Optimiser dans l'incertain Programmation robuste

Option 3A Optimisation 1

Alain Faye

Plan

- Programmation robuste
  - Scenarios
    - Différents critères de robustesse
    - Moyenne ordonnée OWA
  - Ensemble d'incertitude
    - Contraintes avec coefficients incertains
- Programmation avec recours
  - Recours robuste
  - Recours avec OWA
- Modèles en AMPL

Programmation robuste

Données incertaines – Programmation robuste

On a un problème d'optimisation avec des données incertaines

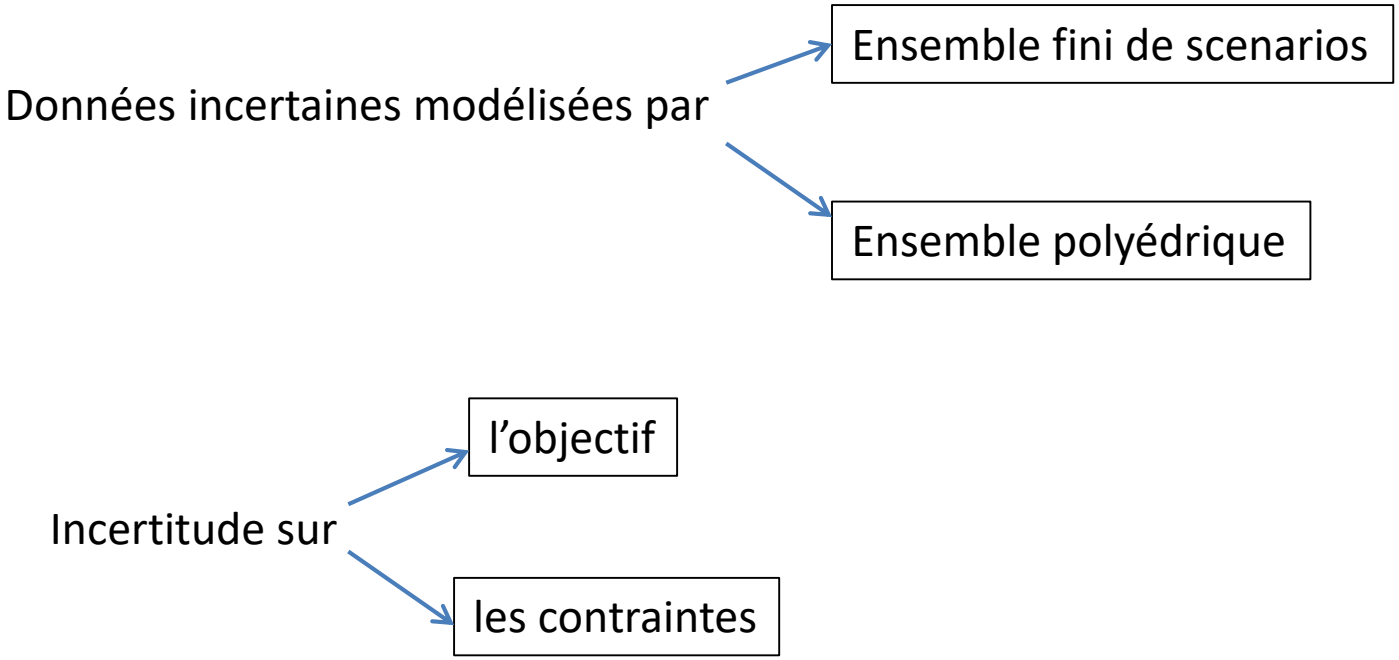
Que faire ?

On pourrait optimiser la valeur moyenne de l'objectif par exemple
- Mais on n'a pas forcément connaissance des lois de probabilité suivies par les données
- De plus une solution qui optimise une valeur moyenne peut être mauvaise pour certaines réalisations des données

L'approche **robuste** consiste à chercher une solution « bonne » quelque soit la réalisation des données

Optimiser sous des données incertaines

Problème d'optimisation avec données incertaines



Approche par scenarios

Incertitude sur l'objectif

Problème d'optimisation $\min_{x \in X} f(x)$

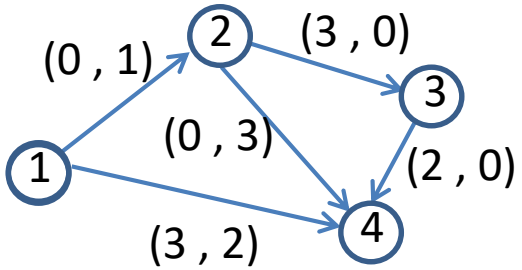
Objectif incertain : *S* ensemble de scenarios possibles

Plusieurs objectifs $f_s \; s \in S$ dépendant des scenarios

On a pu élaborer S un ensemble de scenarios envisageables mais on n'a pas de loi de probabilité sur ces scenarios

Exemple

Plus court chemin de 1 à 4



2 scenarios sur la valeur des arcs

chemins	valeur pour sc.1	valeur pour sc.2
1-2-3-4	5	1
1-2-4	0	4
1-4	3	2

Quel chemin choisir ?

