Two Numerical Methods for Optimizing Matrix Stability

James V. Burke Adrian S. Lewis Michael L. Overton

March 30, 2001

Abstract

Consider the affine matrix family $A(x) = A_0 + \sum_{k=1}^m x_k A_k$, mapping a design vector $x \in \mathbf{R}^m$ into the space of $n \times n$ real matrices. We are interested in the question of how to choose x to optimize the stability of the dynamical system $\dot{z} = A(x)z$. A classic example in control is stabilization by output feedback. We take two approaches. The first is to directly minimize $\alpha(A(x))$, the spectral abscissa (the largest real part of the eigenvalues) of $A(x)$, since this quantity bounds the asymptotic decay rate of the trajectories of the dynamical system. The spectral abscissa $\alpha(X)$ is a continuous but nonsmooth, in fact non-Lipschitz, function of the matrix argument X, and finding a global minimizer of $\alpha(A(x))$ is difficult. We introduce a novel random gradient bundle method for approximating *local* minimizers, motivated by recent work on nonsmooth analysis of the function $\alpha(X)$. Our second approach is to minimize a related function $\alpha_\delta(A(x))$, where δ is a *robustness* parameter in (0,1). The motivation for the definition of the "robust spectral abscissa" $\alpha_\delta(X)$ is that it bounds transient peaks as well as asymptotic decay of trajectories of $\dot{z} = Xz$. The function $\alpha_\delta(X)$ is Lipschitz but typically nonconvex for $\delta \in (0,1)$, approaching $\alpha(X)$ as $\delta \to 0$ and the largest eigenvalue of $\frac{1}{2}(X + X^T)$ as $\delta \to 1$. We use a Newton barrier method to approximate local minimizers of $\alpha_\delta(A(x))$. We compare the results of the two approaches on a number of interesting test cases.

1 Introduction

The spectral abscissa α of a square matrix is the maximum of the real parts of its eigenvalues. It plays a crucial role in the asymptotic analysis of dynamical systems. Consider the system $\dot{z}(t) = Xz(t)$. As is well known, the trajectory norm $\|z(t)\| = \|e^{tX}z(0)\|$ is bounded asymptotically by $e^{\gamma t}\|z(0)\|$,

for any γ greater than $\alpha(X)$, the spectral abscissa of X. In particular, if $\alpha(X) < 0$, the system is asymptotically stable.

Consider the affine matrix family $A(x) = A_0 + \sum_{k=1}^{m} x_k A_k$, mapping a design vector $x \in \mathbf{R}^m$ into $\mathbf{M}^n(\mathbf{R})$, the space of $n \times n$ real matrices. We are interested in choosing x to optimize the stability of the dynamical system $\dot{z}(t) = A(x)z(t)$. If we focus exclusively on asymptotic stability, we have a *spectral abscissa minimization* problem: find a vector $\bar{x}$ that minimizes $\alpha(A(x))$, locally or globally.

For example, the differential equation describing a damped linear oscillator is

$$\ddot{w}(t) + \xi \dot{w}(t) + w(t) = 0.$$

Here $\xi \in \mathbf{R}$ is the parameter that controls damping; when $\xi = 0$, the solutions are pure oscillations. The equivalent first-order system in the vector $z = [w \;\; \dot{w}]^T$ is

$$\dot{z}(t) = A(\xi)\; z(t), \quad \text{where} \quad A(\xi) = \begin{bmatrix} 0 & 1 \\ -1 & -\xi \end{bmatrix}. \tag{1}$$

The trajectory norm $\|z(t)\|$ is bounded asymptotically by $e^{\gamma t}\|z(0)\|$, for any γ greater than the spectral abscissa of $A(\xi)$,

$$\alpha(A(\xi)) = \begin{cases} -\frac{\xi}{2} & \text{if } |\xi| \le 2 \\ -\frac{\xi}{2} + \sqrt{\frac{\xi^2}{4} - 1} & \text{if } |\xi| \ge 2 \end{cases}.$$

This is minimized by choosing $\xi = 2$, yielding $\alpha(A(2)) = -1$. Note that $A(2)$ has a nonderogatory double eigenvalue -1. A *nonderogatory* eigenvalue is one whose geometric multiplicity is one, i.e., the eigenvalue is associated with a single Jordan block. By contrast, a *semisimple* eigenvalue is one whose geometric multiplicity equals its algebraic multiplicity, i.e., the eigenvalue has no Jordan blocks with size two or more.

Another interesting example is

$$A(x) = \begin{bmatrix} -x_1 & 1 & 0 & \cdot & \cdot & 0 \\ x_1 & 0 & 1 & 0 & \cdot & 0 \\ x_2 & 0 & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & 1 \\ x_{n-1} & 0 & \cdot & \cdot & \cdot & 0 \end{bmatrix}. \tag{2}$$

Here $m = n - 1$. Figure 1 shows the contours of the spectral abscissa $\alpha(A(x))$ in the case $n = 3$ ($m = 2$). The spectral abscissa is minimized by

