A Case for using Trend Filtering over Splines

Aaditya Ramdas

ML Department and Statistics Department
Carnegie Mellon University

Joint work (+ borrowing slides) with Ryan Tibshirani

Nonparametric regression

Nonparametric regression: observe $(x_1, y_1), \ldots (x_n, y_n) \in \mathbb{R}^p \times \mathbb{R}$ from model

$$y_i = f(x_i) + \epsilon_i, \quad i = 1, \ldots n$$

Nonparametric regression

Nonparametric regression: observe $(x_1, y_1), \ldots (x_n, y_n) \in \mathbb{R}^p \times \mathbb{R}$ from model

$$y_i = f(x_i) + \epsilon_i, \quad i = 1, \ldots n$$

Errors ϵ_i have zero mean conditional on $X = x_i$. Want to estimate

$$f(x) = \mathbb{E}[Y | X = x]$$

Nonparametric regression

Nonparametric regression: observe $(x_1, y_1), \ldots (x_n, y_n) \in \mathbb{R}^p \times \mathbb{R}$ from model

$$y_i = f(x_i) + \epsilon_i, \quad i = 1, \ldots n$$

Errors ϵ_i have zero mean conditional on $X = x_i$. Want to estimate

$$f(x) = \mathbb{E}[Y | X = x]$$

Rich literature, lots of interesting work (mostly for $p = 1$).

Nonparametric regression

Nonparametric regression: observe $(x_1, y_1), \ldots (x_n, y_n) \in \mathbb{R}^p \times \mathbb{R}$ from model

$$y_i = f(x_i) + \epsilon_i, \quad i = 1, \ldots n$$

Errors ϵ_i have zero mean conditional on $X = x_i$. Want to estimate

$$f(x) = \mathbb{E}[Y|X = x]$$

Rich literature, lots of interesting work (mostly for $p = 1$). E.g.,

- Local polynomials
- Kernels
- Splines
- Wavelets

Nonparametric regression

Nonparametric regression: observe $(x_1, y_1), \ldots (x_n, y_n) \in \mathbb{R}^p \times \mathbb{R}$ from model

$$y_i = f(x_i) + \epsilon_i, \quad i = 1, \ldots n$$

Errors ϵ_i have zero mean conditional on $X = x_i$. Want to estimate

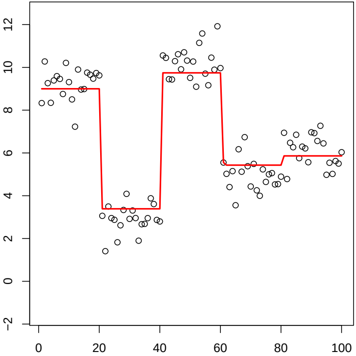
$$f(x) = \mathbb{E}[Y|X = x]$$

Rich literature, lots of interesting work (mostly for $p = 1$). E.g.,

- Local polynomials
- Kernels
- Splines
- Wavelets

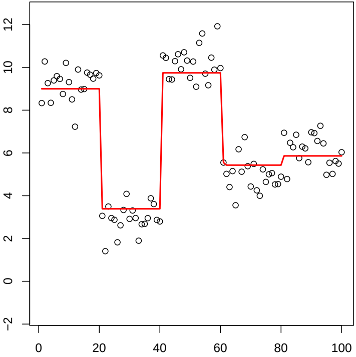
This talk: relative newcomer in nonparametric regression. Assume $p = 1$ and $x_1, \ldots x_n$ are evenly spaced (for now)

Constant-order trend filtering



Setup: we make observations $y = (y_1, \ldots y_n) \in \mathbb{R}^n$ at successive, equally spaced locations

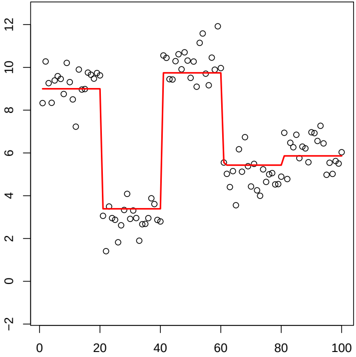
Constant-order trend filtering



Setup: we make observations $y = (y_1, \ldots y_n) \in \mathbb{R}^n$ at successive, equally spaced locations

We want to approximate y by piecewise constant sequence, in red

Constant-order trend filtering



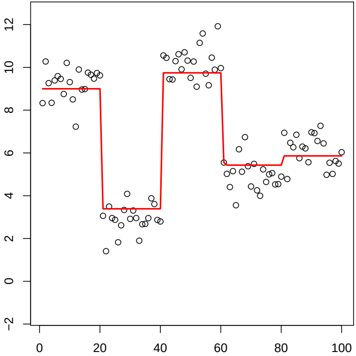
Setup: we make observations $y = (y_1, \dots y_n) \in \mathbb{R}^n$ at successive, equally spaced locations

We want to approximate y by piecewise constant sequence, in red

Given by solving **1-dimensional fused lasso** problem

$$\min_{\beta \in \mathbb{R}^n} \ \frac{1}{2} \sum_{i=1}^{n} (y_i - \beta_i)^2 + \lambda \sum_{i=1}^{n-1} |\beta_i - \beta_{i+1}|$$

Constant-order trend filtering



Setup: we make observations $y = (y_1, \ldots y_n) \in \mathbb{R}^n$ at successive, equally spaced locations

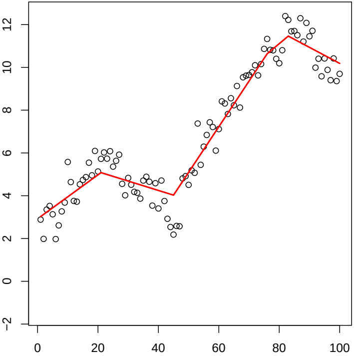
We want to approximate y by piecewise constant sequence, in red

Given by solving **1-dimensional fused lasso** problem

$$\min_{\beta \in \mathbb{R}^n} \frac{1}{2} \sum_{i=1}^{n} (y_i - \beta_i)^2 + \lambda \sum_{i=1}^{n-1} |\beta_i - \beta_{i+1}|$$

(Also called 1-dimensional total variation denoising)

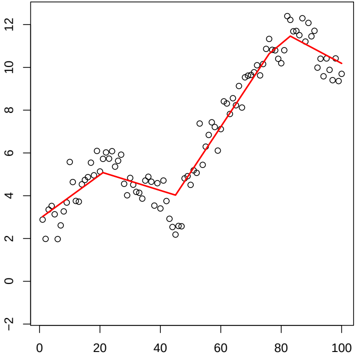
Linear trend filtering



Same setup, but now we believe underlying trend is piecewise linear

(Or well-approximated by such a function)

Linear trend filtering



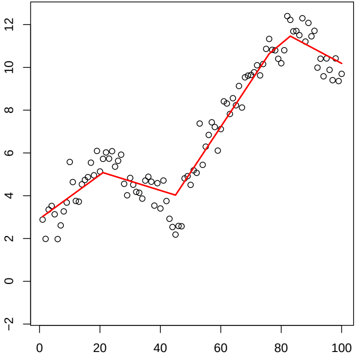
Same setup, but now we believe underlying trend is piecewise linear

