Coordinate Descent Converges Faster with the Gauss-Southwell Rule Than Random Selection: Appendix

Julie Nutini[1], Mark Schmidt[1], Issam H. Laradji[1], Michael Friedlander[2], Hoyt Koepke[3]

[1]University of British Columbia, [2]University of California, Davis, [3]Dato

In this document, we derive several results and present additional experiments that were omitted from the main paper due to space limitations. To allow easy referencing, the sections are numbered according to the location in the main paper that they are referenced.

2 Efficient calculation of GS rules for sparse problems

We first give additional details on how to calculate the GS rule efficiently for sparse instances of problems h_1 and h_2. We will consider the case where each g_i is smooth, but the ideas can be extended to allow a non-smooth g_i. Further, note that the efficient calculation does not rely on convexity, so these strategies can also be used for non-convex problems.

Problem h_2

Problem h_2 has the form

$$h_2(x) := \sum_{i \in V} g_i(x_i) + \sum_{(i,j) \in E} f_{ij}(x_i, x_j),$$

where each g_i and f_{ij} are differentiable and $G = \{V, E\}$ is a graph where the number of vertices $|V|$ is the same as the number of variables n. If all nodes in the graph have a degree (number of neighbours) bounded above by some constant d, we can implement the GS rule in $O(d \log n)$ after an $O(n + |E|)$ time initialization by maintaining the following information about x^k:

1. A vector containing the values $\nabla_i g_i(x_i^k)$.
2. A matrix containing the values $\nabla_i f_{ij}(x_i^k, x_j^k)$ in the first column and $\nabla_j f_{ij}(x_i^k, x_j^k)$ in the second column.
3. The elements of the gradient vector $\nabla h_2(x^k)$ stored in a binary max heap data structure [see Cormen et al., 2001, Chapter 6].

Given the heap structure, we can compute the GS rule in $O(1)$ by simply reading the index value of the root node in the max heap. The costs for initializing these structures are:

1. $O(n)$ to compute $g_i(x_i^0)$ for all n nodes.
2. $O(|E|)$ to compute $\nabla_{ij} f_{ij}(x_i^0, x_j^0)$ for all $|E|$ edges.
3. $O(n + |E|)$ to sum the values in the above structures to compute $\nabla h(x^0)$, and $O(n)$ to construct the initial max heap.

Thus, the one-time initialization cost is $O(n + |E|)$. The costs of updating the data structures after we update $x_{i_k}^k$ to $x_{i_k}^{k+1}$ for the selected coordinate i_k are:

1. $O(1)$ to compute $g_{i_k}(x_{i_k}^{k+1})$.
2. $O(d)$ to compute $\nabla_{ij} f_{ij}(x_i^{k+1}, x_j^{k+1})$ for $(i,j) \in E$ and $i = i_k$ or $j = i_k$ (only d such values exist by assumption, and all other $\nabla_{ij} f_{ij}(x_i, x_j)$ are unchanged).
3. $O(d)$ to update up to d elements of $\nabla h(x^{k+1})$ that differ from $\nabla h(x^k)$ by using differences in changed values of g_i and f_{ij}, followed by $O(d \log n)$ to perform d updates of the heap at a cost of $O(\log n)$ for each update.

The most expensive part of the update is modifying the heap, and thus the total cost is $O(d \log n)$.[1]

Problem h_1

Problem h_1 has the form

$$h_1(x) := \sum_{i=1}^{n} g_i(x_i) + f(Ax),$$

where g_i and f are differentiable, and A is an m by n matrix where we denote column i by a_i and row j by a_j^T. Note that f is a function from $\mathbb{R}^m$ to $\mathbb{R}$, and we assume $\nabla_j f$ only depends on $a_j^T x$. While this is a strong assumption (e.g., it rules out f being the product function), this class includes a variety of notable problems like the least squares and logistic regression models from the main paper. If A has z non-zero elements, with a maximum of c non-zero elements in each column and r non-zero elements in each row, then with a pre-processing cost of $O(z)$ we can implement the GS rule in this setting in $O(cr \log n)$ by maintaining the following information about x^k:

1. A vector containing the values $\nabla_i g_i(x_i^k)$.
2. A vector containing the product Ax^k.
3. A vector containing the values $\nabla f(Ax^k)$.
4. A vector containing the product $A^T \nabla f(Ax^k)$.
5. The elements of the gradient vector $\nabla h_1(x^k)$ stored in a binary max heap data structure.

The heap structure again allows us to compute the GS rule in $O(1)$, and the costs of initializing these structures are:

1. $O(n)$ to compute $g_i(x_i^0)$ for all n variables.
2. $O(z)$ to compute the product Ax^0.
3. $O(m)$ to compute $\nabla f(Ax^0)$ (using that $\nabla_j f$ only depends on $a_j^T x^0$).
4. $O(z)$ to compute $A^T \nabla f(Ax^0)$.
5. $O(n)$ to add the $\nabla_i g_i(x_i^0)$ to the above product to obtain $\nabla h_1(x^0)$ and construct the initial max heap.

As it is reasonable to assume that $z \geq m$ and $z \geq n$ (e.g., we have at least one non-zero in each row and column), the cost of the initialization is thus $O(z)$. The costs of updating the data structures after we update $x_{i_k}^k$ to $x_{i_k}^{k+1}$ for the selected coordinate i_k are:

[1] For less-sparse problems where $n < d \log n$, using a heap is actually inefficient and we should simply store $\nabla h(x^k)$ as a vector. The initialization cost is the same, but we can then perform the GS rule in $O(n)$ by simply searching through the vector for the maximum element.

1. $O(1)$ to compute $g_{i_k}(x_{i_k}^{k+1})$.

2. $O(c)$ to update the product using $Ax^{k+1} = Ax^k + (x_{i_k}^{k+1} - x_{i_k}^k)a_i$, since a_i has at most c non-zero values.

3. $O(c)$ to update up to c elements of $\nabla f(Ax^{k+1})$ that have changed (again using that $\nabla_j f$ only depends on $a_j^T x^{k+1}$).

4. $O(cr)$ to perform up to c updates of the form $A^T\nabla f(Ax^{k+1}) = A^T\nabla f(Ax^k) + (\nabla_j f(Ax^{k+1}) - \nabla_j f(Ax^k))(a_i)^T$, where each update costs $O(r)$ since each a_i has at most r non-zero values.

5. $O(cr \log n)$ to update the gradients in the heap.

The most expensive part is again the heap update, and thus the total cost is $O(cr \log n)$.

4 Relationship between μ_1 and μ

We can establish the relationship between μ and μ_1 by using the known relationship between the 2-norm and the 1-norm,

$$\|x\|_1 \geq \|x\| \geq \frac{1}{\sqrt{n}}\|x\|_1.$$

In particular, if we assume that f is μ-strongly convex in the 2-norm, then for all x and y we have

$$\begin{aligned} f(y) &\geq f(x) + \langle \nabla f(x), y - x\rangle + \frac{\mu}{2}\|y - x\|^2 \\ &\geq f(x) + \langle \nabla f(x), y - x\rangle + \frac{\mu}{2n}\|y - x\|_1^2, \end{aligned}$$

implying that f is at least $\frac{\mu}{n}$-strongly convex in the 1-norm. Similarly, if we assume that a given f is μ_1-strongly convex in the 1-norm then for all x and y we have

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

implying that f is at least μ_1-strongly convex in the 2-norm. Summarizing these two relationships, we have

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

4.1 Analysis for separable quadratic case

We first establish an equivalent definition of strong-convexity in the 1-norm, along the lines of Nesterov [2004, Theorem 2.1.9]. Subsequently, we use this equivalent definition to derive μ_1 for a separable quadratic function.