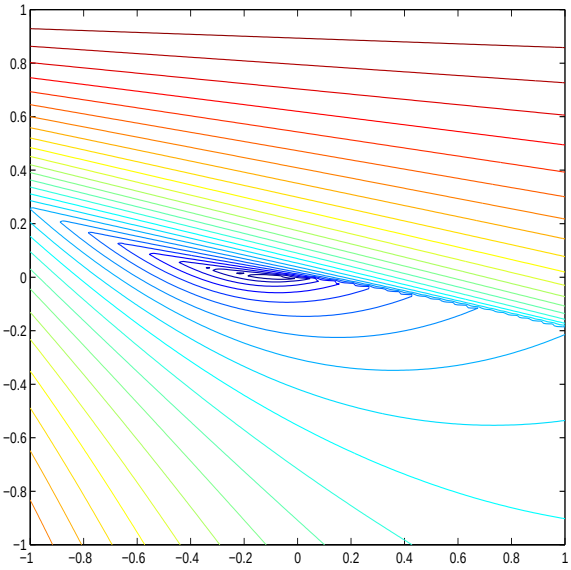

Figure 1: Contours of composite function $\alpha \circ A$

$x_1 = x_2 = 0$, for which $A(x)$ is a Jordan block (with the nonderogatory triple eigenvalue zero). A manifold is clearly visible on which the spectral abscissa is nonsmooth; on this manifold, which is tangent to the x_1 axis at (0,0), $A(x)$ has a nonderogatory double eigenvalue (except at (0,0) where it is triple). As x_2 is increased above 0, keeping $x_1 = 0$, the triple eigenvalue splits into three distinct eigenvalues, a real eigenvalue in the right half plane and a complex conjugate pair in the left, and the spectral abscissa accordingly increases sharply (the growth is $O(x_2^{1/3})$). When x_2 is decreased below 0, the growth in the spectral abscissa is still $O(x_2^{1/3})$, but with asymptotic factor reduced by half; this is because the real eigenvalue is in the left half plane and the complex conjugate pair is in the right half plane. On the other hand, when x_2 is fixed equal to 0 and x_1 is varied, the growth in α is much slower, $O(x_1^{1/2})$, though still non-Lipschitz. This simple example illustrates how challenging spectral abscissa minimization is in general, even locally.

More complicated spectral abscissa minimization problems often arise in control applications. A particularly important example (described in [BT97] as "perhaps the most basic problem in control theory") is *stabilization by output feedback*: given an $n \times n$ matrix $\bar{A}$, an $n \times r$ matrix $\bar{B}$ and an $s \times n$ matrix $\bar{C}$, find (if possible) an $r \times s$ matrix K such that $\bar{A} + \bar{B}K\bar{C}$ is stable, i.e., has all its eigenvalues in the left half of the complex plane. When

interval bounds on the entries of K are specified, the problem is NP-hard [BT97, Nem93]. The complexity of the problem without bounds on K is not known. Since the map $K \mapsto \bar{A} + \bar{B}K\bar{C}$ is affine in K, stabilization by output feedback can be expressed as a spectral abscissa global minimization problem. In a 1995 survey [BGL95], experts in control theory described the characterization of triples $(\bar{A}, \bar{B}, \bar{C})$ such that a stabilizing K exists as a "major open problem". Although we do not expect to devise an efficient algorithm to find global spectral abscissa minimizers, it is clear that reliable and efficient methods to find local minimizers could be of great value in practice. In the past, such methods have not been studied systematically; optimality conditions were not known, and though methods for moving eigenvalues left in the complex plane have certainly been proposed from time to time, they have generally been fairly ad hoc, and usually assume that the eigenvalues in question are simple or perhaps double. The related control literature is huge; besides the surveys already cited, we mention [HHB98, PS99]. Although we confine our attention to the spectral abscissa, spectral radius minimization problems also frequently arise in practice, e.g., in optimizing convergence of iterative processes such as Markov chains.

We say that an eigenvalue of a matrix X is *active* if its real part equals its spectral abscissa $\alpha(X)$. It is not hard to see that the spectral abscissa map

$$X \in \mathbf{M}^n(\mathbf{R}) \mapsto \alpha(X) \in \mathbf{R} \tag{3}$$

is non-Lipschitz at all $\bar{X}$ for which $\bar{X}$ has a non-semisimple active eigenvalue. It is well known [Arn71] that such matrices form a manifold in matrix space, and that the codimension of this manifold is one. Consequently, this manifold transversally intersects the image of almost any affine family $A(x)$, even if m, the number of variables, is only one. An immediate consequence is that the restricted map

$$x \in \mathbf{R}^m \mapsto \alpha(A(x)) \in \mathbf{R} \tag{4}$$

inherits the non-Lipschitz nature of (3), except in trivial cases. Therefore, even if we were presented with a minimizer $\bar{x}$, verification of local optimality requires nonstandard analytical tools. These have been developed at length in [BO01], using techniques from modern nonsmooth analysis. Although we do not discuss these results in the present work, they motivate a good deal of its development. In particular we remark that the key result in [BO01] is that the map (3) is subdifferentially regular [RW98] at $\bar{X}$ if and only if all active eigenvalues of $\bar{X}$ are nonderogatory. It is not a coincidence that the spectral abscissa minimizers for $A(x)$ defined by (1) and (2) are both associated with nonderogatory multiple eigenvalues. On the contrary, this is typical;

minimizing the spectral abscissa tends to make eigenvalues coalesce, and, generically, one expects any such multiple eigenvalues to be nonderogatory. Furthermore, with m variables, one generically expects eigenvalues to have multiplicity at most $m+1$. These phenomena are discussed and analyzed in [BLO01c, BLO01b].

