Stochastic Shortest Path Problem with Delay Excess Penalty

Stefanie Kosuch Abdel Lisser

Laboratoire de Recherche en Informatique (LRI), Université Paris Sud / XI (Orsay)

ROADEF 2010
24 - 26 février 2010
Toulouse

1. Présentation du problème
   - Formulation mathématique
2. Méthode de résolution
   - Gradient projeté et stratégie d'ensemble actif (cas déterministe)
   - Gradient projeté et stratégie d'ensemble actif (cas stochastique)

Outline

Plus court chemin restreint stochastique

Plus court chemin restreint stochastique

- $G = (V, A)$: graphe orienté simple (sans cycle orienté)

Plus court chemin restreint stochastique

- $G = (V, A)$: graphe orienté simple (sans cycle orienté)
- s, t : deux sommets fixés (source/puits)

Plus court chemin restreint stochastique

- $G = (V, A)$: graphe orienté simple (sans cycle orienté)
- s, t : deux sommets fixés (source/puits)
- c_a : coût (déterministe) de l'arc $a \in A$

Plus court chemin restreint stochastique

- $G = (V, A)$: graphe orienté simple (sans cycle orienté)
- s, t : deux sommets fixés (source/puits)
- c_a : coût (déterministe) de l'arc $a \in A$
- δ_a : variable aléatoire représentant le délai sur l'arc $a \in A$

Plus court chemin restreint stochastique

- $G = (V, A)$: graphe orienté simple (sans cycle orienté)
- s, t : deux sommets fixés (source/puits)
- c_a : coût (déterministe) de l'arc $a \in A$
- δ_a : variable aléatoire représentant le délai sur l'arc $a \in A$
- D : maximum délai permis sans pénalité

Plus court chemin restreint stochastique

- $G = (V, A)$: graphe orienté simple (sans cycle orienté)
- s, t : deux sommets fixés (source/puits)
- c_a : coût (déterministe) de l'arc $a \in A$
- δ_a : variable aléatoire représentant le délai sur l'arc $a \in A$
- D : maximum délai permis sans pénalité
- d : penalité en cas de dépassement de D (pénalité / unité de temps)

Plus court chemin restreint stochastique

- $G = (V, A)$: graphe orienté simple (sans cycle orienté)
- s, t : deux sommets fixés (source/puits)
- c_a : coût (déterministe) de l'arc $a \in A$
- δ_a : variable aléatoire représentant le délai sur l'arc $a \in A$
- D : maximum délai permis sans pénalité
- d : penalité en cas de dépassement de D (pénalité / unité de temps)
- Objectif : **Touver un chemin de s à t qui minimise l'espérance des coûts totals**

Plus court chemin stochastique avec délai (SSPD)

$$\min_{x \in \{0,1\}^{|A|}} \sum_{a \in A} c_a x_a + d \cdot \mathbb{E}[[\sum_{a \in A} \delta(a) x_a - D]^+]$$

Plus court chemin stochastique avec délai (SSPD)

$$\min_{x \in \{0,1\}^{|A|}} \sum_{a \in A} c_a x_a + d \cdot \mathbb{E}[[\sum_{a \in A} \delta(a) x_a - D]^+]$$

- $[x]^+ := \max(0, x) = x \cdot \mathbb{1}_{\mathbb{R}^+}(x)$

Plus court chemin stochastique avec délai (SSPD)

$$\min_{x \in \{0,1\}^{|A|}} \sum_{a \in A} c_a x_a + d \cdot \mathbb{E}[[\sum_{a \in A} \delta(a) x_a - D]^+]$$
$$\text{s.t.} \quad Mx = b$$

- $[x]^+ := \max(0, x) = x \cdot \mathbb{1}_{\mathbb{R}^+}(x)$

Référence

B. Verweij, S. Ahmed, A. J. Kleywegt, G. Nemhauser, A. Shapiro :

The Sample Average Approximation Method Applied to Stochastic Routing Problems : A Computational Study (2003)

$\rightarrow$ **NP-difficile**

Référence

B. Verweij, S. Ahmed, A. J. Kleywegt, G. Nemhauser, A. Shapiro :
The Sample Average Approximation Method Applied to Stochastic Routing Problems : A Computational Study (2003)
$\rightarrow$ **NP-difficile**

