LECTURE NOTES ON CONVEX CONSTRAINED OPTIMIZATION AND COORDINATE DESCENT [*]

A. A. ERGÜR AND L. A. HOLLAND

Abstract. This scribe contains lecture notes on foundational methods in convex constrained optimization, with an emphasis on projection-based and projection-free algorithms. We begin by introducing cones and descent cones to formulate optimality conditions in constrained problems. We then present projected gradient descent and characterize its convergence behavior. To address projection inefficiencies, we introduce the Frank-Wolfe algorithm and analyze its performance. The second half of the lecture focuses on coordinate descent methods — including their implementation in empirical risk minimization settings — and provides a detailed convergence analysis in both convex and strongly convex regimes. We conclude by discussing importance sampling and Nesterov-style acceleration for coordinate descent.

Key words. Convex optimization, constrained optimization, projected gradient descent, Frank-Wolfe algorithm, coordinate descent, empirical risk minimization, Lipschitz continuity, strong convexity, accelerated methods

AMS subject classifications. safely ignore

1. Cones and Descent Cones.

1.1. Definition of a Cone.

Let us consider a geometric object known as a **cone**. Formally, a set $C \subseteq \mathbb{R}^n$ is a cone if for any $x_1, x_2 \in C$ and any non-negative scalars $\lambda_1, \lambda_2 \geq 0$, the combination $\lambda_1 x_1 + \lambda_2 x_2$ is also in C. A familiar example is an ice cream cone, in either two or three dimensions. Any point inside the cone, when stretched or scaled positively, remains within the cone. The same holds for linear combinations of such vectors.

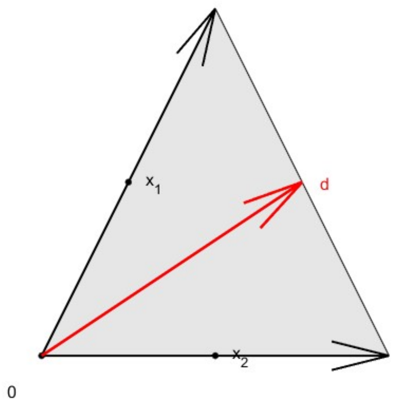

FIG. 1. *A convex cone generated by directions x_1 and x_2 from the origin. The shaded region represents all nonnegative combinations of the two vectors. The red arrow, $d = 0.5x_1 + 0.5x_2$, lies strictly inside the cone and illustrates a valid descent direction in convex optimization. Adapted from Boyd and Vandenberghe's Convex Optimization lecture slides [1].*

1.2. Descent Cones and Optimality.

Now, consider a convex optimization problem where we are minimizing a function $f(x)$ over a convex set $\Omega \subseteq \mathbb{R}^n$. Let

[*]We thank Robbins family for supporting the Algorithmic Foundations of Data Science Course

$x \in \Omega$ be a point of interest. The **descent cone** at **x**, denoted $N_\Omega(x)$, is the set of all vectors $v \in \mathbb{R}^n$ such that for all $y \in \Omega$, the inequality

$$v^T(y - x) \leq 0$$

holds. Geometrically, this cone characterizes the directions from which no further descent is possible.

Theorem: let $f : \Omega \to \mathbb{R}$ be differentiable and Ω be a closed convex set.

1. If $x^* = \arg\min_{x\in\Omega} f(x)$, then $-\nabla f(x^*) \in N_\Omega(x^*)$.
2. If f is convex, $-\nabla f(x^*) \in N_\Omega(x^*)$ implies $x^* = \arg\min_{x\in\Omega} f(x)$
3. If f is strongly convex, then x^* is the unique global minimizer.

This highlights the geometric interpretation of optimality: at the local minimum x^*, the negative gradient vector must lie within the descent cone at that point. This means no feasible direction exists at x^*.