Section 2 of this paper presents an algorithm that can be used to find local minimizers of the spectral abscissa $\alpha(A(x))$. The technique employed is a novel one using randomly generated bundles of gradients. Numerical results are presented in Section 4, showing that the random gradient bundle method is a very effective way to find local minimizers, despite the non-Lipschitz nature of the problem. We defer consideration of convergence analysis for this algorithm to future work, but we have addressed some underlying theoretical questions about random gradient bundles in [BLO01a].

Although pure spectral abscissa minimization is very interesting from both a mathematical and a computational point of view, in practice asymptotic stability is far from the whole story. Transient stability is also of great importance. Techniques for balancing transient and asymptotic stability have been discussed for decades and many approaches are possible. In this paper, we focus on one that has the appeal of being very clean and well motivated.

The following fundamental result characterizes the spectral abscissa of a general square matrix as an optimization problem over the positive definite matrices, a convex cone in $\mathbf{S}^n$, the space of real symmetric matrices. Let $P \succ 0$ denote the condition that $P \in \mathbf{S}^n$ is positive definite. Then, for any square matrix $X \in \mathbf{M}^n(\mathbf{R})$,

$$\alpha(X) = \inf_{P \in \mathbf{S}^n, P \succ 0} \quad \frac{1}{2}\lambda_{\max}(P^{\frac{1}{2}} X P^{-\frac{1}{2}} + P^{-\frac{1}{2}} X^T P^{\frac{1}{2}}),$$

where $\lambda_{\max}$ denotes the largest eigenvalue of a symmetric matrix. In other words, the spectral abscissa $\alpha(X)$ is the optimal value of

$$\inf_{\gamma \in \mathbf{R}, P \in \mathbf{S}^n} \quad \gamma \tag{5}$$

$$\text{subject to} \quad P \succ 0, \quad 2\gamma P \succeq PX + X^T P, \tag{6}$$

where the latter inequality is meant in the semidefinite sense. Furthermore, the infimum is achieved if and only if the active eigenvalues of X are all semisimple. Notice that we can scale P arbitrarily without changing anything. For example, we could constrain $\operatorname{tr} P = 1$. Although this characterization follows easily from fundamental results of Lyapunov, it is surprisingly little known outside the control community. See [LO96] for a

proof from first principles. Note that the last constraint is bilinear in the variables γ and P. In the control community, such a constraint is known as a BMI: bilinear matrix inequality.

Asymptotic decay of trajectories of $\dot{z} = Xz$ is guaranteed if $\gamma < 0$ is feasible in (6), for some $P \succ 0$. However, transient growth is associated with the conditioning of P. It is easy to show, by differentiating the Lyapunov function $z(t)^T P z(t)$, that

$$\|z(t)\| \leq k e^{\gamma t} \|z(0)\|, \tag{7}$$

where $k = \kappa(P)$, the condition number of P. In fact, the Kreiss Matrix Theorem and its variations [TT99] state that enforcing a bound of the form (7) is, in a certain sense, *equivalent* to bounding $\kappa(P)$ in (6).

Introducing the affine map $A(x)$ discussed above, we obtain another formulation for minimization of the spectral abscissa $\alpha(A(x))$,

$$\inf_{\gamma \in \mathbf{R}, x \in \mathbf{R}^m, P \in \mathbf{S}^n} \quad \gamma \tag{8}$$

$$\text{subject to} \quad P \succ 0, \quad 2\gamma P \succeq PA(x) + A(x)^T P. \tag{9}$$

The optimization problem is now over a space whose dimension is typically much larger than m. Furthermore, as already noted, minimizing the spectral abscissa typically leads to nonderogatory multiple eigenvalues and hence to an infimum in (8)–(9) that is not achieved, i.e., any minimizing sequence (γ^r, x^r, P^r) has the condition number of P^r diverging to ∞. In other words, optimizing asymptotic decay forces arbitrarily bad transient peaks in the trajectories from some starting points.

This raises an idea: for any square matrix X and any $\delta \in (0, 1)$, define the *robust spectral abscissa* $\alpha_\delta(X)$ as the optimal value of

$$\min_{\gamma \in \mathbf{R}, P \in \mathbf{S}^n} \quad \gamma$$

$$\text{subject to} \quad \delta I \preceq P \preceq I, \quad 2\gamma P \succeq PX + X^T P.$$

We call δ the *robustness* parameter. The feasible region for P is now compact, and $\kappa(P)$ is bounded by $1/\delta$. Since P may be scaled arbitrarily in (9), we have $\alpha_\delta(X) \to \alpha(X)$ (a non-Lipschitz function) as $\delta \to 0$ and $\alpha_\delta(X) \to \frac{1}{2}\lambda_{\max}(X + X^T)$ (a convex function) as $\delta \to 1$. The function $\alpha_\delta(X)$ is Lipschitz and typically nonconvex for $\delta \in (0, 1)$. Although computing $\alpha_\delta(X)$ is an optimization problem with a BMI constraint, it is tractable via bisection on γ, by checking a sequence of semidefinite feasibility problems. This is how Figure 2 was generated, using $\delta = .03$. Figure 2 shows

