Learning smoothing models of copy number profiles using breakpoint annotations

Toby Dylan Hocking, Gudrun Schleiermacher, Isabelle Janoueix-Lerosey,
Olivier Delattre, Francis Bach, Jean-Philippe Vert

Learning smoothing models of copy number profiles using breakpoint annotations

Toby Dylan Hocking [1,2,3,4], Gudrun Schleiermacher [5],
Isabelle Janoueix-Lerosey [5], Olivier Delattre [5],
Francis Bach [1] and Jean-Philippe Vert [2,3,4]
[1]INRIA – Sierra project-team, Paris, F-75013,
[2] Centre for computational biology, Mines ParisTech, Fontainebleau, F-77300,
[3] Institut Curie, [4] INSERM U900 and [5] INSERM U830, Paris, F-75248, France

January 27, 2012

Abstract

Motivation: Many models have been proposed to detect breakpoints in chromosomal copy number profiles, but it is usually not obvious to decide which is most effective for a given data set. Furthermore, most methods have a smoothing parameter that determines the number of breakpoints and must be chosen using various heuristics.

Results: We present three contributions toward automatic training of smoothing models. First, we propose to select the model and degree of smoothness that maximizes agreement with visual breakpoint region annotations. Second, we develop cross-validation procedures to estimate the error of the trained models. Third, we apply these methods to a new database of annotated neuroblastoma copy number profiles, which we make available as a public benchmark for testing new algorithms. Whereas previous studies have been qualitative or limited to simulated data, our approach is quantitative and suggests which algorithms are fastest and most accurate in practice on real data.

Availability: Copy number profiles can be annotated using a GUI:
`http://pypi.python.org/pypi/annotate_regions`
The annotated neuroblastoma copy number profiles are available in R: `data(neuroblastoma,package="bams")`
`http://cran.r-project.org/web/packages/bams`
Contact: `toby.hocking@inria.fr`

Contents

1   Introduction: the need for smoothing model selection criteria

DNA copy number alterations (CNAs) can result from various types of genomic rearrangements, and are important in the study of many types of cancer (Weinberg, 2006). In particular, clinical outcome of patients with neuroblastoma has been shown to be worse for tumors with segmental alterations or breakpoints in specific genomic regions (Janoueix-Lerosey *et al.*, 2009; Schleiermacher *et al.*, 2010). Thus, to construct an accurate predictive model of clinical outcome for these tumors, we must first accurately detect the precise location of each breakpoint.

In recent years, array comparative genomic hybridization (aCGH) and single nucleotide polymorphism (SNP) microarrays have been developed as genome-wide assays for CNAs. In parallel, there have been many new mathematical models proposed to smooth the noisy signals from these microarray assays in order to recover the CNAs (Hupé *et al.*, 2004; Picard *et al.*, 2005; Venkatraman and Olshen, 2007; Tibshirani and Wang, 2007; Ben-Yaacov and Eldar, 2008; Hoefling, 2009). Each model has different assumptions about the data, and it is not obvious to decide which model is appropriate for a given data set.

Furthermore, most models have parameters that control the degree of smoothness. Varying these smoothing parameters will vary the number of detected breakpoints. Most authors give default values that accurately detect breakpoints on some data, but do not necessarily generalize well to other data. There are some specific criteria for choosing the degree of smoothness in some models (Lavielle, 2005; Zhang and Siegmund, 2007; Zhang *et al.*, 2010), but the mathematical assumptions of these models are not often verified in real noisy microarray data, which can lead to poor estimation of CNAs.

In practice, software tools such as VAMP (La Rosa *et al.*, 2006) are often used to normalize the noisy microarray signals, then plot them against genomic position for interpretation by an expert biologist looking for CNAs. Indeed, to motivate the use of their cghFLasso smoothing model, Tibshirani and Wang (2007) write "The results of a CGH experiment are often interpreted by a biologist, but this is time consuming and not necessarily very accurate."

In contrast, the first contribution of this paper is the idea that the expert interpretation of the biologist is very accurate and in fact valuable for model selection. To tune the smoothness parameter in practice, the biologist will often examine plots of the microarray signal with a smoothed model, changing the smoothness parameter until the model seems to capture all the visible breakpoints the data. In Section 2, we make this intuition concrete by defining a training protocol based on visual annotations that can quantify the accuracy of a smoothing method. We note that using databases of visual annotations is not a new idea, and has been used successfully for object recognition and detection in the field of computer vision (Russell *et al.*, 2008). In bioinformatics, Shah *et al.* (2006) proposed a hidden Markov model with position-specific prior probabilities for copy number variations, but no previous models have attempted to learn the degree of smoothness using breakpoint annotations.

Our second contribution is a protocol to estimate the breakpoint detection ability of the trained smoothing models on real data. In Section 3, we propose to estimate the false positive and false negative rates of the trained models using cross-validation. This provides an objective criterion for deciding which smoothing algorithms are appropriate for which data.