Equivalent definition of strong-convexity

Assume that f is μ_1-strongly convex in the 1-norm, so that for any $x, y \in \mathbb{R}^n$ we have

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

Reversing x and y in the above gives

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

and adding these two together yields

$$\langle \nabla f(y) - \nabla f(x), y - x \rangle \geq \mu_1 \|y - x\|_1^2. \tag{1}$$

Conversely, assume that for all x and y we have

$$\langle \nabla f(y) - \nabla f(x), y - x \rangle \geq \mu_1 \|y - x\|_1^2,$$

and consider the function $g(\tau) = f(x + \tau(y - x))$ for $\tau \in \mathbb{R}$. Then

$$\begin{aligned}
f(y) - f(x) - \langle \nabla f(x), y - x \rangle &= g(1) - g(0) - \langle \nabla f(x), y - x \rangle \\
&= \int_0^1 \frac{dg}{d\tau}(\tau) - \langle \nabla f(x), y - x \rangle \, d\tau \\
&= \int_0^1 \langle \nabla f(x + \tau(y - x)), y - x \rangle - \langle \nabla f(x), y - x \rangle \, d\tau \\
&= \int_0^1 \langle \nabla f(x + \tau(y - x)) - \nabla f(x), y - x \rangle \, d\tau \\
&\geq \int_0^1 \frac{\mu_1}{\tau} \|\tau(y - x)\|_1^2 \, d\tau \\
&= \int_0^1 \mu_1 \tau \|y - x\|_1^2 \, d\tau \\
&= \frac{\mu_1}{2} \tau^2 \|y - x\|_1^2 \Big|_0^1 \\
&= \frac{\mu_1}{2} \|y - x\|_1^2.
\end{aligned}$$

Thus, μ_1-strong convexity in the 1-norm is equivalent to having

$$\langle \nabla f(y) - \nabla f(x), y - x \rangle \geq \mu_1 \|y - x\|_1^2 \quad \forall \; x, y. \tag{2}$$

Strong-convexity constant μ_1 for separable quadratic functions

Consider a strongly convex quadratic function f with a diagonal Hessian $H = \nabla^2 f(x) = \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$, where $\lambda_i > 0$ for all $i = 1, \ldots, n$. We show that in this case

$$\mu_1 = \left(\sum_{i=1}^n \frac{1}{\lambda_i} \right)^{-1}.$$

From the previous section, μ_1 is the minimum value such that (2) holds,

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

Using $\nabla f(x) = Hx + b$ for some b and letting $z = y - x$, we get

$$\begin{aligned}
\mu_1 &= \inf_{x \neq y} \frac{\langle (Hy - b) - (Hx - b), y - x \rangle}{\|y - x\|_1^2} \\
&= \inf_{x \neq y} \frac{\langle H(y - x), y - x \rangle}{\|y - x\|_1^2} \\
&= \inf_{z \neq 0} \frac{z^T H z}{\|z\|_1^2} \\
&= \min_{\|z\|_1 = 1} z^T H z \\
&= \min_{e^T z = 1} \sum_{i=1}^n \lambda_i z_i^2,
\end{aligned}$$

where the last two lines use that the objective is invariant to scaling of z and to the sign of z (respectively), and where e is a vector containing a one in every position. This is an equality-constrained strictly-convex quadratic program, so its solution is given as a stationary point (z^*, η^*) of the Lagrangian,

$$\Lambda(z, \eta) = \sum_{i=1}^n \lambda_i z_i^2 + \eta(1 - e^T z).$$

Differentiating with respect to each z_i for $i = 1, \ldots, n$ and equating to zero, we have for all i that $2\lambda_i z_i^* - \eta^* = 0$, or

$$z_i^* = \frac{\eta^*}{2\lambda_i}. \tag{3}$$

Differentiating the Lagrangian with respect to η and equating to zero we obtain $1 - e^T z^* = 0$, or equivalently

$$1 = e^T z^* = \frac{\eta^*}{2} \sum_j \frac{1}{\lambda_j},$$

which yields

$$\eta^* = 2 \left(\sum_j \frac{1}{\lambda_j} \right)^{-1}.$$

Combining this result for η^* with equation (3), we have

$$z_i^* = \frac{1}{\lambda_i} \left(\sum_j \frac{1}{\lambda_j} \right)^{-1}.$$

This gives the minimizer, so we evaluate the objective at this point to obtain μ_1,

$$\begin{aligned}
\mu_1 &= \sum_{i=1}^{n} \lambda_i (z_i^*)^2 \\
&= \sum_{i=1}^{n} \lambda_i \left(\frac{1}{\lambda_i} \left(\sum_{j=1}^{n} \frac{1}{\lambda_j} \right)^{-1} \right)^2 \\
&= \sum_{i=1}^{n} \frac{1}{\lambda_i} \left(\sum_{j=1}^{n} \frac{1}{\lambda_j} \right)^{-2} \\
&= \left(\sum_{j=1}^{n} \frac{1}{\lambda_j} \right)^{-2} \left(\sum_{i=1}^{n} \frac{1}{\lambda_i} \right) \\
&= \left(\sum_{j=1}^{n} \frac{1}{\lambda_j} \right)^{-1}.
\end{aligned}$$

Interpretation in terms of 'working together'

In this separable quadratic case, μ_1 is given by the harmonic mean of the eigenvalues of the Hessian divided by n. The harmonic mean is dominated by its smallest values, and the harmonic mean divided by n has a particular interpretation in terms of processes 'working together' [Ferger, 1931]. If each λ_i represents the time taken by each process to finish a task (e.g., large values of λ_i correspond to slow workers), then μ is the time needed by the fastest worker to complete the task, and μ_1 is the time needed to complete the task if all processes work together (and have independent effects). Using this interpretation, the GS rule provides the most benefit over random selection when *working together is not efficient*, meaning that if the n processes work together, then the task is not solved much faster than if the fastest worker performed the task alone. This explains the non-intuitive scenario where GS provides the most benefit: if all workers have the same efficiency, then working together solves the problem n times faster. Similarly, if there is one slow worker (large λ_i), then the problem is solved roughly n times faster by working together. On the other hand, if most workers are slow (many large λ_i), then working together has little benefit.

5.1 Gauss-Southwell, exact optimization: convergence rate

We can obtain a faster convergence for GS using exact coordinate optimization for sparse variants of problems h_1 and h_2, by observing that the convergence rate can be expressed in terms of the sequence of $(1 - \mu_1/L_{i_k})$ values,

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

The worst case occurs when the sequence of $(1 - \mu_1/L_{i_k})$ values is as large as possible. However, using exact coordinate optimization guarantees that, after we have updated coordinate i, the GS rule will never select it again until one of its neighbours has been selected. Thus, we can obtain a tighter bound on the worst-case convergence rate using GS with exact coordinate optimization on iteration k, by solving the following combinatorial optimization problem defined on a weighted graph:

Problem 1. *We are given a graph $G = (V, E)$ with n nodes, a number M_i associated with each node i, and an iteration number k. Choose a sequence $\{i_t\}_{t=1}^{k}$ that maximizes the sum of the M_{i_t}, subject to the following*

constraint: after each time node i has been chosen, it cannot be chosen again until after a neighbour of node i has been chosen.

We can use the M_i chosen by this problem to obtain an upper-bound on the sequence of $\log(1-\mu_1/L_i)$ values, and if the largest M_i values are not close to each other in the graph, then this rate can be much faster than the rate obtained by alternating between the largest M_i values. In the particular case of chain-structured graphs, a worst-case sequence can be constructed that spends all but $O(n)$ iterations in one of two solution modes: (i) alternate between two nodes i and j that are connected by an edge with the highest value of $\frac{M_i+M_j}{2}$, or (ii) alternate between three nodes $\{i,j,k\}$ with the highest value of $\frac{M_i+M_j+M_k}{3}$, where there is an edge from i to j and from j to k, but not from i to k. To show that these are the two solution modes, observe that the solution must eventually cycle because there are a finite number of nodes. If you have more than three nodes in the cycle, then you can always remove one node from the cycle to obtain a better average weight for the cycle without violating the constraint. We will fall into mode (i) if the average of M_i and M_j in this mode is larger than the average of M_i, M_j and M_k in the second mode. We can construct a solution to this problem that consists of a 'burn-in' period, where we choose the largest M_i, followed by repeatedly going through the better of the two solution modes up until the final three steps, where a 'burn-out' phase arranges to finish with several large M_i. By setting $M_i = \log(1-\mu_1/L_i)$, this leads to a convergence rate of the form

$$f(x^k) - f(x^*) \leq O\left(\max\{\rho_2^G, \rho_3^G\}^k\right)[f(x^0) - f(x^*)],$$

where ρ_2^G is the maximizer of $\sqrt{(1-\mu_1/L_i)(1-\mu_1/L_j)}$ among all consecutive nodes i and j in the chain, and ρ_3^G is the maximizer of $\sqrt[3]{(1-\mu_1/L_i)(1-\mu_1/L_j)(1-\mu_1/L_k)}$ among consecutive nodes i, j, and k. The $O()$ notation gives the constant due to choosing higher $(1-\mu_1/L_i)$ values during the burn-in and burn-out periods.The implication of this result is that, if the large L_i values are more than two edges away from each other in the graph, the convergence rate can be much faster.

