DOMAIN FINDING ALGORITHM

CSE 549 Computational Biology

Link to GitHub: https://github.com/mdaliejaz/domainFindingAlgorithms.git

Muhammad Ali Ejaz	<110559131>	Kumar Anupam	<110614410>
Supriya Pramod Deshpande	<110369866>	Neha Sudhakar Ellur	<110468264>

Under the guidance of: Dr. Robert Patro

1. INTRODUCTION

A chromosome found in nucleus of a cell. It is an organized structure of DNA and protein, a single piece of coiled DNA containing many genes, regulatory elements and other nucleotide sequences. Chromosomes also contain DNA-bound proteins, which serve to package the DNA and control its functions. Chromosomes vary widely between different organisms.

The structure of the chromosomes has a deep impact on the cell development and cycle. It defines the chromosomal territories, which are known to be arranged radially around the nucleus. It is known to directly correlate with the gene density and size. Through multiple different studies, numerous spatial interactions between neighboring chromosome territories have been found, which can help in studying the influence of chromosome conformation on the functioning of the cells.

2. MAJOR DATA STRUCTURES USED

1. Y – The input matrix
$$Y_{i,j} \sim p(\cdot, \mu_{i,j}), \qquad \mu_{i,j} = E(Y_{i,j})$$
2. T – A temporary data structure to store the means that is used to compute the domain and non-domain region matrices.
3. D – Matrix to compute the domain regions.
4. R – Matrix to compute the non-domain regions
5. delta – Matrix to calculate data for : $\sum_{(i,j \in D_k} l_k(Y_{i,j})$
6. exterior – Matrix to calculate data for : $\sum_{(i,j \in R_k} l_0(Y_{i,j})$
7. I - The matrix which computes the maximum over the sum of delta and exterior.

Given by:

$$C(t_{k-1}, t_k - 1) = \sum_{(i,j \in D_k} l_k(Y_{i,j}) + \sum_{(i,j \in R_k} l_0(Y_{i,j}))$$

$$I_L(\tau) = \max_{1 = t_0 < t_1 < \cdots < t_L = \tau + 1} \sum_{k=1}^{L} C(t_{k-1}, t_k - 1)$$

3. SYSTEM OVERVIEW

a. To compile the code (in the domainFindingAlgorithms folder)

```
R CMD INSTALL HiCseg
```

b. To run the code

i) To load the parameters:

```
library(HiCseg)
```

ii) To load the data

```
data(matrix)
dim=dim(matrix)
n=dim[1]
image(1:n,1:n,matrix,xlab="",ylab="")
```

We form a symmetric matrix of size 200 x 200 with the change points on the x-axis And the log likelihood on the y-axis.

iii) To form the link between C and R

```
result = HiCseg_linkC_R(size_mat, nb_change_max, distrib, mat_data, model)
```

The arguments are defined as below:

- size_mat: Size of the matrix
- nb_change_max: Maximum number of change points
- distrib: The distribution of the data.
  The values can be:
  - **G**: Gaussian Distribution
  - **P**: Poisson Distribution
  - **B**: Negative Binomial Distribution
- mat_data: Data of the matrix
- model: Type of the model
  - **D** – Block Diagonal Matrix
  - **Dplus** – Extended block diagonal Matrix

c. From R

The command that forms the link between C and R internally calls the following C function:

```
tmp=.C("Function_HiC_R",as.integer(size_mat),as.integer(nb_change_max),
          as.character(distrib), as.double(as.vector(mat_data)),
          t_hat=as.integer(rep(0,nb_change_max)),
          J=as.double(rep(0.0,nb_change_max)),
          t_est=as.integer(rep(0,K)),as.character(model))
```

where,

t_hat: Contains the estimated change-points

J: Gives the values of the log-likelihood for different number of change-points up to some constants

t_est_mat: Gives the matrix of the estimated change-points for different possible number of change-points i.e when there is no change point, one change-point, two change-points and so on.

