Optimization with Costly Gradients

By

Gabriel Goh

B.S. (University of California, Davis) 2001

DISSERTATION

Submitted in partial satisfaction of the requirements for the degree of

DOCTOR OF PHILOSOPHY

in

MATHEMATICS

in the

OFFICE OF GRADUATE STUDIES

of the

UNIVERSITY OF CALIFORNIA

DAVIS

Approved:

Committee Member 1

Committee Member 2

Committee Member 3

Committee in Charge

2006

To...

Contents

Gabriel Goh
May 2017

Abstract

Acknowledgments

Thanks!

CHAPTER 1

Introduction

1.1. Functions with Costly Gradients

Optimization plays an important role in many of many branches of statistics and machine learning. Usually, once a model has been specified, it must be fitted. This may take the form of minimzing the loss between the model and the data, for example, or perhaps another operational goal is desired. In either case, this is the point where the reigns are then handed over to an optimization algorithm. Consider the generic unconstrained convex optimization problem f over the reals:

$$\underset{x \in \mathbf{R}^n}{\text{minimize}} \quad f(x), \qquad f : \mathbf{R}^n \to \mathbf{R}. \tag{1.1.1}$$

where f convex and lower semicontinuous [RW98, Definition 1.5]. In the framework of first order optimization [Nem05], f is thought of as a black box. Our access to f is is mediated through a subgradient oracle,

Definition 1.1.1 (0^{th} and 1^{st} order oracle)**.** Let f be a convex function. Then f is equiped with a 0th and 1st order oracle, $\mathbf{oracle}_f$ if

$$(f(x), g) = \mathbf{oracle}_f(x), \qquad f(\bar{x}) \geq f(x) + g^T(\bar{x} - x) \quad \forall \bar{x},$$

Of course, this can be succinctly stated as $g \in \partial f(x)$. We are interested in efficient algorithms for solving Problem 1.1.1. The first, and most direct measure of efficiency is its arithmetical complexity. This is the number of floating point operations necessary to drive $f(x) - \inf_x f(x)$ down to a requisite tolerance on a computer. The second, informational complexity is the number of times which the oracle is called to drive the accuracy of the problem down a similar amount.

To understand the distinctions between these two measures of efficiency, we note that many optimization algorithms involves a trade-off between two different forms of complexity. Often, we begin with a simple model of f. Next, a query to $\mathbf{oracle}_f$ is made. We then update our model of f based on this new information. The model is then queried, in the form of a subproblem, to find a new search point x to query the oracle. The process then repeats. In some extreme cases, such as the center of gravity method[Lev65], the cost of the inner subproblem can be exponential in the number of dimensions, but its informational complexity is optimal. In other cases, such as sub-gradient descent, there is essentially no work in the inner subproblems - but the algorithm requires numerous, repeated calls to the sub-gradient. Most algorithms operate somewhere in between these two extremes - and finding the right balance, while exploiting problem structure, is essential to optimal performance.

In this thesis we are motivated by the modern appetite for complexity and scale, which gives rise to functions f for which a call to $\mathbf{oracle}_f$ requires a high computational load. Such functions arise, for example, when f is defined either implicitly via another optimization problem, or an integral over a set. In either case, the cost of evaluating $\mathbf{oracle}_f$ dominates the complexity of the algorithm. Not only do we need to use gradient information judiciously, in many cases of practical interest we do not have the luxary of even calling $\mathbf{oracle}_f$ once, and we must resort to approximations.

1.2. Constructing Approximate Subgradient Oracles

Approximating an oracle is an important ingredient in much of modern optimization. As we shall soon discuss, large, expensive oracles of this nature typically arise as a byproduct of a numerical procedures. This may be a numerical optimizer, or a Monte-Carlo integrator. Either way, we compute our function via an iterative procedure which can be terminated part way. As we shall see, this partial information is usually enough to yield progress on the optimization. We will refer to such oracles as approximate oracles. By choosing an appropriate time to terminate this computation, this oracle becomes controllable, giving us the liberty of trading off accuracy for computation.

A typical algorithm which takes advantage of this proceeds as follows: in the early parts of the optimization, we only need a rough approximation to $\mathbf{oracle}_f$. And as we move to the stages of refinement and high numerical accuracy, we pursue finer approximations. As we shall explore, this

scaling of the amount of work with the accuracy of the solution is typical. And thus we can get away with nearly the same convergence rates without a single call to $\mathbf{oracle}_f$. We will discuss the construction and examples of approximate oracles in the next two sections.

1.2.1. Parametric Minimizations/Saddle Systems. A first example of a function for which $\mathbf{oracle}_f$ is expensive is one defined implicitly as the minimization of another function. If $h(\cdot,\cdot)$ is a convex-concave saddle function, i.e. it is convex the first argument, and concave in the second, then we can define f as one implicitly taken after maximizing over y,

$$f(x) := \max_y h(x,y) \tag{1.2.1}$$

We are using max instead of sup as we assume h is such that the maximum is achieved for all x. Then we can construct a 0^{th} and 1^{st} order oracle by first solving the inner subproblem, to yield optimal solution y^*, and then differentiate with respect to x. In other words

$$(h(x,y^*),g) = \mathbf{oracle}_f(x) \qquad g \in \partial h(\cdot,y^*)(x) \qquad y^* \in \operatorname*{argmax}_y h(x,y). \tag{1.2.2}$$

This is true as by the definition of the subgradient,

$$h(\bar{x},y^*) \geq h(x,y) + g^T(\bar{x}-x) = f(x) + g^T(\bar{x}-x) \qquad \text{for any } \bar{x},\ \ g \in \partial h(\cdot,y^*)(x)$$

thus matching the Definition 1.1.1. Such functions arise in many contexts, in particular in duality, be it Largrangian or Fenchel (concrete examples follow shortly).

A second example comes from parametrized minimization. Let $f(\cdot,\cdot)$ is a convex function. We define f implicitly by minimizing over a block of variables,

$$f(x) = \min_y h(x,y) \tag{1.2.3}$$

and we find ourselves in a similar situation. If we had access to the optimal solution, then if all the level sets of h are bounded (possibly empty), then by [RW98] Theorem 10.13 we can differentiate under the optimization, to get a first order oracle

$$(h(x,y^*),g) = \mathbf{oracle}_f(x) \qquad g \in \partial h(\cdot,y^*)(x) \qquad y^* \in \operatorname*{argmin}_y h(x,y). \tag{1.2.4}$$

If there is no closed form solution for the optimization, one must resort to numerical methods. Thus, each call to the oracle requires a call to a convex optimization program, one in principle solved to infinite floating point precision. This is very impractical, and as we shall see, but in both cases, it is possible by running the optimization terminating part way, to obtain an lower minorant. The further x is driven towards optimality, the more accurate the oracle becomes. We will discuss three ways of measuring the accuracy of this lower minorant. The first is the epsilon subgradient

$$g \in \partial_\epsilon f(x) \quad \text{and} \quad f(\bar{x}) + \epsilon \geq f(x) + g^T(\bar{x} - x) \quad \forall \bar{x}$$

which measures the distance from the lower bound at x from $f(x)$, see Figure 1.2.1. A computable version of the ϵ-subgradient, when $f(x)$ is unknown, is

Definition 1.2.1 (Inexact Oracle)**.** Let f be a convex function. Then f is equipped with an inexact oracle if we can find

$$(u, l, g) = \textbf{epi-oracle}_f(x, \epsilon), \quad l + g^T(\bar{x} - x) \leq f(\bar{x}) \text{ for all } \bar{x}, \quad f(x) \leq u, \quad u - l \leq \epsilon$$

The ϵ here is computable, as we can obtain (as we shall see in the next section) lower and upper bounds on f from partial computations with an appeal to duality. Since $l > f(x)$, $g \in \partial_\epsilon f(x)$ - this also gives an epsilon subgradient, though a weaker one. And finally, it is clear that $\textbf{epi-oracle}_f(x, 0) = \textbf{oracle}_f(x)$. While the above are measures of vertical distance from $f(x)$, we can also measure the error horizontally, see Figure 1.2.1. This gives rise to the shallow cut oracle, popular in the Ellipsoid method [BGT81] literature, which measures the horizontal distance of the lower bound where it intersects the horizontal line at $f(x)$ from x:

Definition 1.2.2 (Shallow-cut subgradient oracle)**.** Let f be a convex function. Then f is equipped with an shallow-cut oracle if we can find

$$g = \textbf{shallow-oracle}_f(x, \epsilon) \quad \text{and} \quad f(x) + g^T(\bar{x} - x) \leq f(\bar{x}) + \epsilon\|g\| \quad \forall \bar{x} \tag{1.2.5}$$

Let us look at a few concrete examples illustrating how to construct an oracle of this form.

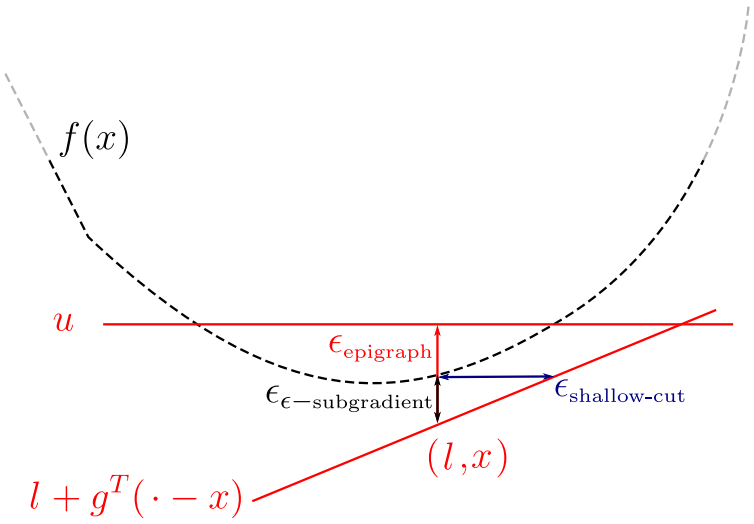

FIGURE 1.2.1. Illustration of a shallow cut with an inexact epigraphical oracle. The dotted line is $f(x)$, the polytope defined by the black solid lines P_k. The black dot is center, and the two red lines are the two cutting planes provided by the oracle.

1.2.1.1. *Saddle Systems.* For the remainder of this subsubsection, we will return to the definition of $h(\cdot,\cdot)$ in 1.2.1 as a saddle function defined on $\mathbf{R}^n \times \mathbf{R}^m$, and f as $f(x) = \sup_y\ h(x,y)$. For a fixed vector x, any vector $\bar{y}$ in the domain of $h(x,\cdot)$ certifies a lower bound $l := h(\bar{x}, x)$. Together with any subgradient $g \in \partial h(\cdot, \bar{y})(x)$, this lower bound generates a global affine minorant to f, i.e.,

$$f(x) \geq h(x,\bar{y}) \geq l + g^T(x - \bar{x}) \qquad g \in \partial h(\cdot,\bar{y})(x)$$

An upper bound is trickier to find, and requires an appeal to duality. This often requires knowledge of the structure of $h(x,\cdot)$. We will, for notational clarity, hide the dependence on x and make h convex by letting $h(v) := -h(x,v)$. Assume that h can be written as the sum of two functions,

$$h(v) = \bar{h}(Av) + \tfrac{1}{2}\|v\|^2. \tag{1.2.6}$$

Assuming strong Fenchel Duality, i.e. $A \cdot \mathrm{dom}(\bar{h})$ is not empty, a weak assumption, we have

$$\begin{aligned} f(x) = \sup -h(x,v) = -\inf_v h(v) &= -\inf_v\{\bar{h}(Av) + \tfrac{1}{2}\|v\|^2\} \qquad \text{(compute fenchel dual)} \\ &= \inf_y\{\bar{h}^*(y) + \tfrac{1}{2}\|A^T y\|^2\} \leq \bar{h}^*(\bar{y}) + \tfrac{1}{2}\|A^T\bar{y}\|^2. \end{aligned}$$

Thus for any $\bar{y}$, we have an the upper bound on f. We do not simply require an upper and lower bound, however. We also require $l - u$ be controllable. This can be achieved in the by applying a primal dual algorithm, noting that $l - u$ is exactly the duality gap. For example, we may apply a first order method to the dual problem

$$\underset{y}{\text{minimize}} \quad \bar{h}^*(y) + \tfrac{1}{2}\|A^T y\|^2$$

for a set number of iterations. Once we have an "acceptable" y, we can recover the primal with $v = -A^T y$. As we get closer to the optimum, y approaches the dual solution, and v approaches the primal solution. The duality gap is precisely ϵ. To summarize, we compute upper and lower bounds

$$u = \bar{h}^*(y) + \tfrac{1}{2}\|A^T y\|^2 \qquad l = -\bar{h}(-AA^T y) - \tfrac{1}{2}\|A^T y\|^2$$

which, with increasing iterations, go down to 0. By applying an optimal first order method to the dual, we can get the duality gap to decrease at a rate of $\mathcal{O}(1/k)$ where k is the number of iterations. (TODO: I've seen this results somewhere. Track it down). We can, alternately, attack the saddle point problem directly with an optimal primal-dual first order method.

$$\begin{aligned} f(x) = -\inf_v h(v) &= -\inf_v \{\bar{h}(Av) + \tfrac{1}{2}\|v\|^2\} \\ &= -\inf_v \left\{ \sup_y \{y^T Av - \bar{h}^*(y)\} + \tfrac{1}{2}\|v\|^2 \right\} = -\sup_y \left\{ \inf_v \{y^T Av + \tfrac{1}{2}\|v\|^2\} - \bar{h}^*(y) \right\} \end{aligned}$$

Any primal-dual solution $(\bar{v}, \bar{y})$ will provide both an upper and a lower bound, as shown

$$f(x) = -\inf_v h(v) = -\sup_y \{-\tfrac{1}{2}\|Ay\|^2 - \bar{h}^*(y)\} \leq \tfrac{1}{2}\|A\bar{y}\|^2 + \bar{h}^*(\bar{y})$$

$$f(x) = -\inf_v h(v) = -\inf_v \{\bar{h}(Av) + \tfrac{1}{2}\|v\|^2\} \geq -\bar{h}(A\bar{v}) - \tfrac{1}{2}\|\bar{v}\|^2$$

Employing an optimal primal dual algorithm, the upper and the lower bounds converge at a rate of $\mathcal{O}(1/k)$ [CLO14].

Example 1.2.3 (Support Vector Machines with Dataset Constraints)**.** The classification problem in machine learning involves predicting labels, here $b_i \in \{-1, 1\}$ from n features, $a_i \in \mathbf{R}^n$. We would

like to find a vector $y \in \mathbf{R}^n$ which defines a linear function $a \mapsto a^T y$ such that for all i

$$a_i^T y < 0 \;\text{ if }\; b_i = -1, \qquad a_i^T y \geq 0 \;\text{ if }\; b_i = 1.$$

If such a seperation is possible, the data is known as "linearly seperable". This is very rare, however, and in general we can only hope to minimize the number mislabeled points by solving

$$\underset{y}{\text{minimize}} \qquad \mathbf{1}^T \mathbb{I}(BAy), \qquad B = \text{diag}(b),\; A = [a_1, \ldots, a_m],\; \mathbb{I}(x) = \begin{cases} 1 & x > 0 \\ 0 & x \leq 0 \end{cases}$$

This has a simple geometric interpretation - we have a hyperplane $\{a \mid y^T a = 0\}$, the "seperating hyperplane", which as the name suggests separates the positively labeled points from the negatively labeled ones. We can also define a weighted version of this problem, where each datapoint is given a weight w_i, which determines its importance (if all points are equal, $w_i = 1$). The weighted version of this problem is

$$\underset{y}{\text{minimize}} \qquad w^T \mathbb{I}(BAy)$$

This problem, unfortunately, is NP-hard. We can employ, however, the convex relaxation $\mathbb{I}(x) \leq \max\{0, 1 - x\}$ and solve the relaxation instead. This objective has the effect of penalizing a misclassified point's distance from the decision boundary. Let 1 be the vector of all 1's. Then we arrive at the following formulation of the problem,

$$\underset{y}{\text{minimize}} \qquad w^T \max\{0, \mathbf{1} - BAy\} + \tfrac{1}{2}\|y\|^2$$

a weighted support vector machine. It is rare that the goal of minimizing misclassification error is the end in itself, however. In this example we wish to increase the expressiveness of this by adding constraints, such as

- Coverage: One may wish to control how often a classifier predicts the positive (or negative) class. For example, one may want to ensure that only 10% of customers are selected to receive a printed catalog due to budget constraints, or perhaps to compensate for a biased training set.

- Churn: We wish for two models to disagree only on a relatively small number of examples near the true decision boundary
- Fairness: A practitioner may be required to guarantee fairness of a learned classifier, in the sense that it makes positive predictions for members of different subgroups at certain rates. For example, one might require that housing loans be given equally to people of different genders. Hardt et al. [2016] identify three types of fairness: (i) demographic parity, in which positive predictions are made at the same rate on each subgroup, (ii) equal opportunity, in which only the true positive rates must match, and (iii) equalized odds, in which both the true positive rates and false positive rates must match
- Recall and Precision: Requirements of real-world classifiers are often expressed in terms of precision and recall, especially when examples are highly imbalanced between positives and negatives. In our framework, we can handle this problem via Neyman-Pearson classification [e.g. Scott and Nowak, 2005, Davenport et al., 2010], in which one seeks to minimize the false negative rate subject to a constraint on the false positive rate.

A key aspect of many of the goals of Section 1 is that we can view them as being classification problems defined on different datasets. For example, we might seek to maximize the accuracy on a set of labeled examples drawn in some biased manner, require that its recall be at least 90% on 50 small datasets sampled in an unbiased manner from 50 different countries, desire low churn relative to a deployed classifier on a large unbiased unlabeled dataset, and require that 100 given egregious examples be classified correctly. For each constraint, we have one dataset i, and we can therefore, we can write our idealized objective as:

$$\underset{y}{\text{minimize}} \quad (w^0)^T \mathbb{I}(BAy) \qquad \text{s.t.} \quad (w^i)^T \mathbb{I}(BAy) \leq u^i$$

This problem can be relaxed using the same technique to yield the convex program

$$\text{minimize} \qquad (w^0)^T \max\{0, e - L^0 A^0 y\} + \tfrac{1}{2}\|y\|^2 \quad \text{s.t} \quad (w^i)^T \max\{0, e - L^i A^i y\} \leq u^i$$

We will solve this via Lagrangian Duality. The Lagrangian is

$$\mathcal{L}(x, y) = y^T u + g(y) \qquad g(y) = (w^0)^T \max\{0, e - L^0 A^0 y\} + \textstyle\sum_{i=1}^m y_i (w^i)^T \max\{0, e - L^i A^i y\} + \frac{1}{2}\|y\|^2$$

Observe g takes the form of a weighted support vector machine, with weights

$$g(y) \stackrel{\text{def}}{=} -\inf_x \left\{ w_\lambda^T \max\{0, \mathbf{1} - BAy\} + \tfrac{1}{2}\|y\|^2 \right\} \quad w_\lambda^i = \begin{cases} w^0 & \text{if } i = 0 \\ \lambda_i w^i & \text{if } i \neq 0 \end{cases}, \quad \begin{matrix} L = [L^0, \ldots, L^m] \\ A = [A^0, \ldots, A^m] \end{matrix}$$

Since strong duality holds, $\sup_{y \geq 0} \inf_x \mathcal{L}(x, y) = \inf_x \sup_{y \geq 0} \mathcal{L}(x, y)$. Therefore each computation of g requires the solution of a weighted SVM. Using Fenchel duality, we can derive the primal dual pairs for the problem

$$-\inf_x \left\{ w_\lambda^T \max\{0, e - BAy\} + \tfrac{1}{2}\|y\|^2 \right\} = \inf \left\{ \mathbf{1}^T v - \tfrac{1}{2}\|A^T v\|^2 \mid 0 \leq v \leq w_\lambda \right\} \tag{1.2.7}$$

And thus, any feasible primal dual pair will give us an upper and lower bound. Given such a pair $(\bar{v}, \bar{y})$, we can write the upper and lower bounds as

$$\begin{aligned} u &= e^T \bar{v} - \tfrac{1}{2}\|A^T \bar{v}\|^2, \qquad 0 \leq \bar{v} \leq w_\lambda \\ l &= w_\lambda^T \max\{0, \mathbf{1} - BA\bar{y}\} + \tfrac{1}{2}\|\bar{y}\|^2 \\ g_i &= w^{iT}(\max\{0, e - B^i A^i \bar{y}\} - v^i) \end{aligned}$$

1.2.1.2. *Constructing an Inexact Oracle - Parametrized Optimization.* The parametrized optimization problem is in some sense the dual of the above problem. Here, finding an upper bound on f is easy, as

$$f(x) = \inf_y h(x, y) \leq h(x, \bar{y})$$

for any $\bar{y}$. Obtaining a lower bound, however, depends on duality.

As in the previous section, we consider the case where our objective, as a function of y has separable structure:

$$h(x, y) = \bar{h}(x, Ay) + \tfrac{1}{2}\|y\|^2$$

an application of Fenchal Duality gives the following lower bound

$$\begin{aligned} f(x) = \inf_y h(x, y) &= \inf_y \left\{ \bar{h}(x, Ay) + \tfrac{1}{2}\|y\|^2 \right\} \\ &= -\inf_v \left\{ \bar{h}(x, \cdot)^*(v) + \tfrac{1}{2}\|A^T v\|^2 \right\} \geq -\bar{h}(x, \cdot)^*(\bar{v}) - \tfrac{1}{2}\|A^T \bar{v}\|^2. \end{aligned}$$

Noting that $g(x) = -\bar{h}(x,\cdot)^*(\bar{v})$ is a convex function, linearizing this will yield the linear lower bound we seek, with

$$l = -h(\bar{x},\cdot)^*(\bar{v}) - \tfrac{1}{2}\|A^T\bar{v}\|^2, \qquad g \in \partial_x[\bar{h}(x,\cdot)^*(\bar{v})](\bar{x}).$$

We are once again in a situation where our upper and lower bounds define a duality gap. Using a primal dual solver will yield a controllable inexact oracle.

Example 1.2.4 (Two-Stage Stochastic Linear Programs)**.** In system design we are often interested in designing a system where there are multiple actors y_i, each acting in their own self interest based on partial information, q_i, W_i, d_i. The system in which they act upon is parametrized by x, and their actions have utility $Q_i(x)$.

$$\underset{x}{\text{minimize}} \quad f(x) = e^T x + \sum_{i=1}^{m} Q_i(x), \qquad Q_i(x) = \inf_y \left\{ q_i^T x \mid W_i y = d_i - T_i x, y \geq 0 \right\}$$

We wish to design an optimal system, taking into account the self interested nature of the individual actors. This is a nested minimization problem - evaluating x requires a solution to a linear program. By LP duality, we can get a lower bound for each individual Q_i, as shown

$$Q_i(x) = \inf_y \left\{ q_i^T x \mid W_i y = d_i - T_i x, y \geq 0 \right\} = \sup_v \left\{ (d_i - T_i x)^T v \mid W_i v \leq q_i \right\}$$

And thus any feasible v furnishes a linear lower bound on an individual Q_i and any feasible y furnishes an upper bound. ϵ here is the duality gap. And hence by using any primal-dual linear programming solver, we can drive the error down arbitrarily.

Example 1.2.5 (Bias in a Support Vector Machine)**.** There is a large amount of research which goes into solving the optimization problem of (1.2.3). Highly optimized codes such as `LIBLINEAR` [FCH+08] scales efficiently to hundred or thousands of datapoints.

The formulation of the SVM in (1.2.3), has a shortcoming. We often desire our objective to be invariant to shifts in the decision boundary, however, i.e. a boundary of $\{a \mid a^T x \leq b\}$ has the same objective for any b. Though this can be shoehormed in by adding a vector of 1's to the data, this

parameter is still penalized in the quadratic term. A SVM which is truly agnostic to b would have the objective

$$\underset{x\in\mathbf{R},y}{\text{minimize}} \qquad h(x,y) = w^T \max\{0, \mathbf{1} - L(Ay - x\mathbf{1})\} + \frac{1}{2}\|y\|^2$$

This formulation of the SVM however, resists the highly efficient dual methods of coordinate descent used in packages such as `LIBLINEAR` [FCH+08] due to the coupling scalar x. A possible approach to solve this would be to keep x fixed, and minimizing simply with respect to y - an SVM without a bias term. The bias term can then be optimized via a 1D optimization problem. Since

$$f(x) = \inf\left\{w_\lambda^T \max\{0, e - BAy\} + \tfrac{1}{2}\|y\|^2\right\} \leq w^T \max\{0, \mathbf{1} - L(A\bar{y} - x\mathbf{1})\} + \tfrac{1}{2}\|\bar{y}\|^2$$

we have the following upper and lower bounds

$$\begin{aligned} u &= \mathbf{1}^T\bar{v} - \tfrac{1}{2}\|A^T\bar{v}\|^2, \qquad 0 \leq \bar{v} \leq w_\lambda \\ l &= w^T \max\{0, \mathbf{1} - L(A\bar{y} - x\mathbf{1})\} \\ g &= \partial(w^T \max\{0, \mathbf{1} - L(A\bar{y} - \cdot\mathbf{1})\})(x) \end{aligned}$$

where the upper bound comes from SVM duality, described in .

1.2.2. Expectations. Another source of costly oracles comes when f is defined implicitly via an integral or expectation. Here we will consider the smooth minimization problem

$$\underset{x}{\text{minimize}} \quad f(x) := \mathbf{E}\, h(x,\xi) = \int h(x,\xi)\, d\xi.$$

where ξ is a random variable the expectation is taken over. If $h(\cdot,\xi)$ is convex, f remains convex - and if no closed form for the integral exists, the integral must be approximated via Monte Carlo or quadrature. We can, as in the previous section define a controllable oracle for functions of this form. There are three different ways of measuring the accuracy of the problem, and hence we have three different kinds of oracles:

Definition 1.2.6 (Stochastic First Order Oracle)**.** Let e be a random variable which depends on x. Then f is equipped with an inexact oracle if we can find

$$g = \textbf{stochastic-oracle}_f(x, \epsilon), \qquad g = \nabla f(x) + e(x).$$

We require the $\mathbf{E}[e(x)] = 0$, and there are three variations of the stochastic oracle.

A. [Variance] $\mathbf{E}\|e(x)\|^2 \leq \epsilon$;

B. [High Probability] $\Pr\left(e(x)_i \geq \delta \mid x\right) \leq \exp\left(-\delta^2/\epsilon\right)$ for all i

C. [Deterministic] $\|e(x)\|^2 \leq \epsilon$.

These conditions are ordered in increasing strength: if (C) holds, then (B) holds by Serfling's inequality (Theorem A.2.2), and if (B) holds, then (A) holds because the exponential bound implies a bound on the second moment, i.e.,

$$\mathbf{E}\left[[e(x)_k]_i^2 \mid \mathcal{F}_{k-1}\right] = \int_0^\infty \Pr([e(x)_k]_i^2 \geq \epsilon \mid \mathcal{F}_{k-1})\, d\epsilon \leq \int_0^\infty \exp\left(-\epsilon^2/U_k\right) d\epsilon < \infty.$$

1.2.2.1. *Constructing Inexact Oracles - Sampling.* The basis of quadrature and monte-carlo approaches to solving an integral all revolve around representative evaluations of the function at well placed points. The degree to which we can approximate this integral depends on the amount of variation that exists within the the function. In the extreme case, if the gradients were all equal at every point, a single sample is sufficient to obtain the true gradient. And if the variation within the function were tremendous, any single point would not suffice as an approximation - taken to the extreme if the function were not integable in ξ. Thus our bounds on the amount of informaiton depend on our intrustments for quantifying the degree of variaiton of h in ξ. There are three ways for quantifying this uncertainty.

Hypothesis 1.2.1 (Uniform bounds)**.** We wish to posit that for all x, ξ one of the following hypothesis hold

A. [Variance] $\frac{1}{m}\sum_{i=0}^{m}\mathbf{E}\|\nabla h(x,\xi_i)-\mathbf{E}[\nabla h(x,\xi_i)]\|^2 \le B_A$;

B. [Exponential Tail] $\sup_x\{\max_\xi[\nabla h(x,\xi)]_i - \min_\xi[\nabla h(x,\xi)]_i\} < B_B$,

C. [Deterministic] $\|\nabla h(x,\xi)\|^2 \le B_C$.

We now discuss concrete examples of this oracle.

1.2.2.2. *Sampling With Replacement.* Since the gradient is a linear operator, under very weak conditions we can move the gradient inside the expectation, $\nabla\mathbf{E}h(x,\xi) = \mathbf{E}\nabla h(x,\xi)$. And hence if we could obtain a uniform sample from ξ_i, we can construct an estimate of the gradient sampling the gradients at these instantiations:

$$g = \mathbf{oracle}_f(x,\epsilon), \quad g = \sum_{i=0}^{m}\nabla F(x,\xi_i), \quad e(x) = \frac{1}{m}\sum_{i=0}^{m}\nabla h(x,\xi_i) - \mathbf{E}[h(x,\xi_i)]$$

Our error term $e(x)$ for this estimator is sampling error, which clearly has expectation 0.