6.2 Gauss-Southwell-Lipschitz rule: convergence rate

The coordinate-descent method with a constant step-size of L_{i_k} uses the iteration

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

Because f is coordinate-wise L_{i_k}-Lipschitz continuous, we obtain the following bound on the progress made by each iteration:

$$\begin{aligned} f(x^{k+1}) &\leq f(x^k) + \nabla_{i_k} f(x^k)(x^{k+1} - x^k)_{i_k} + \frac{L_{i_k}}{2}(x^{k+1} - x^k)_{i_k}^2 \\ &= f(x^k) - \frac{1}{L_{i_k}}(\nabla_{i_k} f(x^k))^2 + \frac{L_{i_k}}{2}\left[\frac{1}{L_{i_k}}\nabla_{i_k} f(x^k)\right]^2 \\ &= f(x^k) - \frac{1}{2L_{i_k}}[\nabla_{i_k} f(x^k)]^2 \\ &= f(x^k) - \frac{1}{2}\left[\frac{\nabla_{i_k} f(x^k)}{\sqrt{L_{i_k}}}\right]^2. \end{aligned} \tag{4}$$

By choosing the coordinate to update according to the Gauss-Southwell-Lipchitz (GSL) rule,

$$i_k = \operatorname*{argmax}_i \frac{|\nabla_i f(x^k)|}{\sqrt{L_i}},$$

we obtain the tightest possible bound on (4). We define the following norm,

$$\|x\|_L = \sum_{i=1}^n \sqrt{L_i}|x_i|, \tag{5}$$

which has a dual norm of

$$\|x\|_L^* = \max_i \frac{1}{\sqrt{L_i}}|x_i|.$$

Under this notation, and using the GSL rule, (4) becomes

$$f(x^{k+1}) \leq f(x^k) - \frac{1}{2}\big(\|\nabla f(x^k)\|_L^*\big)^2,$$

Measuring strong-convexity in the norm $\|\cdot\|_L$, we get

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

Minimizing both sides with respect to y we get

$$\begin{aligned} f(x^*) &\geq f(x) - \sup_y\{\langle -\nabla f(x), y - x\rangle - \frac{\mu_L}{2}\|y - x\|_L^2\} \\ &= f(x) - \left(\frac{\mu_L}{2}\|\cdot\|_L^2\right)^*(-\nabla f(x)) \\ &= f(x) - \frac{1}{2\mu_L}\big(\|\nabla f(x)\|_L^*\big)^2. \end{aligned}$$

Putting these together we get

$$f(x^{k+1}) - f(x^*) \leq (1 - \mu_L)[f(x^k) - f(x^*)]. \tag{6}$$

6.2 Comparing μ_L to μ_1 and μ

By the logic of Section 4 of this document, to establish a relationship between different strong-convexity constants under different norms, it is sufficient to establish the relationships between the squared norms. In this section, we use this to establish the relationship between μ_L defined in (5) and both μ_1 and μ.

Relationship between μ_L and μ_1

We have

$$c\|x\|_1 - \|x\|_L = c\sum_i |x_i| - \sum_i \sqrt{L_i}|x_i| = \sum_i (c - \sqrt{L_i})|x_i|,$$

Assuming $c \geq \sqrt{L}$, where $L = \max_i\{L_i\}$, the expression is non-negative and we get

$$\|x\|_L \leq \sqrt{L}\|x\|_1.$$

By using

$$c\|x\|_L - \|x\|_1 = \sum_i (c\sqrt{L_i} - 1)|x_i|,$$

and assuming $c \geq \frac{1}{\sqrt{L_{min}}}$, where $L_{min} = \min_i\{L_i\}$, this expression is nonnegative and we get

$$\|x\|_1 \leq \frac{1}{\sqrt{L_{min}}}\|x\|_L.$$

The relationship between μ_L and μ_1 is based on the squared norm, so in summary we have

$$\frac{\mu_1}{L} \leq \mu_L \leq \frac{\mu_1}{L_{min}}.$$

Relationship between μ_L and μ

Let $\vec{L}$ denote a vector with elements $\sqrt{L_i}$, and we note that

$$\|\vec{L}\| = \left(\sum_i (\sqrt{L_i})^2\right)^{1/2} = \left(\sum_i L_i\right)^{1/2} = \sqrt{n\bar{L}}, \quad \text{where } \bar{L} = \frac{1}{n}\sum_i L_i.$$

Using this, we have

$$\|x\|_L = x^T(\text{sign}(x) \circ \vec{L}) \leq \|x\|\|\,\text{sign}(x) \circ \vec{L}\| = \sqrt{n\bar{L}}\|x\|.$$

This implies that

$$\frac{\mu}{n\bar{L}} \leq \mu_L.$$

Note that can also show that $\mu_L \leq \frac{\mu}{L_{min}}$, but this is less tight than the upper bound from the previous section because $\mu_1 \leq \mu$.

Equivalence of Gauss-Southwell-Lipschitz and nearest neighbour search

Dhillon et al. [2011] discuss an interesting connection between the GS rule and the nearest neighbour search (NNS) problem for problems of the form

$$\min_{x \in \mathbb{R}^n} F(x) = f(Ax). \tag{7}$$

This is a special case of h_1 with no g_i functions, and its gradient has the special form

$$\nabla F(x) = A^T r(x),$$

where $r(x) = \nabla f(Ax)$. We use the symbol r because $r(x)$ is the residual vector $(Ax - b)$ in the special case of least squares. For this problem structure the GS rule has the form

$$i_k = \underset{i}{\text{argmax}}\, |r(x)^T a_i|,$$

where in this section we again use a_i to denote column i of A for $i = 1, \ldots, n$. Here, we also need a notation for its negation; for this, we also use a_i to denote $-(a_{i-n})$ for $i = (n+1), \ldots, 2n$. Under this notation, Dhillon et al. [2011] propose to approximate the above argmax by solving the following NNS problem

$$i_k = \underset{i \in [2n]}{\text{argmin}}\, \|r(x) - a_i\|,$$

where if i in the argmin is greater than n, we return $(i - n)$. We can justify this approximation using the logic

$$\begin{aligned}
i_k &= \underset{i \in [2n]}{\text{argmin}}\, \|r(x) - a_i\| \\
&= \underset{i \in [2n]}{\text{argmin}}\, \frac{1}{2}\|r(x) - a_i\|^2 \\
&= \underset{i \in [2n]}{\text{argmin}}\, \underbrace{\frac{1}{2}\|r(x)\|^2}_{\text{constant}} - r(x)^T a_i + \frac{1}{2}\|a_i\|^2 \\
&= \underset{i \in [2n]}{\text{argmax}}\, r(x)^T a_i - \frac{1}{2}\|a_i\|^2 \\
&= \underset{i \in [n]}{\text{argmax}}\, |r(x)^T a_i| - \frac{1}{2}\|a_i\|^2.
\end{aligned}$$

Thus, the nearest neighbour search computes an approximation to the GS rule that is biased towards coordinates where $\|a_i\|$ is small. Note that this formulation is equivalent to the GS rule in the special case that $\|a_i\| = 1$ (or any other constant) for all i. Shrivastava and Li [2014] have more recently considered the case where $\|a_i\| \leq 1$ and incorporate powers of $\|a_i\|$ in the NNS to yield a better approximation.

In the next 2 sections, we explore the connection between the GSL rule and the NNS problem. We in particular show the surprising result that for many problems of the form (7) we can formulate the *exact* GSL rule as a NNS problem (i.e., it is not an approximation as it is for the GS rule). We start by doing this for least squares, then consider more general scenarios.