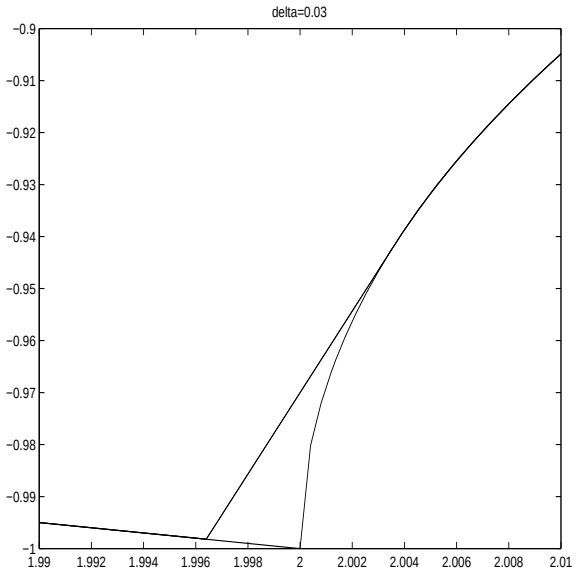

Figure 2: The Spectral Abscissa (below) and Robust Spectral Abscissa (above) for the Damped Linear Oscillator, as a function of damping

$\alpha_\delta(A(\xi))$ (the upper graph) and $\alpha(A(\xi))$ (the lower graph) as functions of ξ for the damped linear oscillator. Efficient interior point methods for solving semidefinite feasibility problems, or, more generally, semidefinite programs, have received much attention in the past decade [NN94, Tod01], especially for applications in control, where semidefinite constraints are known as LMI's: linear matrix inequalities [BGFB94, GN00].[1]

Given an affine family $A(x)$, then, the robust spectral abscissa minimization problem is to minimize $\alpha_\delta(A(x))$, i.e.,

$$\min_{\gamma\in\mathbf{R},x\in\mathbf{R}^m,P\in\mathbf{S}^n} \quad \gamma \tag{10}$$

$$\text{subject to} \quad \delta I \preceq P \preceq I, \quad 2\gamma P \succeq PA(x) + A(x)^T P. \tag{11}$$

Section 3 of the paper presents a Newton barrier method to find local minimizers of this problem. Section 4 of the paper compares the results for pure and robust spectral abscissa minimization for an interesting set of examples. Section 5 presents some conclusions.

[1] We used both the SeDuMi and SDPpack codes to generate the graph shown in Figure 2, obtaining essentially the same results with both. This was an interesting test for SDPpack, since the method it uses to detect infeasibility does not have any theoretical foundation, unlike the one used by SeDuMi.

There are many other possible approaches to balancing transient and asymptotic stability. For example, since including P as a design variable is expensive, one might define P implicitly via a constraint of the form

$$P(A(x) - \gamma I) + (A(x) - \gamma I)^T P = -I,$$

and devise a minimization objective that suitably weights γ and $\kappa(P)$.

A different approach to the balancing of transient and asymptotic stability is provided by the notion of pseudospectrum [Tre97]. Given an $\epsilon \geq 0$, the pseudospectrum of a matrix X is defined as the set of eigenvalues of all matrices Y satisfying $\|Y - X\|_2 \leq \epsilon$. Let $\tilde{\alpha}_\epsilon(X)$ denote the pseudospectral abscissa of X, i.e., the maximum of the real parts of the pseudo-eigenvalues of X (the elements of the pseudospectrum). One of the consequences of the Kreiss Matrix Theorem is that, for any $\delta \in (0, 1)$, there exist ϵ_1, ϵ_2, C_1 and C_2 such that

$$C_1 \tilde{\alpha}_{\epsilon_1}(X) < \alpha_\delta(X) < C_2 \tilde{\alpha}_{\epsilon_2}(X)$$

for every matrix X. Thus, in some sense, the notions of robust spectral abscissa and pseudospectral abscissa are equivalent, although the dependence of $\epsilon_1, \epsilon_2, C_1$ and C_2 on δ is complicated. This justifies the use of the word "robust" in the name for α_δ, since it is immediate from the definition that, if $\tilde{\alpha}_\epsilon(X) = \gamma$, then $\alpha(Y) \leq \gamma$ whenever $\|Y - X\|_2 \leq \epsilon$.

We conclude this introduction by thanking Steve Boyd, Alex Megretski and Nick Trefethen for very helpful conversations regarding the issues discussed above.

2 A Random Gradient Bundle Method

In this section we present a random gradient bundle method to approximate minimizers of general nonsmooth functions. The ideas are simple and we believe they will be useful in a very general setting, although our experience is at present limited to our specific application. The method is inspired by the great success of gradient bundle methods for convex optimization [HUL93]. These and related methods have been extended to be applicable for minimizing nonconvex but Lipschitz functions [Kiw85], but we are not aware of related work for non-Lipschitz functions such as the spectral abscissa. Our basic assumptions are simply that (a) the function f being minimized is continuous and is continuously differentiable almost everywhere and (b) where it is defined, the gradient of f is easily computed.

