Fused lasso

Patrick Breheny

May 6, 2025

Introduction

- Today, we will discuss a different kind of sparsity arising from structure among the features: rather than being grouped, we will consider the case in which features are ordered
- Ordered situations arise in many situations, such as spectroscopic data, temporal data, and spatial data; we will discuss its application to genetics and copy number variation later
- It can also be applied in situations where the features are not naturally ordered, but could be ordered using, say, hierarchical clustering (as could the group lasso)

Fused lasso

- The fused lasso estimates $\widehat{\beta}$ are the values minimizing the following objective function:

$$Q(\boldsymbol{\beta}|\mathbf{X},\mathbf{y}) = \tfrac{1}{2n}\|\mathbf{y}-\mathbf{X}\boldsymbol{\beta}\|_2^2 + \lambda_1\|\boldsymbol{\beta}\|_1 + \lambda_2\sum_{j=1}^{p-1}|\beta_j-\beta_{j+1}|$$

- Note that the penalty consists of two pieces:
    - A lasso penalty that encourages $\beta_j = 0$
    - A fusion penalty that encourages β_j to be equal to β_{j+1} and β_{j-1}

Fused lasso signal approximator

- A special case of the fused lasso that we will concentrate on today is the situation where $\mathbf{X} = \mathbf{I}$
- To make it clear which case we are dealing with, I will use $\hat{\boldsymbol{\theta}}$ to denote the solutions to this problem of minimizing

$$Q(\boldsymbol{\theta}|\mathbf{y}) = \frac{1}{2}\|\mathbf{y} - \boldsymbol{\theta}\|_2^2 + \lambda_1\|\boldsymbol{\theta}\|_1 + \lambda_2 \sum_{j=1}^{n-1} |\theta_j - \theta_{j+1}|$$

- This version of the problem is sometimes called the "fused lasso signal approximator' ', in the sense that it amounts to approximating a one-dimensional signal with a series of zeroes and piecewise constant functions

Coordinate descent: Unsuitable?

- Solving this optimization problem, however, introduces some new challenges that we have not yet encountered
- Recall the two basic conditions necessary for coordinate descent algorithms to converge
  - A differentiable loss function (this was violated in LAD/quantile regression)
  - A separable penalty function (this is violated in the fused lasso)
- As we will see, coordinate descent does not work well at all for solving the fused lasso problem; new tools are needed

Toy data

- To get a better sense of what's going on, let's consider a toy data set: $\mathbf{y} = \{0, 0, 0, 1, 1, 1, 0, 0, 0\}$
- For the purposes of illustration, let $\lambda_1 = 0$ and $\lambda_2 = 1/2$
- We can see that $Q(\mathbf{y}) = 1$, while $Q(\mathbf{0}) = 1.5$, so $Q(\mathbf{y}) < Q(\mathbf{0})$
- Nevertheless, if we start at the initial value $\boldsymbol{\theta} = \mathbf{0}$, the coordinate descent algorithm can never escape zero
- By only considering one-coordinate-at-a-time transitions, the CD algorithm misses the fact that we could simultaneously move $\{\theta_4, \theta_5, \theta_6\}$ and obtain a better solution

ADMM: Introduction

- There are a variety of alternative algorithms we could use here, but this is a good opportunity to discuss a flexible and useful algorithm called the *alternating direction method of multipliers*, or ADMM, algorithm
- ADMM algorithms converge for a wider range of problems than CD; in addition (although we won't focus on this today), they lend themselves to parallelization in a way that CD algorithms do not, which has led to a considerable amount of recent interest in them
- The essence of the ADMM algorithm is that we will introduce new variables $\{\delta_j = \theta_j - \theta_{j+1}\}_{j=1}^{n-1}$ and alternate between updating $\boldsymbol{\theta}$, updating $\boldsymbol{\delta}$, and reconciling their differences

Reframing the problem ($\lambda_1 = 0$ for simplicity)

Specifically, let us reframe the problem as: minimize

$$\frac{1}{2}\|\mathbf{y} - \boldsymbol{\theta}\|_2^2 + \lambda\|\boldsymbol{\delta}\|_1$$

subject to the constraint

$$\mathbf{D}\boldsymbol{\theta} = \boldsymbol{\delta},$$

where $\mathbf{D}$ is the $(n-1) \times n$ matrix of first-order differences:

$$\mathbf{D} = \begin{bmatrix} 1 & -1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & -1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & -1 \end{bmatrix}$$