The third contribution of this paper is a systematic, quantitative comparison of the accuracy of several common smoothing algorithms on a new database of annotated neuroblastoma copy number profiles, which we give in Section 5. For simulated data, Willenbrock and Fridlyand (2005) compared the breakpoint detection ability of GLAD, DNAcopy, and a hidden Markov model by examining false positive and false negative rates. But there are no previous studies that quantitatively compare smoothing models on real data. This is because quantifying smoothing model accuracy using annotated region data is a new idea, and so we make the annotated neuroblastoma copy number profiles available as a benchmark for the community to test new algorithms on real data.

Several authors have recently proposed methods for so-called joint segmentation of multiple CGH profiles, under the hypothesis that each profile shares breakpoints in the exact same location (Vert and Bleakley, 2010; Picard *et al.*, 2011; Ritz *et al.*, 2011). These models are not useful in our setting, since we assume that breakpoints do not occur in the exact same locations across copy number profiles. Instead, we focus on the case of learning a model that will accurately detect an unknown number of breakpoints in a copy number profile.

2 Training smoothing models using breakpoint annotations

The first contribution of this paper is a smoothing model training protocol that uses visually determined breakpoint annotations to quantify model accuracy, which we explain in this section.

2.1 Detecting breakpoints using smoothing models

Assume that we have observed n chromosomal copy number profiles, each with an unknown number of breakpoints at unknown locations. We would like to recover the breakpoints accurately using a model with parameter λ that controls the degree of smoothness. As shown in Figure 1, we represent the d probes on chromosome c of profile i using the following numbers:

$$
\begin{array}{cccccll}
p_1 & \leq & \ldots & \leq & p_d & \in \mathbb{N} & \text{positions on chromosome } c \\
y_1 & , & \ldots & , & y_d & \in \mathbb{R} & \text{logratio measurements} \\
\hat{y}_1^\lambda & , & \ldots & , & \hat{y}_d^\lambda & \in \mathbb{R} & \text{smoothed profile}
\end{array}
$$

We define the breakpoints predicted on this chromosome as the set of positions where there are jumps in the smoothed signal:

$$
\hat{B}_{ic}^\lambda = \left\{ (p_j + p_{j+1})/2 \mid \hat{y}_j^\lambda \neq \hat{y}_{j+1}^\lambda, \ \forall j = 1, \ldots, d-1 \right\} \tag{1}
$$

Then, we define $\hat{B}_i^\lambda$ to be the complete set of genomic breakpoints predicted by algorithm λ for profile i, over all chromosomes c.

2.2 Breakpoint annotations quantify model accuracy

Intuitively, by visual inspection of the noisy signal, it is not obvious to locate the exact location of a breakpoint, but it should be easy to determine whether or not a region contains a breakpoint. So rather than defining annotations in terms of precise breakpoint locations, we instead define them in terms of regions. We define a genomic region R_k as the subset of genomic positions on chromosome c_k between the min $\underline{r}_k$ and max $\overline{r}_k$.

So, we define the breakpoint annotation for profile i in region k as

$$
b_{ik} = \begin{cases} 0 & \text{if profile } i \text{ has no breakpoints in } R_k \\ 1 & \text{if profile } i \text{ has at least 1 breakpoint in } R_k, \end{cases} \tag{2}
$$

which can be determined by visual inspection of the scatterplot of logratio measurements y versus position p.

The idea for model selection is to choose λ such that the predicted breakpoints $\hat{B}_i^\lambda$ agree with the annotations b_{ik}, as shown in Figure 1. To quantify this, for each region k, we predict 0 if there are no predicted breakpoints in the region, and 1 if there is at least 1 predicted breakpoint:

$$
\hat{b}_{ik}^\lambda = \begin{cases} 0 & \text{if } R_k \cap \hat{B}_i^\lambda = \emptyset \\ 1 & \text{otherwise} \end{cases} \tag{3}
$$

We can measure the error of a model at region k on profile i with the indicator function

$$
E_i^k(\lambda) = \begin{cases} 0 & \text{if } b_{ik} = \hat{b}_{ik}^\lambda \\ 1 & \text{otherwise} \end{cases} \tag{4}
$$

