Spatial organization of genes along mammalian chromosomes

EPFL Bachelor Project – Computer Science – Spring 2016

Dario Anongba Varela

Supervisors : Saeed Omidi, Felix Naef, Jake Yeung
Lab : Computational Systems Biology Lab (Felix Naef)

Why we are doing this research

Goals of the research

01 **One:**
Genes coding for functional products are placed along each chromosome, the function of its specific positioning **being yet to be understood**

02 **Two:**
Are genes placed in **clusters** along the chromosomes ?

03 **Three:**
If they are, it would suggest a **functional role of the specific positioning** of genes along a chromosome

The idea

Partitioning algorithm

- How to identify the clusters and partition the genes efficiently ?
- From an O(2^{n-1}) problem to an O(N^2) using dynamic programming

Simple example

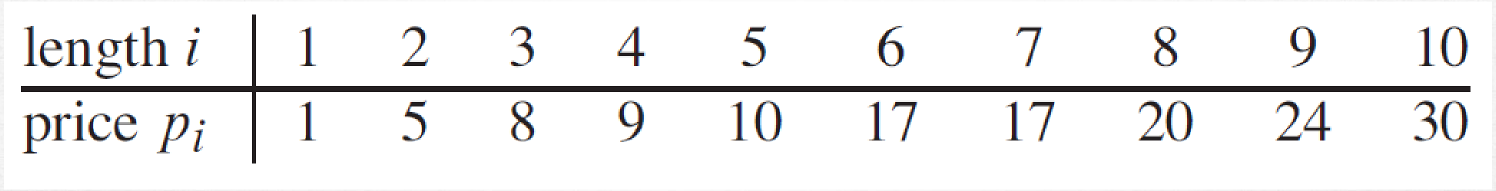
Partitioning algorithm

- A length **n** of a metal rod. Example for n = 4. So $\mathbf{2^3}$ = 8 possible solutions
- A table of prices p_i for rods of lengths **i** = 1,. . .,n

Simple example

Partitioning algorithm

- A length **n** of a metal rod. Example for n = 4. So $\mathbf{2^3}$ = 8 possible solutions
- A table of prices $\boldsymbol{p_i}$ for rods of lengths **i** = 1,…,n

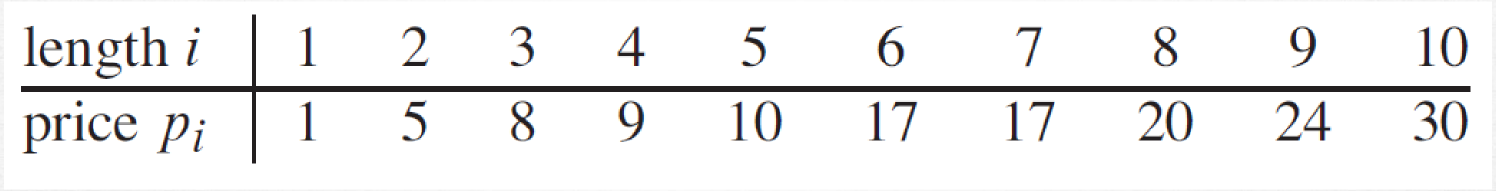


length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

Simple example

Partitioning algorithm

- A length **n** of a metal rod. Example for n = 4. So $\mathbf{2^3}$ = 8 possible solutions
- A table of prices $\boldsymbol{p_i}$ for rods of lengths **i** = 1,. . .,n



length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30

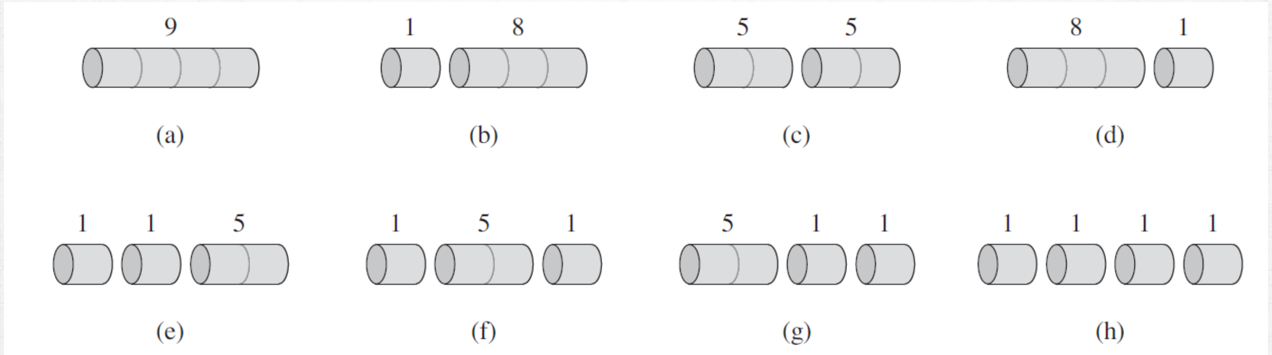
- **Objective** : Decide how to cut the rod into pieces and maximize / minimize the price.

Simple example

Partitioning algorithm

- A length **n** of a metal rod. Example for n = 4. So $\mathbf{2^3}$ = 8 possible solutions
- A table of prices p_i for rods of lengths **i** = 1,. . .,n

length i	1	2	3	4	5	6	7	8	9	10
price p_i	1	5	8	9	10	17	17	20	24	30



Simple example

Partitioning algorithm

- If an optimal solution cuts the rod into k pieces, for some 1 <= k <= n, then an optimal decomposition
- $n = i_1 + i_2 + \ldots + i_k$

Simple example

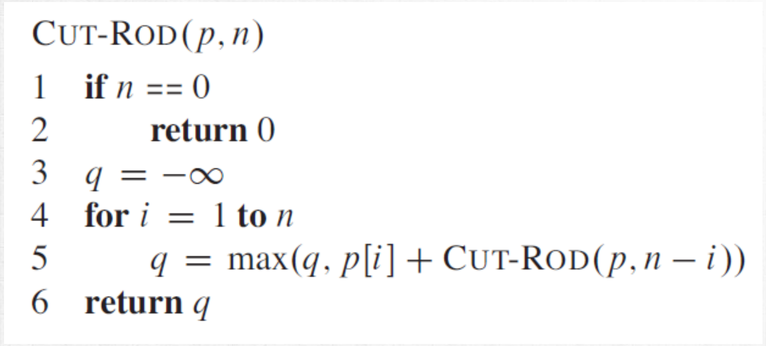
Partitioning algorithm

- If an optimal solution cuts the rod into k pieces, for some 1 <= k <= n, then an optimal decomposition
- $n = i_1 + i_2 + \dots + i_k$
- of the rod into pieces of lengths i1, i2, . . ., ik provides maximum revenue
- $r_n = p_{i_1} + p_{i_2} + \dots + p_{i_k} p_{i_1}$