The corresponding function called is:

```
int Function_HiC_R(int *size, int *maximum_no_change_points,
        char **distribution, double *matrix, int *out_t_hat,
        double *out_log_likelihood, int *out_est_chng_pt, char * *model_type)
```

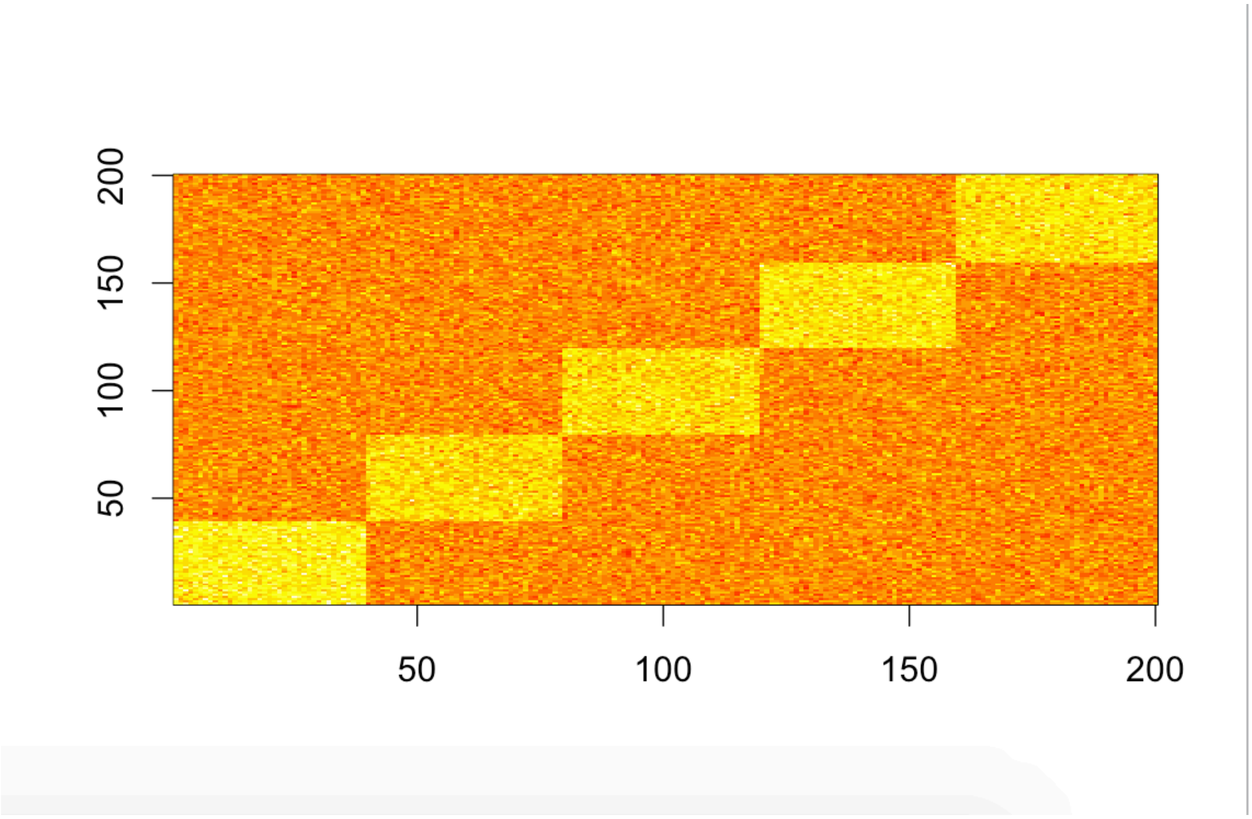
and the output is populated in:

```
out_t_hat
out_log_likelihood
out_est_chng_pt
```

4. OUTPUT

The following are the compilation steps (for each distribution) and they render a framework graph as follows:

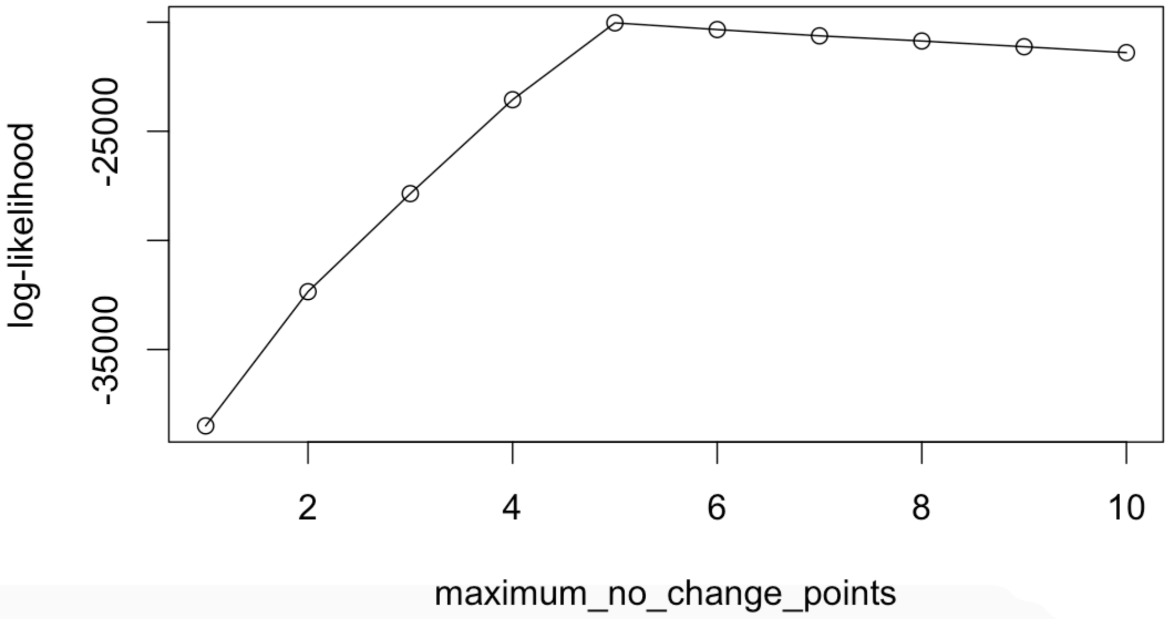
```
> library(HiCseg)
> data(matrix)
> dim=dim(matrix)
> n=dim[1]
> image(1:n,1:n,matrix,xlab="",ylab="")
> library(HiCseg)
> data(matrix)
> dim=dim(matrix)
> n=dim[1]
> image(1:n,1:n,matrix,xlab="",ylab="")
```



3.1 The output matrix for Gaussian with D:

```
> result = HiCseg_linkC_R(200, 10, "G", matrix, "D")
> result$t_est_mat
      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
 [1,]  200    0    0    0    0    0    0    0    0     0
 [2,]   39  200    0    0    0    0    0    0    0     0
 [3,]   39  159  200    0    0    0    0    0    0     0
 [4,]   39  119  159  200    0    0    0    0    0     0
 [5,]   39   79  119  159  200    0    0    0    0     0
 [6,]   39   41   79  119  159  200    0    0    0     0
 [7,]   39   75   77   79  119  159  200    0    0     0
 [8,]   39   73   75   77   79  119  159  200    0     0
 [9,]   39   71   73   75   77   79  119  159  200     0
[10,]   39   41   71   73   75   77   79  119  159   200
> result$J
 [1] -38499.61 -32349.69 -27856.21 -23552.36 -20031.73 -20339.39 -
20626.68 -20862.68
 [9] -21125.81 -21396.09
> result$t_hat
 [1]  39  79 119 159 200   0   0   0   0   0
```

