Putting it all together

Barnabas Poczos & Ryan Tibshirani
Convex Optimization 10-725/36-725

The big optimization toolbox

So far we've learned about:

- First-order methods
- Newton/quasi-Newton methods
- Interior point methods

These comprise a good part of the core tools in optimization, and are a big focus in this field

Given the number of available tools, it may seem overwhelming to choose a method in practice. A fair question: how to know what to use when?

By the way, there's still a lot more out there. Before the course is over we'll also learn:

- Dual methods
- Coordinate descent

Disclaimer

It's not possible to give a complete answer to this question. These are guidelines; any real problem should still be treated carefully

What are some important aspects to think about?

- Assumptions on criterion function
- Assumptions on constraint functions/set
- Ease of implementation (how to choose parameters?)
- Cost of each iteration
- Number of iterations needed

Other important aspects, that we won't consider: parallelization, data storage issues, statistical interplay

Outline

Today:

- Big table of methods
- Case study

	Gradient	Subgrad	Prox grad	Newton	Conj grad	quasi-Newton
Criterion	smooth f	any f	smooth $+$ simple, $f = g + h$	doubly smooth f	doubly smooth f	doubly smooth f
Constraints	projection onto constraint set	projection onto constraint set	constrained prox operator	equality constraints	unconstrained	unconstrained
Opti parameters	fixed step size ($t \leq 1/L$) or line search	diminishing step sizes	fixed step size ($t \leq 1/L$) or line search	pure step size ($t = 1$) or line search	FR & PR: line search, versions that use the Hessian: fixed step size	DFP & BFGS: line search
Iteration cost	cheap (compute gradient)	cheap (compute subgradient)	moderately cheap (evaluate prox)	moderate to expensive (compute Hessian and solve linear system)	moderately cheap (compute gradients, inner products)	moderately cheap (compute gradients, inner products; no matrix inversion, but storage for estimated inverse Hessian)
Rate	$O(\frac{1}{\epsilon})$ [$O(\frac{1}{\sqrt{\epsilon}})$ with acceleration, $O(\log(\frac{1}{\epsilon}))$ with strong convexity]	$O(\frac{1}{\epsilon^2})$	$O(\frac{1}{\epsilon})$ [$O(\frac{1}{\sqrt{\epsilon}})$ with acceleration, $O(\log(\frac{1}{\epsilon}))$ with strong convexity]	$O(\log\log(\frac{1}{\epsilon}))$ (quadratic rate)	superlinear rate, n-step quadratic rate (n steps are as effective as one Newton step)	superlinear rate, n-step quadratic rate (n steps are as effective as one Newton step)

	Barrier method	Primal-dual IPM	ADMM	Coord desc
Criterion	doubly smooth f	doubly smooth f	block separable, $f(x,z) = g(x) + h(z)$	smooth + component-wise separable
Constraints	doubly smooth h_i (ineq constraints)	doubly smooth h_i (ineq constraints)	equality constraints (always) & ineq constraints (sometimes)	component-wise separable constraints
Opti parameters	inner loop: fixed step size or use line search, outer loop: diverging barrier parameter	line search for step size & diverging barrier parameter	fixed augmented Lagrange parameter (theory), or varied by iteration (practice)	none!
Iteration cost	expensive to very expensive (one iteration solves one smoothed problem, by Newton)	moderate to expensive (one iteration performs one Newton step)	cheap to expensive (one iteration solves two subproblems, makes a dual step)	cheap to expensive (one iteration performs a full cycle of component minimizations)
Rate	$O(\log(\frac{1}{\epsilon}))$ (both in terms of iterations and Newton steps)	$O(\log(\frac{1}{\epsilon}))$	not known in general, but known in special cases; practically tends to behave like a first-order method	not known in general, but known in special cases; practically tends to behave faster than first-order methods

Case study: generalized lasso regularization

Consider the problem

$$\min_{\beta \in \mathbb{R}^n} \; f(\beta) + \lambda \|D\beta\|_1$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is a smooth, convex function and $D \in \mathbb{R}^{m \times n}$ is a penalty matrix. This is called a generalized lasso regularization problem

Usual lasso, $D = I$, encodes sparsity in β, while the generalized lasso encodes sparsity in

