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

Julie Nutini JNUTINI@CS.UBC.CA
Mark Schmidt SCHMIDTM@CS.UBC.CA
Issam H. Laradji ISSAMOU@CS.UBC.CA
University of British Columbia

Michael Friedlander MPF@MATH.UCDAVIS.EDU
University of California, Davis

Hoyt Koepke HOYTAK@DATO.COM
Dato

Abstract

There has been significant recent work on the theory and application of randomized coordinate descent algorithms, beginning with the work of Nesterov [*SIAM J. Optim., 22(2), 2012*], who showed that a random-coordinate selection rule achieves the same convergence rate as the Gauss-Southwell selection rule. This result suggests that we should never use the Gauss-Southwell rule, as it is typically much more expensive than random selection. However, the empirical behaviours of these algorithms contradict this theoretical result: in applications where the computational costs of the selection rules are comparable, the Gauss-Southwell selection rule tends to perform substantially better than random coordinate selection. We give a simple analysis of the Gauss-Southwell rule showing that—except in extreme cases—it's convergence rate is faster than choosing random coordinates. Further, in this work we (i) show that exact coordinate optimization improves the convergence rate for certain sparse problems, (ii) propose a Gauss-Southwell-Lipschitz rule that gives an even faster convergence rate given knowledge of the Lipschitz constants of the partial derivatives, (iii) analyze the effect of approximate Gauss-Southwell rules, and (iv) analyze proximal-gradient variants of the Gauss-Southwell rule.

Proceedings of the 32^{nd} International Conference on Machine Learning, Lille, France, 2015. JMLR: W&CP volume 37.

1. Coordinate Descent Methods

There has been substantial recent interest in applying coordinate descent methods to solve large-scale optimization problems, starting with the seminal work of Nesterov (2012), who gave the first global rate of convergence analysis for coordinate descent methods for minimizing convex functions. This analysis suggests that choosing a random coordinate to update gives the same performance as choosing the "best" coordinate to update via the more expensive Gauss-Southwell (GS) rule. (Nesterov also proposed a more clever randomized scheme, which we consider later in this paper.) This result gives a compelling argument to use randomized coordinate descent in contexts where the GS rule is too expensive. However, it also suggests that there is no benefit to using the GS rule in contexts where it is relatively cheap. But in these contexts, the GS rule often substantially outperforms randomized coordinate selection in practice. This suggests that either the analysis of GS is not tight, or that there exists a class of functions for which the GS rule is as slow as randomized coordinate descent.

After discussing contexts in which it makes sense to use coordinate descent and the GS rule, we answer this theoretical question by giving a tighter analysis of the GS rule (under strong-convexity and standard smoothness assumptions) that yields the same rate as the randomized method for a restricted class of functions, but is otherwise faster (and in some cases substantially faster). We further show that, compared to the usual *constant* step-size update of the coordinate, the GS method with exact coordinate optimization has a provably faster rate for problems satisfying a certain sparsity constraint (Section 5). We believe that this is the first result showing a theoretical benefit of exact coordinate optimization; all previous analyses show that these

strategies obtain the same rate as constant step-size updates, even though exact optimization tends to be faster in practice. Further, in Section 6, we propose a variant of the GS rule that, similar to Nesterov's more clever randomized sampling scheme, uses knowledge of the Lipschitz constants of the coordinate-wise gradients to obtain a faster rate. We also analyze approximate GS rules (Section 7), which provide an intermediate strategy between randomized methods and the exact GS rule. Finally, we analyze proximal-gradient variants of the GS rule (Section 8) for optimizing problems that include a separable non-smooth term.

2. Problems of Interest

The rates of Nesterov show that coordinate descent can be faster than gradient descent in cases where, if we are optimizing n variables, the cost of performing n coordinate updates is similar to the cost of performing one full gradient iteration. This essentially means that coordinate descent methods are useful for minimizing convex functions that can be expressed in one of the following two forms:

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

where x_i is element i of x, f is smooth and cheap, the f_{ij} are smooth, $G = \{V, E\}$ is a graph, and A is a matrix. (It is assumed that all functions are convex.)[1] The family of functions h_1 includes core machine-learning problems such as least squares, logistic regression, lasso, and SVMs (when solved in dual form) (Hsieh et al., 2008). Family h_2 includes quadratic functions, graph-based label propagation algorithms for semi-supervised learning (Bengio et al., 2006), and finding the most likely assignments in continuous pairwise graphical models (Rue & Held, 2005).