Simple example

Partitioning algorithm

- If an optimal solution cuts the rod into k pieces, for some 1 <= k <= n, then an optimal decomposition
- $n = i_1 + i_2 + \dots + i_k$
- of the rod into pieces of lengths i1, i2, . . ., ik provides maximum revenue
- $r_n = p_{i_1} + p_{i_2} + \dots + p_{i_k} p_{i_1}$

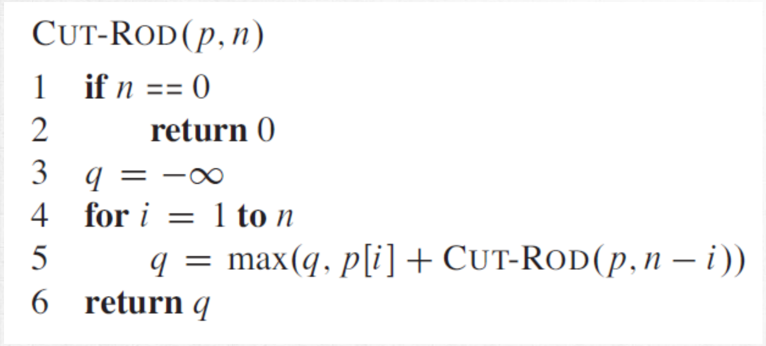


```
CUT-ROD(p, n)
1  if n == 0
2      return 0
3  q = −∞
4  for i = 1 to n
5      q = max(q, p[i] + CUT-ROD(p, n − i))
6  return q
```

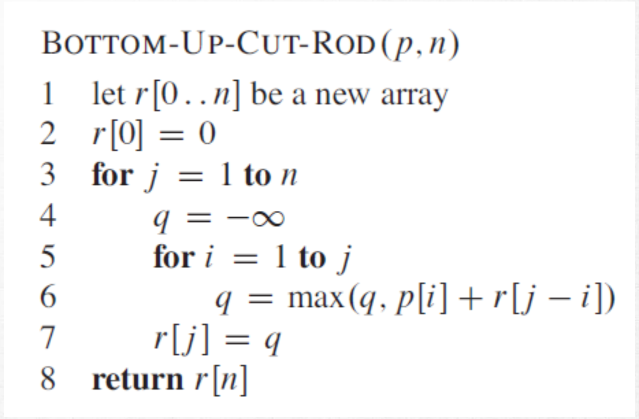
Simple example

Partitioning algorithm

- If an optimal solution cuts the rod into k pieces, for some 1 <= k <= n, then an optimal decomposition
- $n = i_1 + i_2 + \dots + i_k$
- of the rod into pieces of lengths i1, i2, . . ., ik provides maximum revenue
- $r_n = p_{i_1} + p_{i_2} + \dots + p_{i_k} p_{i_1}$



```
CUT-ROD(p, n)
if $n == 0$
    return 0
$q = -\infty$
for $i = 1$ to $n$
    $q = \max(q, p[i] + \text{CUT-ROD}(p, n - i))$
return $q$
```

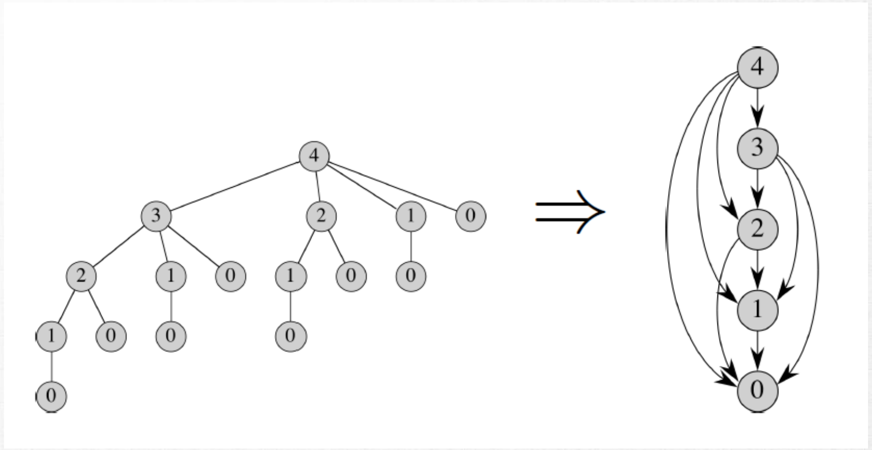


```
BOTTOM-UP-CUT-ROD(p, n)
let $r[0..n]$ be a new array
$r[0] = 0$
for $j = 1$ to $n$
    $q = -\infty$
    for $i = 1$ to $j$
        $q = \max(q, p[i] + r[j - i])$
    $r[j] = q$
return $r[n]$
```

Simple example

Partitioning algorithm

- If an optimal solution cuts the rod into k pieces, for some 1 <= k <= n, then an optimal decomposition
- $n = i_1 + i_2 + \ldots + i_k$
- of the rod into pieces of lengths i1, i2, . . ., ik provides maximum revenue
- $r_n = p_{i_1} + p_{i_2} + \ldots + p_{i_k} p_{i_1}$



Simple example

Partitioning algorithm

- Algorithm only returns the optimal revenue, but we want the whole solution path.

Simple example

Partitioning algorithm

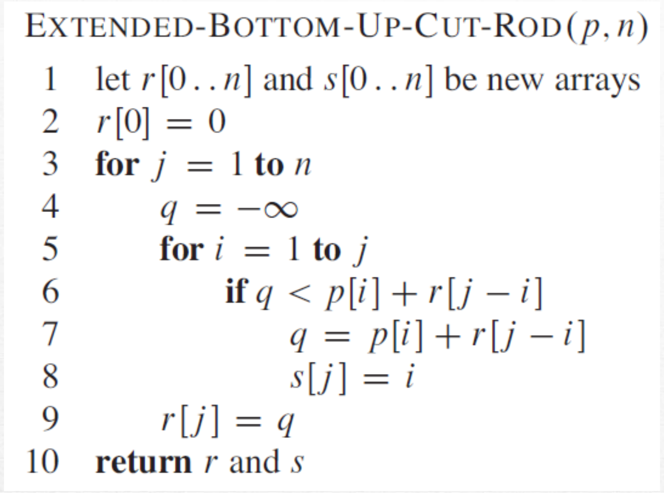
- Algorithm only returns the optimal revenue, but we want the whole solution path.
- Approach : Each cell of the stored table corresponds to a decision: the location of the leftmost cut. Store the decision corresponding to every cell in a separate table

Simple example

Partitioning algorithm

- Algorithm only returns the optimal revenue, but we want the whole solution path.

