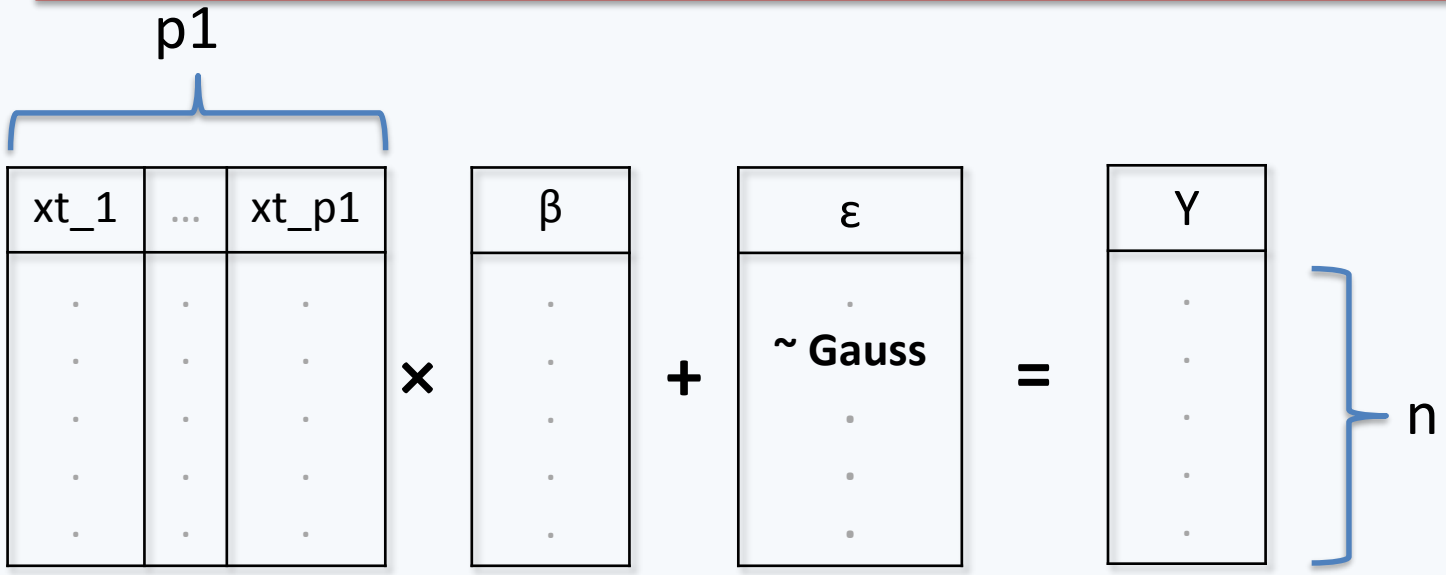
ENSAE ParisTech
École nationale de la statistique et de l'administration économique
université PARIS-SACLAY

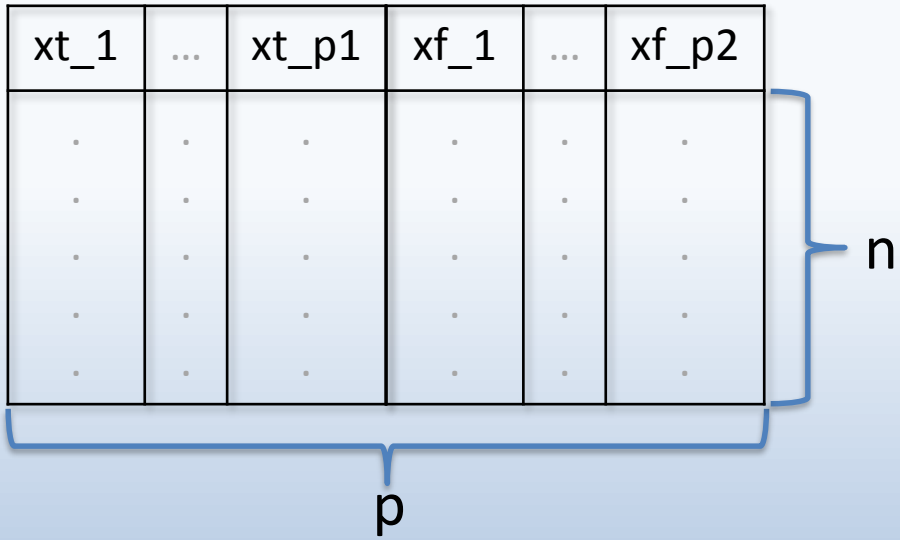
Simulation et Monte Carlo

Sélection de variables

Pierre Le Pelletier de Woillemont – Jingmei Jiang – Julien Sauvan

Méthodologie





On cherche à minimise la fonction:

S → BIC(modele_S)

Où S est un vecteur binaire de dimension p et modele_S est la régression linéaire de Y sur les X correspondants à S=1

Recuit simulé basé sur un noyau de type Metropolis

- Inspiré de la physique thermodynamique: notion d'énergie et de température

- Principe général: en modifiant un état donné du système, on obtient un autre état. S'il améliore le critère que l'on cherche à optimiser, l'énergie diminue. L'objectif est de minimiser l'énergie du système.

Recuit simulé basé sur un noyau de type Metropolis

- Etat initial: énergie initiale E=E0
  température initiale T=T0
- Itérations:
  -Modification de l'état du système
  -Entraîne variation ΔE de l'énergie
  -Si ΔE < 0, on l'applique à l'état courant
  -Sinon, on l'accepte avec une probabilité $e^{(-\Delta E/T)}$ *(règle de Metropolis)*
  -On itère en diminuant la température de façon logarithmique
  -Lorsque le système atteint un équilibre après un nombre suffisant d'itérations, l'algorithme s'arrête.

Recuit simulé basé sur un noyau de type Metropolis

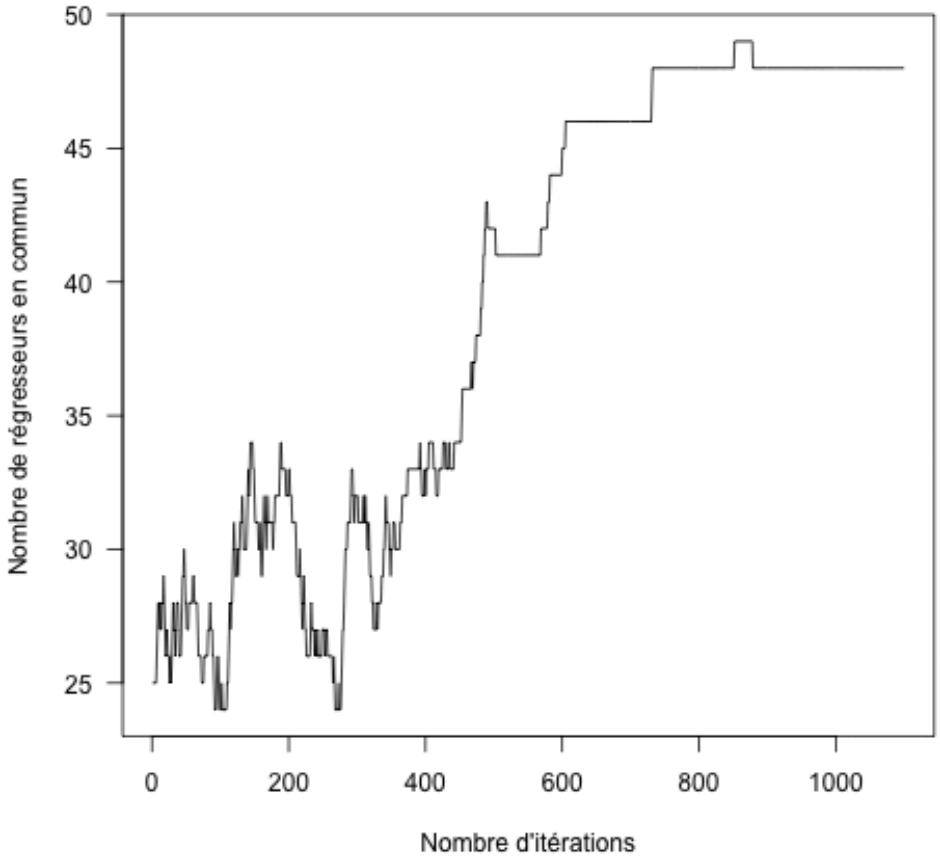
- Noyau de type Metropolis: la modification se fait sur les « voisins »