The augmented Lagrangian

- In general, Lagrange multipliers are a useful way of solving optimization problems with constraints
- The ADMM algorithm uses a modification of this approach in order to achieve greater robustness; we will minimize the *augmented Lagrangian*

$$\frac{1}{2}\|\mathbf{y}-\boldsymbol{\theta}\|_2^2+\lambda\|\boldsymbol{\delta}\|_1+\frac{\rho}{2}\|\mathbf{D}\boldsymbol{\theta}-\boldsymbol{\delta}+\mathbf{u}\|_2^2-\frac{\rho}{2}\|\mathbf{u}\|_2^2,$$

  where $\mathbf{u}$ are the (scaled) Lagrange multipliers (also known as dual variables)
  - The algorithm thus consists of alternately updating $\boldsymbol{\theta}$, $\boldsymbol{\delta}$, and $\mathbf{u}$, all of which have simple, closed forms

ADMM updates

- **Proposition:** Given $\boldsymbol{\delta}$ and $\mathbf{u}$ from iteration k, the value of $\boldsymbol{\theta}$ that minimizes the augmented Lagrangian for iteration $k+1$ is

$$\boldsymbol{\theta} = (\rho \mathbf{D}^\top \mathbf{D} + \mathbf{I})^{-1}[\mathbf{y} + \rho \mathbf{D}^\top (\boldsymbol{\delta} - \mathbf{u})]$$

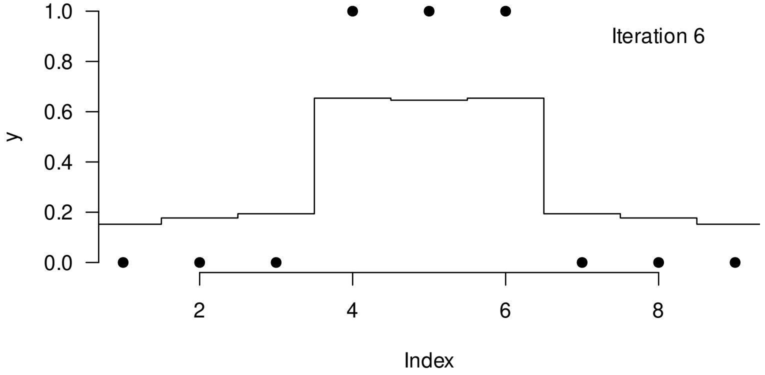
- **Proposition:** Given $\boldsymbol{\theta}$ from iteration $k+1$ and $\mathbf{u}$ from iteration k, the value of $\boldsymbol{\delta}$ that minimizes the augmented Lagrangian for iteration $k+1$ is

$$\boldsymbol{\delta} = \frac{1}{\rho} S\left(\rho(\mathbf{D}\boldsymbol{\theta} + \mathbf{u}), \lambda\right)$$

- To update $\mathbf{u}$, on the other hand, we apply an update with step size ρ:

$$\mathbf{u}^{k+1} = \mathbf{u}^k + \rho(\mathbf{D}\boldsymbol{\theta}^{k+1} - \boldsymbol{\delta}^{k+1})$$

ADMM convergence for the toy data



Remarks

- Recall that $Q(\mathbf{0}) = 1.5$ and $Q(\mathbf{y}) = 1$; we now have $Q(\hat{\boldsymbol{\theta}}) = 0.75$
- In other words, the decoupling between $\boldsymbol{\theta}$ and $\boldsymbol{\delta}$ introduced by the ADMM prevented the algorithm from being stuck at $\mathbf{0}$ and allowed us to reach the global minimum
- The step size ρ affects convergence:
  - ρ too small and $\boldsymbol{\theta}$, $\boldsymbol{\delta}$ remain uncoupled
  - ρ too large and $\boldsymbol{\theta}$, $\boldsymbol{\delta}$ too coupled; don't have the flexibility to reach optimal solution

Path algorithms

- ADMM is a very flexible framework worth knowing about
- In the specific context of the FLSA, however, there are also a variety of exact solutions that can be calculated using an algorithm somewhat analogous to the LARS algorithm for the regular lasso
- The fast solver provided by the R package `flsa` (which we will use in the case study coming up) uses one of these algorithms, not ADMM
- These exact algorithms tend to be quite a bit faster for small problems; for larger problems, and for going outside the FLSA framework, ADMM is often better

Copy number variation

- Broadly speaking, humans have two copies of their genome
- Occasionally however, a region of the genome is duplicated or destroyed; this is known as copy number variation (CNV) and it occurs in all humans
- Copy number variation tends to be more extreme in cancer, however: gains or losses of large regions of the genome often trigger uncontrolled cell growth
- There are a variety of methods for measuring copy number variation in a genome-wide fashion; the data we will look at today comes from a method known as comparative genomic hybridization (CGH)