- **Approach** : Each cell of the stored table corresponds to a decision: the location of the leftmost cut. Store the decision corresponding to every cell in a separate table

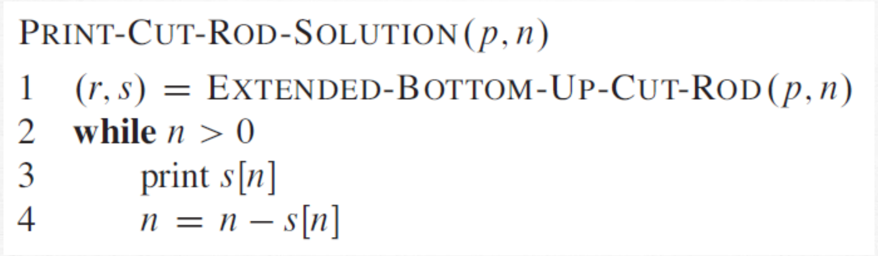


```
EXTENDED-BOTTOM-UP-CUT-ROD$(p, n)$
let $r[0..n]$ and $s[0..n]$ be new arrays
$r[0] = 0$
for $j = 1$ to $n$
    $q = -\infty$
    for $i = 1$ to $j$
        if $q < p[i] + r[j-i]$
            $q = p[i] + r[j-i]$
            $s[j] = i$
    $r[j] = q$
return $r$ and $s$
```

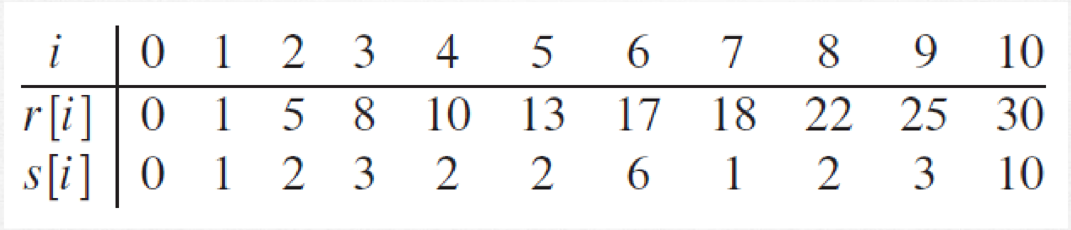
Simple example

Partitioning algorithm

- Algorithm only returns the optimal revenue, but we want the whole solution path.
- **Approach** : Each cell of the stored table corresponds to a decision: the location of the leftmost cut. Store the decision corresponding to every cell in a separate table



```
PRINT-CUT-ROD-SOLUTION(p, n)
1  (r, s) = EXTENDED-BOTTOM-UP-CUT-ROD(p, n)
2  while n > 0
3      print s[n]
4      n = n - s[n]
```



i	0	1	2	3	4	5	6	7	8	9	10
$r[i]$	0	1	5	8	10	13	17	18	22	25	30
$s[i]$	0	1	2	3	2	2	6	1	2	3	10

Real partitioning

Partitioning algorithm

We are actually doing quite the same with our RNA-Seq data

- Where **n** is the number of genes of a chromosome we want to partition
- Genes are ordered in order to follow a cluster logic

- We try to minimize the score instead of maximizing it

Partitioning algorithm

We are actually doing quite the same with our RNA-Seq data

- Where **n** is the number of genes of a chromosome we want to partition
- Genes are ordered in order to follow a cluster logic

- We try to minimize the score instead of maximizing it

What is the equivalent to the **prices** of the rod cutting algorithm ?

Real partitioning

Model selection

- The «price» is computed for each block of genes by :
- $\min\{score\ of\ model\ 1,\ score\ of\ model\ 2\}$
- Where the score is the residue (square of the errors) + a penalty.
- Model 1 is the flat model and model 2 is a circadian (rythmic model) :

- Residue of a block : $\sum_{i=1}^{n}(M_i - \hat{Y}_{ji})^2$, for j = 1, 2 (for model 1 and 2), M is the matrix of RNA-Seq data and $\hat{Y}_j$ is the predictor matrix

Real partitioning

Model selection

- The «price» is computed for each block of genes by :
- $\min\{score\ of\ model\ 1,\ score\ of\ model\ 2\}$
- Where the score is the residue (square of the errors) + a penalty.
- Model 1 is the flat model and model 2 is a circadian (rythmic model) :

- Residue of a block : $\sum_{i=1}^{n}(M_i - \hat{Y}_{ji})^2$, for j = 1, 2 (for model 1 and 2), M is the matrix of RNA-Seq data and $\hat{Y}_j$ is the predictor matrix.

