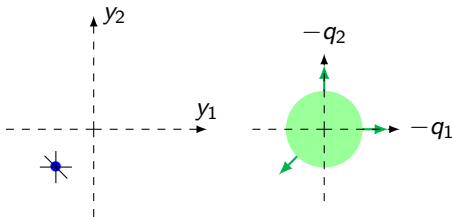
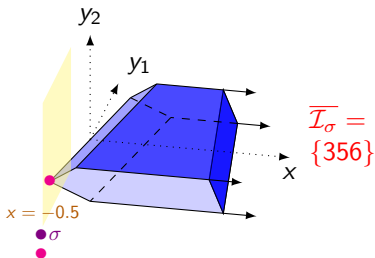
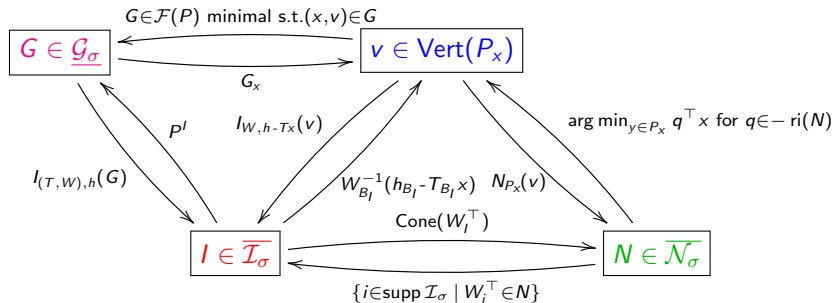
$$\mathcal{I}_\sigma := \mathcal{I}(W, h - T_x) = \mathcal{I}(W, h - T_{x'})$$

$$\mathcal{N}_\sigma := \mathcal{N}(P_x) = \mathcal{N}(P_{x'})$$

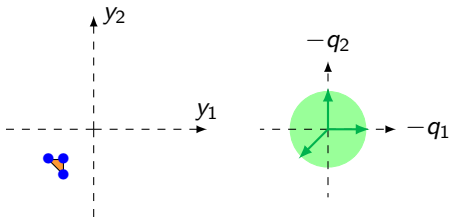
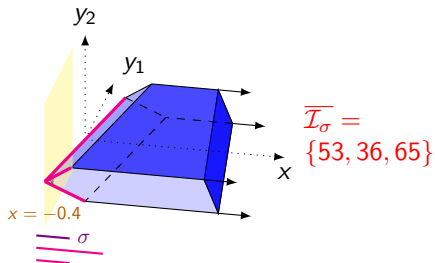
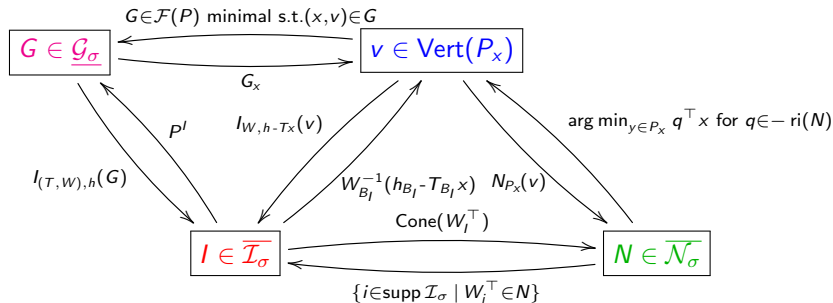
Correspondences



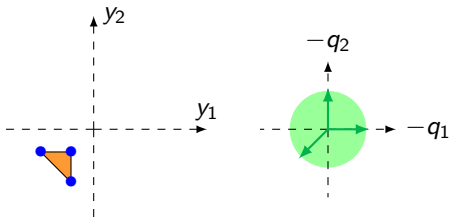
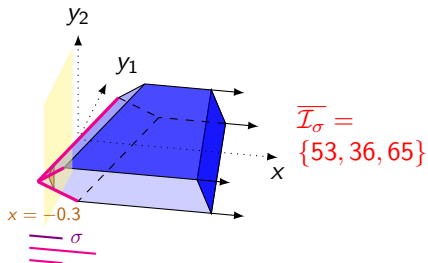
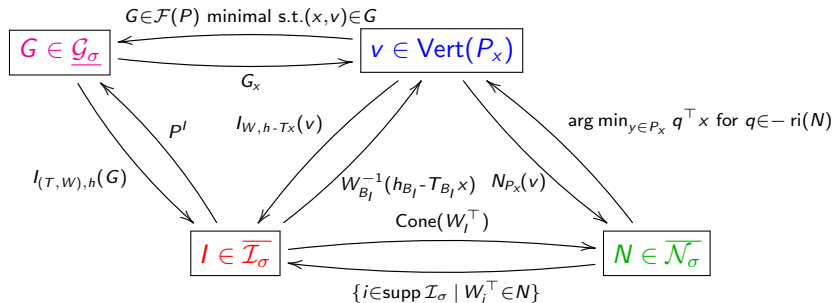
Correspondences



