Segmentor3IsBack : an R package for the fast and exact segmentation of Seq-data

Alice Cleynen, Michel Koskas, Guillem Rigaill, Stephane Robin

ALGORITHMS FOR MOLECULAR BIOLOGY

SOFTWARE ARTICLE Open Access

Segmentor3IsBack: an R package for the fast and exact segmentation of Seq-data

Alice Cleynen[1,2*], Michel Koskas[1,2], Emilie Lebarbier[1,2], Guillem Rigaill[3] and Stéphane Robin[1,2]

Abstract

Background: Change point problems arise in many genomic analyses such as the detection of copy number variations or the detection of transcribed regions. The expanding Next Generation Sequencing technologies now allow to locate change points at the nucleotide resolution.

Results: Because of its complexity which is almost linear in the sequence length when the maximal number of segments is constant, and as its performance had been acknowledged for microarrays, we propose to use the Pruned Dynamic Programming algorithm for Seq-experiment outputs. This requires the adaptation of the algorithm to the negative binomial distribution with which we model the data. We show that if the dispersion in the signal is known, the PDP algorithm can be used, and we provide an estimator for this dispersion. We describe a compression framework which reduces the time complexity without modifying the accuracy of the segmentation. We propose to estimate the number of segments via a penalized likelihood criterion. We illustrate the performance of the proposed methodology on RNA-Seq data.

Conclusions: We illustrate the results of our approach on a real dataset and show its good performance. Our algorithm is available as an *R* package on the CRAN repository.

Keywords: Segmentation algorithm, Exact algorithm, Fast algorithm, RNA-Seq data, Genome annotation, Count data, Data compression

Background

Change-point detection methods have long been used in the analysis of genetic data as an efficient tool in the study of DNA sequences for various purposes. For instance, segmentation methods have been developed for categorical variables with the aim of identifying patterns for gene predictions [1,2], while SNPs have been detected using sequence segmentation [3]. In the last two decades, with the large spread of micro-arrays, change-point methods have been widely used for the analysis of DNA copy number variations and the identification of amplification or deletion of genomic regions in pathologies such as cancer [4-8].

The recent development of Next-Generation Sequencing technologies gives rise to new applications along with new difficulties: (*a*) the increased size of profiles (up to 10^8 data-points when micro-array signals were closer to 10^5), and (*b*) the discrete nature of the output (number of reads starting at each position of the genome). Yet applying segmentation methods to DNA-Seq data with their greater resolution should lead to the analysis of copy-number variation with a much improved precision compared to CGH arrays. Moreover, in the case of poly-(A) RNA-Seq data on lower organisms, since coding regions of the genome are well separated from non-coding regions with lower activity, segmentation methods should allow the identification of transcribed genes as well as address the issue of new transcript discovery. Our objective is therefore to develop a segmentation method to tackle both (*a*) and (*b*) with some specific requirements: the amount of reads falling within a segment should be representative of the biological information associated (relative copy-number of the region, relative level of expression of the gene), and comparison to neighboring regions should be sufficient to label the segment (for instance normal or deleted region of the chromosome in DNA-Seq data, exon or non-transcribed region in RNA-Seq), therefore

*Correspondence: acleynen@jimmy.harvard.edu
[1]AgroParisTech, UMR 518, 16 rue Claude Bernard, 75231 Paris Cedex 05, France
[2]INRA, UMR 518, 16 rue Claude Bernard, 75231 Paris Cedex 05, France
Full list of author information is available at the end of the article

no comparison profile should be needed. This also suppresses the need for normalization, and consequently we wish to analyze the raw count-profile.

Up to now, most methods addressing the analysis of these datasets require some normalization process to allow the use of algorithms which rely on Gaussian-distributed data or which were previously developed for micro-arrays [9-12]. Indeed, methods adapted to count datasets are not numerous and are highly focused on the Poisson distribution. Alteration of genomic sequences can be detected based on the comparison of Poisson processes associated with the read counts of a case and a control sample [13], but this cannot be applied to the detection of transcribed regions in a single condition.

Still, a likelihood ratio statistic was proposed for the localization of a shift in the intensity of a Poisson process [14], and a test statistic was proposed for the existence of a change-point in the Poisson autoregression of order 1 [15].

These last two methods do not require a comparison profile but they only allow for the detection of a single change-point and have too high a time-complexity to be applied to RNA-Seq profiles. Binary Segmentation, a fast heuristic [6], and Pruned Exact Linear Time (PELT) [16], an exact algorithm for optimal segmentation with respect to the likelihood, are both implemented for the Poisson distribution in the **changepoint** package. Even though both are extremely fast, do not require a comparison profile, and analyze count-data, the Poisson distribution is not adapted to our kind of datasets.

A recent study [17] has compared 13 segmentation methods for the analysis of chromosomal copy number profiles and has shown the excellent performance of the Pruned Dynamic Programming (PDP) algorithm [18] proposed in its initial implementation for the analysis of Gaussian data in the *R* package **cghseg**. We propose to use this algorithm, which we have implemented for the Poisson and negative binomial distributions.

In the next section we recall the general segmentation framework and the definition and requirements of the PDP algorithm. Our contributions are given in the third section where we define the negative binomial model and show that it satisfies the PDP algorithm requirements. We also provide a theoretical result for the possibility to compress the data, and finally we give a model selection criterion with theoretical guarantees, which makes the whole approach complete. We conclude with a simulation study, which illustrates the performance of the proposed method.

Segmentation model and algorithm

General segmentation model

The general segmentation problem consists in partitioning a signal of n data-points $\{y_t\}_{t\in[\![1,n]\!]}$ into a given number K of pieces or segments. The model can be written as follows: the observed data $\{y_t\}_{t=1,\dots,n}$ are supposed to be a realization of an independent random process $Y = \{Y_t\}_{t=1,\dots,n}$. This process is drawn from a probability distribution $\mathcal{G}$ which depends on a set of parameters among which one parameter θ is assumed to be affected by $K-1$ abrupt changes, called change-points, such that

$$Y_t \sim \mathcal{G}(\theta_r, \phi) \qquad \text{if } t \in r \quad \text{and} \quad r \in m$$

where m is a partition of $[\![1, n]\!]$ into segments r, θ_r stands for the parameter of segment r and ϕ is constant. The objective is to estimate the change-points or the positions of the segments and the parameters θ_r both resulting from the segmentation. More precisely, we define $\mathcal{M}_{k,t}$ the set of all possible partitions in $k > 0$ regions of the sequence up to point t. We recall that the number of possible partitions is

$$\text{card}(\mathcal{M}_{k,t}) = \binom{t-1}{k-1}.$$

We aim at choosing the partition in $\mathcal{M}_{K,n}$ of minimal loss γ, where the loss is usually taken as the negative log-likelihood of the model. We define the point-additive loss of a segment with given parameter θ as $c(r,\theta) = \sum_{i\in r} \gamma(y_i, \theta)$, therefore its optimal cost is $c(r) = \min_\theta \{c(r,\theta)\}$. This allows us to define the cost of a segmentation m as $\sum_{r\in m} c(r)$ and our goal is to recover the optimal segmentation $M_{K,n}$ and its cost $C_{K,n}$ which are particular cases of the generic optimal segmentation of the signal up to point t in k segments and its cost, defined as:

$$M_{k,t} = \arg\min_{\{m\in\mathcal{M}_{k,t}\}} \left\{ \sum_{r\in m} c(r) \right\}$$

$$\text{and} \quad C_{k,t} = \min_{\{m\in\mathcal{M}_{k,t}\}} \left\{ \sum_{r\in m} c(r) \right\}.$$

