Is Greedy Coordinate Descent a Terrible Algorithm?

Julie Nutini, Mark Schmidt, Issam Laradji,
Michael Friedlander, Hoyt Koepke

University of British Columbia

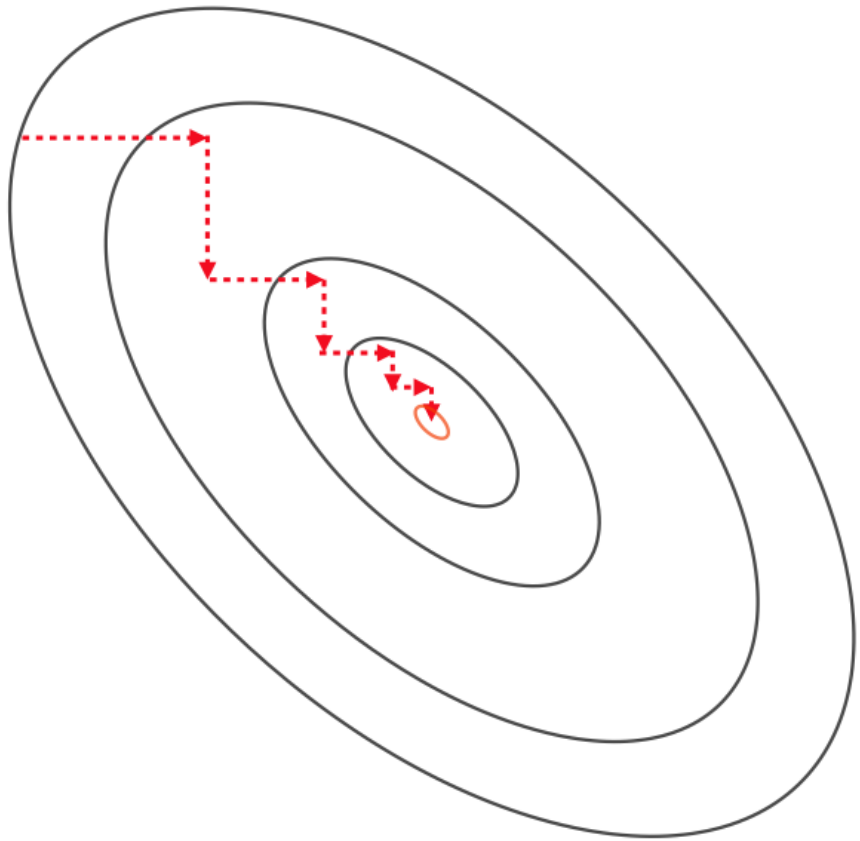
Optimization and Big Data, 2015

Context: Random vs. Greedy Coordinate Descent

- We consider coordinate descent for large-scale optimization:
  1. Select a coordinate to update.
  2. Take a small gradient step along coordinate.

Context: Random vs. Greedy Coordinate Descent

- We consider coordinate descent for large-scale optimization:
  1. Select a coordinate to update.
  2. Take a small gradient step along coordinate.



Context: Random vs. Greedy Coordinate Descent

- We consider coordinate descent for large-scale optimization:
  1. Select a coordinate to update.
  2. Take a small gradient step along coordinate.
- Recent interest began with Nesterov [2010]:
  - Global convergence rate for randomized coordinate selection.
  - Faster than gradient descent if iterations are n times cheaper.

Context: Random vs. Greedy Coordinate Descent

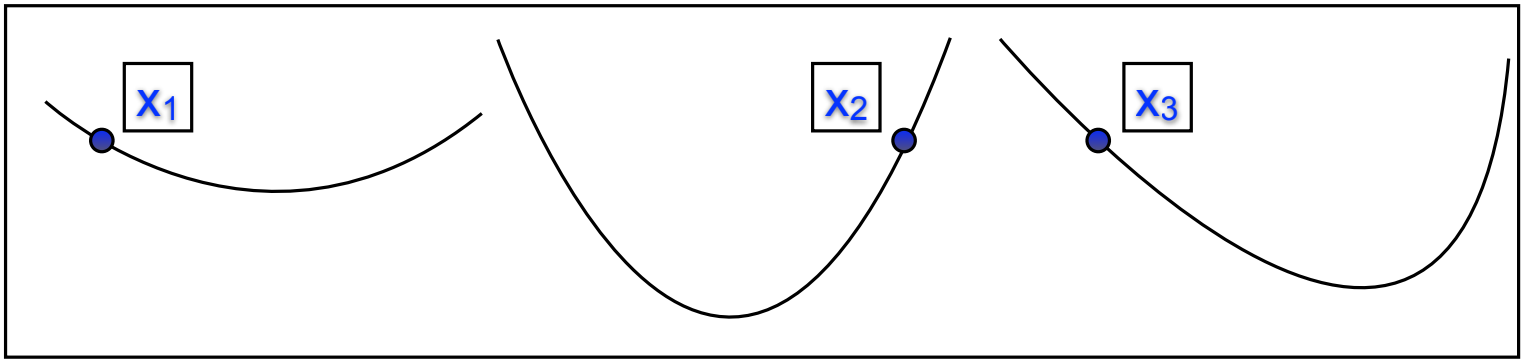
- We consider coordinate descent for large-scale optimization:
  1. Select a coordinate to update.
  2. Take a small gradient step along coordinate.
- Recent interest began with Nesterov [2010]:
  - Global convergence rate for randomized coordinate selection.
  - Faster than gradient descent if iterations are n times cheaper.
- Contrast random with classic Gauss-Southwell rule:

$$\underset{i}{\operatorname{argmax}} |\nabla_i f(x)|.$$

Context: Random vs. Greedy Coordinate Descent

- We consider coordinate descent for large-scale optimization:
  1. Select a coordinate to update.
  2. Take a small gradient step along coordinate.
- Recent interest began with Nesterov [2010]:
  - Global convergence rate for randomized coordinate selection.
  - Faster than gradient descent if iterations are n times cheaper.
- Contrast random with classic Gauss-Southwell rule:

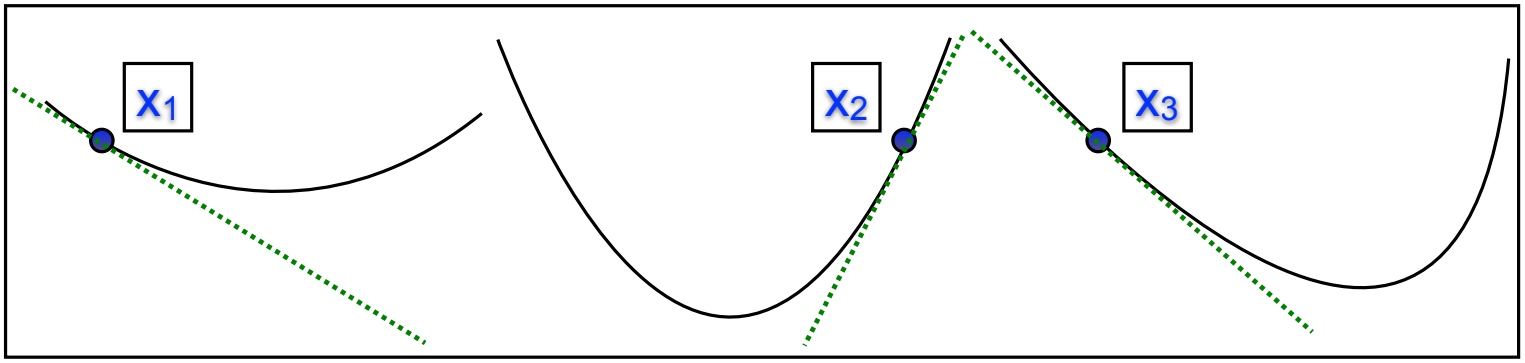
$$\underset{i}{\operatorname{argmax}} |\nabla_i f(x)|.$$



Context: Random vs. Greedy Coordinate Descent

- We consider coordinate descent for large-scale optimization:
  1. Select a coordinate to update.
  2. Take a small gradient step along coordinate.
- Recent interest began with Nesterov [2010]:
  - Global convergence rate for randomized coordinate selection.
  - Faster than gradient descent if iterations are n times cheaper.
- Contrast random with classic Gauss-Southwell rule:

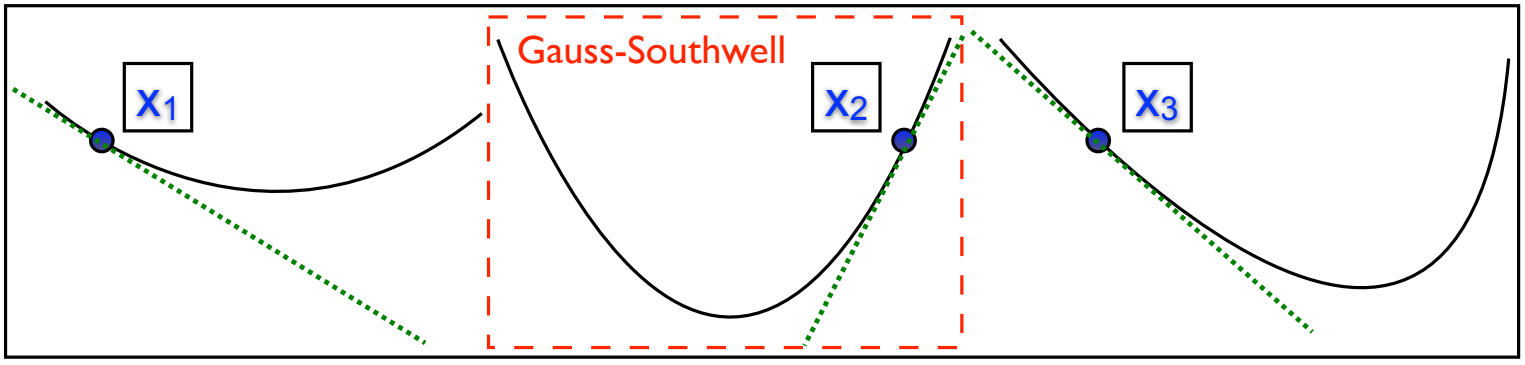
$$\underset{i}{\operatorname{argmax}} |\nabla_i f(x)|.$$



Context: Random vs. Greedy Coordinate Descent

- We consider coordinate descent for large-scale optimization:
  1. Select a coordinate to update.
  2. Take a small gradient step along coordinate.
- Recent interest began with Nesterov [2010]:
  - Global convergence rate for randomized coordinate selection.
  - Faster than gradient descent if iterations are n times cheaper.
- Contrast random with classic Gauss-Southwell rule:

$$\underset{i}{\operatorname{argmax}} |\nabla_i f(x)|.$$



Context: Random vs. Greedy Coordinate Descent

- We consider coordinate descent for large-scale optimization:
  1. Select a coordinate to update.
  2. Take a small gradient step along coordinate.
- Recent interest began with Nesterov [2010]:
  - Global convergence rate for randomized coordinate selection.
  - Faster than gradient descent if iterations are n times cheaper.
- Contrast random with classic Gauss-Southwell rule:

$$\underset{i}{\operatorname{argmax}} |\nabla_i f(x)|.$$

  - Gauss-Southwell (GS) is at least as expensive as random.
  - But Nesterov showed the rate is the same.
  - So greedy is a terrible algorithm and just use random!