The assumptions just mentioned hold for the spectral abscissa map (3) since α is differentiable at X if and only if (i) X either has exactly one real

active eigenvalue, say λ, or exactly one complex conjugate pair of active eigenvalues, say $\lambda \pm i\tilde{\lambda}$, and (ii) this active eigenvalue or conjugate pair of eigenvalues is simple, i.e., has multiplicity one. Suppose this is the case. Then the gradient of (3) is easily obtained by computing the eigenvectors of X. When the active eigenvalue is real, with real right and left eigenvectors v and u satisfying

$$Xv = \lambda v, \quad X^T u = \lambda u, \tag{12}$$

with $u^T v = 1$, it is well known that the gradient is given by

$$\nabla \alpha(X) = uv^T.$$

When a conjugate pair of eigenvalues $\lambda \pm i\tilde{\lambda}$ is active, we break the eigenvectors associated with $\lambda \pm i\tilde{\lambda}$ into real and imaginary parts, so that

$$X(v + i\tilde{v}) = (\lambda + i\tilde{\lambda})(v + i\tilde{v}), \quad X^T(u + i\tilde{u}) = (\lambda - i\tilde{\lambda})(u + i\tilde{u}), \tag{13}$$

with $(u - i\tilde{u})^T(v + i\tilde{v}) = 1$. Then it is easily seen that

$$\nabla \alpha(X) = uv^T + \tilde{u}\tilde{v}^T.$$

It follows that the gradient of (4) is given by

$$\left(\nabla \left(\alpha \circ A\right)(x)\right)_k = u^T A_k v$$

when $A(x)$ has exactly one real active eigenvalue, with corresponding eigenvectors satisfying (12), and

$$\left(\nabla \left(\alpha \circ A\right)(x)\right)_k = u^T A_k v + \tilde{u}^T A_k \tilde{v}$$

when $A(x)$ has exactly one conjugate pair of active eigenvalues, with corresponding left and right eigenvectors satisfying (13).

We illustrate the main idea of the random gradient bundle method by considering the example whose contours are displayed in Figure 1. First, consider an ordinary steepest descent method with a line search, assuming that, at every iterate, $\alpha(A(x))$ is differentiable. Figure 3 plots the steepest descent iterates using plus signs, starting from the initial point $(-1, -1)$. The behavior is typical of the steepest descent method applied to a nonsmooth function; progress is good at first, but then the iterates jam near the manifold on which the function is not differentiable. The iterates of our gradient bundle algorithm are shown using asterisks in the same Figure 3. Instead of calculating only one gradient per iteration, the algorithm evaluates a *bundle of gradients* at points *nearby* the current iterate, chosen by

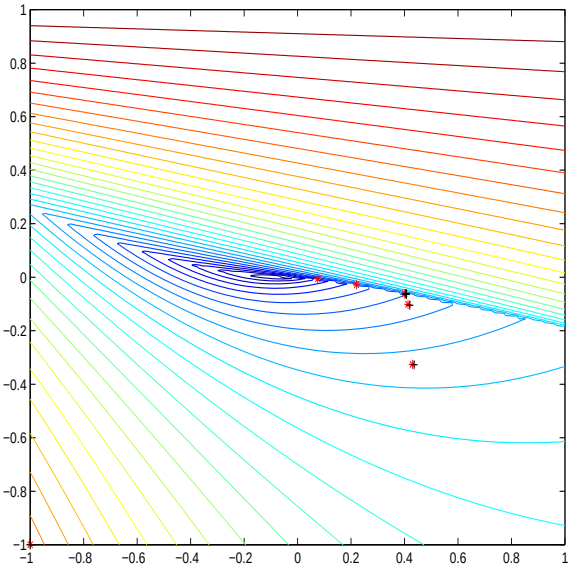

Figure 3: Gradient Bundle (*) and Steepest Descent (+)

sampling from a uniform distribution scaled by a parameter ϵ. The algorithm's progress is similar to the steepest descent method at first, because all the gradients in the bundle are qualitatively similar at first. However, when the iterates approaches the region where the steepest descent method jams, the bundle includes gradients from *both sides* of the manifold on which the spectral abscissa is not smooth. The algorithm is based on a generalized steepest descent method for convex optimization, using a search direction d with the property that $-d$ is the *convex combination of the gradients in the bundle whose 2-norm is minimized* (a convex quadratic subproblem that is easily solved, e.g., by an interior point method). Therefore, in the region where steepest descent jams, the bundle algorithm is able to use the information from both sides of the manifold to "turn the corner" and make progress towards the minimizer, as shown by the asterisks in Figure 3. Eventually, it finds an iterate at which 0 is in the convex hull of its gradient bundle, and it terminates. At this point, we would restart the minimization with a smaller value of ϵ.

We now formally define the random gradient bundle algorithm. It is assumed that f, the function to be minimized, is continuous, and continuously differentiable almost everywhere. When we write $\nabla f(x)$ below, we are implicitly assuming that f is differentiable at x. We restrict our iterates

x to the feasible region defined by

$$\|x\|_\infty \leq \Xi$$

for some moderately large number Ξ.

Algorithm 1 (Gradient Bundle)

0. (Initialization) Choose an initial feasible point $x \in \mathbf{R}^m$, an initial positive value for the sampling radius ϵ, a positive reduction factor $\theta < 1$, a positive integer N defining the number of gradients per bundle, and two positive integers M_1 and M_2 to terminate the iterations. Initialize $k = 1$.

1. (Inner Iteration) Carry out minimization k, controlled by the sampling radius ϵ, as follows. Initialize $j = 0$.

