jointSeg : Segmentation de données génomiques en cancérologie

Morgane Pierre-Jean, Pierre Neuvial

Laboratoire Statistique et Génome
Université d'Évry Val d'Éssonne
UMR CNRS 8071 USC INRA

CERIM
Université Lille 2 Droit et Santé
Faculté de médecine

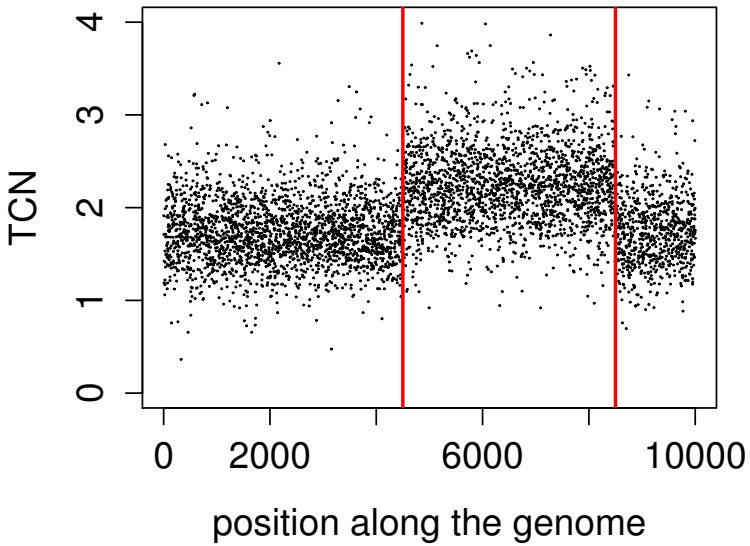
2013-06-28

Sommaire

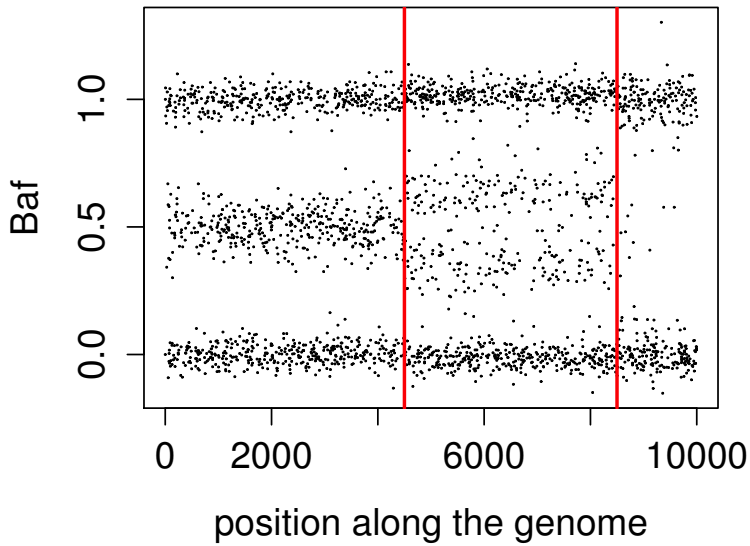
Sommaire

Données et objectif

Total copy number



B allele fraction

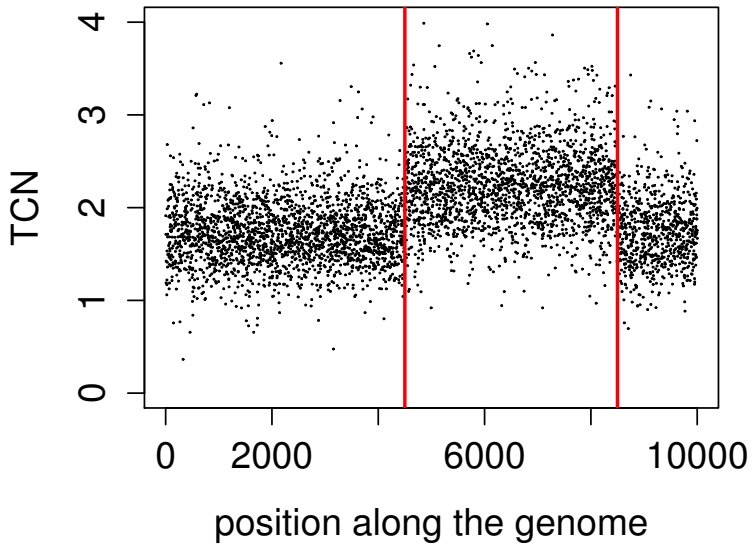


Segmentation

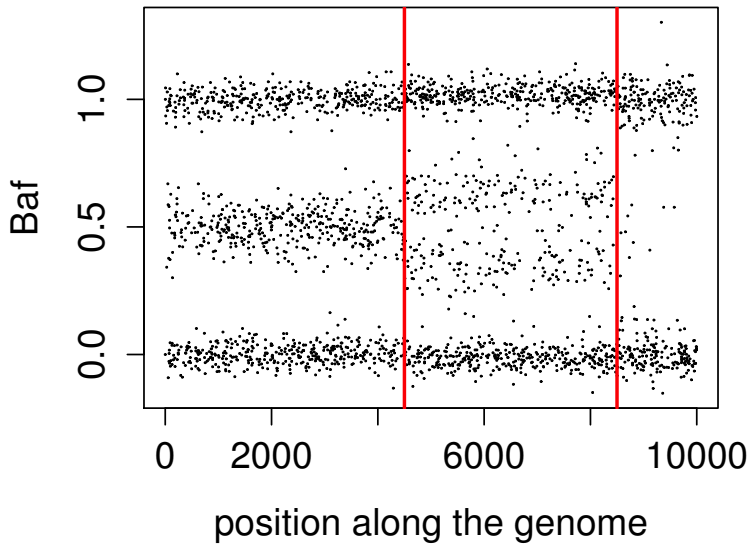
→ **Retrouver les ruptures dans le signal**

Données et objectif

Total copy number



B allele fraction



Segmentation jointe

→ **Retrouver les ruptures dans le signal bidimensionnel**

Contexte général

But des études de nombre de copies d'ADN : Identifier les régions altérées du génome.

- Comprendre les progressions des tumeurs
- Mettre en place des thérapies personnalisées

On s'intéresse à l'identification des ruptures
JOINTSEG est disponible depuis Janvier 2013 sur R-forge :
`http://r-forge.r-project.org/R/?group_id=1562`