Quick overview of the PDP algorithm

Like the original DP algorithm, the pruned DP algorithm is an iterative algorithm based on the minimization of a cost function $C_{k,t}$ which is traditionally decomposed as:

$$C_{k,t} = \min_{\{k-1<\tau<t\}} \left\{ C_{k-1,\tau} + \min_\theta \left[c([\tau+1, t], \theta) \right] \right\} \quad (1)$$

where θ is the parameter of the cost of the last segment, constraints on its possible values being directly related to the support of the loss function γ (for instance θ takes its value in $\mathbb{R}$ in the case of the Gaussian loss, but in $[0,1]$ in the case of the binomial loss). In what follows we will denote by I_s the set of possible values for parameter θ.

The specificity of the PDP algorithm is that it relies on the comparison of candidates for the last change-point position τ through the permutation of the minimizations in (1) and the introduction of the functions:

$$H_{k,t}(\theta) = \min_{k-1<\tau\le t} \left\{ C_{k-1,\tau} + c([\tau+1, t], \theta) \right\},$$

which are the cost of the best partition in k regions up to t, the parameter of the last segment being θ. $C_{k,t}$ is then obtained as $\min_\theta\{H_{k,t}(\theta)\}$.

Then at each iteration k, the PDP algorithm works on a list of last change-point candidates: ListCandidate$_k$. For each of these τs and for each value of t, it updates the set of θs, denoted $S^\tau_{k,t}$ for which this candidate is optimal. If this set is empty, the candidate is discarded, resulting in the pruning and lower complexity of the algorithm.

The foundations of the algorithm can be written as follows.

- Defining $H^\tau_{k,t}(\theta) = C_{k-1,\tau} + \sum_{j=\tau+1}^{t} \gamma(y_j, \theta)$ the optimal cost if the last change is τ and last parameter is θ, then

  (*i*) $H^\tau_{k,t+1}(\theta)$ is obtained from $H^\tau_{k,t}(\theta)$ using:
  $$H^\tau_{k,t+1}(\theta) = H^\tau_{k,t}(\theta) + \gamma(y_{t+1}, \theta);$$

- Defining $I^\tau_{k,t} = \left\{\theta \mid H^\tau_{k,t}(\theta) \leq H^t_{k,t}(\theta)\right\} = \left\{\theta \mid H^\tau_{k,t}(\theta) \leq C_{k-1,t}\right\}$ the set of θ such that τ is better than t in terms of cost, with $\tau < t$, then

  (*ii*) if all $\sum_{j=\tau+1}^{t} \gamma(y_j, \theta)$ are unimodal in θ then $I^\tau_{k,t}$ are intervals. Indeed, since by definition $H^t_{k,t}(\theta) = C_{k-1,t}$ and the cost function does not depend on θ, $I^\tau_{k,t}$ is the set of values for which a unimodal function is smaller than a constant.

- Finally, we introduce $S^\tau_{k,t} = \left\{\theta \mid H^\tau_{k,t}(\theta) \leq H_{k,t}(\theta)\right\}$ the set of θ such that τ is optimal. Then since $H_{k,t}(\theta) = \min_{\tau \leq t}\left\{H^\tau_{k,t}(\theta)\right\}$, $S^\tau_{k,t}$ can be written as $\left\{\theta \mid H^\tau_{k,t}(\theta) = H_{k,t}(\theta)\right\}$ and we obtain that

  (*iii*) $S^\tau_{k,t+1}$ can be updated using:

  - $S^\tau_{k,t+1} = S^\tau_{k,t} \cap I^\tau_{k,t+1}$
  - $S^t_{k,t} = I_s \setminus (\cup_{\tau \in \text{ListCandidate}_k} I^\tau_{k,t})$

  The first assertion follows from the fact that $S^\tau_{k,t+1} = \left\{\theta | H^\tau_{k,t+1} \leq \min_{k \leq \tau \leq t+1} H^\tau_{k,t+1}\right\} = \left\{\theta | H^\tau_{k,t+1} \leq \min\left(H^{t+1}_{k,t+1}, \min_{k \leq \tau \leq t} H^\tau_{k,t+1}\right)\right\}$, the first term in the minimum giving $I^\tau_{k,t+1}$ and the second one giving $S^\tau_{k,t}$. The second assertion trivially follows from the fact that candidate t is optimal on the set of values where no other candidate was optimal.

  (*iv*) once it has been determined that $S^\tau_{k,t}$ is empty, it easily follows from the update equation (*iii*) that the region-border τ can be discarded from the list of candidates *ListCandidate*$_k$:
  $$S^\tau_{k,t} = \emptyset \Rightarrow \qquad \forall\, t' \geq t \quad S^\tau_{k,t'} = \emptyset.$$

Requirements of the pruned dynamic programming algorithm.

Proposition 0.1. *Properties (i) to (iv) are satisfied as soon as the following conditions on the loss $c(r, \theta)$ are met:*

(a) It is point additive,
(b) It is convex with respect to its parameter θ,
(c) It can be stored and updated efficiently.

The proof of those claims can be found in [18]. A pseudo-code of the PDP algorithm is given in the appendix.

It is possible to include an additional penalty term, denoted g as in the pseudo-code, in the loss function. To preserve the point-additivity requirement of the loss, this penalty can only depend on the value of the segment-parameter θ and not on any other characteristics, such as segment length. This is then equivalent to minimizing $C_{k,t} = \min_{\{k-1<\tau<t\}}\left\{C_{k-1,\tau} + \min_\theta\left\{c([\tau+1, t], \theta) + g(\theta)\right\}\right\}$ and can be achieved by adding the penalty value $g(\theta)$ in the initialization of $H^\tau_{k,t}(\theta)$. For example, in the case of RNA-seq data one could add a lasso ($\lambda|\theta|$) or ridge penalty ($\lambda\theta^2$) to encode that *a priori* the coverage in most regions should be close to 0. Our C++ implementation of the PDP algorithm includes the possibility of adding such a penalty term; however we do not provide an `R` interface to this functionality in our `R` package. One of the reasons for this choice is that choosing an appropriate value for λ is not a simple problem.

Contribution

Pruned dynamic programming algorithm for count data

We now show that the PDP algorithm can be applied to the segmentation of RNA-Seq data using a negative binomial model and we propose a criterion for the choice of K. Though not discussed here, our results also hold for the Poisson segmentation model.

Negative binomial model. We consider that in each segment r all y_t are the realization of random variables Y_t which are independent and follow the same negative binomial distribution. Assuming the dispersion parameter ϕ to be known, we will use the natural parametrization from the exponential family (also classically used in R) so that parameter θ_r will be the probability of success. In this framework, θ_r is specific to segment r whereas ϕ is common to all segments.