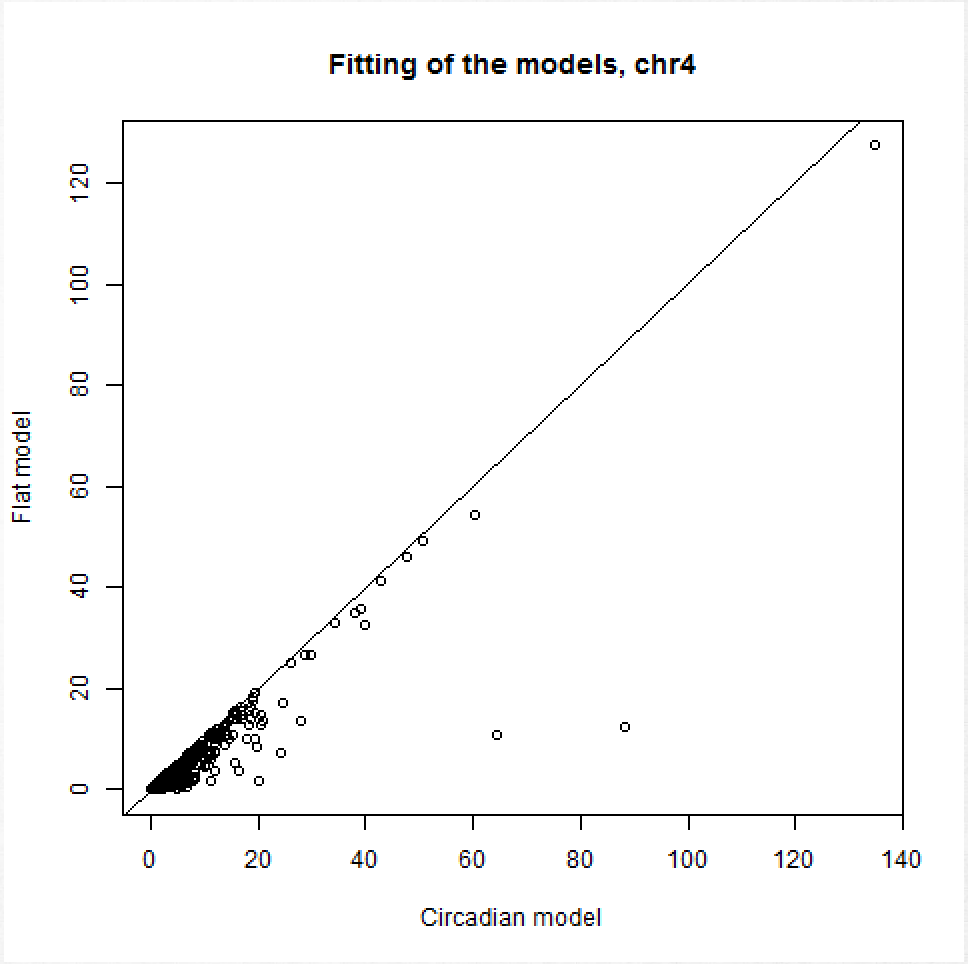
- Flat model : $\hat{Y} = mean(M_i)$

- Circadian model : $\hat{Y} = \alpha \sin(2\pi f) + \beta \sin(2\pi f)$

Real partitioning

Model selection

Circadian model always better or equal than the flat model !



Model selection

- Have to introduce a penalty : Score = residue + penalty

- Penalty = $\sigma^2 \; * (k+1) \; * \log(N)$
- Where **m** is the number of parameters, **N** the total number of data points and $\boldsymbol{\sigma}$ a customizable value.

Real partitioning

Model selection

- Have to introduce a penalty : Score = residue + penalty

- Penalty = $\sigma^2 \; * (k+1) \; * \log(N)$
- Where **m** is the number of parameters, **N** the total number of data points and $\boldsymbol{\sigma}$ a customizable value.

- For example, for chromosome 7 containing 1139 chromosomes and 24 time points, the penalty for model 2 would be:
- Penalty = $\sigma^2 \; * \left((1139+2)+1\right) \; * \log(1139 \; * 24)$

Real partitioning

Model selection

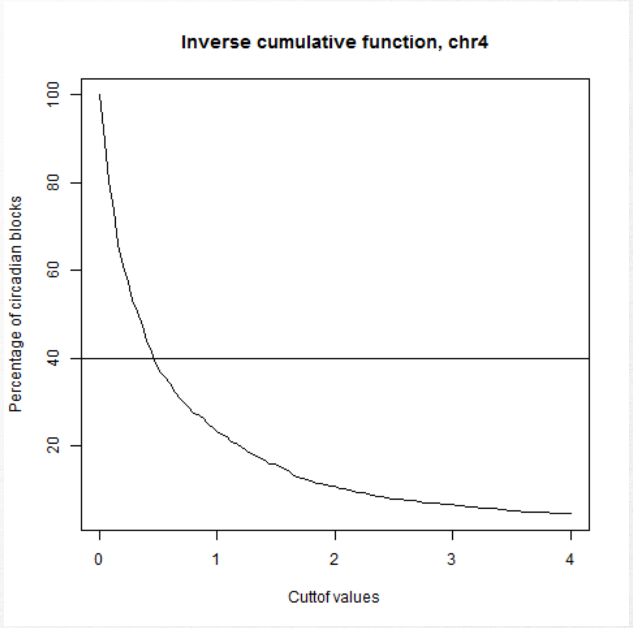
- Instead of using an abstract sigma value, we compute a value of sigma given **the percentage of expected circadian blocks**

- We use the inverse cumulative function to find it :

Real partitioning

Model selection

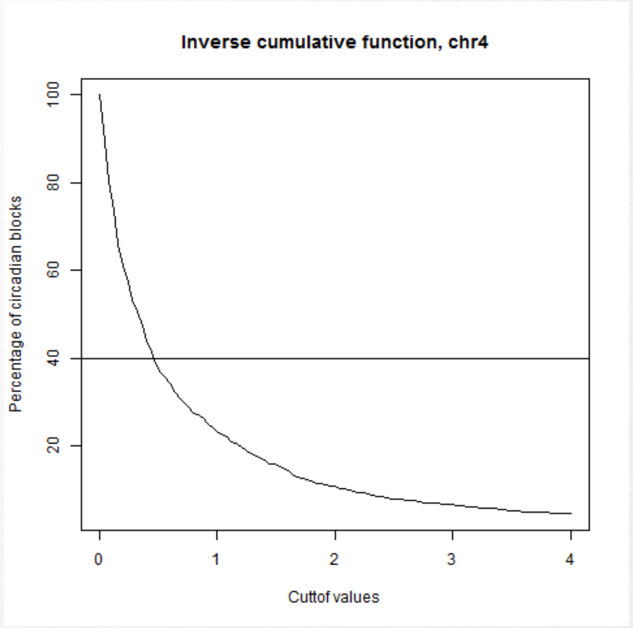
- Instead of using an abstract sigma value, we compute a value of sigma given **the percentage of expected circadian blocks**
- We use the inverse cumulative function to find it :



Real partitioning

Model selection

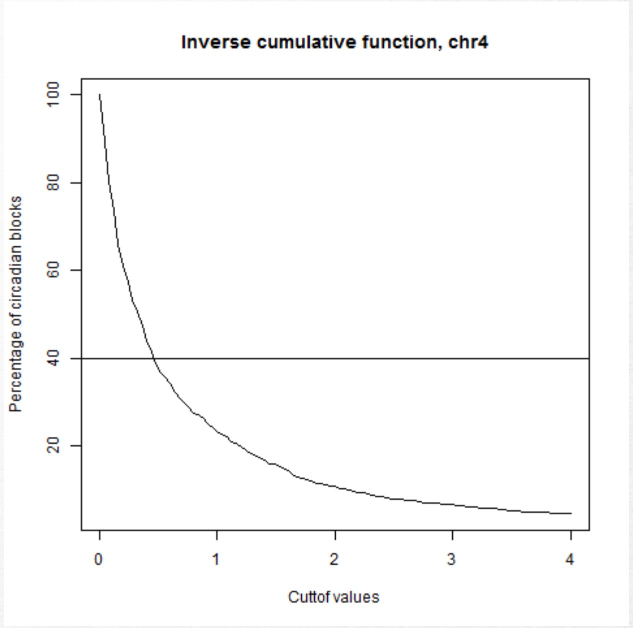
- Instead of using an abstract sigma value, we compute a value of sigma given **the percentage of expected circadian blocks**
- We use the inverse cumulative function to find it :
- **Cuttof** = quantile(res.1 – res.2, 1 – x),
- Where **res.1** and **res.2** are the residues of the flat model and circadian model and **x** is the expected percentage of circadian blocks