Hypothèse

Référence

B. Verweij, S. Ahmed, A. J. Kleywegt, G. Nemhauser, A. Shapiro :
The Sample Average Approximation Method Applied to Stochastic Routing Problems : A Computational Study (2003)
$\rightarrow$ **NP-difficile**

Hypothèse

- Délais suivent distribution continue

Référence

B. Verweij, S. Ahmed, A. J. Kleywegt, G. Nemhauser, A. Shapiro :
The Sample Average Approximation Method Applied to Stochastic Routing Problems : A Computational Study (2003)
$\rightarrow$ **NP-difficile**

Hypothèse

- Délais suivent distribution continue
- Délais sont distibués normallement et indépendemment

Référence

B. Verweij, S. Ahmed, A. J. Kleywegt, G. Nemhauser, A. Shapiro :
The Sample Average Approximation Method Applied to Stochastic Routing Problems : A Computational Study (2003)
$\rightarrow$ **NP-difficile**

Hypothèse

- Délais suivent distribution continue
- Délais sont distibués normallement et indépendemment
- Moyenne μ_a et écart-type σ_a de chaque arc a sont supposé connus

Plus court chemin stochastique avec délai (SSPD)

Reformulation déterministe équivalent

$$\min_{x \in \{0,1\}^{|A|}} \sum_{a \in A} c_a x_a + d \cdot \left[\hat{\sigma} \cdot f\left(\frac{D - \hat{\mu}}{\hat{\sigma}} \right) - (D - \hat{\mu}) \cdot \left[1 - F\left(\frac{D - \hat{\mu}}{\hat{\sigma}} \right) \right] \right]$$

$$\text{s.t.} \quad Mx = b$$

Plus court chemin stochastique avec délai (SSPD)

Reformulation déterministe équivalent

$$\min_{x\in\{0,1\}^{|A|}} \sum_{a\in A} c_a x_a + d \cdot \left[\hat{\sigma}\cdot f\left(\frac{D-\hat{\mu}}{\hat{\sigma}}\right) - (D-\hat{\mu})\cdot\left[1 - F\left(\frac{D-\hat{\mu}}{\hat{\sigma}}\right)\right]\right]$$

$$\text{s.t.} \quad Mx = b$$

- $\hat{\mu} := \sum_{a\in A} \mu_a x_a$
- $\hat{\sigma} := \sqrt{\sum_{a\in A} \sigma_a^2 x_a^2}$

Plus court chemin stochastique avec délai (SSPD)

Reformulation déterministe équivalent

$$\min_{x\in\{0,1\}^{|A|}} \sum_{a\in A} c_a x_a + d \cdot \left[\hat{\sigma} \cdot f\left(\frac{D-\hat{\mu}}{\hat{\sigma}}\right) - (D-\hat{\mu}) \cdot \left[1 - F\left(\frac{D-\hat{\mu}}{\hat{\sigma}}\right)\right]\right]$$

$$\text{s.t.} \quad Mx = b$$

- $\hat{\mu} := \sum_{a\in A} \mu_a x_a$
- $\hat{\sigma} := \sqrt{\sum_{a\in A} \sigma_a^2 x_a^2}$
- f : fonction de densité de la loi normale centrée réduite

Plus court chemin stochastique avec délai (SSPD)

Reformulation déterministe équivalent

$$\min_{x\in\{0,1\}^{|A|}} \sum_{a\in A} c_a x_a + d \cdot \left[\hat{\sigma}\cdot f\left(\frac{D-\hat{\mu}}{\hat{\sigma}}\right) - (D-\hat{\mu})\cdot\left[1 - F\left(\frac{D-\hat{\mu}}{\hat{\sigma}}\right)\right]\right]$$

$$\text{s.t.} \quad Mx = b$$

- $\hat{\mu} := \sum_{a\in A} \mu_a x_a$
- $\hat{\sigma} := \sqrt{\sum_{a\in A} \sigma_a^2 x_a^2}$
- f : fonction de densité de la loi normale centrée réduite
- F : fonction de répartition de la loi normale centrée réduite

Plus court chemin stochastique avec délai (SSPD)