(a) Define a bundle G as the set of N gradients $\{\nabla f(y)\}$, where y takes on N values: the current iterate x, and $N-1$ other vectors differing from x by vectors whose entries are obtained by sampling from a uniform distribution on $[-\epsilon/2, \epsilon/2]$.

(b) Define the search direction

$$d = -\arg\min\{\|v\|_2 : v \in \text{conv } G\}. \tag{14}$$

If $d = 0$, go to Step 2.

(c) Use a line search (see below) to find a steplength satisfying

$$f(x + td) < f(x), \quad t \in (0, \bar{t}\,], \tag{15}$$

where $\bar{t} = \arg\max\{t : \|x + td\|_\infty \leq \Xi\}$.

(d) Replace x by $x + td$. If $t = \bar{t}$, terminate. If $j < M_2$, increment j and return to Step (a).

2. (Decrease Sampling Parameter) If $k < M_1$, replace ϵ by the smaller value $\theta\epsilon$, increment k and return to Step 1. Otherwise terminate.

The condition imposed by the line search is simply a reduction in the value of the objection function. It is standard practice in line searches for smooth and convex optimization to impose a somewhat more demanding "sufficient decrease" condition. However, in the non-Lipschitz case, function values can be extremely sensitive to changes in the argument in the

neighborhood of a minimizer, and the imposition of any sufficient decrease condition often leads to failure in the line search.

The definition of d in (14) guarantees the descent condition

$$\nabla f(x)^T d < 0$$

as long as d is not zero, since $\nabla f(x)$ is one of the gradients in the bundle G. To see this, observe that if d is not 0, there is a separating hyperplane between 0 and conv G, so $-d$, the closest element of conv G to 0 in the 2-norm, must satisfy the condition $h^T(-d) > 0$ for every $h \in$ conv G, including $h = \nabla f(x)$. The descent condition ensures that (15) must hold for sufficiently small t.

We use a very simple line search:

Algorithm 2 (Line Search)

0. (Initialization) Set $s = 0$, $t = 1$. If (15) is violated, go to Step 2.

1. (Doubling Phase) Repeatedly do the following: replace s by the value of t and replace t by $\min(2t, \bar{t})$, until $t = \bar{t}$ or (15) is violated. In the latter case replace t by its previous value s. In both cases, terminate.

2. (Bisection Phase) Repeatedly replace t by $t/2$, until t satisfies (15).

In this simple line search, either a doubling process (Step 1) or a bisection process (Step 2) is executed. The doubling phase is initiated if the initial point satisfies (15) and it continues until a point is found violating (15), at which point the line search terminates with the previous point, or the bound on the step is reached. The bisection phase is initiated if the initial point violates (15) and it continues until a point is found satisfying (15). In exact arithmetic, it must terminate since (15) must hold for t sufficiently small. In practice, we place a limit of 50 on the number of bisection steps, and if this is exceeded, the inner loop in Algorithm 1 is terminated, with control passing to its Step 2.

Algorithm 1 terminates in Step 1(d) if an iterate x is generated with $\|x\|_\infty = \Xi$, the bound on the feasible region being considered. We also experimented with a version of the algorithm that adds bounds to the quadratic program solved in Step 1(b), in order to minimize f over the box defined by $\|x\|_\infty \leq \Xi$, but we do not discuss this here as we prefer to keep the algorithm description as simple as possible.

We implemented Algorithm 1 in MATLAB. If we were able to solve the quadratic program in Step 1(b) of Algorithm 1 exactly, no tolerance would

be needed when comparing d to 0. In practice, a tolerance is needed; we use the test $\|d\|_2 \leq \tau$, where τ is a positive parameter. We use MOSEK [Mos01] to solve the quadratic program. MOSEK implements an interior-point method and is available as a MATLAB toolbox. It is far more efficient than various other MATLAB-callable codes that we tried.

In Section 4, we report numerical results for minimizing the spectral abscissa by Algorithm 1.

3 A Newton Barrier Method

In this section, we briefly describe a Newton barrier method to solve (10)-(11). As in the previous section, we impose an *a priori* upper bound Ξ on the norm of the solution x. Since we are using a barrier method, the 2-norm is a convenient choice. Consider the barrier function $B_\mu : \mathbf{R} \times \mathbf{R}^m \times \mathbf{S}^n \to \mathbf{R}$ defined by

$$B_\mu(\gamma, x, P) = \gamma - \mu \log\det(I - P) - \mu \log\det(P - \delta I)$$
$$-\mu \log\det\Big(2\gamma P - \Big(PA(x) + A(x)^T P\Big)\Big) - \mu \log(\Xi^2 - x^T x).$$

Recall that δ is the fixed robustness parameter. The new parameter μ is a positive *barrier parameter*, and we interpret $\log\det(M)$ to have the value $-\infty$ when its symmetric matrix argument M is not positive definite. The three log determinant terms impose the constraints in (11), and the final log term imposes the bound on $\|x\|$. The use of log barrier functions to impose nonlinear inequality constraints goes back to the 1950's and was used extensively in the 1960's [FM68]. The approach fell out of favor in the 1970's but was revived in the 1980's and 1990's [Wri92], following the explosion of interest in interior point methods for linear and convex programming. The role of the log determinant function in the analysis of interior point methods for convex semidefinite programs is particularly well known, but its use in the implementation of barrier methods for problems with nonlinear semidefinite constraints is also quite well established; see e.g. [LM00, Rin96].