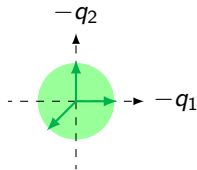
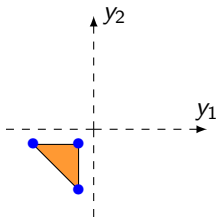
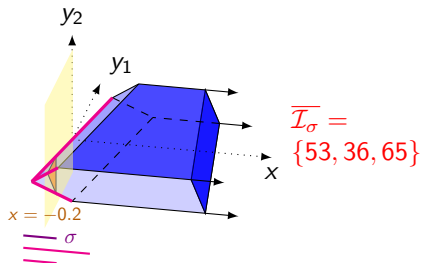
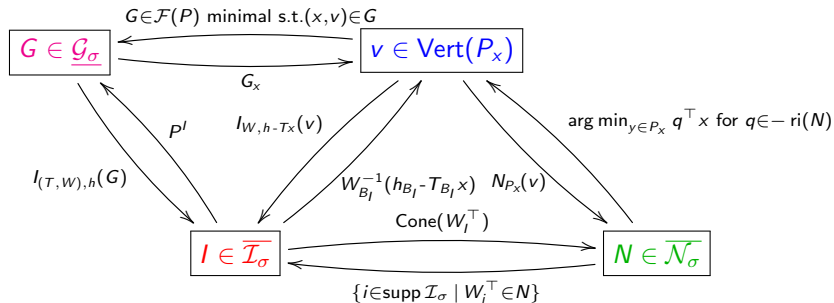
Correspondences



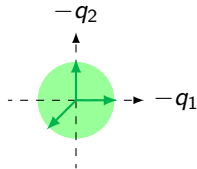
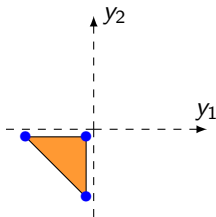
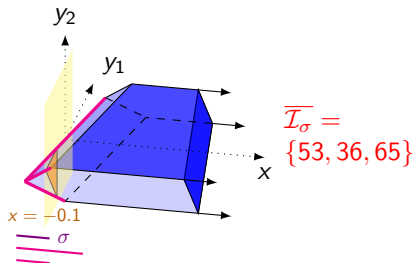
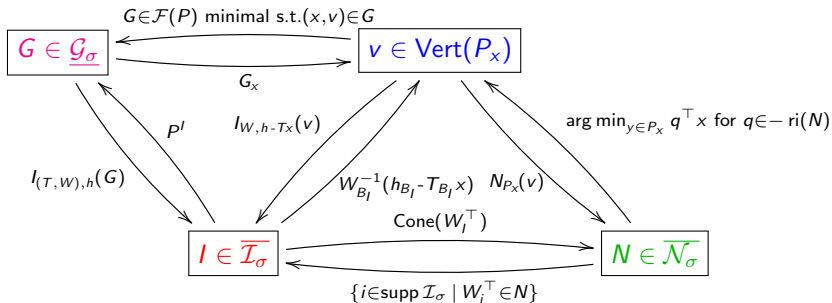
Correspondences



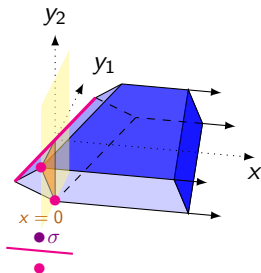
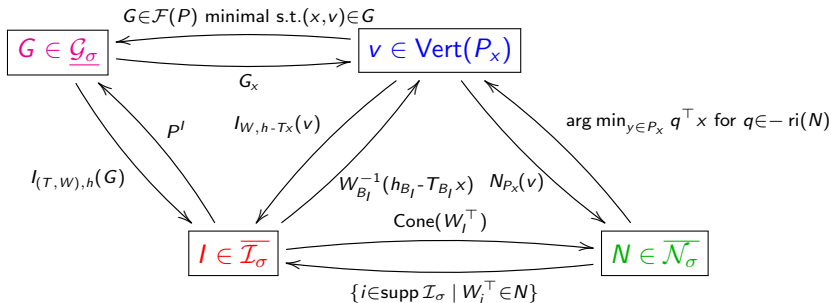
Correspondences