and with respect to an entire profile i using

$$
E_i^{\text{local}}(\lambda) = \sum_k E_i^k(\lambda) \tag{5}
$$

We define the local model as the model obtained by choosing a different λ_i to minimize Equation 5 for each profile i.
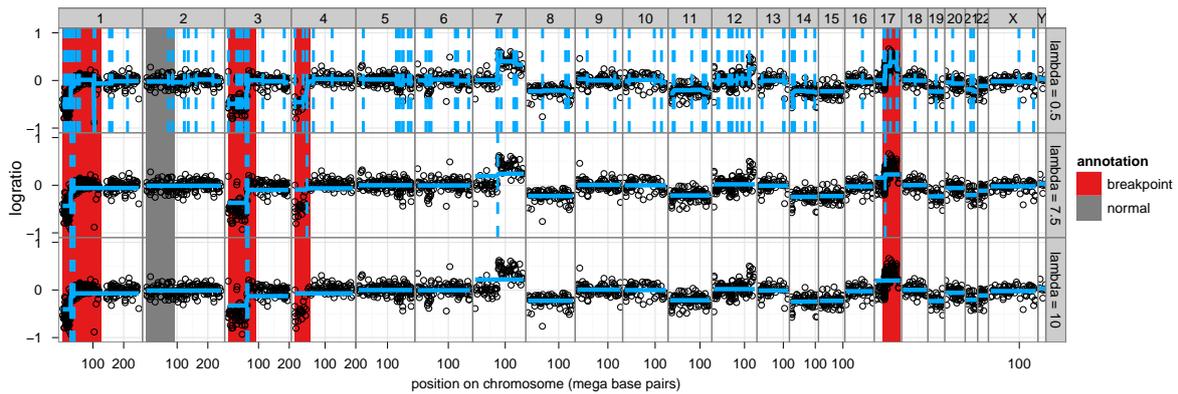
Figure 1: Model agreement to annotated regions can be measured by examining the positions of predicted breakpoints $\hat{B}_i^\lambda$ (dashed vertical blue lines) observed in the smoothing model $\hat{y}^\lambda$ (solid blue lines). Black circles show logratio measurements y plotted against position p for a single profile $i = 375$. Chromosomes are shown in panels from left to right, and different values of the smoothing parameter λ in the flsa model are shown in panels from top to bottom. Models with too many breakpoints ($\lambda = 0.5$) and too few breakpoints ($\lambda = 10$) are suboptimal, so we pick an intermediate model ($\lambda = 7.5$) that maximizes agreement with the annotations, thus detecting a new breakpoint on chromosome 7 which was not annotated.
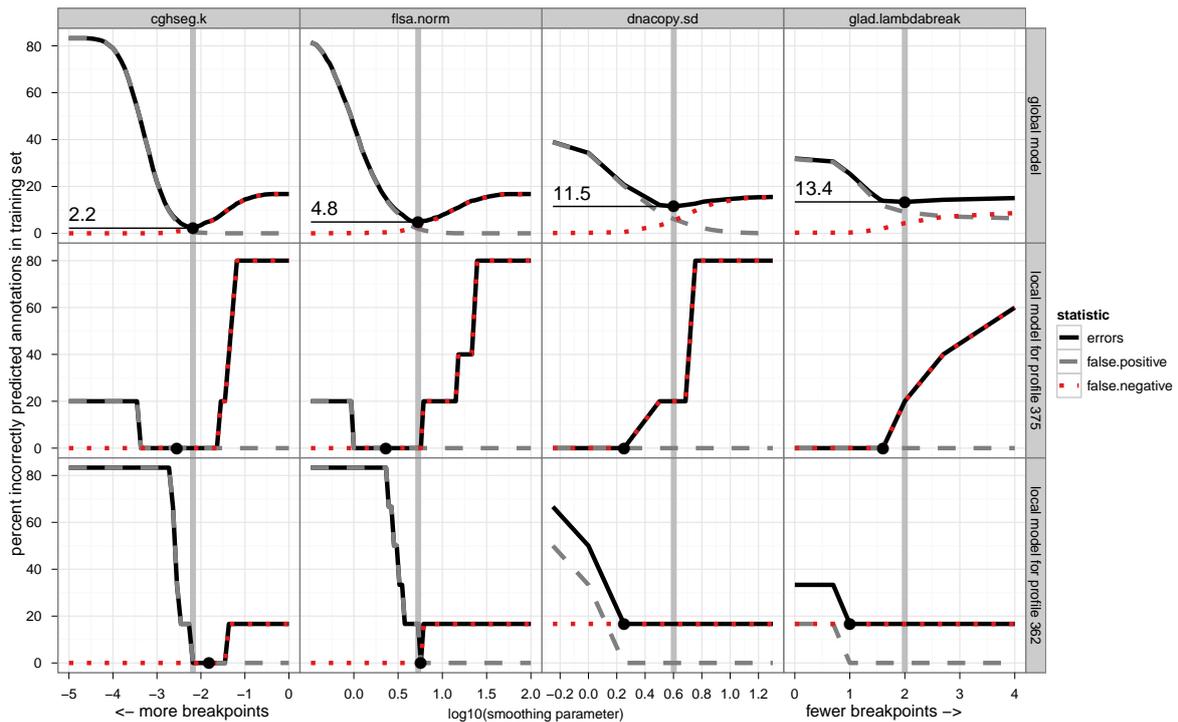
Figure 2: Training error functions for global and local models plotted against smoothing parameter λ. In the top row panels, we plot $E^{\mathrm{global}}(\lambda)$ from Equation 6, and in the other rows, we plot $E_i^{\mathrm{local}}(\lambda)$ from Equation 5. Each column of plots shows the error of a particular algorithm, and the minimum chosen using the global training procedure is shown using a vertical grey line. Note that the local model training error can be reduced by moving from the globally optimal smoothing parameter λ^* to a local value λ_i, as in profile $i = 375$ for algorithms dnacopy.sd and glad.lambdabreak. For the local models trained on single profiles, there are at most 6 training examples, so many smoothing parameters attain the minimum. Thus, we use the protocol described in section 2.3 to pick the best value, shown as a black dot.

4

However, we can learn a globally optimal smoothing parameter λ by minimizing the error with respect to all the profiles:

$$E^{\text{global}}(\lambda) = \sum_{i=1}^{n} E_i^{\text{local}}(\lambda). \tag{6}$$

We define the global model as the model obtained by choosing a smoothing parameter λ^* that minimizes Equation 6.