Real partitioning

Model selection

- Instead of using an abstract sigma value, we compute a value of sigma given **the percentage of expected circadian blocks**
- We use the inverse cumulative function to find it :
- **Cuttof 40%** = 0.4621029
- $\sigma^2 = {}^{cuttof}/_{2 * (\log(N))}$ = 0.02289724



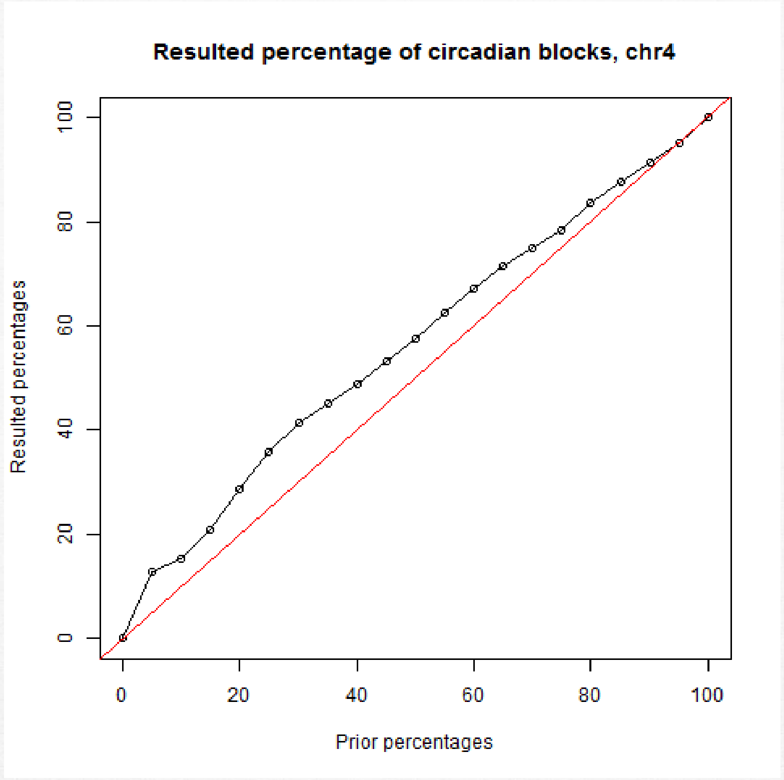
Complete algorithm

Pipeline of computation

Step 1	Input the RNA-Seq data and the parameters
Step 2	Compute the residue of models (this is stored)
Step 3	Compute the cuttof and σ^2
Step 4	Run the algorithm with the given and computed parameters

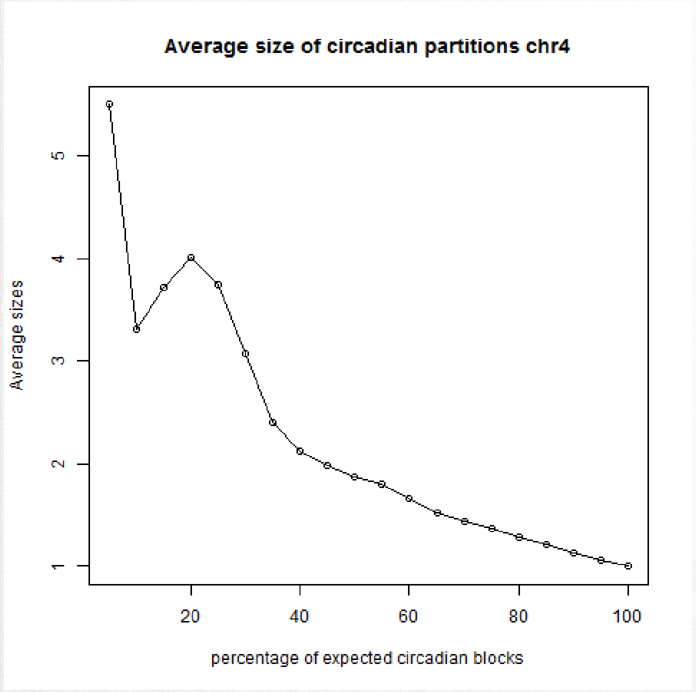
Results

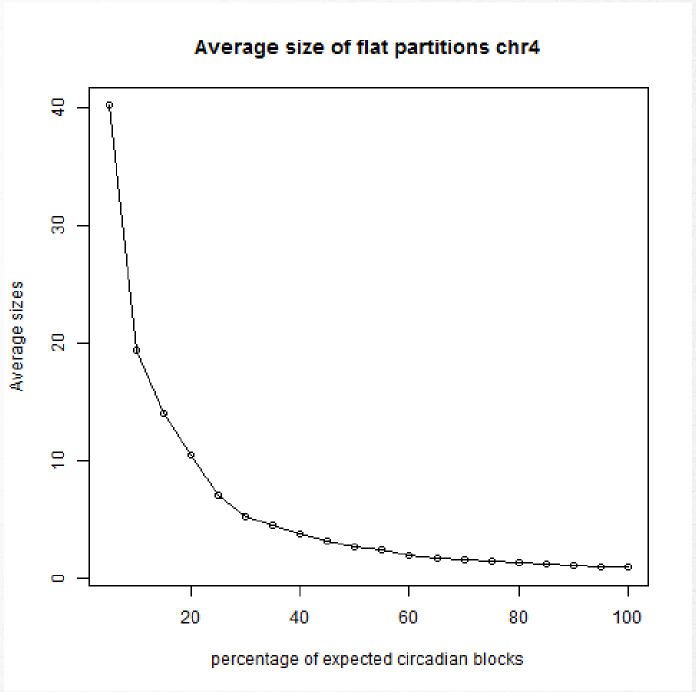
Resulted percentages of circadian blocks