(Or well-approximated by such a function)

Solve **linear trend filtering** problem

$$\min_{\beta \in \mathbb{R}^n} \ \frac{1}{2} \sum_{i=1}^{n} (y_i - \beta_i)^2 + \lambda \sum_{i=1}^{n-2} |\beta_i - 2\beta_{i+1} + \beta_{i+2}|$$

Linear trend filtering



Same setup, but now we believe underlying trend is piecewise linear

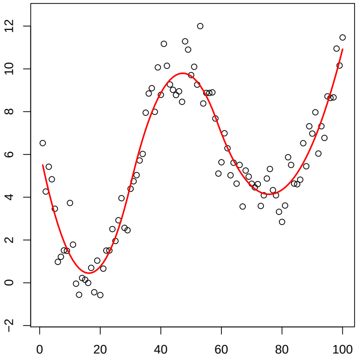
(Or well-approximated by such a function)

Solve **linear trend filtering** problem

$$\min_{\beta \in \mathbb{R}^n} \frac{1}{2} \sum_{i=1}^{n} (y_i - \beta_i)^2 + \lambda \sum_{i=1}^{n-2} |\beta_i - 2\beta_{i+1} + \beta_{i+2}|$$

Note $\beta_i - 2\beta_{i+1} + \beta_{i+2} = 0 \iff \beta_{i+1} = (\beta_i + \beta_{i+2})/2$

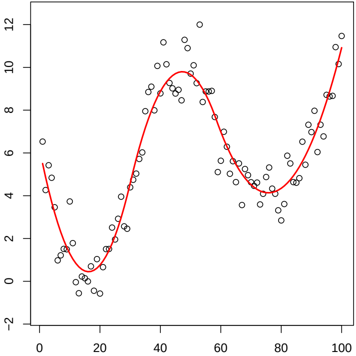
Quadratic trend filtering



Same setup, but now we believe underlying trend is piecewise quadratic

(Or well-approximated by such a function)

Quadratic trend filtering



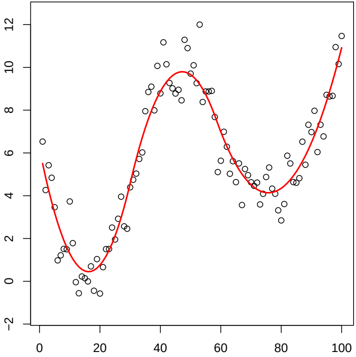
Same setup, but now we believe underlying trend is piecewise quadratic

(Or well-approximated by such a function)

Solve **quadratic trend filtering** problem

$$\min_{\beta \in \mathbb{R}^n} \frac{1}{2} \sum_{i=1}^{n} (y_i - \beta_i)^2 + \lambda \sum_{i=1}^{n-3} |\beta_i - 3\beta_{i+1} + 3\beta_{i+2} - \beta_{i+3}|$$

Quadratic trend filtering



Same setup, but now we believe underlying trend is piecewise quadratic

(Or well-approximated by such a function)

Solve **quadratic trend filtering** problem

$$\min_{\beta \in \mathbb{R}^n} \ \frac{1}{2} \sum_{i=1}^{n} (y_i - \beta_i)^2 + \lambda \sum_{i=1}^{n-3} |\beta_i - 3\beta_{i+1} + 3\beta_{i+2} - \beta_{i+3}|$$

(Where did this come from?)

Why those penalty terms?

Write 1d fused lasso problem as

$$\min_{\beta \in \mathbb{R}^n} \frac{1}{2}\|y-\beta\|_2^2 + \lambda\|D_1\beta\|_1$$

$$\text{where} \quad D_1 = \begin{bmatrix} -1 & 1 & 0 & \dots & 0 & 0 \\ 0 & -1 & 1 & \dots & 0 & 0 \\ & & & \dots & & \\ 0 & 0 & 0 & \dots & -1 & 1 \end{bmatrix} \in \mathbb{R}^{(n-1)\times n}$$

Why those penalty terms?

Write 1d fused lasso problem as

$$\min_{\beta \in \mathbb{R}^n} \frac{1}{2}\|y - \beta\|_2^2 + \lambda \|D_1 \beta\|_1$$

$$\text{where} \quad D_1 = \begin{bmatrix} -1 & 1 & 0 & \dots & 0 & 0 \\ 0 & -1 & 1 & \dots & 0 & 0 \\ & & & \dots & & \\ 0 & 0 & 0 & \dots & -1 & 1 \end{bmatrix} \in \mathbb{R}^{(n-1)\times n}$$

Linear trend filtering replaces penalty by $\|D_2\beta\|_1$, where

$$D_2 = \begin{bmatrix} -1 & 2 & -1 & \dots & 0 & 0 & 0 \\ 0 & -1 & 2 & \dots & 0 & 0 & 0 \\ \dots & & & & & & \\ 0 & 0 & 0 & \dots & -1 & 2 & -1 \end{bmatrix} \in \mathbb{R}^{(n-2)\times(n-1)}$$

Important relationship: note

$$D_2 \quad = \quad \underbrace{D_1^{(n-1)}}_{(n-2)\times(n-1)} \quad \cdot \quad \underbrace{D_1}_{(n-1)\times n}$$

Important relationship: note

$$D_2 \quad = \quad \underbrace{D_1^{(n-1)}}_{(n-2)\times(n-1)} \quad \cdot \quad \underbrace{D_1}_{(n-1)\times n}$$

Using this recursion: for polynomial trend filtering of order k, the penalty term is $\|D_{k+1}\beta\|_1$, where

$$D_{k+1} \quad = \quad \underbrace{D_1^{(n-k)}}_{(n-k-1)\times(n-k)} \quad \cdot \quad \underbrace{D_k}_{(n-k)\times n} \quad \in \quad \mathbb{R}^{(n-k-1)\times n}$$

Important relationship: note

$$D_2 \quad = \quad \underbrace{D_1^{(n-1)}}_{(n-2)\times(n-1)} \quad \cdot \quad \underbrace{D_1}_{(n-1)\times n}$$

Using this recursion: for polynomial trend filtering of order k, the penalty term is $\|D_{k+1}\beta\|_1$, where

$$D_{k+1} \quad = \quad \underbrace{D_1^{(n-k)}}_{(n-k-1)\times(n-k)} \quad \cdot \quad \underbrace{D_k}_{(n-k)\times n} \quad \in \quad \mathbb{R}^{(n-k-1)\times n}$$

This is **discrete derivative operator** of order $k+1$, i.e., kth order trend filtering penalizes discrete $(k+1)$st derivatives

Uneven spacing

This recursion also reveals a way to deal with uneven spacing: if $y_1, \dots y_n$ are observed at $x_1 < \dots < x_n$, then we redefine

$$D_1 = \begin{bmatrix} -\frac{1}{x_2 - x_1} & \frac{1}{x_2 - x_1} & 0 & \dots & 0 & 0 \\ 0 & -\frac{1}{x_3 - x_2} & \frac{1}{x_3 - x_2} & \dots & 0 & 0 \\ & & & \dots & & \\ 0 & 0 & 0 & \dots & -\frac{1}{x_n - x_{n-1}} & \frac{1}{x_n - x_{n-1}} \end{bmatrix}$$