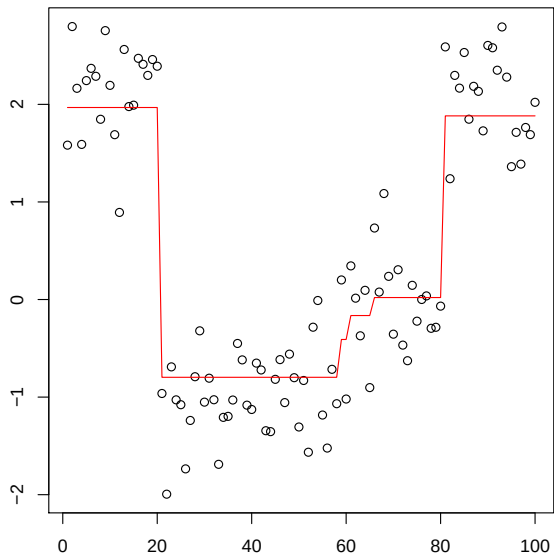
$$D\beta = \begin{pmatrix} D_1\beta \\ \vdots \\ D_m\beta \end{pmatrix}$$

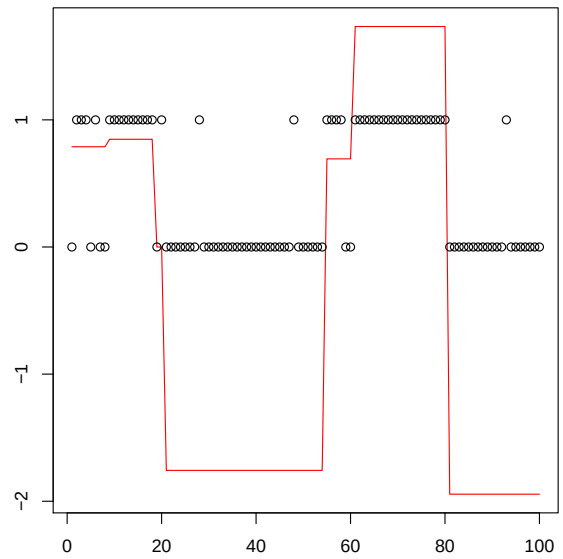
where $D_1, \ldots D_m$ are the rows of D. This can yield interesting structure in β, depending on choice of D

Special case: fused lasso

$$D = \begin{bmatrix} -1 & 1 & 0 & \dots & 0 & 0 \\ 0 & -1 & 1 & \dots & 0 & 0 \\ \vdots & & & & & \\ 0 & 0 & 0 & \dots & -1 & 1 \end{bmatrix}, \text{ so } \|D\beta\|_1 = \sum_{i=1}^{n-1} |\beta_i - \beta_{i+1}|$$

E.g.,





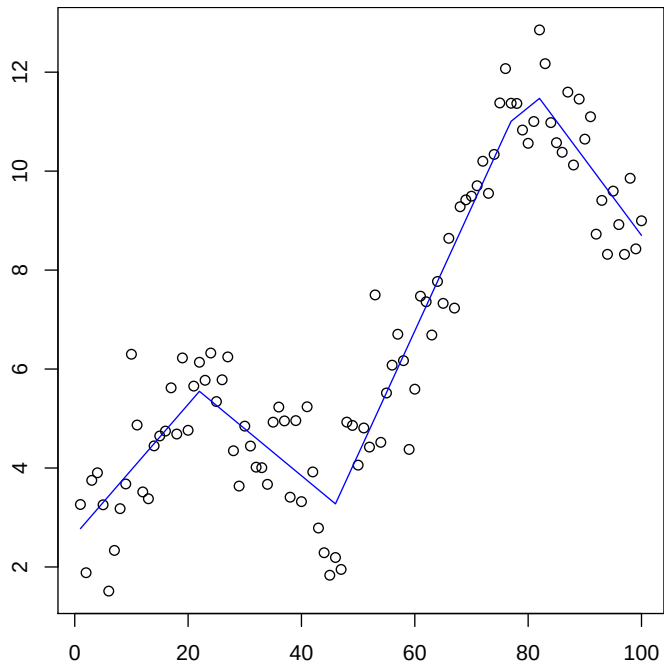
$$f(\beta) = \sum_{i=1}^{n} (y_i - \beta_i)^2$$