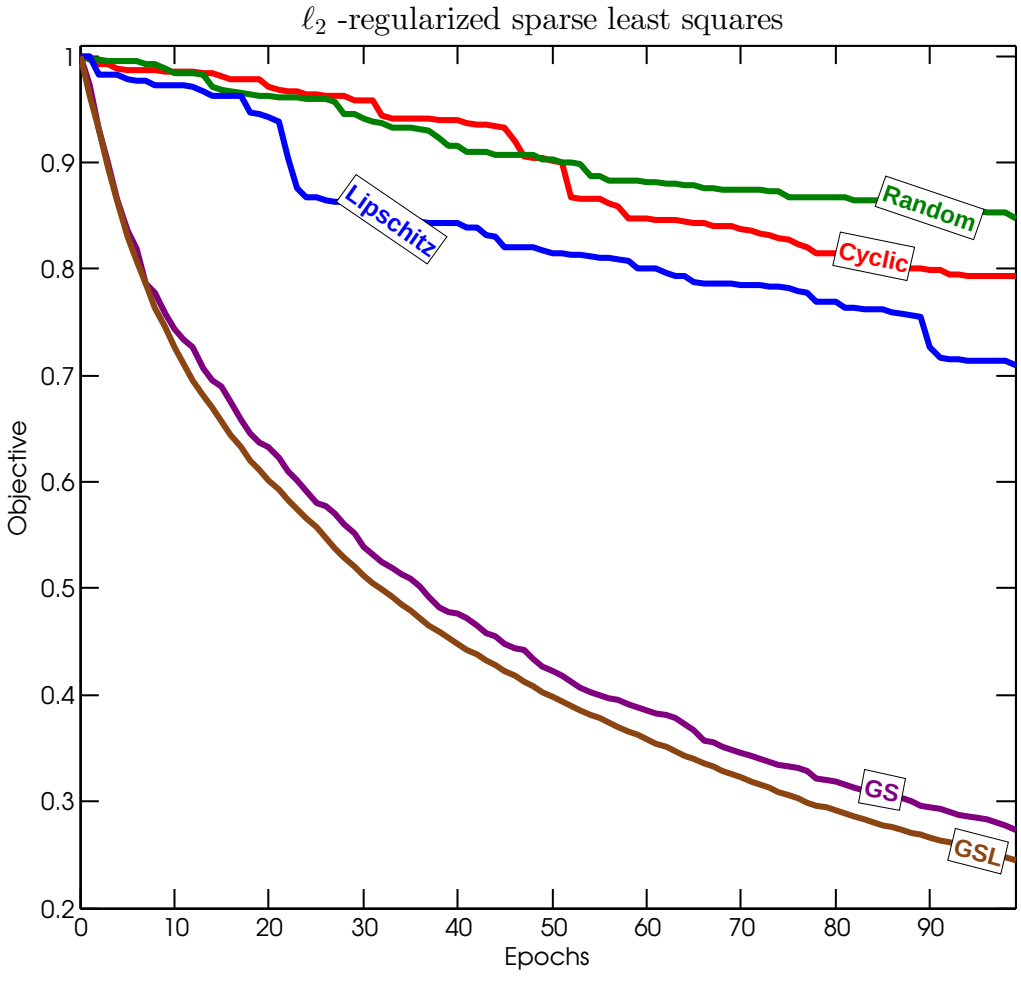
Context: Random vs. Greedy Coordinate Descent

- We consider coordinate descent for large-scale optimization:
  1. Select a coordinate to update.
  2. Take a small gradient step along coordinate.
- Recent interest began with Nesterov [2010]:
  - Global convergence rate for randomized coordinate selection.
  - Faster than gradient descent if iterations are n times cheaper.
- Contrast random with classic Gauss-Southwell rule:

$$\underset{i}{\operatorname{argmax}} |\nabla_i f(x)|.$$

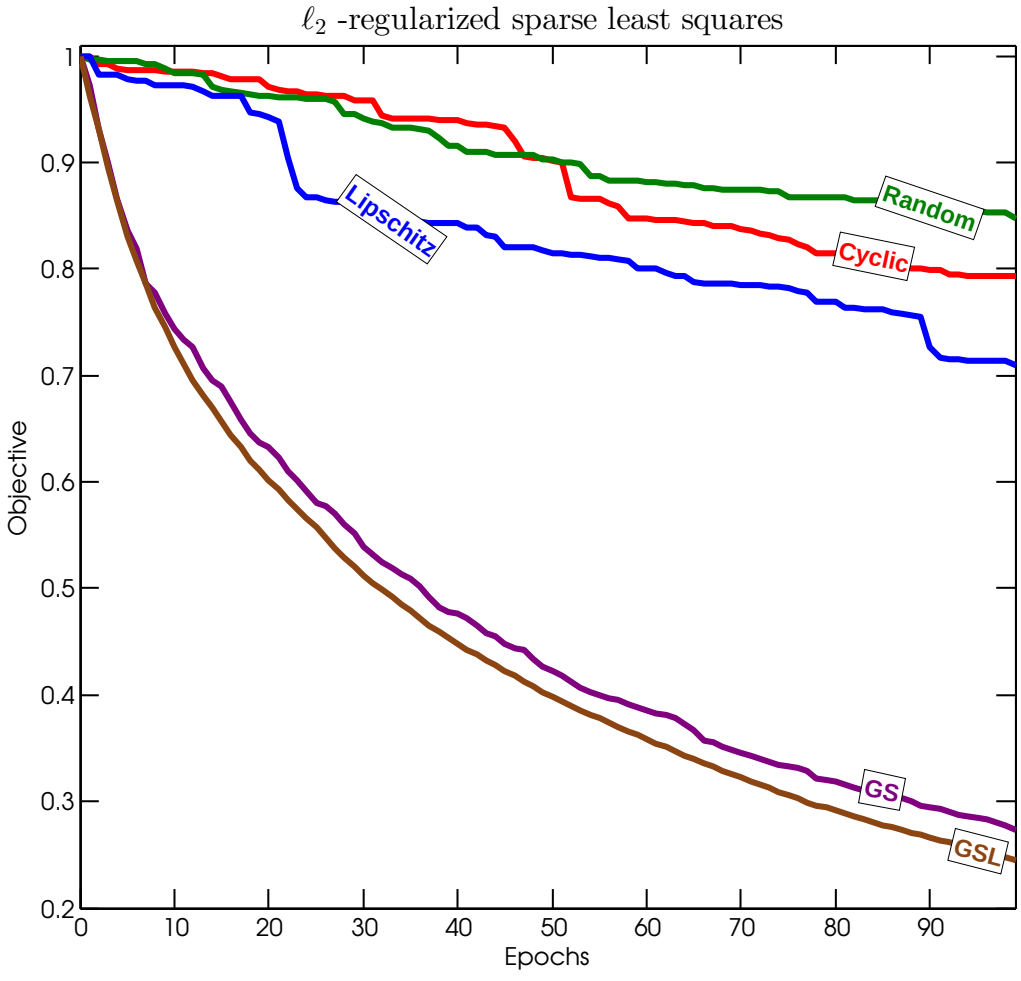
  - Gauss-Southwell (GS) is at least as expensive as random.
  - But Nesterov showed the rate is the same.
  - So greedy is a terrible algorithm and just use random!
- But this theory disagrees with practice...

Context: Random vs. Greedy Coordinate Descent



- If random and GS have similar costs, GS works much better.

Context: Random vs. Greedy Coordinate Descent



- If random and GS have similar costs, GS works much better.
- This work: refined analysis of GS.

Problems where we can apply coordinate descent

- When is coordinate update n times faster than gradient update?

Problems where we can apply coordinate descent

- When is coordinate update n times faster than gradient update?
- There are basically two problems where this is true:

$$h_1(x) = f(Ax) + \sum_{i=1}^{n} g_i(x_i), \quad \text{or} \quad h_2(x) = \sum_{(i,j)\in E} f_{ij}(x_i, x_j) + \sum_{i=1}^{n} g_i(x_i),$$

where f and f_{ij} are smooth, A is a matrix, E are edges in a graph. (g_i can be general non-degenerate convex functions)

Problems where we can apply coordinate descent

- When is coordinate update n times faster than gradient update?
- There are basically two problems where this is true:

$$h_1(x) = f(Ax) + \sum_{i=1}^{n} g_i(x_i), \quad \text{or} \quad h_2(x) = \sum_{(i,j)\in E} f_{ij}(x_i, x_j) + \sum_{i=1}^{n} g_i(x_i),$$

  where f and f_{ij} are smooth, A is a matrix, E are edges in a graph. (g_i can be general non-degenerate convex functions)
- h_1 includes least squares, logistic regression, Lasso, and SVMs.

$$\text{E.g., } \min_{x\in\mathbb{R}^n} \frac{1}{2}\|Ax - b\|^2 + \lambda \sum_{i=1}^{n} |x_i|.$$

Problems where we can apply coordinate descent

- When is coordinate update n times faster than gradient update?
- There are basically two problems where this is true:

$$h_1(x) = f(Ax) + \sum_{i=1}^{n} g_i(x_i), \quad \text{or} \quad h_2(x) = \sum_{(i,j)\in E} f_{ij}(x_i, x_j) + \sum_{i=1}^{n} g_i(x_i),$$

  where f and f_{ij} are smooth, A is a matrix, E are edges in a graph. (g_i can be general non-degenerate convex functions)
- h_1 includes least squares, logistic regression, Lasso, and SVMs.

$$\text{E.g., } \min_{x\in\mathbb{R}^n} \frac{1}{2}\|Ax - b\|^2 + \lambda \sum_{i=1}^{n} |x_i|.$$

- h_2 includes quadratics, graph-based label propagation, and probabilistic graphical models.

$$\text{E.g., } \min_{x\in\mathbb{R}^n} \frac{1}{2}x^T Ax + b^T x = \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij}x_i x_j + \sum_{i=1}^{n} b_i x_i.$$

Problems where can apply Gauss-Southwell

- GS rule may be as expensive as gradient even for h_1 and h_2.

Problems where can apply Gauss-Southwell

- GS rule may be as expensive as gradient even for h_1 and h_2.
  - But there are special cases where GS is $\approx n$ times faster.

Problems where can apply Gauss-Southwell

- GS rule may be as expensive as gradient even for h_1 and h_2.
  - But there are special cases where GS is $\approx n$ times faster.
- Problem h_2:
  - GS efficient if maximum degree is comparable to average degree.
  - You can track the gradients and use a max-heap.

Problems where can apply Gauss-Southwell

- GS rule may be as expensive as gradient even for h_1 and h_2.
  - But there are special cases where GS is $\approx n$ times faster.
- Problem h_2:
  - GS efficient if maximum degree is comparable to average degree.
  - You can track the gradients and use a max-heap.
- Examples:
  - Grid-based models, max degree $= 4$ and average degree ≈ 4.

    [Meshi et al., 2012]
  - Dense quadratic: max degree $= (n-1)$, average degree $= (n-1)$.
  - Facebook graph: max degree < 7000, average is ≈ 200.

Problems where can apply Gauss-Southwell

- GS rule may be as expensive as gradient even for h_1 and h_2.
  - But there are special cases where GS is $\approx n$ times faster.
- Problem h_2:
  - GS efficient if maximum degree is comparable to average degree.
  - You can track the gradients and use a max-heap.
- Examples:
  - Grid-based models, max degree $= 4$ and average degree ≈ 4.

    [Meshi et al., 2012]
  - Dense quadratic: max degree $= (n-1)$, average degree $= (n-1)$.
  - Facebook graph: max degree < 7000, average is ≈ 200.
- Problem h_1:
  - GS efficient if rows and columns of A have $O(\log(n))$ non-zeros.