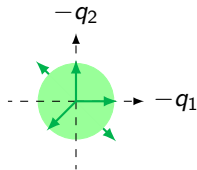
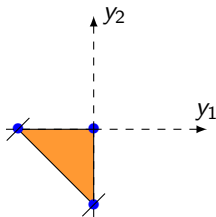
Correspondences



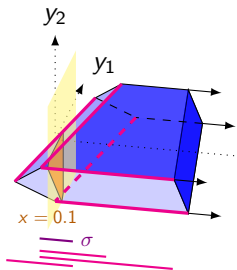
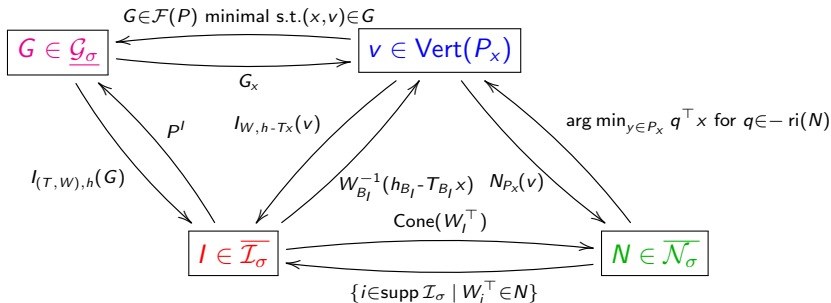
Correspondences



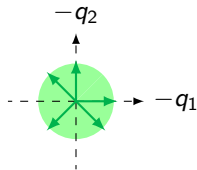
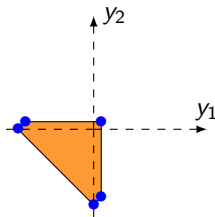
$$\overline{\mathcal{I}}_\sigma = \{523, 346, 65\}$$



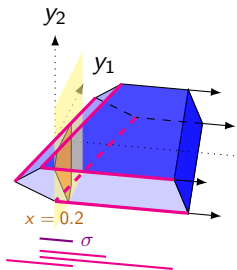
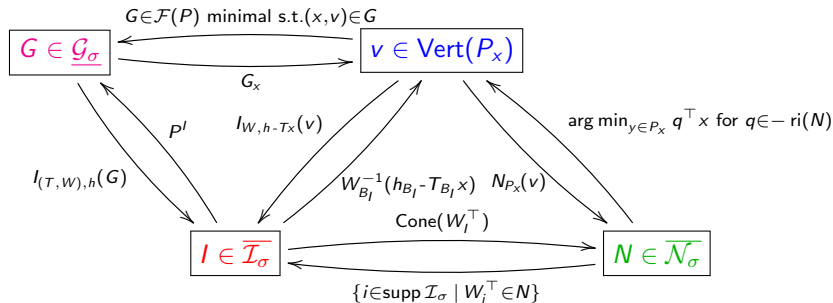
Correspondences



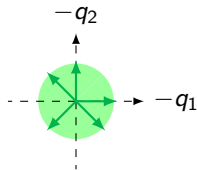
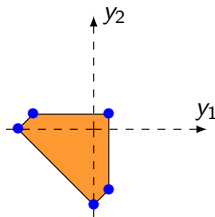
$\overline{\mathcal{I}}_\sigma =$
 $\{52, 23, 34,$
 $46, 65\}$



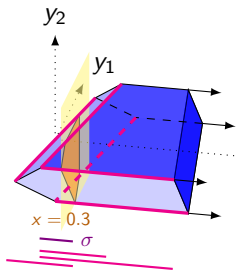
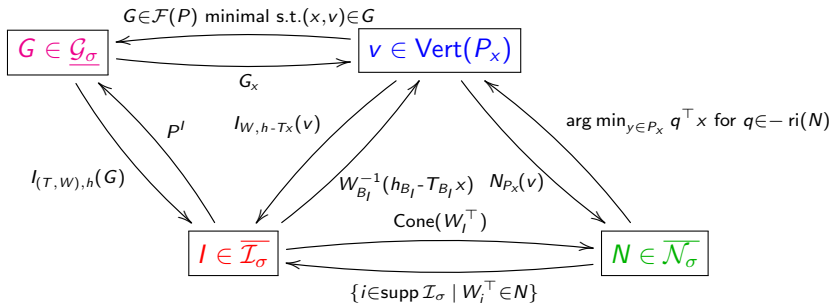
Correspondences



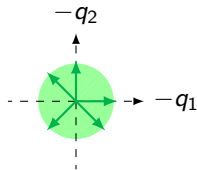
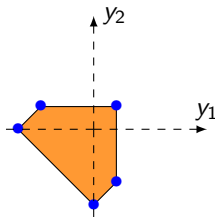
$\overline{\mathcal{I}}_\sigma =$
 $\{52, 23, 34,$
 $46, 65\}$