(A) Variance Stochastic Oracle: The weakest of the three oracles only requires Hypothesis 1.2.1(A), a bound on the variance of the problem. Taking expectations of $\|e(x)\|$,

$$\mathbf{E}\|e(x)\|^2 = \frac{1}{m^2}\sum_{i=0}^{m}\mathbf{E}\|\nabla h(x,\xi_i)-\mathbf{E}[\nabla h(x,\xi_i)]\|^2 \le \frac{B_A}{m}$$

Thus, to achieve an error of ϵ, one needs at least

$$m = \left\lceil \frac{B_A}{\epsilon} \right\rceil \quad \text{samples.}$$

(B) High Probability Stochastic Oracle: The high probability bounds stem from Hoeffling's inequality, applied with random variable $X_i = \nabla f(x,\xi_i)_i$

$$\Pr\left(e(x)_i \ge \delta\right) = \Pr\left(\nabla h(x,\xi_i)_i - \mathbf{E}[\nabla h(x,\xi_i)_i] \ge \delta\right) \le \exp\left(\frac{-2m\delta^2}{B_B^2}\right)$$

which shows that if a random variable is bounded, its tails are not heavy. Therefore, if Hypothesis 1.2.1(A) holds, then we need

$$m = \left\lceil \frac{B_B^2}{2\epsilon} \right\rceil \qquad \text{samples}$$

to get a high probability oracle.

(C) Deterministic Stochastic Oracle: We cannot construct a controllable deterministic oracle. No matter how many samples we take, there is a small but nonzero chance the error will not drop - for example if the same sample repeated every time.

1.2.2.3. *Sampling Without Replacement.* We will note an important special case. When ξ takes on a finite number of values with uniform probability, then f is equivalent to the familiar case of sums of functions

$$h(x, \xi) = \frac{1}{M} \sum_{i=1}^{M} h_i(x),$$

Such sums of functions, typically with very large M, arise very often in machine learning and statistics, as we shall see in the following example.

Example 1.2.7 (Minimizing Loss)**.** A common theme in statistics and machine learning is the fitting of a set of observations, $\{a_i\}_{i=1}^m \in \mathbf{R}^n$ to a set of targets $\{b_i\}_{i=1}^m$. In linear regression, we wish to find a set of weights, x, for which $x^T a_i \approx b_i$. To do so, we minimize over the misfit of a model over a loss function taken over dataset. Therefore

$$h_i(x) = \tfrac{1}{2}(a_i^T x - b_i)^2$$

In binary classification, as discussed in Example 1.2.3 is when b_i are a set of labels, in $\{-1, 1\}$. As in the case of SVMs , we find a convex penalty to the decision boundary. Here we are interested in loss functions which are smooth, such as

$$h_i(x) = \log(1 + \exp[-b_i a_i^T x]) \qquad \text{Logistic Regression}$$

$$h_i(x) = \max\{0, (b_i a_i^T x)^2\} \qquad \text{Smooth SVM}$$

These are just a small sample of possible convex loss functions defined for pieces of data.

When the sum is finite, we can improve over independent sampling - we can take samples without replacement. This reduces the number of samples we need to control ϵ, because as we approach M, our population size, our error goes to 0.

(A) Variance Stochastic Oracle: If we assume

$$\mathbf{E}\|e(x)\|^2 = \left(\frac{M-m}{M}\right)\frac{\mathbf{Var}(\|\nabla h(x,\xi_i)\|)}{m} \leq \left(\frac{M-m}{M}\right)\frac{B_A}{m}$$

Notice this differs from the previous bound by a factor of $1 - m/M$. This term, known as the finite population correction, approaches 0 as m approaches M, shrinking the variance of the estimator dramatically in that regime. Therefore there are

$$m = \left\lceil \frac{B_A(B_A+1)}{B_A + 2M\epsilon} \right\rceil$$

samples needed to get this oracle.

(B) High Probability Stochastic Oracle: The high probability bounds require a variation of Hoeffding's inequality, Serfling's inequality which show

$$\Pr\left(e(x)_i \geq \delta\right) = \Pr\left(\nabla h(x,\xi_i)_i - \mathbf{E}[\nabla h(x,\xi_i)_i] \geq \delta\right) \leq \exp\left(\frac{-2m\delta^2}{B_B^2}\right)$$

Where the final step requires the assumption (C). Therefore we require

$$m = \left\lceil \frac{B_B(B_B+1)}{B_B + 2M\epsilon} \right\rceil \qquad \text{samples}$$

(C) Deterministic Stochastic Oracle: If we assume (C) then the sampling error, can be bounded by

$$\begin{aligned}\|e(x)\|^2 &= \left\| \frac{M-m}{Mm}\sum_{i\in S}\nabla h_i(x) - \frac{1}{M}\sum \nabla h_i(x)\right\|^2 \\ &\leq \left(\frac{M-m}{Mm}\left\|\sum_{i\in S}\nabla h_i(x)\right\| + \frac{1}{M}\left\|\sum_{i=1}^{M}\nabla h_i(x_k)\right\|\right)^2 \leq 4\left(1-\frac{m}{M}\right)^2 B_C\end{aligned}$$

Therefore we require

$$m = \left\lceil B_C\left(1-\sqrt{\frac{\epsilon}{4}}\right)\right\rceil, \qquad \text{samples}$$

to achieve a deterministic oracle

1.3. Algorithms with Exact Oracles

In this section we consider two broad categories of algorithms which use the subgradient informaiton in qualitiatively different ways. The first are the cutting plane methods, which take advantage of the fact that the first order oracle gives a global, lower bound. This gives us clear means of localizing the optimum, any points which can be proven to be above any upper bound on the optimum can be safely ignored. These methods are typically most effective in problems with small dimensions, but yield optimal oracle complexity.

In moderate dimensions, the above methods are no longer suitable. However, when our objective is smooth, the subgradient on a smooth function gives a direction of descent. Here, we treat the subgradient as a source of local information, and move in a step along that direction.

1.3.1. Cutting Plane Methods. In this subsection we will tackle the generic constrained optimization problem

$$\underset{x\in\mathcal{X}}{\text{minimize}} \qquad f(x), \qquad \mathcal{X} \text{ is convex and compact}$$

Let $\mathcal{S}^*$ be the solution set, and x^* be a point in it. A literal reading of the definition of the subgradient oracle lends itself naturally to its use as a cutting plane. Since

$$f(\bar{x}) \geq f(x) \geq \min_x f(x) \quad \forall \bar{x} \text{ such that } g^T(\bar{x} - x) \geq 0.$$

$\mathbf{oracle}_f$ acts as a seperation oracle between x and $\mathcal{S}^*$. With a single call to the $\mathbf{oracle}_f$, we now know with certainty that the set

$$\{x \mid g^T(\bar{x} - x) \geq 0\} \qquad \text{does not contain the optimum.}$$

This lends itself to the following algorithm. We begin with a compact polytope containing at least one point in $\mathcal{S}^*$. Then

$$x_k = \mathbf{center}(P_k),$$

$$g_k = \mathbf{oracle}_f(x_k)$$

$$P_{k+1} = P_k \cap \{x \mid g_k^T(x - x_k) \leq 0\}.$$

There are many choices for **center**, but the main aim is that all hyperplanes through it partition the set into two approximately evenly parts. The most well understood choices for are **center** are the center of gravity [Lev65, New65], the maximum inscribed ellipsoid [Tar88], the volumetric center [Vai89], and the analytic center [Ye96].

Cutting-plane algorithms enjoy a number of attractive properties that set them apart from other first-order methods. Firstly, for a fixed dimension, the center of gravity and maximum inscribed ellipsoid variants achieve optimal oracle complexity, up to a constant factor [Nem94]. Secondly, on any convex function, they achieve a universal, linear rate of convergence [Nem94]. This rate of convergence, surprisingly, is oblivious to the structure of the function itself.

These advantages come at a price. Firstly, each step of the cutting plane algorithm requires the computation of **center**, which may exceed the cost of the original problem. Furthermore, though the convergence rates of cutting plane algorithms do not depend on the conditioning of the problem, it depends strongly on the dimension of the problem, sometimes degrading exponentially as the dimensions increase. It is for these reasons that these algorithms are not practical for problems of any substantial dimension.

An alternate strategy, the Ellipsoid method [BGT81] suggests a different to representing the localizer. Instead of storing all the previous cuts, we maintain an Ellipsoid which contains the optimum. At each iteration, the search point is the center of this Ellipsoid. The cutting plane splits the ellipsoid into two parts, and the next Ellipsoid is the largest ellipsoid which contains this half ellipsoid. Since simple, concise formulas exist for updating these Ellipsoids, this leads to inexpensive computations. And surprisingly, this approach is sufficient to obtain linear convergence

1.3.2. Epigraphical cutting plane algorithms. We note a shortcoming of the shallow-cut subgradient oracle in practice. Because this oracle's accuracy is measured relative to the norm of the gradient, it is not possible to know *a priori* how much computation it requires. Another shortcoming of the algorithm described above is that it does not make use of the objective value. Since the objective value is often computed as a side effect of the gradient computation, discarding this information seems wasteful. The epigraphical cutting plane method, first suggested in [BGVDM94] and also described in [BV07, GV99, Meh00] is a remedy to this problem. It uses the stronger oracle

Definition 1.3.1 (Zeroth and first order oracle)**.**

$$(u, g) = F(x) \qquad f(\bar{x}) \geq u + g^T(\bar{x} - x) \quad \forall \bar{x}, \quad f(x) = u \tag{1.3.1}$$

This oracle gives us a global lower minorant for f, $u + g^T(\cdot - x) \leq t$ and an upper bound on the $f(x^*)$, u. Operating on the lifted space (t, x), the epigraphs of these two linear functions form the cutting planes which separate the current iterate (t_k, x_k) from the solution, $(f(x^*), x^*)$.

$$(t_k, x_k) = \textbf{center}(P_k)$$
$$(u_k, g_k) = \textbf{oracle}_f(x_k)$$
$$P_{k+1} = P_k \cap \{(t, x) \mid t \leq u_k\} \cap \{(t, x) \mid l_k + g_k^T(x - x_k) \leq t\}$$

Unlike the cutting plane algorithm, these two half-spaces generally do not pass through the search point (t_k, x_k). It is guaranteed, however, to do at least better, and cut off the search point. And the cuts are typically deep - they truncate more than is strictly needed, to achieve a greater volume reduction. This comes at the price, however, of lifting the problem an extra dimension.

These methods have strong connections to bundle methods. If we define

$$u_k^* = \min\{u_0, \ldots, u_{k-1}\}, \qquad l_k^*(x) = \max_{i<k}\{l_i + g_i^T(x - x_i)\}.$$

Then l_k^* gives a lower bound on f and u_k^* an upper bound on $f(x^*)$. Kelly's method [Kel60] produces the next iterate by minimizing l_k^* directly. Bundle methods [Lem75, Wol75] regularize this approach work by minimizing l_k^* with a quadratic term. Many variations of these methods exist, which we will not attempt to survey here.

1.3.2.1. *Convergence Rates.* Let us now review a few results regarding the convergence of a few prominent cutting plane methods. To do so, we will require the following lemma, which is a slight variation of a remark in [Tar88]. For our purposes it is helpful to define a slightly more general notion of volume, a d-measure. μ is a d-measure if the volume increases relative to inclusion, $P_1 \subseteq P_2 \Rightarrow \mu(P_1) \leq \mu(P_2)$, it is homogenious with respect to scaling, i.e. $\mu(a + AP) = \det(A)\mu(P)$. Two such measures of μ used in this paper are the volume of μ and the volume of the maximum inscriped ellipse of P.

Lemma 1.3.2. Let P be a convex polytope containing x^*. Then there exists a point on the boundary of P, $\bar{x}$ such that

$$f(\bar{x}) - f(x^*) \leq \left(\max_{x \in C} f(x) - \min_{x \in C} f(x)\right) \left(\frac{\mu(P)}{\mu(C)}\right)^{1/d}$$

Proof. Following [Tar88],

$$\alpha^\star = \max_{a \in \lambda} \{\alpha \mid \alpha \geq 0, x^* + \alpha C \subseteq P\}.$$

Let $x = \text{argmax}_{x \in C} f(x)$. Let $\bar{x}$ be the point in which the line $[x^*, x]$ intersects P. Then

$$\begin{aligned} f(\bar{x}) - f(x^*) &\overset{(a)}{\leq} (f(x) - f(x^*)) \frac{\|\bar{x} - x^*\|}{\|x - x^*\|} \\ &\overset{(b)}{\leq} (f(x) - f(x^*))\, \alpha^\star \overset{(c)}{\leq} (f(x) - f(x^*)) \left(\frac{\mu(P)}{\mu(C)}\right)^{1/n}. \end{aligned}$$

(a) comes from the fact that

$$\frac{f(y) - f(x)}{\|y - x\|} \leq \frac{f(z) - f(x)}{\|z - x\|} \qquad x \in [y, z].$$

(b) comes from the fact that and $x^* + \alpha C \subseteq P$, and (c) comes from the fact that $\alpha^n \text{vol}(C) = \text{vol}(\alpha C) \leq \text{vol}(P)$. □

Observe that for all cutting plane methods, including the ellipsoid method, $\min\{f(x_1), \ldots, f(x_k)\}$ is smaller than all the points in the perimeter. Therefore we can use the above theorem to effectively determine rates of convergence. The convergence rate we get is

$$\min\{f(x_1), \ldots, f(x_k)\} - f(x^*) \leq \text{rate}^{k/n} \cdot \left(\max_{x \in C} f(x) - \min_{x \in C} f(x)\right)$$

For the epigraphical variants of the cutting plane algorithm, we have the following analogous lemma

Lemma 1.3.3. Let $\mathcal{K} = \text{conv}\{\sup_{x \in C} f(x) \times S, (f(x^*), x^*)\}$. Then for an epigraphical polytope

$$P = \{(t, x) \mid t \leq u^*\} \cap \text{epi}(l), \qquad \text{for some convex lower bound } l \text{ on } f$$

Then

$$u^* - f(x^*) \leq \left(\max_{x \in C} f(x) - \min_{x \in C} f(x) \right) \left(\frac{\mu(P)}{\mu(\mathcal{K})} \right)^{1/(n+1)}$$

1.3.3. Proximal Methods.

1.3.3.1. *Smooth Functions with Composite Structure.* While smooth optimization is useful in itself, we can, with a little effort, add a small amount of structurally simple nonsmoothness. We will consider the minimization of problems with additive composite structure, i.e.

$$\underset{x \in \mathbf{R}^n}{\text{minimize}} \quad f(x) + g(x) \tag{1.3.2}$$

Here we consider functions $f : \mathbf{R}^n \to \mathbf{R}$ and $g : \mathbf{R}^n \to \mathbf{R} \cup \{+\infty\}$. Here f is assumed to be smooth, and g is usually simple. In machine learning applications, typically, f is a loss function that penalizes incorrect predictions of a model. f contains the data, and is typically hard to compute - but is smooth. g is a regularizer that encourages desirable structure in the solution. Important examples of nonsmooth regularizers are the 1-norm and total variation, which encourage sparsity in either x or its gradient. Let us explore a few examples of functions with composite structure.

Example 1.3.4 (Feature Selection)**.** The feature selection problem considers the case where only a subset of the features are informative. The rest have no predictive value, and should be omitted to obtain the most parsimonious model.

Example 1.3.5 (Total Variation Denoising)**.**

Example 1.3.6 (Support Vector Machines)**.**

1.3.3.2. *Descent Algorithms for Composite Problems.* In the second part of the thesis, we will think of the subgradient oracle as a source of local information - as a direction of descent. When f is smooth, our oracle gives us gradient information, which can be interpreted as decrease if we move

an tiny amount in d

$$\nabla f(x) = \lim_{\epsilon \to 0} \max_{d} f(x + \epsilon d)$$

This information is exploited in descent methods, such as gradient descent and its accelerated versions. In these algorithms, we make a small step in a descent direction - and secure a small, local amount amount of function descrease at every iteration.

$$x^+ = x - \alpha \nabla f(x)$$

We can generalize slightly to problems where g isn't 0. We begin by discussing an interpreation of a large class of descent methods popular in first order optimization. Given a function f, differentiable at x_k, our guess of the solution at iterate k, we can construct a quadratic model around k as such:

$$f(x) \approx f(x_k) + \nabla f(x_k)^T (x - x_k) + \frac{1}{2}\|x - x_k\|_H^2 = f(x_k) + \frac{\alpha}{2}\|x - x_k - \frac{1}{\alpha}\nabla f(x_k)\|_H^2$$

H we will assume is a invertible, quadratic model of curvature. For different choices of H, we obtain

In composite optimization, we do an identical thing. Pulling g through and completing the square, we get

$$\begin{aligned} f(x) + g(x) &\approx f(x_k) + \nabla f(x_k)^T (x - x_k) + \frac{1}{2}\|x - x_k\|_H^2 + g(x) \\ &= f(x_k) + \frac{\alpha}{2}\|x - x_k - \frac{1}{\alpha}\nabla f(x_k)\|_H^2 + g(x) \end{aligned}$$

By minimizing the approximation, we get the next iterate. Therefore

$$x^+ = \operatorname*{argmin}_{\bar{x}} \left\{ \frac{1}{2}\|\bar{x} - x - H^{-1}\nabla f(x)\|_H^2 + g(\bar{x}) \right\}$$

This is one motivation for the definition of the proximal operator. The objective value which the quadratic optimization takes

$$\mathbf{prox}_g^H(z) := \operatorname*{argmin}_{x \in \mathbf{R}^n} \left\{ \frac{1}{2}\|z - x\|_H^2 + g(x) \right\}, \qquad \mathbf{env}_g^H(z) := \min_{x \in \mathbf{R}^n} \left\{ \frac{1}{2}\|z - x\|_H^2 + g(x) \right\}$$

And the proximal gradient iteration is

$$x^+ = \mathbf{prox}_g^\alpha(x - \alpha^{-1}\nabla f(x)) \tag{1.3.3}$$

TABLE 1.3.1. Preconditioned proximal-gradient methods.

Method	Reference	H
iterative soft thresholding	[BT09]	αI
symmetric rank 1	[BF12]	$\alpha I + ss^T$
identity-minus-rank-1	[KV14]	$\alpha I - ss^T$
proximal L-BFGS	[SvdBFM09, ZhYDR14, ST16]	$\Lambda + SDS^T$
proximal Newton	[LSS14, TDKC15, BNO16]	$\nabla^2 f$

The connection to gradient descent is clear. If $g = 0$, we recover the descent iteration. Like gradient descent, under reasonable hypotheses they are guaranteed to achieve, within k iterations, a function value that is within $\mathcal{O}(1/k)$ of the optimal value using a constant step size, and within $\mathcal{O}(1/k^2)$ using an accelerated variant. [Tse10] outlines a unified view of the many proximal- gradient variations, including accelerated versions such as FISTA [BT09].

For g's which are structurally simple, this it leads to an inexpensive proximal iteration. This is not limited to, but particularly true in the case where g is separable.

When gradients of f are expensive to compute, incorporate into our iterations some approximation of curvature. Suppose that H is a positive-definite matrix. The iteration

$$x^+ = \mathbf{prox}_g^H(x - H^{-1}\nabla f(x)) \tag{1.3.4}$$

is a generalization of proximal-gradient, where x is most recent estimate of the solution, and

$$\mathbf{prox}_g^H(z) := \operatorname*{argmin}_{x \in \mathbf{R}^n} \left\{ \tfrac{1}{2}\|z - x\|_H^2 + g(x) \right\} \tag{1.3.5}$$

is the (scaled) proximal operator of g. The scaled diagonal $H = \alpha I$ is the vanilla proximal operator described in the previous sections. For more general matrices H, however, may lead to proximal operators that dominate the computation. The choice of H remains very much an art - we are trading off complexity in the inner subproblems for less calls to the gradient of f. There is an entire continuum of algorithms that vary based on the choice of H, as illustrated by Table 1.3.1.

Thee table lists a set of algorithms roughly in order of the accuracy with which H approximates the Hessian—when it exists—of the smooth function f. At one extreme is iterative soft thresholding (IST), which approximates the Hessian using a constant diagonal. At the other extreme is proximal Newton, which uses the true Hessian. The quality of the approximation induces a tradeoff between

the number of expected proximal iterations and the computational cost of evaluating the proximal operator at each iteration. Thus H can be considered a preconditioner for the proximal iteration. The proposals offered by [TDKC15] and [BNO16] are flexible in the choice of H, and so those references might also be considered to apply to other methods listed in Table 1.3.1.

1.4. Algorithms with Inexact Oracles

1.4.1. Inexact cutting plane algorithms. The following oracle captures such cases by only requiring an approximate subgradient, as defined by an ϵ-subgradient [Roc70, p. 219].

The ϵ-subgradient, too, gives us a hyperplane that separates x from x^*. However, this hyperplane is not guaranteed to pass through x because

$$f(\bar{x}) \geq f(x) \geq f(x^*) \quad \forall \bar{x} \quad \text{such that} \quad g^T(\bar{x} - x) \geq \epsilon, \quad g \in \partial_\epsilon f(x).$$

If we assume for the moment that $g \neq 0$, then $g^T(\bar{x} - x)$ and so we observe that the hyperplane defined by the ϵ-subgradient is displaced from x in the normalized direction $g/\|g\|$ by the amount $\epsilon/\|g\|$ from x. The smaller the distance $\epsilon/\|g\|$, the more the halfspace slices off P. Therefore this motivates the definition of a scaled inexact oracle wherein we control directly the distance the hyperplane lies from ϵ.

This definition bears resemblance the definition of the shallow-cut oracle described in the classical literature on shallow-cut ellipsoid methods [BGT81, GLS12]. The main difference is the norm in which g is measured. Algorithms which use inexact cutting planes follow this pattern. We start with $P_0 = \mathcal{C}$. Then

$$x_k = \textbf{center}(P_k)$$

$$g_k = \textbf{oracle}_f(x_k, \epsilon_k)$$

$$P_{k+1} = P_k \cap \{x \mid |{g_k}^T(x - x_k) \leq \epsilon\|g\|\}.$$

First, a center is chosen. And then, a shallow cut oracle is called with a specific ϵ_k, one typically determined by P_k, to ensure a consistent rate of decrease. When **center** is the center of gravity, for example, Bertsimas and Vempala [BV04, Theorem 3] give a bound on the guaranteed proportion of decrease in volume if the cutting plane misses the center of gravity by a distance ϵ.

1.4.1.1. *Inexact Bundle Methods.*

1.4.1.2.

1.5. Algorithms with Stochastic Oracles

$$x^+ = x - \alpha g \tag{1.5.1}$$

The use of stochastic oracles have its earliest beginnings in stochastic approximation. For smooth functions, the approximate gradient is used as a drop in replacement of the true gradient

In general, if $\liminf_k \|e_k\| \neq 0$, then we necessarily require $\alpha_k \to 0$ in (1.5.1) in order to ensure optimality of limit points. [CW05, Theorem 3.4] show that the iteration (1.5.1) converges to a solution when $0 < \inf \alpha_k < \sup \alpha_k < 2/L$ and $\sum_{k \in \mathbf{N}} \|e_k\| < \infty$, and also consider other kinds of perturbations that enter into the iteration; no convergence rates are given. [SRB11] link the convergence rate of the iterations (including accelerated variants) to $\mathbf{E}\|e_k\|^2$, which measures the variance in the error, and to error in the computation of the projection. In the case in which $e_k = 0$ has zero mean and finite variance, it is known that the projected-gradient method converges as $\mathcal{O}(1/\sqrt{k})$; see, e.g., [LLZ09]. Contrast these rates to those that can be obtained when $e_k \equiv 0$, and in that case the method has a rate of $\mathcal{O}(1/k)$, and its accelerated variant has a rate of $\mathcal{O}(1/k^2)$, which is an optimal rate; see, e.g., [Nes07] and [BT08].

For the case where $g \equiv 0$ (and hence the projection operator in (1.5.1) is simply the identity map), has been extensively studied. It is well known that if f is strongly convex, deterministic steepest descent without error (i.e., $e_k \equiv 0$) and with a constant stepsize $\alpha_k = 1/L$ converges linearly with a rate constant $\rho < 1$ that depends on the condition number of f; see [LY08, section 8.6]. [BT00] describe conditions for convergence of the iteration (1.5.1) when the steplengths α_k satisfy the conditions $\sum_{k \in \mathbf{N}} \alpha_k = \infty$ and $\sum_{k \in \mathbf{N}} \alpha_k^2 < \infty$. [NB00] show that randomized incremental-gradient methods for (1.2.2.3), with constant steplength $\alpha_k \equiv \alpha$, converge as

$$\mathbf{E}\pi_k \leq \mathcal{O}(1)(\rho^k + \alpha)$$

where $\mathcal{O}(1)$ is a positive constant. This expression is telling because the first term on the right-hand side decreases at a linear rate, and depends on the condition number through ρ; this term is also present for any deterministic first-order method with constant stepsize. For a constant stepsize, [FS12] give non-asymptotic rates that directly depend on the rate at which the error goes to zero, and for the case where f is given by (1.2.2.3), they further show the dependence of the convergence rate on the sample size.

For a non-vanishing stepsize, i.e., $\liminf_k \alpha_k > 0$, [LT93b] show that for a decreasing error sequence that satisfies $\|e_k\| = \mathcal{O}(\|x_{k+1} - x_k\|)$, the function values converge to the optimal value at an asymptotic linear rate.

The convergence in probability of the stochastic-approximation method was first discussed by the classic [RM51] paper. [BT00] give mild conditions on e_k and f under which $f(x_k) \to \inf f(x)$ in probability. More recently, [NJLS09] show that for decreasing steplengths $\alpha_k = \mathcal{O}(1/k)$, these methods achieve a sublinear rate according to $\mathbf{E}\pi_k = \mathcal{O}(1/k)$; the iteration average has similar convergence properties, and it converges sublinearly with overwhelming probability. A similar effort by [NV08] give exponential tail bounds for sequences with any steplength sequence.

1.6. Thesis Outline

CHAPTER 2

The Inexact Epigraphical Cutting Plane Algorithm

2.1. The Inexact Epigraphical Cutting Plane Algorithm

Our proposed algorithm is a modification of the cutting plane algorithms described in Section [] to accomidate We will two consider methods of determining **center**, the center of gravity and the

Algorithm 1: Epigraphical Cutting Plane With Error

$P_0 = C \times [\text{upper bound on } f, \text{lower bound on } f]$
while $\max\{height(P_k), K \cdot size(P_k)^{1/(n+1)}\} \geq TOL$ **do**
1 $(t_k, x_k) \leftarrow \textbf{center}(P_k)$
2 $\epsilon_k \leftarrow \gamma(\min\{u_0, \ldots, u_{k-1}\} - t_k)$
3 $(u_k, l_k, g_k) \leftarrow F(x_k, \epsilon_k)$
4 $P_{k+1} \leftarrow P_k \cap \underbrace{\{(t,x) \mid t \leq u_k\}}_{\text{upper bound}} \cap \underbrace{\{(t,x) \mid l_k + g_k^T(x - x_k) \leq t\}}_{\text{lower bound}}$
5 $k \leftarrow k + 1$
end

center of the maximum inscribed ellipsoid. In this section, we will refer to the height of a convex set to be:

$$\text{height}(P) = \max\{t \mid (t,x) \in P\} - \min\{t \mid (t,x) \in P\}.$$

We will refer to u_k^* as the best upper bound and l_k^* to be the strongest lower bounds.

$$u_k^* = \min\{u_0, \ldots, u_{k-1}\} \qquad l_k^*(x) = \max_{i<k}\{l_i + g_i^T(x - x_i)\}.$$

There is an important property of the localization polytopes P_k our method will exploit. Every polytope P_k can be decomposed into the intersection of two sets, one formed by the intersection of all the lower bounds, other a halfspace formed by the intersection of the upper bounds, i.e.

$$P_k = \text{epi}(l_k^*) \cap \{(t,x) \mid t \leq u_k^*\}$$

The level sets of an epigraph are nested, and $\{x \mid (t,x) \in P_k\}$ is the level set of l^* lower bound at t,

$$t_1 \leq t_2 \Rightarrow \{x \mid (t_1,x) \in P_k\} \subseteq \{x \mid (t_2,x) \in P_k\}$$

and thus every vertical half line emanating from inside the polytope will only exit the polytope at the upper bound. We will refer to this as P_k tapering downwards.

2.1.1. The Center of Gravity Oracle. We begin our exposition with a conceptual algorithm. For this section only, define

$$\textbf{center}(P) := \text{center of gravity of } P = \frac{\int_P z dz}{\int_P dz} \quad \text{and} \quad \text{size}(P) = \text{vol}(P).$$

This definition is valid when P is bounded and has a nonempty interior, which is true for all localization polyopes. The following theorem establishes an important geometric fact about the CG.

Theorem 2.1.1. (Grunbaum's Theorem) Let a be the center of gravity of P. Then for any g,

$$\frac{1}{\exp(1)} \leq \frac{\text{vol}(P \cap \{x \mid g^T(x-a) \leq 0\})}{\text{vol}(P)} \leq 1 - \frac{1}{\exp(1)}$$

The center of gravity can therefore be interpreted as a n dimensional analog of a midpoint, as it divides a convex set into two approximately equal parts.