Critères robustes

On minimise le pire coût

$$\min_{x \in X} \max_{s \in S} f_s(x)$$

chemins	valeur sc.1	valeur sc.2	Max(sc1,sc2)
1-2-3-4	5	1	5
1-2-4	0	4	4
1-4	3	2	3
		Min max =	3

On choisit chemin 1-4

Quoiqu'il arrive le coût ne dépassera pas 3

Critères robustes

On minimise le pire écartement par rapport au cout optimal du scenario

x_s^* = solution optimale pour le scenario *s*

$$\min_{x \in X} \max_{s \in S} \left(f_s(x) - f_s(x_s^*) \right)$$

chemins	val. sc.1	val. sc.2	val. sc1 - min sc1	val. sc2 - min sc2	max
1-2-3-4	5	1	5	0	5
1-2-4	0	4	0	3	3
1-4	3	2	3	1	3
min	0	1			3

On peut choisir chemin 1-4 ou 1-2-4

Quoiqu'il arrive l'écartement par rapport au cout optimal ne dépassera pas 3

Critères robustes

Le critère avec écartement nécessite la connaissance de la valeur optimale pour chaque scenario => coûteux en temps de calcul

Dans la pratique, on utilise généralement le critère avec le coût absolu

Exercices

- Donner un programme mathématique (linéaire) pour calculer le plus court chemin d'un sommet à un autre dans un graphe orienté.

- Pour un graphe avec un ensemble S de scenarios possibles sur les valeurs des arcs, donner le programme mathématique qui calcule le chemin qui minimise le pire des scenarios .

Programmation stochastique

- Ensemble de K scenarios S={s_1,…,s_K}. On connaît la probabilité p_k de chaque scenario k.
- On peut minimiser l'espérance de la fonction objectif: $\min_{x \in X} \sum_{k=1}^{K} p_k \, f_{s_k}(x)$
  - Si $f_s(x)$ est polynomiale, cela revient à minimiser une fonction moyenne

Variante de la programmation stochastique: minimiser la moyenne des pires cas

- Le pire cas arrivera avec une probabilité λ_1 , le deuxième pire cas avec probabilité λ_2 , le troisième pire cas etc…
- Hypothèse prudente : la probabilité du pire cas est la plus haute, la probabilité du deuxième pire cas vient en 2^{è} position etc…
- $\Sigma_i \lambda_i = 1$ et $\lambda_i \geq \lambda_{i+1}$ $\forall i$
- On va minimiser la moyenne ordonnée des cas

Moyenne ordonnée

p critères $c_1, c_2, \cdots, c_p$, p poids décroissants $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_p \geq 0$ avec $\sum_{i=1}^{p} \lambda_i = 1$
On trie les critères dans l'ordre décroissant

$$c_{(1)} \geq c_{(2)} \geq \cdots \geq c_{(p)}$$

(i) désigne le critère en position i

Moyenne ordonnée OWA (Ordered Weighted Average)

$$\lambda_1 c_{(1)} + \lambda_2 c_{(2)} + \cdots + \lambda_p c_{(p)}$$

Moyenne ordonnée

Exemple :

3 critères $c_1 = 4, c_2 = 3, c_3 = 5$

Critères triés $c_{(1)} = 5, c_{(2)} = 4, c_{(3)} = 3$

- Prenons $\lambda_1 = 0{,}6$, $\lambda_2 = 0{,}3$, $\lambda_3 = 0{,}1$
  OWA = $0{,}6 \times 5 + 0{,}3 \times 4 + 0{,}1 \times 3 = 4{,}5$

- Prenons $\lambda_1 = 1$, $\lambda_2 = 0$, $\lambda_3 = 0$
  OWA = 5

OWA et problème d'optimisation

S ensemble de *p* scenarios , objectif dépendant des scenarios f_s $s \in S$
x fixé $f_{(1)}(x) \geq f_{(2)}(x) \geq \cdots \geq f_{(p)}(x)$ le classement dépend de x

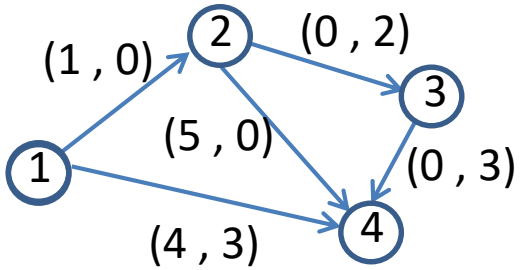
On minimise la moyenne ordonnée

$$\min_{x \in X} OWA(x) = \lambda_1 f_{(1)}(x) + \lambda_2 f_{(2)}(x) + \cdots + \lambda_p f_{(p)}(x)$$

Note : pour $\lambda_1 = 1$, $\lambda_2 = 0, \cdots, \lambda_p = 0$ on retrouve l'optimisation robuste puisque l'on minimise le pire coût

Exemple OWA

Plus court chemin de 1 à 4



2 scenarios sur la valeur des arcs

chemins	valeur pour sc.1	valeur pour sc.2
1-2-3-4	1	5
1-2-4	6	0
1-4	4	3

Exemple OWA

chemins	valeur pour sc.1	valeur pour sc.2	OWA λ_1=0,6 λ_2=0,4	OWA λ_1=1 λ_2=0
1-2-3-4	1	5	3,4	5
1-2-4	6	0	3,6	6
1-4	4	3	3,6	4
		min	3,4	4

Avec une vision robuste, on choisit le chemin 1-4. On ne paiera pas plus que 4.

Si on estime que le second pire scenario a une chance (0,4) de se produire, on choisit le chemin 1-2-3-4. On peut certes payer 5 si on n'a pas de chance mais sinon on peut payer 1.

Programme math. pour calculer OWA

x fixé, il faut classer les $f_s(x)$ $s \in S$ par ordre décroissant
et les affecter aux λ_s qui eux sont déjà classés par ordre décroissant

On définit les variables binaires suivantes :

$y_{ij} = 1$ si on affecte $f_i(x)$ à λ_j , 0 sinon

Pour obtenir $OWA(x) = \lambda_1 f_{(1)}(x) + \lambda_2 f_{(2)}(x) + \cdots + \lambda_p f_{(p)}(x)$,
il suffit de chercher l'affectation y qui maximise $\sum_{j=1}^{p} \lambda_j \sum_{i=1}^{p} f_i(x) y_{ij}$

En effet, pour maximiser cette somme de produits,
on affecte le plus gros au plus gros, le deuxième plus gros au deuxième plus etc ..