Uneven spacing

This recursion also reveals a way to deal with uneven spacing: if $y_1, \ldots y_n$ are observed at $x_1 < \ldots < x_n$, then we redefine

$$D_1 = \begin{bmatrix} -\frac{1}{x_2-x_1} & \frac{1}{x_2-x_1} & 0 & \ldots & 0 & 0 \\ 0 & -\frac{1}{x_3-x_2} & \frac{1}{x_3-x_2} & \ldots & 0 & 0 \\ & & & \ldots & & \\ 0 & 0 & 0 & \ldots & -\frac{1}{x_n-x_{n-1}} & \frac{1}{x_n-x_{n-1}} \end{bmatrix}$$

and carry forward recursion as before,

$$D_{k+1} \;=\; \underbrace{D_1^{(n-k)}}_{(n-k-1)\times(n-k)} \cdot \underbrace{D_k}_{(n-k)\times n} \;\in\; \mathbb{R}^{(n-k-1)\times n}, \quad k = 1, 2, \ldots$$

Uneven spacing

This recursion also reveals a way to deal with uneven spacing: if $y_1, \ldots y_n$ are observed at $x_1 < \ldots < x_n$, then we redefine

$$D_1 = \begin{bmatrix} -\frac{1}{x_2-x_1} & \frac{1}{x_2-x_1} & 0 & \ldots & 0 & 0 \\ 0 & -\frac{1}{x_3-x_2} & \frac{1}{x_3-x_2} & \ldots & 0 & 0 \\ & & & \ldots & & \\ 0 & 0 & 0 & \ldots & -\frac{1}{x_n-x_{n-1}} & \frac{1}{x_n-x_{n-1}} \end{bmatrix}$$

and carry forward recursion as before,

$$D_{k+1} \;=\; \underbrace{D_1^{(n-k)}}_{(n-k-1)\times(n-k)} \;\cdot\; \underbrace{D_k}_{(n-k)\times n} \;\in\; \mathbb{R}^{(n-k-1)\times n}, \quad k = 1, 2, \ldots$$

For the rest of this talk, assume even spacing for simplicity; results can be extended to uneven case

Outline

- Theory
- Algorithms
- Neuroscience example
- Extensions

What do we know about trend filtering?

Not a whole lot so far!

What do we know about trend filtering?

Not a whole lot so far!

- Idea and name attributed to Kim et al. (2009), but essentially same idea appears earlier in Mammen and van de Geer (1997)

What do we know about trend filtering?

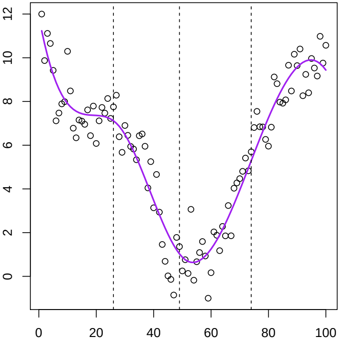
Not a whole lot so far!

- Idea and name attributed to Kim et al. (2009), but essentially same idea appears earlier in Mammen and van de Geer (1997)
- Key property: trend filtering estimates can be viewed as piecewise polynomials, where knots are chosen adaptively

What do we know about trend filtering?

Not a whole lot so far!

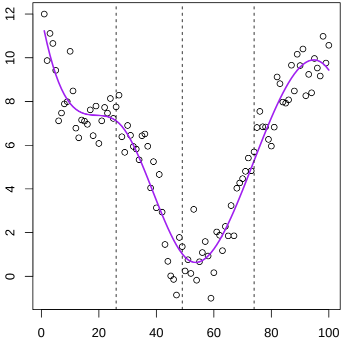
- Idea and name attributed to Kim et al. (2009), but essentially same idea appears earlier in Mammen and van de Geer (1997)
- Key property: trend filtering estimates can be viewed as piecewise polynomials, where knots are chosen adaptively



What do we know about trend filtering?

Not a whole lot so far!

- Idea and name attributed to Kim et al. (2009), but essentially same idea appears earlier in Mammen and van de Geer (1997)
- Key property: trend filtering estimates can be viewed as piecewise polynomials, where knots are chosen adaptively

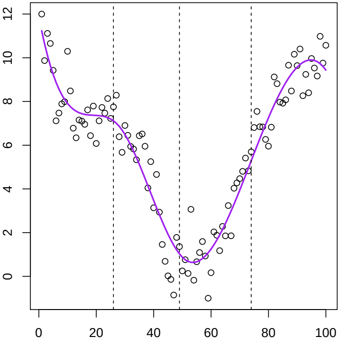


- Adaptive selection of knots comes from use of ℓ_1 penalty $\|D\beta\|_1$

What do we know about trend filtering?

Not a whole lot so far!

- Idea and name attributed to Kim et al. (2009), but essentially same idea appears earlier in Mammen and van de Geer (1997)
- Key property: trend filtering estimates can be viewed as piecewise polynomials, where knots are chosen adaptively

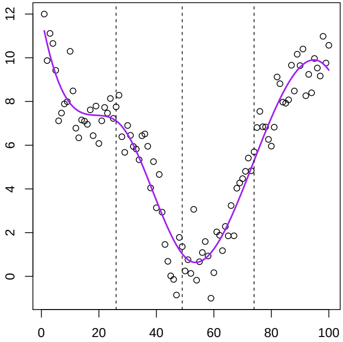


- Adaptive selection of knots comes from use of ℓ_1 penalty $\|D\beta\|_1$
- Smoothing splines are similar but use an ℓ_2 penalty of form $\beta^T \Omega \beta$

What do we know about trend filtering?

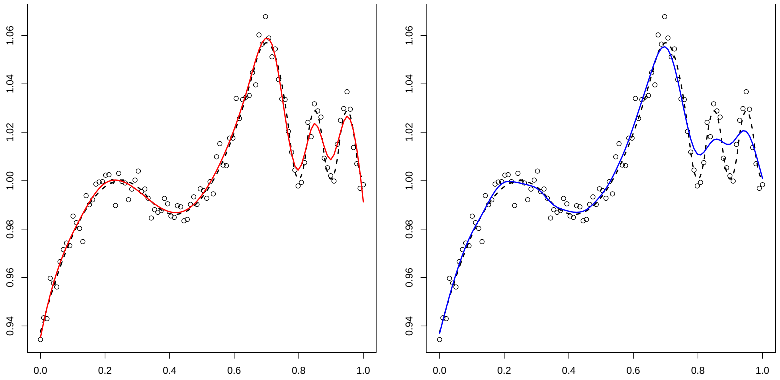
Not a whole lot so far!