Equivalence of GSL and NNS for least squares

The classic least squares problem is defined by

$$\min_{x \in \mathbb{R}^n} f(x) \equiv \frac{1}{2}\|Ax - b\|^2,$$

where we will use the same notation as Section 2 for problem h_1 for rows and columns of A. The gradient and Hessian for this problem have the form

$$\nabla f(x) = A^T(Ax - b), \quad \nabla^2 f(x) = A^T A.$$

Using $r(x) = Ax - b$, the gradient elements have the form

$$\nabla_i f(x) = r(x)^T a_i.$$

Since the diagonals of the Hessian are constant, we have that

$$L_i = \nabla^2_{ii} f(x) = (a_i)^T a_i = \|a_i\|^2.$$

Using these properties, we can compute the GSL rule by finding the index i corresponding to a solution of a normalized NNS problem

$$i_k = \underset{i \in [2n]}{\operatorname{argmin}} \left|\left| r(x) - \frac{a_i}{\|a_i\|} \right|\right|. \tag{8}$$

The exactness of this formula follows because

$$\begin{aligned}
i_k &= \underset{i \in [2n]}{\operatorname{argmin}} \frac{1}{2}\|r(x) - a_i/\|a_i\|\|^2 \\
&= \underset{i \in [2n]}{\operatorname{argmin}} \underbrace{\frac{1}{2}\|r(x)\|^2}_{\text{constant}} - \frac{r(x)^T a_i}{\|a_i\|} + \underbrace{\frac{1}{2}\frac{\|a_i\|^2}{\|a_i\|^2}}_{\text{constant}} \\
&= \underset{i \in [n]}{\operatorname{argmax}} \frac{|r(x)^T a_i|}{\|a_i\|} \\
&= \underset{i \in [n]}{\operatorname{argmax}} \frac{|\nabla_i f(x)|}{\sqrt{L_i}}.
\end{aligned}$$

Thus, the form of the Lipschitz constant conveniently removes the bias towards smaller values of $\|a_i\|$ when we try to formulate the classic GS rule as a NNS problem.

Equivalence of GSL and NNS for linear prediction

Consider the more general scenario where we have

$$\min_{x \in \mathbb{R}^n} F(x) = \sum_{i=1}^{m} f(a_i^T x),$$

for some twice-differentiable univariate function f where f' is γ-Lipschitz continuous. This includes least squares and logistic regression as a special case, and indeed this is a common abstraction in machine learning and statistics. For this problem structure we have

$$\nabla_i F(x) = \sum_{j=1}^{m} a_{ij} f'(a_j^T x) = r(x)^T a_i,$$

where we define $r(x) = [f'(a_1^T x), f'(a_i^T x), \ldots, f'(a_m^T x)]^T$. The diagonals of the Hessian have the form

$$\nabla^2_{ii} F(x) = \sum_{j=1}^{m} a_{ij}^2 f''(a_j^T x).$$

By using that f' is γ-Lipschitz continuous we have

$$\begin{aligned}
L_i &= \sup_{x \in \mathbb{R}^n} \nabla^2_{ii} F(x) \\
&= \sup_{x \in \mathbb{R}^n} \sum_{j=1}^{m} a_{ij}^2 f''(a_j^T x) \\
&\leq \sum_{j=1}^{m} a_{ij}^2 \sup_{x \in \mathbb{R}^n} f''(a_j^T x) \\
&\leq \|a_i\|^2 \sup_{x \in \mathbb{R}} f''(x) \\
&= \gamma \|a_i\|^2,
\end{aligned}$$

where the inequalities will typically hold with equality (e.g., because we can typically achieve the supremum with a common x like $x = 0$, and because each a_j will have at least one non-zero element so $a_j^T x$ spans $\mathbb{R}$). We now show that the normalized NNS problem (8) is also equivalent to the GSL rule for this problem,

$$\begin{aligned}
i_k &= \operatorname*{argmin}_{i \in [2n]} \frac{1}{2} \|r(x) - a_i / \|a_i\|\|^2 \\
&= \operatorname*{argmin}_{i \in [2n]} \underbrace{\frac{1}{2} \|r(x)\|^2}_{\text{constant}} - \frac{r(x)^T a_i}{\|a_i\|} + \underbrace{\frac{1}{2} \frac{\|a_i\|^2}{\|a_i\|^2}}_{\text{constant}} \\
&= \operatorname*{argmax}_{i \in [n]} \frac{|r(x)^T a_i|}{\sqrt{\gamma} \|a_i\|} \\
&= \operatorname*{argmax}_{i \in [n]} \frac{|\nabla_i f(x)|}{\sqrt{L_i}},
\end{aligned}$$

where we have used that $\gamma > 0$. Interestingly, we thus *do not need to know* γ to implement the GSL rule as a NNS problem. If we had a different function f_i for each training example and they each had a different γ_i, it would break the equivalence of GSL with normalized NNS. On the other hand, the GSL rule is equivalent to a NNS for general functions of the form $f(Ax)$, whenever for all i we have that $L_i = \gamma \|a_i\|^2$ for some constant γ.

7.2 Approximate Gauss-Southwell with additive error

In the additive error regime, the approximate Gauss-Southwell rule chooses an i_k satisfying

$$|\nabla_{i_k} f(x^k)| \geq \|\nabla f(x^k)\|_\infty - \epsilon_k, \quad \text{where } \epsilon_k \geq 0 \ \ \forall k,$$

and we note that we can assume $\epsilon_k \leq \|\nabla f(x^k)\|_\infty$ without loss of generality because we must always choose an i with $|\nabla_{i_k} f(x^k)| \geq 0$. Applying this to our bound on the iteration progress, we get

$$\begin{aligned} f(x^{k+1}) &\leq f(x^k) - \frac{1}{2L}\left[\nabla_{i_k} f(x^k)\right]^2 \\ &\leq f(x^k) - \frac{1}{2L}\left(\|\nabla f(x^k)\|_\infty - \epsilon_k\right)^2 \\ &= f(x^k) - \frac{1}{2L}\left(\|\nabla f(x^k)\|_\infty^2 - 2\epsilon_k\|\nabla f(x^k)\|_\infty + \epsilon_k^2\right) \\ &= f(x^k) - \frac{1}{2L}\|\nabla f(x^k)\|_\infty^2 + \frac{\epsilon_k}{L}\|\nabla f(x^k)\|_\infty - \frac{\epsilon_k^2}{2L} \end{aligned} \tag{9}$$

We first give a result that assumes f is L_1-Lipschitz continuous in the 1-norm. This implies an inequality that we prove next, followed by a convergence rate that depends on L_1. However, note that $L \leq L_1 \leq Ln$, so this potentially introduces a dependency on n. We subsequently give a slightly less concise result that has a worse dependency on ϵ but does not rely on L_1.

Gradient bound in terms of L_1

We say that ∇f is L_1-Lipschitz continuous in the 1-norm if we have for all x and y that

$$\|\nabla f(x) - \nabla f(y)\|_\infty \leq L_1\|x - y\|_1.$$

Similar to Nesterov [2004, Theorem 2.1.5], we now show that this implies

$$f(y) \geq f(x) + \langle \nabla f(x), y - x\rangle + \frac{1}{2L_1}\|\nabla f(y) - \nabla f(x)\|_\infty^2, \tag{10}$$

and subsequently that

$$\|\nabla f(x^k)\|_\infty = \|\nabla f(x^k) - \nabla f(x^*)\|_\infty \leq \sqrt{2L_1(f(x^k) - f(x^*))} \leq \sqrt{2L_1(f(x^0) - f(x^*))}, \tag{11}$$

where we have used that $f(x^k) \leq f(x^{k-1})$ for all k and any choice of i_{k-1} (this follows from the basic bound on the progress of coordinate descent methods).

We first show that ∇f being L_1-Lipschitz continuous in the 1-norm implies that

$$f(y) \leq f(x) + \langle \nabla f(x), y - x\rangle + \frac{L_1}{2}\|y - x\|_1^2,$$

for all x and y. Consider the function $g(\tau) = f(x + \tau(y - x))$ with $\tau \in \mathbb{R}$. Then

$$\begin{aligned}
f(y) - f(x) - \langle \nabla f(x), y - x \rangle &= g(1) - g(0) - \langle \nabla f(x), y - x \rangle \\
&= \int_0^1 \frac{dg}{d\tau}(\tau) - \langle \nabla f(x), y - x \rangle \, d\tau \\
&= \int_0^1 \langle \nabla f(x + \tau(y - x)), y - x \rangle - \langle \nabla f(x), y - x \rangle \, d\tau \\
&= \int_0^1 \langle \nabla f(x + \tau(y - x)) - \nabla f(x), y - x \rangle \, d\tau \\
&\leq \int_0^1 \|\nabla f(x + \tau(y - x)) - \nabla f(x)\|_1 \|y - x\|_\infty \, d\tau \\
&\leq \int_0^1 L_1 \tau \|y - x\|_1 \, \|y - x\|_\infty \, d\tau \\
&\leq \int_0^1 L_1 \tau \|y - x\|_1^2 \, d\tau \\
&= \frac{L_1}{2} \tau^2 \|y - x\|_1^2 \Big|_0^1 \\
&= \frac{L_1}{2} \|y - x\|_1^2.
\end{aligned}$$