L'affectation ordonnée des poids λ maximise la moyenne

Exemple :

3 critères $c_1 = 4, c_2 = 3, c_3 = 5$

Critères triés $c_{(1)} = 5, c_{(2)} = 4, c_{(3)} = 3$
Prenons $\lambda_1 = 0{,}6$, $\lambda_2 = 0{,}3$, $\lambda_3 = 0{,}1$

$$\lambda_2 c_{(1)} + \lambda_1 c_{(2)} + \lambda_3 c_{(3)} = 0{,}3 \times 5 + 0{,}6 \times 4 + 0{,}1 \times 3 = 4{,}2$$

$$\lambda_1 c_{(1)} + \lambda_2 c_{(2)} + \lambda_3 c_{(3)} = 0{,}6 \times 5 + 0{,}3 \times 4 + 0{,}1 \times 3 = 4{,}5$$

Minimiser la moyenne ordonnée

L'opération se passe en 2 temps

1. Pour x fixé, affecter les poids λ de façon ordonnée pour obtenir la moyenne ordonnée
2. Ensuite, minimiser sur x la moyenne ordonnée

Exercice: écrire un PL pour calculer et minimiser cette moyenne ordonnée

Programme pour calculer OWA

$$\max_{y} \sum_{j=1}^{p} \lambda_j \sum_{i=1}^{p} f_i(x) y_{ij}$$

Contraintes d'affectation

$$\begin{cases} \sum_{i=1}^{p} y_{ij} = 1 \ \ \forall j = 1, \dots, p \\ \sum_{j=1}^{p} y_{ij} = 1 \ \ \forall i = 1, \dots, p \end{cases}$$

Les contraintes binaires $y_{ij} \in \{0,1\}$ peuvent être relâchées en $y_{ij} \geq 0$

On obtient donc pour x fixé un PL en y. Cependant on a des produits $f_i(x) y_{ij}$ et quand x n'est plus fixé, le programme n'est plus linéaire.

Programme pour calculer OWA

On prend alors le dual du problème d'affectation précédent

α_i variable duale associée à la contrainte d'affectation i
β_j variable duale associée à la contrainte d'affectation j

$$\min_{\alpha,\beta} \sum_{i=1}^{p} (\alpha_i + \beta_i)$$

sous les contraintes

$$\alpha_i + \beta_j \geq \lambda_j f_i(x) \;\; i, j = 1, \ldots, p$$

Cette fois $f_i(x)$ est multiplié par une constante λ_j

Programme pour minimiser OWA

Donc pour minimiser OWA, on n'a plus qu'à résoudre :

$$\min_{x,\,\alpha,\,\beta} \sum_{i=1}^{p} (\alpha_i + \beta_i)$$

sous les contraintes

$$\begin{cases} \alpha_i + \beta_j \geq \lambda_j f_i(x) \;\; i,j = 1,\dots,p \\ x \in X \end{cases}$$

Exercices

- Donner un programme mathématique (linéaire) pour calculer le plus court chemin d'un sommet à un autre dans un graphe orienté.

- Pour un graphe avec un ensemble S de scenarios possibles sur les valeurs des arcs, donner le programme mathématique qui calcule le chemin qui minimise la moyenne ordonnée de la valeur des chemins.

Ensemble d’incertitude

Ensemble d'incertitude

Scenarios sont en nombre fini
Pas toujours simple de définir un scenario

<u>Généralisation</u>

Chacune des données incertaines est à l'intérieur d'un intervalle donné

Donnée $\tilde{d} \in [1,2]$ plutôt que dire $\tilde{d} = 1 \; ou \; \tilde{d} = 2$

Ensemble d'incertitude

Modèle Bertsimas et Sim

$\tilde{d}$ donnée incertaine dans un intervalle donné : $\tilde{d} \in \left[\bar{d} - \hat{d}, \bar{d} + \hat{d}\right]$
$\bar{d}$ valeur nominale
$\hat{d} > 0$ variation maximum autour de la valeur nominale

Soit de façon équivalente $\tilde{d} = \bar{d} + \hat{d}\Delta$ avec $\Delta \in [-1,1]$

Δ variable aléatoire mais on ne connait pas sa loi de probabilité

On dit simplement que toutes les données ne peuvent pas être en même temps éloignées de leur valeurs nominales.

Seul un certain nombre Γ pourront être différentes de leur valeur nominale.

Ensemble d'incertitude

Modèle Bertsimas et Sim

n données certaines $\tilde{d}_i \quad i = 1, \dots, n$

$\tilde{d}_i = \bar{d}_i + \hat{d}_i \Delta_i$ avec $\Delta_i = [-1,1]$

Ensemble d'incertitude

$$D = \left\{ \Delta \in [-1,1]^n : \sum_{i=1}^{n} |\Delta_i| \leq \Gamma \right\}$$