We have $E(Y_t) = \phi(1-\theta)/\theta$ and $Var(Y_t) = \phi(1-\theta)/\theta^2$. We choose the loss as the negative log-likelihood associated with data-point t belonging to segment r: $-\phi\log(\theta_r) - y_t\log(1-\theta_r) + A(\phi, y_t)$, or more simply $\gamma(y_t, \theta_r) = -\phi\log(\theta_r) - y_t\log(1-\theta_r)$ since A is a function that does not depend on θ_r.

Validity of the pruned dynamic programming algorithm for the negative binomial model

Proposition 0.2. *Assuming parameter ϕ to be known, the negative binomial model satisfies (a), (b) and (c):*

(a) As we assume that Y_t are independent, we indeed have that the loss is point additive: $c(r,\theta) = \sum_{t\in r}\gamma(y_t,\theta)$.
(b) As $\gamma(y_t,\theta) = -\phi\log(\theta) - y_t\log(1-\theta)$ is convex with respect to θ, $c(r,\theta)$ is also convex as the sum of convex functions.
(c) Finally, we have $c(r,\theta) = -n_r\phi\log(\theta) + \sum_{t\in r} y_t\log(1-\theta)$ (where n_r is the length of segment r). This function can be stored and updated using only two doubles: one for $-n_r\phi$, say d_1, and the other for $\sum_{t\in r} y_t$, say d_2. Then at step $t+1$ as the new datapoint y_{t+1} is considered, these doubles are simply updated as $d_1 \leftarrow d_1 + \phi$ and $d_2 \leftarrow d_2 + y_{t+1}$.

Estimation of the overdispersion parameter. We propose to estimate ϕ using a modified version of Johnson *et. al*'s estimator [19]: compute the moment estimator of ϕ on each sliding window of size h using the formula $\phi = \mathbb{E}(Y)^2/(Var(Y) - \mathbb{E}(Y))$ and keep the median $\widehat{\phi}$.

Taking into account a positional bias. It is possible that the assumption that the counts share the same distribution in a segment might not be verified. For instance in the case of RNA-Seq data the number of reads can be affected by the location in the transcribed region or by the GC-content of the fragment. The pruned dynamic programming algorithm only requires a vector of integers as input, it is therefore possible to apply any kind of normalization process that preserves the count-specificity of the data prior to segmentation. For instance, a method such as that which has resulted in the publication of the data used in the illustration [20] can be applied. A comparison of the main normalization methods can for example be found in Bullard *et. al.*'s paper [21].

C++ implementation of the PDP algorithm

We implemented the PDP algorithm in C++ having in mind the possibility of adding new loss functions in potential future applications. The difficulties we had to face were the versatility of the program to be designed and the design of the objects it could work on. Indeed, the use of full templates implied that we used stable sets of objects for the operations that were to be performed.

Namely:

- The sets were to be chosen in a *tribe*. This means that they all belong to a set $\mathcal{S}$ of sets such that any set $s \in \mathcal{S}$ can be conveniently handled and stored in the computer. A set of sets $\mathcal{S}$ is said to be *acceptable* if it satisfies the following:
  1. If s belongs to $\mathcal{S}$, $\mathbb{R}\setminus s \in \mathcal{S}$
  2. If $s_1, s_2 \in \mathcal{S}$, $s_1 \cap s_2 \in \mathcal{S}$
  3. If $s_1, s_2 \in \mathcal{S}$, $s_1 \cup s_2 \in \mathcal{S}$

  For instance, the set $\mathcal{S}$ of intervals is a tribe since the complementary, the union and the intersection of intervals form a union of intervals. This property ensures that the sets $I^\tau_{k,t}$ and $S^\tau_{k,t}$ can be updated and stored efficiently (only two doubles are required to store an interval) to take full advantage of the pruning process.
- The cost functions were chosen in a set $\mathcal{F}$ such that
  1. Each function may be conveniently handled and stored by the software.
     For instance, for the Gaussian loss it suffices to store the three coefficients of a second order polynomial.
  2. For any $f \in \mathcal{F}$ and any constant c, $f(x) \leq c$ can be easily solved and the set of solutions belongs to an acceptable set of sets
  3. For any $f, g \in \mathcal{F}$, $f + g \in \mathcal{F}$.
     These two points ensure that the cost (and penalty) functions can be easily updated and compared so that the sets $I^\tau_{k,t}$ of each candidate τ can be updated and candidates eventually discarded.

Thus we defined two collections for the sets of sets $\mathcal{S}$, intervals and parallelepipeds, and implemented the loss functions corresponding to negative binomial, Poisson or normal distributions. The program is thus designed in a way that any user can add his own cost function or acceptable set of probability function and use it without rewriting a line in the code.

Compression of the signal

In the case of count data, and in particular in the analysis of RNA-Seq data, it is very likely that we observe plateaux, that is regions between two arbitrary positions t_1 and t_2 ($> t_1$) where the signal is constant:

$$\forall t,\ t_1 \leq t \leq t_2,\ y_t = y_{t_1} = y_{t_2}.$$

Then we have the following proposition, the proof of which is given in the appendix.

Proposition 0.3. *There exists a segmentation m in K or fewer segments without any change-point in the plateaux such that the optimal cost of m is equal to $C_{K,n}$.*

This proposition proves the arguably intuitive idea that having a change-point between t_1 and t_2 is never beneficial in terms of cost. When searching for the best segmentation of the data, it is therefore unnecessary to

look for change-points in plateaux. In other words a plateau starting at position t_1 and ending at position t_2 can be considered as a unique data point with value y_{t_1} and weight $t_2 - t_1 + 1$. At worst the size of the compressed signal is equal to the minimum between two times the number of reads and the length of the chromosome arm. Thus, if the number of reads is very large, the two-step algorithm (compression and pruned dynamic programming) does not change the worst case complexity. However, in most cases the number of reads is much smaller than the size of the considered chromosome. Thus compression is efficient and allows for a significant reduction in the overall run-time. Furthermore, in the case of RNA-Seq data we do not expect reads to be evenly scattered. On the contrary they are concentrated in transcribed regions and between those regions we expect large plateaux of 0 allowing for an efficient compression (for instance only 2% of the human chromosome contains coding regions).