`glioma` data

- The data we will look at today is a popular benchmark in the field
- It consists of CGH data from two glioblastoma tumors (chromosome 7 in one patient, chromosome 13 in another) spliced together in order to create a challenging data set for CNV detection:
  - Both gains and losses are present
  - The copy number changes occur over both short and large scales
- CGH data is typically reported on the $\log_2$ ratio scale, so that 0 means 2 copies (i.e., a normal number of copies), $\log_2(3/2) = 1$ means a gain of a copy, and $\log_2(1/2) = -1$ means the loss of a copy

The **flsa** package

- There is a nice package called **flsa** for fitting the special case of the fused lasso signal approximator
- Its basic usage is

```
flsa(y, lambda1 = 0, lambda2 = 1/2)
```

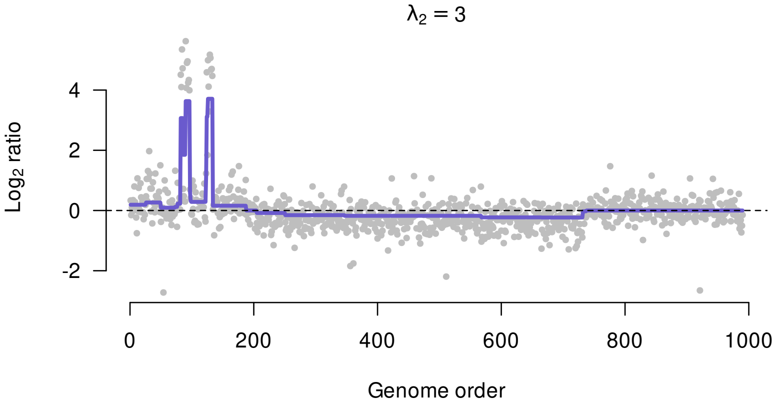
- Often, however, it is best to fit the whole path with

```
fit <- flsa(y)
```

followed by

```
flsaGetSolution(fit, lambda1 = 0.1, lambda2 = 1/2)
```

Fused lasso solution



Two-dimensional fused lasso

- The fused lasso, as we have presented it, accounts for one-dimensional ordering
- Of course, two-dimensional ordering is also common: spatial statistics, images
- Consider, then, the two-dimensional fused lasso (which we present here in signal approximator form):

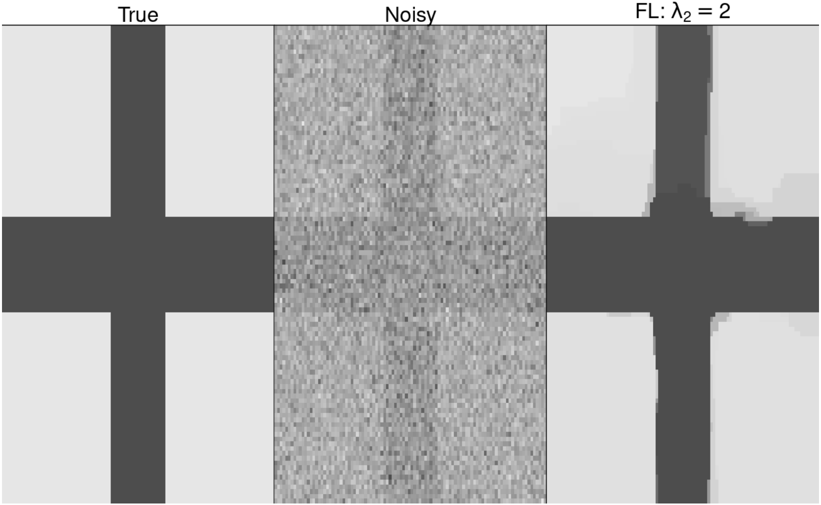
$$Q(\boldsymbol{\beta}) = \frac{1}{2}\|\mathbf{Y} - \boldsymbol{\Theta}\|_F^2 + \lambda \sum_{i,j} \left(|\theta_{i,j} - \theta_{i+1,j}| + |\theta_{i,j} - \theta_{i,j+1}|\right),$$

where $\|\mathbf{A}\|_F$ is the *Frobenius norm*: $\|\mathbf{A}\|_F = \sqrt{\sum_{i,j} a_{ij}^2}$

Image de-noising

- A major application of the two-dimensional fused lasso is in image processing
- The idea here is that there exists a “true” image, but we only see a noisy image, from which we would like to recover the true image
- In this context, the two-dimensional fused lasso is known as *total variation de-noising*; this idea predates the fused lasso, although recent advances in convex optimization have led to better algorithms