$$f(\beta) = \sum_{i=1}^{n} (-z_i \beta_i + \log(1 + e^{-\beta_i}))$$

Special case: linear trend filtering

$$D = \begin{bmatrix} 1 & -2 & 1 & \dots & 0 & 0 \\ 0 & 1 & -2 & \dots & 0 & 0 \\ \vdots & & & & & \\ 0 & 0 & 0 & \dots & -2 & 1 \end{bmatrix}, \text{ so } \|D\beta\|_1 = \sum_{i=1}^{n-2} |\beta_i - 2\beta_{i+1} - \beta_{i+2}|$$

E.g.,



$$f(\beta) = \sum_{i=1}^{n} (y_i - \beta_i)^2$$

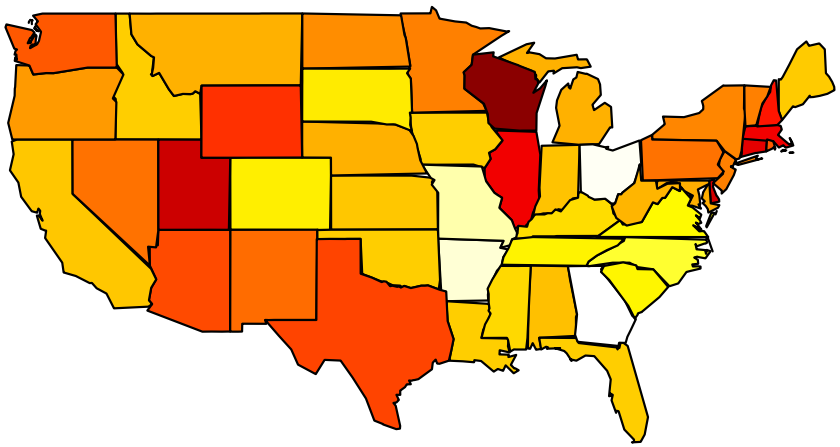
Special case: fused lasso on a graph, $G = (\{1, \ldots n\}, E)$. Here D is $|E| \times n$, and if $e_\ell = (i, j)$, then D has ℓth row

$$D_\ell = (0, \ldots \underset{\uparrow}{-1}_{i}, \ldots \underset{\uparrow}{1}_{j}, \ldots 0)$$

so

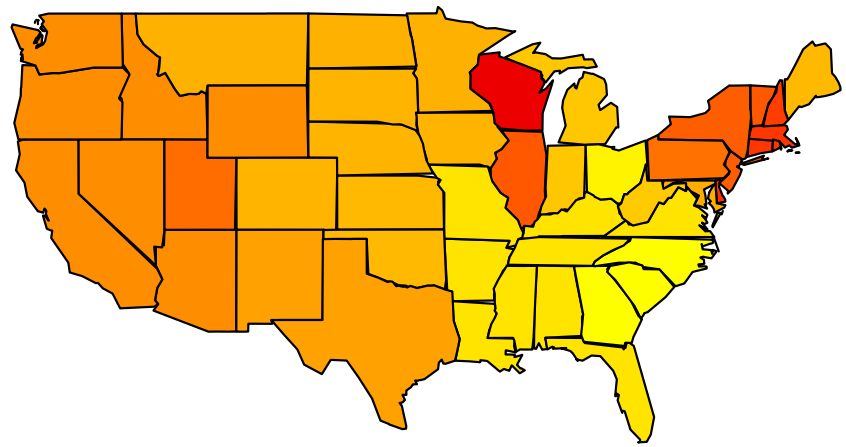
$$\|D\beta\|_1 = \sum_{(i,j) \in E} |\beta_i - \beta_j|$$

E.g., $f(\beta) = \sum_{i=1}^n (y_i - \beta_i)^2$,



observed data

$y_1, \ldots y_n$



fused lasso solution

$\hat{\beta}_1, \ldots \hat{\beta}_n$

Special case: equality constrained lasso regularization problem

Consider the problem

$$\min_{\beta \in \mathbb{R}^n} \; f(\beta) + \lambda \|\beta\|_1 \;\; \text{subject to} \;\; A\beta = 0$$