Model selection

The last issue concerns the estimate of the number of segments K. This model selection issue can be solved using a penalized log-likelihood criterion for which the choice of a good penalty function is crucial. This kind of procedure typically requires computation of the optimal segmentations in all $k = 1, \ldots, K_{\max}$ segments where $K_{\max}$ is generally chosen smaller than n. The most popular criteria (AIC [22] and BIC [23]) failed in the segmentation context due to the discrete nature of the change-points. Indeed, additionally to being an asymptotic criterion in a framework where the collection of possible models grows polynomially with n, the BIC criterion uses a Laplace approximation requiring differentiability conditions of the likelihood function which are not satisfied by the segmentation model [24]. From a non-asymptotic point of view and for the negative binomial model, the following criterion was proposed [25]: denoting $\hat{m}_K$ the optimal segmentation of the data in K segments,

$$\hat{K} = \arg\min_{K \in 1:K_{\max}} \left\{ \sum_{r \in \hat{m}_K} \sum_{t \in r} \left[-\phi \log \frac{\phi}{\phi + \bar{y}_r} - y_t \log\left(1 - \frac{\phi}{\phi + \bar{y}_r}\right) \right] + \beta K \left(1 + 4\sqrt{1.1 + \log\left(\frac{n}{K}\right)}\right)^2 \right\}, \quad (2)$$

where $\bar{y}_r = \dfrac{\sum_{t \in r} y_t}{\hat{n}_r}$ and $\hat{n}_r$ is the size of segment r. The first term corresponds to the cost of the optimal segmentation while the second is a penalty term which depends on the dimension K and on a constant β that has to be tuned according to the data (see the next section). With this choice of penalty, a so-called oracle penalty, the resulting estimator satisfies an oracle-type inequality. A more complete performance study is done in [25] and showed that the proposed criterion outperforms the existing ones.

Implementation

The Pruned Dynamic Programming algorithm is available in the function `Segmentor` of the *R* package **Segmentor3IsBack**. Version 1.7 of this package contains the compression process which is performed by default in the case of count data. The user can choose the distribution with the slot `model` (1 for Poisson, 2 for Gaussian homoscedastic, 3 for negative binomial and 4 for segmentation of the variance). It returns an S4 object of class Segmentor which can later be processed for other purposes. The function `SelectModel` provides four criteria for choosing the optimal number of segments: AIC [22], BIC [23], the modified BIC [24] (available for Gaussian and Poisson distribution) and oracle penalties (available for the Gaussian distribution [26] and for the Poisson and negative binomial [25] as described previously). This latter kind of penalty requires tuning a constant according to the data, which is done using the slope heuristic [27].

Figure 1 (which is detailed in the Results and discussion Section) was obtained with the following 4 lines of code (assuming the data was contained in vector `x`):

```
Seg<-Segmentor(x,model=3,Kmax=200)

Kchoose<-SelectModel(Seg, penalty=
"oracle")

plot(sqrt(x),col='dark red')

abline(v=getBreaks(Seg)[Kchoose,
1:Kchoose],col='blue')
```

The function `BestSegmentation` allows us, for a given K, to find the optimal segmentation with a change-point at location t (slot `$bestSeg`). It also provides, through the slot `$bestCost`, the cost of the optimal segmentation with t for j^{th} change-point. Figure 2(Left) illustrates this result for the optimal segmentations in 4 segments of a signal simulated with only 3 segments. We can see for instance that any choice of first change-point location between 1 and 2000 yields almost the same cost (the minimum is obtained for $t = 1481$), and thus the optimal segmentation is not clearly better than the next best segmentations. On the contrary, the same function with 3 segments shows that the optimal segmentation outperforms all other segmentations in 3 segments.

Results and discussion

Performance study

We designed a simulation study on the negative binomial distribution to assess the performance of the PDP algorithm in terms of computational efficiency without using the compression option, while studying the impact of the overdispersion parameter ϕ by comparing the results for two different values of this parameter. After running different estimators (median on sliding windows

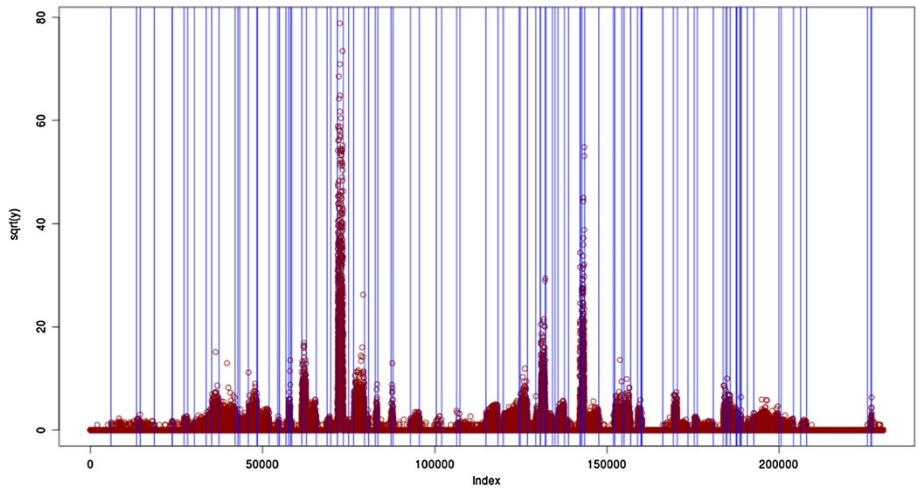

Figure 1 Segmentation of the yeast chromosome 1 using the negative binomial loss. The model selection procedure chooses $K = 125$ segments, most of which correspond to the official annotation, with segments corresponding to transcribed regions surrounding official genes.

of maximum, quasi-maximum likelihood and moment estimators) on several real RNA-Seq data (whole chromosome and genes of various sizes), we fixed $\phi_1 = 0.3$ as a typical value for highly dispersed data as observed in real RNA-Seq data and chose $\phi_2 = 2.3$ for comparison with a reasonably dispersed dataset. For each value, we simulated datasets of size n with various densities of number of segments K, and only two possible values for the parameter p_J: 0.8 on even segments (corresponding to low signal) and 0.2 on odd segments for a higher signal. We had n vary on a logarithmic scale between 10^3 and 10^6 and K between $\sqrt{n}/6$ and $\sqrt{n}/3$. For each configuration, we segmented the signal up to $K_{\max} = \sqrt{n}$ twice: once with the known value of ϕ and once with our estimator $\widehat{\phi}$ as described above. We started with a window width $h = 15$. When the estimate was negative, we doubled h and repeated the experience until the median was positive.

Each configuration was simulated 100 times.

For our analysis we checked the run-time on a standard laptop, and assessed the quality of the segmentation using the Rand Index $\mathcal{I}$. Specifically, let C_t be the true index

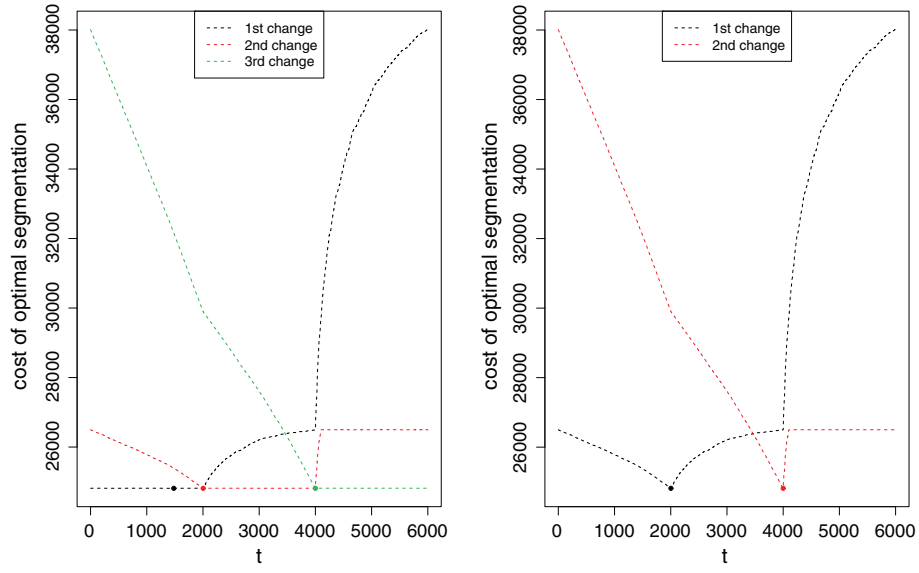

Figure 2 Cost of optimal segmentation in 4 and 3 segments. Cost of optimal segmentation depending on the location of the j^{th} change-point when the number of segments is 4 (Left) and 3 (Right) and the signal was simulated with 3 segments. Illustration of the output of function `BestSegmentation`.