- Idea and name attributed to Kim et al. (2009), but essentially same idea appears earlier in Mammen and van de Geer (1997)
- Key property: trend filtering estimates can be viewed as piecewise polynomials, where knots are chosen adaptively



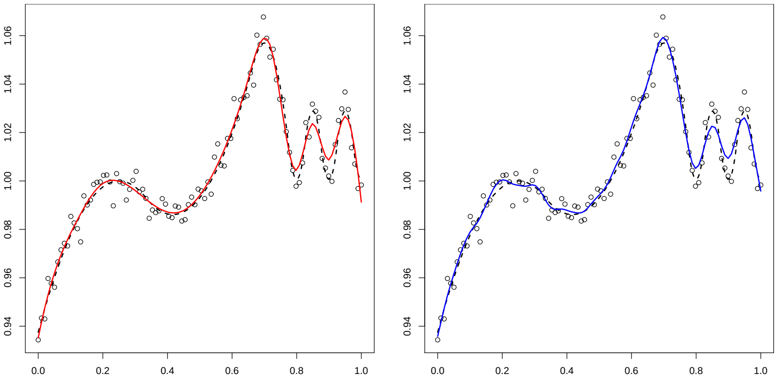
- Adaptive selection of knots comes from use of ℓ_1 penalty $\|D\beta\|_1$
- Smoothing splines are similar but use an ℓ_2 penalty of form $\beta^T \Omega \beta$
- Big difference: trend filtering can achieve exact zeros in $(k+1)$st derivative, smoothing splines cannot

Cubic trend filtering | Smoothing spline



$\widehat{\mathrm{df}} = 16$ | $\mathrm{df} = 16$

Cubic trend filtering | Smoothing spline



$\widehat{\mathrm{df}} = 16$ | $\mathrm{df} = 23$

Asymptotic convergence rate

Recall: we observe $(x_1, y_1), \ldots (x_n, y_n) \in \mathbb{R} \times \mathbb{R}$ from model

$$y_i = f(x_i) + \epsilon_i, \quad i = 1, \ldots n$$

and assume $x_1, \ldots x_n$ evenly spaced (hence fixed, nonrandom)

Asymptotic convergence rate

Recall: we observe $(x_1, y_1), \ldots (x_n, y_n) \in \mathbb{R} \times \mathbb{R}$ from model

$$y_i = f(x_i) + \epsilon_i, \quad i = 1, \ldots n$$

and assume $x_1, \ldots x_n$ evenly spaced (hence fixed, nonrandom)

Theorem (Mammen and van de Geer, 1997): Assume errors ϵ_i, $i = 1, \ldots n$ are independent with sub-Gaussian tails, and $f^{(k)}$ has bounded total variation. Then the trend filtering estimate of order k with $\lambda = \Theta(n^{1/(2k+1)})$ satisfies

$$\frac{1}{\sqrt{n}} \|\hat{\beta} - f\|_2 = O_P(n^{-k/(2k+1)})$$

Asymptotic convergence rate

Recall: we observe $(x_1, y_1), \dots (x_n, y_n) \in \mathbb{R} \times \mathbb{R}$ from model

$$y_i = f(x_i) + \epsilon_i, \quad i = 1, \dots n$$

and assume $x_1, \dots x_n$ evenly spaced (hence fixed, nonrandom)

Theorem (Mammen and van de Geer, 1997): Assume errors ϵ_i, $i = 1, \dots n$ are independent with sub-Gaussian tails, and $f^{(k)}$ has bounded total variation. Then the trend filtering estimate of order k with $\lambda = \Theta(n^{1/(2k+1)})$ satisfies

$$\frac{1}{\sqrt{n}} \|\hat{\beta} - f\|_2 = O_P(n^{-k/(2k+1)})$$

Trend filtering achieves the **minimax** rate of $n^{-k/(2k+1)}$ over assumed problem class (Nemirovskii et al., 1985).

Asymptotic convergence rate

Recall: we observe $(x_1, y_1), \ldots (x_n, y_n) \in \mathbb{R} \times \mathbb{R}$ from model

$$y_i = f(x_i) + \epsilon_i, \quad i = 1, \ldots n$$

and assume $x_1, \ldots x_n$ evenly spaced (hence fixed, nonrandom)

Theorem (Mammen and van de Geer, 1997): Assume errors ϵ_i, $i = 1, \ldots n$ are independent with sub-Gaussian tails, and $f^{(k)}$ has bounded total variation. Then the trend filtering estimate of order k with $\lambda = \Theta(n^{1/(2k+1)})$ satisfies

$$\frac{1}{\sqrt{n}} \|\hat{\beta} - f\|_2 = O_P(n^{-k/(2k+1)})$$

Trend filtering achieves the **minimax** rate of $n^{-k/(2k+1)}$ over assumed problem class (Nemirovskii et al., 1985). This rate cannot be achieved by estimates that are linear in observations, e.g., kernels and smoothing splines (Donoho and Johnstone, 1992)

How do we actually get solutions?

Trend filtering problem is generally **much harder** to solve than other nonparametric regression problems (e.g., smoothing splines, kernels, wavelets)

How do we actually get solutions?

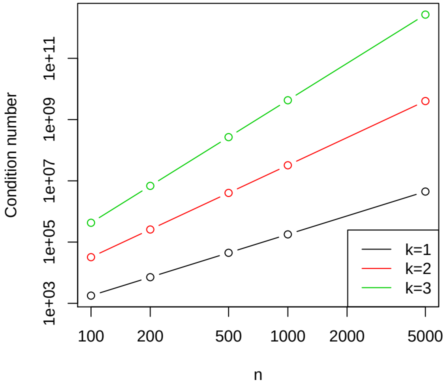
Trend filtering problem is generally **much harder** to solve than other nonparametric regression problems (e.g., smoothing splines, kernels, wavelets)

- Can apply many generic convex optimization techniques, but performance is bad: discrete derivative operator D is very ill-conditioned (note $D = D_k$, worse for larger k)

How do we actually get solutions?

Trend filtering problem is generally **much harder** to solve than other nonparametric regression problems (e.g., smoothing splines, kernels, wavelets)

- Can apply many generic convex optimization techniques, but performance is bad: discrete derivative operator D is very ill-conditioned (note $D = D_k$, worse for larger k)

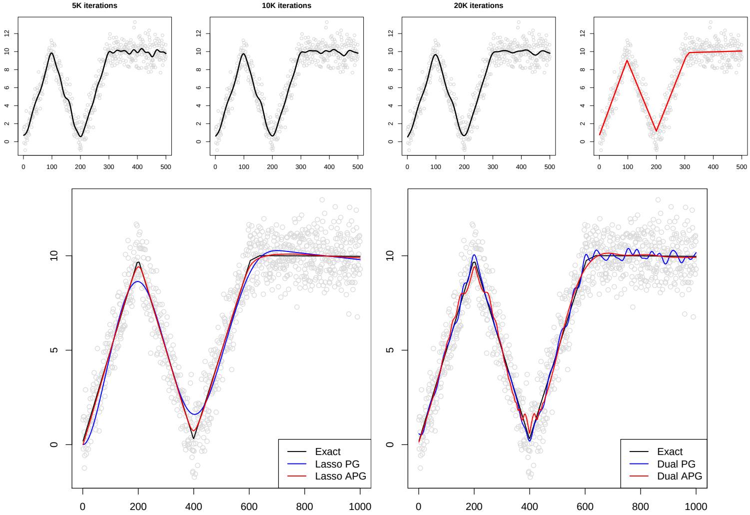


How do we actually get solutions?