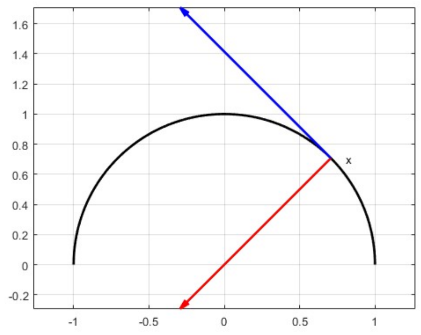

FIG. 2. *At a boundary point of a convex set, the blue arrow represents a feasible direction in the tangent cone, and the red arrow represents a vector in the normal cone. The optimality condition $-\nabla f(x^*) \in N_\Omega(x^*)$ geometrically implies that the gradient must lie within the normal cone.*

2. Projection Onto Convex Sets.

In constrained optimization, it is often necessary to project points back onto the feasible set after taking an update step.

2.1. Definition of Projection.

An important operation in constrained optimization is **projection**. Given a point $z \in \mathbb{R}^n$ and a convex set Ω, the projection $P_\Omega(z)$ is defined as:

$$P_\Omega(z) := \arg\min_{x\in\Omega} \|z - x\|^2$$

This is the point in Ω that is closest to z in the Euclidean sense. When Ω is convex, the objective function $\|z - x\|^2$ is strongly convex, and thus the projection exists and is unique.

2.2. Projection Characterization.

A key characterization of projections onto convex sets is:

$$x = P_\Omega(z) \leftrightarrow x - z \in N_\Omega(x)$$

That is, the residual vector $x - z$ must lie within the descent cone at x. This also leads to the inequality:

$$(P_\Omega(z) - y)^T(z - P_\Omega(z)) \geq 0, \forall y \in \Omega$$

This variational inequality confirms that $P_\Omega(z)$ is indeed the closest feasible point to z [1].

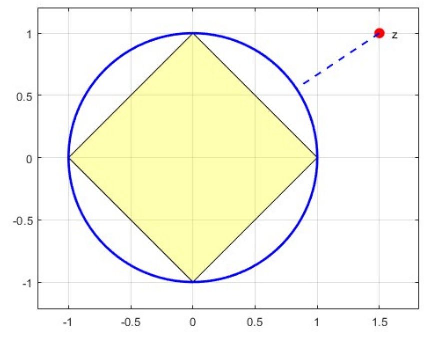

Fig. 3. *Geometric visualization of projection onto different convex sets: the unit ℓ_2 ball (blue circle) and the ℓ_1 ball (yellow diamond). The red point z lies outside both sets, and its projection onto the ℓ_2 ball is shown with a dashed arrow. This illustrates the importance of projection geometry in constrained optimization.*

3. Projected Gradient Descent.

Now, we present a natural algorithm for constrained optimization: **projected gradient descent**. Standard gradient descent may take steps that violate constraints. To maintain feasibility, we modify the update rule by projecting back to Ω.

3.1. Algorithm.

Let $f : \mathbb{R}^n \to \mathbb{R}$ be differentiable and convex. Let $\Omega \subseteq \mathbb{R}^n$ be a closed convex set. The update rule is:

$$x_{k+1} = P_\Omega(x_k - \alpha_k \nabla f(x_k))$$

This ensures that each iterate remains feasible $x_{k+1} \in \Omega$

3.2. Termination and Optimality.

For a point $x^* \in \Omega$, the following are equivalent:

- $x^* = \arg\min_{x\in\Omega} f(x)$
- $-\nabla f(x^*) \in N_\Omega(x^*)$
- $P_\Omega(x^* - \lambda \nabla f(x^*)) = x^*, \forall \lambda 0$

This leads to a termination condition: if the projected step does not move to the point, i.e.,

$$P_\Omega(x_k - \alpha_k \nabla f(x_k)) = x_k$$

then x_k is optimal (assuming f is convex).