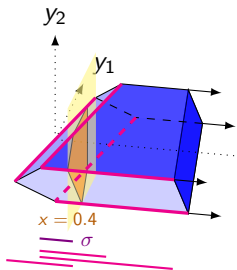
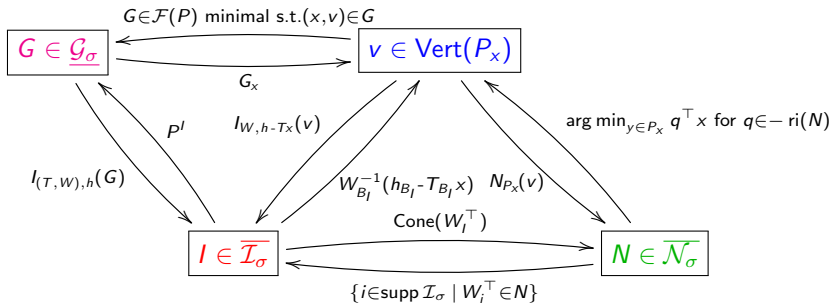
Correspondences



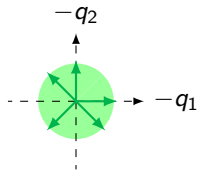
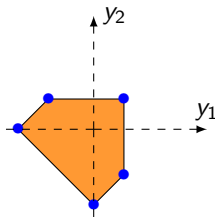
$\overline{\mathcal{I}}_\sigma =$
 $\{52, 23, 34,$
 $46, 65\}$



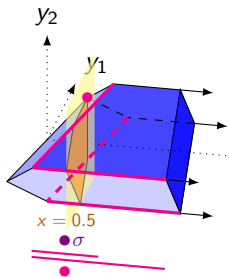
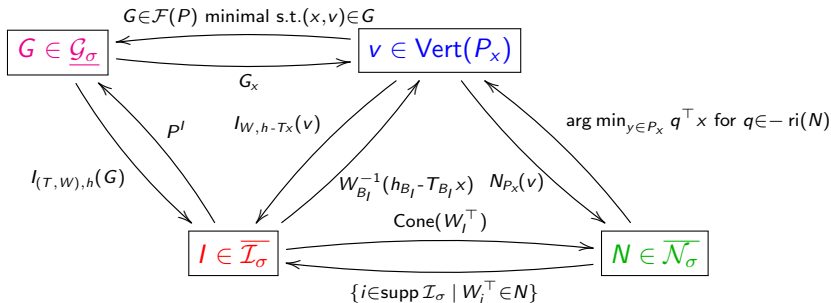
Correspondences



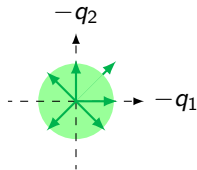
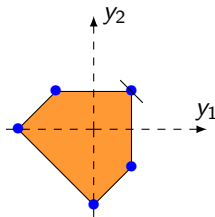
$\overline{\mathcal{I}}_\sigma =$
 $\{52, 23, 34,$
 $46, 65\}$



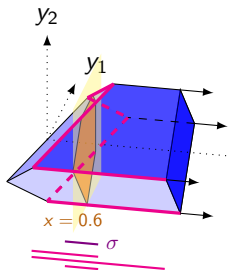
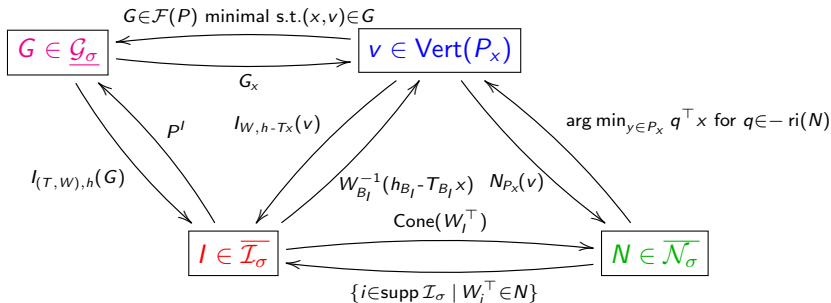
Correspondences



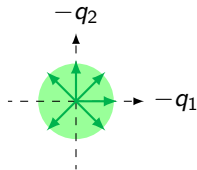
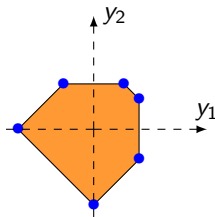
$\overline{\mathcal{I}}_\sigma =$
 $\{52, 23, 34,$
 $46, 615\}$