    (iteration cost of $O((\log n)^3)$)

Problems where can apply Gauss-Southwell

- GS rule may be as expensive as gradient even for h_1 and h_2.
  - But there are special cases where GS is $\approx n$ times faster.
- Problem h_2:
  - GS efficient if maximum degree is comparable to average degree.
  - You can track the gradients and use a max-heap.
- Examples:
  - Grid-based models, max degree $= 4$ and average degree ≈ 4.

    [Meshi et al., 2012]
  - Dense quadratic: max degree $= (n-1)$, average degree $= (n-1)$.
  - Facebook graph: max degree < 7000, average is ≈ 200.
- Problem h_1:
  - GS efficient if rows and columns of A have $O(\log(n))$ non-zeros.

    (iteration cost of $O((\log n)^3)$)
  - GS can be approximated as nearest neighbour problem.

    [Dhillon et al., 2011, Shrivastava & Li, 2014].

Notation and Assumptions

- We focus on the convex optimization problem

$$\min_{x \in \mathbb{R}^n} \; f(x),$$

where ∇f is coordinate-wise L-Lipschitz continuous,

$$|\nabla_i f(x + \alpha e_i) - \nabla_i f(x)| \leq L|\alpha|.$$

Notation and Assumptions

- We focus on the convex optimization problem

$$\min_{x \in \mathbb{R}^n} \; f(x),$$

where ∇f is coordinate-wise L-Lipschitz continuous,

$$|\nabla_i f(x + \alpha e_i) - \nabla_i f(x)| \leq L|\alpha|.$$

- We focus on the case where f is μ-strongly convex, meaning that

$$x \mapsto f(x) - \frac{\mu}{2}\|x\|^2,$$

is a convex function for some $\mu > 0$.

Notation and Assumptions

- We focus on the convex optimization problem

$$\min_{x \in \mathbb{R}^n} \; f(x),$$

  where ∇f is coordinate-wise L-Lipschitz continuous,

$$|\nabla_i f(x + \alpha e_i) - \nabla_i f(x)| \leq L|\alpha|.$$

- We focus on the case where f is μ-strongly convex, meaning that

$$x \mapsto f(x) - \frac{\mu}{2}\|x\|^2,$$

  is a convex function for some $\mu > 0$.
- If twice-differentiable, equivalent to

$$\nabla^2_{ii} f(x) \leq L, \quad \nabla^2 f(x) \succeq \mu I.$$

Convergence of Randomized Coordinate Descent

- Coordinate descent with constant step size $\frac{1}{L}$ uses

$$x^{k+1} = x^k - \frac{1}{L}\nabla_{i_k} f(x^k) e_{i_k},$$

for some variable i_k.

Convergence of Randomized Coordinate Descent

- Coordinate descent with constant step size $\frac{1}{L}$ uses

$$x^{k+1} = x^k - \frac{1}{L}\nabla_{i_k} f(x^k) e_{i_k},$$

for some variable i_k.

- With i_k chosen uniformly from $\{1, 2, \ldots, n\}$ [Nesterov, 2010],

$$\mathbb{E}[f(x^{k+1})] - f(x^*) \leq \left(1 - \frac{\mu}{Ln}\right)[f(x^k) - f(x^*)].$$

Convergence of Randomized Coordinate Descent

- Coordinate descent with constant step size $\frac{1}{L}$ uses

$$x^{k+1} = x^k - \frac{1}{L}\nabla_{i_k} f(x^k) e_{i_k},$$

for some variable i_k.

- With i_k chosen uniformly from $\{1, 2, \ldots, n\}$ [Nesterov, 2010],

$$\mathbb{E}[f(x^{k+1})] - f(x^*) \leq \left(1 - \frac{\mu}{Ln}\right)[f(x^k) - f(x^*)].$$

- Compare this to the rate of gradient descent,

$$f(x^{k+1}) - f(x^*) \leq \left(1 - \frac{\mu}{L_f}\right)[f(x^k) - f(x^*)].$$

- Since $Ln \geq L_f \geq L$, coordinate descent is slower *per iteration*, but n coordinate iterations are faster than one gradient iteration.

Classic Analysis of Gauss-Southwell Rule

- GS rule chooses coordinate with largest directional derivative,

$$i_k = \underset{i}{\text{argmax}} \, |\nabla_i f(x^k)|.$$

Classic Analysis of Gauss-Southwell Rule

- GS rule chooses coordinate with largest directional derivative,

$$i_k = \underset{i}{\text{argmax}} \, |\nabla_i f(x^k)|.$$

- From Lipschitz-continuity assumption this rule satisfies

$$f(x^{k+1}) \leq f(x^k) - \frac{1}{2L} \|\nabla f(x^k)\|_\infty^2.$$

Classic Analysis of Gauss-Southwell Rule

- GS rule chooses coordinate with largest directional derivative,

$$i_k = \underset{i}{\operatorname{argmax}} |\nabla_i f(x^k)|.$$

- From Lipschitz-continuity assumption this rule satisfies

$$f(x^{k+1}) \leq f(x^k) - \frac{1}{2L}\|\nabla f(x^k)\|_\infty^2.$$

- From strong-convexity we have

$$f(x^*) \geq f(x^k) - \frac{1}{2\mu}\|\nabla f(x^k)\|^2.$$

Classic Analysis of Gauss-Southwell Rule

- GS rule chooses coordinate with largest directional derivative,

$$i_k = \underset{i}{\operatorname{argmax}} |\nabla_i f(x^k)|.$$

- From Lipschitz-continuity assumption this rule satisfies

$$f(x^{k+1}) \leq f(x^k) - \frac{1}{2L}\|\nabla f(x^k)\|_\infty^2.$$

- From strong-convexity we have

$$f(x^*) \geq f(x^k) - \frac{1}{2\mu}\|\nabla f(x^k)\|^2.$$

- Using $\|\nabla f(x^k)\|^2 \leq n\|\nabla f(x^k)\|_\infty^2$, we get

$$f(x^{k+1}) - f(x^*) \leq \left(1 - \frac{\mu}{Ln}\right)[f(x^k) - f(x^*)],$$

same rate as randomized [Boyd & Vandenberghe, 2004, §9.4.3].

Classic Analysis of Gauss-Southwell Rule

- GS rule chooses coordinate with largest directional derivative,

$$i_k = \underset{i}{\operatorname{argmax}} |\nabla_i f(x^k)|.$$

- From Lipschitz-continuity assumption this rule satisfies

$$f(x^{k+1}) \leq f(x^k) - \frac{1}{2L}\|\nabla f(x^k)\|_\infty^2.$$

- From strong-convexity we have

$$f(x^*) \geq f(x^k) - \frac{1}{2\mu}\|\nabla f(x^k)\|^2.$$

- Using $\|\nabla f(x^k)\|^2 \leq n\|\nabla f(x^k)\|_\infty^2$, we get

$$f(x^{k+1}) - f(x^*) \leq \left(1 - \frac{\mu}{Ln}\right)[f(x^k) - f(x^*)].$$

same rate as randomized [Boyd & Vandenberghe, 2004, §9.4.3].

Refined Gauss-Southwell Analysis

- To avoid norm inequality, measure strong-convexity in 1-norm.

Refined Gauss-Southwell Analysis

- To avoid norm inequality, measure strong-convexity in 1-norm.
- E.g., find the maximum μ_1 such that

$$x \mapsto f(x) - \frac{\mu_1}{2}\|x\|_1^2,$$

is convex.

Refined Gauss-Southwell Analysis

- To avoid norm inequality, measure strong-convexity in 1-norm.
- E.g., find the maximum μ_1 such that

$$x \mapsto f(x) - \frac{\mu_1}{2}\|x\|_1^2,$$

  is convex.
- We now have that

$$f(x^*) \geq f(x^k) - \frac{1}{2\mu_1}\|\nabla f(x^k)\|_\infty^2.$$

Refined Gauss-Southwell Analysis

- To avoid norm inequality, measure strong-convexity in 1-norm.
- E.g., find the maximum μ_1 such that

$$x \mapsto f(x) - \frac{\mu_1}{2}\|x\|_1^2,$$

  is convex.
- We now have that

$$f(x^*) \geq f(x^k) - \frac{1}{2\mu_1}\|\nabla f(x^k)\|_\infty^2.$$

- This gives a rate of

$$f(x^{k+1}) - f(x^*) \leq \left(1 - \frac{\mu_1}{L}\right)[f(x^k) - f(x^*)].$$

Refined Gauss-Southwell Analysis

- To avoid norm inequality, measure strong-convexity in 1-norm.
- E.g., find the maximum μ_1 such that

$$x \mapsto f(x) - \frac{\mu_1}{2}\|x\|_1^2,$$

  is convex.
- We now have that

$$f(x^*) \geq f(x^k) - \frac{1}{2\mu_1}\|\nabla f(x^k)\|_\infty^2.$$

- This gives a rate of

$$f(x^{k+1}) - f(x^*) \leq \left(1 - \frac{\mu_1}{L}\right)[f(x^k) - f(x^*)].$$

- The relationship between μ and μ_1 is given by

$$\frac{\mu}{n} \leq \mu_1 \leq \mu.$$

Refined Gauss-Southwell Analysis

- To avoid norm inequality, measure strong-convexity in 1-norm.
- E.g., find the maximum μ_1 such that

$$x \mapsto f(x) - \frac{\mu_1}{2}\|x\|_1^2,$$

  is convex.
- We now have that

$$f(x^*) \geq f(x^k) - \frac{1}{2\mu_1}\|\nabla f(x^k)\|_\infty^2.$$

- This gives a rate of

$$f(x^{k+1}) - f(x^*) \leq \left(1 - \frac{\mu_1}{L}\right)[f(x^k) - f(x^*)].$$

- The relationship between μ and μ_1 is given by

$$\frac{\mu}{n} \leq \mu_1 \leq \mu.$$

- GS bound is the same as random when $\mu_1 = \mu/n$.

Refined Gauss-Southwell Analysis

- To avoid norm inequality, measure strong-convexity in 1-norm.
- E.g., find the maximum μ_1 such that

$$x \mapsto f(x) - \frac{\mu_1}{2}\|x\|_1^2,$$

  is convex.
- We now have that

$$f(x^*) \geq f(x^k) - \frac{1}{2\mu_1}\|\nabla f(x^k)\|_\infty^2.$$

- This gives a rate of

$$f(x^{k+1}) - f(x^*) \leq \left(1 - \frac{\mu_1}{L}\right)[f(x^k) - f(x^*)].$$

- The relationship between μ and μ_1 is given by

$$\frac{\mu}{n} \leq \mu_1 \leq \mu.$$

- GS bound is the same as random when $\mu_1 = \mu/n$.
- Otherwise, GS can be faster by as large as n.

Comparison for Separable Quadratic

- In f is a quadratic with diagonal Hessian, we can show

$$\mu = \min_i \lambda_i, \quad \text{and} \quad \mu_1 = \frac{1}{\sum_{i=1}^{n} \frac{1}{\lambda_i}}.$$

Comparison for Separable Quadratic

- In f is a quadratic with diagonal Hessian, we can show

$$\mu = \min_i \lambda_i, \quad \text{and} \quad \mu_1 = \frac{1}{\sum_{i=1}^{n} \frac{1}{\lambda_i}}.$$

- If all λ_i equal:
  - There is no advantage to GS ($\mu_1 = \mu/n$).

