Is Greedy Coordinate Descent a Terrible Algorithm?

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

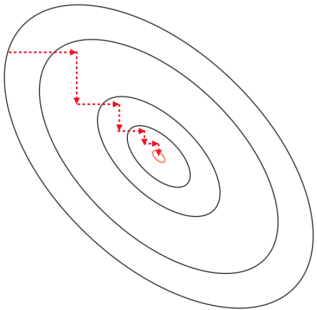
University of British Columbia

Coordinate Descent for Large-Scale Optimizaiton

- We consider the basic convex optimization problem:

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

  where f is differentiable and n is large.
- A popular approach is coordinate descent:
  1. Select a coordinate to update.
  2. Take a small gradient step along coordinate.



Why use coordinate descent?

- Theoretically, it is a provably bad algorithm:
  - The convergence rate is slower than gradient descent.
  - The iteration cost can be similar to gradient descent.
- But it is widely-used in practice:
  - Nothing works better for certain problems.
  - Certain fields think it is the 'ultimate' algorithm.
- ??
- Renewed theoretical interest began with Nesterov [2010]:
  - Global convergence rate for randomized coordinate selection.
  - Faster than gradient descent if iterations are n times cheaper.

Problems Suitable for Coordinate Descent

- Coordinate update is n times faster than gradient update when f has the form:

$$g(x) = \underbrace{\sum_{i=1}^{n} f_i(x_i)}_{\text{separable}} + \underbrace{\sum_{i=1}^{n}\sum_{j=1}^{n} f_{ij}(x_i, x_j)}_{\text{pairwise separable}} + \underbrace{f(Ax)}_{\text{linear composition}} .$$

  - f_i general convex functions (can be non-smooth).
  - f_{ij} and f are smooth.
  - A is a matrix and f is cheap.

- Key implementation ideas:
  - Separable part costs $O(1)$ for 1 partial derivative.
  - Pairwise part costs $O(n)$ for 1 partial derivative, instead of $O(n^2)$.
  - Linear composition costs $O(m)$ for 1 partial derivative by tracking Ax, instead of $O(mn)$.

Problems Suuitable for Coordinate Descent

- Examples: least squares, logistic regression, L1-regularization, SVMs.

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

- More examples: quadratics, graph-based label propagation, graphical models.

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

- There are many more examples where coordinate descent is n-times faster:
  - Matrix/tensor factorization, log-determinant problems, convex sub-modular extensions.

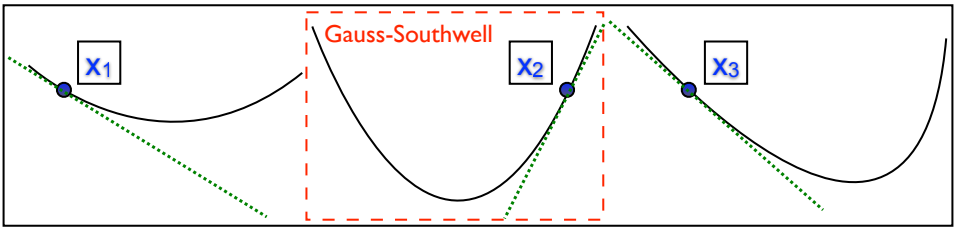
Context: Random vs. Greedy Coordinate Descent

This talk:

- Instead of random, consider classic steepest descent rule:

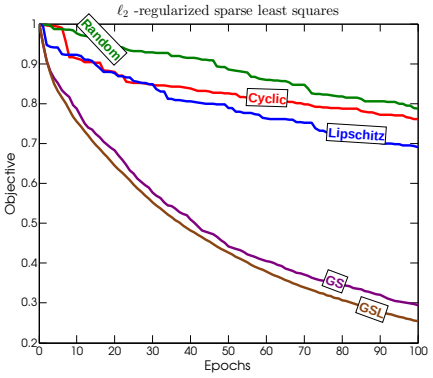
$$\underset{i}{\text{argmax}} |\nabla_i f(x)|,$$

  also known as the greedy rule or the Gauss-Southwell (GS) rule.