Sommaire

Simulations

Principe

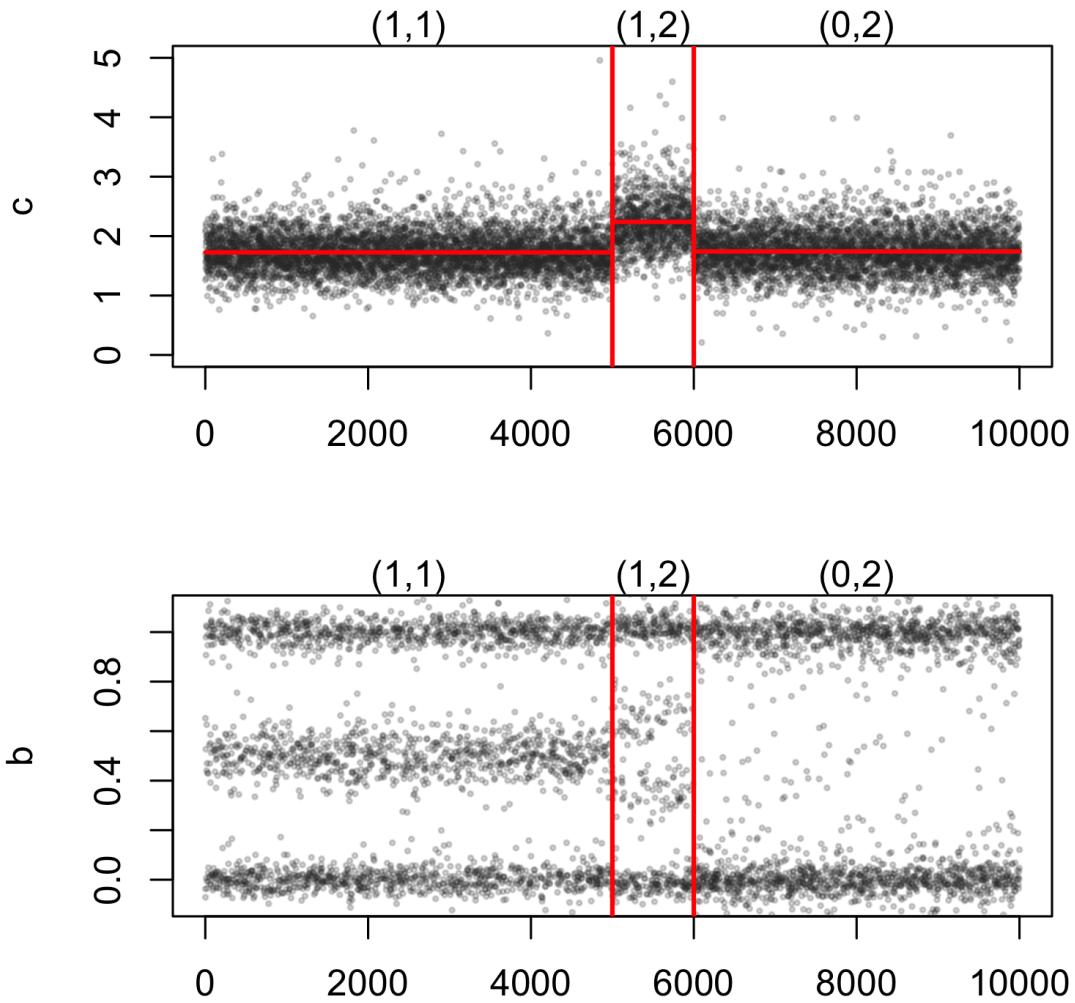
A partir de vraies données → Création de données synthétiques par rééchantillonnage

Avantage :

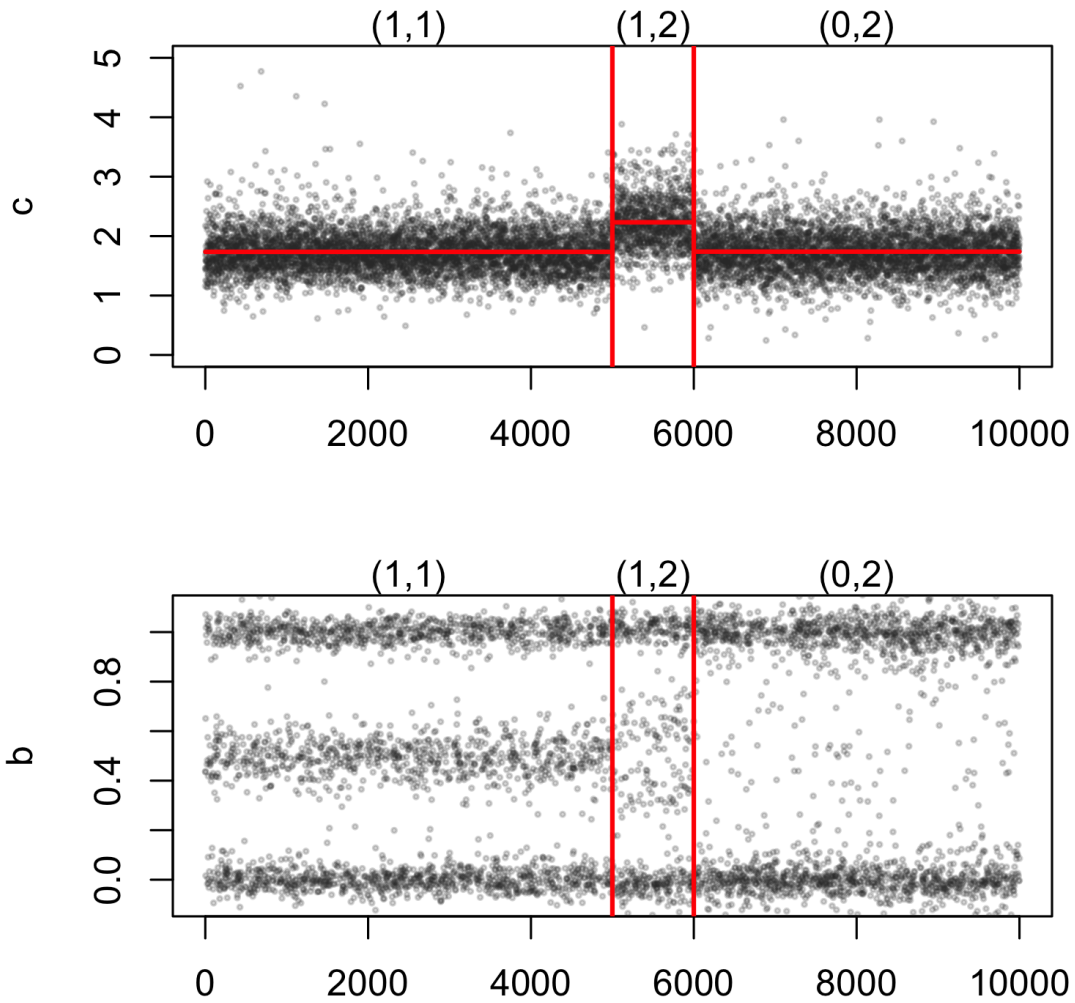
- Petit nombre de paramètres
- Interprétation biologique claire
  - proportion de cellules tumorales
  - longueur du signal
  - nombre de ruptures

```
 library(jointSeg)
data <- loadCnRegionData(platform="Affymetrix",
tumorFraction=1)
sim <- getCopyNumberDataByResampling(1e4, nBkp=2,
regData=data)
```

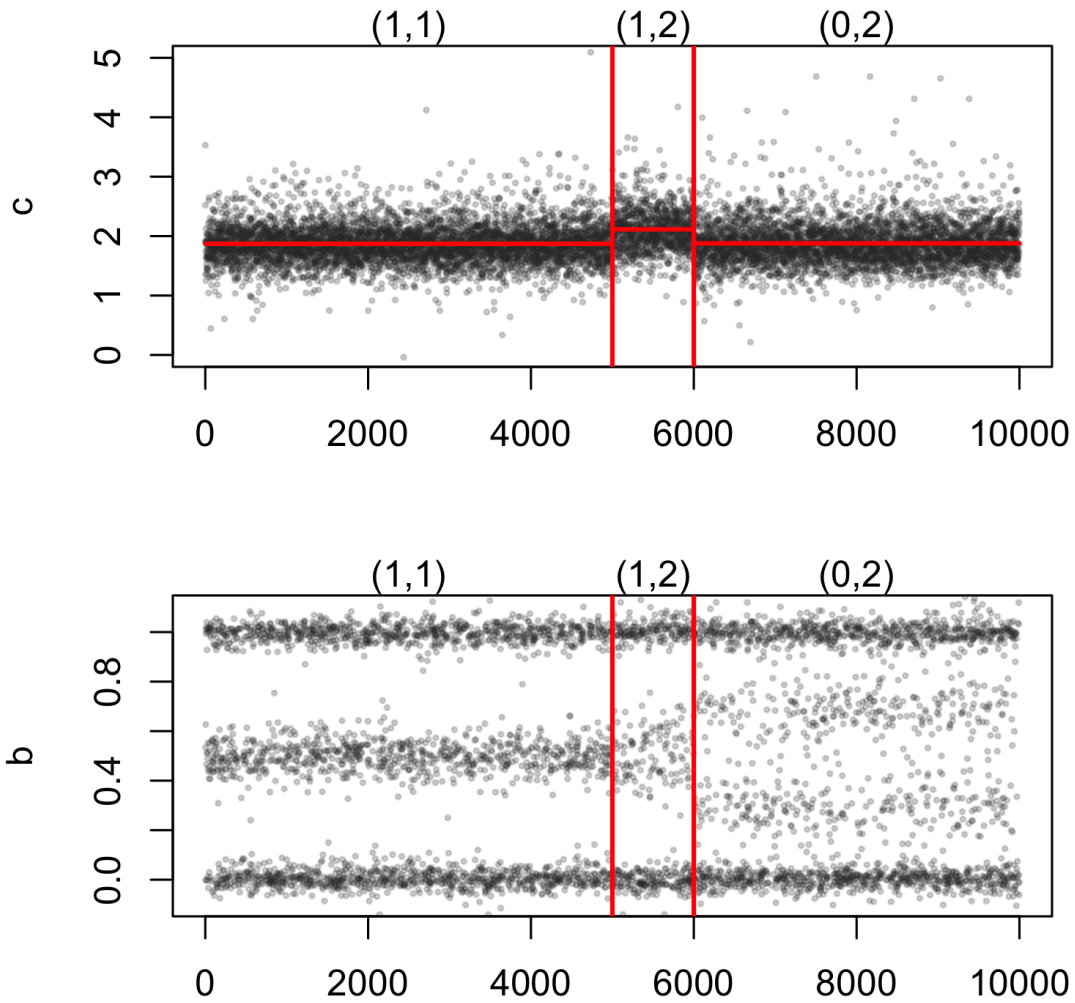
Exemple : data set 1, 100% de cellules tumorales