Ensemble d'incertitude

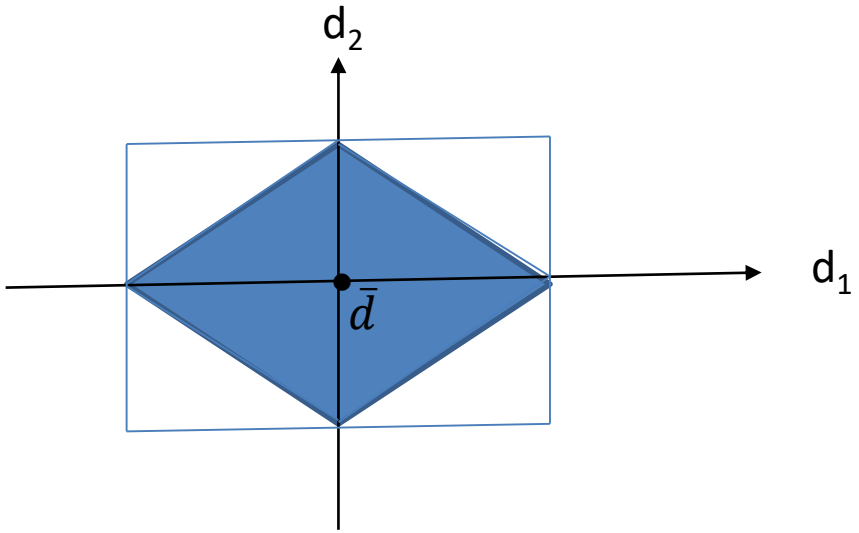
Modèle Bertsimas et Sim

Soit $\hat{d} = sup\{\hat{d}_i:\ i = 1, \dots, n\}$

D est inclus dans la boule (fermée) centrée en le vecteur $\bar{d}$ des valeurs nominales et de rayon $\hat{d}\Gamma$ où la distance est définie par la norme 1 .

$$\tilde{d}_i - \bar{d}_i = \hat{d}_i\Delta_i \leq \hat{d}\Delta_i \Rightarrow \sum_{i=1}^{n} \left|\tilde{d}_i - \bar{d}_i\right| \leq \hat{d} \sum_{i=1}^{n} |\Delta_i| \leq \hat{d}\Gamma$$

C'est-à-dire $\left\|\tilde{d} - \bar{d}\right\|_1 \leq \hat{d}\Gamma$



Contrainte avec des coefficients incertains

$\sum_{i=1}^{n} \tilde{a}_i x_i \leq b$ avec coefficients incertains $\tilde{a}_i = \bar{a}_i + \hat{a}_i \Delta_i \; et \; \Delta_i \in [-1,1]$

La contrainte s'écrit

$$\sum_{i=1}^{n} \bar{a}_i x_i + \sum_{i=1}^{n} \hat{a}_i \Delta_i x_i \leq b$$

On veut une solution x valable quelque soit la réalisation des coefficients dans l'ensemble D d'incertitude.

On remplace la partie incertaine (rouge) par sa valeur maximum dans D.

$$\max_{\Delta \in D} \sum_{i=1}^{n} \hat{a}_i \Delta_i x_i$$

Contrainte avec des coefficients incertains

Protection contre l'incertitude : $\max_{\Delta \in D} \sum_{i=1}^{n} \hat{a}_i \Delta_i x_i$

Pour maximiser, si $x_i > 0$ on prend $\Delta_i \geq 0$ et si $x_i < 0$ on prend $\Delta_i \leq 0$
Donc $\Delta_i x_i = |\Delta_i||x_i|$

La protection se réécrit : $\max_{\Delta} \sum_{i=1}^{n} \hat{a}_i |\Delta_i||x_i| \ \ s.c. \begin{cases} \sum_{i=1}^{n} |\Delta_i| \leq \Gamma \\ \Delta_i \in [-1,1] \ i = 1, \dots, n \end{cases}$

Soit en posant $z_i = |\Delta_i|$: $\max_{z} \sum_{i=1}^{n} \hat{a}_i z_i |x_i| \ \ s.c. \begin{cases} \sum_{i=1}^{n} z_i \leq \Gamma \\ z_i \in [0,1] \ i = 1, \dots, n \end{cases}$

Pour x fixé, ce programme est linéaire.
En raison des produits $z_i|x_i|$, ce n'est plus le cas quand x varie.
On passe au dual.

Contrainte avec des coefficients incertains

Protection contre l'incertitude : $\max_{z} \sum_{i=1}^{n} \hat{a}_i z_i |x_i| \;\; s.c. \begin{cases} \sum_{i=1}^{n} z_i \leq \Gamma \\ z_i \in [0,1] \; i = 1, \dots, n \end{cases}$

On passe au dual. $\min_{\mu,\lambda} \mu\Gamma + \sum_{i=1}^{n} \lambda_i \;\; s.c. \begin{cases} \mu + \lambda_i \geq \hat{a}_i |x_i| \; i = 1, \dots, n \quad (1) \\ \mu \geq 0, \lambda_i \geq 0 \; i = 1, \dots, n \quad (2) \end{cases}$

La contrainte s'écrit

$\sum_{i=1}^{n} \bar{a}_i x_i + \min_{\mu,\lambda} \mu\Gamma + \sum_{i=1}^{n} \lambda_i \leq b$ avec (1), (2)

Maintenant si cette contrainte est dans un programme d'optimisation
on peut enlever le min car les variables μ, λ se positionneront de façon à minimiser $\mu\Gamma + \sum_{i=1}^{n} \lambda_i$ afin que le problème soit moins contraint

Contrainte avec des coefficients incertains

Exercice 1.
On considère un problème de sac-à-dos en variables binaires. Les poids des objets sont incertains.
On veut une solution qui maximise la valeur du sac et robuste par rapport à l'incertitude.
En supposant qu'au plus Γ poids vont dévier de leur valeur nominale, donner le PL en 0-1 qui donne une telle solution.