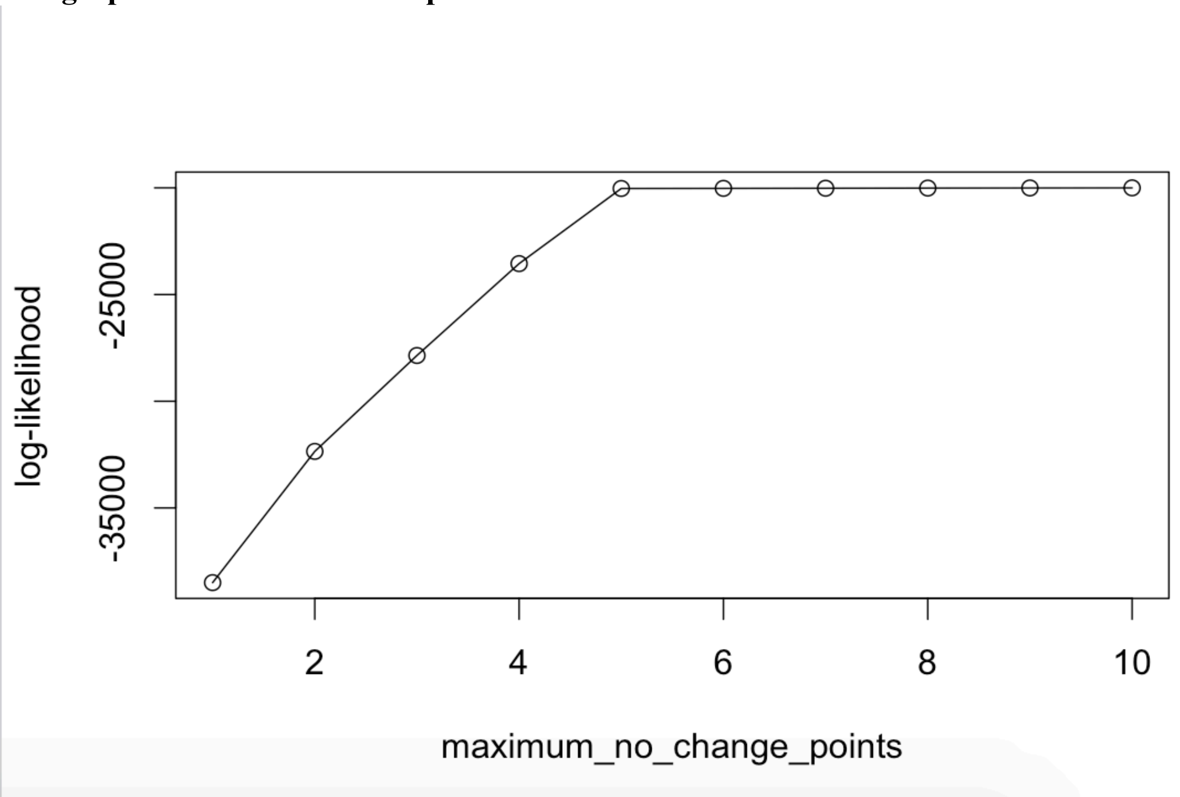
The graph for Gaussian with D:



3.2 The output matrix for Gaussian with Dplus:

```
> result = HiCseg_linkC_R(200, 10, "G", matrix, "Dplus")
> result$t_est_mat
      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
 [1,]  200    0    0    0    0    0    0    0    0     0
 [2,]   39  200    0    0    0    0    0    0    0     0
 [3,]   39  159  200    0    0    0    0    0    0     0
 [4,]   39  119  159  200    0    0    0    0    0     0
 [5,]   39   79  119  159  200    0    0    0    0     0
 [6,]   36   39   79  119  159  200    0    0    0     0
 [7,]    1    4   39   79  119  159  200    0    0     0
 [8,]    1    4   13   39   79  119  159  200    0     0
 [9,]    1    4   13   36   39   79  119  159  200     0
[10,]    1    4   25   30   37   39   79  119  159   200
> result$J
 [1] -38499.61 -32351.21 -27855.96 -23551.54 -20030.26 -20024.89
      -20018.87 -20011.23
 [9] -20005.91 -20001.32
> result$t_hat
 [1]   1   4  25  30  37  39  79 119 159 200
> plot(result$J,type="o",xlab="maximum_no_change_points",ylab="log-
likelihood")
```