Reformulation déterministe équivalent

$$\min_{x\in\{0,1\}^{|A|}} \sum_{a\in A} c_a x_a + d \cdot \left[\hat{\sigma}\cdot f\left(\frac{D-\hat{\mu}}{\hat{\sigma}}\right) - (D-\hat{\mu})\cdot\left[1 - F\left(\frac{D-\hat{\mu}}{\hat{\sigma}}\right)\right]\right]$$

$$\text{s.t.} \quad Mx = b$$

- $\hat{\mu} := \sum_{a\in A} \mu_a x_a$
- $\hat{\sigma} := \sqrt{\sum_{a\in A} \sigma_a^2 x_a^2}$
- f : fonction de densité de la loi normale centrée réduite
- F : fonction de répartition de la loi normale centrée réduite

Outline

Méthode de Résolution

Méthode de Résolution

- Utiliser un algorithme **branch-and-bound** pour trouver le chemin optimal
  $\longrightarrow$ Parcours en profondeur de G

Méthode de Résolution

- Utiliser un algorithme **branch-and-bound** pour trouver le chemin optimal
  $\longrightarrow$ Parcours en profondeur de G
- Calculer bornes inférieures :
  Resoudre la **relaxation continue du** *SSPD*

Méthode de Résolution

- Utiliser un algorithme **branch-and-bound** pour trouver le chemin optimal
  $\longrightarrow$ Parcours en profondeur de G
- Calculer bornes inférieures :
  Resoudre la **relaxation continue du** *SSPD*
- Utiliser un **gradient projeté stochastique** pour trouver solution de relaxation

Méthode de Résolution

- Utiliser un algorithme **branch-and-bound** pour trouver le chemin optimal
  $\longrightarrow$ Parcours en profondeur de G
- Calculer bornes inférieures :
  Resoudre la **relaxation continue du** *SSPD*
- Utiliser un **gradient projeté stochastique** pour trouver solution de relaxation
- Gérer les inégalités $x_a \geq 0$ avec une stratégie d'**ensemble actif**

Gradient projeté

Fonction objectif : $J(x)$

Gradient projeté

Fonction objectif : $J(x)$

Basé sur l'algorithme **descente de gradient** :

Gradient projeté

Fonction objectif : $J(x)$

Basé sur l'algorithme **descente de gradient** :

$$x^k = x^{k-1} - \alpha \nabla_x J(x^{k-1})$$

Gradient projeté

Fonction objectif : $J(x)$

Basé sur l'algorithme **descente de gradient** :

$$x^k = x^{k-1} - \alpha \nabla_x J(x^{k-1})$$

Gestion des **contraintes d'égalité linéaires** ($Mx = b$) :

Gradient projeté

Fonction objectif : $J(x)$

Basé sur l'algorithme **descente de gradient** :

$$x^k = x^{k-1} - \alpha \nabla_x J(x^{k-1})$$

Gestion des **contraintes d'égalité linéaires** ($Mx = b$) :

- Projeter le gradient sur le noyeau de M

Gradient projeté

Fonction objectif : $J(x)$

Basé sur l'algorithme **descente de gradient** :

$$x^k = x^{k-1} - \alpha \nabla_x J(x^{k-1})$$

Gestion des **contraintes d'égalité linéaires** ($Mx = b$) :

- Projeter le gradient sur le noyeau de M
- Matrice de projection : $P := \mathbb{I}_n - M^T(MM^T)^{-1}M$

Gradient projeté

Fonction objectif : $J(x)$

Basé sur l'algorithme **descente de gradient** :

$$x^k = x^{k-1} - \alpha \nabla_x J(x^{k-1})$$

Gestion des **contraintes d'égalité linéaires** ($Mx = b$) :

- Projeter le gradient sur le noyeau de M
- Matrice de projection : $P := \mathbb{I}_n - M^T(MM^T)^{-1}M$

$$x^{k-1}M = b \Rightarrow$$

Gradient projeté

Fonction objectif : $J(x)$

Basé sur l'algorithme **descente de gradient** :

$$x^k = x^{k-1} - \alpha \nabla_x J(x^{k-1})$$

Gestion des **contraintes d'égalité linéaires** ($Mx = b$) :

- Projeter le gradient sur le noyeau de M
- Matrice de projection : $P := \mathbb{I}_n - M^T(MM^T)^{-1}M$

$$x^{k-1}M = b \Rightarrow$$
$$x^k M = x^{k-1}M - \alpha \cdot P \cdot \nabla_x J(x^{k-1}) \cdot M = b$$

Gestion des **inégalités de non-négativité**

Si x^k ne satisfait pas une des inégalités $x_i \geq 0$:

Gestion des **inégalités de non-négativité**

Si x^k ne satisfait pas une des inégalités $x_i \geq 0$:

- Raccourcir le pas α :

Gestion des **inégalités de non-négativité**

Si x^k ne satisfait pas une des inégalités $x_i \geq 0$:

- Raccourcir le pas α :
- $I_-^k := \{i \mid x_i^k < 0\}$

$$\overline{\alpha} = \min_{i \in I_-^k} \left\{ \frac{x_i^{k-1}}{(P \cdot \nabla_x J(x^{k-1}))_i} \right\}$$

Gestion des **inégalités de non-négativité**

Si x^k ne satisfait pas une des inégalités $x_i \geq 0$:

- Raccourcir le pas α :
- $I_-^k := \{i \mid x_i^k < 0\}$

$$\overline{\alpha} = \min_{i \in I_-^k} \left\{ \frac{x_i^{k-1}}{(P \cdot \nabla_x J(x^{k-1}))_i} \right\}$$

- Si toutes inégalités strictement satisfaites par $x^{k-1} \Rightarrow \overline{\alpha} > 0$

Gestion des **inégalités de non-négativité**

Si x^k ne satisfait pas une des inégalités $x_i \geq 0$:

- Raccourcir le pas α :
- $I_-^k := \{i \mid x_i^k < 0\}$

$$\overline{\alpha} = \min_{i \in I_-^k} \left\{ \frac{x_i^{k-1}}{(P \cdot \nabla_x J(x^{k-1}))_i} \right\}$$

- Si toutes inégalités strictement satisfaites par $x^{k-1} \Rightarrow \overline{\alpha} > 0$
- Sinon...

Stratégie de l'ensemble actif

Si x^{k-1} satisfait une ou plusieurs inégalités avec égalité

Stratégie de l'ensemble actif

Si x^{k-1} satisfait une ou plusieurs inégalités avec égalité

- Transformer inégalités en égalités
  $\rightarrow$ **ensemble actif** $\mathcal{A}$

Stratégie de l'ensemble actif

Si x^{k-1} satisfait une ou plusieurs inégalités avec égalité

- Transformer inégalités en égalités
  $\rightarrow$ **ensemble actif** $\mathcal{A}$

Stratégie de l'ensemble actif

Si x^{k-1} satisfait une ou plusieurs inégalités avec égalité

- Transformer inégalités en égalités
  $\rightarrow$ **ensemble actif** $\mathcal{A}$
  $\rightarrow M_{\mathcal{A}}$

Stratégie de l'ensemble actif

Si x^{k-1} satisfait une ou plusieurs inégalités avec égalité

- Transformer inégalités en égalités
  $\rightarrow$ **ensemble actif** $\mathcal{A}$
  $\rightarrow M_{\mathcal{A}}$
- Projeter gradient $\nabla_x J(x^{k-1})$ sur noyeau de $M_{\mathcal{A}}$

Stratégie de l'ensemble actif

Si x^{k-1} satisfait une ou plusieurs inégalités avec égalité

- Transformer inégalités en égalités
  $\rightarrow$ **ensemble actif** $\mathcal{A}$
  $\rightarrow M_{\mathcal{A}}$
- Projeter gradient $\nabla_x J(x^{k-1})$ sur noyeau de $M_{\mathcal{A}}$
- $\Longrightarrow x^k$ satisfait ces mêmes inégalités avec égalité

Stratégie de l'ensemble actif

Si x^{k-1} satisfait une ou plusieurs inégalités avec égalité

- Transformer inégalités en égalités
  $\rightarrow$ **ensemble actif** $\mathcal{A}$
  $\rightarrow M_{\mathcal{A}}$
- Projeter gradient $\nabla_x J(x^{k-1})$ sur noyeau de $M_{\mathcal{A}}$
- $\Longrightarrow x^k$ satisfait ces mêmes inégalités avec égalité
- Si x^k viole d'autres inégalités : $\overline{\alpha} > 0$

Si $P \cdot \nabla_x J(x) = 0$

Si $P \cdot \nabla_x J(x) = 0$

- Optimum "local" trouvé

Si $P \cdot \nabla_x J(x) = 0$

- Optimum "local" trouvé
- Calculer "Multiplicateurs de Lagrange" des égalités

$$\lambda = (M_{\mathcal{A}}(M_{\mathcal{A}})^T)^{-1} M_{\mathcal{A}} \nabla_x J(x)$$

Si $P \cdot \nabla_x J(x) = 0$

- Optimum "local" trouvé
- Calculer "Multiplicateurs de Lagrange" des égalités

$$\lambda = (M_{\mathcal{A}}(M_{\mathcal{A}})^T)^{-1} M_{\mathcal{A}} \nabla_x J(x)$$

- Si $\lambda_{I_{\mathcal{A}}} \geq 0$ pour l'ensemble actif : STOP.