Même question en supposant maintenant que ce sont les utilités des objets qui sont incertaines

Exercice 2.
Montrer que si les données incertaines sont dans l'objectif, on peut toujours se ramener au cas d'incertitude sur une contrainte

Contrainte avec des coefficients incertains

Garantie en probabilité

Bertsimas et Sim ont donné une borne sur la probabilité que la contrainte incertaine soit violée

Si les var. aléatoires $\tilde{a}_i$ sont indépendantes
Si $\tilde{a}_i$ est symétriquement distribué de part et d'autre de sa valeur nominale $\bar{a}_i \ \forall i$

Si x^* est une solution satisfaisant la contrainte protégée avec le paramètre Γ

$$Proba\left(\sum_{i=1}^{n} \tilde{a}_i x_i \ ^* > b\right) \leq exp\left(-\frac{\Gamma^2}{2n}\right)$$

D.Bertsimas and M.Sim. The price of robustness. Operations Research 52(1): 35-53 (2004)

Programmation robuste avec recours

Programmation robuste avec recours

2 niveaux de décision
Niveau 1 : décision que l'on prend maintenant

Niveau 2 : décision que l'on prendra plus tard une fois les données révélées

$$\min_{x \in X} cx + Q(x)$$

où $Q(x) = \max_{\xi} \min_{y} \{qy : Ax + By \leq h\}$

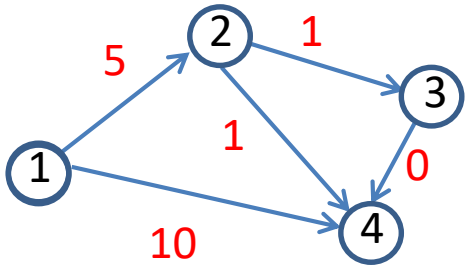
où $\xi = (q, A, B, h)$ représente les données incertaines

x décision de niveau 1
y décision de niveau 2 (variables de recours)

$Q(x)$ est le problème de recours. Il dépend des décisions prises au niveau 1

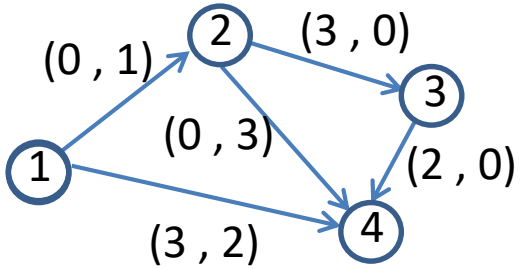
Exemple

On doit construire un réseau afin d'aller d'un point à un autre par le plus court chemin.
Mais les valeurs de parcours des arcs sont incertaines



Coûts de construction des arcs

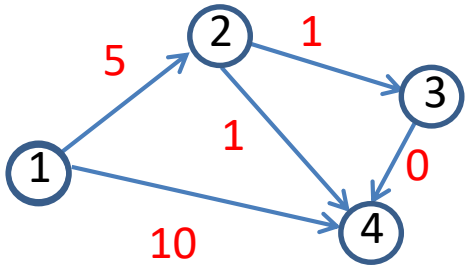
Plus court chemin de 1 à 4



2 scenarios sur la valeur des arcs

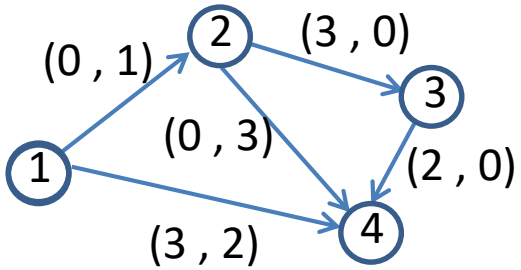
Exemple

On doit construire un réseau afin d'aller d'un point à un autre par le plus court chemin.
Mais les valeurs de parcours des arcs sont incertaines



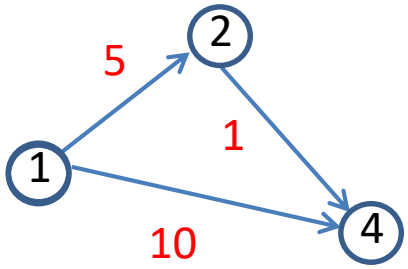
Plus court chemin de 1 à 4

Décision maintenant
Variables x construction du réseau
Quels arcs construit-on ?



2 scenarios sur la valeur des arcs
Pour chaque scenario *s*, il y a les variables de recours y^s qui représentent les arcs parcourus dans le plus court chemin

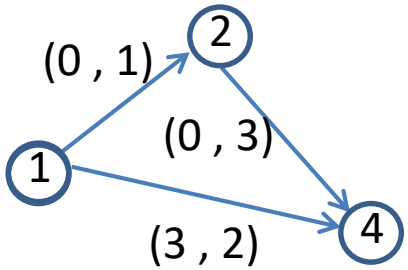
Exemple



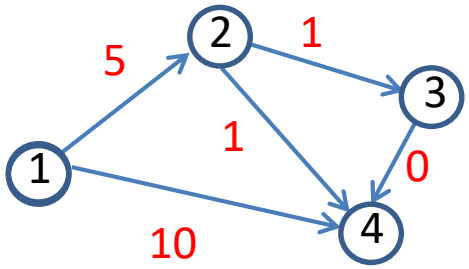
Si je construis ce réseau coût = 16

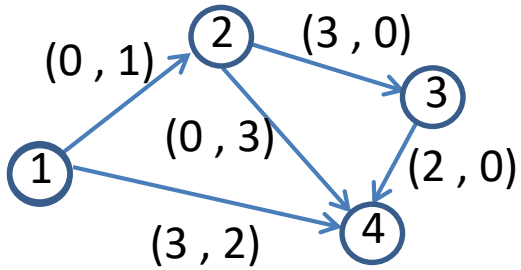
Plus court chemin de 1 à 4

2 scenarios sur la valeur des arcs
Pour chaque scenario
Sc.1 : plus court chemin 1-2-4 valeur=0
Sc.2 : plus court chemin 1-4 valeur=2