- GS is at least as expensive as random.
- But Nesterov showed the rate is the same.
- But this theory disagrees with practice...

Context: Random vs. Greedy Coordinate Descent



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

Gauss-Southwell??????????????????????????????

- How is computing max(gradient) n-times cheaper than computing the gradient?

- Consider a quadratic $f(x) = x^T Ax + b^t x$ with a very-sparse A:
  - For example, 10 non-zeroes per column.
  - In this case, only 10 gradients change when you change one variable.
  - You can efficiently track the max using a max-heap structure [Meshi et al., 2012].

- For pairwise objectives like quadratics, need max-degree $\approx$ average-degree.
  - Grid-based models, max degree $= 4$ and average degree ≈ 4.
  - Dense quadratic: max degree $= (n-1)$, average degree $= (n-1)$.
    - Gradient costs $O(n^2)$, updating 1 variable and tracking max(gradient) costs $O(n)$.
  - Facebook graph: max degree < 7000, average is ≈ 200.

- For some problems, Gauss-Southwell is approximated by nearest neighbours search.

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 $f(x) - \frac{\mu}{2}\|x\|^2$ is convex.
  - For some $\mu > 0$.

- If f is twice-differentiable, these assumptions are equivalent to

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

- We'll analyze coordinate descent with constant step size $\frac{1}{L}$,

$$x^{k+1}_{i_k} = x^k_{i_k} - \frac{1}{L}\nabla_{i_k} f(x^k).$$

Convergence of Randomized Coordinate Descent

- Convergence rate of gradient descent with step-size $1/L_f$ is

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

  so we require $O(\frac{L_f}{\mu}\log(1/\epsilon))$ iterations to reach accuracy ϵ.

- With i_k chosen uniformly, coordinate descent has [Nesterov, 2010]

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

  so we require $O(\frac{Ln}{\mu}\log(1/\epsilon))$ iterations to reach accuracy ϵ.

- Assuming "n-times cheaper", we need $O(\frac{L}{\mu}\log(1/\epsilon))$ in terms of gradient cost.
  - But $L \leq L_f$ so coordinate descent has a better bound.

Classic Analysis of Gauss-Southwell Rule

- GS rule chooses coordinate with largest directional derivative,

$$i_k \in \operatorname*{argmax}_i |\nabla_i f(x^k)|.$$

- 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 random [Boyd & Vandenberghe, 2004, §9.4.3].

Classic Analysis of Gauss-Southwell Rule

- GS rule chooses coordinate with largest directional derivative,

$$i_k \in \operatorname*{argmax}_i |\nabla_i f(x^k)|.$$

- 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 random [Boyd & Vandenberghe, 2004, §9.4.3].

Refined Gauss-Southwell Analysis

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

$$f(y) \geq f(x) + \langle \nabla f(x), y - x \rangle + \frac{\mu_1}{2}\|y - x\|_1^2.$$

- Using convex conjugate of $\|\cdot\|_1^2$ we now have that

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

- Combining with $\|\cdot\|_\infty^2$ in the GS progress bound gives a rate of

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

- This is the same as random if $\mu_1 = \mu/n$.
- The relationship between μ and μ_1 is given by

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

- Worst case same as random, but may be faster by factor up to n.
  - In the paper, we also analyze approximate GS rules and exact coordinate optimization.

Comparison for Separable Quadratic

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

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

- μ_1 is harmonic mean of λ_i divided by n:
  - Time needed for workers "working together" to finish task is μ_1 [Ferger, 1931].
  - Dominated by minimum λ_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.

Fast Convergence with Bias Term

- Consider the linear-prediction framework in statistics,

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

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

Rules Depending on Lipschitz Constants

What about non-uniform randomized sampling?

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

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

- Assume that we know the L_i or approximate them.
  - For example, we have $L = \|a_i\|^2 + \lambda$ for L2-regularized least squares,

$$\operatorname*{argmin}_x \|Ax - b\|^2 + \frac{\lambda}{2}\|x\|^2.$$

- 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 distinct.
- If we know gradients and L_i then should we use GS or Lipschitz sampling?