3.3. Example: Optimization Over the Probability Simplex.

Consider the probability simplex:

$$\Delta_n := \{x \in \mathbb{R}^n : x_i \geq 0, \sum_{i=1}^{n} x_i = 1\}$$

This is a common feasible set in applications involving probability distributions. Suppose we wish to minimize:

$$f(x) = \langle x, c\rangle^2$$

Gradient descent alone may yield infeasible (e.g., negative) values. Projected gradient descent ensures that each step respects the simplex constraint, making it particularly suitable for applications in machine learning and statistics.

4. The Frank-Wolfe Algorithm.

The **Frank-Wolfe algorithm**, also known as the **conditional gradient method**, is an alternative to projected gradient descent for constrained convex optimization [2]. It is particularly effective when feasible region $K \subseteq \mathbb{R}^n$ is convex and has a structure that makes linear minimization easier than projection.

4.1. Problem Setup.

We aim to solve:

$$\min_{x \in K)} f(x)$$

where:

- $f : \mathbb{R}^n \to \mathbb{R}$ is convex and differentiable
- K is a compact convex set

4.2. Algorithm Steps.

At each iteration k, the Frank-Wolfe algorithm performs the following steps:

1. **Linear Minimization Setup**
   Compute:
   $$\overline{x}_{k+1} := \arg\min_{x \in K} \nabla f(x_k)^T x$$
   This step finds a feasible direction that minimizes the linear approximation of f at x_k.
2. **Update Step**
   Update the iterate using:
   $$x_{k+1} = (1 - \alpha_k)x_k + \alpha_k \overline{x}_k$$
   or equivalently:
   $$x_{k+1} = x_k + \alpha_k(\overline{x}_k - x_k)$$
   Where $\alpha_k \in [0, 1]$ is a step size (typically $\alpha_k = \frac{2}{k+2}$

This avoids the need for a potentially expensive projection step.

4.3. Convergence Guarantee.

We can guarantee sublinear convergence under standard assumptions. **Theorem**: Let f be convex and L-smooth (i.e., ∇f is Lipschitz continuous). Let

$$D := \max_{x,y \in K)} \|x - y\|$$

Then using step size $\alpha_k = \frac{2}{k+2}$, we have:

$$f(x_k) - f(x^*) \leq \frac{2LD^2}{k+2}$$

So to achieve accuracy ϵ, we require:

$$t \geq \frac{2LD^2}{\epsilon}$$

4.4. Illustrative Example.

Let the feasible set be:

$$K = conv\{a_!, a_2, \ldots, a_N\}, a_i \in \mathbb{R}^2$$

and consider the problem:

$$\min_{x \in K} f(x) = \frac{1}{2}\|x - z\|^2$$

This is simply projecting z onto the convex hull K, but using Frank-Wolfe instead of orthogonal projection.

- The gradient $\nabla f(x) = x - z$
- The linear minimization step becomes:

$$\overline{x}_k = \arg\min_{a \in K} \langle x_k - z, a \rangle$$

This is solved by selecting the vertex $a_i \in K$ that minimizes the inner product. The Frank-Wolfe iteration then becomes:

$$x_{k+1} = (1 - \alpha_k)x_k + \alpha_k a_i$$

which converges to the projection of z onto K.

5. Coordinate Descent.

Coordinate descent is especially effective for composite objectives with separable regularization terms [5]. At each step, it updates only one coordinate of the variable vector, keeping the others fixed. This makes each step computationally cheap and efficient, especially when the objective function is separable or has a structure that supports efficient partial updates.

5.1. The Basic Algorithm.

Let $f : \mathbb{R}^n \to \mathbb{R}$ be a (typically convex) differentiable function. At iteration k, we select a coordinate $i_k \in \{1, \dots, n\}$, and perform the update:

$$x_{k+1} = x_k - \alpha_k (\nabla f(x))_{i_k}$$

or

$$x_{k+1} = x_k - \alpha_k \cdot \frac{\partial f}{\partial x_{i_k}} \cdot e_{i_k}$$

This can be interpreted as performing one dimensional descent along a single axis at each step

5.2. Why Coordinate Descent?.

- **Simplicity:** The method is simple to implement.
- **Low Per-Iteration Cost:** Only one partial derivative a one dimensional update are needed
- **Suitability for High Dimensions:** In problems with huge n, full gradients are expensive. Coordinate updates avoid this.
- **Practical Efficiency:** Especially effective when the cost of computing full gradients is high or data is sparse.

