# A Tutorial on Using the R Package FLLat

Gen Nowak, Trevor Hastie, Jonathan R. Pollack, Robert Tibshirani and Nicholas Johnson

November 2, 2024

## 1 Introduction

This document provides a brief tutorial on using the FLLat package, which implements the Fused Lasso Latent Feature (FLLat) model (Nowak and others, 2011). The FLLat model is designed to identify regions of copy number variation (CNV) in multi-sample aCGH data. In particular, it takes advantage of any similarities shared among samples while maintaining the ability to identify any potential heterogeneity. The model is described in more detail below.

## 2 Fused Lasso Latent Feature model

Suppose we have a group of $S$ samples of aCGH data, with each sample consisting of the observed log intensity ratios at $L$ probe locations. The FLLat model assumes that there exists a fixed number of features that can describe these samples and models each sample as a weighted combination of these features. Formally, the model is given by:

$$y_{ls} = \sum_{j=1}^{J} \beta_{lj}\theta_{js} + \epsilon_{ls}, \tag{1}$$

where $y_{ls}$ denotes the observed log intensity ratio at probe location $l$ for sample $s$, $\beta_{lj}$ denotes the value of the $j$th feature at probe location $l$, $\theta_{js}$ denotes the weight applied to the $j$th feature for sample $s$ and $J$ is the number of features. It is a little easier to visualize when (1) is written using matrix notation:

$$\boldsymbol{Y} = \boldsymbol{B\Theta} + \boldsymbol{E}. \tag{2}$$

In (2), each column of $\boldsymbol{Y}$ is a sample, each column of $\boldsymbol{B}$ is a feature and each column of $\boldsymbol{\Theta}$ are the weights applied to the features to generate a sample.

We fit the model by minimizing the following criterion:

$$F(\boldsymbol{B}, \boldsymbol{\Theta}) = \|\boldsymbol{Y} - \boldsymbol{B\Theta}\|_F^2 + \sum_{j=1}^{J} P_{\lambda_1,\lambda_2}(\boldsymbol{\beta}_{\cdot j}), \tag{3}$$

where $P_{\lambda_1,\lambda_2}(\boldsymbol{\beta}_{\cdot j}) = \lambda_1 \sum_{l=1}^{L} |\beta_{lj}| + \lambda_2 \sum_{l=2}^{L} |\beta_{lj} - \beta_{l-1,j}|$. We have applied a fused lasso penalty to each feature, encouraging the features to be both sparse and smooth. This helps us to identify the regions of CNV. Additionally, the $L_2$ norm of each row of $\Theta$ is constrained to be less than or equal to 1.

## 3 Using the FLLat Package

There are four main functions in the FLLat package: FLLat, FLLat.BIC, FLLat.PVE and FLLat.FDR. The FLLat function fits the FLLat model for given values of $J$ (the number of features), $\lambda_1$ and $\lambda_2$ (the fused lasso tuning parameters). The FLLat.BIC function determines the optimal values of $\lambda_1$ and $\lambda_2$ for a given value of $J$. The FLLat.PVE function aids in choosing an appropriate value of $J$. Finally, the FLLat.FDR function estimates the false discovery rate (FDR) for a fitted FLLat model. Unless the user has some pre-specified values of $J$, $\lambda_1$ and $\lambda_2$ which they would like to use, the usual steps would be to first run FLLat.PVE to

choose $J$, then run `FLLat.BIC` to choose $\lambda_1$ and $\lambda_2$. We will demonstrate these steps on a simulated data set, `simaCGH`, that is included in the package. This data set consists of 20 samples and 1000 probes and was generated using model (1) with 5 features. Further simulation details can be found in Nowak and others (2011).