2.3 Picking the optimal model

We assume that λ is a tuning parameter that will be directly related to the number of breakpoints observed in the smoothed signal. In other words, this training procedure requires that the number of breakpoints predicted by the algorithm is monotonic in λ, which is usually the case.

Fix a set of smoothing parameters $\lambda \in \Lambda$, and run the smoothing algorithm with each of these parameters. Intuitively, we should pick the value of λ that maximizes agreement with annotation data. For global models, we attempt to minimize Equation 6, and there is usually one best value, λ^*.

However, for the local model for profile i, we want to minimize the local error as defined in Equation 5. Since the training set consists of only the annotations of one profile i, there may be no unique smoothing parameter λ that minimizes the error. We propose to pick between models that achieve the minimum number of errors based on the shape of the error curve, and these cases are illustrated in Figure 2.

1. When the minimum error is achieved in a range of intermediate parameter values, we pick a value in the middle. This occurs in the local error curves shown for flsa.norm and cghseg.k.

2. When the minimum is attained by the model with the most breakpoints, we pick the model with the fewest breakpoints that has the same error. This attempts to minimize the false positive rate. This occurs for profile $i = 375$ with models dnacopy.sd and glad.lambdabreak.

3. When the minimum is attained by the model with the fewest breakpoints, we pick the model with the most breakpoints that has the same error. This attempts to minimize the false negative rate, and occurs for profile $i = 362$ with models dnacopy.sd and glad.lambdabreak.

3 Estimating the error of the trained model using cross-validation

The second contribution of this paper are cross-validation procedures that estimate the generalization error of the trained smoothing models. The main idea is to use a training set of annotations to learn the parameter of a smoothing model, then quantify the error of the model using a test set of annotations. Notably, this enables quantitative comparison of smoothing models on copy number profiles from real microarray data.

3.1 Leave-one-out cross-validation for comparing local and global models

To compare the breakpoint detection performance of local and global models on un-annotated regions, we propose leave-one-out cross-validation on regions.

- For each annotated region k:

  1. Designate R_k as the test region, and set aside the annotations in this region from all the profiles.
  2. Using all the other annotations as a training set, pick the best λ using the protocol described in Section 2.3. For local models we learn a profile-specific λ_i that minimizes E_i^{local}, and for global models we learn a global λ that minimizes E^{global}.
  3. To estimate how the model generalizes, count the errors of the learned model in the test region R_k.

- To estimate the ability of the trained model to predict breakpoints at a general un-annotated region, take the mean test error over all regions.

3.2 n/t-fold cross-validation to estimate error on un-annotated profiles

Since the annotation process is time-consuming, we are interested in training an accurate breakpoint detector with as few annotations as possible. Thus we would like to answer the following question: how many profiles t do I need to annotate before I get a global model that will generalize well to all the other profiles?

To answer this question, we estimate the error of a global model trained on the annotations from t profiles using cross-validation. We divide the set of n annotated profiles into exactly $\lfloor n/t \rfloor$ folds, each with approximately t profiles. For each fold, we consider its annotations a training set for a global model, and combine the other folds as a test set to quantify the model error. The final estimate of generalization error is then the average model error over all folds.

min $= \underline{r}_k$	max $= \overline{r}_k$	chrom c_k	breakpoint	normal	(all)
0	125000000	1	103	464	567
0	93300000	2	110	464	574
0	91000000	3	43	531	574
0	50400000	4	35	534	569
53700000	135006516	11	107	464	571
24000000	81195210	17	175	388	563
		(all)	573	2845	3418

Table 1: Counts of normal and breakpoint annotations in the neuroblastoma data set, conditional on region. Min and max limits of each region are shown in base pairs, in reference to the Hg19 Human genome assembly.

4 Data and models

4.1 Neuroblastoma copy number data

We analyzed a new data set of $n = 575$ copy number profiles from aCGH and SNP microarray experiments on neuroblastoma tumors taken from patients at diagnosis at the Institut Curie. In this article we analyze the normalized logratio measurements of these microarrays, which we have made available in R package `bams` on CRAN.

Six chromosome arms known to be associated with prognostic impact were annotated in the microarray data set (Janoueix-Lerosey *et al.*, 2009). Each region $R_1, \ldots, R_6$ was defined by the start and end of a chromosome arm, and the genomic coordinates of these regions are given in Table 1.

For each profile i, our domain expert annotated each region k by examining the plotted profile in VAMP (La Rosa *et al.*, 2006) and recording 0 or 1 in a spreadsheet, according to the definition of breakpoint annotations in Equation 2. Table 1 shows counts of annotations per region, and Table 2 shows counts of annotations per profile. Some profiles have less than 6 annotations since regions for which breakpoints could not be determined by visual inspection were not included in the analysis. The annotations are shown as colored rectangles in Figure 1, and are available in R package `bams` on CRAN.

4.2 Smoothing models

In this study we considered smoothing models from the bioinformatics literature with free software implementations available as R packages on CRAN, R-Forge, or Bioconductor (R Development Core Team, 2011; Theußl and Zeileis, 2009; Gentleman *et al.*, 2004).