In general, the GS rule for problem h_2 is as expensive as a full gradient evaluation. However, the structure of G often allows efficient implementation of the GS rule. For example, if each node has at most d neighbours, we can track the gradients of all the variables and use a max-heap structure to implement the GS rule in $O(d \log n)$ time (Meshi et al., 2012). This is similar to the cost of the randomized algorithm if $d \approx |E|/n$ (since the average cost of the randomized method depends on the average degree). This condition is true in a variety of applications. For example, in spatial statistics we often use two-dimensional grid-structured graphs, where the maximum degree is four and the average degree is slightly less than 4. As another example, for applying graph-based label propagation on the Facebook graph (to detect the spread of diseases, for example), the average number of friends is around 200 but no user has more than seven thousand friends.[2] The maximum number of friends would be even smaller if we removed edges based on proximity. A non-sparse example where GS is efficient is complete graphs, since here the average degree and maximum degree are both $(n-1)$. Thus, the GS rule is efficient for optimizing dense quadratic functions. On the other hand, GS could be very inefficient for star graphs.

If each column of A has at most c non-zeroes and each row has at most r non-zeroes, then for many notable instances of problem h_1 we can implement the GS rule in $O(cr \log n)$ time by maintaining Ax as well as the gradient and again using a max-heap (see Appendix 2). Thus, GS will be efficient if cr is similar to the number of non-zeroes in A divided by n. Otherwise, Dhillon et al. (2011) show that we can approximate the GS rule for problem h_1 with no g_i functions by solving a nearest-neighbour problem. Their analysis of the GS rule in the convex case, however, gives the same convergence rate that is obtained by random selection (although the constant factor can be smaller by a factor of up to n). More recently, Shrivastava & Li (2014) give a general method for approximating the GS rule for problem h_1 with no g_i functions by writing it as a maximum inner-product search problem.

3. Existing Analysis

We are interested in solving the convex optimization problem

$$\min_{x \in \mathbb{R}^n} f(x), \tag{1}$$

where ∇f is coordinate-wise L-Lipschitz continuous, i.e., for each $i = 1, \ldots, n$,

$$|\nabla_i f(x + \alpha e_i) - \nabla_i f(x)| \leq L|\alpha|, \quad \forall x \in \mathbb{R}^n \text{ and } \alpha \in \mathbb{R},$$

where e_i is a vector with a one in position i and zero in all other positions. For twice-differentiable functions, this is equivalent to the assumption that the diagonal elements of the Hessian are bounded in magnitude by L. In contrast, the typical assumption used for gradient methods is that ∇f is L^f-Lipschitz continuous (note that $L \leq L^f \leq Ln$). The coordinate descent method with constant step-size is based on the iteration

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

The randomized coordinate selection rule chooses i_k uniformly from the set $\{1, 2, \ldots, n\}$. Alternatively, the GS

[1] We could also consider slightly more general cases like functions that are defined on hyper-edges (Richtárik & Takáč, 2015), provided that we can still perform n coordinate updates for a similar cost to one gradient evaluation.

[2] https://recordsetter.com/world-record/facebook-friends

rule

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

chooses the coordinate with the largest directional derivative. Under either rule, because f is coordinate-wise Lipschitz continuous, we obtain the following bound on the progress made by each iteration:

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

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

$$f(y) \geq f(x) + \langle \nabla f(x), y - x \rangle + \frac{\mu}{2}\|y - x\|^2, \quad \forall x, y \in \mathbb{R}^n, \tag{3}$$

which implies that

$$f(x^*) \geq f(x^k) - \frac{1}{2\mu}\|\nabla f(x^k)\|^2, \tag{4}$$

where x^* is the optimal solution of (1). This bound is obtained by minimizing both sides of (3) with respect to y.

3.1. Randomized Coordinate Descent

Conditioning on the σ-field $\mathcal{F}_{k-1}$ generated by the sequence $\{x^0, x^1, \ldots, x^{k-1}\}$, and taking expectations of both sides of (2), when i_k is chosen with uniform sampling we obtain

$$\begin{aligned} \mathbb{E}[f(x^{k+1})] &\leq \mathbb{E}\left[f(x^k) - \frac{1}{2L}\left(\nabla_{i_k} f(x^k)\right)^2\right] \\ &= f(x^k) - \frac{1}{2L}\sum_{i=1}^{n}\frac{1}{n}\left(\nabla_i f(x^k)\right)^2 \\ &= f(x^k) - \frac{1}{2Ln}\|\nabla f(x^k)\|^2. \end{aligned}$$

Using (4) and subtracting $f(x^*)$ from both sides, we get

$$\mathbb{E}[f(x^{k+1})] - f(x^*) \leq \left(1 - \frac{\mu}{Ln}\right)[f(x^k) - f(x^*)]. \tag{5}$$

This is a special of case of Nesterov (2012, Theorem 2) with $\alpha = 0$ in his notation.