- First order methods?

How do we actually get solutions?

- First order methods?



$n = 1000$, estimated solution after $20,000$ iterations.

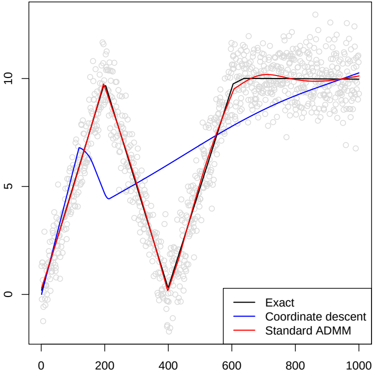
How do we actually get solutions?

How do we actually get solutions?

- Let us try to solve this problem via ADMM (Alternating Direction Method of Multipliers).

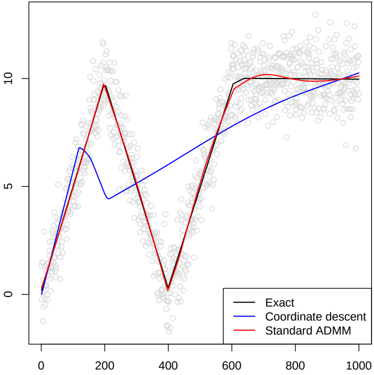
How do we actually get solutions?

- Let us try to solve this problem via ADMM (Alternating Direction Method of Multipliers).



How do we actually get solutions?

- Let us try to solve this problem via ADMM (Alternating Direction Method of Multipliers).



After 5000 iterations, still not good enough...

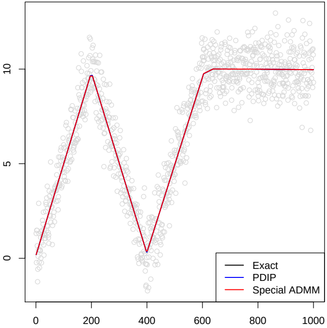
A Specialized ADMM

A Specialized ADMM

- Kim et al. (2009) propose specialized primal-dual interior point method for linear trend filtering.
- This is the current state of the art - way better than first order methods, coordinate descent, ADMM, etc.

A Specialized ADMM

- Kim et al. (2009) propose specialized primal-dual interior point method for linear trend filtering.
- This is the current state of the art - way better than first order methods, coordinate descent, ADMM, etc.
- Our proposal: A Specialized ADMM.



After just **twenty (yes, 20)** iterations.

A Specialized ADMM

Standard ADMM:

$$\min_{\beta \in \mathbb{R}^n,\, \alpha \in \mathbb{R}^{n-k-1}} \ \frac{1}{2}\|y-\beta\|_2^2 + \lambda\|\alpha\|_1 \ \ \text{subject to} \ \ \alpha = D^{(k+1)}\beta.$$

A Specialized ADMM

Standard ADMM:

$$\min_{\beta \in \mathbb{R}^n,\, \alpha \in \mathbb{R}^{n-k-1}} \ \frac{1}{2}\|y - \beta\|_2^2 + \lambda \|\alpha\|_1 \ \text{ subject to } \ \alpha = D^{(k+1)}\beta.$$

Specialized ADMM:

$$\min_{\beta \in \mathbb{R}^n,\, \alpha \in \mathbb{R}^{n-k}} \ \frac{1}{2}\|y - \beta\|_2^2 + \lambda \|D^{(1)}\alpha\|_1 \ \text{ subject to } \ \alpha = D^{(k)}\beta,$$

A Specialized ADMM

Standard ADMM:

$$\min_{\beta \in \mathbb{R}^n,\, \alpha \in \mathbb{R}^{n-k-1}} \; \frac{1}{2}\|y - \beta\|_2^2 + \lambda \|\alpha\|_1 \;\; \text{subject to} \;\; \alpha = D^{(k+1)}\beta.$$

Specialized ADMM:

$$\min_{\beta \in \mathbb{R}^n,\, \alpha \in \mathbb{R}^{n-k}} \; \frac{1}{2}\|y - \beta\|_2^2 + \lambda \|D^{(1)}\alpha\|_1 \;\; \text{subject to} \;\; \alpha = D^{(k)}\beta,$$

At every iteration:

$$\begin{aligned}
\beta &\leftarrow \big(I + \rho (D^{(k)})^T D^{(k)}\big)^{-1}\big(y + \rho (D^{(k)})^T(\alpha + u)\big), \\
\alpha &\leftarrow \operatorname*{argmin}_{\alpha \in \mathbb{R}^{n-k}} \; \frac{1}{2}\|\alpha - (D^{(k)}\beta - u)\|_2^2 + \lambda/\rho \|D^{(1)}\alpha\|_1, \\
u &\leftarrow u + \alpha - D^{(k)}\beta.
\end{aligned}$$

α-Update for Standard ADMM:

$$\alpha \leftarrow \underset{\alpha \in \mathbb{R}^{n-k-1}}{\operatorname{argmin}} \ \frac{1}{2}\|\alpha - (D^{(k+1)}\beta - u)\|_2^2 + \lambda/\rho\|\alpha\|_1,$$

α-Update for Standard ADMM:

$$\alpha \leftarrow \underset{\alpha \in \mathbb{R}^{n-k-1}}{\operatorname{argmin}} \ \frac{1}{2}\|\alpha - (D^{(k+1)}\beta - u)\|_2^2 + \lambda/\rho\|\alpha\|_1,$$

This is just soft-thresholding the vector $(D^{(k+1)}\beta - u)$!

α-Update for Standard ADMM:

$$\alpha \leftarrow \underset{\alpha \in \mathbb{R}^{n-k-1}}{\text{argmin}} \ \frac{1}{2}\|\alpha - (D^{(k+1)}\beta - u)\|_2^2 + \lambda/\rho\|\alpha\|_1,$$

This is just soft-thresholding the vector $(D^{(k+1)}\beta - u)$!

α-Update for Specialized ADMM:

$$\alpha \leftarrow \underset{\alpha \in \mathbb{R}^{n-k}}{\text{argmin}} \ \frac{1}{2}\|\alpha - (D^{(k)}\beta - u)\|_2^2 + \lambda/\rho\|D^{(1)}\alpha\|_1,$$

α-Update for Standard ADMM:

$$\alpha \leftarrow \underset{\alpha \in \mathbb{R}^{n-k-1}}{\operatorname{argmin}} \ \frac{1}{2}\|\alpha - (D^{(k+1)}\beta - u)\|_2^2 + \lambda/\rho\|\alpha\|_1,$$

This is just soft-thresholding the vector $(D^{(k+1)}\beta - u)$!

α-Update for Specialized ADMM:

$$\alpha \leftarrow \underset{\alpha \in \mathbb{R}^{n-k}}{\operatorname{argmin}} \ \frac{1}{2}\|\alpha - (D^{(k)}\beta - u)\|_2^2 + \lambda/\rho\|D^{(1)}\alpha\|_1,$$

Solved exactly by Dynamic Programming in linear time!