Reparametrize as follows: let $D \in \mathbb{R}^{n \times p}$ have columns that span $\text{null}(A)$; then $A\beta = 0 \iff \beta = D\theta$ for some $\theta \in \mathbb{R}^p$

Therefore the above is equivalent to the problem

$$\min_{\theta \in \mathbb{R}^p} \; f(D\theta) + \lambda \|D\theta\|_1$$

Note that D here is generically dense and unstructured

Generalized lasso algorithms

So, how to solve

$$\min_{\beta \in \mathbb{R}^n} \; f(\beta) + \lambda \|D\beta\|_1 \; ?$$

For the subgradient method, we repeat updates in the direction

$$\Delta = \nabla f(\beta) + \lambda D^T \gamma$$

where $\gamma \in \partial \|x\|_1$ evaluated at $x = D\beta$, i.e.,

$$\gamma_i \in \begin{cases} \{\mathrm{sign}\big((D\beta)_i\big)\} & \text{if } D\beta_i \neq 0 \\ [-1, 1] & \text{if } D\beta_i = 0 \end{cases}, \quad i = 1, \ldots m$$

This downside (as usual) is that convergence is slow. However, if $(D\beta)_i \neq 0$ for $i \in S$, then

$$\Delta = \nabla f(\beta) + \lambda \sum_{i \in S} \mathrm{sign}\big((D\beta)_i\big) \cdot D_i$$

is a proper update direction, where D_i is the ith row of D

What about generalized gradient descent? Prox operator is

$$\text{prox}_t(\beta) = \underset{z \in \mathbb{R}^n}{\text{argmin}} \ \frac{1}{2t}\|\beta - z\|_2^2 + \lambda \|Dz\|_1$$

This is not easy for a generic D (as opposed to soft-thresholding, corresponding to $D = I$) ... in fact, this itself is a pretty hard optimization problem

Nothing else obviously applicable (of what we learned so far) ... so derive the dual problem

In fact, it is never a bad idea to look at the dual, even when you think you have a good approach for the primal problem!

The dual of

$$\min_{\beta \in \mathbb{R}^n} \; f(\beta) + \lambda \|D\beta\|_1$$

is

$$\min_{u \in \mathbb{R}^m} \; f^*(-D^T u) \;\; \text{subject to} \;\; \|u\|_\infty \leq \lambda$$

Here f^* is the conjugate of f. Note that u is m dimensional (the number of rows of D) while β is n dimensional

The primal and dual solutions $\hat{\beta}$ and $\hat{u}$ are related by

$$\nabla f(\hat{\beta}) + D^T \hat{u} = 0$$

and

$$\hat{u}_i \in \begin{cases} \{\lambda\} & \text{if } (D\hat{\beta})_i > 0 \\ \{-\lambda\} & \text{if } (D\hat{\beta})_i < 0 \\ [-\lambda, \lambda] & \text{if } (D\hat{\beta})_i = 0 \end{cases}, \quad i = 1, \ldots m$$

So $\hat{u}_i \in (-\lambda, \lambda) \Longrightarrow (D\hat{\beta})_i = 0$

$$\text{Primal} \quad : \quad \min_{\beta \in \mathbb{R}^n} \; f(\beta) + \lambda \|D\beta\|_1$$

$$\text{Dual} \quad : \quad \min_{u \in \mathbb{R}^m} \; f^*(-D^T u) \;\; \text{subject to} \;\; \|u\|_\infty \leq \lambda$$

General trend is that:

- For large λ, many components $(D\hat{\beta})_i$ are zero, and many components $\hat{u}_i$ are strictly between $-\lambda$ and λ
- For small λ, many components $(D\hat{\beta})_i$ are nonzero, and many components $\hat{u}_i$ are equal to λ or $-\lambda$

When many $(D\hat{\beta})_i$ are zero, there are fewer "effective parameters" to fit in the primal; when many $\hat{u}_i$ are $\pm\lambda$, the same is true in the dual

Therefore, keep in mind:

- Large $\lambda \implies$ easier problem for primal algorithms
- Small $\lambda \implies$ easier problem for dual algorithms

How to solve the dual

$$\min_{u \in \mathbb{R}^m} \; f^*(-D^T u) \;\; \text{subject to} \;\; \|u\|_\infty \leq \lambda \;\; ?$$