3.2. Gauss-Southwell

We now consider the progress implied by the GS rule. By the definition of i_k,

$$(\nabla_{i_k} f(x^k))^2 = \|\nabla f(x^k)\|_\infty^2 \geq (1/n)\|\nabla f(x^k)\|^2. \tag{6}$$

Applying this inequality to (2), we obtain

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

which together with (4), implies that

$$f(x^{k+1}) - f(x^*) \leq \left(1 - \frac{\mu}{Ln}\right)[f(x^k) - f(x^*)]. \tag{7}$$

This is a special case of Boyd & Vandenberghe (2004, §9.4.3), viewing the GS rule as performing steepest descent in the 1-norm. While this is faster than known rates for cyclic coordinate selection (Beck & Tetruashvili, 2013) and holds deterministically rather than in expectation, this rate is the same as the randomized rate given in (5).

4. Refined Gauss-Southwell Analysis

The deficiency of the existing GS analysis is that too much is lost when we use the inequality in (6). To avoid the need to use this inequality, we instead measure strong-convexity in the 1-norm, i.e.,

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

which is the analogue of (3). Minimizing both sides with respect to y, we obtain

$$\begin{aligned} f(x^*) &\geq f(x) - \sup_y\{\langle -\nabla f(x), y - x\rangle - \frac{\mu_1}{2}\|y - x\|_1^2\} \\ &= f(x) - \left(\frac{\mu_1}{2}\|\cdot\|_1^2\right)^*(-\nabla f(x)) \\ &= f(x) - \frac{1}{2\mu_1}\|\nabla f(x)\|_\infty^2, \end{aligned} \tag{8}$$

which makes use of the convex conjugate $(\frac{\mu_1}{2}\|\cdot\|_1^2)^* = \frac{1}{2\mu_1}\|\cdot\|_\infty^2$ (Boyd & Vandenberghe, 2004, §3.3). Using (8) in (2), and the fact that $(\nabla_{i_k} f(x^k))^2 = \|\nabla f(x^k)\|_\infty^2$ for the GS rule, we obtain

$$f(x^{k+1}) - f(x^*) \leq \left(1 - \frac{\mu_1}{L}\right)[f(x^k) - f(x^*)]. \tag{9}$$

It is evident that if $\mu_1 = \mu/n$, then the rates implied by (5) and (9) are identical, but (9) is faster if $\mu_1 > \mu/n$. In Appendix 4, we show that

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

Thus, at one extreme the GS rule obtains the same rate as uniform selection ($\mu_1 \approx \mu/n$). However, at the other extreme, it could be faster than uniform selection by a factor of n ($\mu_1 \approx \mu$). This analysis, that the GS rule only obtains the same bound as random selection in an extreme case, supports the better practical behaviour of GS.

4.1. Comparison for Separable Quadratic

We illustrate these two extremes with the simple example of a quadratic function with a diagonal Hessian $\nabla^2 f(x) = \text{diag}(\lambda_1, \dots, \lambda_n)$. In this case,

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

We prove the correctness of this formula for μ_1 in Appendix 4.1. The parameter μ_1 achieves its lower bound when all λ_i are equal, $\lambda_1 = \cdots = \lambda_n = \alpha > 0$, in which case

$$\mu = \alpha \quad \text{and} \quad \mu_1 = \alpha/n.$$

Thus, uniform selection does as well as the GS rule if all elements of the gradient change at *exactly* the same rate. This is reasonable; under this condition, there is no apparent advantage in selecting the coordinate to update in a clever way. Intuitively, one might expect that the favourable case for the Gauss-Southwell rule would be where one λ_i is much larger than the others. However, in this case, μ_1 is again similar to μ/n. To achieve the other extreme, suppose that $\lambda_1 = \beta$ and $\lambda_2 = \lambda_3 = \cdots = \lambda_n = \alpha$ with $\alpha \geq \beta$. In this case, we have $\mu = \beta$ and

$$\mu_1 = \frac{\beta\alpha^{n-1}}{\alpha^{n-1} + (n-1)\beta\alpha^{n-2}} = \frac{\beta\alpha}{\alpha + (n-1)\beta}.$$

If we take $\alpha \to \infty$, then we have $\mu_1 \to \beta$, so $\mu_1 \to \mu$. This case is much less intuitive; GS is n times faster than random coordinate selection if one element of the gradient changes much more *slowly* than the others. Appendix 4.1 gives a physical interpretation of μ and μ_1 in terms of independent processes 'working together' (Ferger, 1931).

4.2. Fast Convergence with Bias Term

Consider the standard linear-prediction framework,

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

where we have included a bias variable β (an example of problem h_1). Typically, the regularization parameter σ of the bias variable is set to be much smaller than the regularization parameter λ of the other covariates, to avoid biasing against a global shift in the predictor. Assuming that there is no hidden strong-convexity in the sum, this problem has the structure described in the previous section ($\mu_1 \approx \mu$) where GS has the most benefit over random selection.