6. Coordinate-wise Optimality.

Now we ask the question: if at a point $x \in \mathbb{R}^n$, the function $f(x)$ cannot be improved by adjusting any individual coordinate, is x a global minimizer?

6.1. Case 1: f is Convex and Differentiable.

Suppose $f : \mathbb{R}^n \to \mathbb{R}$ is convex and differentiable, and for all $i \in \{1, \dots, n\}$ and all scalars δ,

$$f(x + \delta e_i) \geq f(x)$$

Where $e_i = (0, \dots, 1, \dots, 0) \in \mathbb{R}$, the ith standard basis vector. This means no descent direction exists along any coordinate. Then:

$$\frac{\partial f}{\partial x_i} = 0, \forall i \to \nabla f(x) = 0$$

Since f is convex, we conclude that x is a global minimizer.

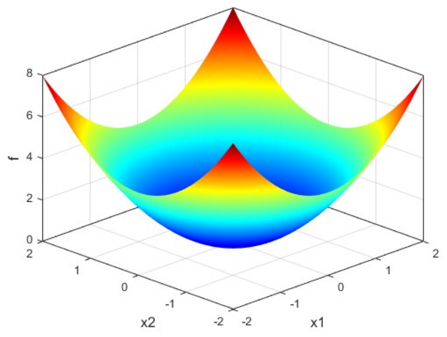

FIG. 4. *If f is convex and differentiable, and minimized along each coordinate direction, then x is a global minimizer. Adapted from Tibshirani [4].*

6.2. Case 2: f is Convex but Not Differentiable. Now suppose f is convex, but possibly not differentiable. Can coordinate-wise optimality still guarantee global optimality? **No**.

There are convex, non-differentiable functions where all coordinate-wise updates fail to improve the function, yet the point is not a global minimizer. These functions have subgradients that can allow descent in directions outside the coordinate axes.

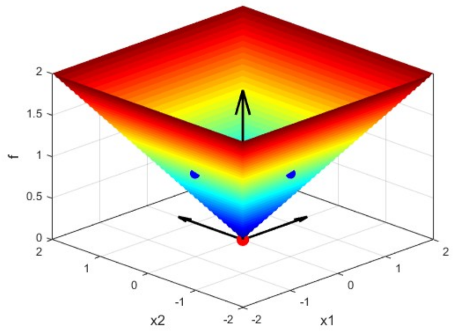

FIG. 5. *A convex but non-differentiable function $f(x_1, x_2) = \max(| x_1 |, | x_2 |)$. The red dot marks the global minimizer at the origin. The blue dots represent local minima along the coordinate directions when the other variable is held fixed. Despite minimizing along each coordinate, the process does not necessarily converge to the global minimizer, demonstrating that coordinate-wise minimization does not imply global optimality in the absence of differentiability. Adapted from Tibshirani [4].*

6.3. Case 3: Separable Non-Differentiable Convex Functions. Now assume the composite structure:

$$f(x) = g(x) + \sum_{i=1}^{n} h_i(x_i)$$

where:

- g is convex and differentiable

- Each h_i os convex
- The h_i terms are separable

Claim: If x is coordinate-wise optimal, then x is a global minimizer. **Proof**: For any $y \in \mathbb{R}^n$,

$$f(y) - f(x) = g(y) - g(x) + \sum i = 1n[h_i(y_i) - h_i(x_i)]$$

By convexity:

$$g(y) - g(x) \geq \nabla g(x)^T(y - x) = \sum_{i=1}^{n} \nabla_i g(x)(y_i - x_i)$$

Therefore:

$$f(y) - f(x) \geq \sum_{i=1}^{n} [\nabla_i g(x)(y_i - x_i) + h_i(y_i) - h_i(x_i)]$$

Since each term is non-negative due to coordinate-wise optimality, the whole sum is non-negative.