Generalized gradient is now tractable, because the prox operator

$$\text{prox}_t(u) = \underset{z \in \mathbb{R}^m}{\text{argmin}} \; \frac{1}{2t}\|u - z\|_2^2 \;\; \text{subject to} \;\; \|z\|_\infty \leq \lambda$$

is easy. Note that this is just projection onto a box, i.e., the point $\hat{z} = \text{prox}_t(u)$ has components

$$\hat{z}_i = \begin{cases} \lambda & \text{if } u_i > \lambda \\ -\lambda & \text{if } u_i < -\lambda \\ u_i & \text{if } u_i \in [-\lambda, \lambda] \end{cases}, \quad i = 1, \ldots m$$

Can also rewrite dual problem as

$$\min_{u \in \mathbb{R}^m} \; f^*(-D^T u) \;\; \text{subject to} \;\; -\lambda \leq u_i \leq \lambda, \;\; i = 1, \ldots m$$

These are just linear constraints, so we can easily apply an interior point method

With the barrier method, we repeatedly solve

$$\min_{u \in \mathbb{R}^m} \; t \cdot f^*(-D^T u) + \phi(u)$$

for larger and larger values of the barrier parameter t, where

$$\phi(u) = -\sum_{i=1}^{m} \big(\log(\lambda - u_i) + \log(u_i + \lambda) \big)$$

How efficient is this?

We will solve the inner problem

$$\min_{u \in \mathbb{R}^m} \; t \cdot f^*(-D^T u) + \phi(u)$$

using Newton's method. Let $F(u) = t f^*(-D^T u) + \phi(u)$; then the Newton updates are given by

$$\Delta = -\big(\nabla^2 F(u)\big)^{-1} \nabla F(u)$$

Note that

$$\nabla F(u) = -t \cdot D\big(\nabla f^*(-D^T u)\big) + \nabla \phi(u)$$
$$\nabla^2 F(u) = t \cdot D\big(\nabla^2 f^*(-D^T u)\big) D^T + \nabla^2 \phi(u)$$

To compute the Newton update Δ, we need to invert $\nabla^2 F(u)$.

- Second term: $\nabla^2 \phi(u)$ is a diagonal matrix
- First term: if both $\nabla^2 f^*(x)$ and D are structured matrices, then $D \nabla^2 f^*(x) D^T$ can too be structured

Putting it all together:

- Primal subgradient method: iterations are cheap (we sum up rows of D over active set S), but convergence is slow
- Primal generalized gradient: iterations involve evaluating

$$\text{prox}_t(\beta) = \underset{z \in \mathbb{R}^n}{\text{argmin}} \ \frac{1}{2t}\|\beta - z\|_2^2 + \lambda \|Dz\|_1$$

  which is generally expensive, convergence is medium
- Dual generalized gradient: iterations involve projecting onto a box (very efficient), convergence is medium
- Dual barrier method: Newton iterations involve solving linear system in

$$t \cdot D\big(\nabla^2 f^*(-D^T u)\big) D^T + \nabla^2 \phi(u)$$

  which may or may not be expensive, convergence is fast

Back to some examples

Suppose that we are studying the linear trend filtering problem, so

$$D = \begin{bmatrix} 1 & -2 & 1 & \dots & 0 & 0 \\ 0 & 1 & -2 & \dots & 0 & 0 \\ \vdots & & & & & \\ 0 & 0 & 0 & \dots & -2 & 1 \end{bmatrix},$$

and the loss is either least squares loss $f(\beta) = \sum_{i=1}^n (y_i - \beta_i)^2$, or logistic loss $f(\beta) = \sum_{i=1}^n (-z_i\beta_i + \log(1 + e^{-\beta_i}))$

Suppose further that we desire a solution at a fairly high level of accuracy—otherwise, we notice "wiggles" in the plotted trend $\hat{\beta}_1, \dots \hat{\beta}_n$

What algorithm to use?