To subsequently show (10), fix $x \in \mathbb{R}^n$ and consider the function

$$\phi(y) = f(y) - \langle \nabla f(x), y \rangle,$$

which is convex on $\mathbb{R}^n$ and also has an L_1-Lipschitz continuous gradient in the 1-norm, as

$$\begin{aligned}
\|\phi'(y) - \phi'(x)\|_\infty &= \|(\nabla f(y) - \nabla f(x)) - (\nabla f(x) - \nabla f(x))\|_\infty \\
&= \|\nabla f(y) - \nabla f(x)\|_\infty \\
&\leq L_1 \|y - x\|_1.
\end{aligned}$$

As the minimizer of ϕ is x (i.e., $\phi'(x) = 0$), for any $y \in \mathbb{R}^n$ we have

$$\begin{aligned}
\phi(x) = \min_v \phi(v) &\leq \min_v \phi(y) + \langle \phi'(y), v - y \rangle + \frac{L_1}{2} \|v - y\|_1^2 \\
&= \phi(y) - \sup_v \langle -\phi'(y), v - y \rangle - \frac{L_1}{2} \|v - y\|_1^2 \\
&= \phi(y) - \frac{1}{2L_1} \|\phi'(y)\|_\infty^2.
\end{aligned}$$

Substituting in the definition of ϕ, we have

$$\begin{aligned}
& f(x) - \langle \nabla f(x), x \rangle \leq f(y) - \langle \nabla f(x), y \rangle - \frac{1}{2L_1} \|\nabla f(y) - \nabla f(x)\|_\infty^2 \\
\iff \quad & f(x) \leq f(y) + \langle \nabla f(x), x - y \rangle - \frac{1}{2L_1} \|\nabla f(y) - \nabla f(x)\|_\infty^2 \\
\iff \quad & f(y) \geq f(x) + \langle \nabla f(x), y - x \rangle + \frac{1}{2L_1} \|\nabla f(y) - \nabla f(x)\|_\infty^2.
\end{aligned}$$

Additive error bound in terms of L_1

Using (11) in (9) and noting that $\epsilon_k \geq 0$, we obtain

$$\begin{aligned} f(x^{k+1}) &\leq f(x^k) - \frac{1}{2L}\|\nabla f(x^k)\|_\infty^2 + \frac{\epsilon_k}{L}\|\nabla f(x^k)\|_\infty - \frac{\epsilon_k^2}{2L} \\ &\leq f(x^k) - \frac{1}{2L}\|\nabla f(x^k)\|_\infty^2 + \frac{\epsilon_k}{L}\sqrt{2L_1(f(x^0) - f(x^*))} - \frac{\epsilon_k^2}{2L} \\ &\leq f(x^k) - \frac{1}{2L}\|\nabla f(x^k)\|_\infty^2 + \epsilon_k\frac{\sqrt{2L_1}}{L}\sqrt{f(x^0) - f(x^*)}. \end{aligned}$$

Applying strong convexity (taken with respect to the 1-norm), we get

$$f(x^{k+1}) - f(x^*) \leq \left(1 - \frac{\mu_1}{L}\right)\left[f(x^k) - f(x^*)\right] + \epsilon_k\frac{\sqrt{2L_1}}{L}\sqrt{f(x^0) - f(x^*)},$$

which implies

$$\begin{aligned} f(x^{k+1}) - f(x^*) &\leq \left(1 - \frac{\mu_1}{L}\right)^k\left[f(x^0) - f(x^*)\right] + \sum_{i=1}^k\left(1 - \frac{\mu_1}{L}\right)^{k-i}\epsilon_i\frac{\sqrt{2L_1}}{L}\sqrt{f(x^0) - f(x^*)} \\ &= \left(1 - \frac{\mu_1}{L}\right)^k\left[f(x^0) - f(x^*) + \sqrt{f(x^0) - f(x^*)}A_k\right], \end{aligned}$$

where

$$A_k = \frac{\sqrt{2L_1}}{L}\sum_{i=1}^k\left(1 - \frac{\mu_1}{L}\right)^{-i}\epsilon_i.$$

Additive error bound in terms of L

By our additive error inequality, we have

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

Using this again in (9) we get

$$\begin{aligned} f(x^{k+1}) &\leq f(x^k) - \frac{1}{2L}\|\nabla f(x^k)\|_\infty^2 + \frac{\epsilon_k}{L}\|\nabla f(x^k)\|_\infty - \frac{\epsilon_k^2}{2L} \\ &\leq f(x^k) - \frac{1}{2L}\|\nabla f(x^k)\|_\infty^2 + \frac{\epsilon_k}{L}\left(|\nabla_{i_k} f(x^k)| + \epsilon_k\right) - \frac{\epsilon_k^2}{2L} \\ &= f(x^k) - \frac{1}{2L}\|\nabla f(x^k)\|_\infty^2 + \frac{\epsilon_k}{L}|\nabla_{i_k} f(x^k)| + \frac{\epsilon_k^2}{2L}. \end{aligned}$$

Further, from our basic progress bound that holds for any i_k we have

$$f(x^*) \leq f(x^{k+1}) \leq f(x^k) - \frac{1}{2L}\left[\nabla_{i_k} f(x^k)\right]^2 \leq f(x^0) - \frac{1}{2L}\left[\nabla_{i_k} f(x^k)\right]^2,$$

which implies

$$|\nabla_{i_k} f(x^k)| \leq \sqrt{2L(f(x^0) - f(x^*))}.$$

and thus that

$$\begin{aligned} f(x^{k+1}) &\leq f(x^k) - \frac{1}{2L}\|\nabla f(x^k)\|_\infty^2 + \frac{\epsilon_k}{L}\sqrt{2L(f(x^0) - f(x^*))} + \frac{\epsilon_k^2}{2L} \\ &= f(x^k) - \frac{1}{2L}\|\nabla f(x^k)\|_\infty^2 + \epsilon_k\sqrt{\frac{2}{L}}\sqrt{f(x^0) - f(x^*)} + \frac{\epsilon_k^2}{2L}. \end{aligned}$$

Applying strong convexity and applying the inequality recursively we obtain

$$\begin{aligned} f(x^{k+1}) - f(x^*) &\leq \left(1 - \frac{\mu_1}{L}\right)^k \left[f(x^0) - f(x^*)\right] + \sum_{i=1}^{k} \left(1 - \frac{\mu_1}{L}\right)^{k-i} \left(\epsilon_i\sqrt{\frac{2}{L}}\sqrt{f(x^0) - f(x^*)} + \frac{\epsilon_i^2}{2L}\right) \\ &= \left(1 - \frac{\mu_1}{L}\right)^k \left[f(x^0) - f(x^*) + A_k\right], \end{aligned}$$

where

$$A_k = \sum_{i=1}^{k} \left(1 - \frac{\mu_1}{L}\right)^{-i} \left(\sqrt{\frac{2}{L}}\epsilon_i\sqrt{f(x^0) - f(x^*)} + \frac{\epsilon_i^2}{2L}\right).$$

Although uglier than the expression depending on L_1, this expression will tend to be smaller unless ϵ_k is not small.

8. Convergence Analysis of GS-s, GS-r, and GS-q Rules

In this section, we consider problems of the form

$$\min_{x \in \mathbb{R}^n} F(x) = f(x) + g(x) = f(x) + \sum_{i=1}^{n} g_i(x_i),$$

where f satisfies our usual assumptions, but the g_i can be non-smooth. We first introduce some notation that will be needed to state our result for the GS-q rule, followed by stating the result and then showing that it holds in two parts. We then turn to showing that the rule cannot hold in general for the GS-s and GS-r rules.

Notation and basic inequality

To analyze this case, an important inequality we will use is that the L-Lipschitz-continuity of $\nabla_i f$ implies that for all x, i, and d that

$$\begin{aligned} F(x + de_i) = f(x + de_i) + g(x + de_i) &\leq f(x) + \langle \nabla f(x), de_i \rangle + \frac{L}{2}d^2 + g(x + de_i) \\ &= f(x) + g(x) + \langle \nabla f(x), de_i \rangle + \frac{L}{2}d^2 + g_i(x_i + d) - g_i(x_i) \\ &= F(x) + V_i(x, d), \end{aligned} \tag{12}$$

where

$$V_i(x, d) \equiv \langle \nabla f(x), de_i \rangle + \frac{L}{2}d^2 + g_i(x_i + d) - g_i(x_i).$$

Notice that the GS-q rule is defined by

$$i_k = \operatorname*{argmin}_i \{\min_d V_i(x, d)\},$$

We use the notation $d_i^k = \text{argmin}_d V_i(x^k, d)$ and we will use d^k to denote the vector containing these values for all i. When using the GS-q rule, the iteration is defined by

$$\begin{aligned} x^{k+1} &= x^k + d_{i_k} e_{i_k} \\ &= x^k + \underset{d}{\text{argmin}}\{V_{i_k}(x, d)\} e_{i_k}. \end{aligned} \tag{13}$$

In this notation the GS-r rule is given by

$$j_k = \underset{i}{\text{argmax}} |d_i^k|.$$

We will use the notation x_+^k to be the step that would be taken at x_k if we update coordinate j_k according the GS-r rule

$$x_+^k = x^k + d_{j_k} e_{j_k}.$$

From the optimality of d_i^k, we have for any i that

$$-L[(x_i^k - \frac{1}{L}\nabla_i f(x^k)) - (x_i^k + d_i^k)] \in \partial g_i(x_i^k + d_i^k), \tag{14}$$

and we will use the notation s_i^k for the unique element of $\partial g_j(x_j^k + d_j^k)$ satisfying this relationship. We use s^k to denote the vector containing these values.