5. Rates with Different Lipschitz Constants

Consider the more general scenario where we have a Lipschitz constant L_i for the partial derivative of f with respect to each coordinate i,

$$|\nabla_i f(x + \alpha e_i) - \nabla_i f(x)| \leq L_i|\alpha|, \quad \forall x \in \mathbb{R}^n \text{ and } \alpha \in \mathbb{R},$$

and we use a coordinate-dependent step-size at each iteration:

$$x^{k+1} = x^k - \frac{1}{L_{i_k}} \nabla_{i_k} f(x^k) e_{i_k}. \tag{10}$$

By the logic of (2), in this setting we have

$$f(x^{k+1}) \leq f(x^k) - \frac{1}{2L_{i_k}} [\nabla_{i_k} f(x^k)]^2, \tag{11}$$

and thus a convergence rate of

$$f(x^k) - f(x^*) \leq \left[\prod_{j=1}^k \left(1 - \frac{\mu_1}{L_{i_j}}\right)\right] [f(x^0) - f(x^*)]. \tag{12}$$

Noting that $L = \max_i\{L_i\}$, we have

$$\prod_{j=1}^k \left(1 - \frac{\mu_1}{L_{i_j}}\right) \leq \left(1 - \frac{\mu_1}{L}\right)^k. \tag{13}$$

Thus, the convergence rate based on the L_i will be faster, provided that at least one iteration chooses an i_k with $L_{i_k} < L$. In the worst case, however, (13) holds with equality even if the L_i are distinct, as we might need to update a coordinate with $L_i = L$ on every iteration. (For example, consider a separable function where all but one coordinate is initialized at its optimal value, and the remaining coordinate has $L_i = L$.) In Section 6, we discuss selection rules that incorporate the L_i to achieve faster rates whenever the L_i are distinct, but first we consider the effect of exact coordinate optimization on the choice of the L_{i_k}.

5.1. Gauss-Southwell with Exact Optimization

For problems involving functions of the form h_1 and h_2, we are often able to perform exact (or numerically very precise) coordinate optimization, even if the objective function is not quadratic (e.g., by using a line-search or a closed-form update). Note that (12) still holds when using exact coordinate optimization rather than using a step-size of $1/L_{i_k}$, as in this case we have

$$\begin{aligned} f(x^{k+1}) &= \min_\alpha\{f(x^k + \alpha e_{i_k})\} \\ &\leq f\left(x^k - \frac{1}{L_{i_k}} \nabla_{i_i} f(x^k) e_{i_k}\right) \\ &\leq f(x^k) - \frac{1}{2L_{i_k}} [\nabla_{i_k} f(x^k)]^2, \end{aligned} \tag{14}$$

which is equivalent to (11). However, in practice using exact coordinate optimization leads to better performance. In this section, we show that using the GS rule results in a

convergence rate that is indeed faster than (9) for problems with distinct L_i when the function is quadratic, or when the function is not quadratic but we perform exact coordinate optimization.

The key property we use is that, after we have performed exact coordinate optimization, we are guaranteed to have $\nabla_{i_k} f(x^{k+1}) = 0$. Because the GS rule chooses $i_{k+1} = \text{argmax}_i |\nabla_i f(x^{k+1})|$, we cannot have $i_{k+1} = i_k$, unless x^{k+1} is the optimal solution. Hence, we never choose the same coordinate twice in a row, which guarantees that the inequality (13) is strict (with distinct L_i) and exact coordinate optimization is faster. We note that the improvement may be marginal, as we may simply alternate between the two largest L_i values. However, consider minimizing h_2 when the graph is sparse; after updating i_k, we are guaranteed to have $\nabla_{i_k} f(x^{k+m}) = 0$ for all future iterations $(k+m)$ until we choose a variable i_{k+m-1} that is a neighbour of node i_k in the graph. Thus, if the two largest L_i are not connected in the graph, GS cannot simply alternate between the two largest L_i.

By using this property, in Appendix 5.1 we show that the GS rule with exact coordinate optimization for problem h_2 under a chain-structured graph has 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 implication of this result is that, if the large L_i values are more than two edges from each other in the graph, then we obtain a much better convergence rate. We conjecture that for general graphs, we can obtain a bound that depends on the largest value of ρ_2^G among all nodes i and j connected by a path of length 1 or 2. Note that we can obtain similar results for problem h_1, by forming a graph that has an edge between nodes i and j whenever the corresponding variables are both jointly non-zero in at least one row of A.

6. Rules Depending on Lipschitz Constants

If the L_i are known, Nesterov (2012) showed that we can obtain a faster convergence rate by sampling proportional to the L_i. We review this result below and compare it to the GS rule, and then propose an improved GS rule for this scenario. Although in this section we will assume that the L_i are known, this assumption can be relaxed using a backtracking procedure (Nesterov, 2012, §6.1).