Comparison for Separable Quadratic

- In f is a quadratic with diagonal Hessian, we can show

$$\mu = \min_i \lambda_i, \quad \text{and} \quad \mu_1 = \frac{1}{\sum_{i=1}^{n} \frac{1}{\lambda_i}}.$$

- If all λ_i equal:
  - There is no advantage to GS ($\mu_1 = \mu/n$).
- With one very large λ_i:
  - Here you would think that GS would be faster.

Comparison for Separable Quadratic

- In f is a quadratic with diagonal Hessian, we can show

$$\mu = \min_i \lambda_i, \quad \text{and} \quad \mu_1 = \frac{1}{\sum_{i=1}^{n} \frac{1}{\lambda_i}}.$$

- If all λ_i equal:
  - There is no advantage to GS ($\mu_1 = \mu/n$).
- With one very large λ_i:
  - Here you would think that GS would be faster.
  - But GS and random are still similar ($\mu_1 \approx \mu/n$).

Comparison for Separable Quadratic

- In f is a quadratic with diagonal Hessian, we can show

$$\mu = \min_i \lambda_i, \quad \text{and} \quad \mu_1 = \frac{1}{\sum_{i=1}^{n} \frac{1}{\lambda_i}}.$$

- If all λ_i equal:
  - There is no advantage to GS ($\mu_1 = \mu/n$).
- With one very large λ_i:
  - Here you would think that GS would be faster.
  - But GS and random are still similar ($\mu_1 \approx \mu/n$).
- With one very small λ_i:
  - Here GS bound can be better by a factor of n ($\mu_1 \approx \mu$).
  - In this case, GS can actually be faster than gradient descent.

Comparison for Separable Quadratic

- In f is a quadratic with diagonal Hessian, we can show

$$\mu = \min_i \lambda_i, \quad \text{and} \quad \mu_1 = \frac{1}{\sum_{i=1}^{n} \frac{1}{\lambda_i}}.$$

- If all λ_i equal:
  - There is no advantage to GS ($\mu_1 = \mu/n$).
- With one very large λ_i:
  - Here you would think that GS would be faster.
  - But GS and random are still similar ($\mu_1 \approx \mu/n$).
- With one very small λ_i:
  - Here GS bound can be better by a factor of n ($\mu_1 \approx \mu$).
  - In this case, GS can actually be faster than gradient descent.
- μ_1 is harmonic mean of λ_i divided by n, $H(\lambda)/n$:
  - $H(\lambda)$ is dominated by minimum of its arguments.
  - If each worker takes λ_i time to finish a task on their own, $H(\lambda)/n$ is time needed when 'working together' [Ferger, 1931].

Fast Convergence with Bias Term

- Consider the linear-prediction framework in statistics,

$$\underset{x,\beta}{\operatorname{argmin}} \sum_{i=1}^{n} f(a_i^T x + \beta) + \frac{\lambda}{2}\|x\|^2 + \frac{\sigma}{2}\beta^2,$$

  where we've included a bias β.

Fast Convergence with Bias Term

- Consider the linear-prediction framework in statistics,

$$\underset{x,\beta}{\operatorname{argmin}} \sum_{i=1}^{n} f(a_i^T x + \beta) + \frac{\lambda}{2}\|x\|^2 + \frac{\sigma}{2}\beta^2,$$

  where we've included a bias β.
- Typically $\sigma << \lambda$ to avoid biasing against a global shift.
- This is an instance of h_1 where GS has the most benefit.

Rates with Different Lipschitz Constants

- Consider the case where we have an L_i for each coordinate,

$$|\nabla_i f(x + \alpha e_i) - \nabla_i f(x)| \leq L_i |\alpha|,$$

and we use a coordinate-dependent stepsize,

$$x^{k+1} = x^k - \frac{1}{L_{i_k}} \nabla_{i_k} f(x^k) e_{i_k}.$$

Rates with Different Lipschitz Constants

- Consider the case where we have an L_i for each coordinate,

$$|\nabla_i f(x + \alpha e_i) - \nabla_i f(x)| \leq L_i |\alpha|,$$

and we use a coordinate-dependent stepsize,

$$x^{k+1} = x^k - \frac{1}{L_{i_k}} \nabla_{i_k} f(x^k) e_{i_k}.$$

- In this setting, we get a rate of

$$f(x^k) - f(x^*) \leq \left[\prod_{j=1}^{k} \left(1 - \frac{\mu_1}{L_{i_j}} \right) \right] [f(x^0) - f(x^*)].$$

Rates with Different Lipschitz Constants

- Consider the case where we have an L_i for each coordinate,

$$|\nabla_i f(x + \alpha e_i) - \nabla_i f(x)| \leq L_i|\alpha|,$$

  and we use a coordinate-dependent stepsize,

$$x^{k+1} = x^k - \frac{1}{L_{i_k}} \nabla_{i_k} f(x^k) e_{i_k}.$$

- In this setting, we get a rate of

$$f(x^k) - f(x^*) \leq \left[\prod_{j=1}^{k} \left(1 - \frac{\mu_1}{L_{i_j}} \right) \right] [f(x^0) - f(x^*)].$$

- Since $L = \max_i L_i$, this is faster if $L_{i_k} < L$ for any i_k.
- But rate is the same in the worst case, even if L_i are distinct.
- Let's consider the effect of exact coordinate optimization on L_{i_k}.

Gauss-Southwell with Exact Optimization

- Exact coordinate optimization chooses the stepsize minimizing f.
- We can get the same rates for randomized/GS because

$$\begin{aligned} f(x^{k+1}) &= \min_{\alpha}\{f(x^k - \alpha\nabla_{i_k} f(x^k)e_{i_k})\} \\ &\leq f\left(x^k - \frac{1}{L_{i_k}}\nabla_{i_i} f(x^k)e_{i_k}\right) \\ &\leq f(x^k) - \frac{1}{2L_{i_k}}[\nabla_{i_k} f(x^k)]^2, \end{aligned}$$

Gauss-Southwell with Exact Optimization

- Exact coordinate optimization chooses the stepsize minimizing f.
- We can get the same rates for randomized/GS because

$$\begin{aligned} f(x^{k+1}) &= \min_{\alpha}\{f(x^k - \alpha\nabla_{i_k} f(x^k)e_{i_k})\} \\ &\leq f\left(x^k - \frac{1}{L_{i_k}}\nabla_{i_i} f(x^k)e_{i_k}\right) \\ &\leq f(x^k) - \frac{1}{2L_{i_k}}[\nabla_{i_k} f(x^k)]^2, \end{aligned}$$

- But theory again disagrees with practice:
  - Empirically, exact optimization is much faster.

Gauss-Southwell with Exact Optimization and Sparsity

- For dense problems, exact optimization bound is a little better:
  - After an exact update, we have $\nabla_{i_k} f(x^{k+1}) = 0$.

Gauss-Southwell with Exact Optimization and Sparsity

- For dense problems, exact optimization bound is a little better:
  - After an exact update, we have $\nabla_{i_k} f(x^{k+1}) = 0$.
  - Since $i_{k+1} = \text{argmax}_i |\nabla_i f(x^{k+1})|$, we never have $i_{k+1} = i_k$.

Gauss-Southwell with Exact Optimization and Sparsity

- For dense problems, exact optimization bound is a little better:
  - After an exact update, we have $\nabla_{i_k} f(x^{k+1}) = 0$.
  - Since $i_{k+1} = \text{argmax}_i |\nabla_i f(x^{k+1})|$, we never have $i_{k+1} = i_k$.

$$\nabla f(x^k) = \begin{bmatrix} 0.67 \\ -1.21 \\ 0.72 \\ \mathbf{1.63} \\ 0.49 \end{bmatrix},$$

Gauss-Southwell with Exact Optimization and Sparsity

- For dense problems, exact optimization bound is a little better:
  - After an exact update, we have $\nabla_{i_k} f(x^{k+1}) = 0$.
  - Since $i_{k+1} = \text{argmax}_i |\nabla_i f(x^{k+1})|$, we never have $i_{k+1} = i_k$.

$$\nabla f(x^k) = \begin{bmatrix} 0.67 \\ -1.21 \\ 0.72 \\ \mathbf{1.63} \\ 0.49 \end{bmatrix}, \qquad \nabla f(x^{k+1}) = \begin{bmatrix} 0.65 \\ \mathbf{-1.31} \\ 0.81 \\ 0 \\ 0.53 \end{bmatrix}.$$

Gauss-Southwell with Exact Optimization and Sparsity

- For dense problems, exact optimization bound is a little better:
  - After an exact update, we have $\nabla_{i_k} f(x^{k+1}) = 0$.
  - Since $i_{k+1} = \text{argmax}_i |\nabla_i f(x^{k+1})|$, we never have $i_{k+1} = i_k$.

$$\nabla f(x^k) = \begin{bmatrix} 0.67 \\ -1.21 \\ 0.72 \\ \mathbf{1.63} \\ 0.49 \end{bmatrix}, \qquad \nabla f(x^{k+1}) = \begin{bmatrix} 0.65 \\ \mathbf{-1.31} \\ 0.81 \\ 0 \\ 0.53 \end{bmatrix}.$$

  - If L_i are distinct, worst case is alternating between largest two L_i.

Gauss-Southwell with Exact Optimization and Sparsity

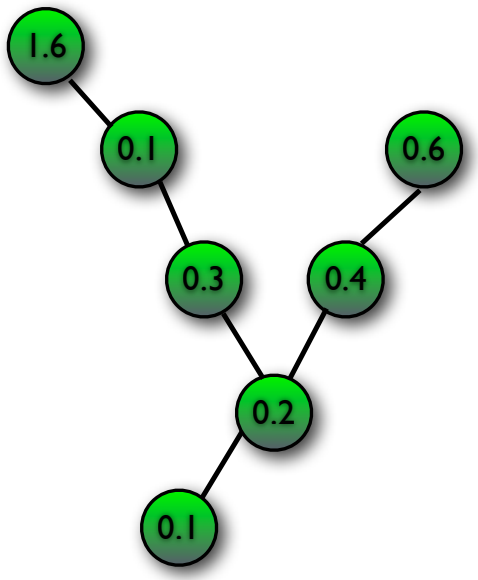
- For sparse instances of h_2, exact can be much better:

Gauss-Southwell with Exact Optimization and Sparsity

- For sparse instances of h_2, exact can be much better:
  - After an exact update we have $\nabla_{i_k} f(x^{k+m}) = 0$,
    for all m until i_{k+m-1} is a neighbour of node i_k in the graph.
  - We never alternate between large L_i that aren't neighbours.

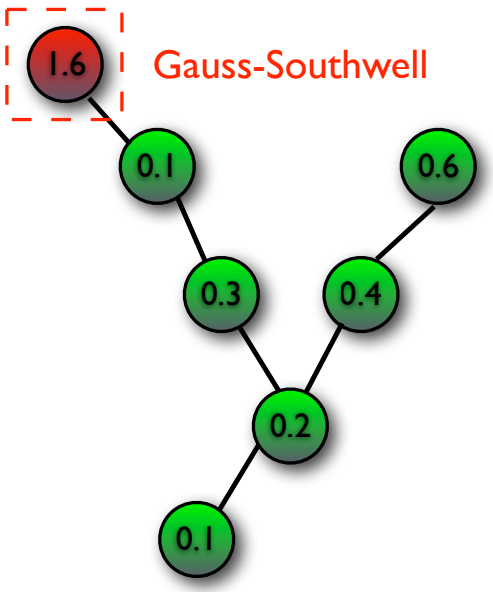
Gauss-Southwell with Exact Optimization and Sparsity

- For sparse instances of h_2, exact can be much better:
  - After an exact update we have $\nabla_{i_k} f(x^{k+m}) = 0$,
    for all m until i_{k+m-1} is a neighbour of node i_k in the graph.
  - We never alternate between large L_i that aren't neighbours.