We used version 1.03 of the `flsa` package from CRAN to calculate the Fused Lasso Signal Approximator as described by Hoefling (2009). The FLSA solves the following optimization problem:

$$\hat{y}^\lambda = \arg\min_{\beta \in \mathbb{R}^d} \frac{1}{2} \sum_{i=1}^d (y_i - \beta_i)^2 + \lambda_1 \sum_{i=1}^d |\beta_i| + \lambda_2 \sum_{i=1}^{d-1} |\beta_i - \beta_{i+1}|. \tag{7}$$

We define a grid of values $\lambda \in \{10^{-5}, \ldots, 10^{12}\}$, take $\lambda_1 = 0$, and consider the following parameterizations for λ_2:

- flsa: $\lambda_2 = \lambda$.

- flsa.norm: $\lambda_2 = \lambda d \times 10^6 / l$ where d is the number of points and l is the length of the chromosome in base pairs.

We used version 1.29.0 of the `DNAcopy` package from Bioconductor to fit the circular binary segmentation model of Venkatraman and Olshen (2007). We varied the degree of smoothness by adjusting the `undo.SD`, `undo.prune`, and `alpha` parameters of the `segment` function.

We used version 0.2-1 of the `cghFLasso` package from CRAN, which implements the method of Tibshirani and Wang (2007), but does not provide any smoothness parameters for breakpoint detection.

Normal annotations	Breakpoint annotations						
	0	1	2	3	4	5	6
0	0	0	0	1	0	0	2
1	0	0	0	0	3	9	0
2	0	0	0	5	29	0	0
3	0	1	3	60	0	0	0
4	0	8	64	0	0	0	0
5	8	47	0	0	0	0	0
6	335	0	0	0	0	0	0

Table 2: Counts of profiles in the neuroblastoma data set, conditional on number of annotations. Note that most profiles have more normal regions than breakpoint regions. For example, 335 profiles have all 6 regions annotated as normal.

We used version 2.17.0 of the `GLAD` package from Bioconductor to fit the GLAD adaptive weights smoothing model of Hupé *et al.* (2004). We varied the degree of smoothness by adjusting the `lambdabreak` and `MinBkpWeight` parameters of the `daglad` function. For the glad.haarseg model, we used the `smoothfunc="haarseg"` option and varied the `breaksFdrQ` parameter to fit the wavelet smoothing model of Ben-Yaacov and Eldar (2008).

We used version 0.01 of the `cghseg` package from R-Forge to fit the maximum-likelihood piecewise constant smoothing model of Picard *et al.* (2005). We used the `segmeanCO` function with `kmax=20` to obtain the maximum-likelihood piecewise constant smoothing model $\hat{y}^k$ for $k = 1, \ldots, 20$ segments. Lavielle (2005) suggested penalizing k breakpoints in a signal sampled at d points using λk, and varying λ as a tuning parameter. We implemented this model selection criterion as the cghseg.k model, for which we define the optimal number of segments

$$k^*(\lambda) = \underset{k \in \{1, \ldots, 20\}}{\arg\min} \ \lambda k + \frac{1}{d} \sum_{i=1}^{d} (y_i - \hat{y}_i^k)^2, \tag{8}$$

and the optimal smoothing $\hat{y}^\lambda = \hat{y}^{k^*(\lambda)}$. For the cghseg.mBIC model, we used the modified Bayesian information criterion described by Zhang and Siegmund (2007), which has no smoothness parameter, and is implemented in the `uniseg` function of the `cghseg` package.

5 Results

All the algorithms from Section 4.2 were applied to all the annotated neuroblastoma copy number profiles in the data set described in Section 4.1. However, the dnacopy.prune algorithm was too slow (> 24 hours) for some of the profiles with many data points, so these profiles were excluded from the analysis of dnacopy.prune. Note that to decrease computation time, the model fitting may be trivially parallelized for profiles, algorithms, and smoothing parameter values.

5.1 Among global models, cghseg.k exhibits the smallest training error in the neuroblastoma data

The global and local training procedures were applied to the entire set of annotated profiles. Training error curves for flsa.norm, cghseg.k, dnacopy.sd, and glad.lambdabreak are shown in Figure 2. Note that the global curves do not achieve zero training error but the local curves often do, suggesting that the local training strategy may be useful in some cases. Also note the inflexibility of the dnacopy.sd and glad.lambdabreak models, which do not detect a breakpoint in profile $i = 362$, even at the smallest parameter value, corresponding to the model with the most breakpoints. Finally, note the minimum error of 2.2% achieved by cghseg.k, the global model with the smallest training error.