of the segment to which base t belongs and let $\hat{C}_t$ be the index estimated by the method, then

$$\mathcal{I} = \frac{2\sum_{s=1}^{n}\sum_{t>s}\left[\mathbf{1}_{C_t=C_s}\mathbf{1}_{\hat{C}_t=\hat{C}_s} + \mathbf{1}_{C_t\neq C_s}\mathbf{1}_{\hat{C}_t\neq\hat{C}_s}\right]}{(n-1)(n-2)}.$$

Figure 3 shows, for the particular case of $K = \sqrt{n}/3$, the almost linear complexity of the algorithm in the size n of the signal. As the maximal number of segments $K_{\max}$ considered increased with n, we normalized the run-time to allow comparison. This underlines an empirical complexity smaller than $\mathcal{O}(K_{\max} n \log n)$, and independent of the value of ϕ or its knowledge. Moreover, the algorithm, and therefore the pruning, is faster when the overdispersion is high, a phenomenon already encountered with the L^2 loss when the distribution of errors is Cauchy. However, the knowledge of the true value of ϕ does not affect the run-time of the algorithm. Figure 4 illustrates through the Rand Index the quality of the proposed segmentation for a few values of n. Even though the indexes are slightly lower for ϕ_1 than for ϕ_2 (see left panel), they range between 0.94 and 1 showing a great quality in the results. Moreover, the knowledge of ϕ does not increase the quality (see right panel), which validates the use of our estimator. We can therefore conclude that the run-time of our algorithm without compression is roughly $40 \times K_{max} \times n/10^6$s.

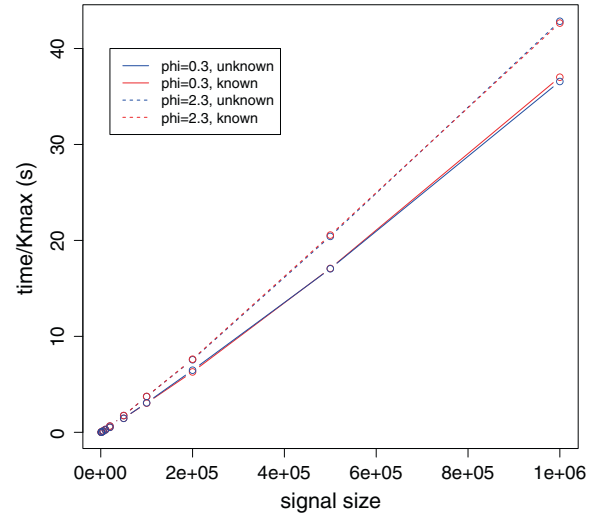

Figure 3 Run-time analysis for segmentation with negative binomial distribution. This figure displays the normalized (by $K_{\max}$) run-time in seconds of the **Segmentor3IsBack** package for the segmentation of signals with increasing length n, for two values of the dispersion ϕ, and with separate analyses for a known value or an estimated value. While the algorithm is faster for more over-dispersed data, the estimation of the parameter does not slow the processing.

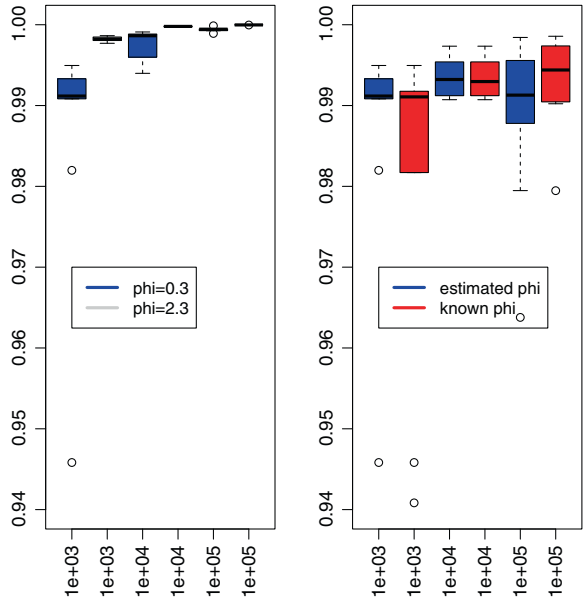

Figure 4 Rand Index for the quality of the segmentation. This figure displays the boxplot of the Rand Index computed for each of the hundred simulations performed in the following situations: comparing the values with ϕ_1 and ϕ_2 when estimated (left figure), and comparing the impact of estimating ϕ_1 (right figure). While the estimation does not decrease the quality of the segmentation, the value of the dispersion affects the recovery of the true change-points.

Yeast RNAseq experiment

We applied our algorithm to the segmentation of chromosome 1 of the *S. Cerevisiae* (yeast) using RNA-Seq data from the Sherlock Laboratory at Stanford University [20], publicly available from the NCBI's Sequence Read Archive (SRA, http://www.ncbi.nlm.nih.gov/sra, accession number SRA048710). We selected the number of segments using our oracle penalty described in the previous section. An existing annotation of translated regions (*i.e.* excluding un-translated regions (UTR)) is available on the Saccharomyces Genome Database (SGD) at http://www.yeastgenome.org, which allows us to validate our results.

With a run-time of 27 minutes without compression, and 5.4 minutes with compression (for a signal length of 230218), we selected 125 segments with the negative binomial distribution. Most of those segments (all but 3) can be related to the official annotation, however as expected segments corresponding to transcribed regions (as opposed to intergenic regions) were found to surround known genes from the SGD due to the difference between transcribed and translated regions. Figure 1 illustrates the result.

We compared our segmentation with that corresponding to the SGD annotation through the Hellinger distance by fitting a negative binomial distribution on each segment and repeated this comparison with the other two algorithms able to process long count datasets: PELT [16] and Binary Segmentation [6], both implemented in the R package `changepoint` for the Poisson distribution. For

fair comparison, we also used the PDP algorithm for the Poisson loss. Figure 5, together with Table 1 which gives the estimated number of segments, the overall Hellinger score $\left(\sum_t H_t/n\right)$ and the number of change-points falling within annotated translated regions, illustrates the result and shows that we outperform the other approaches. Moreover, most of the Hellinger peaks observed can be explained by the fact that we are comparing the annotation of transcribed regions with that of translated regions.