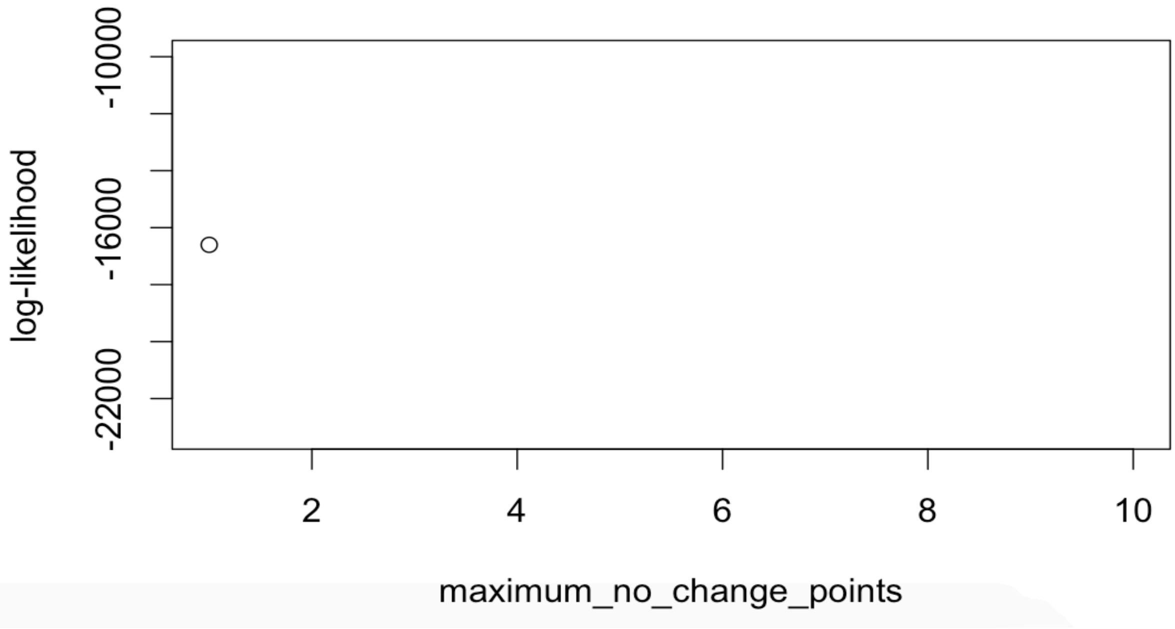
The graph for Gaussian with Dplus:



3.3 The output matrix for Poisson with D:

```
> result = HiCseg_linkC_R(200, 10, "P", matrix, "D")
> result$t_est_mat
      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
 [1,]  200    0    0    0    0    0    0    0    0     0
 [2,]    1  200    0    0    0    0    0    0    0     0
 [3,]    0    1  200    0    0    0    0    0    0     0
 [4,]    1    0    1  200    0    0    0    0    0     0
 [5,]    0    1    0    1  200    0    0    0    0     0
 [6,]    1    0    1    0    1  200    0    0    0     0
 [7,]    0    1    0    1    0    1  200    0    0     0
 [8,]    1    0    1    0    1    0    1  200    0     0
 [9,]    0    1    0    1    0    1    0    1  200     0
[10,]    1    0    1    0    1    0    1    0    1   200
> result$J
 [1] -16603.34       NaN       NaN       NaN       NaN       NaN
     NaN       NaN
 [9]       NaN       NaN
> result$t_hat
 [1] 200   0   0   0   0   0   0   0   0   0
> plot(result$J,type="o",xlab="maximum_no_change_points",ylab="log-
likelihood")
```