The ROC curves for the training error of the global models for each algorithm are traced in Figure 3. It is clear that the default parameters of each algorithm show relatively large false positive rates. In contrast, the models chosen by maximizing agreement with the breakpoint annotation data exhibit smaller false positive rates at the cost of smaller true positive rates. The ROC curves suggest that the cghseg.k algorithm is the most discriminative for breakpoint detection in the neuroblastoma data.
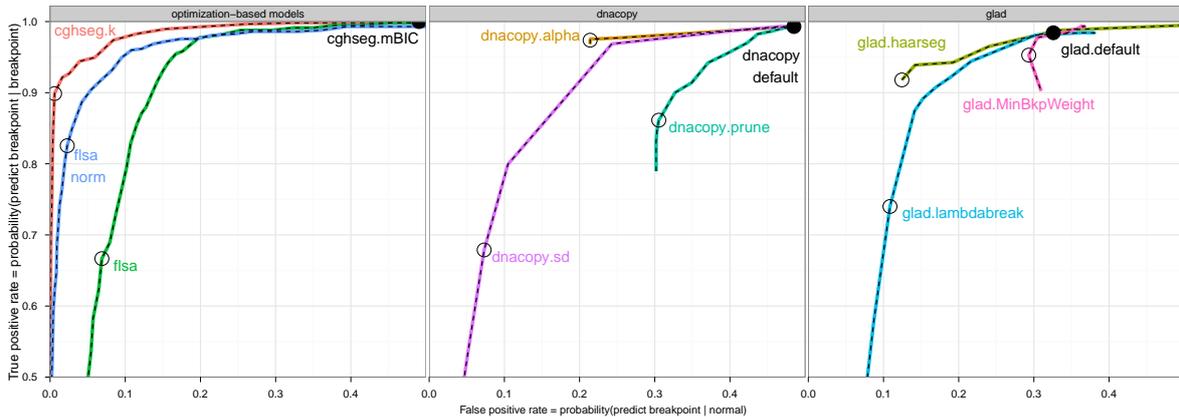

Figure 3: ROC curves for the training error with respect to the breakpoint annotation data are shown as colored lines. The curves are shown in 3 panels zoomed to the upper left region of ROC space to avoid visual clutter. Each curve is traced by plotting the error of a model as the degree of smoothness is varied, and an empty black circle shows the global model chosen by minimizing the error with respect to all annotations. Algorithms with no tuning parameters are shown as black dots. Note that some ROC curves appear incomplete since some segmentation algorithms are not flexible enough for the task of breakpoint detection, even though we ran each algorithm on a very large range of smoothness parameter values.

5.2   Global models generalize better than local models

The leave-one-out cross-validation protocol was used to contrast the test error of the models trained using the local and global procedures. Table 3 shows the error, false positive, and false negative rates of each model, averaged over the 6 test regions.

It is clear that the training procedure makes no difference for models glad.default, dnacopy.default, cghseg.mBIC, and cghFLasso, which have no smoothness parameters. The large error of these models suggest that the assumptions of their default parameter values do not hold in the neuroblastoma data set. More generally, these error rates suggest that smoothness parameter tuning is critically important to obtain an accurate smoothing of real copy number profiles.

For dnacopy.prune, glad.MinBkpWeight, glad.lambdabreak, and flsa, there appears to be little difference between the local and global training procedures. For models flsa.norm and cghseg.k, there seems to be a clear advantage for the global models which share information between profiles. Indeed, the cghseg.k model shows the minimal estimated error of only 2.2% on these data, with a moderately fast median training time of 2.1 seconds.

Finally, note that the false positive rate of locally trained models is higher than the false negative rate for most algorithms. This can be explained by the larger fraction of normal annotations present in the training set, and the fact that many profiles have only normal annotations (Table 2).

	Global			Local			Timings
	errors	FP	FN	errors	FP	FN	seconds
cghseg.k	2.2	0.6	11.6	11.1	13.0	7.0	2.10
flsa.norm	6.7	3.6	18.5	14.8	15.2	10.5	0.08
dnacopy.sd	11.5	7.6	32.2	14.4	12.3	26.9	51.62
glad.haarseg	11.8	12.6	8.0	20.0	23.8	2.3	29.51
glad.lambdabreak	14.1	12.3	23.0	15.5	15.1	18.0	14.44
flsa	16.0	12.7	36.6	14.4	15.9	10.3	0.04
dnacopy.alpha	18.4	21.9	2.6	25.5	30.8	1.1	25.93
glad.MinBkpWeight	25.2	30.0	4.3	23.9	28.1	4.7	40.88
*glad.default	27.4	33.3	1.2	27.4	33.3	1.2	1.13
dnacopy.prune	27.9	31.9	17.1	31.1	35.8	14.8	35.17
*dnacopy.default	40.5	49.3	0.5	40.5	49.3	0.5	1.78
*cghseg.mBIC	40.9	49.4	0.0	40.9	49.4	0.0	1.47
*cghFLasso	80.8	97.2	0.0	80.8	97.2	0.0	0.14

Table 3: Leave-one-out cross-validation over the 6 annotated regions was used to estimate breakpoint detection error, false positive (FP), and false negative (FN) rates. Each line shows the performance of one of the models described in Section 4.2. For models marked with an *asterisk, there are no smoothness parameters that can be learned using annotations. For the other models, global and local training procedures described in Section 2.3 were used to learn smoothness parameters. The global error is used to order the rows of the table. The Timings column shows the median time to fit the sequence of smoothing models for a single profile.

5.3 Only a few profiles need to be annotated for a good global model

Finally, to estimate the generalization error of a global model trained on a relatively small training set of t annotated profiles, we applied the n/t-fold cross-validation procedure to the data.

For several training set sizes t, we plot the error of the glad.lambdabreak, dnacopy.sd, flsa.norm, and cghseg.k models in Figure 4. It shows that adding more annotations to the training set decreases the error in general, but at a diminishing rate. The error curves flatten out near $t = 10$, suggesting that annotating 10 profiles is sufficient to get performance just as good as if all the profiles were annotated. Futhermore, it is clear that the minimum error is model-dependent, and we conjecture that it is also annotator-dependent.