6.1. Lipschitz Sampling

Taking the expectation of (11) under the distribution $p_i = L_i / \sum_{j=1}^n L_j$ and proceeding as before, we obtain

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

where $\bar{L} = \frac{1}{n}\sum_{j=1}^n L_j$ is the average of the Lipschitz constants. This was shown by Leventhal & Lewis (2010) and is a special case of Nesterov (2012, Theorem 2) with $\alpha = 1$ in his notation. This rate is faster than (5) for uniform sampling if any L_i differ.

Under our analysis, this rate may or may not be faster than (9) for the GS rule. On the one extreme, if $\mu_1 = \mu/n$ and any L_i differ, then this Lipschitz sampling scheme is faster than our rate for GS. Indeed, in the context of the problem from Section 4.1, we can make Lipschitz sampling faster than GS by a factor of nearly n by making one λ_i much larger than all the others (recall that our analysis shows no benefit to the GS rule over randomized selection when only one λ_i is much larger than the others). At the other extreme, in our example from Section 4.1 with many large α and one small β, the GS and Lipschitz sampling rates are the same when $n = 2$, with a rate of $(1 - \beta/(\alpha+\beta))$. However, the GS rate will be faster than the Lipschitz sampling rate for any $\alpha > \beta$ when $n > 2$, as the Lipschitz sampling rate is $(1 - \beta/((n-1)\alpha + \beta))$, which is slower than the GS rate of $(1-\beta/(\alpha+(n-1)\beta))$.

6.2. Gauss-Southwell-Lipschitz Rule

Since neither Lipschitz sampling nor GS dominates the other in general, we are motivated to consider whether faster rules are possible by combining the two approaches. Indeed, we obtain a faster rate by choosing the i_k that minimizes (11), leading to the 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. Following a similar argument to Section 4, but using (11) in place of (2), the GSL rule obtains a convergence rate of

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

where μ_L is the strong-convexity constant with respect to the norm $\|x\|_L = \sum_{i=1}^n \sqrt{L_i}|x_i|$. This is shown in Appendix 6.2, where we also show that

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

Thus, the GSL rule is always at least as fast as the fastest of the GS rule and Lipschitz sampling. Indeed, it can be

more than a factor of n faster than using Lipschitz sampling, while it can obtain a rate closer to the minimum L_i, instead of the maximum L_i that the classic GS rule depends on.

An interesting property of the GSL rule for quadratic functions is that it is the *optimal* myopic coordinate update. That is, if we have an oracle that can choose the coordinate and the step-size that decreases f by the largest amount,

$$f(x^{k+1}) = \underset{i,\alpha}{\text{argmin}}\{f(x^k + \alpha e_i)\}, \tag{15}$$

this is equivalent to using the GSL rule and the update in (10). This follows because (11) holds with equality in the quadratic case, and the choice $\alpha_k = 1/L_{i_k}$ yields the optimal step-size. Thus, although faster schemes could be possible with non-myopic strategies that cleverly choose the sequence of coordinates or step-sizes, if we can only perform one iteration, then the GSL rule cannot be improved.

For general f, (15) is known as the *maximum improvement* (MI) rule. This rule has been used in the context of boosting (Rätsch et al., 2001), graphical models (Della Pietra et al., 1997; Lee et al., 2006; Scheinberg & Rish, 2009), Gaussian processes (Bo & Sminchisescu, 2008), and low-rank tensor approximations (Li et al., 2015). Using an argument similar to (14), our GSL rate also applies to the MI rule, improving existing bounds on this strategy. However, the GSL rule is much cheaper and does not require any special structure (recall that we can estimate L_i as we go).

A further interesting property of the GSL rule is that it has a stronger connection to the nearest neighbour problem than the GS rule. In particular, in Appendix 6.2 we show that under weak conditions the GSL rule is *equivalent* to a *normalized* nearest neighbour problem for the standard empirical risk minimization framework with a linear predictor, $F(x) = \sum_{i=1}^n f(a_i^T x)$, for a twice-differentiable loss f. This includes problems like least squares and logistic regression, and note that this equivalence is not true for the classic GS rule. Surprisingly, this strategy allows us to compute the GSL rule in this context *even if we do not the know the* L_i.

7. Approximate Gauss-Southwell

In many applications, computing the exact GS rule is too inefficient to be of any practical use. However, a computationally cheaper *approximate* GS rule might be available. Approximate GS rules under multiplicative and additive errors were considered by Dhillon et al. (2011) in the convex case, but in this setting the convergence rate is similar to the rate achieved by random selection. In this section, we give rates depending on μ_1 for approximate GS rules.