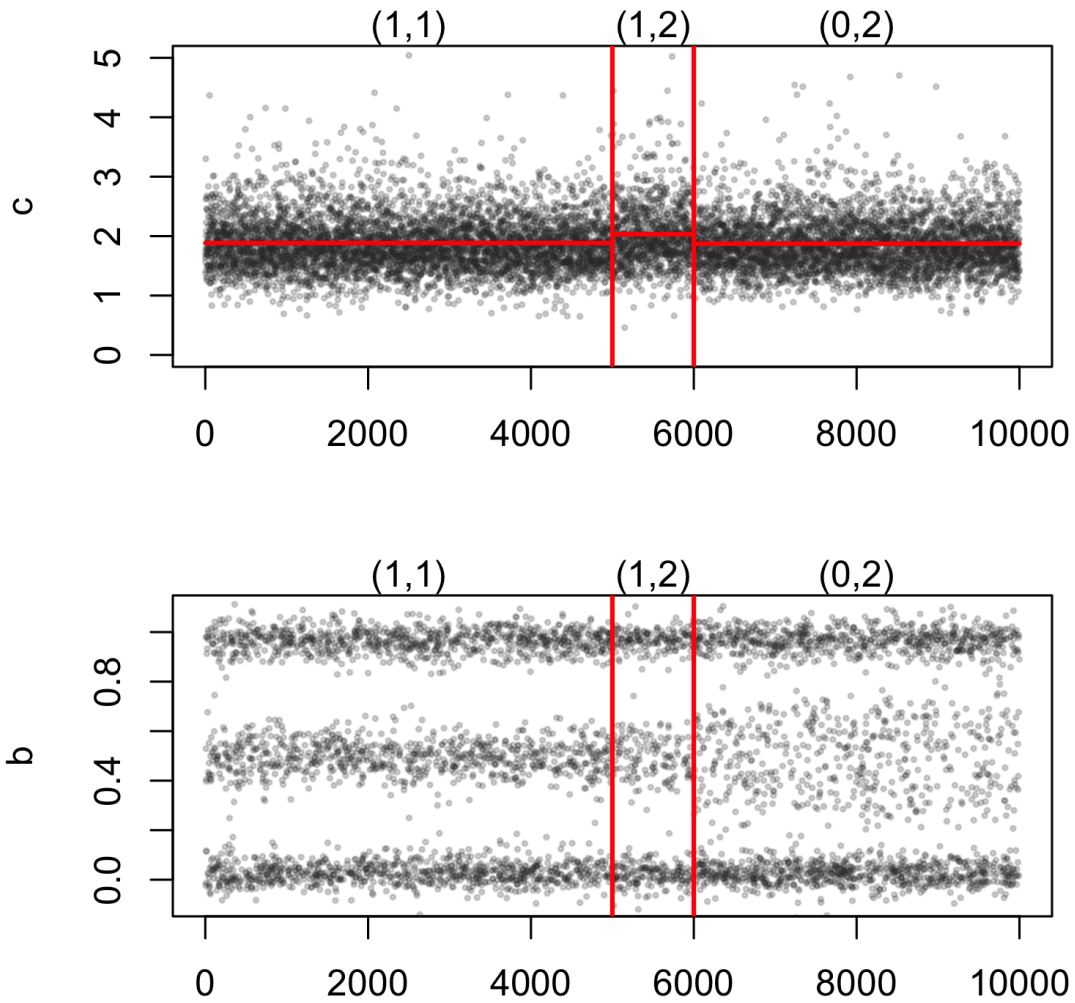
Exemple : data set 1, 100% de cellules tumorales (2)



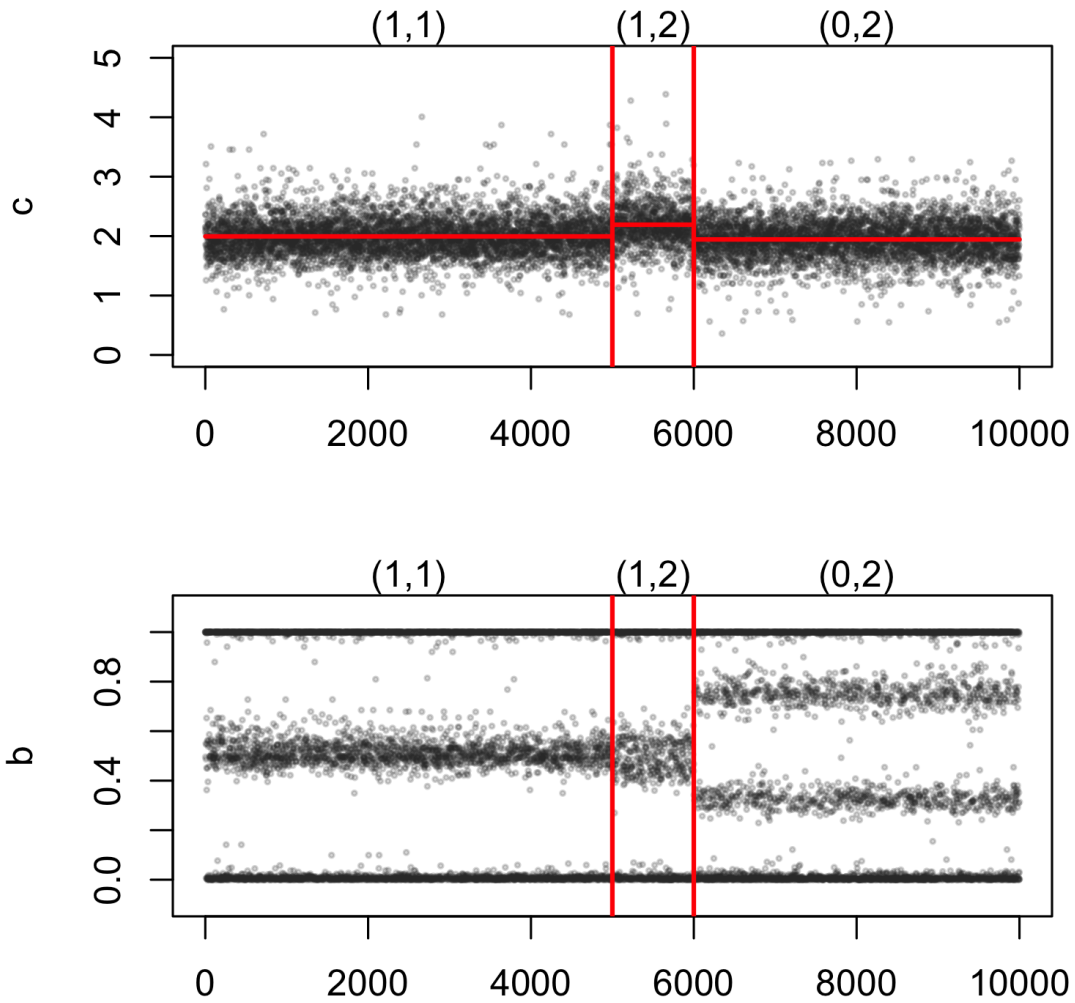
Exemple : data set 1, 70% de cellules tumorales