This suggests the following protocol: annotate breakpoints in about 10 profiles, then use those annotations to train a global model. In Table 4, we used $n/10$-fold cross-validation to estimate the error rates of models trained using this protocol. These error estimates are slightly larger, but the model ordering is mostly unchanged, with respect to the leave-one-out cross-validated estimates of the global model error rates in Table 3. In particular, cghseg.k still shows the best performance on these data, with an estimated generalization error of 3.4%.
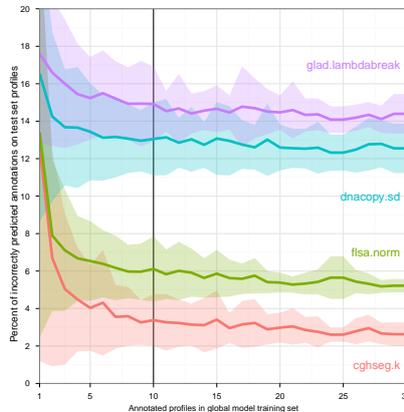

Figure 4: Cross-validation was used to estimate the generalization error of the global models with different sized training sets for several breakpoint detection algorithms. For each training set size t, the profiles were partitioned into training sets of approximately size t, then were evaluated using the annotations from all the other profiles. Results on these data indicate decreasing error (lines) and standard deviation (shaded bands) as the training set increases, with diminishing returns after approximately $t = 10$, indicated with a vertical black line, and shown in detail for all algorithms in Table 4.

model	errors	sd	FP	sd	FN	sd
cghseg.k	3.4	1.4	1.6	1.8	12.2	8.3
flsa.norm	6.1	1.7	3.2	2.7	20.5	13.8
glad.haarseg	12.6	1.5	13.7	2.0	7.1	1.2
dnacopy.sd	13.0	1.9	6.0	4.9	47.8	24.3
flsa	13.5	1.7	8.4	5.4	38.6	30.9
glad.lambdabreak	14.9	2.0	12.7	4.7	25.7	17.3
dnacopy.alpha	19.0	1.0	22.4	1.2	2.4	0.1
glad.MinBkpWeight	26.8	1.5	31.3	2.1	4.3	3.1
*glad.default	27.4	0.1	32.6	0.2	1.6	0.1
dnacopy.prune	28.5	1.6	31.7	2.4	12.8	2.9
*dnacopy.default	40.5	0.1	48.5	0.2	0.7	0.0
*cghseg.mBIC	40.8	0.1	49.1	0.2	0.0	0.0
*cghFLasso	80.9	0.1	97.2	0.0	0.0	0.0

Table 4: The $n/10$-fold cross-validation protocol was used to estimate error, false positive (FP), and false negative (FN) rates, shown in percent. Algorithms with an *asterisk have no smoothness parameters, but the smoothness parameter of the other algorithms was chosen using annotations from approximately $t = 10$ profiles.

6 Discussion and conclusions

We have proposed to train smoothing models using annotations determined by visual inspection of the copy number profiles. We have demonstrated that this approach allows quantitative comparison of smoothing models on real data, and gives an objective criterion for choosing the degree of smoothness.

The clear drawback of annotation-based model selection is the time required to create the annotations. However, we have shown diminishing returns after about $t = 10$ annotated profiles, so not much time should be needed in general to get good performance with models trained using annotations. Furthermore, we propose to streamline the annotation process by using a simple, portable, free software Python annotation GUI which is available as package `annotate_regions` from the Python Package Index.

In contrast with our results, in a previous work on automatic parameter tuning of smoothing models, Zhang *et al.* (2010) claim that local models should be better in some sense: "it is clear that the advantages of selecting individual-specific λ values outweigh the benefit of selecting constant λ values that maximize overall performance." However, they do not demonstrate this claim explicitly, and one of the contributions of this work is to show that global models often generalize better than local models, according to our leave-one-out estimates.

We have also shown that learning a global smoothness parameter on a limited set of annotations can generalize well to un-annotated profiles. However, the smoothness parameterization must be carefully chosen. For example, the flsa.norm algorithm scales the smoothness parameter λ by the number of points and the length of the chromosome, and results in lower error rates than the unscaled flsa algorithm. We are interested in investigating other parameterizations which could reduce the error even further.

In this work we have characterized the performance of several common algorithms on a set of annotated neuroblastoma copy number profiles, but it will be interesting to apply annotation-based model training to other algorithms and data sets. In particular, the ordering of algorithms may change when we apply these procedures to data generated by different microarrays and annotators. However, it is clear that in any data set, our approach will favor algorithms that capture breakpoints that agree with the annotator's visual definition of a breakpoint.

We have solved the problem of smoothness parameter selection using breakpoint annotations, but the biological question of detecting CNAs remains. By constructing a database of annotated regions of CNAs, we could use a similar approach to train models that detect CNAs. Annotations could be actual copy number $(0, 1, 2, 3, \ldots)$ or some simplification (normal, deletion, amplification). For the future, we will be interested in developing joint breakpoint detection and copy number calling models that directly use these annotation data as constraints or as part of the model likelihood.