Si $P \cdot \nabla_x J(x) = 0$

- Optimum "local" trouvé
- Calculer "Multiplicateurs de Lagrange" des égalités

$$\lambda = (M_{\mathcal{A}}(M_{\mathcal{A}})^T)^{-1} M_{\mathcal{A}} \nabla_x J(x)$$

- Si $\lambda_{I_{\mathcal{A}}} \geq 0$ pour l'ensemble actif : STOP.
- Sinon : Enléver contrainte de $\mathcal{A}$ avec λ_i corr. minimal

Gradient projeté stochastique

Fonction objectif : $J(x) = \mathbb{E}[j(x, \delta)]$

Basé sur l'algorithme **gradient stochastique** :

Gradient projeté stochastique

Fonction objectif : $J(x) = \mathbb{E}[j(x, \delta)]$

Basé sur l'algorithme **gradient stochastique** :

- Utiliser gradient de la fonction j

Gradient projeté stochastique

Fonction objectif : $J(x) = \mathbb{E}[j(x, \delta)]$

Basé sur l'algorithme **gradient stochastique** :

- Utiliser gradient de la fonction j
- Tirage indépendant de δ à chaque itération

$$x^k = x^{k-1} - \alpha \nabla_x j(x^{k-1}, \hat{\delta}^k)$$

Gradient projeté stochastique

Fonction objectif : $J(x) = \mathbb{E}[j(x, \delta)]$

Basé sur l'algorithme **gradient stochastique** :

- Utiliser gradient de la fonction j
- Tirage indépendant de δ à chaque itération

$$x^k = x^{k-1} - \alpha \nabla_x j(x^{k-1}, \hat{\delta}^k)$$

- Procédure **Monte-Carlo** pour approcher l'espérance

Gradient projeté stochastique

Fonction objectif : $J(x) = \mathbb{E}[j(x, \delta)]$

Basé sur l'algorithme **gradient stochastique** :

- Utiliser gradient de la fonction j
- Tirage indépendant de δ à chaque itération

$$x^k = x^{k-1} - \alpha \nabla_x j(x^{k-1}, \hat{\delta}^k)$$

- Procédure **Monte-Carlo** pour approcher l'espérance

Gradient projeté stochastique

Fonction objectif : $J(x) = \mathbb{E}[j(x, \delta)]$

Basé sur l'algorithme **gradient stochastique** :

- Utiliser gradient de la fonction j
- Tirage indépendant de δ à chaque itération

$$x^k = x^{k-1} - \alpha \nabla_x j(x^{k-1}, \hat{\delta}^k)$$

- Procédure **Monte-Carlo** pour approcher l'espérance

Modifications supplémentaires

Ajouter des contraintes dû à un "mauvais tirage"

Modifications supplémentaires

Ajouter des contraintes dû à un "mauvais tirage"

$\rightarrow$ Méthode randomisée pour re-enléver contraintes

Modifications supplémentaires

Ajouter des contraintes dû à un "mauvais tirage"

$\rightarrow$ Méthode randomisée pour re-enléver contraintes

Modifications supplémentaires

Ajouter des contraintes dû à un "mauvais tirage"

$\rightarrow$ Méthode randomisée pour re-enléver contraintes

Retirer une contrainte dû à un "mauvais tirage"

Modifications supplémentaires

Ajouter des contraintes dû à un "mauvais tirage"

$\rightarrow$ Méthode randomisée pour re-enléver contraintes

Retirer une contrainte dû à un "mauvais tirage"

- Faudra prob. rajouter cette contrainte qq. itérations plus tard

Modifications supplémentaires

Ajouter des contraintes dû à un "mauvais tirage"

$\rightarrow$ Méthode randomisée pour re-enléver contraintes

Retirer une contrainte dû à un "mauvais tirage"