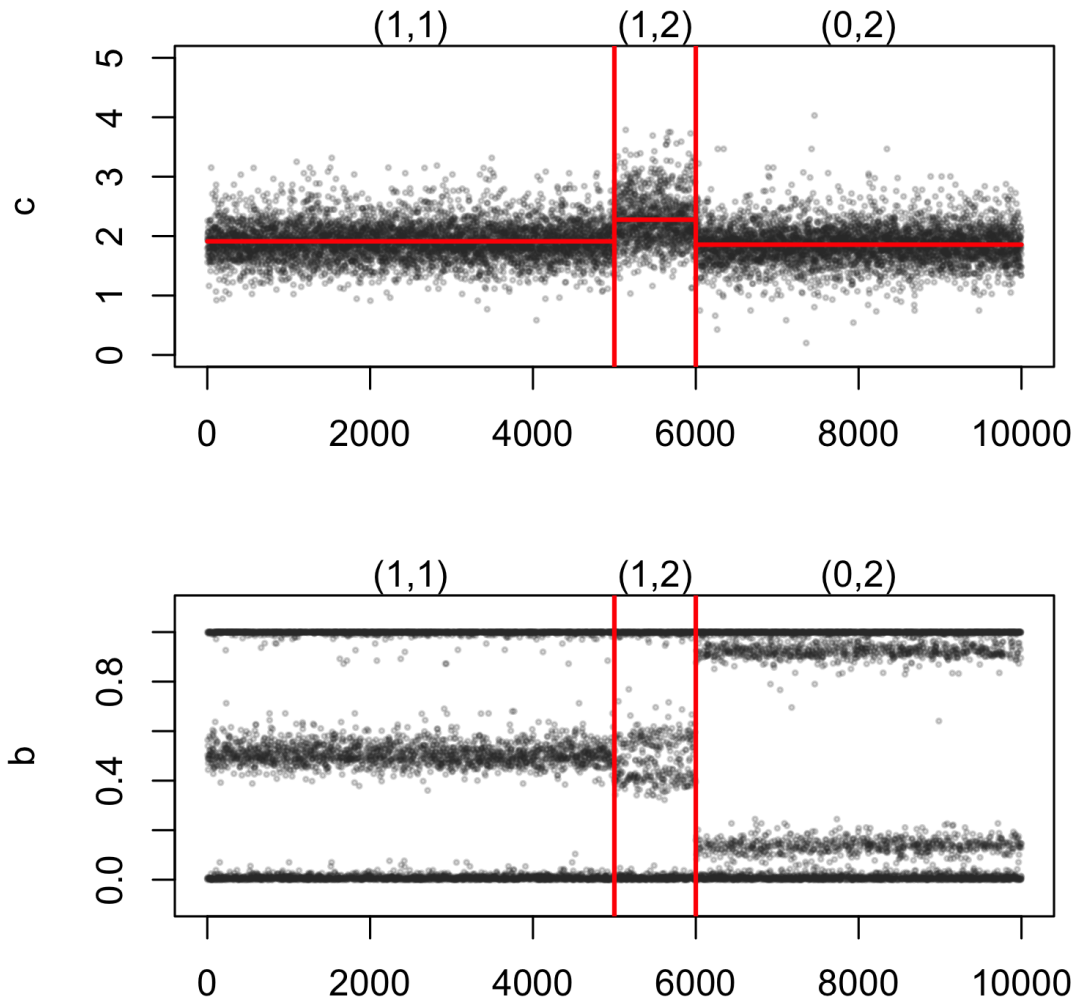
Exemple : data set 1, 50% de cellules tumorales



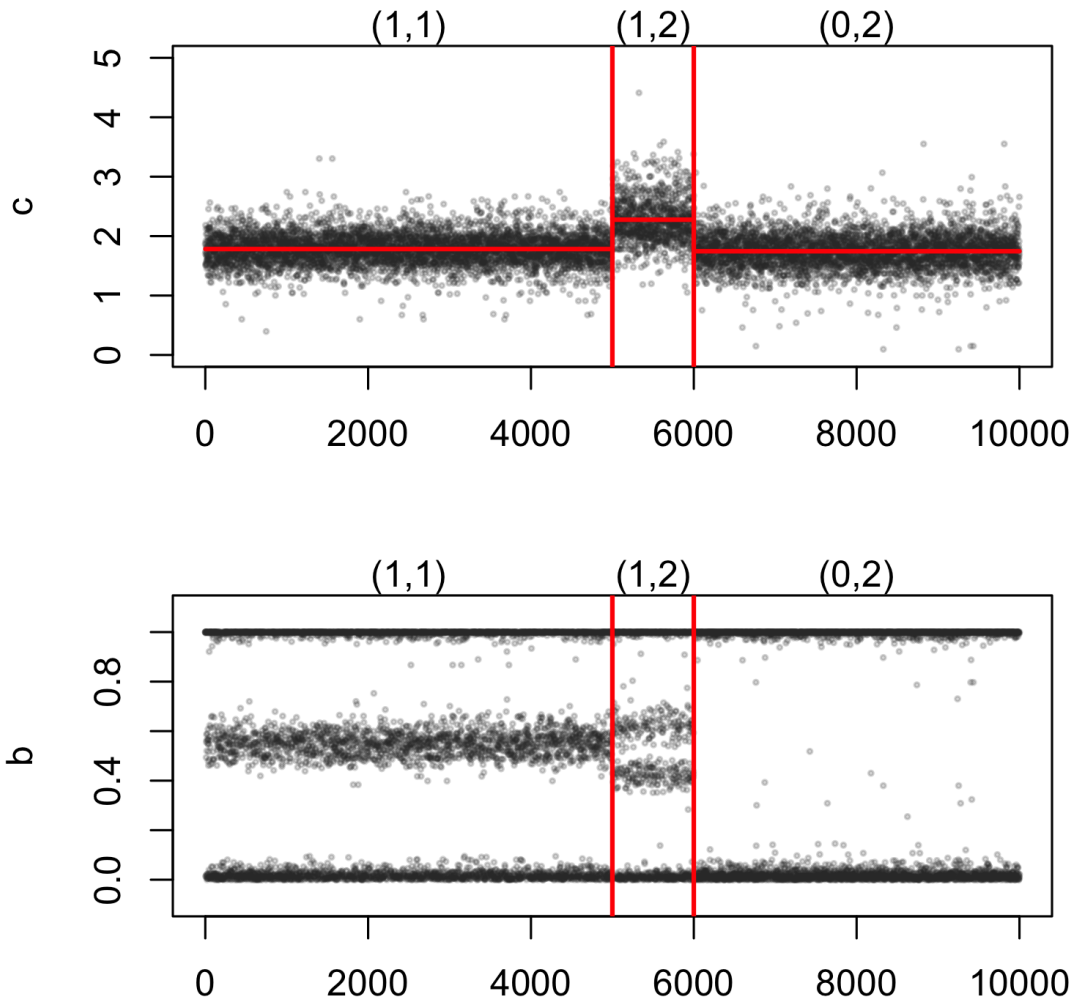
Exemple : data set 2, 50% de cellules tumorales



Exemple : data set 2, 79% de cellules tumorales



Exemple : data set 2, 100% de cellules tumorales



Utilisation de plusieurs méthodes de segmentation existantes via une interface unifiée

Types de méthodes existantes

- Approches exactes par programmation dynamique (`cghseg`) ;
- Segmentation binaire (`CBS`, `PSCBS`)
- Régression pénalisée de type fused Lasso (`GFLARS`, portage en R d'un code Matlab)
- Modèle de Markov caché (`PSCN`).

Méthode implémentée

- `RBS` for Recursive Binary Segmentation
  $\rightarrow$ combinaison de `CART` et la programmation dynamique.