Lemma 2.1.2. The volumes decrease at the rate

$$\frac{\text{vol}(P_{k+1})}{\text{vol}(P_k)} \leq 1 - \frac{1-\gamma}{\exp(1)}$$

PROOF. **Case 1: (Deep Cut)** $l_k \geq t_k$ **or** $l_k \leq t_k$ **and** $u_k \leq t_k$ The intersection of the cutting planes do not contain (t_k, x_k), therefore by Theorem 2.1.7,

$$\frac{\text{vol}(P_{k+1})}{\text{vol}(P_k)} \leq 1 - \frac{1}{\exp(1)}$$

Case 2: (Shallow Cut) $l_k < t_k$ **and** $u_k > t_k$. See Figure 2.1.1 for an illustration of this case. By the definition of ϵ,

$$u_k - t_k \leq u_k - l_k \leq \gamma(u_k^* - t_k) \quad \Rightarrow \quad t_k + \gamma(u_k^* - t_k) \geq u_k$$

Define $\bar{P}_k$ to be the region above the upper cutting plane, compressed by a small amount from the top

$$\bar{P}_k = \Sigma\big(P_k \cap \{(t,x) \mid t \geq t_k\} - (u_k^*, 0, \ldots, 0)\big) + (u_k^*, 0, \ldots, 0)$$
$$\Sigma = \operatorname{diag}(1-\gamma, 1, \ldots, 1)$$

The bottommost point of $\bar{P}_k$ is $(t_k + \gamma(u_k^* - t_k), x_k)$ which lies above u_k. Furthermore because the set tapers downwards, so $P_k \cap \{(t,x) \mid t \geq u_k\} \supseteq \bar{P}_k$. Therefore, taking volumes on both sides

$$\operatorname{vol}(P_k \cap \{(t,x) \mid t \geq u_k\}) \geq (1-\gamma) \cdot \operatorname{vol}(P_k \cap \{(t,x) \mid t \geq t_k\}) \tag{2.1.1}$$

Now

$$\begin{aligned}
\operatorname{vol}(P_{k+1}) &\overset{(a)}{\leq} \operatorname{vol}(P_k) - \operatorname{vol}(P_k \cap \{(t,x) \mid t \geq u_k\}) \\
&\overset{(b)}{\leq} \operatorname{vol}(P_k) - (1-\gamma) \cdot \operatorname{vol}(P_k \cap \{(t,x) \mid t \geq t_k\}) \\
&\overset{(c)}{\leq} \operatorname{vol}(P_k) - (1-\gamma)e^{-1} \cdot \operatorname{vol}(P_k) = (1 - (1-\gamma)e^{-1}) \cdot \operatorname{vol}(P_k)
\end{aligned}$$

(a) comes from the fact that P_{k+1} is P_k intersected with two cutting planes. The right hand side of (a) is the volume of P_k intersected with just the upper bound. (b) comes from (2.1.1). (c) comes from Theorem 2.1.1. Combing the two results, we obtain the theorem. □

Combining the above result with Lemma 1.3.3, we get a final theorem on the convergence rate of our algorithm.

Theorem 2.1.3. For all k

$$u_k^* - f(x^*) \leq K\left(1 - \frac{1-\gamma}{\exp(1)}\right)^{k/(n+1)} \qquad K = (u_0 - f(x^*))\left(\frac{\operatorname{vol}(P_0)}{\operatorname{vol}(\mathcal{K})}\right)^{1/(n+1)}$$

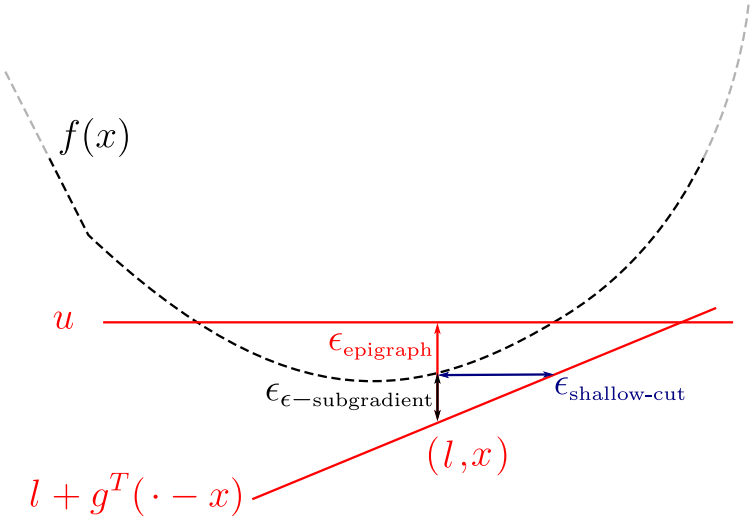

FIGURE 2.1.1. Illustration of a shallow cut with an inexact epigraphical oracle. The dotted line is $f(x)$, the polytope defined by the black solid lines P_k. The black dot is center, and the two red lines are the two cutting planes provided by the oracle.

and when Algorithm 1 with **center** being the center of gravity terminates in no more than

$$k = \left\lceil \frac{(n+1)\log(K/\text{TOL})}{-\log(1-(1-\gamma)/\exp(1))} \right\rceil \in \mathcal{O}\left(n \log \frac{1}{\text{TOL}}\right).$$

We now focus our attention on the amount of work involved in each oracle call. Recall that the cost of invoking the oracle is inversely proportional to ϵ_k. Hence we will show a lower bound on ϵ_k in the next few lemmas.

Lemma 2.1.4. For all k,

$$\epsilon_k \geq \frac{\gamma \cdot \text{height}(P_k)}{(1-\exp(1))(n+1)}$$

PROOF. We first construct a conic inner approximation $\mathcal{K}_k$, and a cylindrical outer approximation $\mathcal{C}_k$ of P_k. See Figure 2.1.2 for an illustration of these approximations. Let $\hat{x}^* \in \operatorname{argmin} l_k^*(x)$, then

$$\mathcal{K}_k := \text{conv}\{u_k^* \times A_k, (l_k^*(\hat{x}^*), \hat{x}^*)\} \subseteq P_k \subseteq [l_k^*(\hat{x}^*), u_k^*] \times A_k =: \mathcal{C}_k$$

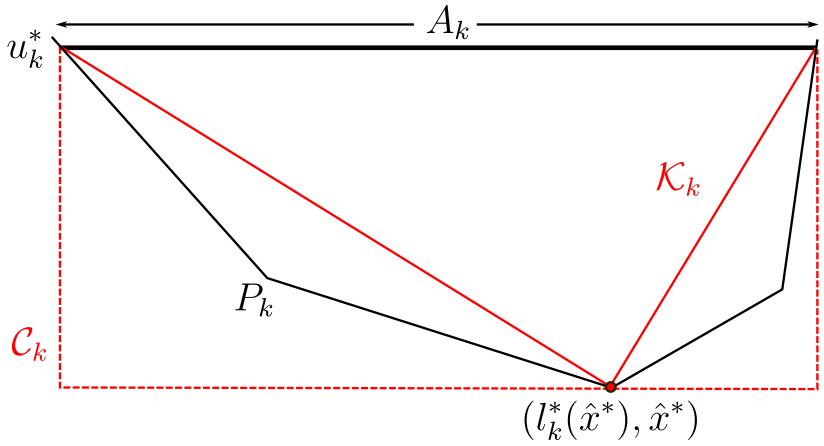

FIGURE 2.1.2. Illustration of outer cylindrical approximation $\mathcal{C}_k$ and inner conic approximation $\mathcal{K}_k$ of $\mathcal{P}_k$

Taking volumes of these sets, we get

$$\text{vol}(P_k) \geq \text{vol}(\mathcal{K}_k) = \text{height}(P_k) \cdot \text{vol}(A_k)/(n+1)$$

$$\text{vol}(P_k \cap \{(t,x) \mid t \geq h\}) \leq \text{vol}(\mathcal{C}_k \cap \{(t,x) \mid t \geq h\}) = (u_k^* - h) \cdot \text{vol}(A_k)$$

Here $\text{vol}(A_k)$ refers to the volume of the n dimensional polytope, not the volume of A_k as a set embedded in $n+1$ dimensional space (which is 0). The final inequalities come from the volume of a cone, and a cylinder, respectively. Now,

$$\begin{aligned}\frac{\epsilon_k(n+1)}{\gamma \cdot \text{height}(P_k)} &\overset{(a)}{=} \frac{(u_k^* - t_k) \cdot \text{vol}(A_k)}{\text{height}(P_k) \cdot \text{vol}(A_k)/(n+1)} \\ &\overset{(b)}{\geq} \frac{\text{vol}(P_k \cap \{(t,x) \mid t \geq t_k\}))}{\text{vol}(P_k)} \overset{(c)}{\geq} 1 - \exp(-1)\end{aligned}$$

(a) comes from the definition of ϵ_k, (b) from our inner and outer approximating bounds, in particular the outer bound with $h = t_k$. And finally (c) from Theorem 2.1.1. We obtain the final result from a simple rearrangement. □

Theorem 2.1.5. Define the total amount of work for each oracle call $F(x, \epsilon) = \epsilon^{-1}$. Assume the algorithm is run till termination. Then the total amount of work involved is

$$\sum \frac{1}{\epsilon_k} \leq 7 \cdot \frac{n+1}{\gamma \text{TOL}} \left\lceil (n+1) \log\left(\frac{K}{\text{TOL}}\right) \right\rceil \in \mathcal{O}\left(\frac{1}{\text{TOL}} \log \frac{1}{\text{TOL}}\right)$$

PROOF. The algorithm terminates when either

$$\text{height}(P_k) \leq \text{TOL} \qquad \text{or} \qquad \text{vol}(P_k) \leq (\epsilon/K)^{n+1}.$$

Lemma 2.1.4 implies the work involved at each iteration will be no more than $1/(100(n+1)\epsilon)$. Theorem 2.1.10 gives a bound on the total number of iterations. This gives

$$\sum \frac{1}{\epsilon_k} \leq \frac{(1-\exp(-1))(n+1)}{\gamma \text{TOL}} \left\lceil \frac{(n+1)\log(K/\text{TOL})}{-\log(1-(1-\gamma)/\exp(1))} \right\rceil$$

Optimizing for γ, we get $\gamma \approx 0.473$. Computing the numerical values and rounding, we obtain the final lower bound. □

Example 2.1.6 (Errors in Epigraphical and shallow-cut oracles)**.** This example demonstrates how the amount of work in the epigraphical cutting plane scales with the amount of uncertainty in the localization polytope. Consider running one iteration of either the shallow-cut cutting plane method (Definition 1.2.2) or the epigraphical variant (Definition 1.2.1) on the one dimensional problem

$$\text{minimum} \quad |x| \quad \text{s.t.} \quad x \geq 0.$$

We use this problem to compare the amount of work required in a single call to the oracle for both algorithms. Let P_x be the localization polytope for the cutting-plane method, and let $\bar{P}_x$ be the localization polytope for the epigraphical cutting plane. For the purposes of fair comparison, we will we will assume the uncertainty in x is fixed, i.e.,

$$\bar{P}_x = \{x \mid (t,x) \in P_x\} = [0,x].$$

Inexact Cutting Plane. We continue with our running assumption that evaluating $g \in \partial_\epsilon \|\cdot\|(x)$ requires $1/\epsilon$ amount of work. Note that g is required to satisfy (1.2.5), and thus the oracle requires

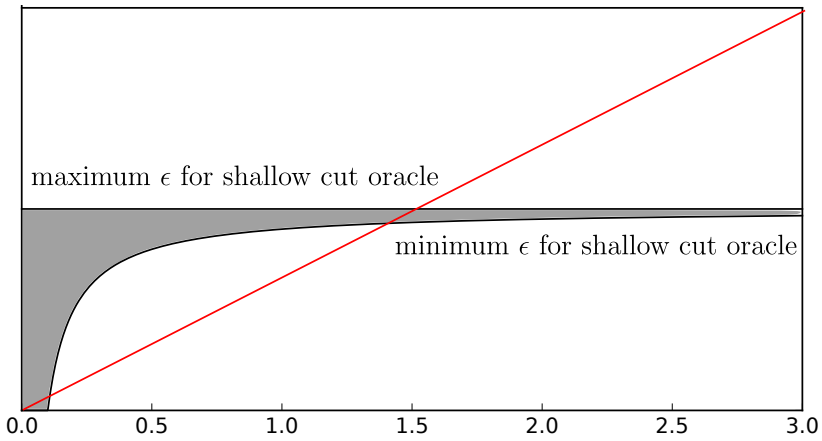

FIGURE 2.1.3. Bounds on amount of work for shallow cut an epigraphical cutting plane methods. The shaded gray region represents the upper and lower bounds for the ϵ needed in the ϵ-subgradient, while the red line represents the amount of work needed in the inexact shallow cut oracle. In epigraphical form, ϵ increases linearly with the size of the polytope.

a cost of $1/(\epsilon\|g\|)$. We can lower bound the amount of work by upper bounding $\|g\|$, which is 1 since

$$g \in \partial_\delta |\cdot|(x) = \begin{cases} [-1, -1 - \delta/x] & x \leq -\delta/2 \\ [-1, 1] & -\delta/2 < x < \delta/2 \\ [1 - \delta/x, 1] & x \geq \delta/2. \end{cases}$$

Therefore the amount of work required is at least $1/\epsilon$.

Inexact Epigraphical Cutting Plane. To find the tightest bound on ϵ, we must find the smallest localization polytope with the property that its projection onto the second coordinate is $[0, x]$. This occurs in the triangle

$$P_x = \mathbf{epi}(|\cdot|) \cap \{(x, t) \mid t \leq x\}.$$

Using the well known fact that the centroid of a right angled triangle lies $1/3$ of the way from the top, the ϵ which we call our oracle is $\epsilon = \gamma x/3$, which increases linearly with x.

While the amount of work when far from x is fixed in the shallow cut case, it dropes linearly with the size in the epigraphical case. For a more detailed illustration of the bounds on the amount of work in the shallow cut oracle, refer to Figure 2.1.3

2.1.2. The Maximum Inscribed Ellipsoid Oracle. [Tar88] proposed an alternate oracle as to remedy the problem of the intractability of the center of gravity. [Tar88] noted that the center of the the maximum inscribed ellipse of a polytope shares many of the properties of the center of gravity, but has the advantage of being computable in polynomial time. Define an ellipse to be

$$\mathcal{E} = \{Ax + a \mid \|x\| \leq 1\}.$$

Let P be a convex, compact set. Define the Maximum Inscribed Ellipsoid (MIE) of P to be ellipse of largest volume contained within P. For this section only, let

$$\textbf{center}(P) := \text{center of MIE of } P.$$

We will also define $\mu(P)$ to be to be the volume of this ellipse. Define, for this section only,

$$\text{size}(P) = \mu(P) = \max\{\text{vol}(\mathcal{E}) \mid \mathcal{E} \text{ is an ellipse such that } \mathcal{E} \subseteq P\}$$

The use of max in place of sup is justified as the set is compact. If

$$P = \{x \mid g_i^T x \leq b_i\}$$

The MIE can be computed by solving

$$\text{maximize } \log\det A \qquad \text{s.t.} \quad \|Ag_i\|_2 + g_i^T a \leq b_i$$

Like the center of gravity, a hyperplane going through the center of the MIE divides a convex set into two approximately even parts:

Theorem 2.1.7. [Tar88] Let $\mathcal{E}$ be the maximum volume ellipse of P. Then for any g,

$$\frac{\mu(P \cap \{x \mid g^T(x - a) \leq 0\})}{\mu(P)} \leq 0.843$$

where a is the center of $\mathcal{E}$.

Here we will require a slightly stronger result, Theorem γ of [Tar88]

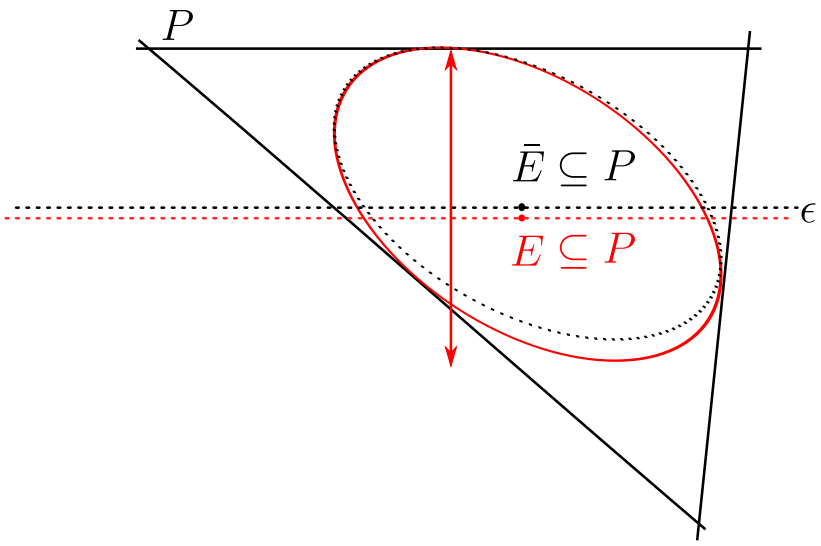

FIGURE 2.1.4. Vertical compression of MIE to an approximate MIE

Lemma 2.1.8. Let $\mathcal{E}$ be an ellipse inscribed in P with relative error γ, i.e.

$$\text{vol}(\mathcal{E}) = (1-\gamma)\mu(P),$$

where a is the center of this ellipse. Then

$$\frac{\mu(P \cap \{x \mid g^T(x-a) \leq 0)}{\mu(P)} \leq \frac{0.843}{(1-\gamma)^2}$$

Then for P_k's generated by Algorithm 1,

Lemma 2.1.9. The volumes decrease at the rate

$$\frac{\mu(P_{k+1})}{\mu(P_k)} \leq \frac{0.843}{(1-\gamma)^2}$$

PROOF. We will split the analysis into two cases.

Case 1: (Deep Cut) $l_k \geq t_k$ **or** $l_k \leq t_k$ **and** $u_k \leq t_k$ The intersection of the cutting planes do not contain (t_k, x_k), therefore by Theorem 2.1.7,

$$\frac{\mu(P_{k+1})}{\mu(P_k)} \leq 0.843$$

Case 2: (Shallow Cut) $l_k < t_k$ **and** $u_k > t_k$. See Figure 2.1.1 for an illustration of this case. In this event, our choice of duality gap ensures that

$$u_k - t_k \leq u_k - l_k \leq \gamma(u_k^* - t_k) \quad \Rightarrow \quad t_k + \gamma(u_k^* - t_k) \geq u_k \tag{2.1.2}$$

Consider the ellipse formed by the following compressive transformation

$$\bar{\mathcal{E}} = \Lambda\left(\mathcal{E} - (u_k^*, 0, \ldots, 0)\right) + (u_k^*, 0, \ldots, 0)$$
$$\Lambda = \operatorname{diag}([1-\gamma, 1, \ldots, 1])$$

The center of this compressed ellipse is $(t_k + \gamma(u_k^* - t_k), x_k)$ which lies at or above (u_k, x_k), by (2.1.2). Therefore the horizontal upper bound cuts off the center of $\bar{\mathcal{E}}$. Also observe that $\bar{\mathcal{E}} \subseteq P$ as P tapers downwards, and the vertical region that lies above $\mathcal{E}$ is all contained in P. Furthermore,

$$\operatorname{vol}(\bar{\mathcal{E}}) = \det(\Lambda) \cdot \operatorname{vol}(\mathcal{E}) = (1-\gamma) \cdot \operatorname{vol}(\mathcal{E})$$

Therefore, $\bar{\mathcal{E}}$ is a $(1-\gamma)$-approximate ellipsoid, and by Lemma 2.1.8

$$\frac{\mu(P_{k+1})}{\mu(P_k)} \leq \frac{0.843}{(1-\gamma)^2}.$$

Combining these two cases yields the final result. □

Applying this result inductively, we get

$$\mu(P_k) \leq \left(\frac{0.843}{(1-\gamma)^2}\right)^k \mu(P_0)$$

Combining the above two results, we get a final theorem on the convergence rate of our algorithm.

Theorem 2.1.10. For all k in Algorithm 1,

$$u_k^* - f(x^*) \leq K\left(\frac{(1-\gamma)^2}{0.483}\right)^{k/(n+1)} \qquad K = (u_0 - f(x^*))\left(\frac{\mu(P_0)}{\mu(\mathcal{K})}\right)^{1/(n+1)}$$

and therefore it terminates in no more than

$$k = \left\lceil \frac{(n+1)\log(K/\mathrm{TOL})}{-\log((1-\gamma)^2/\exp(1))} \right\rceil \in \mathcal{O}\left(n \log \frac{1}{\mathrm{TOL}}\right).$$

iterations with

$$u_k^* - f(x^*) \leq \epsilon$$

We now show a lower bound on ϵ_k, an analog of Lemma 2.1.4.

Lemma 2.1.11. For all k,

$$\epsilon_k \geq \frac{\gamma \operatorname{height}(P_k)}{n+1}$$

PROOF. Since $u_k^* - t_k = \operatorname{height}(P_k \cap \{(t,x) \mid t \geq t_k\})$, we have from Lemma 2.3.4 and the definition of ϵ_k

$$\epsilon_k = \gamma\,(u_k^* - t_k) \geq \frac{\gamma \operatorname{height}(P_k)}{n+1}.$$

□

Finally, we have the analog of Theorem 2.1.5. The result is identical, apart from constants. The proof of this theorem is identical to that of Theorem 2.1.5.

Theorem 2.1.12. Define the total amount of work for each oracle call $F(x,\epsilon) = \epsilon^{-1}$. Assume the algorithm is run till termination. Then the total amount of work involved is

$$\sum \frac{1}{\epsilon_k} \leq 30 \cdot \frac{n+1}{\gamma \mathrm{TOL}} \left\lceil (n+1) \log\left(\frac{K}{\mathrm{TOL}}\right) \right\rceil \in \mathcal{O}\left(\frac{1}{\mathrm{TOL}} \log \frac{1}{\mathrm{TOL}}\right)$$

PROOF. From the bound on the total number of iterations, we get

$$\sum \frac{1}{\epsilon_k} \leq \frac{n+1}{\gamma \mathrm{TOL}} \left\lceil \frac{(n+1)\log(K/\mathrm{TOL})}{-\log((1-\gamma)^2/\exp(1))} \right\rceil \in \mathcal{O}\left(n \log \frac{1}{\mathrm{TOL}}\right).$$

Optimizing for γ numerically, we find that the optimum occurs at $\gamma = 0.41$. Substituting this into the equation, we get the final result. □

Algorithm 2: Epigraphical Cutting Plane With Error

$P_0 = C \times [\text{upper bound on } f, \text{lower bound on } f]$
while $\max\{height(P_k), K \cdot size(P_k)^{1/(n+1)}\} \geq TOL$ **do**
1 $(t_k, x_k) \leftarrow \textbf{center}(P_k)$
2 $\epsilon_k \leftarrow \gamma(\min\{u_0, \ldots, u_{k-1}\} - t_k)$
3 $(u_k, l_k, g_k) \leftarrow F(x_k, \epsilon_k)$
4 $P_{k+1} \leftarrow P_k \cap \underbrace{\{(t,x) \mid t \leq u_k\}}_{\text{upper bound}} \cap \underbrace{\{(t,x) \mid l_k + g_k^T(x - x_k) \leq t\}}_{\text{lower bound}}$
5 $k \leftarrow k + 1$
end

2.2. Strongly Convex Functions

We can modify our algorithm to exploit additional structure, if available, in the problem. Consider the regularized problem

$$\text{minimize} \quad f(x) + \tfrac{S}{2}\|x\|^2 \qquad x \in C,$$

where f is convex, real valued and has a quadratic upper bound at the optimum,

$$f(x) \leq \tfrac{L}{2}\|x - x^*\|^2 + f(x^*). \tag{2.2.1}$$

The extra regularization parameter allows us to generate parabolic, rather than linear lower bounds on the problem:

$$\text{for any } x, \epsilon \quad (u, l, g) = F(x, \epsilon), \qquad l + g^T(\bar{x} - x) + \tfrac{S}{2}\|\bar{x}\|^2 \leq f(\bar{x}) \text{ for all } \bar{x}.$$

It is a simple matter to incorporate these lower bounds in this problem, and this results in Algorithm (2). These two modifications will enable us to prove a stronger result on the total work involved. Indeed, in contrast to the $\mathcal{O}(\epsilon^{-1} \log \epsilon^{-1})$ amount of work required before, we now require $\mathcal{O}(\epsilon^{-1})$ work. For simplicity, we will assume that **center** is the center of the MIE, and size $= \mu$ though all the results here can be proven for the CG in a straightfoward manner.

Let us consider why such a regularized system might arise in practice. First, note that any unregularized, but strongly convex problem can be written in this form (by adding and subtracting a quadratic term). Therefore this regularized form of the problem is merely syntactic sugar - we are really dealing with strongly convex functions. The second assumption of smoothness holds true if f

is smooth and has L-Lipshitz gradients. This is true, for example, if f is a dual of a strongly convex primal problem. (TODO: Try to work in a reference to Geometric Descent, a similar algorithm which uses parabolic lower bounds. [DFR16])

In the context of constrained optimization, the regularized problem can be interpreted as using a squared penalty in lieu of hard constraints, as the next example demonstrates.

Example 2.2.1. As noted in the motivation of the introduction, we are interested in duals of optimization problems with few constraints. Recall the Lagrangian

$$\mathcal{L}(v, x) = h_0(v) + x_1 h_1(v) + \cdots + x_n h_n(v).$$

Of course, as we have shown, maximizing the variable x over the set $\mathbf{R}_+$ will recover the original constrained problem. Instead of doing that let us regularize the dual variables with a quadratic term, as shown:

$$\bar{\mathcal{L}}(x, v) = h_0(v) + x_1 h_1(v) + \cdots + x_n h_n(v) - \tfrac{S}{2}\|x\|^2.$$

Now taking infimums over v, we see this is exactly the regularized problem

$$-\inf_v \bar{\mathcal{L}}(x, v) = f(x) + \tfrac{S}{2}\|x\|^2.$$

Taking supremums instead over x over $\bar{\mathcal{L}}$, we arrive at an interpretation for the regularization. We have replaced the hard constraints with smooth, quadratic penalties in the objective,

$$\sup_{x \geq 0} \bar{\mathcal{L}}(x, v) = h_0(v) + \tfrac{1}{S}\big(\max\{0, h_1(v)^2\} + \cdots + \max\{0, h_n(v)^2\}\big).$$

If we have upper bounds on the Lagrange multipliers (our variables x), $0 \leq x \leq \alpha$, then the quadratic penalties turn into huber penalties.

Lemma 2.2.2. For all k,

$$u_k^* - f(x^*) \leq \mathcal{O}(1) \cdot n[L^n \mu(P_k)^2]^{1/(2+n)}.$$

where $\mathcal{O}(1)$ is a universal constant.

PROOF. From Assumption 2.2.1 we have the following quadratic upper bound on f,

$$v := \tfrac{L}{2}\|\cdot - x^*\|^2 + f(x^*),$$

i.e. $v \geq f$. Define the two sets $\mathcal{K}_k$, $\mathcal{P}_k$ for which $\mathcal{K}_k \subseteq \mathcal{P}_k \subseteq P_k$,

$$\mathcal{K}_k = \operatorname{conv}\{u_k^* \times \operatorname{lvl}(v, u_k^*), (f(x^*), x^*)\}. \quad \mathcal{P}_k = \operatorname{epi}(v) \cap \{(t, x) \mid t \leq u_k^*\},$$

Also, let $\mathcal{K}$ be a unit circular cone

$$\mathcal{K} = \operatorname{conv}\big\{\{(1, x) \mid \tfrac{1}{2}\|x\|^2 \leq 1\}, 0\big\}.$$

First note that $\mathcal{K}_k$ and $\mathcal{K}$, our standard cone, are related by a affine transformation. This transform which first involves the translation of the tip of the cones to $(0, 0)$, and then a scaling.

$$\mathcal{K} = \Sigma_k^{-1}(\mathcal{K}_k - (f(x^*), x^*)) \tag{2.2.2}$$

$$\Sigma_k = \operatorname{diag}([u_k - f(x^*), ([u_k - f(x^*)]/L)^{1/2}, \ldots, ([u_k - f(x^*)/L])^{1/2}) \tag{2.2.3}$$

Therefore, the following chain of inequalities follow:

$$\begin{aligned}\mu(P_k) &\overset{(a)}{\geq} \mu(\mathcal{P}_k) \\ &\overset{(b)}{\geq} \mu(\mathcal{K}_k) \overset{(c)}{=} \det(\Sigma_k)\mu(\mathcal{K}) = \mu(\mathcal{K})[u_k - f(x^*)]^{(2+n)/2}/L^{n/2}\end{aligned}$$

(a) comes from Assumption (2.2.1), which implies $P_k \subseteq \mathcal{P}_k$. (b) follows from the fact that $\mathcal{P}_k \supseteq \mathcal{K}_k$, by construction ($\mathcal{K}_k$ is a circular cone inscribed in the parabola $\mathcal{P}_k$.) (c) comes from translating and scaling $\mathcal{K}_k$ into the standard cone, equation (2.2.2), the affine invariance of the MIE, and the affine homogeneity of μ. Taking powers of $2/(2+n)$ on both sides, we get the final result

$$u_k^* - f(x^*) \leq \frac{L^{n/(2+n)}\mu(P_k)^{2/(2+n)}}{\mu(\mathcal{K})^{2/(2+n)}} \leq \mathcal{O}(1) \cdot n[L^n \mu(P_k)^2]^{1/(2+n)}.$$

For the final inequality, we use the fact that

$$\mu(\mathcal{K})^{2/(2+n)} \geq \frac{\omega_n^{2/(2+n)}}{4^{\frac{2n+2}{2+n}}} \geq n\mathcal{O}(1)$$

for all n. □

This theorem can be seen as a quadratic improvement over Lemma 1.3.2. In Lemma 1.3.2 we proved $u_k^* - f(x^*) \lessapprox \mu(P_k)^{1/(n+1)}$. This theorem improves that bound to $u_k^* - f(x^*) \lessapprox \mu(P_k)^{2/(2+n)}$. This fact allows us to relax the termination condition by a squared factor. Note too, that this improvement does not depend on the parabolic cutting planes, and hence can be integrated with the results in the previous section. This results in

Theorem 2.2.3. Algorithm 1 terminates in no more than

$$k = \left\lceil \frac{(2+n)\log(1/\epsilon) + 2\log\mu(P_0) + n\log L}{\log(1/0.878^2)} \right\rceil \in \mathcal{O}\left(n\log\frac{1}{\epsilon}\right)$$

iterations. When Algorithm 2 terminates,

$$u_k^* - f(x^*) \leq \epsilon$$

The improvement above is not significant, and does not change the algorithms complexity class. The above statement merely establishes that this new stopping criteria is valid. The next theorem establishes a lower bound on ϵ_k as a function of $\mu(P_k)$. The parabolic lower bounds, intuitively, prevent P_k from becoming too flat.