Table 1 Comparison of algorithm performance on real data

Algorithm	Number of segments	Hellinger score	False positives
PDPA- negative binomial	125	0.0120	39
PDPA- Poisson	106	0.0187	77
PELT	3416	0.0188	3003
Binary Segmentation	2408	0.0151	2072

Overall Hellinger score of each of the segmentation algorithms, and number of estimated change-points falling within regions annotated as translated (thus considered as false positives).

Analysis of complex organisms

The issues raised in the analysis of RNA-Seq and DNA-Seq data differ. In the first case, the number of segments that we hope to select is roughly twice the number of expressed exons, therefore the order of K_{max} varies from 10^2 (small chromosomes from lower organisms, e.g. yeast) to 10^4 (large chromosomes from higher organisms, e.g. human). However, when aligned to a reference genome, RNA-Seq data is expected to present large plateaux of zeros at non-coding regions (for instance, 98% of the human genome) and at non expressed regions. The compression option of our algorithm then allows us to reduce the size of the profile by a factor of 10 to 10^3. Moreover, it is well-known that centromere regions are large non-coding regions where no change-point is expected, and we therefore propose to divide the profile into two parts at such regions. As a proof of concept we ran our algorithm with compression on an RNA-seq profile of the small arm of the 4th chromosome of *Arabidopsis*

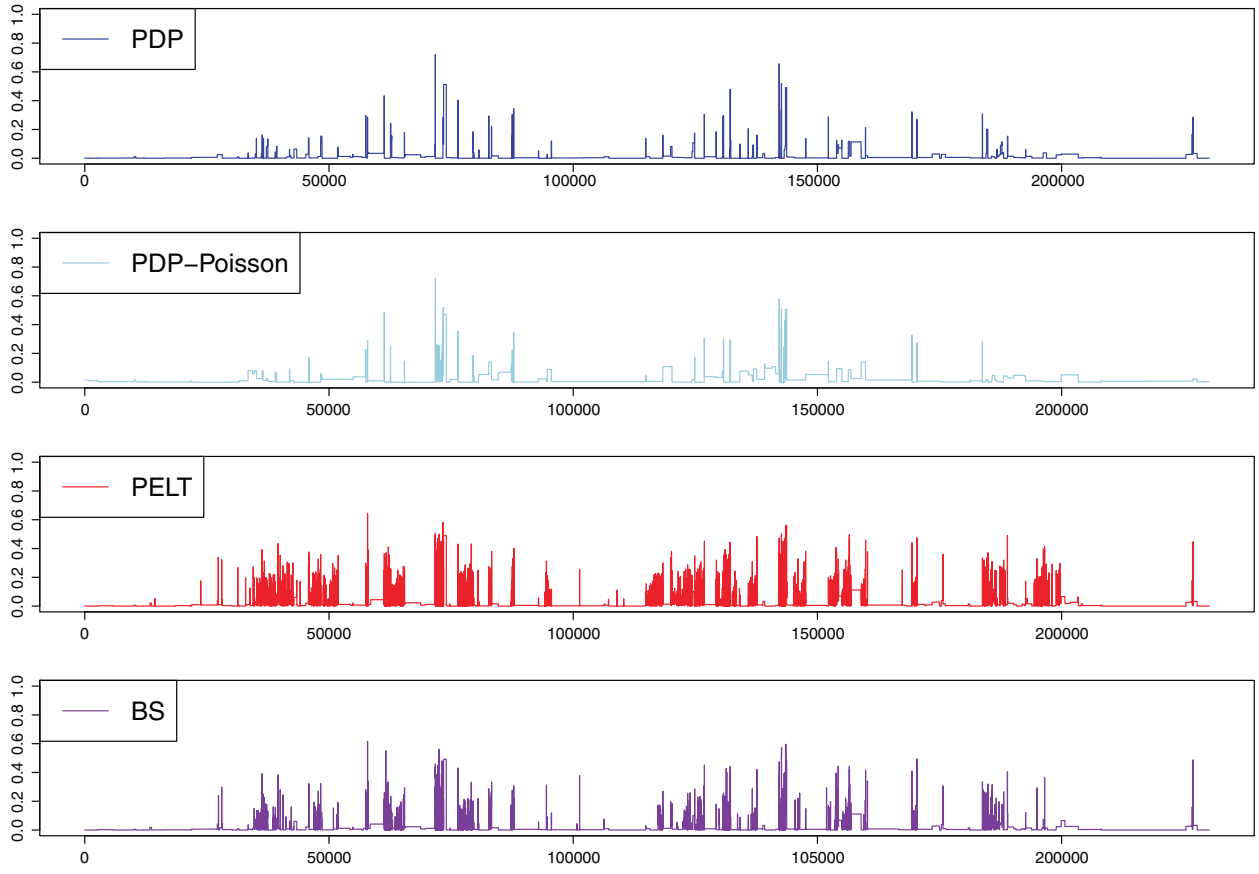

Figure 5 Comparison of proposed segmentation with annotation and PELT and Binary Segmentation algorithms. Each figure gives the Hellinger distance between the estimated segmentation of the algorithm (top: PDP with negative binomial, then: PDP with Poisson, third: PELT, and bottom: Binary segmentation) and that of the SGD. The PDP algorithm with the negative binomial distribution seems to outperform other algorithms.

Thaliana $\left(n = 4.10^6,\ K_{max} = 6.10^3\right)$ and selected 4289 segments after a compression factor of 10 and a run-time of 19 hours on a 2.4Ghz computer. The data was kindly provided by some of our collaborators.

DNA-Seq data on the other hand will present much smaller plateaux. While this implies that the compression will be less efficient, the profile can still be summarized into a dataset the length of which will be smaller than the total amount of mapped reads.Most importantly, in these experiments the expected number of segments is drastically smaller as the number of chromosomic aberrations is generally limited to less than one hundred per chromosome, even in pathologies such as cancer.

Conclusion

Segmentation has been a useful tool for the analysis of biological datasets for a few decades. We propose to extend its application with the use of the Pruned Dynamic Programming algorithm for count datasets such as outputs of sequencing experiments. We show that the negative binomial distribution can be used to model such datasets on the condition that the overdispersion parameter is known and have proposed an estimator of this parameter that performs well in our segmentation framework.

We propose to choose the number of segments using our oracle penalty criterion, which makes the package fully operational. This package also allows the use of other criteria such as AIC or BIC. Similarly, the algorithm is not restricted to the negative binomial distribution but also allows the use of Poisson and Gaussian losses for instance and could easily be adapted to other convex one-parameter losses.

With its empirical complexity of $\mathcal{O}(K_{\max} n \log n)$, it can be applied to large signals such as read-alignment of whole chromosomes, and we illustrated its result on a real dataset from the yeast genomes. Moreover, this algorithm can be used as a base for further analysis. For example, [28] use it to initialize their Hidden Markov Model to compute change-point location probabilities.