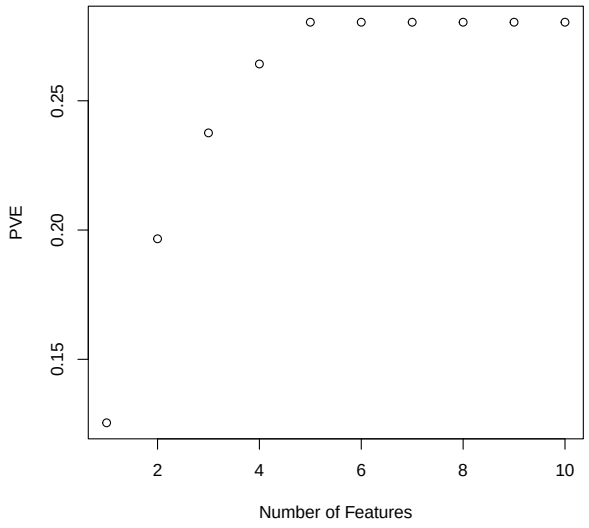
### 3.1 FLLat.PVE

The first step is to use the `FLLat.PVE` function. Given a sequence of values of $J$, this function calculates the percentage of variation explained (PVE) for each $J$, where the PVE is defined as:

$$\text{PVE}_J = 1 - \frac{\sum_{s=1}^{S}\sum_{l=1}^{L}\left(y_{ls} - \sum_{j=1}^{J}\hat{\beta}_{lj}\hat{\theta}_{js}\right)^2}{\sum_{s=1}^{S}\sum_{l=1}^{L}\left(y_{ls} - \bar{y}_s\right)^2}, \tag{4}$$

with $\hat{\beta}_{lj}$ and $\hat{\theta}_{js}$ denoting the estimates produced by the FLLat model and $\bar{y}_s = \left(\sum_{l=1}^{L} y_{ls}\right)/L$. The idea is that after a certain point, additional features will not significantly improve the estimated fit and the PVE will begin to flatten out. We can then choose the point at which the PVE begins to flatten out as the appropriate value of $J$. The code below runs `FLLat.PVE` on the data set `simaCGH` for $J = 1, \ldots, 10$ and also plots the resulting PVEs:

```
> library(FLLat)
> data(simaCGH)
> result.pve <- FLLat.PVE(simaCGH,J.seq=1:(ncol(simaCGH)/2))
> plot(result.pve)
```




### 3.2 FLLat.BIC

We see from the PVE plot that the PVEs begin to flatten out after $J = 5$. Having chosen an appropriate value of $J$, we can run the `FLLat.BIC` function with this value of $J$ to choose the optimal tuning parameters. The `FLLat.BIC` function first re-parameterizes $\lambda_1$ and $\lambda_2$ in terms of $0 < \alpha < 1$ and $\lambda_0$ such that $\lambda_1 = \alpha\lambda_0$ and $\lambda_2 = (1-\alpha)\lambda_0$. It then searches over a two dimensional grid of $\alpha$ and $\lambda_0$ values and chooses the optimal

values by minimizing the following criterion:

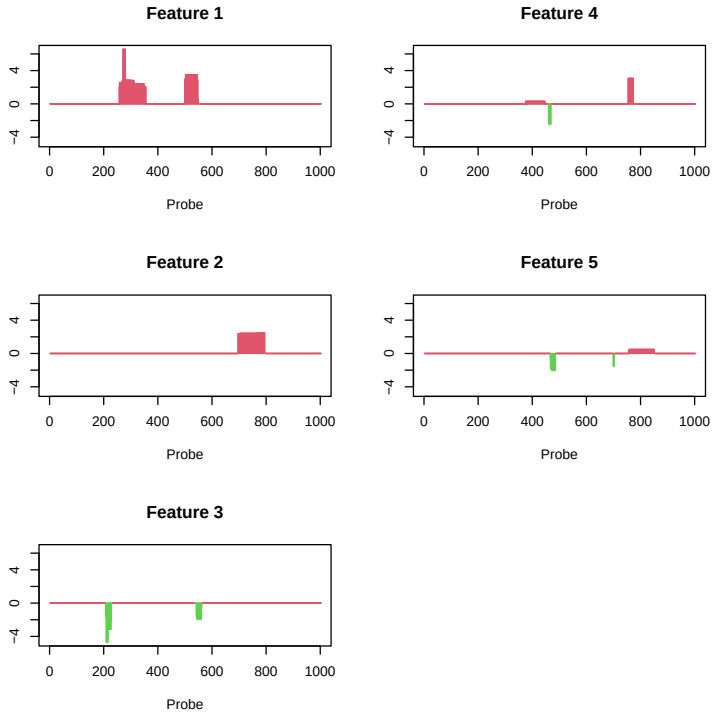
$$(SL) \times \log\left(\frac{\left\|\boldsymbol{Y} - \hat{\boldsymbol{B}}\hat{\boldsymbol{\Theta}}\right\|_F^2}{SL}\right) + k_{\alpha,\lambda_0} \log(SL), \tag{5}$$