Recursive Binary Segmentation (RBS)

Modèle statistique pour K ruptures aux positions $(t_1, ...t_K)$:
$\forall j \in [t_{k-1}, t_k[\qquad c_j = \gamma_j + \epsilon_j \quad$ où $\gamma_j = \Gamma_k$

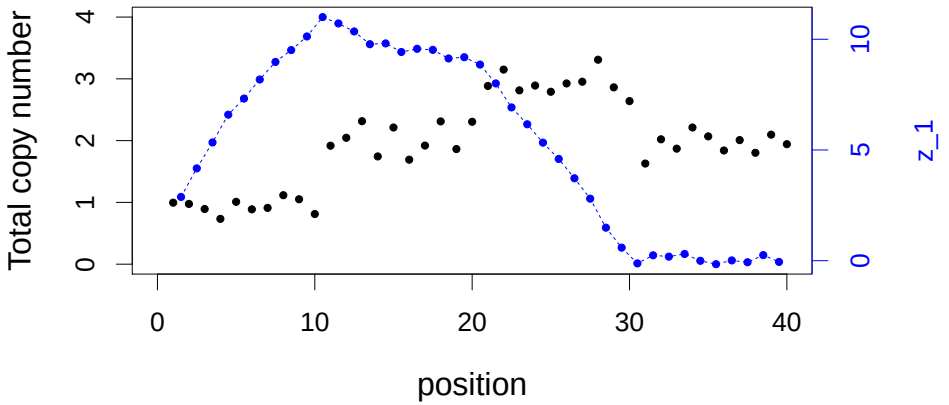
- Cas simple : dimension égale à $(d = 1)$:
- Hypothèse : $\mathcal{H}_0$: Pas de ruptures vs $\mathcal{H}_1$: Exactement une rupture.
- Statistique du rapport de vraisemblance : $\max_{1 \leq i \leq n} |Z_i|$

$$Z_i = \frac{\left(\frac{S_i}{i} - \frac{S_n - S_i}{n-i} \right)}{\sqrt{\frac{1}{i} + \frac{1}{n-i}}}, \qquad (1)$$

Et $S_i = \sum_{1 \leq l \leq i} c_l$

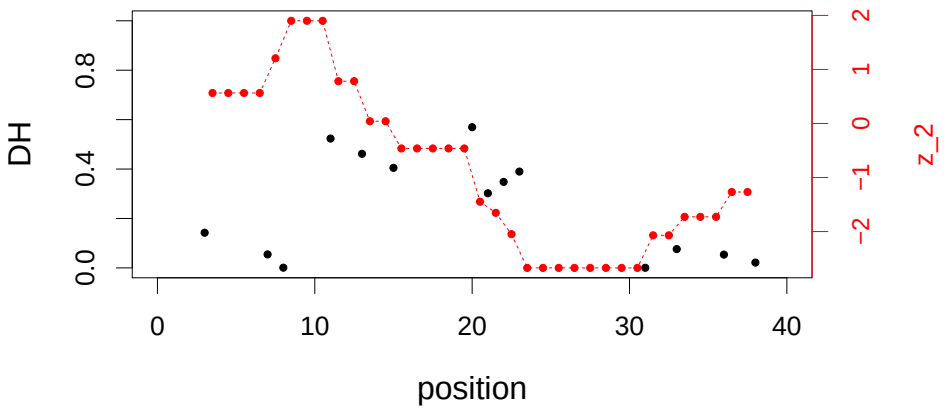
Si $(d > 1)$: statistique du rapport de vraisemblance devient $\max_{1 \leq i \leq n} \|Z_i\|_2^2$

Recursive Binary Segmentation (RBS)



Compléxité : $O(dnlog(K))$

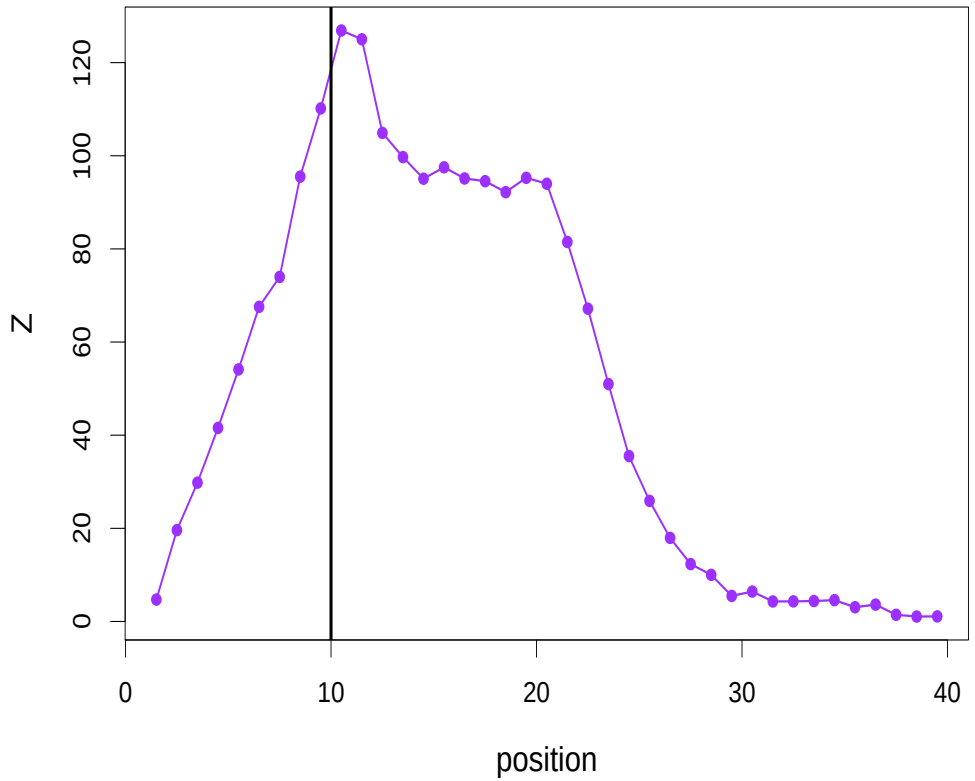
- Première rupture
- Pour chaque i : on calcule Z_i : $t_1 = \arg\max_{1 \leq i \leq n} \|Z_i\|_2^2$



Recursive Binary Segmentation (RBS)

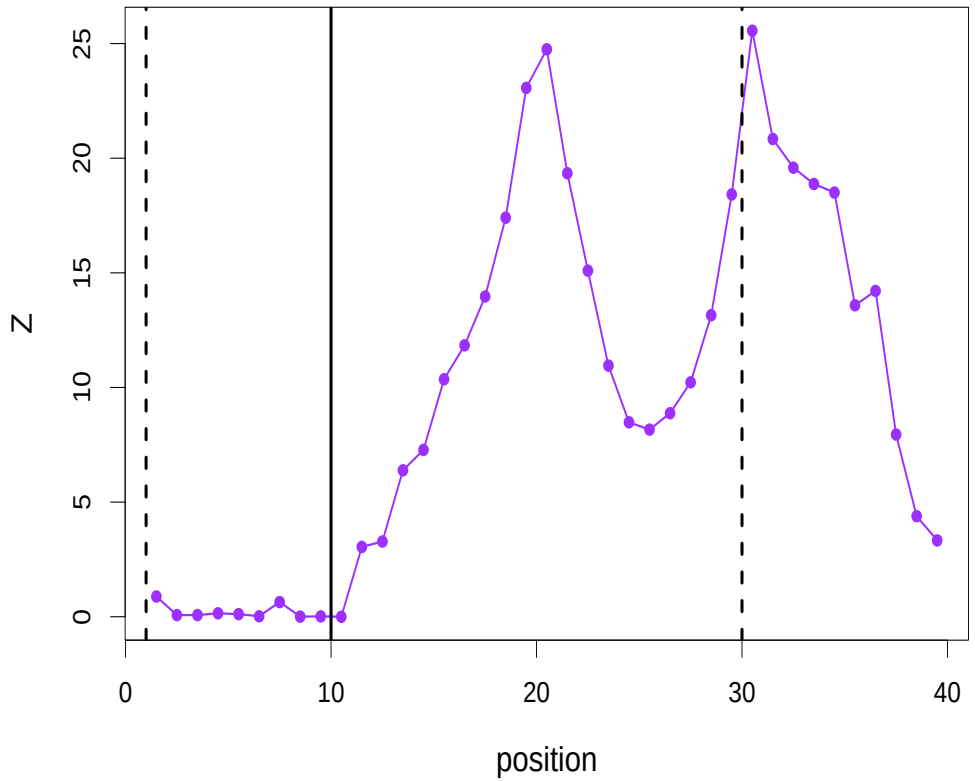
Compléxité : $O(dnlog(K))$

- Première rupture
- Pour chaque i : on calcule Z_i : $t_1 = \arg\max_{1 \leq i \leq n} \|Z_i\|_2^2$