α-Update for Standard ADMM:

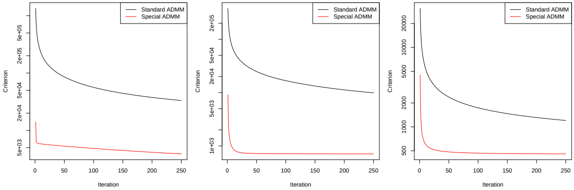
$$\alpha \leftarrow \underset{\alpha \in \mathbb{R}^{n-k-1}}{\operatorname{argmin}} \ \frac{1}{2}\|\alpha - (D^{(k+1)}\beta - u)\|_2^2 + \lambda/\rho\|\alpha\|_1,$$

This is just soft-thresholding the vector $(D^{(k+1)}\beta - u)$!

α-Update for Specialized ADMM:

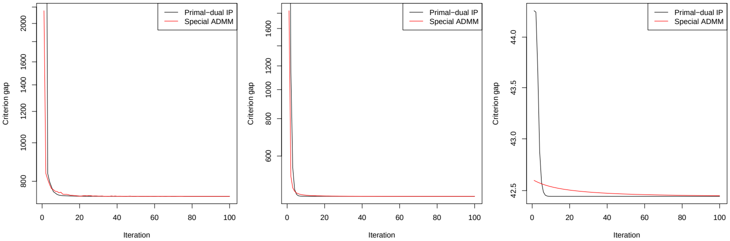
$$\alpha \leftarrow \underset{\alpha \in \mathbb{R}^{n-k}}{\operatorname{argmin}} \ \frac{1}{2}\|\alpha - (D^{(k)}\beta - u)\|_2^2 + \lambda/\rho\|D^{(1)}\alpha\|_1,$$

Solved exactly by Dynamic Programming in linear time!



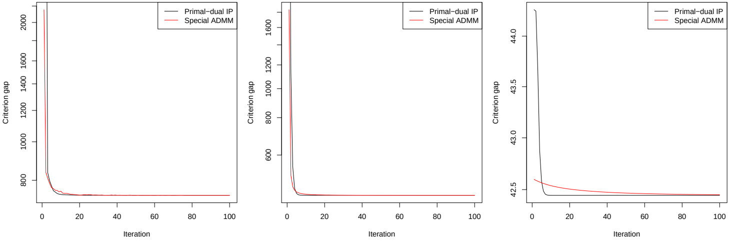
Doppler function, $k = 2$, $n = 10,000$, high, medium and low λ.

Specialized ADMM vs. Primal-Dual IP

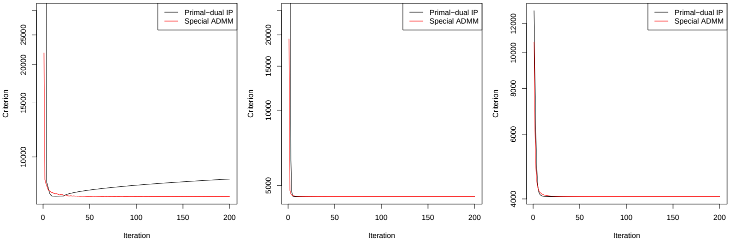


Sinusoidal function, $k = 1$, $n = 10,000$, high, medium and low λ.

Specialized ADMM vs. Primal-Dual IP

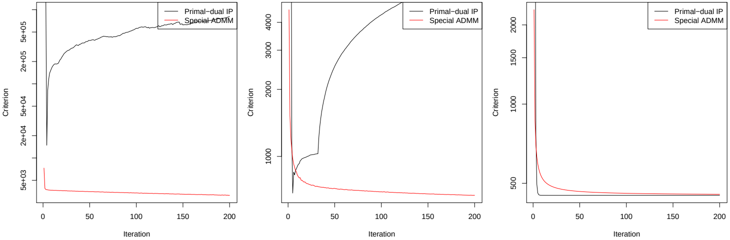


Sinusoidal function, $k = 1$, $n = 10,000$, high, medium and low λ.



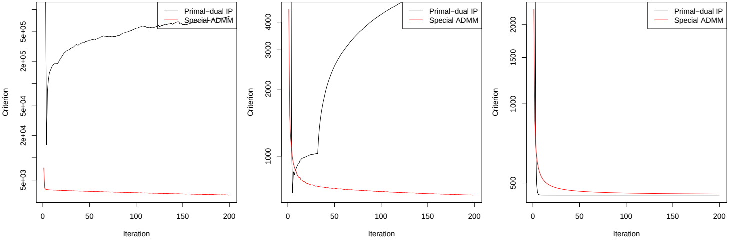
Sinusoidal function, $k = 1$, $n = 100,000$, high, medium and low λ.

Specialized ADMM vs. Primal-Dual IP

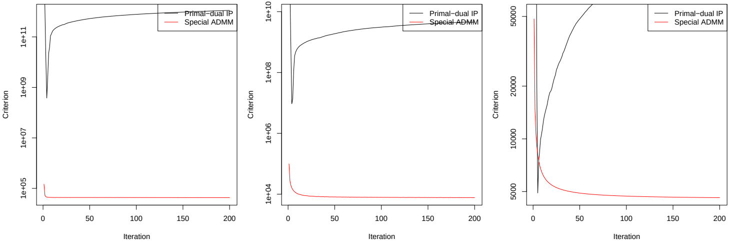


Sinusoidal function, $k = 2$, $n = 10,000$, high, medium and low λ.

Specialized ADMM vs. Primal-Dual IP

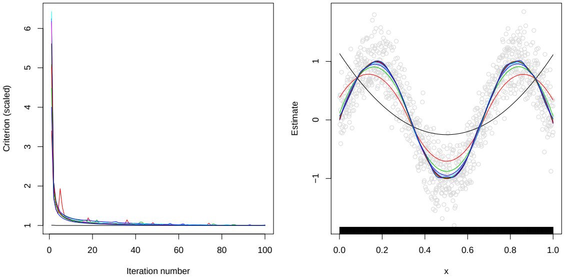


Sinusoidal function, $k = 2$, $n = 10,000$, high, medium and low λ.

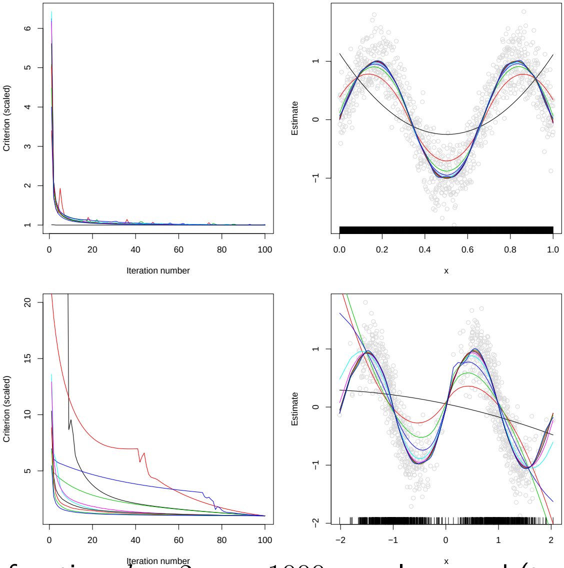


Sinusoidal function, $k = 2$, $n = 100,000$, high, medium and low λ.