- Pseudo-code:

```
s := s0
e := E(s)
k := 0
tant que k < kmax et e > emax
  sn := voisin(s)
  en := E(sn)
  si en < e ou aléatoire() < P(en - e, temp(k/kmax)) alors
    s := sn; e := en
  k := k + 1
retourne s
```

Recuit simulé basé sur un noyau de type Metropolis

Convergence de l'algorithme



Recuit simulé basé sur un noyau de Gibbs

$$s_i^0 = \begin{cases} 1, & \text{si on choisit le i-ème variable, i = 1,2,…,p} \\ 0, & \text{sinon} \end{cases}$$

Distribution de probabilité (Gibbs):

$$p(s_1^{(1)}, T_p) = \frac{\exp(-\frac{BIC(s_1^{(0)}=1)}{T_p})}{\exp(-\frac{BIC(s_1^{(0)}=1)}{T_p}) + \exp(-\frac{BIC(s_1^{(0)}=0)}{T_p})}$$

$$s_1^1 \sim B(p(s_1^{(1)}, T_p))$$

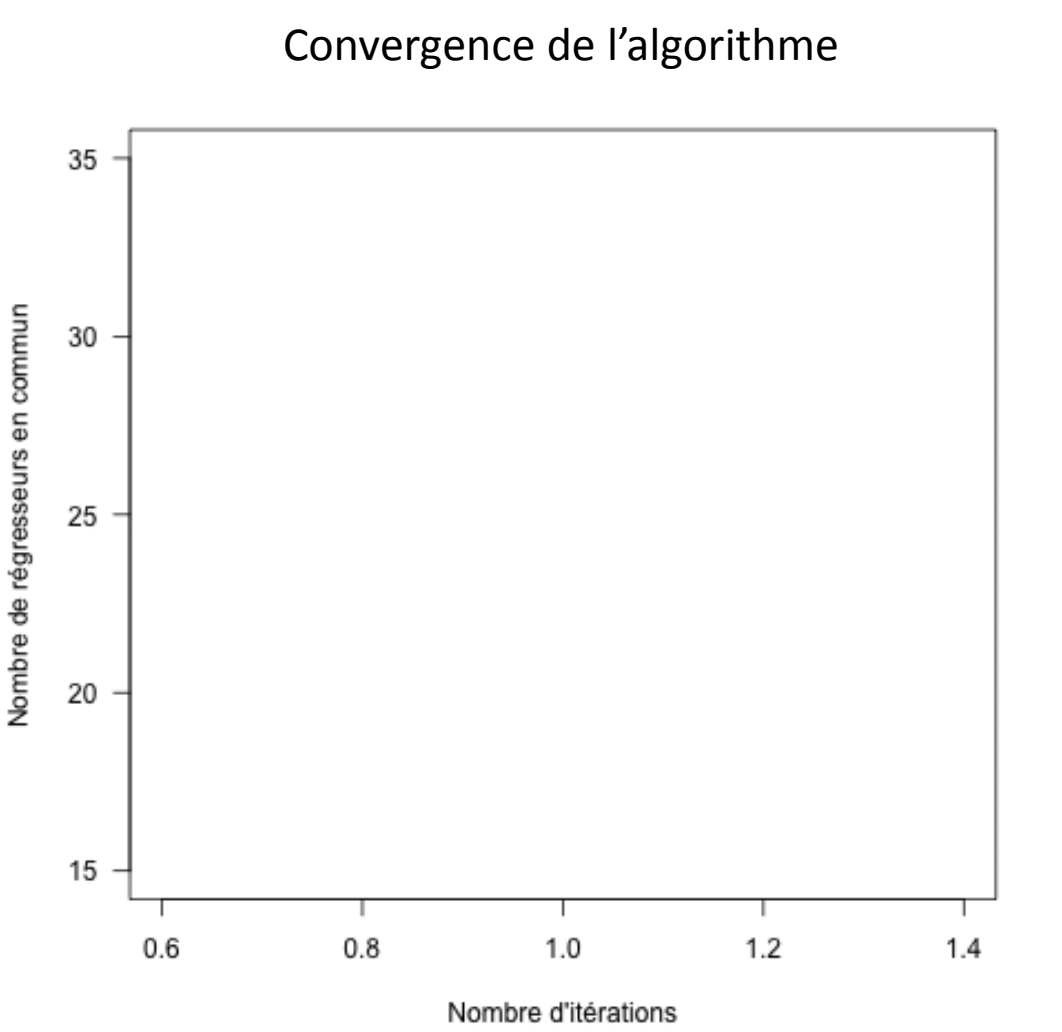
Échantillonneur de Gibbs:

$$\begin{cases} s_1^{(t+1)} \sim B(p(s_1 \mid s_2^{(t)}, s_3^{(t)}, \ldots, s_p^{(t)})) \\ s_2^{(t+1)} \sim B(p(s_2 \mid s_1^{(t+1)}, s_3^{(t)}, \ldots, s_p^{(t)})) \\ \vdots \\ s_p^{(t+1)} \sim B(p(s_p \mid s_1^{(t+1)}, s_2^{(t+1)}, \ldots, s_{p-1}^{(t+1)})) \end{cases}$$

Décroissance de température:

$$T_{p+1} = T_p / \log(1+t)$$

Recuit simulé basé sur un noyau de Gibbs



Algorithme Cross-Entropy

$\theta = \{0.5\}^p \rightarrow$

S_1	...	S_N
$\sim \text{Ber}(\theta)$	. . .	$\sim \text{Ber}(\theta)$

→ *Trie par le BIC* →

S_i	BIC_i
.	.
.	.
.	.
.	.

On sélectionne les meilleurs modèles

On redéfinit θ comme l'estimateur $\tilde{\theta}$ d'une Bernoulli de dimension p; uniquement à partir de l'échantillon « *préférentiel* »

$\theta = \tilde{\theta} \rightarrow$

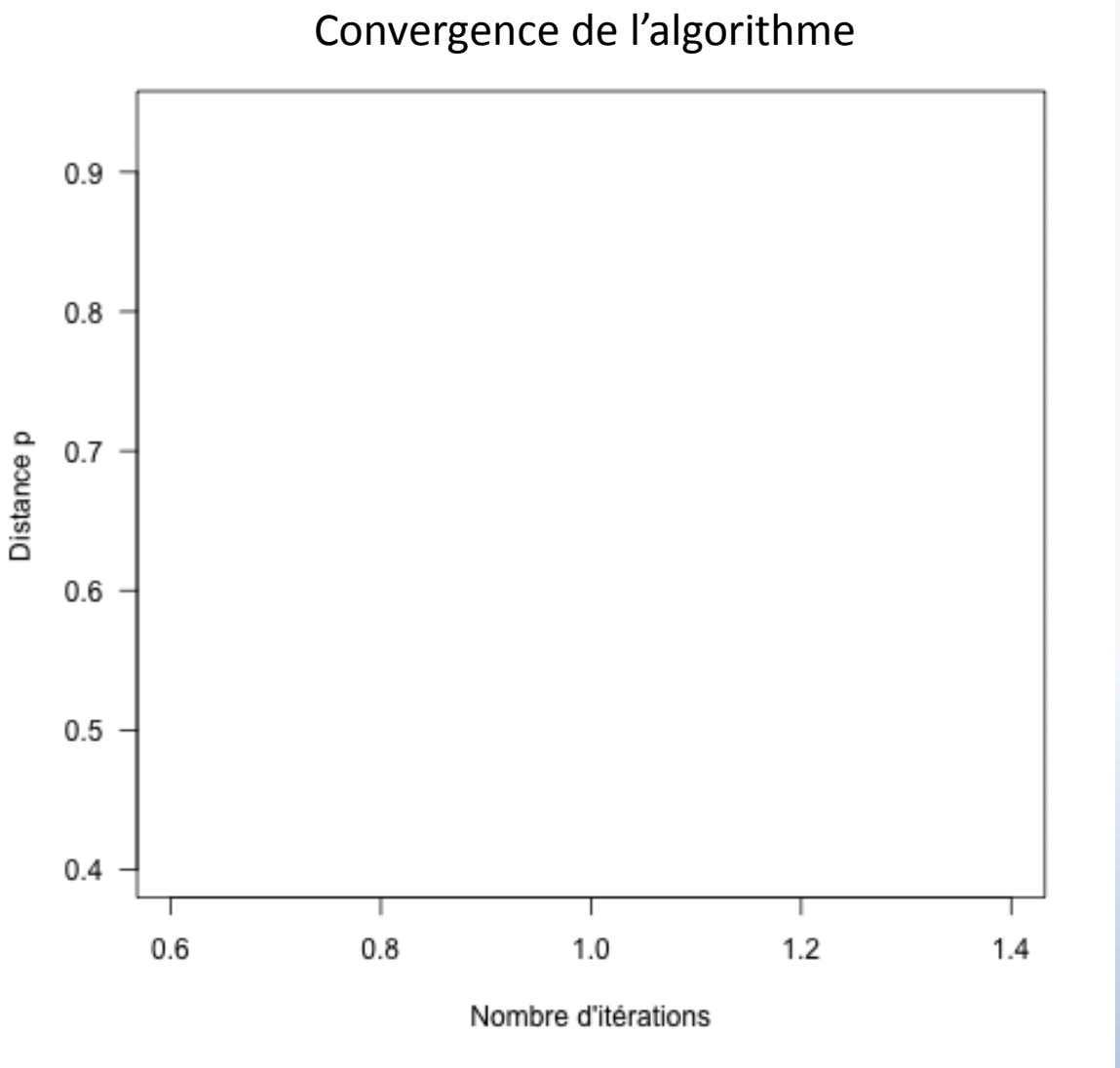
S_1	...	S_N
$\sim \text{Ber}(\theta)$	. . .	$\sim \text{Ber}(\theta)$