Lemma 2.2.4. For all k

$$\epsilon_k \geq \mathcal{O}(1) \cdot [S^n \mu(P_k)^2]^{1/(2+n)}$$

Where $\mathcal{O}(1)$ is a universal constant.

Proof. Let l_k^* be the strongest lower bound at iteration k,

$$l_k^* = \max_{i<k}\{l_i + g_i^T(\cdot - x_i) + \tfrac{S}{2}\|\cdot\|^2\}.$$

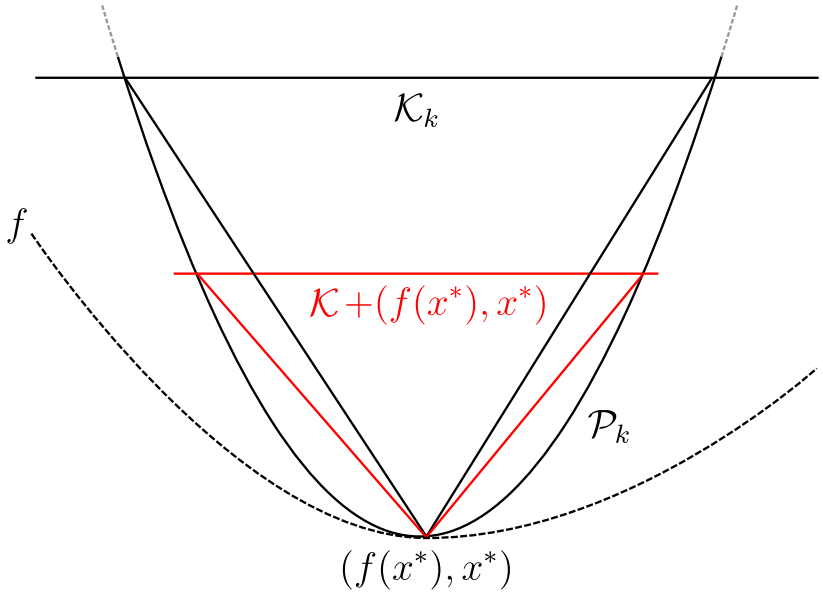

FIGURE 2.2.1. Illustration of inner approximating cones, and the rescaling by Σ_k necessary to transform it into the unit cone.

Since each function within the max is strongly convex with modulus S, so must l_k^* (strong convexity is preserved by the max). Let $\hat{x}^*$ be the minimizer of l_k^*. Since $0 \in \partial l_k^*(\hat{x}^*)$ (see Figure 2.2.2),

$$l_k^*(x) \geq l_k^*(\hat{x}^*) + 0^T(x - \hat{x}^*) + \tfrac{S}{2}\|x - \hat{x}^*\|^2 \geq l_k^*(\hat{x}^*) + \tfrac{S}{2}\|x - \hat{x}^*\|^2,$$

This implies $\operatorname{epi}(l_k^*) \subseteq \operatorname{epi}(l_k^*(\hat{x}^*) + \frac{S}{2}\|\cdot - \hat{x}^*\|^2)$ and therefore

$$P_k = \operatorname{epi}(l_k^*) \cap \{(t,x) \mid t \leq u_k^*\} \tag{2.2.4}$$

$$\subseteq \operatorname{epi}(l_k^*(\hat{x}^*) + \tfrac{S}{2}\|\cdot - \hat{x}^*\|^2) \cap \{(t,x) \mid t \leq u_k^*\} =: \mathcal{P}_k \tag{2.2.5}$$

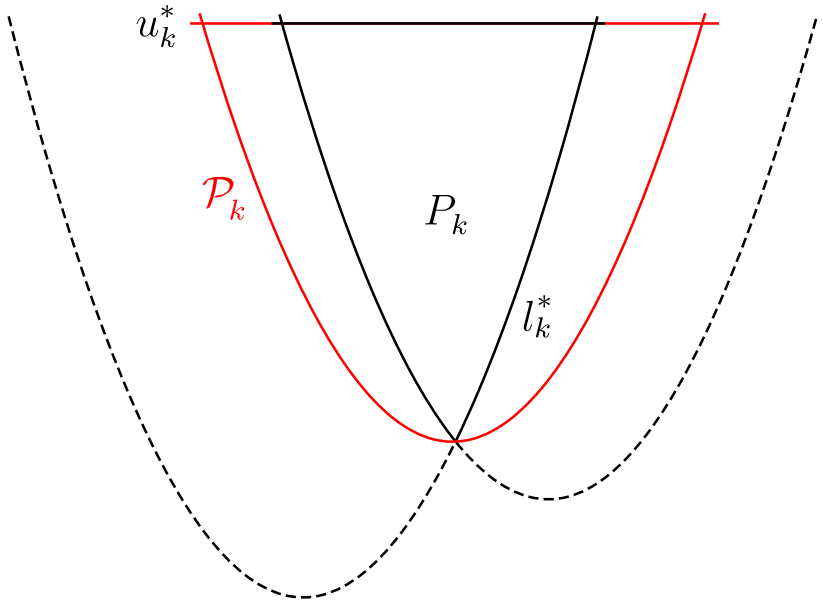

FIGURE 2.2.2. Geometric Illustration of the parabolic outer approximation of P_k, i.e. $P_k \subseteq \mathcal{P}_k$.

See Figure 2.2.2 for an illustration. Lete ω_n be the volume of a unit n-ball.

$$\begin{aligned}
\mu(P_k) &\overset{(a)}{\leq} \text{vol}(P_k) \\
&\overset{(b)}{\leq} \text{vol}(\mathcal{P}_k) \\
&\overset{(c)}{=} \frac{2\omega_n}{n[S/2]^{n/2}} \cdot \text{height}(\mathcal{P}_k)^{(2+n)/2} \\
&\overset{(d)}{=} \frac{2\omega_n}{n[S/2]^{n/2}} \cdot \text{height}(P_k)^{(2+n)/2} \\
&\overset{(e)}{\leq} \frac{2\omega_n}{n[S/2]^{n/2}} \cdot [(n+1)\epsilon_k]^{(2+n)/2}
\end{aligned}$$

(a) comes from the fact that $\mu(P_k) = \text{vol}(\mathcal{E}_k)$, and $\mathcal{E}_k$, being an inscribed ellipse, is a subset of P_k. (b) comes from (2.2.4). Finally, (c) is the volume of a parabola. (c) is from construction (see Figure 2.2.2), and (d) comes from Lemma 2.3.4. Taking powers of $2/(2+n)$ on both sides, we obtain

$$\epsilon_k \geq \frac{n^{2/(2+n)}}{2} \cdot \frac{[\mu(P_k)^2 S^n]^{1/(2+n)}}{(n+1)[\omega_n]^{2/(2+n)}} \geq \mathcal{O}(1) \cdot [S^n \mu(P_k)^2]^{1/(2+n)}$$

The final inequality comes from the Lemma 2.3.5. □

Finally, we bound the total amount of work involved in the algorithm. We assume that the amount of work is inversely proportional to ϵ.

Theorem 2.2.5. Assume the algorithm is run till termination. Then for $\epsilon > 0$

$$\sum \frac{1}{\epsilon_k} \leq \frac{L}{S} \cdot \frac{\mathcal{O}(1)}{1 - 0.878^{2/(2+n)}} \cdot \frac{n}{\epsilon}$$

Where $\mathcal{O}(1)$ represents a universal constant.

PROOF. Let $\mu_k := \mu(P_k)^{2/(2+n)}$. Then we have the following three guarantees.

$$\begin{aligned}
&\epsilon_k \geq C_1 \mu_k && \text{(Lower bound on } \epsilon_k\text{, Lemma 2.2.4)} \\
&\mu_k \geq C_2 \epsilon && \text{(Termination criteria, Algorithm 2)} \\
&\mu_k \geq 0.878^{-2/(2+n)} \cdot \mu_{k+1} && \text{(Convergence guarantee, Lemma 2.1.9)}
\end{aligned}$$

where

$$C_1 = \mathcal{O}(1) \cdot S^{n/(2+n)}, \qquad C_2 = \mathcal{O}(1) \cdot L^{-n/(2+n)}$$

Therefore, the total cost of the iteration is

$$\sum \frac{1}{\epsilon_k} \leq \sup\{\alpha_0, \alpha_1, \dots\},$$

where α_j is the total amount of work performed if the algorithm ran for j steps. We can upper bound the total amount of work performed by performing an optimization over the three constraints described above. Let $\beta = 0.878^{-2/(2+n)}$. Then

$$\begin{aligned}
\alpha_j &\leq \sup\{ \textstyle\sum_{k=0}^{j} \epsilon_k^{-1} \mid \mu_0 \geq \beta\mu_1, \cdots, \mu_{j-1} \geq \beta\mu_j \text{ and } \mu_k \geq C_2\epsilon, \epsilon_k \geq C_1\mu_k, \forall k\} \\
&= \sup\{ \textstyle\sum_{k=0}^{j} \epsilon_k^{-1} \mid \mu_0 \geq \beta\mu_1, \cdots, \mu_{j-1} \geq \beta\mu_j \text{ and } \mu_k \geq C_2\epsilon, \epsilon_k = C_1\mu_k, \forall k\} \\
&= \sup\{ \textstyle\sum_{k=0}^{j} [C_1\mu_k]^{-1} \mid \mu_0 \geq \beta\mu_1, \cdots, \mu_{j-1} \geq \beta\mu_j, \mu_j \geq C_2\epsilon\} \\
&\overset{(1)}{=} \frac{\sum_{k=0}^{j} \beta^{-k}}{C_1C_2\epsilon} \leq \frac{\sum_{k=0}^{\infty} \beta^{-k}}{C_1C_2\epsilon} = \frac{1}{(1-\beta^{-1})C_1C_2\epsilon}
\end{aligned}$$

Inequality (1) comes from Lemma 2.3.3. Our objective function is convex, and bounded from above on the feasible set (the function is decreasing, and the feasible set does not contain 0). By writing the constraints as

$$\begin{pmatrix} 1 & -\beta & & \\ & \ddots & \ddots & \\ & & 1 & -\beta \\ & & & 1 \end{pmatrix} \begin{pmatrix} \mu_0 \\ \vdots \\ \mu_{j-1} \\ \mu_j \end{pmatrix} \geq \begin{pmatrix} 0 \\ \vdots \\ 0 \\ C_2\epsilon \end{pmatrix}.$$

The statement of the lemma states that the optimum is achieved at equality. Therefore the optimum is achieved at $\mu_k = \beta^{j-k} C_2 \epsilon$. Substituting this back into the objective and reversing the order of summation yields the equality. Finally,

$$\frac{1}{C_1 C_2} = \frac{\mathcal{O}(1) \cdot nL^{n/(2+n)}}{\mathcal{O}(1) \cdot S^{n/(2+n)}} \leq \mathcal{O}(1) \cdot \frac{nL}{S}$$

as $L/S \geq 1$ which implies $(L/S)^p \leq L/S$ for $p \leq 1$ □

Surprisingly, there are no constants in the above proof which involve $\mu(P_0)$. This counterintuitive fact means the volume of the initial localization polytope plays no role in the optimization. How can this be possible? Perhaps this can be understood with the following intuition. Since the iterates at the beginning get exponentially cheaper as the polytope P_0 increases, in a sense, early iterations contribute very little to the work involved. In practice, of course, this is not the case. There is some, finite fixed cost in solving any optimization problem, which cannot be atomically divided. However we believe that this property still translates to much better performance in practice.

2.3. Some General Facts

Lemma 2.3.1. (**Volume of a Parabolid**) Let ω_n be the volume of the n unit ball. Then

$$\text{vol}(\{(t, x) \mid \tfrac{K}{2}\|x\|^2 \leq t \leq h\}) = \frac{2\omega_n}{n[K/2]^{n/2}} \cdot h^{(2+n)/2}.$$

Proof. This proof is an application of the "disk method" in calculus. Since

$$\frac{K}{2}\|x\|^2 \leq t \iff \|x\| \leq \sqrt{2t/K},$$

each slice of $\mathcal{P}$ in the first dimension is a $n-$ball of radius $\sqrt{2t/K}$, with area $\omega_n(2t/K)^{n/2}$. We integrate this from 0 to h, i.e.,

$$\text{vol}(\mathcal{P}) = \omega_n \int_0^h [2t/K]^{n/2} dt = \frac{\omega_n}{[K/2]^{n/2}} \int_0^h t^{n/2} dt = \frac{2\omega_n}{n[K/2]^{n/2}} \cdot h^{(n+2)/2}.$$

□

Lemma 2.3.2. (Volume approximations for MIE of the unit circular cone) Let

$$\mathcal{K} = \text{conv}\big\{\{(1,x) \mid \tfrac{1}{2}\|x\|^2 \leq 1\}, 0\big\}.$$

Then

$$\frac{\omega_{n+1}}{4^{n+1}} \leq \mu(\mathcal{K}) \leq \frac{2^{n/2+1}\omega_n}{n}$$

PROOF. (**Lower Bound)** We can inscribe a $n+1$ dimensional ball of radius $1/(2\sqrt{1.25}+1)$ in the unit cone. Since the MIE is larger than this,

$$\frac{\omega_{n+1}}{(2\sqrt{1.25}+1)^{n+1}} \leq \mu(\mathcal{K}).$$

(**Upper Bound)** Since $\mu(\mathcal{K}) \leq \text{vol}(\mathcal{K})$, we can use Lemma 2.3.1 □

Lemma 2.3.3. (**Maximization of a convex function over a cone**) Let f be a convex function, and A be a square, invertible matrix. Furthermore assume that f is bounded on the cone $\{x \mid Ax \geq b\}$. Then

$$\sup_x \{f(x) \mid Ax \geq b\} = f(A^{-1}b).$$

PROOF. We transform the problem to

$$\sup_x \{f(x) \mid Ax \geq b\} = \sup_x \{f(A^{-1}(x+b)) \mid x \geq 0\} = \sup_x \{g(x) \mid x \geq 0\}$$

Where $g(x) = f(A^{-1}(x+b))$. By assumption, the function g is bounded, and therefore is decreasing on every half line [Roc70], and achieves the maximum at $g(0)$. Therefore, g must attain a global minimum at $g(0) = f(A^{-1}b)$. □

Lemma 2.3.4. Let $P \subseteq R^{n+1}$ be a compact polytope, and assume that the center of the maximum volume elipsoid of P is a. Then

$$\frac{1}{n+1} \leq \frac{\text{height}(P \cap \{(t,x) \mid t \geq a_1\})}{\text{height}(P)} \leq \frac{n}{n+1}$$

PROOF. Let $\mathcal{E}$ be the MIE of P. Assume, w.l.o.g, $\mathcal{E}$ is centered at 0. By [Tar88] we know that

$$\mathcal{E} \subseteq P \subseteq n\mathcal{E}$$

Define the Proj to be the projection of the set P onto the first coordinate,

$$\text{Proj}(P) = \{t \mid (t, \bar{x}) \in P\}$$

Since $\mathcal{E}, P, n\mathcal{E}$ are all convex, their corrosponding projections are intervals. Since $\mathcal{E}, n\mathcal{E}$ are both ellipses centered at 0, we can assume then without loss of generality (since the ratios are preserved)

$$\text{Proj}(\mathcal{E}) := [-1,1], \quad \text{Proj}(P) := [-x_1, x_2], \quad \text{Proj}(n\mathcal{E}) := [-n,n]$$

Using the fact that

$$\text{Proj}(\mathcal{E}) \subseteq \text{Proj}(P) \subseteq \text{Proj}(n\mathcal{E})$$

we have

$$[-1,1] \subseteq [-x_1, x_2] \subseteq [-n,n]$$

Then for $H = \{(t,x) \mid t \geq a_1\})$

$$\frac{\text{height}(P \cap H)}{\text{height}(P)} = \frac{x_1}{x_2 + x_1} \leq \sup \left\{ \frac{x_1}{x_2 + x_1} \;\middle|\; \begin{array}{l} 1 \leq x_1 \leq n \\ 1 \leq x_2 \leq n \end{array} \right\} = \frac{n}{n+1}$$

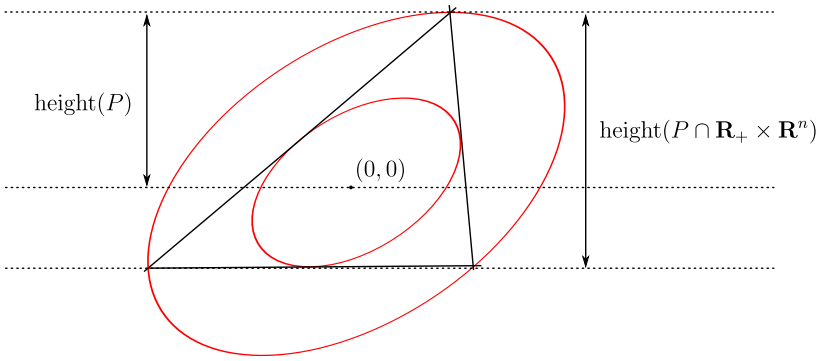

FIGURE 2.3.1. Case where lower bound is tight in $2d$. The $n-$dimensional simplex achieves this bound in general.

The final equality can be found by solving a simple $2d$ optimization problem over a box, for which we can exhaustively search all critical points. Replacing the sup with an inf, we obtain the other bound. □

Lemma 2.3.5. Let ω_n be the volume of a unit sphere. Then

$$\frac{2}{n+1} \leq \omega_n^{2/(2+n)} \leq \frac{12}{n+1}$$

PROOF. Multiplying both sides by $n+1$, we get

$$\begin{aligned}
(n+1)\omega_n^{2/(2+n)} &= (n+1)\left(\frac{\pi^{\frac{n}{2}}}{\Gamma(\frac{n}{2}+1)}\right)^{2/(2+n)} \\
&\overset{(a)}{\leq} (n+1)\left(\frac{\pi^{\frac{n}{2}}}{\sqrt{2\pi}n^{\frac{n+3}{2}}e^{-\frac{n+2}{2}}}\right)^{2/(2+n)} \\
&= (n+1)\frac{\pi^{\frac{n}{2+n}}}{\sqrt{2\pi}n^{\frac{n+3}{n+2}}e^{-1}} \\
&= \frac{n+1}{n^{1+\frac{1}{n+2}}}\frac{\pi^{\frac{n}{2+n}}}{\sqrt{2\pi}e^{-1}} \\
&\leq \left(1+\frac{1}{n}\right)\frac{3\pi e}{\sqrt{2\pi}} \leq \frac{6\pi e}{\sqrt{2\pi}} \leq 12
\end{aligned}$$

(a) comes from sterling's lower bound. The lower bound is proved in an analogous way

$$\begin{aligned}
(n+1)\omega_n^{2/(2+n)} &= (n+1)\left(\frac{\pi^{\frac{n}{2}}}{\Gamma(\frac{n}{2}+1)}\right)^{2/(2+n)} \\
&\geq (n+1)\left(\frac{\pi^{\frac{n}{2}}}{\sqrt{2\pi}\cdot n^{n+\frac{1}{2}}e^{\frac{1}{12n}-n}}\right)^{2/(2+n)} \\
&= (n+1)\frac{\pi^{\frac{n}{2+n}}}{\sqrt{2\pi}\cdot n^{\frac{n+3}{2+n}}e^{\left(\frac{2}{2+n}\right)\left(\frac{1}{12n}-n\right)}} \\
&\geq \frac{n+1}{n}\frac{10}{\sqrt{2\pi}}
\end{aligned}$$

□

CHAPTER 3

Proximal Gradient With Error

We will assume for simplicity that $\alpha_k \equiv 1/L$. We will make the following blanket assumptions about f. First, the solution set $\mathcal{S}^*$ of (1.3.2) is nonempty. For all x and y, there exist positive constants L and $\tau \geq 1$ such that

$$\|\nabla f(y) - \nabla f(x)\| \leq L \|y - x\|, \tag{3.0.1a}$$

$$\min_{\bar{x} \in \mathcal{S}^*} \|x - \bar{x}\| \leq \tau \|x - \mathbf{prox}_{1/L}(x - \tfrac{1}{L}\nabla f(x))\| \qquad \forall x \in \mathrm{dom}(g) \tag{3.0.1b}$$

Assumption (3.0.1a) asserts the Lipschitz continuity of the gradient of f. Assumption (3.0.1b) is an error bound on the generalized residual. This generalized residual has been explored in local contexts in Tseng and Yun [TY09] and Luo and Tseng [LT93a]; for simplicity our assumption is stronger, however, requiring the bound to be global.

τ can be seen as a nonsmooth proxy for the condition number. If $g \equiv 0$ and f is strongly convex with parameter μ, then (3.0.1b) holds with $\tau = L/\mu$. More generally, if g is an indicator function on a polyhedral set and f is strongly convex, this bound holds globally, with parameter $\tau = (L+1)/\mu$. [Pan87] Theorem 3.1. Recently, a global version of this error bound has been developed for non-strongly convex functions which degrades with the size of the neighborhood see [WL14] Theorem 18. We can use such a bound if the function g is an indicator over a polyhedral set, and f can be written in the form

$$f(x) = r(Ax) + b^T x$$

for any A, b, and r is strongly convex. If the above two conditions hold, then

Lemma 3.0.1. Let $\pi_k = f(x_k) - \min f(x)$. Then after k iterations of algorithm (1.3.3),

$$\pi_k \leq \rho^k \pi_0 + \frac{1}{\vartheta} \sum_{i=0}^{k-1} \rho^{k-1-i} \|e_i\|^2,$$

where

$$\rho = 1 - \frac{1}{1+40\tau^2} \in (0,1) \qquad \text{and} \qquad \vartheta = L \cdot \left(\frac{1}{40\tau^2} + 1\right) > 0.$$

The proof of this result follows the template laid out by Luo and Tseng [LT93b, Theorem 3.1], modified to keep the error term e_k explicit. So [So13] also provides a similar derivation for the case where $g \equiv 0$, in which case it seems possible to obtain tighter constants ρ and ϑ. If additionally $\|e_k\| = 0$, then the result reduces to the well-known fact that steepest descent decreases the objective value linearly. The convergence rate, as expected, is a function of the condition number, We note that the constants are invariant to scalings of $f + g$.

Lemma 3.0.2 (Three-point property with error). For all $y \in \operatorname{dom}(g)$,

$$g(y) \geq g(x_{k+1}) + (\nabla f(x_k) + e_k)^T x_{k+1} - y + \frac{L}{2}\|x_{k+1} - x_k\|^2 + \frac{L}{2}\|y - x_{k+1}\|^2 - \frac{L}{2}\|y - x_k\|^2.$$

Proof. Let $\psi_k(x) := g(x) + f(x_k) + \langle \nabla f(x_k) + e_k, x - x_k\rangle + \frac{L}{2}\|x - x_k\|^2$. Because ψ_k is strongly convex,

$$\psi_k(y) \geq \psi_k(x) + q^T y - x + \frac{L}{2}\|y - x\|^2 \quad \text{for all } x, y \text{ and all } q \in \partial\psi_k(x).$$

Choose $x = x_{k+1} := \operatorname{argmin} \phi_k(x)$. Because $0 \in \partial\psi_k(x_{k+1})$, we have

$$\psi_k(y) \geq \psi_k(x_{k+1}) + \frac{L}{2}\|y - x_{k+1}\|^2,$$

which, after simplifying, yields the required result. □

Lemma 3.0.3. Let $\bar{x}_k$ be the projection of x_k onto $\mathcal{S}^*$. Then

$$\|x_k - \bar{x}_k\| \leq \tau\|x_k - x_{k+1}\| + \frac{\tau}{L}\|e_k\|; \tag{3.0.2a}$$

$$\|x_k - \bar{x}_k\|^2 \leq 2\tau^2\|x_k - x_{k+1}\|^2 + \frac{5}{4}(\tau^2/L^2)\|e_k\|^2; \tag{3.0.2b}$$

$$\|x_{k+1} - \bar{x}_k\| \leq (1+\tau)\|x_k - x_{k+1}\| + \frac{\tau}{L}\|e_k\|; \tag{3.0.2c}$$

$$\|x_{k+1} - \bar{x}_k\|^2 \leq \frac{1}{2}[2 + 5\tau + 3\tau^2]\|x_k - x_{k+1}\|^2 + \tfrac{1}{2L^2}[3\tau^2 + \tau]\|e_k\|^2. \tag{3.0.2d}$$

PROOF. For all k,

$$\begin{aligned}
\|x_k - \bar{x}_k\| &\overset{(i)}{\leq} \tau\|x_k - [x_k - \tfrac{1}{L}\nabla f(x_k)]_+\| \\
&\leq \tau\|x_k - x_{k+1}\| + \tau\|x_{k+1} - [x_k - \tfrac{1}{L}\nabla f(x_k)]_+\| \\
&= \tau\|x_k - x_{k+1}\| + \tau\|[x_k - \tfrac{1}{L}(\nabla f(x) + e_k)]_+ - [x_k - \tfrac{1}{L}\nabla f(x_k)]_+\| \\
&\overset{(ii)}{\leq} \tau\|x_k - x_{k+1}\| + \tfrac{\tau}{L}\|e_k\|,
\end{aligned}$$

where (i) follows from Assumption (3.0.1b) and (ii) follows from the nonexpansiveness of the proximal operator.

Part (3.0.2b)**.** Square both sides of (3.0.2a) and then apply the inequality

$$ab \leq \frac{a^2}{2\alpha} + \frac{\alpha b^2}{2}, \quad \forall \alpha > 0, \tag{3.0.3}$$

to bound the cross terms:

$$\begin{aligned}
\|x_k - \bar{x}_k\|^2 &\leq \tau^2\|x_k - x_{k+1}\|^2 + (\tau/L)^2\|e_k\|^2 + (\tau^2/L)\|x_k - x_{k+1}\|\|e_k\| \\
&\leq \left(\tau^2 + \tfrac{\tau^2\alpha}{2L}\right)\|x_k - x_{k+1}\|^2 + \left(\tfrac{\tau^2}{L^2} + \tfrac{\tau^2}{2L\alpha}\right)\|e_k\|^2 \qquad (\forall \alpha > 0) \\
&\leq 2\tau^2\|x_k - x_{k+1}\|^2 + \tfrac{5}{4}(\tau^2/L^2)\|e_k\|^2.
\end{aligned}$$

Part (3.0.2c)**.** Use the triangle inequality and (3.0.2a):

$$\|x_{k+1} - \bar{x}_k\| \leq \|x_{k+1} - x_k\| + \|x_k - \bar{x}_k\| \leq (1+\tau)\|x_k - x_{k+1}\| + (\tau/L)\|e_k\|.$$

Part (3.0.2d)**.** Square both sides above, and use the same technique used in Part (3.0.2b) to bound the cross-terms:

$$\|x_{k+1} - \bar{x}_k\|^2 \leq \tfrac{1}{2}(2 + 5\tau + 3\tau^2)\|x_k - x_{k+1}\|^2 + \tfrac{1}{2L^2}(3\tau^2 + \tau)\|e_k\|^2.$$

□

Lemma 3.0.4 (Sufficient decrease)**.** For all k,

$$\pi_{k+1} \leq \left(1 - \frac{1}{1+40\tau^2}\right)\pi_k + \frac{1}{L}\cdot\frac{40\tau^2}{1+40\tau^2}\|e_k\|^2.$$

Proof. First, specialize Lemma 3.0.2 with $y = x_k$:

$$g(x_{k+1}) \leq g(x_k) - \langle \nabla f(x_k) + e_k, x_{k+1} - x_k\rangle - L\|x_{k+1} - x_k\|^2. \tag{3.0.4}$$

Then,

$$\begin{aligned}
h(x_{k+1}) &\overset{(i)}{\leq} f(x_k) + \langle \nabla f(x_k), x_{k+1} - x_k\rangle + \frac{L}{2}\|x_{k+1} - x_k\|^2 + g(x_{k+1}) \\
&\overset{(ii)}{\leq} f(x_k) + \langle \nabla f(x_k), x_{k+1} - x_k\rangle + \frac{L}{2}\|x_{k+1} - x_k\|^2 + g(x_k) \\
&\qquad - \langle \nabla f(x_k) + e_k, x_{k+1} - x_k\rangle - L\|x_{k+1} - x_k\|^2 \\
&= h(x_k) - \langle e_k, x_{k+1} - x_k\rangle - \frac{L}{2}\|x_k - x_{k+1}\|^2 \\
&\leq h(x_k) + \tfrac{1}{2\alpha}\|e_k\|^2 + \left(\frac{\alpha}{2} - \frac{L}{2}\right)\|x_k - x_{k+1}\|^2,
\end{aligned}$$

where (i) uses Assumption (3.0.1a) and (ii) uses the (3.0.4). Choose $\alpha = L/2$ and rearrange terms to obtain the required result. □

We now proceed with the proof of Lemma 3.0.1. Let $\bar{x}_k$ be the projection of x_k onto the solution set $\mathcal{S}^*$. By the mean value theorem,

$$f(x_{k+1}) - f(\bar{x}_k) = \langle \nabla f(\xi), x_{k+1} - \bar{x}_k\rangle\,. \tag{3.0.5}$$

From Lemma 3.0.2, we have

$$\begin{aligned}
g(x_{k+1}) - g(\bar{x}_k) &\leq -\langle \nabla f(x_k) + e_k, x_{k+1} - \bar{x}_k\rangle - \frac{L}{2}\|x_{k+1} - x_k\|^2 - \frac{L}{2}\|\bar{x}_k - x_{k+1}\|^2 + \frac{L}{2}\|\bar{x}_k - x_k\|^2 \\
&\leq -\langle \nabla f(x_k) + e_k, x_{k+1} - \bar{x}_k\rangle + \frac{L}{2}\|\bar{x}_k - x_k\|^2. \qquad (3.0.6)
\end{aligned}$$