Convergence bound for GS-q rule

Under this notation, we can show that coordinate descent with the GS-q rule satisfies the bound

$$F(x^{k+1}) - F(x^*) \leq \min\left\{\left(1 - \frac{\mu}{Ln}\right)[f(x^k) - f(x^*)], \left(1 - \frac{\mu_1}{L}\right)[f(x^0) - f(x^*)] + \epsilon_k\right\}, \tag{15}$$

where

$$\epsilon_k \leq \frac{\mu_1}{L}\left(g(x_+^k) - g(x^k + d^k) + \langle s^k, (x^k + d^k) - x_+^k\rangle\right),$$

We note that if g is linear then $\epsilon^k = 0$ and this convergence rate reduces to

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

Otherwise, ϵ_k depends how far $g(x_+^k)$ lies above a particular linear underestimate extending from $(x^k + d^k)$, as well as the conditioning of f. We show this result by first showing that the GS-q rule makes at least as much progress as randomized selection (first part of the min), and then showing that the GS-q rule also makes at least as much progress as the GS-r rule (second part of the min).

GS-q is at least as fast as random

Our argument in this section follows a similar approach to Richtárik and Takáč [2014]. In particular, combining (12) and (13) we have the following upper bound on the iteration progress

$$\begin{aligned}
F(x^{k+1}) &\leq F(x^k) + \min_{i\in\{1,2,\dots,n\}}\left\{\min_{d\in\mathbb{R}} V_i(x^k,d)\right\},\\
&= F(x^k) + \min_{i\in\{1,2,\dots,n\}}\left\{\min_{y\in\mathbb{R}^n} V_i(x^k,y_i-x_i^k)\right\},\\
&= F(x^k) + \min_{y\in\mathbb{R}^n}\left\{\min_{i\in\{1,2,\dots,n\}} V_i(x^k,y_i-x_i^k)\right\},\\
&\leq F(x^k) + \min_{y\in\mathbb{R}^n}\left\{\frac{1}{n}\sum_{i=1}^n V_i(x^k,y_i-x^k)\right\}\\
&= F(x^k) + \frac{1}{n}\min_{y\in\mathbb{R}^n}\left\{\langle\nabla f(x^k),y-x^k\rangle + \frac{L}{2}\|y-x^k\|^2 + g(y) - g(x^k)\right\}\\
&= \left(1-\frac{1}{n}\right)F(x^k) + \frac{1}{n}\min_{y\in\mathbb{R}^n}\left\{f(x^k) + \langle\nabla f(x^k),y-x^k\rangle + \frac{L}{2}\|y-x^k\|^2 + g(y)\right\}.
\end{aligned}$$

From strong convexity of f, we have that F is also μ-strongly convex and that

$$\begin{aligned}
f(x^k) &\leq f(y) - \langle\nabla f(x^k),y-x^k)\rangle - \frac{\mu}{2}\|y-x^k\|^2,\\
F(\alpha x^* + (1-\alpha)x^k) &\leq \alpha f(x^*) + (1-\alpha)f(x^k) - \frac{\alpha(1-\alpha)\mu}{2}\|x^k-x^*\|^2,
\end{aligned}$$

for any $y\in\mathbb{R}^n$ and any $\alpha\in[0,1]$ [see Nesterov, 2004, Theorem 2.1.9]. Using these gives us

$$\begin{aligned}
&F(x^{k+1})\\
&\leq \left(1-\frac{1}{n}\right)F(x^k) + \frac{1}{n}\min_{y\in\mathbb{R}^n}\left\{f(y) - \frac{\mu}{2}\|y-x\|^2 + \frac{L}{2}\|y-x^k\|^2 + g(y)\right\}\\
&= \left(1-\frac{1}{n}\right)F(x^k) + \frac{1}{n}\min_{y\in\mathbb{R}^n}\left\{F(y) + \frac{L-\mu}{2}\|y-x^k\|^2\right\}\\
&\leq \left(1-\frac{1}{n}\right)F(x^k) + \frac{1}{n}\min_{\alpha\in[0,1]}\left\{F(\alpha x^* + (1-\alpha)x^k) + \frac{\alpha^2(L-\mu)}{2}\|x^k-x^*\|^2\right\}\\
&\leq \left(1-\frac{1}{n}\right)F(x^k) + \frac{1}{n}\min_{\alpha\in[0,1]}\left\{\alpha F(x^*) + (1-\alpha)F(x^k) + \frac{\alpha^2(L-\mu)-\alpha(1-\alpha)\mu}{2}\|x^k-x^*\|^2\right\}\\
&\leq \left(1-\frac{1}{n}\right)F(x^k) + \frac{1}{n}\left[\alpha^* F(x^*) + (1-\alpha^*)F(x^k)\right] \qquad \left(\text{choosing } \alpha^* = \frac{\mu}{L}\in(0,1]\right)\\
&= \left(1-\frac{1}{n}\right)F(x^k) + \frac{\alpha^*}{n}F(x^*) + \frac{(1-\alpha^*)}{n}F(x^k)\\
&= F(x^k) - \frac{\alpha^*}{n}[F(x^k) - F(x^*)].
\end{aligned}$$

Subtracting $F(x^*)$ from both sides of this inequality gives us

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

GS-q is at least as fast as GS-r

In this section we derive the right side of the bound (15) for the GS-r rule, but note it also applies to the GS-q rule because from (12) and (13) we have

$$\begin{aligned} F(x^{k+1}) &\leq F(x^k) + \min_i V_i(x, d_i^k) \quad \text{(GS-}q\text{ rule)} \\ &\leq F(x^k) + V_{j_k}(x, d_{j_k}^k) \quad (j_k \text{ selected by the GS-}r\text{ rule}) \end{aligned}$$

Note that we lose progress by considering a bound based on the GS-r rule, but its connection to the ∞-norm will make it easier to derive an upper bound.

By the convexity of g_{j_k} we have

$$\begin{aligned} g_{j_k}(x_{j_k}^k) &\geq g_{j_k}(x_{j_k}^k + d_{j_k}^k) + s_{j_k}^k(x_{j_k}^k - (x_{j_k}^k + d_{j_k}^k)) \\ &= g_{j_k}(x_{j_k}^k + d_{j_k}^k) - (-Ld_{j_k}^k - \nabla_{j_k} f(x^k))(d_{j_k}^k) \\ &= g_{j_k}(x_{j_k}^k + d_{j_k}^k) + \nabla_{j_k} f(x^k) d_{j_k}^k + L(d_{j_k}^k)^2, \end{aligned}$$

where s_i^k is defined by (14). Using this we have that

$$\begin{aligned} F(x^{k+1}) &\leq F(x^k) + V_j(x, d_{j_k}^k) \\ &= F(x^k) + \nabla_j f(x^k)(d_{j_k}^k) + \frac{L}{2}(d_{j_k}^k)^2 + g_i(x_{j_k}^k + d_{j_k}^k) - g_i(x_{j_k}^k) \\ &\leq F(x^k) + \nabla_j f(x^k)(d_{j_k}^k) + \frac{L}{2}(d_{j_k}^k)^2 - \nabla_{j_k} f(x^k) d_{j_k}^k - L(d_{j_k}^k)^2 \\ &= F(x^k) - \frac{L}{2}(d_{j_k}^k)^2. \end{aligned}$$

Adding and subtracting $F(x^*)$ and noting that j_k is selected using the GS-r rule, we obtain the upper bound

$$F(x^{k+1}) - F(x^*) \leq F(x^k) - F(x^*) - \frac{L}{2}||d^k||_\infty^2. \tag{16}$$