- Faudra prob. rajouter cette contrainte qq. itérations plus tard
- $\Rightarrow$ prochaine solution locale moins bonne

Modifications supplémentaires

Ajouter des contraintes dû à un "mauvais tirage"

$\rightarrow$ Méthode randomisée pour re-enléver contraintes

Retirer une contrainte dû à un "mauvais tirage"

- Faudra prob. rajouter cette contrainte qq. itérations plus tard
- $\Rightarrow$ prochaine solution locale moins bonne
- Deux possibilités :

Modifications supplémentaires

Ajouter des contraintes dû à un "mauvais tirage"

$\rightarrow$ Méthode randomisée pour re-enléver contraintes

Retirer une contrainte dû à un "mauvais tirage"

- Faudra prob. rajouter cette contrainte qq. itérations plus tard
- $\Rightarrow$ prochaine solution locale moins bonne
- Deux possibilités :
  1. Choisir autre contrainte à enléver

Modifications supplémentaires

Ajouter des contraintes dû à un "mauvais tirage"

$\rightarrow$ Méthode randomisée pour re-enléver contraintes

Retirer une contrainte dû à un "mauvais tirage"

- Faudra prob. rajouter cette contrainte qq. itérations plus tard
- $\Rightarrow$ prochaine solution locale moins bonne
- Deux possibilités :
  1. Choisir autre contrainte à enléver
  2. Forcer contrainte à rester strictement respectée

Estimation du gradient de j

Fonction objectif du SSPD :

$J(x) = \mathbb{E}[j(x,\delta)] = \sum_{a\in A} c_a x_a + d \cdot \mathbb{E}[[\sum_{a\in A} \delta(a) x_a - D]^+]$

Estimation du gradient de j

Fonction objectif du SSPD :

$J(x) = \mathbb{E}[j(x,\delta)] = \sum_{a\in A} c_a x_a + d \cdot \mathbb{E}[[\sum_{a\in A} \delta(a) x_a - D]^+]$

$$\Rightarrow j(x,\delta) = \sum_{a\in A} c_a x_a + d \cdot [\sum_{a\in A} \delta(a) x_a - D]^+$$

Estimation du gradient de j

Fonction objectif du SSPD :

$J(x) = \mathbb{E}[j(x,\delta)] = \sum_{a\in A} c_a x_a + d \cdot \mathbb{E}[[\sum_{a\in A} \delta(a)x_a - D]^+]$

$$\Rightarrow j(x,\delta) = \sum_{a\in A} c_a x_a + d \cdot [\sum_{a\in A} \delta(a)x_a - D]^+$$

- j différentiable p.r. à x sur $\mathbb{R}^n \setminus \{x \in \mathbb{R}^n | \sum_{a\in A} \delta(a)x_a = D\}$

Estimation du gradient de j

Fonction objectif du SSPD :

$J(x) = \mathbb{E}[j(x,\delta)] = \sum_{a\in A} c_a x_a + d \cdot \mathbb{E}[[\sum_{a\in A} \delta(a)x_a - D]^+]$

$$\Rightarrow j(x,\delta) = \sum_{a\in A} c_a x_a + d \cdot [\sum_{a\in A} \delta(a)x_a - D]^+$$

- j différentiable p.r. à x sur $\mathbb{R}^n \setminus \{x \in \mathbb{R}^n | \sum_{a\in A} \delta(a)x_a = D\}$
- $\{x \in \mathbb{R}^n | \sum_{a\in A} \delta(a)x_a = D\}$ ensemble négligeable

Estimation du gradient de j

Fonction objectif du SSPD :

$J(x) = \mathbb{E}[j(x,\delta)] = \sum_{a\in A} c_a x_a + d \cdot \mathbb{E}[[\sum_{a\in A} \delta(a)x_a - D]^+]$

$$\Rightarrow j(x,\delta) = \sum_{a\in A} c_a x_a + d \cdot [\sum_{a\in A} \delta(a)x_a - D]^+$$

- j différentiable p.r. à x sur $\mathbb{R}^n \setminus \{x \in \mathbb{R}^n | \sum_{a\in A} \delta(a)x_a = D\}$
- $\{x \in \mathbb{R}^n | \sum_{a\in A} \delta(a)x_a = D\}$ ensemble négligeable

Estimation du gradient de j

Fonction objectif du SSPD :