Results

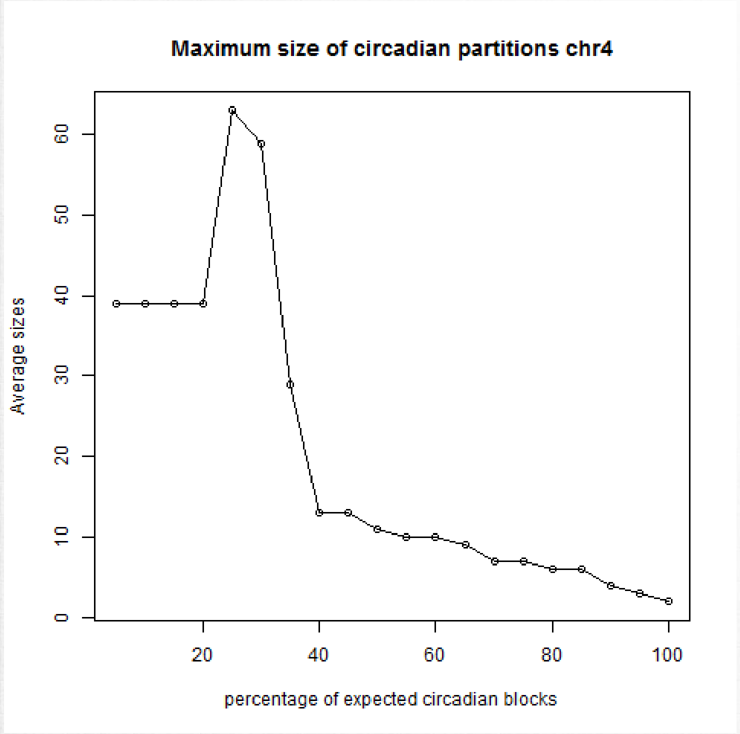
Average size of partitions

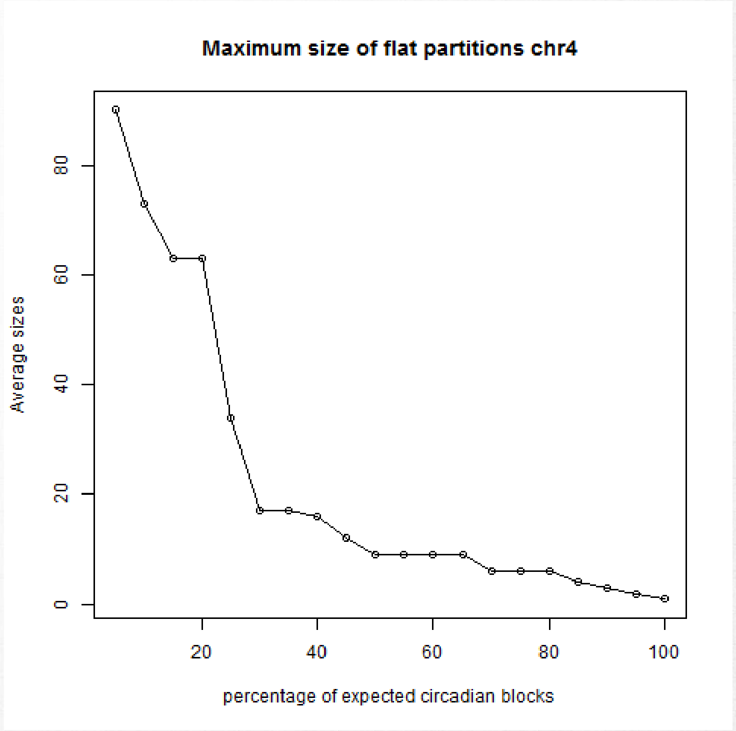




Results

Maximum sizes of partitions





Results

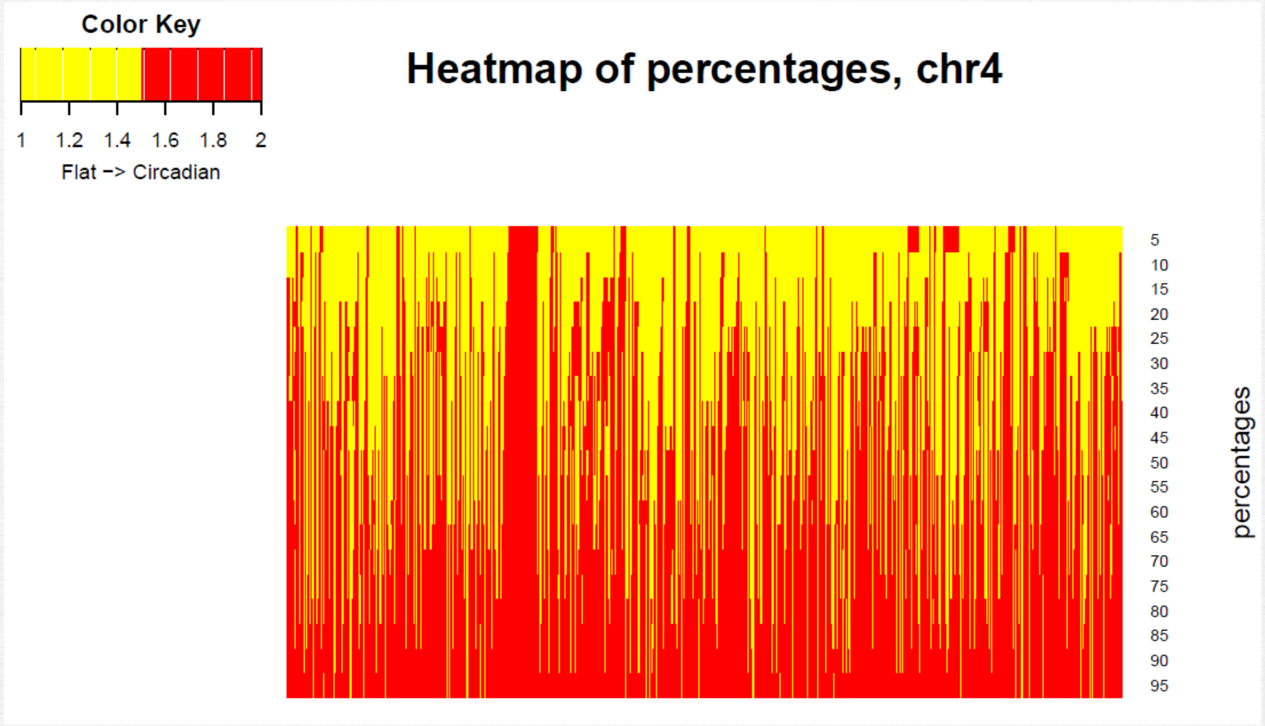
Do clusters of genes actually exist ?