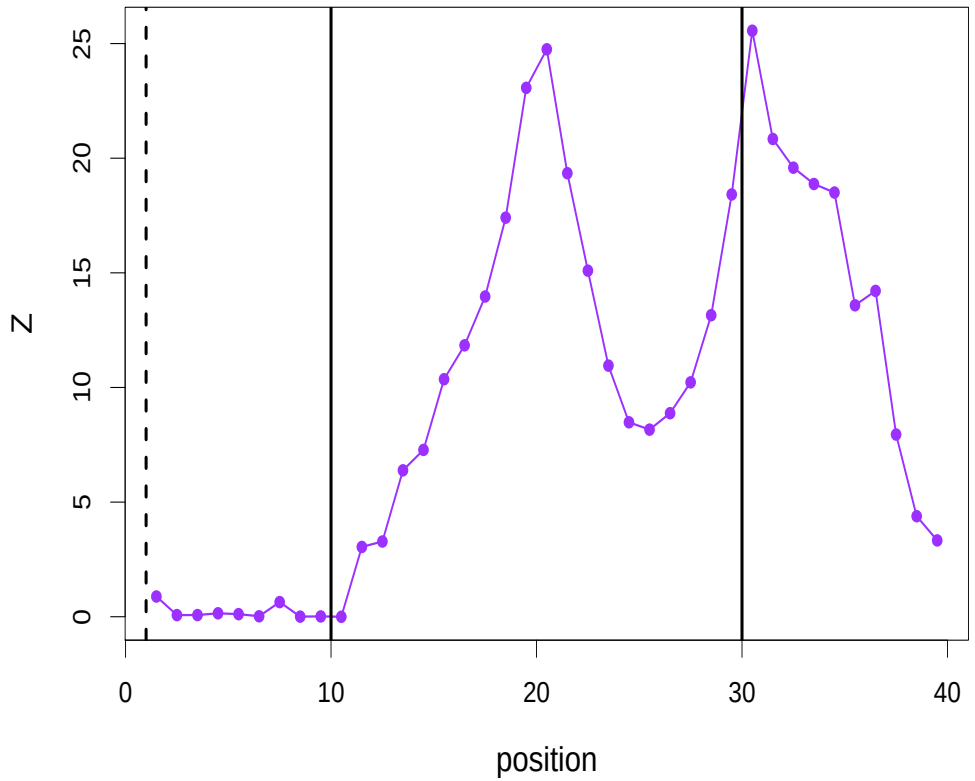
Recursive Binary Segmentation (RBS)

- Deuxième rupture :
  - $\max_{1 \leq i \leq t_1} \|Z_i\|_2^2$
  - $\max_{t_1 < i \leq n} \|Z_i\|_2^2$
- Calcul du RSE pour chaque segment.
- On garde le segment qui apporte le plus de gain au niveau du RSE
- Ajout de la rupture dans l'active set



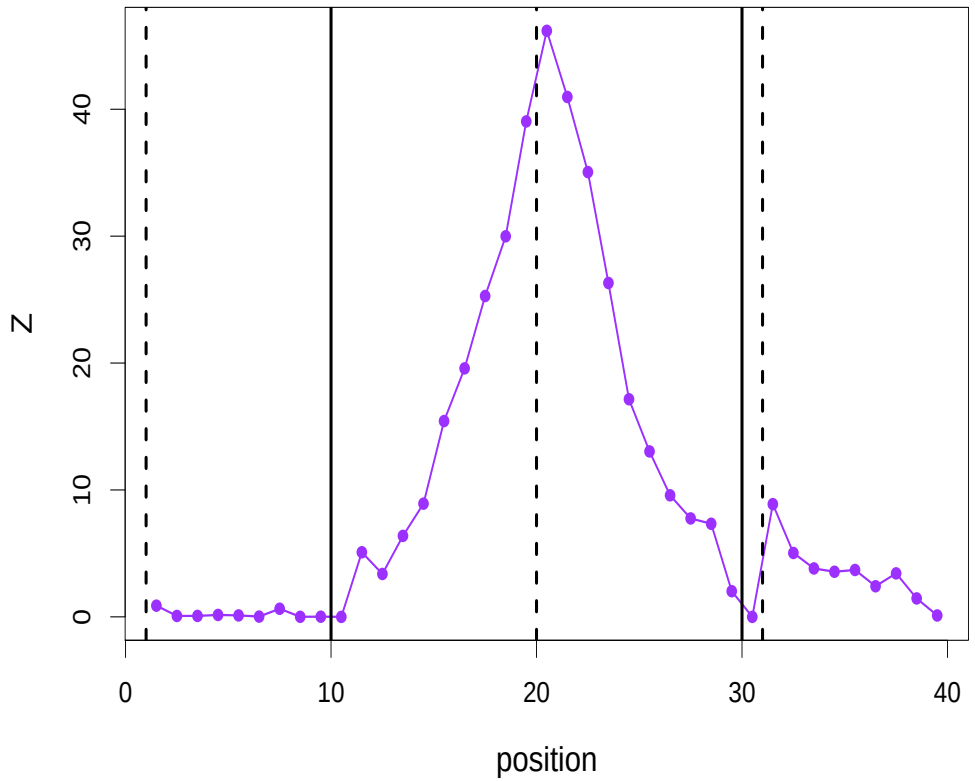
Recursive Binary Segmentation (RBS)

- Deuxième rupture :
  - $\max_{1 \leq i \leq t_1} \|Z_i\|_2^2$
  - $\max_{t_1 < i \leq n} \|Z_i\|_2^2$
- Calcul du RSE pour chaque segment.
- On garde le segment qui apporte le plus de gain au niveau du RSE
- Ajout de la rupture dans l'active set



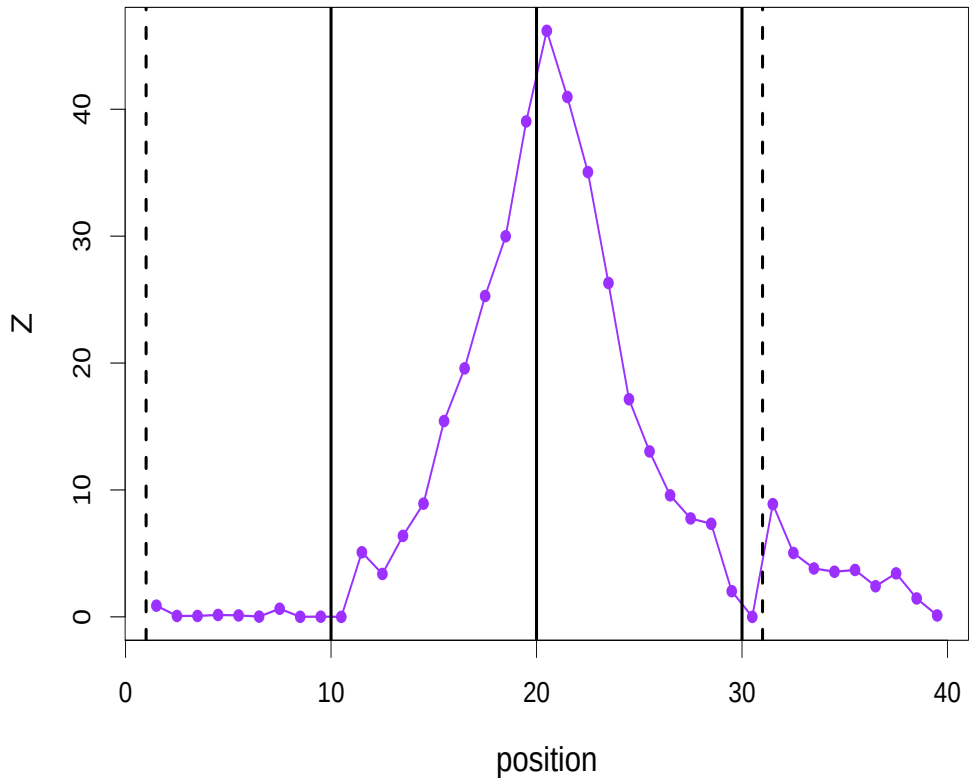
Recursive Binary Segmentation (RBS)