$J(x) = \mathbb{E}[j(x,\delta)] = \sum_{a\in A} c_a x_a + d \cdot \mathbb{E}[[\sum_{a\in A} \delta(a)x_a - D]^+]$

$$\Rightarrow j(x,\delta) = \sum_{a\in A} c_a x_a + d \cdot [\sum_{a\in A} \delta(a)x_a - D]^+$$

- j différentiable p.r. à x sur $\mathbb{R}^n \setminus \{x \in \mathbb{R}^n | \sum_{a\in A} \delta(a)x_a = D\}$
- $\{x \in \mathbb{R}^n | \sum_{a\in A} \delta(a)x_a = D\}$ ensemble négligeable

"Définition" du gradient de j :

Estimation du gradient de j

Fonction objectif du SSPD :

$J(x) = \mathbb{E}[j(x,\delta)] = \sum_{a\in A} c_a x_a + d \cdot \mathbb{E}[[\sum_{a\in A} \delta(a)x_a - D]^+]$

$$\Rightarrow j(x,\delta) = \sum_{a\in A} c_a x_a + d \cdot [\sum_{a\in A} \delta(a)x_a - D]^+$$

- j différentiable p.r. à x sur $\mathbb{R}^n \setminus \{x \in \mathbb{R}^n | \sum_{a\in A} \delta(a)x_a = D\}$
- $\{x \in \mathbb{R}^n | \sum_{a\in A} \delta(a)x_a = D\}$ ensemble négligeable

"Définition" du gradient de j :

$$\nabla_x j(x,\delta) := \begin{cases} c & \text{si } \left[\sum_{a\in A} \delta(a)x_a - D \leq 0\right] \\ c + d\cdot\delta & \text{sinon} \end{cases}$$

Outline

Problème : Contraintes linéairement dépendentes

- Contraintes $Mx = b$ indépendentes

Problème : Contraintes linéairement dépendentes

- Contraintes $Mx = b$ indépendentes
- $\{Mx = b\} \cup \mathcal{A}$ peut contenir contraintes dépendentes

Problème : Contraintes linéairement dépendentes

- Contraintes $Mx = b$ indépendentes
- $\{Mx = b\} \cup \mathcal{A}$ peut contenir contraintes dépendentes
- Problème : après suppression de contrainte :
  gradient projeté reste 0
  $\Rightarrow$ l'algorithme boucle

Problème : Contraintes linéairement dépendentes

- Contraintes $Mx = b$ indépendentes
- $\{Mx = b\} \cup \mathcal{A}$ peut contenir contraintes dépendentes
- Problème : après suppression de contrainte :
  gradient projeté reste 0
  $\Rightarrow$ l'algorithme boucle

Problème : Contraintes linéairement dépendentes

- Contraintes $Mx = b$ indépendentes
- $\{Mx = b\} \cup \mathcal{A}$ peut contenir contraintes dépendentes
- Problème : après suppression de contrainte :
  gradient projeté reste 0
  $\Rightarrow$ l'algorithme boucle

Solution ?

- Ajouter à $\mathcal{A}$ seulement contraintes indépendentes

Problème : Contraintes linéairement dépendentes

- Contraintes $Mx = b$ indépendentes
- $\{Mx = b\} \cup \mathcal{A}$ peut contenir contraintes dépendentes
- Problème : après suppression de contrainte :
  gradient projeté reste 0
  $\Rightarrow$ l'algorithme boucle

Solution ?

- Ajouter à $\mathcal{A}$ seulement contraintes indépendentes
- Problème : après suppr. de la contrainte avec mult. min.
  $\exists i$ t.q. $x_i^{k-1} = 0$ et $x_i^k < 0$

Problème : Contraintes linéairement dépendentes

- Contraintes $Mx = b$ indépendentes
- $\{Mx = b\} \cup \mathcal{A}$ peut contenir contraintes dépendentes
- Problème : après suppression de contrainte :
  gradient projeté reste 0
  $\Rightarrow$ l'algorithme boucle

Solution ?

- Ajouter à $\mathcal{A}$ seulement contraintes indépendentes
- Problème : après suppr. de la contrainte avec mult. min.
  $\exists i$ t.q. $x_i^{k-1} = 0$ et $x_i^k < 0$
- $\Rightarrow \overline{\alpha} = 0$
  $\Rightarrow$ l'algorithme boucle

Merci pour votre attention !

Merci pour votre attention !

Questions ? *Réponses ?*