Le coût de cette solution est (au pire) 16+max{0,2}=18

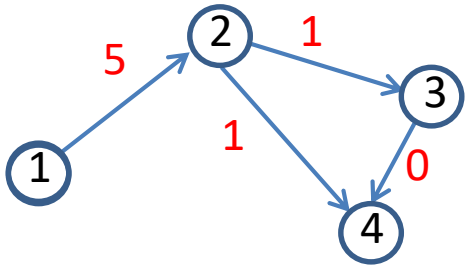




réseau	Coût de construction	Sc.1 plus court chemin de 1 à 4	Sc.2 plus court chemin de 1 à 4	Val. max pour aller de 1 à 4	Cout total construction + max plus court chemin	
1-2-3-4	**6**	5	1	**5**	**11**	
1-2-4	**6**	0	4	**4**	**10**	
1-4	**10**	3	2	**3**	**13**	
1-2-3-4 2-4	**7**	0	1	**1**	**8**	Sol. opt.
1-2-3-4 1-4	**16**	3	1	**3**	**19**	
1-2-4 1-4	**16**	0	2	**2**	**18**	
1-2-3-4 2-4 1-4	**17**	0	1	**1**	**18**	

Commentaires de la solution

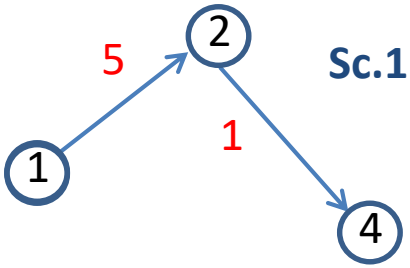
La solution optimale robuste est le graphe formé des arcs 1-2-3-4 et 2-4
Cette solution ne coutera jamais plus de 8 (tenant compte du chemin de 1 à 4)



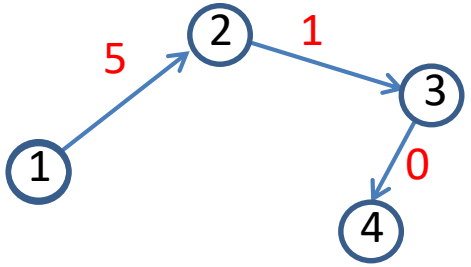
Solution robuste

- Si on savait que c'est le scenario 1 qui va se réaliser alors on prendrait la solution formée par le graphe 1-2-4 de coût total 6 (avec chemin de 1 à 4)
- Si on savait que c'est le scenario 2 qui va se réaliser alors on prendrait la solution formée par le graphe 1-2-3-4 de coût total 7 (avec chemin de 1 à 4)

Dans les 2 cas, c'est inférieur à 8, bien sûr, mais pas très éloigné.



Sc.1



Sc.2

Par contre,
- si on choisissait le graphe 1-2-4 et si le scenario 2 se réalisait : coût total 10
- si on choisissait le graphe 1-2-3-4 et si le scenario 1 se réalisait : coût total 11

Ces 2 solutions coûteront plus que 8.

Programmation avec recours assoupli

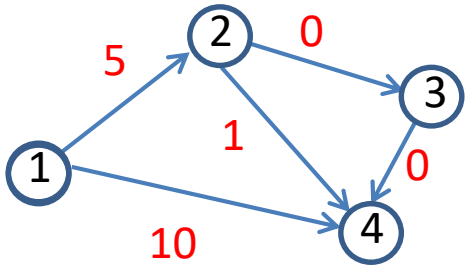
On peut « assouplir » le problème de recours
Au lieu de minimiser le pire scenario
on minimise la moyenne ordonnée OWA des scenarios

$$\min_{x \in X} cx + Q(x)$$

où $Q(x) = \underset{\xi}{\text{OWA}} \min_{y}\{qy : Ax + By \leq h\}$

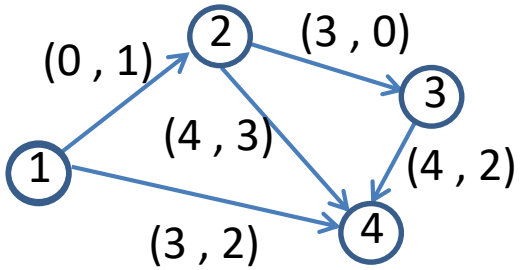
où $\xi = (q, A, B, h)$ représente les données incertaines

Exemple recours avec OWA

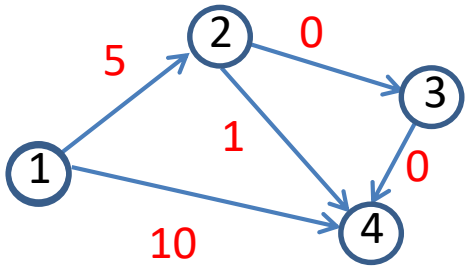


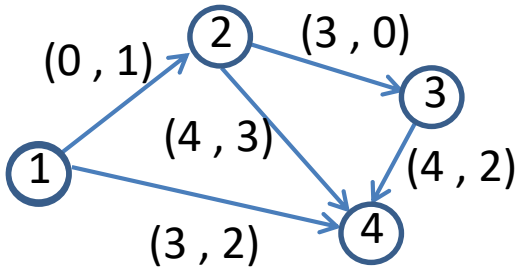
Coûts de construction des arcs

Plus court chemin de 1 à 4



2 scenarios sur la valeur des arcs





réseau	Coût de construction	Sc.1 plus court chemin de 1 à 4	Sc.2 plus court chemin de 1 à 4	Val. max sc.1 et 2 pour aller de 1 à 4	Cout total construction + max plus court chemin	OWA de sc.1 et 2 avec $\lambda_1=\lambda_2=½$	Cout total construction + OWA	
1-2-3-4	**5**	7	3	**7**	**12**	**5**	**10**	
1-2-4	**6**	4	4	**4**	**10**	**4**	**10**	
1-4	**10**	3	2	**3**	**13**	**2,5**	**12,5**	
1-2-3-4 2-4	**6**	4	3	**4**	**10**	**3,5**	**9,5**	sol.opt.
1-2-3-4 1-4	**15**	3	2	**3**	**18**	**2,5**	**17,5**	
1-2-4 1-4	**16**	3	2	**3**	**19**	**2,5**	**18,5**	
1-2-3-4 2-4 1-4	**16**	3	2	**3**	**19**	**2,5**	**18,5**	