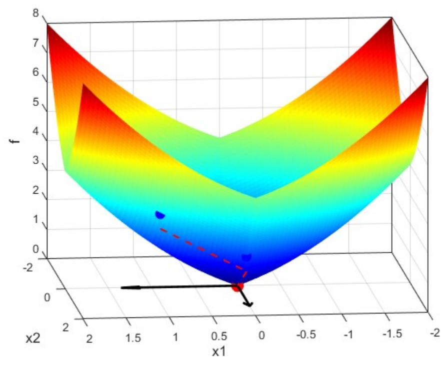

FIG. 6. *Visualization of a composite convex function of the form $f(x) = g(x) + \sum_i h_i(x_i)$, where $g(x) = \frac{1}{2}(x_1^2 + x_2^2)$ is convex and differentiable, and $h_i(x_i) = |x_i|$ are convex but non-differentiable. The red point denotes the global minimizer at the origin. Blue points indicate coordinate-wise minimization steps, and the dashed red lines show the descent path. Despite non-smoothness, the structure of the function ensures that coordinate-wise optimality implies global optimality. Adapted from Tibshirani [4].*

7. Empirical Risk Minimization.

Many large-scale optimization problems in machine learning are instances of **Empirical Risk Minimization** (ERM). The general ERM objective balances a data fidelity term with a regularization term and can be written as:

$$f(x) = \frac{1}{N} \sum_{j=1}^{N} f_j(A_j x) + \lambda \sum_{i=1}^{n} g_i(x_i)$$

Where:

- $x \in \mathbb{R}^n$ is the parameter we wish to optimize,
- $A \in \mathbb{R}^{N \times n}$ is the data matrix with rows $a_j \in \mathbb{R}^n$
- f_j are convex loss functions
- g_i are convex regularization terms

This form naturally arises in supervised learning problems, especially for classification and regression tasks.

7.1. Gradient Computation. Assuming all f_j and g_i are differentiable, we can compute the gradients via the chain rule:

- For a single term $f_j(A_j x)$ the gradient is:

$$(\nabla(f_j(A_j x)))_u = \langle A_j, \nabla f_j(x)\rangle_{A_{ju}}$$

- Aggregating over all samples an adding regularization, we get the full gradient:

$$(\nabla f(x))_i = \frac{1}{N}(\sum_{j=1}^{N} A_{ji} \cdot \nabla f_j(x)) + \lambda g_i'(x_i)$$

Thus, each coordinate i of the gradient depends only on the ith column of A and the current value x_i.

7.2. Coordinate Descent Update. In coordinate descent, we update only one coordinate i_k at each step:

$$x_{k+1} = x_k - \alpha_k (\nabla f(x_k))_{i_k} e_{i_k}$$

Where e_{i_k} is the i_kth unit vector. This means that each iteration only requires computing a partial gradient, which is computationally cheaper than computing the full gradient.

- **Backpropagation cost:** Only need to backpropagate through one f_j
- **Regularization:** $g_i'(x_i)$ is typically simple

7.3. Covariance Estimation via Maximum Likelihood. Suppose we observe samples $x_1, \dots, x_N \in \mathbb{R}^n$, drawn from a multivariate Gaussian $\mathcal{N}(0, \Sigma)$. The goal is to estimate the inverse covariance matrix $\Theta = \Sigma^{-1}$. The log-likelihood of the data is:

$$\log p(X \mid \Sigma) = -\log\det(\Theta^{-1}) + \langle X^T X, \Theta\rangle$$

Thus, the negative log-likelihood plus regularization leads to the objective:

$$\min_{\Theta \succeq 0} [-\log\det(\Theta) + \langle X^T X, \Theta\rangle + \lambda\|\Theta\|_2^2]$$

If we seek sparse estimates, we instead use:

$$\min_{\Theta \succeq 0} [-\log\det(\Theta) + \langle X^T X, \Theta\rangle + \lambda\|\Theta\|_1]$$

This is known as the **Graphical Lasso**, and it fits the ERM + sparsity regularization template. Coordinate descent is commonly used to solve such problems.