Also note that

$$\begin{aligned}
\langle \nabla f(\xi) - \nabla f(x_k), x_{k+1} - \bar{x}_k \rangle &\leq \|\nabla f(\xi) - \nabla f(x_k)\| \|x_{k+1} - \bar{x}_k\| \\
&\overset{(i)}{\leq} L\|\xi - x_k\| \|x_{k+1} - \bar{x}_k\| \\
&\leq L[\|x_{k+1} - x_k\| + \|x_k - \bar{x}_k\|] \cdot \|x_{k+1} - \bar{x}_k\| \\
&\leq [L(1+\tau)\|x_k - x_{k+1}\| + \tau\|e_k\|] \cdot [(1+\tau)\|x_k - x_{k+1}\| + \tfrac{\tau}{L}\|e_k\|] \\
&= L(1+\tau)^2\|x_k - x_{k+1}\|^2 + 2[\tau(1+\tau)]\|x_k - x_{k+1}\|\|e_k\| + \tau^2/L\|e_k\|^2 \\
&\leq [L(1+\tau)^2 + \tfrac{1}{\alpha}\tau(1+\tau)]\|x_k - x_{k+1}\|^2 + [\tau^2/L + \alpha\tau(1+\tau)]\|e_k\|^2 \\
&\leq L(1 + 3\tau + 2\tau^2)\|x_k - x_{k+1}\|^2 + \tfrac{1}{L}(2\tau^2 + \tau)\|e_k\|^2,
\end{aligned}$$

where (i) follows from (3.0.1a). In the steps which follow, we apply the relevant inequalities in Lemma 3.0.3, group terms, bound every cross term using (3.0.3), and repeat the process until we reach the final result:

$$\begin{aligned}
h(x_{k+1}) - h(\bar{x}_k) &\overset{(i)}{\leq} \langle \nabla f(\xi) - \nabla f(x_k), x_{k+1} - \bar{x}_k \rangle - \langle e_k, x_{k+1} - \bar{x}_k \rangle + \frac{L}{2}\|\bar{x}_k - x_k\|^2 \\
&\leq L(1 + 3\tau + 2\tau^2)\|x_k - x_{k+1}\|^2 + \tfrac{1}{L}(2\tau^2 + \tau)\|e_k\|^2 \\
&\quad + \frac{\alpha}{2}\|e_k\|^2 + \tfrac{1}{2\alpha}\|x_{k+1} - \bar{x}_k\|^2 + \frac{L}{2}\|\bar{x}_k - x_k\|^2 \qquad \forall \alpha > 0 \\
&\leq L(1 + 3\tau + 2\tau^2)\|x_k - x_{k+1}\|^2 + \tfrac{1}{L}(2\tau^2 + \tau)\|e_k\|^2 \\
&\quad + \frac{\alpha}{2}\|e_k\|^2 + \tfrac{1}{4\alpha}[2 + 5\tau + 3\tau^2]\|x_k - x_{k+1}\|^2 \\
&\quad + \tfrac{1}{4L^2\alpha}[3\tau^2 + \tau]\|e_k\|^2 + L\tau^2\|x_k - x_{k+1}\|^2 + \tfrac{5L}{8}(\tau^2/L^2)\|e_k\|^2 \\
&\leq \left(L(1 + 3\tau + 2\tau^2) + \tfrac{1}{4\alpha}[2 + 5\tau + 3\tau^2] + L\tau^2\right)\|x_k - x_{k+1}\|^2 + \\
&\quad \left(\tfrac{1}{L}(2\tau^2 + \tau) + \frac{\alpha}{2} + \tfrac{1}{4L^2\alpha}[3\tau^2 + \tau] + \tfrac{5L}{8}(\tau^2/L^2)\right)\|e_k\|^2 \\
&\overset{(ii)}{\leq} 10L\tau^2\|x_k - x_{k+1}\|^2 + \tfrac{1}{L}10\tau^2\|e_k\|^2 \\
&\overset{(iii)}{\leq} 40\tau^2[h(x_k) - h(x_{k+1})] + (4/L^2 + \tfrac{1}{L}10\tau^2)\|e_k\|^2 \\
&\leq 40\tau^2[h(x_k) - h(x_{k+1})] + \tfrac{1}{L}40\tau^2\|e_k\|^2.
\end{aligned}$$

In the steps above, (i) follows by add inequalities (3.0.5) and (3.0.6). Also, we make use of Lemma 3.0.3 to bound all stray terms in terms of $\|x_k - x_{k+1}\|^2$ and $\|e_k\|^2$, and Equation (3.0.3) to bound the cross-terms. In (ii) we make use of the assumption that $\tau \geq 1$ and set $\alpha = 1/L$. Finally, in (iii) we make use of Lemma 3.0.4 to transition from a bound on the distance between successive iterates x_k to differences in successive values $h(x_k)$. Rearranging terms, we get

$$(1 + 40\tau^2)h(x_{k+1}) - (1 + 40\tau^2)h(\bar{x}_k) \leq 40\tau^2(h(x_k) - h(\bar{x}_k)) + \frac{1}{L}40\tau^2\|e_k\|^2,$$

which is true if and only if the desired result holds:

$$\pi_{k+1} \leq \left(1 - \frac{1}{1 + 40\tau^2}\right)\pi_k + \frac{1}{L} \cdot \frac{40\tau^2}{1 + 40\tau^2}\|e_k\|^2.$$

Example 3.0.5 (Gradient descent with independent Gaussian noise, part I)**.** Let $e_k \sim N(0, \sigma^2 I)$. Because $\|e_k\|^2$ is a sum of n independent Gaussians, it follows a chi-squared distribution with mean $\mathbf{E}\|e_k\|^2 = n\sigma^2$. Therefore,

$$\mathbf{E}\pi_k - \rho^k\pi_0 \leq \frac{1}{\vartheta}\sum_{i=0}^{k-1}\rho^{k-1-i}\mathbf{E}\|e_i\|^2 = \frac{n\sigma^2}{\vartheta}\sum_{i=0}^{k-1}\rho^{k-1-i}. \tag{3.0.7}$$

Take the limit inferior of both sides of (3.0.7), and note that $\lim_{k\to\infty}\sum_{i=0}^{k-1}\rho^{k-1-i} = 1/(1-\rho)$. Use the values of the constants in Lemma 3.0.1 to obtain the bound

$$\mathbf{E}\liminf_{k\to\infty}\pi_k \leq \liminf_{k\to\infty}\mathbf{E}\pi_k \leq \frac{20\tau^2}{L}\,n\sigma^2,$$

where the first inequality follows from the application of Fatou's Lemma [RF10, Ch. 4]. Hence, even though $\lim_{k\to\infty}\pi_k$ may not exist, we can still provide a lower bound on the distance to optimality that is proportional to the variance of the error term.

3.1. Bounds as functions of Errors

3.1.1. Deterministic Bounds.

3.1.2. Bounds in expectation. Since π_k and e_k are both random variables, we can take expectations on both sides to obtain

$$\mathbf{E}\,\pi_k \leq \rho^k \pi_0 + \frac{1}{\vartheta}\sum_{i=0}^{k-1} \rho^{k-1-i}\,\mathbf{E}\,\|e_i\|^2,$$

3.1.3. Probabilistic bounds for gradient descent with random error. An immediate consequence of Lemma 3.0.1 is a tail bound via Markov's inequality:

$$\Pr(\pi_k - \rho^k\pi_0 \geq \epsilon) \leq \Pr\left(\frac{1}{\vartheta}\sum_{i=0}^{k-1}\rho^{k-1-i}\|e_i\|^2 \geq \epsilon\right) \leq \frac{1}{\vartheta\epsilon}\sum_{i=0}^{k-1}\rho^{k-1-i}\mathbf{E}\|e_i\|^2.x$$

This inequality is too weak, however, to say anything meaningful about the confidence in our solution after a finite number of iterations. We are instead interested in Chernoff-type bounds that are exponentially decreasing in ϵ, and in the parameters that control the size of the error.

The first bound (section 3.1.4) that we develop makes no assumption on the relation of the gradient errors between iterations, i.e., the error sequence may or may not be history dependent, and we thus refer to this as a generic error sequence. The second bound (section 3.1.5) makes the stronger assumption about the relationship of the errors between iterations.

3.1.4. Generic error sequence. Our first exponential tail bounds are defined in terms of the moment-generating function

$$\gamma_k(\theta) := \mathbf{E}\exp(\theta\|e_k\|^2)$$

of the error norms $\|e_k\|^2$. We make the convention that $\gamma_k(\theta) = +\infty$ for $\theta \notin \mathrm{dom}\gamma_k$.

Theorem 3.1.1 (Tail bound for generic errors)**.** For algorithm (1.3.3),

$$\Pr(\pi_k - \rho^k\pi_0 \geq \epsilon) \leq \inf_{\theta>0}\left\{\frac{\exp(-\theta\vartheta\epsilon/R_k)}{R_k}\sum_{i=0}^{k-1}\rho^{k-1-i}\gamma_i(\theta)\right\}. \tag{3.1.1a}$$

If $\gamma_k \equiv \gamma$ for all k (i.e., the error norms $\|e_k\|^2$ are identically distributed), then the bound simplifies to

$$\Pr(\pi_k - \rho^k\pi_0 \geq \epsilon) \leq \inf_{\theta>0}\left\{\exp(-\theta\vartheta\epsilon/R_k)\gamma(\theta)\right\}. \tag{3.1.1b}$$

PROOF. By the definition of R_k, $\left(\sum_{i=0}^{k-1} \rho^{k-1-i}\right)/R_k = 1$. Thus, for $\theta > 0$,

$$\begin{aligned}\mathbf{E}\exp\left(\theta\sum_{i=0}^{k-1}\rho^{k-1-i}\|e_i\|^2\right) &= \mathbf{E}\exp\left(\sum_{i=0}^{k-1}\frac{\rho^{k-1-i}}{R_k}\theta R_k\|e_i\|^2\right)\\ &\overset{(i)}{\leq} \mathbf{E}\sum_{i=0}^{k-1}\frac{\rho^{k-1-i}}{R_k}\exp(\theta R_k\|e_i\|^2) \overset{(ii)}{=} \frac{1}{R_k}\sum_{i=0}^{k-1}\rho^{k-1-i}\gamma_i(\theta R_k),\end{aligned}$$

where (i) follows from the convexity of $\exp(\cdot)$, and (ii) follows from the linearity of the expectation operator and the definition of γ_i. Together with Markov's inequality, the above implies that for all $\theta > 0$,

$$\begin{aligned}\Pr\left(\sum_{i=0}^{k-1}\rho^{k-1-i}\|e_i\|^2 \geq \epsilon\right) &= \Pr\left(\exp\left[\theta\sum_{i=0}^{k-1}\rho^{k-1-i}\|e_i\|^2\right] \geq \exp(\theta\epsilon)\right)\\ &\leq \exp(-\theta\epsilon)\mathbf{E}\exp\left(\theta\sum_{i=0}^{k-1}\rho^{k-1-i}\|e_i\|^2\right)\\ &\leq \frac{\exp(-\theta\epsilon)}{R_k}\sum_{i=0}^{k-1}\rho^{k-1-i}\gamma_i(\theta R_k). \qquad (3.1.2)\end{aligned}$$

This inequality, together with Lemma 3.0.1, implies that for all $\theta > 0$,

$$\Pr\left(\pi_k - \rho^k\pi_0 \geq \epsilon\right) \leq \Pr\left(\frac{1}{\vartheta}\sum_{i=0}^{k-1}\rho^{k-1-i}\|e_i\|^2 \geq \vartheta\epsilon\right) \leq \frac{\exp(-\theta\vartheta\epsilon)}{R_k}\sum_{i=0}^{k-1}\rho^{k-1-i}\gamma_i(\theta R_k),$$

where we use the elementary fact that $\Pr(X \geq \epsilon) \leq \Pr(Y \geq \epsilon)$ if $X \leq Y$ almost surely. Redefine θ as θR_k, and take the infimum of the right-hand side over $\theta > 0$, which gives the required inequality (3.1.1a). The simplified bound (3.1.1b) follows directly from the definition of R_k. □

When the errors are identically distributed, there is an intriguing connection between the tail bounds described in Theorem 3.1.1 and the convex conjugate of the cumulant-generating function of that distribution, i.e., $(\log\circ\gamma)^*$.

Theorem 3.1.2 (Tail bound for identically-distributed errors)**.** Suppose that the error norms $\|e_k\|^2$ are identically distributed. Then for algorithm (1.3.3),

$$\log\Pr(\pi_k - \rho^k\pi_0 \geq \epsilon) \leq -\left[\log\gamma(\cdot)\right]^*(\vartheta\epsilon/R_k).$$

PROOF. Take the log of both sides of (3.1.1b) to get

$$\log \Pr(\pi_k - \rho^k \pi_0 \geq \epsilon) \leq \log \inf_{\theta>0} \{\exp(-\theta\vartheta\epsilon/R_k)\, \gamma(\theta)\} = -\sup_{\theta>0} \{(\vartheta\epsilon/R_k)\theta - \log \gamma(\theta)\},$$

which we recognize as the negative of the conjugate of $\log \circ \gamma$ evaluated at $\vartheta\epsilon/R_k$. □

Note that these bounds are invariant with regard to scaling, in the sense that if the objective function f is scaled by some $\alpha > 0$, then the bounds hold for $\alpha\epsilon$.

The following example illustrates an application of this tail bound to the case in which the errors follow a simple distribution with a known moment-generating function.

Example 3.1.3 (Gradient descent with independent Gaussian noise, part II)**.** As in Example 3.0.5, let $e_k \sim N(0, \sigma^2 I)$. Then e_k is a scaled chi-squared distribution with moment-generating function

$$\gamma_k(\theta) = (1 - 2\sigma^2\theta)^{-n/2}, \quad \theta \in \left[0, \frac{1}{2\sigma^2}\right).$$

Note that

$$[\log \gamma(\cdot)]^*(\mu) = \frac{\mu - n\sigma^2}{2\sigma^2} + \frac{n}{2} \log(n\sigma^2/\mu) \text{for} \mu > n\sigma^2.$$

We can then apply Corollary 3.1.2 to this case to deduce the bound

$$\Pr(\pi_k - \rho^k \pi_0 \geq \epsilon) \leq \left(\frac{\exp(1)}{n} \cdot \frac{\vartheta\epsilon}{\sigma^2 R_k}\right)^{n/2} \exp\left(-\frac{\vartheta\epsilon}{2\sigma^2 R_k}\right) \text{for} \epsilon > \frac{n\sigma^2 R_k}{\vartheta}.$$

The bound can be further simplified by introducing an additional perturbation $\delta > 0$ that increases the base of the exponent:

$$\Pr(\pi_k - \rho^k \pi_0 \geq \epsilon) = \mathcal{O}\left[\exp\left(-\delta \frac{\vartheta\epsilon}{2\sigma^2 R_k}\right)\right] \text{ for all} \delta \in [0, 1), \tag{3.1.3}$$

which highlights the exponential decrease of the bound in terms of ϵ.

3.1.5. Unconditionally bounded error sequence. In contrast to the previous section, we now assume that there exists a deterministic bound on the conditional expectation $\mathbf{E}\left[\exp(\theta\|e_k\|^2) \mid \mathcal{F}_{k-1}\right]$. We say that this bound holds unconditionally because it holds irrespective of the history of the error sequence.

Lemma 3.1.4. Assume that $\mathbf{E}\left[\exp(\theta\|e_k\|^2) \mid \mathcal{F}_{k-1}\right]$ is finite over $[0, \sigma)$, for some $\sigma > 0$. Therefore there exists, for each k, a deterministic function $\gamma_k : \mathbf{R}_+ \to \mathbf{R}_+ \cup \{\infty\}$ such that

$$\gamma_k(0) = 1 \qquad \text{and} \qquad \mathbf{E}\left[\exp(\theta\|e_k\|^2) \mid \mathcal{F}_{k-1}\right] \leq \gamma_k(\theta).$$

(Thus, the bound is tight at $\theta = 0$.)

The existence of such a function in fact implies a bound on the moment-generating function of $\|e_k\|^2$. In particular,

$$\gamma_k(\theta) := \mathbf{E}\exp(\theta\|e_k\|^2) = \mathbf{E}\left[\mathbf{E}\left[\exp(\theta\|e_k\|^2) \mid \mathcal{F}_{k-1}\right]\right] \leq \mathbf{E}\gamma_k(\theta) = \gamma_k(\theta). \tag{3.1.4}$$

The converse, however, is not necessarily true. To see this, consider the case in which the errors $e_1, \ldots, e_{k-1}$ are independent Bernoulli-distributed random variables, and e_k is a deterministic function of all the previous errors, e.g., $\Pr(e_i = 0) = \Pr(e_i = 1) = 1/2$ for $i = 1, \ldots, k-1$, and the error on the last iteration is completely determined by the previous errors:

$$e_k = \begin{cases} 1 & \text{if } e_1 = e_2 = \cdots = e_{k-1}, \\ 0 & \text{otherwise.} \end{cases}$$

Therefore, $\Pr(e_k = 1) = (1/2)^{k-1}$ and $\Pr(e_k = 0) = 1 - (1/2)^{k-1}$, and the moment-generating function of e_k is $\gamma_k(\theta) = 1 - 2^{1-k}(1 + \exp\theta)$. Then,

$$\mathbf{E}[\exp(\theta e_k^2) \mid e_1, \ldots, e_{k-1}] = \begin{cases} \exp\theta & \text{if } e_1 = e_2 = \cdots = e_{k-1}, \\ 1 & \text{otherwise,} \end{cases}$$

whose tightest deterministic upper bound is $\gamma_k(\theta) = \exp\theta$. However, $\gamma_k(\theta) \geq \gamma_k(\theta)$ for all $\theta \geq 0$. The following result is analogous to Theorem 3.1.1.

Theorem 3.1.5 (Tail bounds for unconditionally bounded errors)**.** Suppose that Assumption 3.1.4 holds. Then for algorithm (1.3.3),

$$\Pr(\pi_k - \rho^k \pi_0 \geq \epsilon) \leq \inf_{\theta > 0} \left\{ \exp(-\theta \vartheta \epsilon) \prod_{i=0}^{k-1} \gamma_i(\theta \rho^{k-i-1}) \right\}.$$

Proof. The proof follows the same outline as many martingale-type inequalities [Azu67, CL06]. We obtain the following relationships:

$$\begin{aligned}
\mathbf{E} \exp\left[\theta \sum_{i=0}^{k-1} \rho^{k-1-i} \|e_i\|^2\right] &\overset{(i)}{=} \mathbf{E}\left[\mathbf{E}\left[\exp\left[\theta \sum_{i=0}^{k-1} \rho^{k-1-i}\|e_i\|^2\right] \middle| \mathcal{F}_k - 2\right]\right] \\
&= \mathbf{E}\left[\mathbf{E}\left[\exp\left[\theta\rho^0\|e_{k-1}\|^2 + \theta \sum_{i=0}^{k-2} \rho^{k-1-i}\|e_i\|^2\right] \middle| \mathcal{F}_{k-2}\right]\right] \\
&\overset{(ii)}{=} \mathbf{E}\left[\exp\left[\theta \sum_{i=0}^{k-2} \rho^{k-1-i}\|e_i\|^2\right] \mathbf{E}\left[\exp\left(\theta\|e_{k-1}\|^2\right) \middle| \mathcal{F}_k - 2\right]\right] \\
&\overset{(iii)}{\leq} \mathbf{E}\left[\exp\left[\theta \sum_{i=0}^{k-2} \rho^{k-i-1}\|e_i\|^2\right]\right] \gamma_{k-1}(\theta) \\
&\overset{(iv)}{\leq} \prod_{i=0}^{k-1} \gamma_i(\theta\rho^{k-i-1}),
\end{aligned}$$

where (i) follows from the law of total expectations, i.e., $\mathbf{E}_Y[\mathbf{E}[X|Y]] = \mathbf{E}[X]$; (ii) follows from the observation that the random variable $\exp(\theta \sum_{i=0}^{k-2} \rho^{k-1-i}\|e_i\|^2)$ is a deterministic function of $e_0, \ldots, e_k - 2$, and hence is measurable with respect to $\mathcal{F}_{k-1}$ and can be factored out of the expectation; (iii) uses Assumption 3.1.4; and to obtain (iv) we simply repeat the process recursively.

Thus, we now have a bound on the moment-generating function of the discounted sum of errors $\theta \sum_{i=0}^{k-1} \rho^{k-1-i}\|e_i\|^2$, and we can continue by using the same approach used to derive (3.1.2). The remainder of the proof follows that of Theorem 3.1.1, except that the sums over $i = 0, \ldots, k$ are replaced by products over that same range. □

In an application where both γ_k and γ_k are available, it is not true in general that either of the bounds obtained in Theorems 3.1.1 and 3.1.5 are tighter than the other. When only a bound γ_k that satisfies Assumption 3.1.4 is available, however, (which is the case in the sampling application we shall describe) we could leverage (3.1.4) and apply Theorem 3.1.1 to obtain a valid bound in

terms of γ_k by simply substituting it for γ_k. However, as shown below, in this case it is better to apply Theorem 3.1.5 because it yields a uniformly better bound:

$$\Pr\left(\pi_k - \rho^k \pi_k \geq \epsilon\right) \leq \inf_{\theta > 0} \left\{ \exp\left(-\theta\vartheta\epsilon + \sum_{i=0}^{k-1} \log \gamma_i\big(\theta \rho^{k-1-i}\big) \right) \right\}, \tag{3.1.5}$$

while Theorem 3.1.1 (with γ_k replaced by γ_k) gives us

$$\Pr\left(\pi_k - \rho^k \pi_0 \geq \epsilon\right) \leq \inf_{\theta > 0} \left\{ \exp\left(-\theta\vartheta\epsilon + \log\left[\frac{1}{R_k} \sum_{i=0}^{k-1} \rho^{k-1-i} \gamma_i(\theta R_k) \right] \right) \right\}, \tag{3.1.6}$$

where we rescale θ by R_k. A direct comparison of the two bounds show that they only differ by one term:

$$\log\left[\frac{1}{R_k} \sum_{i=0}^{k-1} \rho^{k-i-1} \gamma_i(\theta R_k) \right] \quad \text{vs.} \quad \sum_{i=0}^{k-1} \log \gamma_i(\theta \rho^{k-1-i}).$$

Because $R_k = \sum_{i=0}^{k} \rho^{k-i-1}$, the term in the log on the left is a convex combination of the functions γ_i. Therefore,

$$\begin{aligned}
\log\left[\frac{1}{R_k} \sum_{i=0}^{k-1} \rho^{k-i-1} \gamma_i(\theta R_k) \right] &\overset{(i)}{\geq} \sum_{i=0}^{k-1} \frac{\rho^{k-1-i}}{R_k} \log \gamma_i(\theta R_k) \\
&\overset{(ii)}{\geq} \sum_{i=0}^{k-1} \log \gamma_i(\theta R_k \, \rho^{k-1-i} / R_k) \\
&= \sum_{i=0}^{k-1} \log \gamma_i(\theta \rho^{k-1-i}),
\end{aligned}$$

where (i) is an application of Jensen's inequality and the concavity of log, and (ii) follows from the convexity of the cumulant generating function. It is then evident that (3.1.5) implies (3.1.6).

As with Corollary 3.1.2, by taking logs of both sides above, a connection can be made between our bound and the infimal convolution when $\bar{\gamma}$ is log-concave:

$$\log \Pr(\pi_k - \rho^k \pi_0 \geq \epsilon) \leq \left[\bigotimes_{i=0}^{k-1} [\log \gamma_i(\,\cdot\, \rho^{k-i-1})]^* \right] (\vartheta\epsilon / R_k),$$

where $\otimes$ denotes the infimal convolution operator.

Example 3.1.6 (Gradient descent with independent Gaussian noise, part III)**.** As in Example 3.1.3, let $e_k \sim N(0, \sigma^2 I)$. Because the errors e_k are independent, $\mathbf{E}\big[\exp(\theta\|e_k\|^2) \,|\, \mathcal{F}_{k-1}\big] = \mathbf{E}\exp(\theta\|e_k\|^2) = \gamma_k(\theta)$, which satisfies Assumption 3.1.4 with $\gamma_k(\theta) := \gamma_k(\theta)$. Apply Theorem 3.1.5 to obtain the bound

$$\Pr\left(\pi_k - \rho^k \pi_0 \geq \epsilon\right) \leq \inf_{\theta > 0} \left\{ \exp(-\theta\vartheta\epsilon) \cdot \prod_{i=0}^{k-1} (1 - 2\sigma^2\theta\rho^{k-1-i})^{-n/2} \right\}. \tag{3.1.7}$$

Apply Lemma A.1.2 to obtain

$$\Pr\left(\pi_k - \rho^k \pi_0 \geq \epsilon\right) \leq \left(\frac{\exp(1)}{n\alpha} \cdot \frac{\vartheta\epsilon}{\sigma^2}\right)^{\frac{n\alpha}{2}} \exp\left(-\frac{\vartheta\epsilon}{\sigma^2}\right) \text{for} \epsilon > \frac{n\alpha\sigma^2}{\vartheta},$$

where $\alpha = 1 - (\log\rho)^{-1}$. We simplify the bound to obtain

$$\Pr\left(\pi_k - \rho^k \pi_0 \geq \epsilon\right) = \mathcal{O}\left[\exp\left(-\delta \cdot \frac{\vartheta\epsilon}{\sigma^2}\right)\right] \text{for all} \delta \in (0,1); \tag{3.1.8}$$

cf. (3.1.3).

As an aside, we note that we can easily accommodate correlated noise, i.e., $e_k \sim N(0, \Sigma^2)$ where Σ is an $n \times n$ positive definite matrix. The error $\|e_k\|^2$ then has the distribution of a sum of chi-squared random variables that are weighted according to the eigenvalues σ_j of Σ [Imh61]:

$$\|e_k\|^2 \sim \sum_{j=1}^{n} \sigma_j^2 \chi_1^2,$$

and so the above tail bounds hold with $\sigma = \sigma_{max}$.

The bounds obtained in Examples 3.1.3 and 3.1.6 illustrate the relative strengths of Theorems 3.1.1 and 3.1.5. Comparing (3.1.3) and (3.1.8), we see that the asymptotic bounds only differ by a factor of $1/R_k$. Hence, for large ϵ, the bound in Example 3.1.3 is uniformly weaker than the bound in Example 3.1.6. Note that this holds despite the simplification (i.e., Lemma A.1.2) used to simply (3.1.7).

CHAPTER 4

Proximal Quasi Newton

In this section we will show an important family of functions g and H, the proximal operator (1.3.5) can be computed efficiently via an interior method. This approach builds on the work of [ABP13], who define the class of quadratic- support functions and outline a particular interior algorithm for their optimization. Our approach is specialized to the case where the quadratic-support function appears inside of a proximal computation. Together with the correct dualization approach (§4.3) , this yields a particularly efficient interior implementation when the data that define g and H have special structure (§4.4). The proximal quasi-Newton method serves as a showcase for how this technique can be used within a broader algorithmic context (§4.6).

4.1. Quadratic-support functions

[ABP13] introduce the notion of a quadratic-support (QS) function, which is a generalization of sublinear support functions [RW98, Ch. 8E]. Here we introduce a slightly more general definition than the version implemented by Aravkin et al. We retain the "QS" designation because the quadratic term, which is an essential feature of their definition, can also be expressed by the version we use here.

Let $\mathbf{B}_p = \{z \mid \|z\|_p \leq 1\}$ and $\mathcal{K} = \mathcal{K}_1 \times \cdots \times \mathcal{K}_k$, where each cone $\mathcal{K}_i$ is either a nonnegative orthant $\mathbf{R}^m_+$ or a second-order cone $\mathcal{Q}^m = \{(\tau, z) \in \mathbf{R} \times \mathbf{R}^{m-1} | z \in \tau \mathbf{B}_2\}$. (The size m of the cones may of course be different for each index i.) The notation $Ay \succeq_{\mathcal{K}} b$ means that $Ay - b \in \mathcal{K}$, and $\tau \mathbf{B}_p \equiv \{\tau z | z \in \mathbf{B}_p\}$. The indicator on a convex set U is denoted as

$$\delta(x \mid U) = \begin{cases} 0 & \text{if } x \in U, \\ +\infty & \text{otherwise.} \end{cases}$$

Unless otherwise specified, x is an n-vector. To help disambiguate dimensions, the p-by-p identity matrix is denoted by I_p, and the p-vector of all ones by $\mathbf{1}_p$.

We consider the class of functions $g : \mathbf{R}^n \to \mathbf{R} \cup \{+\infty\}$ that have the conjugate representation

$$g(x) = \sup_y \{y^T(Bx + d) \mid y \in \mathcal{Y}\}, \quad \text{where} \quad \mathcal{Y} = \{y \in \mathbf{R}^\ell \mid Ay \succeq_\mathcal{K} b\}. \tag{4.1.1}$$

(The term "conjugate" alludes to the implicit duality, since g may be considered as the conjugate of the indicator function to the set $\mathcal{Y}$.) We assume throughout that the feasible set $\mathcal{Y}$ is nonempty. If $\mathcal{Y}$ contains the origin, the QS function g is nonnegative for all x, and we can then consider it to be a penalty function. This is automatically true, for example, if $b \leq 0$.

The formulation (4.1.1) is close to the standard definition of a sublinear support function [Roc70, §13], which is recovered by setting $d = 0$ and $B = I$, and letting $\mathcal{Y}$ be any convex set. Unlike a standard support function, g is not positively homogeneous if $d \neq 0$. This is a feature that allows us to capture important classes of penalty functions that are not positively homogeneous, such as piecewise quadratic functions, the "deadzone" penalty, or indicators on certain constraint sets. These are examples that are not representable by the standard definition of a support function. Our definition springs from the quadratic-support function definition introduced by [ABP13], who additionally allow for an explicit quadratic term in the objective and for $\mathcal{Y}$ to be any nonempty convex set. The concrete implementation considered by [ABP13], however, is restricted to the case where $\mathcal{Y}$ is polyhedral. In contrast, we also allow $\mathcal{K}$ to contain second-order cones. Therefore, any quadratic objective terms in the [ABP13] definition can be "lifted" and turned into a linear term with a second-order-cone constraint (see Example 4.1.2). Our definition is thus no less general.