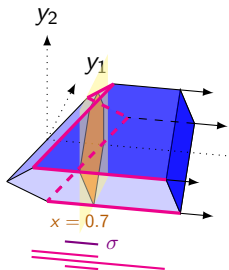
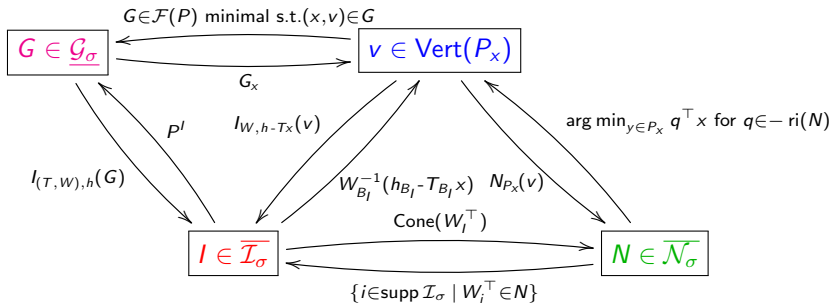
Correspondences



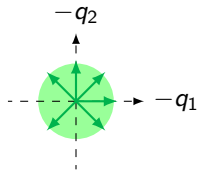
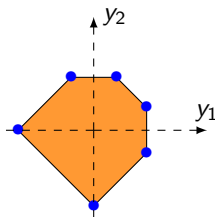
$\overline{\mathcal{I}}_\sigma =$
 $\{52, 23, 34,$
 $46, 61, 15\}$



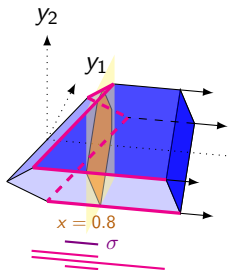
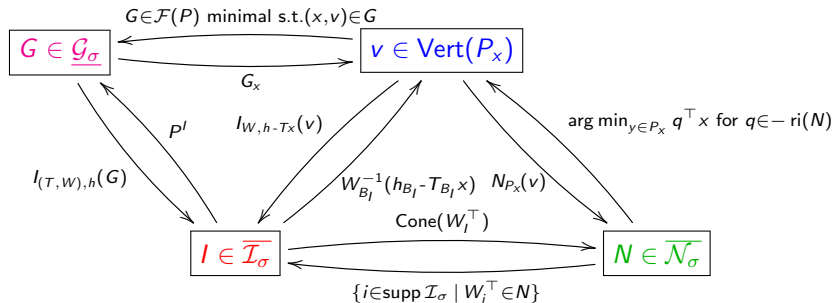
Correspondences



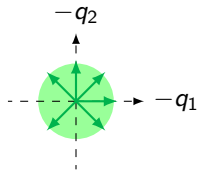
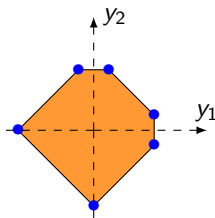
$\overline{\mathcal{I}}_\sigma =$
 $\{52, 23, 34,$
 $46, 61, 15\}$



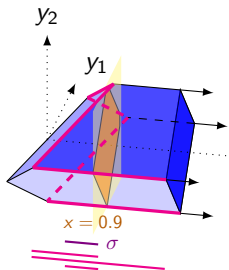
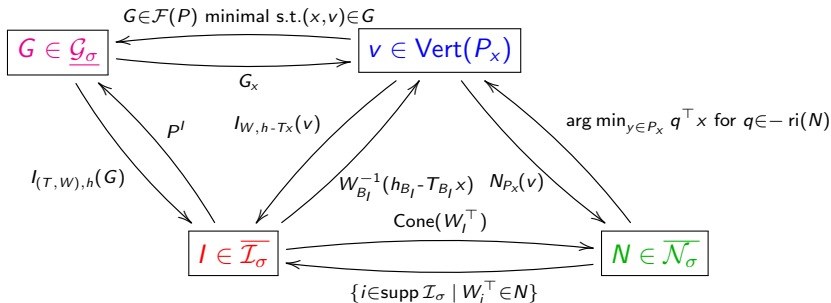
Correspondences



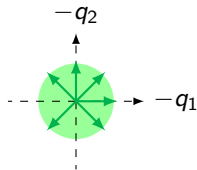
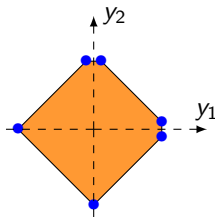
$\overline{\mathcal{I}}_\sigma =$
 $\{52, 23, 34,$
 $46, 61, 15\}$