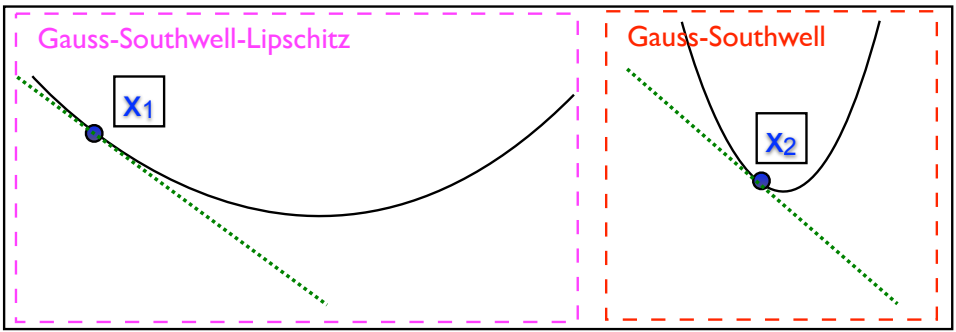
Gauss-Southwell-Lipschitz Rule

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

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

  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 satisfies the inequality

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

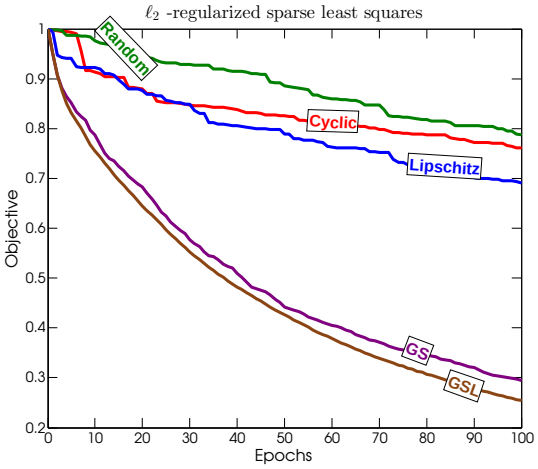
so GSL is at least as fast as GS and Lipschitz sampling.

- GSL using $\frac{1}{L_{i_k}}$ is unimprovable for quadratics,

$$f(x^{k+1}) = \min_{i,\alpha}\{f(x^k + \alpha e_i)\}.$$

- Gives tighter bound on maximum improvement rule.

GS vs. GSL Rule



- GSL rule gives modest but consistent improvements.
  - Improvement is large if we update multiple variables.

Gauss-Southwell-Lipschitz as Nearest Neighbour

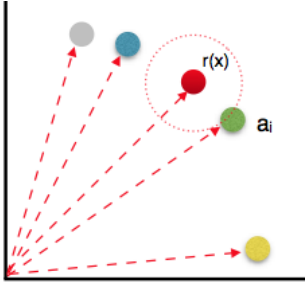
- An important problem class in machine learning is objectives of the form

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

where GS rule has the form

$$j_k \in \operatorname*{argmax}_j |\nabla_j f(x^k)| \equiv \operatorname*{argmax}_j |r(x^k)^T a^j|.$$

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



Gauss-Southwell-Lipschitz as Nearest Neighbour

- An important problem class in machine learning is objectives of the form

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

where GS rule has the form

$$j_k \in \operatorname*{argmax}_j |\nabla_j f(x^k)| \equiv \operatorname*{argmax}_j |r(x^k)^T a^j|.$$