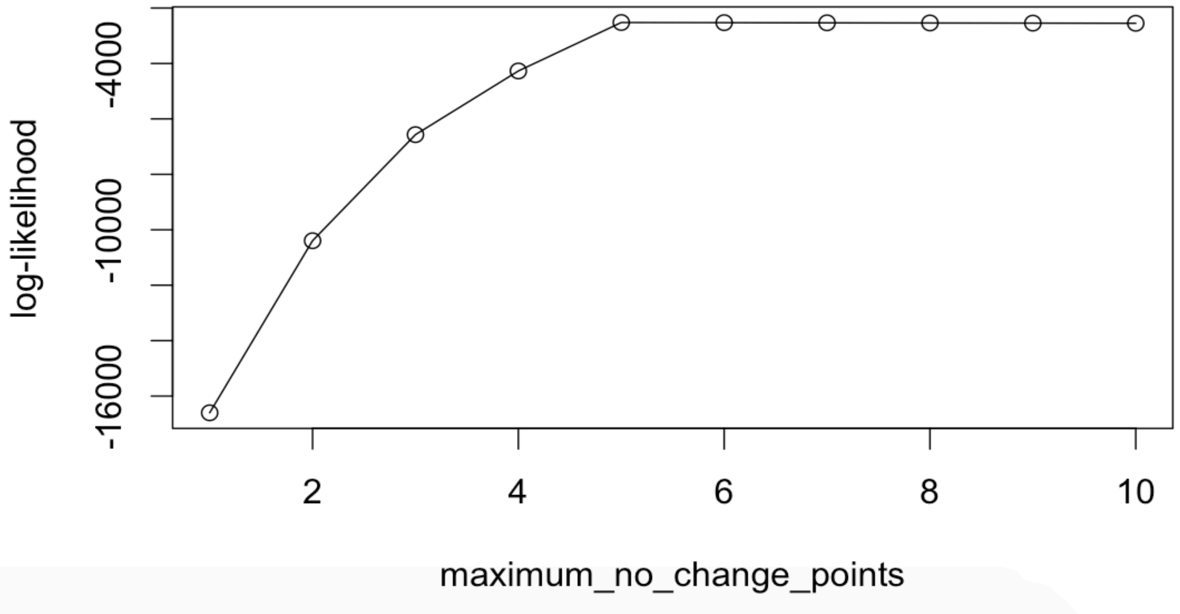
The graph for Poisson with D:



3.4 The output matrix for Poisson with Dplus:

```
> result = HiCseg_linkC_R(200, 10, "P", matrix, "Dplus")
> result$t_est_mat
      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
 [1,]  200    0    0    0    0    0    0    0    0     0
 [2,]  118  200    0    0    0    0    0    0    0     0
 [3,]   39  118  200    0    0    0    0    0    0     0
 [4,]   39   79  157  200    0    0    0    0    0     0
 [5,]   39   78  119  157  200    0    0    0    0     0
 [6,]   37   39   78  119  157  200    0    0    0     0
 [7,]   35   37   39   78  119  157  200    0    0     0
 [8,]   33   35   37   39   78  119  157  200    0     0
 [9,]   31   33   35   37   39   78  119  157  200     0
[10,]   29   31   33   35   37   39   78  119  157   200
> result$J
 [1] -16603.338 -10394.637  -6567.922  -4270.622  -2522.197  -2527.373
    -2533.144  -2538.717
 [9] -2544.630  -2550.477
> result$t_hat
 [1]  39  78 119 157 200   0   0   0   0   0
> plot(result$J,type="o",xlab="maximum_no_change_points",ylab="log-
likelihood")
```