→ *Trie par le BIC* →

S_i	BIC_i
.	.
.	.
.	.
.	.

On sélectionne les meilleurs modèles

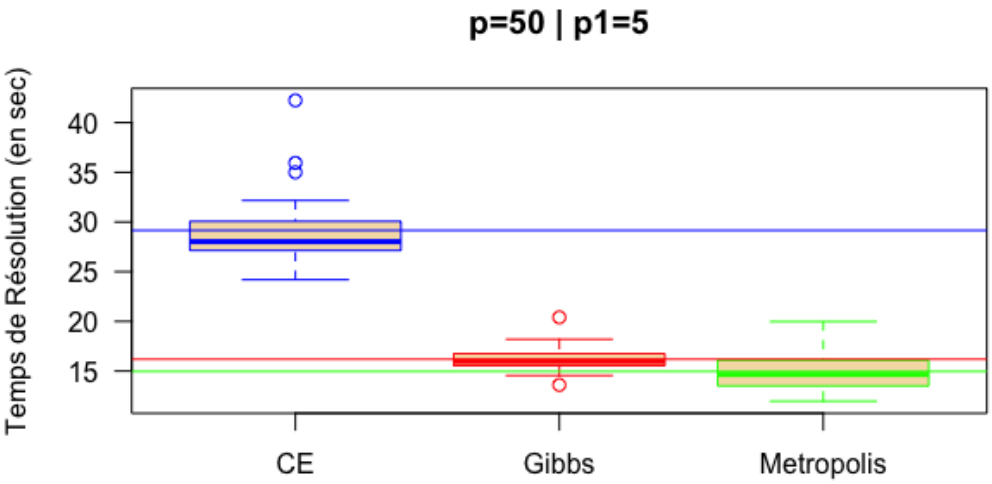
Algorithme Cross-Entropy

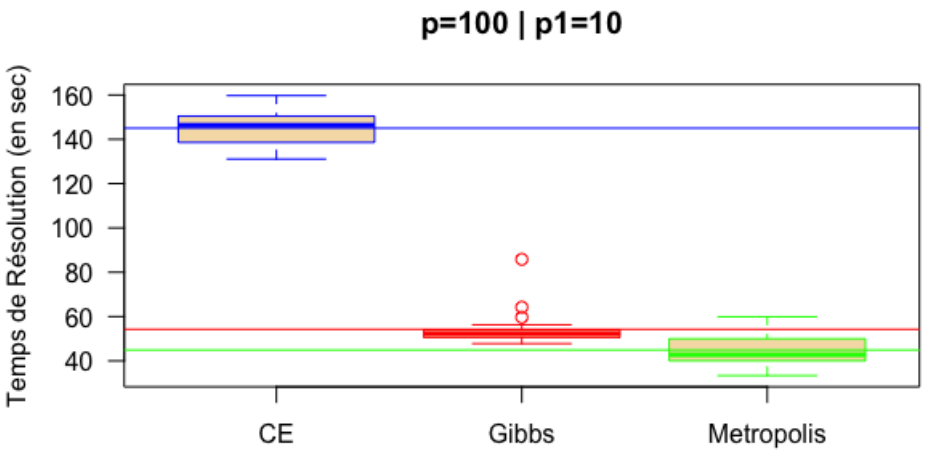


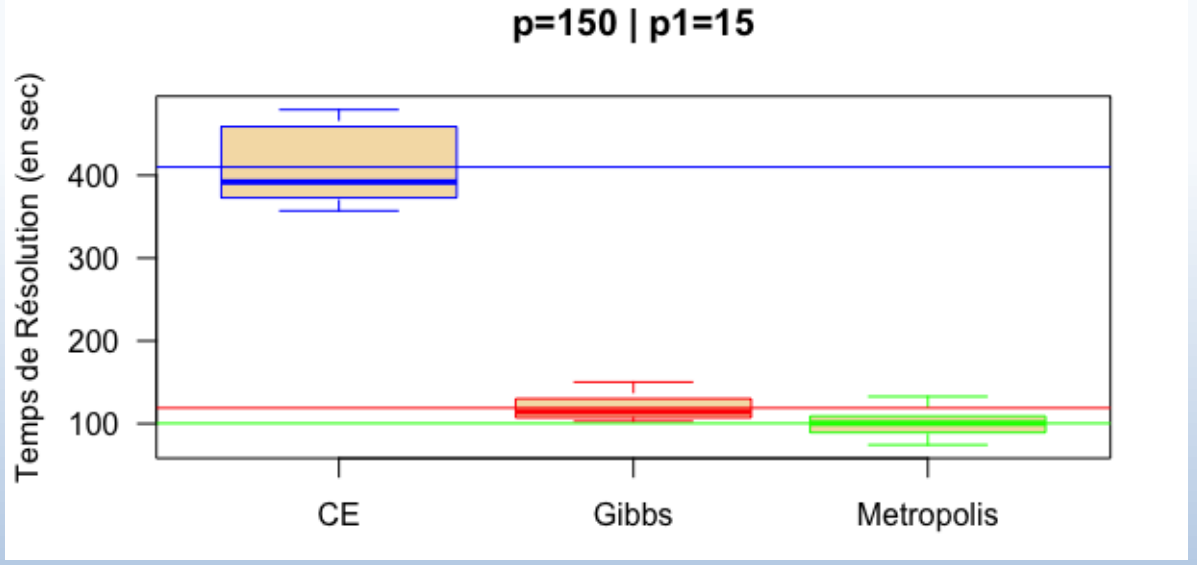
Comparaison

- 2 critères de comparaison: vitesse des algorithmes et qualité de la sélection de variables
- Mesure de la qualité de la sélection de variables: comparaison entre le « vrai » vecteur binaire et les vecteurs de résultats donnés par chaque algorithme.

Vitesse des algorithmes







Précision des algorithmes

Performance des algorithmes pour p=50

p1	Metropolis	Gibbs	Cross Entropy
5 (p1 << p2)	49,37 (0,89)	49,7 (0,60)	**49,83** (0,38)
15 (p1 < p2)	**49,53** (0,73)	49,47 (0,78)	**49,53** (0,73)
25 (p1 = p2)	**49,83** (0,53)	**49,83** (0,53)	49,77 (0,57)
40 (p1 > p2)	49,87 (0,43)	**49,90** (0,40)	49,83 (0,46)

Performance des algorithmes