Availability and requirements

- Project name: Segmentor3IsBack
- Project home page: http://cran.r-project.org/web/packages/Segmentor3IsBack/index.html
- Operating systems: Platform independent
- Programming language: C++ code embedded in *R* package
- License: GNU GPL
- Any restrictions to use by non-academics: none

Appendix

Pseudo-code of the PDP algorithm

Algorithm 1 The PDP algorithm

Input: y_i a sequence of n data-points, K an integer
I_s a set of possible values for θ
γ the loss function, g a penalty function
Output: $C_{k,t}$ a matrix of floats of size $K \times n$
$M_{k,t}$ a matrix of integers of size $K \times n$

Initialize

For $t \in \{1, \dots n\}$
$C_{1,t} = min_{\theta \in I_s} \left\{ \sum_{i=1}^{t} \gamma(y_i, \theta) + g(\theta) \right\}$
$M_{1,t} = 0$
End For

Main

For $k \in \{2, \dots K\}$
$\text{ListCandidate}_k = \{k-1\}$
$H_{k,k-1}^{k-1}(\theta) = C_{k-1,k-1} + g(\theta)$
$S_{k,k-1}^{k-1} = I_s$
$M_{k,k} = k-1$
For $t \in \{k, \dots n\}$
For $\tau \in \text{ListCandidate}_k$
$H_{k,t}^{\tau}(\theta) = H_{k,t-1}^{\tau}(\theta) + \gamma(y_t, \theta)$
$I_{k,t}^{\tau} = \left\{ \theta | H_{k,t}^{\tau}(\theta) \leqslant C_{k-1,t} + g(\theta) \right\}$
$S_{k,t}^{\tau} = S_{k,t-1}^{\tau} \cap I_{k,t}^{\tau}$
EndFor
$C_{k,t} = \min_{\tau \in \text{ListCandidate}_k} \left\{ \min_{\theta \in I_s} H_{k,t}^{\tau}(\theta) \right\}$
$M_{k,t} = \arg\min_{\tau \in \text{ListCandidate}_k} \left\{ \min_{\theta \in I_s} H_{k,t}^{\tau}(\theta) \right\}$
For $\tau \in \text{ListCandidate}_k$
If $[S_{k,t}^{\tau} = \emptyset]$
$\text{ListCandidate}_k = \text{ListCandidate}_k \setminus \{\tau\}$
EndIf
EndFor
$S_{k,t}^{t} = I_s \setminus (\cup_{\tau \in \text{ListCandidate}_k} I_{k,t}^{\tau})$
If $[S_{k,t}^{t} \neq \emptyset]$
$\text{ListCandidate}_k = \text{ListCandidate}_k \cup \{t\}$
$H_{k,t}^{t}(\theta) = C_{k-1,t} + g(\theta)$
EndIf
EndFor
EndFor

Proof of Proposition 0.3

Searching for one change-point Let us first consider a segmentation in 2 segments with a breakpoint at t. We define $P_t(\theta_1, \theta_2)$, the cost of this segmentation given some parameter θ_1 for the first segment and θ_2 for the second segment:

$$P_t(\theta_1, \theta_2) = \sum_{i=1}^{t} \gamma(y_i, \theta_1) \ + \ \sum_{i=t+1}^{n} \gamma(y_i, \theta_2).$$

The optimal cost P_t is:

$$P_t = min_{\theta_1} \left\{ \sum_{i=1}^{t} \gamma(y_i, \theta_1) \right\} + min_{\theta_2} \left\{ \sum_{i=t+1}^{n} \gamma(y_i, \theta_2) \right\}.$$

Having these notations, let us prove the following lemma:

Lemma 0.4.

- *If* $t_1 = 1$ *and* $t_2 = n$ *then* $\forall\, t\ P_t \geq C_{1,n}$
- *If* $t_1 = 1$ *and* $t_2 < n$ *then* $\forall\, t_1 - 1 \leq t \leq t_2$ *we have* $P_t \geq P_{t_2}$

- *If* $t_1 > 1$ *and* $t_2 = n$ *then* $\forall\, t_1 - 1 \leq t \leq t_2$ *we have* $P_t \geq P_{t_1-1}$
- *If* $t_1 > 1$ *and* $t_2 < n$ *then* $\forall\, t_1 - 1 \leq t \leq t_2$ *we have* $P_t \geq min\left\{P_{t_1-1}, P_{t_2}\right\}$

Proof

First scenario [$t_1 = 1$ and $t_2 = n$] We have:

$$P_t = t.min_{\theta_1}\left\{\gamma(y_1,\theta_1)\right\} + (n-t).min_{\theta_2}\left\{\gamma(y_1,\theta_2)\right\} = C_{1,n}.$$

Thus we get: $P_t \geq C_{1,n}$.

Second scenario [$t_1 = 1$ and $t_2 < n$] For any t such that $t \leq t_2$ we have:

$$P_t = t.min_\theta\left\{\gamma(y_1,\theta)\right\} + min_\theta\left\{(t_2-t)\gamma(y_1,\theta) + \sum_{i=t_2+1}^{n}\gamma(y_i,\theta)\right\}.$$

Thus we have:

$$P_t \geq t.min_\theta\left\{\gamma(y_1,\theta_1)\right\} + (t_2-t).min_\theta\left\{\gamma(y_1,\theta)\right\} + min_\theta\left\{\sum_{i=t_2+1}^{n}\gamma(y_i,\theta)\right\}.$$

And we get $\forall\, t \leq t_2 \quad P_t \geq P_{t_2}$.

Third scenario [$t_1 > 1$ and $t_2 = n$] We get $\forall\, t_1 - 1 \leq t \quad P_t \geq P_{t_1-1}$ by reversing the index and using scenario 2.

Fourth scenario [$t_1 > 1$ and $t_2 < n$] For any t such that $t_1 - 1 \leq t \leq t_2$ we obtain:

$$P_t(\theta_1,\theta_2) = \sum_{i=1}^{t_1-1}\gamma(y_i,\theta_1) + \sum_{i=t_2+1}^{n}\gamma(y_i,\theta_2) + (t-t_1+1)\gamma(y_{t_1},\theta_1) + (t_2-t)\gamma(y_{t_1},\theta_2).$$

Thus, for fixed θ_1 and θ_2 and for $t \in [t_1-1, t_2]$, $P_t(\theta_1,\theta_2)$ is a linear function of t. Thus we obtain that for any θ_1 and θ_2:

$$P_t(\theta_1,\theta_2) \geq min\left\{P_{t_1-1}(\theta_1,\theta_2), P_{t_2}(\theta_1,\theta_2)\right\} \geq min\left\{P_{t_1-1}, P_{t_2}\right\}.$$

As this is true for any θ_1 and θ_2 we get $P_t \geq min\left\{P_{t_1-1}, P_{t_2}\right\}$ ■

Proof of the main proposition Assume that we have a segmentation m in $\mathcal{M}_{K,n}$ with a breakpoint τ_k in a plateau. Then applying lemma 0.4 on the sequence $\{y_i\}_{i\in\{\tau_{k-1},\ldots\tau_{k+1}\}}$ we see that τ_k can either be discarded or moved to $t_1 - 1$ or t_2 without increasing the cost. Thus there exists a segmentation in K or fewer segments without any change-point in the plateau such that its optimal cost is $C_{K,n}$. ■

This theorem is more subtle than we might have thought based on our intuition. It does not mean that a change-point in a plateau is never optimal but only that it is not necessary to have change-points in plateaux to achieve optimality.