Modèle mathématique pour programmation robuste avec recours

S ensemble de scenarios
On met une variable y par scenario
γ représente la valeur Q(x) du problème de recours

$$\gamma \geq \max_{s=1,\dots,|S|} q^s y^s$$

$$\min_{x,\gamma,y} cx + \gamma$$

$$\text{s.c.} \begin{cases} A^s x + B^s y^s \leq h^s & s = 1, \dots, |S| \\ \gamma \geq q^s y^s & s = 1, \dots, |S| \\ x \in X \end{cases}$$

Modèle mathématique pour programmation robuste avec recours OWA

S ensemble de scenarios
On met une variable y par scenario
$\sum_{s=1}^{|S|}(\alpha_s + \beta_s)$ représente la valeur OWA(x) (moyenne ordonnée des scénarios)
$\alpha_s + \beta_{s'} \geq \lambda_{s'} \times q^s y^s$ contraintes pour construire OWA
$\lambda_s \;\; s = 1, \dots, |S|$ sont les poids choisis pour la moyenne ordonnée

$$\min_{x,\alpha,\beta,y} cx + \sum_{s=1}^{|S|}(\alpha_s + \beta_s)$$

$$\text{s.c.} \begin{cases} A^s x + B^s y^s \leq h^s & s = 1, \dots, |S| \\ \alpha_s + \beta_{s'} \geq \lambda_{s'} \times q^s y^s & s, s' = 1, \dots, |S| \\ x \in X \end{cases}$$

Remarque sur la modélisation des problèmes robustes avec ou sans recours

Programmation robuste sans recours

$$\min_{y} \max_{s \in S} f_s(y) \quad \Longleftrightarrow \quad \min_{\gamma, y} \gamma \quad \text{s.c. } \gamma \geq f_s(y) \quad \forall s \in S$$

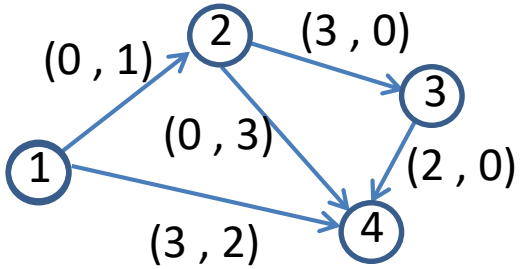
Programmation robuste avec recours:
problème de recours

$$\max_{s \in S} \min_{y} f_s(y) \quad \Longleftrightarrow \quad \min_{\gamma, y^s\, s \in S} \gamma \quad \text{s.c. } \gamma \geq f_s(y^s) \quad \forall s \in S$$

Dans le 1er cas : un y pour tous les scenarios
Dans le 2ème cas : un y^s par scenario

Exemple

Plus court chemin de 1 à 4
2 scenarios sur longueurs des arcs



	sc. 1	sc. 2	max	
ch. 1-2-3-4	5	1	5	
ch. 1-2-4	0	4	4	
ch. 1-4	3	2	3	$\min_{ch}$ max
min	0	1		
		$\max_{sc}$ min		

min max : y = ch. 1-4
max min : y^{sc1} = ch. 1-2-4, y^{sc2} = ch. 1-2-3-4

Problème robuste avec recours K-adaptable

On partitionne l'ensemble d'incertitude (ou les scenarios) en K sous-ensembles $S^1,\dots,S^K$
Et on prend une variable de recours par sous-ensemble S^k

Les 2 cas extrêmes:
K=1 c'est le problème robuste (pas de recours)
K=|S| c'est le problème robuste avec recours (total)

$$\min_{\gamma, y^k \; k=1,\dots,K} \gamma$$
$$\text{s.c. } \gamma \geq f_s\left(y^k\right) \quad \forall s \in S^k, k = 1, \dots, K$$

Pour chaque scenario $s \in S^k$ on prendra la même décision y^k

D. Bertsimas, C. Caramanis. Finite adaptabilty in multistage linear optimisation
IEEE Transactions in Automatic Control, Vol. 55 (2), p.2751-2766 (2010)

Exercices

- Modéliser le problème de la construction de réseau robuste avec recours
- Modéliser le problème de la construction de réseau robuste avec recours assoupli par OWA

par des programmes linéaires en nombres entiers PLNE

Références

D.Bertsimas and M.Sim. The price of robustness. Operations Research 52(1): 35-53 (2004)

D. Bertsimas, C. Caramanis. Finite adaptabilty in multistage linear optimisation
IEEE Transactions in Automatic Control, Vol. 55 (2), p.2751-2766 (2010)

Cédric Hervet. Thèse de l'Ecole Polytechnique en Mathématiques Appliquées
Optimization of Optical Network Deployment. Considerations on demand uncertainty
Soutenue le 18 Décembre 2013