Primal subgradient and primal generalized gradient are out

As for dual algorithms, one can check that the conjugate f^* can be derived in closed form for both the least squares and logisitic losses. Moreover, $\nabla^2 f^*(x)$ is a diagonal matrix in both of these cases

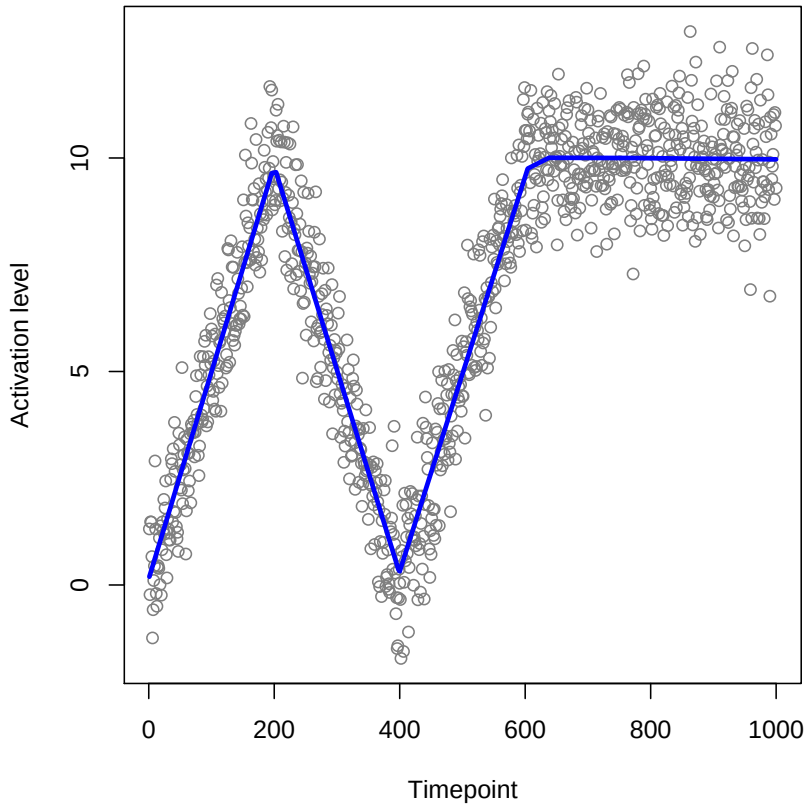
Therefore the Newton steps in the barrier method involve solving a linear system in

$$DW(u)D^T$$

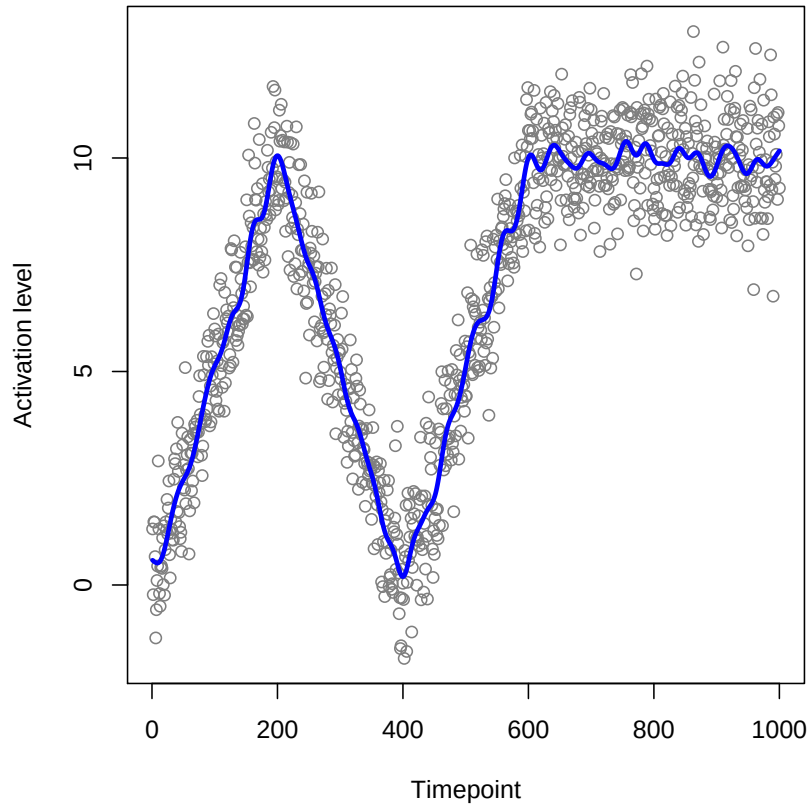
where $W(u)$ is a diagonal matrix; this is a highly sparse, structured matrix, and so these systems can be solved very efficiently. Hence, the barrier method is the way to go