Recall that we use x_+^k to denote the iteration that would result if we chose j_k and actually performed the GS-r update. Using the Lipschitz continuity of the gradient and definition of the GS-q rule again, we have

$$\begin{aligned} F(x^{k+1}) &\leq F(x^k) + \nabla f(x^k)^T(x^{k+1} - x^k) + \frac{L}{2}||x^{k+1} - x^k||^2 + g(x^{k+1}) - g(x^k) \\ &\leq F(x^k) + \nabla f(x^k)^T(x_+^k - x^k) + \frac{L}{2}||x_+^k - x^k||^2 + g(x_k^+) - g(x^k) \\ &= f(x^k) + \nabla f(x^k)^T(x_+^k - x^k) + \frac{L}{2}\|d^k\|_\infty^2 + g(x_+^k) \end{aligned}$$

By the strong-convexity of f, for any $y \in \mathbb{R}^N$ we have

$$f(x^k) \leq f(y) - \nabla f(x^k)^T(y - x^k) - \frac{\mu_1}{2}\|y - x^k\|_1^2,$$

and using this we obtain

$$F(x^{k+1}) \leq f(y) + \nabla f(x^k)^T(x_+^k - y) - \frac{\mu_1}{2}\|y - x^k\|_1^2 + \frac{L}{2}\|d^k\|_\infty^2 + g(x_+^k). \tag{17}$$

By the convexity of g and $s^k \in \partial g(x^k + d^k)$, we have

$$g(y) \geq g(x^k + d^k) + \langle s^k, y - (x^k + d^k)\rangle.$$

Combining (17) with the above inequality, we have

$$F(x^{k+1}) - F(y) \leq \langle \nabla f(x^k), x_+^k - y\rangle - \frac{\mu_1}{2}\|y - x^k\|_1^2 + \frac{L}{2}\|d^k\|_\infty^2 + g(x_+^k) - g(x^k + d^k) + \langle s^k, (x^k + d^k) - y\rangle.$$

We add and subtract $\langle s^k, x_+^k\rangle$ on the right-hand side to get

$$F(x^{k+1}) - F(y) \leq \langle \nabla f(x^k) + s^k, x_+^k - y\rangle - \frac{\mu_1}{2}\|y - x^k\|_1^2 + \frac{L}{2}\|d^k\|_\infty^2 + g(x_+^k) - g(x^k + d^k) + \langle s^k, (x^k + d^k) - x_+^k\rangle.$$

Let $c^k = g(x_+^k) - g(x^k + d^k) + \langle s^k, (x^k + d^k) - x_+^k\rangle$, which is non-negative by the convexity g. Making this substitution, we have

$$F(y) \geq F(x^{k+1}) + \langle -Ld^k, y - x_+^k\rangle + \frac{\mu_1}{2}\|y - x^k\|_1^2 - \frac{L}{2}\|d^k\|_\infty^2 - c^k.$$

Now add and subtract $\langle -Ld^k, x^k\rangle$ to the right-hand side and use (14) to get

$$F(y) \geq F(x^{k+1}) + \langle -Ld^k, y - x^k\rangle + \frac{\mu_1}{2}\|y - x^k\|_1^2 - \frac{L}{2}\|d^k\|_\infty^2 - L\langle d^k, x^k - x_+^k\rangle - c^k.$$

Minimizing both sides with respect to y results in

$$\begin{aligned} F(x^*) &\geq F(x^{k+1}) - \frac{L^2}{2\mu_1}\|d^k\|_\infty^2 - \frac{L}{2}\|d^k\|_\infty^2 - L\langle d^k, x^k - x_+^k\rangle - c^k \\ &\geq F(x^{k+1}) - \frac{L^2}{2\mu_1}\|d^k\|_\infty^2 - \frac{L}{2}\|d^k\|_\infty^2 + L\|d^k\|_\infty^2 - c^k \\ &= F(x^{k+1}) - \frac{L(L-\mu_1)}{2\mu_1}\|d^k\|_\infty^2 - c^k, \end{aligned}$$

where we've used that $x_+^k = x^k + d_{j_k}^k e_{j_k}$ and $|d_{j_k}^k| = \|d^k\|_\infty$. Combining this with equation (16), we get

$$\begin{aligned} F(x^{k+1}) - F(x^*) &\leq F(x^k) - F(x^*) - \frac{L}{2}\|d^k\|_\infty^2 \\ F(x^{k+1}) - F(x^*) &\leq F(x^k) - F(x^*) - \frac{\mu_1}{(L-\mu_1)}\left[F(x^{k+1}) - F(x^*) - c^k\right] \\ \left(1 + \frac{\mu_1}{(L-\mu_1)}\right)\left[F(x^{k+1}) - F(x^*)\right] &\leq F(x^k) - F(x^*) + \epsilon^k\frac{\mu_1}{(L-\mu_1)} \\ F(x^{k+1}) - F(x^*) &\leq \frac{(L-\mu_1)}{L}\left[F(x^k) - F(x^*)\right] + c^k\frac{\mu_1}{L} \\ F(x^{k+1}) - F(x^*) &\leq \left(1 - \frac{\mu_1}{L}\right)\left[F(x^k) - F(x^*)\right] + c^k\frac{\mu_1}{L}. \end{aligned}$$

Lack of progress of the GS-s rule

We now show that the rate $(1 - \mu_1/L)$, and even the slower rate $(1 - \mu/Ln)$, cannot hold for the GS-s rule. We do this by constructing a problem where an iteration of the GS-s method does not make sufficient progress. In particular, consider the bound-constrained problem

$$\min_{x \in C} f(x) = \frac{1}{2}\|Ax - b\|_2^2,$$

where $C = \{x \ : \ x \geq 0\}$, and

$$A = \begin{pmatrix} 1 & 0 \\ 0 & 0.7 \end{pmatrix}, \quad b = \begin{pmatrix} -1 \\ -3 \end{pmatrix}, \quad x^0 = \begin{pmatrix} 1 \\ 0.1 \end{pmatrix}, \quad x^* = \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$$

We thus have that

$$\begin{aligned} f(x^0) &= \frac{1}{2}((1+1)^2 + (.07+3)^2) \approx 6.7 \\ f(x^*) &= \frac{1}{2}((-1)^2 + (-3)^2) = 5 \\ \nabla f(x^0) &= A^T(Ax_0 - b) \approx \begin{pmatrix} 2.0 \\ 2.1 \end{pmatrix} \\ \nabla^2 f(x) &= A^T A = \begin{pmatrix} 1 & 0 \\ 0 & 0.49 \end{pmatrix}. \end{aligned}$$

The parameter values for this problem are

$$\begin{aligned} &n = 2 \\ &\mu = \lambda_{min} = 0.49 \\ &L = \lambda_{max} = 1 \\ &\mu_1 = \left(\frac{1}{\lambda_1} + \frac{1}{\lambda_2}\right)^{-1} = 1 + \frac{1}{0.49} \approx 0.33, \end{aligned}$$

where the λ_i are the eigenvalues of $A^T A$, and μ and μ_1 are the corresponding strong-convexity constants for the 2-norm and 1-norm, respectively.

The proximal operator of the indicator function is the projection onto the set C, which involves setting negative elements to zero. Thus, our iteration update is given by

$$x^{k+1} = \text{prox}_{\delta_C}[x^k - \frac{1}{L}\nabla_{i_k} f(x^k) e_{i_k}] = \max(x^k - \frac{1}{L}\nabla_{i_k} f(x^k) e_{i_k}, 0),$$

For this problem, the GS-s rule is given by

$$i = \underset{i}{\text{argmax}} \, |\eta_i^k|,$$

where

$$\eta_i^k = \begin{cases} \nabla_i f(x^k), & \text{if } x_i^k \neq 0 \text{ or } \nabla_i f(x^k) < 0 \\ 0, & \text{otherwise} \end{cases}.$$

Based on the value of $\nabla f(x^0)$, the GS-s rule thus chooses to update coordinate 2, setting it to zero and obtaining

$$f(x^1) = \frac{1}{2}((1+1)^2 + (-3)^2) = 6.5.$$

Thus we have

$$\frac{f(x^1) - f(x^*)}{f(x^0) - f(x^*)} \approx \frac{6.5 - 5}{6.7 - 5} \approx 0.88,$$

even though the bounds obtain the faster rates of

$$\begin{aligned} \left(1 - \frac{\mu}{Ln}\right) &= \left(1 - \frac{0.49}{2}\right) \approx 0.76, \\ \left(1 - \frac{\mu_1}{L}\right) &\approx (1 - 0.33) = 0.67. \end{aligned}$$

Thus, the GS-s rule does not satisfy either bound. On the other hand, the GS-r and GS-q rules are given in this context by

$$i_k = \underset{i}{\operatorname{argmax}} \left| \max\left(x^k - \frac{1}{L}\nabla_i f(x^k)e_i, 0\right) - x^k \right|,$$

and thus both these rules choose to update coordinate 1, setting it to zero to obtain $f(x^1) \approx 5.2$ and a progress ratio of

$$\frac{f(x^1) - f(x^*)}{f(x^0) - f(x^*)} \approx \frac{5.2 - 5}{6.7 - 5} \approx 0.12,$$

which clearly satisfies both bounds.