7.1. Multiplicative Errors

In the multiplicative error regime, the approximate GS rule chooses an i_k satisfying

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

for some $\epsilon_k \in [0, 1)$. In this regime, our basic bound on the progress (2) still holds, as it was defined for any i_k. We can incorporate this type of error into our lower bound (8) to obtain

$$\begin{aligned} f(x^*) &\geq f(x^k) - \frac{1}{2\mu_1}\|\nabla f(x^k)\|_\infty^2 \\ &\geq f(x^k) - \frac{1}{2\mu_1(1-\epsilon_k)^2}|\nabla_{i_k} f(x^k)|^2. \end{aligned}$$

This implies a convergence rate of

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

Thus, the convergence rate of the method is nearly identical to using the exact GS rule for small ϵ_k (and it degrades gracefully with ϵ_k). This is in contrast to having an error in the gradient (Friedlander & Schmidt, 2012), where the error ϵ must decrease to zero over time.

7.2. Additive Errors

In the additive error regime, the approximate GS rule chooses an i_k satisfying

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

for some $\epsilon_k \geq 0$. In Appendix 7.2, we show that under this rule, we have

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

where

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

where L_1 is the Lipschitz constant of ∇f with respect to the 1-norm. However, note that L_1 could be substantially larger than L, so in Appendix 7.2 we also give a bound with a worse dependence on ϵ_k that does not rely on L_1. This regime is closer to the case of having an error in the gradient, as to obtain convergence the ϵ_k must decrease to zero. This result implies that a sufficient condition for the algorithm to obtain a linear convergence rate is that the errors ϵ_k converge to zero at a linear rate. Further, if the errors satisfy $\epsilon_k = O(\rho^k)$ for some $\rho < (1 - \mu_1/L)$, then the convergence rate of the method is the same as if we used an exact GS rule. On the other hand, if ϵ_k does not decrease to zero, we may end up repeatedly updating the same wrong coordinate and the algorithm will not converge (though we could switch to the randomized method if this is detected).

8. Proximal-Gradient Gauss-Southwell

One of the key motivations for the resurgence of interest in coordinate descent methods is their performance on problems of the form

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

where f is smooth and convex and the g_i are convex, but possibly non-smooth. This includes problems with ℓ_1-regularization, and optimization with lower and/or upper bounds on the variables. Similar to proximal-gradient methods, we can apply the proximal operator to the coordinate 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

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

With random coordinate selection, Richtárik & Takáč (2014) show that this method has a convergence rate of

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

similar to the unconstrained/smooth case.

There are several generalizations of the GS rule to this scenario. Here we consider three possibilities, all of which are equivalent to the GS rule if the g_i are not present. First, the GS-s rule chooses the coordinate with the most negative directional derivative. This strategy is popular for ℓ_1-regularization (Shevade & Keerthi, 2003; Wu & Lange, 2008; Li & Osher, 2009) and in general is given by (see Bertsekas, 1999, §8.4)

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

However, the length of the step ($\|x^{k+1} - x^k\|$) could be arbitrarily small under this choice. In contrast, the GS-r rule chooses the coordinate that maximizes the length of the step (Tseng & Yun, 2009; Dhillon et al., 2011),

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

This rule is effective for bound-constrained problems, but it ignores the change in the non-smooth term ($g_i(x_i^{k+1}) - g_i(x_k^k)$). Finally, the GS-$q$ rule maximizes progress assuming a quadratic upper bound on f (Tseng & Yun, 2009),

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

While the least intuitive rule, the GS-q rule seems to have the best theoretical properties. Further, if we use L_i in place of L in the GS-q rule (which we call the GSL-q strategy), then we obtain the GSL rule if the g_i are not present. In contrast, using L_i in place of L in the GS-r rule (which we call the GSL-r strategy) does not yield the GSL rule as a special case.

In Appendix 8, we show that using the GS-q rule yields a convergence rate of

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

where ϵ_k is bounded above by a measure of the non-linearity of the g_i along the possible coordinate updates. Note that ϵ_k goes to zero as k increases and we conjecture that the above bound holds with $\epsilon_k = 0$. In contrast, in Appendix 8 we show that the above rate does not hold with $\epsilon_k = 0$ for the GS-s or GS-r rule, even if you replace the minimum by a maximum. Thus, any bounds for the GS-s and GS-r rules would be slower than the expected rate under random selection, while the GS-q rule leads to a better bound.

9. Experiments

We compared the efficacy of different coordinate selection rules on the following simple instances of h_1. In Appendix 9, we report experimental results on an instance of h_2.

ℓ_2-regularized sparse least squares: Here we consider the problem

$$\min_x \frac{1}{2n}\|Ax - b\|^2 + \frac{\lambda}{2}\|x\|^2,$$

an instance of problem h_1. We set A to be an m by n matrix with entries sampled from a $\mathcal{N}(0,1)$ distribution (with $m = 1000$ and $n = 1000$). We then added 1 to each entry (to induce a dependency between columns), multiplied each column by a sample from $\mathcal{N}(0,1)$ multiplied by ten (to induce different Lipschitz constants across the coordinates), and only kept each entry of A non-zero with probability $10\log(n)/n$ (a sparsity level that allows the Gauss-Southwell rule to be applied with cost $O(\log^3(n))$. We set $\lambda = 1$ and $b = Ax + e$, where the entries of x and e were drawn from a $\mathcal{N}(0,1)$ distribution. In this setting, we used a step-size of $1/L_i$ for each coordinate i, which corresponds to exact coordinate optimization.