- We ran the algorithm for chromosomes with randomly permuted genes
- 50 times each

- Comparison of sizes of circadian partitions

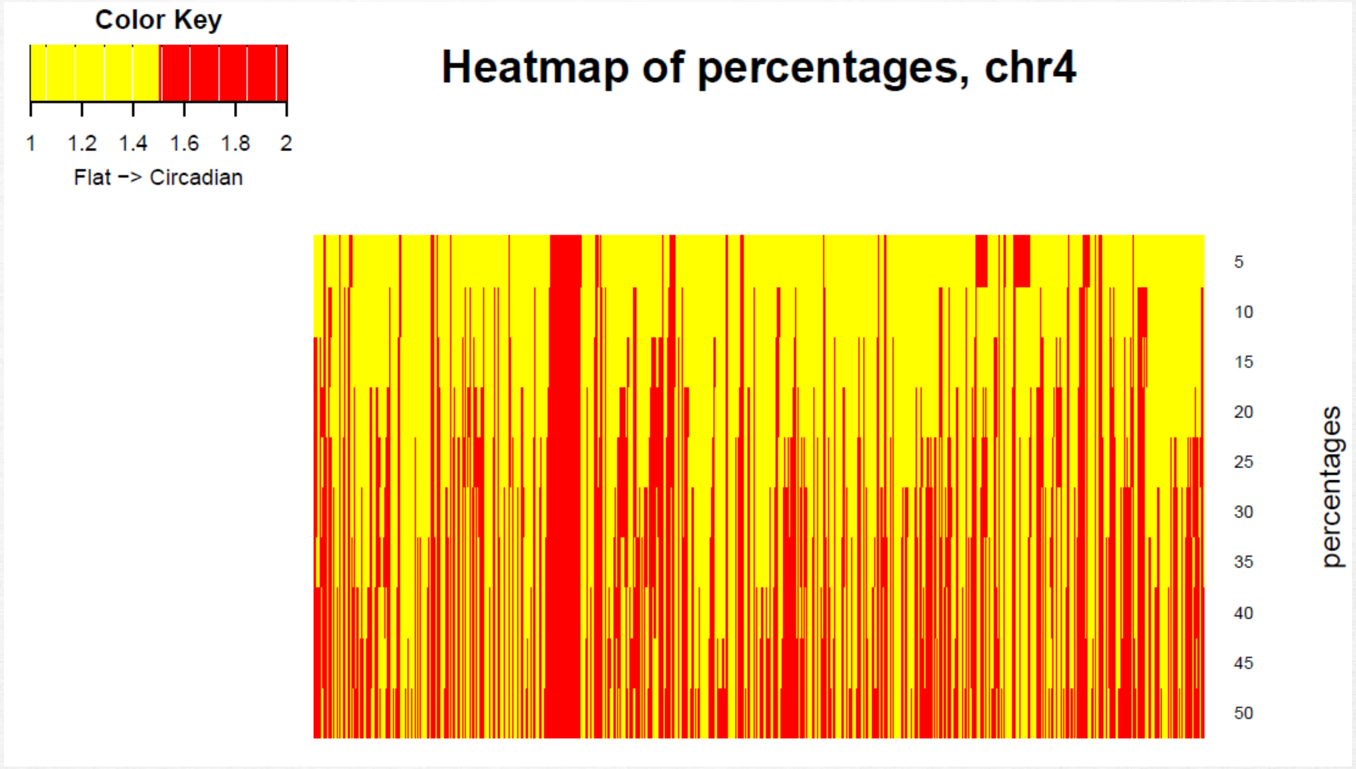
Results

Heatmap percentages 5 to 95



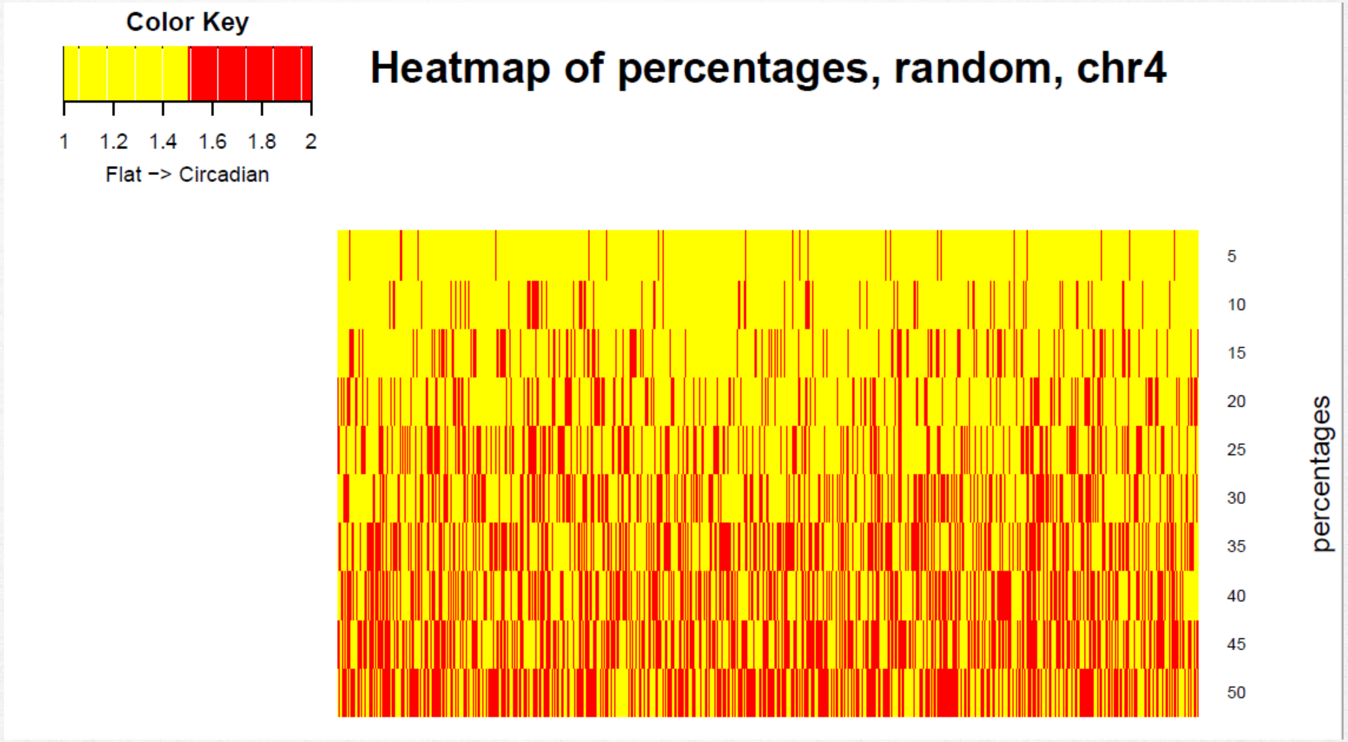
Results

Heatmap percentages 5 to 50



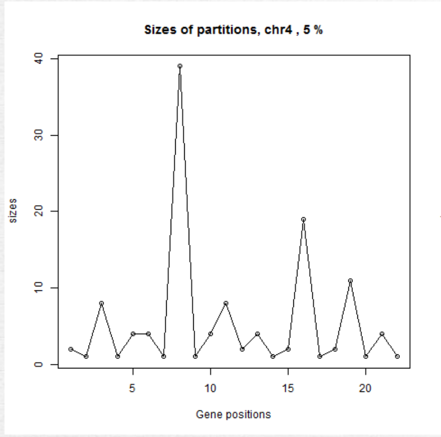
Results

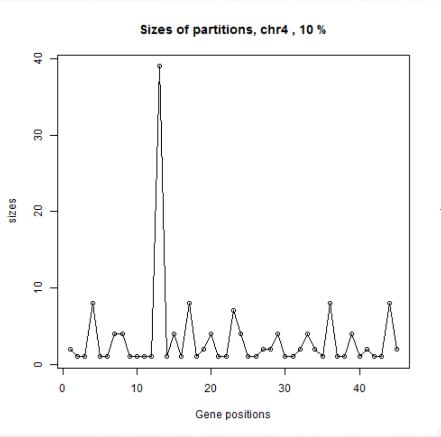
Heatmap randomized chromosome

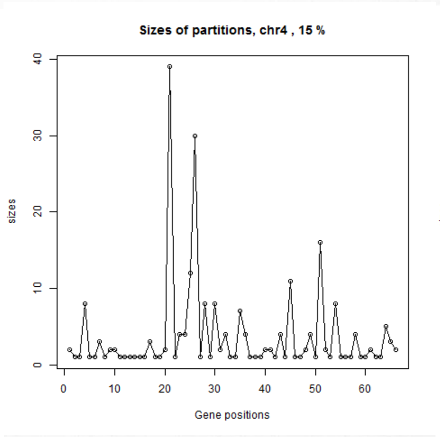


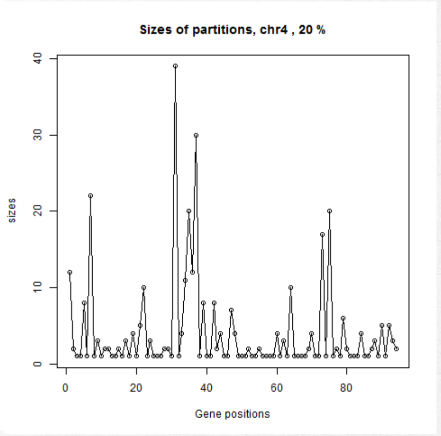
Results

Do clusters of genes actually exist ?



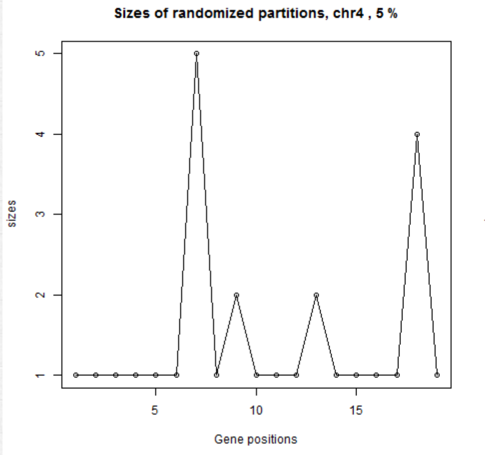


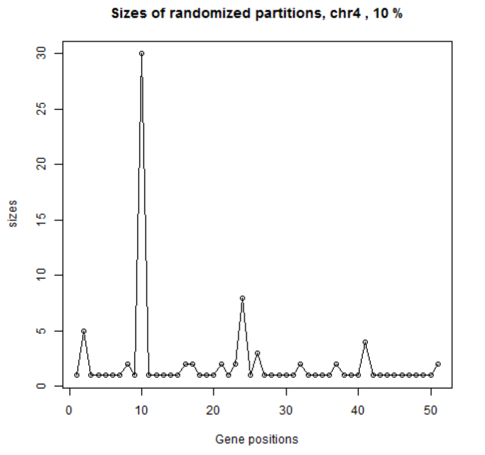