Lack of progress of the GS-r rule

We now turn to showing that the GS-r rule does not satisfy these bounds in general. It will not be possible to show this for a simple bound-constrained problem since the GS-r and GS-q rules are equivalent for these problems. Thus, we consider the following ℓ_1-regularized problem

$$\min_{x \in \mathbb{R}^2} \frac{1}{2}\|Ax - b\|_2^2 + \lambda\|x\|_1 \equiv F(x).$$

We use the same A as the previous section, so that n, μ, L, and μ_1 are the same. However, we now take

$$b = \begin{pmatrix} 2 \\ -1 \end{pmatrix}, \quad x_0 = \begin{pmatrix} 0.4 \\ 0.5 \end{pmatrix}, \quad x_* = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad \lambda = 1,$$

so we have

$$f(x_0) \approx 3.1, \quad f(x_*) = 2$$

The proximal operator of the absolute value function is given by the soft-threshold function, and our coordinate update of variable i_k is given by

$$x_{i_k}^{k+1} = \operatorname{prox}_{\lambda|\cdot|}[x_{i_k}^{k+\frac{1}{2}}] = \operatorname{sgn}(x_{i_k}^{k+\frac{1}{2}}) \cdot \max(x_{i_k}^{k+\frac{1}{2}} - \lambda/L, 0),$$

where we have used the notation

$$x_i^{k+\frac{1}{2}} = x_i^k - \frac{1}{L}\nabla_i f(x^k)e_i.$$

The GS-r rule is defined by

$$i_k = \underset{i}{\operatorname{argmax}} |d_i^k|,$$

where $d_i^k = \operatorname{prox}_{\lambda|\cdot|}[x_i^{k+\frac{1}{2}}] - x_i^k$ and in this case

$$d^0 = \begin{pmatrix} 0.6 \\ -0.5 \end{pmatrix}.$$

Thus, the GS-r rule chooses to update coordinate 1. After this update the function value is

$$F(x^1) \approx 2.9,$$

so the progress ratio is

$$\frac{F(x^1) - F(x^*)}{F(x^0) - F(x^*)} \approx \frac{2.9 - 2}{3.1 - 2} \approx 0.84.$$

However, the bounds suggest faster progress ratios of

$$\left(1 - \frac{\mu}{Ln}\right) \approx 0.76,$$

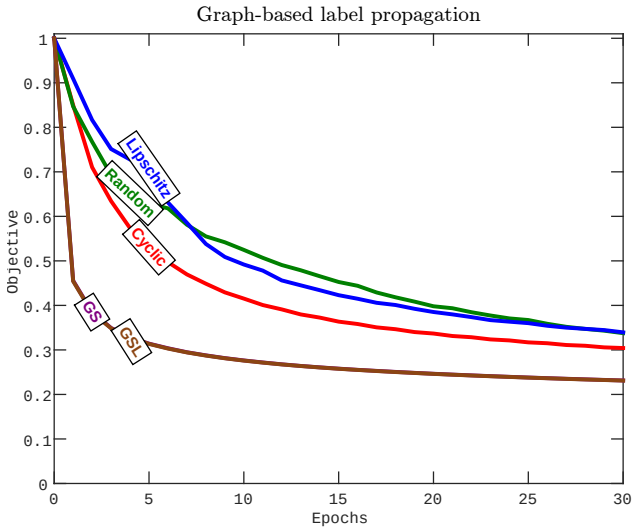

Figure 1: Comparison of coordinate selection rules for graph-based semi-supervised learning.

$$\left(1 - \frac{\mu_1}{L}\right) \approx 0.67,$$

so the GS-r rule does not satisfy either bound. In contrast, in this setting the GS-q rule chooses to update coordinate 2 and obtains $F(x^1) \approx 2.2$, obtaining a progress ratio of

$$\frac{F(x^1) - F(x^*)}{F(x^0) - F(x^*)} \approx \frac{2.2 - 2}{3.1 - 2} \approx 0.16,$$

which satisfies both bounds by a substantial margin. Indeed, we used a genetic algorithm to search for a setting of the parameters of this problem (values of x^0, λ, b, and the diagonals of A) that would make the GS-q not satisfy the bound depending on μ_1, and it easily found counter-examples for the GS-s and GS-r rules but was not able to produce a counter example for the GS-q rule.

9. Experiments on Graph-Based Label-Propagation

Here, we consider an instance of problem h_2, performing label propagation for semi-supervised learning in the 'two moons' dataset [Zhou et al., 2004]. We generate 500 samples from this dataset, randomly label five points in the data, and connect each node to its five nearest neighbours. This high level of sparsity is typical of graph-based methods for semi-supervised learning, and allows the exact Gauss-Southwell rule to be implemented efficiently. We use the quadratic labeling criterion of Bengio et al. [2006], which allows exact coordinate optimization and is normally optimized with cyclic coordinate descent. We plot the performance under different selection rules in Figure 1. Here, we see that even cyclic coordinate descent outperforms randomized coordinate descent, but that the GS and GSL rules give even better performance. We note that the GS and GSL rules perform similarly on this problem since the Lipschitz constants do not vary much.

Runtime Experiments

In Figure 2 we plot the objective against the runtime for the ℓ_2-regularized sparse least squares problem from the main paper. Although runtimes are very sensitive to exact implementation details and we believe that more clever implementations than our naive Python script are possible, this figure does show that the GS and GSL rules offer benefits in terms of runtime with our implementation and test hardware.

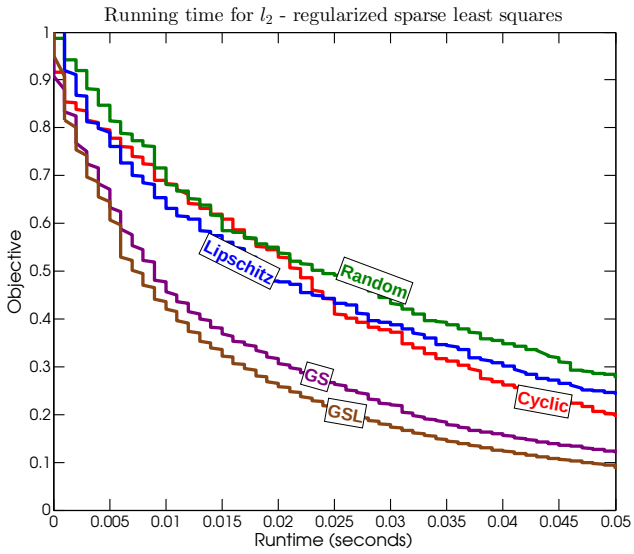

Figure 2: Comparison of coordinate selection rules for ℓ_2-regularized sparse least squares.

References

Y. Bengio, O. Delalleau, and N. Le Roux. Label propagation and quadratic criterion. *Semi-Supervised Learning*, pages 193–216, 2006.

T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. MIT Press Cambridge, second edition, 2001.

I. S. Dhillon, P. K. Ravikumar, and A. Tewari. Nearest neighbor based greedy coordinate descent. *Advances in Neural Information Processing Systems*, 2011.

W. F. Ferger. The nature and use of the harmonic mean. *Journal of the American Statistical Association*, 26(173):36–40, 1931.

Y. Nesterov. *Introductory Lectures on Convex Optimization: A Basic Course*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 2004.

P. Richtárik and M. Takáč. Iteration complexity of randomized block-coordinate descent methods for minimizing a composite function. *Mathematical Programming*, 144:1–38, 2014.

A. Shrivastava and P. Li. Asymmetric LSH (ALSH) for sublinear time maximum inner product search (MIPS). *Advances in Neural Information Processing Systems*, 2014.

D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf. Learning with local and global consistency. *Advances in Neural Information Processing Systems*, 2004.