- Troisième rupture :
  - $\max_{1 \leq i \leq t_1} \|Z_i\|_2^2$
  - $\max_{t_1 < i \leq t_2} \|Z_i\|_2^2$
  - $\max_{t_2 < i \leq n} \|Z_i\|_2^2$
- Calcul du RSE pour chaque segment.
- On garde le segment qui apporte le plus de gain au niveau du RSE
- Ajout de la rupture dans l'active set



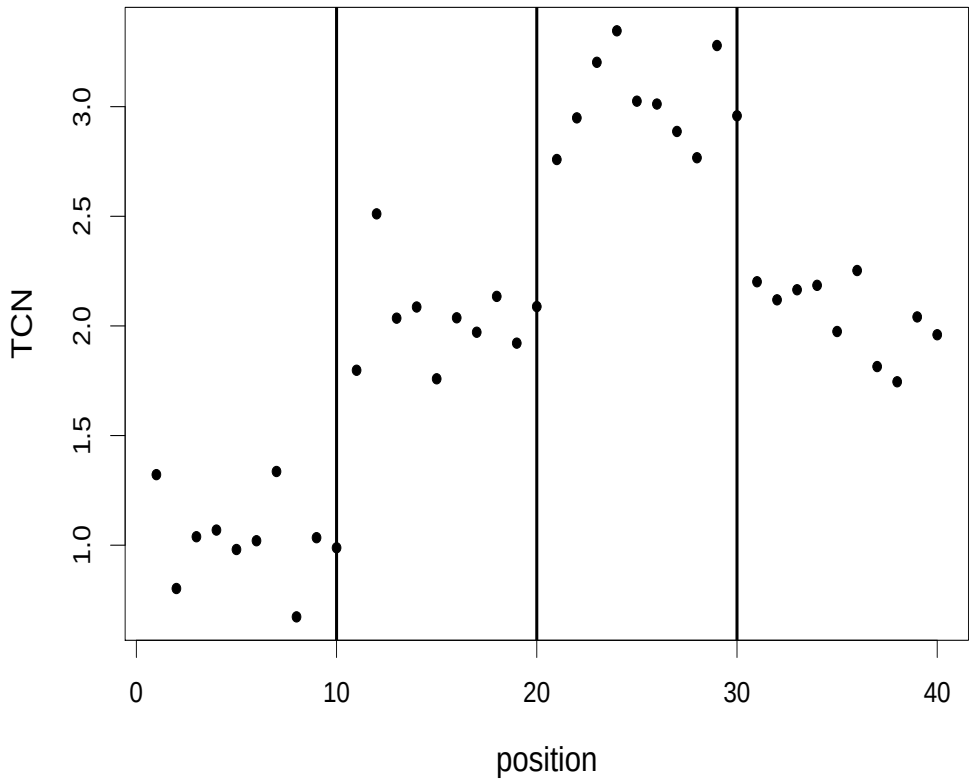
Recursive Binary Segmentation (RBS)

- Troisième rupture :
  - $\max_{1 \leq i \leq t_1} \|Z_i\|_2^2$
  - $\max_{t_1 < i \leq t_2} \|Z_i\|_2^2$
  - $\max_{t_2 < i \leq n} \|Z_i\|_2^2$
- Calcul du RSE pour chaque segment.
- On garde le segment qui apporte le plus de gain au niveau du RSE
- Ajout de la rupture dans l'active set



Recursive Binary Segmentation (RBS)

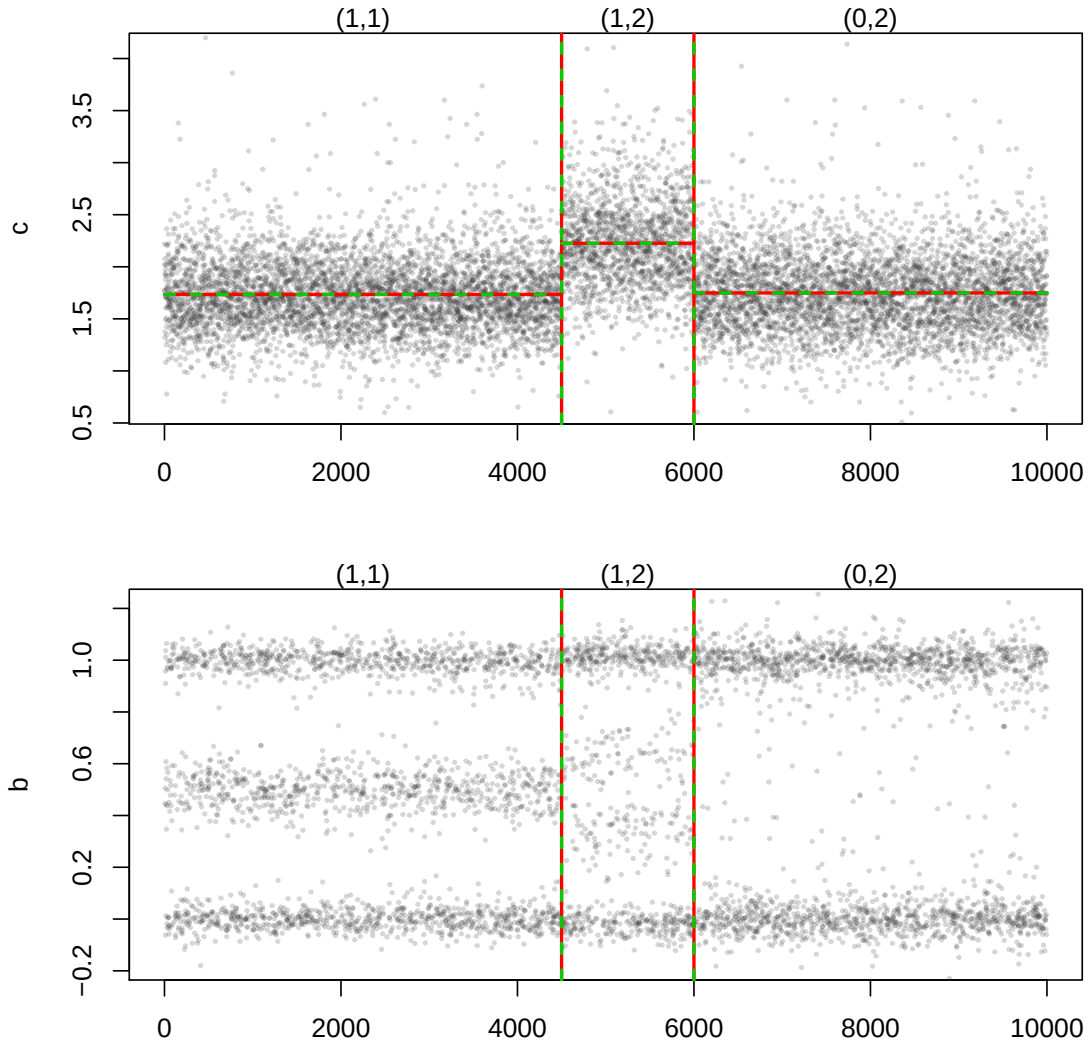
- Troisième rupture :
  - $\max_{1 \leq i \leq t_1} \|Z_i\|_2^2$
  - $\max_{t_1 < i \leq t_2} \|Z_i\|_2^2$
  - $\max_{t_2 < i \leq n} \|Z_i\|_2^2$
- Calcul du RSE pour chaque segment.
- On garde le segment qui apporte le plus de gain au niveau du RSE
- Ajout de la rupture dans l'active set