This expressive class of functions includes many penalty functions commonly used in machine learning; [ABP13] give many other examples. In addition, they show how to interpret QS functions as the negative log of an associated probability density, which makes these functions relevant to maximum a posteriori estimation. In the remainder of this section we provide some examples that illustrate various regularizing functions and constraints that can be expressed as QS functions.

Example 4.1.1 (1-norm regularizer)**.** The 1-norm has the QS representation

$$\|x\|_1 = \sup_y \{\, y^T x \mid y \in \mathbf{B}_\infty \,\},$$

where

$$A = \begin{pmatrix} I_n \\ -I_n \end{pmatrix}, \quad b = -\begin{pmatrix} \mathbf{1}_n \\ \mathbf{1}_n \end{pmatrix}, \quad d = 0, \quad B = I_n, \quad \mathcal{K} = \mathbf{R}^{2n}_+, \tag{4.1.2}$$

Example 4.1.2 (2-norm)**.** This simple example illustrates how the QS representation (4.1.1) can represent the 2-norm, which is not possible using the QS formulation described by [ABP13] where the constraints are polyhedral. With our definition, the 2-norm has the QS representation

$$\|x\|_2 = \sup_y \{y^T x \mid y \in \mathbf{B}_2\} = \sup_y \{y^T x \mid (1, y) \succeq_{\mathcal{K}} 0\},$$

where

$$A = \begin{pmatrix} 0 \\ I_n \end{pmatrix}, \quad b = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad d = 0, \quad B = I_n, \quad \mathcal{K} = \mathcal{Q}^{n+1}. \tag{4.1.3}$$

Example 4.1.3 (Polyhedral norms)**.** Any polyhedral seminorm is a support function, e.g., $\|Bx\|_1$ for some matrix B. In particular, if the set $\{y|Ay \geq b\}$ contains the origin and is centro-symmetric, then

$$\|x\| := \sup_y \{y^T Bx \mid Ay \geq b\}$$

defines a norm if B is nonsingular, and a seminorm otherwise. This is a QS function with $d := 0$ (as will be the case for any positively homogeneous QS function) and $\mathcal{Y} := \{y \mid Ay \geq b\}$.

Example 4.1.4 (Quadratic function)**.** This example justifies the term "quadratic" in our modified definition, even though there are no explicit quadratic terms. It also illustrates the roles of the terms B and d. The quadratic function can be written as

$$\tfrac{1}{2}\|x\|_2^2 = \sup_{y,t} \{y^T x - \tfrac{1}{2}t \mid \|y\|_2^2 \leq t\} = \sup_{y,t} \left\{ \begin{pmatrix} y \\ t \end{pmatrix}^T \left[\begin{pmatrix} I_n \\ 0 \end{pmatrix} x - \begin{pmatrix} 0 \\ \frac{1}{2} \end{pmatrix} \right] \middle| \, \|y\|_2^2 \leq t \right\}.$$

Use the derivation in B.1 to obtain the QS representation with parameters

$$A = \begin{pmatrix} 0 & \frac{1}{2} \\ 0 & \frac{1}{2} \\ I_n & 0 \end{pmatrix}, \quad b = \begin{pmatrix} \frac{1}{2} \\ -\frac{1}{2} \\ 0 \end{pmatrix}, \quad d = \begin{pmatrix} 0 \\ \frac{1}{2} \end{pmatrix}, \quad B = \begin{pmatrix} I_n \\ 0 \end{pmatrix}, \quad \mathcal{K} = \mathcal{Q}^{n+2}.$$

Example 4.1.5 (1-norm indicator)**.** This example is closely related to the 1-norm regularizer in Example 4.1.1, except that the QS function is used to express the constraint $\|x\|_1 \leq 1$ via an indicator function $g = \delta(\cdot \mid \mathbf{B}_1)$. Write the indicator to the 1-norm ball as the conjugate of the infinity norm, which gives

(4.1.4)

$$\delta(x \mid \mathbf{B}_1) = \sup_y \{y^T x - \|y\|_\infty\} = \sup_{y,\tau} \{y^T x - \tau \mid y \in \tau \mathbf{B}_\infty\} = \sup_{y,\tau} \{y^T x - \tau \mid -\tau \mathbf{1}_n \leq y \leq \tau \mathbf{1}_n\}.$$

This is a QS function with parameters

$$A = \begin{pmatrix} -I_n & \mathbf{1}_n \\ I_n & \mathbf{1}_n \end{pmatrix}, \quad b = 0, \quad d = \begin{pmatrix} 0 \\ -1 \end{pmatrix}, \quad B = \begin{pmatrix} I_n \\ 0 \end{pmatrix}, \quad \mathcal{K} = \mathbf{R}^{2n}.$$

Example 4.1.6 (Indicators on polyhedral cones)**.** Consider the following polyhedral cone and its polar:

$$U = \{x \mid Bx \leq 0\} \quad \text{and} \quad U^\circ = \{B^T y \mid |y \leq 0\}.$$

Use the support-function representation of a cone in terms of its polar to obtain

$$\delta(x \mid U) = \delta^*(\cdot \mid U^\circ)(x) = \sup_y \{y^T B x \mid y \leq 0\}, \tag{4.1.5}$$

which is an example of an elementary QS function. (See [RW98] for definitions of the polar of a convex set, and the convex conjugate.) A concrete example is the positive orthant, obtained by choosing $B = I_n$. An important example, used in isotonic regression [BC90], is the monotonic cone

$$U := \{x \mid x_i \geq x_j, \ \forall (i,j) \in \mathbf{E}\},$$

Here, $\mathbf{E}$ is the set of edges in a graph $\mathbf{G} = (\mathbf{V}, \mathbf{E})$ that describes the relationships between variables in $\mathbf{V}$. If we set B to be the incidence matrix for the graph, (4.1.5) then corresponds to the indicator on the monotonic cone U.

Example 4.1.7 (Distance to a cone)**.** The distance to a cone U that is a combination of polyhedral and second-order cones can be represented as a QS function:

$$\begin{aligned}\inf_{x \in U} \|x - y\|_2 &= \inf_x \{\|x - y\|_2 + \delta(x \mid U)\} \\ &= \left[\delta(\cdot \mid \mathbf{B}_2) + \delta(\cdot \mid U^\circ)\right]^*(y) = \sup\left\{y^T x \mid y \in \mathbf{B}_2 \cap U^\circ\right\}.\end{aligned}$$

The second equality follows from the relationship between infimal convolution and conjugates [Roc70, §16.4]. When U is the positive orthant, for example, $g(x) = \|\max\{0, x\}\|_2$, where the *max* operator is taken elementwise.

4.2. Building quadratic-support functions

Quadratic-support functions are closed under addition, composition with an affine map, and infimal convolution with a quadratic function. In the following, let g_i be QS functions with parameters A_i, b_i, d_i, B_i, and $\mathcal{K}_i$ (with $i = 0, 1, 2$). The rules for addition and composition are described in [ABP13], which are here summarized and amplified.

Addition rule: The function

$$h(x) := g_1(x) + g_2(x)$$

is QS with parameters

$$A = \begin{pmatrix} A_1 & \\ & A_2 \end{pmatrix}, \quad b = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}, \quad d = \begin{pmatrix} d_1 \\ d_2 \end{pmatrix}, \quad B = \begin{pmatrix} B_1 \\ B_2 \end{pmatrix}, \quad \mathcal{K} = \mathcal{K}_1 \times \mathcal{K}_2.$$

Concatenation rule: The function

$$h(x) := g_0(x^1) + \cdots + g_0(x^k),$$

where each partition $x^i \in \mathbf{R}^n$, is QS with parameters

$$(A,\, B) = I_k \otimes (A_0,\, B_0), \qquad (b,\, d) = \mathbf{1}_k \otimes (b_0,\, d_0), \qquad \mathcal{K} = \mathcal{K}_0 \times \overset{(k)}{\cdots} \times \mathcal{K}_0. \tag{4.2.1}$$

where the symbol $\otimes$ denotes the Kronecker product. The rule for concatenation follows from the rule for addition of QS functions.

Affine composition rule: The function

$$h(x) := g_0(Px - p)$$

is QS with parameters

$$A = A_0, \quad b = b_0, \quad d = d_0 - B_0 p, \quad B = B_0 P.$$

Moreau-Yosida regularization: The Moreau-Yosida envelope of g_0 is the value of the proximal operator, i.e.,

$$\mathbf{env}_{g_0}^H(z) := \inf_x \{\tfrac{1}{2}\|z - x\|_H^2 + g_0(x)\}. \tag{4.2.2}$$

It follows from [BH13, Proposition 4.10] that

$$\mathbf{env} H g_0(z) = \sup_y \left\{ y^T(B_0 x + d_0) - \frac{1}{2} y^T B_0 H^{-1} B_0^T y \mid A_0 y \preceq_{\mathcal{K}_0} b_0 \right\},$$

which is a QS function with parameters

$$A = \begin{pmatrix} 0 & \frac{1}{2} \\ 0 & \frac{1}{2} \\ R & 0 \\ A_0 & 0 \end{pmatrix}, \quad b = \begin{pmatrix} \frac{1}{2} \\ -\frac{1}{2} \\ 0 \\ b_0 \end{pmatrix}, \quad d = \begin{pmatrix} d_0 \\ -\frac{1}{2} \end{pmatrix}, \quad B = \begin{pmatrix} B_0 \\ 0 \end{pmatrix}, \quad \mathcal{K} = \mathcal{Q}^{n+2} \times \mathcal{K}_0,$$

with Cholesky factorization $R^T R = B_0 H^{-1} B_0^T$. The derivation is given in B.1, where we take $Q = B_0 H^{-1} B_0^T$.

Example 4.2.1 (Sums of norms)**.** In applications of group sparsity [YL06, JMBO10], various norms are applied to all partitions of $x = (x^1, \ldots, x^p)$, which possibly overlap. This produces the QS function

$$g(x) = \|x^1\| + \cdots + \|x^p\|, \tag{4.2.3}$$

where each norm in the sum may be different. In particular, consider the case of adding two norms $g(x) = \|x^1\|_\triangledown + \|x^2\|_\triangle$. (The extension to adding three or more norms follows trivially.) First, we introduce matrices P_i that restrict x to partition i, i.e., $x^i = P_i x$, for $i = 1, 2$. Then

$$g(x) = \|P_1 x\|_\triangledown + \|P_2 x\|_\triangle.$$

Then we apply the affine-composition and addition rules to determine the corresponding quantities that define the QS representation of g:

$$A = \begin{pmatrix} A_\triangledown & \\ & A_\triangle \end{pmatrix}, \quad b = \begin{pmatrix} b_\triangledown \\ b_\triangle \end{pmatrix}, \quad d = 0, \quad B = \begin{pmatrix} B_\triangledown P_1 \\ B_\triangle P_2 \end{pmatrix}, \quad \mathcal{K} = \mathcal{K}_\triangledown \times \mathcal{K}_\triangle,$$

where A_i, b_i, B_i, and $\mathcal{K}_i$ (with $i = \triangledown, \triangle$) are the quantities that define the QS representation of the individual norms. (Necessarily, $d = 0$ because the result is a norm and therefore positive homogeneous.) In the special case where $\|\cdot\|_\triangledown$ and $\|\cdot\|_\triangle$ are both the 2-norm, then A_i, b_i, B_i, and $\mathcal{K}_i$ are given by (4.1.3) in Example 4.1.2.

Example 4.2.2 (Graph-based 1-norm and total variation)**.** A variation of Example 4.2.1 can be used to define a variety of interesting norms, including the graph-based 1-norm regularizer used in machine learning [CMMP13], and the isotropic and anisotropic versions of total variation (TV), important in image processing [LZOX14]. Let

$$g(x) = \|Nx\|_G \quad \text{with} \quad \|z\|_G = \sum_{i=1}^{p} \|z^i\|_2,$$

where z^i is a partition of z and N is an m-by-n matrix. For anisotropic TV and the graph-based 1-norm regularizer, N is the adjacency matrix of a graph, and each partition z^i has a single unique element, so $g(x) = \|Nx\|_1$. For isotropic TV, each partition captures neighboring pairs of variables,

and N is a finite-difference matrix. The QS addition and affine-composition rules can be combined to derive the parameters of g. When $p = m$ (i.e., each z^i is a scalar), we are summing n absolute-value functions, and we use (4.1.2) and (4.2.1) to obtain

$$A = I_m \otimes \begin{pmatrix} 1 \\ -1 \end{pmatrix}, \quad b = \mathbf{1}_m \otimes \begin{pmatrix} -1 \\ -1 \end{pmatrix}, \quad d = 0, \quad B = N, \quad \mathcal{K} = \mathbf{R}_+^{2m}. \tag{4.2.4}$$

Now consider the variation where $p = m/2$, (i.e., each partition has size 2), which corresponds to summing $m/2$ two-dimensional 2-norms. Use (4.1.3) to obtain

$$A = I_{m/2} \otimes \begin{pmatrix} 0 \\ I_2 \end{pmatrix}, \quad b = \mathbf{1}_{m/2} \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad d = 0, \quad B = N, \quad \mathcal{K} = \mathcal{Q}^2 \times \overset{(m/2)}{\cdots} \times \mathcal{Q}^2.$$

4.3. The proximal operator as a conic QP

We describe in this section how the proximal map (1.3.4) can be obtained as the solution of a quadratic optimization problem (QP) over conic constraints,

$$\text{minimize}_y \quad \tfrac{1}{2} y^T Q y - c^T y \quad \text{such that} \quad Ay \succeq_{\mathcal{K}} b, \tag{4.3.1}$$

for some positive semidefinite ℓ-by-ℓ matrix Q and a convex cone $\mathcal{K} = \mathcal{K}_1 \times \cdots \times \mathcal{K}_k$. The transformation to a conic QP is not immediate because the definition of the QS function implies that the proximal map involves nested optimization. Duality, however, furnishes a means for simplifying this problem.

Proposition 4.3.1. Let g be a QS function. The following problems are dual pairs:

$$\text{minimize}_x \quad \tfrac{1}{2}\|z - x\|_H^2 + g(x), \tag{4.3.2a}$$

$$\text{minimize}_{Ay \succeq_{\mathcal{K}} b} \quad \tfrac{1}{2} y^T B H^{-1} B^T y - (d + Bz)^T y. \tag{4.3.2b}$$

If strong duality holds, the corresponding primal-dual solutions are related by

$$Hx + B^T y = Hz. \tag{4.3.3}$$

PROOF. Let

$$h_1(x) := \tfrac{1}{2}\|x - z\|_H^2 \quad \text{and} \quad h_2(x) := \sup_{y \in \mathcal{Y}} \{y^T(x + d)\}.$$

If strong duality holds, it follows from [Ber09, Prop. 5.3.8] that

$$\inf_x \{h_1(x) + h_2(Bx)\} = -\inf_y \{h_1^*(-B^T y) + h_2^*(y)\}, \tag{4.3.4}$$

where

$$h_1^*(y) = \tfrac{1}{2}\|y\|_{H^{-1}}^2 + z^T y \quad \text{and} \quad h_2^*(y) = \delta(z \mid \mathcal{Y}) - d^T y$$

are the Fenchel conjugates of h_1 and h_2, and the infima on both sides are attained. (See [Roc70, §12] for the convex calculus of Fenchel conjugates.) The right-hand side of (4.3.4) is precisely the dual problem (4.3.2b). It also follows from Fenchel duality that the pair (x, y) is optimal only if

$$x \in \operatorname{argmin}_x \{h_1(x) + y^T B x\}.$$

Differentiate this objective to obtain (4.3.3). □

Strong duality holds when $B \cdot \operatorname{ri} \operatorname{dom}(h_1) \cap \operatorname{ri} \operatorname{dom}(h_2) \neq \emptyset$. This holds, for example, when the interior of the domain of g is nonempty, since

$$\operatorname{int} \operatorname{dom}(g) \neq \emptyset \iff \operatorname{im} B \cap \operatorname{int} \operatorname{dom}(h_2) \neq \emptyset \Rightarrow B \cdot \operatorname{ri} \operatorname{dom}(h_1) \cap \operatorname{ri} \operatorname{dom}(h_2) \neq \emptyset.$$

In all of the examples in this paper, this condition holds true.

4.4. Primal-dual methods for conic QP

Proposition 4.3.1 provides a means of evaluating the proximal map of QS functions via conic quadratic optimization. There are many algorithms for solving convex conic QPs, but primal-dual methods offer a particularly efficient approach that can leverage the special structure that defines the class of QS functions. A detailed discussion of the implementation of primal-dual methods for conic optimization is given by [Van10]. Here we summarize the main aspects that pertain to implementing these methods efficiently in our context.

The standard development of primal-dual methods for (4.3.1) is based on perturbing the optimality conditions, which can be stated as follows. The solution y, together with slack and dual

vectors s and v, must satisfy

$$Qy - A^T v = c, \quad v \succeq_{\mathcal{K}} 0, \quad Sv = 0,$$

where the matrix S is block diagonal, and each m_i-by-m_i block S_i is either a diagonal or arrow matrix depending on the type of cone, i.e.,

$$S_i = \begin{cases} \mathbf{diag}(s_i) & \text{if } \mathcal{K}_i = \mathbf{R}_+^{m_i}, \\ \mathbf{arrow}(s_i) & \text{if } \mathcal{K}_i = \mathcal{Q}^{m_i}, \end{cases} \qquad \mathbf{arrow}(u) := \begin{pmatrix} u_0 & \bar{u}^T \\ \bar{u} & u_0 I \end{pmatrix} \quad \text{for} \quad u = (u_0, \bar{u}).$$

See [Van10] for further details. Now replace the complementarity condition $Sv = 0$ with its perturbation $Sv = \mu e$, where μ is a positive parameter and $e = (e^1, \ldots, e^k)$, with each partition defined by

$$e^i = \begin{cases} (1, 1, \ldots, 1) & \text{if } \mathcal{K}_i = \mathbf{R}_+^{m_i}, \\ (1, 0, \ldots, 0) & \text{if } \mathcal{K}_i = Q^{m_i}. \end{cases}$$

A Newton-like method is applied to the perturbed optimality conditions, which we phrase as the root of the function

$$R_\mu : \begin{pmatrix} y \\ v \\ s \end{pmatrix} \mapsto \begin{pmatrix} r_d \\ r_p \\ r_\mu \end{pmatrix} := \begin{pmatrix} Qy - A^T v - c \\ Ay - s - b \\ Sv - \mu e \end{pmatrix}. \tag{4.4.1}$$

Each iteration of the method proceeds by systematically choosing the perturbation parameter μ (ensuring it decreases), and obtaining each search direction as the solution of the Newton system

$$\begin{pmatrix} Q & -A^T & 0 \\ A & 0 & -I \\ 0 & S & V \end{pmatrix} \begin{pmatrix} \Delta y \\ \Delta v \\ \Delta s \end{pmatrix} = - \begin{pmatrix} r_d \\ r_p \\ r_\mu \end{pmatrix}, \qquad \begin{pmatrix} y^+ \\ v^+ \\ s^+ \end{pmatrix} = \begin{pmatrix} y \\ v \\ s \end{pmatrix} + \alpha \begin{pmatrix} \Delta y \\ \Delta v \\ \Delta s \end{pmatrix}. \tag{4.4.2}$$

The steplength α is chosen to ensure that (v^+, s^+) remain in the strict interior of the cone.

One approach to solving for the Newton direction is to apply block Gaussian elimination to (4.4.2), and obtain the search direction via the following systems:

$$(Q + A^T S^{-1} V A)\Delta x = r_d + A^T S^{-1}(V r_p + r_\mu), \tag{4.4.3a}$$

$$\Delta v = S^{-1}(V r_p + r_\mu - V A \Delta x), \tag{4.4.3b}$$

$$\Delta s = V^{-1}\left(r_\mu - S\Delta v\right). \tag{4.4.3c}$$

In practice, the matrices S and V are rescaled at each iteration in order to yield search directions with favorable properties. In particular, the Nesterov-Todd rescaling redefines S and V so that $SV^{-1} = \mathbf{block}(u)$ for some vector u, where

$$\mathbf{block}(u)_i = \begin{cases} \mathbf{diag}(u_i) & \text{if } \mathcal{K}_i = \Re_+^{m_i}, \\ (2u_i u_i^T - [u_i^T J u_i]J)^2 & \text{if } \mathcal{K}_i = \mathcal{Q}^{m_i}, \end{cases} \qquad J = \begin{pmatrix} 1 & 0 \\ 0 & -I_{(m_i-1)} \end{pmatrix}. \tag{4.4.4}$$

The cost of the overall approach is therefore determined by the cost of solving, at each iteration, linear systems with the matrix

$$\mathcal{L}(u) := Q + A^T \mathbf{block}(u)^{-1} A, \tag{4.4.5}$$

which now defines the system (4.4.3a).

4.5. Evaluating the proximal operator

We now describe how to use Proposition 4.3.1 to transform a proximal operator (1.3.5) into a conic QP that can be solved by the interior algorithm described in §4.4. In particular, to evaluate $\mathbf{prox}Hg(x)$ we solve the conic QP (4.3.1) with the definitions

$$Q := BH^{-1}B^T, \qquad c := d + Bx; \tag{4.5.1}$$

the other quantities A, b, and the cone $\mathcal{K}$, appear verbatim. Algorithm 3 summarizes the procedure. As we note in §4.4, the main cost of this procedure is incurred in Step 1, which requires repeatedly solving linear systems that involve the linear operator (4.4.5). Together with (4.5.1), these matrices

Algorithm 3: Evaluating $\mathbf{prox}_H^g(x)$

INPUT : x, H, and QS function g as defined by parameters A, b, d, B, $\mathcal{K}$

OUTPUT : $\mathbf{prox}_H^g(x)$

Step 1: Apply interior method to QP (4.3.2b) to obtain y^*.

Step 2: Return $H^{-1}(c - B^T y^*)$.

Algorithm 4: Inverse via the Sherman-Woodbury identity

function `SWinv`(D, U, M)

1 $D_1 \leftarrow D^{-1}$

2 $U_1 \leftarrow D_1 U$

3 $M_1 \leftarrow (M^{-1} + U^T U_1)^{-1}$

4 **return** D_1, U_1, M_1

end

have the form

$$\mathcal{L}(u) = BH^{-1}B^T + A^T \mathbf{block}(u)^{-1} A. \tag{4.5.2}$$

Below we offer a tour of several examples, ordered by level of simplicity, to illustrate the details involved in the application of our technique. The Sherman-Woodbury (SW) identity

$$(D + UMU^T)^{-1} = D^{-1} - D^{-1}U(M^{-1} + U^T D^{-1} U)^{-1} U^T D^{-1},$$

valid when $M^{-1} + U^T D^{-1} U$ is nonsingular, proves useful for taking advantage of certain structured matrices that arise when solving (4.5.2). Some caution is needed, however, because it is known that the SW identity can be numerically unstable [Yip86].

For our purposes, it is useful to think of the SW formula as a routine that takes the elements (D, U, M) that define a linear operator $D + UMU^T$, and returns the elements (D_1, U_1, M_1) that define the inverse operator $D_1 + U_1 M_1 U_1^T = (D + UMU^T)^{-1}$. We assume that D and M are nonsingular. Algorithm 4 summarizes the operations needed to compute the elements of the inverse operator.

Typically, D is a structured operator that admits a fast algorithm for solving linear systems with any right-hand side, and U and M are stored explicitly as dense matrices. Step 1 computes a new operator D_1 that simply interchanges the multiplication and inversion operations of D. Step 2

applies the operator D_1 to every column of U (typically a tall matrix with few columns). Step 3 requires inverting a small matrix.

Example 4.5.1 (1-norm regularizer; cf. Example 4.1.1)**.** Example 4.1.1 gives the QS representation for $g(x) = \|x\|_1$, and the required expressions for A, B, and $\mathcal{K}$. Because $\mathcal{K}$ is the nonnegative orthant, $\mathbf{block}(u) = \mathbf{diag}(u)$; cf. (4.4.4). With the definitions of A and B, the linear operator $\mathcal{L}$ in (4.5.2) simplifies to

$$\mathcal{L}(u) = H^{-1} + A^T\mathbf{diag}(u)A = H^{-1} + \Sigma,$$

where Σ is a positive-definite diagonal matrix that depends on u. If it happens that the preconditioner H has a special structure such that $H + \Sigma^{-1}$ is easily invertible, it may be convenient to apply the SW identity to obtain equivalent formulas for the inverse

$$\mathcal{L}(u)^{-1} = (H^{-1} + \Sigma)^{-1} = H - H(H + \Sigma^{-1})^{-1}H.$$

Banded, chordal, and diagonal-plus-low-rank matrices are examples of specially structured matrices that make one of these formulas for $\mathcal{L}^{-1}$ efficient. They yield the efficiency because subtracting the diagonal matrix Σ preserves the structure of either H or H^{-1}.

In the important special case where $H = \mathbf{diag}(h)$ is diagonal, each component i of the proximal operator for the 1-norm can be obtained directly via the formula

$$[\mathbf{prox}_g^H(x)]_i = \mathbf{sign}(x_i) \cdot \max\{|x_i - 1|/h_{ii}, 0\},$$

where h_{ii} are the diagonal elements of H. This corresponds to the well-known soft-thresholding operator. No simple formula exists, however, for more general matrices.

Example 4.5.2 (Graph-based 1-norm)**.** Consider the graph-based 1-norm function from Example 4.2.2 induced by a graph $\mathcal{G}$ with adjacency matrix N. Substitute the definitions of A and B

from (4.2.4) into the formula for $\mathcal{L}$ and simplify to obtain

$$\mathcal{L}(u) = NH^{-1}N^T + A^T\mathbf{diag}(u)A = NH^{-1}N^T + \Sigma,$$

where $\Sigma := A^T\mathbf{diag}(u)A$ is a positive-definite diagonal matrix. (As with Example 4.5.1, $\mathcal{K}$ is the positive orthant, and thus $\mathbf{block}(u) = \mathbf{diag}(u)$.) Linear systems of the form $\mathcal{L}(u)p = q$ then can be solved with the following sequence of operations outlined in Algorithm 5, in which we assume that $H = \Lambda + UMU^T$, where Λ is diagonal.

Algorithm 5: Solving the system $\mathcal{L}(u)p = q$ for the graph-based 1-norm.

1 $(\Lambda_1, U_1, M_1) \leftarrow$ `SWinv`(Λ, U, M)	[$H^{-1} \equiv \Lambda_1 + U_1M_1U_1^T$]
2 $\Sigma_1 \leftarrow N\Lambda_1N^T + \Sigma$	
3 $(\Sigma_2, U_2, M_2) \leftarrow$ `SWinv`(Σ_1, NU_1, M_1)	[$\mathcal{L}(u)^{-1} \equiv \Sigma_2 + U_2M_2U_2^T$]
4 $p \leftarrow \Sigma_2 q + U_2M_2U_2^Tq$	[`solve` $\mathcal{L}(u)p = q$]
5 **return** p	

Observe from the definition of H and the definition of Σ_1 in Step 2 that

$$\mathcal{L}(u) = \Sigma_1 + NU_1M_1U_1^TN^T,$$

and then Step 3 computes the quantities that define the inverse of $\mathcal{L}$. The bulk of the work in the above algorithm happens in Step 3, where $\Sigma_2 \equiv \Sigma_1^{-1}$ is applied to each column of NU_1 (see Step 2 of the `SWinv` function), and in Step 4, where Σ_2 is applied to q. Below we give two special cases where it is possible to take advantage of the structure of N and H in order to apply Σ_2 efficiently to a vector.

1-dimensional total variation.: Suppose that the graph $\mathcal{G}$ is a path. Then the $(n-1) \times n$ adjacecy matrix is given by

$$N = \begin{pmatrix} -1 & 1 & & \\ & \ddots & \ddots & \\ & & -1 & 1 \end{pmatrix}.$$

The matrix $\Sigma_1 := N\Lambda^{-1}N^T + \Sigma$ (see Step 2 of the above algorithm) is tridiagonal, and hence equations of the form $\Sigma_1 q = p$ can be solved efficiently using standard techniques, e.g., [GL89, Algorithm 4.3.6].

Chordal graphs.: If the graph $\mathcal{G}$ is chordal, than the matrix $N^T D N$ is also chordal when D is diagonal. This implies that it can be factored in time linear with the number of edges of the graph [ADV10]. We can use this fact to apply $\Sigma_2 \equiv \Sigma_1^{-1}$ efficiently, as follows: let $(\Sigma_3, U_3, M_3) = \texttt{SWinv}(\Sigma, N, \Lambda_1)$, which implies

$$\Sigma_2 := \Sigma_3 + U_3 M_3 U_3^T, \quad \text{where} \quad \Sigma_3 := \Sigma^{-1}, \quad M_3 := (N^T\Sigma^{-1}N + \Lambda_1)^{-1}.$$

Because $N^T\Sigma^{-1}N$ is chordal, so is M_3, and any methods efficient for solving with chordal matrices can be used when applying Σ_2.