Fused lasso solution



Piecewise constant trends

- In the context of nonparametric regression, the ℓ_1 penalty on the coefficients themselves typically does not make sense
- Minimizing the objective function

$$Q(\boldsymbol{\theta}|\mathbf{y}) = \frac{1}{2}\|\mathbf{y} - \boldsymbol{\theta}\|_2^2 + \lambda \sum_{j=1}^{n-1} |\theta_j - \theta_{j+1}|$$

  encourages the fit, or trend, to be a piecewise constant function
- This has applications in changepoint detection, although often one prefers a continuous solution rather than a piecewise constant

Trend filtering

- Consider, then, the following variation:

$$Q(\boldsymbol{\theta}|\mathbf{y}) = \frac{1}{2}\|\mathbf{y} - \boldsymbol{\theta}\|_2^2 + \lambda \sum_{j=2}^{n-1} |\theta_{j-1} - 2\theta_j + \theta_{j+1}|$$

- Note that penalty is zero when (and only when) the estimates θ_{j-1}, θ_j, and θ_{j+1} are on a line: this penalty encourages piecewise linear solutions
- The same logic can be extended to piecewise polynomials of any order; the idea is known in general as *trend filtering*

Encouraging monotone solutions

- One final variation on this idea: suppose we replace the absolute value in the penalty with the positive part $(\cdot)_+$:

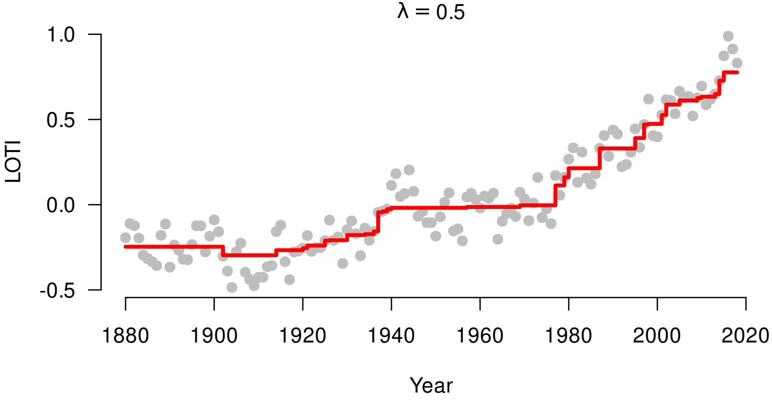
$$Q(\beta|\mathbf{X}, \mathbf{y}) = \frac{1}{2}\|\mathbf{y} - \boldsymbol{\theta}\|_2^2 + \lambda \sum_{j=1}^{n-1} (\theta_j - \theta_{j+1})_+$$

- In other words, increasing values of θ are not penalized at all, but decreasing values are penalized as in the fused lasso
- Such a method might be useful in fitting a line to data in situations where we expect a monotone relationship

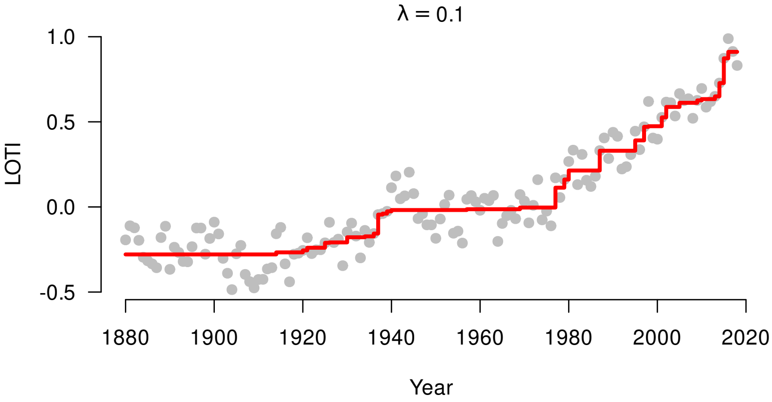
Isotonic regression

- This problem (fitting a monotone line to data) has a long history in statistics dating back to the 1950s, and is known as *isotonic regression*
- The modification of the fused lasso introduced on the previous slide is one way to solve this problem: by setting λ large enough, we can force the solution to be monotone
- However, by merely encouraging monotonicity rather than requiring it, we can also accomplish something new; this idea is known as *nearly isotonic regression*

Global warming: Fused lasso



Global warming: Nearly isotonic regression



Global warming: Trend filtering