As the title of [FM68] suggests, the standard approach for solving nonlinear programs by barrier methods is to solve a sequence of unconstrained problems, i.e., minimize a sequence of barrier functions, ideally with monotonically decreasing values for the barrier parameter μ. As is also standard, we use Newton's method to solve these unconstrained problems. The barrier function $B_\mu(\gamma, x, P)$ is smooth but generally nonconvex when the robustness parameter δ is less than one. By definition, the barrier function must be locally convex at any minimizer of $B_\mu(\gamma, x, P)$, but the smaller the parameter

δ is, the smaller is the region around the solution where the barrier function is convex. Consequently, it is essential to implement Newton's method in a way that recognizes and deals with nonconvexity. Indeed, the barrier function $B_\mu(\gamma, x, P)$ makes an excellent class of test functions for Newton barrier codes.

The gradient and Hessian of $B_\mu(\gamma, x, P)$ are derived in Appendix A. Let us write $v = (\gamma, x^T, \mathrm{svec}(P)^T)^T \in \mathbf{R} \times \mathbf{R}^m \times \mathbf{R}^{n(n+1)/2} = \mathbf{R}^p$, where svec is the standard isometry from $\mathbf{S}^n$ to $\mathbf{R}^{n(n+1)/2}$ and $p = m + 1 + n(n+1)/2$. In order to implement Newton's method for minimizing $B_\mu(\gamma, x, P)$, we actually work in the space $\mathbf{R}^p$. We denote the associated barrier function in this space by $\hat{B}_\mu(v)$. The formula for $\nabla^2 \hat{B}_\mu(v)$ involves symmetrized Kronecker products [AHO98], [TTT98]. We omit the details, but we note that it is important to use symmetrized Kronecker products, not standard Kronecker products, since $P \in \mathbf{S}^n$, not $\mathbf{M}^n(\mathbf{R})$. If Newton's method is implemented in $\mathbf{R} \times \mathbf{R}^m \times \mathbf{R}^{n^2}$ instead of $\mathbf{R} \times \mathbf{R}^m \times \mathbf{R}^{n(n+1)/2}$, so that standard Kronecker products are used instead of symmetrized Kronecker products, the resulting Hessian of the barrier function in this space is necessarily singular. Our MATLAB m-file for computing $\nabla \hat{B}_\mu(v)$ and $\nabla^2 \hat{B}_\mu(v)$ is available at [Dat].

There are two standard approaches to implementing Newton's method for minimizing smooth nonconvex functions: trust region methods and modifed Cholesky factorization methods, the latter in conjunction with a line search. We chose to take neither approach, but to implement a simpler algorithm that may not have received much attention since the 1950's. We start by computing the standard Cholesky factorization of the Hessian matrix $\nabla^2 \hat{B}_\mu(v)$. If the factorization breaks down, i.e., a negative or zero pivot is encountered, we compute the spectral factorization of $\nabla^2 \hat{B}_\mu(v)$, and shift all the eigenvalues up by $\nu - \lambda_{\min}(\nabla^2 \hat{B}_\mu(v))$, where $\nu / \|\nabla^2 \hat{B}_\mu(v)\|$ is small. This yields an excellent modified Newton search direction, dominated by the eigenvector corresponding to $\lambda_{\min}(\nabla^2 \hat{B}_\mu(v))$. Computing the Cholesky *and* spectral factorization is in principle much more expensive than computing a modified Cholesky factorization such as described in [GMW81], but in MATLAB it is much more efficient than implementing the modified Cholesky algorithm as an m-file, and in any case, in our application, the cost is trivial, since forming the Hessian matrix is far more expensive than computing its eigenvalues.

We now summarize the Newton barrier method. We state it so that it is applicable to any family of barrier functions $\hat{B}_\mu(v)$. In our application, given an initial iterate x, we initialize γ to $\frac{1}{2}\lambda_{\max}(A(x) + A(x)^T) + 1$ and P to $\frac{1}{2}I$.

Algorithm 3 (Newton Barrier)

0. (Initialization) Choose an initial point v with $\hat{B}_\mu(v) < \infty$, a positive reduction factor $\theta < 1$, an increase factor $\tilde{\theta}$ satisfying $1 < \tilde{\theta} < 1/\theta$, a very small positive shift parameter ν, a termination tolerance τ, and two positive integers M_1 and M_2 to terminate the iterations. Initialize $k = 1$.

1. (Inner Iteration) Carry out minimization k, controlled by the barrier parameter μ, as follows. Initialize $j = 0$, and set $\hat{v}$ to v (i.e., save a copy of the starting point).

(a) Compute the gradient $\nabla \hat{B}_\mu(v)$. If $\|\nabla \hat{B}_\mu(v)\| \leq \tau$, go to Step 3. Compute the Hessian matrix $\nabla^2 \hat{B}_\mu(v)$ and its Cholesky factorization $\nabla^2 \hat{B}_\mu(v) = LL^T$, with L lower triangular, or, if this breaks down, the spectral factorization $\nabla^2 \hat{B}_\mu(v) = Q\Lambda Q^T$, with Λ diagonal and $Q^T Q = I$, and with the least diagonal entry in Λ denoted $\Lambda_{\min}$.