where $k_{\alpha,\lambda_0} = \sum_{j=1}^{J} k_{\alpha,\lambda_0}(j)$ and $k_{\alpha,\lambda_0}(j)$ is the number of unique non-zero elements in the $j$th feature. This criterion attempts to strike a balance between over-fit and under-fit models. The following code runs the `FLLat.BIC` function for $J = 5$:

```
> result.bic <- FLLat.BIC(simaCGH,J=5)
> result.bic$lam1

[1] 1.027413

> result.bic$lam2

[1] 9.246716
```

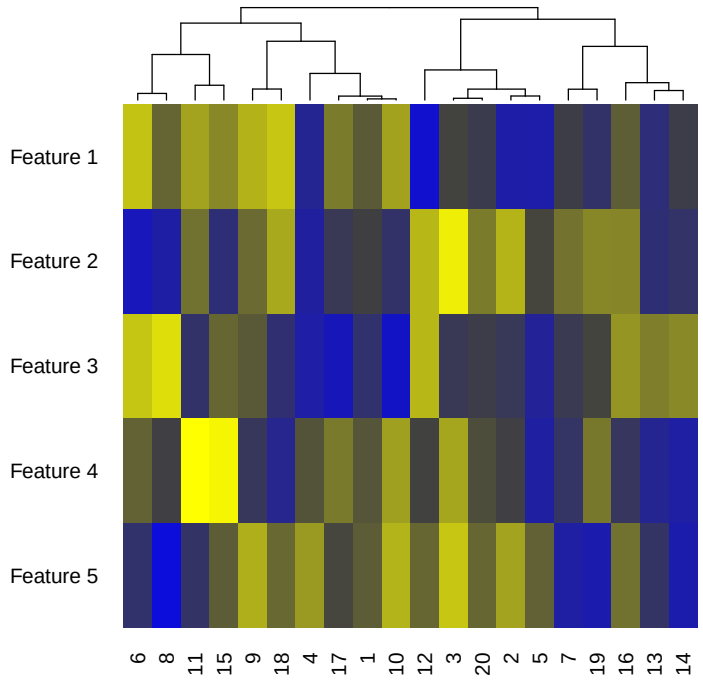
The `FLLat.BIC` function also returns the fitted FLLat model (`opt.FLLat`) for the optimal values of the tuning parameters. We can plot the estimated features, which are plotted in order of decreasing magnitude (where the magnitude of a feature is given by $\sum_{l=1}^{L} \hat{\beta}_{lj}^2$):

```
> plot(result.bic$opt.FLLat)
```




These features indicate the main regions of CNV that are displayed by this group of 20 samples. We can also plot a heatmap of the estimated weights:

```
> plot(result.bic$opt.FLLat,type="weights")
```




The yellow and blue denote positive and negative weights, respectively, and indicate how prominently each feature appears in each sample. Also, the clustering of the samples (based on their weights) seems to show the existence of two main groups of samples.