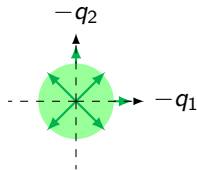
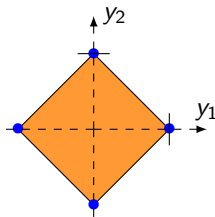
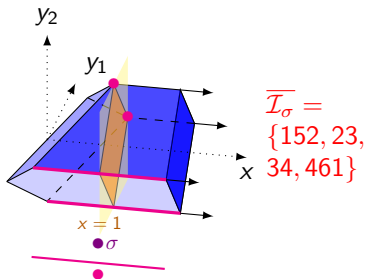
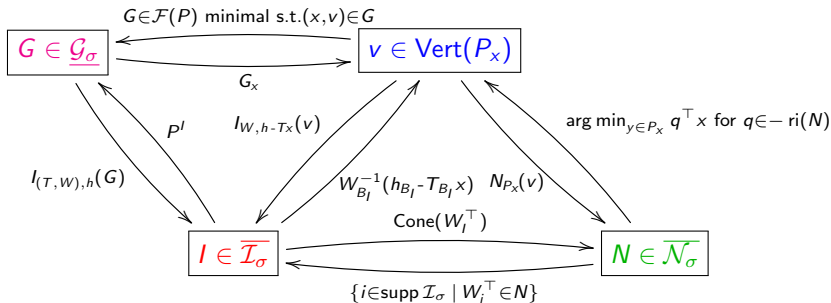
Correspondences



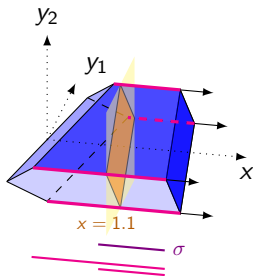
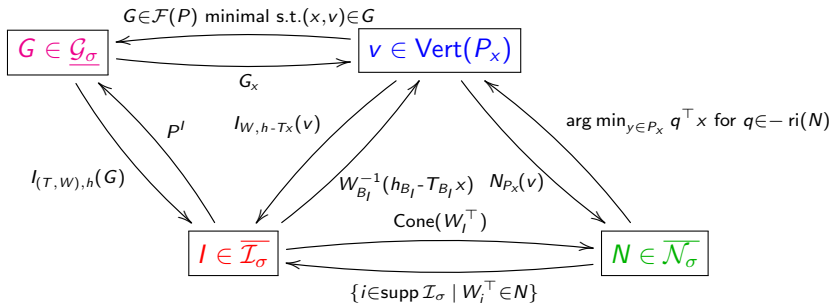
$\overline{\mathcal{I}}_\sigma =$
 $\{52, 23, 34,$
 $46, 61, 15\}$



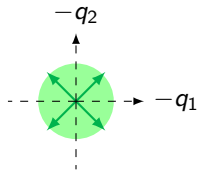
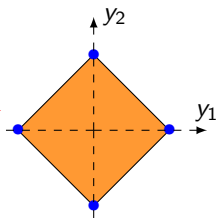
Correspondences



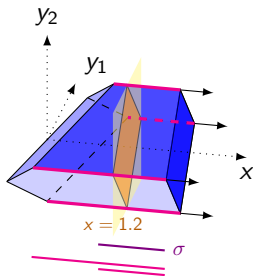
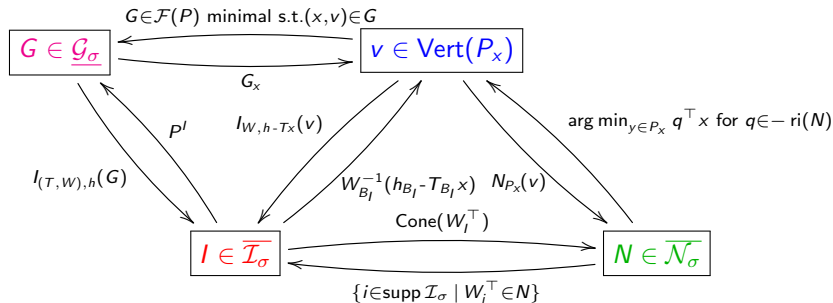
Correspondences



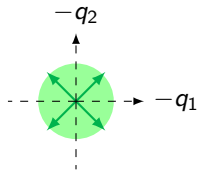
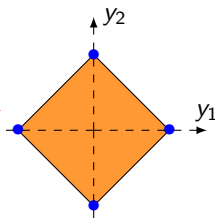
$$\overline{\mathcal{I}}_\sigma = \{12, 23, 34, 41\}$$