8. General Sparse Optimization. A common structure in high-dimensional optimization, particularly in graphical models and signal processing, is a sum over local terms:

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

Where:

- $G = (V, E)$ is a graph,
- f_{ij} is a convex function coupling variables x_i and x_j,
- g_i is a convex regularizer on individual coordinates.

This structure is general enough to cover:

- Total variation denoising,
- Graph-structured Lasso,
- Pairwise Markov Random Field (MRF) models

9. Convergence Analysis of Coordinate Descent.

We now analyze how and when coordinante descent converges, especially in a random setting. Suppose:

- $f : \Omega \to \mathbb{R}$ is convex and differentiable,
- $\Omega \subseteq \mathbb{R}^n$ is convex,
- $x_0 \in \Omega$ is the starting point.

Define the following sets:

- The sublevel set at x_0:
$$L_0 := \{x \in \Omega : f(x) \leq f(x_0)\}$$
- The optimal set:
$$C := \{x \in \Omega : f(x) = \min_{u \in \Omega} f(u)\}$$
- The worst-case distance to optimum within the sublevel set:
$$R_0 := \max_{x \in L_0} \min_{y \in C} \|x - y\|_2$$

9.1. Smoothness Assumption per Coordinate.

Assume for each coordinate $i \in \{1, \ldots, m\}$ the gradient is Lipschitz continuous:

$$\mid (\nabla f(x + te_i))_i - (\nabla f(x))_i \mid \leq L_i \mid t \mid$$

Define:
$$L := \max_{1 \leq i \leq n} L_i$$

9.2. Sublinear Convergence in the Convex Case.

Select coordinate $i_k \in \{1, \ldots, n\}$ uniformly at random at each step. Using step size:

$$\alpha_k = \frac{1}{L}$$

We obtain the expected function error:

$$\mathbb{E}[f(x_k]) - f(x^*)] \leq \frac{2nL_{\max}R_0^2}{k}$$

This is a standard $\mathcal{O}(1/k)$ convergence rate, typical of first-order methods.

9.3. Linear Convergence for Strongly Convex f.

If f is m-strongly convex. That is:
$$f(y) - f(x) \geq \nabla f(x)^T(y - x) + \frac{m}{2}\|y - x\|^2$$

Then we get geometric convergence:

$$\mathbb{E}[f(x_k) - f(x^*)] \leq \left(1 - \frac{m}{nL_{\max}}\right)^k (f(x_0) - f(x^*))$$

9.4. Proof: Convex Case.

Assume coordinate $i_k = 1$ is selected. Form coordinate smoothness:

$$f(x_{k+1}) \leq f(x_k) - \frac{1}{2L_{\max}} \mid (\nabla f(x_k))_1 \mid^2$$

Take the expectation over the coordinate choice:

$$\mathbb{E}[f(x_k) - f(x^*)] \leq -\frac{1}{2nL_{max}}\|f(x_k)\|^2$$

Convexity implies:

$$f(x_k) - f(x^*) \leq R_0 \|\nabla f(x_k)\| \Rightarrow \|\nabla f(x_k)\| \geq \frac{f(x_k) - f(x^*)}{R_0}$$

Combining:

$$\mathbb{E}[f(x_k) - f(x^*)] - \mathbb{E}[f(x_{k+1} - f(x^*)] \geq \frac{(f(x_k) - f(x^*))^2}{2nLR_0^2}$$

9.5. Improved Version with Coordinate-wise Lipschitz Constants. Instead of assuming uniform Lipschitz constant, assign probabilities proportional to sensitivity:

$$\mathbb{P}(i_k = i) = \frac{L_i}{(L_1 + L_2, \ldots, L_N)}, \alpha_i = \frac{1}{2L_i}$$

Then:

$$\mathbb{E}[f(x_{k+1} - f(x_k)] \leq -\frac{1}{2(L_1 + L_2 + \ldots + L_N)} \|\nabla f(x_k)\|^2$$

and the convergence bound becomes:

$$\mathbb{E}[f(x_k) - f(x^*)] \leq \frac{(L_1 + L_2 + \ldots + L_N)R_0^2}{k}$$

This refinement leads to slightly better constants, and often better practical behavior.