An example with uneven points



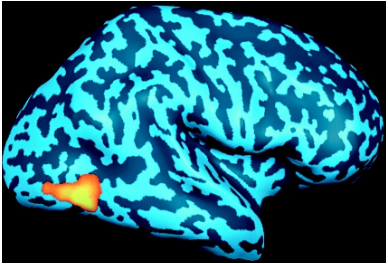
An example with uneven points



Sinusoidal function, $k = 2$, $n = 1000$, evenly spaced (top) vs. mixture of gaussians (bottom).

Object recognition in the brain

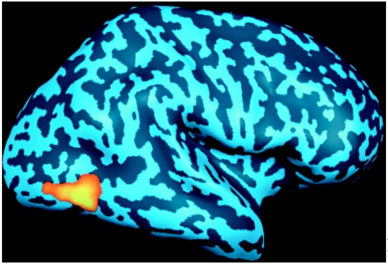
Lateral occipital complex (LOC): region of the occipital lobe believed to play a role in object recognition


[1]

[1](From http://www.siemens.com/innovation/en/publikationen/publications_pof/pof_spring_2007/functional_mr_imaging.htm)

Object recognition in the brain

Lateral occipital complex (LOC): region of the occipital lobe believed to play a role in object recognition

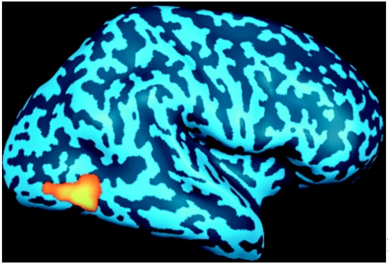
[1]

Question: *how long does it take LOC to pick up differences between objects?*

[1](From `http://www.siemens.com/innovation/en/publikationen/publications_pof/pof_spring_2007/functional_mr_imaging.htm`)

Object recognition in the brain

Lateral occipital complex (LOC): region of the occipital lobe believed to play a role in object recognition

[1]

Question: *how long does it take LOC to pick up differences between objects?*

Experimental data from Yang Xu, Ph.D. student in Machine Learning at Carnegie Mellon University (advisor: Rob Kass)

[1](From `http://www.siemens.com/innovation/en/publikationen/publications_pof/pof_spring_2007/functional_mr_imaging.htm`)

Measuring tool: magnetoencephalography (MEG), high temporal resolution

Measuring tool: magnetoencephalography (MEG), high temporal resolution

Simple setup:

Measuring tool: magnetoencephalography (MEG), high temporal resolution

Simple setup:

- Show someone a face:

Measuring tool: magnetoencephalography (MEG), high temporal resolution

Simple setup:

- Show someone a face:

- Record activity (magnetic responses) using MEG over 300 ms window

Measuring tool: magnetoencephalography (MEG), high temporal resolution

Simple setup:

- Show someone a face:

- Record activity (magnetic responses) using MEG over 300 ms window
- Do this 191 more times (191 more faces)

Measuring tool: magnetoencephalography (MEG), high temporal resolution

Simple setup:

- Show someone a face:

- Record activity (magnetic responses) using MEG over 300 ms window
- Do this 191 more times (191 more faces)

- Show someone a house:

Measuring tool: magnetoencephalography (MEG), high temporal resolution

Simple setup:

- Show someone a face:

- Record activity (magnetic responses) using MEG over 300 ms window
- Do this 191 more times (191 more faces)

- Show someone a house:

- Record activity (magnetic responses) using MEG over 300 ms window

Measuring tool: magnetoencephalography (MEG), high temporal resolution

Simple setup:

- Show someone a face:

- Record activity (magnetic responses) using MEG over 300 ms window
- Do this 191 more times (191 more faces)

- Show someone a house:

- Record activity (magnetic responses) using MEG over 300 ms window
- Do this 191 more times (191 more houses)

Question: *at what timepoint does the LOC start to process faces and houses differently?*

Question: *at what timepoint does the LOC start to process faces and houses differently?*

Data processing:

Question: *at what timepoint does the LOC start to process faces and houses differently?*

Data processing:

- MEG recordings are actually made at multiple spatial locations across LOC

Question: *at what timepoint does the LOC start to process faces and houses differently?*

Data processing:

- MEG recordings are actually made at multiple spatial locations across LOC
- Hence at each time point t, we have two arrays

$$F_{ij}(t) \quad \text{and} \quad H_{ij}(t)$$

with i indexing pictures, j indexing locations

Question: *at what timepoint does the LOC start to process faces and houses differently?*

Data processing:

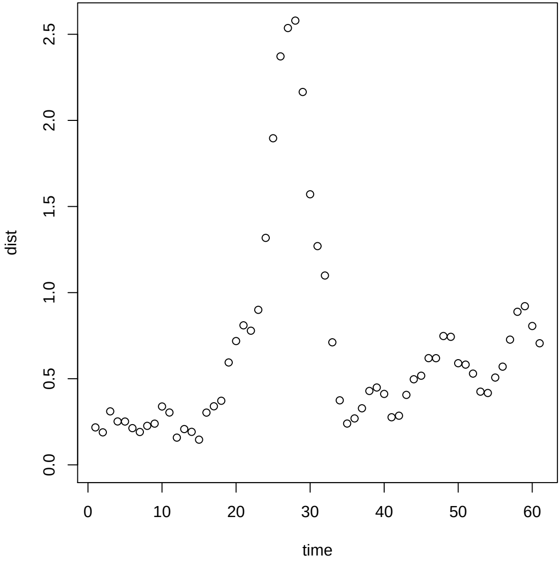
- MEG recordings are actually made at multiple spatial locations across LOC
- Hence at each time point t, we have two arrays

$$F_{ij}(t) \quad \text{and} \quad H_{ij}(t)$$

  with i indexing pictures, j indexing locations
- As a distance measure at t, we compute the sample Mahalanobis distance

$$\Delta_t = d_{\mathsf{Mahalanobis}}\big(F(t), H(t)\big)$$

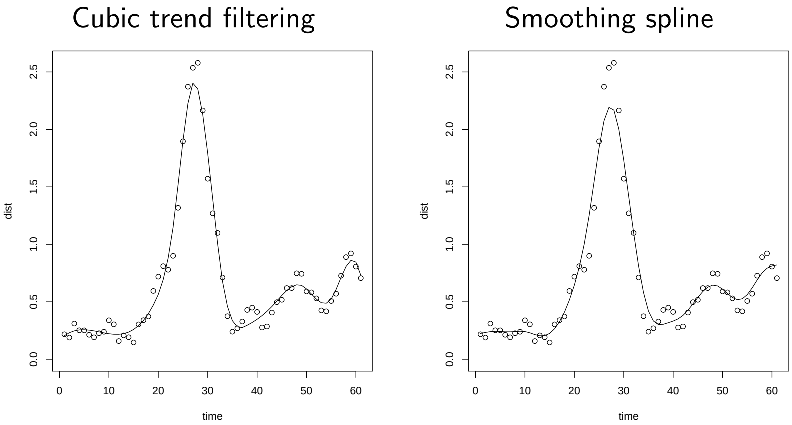
  (Just choosing one as reference distribution)



dist
time
0.0
0.5
1.0
1.5
2.0
2.5
0
10
20
30
40
50
60

Could fit trend filtering or smoothing spline, but these methods would never zero out a region