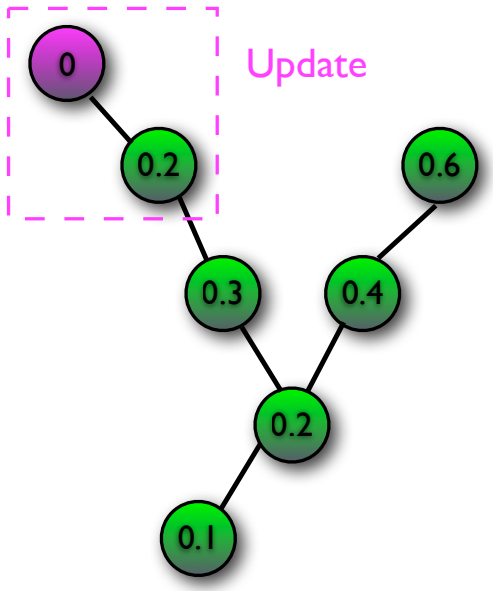
Gauss-Southwell with Exact Optimization and Sparsity

- For sparse instances of h_2, exact can be much better:
  - After an exact update we have $\nabla_{i_k} f(x^{k+m}) = 0$,
    for all m until i_{k+m-1} is a neighbour of node i_k in the graph.
  - We never alternate between large L_i that aren't neighbours.



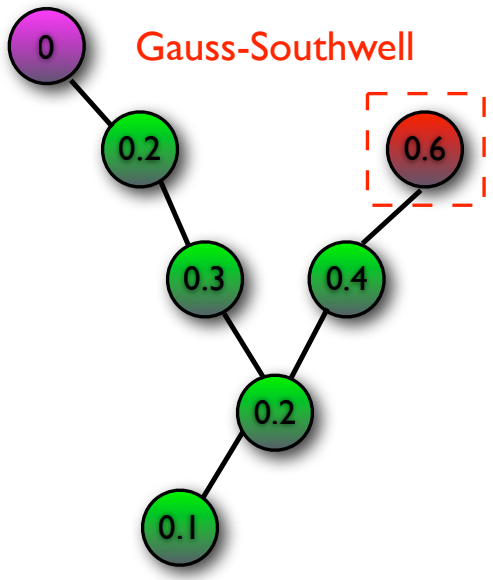
Gauss-Southwell with Exact Optimization and Sparsity

- For sparse instances of h_2, exact can be much better:
  - After an exact update we have $\nabla_{i_k} f(x^{k+m}) = 0$,
    for all m until i_{k+m-1} is a neighbour of node i_k in the graph.
  - We never alternate between large L_i that aren't neighbours.



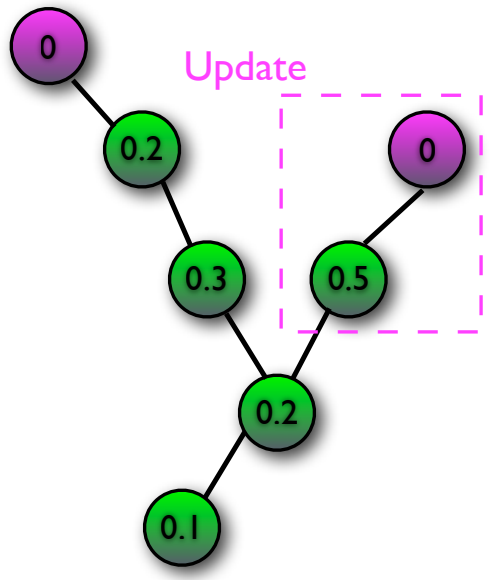
Gauss-Southwell with Exact Optimization and Sparsity

- For sparse instances of h_2, exact can be much better:
  - After an exact update we have $\nabla_{i_k} f(x^{k+m}) = 0$,
    for all m until i_{k+m-1} is a neighbour of node i_k in the graph.
  - We never alternate between large L_i that aren't neighbours.



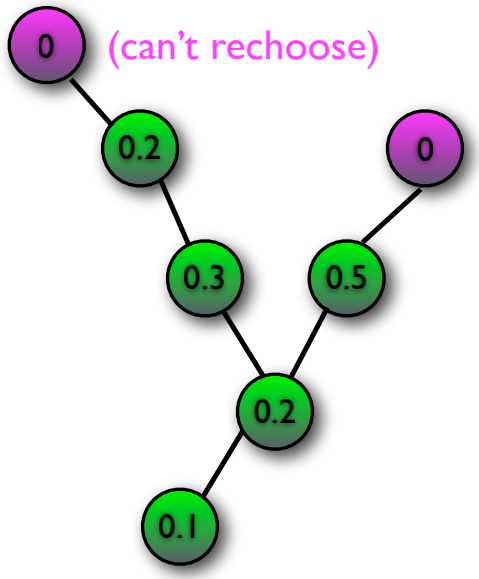
Gauss-Southwell with Exact Optimization and Sparsity

- For sparse instances of h_2, exact can be much better:
  - After an exact update we have $\nabla_{i_k} f(x^{k+m}) = 0$,
    for all m until i_{k+m-1} is a neighbour of node i_k in the graph.
  - We never alternate between large L_i that aren't neighbours.



Gauss-Southwell with Exact Optimization and Sparsity

- For sparse instances of h_2, exact can be much better:
  - After an exact update we have $\nabla_{i_k} f(x^{k+m}) = 0$,
    for all m until i_{k+m-1} is a neighbour of node i_k in the graph.
  - We never alternate between large L_i that aren't neighbours.



Gauss-Southwell with Exact Optimization and Sparsity

- For sparse instances of h_2, exact can be much better:
  - By bounding the worst-case sequence of L_i values, we have

$$f(x^k) - f(x^*) = O\left(\left(1 - \frac{\mu_1}{\max\{L_2^G, L_3^G\}}\right)^k\right)[f(x^0) - f(x^*)].$$

  - L_2^G is the largest average between neighbours.
  - L_3^G is the largest average 3-node path.

Gauss-Southwell with Exact Optimization and Sparsity

- For sparse instances of h_2, exact can be much better:
  - By bounding the worst-case sequence of L_i values, we have

$$f(x^k) - f(x^*) = O\left(\left(1 - \frac{\mu_1}{\max\{L_2^G, L_3^G\}}\right)^k\right)[f(x^0) - f(x^*)].$$

  - L_2^G is the largest average between neighbours.
  - L_3^G is the largest average 3-node path.
  - This is much faster if the large L_i are not neighbours.
- Similar for h_1: edges between variables non-zero in same row.

Rules Depending on Lipschitz Constants

- Assume that we know the L_i or approximate them.

Rules Depending on Lipschitz Constants

- Assume that we know the L_i or approximate them.
- Nesterov [2010] shows that sampling proportional to L_i yields

$$\mathbb{E}[f(x^{k+1})] - f(x^*) \leq \left(1 - \frac{\mu}{n\bar{L}}\right)[f(x^k) - f(x^*)],$$

  where $\bar{L} = \frac{1}{n}\sum_{i=1}^{n} L_i$.
- Faster than uniform sampling when the L_i are disinct.

Rules Depending on Lipschitz Constants

- Assume that we know the L_i or approximate them.
- Nesterov [2010] shows that sampling proportional to L_i yields

$$\mathbb{E}[f(x^{k+1})] - f(x^*) \leq \left(1 - \frac{\mu}{n\bar{L}}\right)[f(x^k) - f(x^*)],$$

  where $\bar{L} = \frac{1}{n}\sum_{i=1}^{n} L_i$.
- Faster than uniform sampling when the L_i are disinct.
- This could be faster or slower than the GS rule.
- In the separalbe quadratic case:
  - With one large λ_i, Lipschitz sampling is faster.
  - With one small λ_i, GS is faster.
- So which should we use?

Gauss-Southwell-Lipschitz Rule

- We obtain a faster rate by using L_i in the GS rule,

$$i_k = \underset{i}{\text{argmax}} \frac{|\nabla_i f(x^k)|}{\sqrt{L_i}},$$

  which we call the Gauss-Southwell-Lipschitz (GSL) rule.

- Intuition: if gradients are similar, more progress if L_i is small.

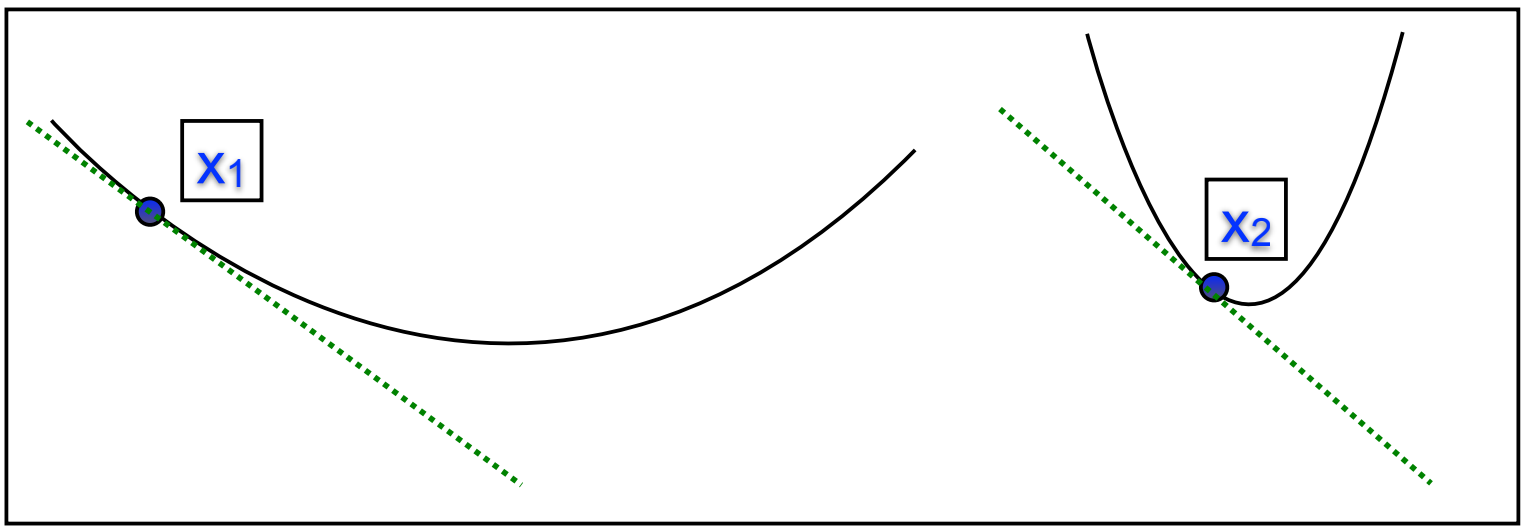
Gauss-Southwell-Lipschitz Rule

- We obtain a faster rate by using L_i in the GS rule,

$$i_k = \underset{i}{\text{argmax}} \frac{|\nabla_i f(x^k)|}{\sqrt{L_i}},$$

  which we call the Gauss-Southwell-Lipschitz (GSL) rule.

- Intuition: if gradients are similar, more progress if L_i is small.