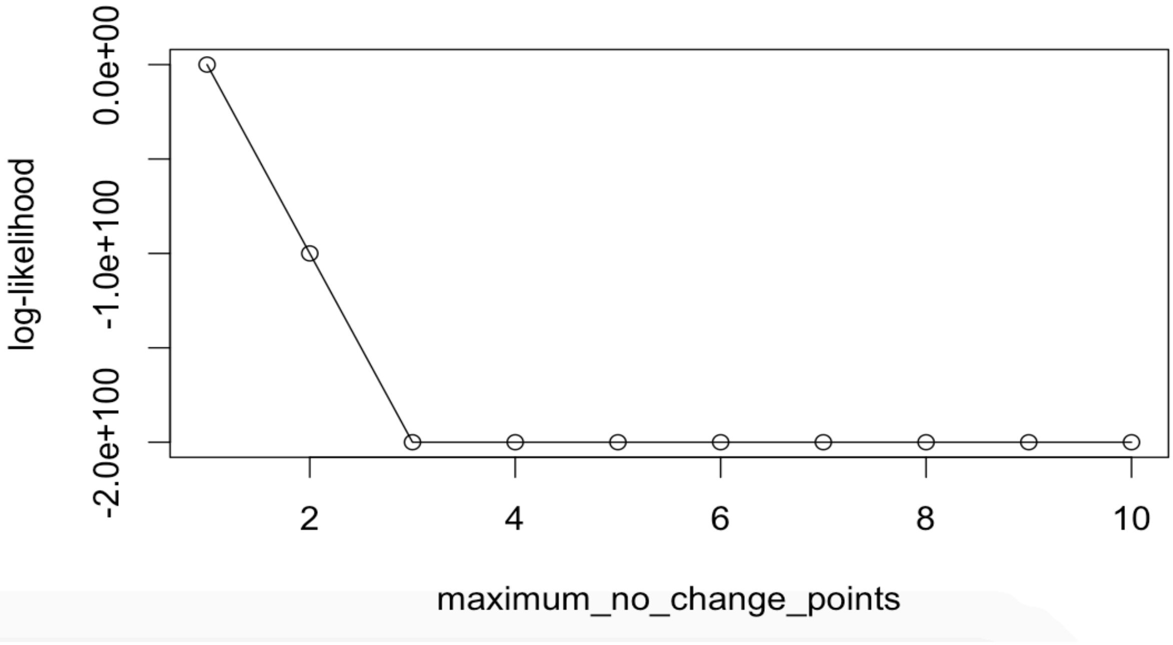
The graph for Poisson with Dplus:



3.5 The output matrix for Negative Binomial with D:

```
> result = HiCseg_linkC_R(200, 10, "B", matrix, "D")
> result$t_est_mat
      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
 [1,]  200    0    0    0    0    0    0    0    0     0
 [2,]    1  200    0    0    0    0    0    0    0     0
 [3,]    0    1  200    0    0    0    0    0    0     0
 [4,]    1    0    1  200    0    0    0    0    0     0
 [5,]    0    1    0    1  200    0    0    0    0     0
 [6,]    1    0    1    0    1  200    0    0    0     0
 [7,]    0    1    0    1    0    1  200    0    0     0
 [8,]    1    0    1    0    1    0    1  200    0     0
 [9,]    0    1    0    1    0    1    0    1  200     0
[10,]    1    0    1    0    1    0    1    0    1   200
> result$J
 [1] -1.337369e+00 -1.000000e+100 -2.000000e+100 -2.000000e+100
     -2.000000e+100
 [6] -2.000000e+100 -2.000000e+100 -2.000000e+100 -2.000000e+100
     -2.000000e+100
> result$t_hat
 [1] 200   0   0   0   0   0   0   0   0   0
> plot(result$J,type="o",xlab="maximum_no_change_points",ylab="log-
likelihood")
```