Modèles en AMPL

modèle prog. robuste avec recours en AMPL

```
# programmation robuste avec recours
# construction d'un reseau pour aller d'un sommet depart vers une arrivee
# couts construction connus
# incertitude sur les couts de parcours des arcs sous forme de scenarios
#
# variable x selection arcs pour construction du reseau
# variable y parcours des arcs dans le plus court chemin
#
param nb_som; # nbre sommets du graphe
param nb_scen;# nbre scen des couts de parcours
param d{i in 1..nb_som}; # demandes
param E{i in 1..nb_som,j in 1..nb_som}; # matrice adjacence du graphe
param c{i in 1..nb_som,j in 1..nb_som}; # cout construction
param q{i in 1..nb_som,j in 1..nb_som,k in 1..nb_scen}; # cout parcours arcs
#
# variables
#
var x{i in 1..nb_som,j in 1..nb_som} binary; # choix arc a construire
var y{i in 1..nb_som,j in 1..nb_som,k in 1..nb_scen} >=0, integer;
var gamma;
#
# modele
#
minimize f:sum{i in 1..nb_som,j in 1..nb_som}E[i,j]*c[i,j]*x[i,j]+gamma;
# contraintes
contr_flot{i in 1..nb_som,k in 1..nb_scen}:
sum{j in 1..nb_som}E[j,i]*y[j,i,k]-sum{j in 1..nb_som}E[i,j]*y[i,j,k]=d[i];
contr_gamma{k in 1..nb_scen}:
gamma>=sum{i in 1..nb_som,j in 1..nb_som}E[i,j]*q[i,j,k]*y[i,j,k];
contr_construction{i in 1..nb_som,j in 1..nb_som:E[i,j]<>0}:
sum{k in 1..nb_scen}y[i,j,k]<=nb_scen*x[i,j]; # on ne peut utiliser que arc construit
```

modèle prog. robuste avec recours – les DATA en AMPL

```
param nb_som:=4;
param nb_scen:=2;
# matrice adjacence du graphe
param E: 1 2 3 4 :=
1  0 1 0 1
2  0 0 1 1
3  0 0 0 1
4  0 0 0 0
;
# demande pour definir sommet depart -1 et arrivee +1
param d:=
1  -1
2  0
3  0
4  1;
# cout construction
param c: 1 2 3 4:=
1  0 5 0 10
2  0 0 1 1
3  0 0 0 0.0
4  0 0 0 0
;
# couts de parcours des arcs
param q:=
[*,*,1] :  1 2 3 4 :=
      1  0 0.0 0 3
      2  0 0 3 0.0
      3  0 0 0 2
      4  0 0 0 0
[*,*,2] :  1 2 3 4 :=
      1  0 1 0 2
      2  0 0 0.0 3
      3  0 0 0 0.0
      4  0 0 0 0
       ;
```

Modèle prog. robuste avec recours OWA en AMPL

```
# programmation robuste avec recours assouplie avec OWA (moyenne ordonee)
# construction d'un reseau pour aller d'un sommet depart vers une arrivee
# couts construction connus
# incertitude sur les couts de parcours des arcs sous forme de scenarios
#
# variable x selection arcs pour construction du reseau
# variable y parcours des arcs dans le plus court chemin
#
param nb_som; # nbre sommets du graphe
param nb_scen;# nbre scen des couts de parcours
param d{i in 1..nb_som}; # demandes
param E{i in 1..nb_som,j in 1..nb_som}; # matrice adjacence du graphe
param c{i in 1..nb_som,j in 1..nb_som}; # cout construction
param q{i in 1..nb_som,j in 1..nb_som,k in 1..nb_scen}; # cout parcours arcs
param lambda{k in 1..nb_scen};
#
# variables
#
var x{i in 1..nb_som,j in 1..nb_som} binary; # choix arc a construire
var y{i in 1..nb_som,j in 1..nb_som,k in 1..nb_scen} >=0, integer;
var alpha{k in 1..nb_scen};
var beta{k in 1..nb_scen};
#
# modele
#
minimize f:
sum{i in 1..nb_som,j in 1..nb_som}E[i,j]*c[i,j]*x[i,j]+sum{k in 1..nb_scen}(alpha[k]+beta[k]);
# contraintes
# flot
contr_flot{i in 1..nb_som,k in 1..nb_scen}:
sum{j in 1..nb_som}E[j,i]*y[j,i,k]-sum{j in 1..nb_som}E[i,j]*y[i,j,k]=d[i];
# contrainte pour calculer OWA
contr_OWA{k in 1..nb_scen,k_pr in 1..nb_scen}:
alpha[k]+beta[k_pr]>=lambda[k_pr]*sum{i in 1..nb_som,j in 1..nb_som}E[i,j]*q[i,j,k]*y[i,j,k];
# on ne peut passer par un arc i,j que s'il est construit
contr_construction{i in 1..nb_som,j in 1..nb_som:E[i,j]<>0}:
sum{k in 1..nb_scen}y[i,j,k]<=nb_scen*x[i,j];
```

Modèle prog. robuste avec recours OWA – les DATA en AMPL

```
param nb_som:=4;
param nb_scen:=2;
# on doit avoir lambda[1]>=lambda[2]>=... et la somme =1
param lambda:=
1  0.75
2  0.25
;
# matrice adjacence du graphe
param E: 1 2 3 4 :=
1  0 1 0 1
2  0 0 1 1
3  0 0 0 1
4  0 0 0 0
;
# demande pour definir sommet depart -1 et arrivee +1
param d:=
1   -1
2   0
3   0
4   1;
# cout construction
param c: 1 2 3 4:=
1  0 5 0 10
2  0 0 0.0 1
3  0 0 0 0.0
4  0 0 0 0
;
# couts de parcours des arcs
param q:=
[*,*,1] :  1 2 3 4 :=
      1  0 0.0 0 3
      2  0 0 3 4
      3  0 0 0 4
      4  0 0 0 0
[*,*,2] :  1 2 3 4 :=
      1  0 1 0 2
      2  0 0 0.0 3
      3  0 0 0 2
      4  0 0 0 0
       ;
```