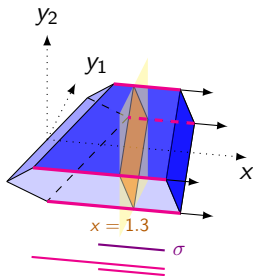
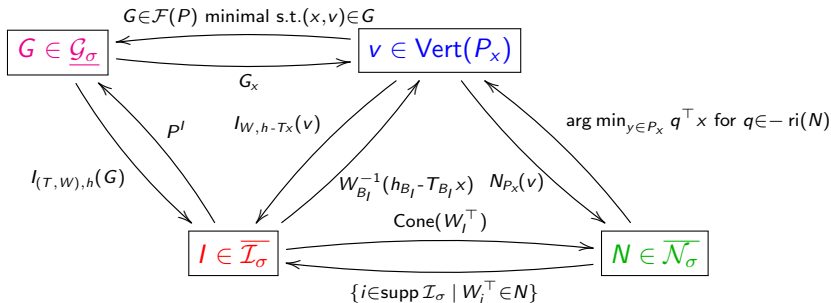
Correspondences



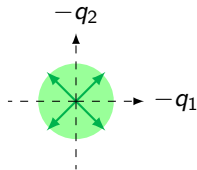
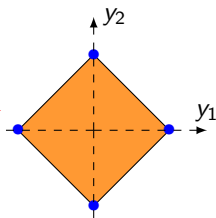
$$\overline{\mathcal{I}}_\sigma = \{12, 23, 34, 41\}$$



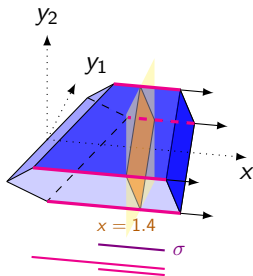
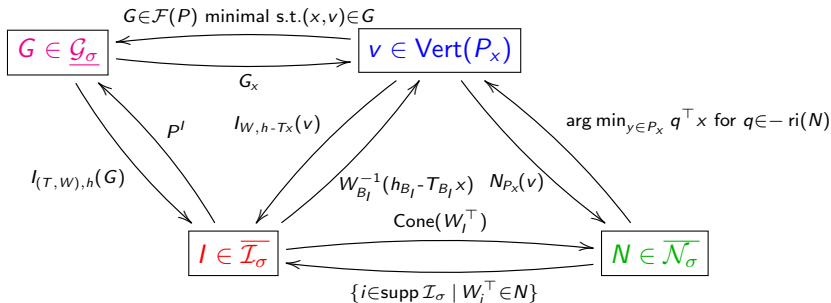
Correspondences



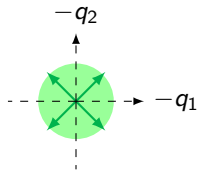
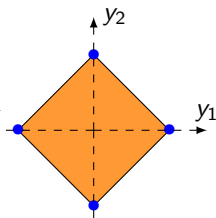
$$\overline{\mathcal{I}}_\sigma = \{12, 23, 34, 41\}$$



Correspondences



$$\overline{\mathcal{I}}_\sigma = \{12, 23, 34, 41\}$$



H-representation of projection of faces

Let $I \in \mathcal{I}((T, W), h)$ be a set of indices

$$x \in \pi(P^I) \iff \begin{cases} \exists y \in \mathbb{R}^m, & (x, y) \in P^I \end{cases}$$

H-representation of projection of faces

Let $I \in \mathcal{I}((T, W), h)$ be a set of indices

$$x \in \pi(P^I) \iff \begin{cases} \exists y \in \mathbb{R}^m, & T_I x + W_I y = h_I \\ \forall j \in [q] \setminus I, & T_j x + W_j y \leq h_j \end{cases} \iff I \in \mathcal{I}(W, h - Tx)$$

H-representation of projection of faces

Let $I \in \mathcal{I}((T, W), h)$ be a set of indices

$$x \in \text{ri } \pi(P^I) \iff \begin{cases} \exists y \in \mathbb{R}^m, & T_I x + W_I y = h_I \\ \forall j \in [q] \setminus I, & T_j x + W_j y < h_j \end{cases} \iff I \in \mathcal{I}(W, h - Tx)$$

H-representation of projection of faces

Let $I \in \mathcal{I}((T, W), h)$ be a set of indices from which we can extract a basis (i.e. $\text{rg}(W_I^\top) = m$) and let B such a basis

$$x \in \text{ri } \pi(P^I) \iff \begin{cases} \exists y \in \mathbb{R}^m, & T_B x + W_B y = h_B \\ \forall i \in I \setminus B, & T_i x + W_i y = h_i \\ \forall j \in [q] \setminus I, & T_j x + W_j y < h_j \end{cases} \iff I \in \mathcal{I}(W, h - Tx)$$