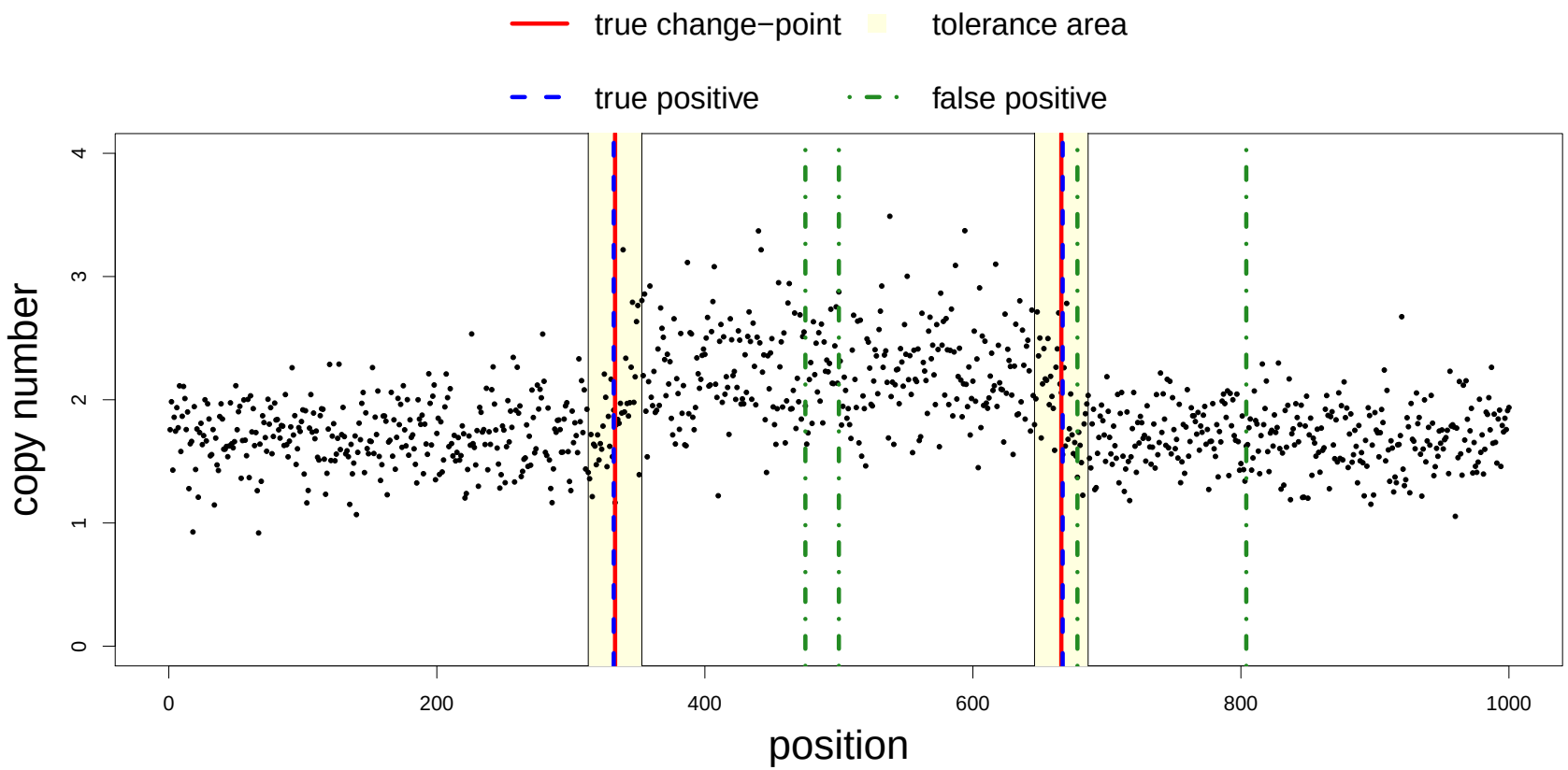
Structure des données et fonction de segmentation

```
 > resRBS <- PSSeg(data=sim$profile, K=20, flavor="RBS")
> resRBS$bestBkp
[1] 4500 6000
> rescghseg <- PSSeg(data=sim$profile, K=20, flavor="cghseg",
statistic='c') > rescghseg$bestBkp
[1] 4500 6000
> resPSCN <- PSSeg(data=sim$profile, flavor="PSCN")
[1] 4502 6001 9578
> resPSCBS <- PSSeg(data=sim$profile, flavor="PSCBS")
[1] 4500 6000
```

```
> plotSeg(sim$profile, list(true=sim$bkp,
est=resRBS$bestBkp))
```



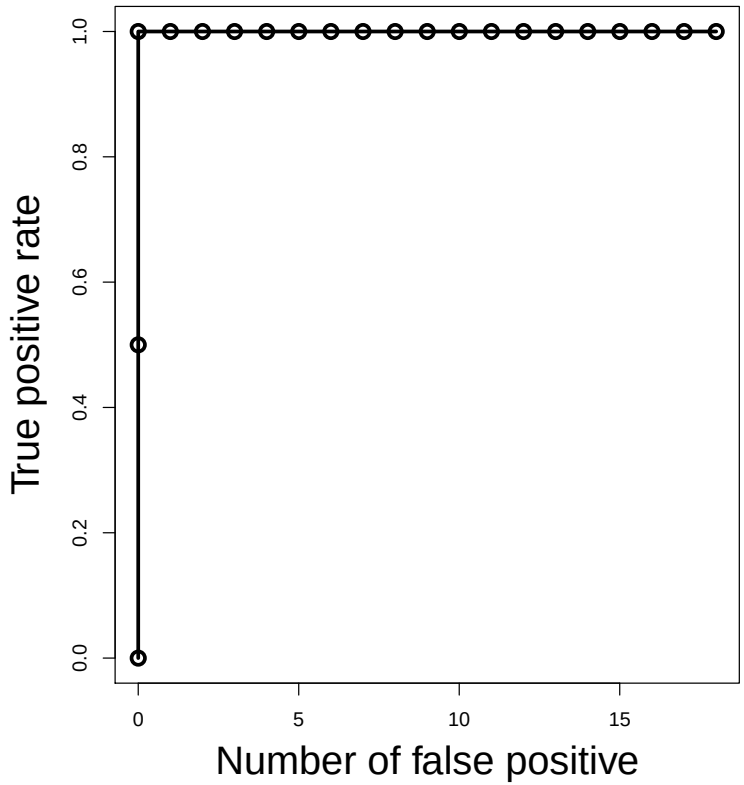
Evaluation : Vrais positifs, faux positifs




```
getTprTnr(candidates = resRBS$bestBkp, relax = -1, len = 10000,
tol = 2, trueBkp=sim$bkp)
```

Evaluation

```
tptn <- sapply(1:20, function(x){getTprTnr(candidates
= resRBS$initBkp[1:x], relax = -1, len = 10000, tol =
2, trueBkp=sim$bkp)})
```



Paramètre `tol`

- Largeur de la zone de tolérance

Paramètre `relax`

- relax = -1 : Ce qui a été présenté
- relax = 0 : Exactement une rupture dans la zone de tolérance
- relax = 1 : Au moins une rupture dans la zone de tolérance

Sommaire

Conclusion et perspectives

Résumé

- Mise en place de simulations à vérité connue
- 2 technologies disponibles (Affymetrix et Illumina)
- Paramètres biologiques interprétables
- Regroupement des différentes méthodes de segmentation
- Évaluation des performances des méthodes

`http://r-forge.r-project.org/R/?group_id=1562`

Merci à Pierre Neuvial, Guillem Rigaill et Cyril Dalmasso
Merci de votre attention