10. Nesterov Accelerated Coordinate Descent. Standard coordinate descent achieves sublinear convergence in convex settings and linear convergence for strongly convex functions. However, it's possible to further accelerate convergence — even in the coordinate-wise case — using Nesterov's acceleration technique, originally designed for full-gradient methods.

10.1. Non-uniform Sampling Based on Coordinate Smoothness. Recall that previously we assumed each coordinate i is sampled uniformly or proportionally to its Lipschitz constantL_i. In Nesterov's accelerated version, coordinates are sampled using the square root of the coordinate-wise Lipschitz constants:

$$\mathbb{P}(i_k = i) = \frac{\sqrt{L_i}}{\sum_{j=1}^{n} \sqrt{L_j}}$$

This reflects the idea that faster moving coordinated still have influence, but the square root makes the distribution less aggressive than proportional weighting.

10.2. Accelerated Convergence Rate. With the above sampling strategy, the accelerated coordinate descent method achieves a faster convergence rate than ordinary coordinate descent[3]. **Theorem (Accelerated Coordinate Descent)** Let $f : \mathbb{R}^n \to \mathbb{R}$ be convex and differentiable, and assume the coordinate-wise Lipschitz continuity conditions hold. Then, using Nesterov's accelerated method with square-root sampling:

$$\mathbb{E}[f(x_k) - f(x^*)] \leq \frac{2R_0^2(\sum_{i=1}^{n} \sqrt{L_i})^2}{k^2}$$

This is a quadratic convergence rate in terms of $\frac{1}{k^2}$, matching the best known rates for full gradient accelerated methods, but applied in a coordinate wise manner.

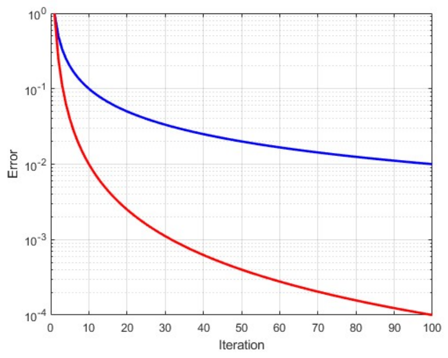

Fig. 7. *Comparison of convergence rates for standard coordinate descent (blue) and Nesterov-accelerated coordinate descent (red). Acceleration improves the rate from $\mathcal{O}(1/k)$ to $\mathcal{O}(1/k^2)$, achieving faster convergence for smooth convex problems.*

REFERENCES

[1] S. Boyd and L. Vandenberghe, Convex Optimization, Cambridge University Press, 2004.
[2] M. Jaggi, Revisiting frank-wolfe: Projection-free sparse convex optimization, in Proceedings of the 30th International Conference on Machine Learning (ICML), vol. 28 of Proceedings of Machine Learning Research, PMLR, 2013, pp. 427–435, https://proceedings.mlr.press/v28/jaggi13.pdf.
[3] Y. Nesterov, Efficiency of coordinate descent methods on huge-scale optimization problems, SIAM Journal on Optimization, 22 (2012), pp. 341–362.
[4] R. Tibshirani, Convex optimization lecture 24: Coordinate descent. https://stat.cmu.edu/~ryantibs/convexopt-F13/lectures/24-coord-desc.pdf, 2013. Carnegie Mellon University lecture notes.
[5] S. J. Wright, Coordinate descent algorithms, Mathematical Programming, 151 (2015), pp. 3–34.