H-representation of projection of faces

Let $I \in \mathcal{I}((T, W), h)$ be a set of indices from which we can extract a basis (i.e. $\text{rg}(W_I^\top) = m$) and let B such a basis

$$x \in \text{ri } \pi(P^I) \iff \begin{cases} \exists y \in \mathbb{R}^m, & y = W_B^{-1}(h_B - T_B x) \\ \forall i \in I \setminus B, & T_i x + W_i y = h_i \\ \forall j \in [q] \setminus I, & T_j x + W_j y < h_j \end{cases} \iff I \in \mathcal{I}(W, h - T x)$$

H-representation of projection of faces

Let $I \in \mathcal{I}((T, W), h)$ be a set of indices from which we can extract a basis (i.e. $\text{rg}(W_I^\top) = m$) and let B such a basis

$$x \in \text{ri } \pi(P^I) \iff \begin{cases} \forall i \in I \setminus B, & T_i x + W_i W_B^{-1}(h_B - T_B x) = h_i \\ \forall j \in [q] \setminus I, & T_j x + W_j W_B^{-1}(h_B - T_B x) < h_j \end{cases}$$

H-representation of projection of faces

Let $I \in \mathcal{I}((T, W), h)$ be a set of indices from which we can extract a basis (i.e. $\text{rg}(W_I^\top) = m$) and let B such a basis

$$x \in \text{ri}(\pi(P^I)) \iff \begin{cases} \forall i \in I \setminus B, & (v_i^B)^\top x = u_i^B \\ \forall j \in [q] \setminus I, & (v_j^B)^\top x < u_j^B \end{cases} \iff I \in \mathcal{I}(W, h - Tx)$$

where

$$v_i^B := (T_i - W_i W_B^{-1} T_B)^\top$$

$$u_i^B := h_i - W_i W_B^{-1} h_B$$

H-representation of chambers

Let $\sigma \in \mathcal{C}(P, \pi)$

$$x \in \bigcap_{I \in \overline{\mathcal{I}_\sigma}} \text{ri}(\pi(P^I)) \iff \begin{cases} \forall I \in \overline{\mathcal{I}_\sigma}, \\ \forall i \in I \setminus B_I, \quad (v_i^{B_I})^\top x = u_i^{B_I} \\ \forall j \in [q] \setminus I, \quad (v_j^{B_I})^\top x < u_j^{B_I} \end{cases} \iff \mathcal{I}(W, h - Tx) = \mathcal{I}_\sigma$$

where

$$\begin{aligned} v_i^B &:= (T_i - W_i W_B^{-1} T_B)^\top \\ u_i^B &:= h_i - W_i W_B^{-1} h_B \end{aligned}$$

with B_I basis $\subset I$ and

$$\begin{aligned} \mathcal{G}_\sigma &:= \{F \in \mathcal{F}(P) \mid \text{ri}(\sigma) \subset \text{ri}(\pi(F))\} \\ \mathcal{I}_\sigma &:= \{I \in \mathcal{I}((T, W), h) \mid \text{ri}(\sigma) \subset \text{ri}(\pi(P^I))\} \end{aligned}$$

We have $\sigma = \bigcap_{G \in \mathcal{G}_\sigma} \pi(G) = \bigcap_{I \in \mathcal{I}_\sigma} \pi(P^I)$

H-representation of chambers

Let $\sigma \in \mathcal{C}(P, \pi)$

$$x \in \text{ri}(\sigma) \iff \begin{cases} \forall I \in \overline{\mathcal{I}_\sigma}, \\ \forall i \in I \setminus B_I, \quad (v_i^{B_I})^\top x = u_i^{B_I} \\ \forall j \in [q] \setminus I, \quad (v_j^{B_I})^\top x < u_j^{B_I} \end{cases} \iff \mathcal{I}(W, h - Tx) = \mathcal{I}_\sigma$$

where

$$\begin{aligned} v_i^B &:= (T_i - W_i W_B^{-1} T_B)^\top \\ u_i^B &:= h_i - W_i W_B^{-1} h_B \end{aligned}$$

with B_I basis $\subset I$ and

$$\begin{aligned} \mathcal{G}_\sigma &:= \{F \in \mathcal{F}(P) \mid \text{ri}(\sigma) \subset \text{ri}(\pi(F))\} \\ \mathcal{I}_\sigma &:= \{I \in \mathcal{I}((T, W), h) \mid \text{ri}(\sigma) \subset \text{ri}(\pi(P^I))\} \end{aligned}$$

We have $\sigma = \cap_{G \in \underline{\mathcal{G}}_\sigma} \pi(G) = \cap_{I \in \overline{\mathcal{I}}_\sigma} \pi(P^I)$