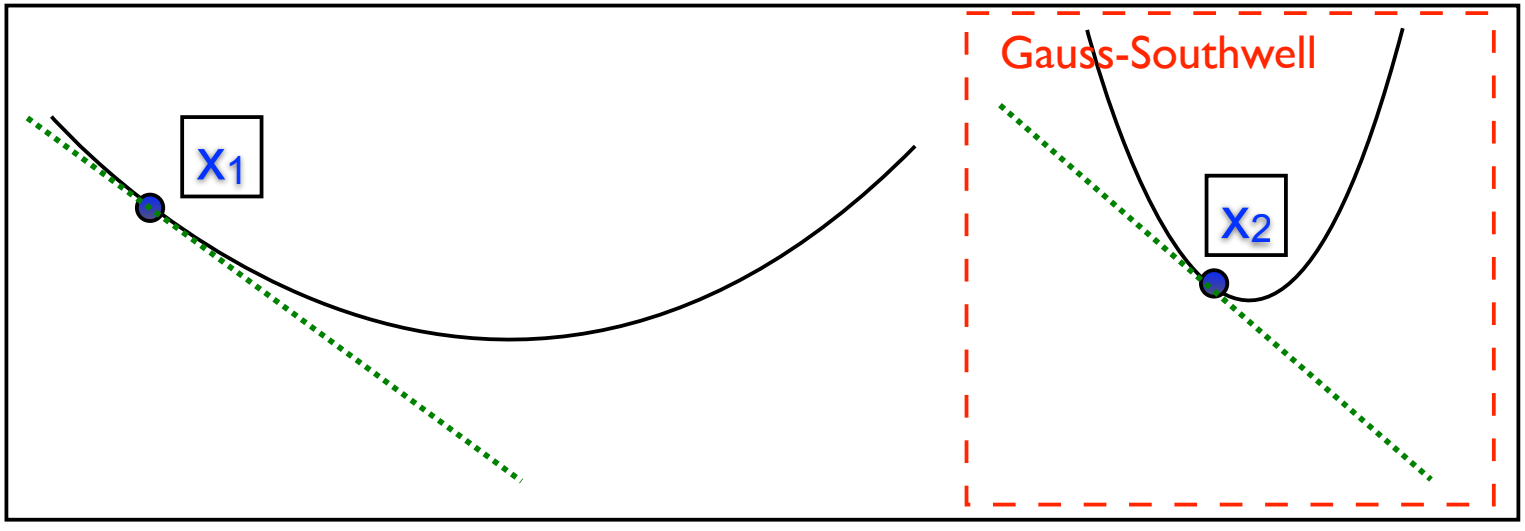
Gauss-Southwell-Lipschitz Rule

- We obtain a faster rate by using L_i in the GS rule,

$$i_k = \underset{i}{\text{argmax}} \frac{|\nabla_i f(x^k)|}{\sqrt{L_i}},$$

  which we call the Gauss-Southwell-Lipschitz (GSL) rule.

- Intuition: if gradients are similar, more progress if L_i is small.



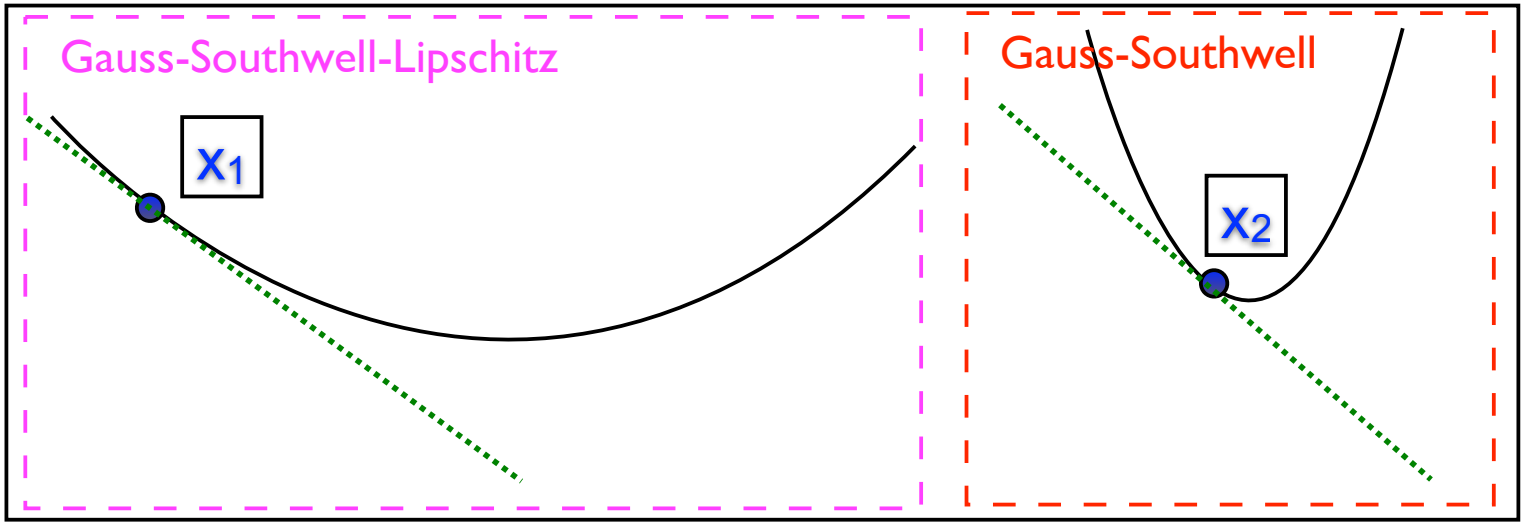
Gauss-Southwell-Lipschitz Rule

- We obtain a faster rate by using L_i in the GS rule,

$$i_k = \underset{i}{\text{argmax}} \frac{|\nabla_i f(x^k)|}{\sqrt{L_i}},$$

  which we call the Gauss-Southwell-Lipschitz (GSL) rule.

- Intuition: if gradients are similar, more progress if L_i is small.



Gauss-Southwell-Lipschitz Rule

- The GSL rule obtains a rate of

$$f(x^{k+1}) - f(x^k) \leq (1 - \mu_L)[f(x^k) - f(x^*)].$$

where μ_L is strong-convexity constant in $\|x\|_L = \sum_{i=1}^{n} \sqrt{L_i}|x_i|$.

Gauss-Southwell-Lipschitz Rule

- The GSL rule obtains a rate of

$$f(x^{k+1}) - f(x^k) \leq (1 - \mu_L)[f(x^k) - f(x^*)].$$

  where μ_L is strong-convexity constant in $\|x\|_L = \sum_{i=1}^n \sqrt{L_i}|x_i|$.

- We have that

$$\max\left\{\frac{\mu}{n\bar{L}}, \frac{\mu_1}{L}\right\} \leq \mu_L \leq \frac{\mu_1}{\min_i\{L_i\}},$$

  so GSL is at least as fast as GS and Lipschitz sampling.

Gauss-Southwell-Lipschitz Rule

- The GSL rule obtains a rate of

$$f(x^{k+1}) - f(x^k) \leq (1 - \mu_L)[f(x^k) - f(x^*)].$$

  where μ_L is strong-convexity constant in $\|x\|_L = \sum_{i=1}^n \sqrt{L_i}|x_i|$.

- We have that

$$\max\left\{\frac{\mu}{n\bar{L}}, \frac{\mu_1}{L}\right\} \leq \mu_L \leq \frac{\mu_1}{\min_i\{L_i\}},$$

  so GSL is at least as fast as GS and Lipschitz sampling.

- GSL using $\frac{1}{L_{i_k}}$ is optimal myopic coordinate update for quadratics,

$$f(x^{k+1}) = \operatorname*{argmin}_{i,\alpha}\{f(x^k + \alpha e_i)\}.$$

Gauss-Southwell-Lipschitz Rule

- The GSL rule obtains a rate of

$$f(x^{k+1}) - f(x^k) \leq (1 - \mu_L)[f(x^k) - f(x^*)].$$

  where μ_L is strong-convexity constant in $\|x\|_L = \sum_{i=1}^n \sqrt{L_i}|x_i|$.

- We have that

$$\max\left\{\frac{\mu}{n\bar{L}}, \frac{\mu_1}{L}\right\} \leq \mu_L \leq \frac{\mu_1}{\min_i\{L_i\}},$$

  so GSL is at least as fast as GS and Lipschitz sampling.

- GSL using $\frac{1}{L_{i_k}}$ is optimal myopic coordinate update for quadratics,

$$f(x^{k+1}) = \operatorname*{argmin}_{i,\alpha}\{f(x^k + \alpha e_i)\}.$$

- Analysis gives tighter bound on maximum improvement rule, used in certain applications.

[Della Pietra et al., 1997, Lee et al., 2006]

Gauss-Southwell-Lipschitz as Nearest Neighbour

- Consider a special case of h_1,

$$\min_x \; h_1(x) = \sum_{i=1}^{n} f(a_i^T x),$$

where GS rule has the form

$$i_k = \operatorname*{argmax}_i |a_i^T r(x^k)|.$$

Gauss-Southwell-Lipschitz as Nearest Neighbour

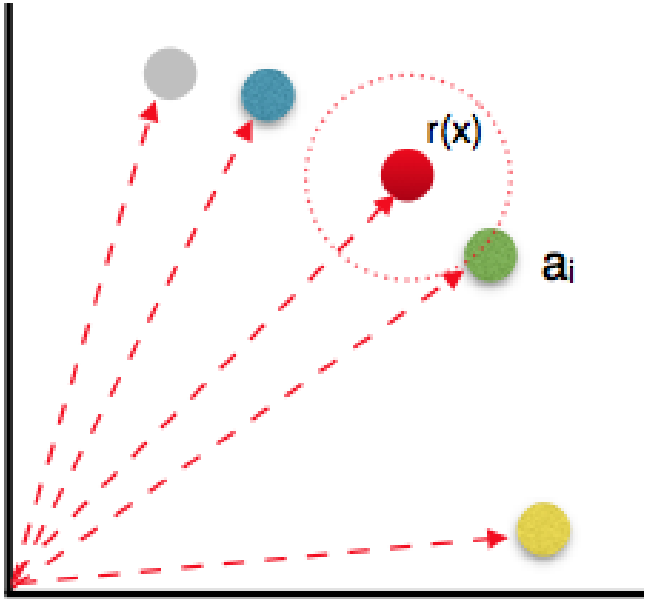
- Consider a special case of h_1,

$$\min_x \; h_1(x) = \sum_{i=1}^{n} f(a_i^T x),$$

where GS rule has the form

$$i_k = \operatorname*{argmax}_i |a_i^T r(x^k)|.$$

- Dhillon et al. [2011] approximate GS as nearest neighbour,



Gauss-Southwell-Lipschitz as Nearest Neighbour

- Consider a special case of h_1,

$$\min_x \; h_1(x) = \sum_{i=1}^{n} f(a_i^T x),$$

where GS rule has the form

$$i_k = \operatorname*{argmax}_i |a_i^T r(x^k)|.$$

- Dhillon et al. [2011] approximate GS as nearest neighbour,

$$\operatorname*{argmin}_i \frac{1}{2}\|r(x^k) - a_i\|^2 = \frac{1}{2}\|r(x^k)\|^2 - a_i^T r(x^k) + \frac{1}{2}\|a_i\|^2.$$

(use a_i and $-a_i$ to get absolute value)

- Approximation is exact if $\|a_i\| = 1$ for all i.