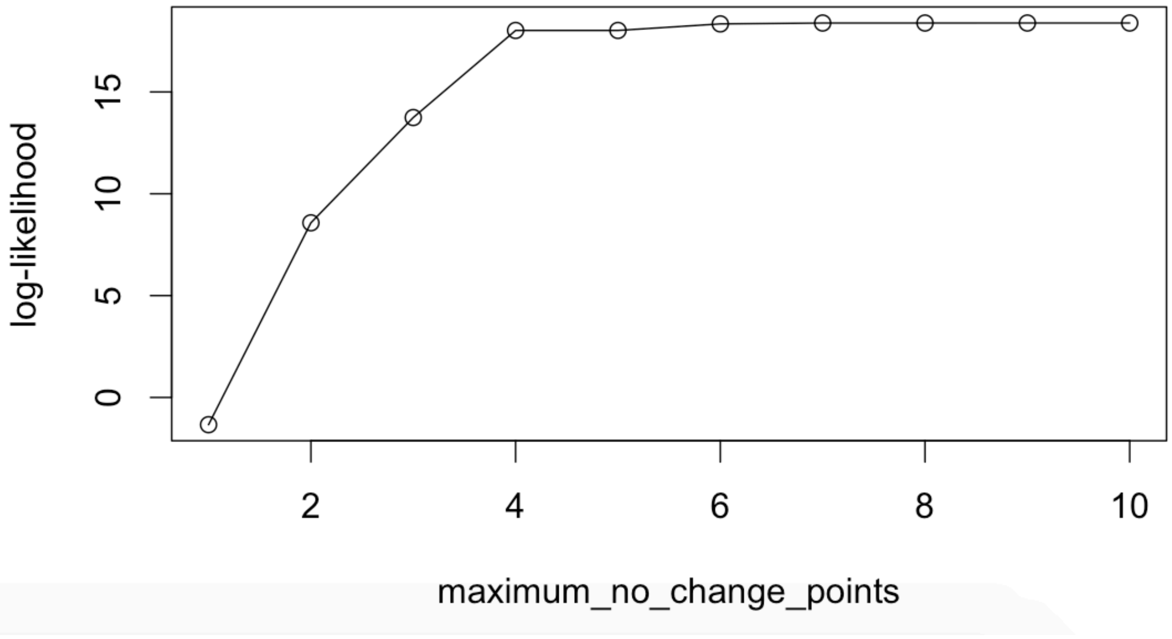
The graph for Negative Binomial with D:



3.6 The output matrix for Negative Binomial with Dplus:

```
> result = HiCseg_linkC_R(200, 10, "B", matrix, "Dplus")
> result$t_est_mat
      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
 [1,]  200    0    0    0    0    0    0    0    0     0
 [2,]  118  200    0    0    0    0    0    0    0     0
 [3,]   79  157  200    0    0    0    0    0    0     0
 [4,]   39  119  190  200    0    0    0    0    0     0
 [5,]   36   39  119  190  200    0    0    0    0     0
 [6,]   39   79  156  158  193  200    0    0    0     0
 [7,]   39   63   79  156  158  193  200    0    0     0
 [8,]   36   39   63   79  156  158  193  200    0     0
 [9,]    1    4   39   63   79  156  158  193  200     0
[10,]    1    4   13   39   63   79  156  158  193   200
> result$J
 [1] -1.337369  8.573798 13.742826 18.018267 18.018368 18.340375
     18.383088 18.383189
 [9] 18.383305 18.383434
> result$t_hat
 [1]   1   4  13  39  63  79 156 158 193 200
> plot(result$J,type="o",xlab="maximum_no_change_points",ylab="log-
likelihood")
```

The graph for Negative Binomial with Dplus:



5. CONCLUSION & FUTURE SCOPE OF WORK

- We observed that the graph for the Negative binomial with D differed from graphs of other distributions in the original implementation. The log-likelihood curve for the former decreases with increase in number of change points. Further analysis is required to assert whether the graph above is as intended or should it be similar to others.

- A fix in the original implementation can render the graph (Negative binomial with D) similar to other graphs. This behavior has been documented in our code.

- The code can be further refactored and should be taken up in the future versions.

- The current code does not take into account the penalty for increasing the number of domains. Future versions of the code can assign a penalty for increases in the number of domains.

6. REFERENCES

1. Two-dimensional segmentation for analyzing Hi-C data
   - Celine Levy-Leduc, M. Delattre, T. Mary-Huard and S. Robin
2. Multiple change-point estimation with a total variation penalty.
   - Z. Harchaoui and C. Levy-Leduc (2010).
3. Optimal detection of changepoints with a linear computational cost.
   - R. Killick (2012)
4. Pruned dynamic programming for optimal multiple change-point detection.
   G. Rigaill (2010)