Acknowledgements

References

Ben-Yaacov, E. and Eldar, Y. C. (2008). A Fast and Flexible Method for the Segmentation of aCGH Data. *Bioinformatics*, **24**(16), i139–i145.

Gentleman, R. C., Carey, V. J., Bates, D. M., and others (2004). Bioconductor: Open software development for computational biology and bioinformatics. *Genome Biology*, **5**, R80.

Hoefling, H. (2009). A path algorithm for the Fused Lasso Signal Approximator. arXiv:0910.0526.

Hupé, P., Stransky, N., Thiery, J.-P., Radvanyi, F., and Barillot, E. (2004). Analysis of array CGH data: from signal ratio to gain and loss of DNA regions. *Bioinformatics*, **20**(18), 3413–3422.

Janoueix-Lerosey, I., Schleiermacher, G., Michels, E., Mosseri, V., Ribeiro, A., Lequin, D., Vermeulen, J., Couturier, J., Peuchmaur, M., Valent, A., Plantaz, D., Rubie, H., Valteau-Couanet, D., Thomas, C., Combaret, V., Rousseau, R., Eggert, A., Michon, J., Speleman, F., and Delattre, O. (2009). Overall genomic pattern is a predictor of outcome in neuroblastoma. *Journal of Clinical Oncology*, **27**(7), 1026–1033.

La Rosa, P., Viara, E., Hupé, P., Pierron, G., Liva, S., Neuvial, P., Brito, I., Lair, S., Servant, N., Robine, N., Mani, E., Brennetot, C., Janoueix-Lerosey, I., Raynal, V., Gruel, N., Rouveirol, C., Stransky, N., Stern, M.-H., Delattre, O., Aurias, A., Radvanyi, F., and Barillot, E. (2006). VAMP: Visualization and analysis of array-CGH, transcriptome and other molecular profiles. *Bioinformatics*, **22**(17), 2066–2073.

Lavielle, M. (2005). Using penalized contrasts for the change-point problem. *Signal Processing*, **85**, 1501–1510.

Picard, F., Robin, S., Lavielle, M., Vaisse, C., and Daudin, J.-J. (2005). A statistical approach for array CGH data analysis. *BMC Bioinformatics*, **6**(27).

Picard, F., Lebarbier, E., Hoebeke, M., Rigaill, G., Thiam, B., and Robin, S. (2011). Joint segmentation, calling, and normalization of multiple CGH profiles. *Biostatistics*, **12**(3), 413–428.

R Development Core Team (2011). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0.

Ritz, A., Paris, P., Ittmann, M., Collins, C., and Raphael, B. (2011). Detection of recurrent rearrangement breakpoints from copy number data. *BMC Bioinformatics*, **12**(1), 114.

Russell, B. C., Torralba, A., Murphy, K. P., and Freeman, W. T. (2008). LabelMe: a database and web-based tool for image annotation. *International Journal of Computer Vision*, **77**(1–3), 157–173.

Schleiermacher, G., Janoueix-Lerosey, I., Ribeiro, A., Klijanienko, J., Couturier, J., Pierron, G., Mosseri, V., Valent, A., Auger, N., Plantaz, D., Rubie, H., Valteau-Couanet, D., Bourdeaut, F., Combaret, V., Bergeron, C., Michon, J., and Delattre, O. (2010). Accumulation of segmental alterations determines progression in neuroblastoma. *J Clin Oncol*, **28**(19), 3122–3130.

Shah, S. P., Xuan, X., DeLeeuw, R. J., Khojasteh, M., Lam, W. L., Ng, R., and Murphy, K. P. (2006). Integrating copy number polymorphisms into array CGH analysis using a robust HMM. *Bioinformatics*, **22**(14), 431–439.

Theußl, S. and Zeileis, A. (2009). Collaborative Software Development Using R-Forge. *The R Journal*, **1**(1), 9–14.

Tibshirani, R. and Wang, P. (2007). Spatial smoothing and hot spot detection for CGH data using the fused lasso. *Biostatistics*.

Venkatraman, E. S. and Olshen, A. B. (2007). A faster circular binary segmentation algorithm for the analysis of array CGH data. *Bioinformatics*, **23**(6), 657–663.

Vert, J.-P. and Bleakley, K. (2010). Fast detection of multiple change-points shared by many signals using group LARS. In J. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel, and A. Cullota, editors, *Advances in Neural Information Processing Systems 23 (NIPS)*, pages 2343–2351.

Weinberg, R. A. (2006). *The Biology of Cancer*. Garland Science, first edition.

Willenbrock, H. and Fridlyand, J. (2005). A comparison study: applying segmentation to array CGH data for downstream analysis. *Bioinformatics*, **21**(22), 4084–4091.

Zhang, N. R. and Siegmund, D. O. (2007). A Modified Bayes Information Criterion with Applications to the Analysis of Comparative Genomic Hybridization Data. *Biometrics*, **63**, 22–32.

Zhang, Z., Lange, K., Ophoff, R., and Sabatti, C. (2010). Reconstructing DNA copy number by penalized estimation and imputation. *The Annals of Applied Statistics*, **4**, 1749–1773.