ℓ_2-regularized sparse logistic regression: Here we consider the problem

$$\min_x \frac{1}{n}\sum_{i=1}^{n}\log(1 + \exp(-b_i a_i^T x)) + \frac{\lambda}{2}\|x\|^2.$$

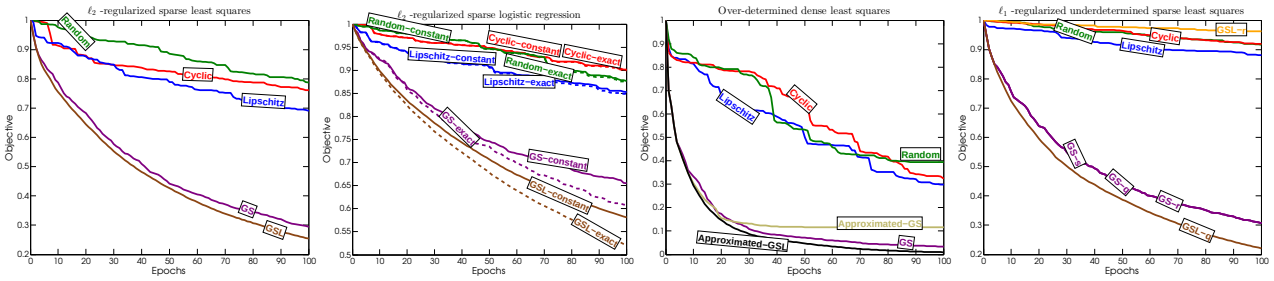

Figure 1. Comparison of coordinate selection rules for 4 instances of problem h_1.

We set the a_i to be the rows of A from the previous problem, and set $b = \text{sign}(Ax)$, but randomly flipping each b_i with probability 0.1. In this setting, we compared using a step-size of $1/L_i$ to using exact coordinate optimization.

Over-determined dense least squares: Here we consider the problem

$$\min_x \frac{1}{2n}\|Ax - b\|^2,$$

but, unlike the previous case, we do not set elements of A to zero and we make A have dimension 1000 by 100. Because the system is over-determined, it does not need an explicit strongly-convex regularizer to induce global strong-convexity. In this case, the density level means that the exact GS rule is not efficient. Hence, we use a balltree structure (Omohundro, 1989) to implement an efficient approximate GS rule based on the connection to the nearest neighbour problem discovered by Dhillon et al. (2011). On the other hand, we can compute the exact GSL rule for this problem as a nearest neighbour problem.

ℓ_1-regularized underdetermined sparse least squares: Here we consider the non-smooth problem

$$\min_x \frac{1}{2n}\|Ax - b\|^2 + \lambda\|x\|_1.$$

We generate A as we did for the ℓ_2-regularized sparse least squares problem, except with the dimension 1000 by 10000. This problem is not globally strongly-convex, but will be strongly-convex along the dimensions that are non-zero in the optimal solution.

We plot the objective function (divided by its initial value) of coordinate descent under different selection rules in Figure 1. Even on these simple datasets, we see dramatic differences in performance between the different strategies. In particular, the GS rule outperforms random coordinate selection (as well as cyclic selection) by a substantial margin in all cases. The Lipschitz sampling strategy can narrow this gap, but it remains large (even when an approximate GS rule is used). The difference between GS and randomized selection seems to be most dramatic for the ℓ_1-regularized problem; the GS rules tend to focus on the non-zero variables while most randomized/cyclic updates focus on the zero variables, which tend not to move away from zero.[3] Exact coordinate optimization and using the GSL rule seem to give modest but consistent improvements. The three non-smooth GS-$*$ rules had nearly identical performance despite their different theoretical properties. The GSL-q rule gave better performance than the GS-$*$ rules, while the the GSL-r variant performed worse than even cyclic and random strategies. We found it was also possible to make the GS-s rule perform poorly by perturbing the initialization away from zero. While these experiments plot the performance in terms of the number of iterations, in Appendix 9 we show that the GS-$*$ rules can also be advantageous in terms of runtime.