Example 4.5.3 (1-norm constraint; cf. Example 4.1.5)**.** Example 4.1.5 gives the QS representation for the indicator function on the 1-norm ball. Because the constraints on y in (4.1.4) involve only bound constraints, $\mathbf{diag}(u) = \mathbf{diag}(u)$. With the definitions of A and B from Example 4.1.5, the linear operator $\mathcal{L}$ has the form

$$\mathcal{L}(u) = \begin{pmatrix} 0 \\ I_n \end{pmatrix} H^{-1} \begin{pmatrix} 0 & I_n \end{pmatrix} + \begin{pmatrix} \mathbf{1}_n^T & \mathbf{1}_n^T \\ -I_n & I_n \end{pmatrix} \begin{pmatrix} \mathbf{diag}(u_1) & \\ & \mathbf{diag}(u_2) \end{pmatrix} \begin{pmatrix} \mathbf{1}_n & -I_n \\ \mathbf{1}_n & I_n \end{pmatrix},$$

where $u = (u_1, u_2)$. Thus, $\mathcal{L}$ simplifies to

$$\mathcal{L}(u) = \begin{pmatrix} \mathbf{1}_n^T u & (u^-)^T \\ u^- & H^{-1} + \Sigma \end{pmatrix} \text{where} \Sigma := \mathbf{diag}(u^+), \qquad \begin{aligned} u^+ &:= \quad u_1 + u_2, \\ u^- &:= -u_1 + u_2. \end{aligned}$$

Systems that involve $\mathcal{L}$ can be solved by pivoting on the block $(H^{-1} + \Sigma)$. The cases where this approach is efficient are exactly those that are efficient in the case of Example 4.5.1.

Example 4.5.4 (2-norm; cf. Example 4.1.2)**.** Example 4.1.2 gives the QS representation for the 2-norm function. Because $\mathcal{K} = Q^n$, then $\mathbf{block}(u) = (2uu^T - [u^T J u]J)^2$, where $u = (u_0, \bar{u})$ and J is specified in (4.4.4). With the expressions for A and B from Example 4.1.2, the linear operator $\mathcal{L}$

reduces to

$$\mathcal{L}(u) = H^{-1} + \alpha I_n + vv^T, \text{with} \alpha = (u^T J u)^2,\ v = \sqrt{8u_0} \cdot \bar{u}. \tag{4.5.3}$$

This amounts to a perturbation of H^{-1} by a multiple of the identity, followed by a rank-1 update. Therefore, systems that involve $\mathcal{L}$ can be solved at the cost of solving systems with $H + \alpha I_n$ (for some scalar α).

Of course, the proximal map of the 2-norm is easily computed by other means; our purpose here is to use this as a building block for more useful penalties, such as Example 4.2.1, which involves the sum-of-norms function shown in (4.2.3). Suppose that the p partitions do not overlap, and have size n_i for $i = 1, \ldots, p$. The operator $\mathcal{L}$ in (4.5.3) generalizes to

$$\mathcal{L}(u) = H^{-1} + \underbrace{\begin{pmatrix} \alpha_1 I_{n_1} + v^1 (v^1)^T & & \\ & \ddots & \\ & & \alpha_p I_{n_p} + v^p (v^p)^T \end{pmatrix}}_{W}, \quad \begin{aligned} u^i &= (u_0^i, \bar{u}^i) \\ \alpha_i &= (u^{iT} J u^i)^2 \\ v^i &= \sqrt{8u_0^i} \cdot \bar{u}^i, \end{aligned}$$

where each vector u_i has size $n_i + 1$.

When p is large, we can treat each diagonal block of W as an individual (small) diagonal-plus-rank-1 matrix. If H^{-1} is diagonal-plus-low-rank, for example, the diagonal part of H^{-1} can be subsumed into W. In that case, each diagonal block in W remains diagonal-plus-rank-1, which can be inverted in parallel by handling each block individually. Subsequently, the inverse of $\mathcal{L}$ can be obtained by a second correction.

Another approach, when p is small, is to consider W as a diagonal-plus-rank-p matrix:

$$W = \begin{pmatrix} \alpha_1 I_{n_1} & & \\ & \ddots & \\ & & \alpha_p I_{n_p} \end{pmatrix} + \begin{pmatrix} v^1 & & \\ & \ddots & \\ & & v^p \end{pmatrix} \begin{pmatrix} v^1 & & \\ & \ddots & \\ & & v^p \end{pmatrix}^T.$$

This representation is convenient: systems involving $\mathcal{L}$ can be solved efficiently in a manner identical to that of Example 4.5.1 because W is a diagonal-plus-low-rank matrix.

Example 4.5.5 (separable QS functions)**.** Suppose that g is separable, i.e.,

$$g(x) = \gamma(x_1) + \cdots + \gamma(x_n),$$

where $\gamma : \mathbf{R} \to \mathbf{R}$ is a QS function with parameters $(A_\gamma, b_\gamma, B_\gamma, d_\gamma, \mathbf{R}_+^{np})$, and p is an integer parameter that depends on γ. The parameters A and B for g follow from the concatenation rule (4.2.1), and $A = (I_n \otimes A_\gamma)$ and $B = (I_n \otimes B_\gamma)$. Thus, the linear operator $\mathcal{L}$ is given by

$$\mathcal{L}(u) = (I_n \otimes B_\gamma)H^{-1}(I_n \otimes B_\gamma)^T + (I_n \otimes A_\gamma)^T \mathbf{diag}(u)(I_n \otimes A_\gamma).$$

Apply the SW identity to obtain

$$\mathcal{L}(u)^{-1} = \Lambda^{-1} - \Lambda^{-1}(I_n \otimes B_\gamma)(H + \Sigma)^{-1}(I_n \otimes B_\gamma)^T\Lambda^{-1},$$

where $\Lambda = \mathbf{diag}(\Lambda_1, \ldots, \Lambda_n)$,

$$\Lambda_i = A_\gamma^T \mathbf{diag}(u^i) A_\gamma, \quad \text{and} \quad \Sigma = \mathbf{diag}(B_\gamma^T \Lambda_1^{-1} B_\gamma, \ldots, B_\gamma^T \Lambda_n^{-1} B_\gamma).$$

Because the function γ takes a scalar input, B_γ is a vector. Hence Σ is a diagonal matrix. Note too that Λ is a block diagonal matrix with n blocks each of size p. We can then solve the system $\mathcal{L}(u)p = q$ with the following steps:

1 $q_1 \leftarrow (I_n \otimes B_\gamma)^T \Lambda^{-1} q$

2 $q_2 \leftarrow (H + \Sigma)^{-1} q_1$

3 $q_3 \leftarrow \Lambda^{-1} q_2 - \Lambda^{-1}(I_n \otimes B_\gamma) q_2$

The cost of solving systems with the operator $\mathcal{L}$ is dominated by solves with the block diagonal matrix Λ (Steps 1 and 3) and $H + \Sigma$ (Step 2). The cost of the latter linear solve is explored in Example 4.5.1.

4.6. A proximal quasi-Newton method

We now turn to the proximal-gradient method discussed in §1.1. Our primary goal is to demonstrate the feasibility of the interior approach for evaluating proximal operators of QS functions.

A secondary goal is to illustrate how this technique leads to an efficient extension of the quasi-Newton method for nonsmooth problems of practical interest.

We follow [ST16] and implement a limited-memory BFGS (L-BFGS) variant of the proximal-gradient method that has no linesearch and that approximately evaluates the proximal operator. Scheinberg and Tang establish a sublinear rate of convergence for this method when the Hessian approximations are suitably modified by adding a scaled identity matrix, and when the scaled proximal maps are evaluated with increasing accuracy. In their proposal, the accuracy of the proximal evaluation is based on bounding the value of the approximation to (4.2.2). We depart from this criterion, however, and instead use the residual (4.4.1) obtained by the interior solver to determine the required accuracy. In particular, we require that the optimality criterion of the interior algorithm used to evaluate the operator is a small multiplicative constant κ of the current optimality of the outer proximal-gradient iterate, i.e.,

$$\|R_\mu(y, v, s)\| \leq \kappa \|x_k - \mathbf{prox}_g(x_k - \nabla f(x_k))\|.$$

This heuristic is reminiscent of the accuracy required of the linear solves used by an inexact Newton method for root finding [DES82]. Note that the proximal map $\mathbf{prox}_g \equiv \mathbf{prox}_g^I$ used above is unscaled, which in many cases can be easily computed when g is seperable.

4.6.1. Limited-memory BFGS updates. Here we give a brief outline the L-BFGS method for obtaining Hessian approximations of a smooth function f. We follow the notation of [NW99, §6.1], who use H_k to denote the current approximation to the *inverse* of the Hessian of f. Let x_k and x_{k-1} be two consecutive iterates, and define the vectors

$$s_k = x_{k+1} - x_k, \quad \text{and} \quad y_k = \nabla f(x_{k+1}) - \nabla f(x_k).$$

A "full memory" BFGS method updates the approximation H_k via the recursion

$$H_0 = \sigma I, \qquad H_{k+1} = H_k - \frac{H_k s_k s_k^T H_k}{s_k^T H_k s_k} + \frac{y_k y_k^T}{y_k^T s_k},$$

for some positive parameter σ that defines the initial approximation. The limited-memory variant of the BFGS update (L-BFGS) maintains the most recent m pairs (s_k, y_k), discarding older vectors.

In all cases, $m \ll n$, e.g., $m = 10$. The globalization strategy advocated by [ST16] may add a small multiple of the identity to H_k. This modification takes the place of a potentially expensive linesearch, and the correction is increased at each iteration if a certain condition for decrease is not satisfied.

Each interior iteration for evaluating the proximal operator depends on solving linear systems with $\mathcal{L}$ in (4.5.2). In all of the experiments presented below, each interior iteration has a cost that is linear in the number of variables n.

4.7. Numerical experiments

We have implemented the proximal quasi-Newton method as a Julia package [BEKS14], called `QSip` designed for problems of the form (3.1.1a), where f is smooth and g is a QS function. The code is available at the URL

`https://github.com/MPF-Optimization-Laboratory/QSip.jl`

A primal-dual interior method, based on ideas from the CVXOPT software package [ADV10], is used for Algorithm 3. We consider below several examples. The first three examples apply the `QSip` solver to minimize benchmark least-squares problems with different nonsmooth regularizers that are QS representable; the last example applies the solver to a sparse logistic-regression problem on a standard data set.

4.7.1. Timing the proximal operator. The examples that we explored in §4.5 have a favorable structure that allows each interior iteration for evaluating the proximal map $\mathbf{prox}_g^H(x)$ to scale linearly with problem size. In this section we verify this behavior empirically for problems with the structure

$$H = I + UU^T, \qquad g(x) = \|x\|_1, \qquad U \in \mathbf{R}^{n\times k} \tag{4.7.1}$$

for different values of k and n. This choice of diagonal-plus-low-rank matrices is designed to mimic the structure of matrices that appear in L-BFGS. Here U and x are chosen with random normal entries. As described in Example 4.1.1, the system $\mathcal{L}(u)$ is inverted in linear time using the SW identity.

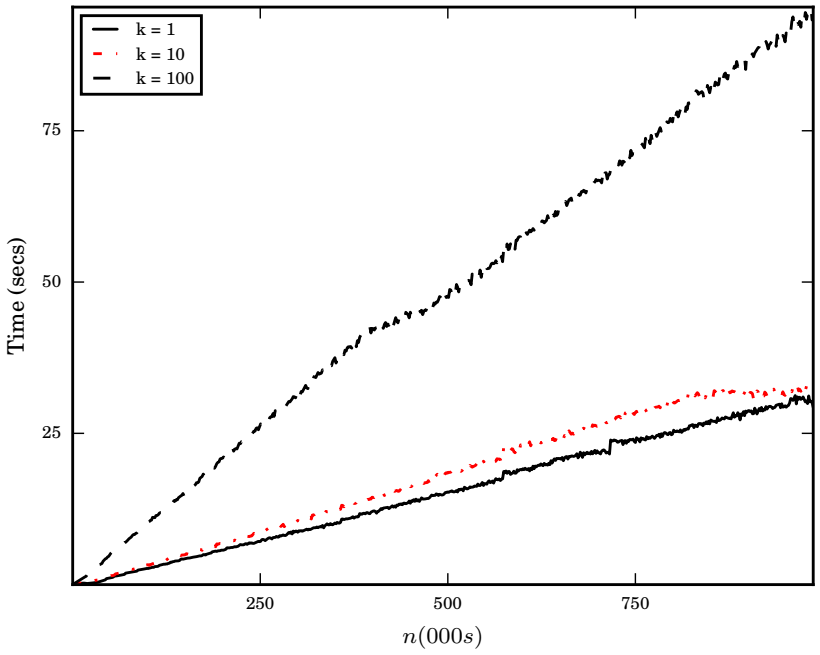

FIGURE 4.7.1. Time taken to compute $\mathbf{prox}_g^H(x)$ versus n, for $k = 1, 10, 100$; see (4.7.1).

We evaluate the proximal map on 100 random instances for each combination of k and n, and plot in Figure 1 the average time needed to reach an accuracy of 10^{-7}, as measured by the optimality conditions in the interior algorithm. Because in practice the number of iterations of the interior method is almost independent of the size of the problem, the time taken to compute the proximal map is a predictable, linear function of the size of the problem.

4.7.2. Synthetic least-square problems. The next set of examples all involve the least-squares objective

$$f(x) = \tfrac{1}{2}\|Ax - b\|_2^2. \tag{4.7.2}$$

Two different procedures are used to construct matrices A, as described in the following sections. In all cases, we follow the testing approach described by [Lor13] for constructing a test problem with a known solution: fix a vector $x^\star$ and choose $b = Ax^\star - A^{-T}v$, where $v \in \partial g(x^\star)$. Note that

$$\partial(f+g)(x^\star) = A^T(Ax^\star - [Ax^\star - A^{-T}v]) + \partial g(x^\star) = \partial g(x^\star) - v.$$

Because $v \in \partial g(x^\star)$, the above implies that $0 \in \partial(f+g)(x^\star)$, and hence $x^\star$ minimizes the objective $f+g$. In the next three sections, we apply `QSip` in turn to problems with g equal to the 1-norm, the group LASSO (i.e., sum of 2-norm functions), and total variation.

4.7.2.1. *One-norm regularization.* In this experiment we choose $g = \|\cdot\|_1$, which gives the 1-norm regularized least-squares problem, often used in applications of sparse optimization. Following the details in Example 4.1.1, the system $\mathcal{L}(u)$ is a diagonal-plus-low-rank matrix, which we invert using the SW identity.

The matrix A in (4.7.2) is a 2000-by-2000 lower triangular matrix with all nonzero entries equal to 1. The bandwidth p of A is adjustable, and determines its coherence

$$\text{coherence}(A) = \max_{i \neq j} \frac{a_i^T a_j}{\|a_i\|\|a_j\|} = \sqrt{\frac{p-1}{p}},$$

where a_i is the ith column. As observed by [Lor13], the difficulty of 1-norm regularized least-squares problems are strongly influenced by the coherence. Our experiments use matrices A with bandwidth $p = 500, 1000, 2000$.

Figure 4.7.2 shows the results of applying the `QSip` solver with a memories $k = 1, 10$, labeled "QSIP mem = k". We also consider comparisons against two competitive proximal-based methods. The first is a proximal-gradient algorithm that uses the Barzilai-Borwein steplength [BB88, WNF09]. This is our own implementation of the method, and is labeled "Barzilai-Borwein" in the figures. The second is the proximal quasi-Newton method implemented by [BF12], which is based on a symmetric-rank-1 Hessian approximation; this code is labeled "PG-SR1". The `QSip` solver with memory of 10 outperforms the other solvers. The quasi-Newton approximation benefits problems with high coherence (p large) more than problems with low coherence (p small). In all cases, the experiments reveal that the additional cost involved in evaluating a proximal operator (via an interior method) is balanced by the overall cost of the algorithm, both in terms of iterations (i.e., matrix-vector products with A) and time.

4.7.2.2. *The effect of conditioning.* It is well known that the proximal-gradient method converges slowly for ill conditioned problems. The proximal L-BFGS method may help to improve convergence in such situations. We investigate the observed convergence rate of the proximal L-BFGS approach on a family of least-squares problems with 1-norm regularization with varying degrees of ill conditioning.

For these experiments, we take A in (4.7.2) as the 2000-by-2000 matrix

$$A = \alpha_L \begin{pmatrix} T & 0 \\ 0 & 0 \end{pmatrix} + \alpha_\mu I,$$

where T is a 1000-by-1000 tridiagonal matrix with constant diagonal entries equal to 2, and constant sub- and super-diagonal entries equal to -1. The parameter α_L/α_μ controls the conditioning of A, and hence the conditioning of the Hessian A^TA of f.

We run L-BFGS with 4 different memories ("mem"): 0 (i.e., proximal gradient with a Barzilai-Borwein steplength), 1, 10, and 100. We terminate the algorithm either when the error drops beneath 10^{-8}, or the method reaches 10^3 iterations. Our method of measuring the observed convergence (OC) computes the line of best fit to the log of optimality versus k, which results in the quantity

$$\text{Observed Convergence} := \frac{\sum_{k=0}^{N} k \cdot \log \|x_k - x_*\|}{\sum_{k=0}^{N} \log \|x_k - x_*\|},$$

where N is the total number of iterations.

The plot in Figure 4.7.3 shows the ratio of the OC for L-BFGS relative to the observed convergence of proximal gradient (PG). This quantity can be interpreted the amount of work that a single quasi-Newton step performs relative to the number of PG iterations. The plot reveals that the quasi-Newton method is faster at all condition numbers, but is especially effective for problems with moderate conditioning. Also, using a higher quasi-Newton memory almost always lowers the number of iterations. This benefit is most pronounced when the problem conditioning is poor.

Together with §4.7.1, this section gives a broad picture of the trade-off between the proximal quasi-Newton and proximal gradient methods. The time required for each proximal gradient iteration is dominated by the cost of the gradient computation because the evaluation of the unscaled proximal operator is often trivial. On the other hand, the proximal quasi-Newton iteration additionally requires evaluating the scaled proximal operator. Therefore, the proximal quasi-Newton method is most appropriate when this cost is small relative to the gradient evaluation.

4.7.2.3. *Group LASSO.* Our second experiment is based on the sum-of-norms regularizer described in Examples 4.2.1 and 4.5.4. In this experiment, the n-vector (with $n = 2000$) is partitioned into $p = 5$ disjoint blocks of equal size. The matrix A is fully lower triangular.

Figure 4.7.4 clearly shows that the `QSip` solver outperforms the PG method with the Barzilai-Borwein step size. Although we required `QSip` to exit with a solution estimate accurate within 6 digits (i.e., $\log \|x - x^*\| \leq 10^{-6}$), the interior solver failed to achieve the requested accuracy because of numerical instability with the SW formula used for solving the Newton system. This raises the question of how to use efficient alternatives to the SW update that are numerically stable and can still leverage the structure of the problem.

4.7.2.4. *1-dimensional total variation.* Our third experiment sets

$$g(x) = \sum_{i=1}^{n-1} |x_{i+1} - x_i|,$$

which is the anisotropic total-variation regularizer described in Examples 4.2.2 and 4.5.2. The matrix A is fully lower triangular. Figure 4.7.5 compares the convergence behavior of `QSip` with the Barzilai-Borwein proximal solver. The Python package `prox-tv` [BS11, BS14] was used for the evaluation of the (unscaled) proximal operator, needed by the Barzilai-Borwein solver. The `QSip` solver, with memories of 1 and 10, outperformed the Barzilai-Borwein solver.

4.7.3. Sparse logistic regression. This next experiment tests `QSip` on the sparse logistic-regression problem problem

$$\underset{x}{\text{minimize}} \quad \frac{1}{N} \sum_{i=1}^{N} \log(1 + \exp[a_i^T x]) + \lambda \|x\|_1,$$

where N is the number of observations. The *Gisette* [GGBHD04] and *Epsilon* [Pas16] datasets, standard benchmarks from the UCI Machine Learning Repository [Lic13], are used for the feature vectors a_i. *Gisette* has $5K$ parameters and $13.5K$ observations; *Epsilon* has $2K$ parameters with $400K$ observations. These datasets were chosen for their large size and modest number of parameters. In all of these experiments, $\lambda = 0.01$.

Figure 4.7.6 compares `QSip` to the Barzilai-Borwein solver, and to newGLMNet [KKL+07], a state-of-the-art solver for sparse logistic regression. (Other possible comparisons include the implementation of [ST16], which we do not include because of difficulty compiling that code.) Because we do not know a priori the solution for this problem, the vertical axis measures the log of the optimality residual $\|x_k - \mathbf{prox}_g(x_k - \nabla f(x_k))\|_\infty$ of the current iterate. (The norm of

this residual necessarily vanishes at the solution.) On the *Gisette* dataset, Barzilai-Borwein and newGLMNNet are significantly faster than the proximal quasi-Newton implementation. On the *Epsilon* dataset, however, the quasi-Newton is faster at all levels of accuracy.

4.8. Conclusion

Much of our discussion revolves around techniques for solving the Newton systems (4.4.2) that arise in the implementation of an interior method for solving QPs. The Sherman-Woodbury formula features prominently because it is a convenient vehicle for taking advantage of the structure of the Hessian approximations and the structured matrices that typically define QS functions. Other alternatives, however, may be preferable, depending on the application.

For example, we might choose to reduce the 3-by-3 matrix in (4.4.2) to an equivalent symmetrized system

$$\begin{pmatrix} -Q & A^T \\ A & D \end{pmatrix} \begin{pmatrix} \Delta y \\ \Delta s \end{pmatrix} = - \begin{pmatrix} -r_d \\ r_p + V^{-1} r_\mu \end{pmatrix}$$

with $D := V^{-1}S$. As described by [BW08], Krylov-based method, such as MINRES [PS75], may be applied to a preconditioned system, using the preconditioner

$$P = \begin{pmatrix} -\mathcal{L}(u) & \\ & D \end{pmatrix},$$

where $\mathcal{L}(u)$ is defined in (4.4.5). This "ideal" preconditioner clusters the spectrum into three distinct values, so that in exact arithmetic, MINRES would converge in three iterations. The application of the preconditioner requires solving systems with $\mathcal{L}$ and D, and so all of the techniques discussed in §4.5 apply. One benefit, however, which we have not explored here, is that the preconditioning approach allows us to approximate $\mathcal{L}^{-1}(u)$, rather than to compute it exactly, which may yield computational efficiencies for some problems.

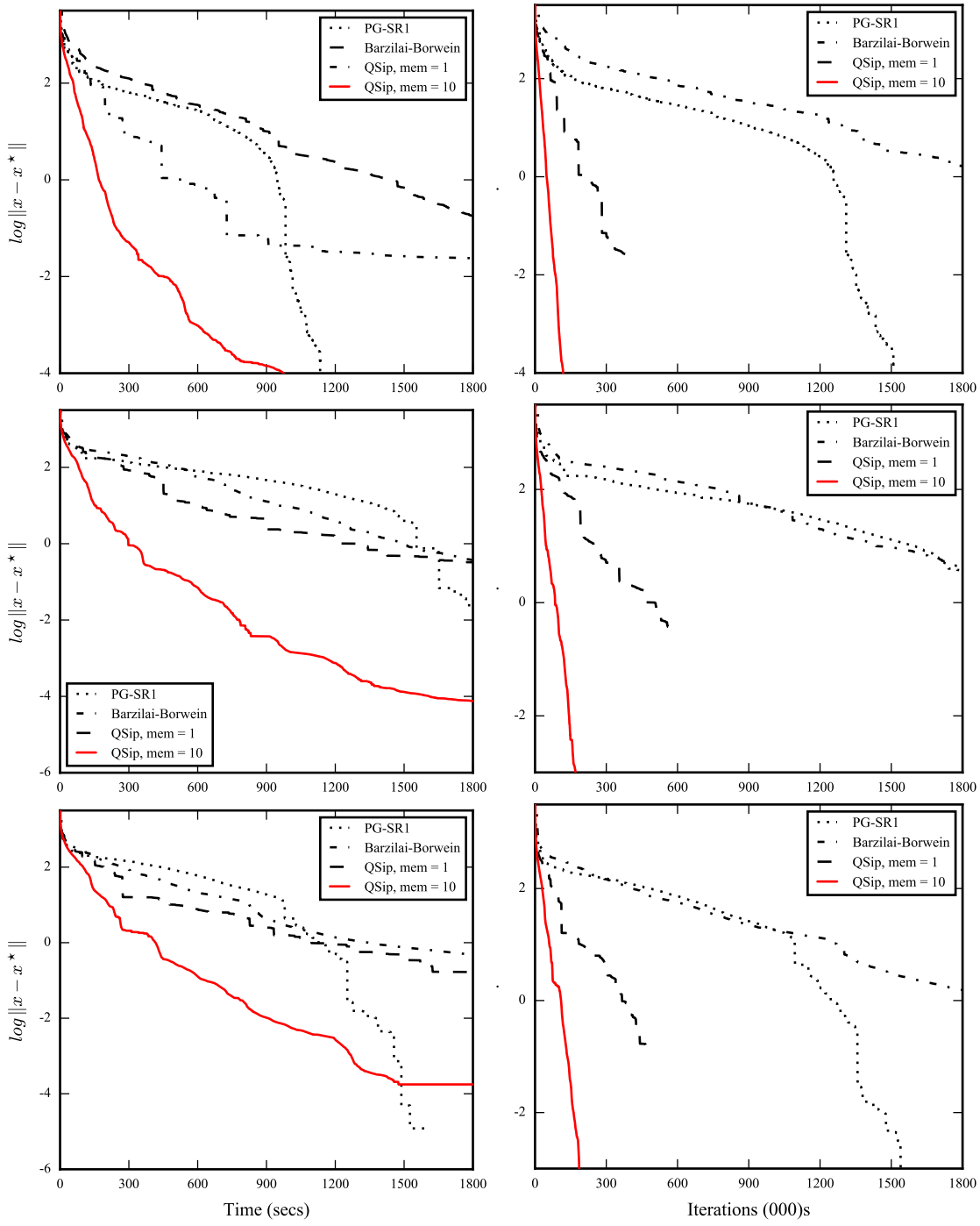

FIGURE 4.7.2. Performance of solvers applied to 1-norm regularized least-squares problems of increasing difficulty. The left and right columns, respectively, track the distance of the current solution estimate to the true solution versus time and iteration number. Top row: $p = 2000$ (highest coherence); middle row: $p = 1000$; bottom row: $p = 500$ (lowest coherence).

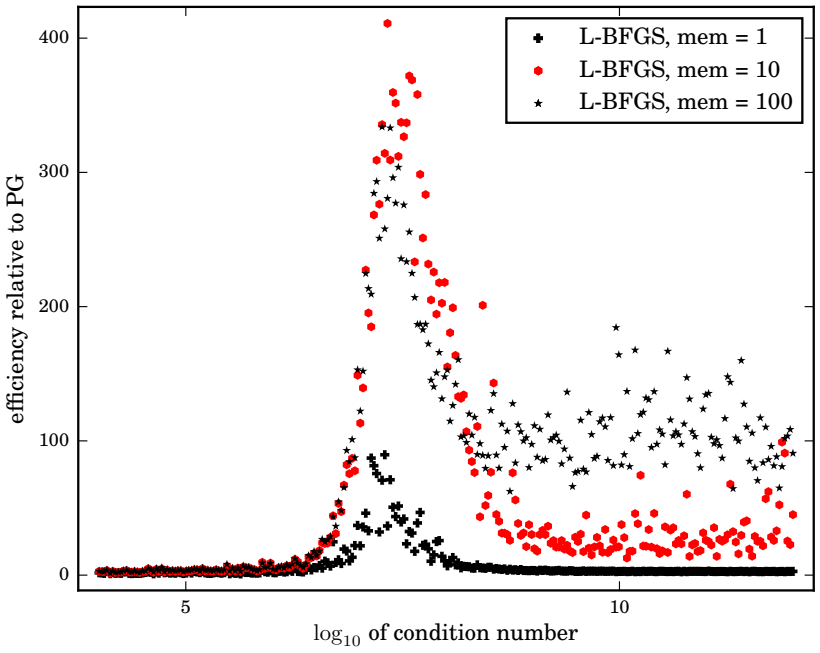

FIGURE 4.7.3. Performance of the proximal quasi-Newton method relative to proximal gradient for problems of varying condition number.

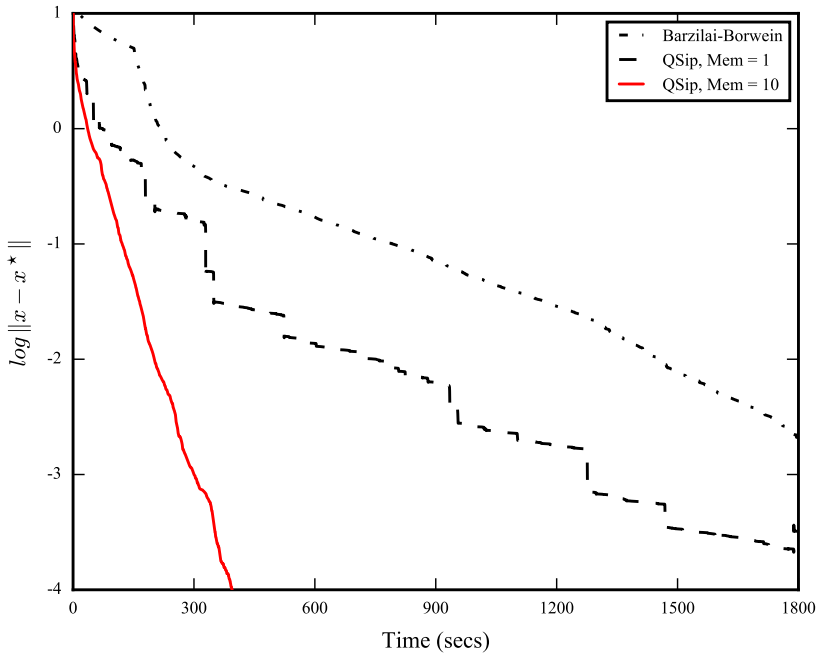

FIGURE 4.7.4. Performance of solvers applied to a group-Lasso problem. The horizontal axis measures elapsed time; the vertical axis measures distance to the solution.