Gauss-Southwell-Lipschitz as Nearest Neighbour

- Consider a special case of h_1,

$$\min_x \; h_1(x) = \sum_{i=1}^{n} f(a_i^T x),$$

where GS rule has the form

$$i_k = \operatorname*{argmax}_i |a_i^T r(x^k)|.$$

- Dhillon et al. [2011] approximate GS as nearest neighbour,

$$\operatorname*{argmin}_i \frac{1}{2}\|r(x^k) - a_i\|^2 = \frac{1}{2}\|r(x^k)\|^2 - a_i^T r(x^k) + \frac{1}{2}\|a_i\|^2.$$

(use a_i and $-a_i$ to get absolute value)

- Approximation is exact if $\|a_i\| = 1$ for all i.
- Using $L_i = \gamma\|a_i\|^2$, exact GSL as a nearest neighbour problem,

$$\operatorname*{argmin}_i \frac{1}{2}\left\|r(x^k) - \frac{a_i}{\sqrt{\gamma}\|a_i\|}\right\|^2 = \frac{1}{2}\|r(x^k)\|^2 - \frac{a_i^T r(x^k)}{\sqrt{\gamma}\|a_i\|} + \frac{1}{2\gamma}.$$

Approximate Gauss-Southwell

- In many applications, can approximate GS rule.

Approximate Gauss-Southwell

- In many applications, can approximate GS rule.
- With multiplicative errror,

$$|\nabla_{i_k} f(x^k)| \geq \|\nabla f(x^k)\|_\infty (1 - \epsilon_k),$$

we have a fast rate and do not need $\epsilon_k \to 0$,

$$f(x^{k+1}) - f(x^*) \leq \left(1 - \frac{\mu_1 (1 - \epsilon_k)^2}{L}\right) [f(x^k) - f(x^*)].$$

Approximate Gauss-Southwell

- In many applications, can approximate GS rule.
- With multiplicative errror,

$$|\nabla_{i_k} f(x^k)| \geq \|\nabla f(x^k)\|_\infty (1 - \epsilon_k),$$

  we have a fast rate and do not need $\epsilon_k \to 0$,

$$f(x^{k+1}) - f(x^*) \leq \left(1 - \frac{\mu_1 (1 - \epsilon_k)^2}{L}\right) [f(x^k) - f(x^*)].$$

- With additive error,

$$|\nabla_{i_k} f(x^k)| \geq \|\nabla f(x^k)\|_\infty - \epsilon_k,$$

  we have a fast rate if $\epsilon_k \to 0$ fast enough.
- With constant additive error, only get a certain solution accuracy.

Proximal Coordinate Descent

- Important application of coordinate descent is for problems

$$\min_{x \in \mathbb{R}^n} F(x) \equiv f(x) + \sum_i g_i(x_i),$$

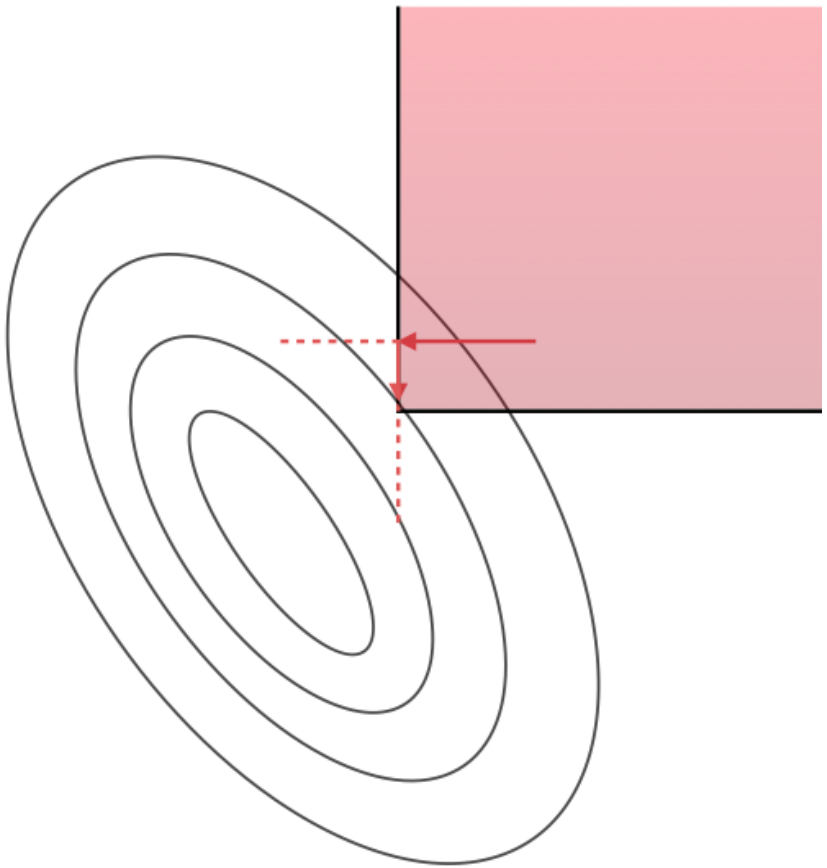
  where f is smooth and g_i might be non-smooth.
- E.g., ℓ_1-regularization or bound constraints.

Proximal Coordinate Descent

- Important application of coordinate descent is for problems

$$\min_{x \in \mathbb{R}^n} F(x) \equiv f(x) + \sum_i g_i(x_i),$$

  where f is smooth and g_i might be non-smooth.
- E.g., ℓ_1-regularization or bound constraints.



Proximal Coordinate Descent

- Important application of coordinate descent is for problems

$$\min_{x \in \mathbb{R}^n} F(x) \equiv f(x) + \sum_i g_i(x_i),$$

  where f is smooth and g_i might be non-smooth.
- E.g., ℓ_1-regularization or bound constraints.
- Here we can apply proximal-gradient style of update,

$$x^{k+1} = \text{prox}_{\frac{1}{L} g_{i_k}} \left[x^k - \frac{1}{L} \nabla_{i_k} f(x^k) e_{i_k} \right],$$

  where

$$\text{prox}_{\alpha g}[y] = \underset{x \in \mathbb{R}^n}{\text{argmin}} \frac{1}{2} \|x - y\|^2 + \alpha g(x).$$

Proximal Coordinate Descent

- Important application of coordinate descent is for problems

$$\min_{x \in \mathbb{R}^n} F(x) \equiv f(x) + \sum_i g_i(x_i),$$

  where f is smooth and g_i might be non-smooth.

- E.g., ℓ_1-regularization or bound constraints.
- Here we can apply proximal-gradient style of update,

$$x^{k+1} = \text{prox}_{\frac{1}{L} g_{i_k}} \left[x^k - \frac{1}{L} \nabla_{i_k} f(x^k) e_{i_k} \right],$$

  where

$$\text{prox}_{\alpha g}[y] = \underset{x \in \mathbb{R}^n}{\text{argmin}} \frac{1}{2} \|x - y\|^2 + \alpha g(x).$$

- Richtárik and Takac [2014] show that

$$\mathbb{E}[F(x^{k+1}) - F(x^k)] \leq \left(1 - \frac{\mu}{Ln}\right) [F(x^k) - F(x^*)],$$

  the same rate as if non-smooth g_i was not there.

Proximal Gauss-Southwell

There are several generalizations of GS to this setting:

- GS-s: Minimize directional derivative,

$$i_k = \underset{i}{\text{argmax}} \left\{ \min_{s \in \partial g_i} |\nabla_i f(x^k) + s| \right\}.$$

Proximal Gauss-Southwell

There are several generalizations of GS to this setting:

- GS-s: Minimize directional derivative,

$$i_k = \underset{i}{\text{argmax}} \left\{ \min_{s \in \partial g_i} |\nabla_i f(x^k) + s| \right\}.$$

  - Commonly-used for ℓ_1-regularization, but $\|x^{k+1} - x^k\|$ could be tiny.

Proximal Gauss-Southwell

There are several generalizations of GS to this setting:

- GS-s: Minimize directional derivative,

$$i_k = \underset{i}{\text{argmax}} \left\{ \min_{s \in \partial g_i} |\nabla_i f(x^k) + s| \right\}.$$

  - Commonly-used for ℓ_1-regularization, but $\|x^{k+1} - x^k\|$ could be tiny.

- GS-r: Maximize how far we move,

$$i_k = \underset{i}{\text{argmax}} \left\{ \left| x_i^k - \text{prox}_{\frac{1}{L} g_{i_k}} \left[x_i^k - \frac{1}{L} \nabla_{i_k} f(x^k) \right] \right| \right\}.$$

  - Effective for bound constraints, but ignores $g_i(x_i^{k+1}) - g_i(x_i^k)$.

Proximal Gauss-Southwell

There are several generalizations of GS to this setting:

- GS-s: Minimize directional derivative,

$$i_k = \underset{i}{\operatorname{argmax}} \left\{ \min_{s \in \partial g_i} |\nabla_i f(x^k) + s| \right\}.$$

  - Commonly-used for ℓ_1-regularization, but $\|x^{k+1} - x^k\|$ could be tiny.

- GS-r: Maximize how far we move,

$$i_k = \underset{i}{\operatorname{argmax}} \left\{ \left| x_i^k - \operatorname{prox}_{\frac{1}{L} g_{i_k}} \left[x_i^k - \frac{1}{L} \nabla_{i_k} f(x^k) \right] \right| \right\}.$$

  - Effective for bound constraints, but ignores $g_i(x_i^{k+1}) - g_i(x_i^k)$.

- GS-q: Maximize progress under quadratic approximation of f.

$$i_k = \underset{i}{\operatorname{argmin}} \left\{ \min_d f(x^k) + \nabla_i f(x^k) d + \frac{L d^2}{2} + g_i(x_i^k + d) - g_i(x_i^k) \right\}.$$

  - Least intuitive, but has the best theoretical properties.
  - Generalizes GSL if you use L_i instead of L.

Proximal Gauss-Southwell Convergence Rate

- For the GS-q rule, we show that

$$f(x^{k+1}) - f(x^k) \leq \min\left\{ \left(1 - \frac{\mu}{Ln}\right)[f(x^k) - f(x^*)], \left(1 - \frac{\mu_1}{L}\right)[f(x^0) - f(x^*)] + \epsilon_k \right\},$$

  where $\epsilon^k \to 0$ measures non-linearity of g_i that are not updated.

- We conjecture that the above always holds with $\epsilon_k = 0$.

Proximal Gauss-Southwell Convergence Rate

- For the GS-q rule, we show that

$$f(x^{k+1}) - f(x^k) \leq \min\left\{ \left(1 - \frac{\mu}{Ln}\right)[f(x^k) - f(x^*)], \left(1 - \frac{\mu_1}{L}\right)[f(x^0) - f(x^*)] + \epsilon_k \right\},$$

  where $\epsilon^k \to 0$ measures non-linearity of g_i that are not updated.

- We conjecture that the above always holds with $\epsilon_k = 0$.

- The above rate does not hold for GS-s or GS-r.
  (even if you change min to max)

Proximal Gauss-Southwell Convergence Rate

- For the GS-q rule, we show that

$$f(x^{k+1}) - f(x^k) \leq \min\left\{ \left(1 - \frac{\mu}{Ln}\right)[f(x^k) - f(x^*)], \left(1 - \frac{\mu_1}{L}\right)[f(x^0) - f(x^*)] + \epsilon_k \right\},$$

  where $\epsilon^k \to 0$ measures non-linearity of g_i that are not updated.