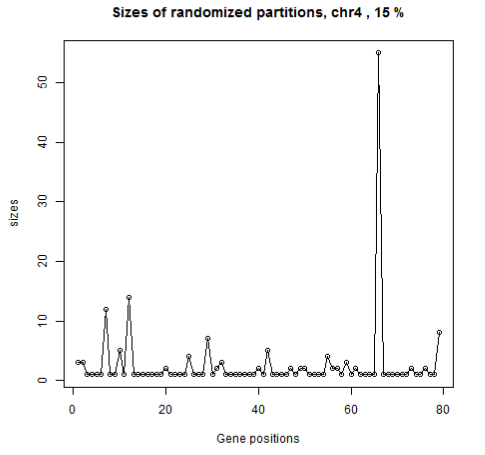


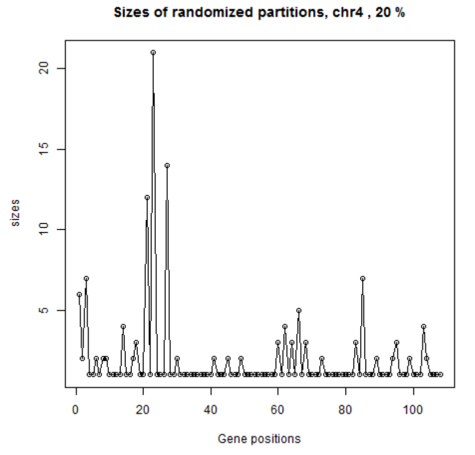
Results

Do clusters of genes actually exist ?









A final review

CONCLUSIONS

- Seems like the positioning is not random (heatmap)
- But the partitioning seems doesn’t seem optimal → Penalty is the problem ?
- Seems Genes are actually placed in clusters along the chromosome

A final review

FUTURE RESEARCH

- Find a better way to penalize models
- Increase number of models
- Use different data with to do the partitioning (we can prove that genes are positioned in clusters using different data)
- Use the randomized data to prove that clusters do exist

THANK YOU