Abbreviations

PELT: Pruned exact linear time; PDP: Pruned dynamic programming; AIC: Akaike information criterion; BIC: Bayesian information criterion; NCBI: National Center for Biotechnology Information; SGD: Saccharomyces genome database.

Competing interests

The authors have no competing interest to declare.

Authors' contributions

AC co-wrote the C++ code, wrote the R-package, performed data analysis and co-wrote the manuscript. MK co-wrote the C++ code. EL co-supervised the work and co-wrote the manuscript. GR co-wrote the C++ code, and co-wrote the manuscript. SR co-wrote the manuscript and co-supervised the work. All authors read and approved the final manuscript.

Acknowledgements

We thank Véronique Brunaud for providing the RNA-seq profile of *Arabidopsis Thaliana*. We also thank our anonymous referee for helpful comments on the presentation of the algorithm.

Author details

[1]AgroParisTech, UMR 518, 16 rue Claude Bernard, 75231 Paris Cedex 05, France. [2]INRA, UMR 518, 16 rue Claude Bernard, 75231 Paris Cedex 05, France. [3]Unité de Recherche en Génomique Végétale (URGV) INRA-CNRS-Université d'Evry Val d'Essonne, 2 Rue Gaston Crémieux, 91057 Evry Cedex, France.

Received: 13 May 2013 Accepted: 3 March 2014
Published: 10 March 2014

References

1. Braun JV, Muller HG: **Statistical methods for DNA sequence segmentation.** *Stat Sci* 1998, **13**(2):142–162.
2. Durot C, Lebarbier E, Tocquet AS: **Estimating the joint distribution of independent categorical variables via model selection.** *Bernoulli* 2009, **15:**475–507.
3. Bockhorst J, Jojic N: **Discovering patterns in biological sequences by optimal segmentation.** In *Proceedings of the 23rd Conference in Uncertainty in Artificial Intelligence*: AUAI Press; 2007.
4. Zhang Z, Lange K, Sabatti C: **Reconstructing DNA copy number by joint segmentation of multiple sequences.** *BMC Bioinformatics* 2012, **13:**205.
5. Erdman C, Emerson JW: **A fast Bayesian change point analysis for the segmentation of microarray data.** *Bioinformatics* 2008, **24**(19):2143–2148.
6. Olshen AB, Venkatraman ES, Lucito R, Wigler M: **Circular binary segmentation for the analysis of array-based DNA copy number data.** *Biostat (Oxford, England)* 2004, **5**(4):557–572.
7. Picard F, Robin S, Lavielle M, Vaisse C, Daudin J: **A statistical approach for array CGH data analysis.** *BMC Bioinformatics* 2005, **6:**27.
8. Picard F, Lebarbier E, Hoebeke M, Rigaill G, Thiam B, Robin S: **Joint segmentation, calling and normalization of multiple CGH profiles.** *Biostatistics* 2011, **12**(3):413–428.
9. Chiang DY, Getz G, Jaffe DB, O'Kelly MJ, Zhao X, Carter SL, Russ C, Nusbaum C, Meyerson M, Lander ES: **High-resolution mapping of copy-number alterations with massively parallel sequencing.** *Nat Methods* 2009, **6:**99–103.
10. Xie C, Tammi MT: **CNV-seq, a new method to detect copy number variation using high-throughput sequencing.** *BMC Bioinformatics* 2009, **10:**80.

11. Yoon S, Xuan Z, Makarov V, Ye K, Sebat J: **Sensitive and accurate detection of copy number variants using read depth of coverage.** *Genome Res* 2009, **19:**1586–1592.
12. Boeva V, Zinovyev A, Bleakley K, Vert JP, Janoueix-Lerosey I, Delattre O, Barillot E: **Control-free calling of copy number alterations in deep-sequencing data using GC-content normalization.** *Bioinformatics (Oxford, England)* 2011, **27:**268–9.
13. Shen JJ, Zhang NR: **Change-point model on nonhomogeneous Poisson processes with application in copy number profiling by next-generation DNA sequencing.** *Ann Appl Stat* 2012, **6**(2):476–496.
14. Rivera C, Walther G: **Optimal detection of a jump in the intensity of a Poisson process or in a density with likelihood ratio statistics.** *Scand J Stat* 2013, **40**(4):752–769.
15. Franke J, Kirch C, Kamgaing JT: **Changepoints in times series of counts.** *J Time Series Anal* 2012, **33**(5):757–770.
16. Killick R, Fearnhead P, Eckley I: **Optimal detection of changepoints with a linear computational cost.** *J Am Stat Assoc* 2012, **107**(500):1590–1598.
17. Hocking TD, Schleiermacher G, Janoueix-Lerosey I, Boeva V, Cappo J, Delattre O, Bach F, Vert J-P: **Learning smoothing models of copy number profiles using breakpoint annotations.** *BMC Bioinformatics* 2013, **14**(1):164.
18. Rigaill G: **Pruned dynamic programming for optimal multiple change-point detection.** *Arxiv:1004.0887* 2010. [http://arxiv.org/abs/1004.0887]
19. Johnson N, Kemp A, Kotz S: *Univariate Discrete Distributions*: John Wiley & Sons Inc.; 2005.
20. Risso D, Schwartz K, Sherlock G, Dudoit S: **GC-Content normalization for RNA-Seq data.** *BMC Bioinformatics* 2011, **12:**480.
21. Bullard J, Purdom E, Hansen K, Dudoit S: **Evaluation of statistical methods for normalization and differential expression in mRNA-Seq experiments.** *BMC Bioinformatics* 2010, **11:**94.
22. Akaike H: **A new look at the statistical model identification.** *Automatic Control IEEE Trans* 1974, **19**(6):716–723.
23. Yao Y: **Estimation of a noisy discrete-time step function: Bayes and empirical Bayes approaches.** *Ann Stat* 1984, **12**(4):1434–1447.
24. Zhang NR, Siegmund DO: **A modified Bayes information criterion with applications to the analysis of comparative genomic hybridization data.** *Biometrics* 2007, **63:**22–32. [PMID: 17447926].
25. Cleynen A, Lebarbier E: **Segmentation of the poisson and negative binomial rate models: a penalized estimator.** *Esaim: P & S* 2014. arXiv preprint arXiv:1301.2534.
26. Lebarbier E: **Detecting multiple change-points in the mean of Gaussian process by model selection.** *Signal Process* 2005, **85**(4):717–736.
27. Arlot S, Massart P: **Data-driven calibration of penalties for least-squares regression.** *J Mach Learn Res* 2009, **10:**245–279. (electronic).
28. Luong TM, Rozenholc Y, Nuel G: **Fast estimation of posterior probabilities in change-point analysis through a constrained hidden Markov model.** *Comput Stat Data Anal* 2013.

doi:10.1186/1748-7188-9-6
Cite this article as: Cleynen *et al.*: Segmentor3IsBack: an R package for the fast and exact segmentation of Seq-data. *Algorithms for Molecular Biology* 2014 **9**:6.