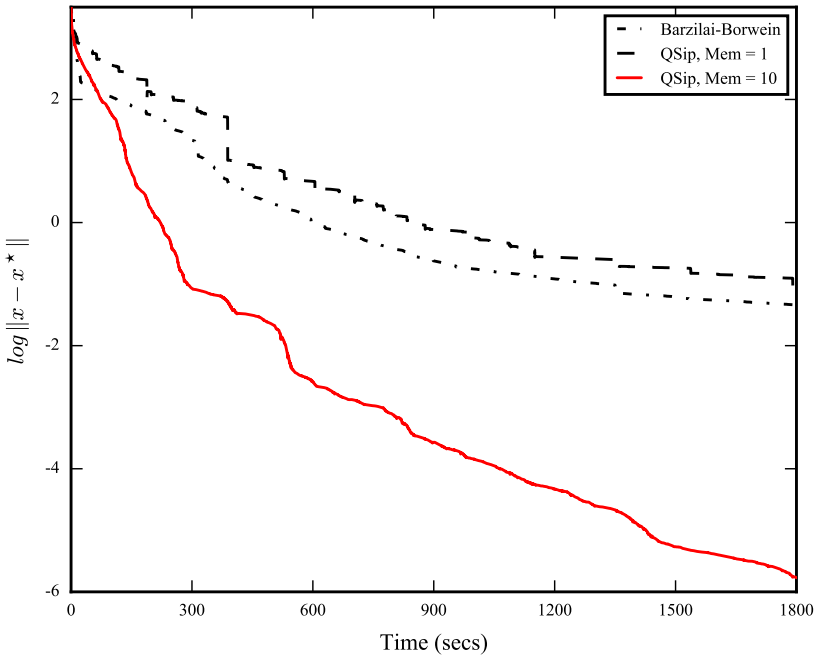

FIGURE 4.7.5. Performance of the `QSip` solver applied to a 1-dimensional total-variation problem.

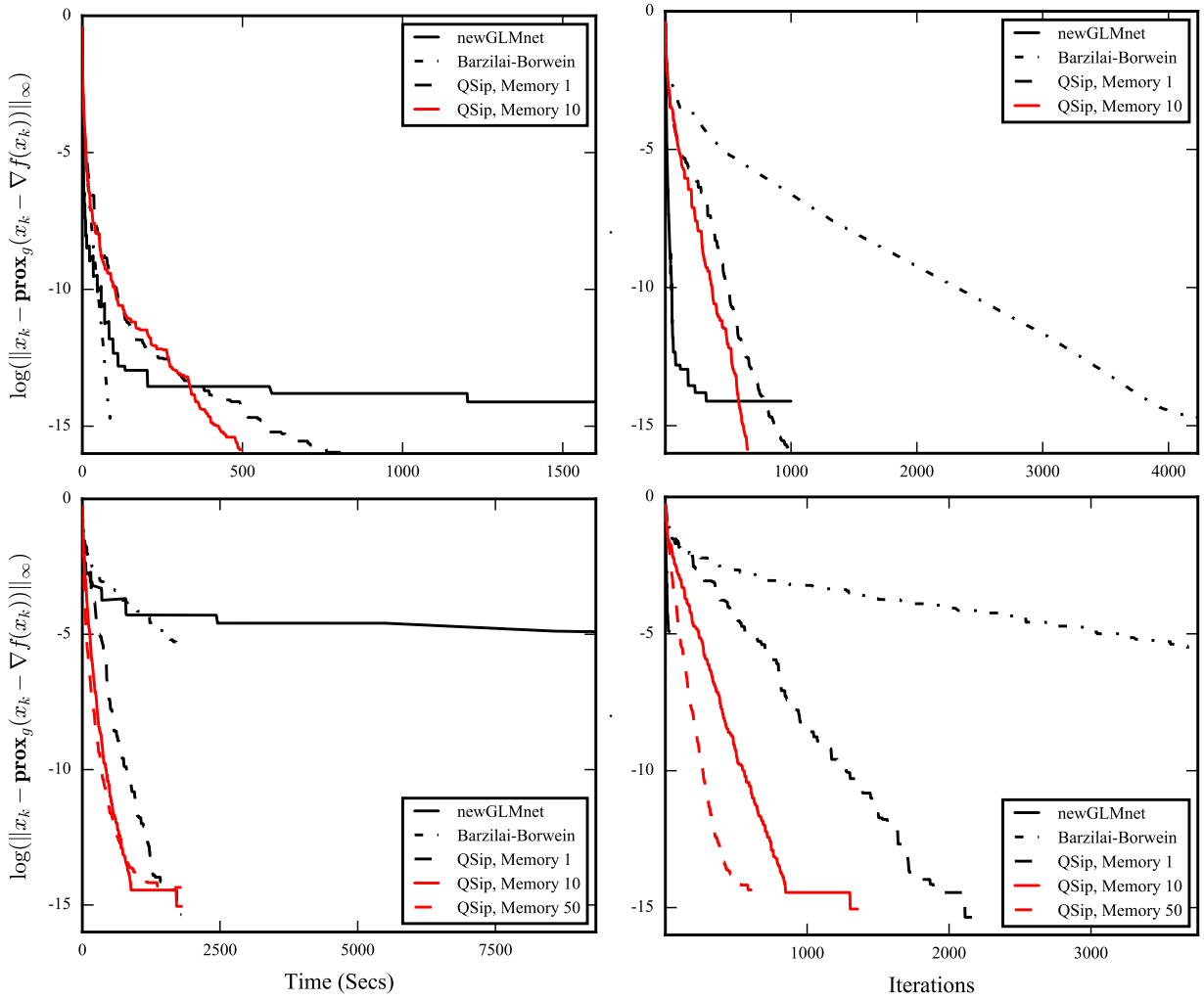

FIGURE 4.7.6. Performance of solvers on a sparse logistic-regression problem. Top row: *Gisette* dataset; bottom row: *Epsilon* dataset. The left and right columns, respectively, track the optimality of the current solution estimate versus elapsed time and iteration number.

APPENDIX A

A Title of Appendix A

A.1. Auxiliary results

Lemma A.1.1. Suppose that

$$\phi_1(k) = \mathcal{O}(k^{\mathcal{O}(1)}) \exp(-\mathcal{O}(k^{\mathcal{O}(1)})),$$
$$\phi_2(k) = \exp(\mathcal{O}(k^{\mathcal{O}(1)})) \exp(-\exp(\mathcal{O}(k^{\mathcal{O}(1)}))),$$

where $\mathcal{O}(1)$ stands for positive constants. Then for each $A > 0$ there exists a positive constant C_A such that

$$\phi_1(k) \leq C_A k^{-A}, \tag{A.1.1}$$

$$\phi_2(k) \leq C_A A^{-k}. \tag{A.1.2}$$

Proof. The statement follows by taking the logarithms on both sides of (A.1.1) and (A.1.2). □

Lemma A.1.2. For $y \in (0,1)$ and $x \in [0,1]$,

$$(1-x)^{1-1/\log y} \leq \prod_{i=0}^{\infty}(1-xy^i). \tag{A.1.3}$$

Proof. To prove the lower bound, we use the following fact:

$$\ln(1-x) \geq -\frac{x}{1-x} \quad \text{for all} \quad x \in [0,1).$$

Therefore,

$$\begin{aligned}\prod_{i=1}^{\infty}(1-xy^i) &= \exp\left(\sum_{i=1}^{\infty}\log\left(1-xy^i\right)\right)\\ &\geq \exp\left(\sum_{i=1}^{\infty}-\frac{y^i}{1/x-y^i}\right)\\ &\geq \exp\left(-\int_0^{\infty}\frac{y^i}{1/x-y^i}di\right)\\ &= \exp\left(-\frac{\log(1-x)}{\log(y)}\right)\geq (1-x)^{-1/\log y}.\end{aligned}$$

Thus,

$$\prod_{i=0}^{\infty}(1-xy^i) = (1-x)\prod_{i=1}^{\infty}(1-xy^i)\geq (1-x)^{1-1/\log y},$$

as required. □

Lemma A.1.3. For $y\in(0,1)$ and $x\in[0,1]$,

$$\exp\left(-\frac{\log(1-x/y)-\log(1-xy^{N+1})}{\log(y)}\right)\leq\prod_{i=0}^{N}(1-xy^i).$$

Proof. Similar to the proof of the previous inequality

$$\begin{aligned}\prod_{i=1}^{N}(1-xy^i) &= \exp\left(\sum_{i=1}^{N}\log\left(1-xy^i\right)\right)\\ &\geq \exp\left(\sum_{i=1}^{N}-\frac{xy^i}{1-xy^i}\right)\\ &\geq \exp\left(-\int_0^{N}\frac{xy^i}{1-xy^i}di\right)\\ &\geq \exp\left(-\frac{\log(1-x)-\log\left(1-xy^N\right)}{\log(y)}\right).\end{aligned}$$

Thus,

$$\prod_{i=0}^{N}(1-xy^i) = \prod_{i=1}^{N+1}(1-(x/y)y^i)$$
$$\geq \exp\left(-\frac{\log(1-x/y)-\log(1-xy^{N+1})}{\log(y)}\right),$$

as required. □

Lemma A.1.4. Let $k>0$, $\mu>0$, and $\epsilon>0$. Then for $y\in(0,1)$ and $x\in(0,1]$,

$$\inf_{\theta>0}\left\{\exp(-\theta\epsilon\nu)\prod_{i=0}^{N-1}\left(1-\theta xy^i\right)^{-k}\right\} \leq \left(\frac{\exp(1)}{\alpha}\cdot\frac{\epsilon\nu}{x}\right)^{\alpha}\exp\left(-\frac{\epsilon\nu}{x}\right),$$

where $\alpha=\frac{1}{k}\left(\frac{1}{\log(1/y)}+1\right)$.

Proof. By inverting both sides of (A.1.3) we obtain the following inequality

$$\prod_{i=0}^{\infty}(1-xy^i)^{-k} \leq \exp\left(-\log(1-x)\left[\frac{1}{\log(1/y)}+1\right]\right). \tag{A.1.4}$$

Therefore, for $\epsilon\geq\alpha x/v$,

$$\begin{aligned}
&\inf_{\theta>0}\left\{\exp(-\theta\epsilon\nu)\prod_{i=0}^{N-1}(1-\theta xy^i)^{-k}\right\}\\
&\quad\leq \inf_{\theta>0}\left\{\exp(-\theta\epsilon\nu)\prod_{i=0}^{\infty}(1-\theta xy^i)^{-k}\right\}\\
&\quad\overset{(i)}{\leq} \inf_{\theta>0}\left\{\exp\left(-\frac{1}{k}\left[\frac{1}{\log(1/y)}+1\right]\log(1-\theta x)-\theta v\epsilon\right)\right\}\\
&\quad= \inf_{\theta>0}\left\{\exp\left(-\alpha\log(1-\theta x)-\theta\epsilon\nu\right)\right\}\\
&\quad\overset{(ii)}{=} \exp\left(-\alpha\log\left(1-\left(\frac{1}{x}-\frac{\alpha}{v\epsilon}\right)x\right)-\left(\frac{1}{x}-\frac{\alpha}{v\epsilon}\right)v\epsilon\right)\\
&\quad= \left(\frac{\exp(1)}{\alpha}\cdot\frac{\epsilon\nu}{x}\right)^{\alpha}\exp\left(-\frac{\epsilon\nu}{x}\right),
\end{aligned}$$

where (i) follows from (A.1.4); and (ii) uses the substitution $\theta=1/x-\alpha/v\epsilon$, which can be shown to be the optimal choice of θ. Because $\theta>0$, $\epsilon>\alpha x/v$. □

For the remainder of this section, define the sample average to be

$$S_m := \frac{1}{m} \sum_{i}^{m} X_i$$

for a sequence of random variables $\{X_1, \ldots, X_m\}$.

A.2. Sampling Bounds

Theorem A.2.1 ([Hoe63, Theorem 2])**.** Consider independent random variables $\{X_1, \ldots, X_m\}$, $X_i : \Omega \to \Re$. If the random variables are bounded, i.e.,

$$d := \sup_{\omega \in \Omega} X_i(\omega) - \inf_{\omega \in \Omega} X_i(\omega)$$

is finite, then

$$\Pr\left(S_m - \mathbf{E}S_m \geq \epsilon\right) \leq \exp\left(-2m\epsilon^2/d^2\right)$$

Theorem A.2.2 ([Ser74, Corollary 1.1])**.** Let $x_1, \ldots, x_M$ be a population, $\{X_1, \ldots, X_m\}$ be samples drawn without replacement from the population, and let

$$d := \max_i x_i - \min_i x_i.$$

Then

$$\Pr\left(S_m - \mathbf{E}S_m \geq \epsilon\right) \leq \exp\left(-\epsilon^2/\eta_m\right), \quad \text{where} \quad \eta_m = \frac{d^2}{2m}\left(1 - \frac{m-1}{M}\right).$$

Because η_m is strictly decreasing in m, the Serfling bound is uniformly better than the Hoeffding bound. Note that the Serfling bound is not tight: in particular, when $M = m$ (i.e., $S_m = \mathbf{E}S_m$), the bound is not zero (except for degenerate population).

APPENDIX B

A Title of Appendix B

B.1. QS Representation for a quadratic

Here we derive the QS representation of a support function that includes an explicit quadratic term:

$$g(x) = \sup_y \{y^T(B_0x + d_0) - \tfrac{1}{2}y^TQy \mid A_0y \succeq_{\mathcal{K}_0} b_0\}.$$

Let R be such that $R^TR = Q$. We can then write the quadratic function in the objective as a constraint its epigraph, i.e.,

$$g(x) = \sup_{y,\,t} \{y^T(B_0x + d_0) - \tfrac{1}{2}t \mid A_0y \succeq_{\mathcal{K}} b_0,\ \|Ry\|^2 \le t\}.$$

Next we write the constraint $\|Ry\|^2 \le t$ as a second-order cone constraint:

$$\begin{aligned}
\|Ry\|^2 \le t &\iff \|Ry\|^2 \le \frac{(t+1)^2 - (t-1)^2}{4} \\
&\iff \|Ry\|^2 + \left(\frac{t-1}{2}\right)^2 \le \left(\frac{t+1}{2}\right)^2 \\
&\iff \sqrt{\|Ry\|^2 + \left(\frac{t-1}{2}\right)^2} \le \frac{t+1}{2} \\
&\iff \left\| \begin{pmatrix} 0 & \frac{1}{2} \\ R & 0 \end{pmatrix} \begin{pmatrix} y \\ t \end{pmatrix} + \begin{pmatrix} -\frac{1}{2} \\ 0 \end{pmatrix} \right\| \le \frac{t+1}{2} \\
&\iff \begin{pmatrix} 0 & \frac{1}{2} \\ 0 & \frac{1}{2} \\ R & 0 \end{pmatrix} \begin{pmatrix} y \\ t \end{pmatrix} \preceq_Q \begin{pmatrix} \frac{1}{2} \\ -\frac{1}{2} \\ 0 \end{pmatrix}.
\end{aligned}$$

Concatenating this with the original constraints gives a QS function with parameters

$$A = \begin{pmatrix} 0 & \frac{1}{2} \\ 0 & \frac{1}{2} \\ R & 0 \\ A_0 & 0 \end{pmatrix}, \quad b = \begin{pmatrix} \frac{1}{2} \\ -\frac{1}{2} \\ 0 \\ b_0 \end{pmatrix}, \quad d = \begin{pmatrix} d_0 \\ -\frac{1}{2} \end{pmatrix}, \quad B = \begin{pmatrix} B_0 \\ 0 \end{pmatrix}, \quad \mathcal{K} = \mathcal{Q}^{n+2} \times \mathcal{K}_0.$$

Bibliography

[ABP13] A. Y. Aravkin, J. V. Burke, and G. Pillonetto, *Sparse/robust estimation and Kalman smoothing with nonsmooth log-concave densities: Modeling, computation, and theory*, J. Mach. Learn. Res. **14** (2013), no. 1, 2689–2728.

[ADV10] M. Andersen, J. Dahl, and L. Vandenberghe, *Implementation of nonsymmetric interior-point methods for linear optimization over sparse matrix cones*, Math. Program. Comp. **2** (2010), no. 3-4, 167–201 (English).

[Azu67] K. Azuma, *Weighted sums of certain dependent random variables*, Tohoku Mathematical Journal **19** (1967), no. 3, 357–367.

[BB88] J. Barzilai and J. M. Borwein, *Two-point step size gradient methods*, IMA J. Numer. Anal. **8** (1988), 141–148.

[BC90] M. J. Best and N. Chakravarti, *Active set algorithms for isotonic regression; a unifying framework*, Math. Program. **47** (1990), no. 1, 425–439.

[BEKS14] J. Bezanson, A. Edelman, S. Karpinski, and V. B. Shah, *Julia: A fresh approach to numerical computing*, November 2014, 1411.1607.

[Ber09] D. P. Bertsekas, *Convex optimization theory*, Athena Scientific Belmont, MA, 2009.

[BF12] S. Becker and J. Fadili, *A quasi-newton proximal splitting method*, Advances in Neural Information Processing Systems 25 (F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, eds.), Curran Associates, Inc., 2012, pp. 2618–2626.

[BGT81] R. G. Bland, D. Goldfarb, and M. J. Todd, *The ellipsoid method: A survey*, Operations research **29** (1981), no. 6, 1039–1091.

[BGVDM94] O. Bahn, J.-L. Goffin, J.-P. Vial, and O. Du Merle, *Experimental behavior of an interior point cutting plane algorithm for convex programming: an application to geometric programming*, Discrete Applied Mathematics **49** (1994), no. 1-3, 3–23.

[BH13] J. V. Burke and T. Hoheisel, *Epi-convergent smoothing with applications to convex composite functions*, SIAM J. Optim. **23** (2013), no. 3, 1457–1479.

[BNO16] R. H. Byrd, J. Nocedal, and F. Oztoprak, *An inexact successive quadratic approximation method for L-1 regularized optimization*, Math. Program. **157** (2016), no. 2, 375–396.

[BS11] A. Barbero and S. Sra, *Fast Newton-type methods for total variation regularization.*, Intern. Conf. on Machine Learning (L. Getoor and T. Scheffer, eds.), Omnipress, 2011, pp. 313–320.

[BS14] A. Barbero and S. Sra, *Modular proximal optimization for multidimensional total-variation regularization*, 2014, arXiv 0902.0885.

[BT00] D. P. Bertsekas and J. N. Tsitsiklis, *Gradient convergence in gradient methods with errors*, SIAM J. Optim. **10** (2000), no. 3, 627–642.

[BT08] A. Beck and M. Teboulle, *A fast iterative shrinkage-thresholding algorithm for linear inverse problems*, Tech. report, Department of Industrial Engineering and Management, Technion, Haifa, 2008.

[BT09] A. Beck and M. Teboulle, *A fast iterative shrinkage-thresholding algorithm for linear inverse problems*, SIAM J. Imag. Sci. **2** (2009), no. 1, 183–202.

[BV04] D. Bertsimas and S. Vempala, *Solving convex programs by random walks*, Journal of the ACM **51** (2004), no. 4, 540–556.

[BV07] S. Boyd and L. Vandenberghe, *Localization and cutting-plane methods*, From Stanford EE 364b lecture notes (2007).

[BW08] M. Benzi and A. J. Wathen, *Some preconditioning techniques for saddle point problems*, Model order reduction: theory, research aspects and applications, Springer, 2008, pp. 195–211.

[CL06] F. Chung and L. Lu, *Concentration inequalities and martingale inequalities: a survey*, Internet Mathematics **3** (2006), no. 1, 79–127.

[CLO14] Y. Chen, G. Lan, and Y. Ouyang, *Optimal primal-dual methods for a class of saddle point problems*, SIAM Journal on Optimization **24** (2014), no. 4, 1779–1814.

[CMMP13] H. H. Chin, A. Madry, G. L. Miller, and R. Peng, *Runtime guarantees for regression problems*, Proceedings of the 4th conference on Innovations in Theoretical Computer Science, ACM, 2013, pp. 269–282.

[CW05] P. L. Combettes and V. R. Wajs, *Signal recovery by proximal forward-backward splitting*, Multiscale Model. Simul. **4** (2005), no. 4, 1168–1200.

[DES82] R. S. Dembo, S. C. Eisenstat, and T. Steihaug, *Inexact Newton methods*, SIAM J. Numer. Anal. **19** (1982), no. 2, 400–408.

[DFR16] D. Drusvyatskiy, M. Fazel, and S. Roy, *An optimal first order method based on optimal quadratic averaging*, arXiv preprint arXiv:1604.06543 (2016).

[FCH+08] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, *LIBLINEAR: A library for large linear classification*, Journal of Machine Learning Research **9** (2008), 1871–1874.

[FS12] M. P. Friedlander and M. Schmidt, *Hybrid deterministic-stochastic methods for data fitting*, SIAM Journal on Scientific Computing **34** (2012), no. 3, A1380–A1405.

[GGBHD04] I. Guyon, S. Gunn, A. Ben-Hur, and G. Dror, *Result analysis of the NIPS 2003 feature selection challenge*, Advances Neural Inform. Processing Systems **17** (2004), 545–552.

[GL89] G. H. Golub and C. F. V. Loan, *Matrix computations*, second ed., Johns Hopkins University Press, Baltimore, 1989.

[GLS12] M. Grötschel, L. Lovász, and A. Schrijver, *Geometric algorithms and combinatorial optimization*, vol. 2, Springer Science & Business Media, 2012.

[GV99] J.-L. Goffin and J.-P. Vial, *A two-cut approach in the analytic center cutting plane method*, Mathematical methods of Operations Research **49** (1999), no. 1, 149–169.

[Hoe63] W. Hoeffding, *Probability inequalities for sums of bounded random variables*, J. American Stat. Assoc. **58** (1963), no. 301, 13–30.

[Imh61] J. P. Imhof, *Computing the distribution of quadratic forms in normal variables*, Biometrika (1961), 419–426.

[JMBO10] R. Jenatton, J. Mairal, F. R. Bach, and G. R. Obozinski, *Proximal methods for sparse hierarchical dictionary learning*, Proc. 27th Intern. Confer. Machine Learning (ICML-10), 2010, pp. 487–494.

[Kel60] J. E. Kelley, Jr, *The cutting-plane method for solving convex programs*, Journal of the society for Industrial and Applied Mathematics **8** (1960), no. 4, 703–712.

[KKL+07] S.-J. Kim, K. Koh, M. Lustig, S. Boyd, and D. Gorinevsky, *An interior-point method for large-scale L1-regularized least squares*, IEEE J. Sel. Top. Signal Process. **1** (2007), no. 4, 606–617.

[KV14] S. Karimi and S. Vavasis, *IMRO: a proximal quasi-Newton method for solving l_1-regularized least squares problem*, 2014, arXiv:1401.4220.

[Lem75] C. Lemaréchal, *An extension of davidon methods to non differentiable problems*, Nondifferentiable optimization (1975), 95–109.

[Lev65] A. Y. Levin, *On an algorithm for the minimization of convex functions*, Soviet Mathematics Doklady, vol. 160, 1965, pp. 1244–1247.

[Lic13] M. Lichman, *UCI machine learning repository*, 2013.

[LLZ09] J. Langford, L. Li, and T. Zhang, *Sparse online learning via truncated gradient*, J. Mach. Learn. Res. **10** (2009), 777–801.

[Lor13] D. A. Lorenz, *Constructing test instances for basis pursuit denoising*, IEEE Trans. Sig. Proc. **61** (2013), no. 5, 1210–1214.

[LSS14] J. D. Lee, Y. Sun, and M. A. Saunders, *Proximal Newton-type methods for minimizing composite functions*, SIAM J. Optim. **24** (2014), no. 3, 1420–1443.

[LT93a] Z. Q. Luo and P. Tseng, *Error bounds and convergence analysis of feasible descent methods: a general approach*, Annals of Operations Research **46** (1993).

[LT93b] Z. Luo and P. Tseng, *Error bounds and convergence analysis of feasible descent methods: A general approach*, Ann. Oper. Res. **46** (1993), no. 1, 157–178.

[LY08] D. Luenberger and Y. Ye, *Linear and nonlinear programming*, Springer Verlag, 2008.

[LZOX14] Y. Lou, T. Zeng, S. Osher, and J. Xin, *A weighted difference of anisotropic and isotropic total variation model for image processing*, Tech. report, Department of Mathematics, UCLA, 2014.

[Meh00] S. Mehrotra, *Volumetric center method for stochastic convex programs using sampling*, (2000).

[NB00] A. Nedic and D. Bertsekas, *Convergence rate of incremental subgradient algorithms*, Stochastic Optimization: Algorithms and Applications (2000), 263–304.

[Nem94] A. Nemirovski, *Efficient methods in convex programming*, Lecture notes (1994).

[Nem05] A. Nemirovski, *Efficient methods in convex programming*, (2005).

[Nes07] Y. Nesterov, *Gradient methods for minimizing composite objective function*, Tech. rep., Center for Operations Research and Econometrics (CORE), Catholic University of Louvain, 2007.

[New65] D. J. Newman, *Location of the maximum on unimodal surfaces*, Journal of the ACM (JACM) **12** (1965), no. 3, 395–398.

[NJLS09] A. Nemirovski, A. Juditsky, G. Lan, and A. Shapiro, *Robust stochastic approximation approach to stochastic programming*, SIAM J. Optim. **19** (2009), no. 4, 1574–1609.

[NV08] Y. Nesterov and J.-P. Vial, *Confidence level solutions for stochastic programming*, Automatica **44** (2008), no. 6, 1559–1568.

[NW99] J. Nocedal and S. J. Wright, *Numerical optimization*, Springer, New York, 1999.

[Pan87] J.-S. Pang, *A posteriori error bounds for the linearly constrained variational inequality problem*, Math. Oper. Res. **12** (1987).

[Pas16] `http://largescale.ml.tu-berlin.de/instructions`, 2016, Accessed: 2016-11-22.

[PS75] C. C. Paige and M. A. Saunders, *Solution of sparse indefinite systems of linear equations*, siamnumanal **12** (1975), 617–629.

[RF10] H. L. Royden and P. M. Fitzpatrick, *Real analysis*, Prentice Hall, Boston 2010, 4th edition.

[RM51] H. Robbins and S. Monro, *A stochastic approximation method*, The Annals of Mathematical Statistics (1951), 400–407.

[Roc70] R. T. Rockafellar, *Convex analysis*, Princeton University Press, Princeton, 1970.

[RW98] R. T. Rockafellar and R. J. B. Wets, *Variational analysis*, vol. 317, Springer, 1998, Corrected 3rd printing.

[Ser74] R. Serfling, *Probability inequalities for the sum in sampling without replacement*, Ann. Statist. **2** (1974), no. 1, 39–48.

[So13] A. M.-C. So, *Non-asymptotic convergence analysis of inexact gradient methods for machine learning without strong convexity*, `http://www.se.cuhk.edu.hk/~manchoso/papers/inexact_GM_conv.pdf`, August 2013.

[SRB11] M. Schmidt, N. L. Roux, and F. Bach, *Convergence rates of inexact proximal-gradient methods for convex optimization*, arXiv preprint arXiv:1109.2415 (2011).

[ST16] K. Scheinberg and X. Tang, *Practical inexact proximal quasi-Newton method with global complexity analysis*, Math. Program. **160** (2016), no. 1, 495–529.

[SvdBFM09] M. Schmidt, E. van den Berg, M. P. Friedlander, and K. Murphy, *Optimizing costly functions with simple constraints: a limited-memory projected quasi-Newton algorithm*, Proc. 12th Inter. Conf. Artificial Intelligence and Stat., April 2009, pp. 448–455.

[Tar88] E. Tarasov, Khachiyan, *The method of inscribed ellipsoids*, Soviet Mathematics Doklady **298** (1988), no. 5, 1081–1085.

[TDKC15] Q. Tran-Dinh, A. Kyrillidis, and V. Cevher, *Composite self-concordant minimization*, J. Machine Learning Research **16** (2015), 371–416.

[Tse10] P. Tseng, *Approximation accuracy, gradient methods, and error bound for structured convex optimization*, Math. Program. **125** (2010), 263–295.

[TY09] P. Tseng and S. Yun, *A coordinate gradient descent method for nonsmooth separable minimization*, Math. Program. **117** (2009).

[Vai89] P. M. Vaidya, *A new algorithm for minimizing convex functions over convex sets*, Foundations of Computer Science, 1989., 30th Annual Symposium on, IEEE, 1989, pp. 338–343.

[Van10] L. Vandenberghe, *The CVXOPT linear and quadratic cone program solvers*, 2010.

[WL14] P.-W. Wang and C.-J. Lin, *Iteration complexity of feasible descent methods for convex optimization.*, Journal of Machine Learning Research **15** (2014), no. 1, 1523–1548.

[WNF09] S. J. Wright, R. D. Nowak, and M. A. Figueiredo, *Sparse reconstruction by separable approximation*, IEEE Trans. Sig. Proc. **57** (2009), no. 7, 2479–2493.

[Wol75] P. Wolfe, *A method of conjugate subgradients for minimizing nondifferentiable functions*, Nondifferentiable optimization, Springer, 1975, pp. 145–173.

[Ye96] Y. Ye, *Complexity analysis of the analytic center cutting plane method that uses multiple cuts*, Mathematical Programming **78** (1996), no. 1, 85–104.

[Yip86] E. Yip, *A note on the stability of solving a rank-p modification of a linear system by the Sherman-Morrison-Woodbury formula*, SIAM J. Sci. Stat. Comput. **7** (1986), no. 2, 507–513.

[YL06] M. Yuan and Y. Lin, *Model selection and estimation in regression with grouped variables*, J. Royal Stat. Soc. B. **68** (2006).

[ZhYDR14] K. Zhong, E. hsu Yen, I. S. Dhillon, and P. K. Ravikumar, *Proximal quasi-newton for computationally intensive l1-regularized m-estimators*, Advances Neural Inform. Processing Systems 27 (Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Weinberger, eds.), Curran Associates, Inc., 2014, pp. 2375–2383.