### 3.3 `FLLat.FDR`

The `FLLat.FDR` function can now be used to estimate the FDR for this fitted FLLat model. Before we give an example of using the `FLLat.FDR` function, we will first briefly describe how the FDR is defined and estimated for the FLLat model. To identify regions of CNV (i.e., significant aberrations), we are testing the following hypothesis for each probe location within each sample:

$$H_0(l,s) = \text{no aberration at probe location } l \text{ for sample } s.$$

We can use a fixed threshold and the fitted values from the FLLat model, $\hat{\boldsymbol{Y}} = \hat{\boldsymbol{B}}\hat{\boldsymbol{\Theta}}$, to help us identify significant aberrations in the following manner. Given a threshold $T$, we declare probe location $l$ for sample $s$ to be a significant aberration if $|\hat{y}_{ls}| \geq T$. In other words, every location with an absolute fitted value at least as large as $T$ is declared a significant aberration. Based on these declared significant aberrations, the FDR is defined as:

$$\text{FDR}(T) = \mathbf{E}\left(\frac{\sum_{s=1}^{S}\sum_{l=1}^{L} I\left(|\hat{y}_{ls}| \geq T\right) I\left(H_0(l,s) \text{ is true}\right)}{\sum_{s=1}^{S}\sum_{l=1}^{L} I\left(|\hat{y}_{ls}| \geq T\right)}\right) \tag{6}$$

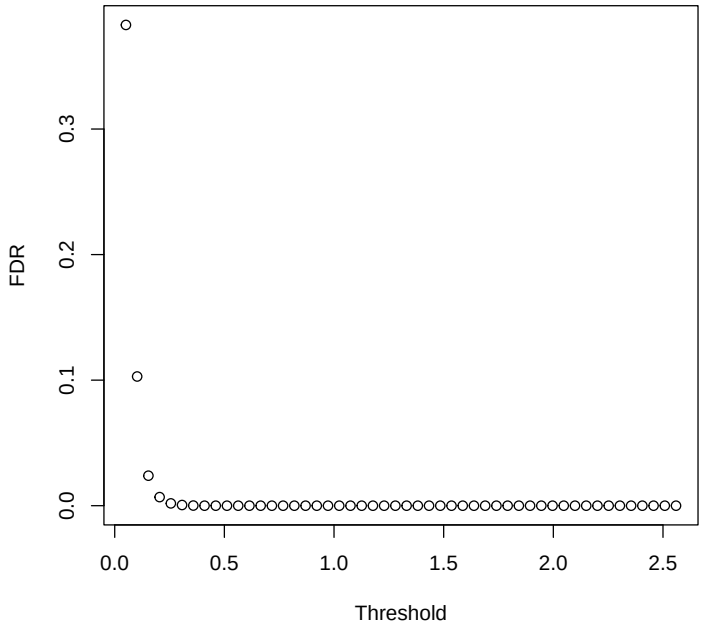
That is, the FDR is the expected proportion of declared aberrations that are not true aberrations. In order to estimate the FDR, in particular the numerator in (6), we need to approximate the null distribution of the data. This is done by repeatedly applying the FLLat model to permuted versions of the data $Y$, where each permuted data set is obtained by randomly permuting the probe locations within every sample. We can then estimate the FDR by

$$\widehat{\text{FDR}}(T) = \frac{\pi_0 \sum_{k=1}^{K}\left(\sum_{s=1}^{S}\sum_{l=1}^{L} I\left(|\hat{y}^k_{ls}| \geq T\right)\right) \Big/ K}{\sum_{s=1}^{S}\sum_{l=1}^{L} I\left(|\hat{y}_{ls}| \geq T\right)}, \tag{7}$$

where $\hat{y}_{ls}^k$ is the fitted value obtained from applying the FLLat model to the $k$th permuted data set, $K$ is the number of permuted data sets and $\pi_0$ is the proportion of true null hypotheses. The value of $\pi_0$ can be set to 1 to give an upper bound for the FDR, or it can be set to something less than 1 if some prior knowledge is available and suggests so.