Dual generalized gradient descent admits very efficient iterations, but takes far too long to converge to high accuracy (suffers also from the poor conditioning of D here)

Recall example from our first lecture:



Dual interior point method
30 iterations



Dual generalized gradient
10,000 iterations

The same story holds when D corresponds to the fused lasso on a general graph: the barrier method has very efficient Newton steps, and is the preferred method

However, for the simple (1-dimensional) fused lasso,

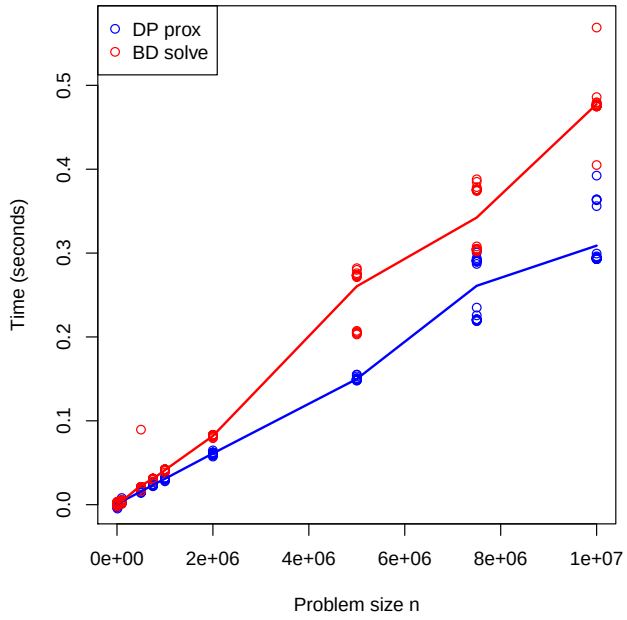
$$D = \begin{bmatrix} -1 & 1 & 0 & \dots & 0 & 0 \\ 0 & -1 & 1 & \dots & 0 & 0 \\ \vdots & & & & & \\ 0 & 0 & 0 & \dots & -1 & 1 \end{bmatrix}$$

the prox function in the primal is

$$\text{prox}_t(\beta) = \underset{z \in \mathbb{R}^n}{\text{argmin}} \ \frac{1}{2t}\|\beta - z\|_2^2 + \lambda \sum_{i=1}^{n} |z_i - z_{i+1}|$$

and this can be evaluated directly using a specialized, fast dynamic programming algorithm[1]

[1] Johnson (2013), "A dynamic programming algorithm for the fused lasso and L_0-segmentation"



1 iteration of primal generalized gradient versus 1 iteration of dual barrier method

Hence, for D corresponding to the 1-dimensional fused lasso, the primal generalized gradient descent algorithm is preferred

This is because it requires a simpler implementation (once the prox has been given to us), and it applies to any smooth f (not just the least squares loss or logistic regression loss)

Also, it favors the large λ setting, which is typically (statisically) the setting we are interested in

Instead conder a problem with a dense, generic D, e.g., stemming from an equality constrained lasso problem

Primal prox is intractable, and the dual barrier method has costly Newton steps

But (provided that we can form f^*), dual generalized gradient descent still features efficient iterations: after evaluating

$$u + t \cdot D\big(\nabla f^*(-D^T u)\big)$$

we just project onto the box $[-\lambda, \lambda]^m$

So, especially if we do not need a highly accurate solution, dual generalized gradient is the best method

Finally, consider a problem in which D is dense and so massive that even fitting it in memory is a burden

Depending on f and its gradient, subgradient method may be the only feasible algorithm; recall that the subgradient update direction can be taken as

$$\nabla f(\beta) + \lambda \sum_{i \in S} \operatorname{sign}\big((D\beta)_i\big) \cdot D_i$$

where S is the set of all i such that $(D\beta)_i \neq 0$

If λ is large enough so that many $(D\beta)_i = 0$, then we only need to fit a small part of D in memory (or, read a small part of D from a file) to perform subgradient updates