- We conjecture that the above always holds with $\epsilon_k = 0$.
- The above rate does not hold for GS-s or GS-r.
  (even if you change min to max)
- But one final time theory disagrees with practice:

Proximal Gauss-Southwell Convergence Rate

- For the GS-q rule, we show that

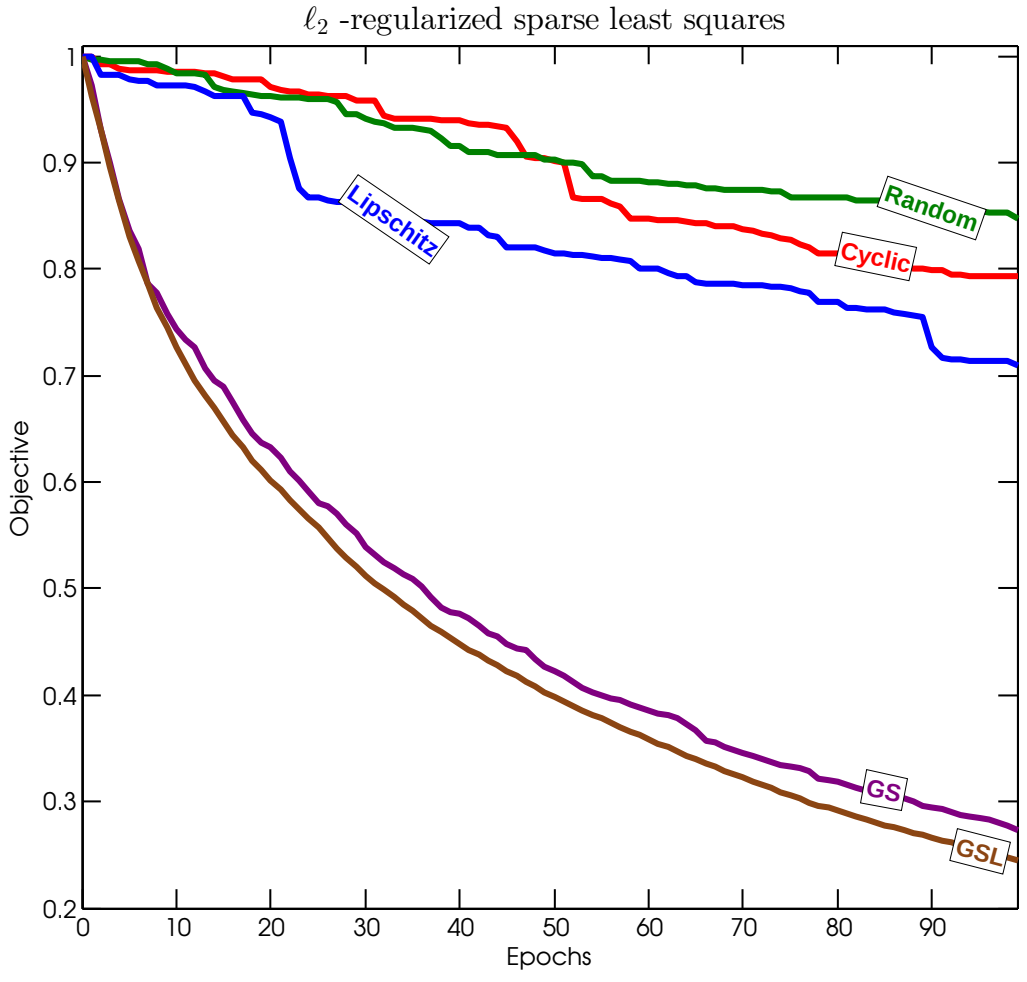
$$f(x^{k+1}) - f(x^k) \leq \min\left\{ \left(1 - \frac{\mu}{Ln}\right)[f(x^k) - f(x^*)], \left(1 - \frac{\mu_1}{L}\right)[f(x^0) - f(x^*)] + \epsilon_k \right\},$$

  where $\epsilon^k \to 0$ measures non-linearity of g_i that are not updated.

- We conjecture that the above always holds with $\epsilon_k = 0$.
- The above rate does not hold for GS-s or GS-r.
  (even if you change min to max)
- But one final time theory disagrees with practice:
  - All three rules seem to work pretty well.
  - Though GS-r works badly if you use the L_i.

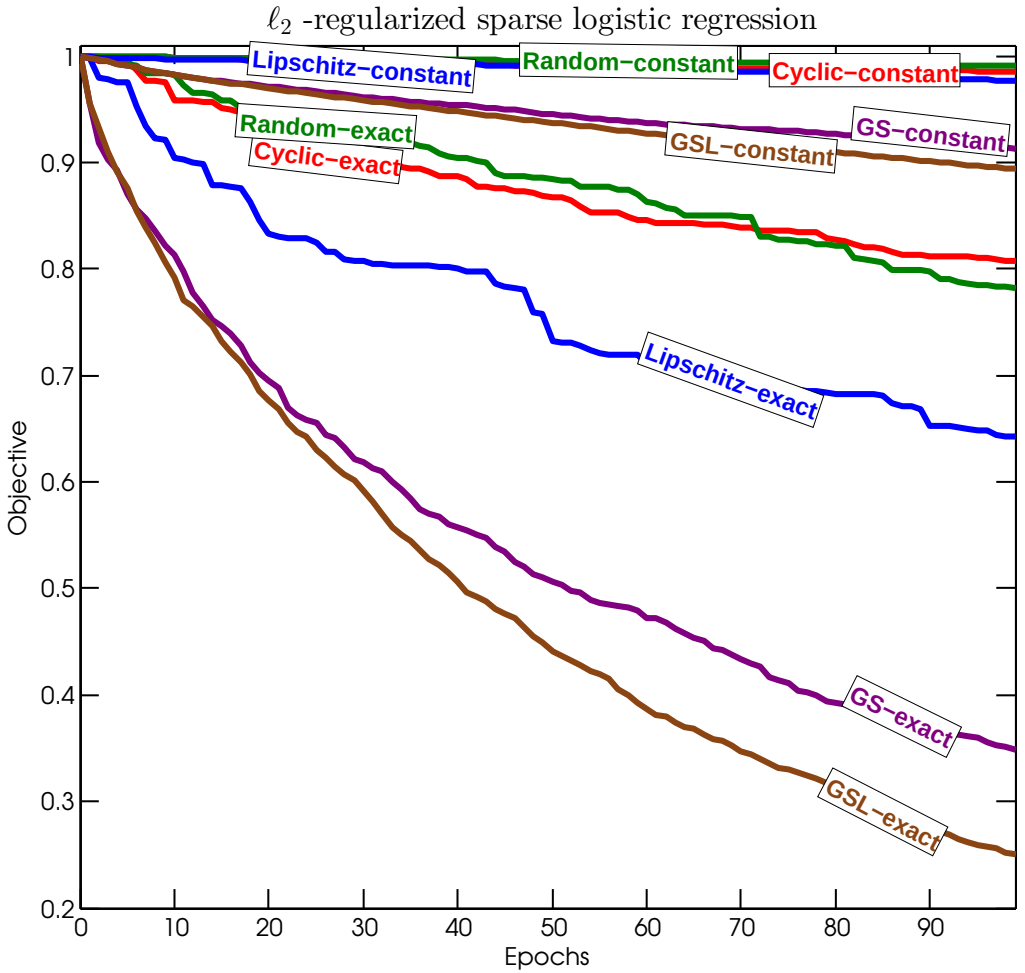
Experiment 1: Sparse ℓ_2-Regularized Least Squares

Least squares with ℓ_2-regularization and very sparse matrix.



Experiment 2: Sparse ℓ_2-Regularized Logistic

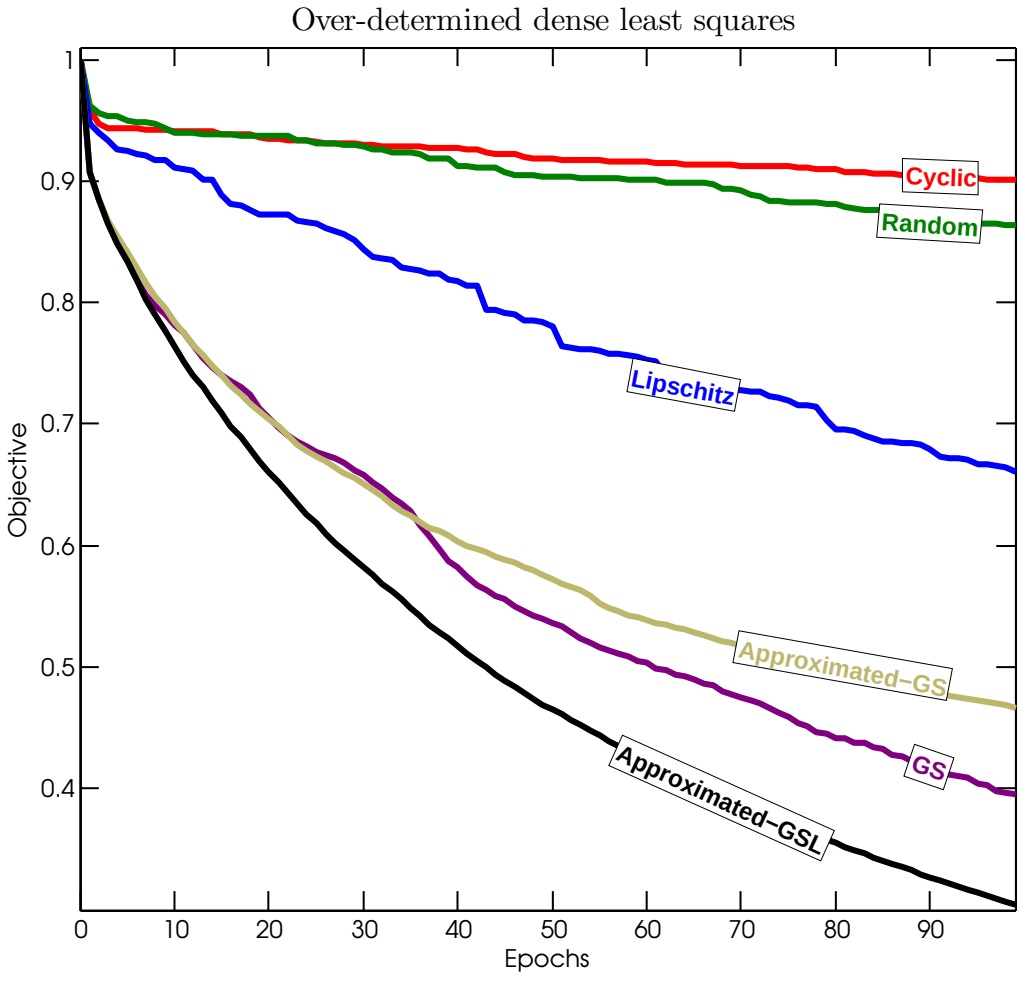
Logistic regression with ℓ_2-regularization and very sparse matrix.



Exact optimization makes a bigger difference than coordinate selection.

Experiment 3: Over-determined least squares

Least squares with dense matrix and nearest neighbour GS.



Approximate GS is still faster than random sampling.

Discussion

- GS not always practical.
- But even approximate GS rules may outperform random.

Discussion

- GS not always practical.
- But even approximate GS rules may outperform random.
- We've given a justification for line-search in certain scenarios.
- We proposed GSL rule, and approximate/proximal variants.

Discussion

- GS not always practical.
- But even approximate GS rules may outperform random.
- We've given a justification for line-search in certain scenarios.
- We proposed GSL rule, and approximate/proximal variants.
- Analysis extends to block updates.
- Could be used for accelerated/parallel methods [Fercocq & Richtárik, 2013], primal-dual methods [Shalev-Schwartz & Zhang, 2013], and without strong-convexity [Luo & Tseng, 1993].