10. Discussion

It is clear that the GS rule is not practical for every problem where randomized methods are applicable. Nevertheless, we have shown that even approximate GS rules can obtain better convergence rate bounds than fully-randomized methods. We have given a similar justification for the use of exact coordinate optimization, and we note that our argument could also be used to justify the use of exact coordinate optimization within randomized coordinate descent methods (as used in our experiments). We have also proposed the improved GSL rule, and considered approximate/proximal variants. We expect our analysis also applies to block updates by using mixed norms $\|\cdot\|_{p,q}$, and could be used for accelerated/parallel methods (Fercoq & Richtárik, 2013), for primal-dual rates of dual coordinate ascent (Shalev-Shwartz & Zhang, 2013), for successive projection methods (Leventhal & Lewis, 2010), for boosting algorithms (Rätsch et al., 2001), and for scenarios without strong-convexity under general error bounds (Luo & Tseng, 1993).

[3] To reduce the cost of the GS-s method in this context, Shevade & Keerthi (2003) consider a variant where we first compute the GS-s rule for the non-zero variables and if an element is sufficiently large then they do not consider the zero variables .

Acknowledgements

We would like to thank the anonymous referees for their useful comments that significantly improved the paper. Julie Nutini is funded by an NSERC Canada Graduate Scholarship.

References

Beck, A. and Tetruashvili, L. On the convergence of block coordinate descent type methods. *SIAM Journal on Optimization*, 23(4):2037–2060, 2013.

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

Bertsekas, D. P. *Nonlinear Programming*. Athena Scientific, second edition, 1999.

Bo, L. and Sminchisescu, C. Greedy block coordinate descent for large scale gaussian process regression. *Uncertainty in Artificial Intelligence*, 2008.

Boyd, S. P. and Vandenberghe, L. *Convex Optimization*. Cambridge University Press, 2004.

Della Pietra, S., Della Pietra, V., and Lafferty, J. Inducing features of random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(4):380–393, 1997.

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

Fercoq, O. and Richtárik, P. Accelerated, parallel and proximal coordinate descent. *arXiv:1312.5799*, 2013.

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

Friedlander, M. P. and Schmidt, M. Hybrid deterministic-stochastic methods for data fitting. *SIAM Journal on Scientific Computing*, 34(3):A1380–A1405, 2012.

Hsieh, C.-J., Chang, K.-W., Lin, C.-J., Keerthi, S. S., and Sundararajan, S. A dual coordinate descent method for large-scale linear SVM. *International Conference on Machine Learning*, 2008.

Lee, S.-I., Ganapathi, V., and Koller, D. Efficient structure learning of Markov networks using ℓ_1-regularization. *Advances in Neural Information Processing Systems*, 2006.

Leventhal, D. and Lewis, A. S. Randomized methods for linear constraints: convergence rates and conditioning. *Mathematics of Operations Research*, 35(3):641–654, 2010.

Li, Y. and Osher, S. Coordinate descent optimization for ℓ_1 minimization with application to compressed sensing; a greedy algorithm. *Inverse Problems and Imaging*, 3(3): 487–503, 2009.

Li, Z., Uschmajew, A., and Zhang, S. On convergence of the maximum block improvement method. *SIAM Journal on Optimization*, 25(1):210–233, 2015.

Luo, Z.-Q. and Tseng, P. Error bounds and convergence analysis of feasible descent methods: a general approach. *Annals of Operations Research*, 46(1):157–178, 1993.

Meshi, O., Jaakkola, T., and Globerson, A. Convergence rate analysis of MAP coordinate minimization algorithms. *Advances in Neural Information Processing Systems*, 2012.

Nesterov, Y. Efficiency of coordinate descent methods on huge-scale optimization problems. *SIAM Journal on Optimization*, 22(2):341–362, 2012.

Omohundro, S. M. Five balltree construction algorithms. Technical report, International Computer Science Institute, Berkeley, 1989.

Rätsch, G., Mika, S., and Warmuth, M. K. On the convergence of leveraging. *Advances in Neural Information Processing Systems*, 2001.

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

Richtárik, P. and Takáč, M. Parallel coordinate descent methods for big data optimization. *Mathematical Programming*, pp. 1–52, 2015.

Rue, H. and Held, L. *Gaussian Markov Random Fields: Theory and Applications*. CRC Press, 2005.

Scheinberg, K. and Rish, I. SINCO - a greedy coordinate ascent method for sparse inverse covariance selection problem. *Optimization Online*, 2009.

Shalev-Shwartz, S. and Zhang, T. Stochastic dual coordinate ascent methods for regularized loss minimization. *Journal of Machine Learning Research*, 14:567–599, 2013.

Shevade, S. K. and Keerthi, S. S. A simple and efficient algorithm for gene selection using sparse logistic regression. *Bioinformatics*, 19(17):2246–2253, 2003.

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

Tseng, P. and Yun, S. A coordinate gradient descent method for nonsmooth separable minimization. *Mathematical Programming*, 117:387–423, 2009.

Wu, T. T. and Lange, K. Coordinate descent algorithms for lasso penalized regression. *The Annals of Applied Statistics*, 2(1):224–244, 2008.