Could fit trend filtering or smoothing spline, but these methods would never zero out a region



(Both with 13 degrees of freedom)

Sparse trend filtering

Sparse trend filtering: additionally penalize the magnitude of the coefficients directly, i.e., solve

Sparse trend filtering

Sparse trend filtering: additionally penalize the magnitude of the coefficients directly, i.e., solve

$$\min_{\beta \in \mathbb{R}^n} \; \frac{1}{2}\sum_{i=1}^{n}(y_i - \beta_i)^2 + \lambda \sum_{i=1}^{n-k-1} \left| \sum_{j=i}^{i+k+1} (-1)^{j-i} \binom{k+1}{j-i} \beta_j \right| + \lambda\gamma \sum_{i=1}^{n} |\beta_i|$$

Sparse trend filtering

Sparse trend filtering: additionally penalize the magnitude of the coefficients directly, i.e., solve

$$\min_{\beta \in \mathbb{R}^n} \; \frac{1}{2}\sum_{i=1}^n (y_i - \beta_i)^2 + \lambda \sum_{i=1}^{n-k-1} \left| \sum_{j=i}^{i+k+1} (-1)^{j-i} \binom{k+1}{j-i} \beta_j \right| + \lambda\gamma \sum_{i=1}^n |\beta_i|$$

or

$$\min_{\beta \in \mathbb{R}^n} \; \frac{1}{2}\|y - \beta\|_2^2 + \lambda \|D_{k+1}\beta\|_1 + \lambda\gamma\|\beta\|$$

Sparse trend filtering

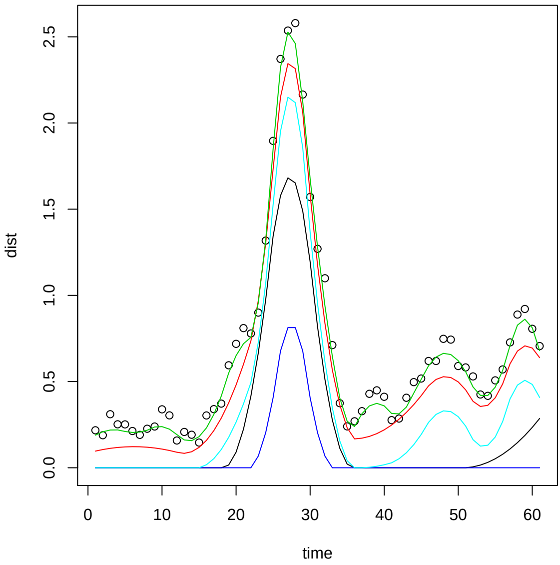
Sparse trend filtering: additionally penalize the magnitude of the coefficients directly, i.e., solve

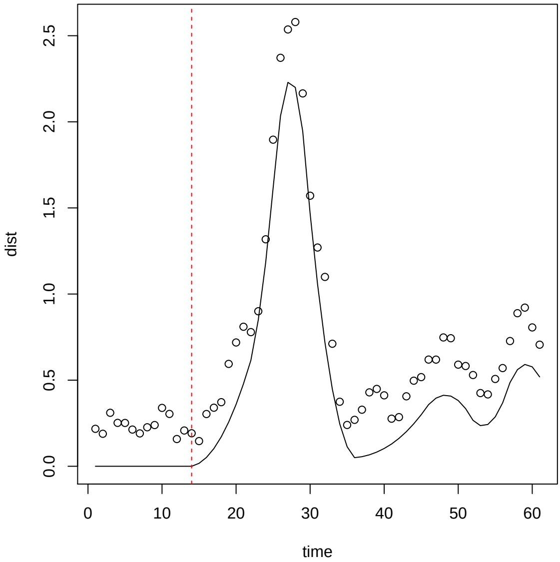
$$\min_{\beta \in \mathbb{R}^n} \; \frac{1}{2} \sum_{i=1}^{n} (y_i - \beta_i)^2 + \lambda \sum_{i=1}^{n-k-1} \left| \sum_{j=i}^{i+k+1} (-1)^{j-i} \binom{k+1}{j-i} \beta_j \right| + \lambda\gamma \sum_{i=1}^{n} |\beta_i|$$

or

$$\min_{\beta \in \mathbb{R}^n} \; \frac{1}{2} \|y - \beta\|_2^2 + \lambda \|D_{k+1}\beta\|_1 + \lambda\gamma \|\beta\|$$

Now we have two tuning parameters: λ and γ


dist
time
0.0
0.5
1.0
1.5
2.0
2.5
0
10
20
30
40
50
60



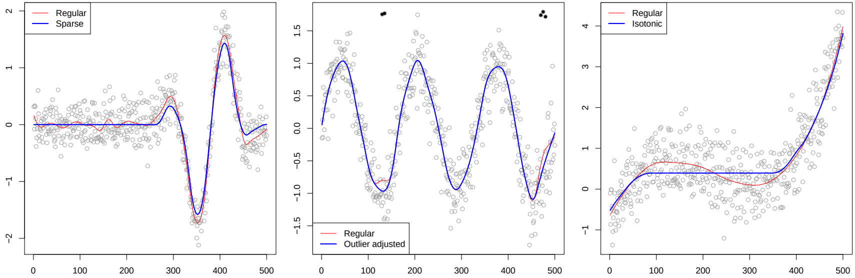
Leaves zero at $t = 14$, i.e. $\approx$ 70 ms, consistent with literature

Other Extensions - Easy to Derive

Key advantage of our ADMM over PDIP - easy to extend!

Other Extensions - Easy to Derive

Key advantage of our ADMM over PDIP - easy to extend!



Sparsity (left), Outlier detection (middle), isotonic (right).

Summary

Trend Filtering is a new and competitive alternative to splines.

Summary

Trend Filtering is a new and competitive alternative to splines.

- Minimax optimal, if you believe underlying function (or its derivatives) have bounded total variation (is piecewise constant/linear/...).
- Computationally efficient and numerically robust schemes are now available for large problems.
- Experiments on real and simulated data are very promising.
- Extensions are really easy!

Summary

Trend Filtering is a new and competitive alternative to splines.

- Minimax optimal, if you believe underlying function (or its derivatives) have bounded total variation (is piecewise constant/linear/...).
- Computationally efficient and numerically robust schemes are now available for large problems.
- Experiments on real and simulated data are very promising.
- Extensions are really easy!

People should try it out and develop their own opinions (see function `trendfilter`, in R package `genlasso`).

Acknowledgements

Ryan Tibshirani (CMU)

Thank you for listening