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

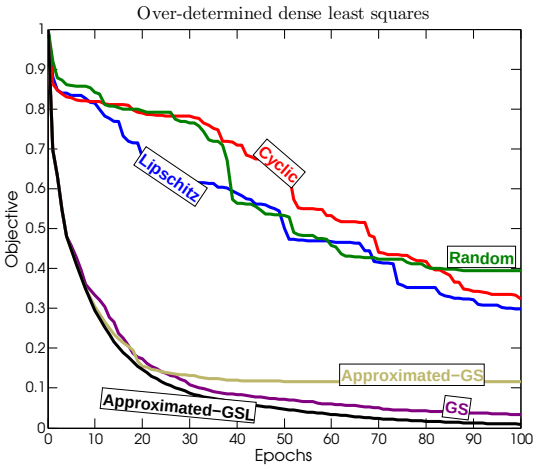
$$\operatorname*{argmin}_j \|r(x^k) - a^j\|^2 \equiv \operatorname*{argmax}_j \left\{ |\nabla_j f(x^k)| - \frac{1}{2}\|a^j\|^2 \right\}.$$

  - Exact if all $\|a^j\|$ are equal, otherwise it's biased toward large $\|a^j\|$.

- Usually $L_j = \gamma\|a^j\|^2$, and exact GSL is normalized nearest neighbours,

$$\operatorname*{argmin}_j \left\| r(x^k) - \frac{a^j}{\|a^j\|} \right\|^2 \equiv \operatorname*{argmax}_j \left\{ \frac{|\nabla_j f(x^k)|}{\sqrt{L_j}} \right\}.$$

Random vs. Approximate GS and GSL



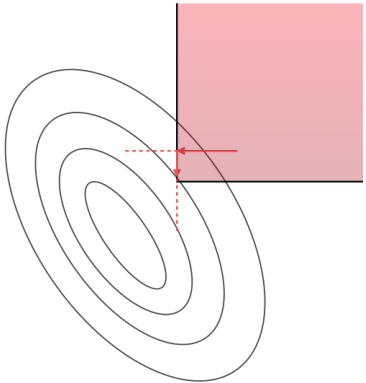
Approximate GS can still be faster than random sampling.

- And GSL rule is computed exactly via the nearest neighbour “approximation”.

Proximal Coordinate Descent

- Coordinate descent is popular for bound-constrained and L1-regularized problems,

$$\operatorname*{argmin}_{x \geq 0} \|Ax - b\|^2, \quad \operatorname*{argmin}_{x} \|Ax - b\|^2 + \lambda \|x\|_1.$$



Proximal Coordinate Descent

- Let's consider the general problem

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

where f is smooth and g_i might be non-smooth or enforce constraints.

- Here we can apply exact coordinate optimization or proximal-gradient update,

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

- Richtárik and Takác [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

In the literature there are multiple generalizations of GS to these problems:

- GS-s: Minimize directional derivative,

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

  - Used for ℓ_1-regularization, but $f(x^{k+1}) - f(x^k)$ 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 $g_i(x_i^{k+1}) - g_i(x_i^k)$.

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

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

  - 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

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

  the same rate as random selection.

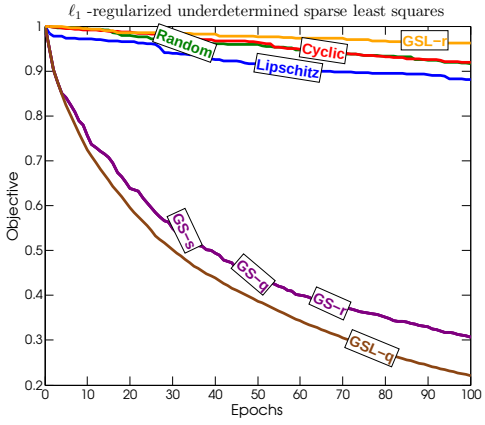
- This rate does not hold for GS-s or GS-r: they can be slower than random.
  - But again theory disagrees with practice (work similarly in practice).

- For piecewise-linear g_i we can get an asymptotic rate depending on μ_1,

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

  for the GS-q rule (under a non-degeneracy condition on subdifferential at solution).

Comparison of Proximal Gauss-Southwell Rules



- All three rules seem to work pretty well.
  - But you can make GS-s work badly with poor initialization.
  - And GS-r works badly if you use the L_i.

Summary

- GS is not always practical.
  - But it is efficient for certain problems.
  - And even approximations to it tend to converge faster than 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].