The following code estimates the FDR for our fitted FLLat model over a range of 50 threshold values, determines the threshold value which controls the FDR at 0.05 and plots the FDRs against the threshold values.

```
> set.seed(1364)
> result.fdr <- FLLat.FDR(simaCGH,result.bic$opt.FLLat)
> result.fdr$thresh.control

[1] 0.1535668

> plot(result.fdr)
```




### 3.4 `predict.FLLat`

There is also a `predict` method for fitted FLLat models (objects of class `FLLat`). Given a set of new data and a fitted FLLat model, the `predict.FLLat` function returns the predicted values for the new data and the predicted weights for producing these predicted values. Specifically, the `predict.FLLat` function takes the estimated features from the fitted FLLat model, $\hat{\boldsymbol{B}}$, and calculates the new weights for predicting the new data $\boldsymbol{Y}^*$ by minimizing the residual sum of squares:

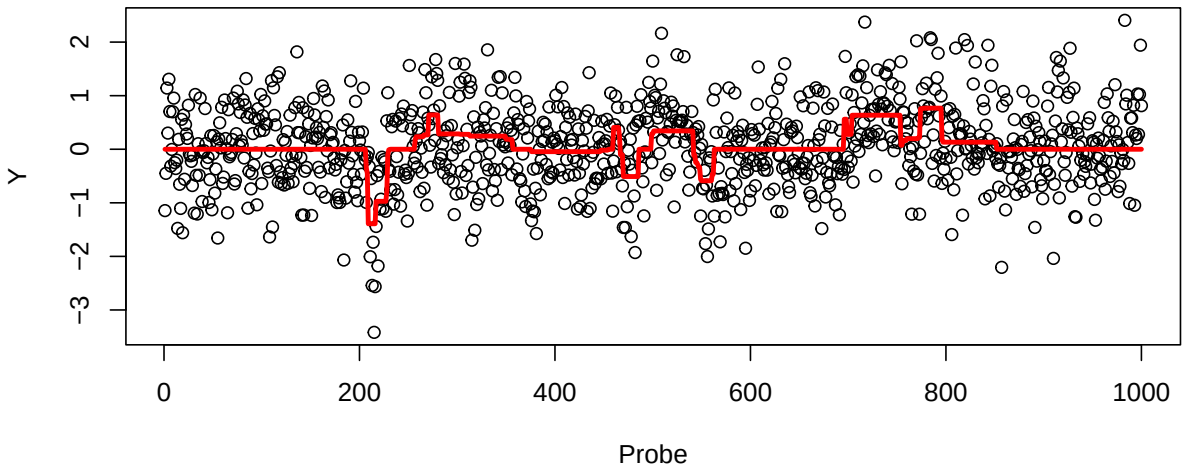
$$\hat{\boldsymbol{\Theta}}^* = \underset{\boldsymbol{\Theta}:\|\boldsymbol{\theta}_{j\cdot}\|^2 \leq 1}{\arg\min} \left\| \boldsymbol{Y}^* - \hat{\boldsymbol{B}}\boldsymbol{\Theta} \right\|_F^2,$$

where the minimization is taken over all $\boldsymbol{\Theta}$ for which the $L_2$ norm of every row is not greater than 1. Note that predicted values and weights can only be calculated if the number of probes in the new data is the same as that in the original data used to produce the fitted FLLat model. Also, for the predictions to be meaningful, the new data should be similar in scale to the original data. If the `predict.FLLat` function is

not given any new data as an argument, it simply returns the fitted values and the estimated weights from the fitted FLLat model.

The following code divides the simulated data set into a training set of 15 samples and a test set of 5 samples, fits a FLLat model to the training set (with pre-specified values of $J$, $\lambda_1$ and $\lambda_2$) and, using this fitted FLLat model, calculates the predicted values and weights on the test set. The code also plots the predicted values superimposed over the data for one of the samples in the test set.

```
> tr.dat <- simaCGH[,1:15]
> tst.dat <- simaCGH[,16:20]
> result.tr <- FLLat(tr.dat,J=5,lam1=1,lam2=9)
> tst.pred <- predict(result.tr,newY=tst.dat)
> plot(tst.dat[,1],xlab="Probe",ylab="Y")
> lines(tst.pred$pred.Y[,1],col="red",lwd=3)
```




## References

G. Nowak, T. Hastie, J. R. Pollack and R. Tibshirani. A Fused Lasso Latent Feature Model for Analyzing Multi-Sample aCGH Data. *Biostatistics*, 2011, doi: 10.1093/biostatistics/kxr012