(b) Define the search direction

$$d = -L^{-T} L^{-1} \nabla \hat{B}_\mu(v), \tag{16}$$

or, if the Cholesky factorization broke down,

$$d = -Q \left(\Lambda + \sigma I\right)^{-1} Q^T \nabla \hat{B}_\mu(v), \tag{17}$$

where $\sigma = \nu \|\nabla^2 \hat{B}_\mu(v)\| - \Lambda_{\min}$.

(c) Use a line search to find a positive steplength satisfying

$$\hat{B}_\mu(x + td) < \hat{B}_\mu(x). \tag{18}$$

(d) Replace x by $x + td$. If $j < M_2$, increment j and return to Step (a).

2. (Increase Barrier Parameter) If $k < M_1$, replace μ by the larger value $\tilde{\theta}\mu$, discard the current iterate v and replace it by $\hat{v}$, increment k and return to Step 1. Otherwise terminate.

3. (Decrease Barrier Parameter) If $k < M_1$, replace μ by the smaller value $\theta\mu$, increment k and return to Step 1. Otherwise terminate.

For simplicity, we use the line search of the previous section (Algorithm 2), although it would be more standard to impose a sufficient decrease condition since the barrier function is smooth.

Note the parallel inner-outer iteration structure of Algorithms 1 and 3. The outer iteration of the former reduces the sampling parameter ϵ, while the outer iteration of the latter reduces the barrier parameter μ. A key difference concerns the behavior when the maximum number of iterates allowed for the inner iteration is reached. Algorithm 1 proceeds to reduce ϵ anyway, since the spectral abscissa is monotonically decreasing throughout the algorithm. There is no corresponding monotonicity for Algorithm 3 to rely on, so if an inner iteration fails to reduce $\|\hat{B}_\mu(v)\|$ to the desired level, the algorithm backtracks, trying a less aggressive choice of μ. An alternative strategy would be to check whether γ, the bound on the robust spectral abscissa and the first component of v, is lower than the corresponding first component of the inner iteration starting point $\hat{v}$.

4 Numerical Results

We have expermented with various affine matrix families $A(x)$, applying Algorithm 1 (Gradient Bundle) to find local minimizers of the spectral abscissa $\alpha(A(x))$ and Algorithm 3 (Newton Barrier) to find local minimizers of the robust spectral abscissa $\alpha_\delta(A(x))$. We focus our attention on small values of n, the matrix dimension, and m, the number of variables, because, as we shall see, even small problems are quite challenging. The reason for this is that, for most of our examples, local minimizers of the spectral abscissa are associated with at least one active nonderogatory multiple eigenvalue (see the relevant discussion in Section 1). Suppose that the optimal matrix has a nonderogatory multiple eigenvalue[2] with multiplicity $k > 2$. Then, as is well known (see e.g. [MBO97]), perturbing the matrix by noise with norm β typically increases the spectral abscissa by an amount whose magnitude is of order $\beta^{1/k}$. If β is the machine precision for the IEEE double format arithmetic used by MATLAB and $k = 5$, the spectral abscissa is perturbed by approximately 0.001. Thus, the optimal value of the spectral abscissa is extremely sensitive to even very small perturbations.

Before presenting the results, we specify the parameters defining the algorithms. For both algorithms, we initialized x randomly, drawing each component from a normal distribution with mean 0 and variance 1, and we

[2]To see why we exclude $k = 2$, see (1). In this case, it may be only the imaginary part that increases rapidly, depending on the perturbation.

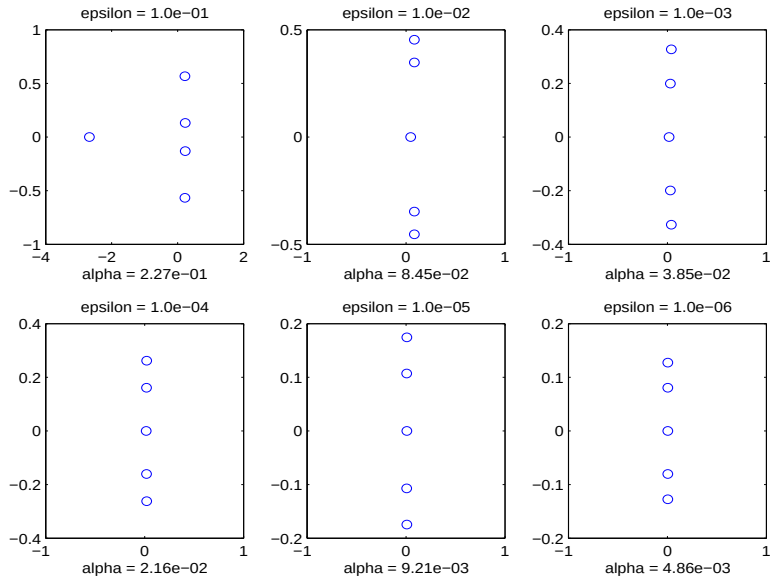

Figure 4: Example OneBlock, Gradient Bundle

set Ξ, the bound on $\|x\|$, to 1000, the tolerance τ to 10^{-6}, and the outer and inner iteration limits M_1 and M_2 to 6 and 100 respectively. For Gradient Bundle, we set the initial value for the sampling radius ϵ to 0.1, its reduction factor θ to 0.1, and N, the number of gradients per bundle, to $2m$ (twice the number of variables). For Newton Barrier, we set the initial value for the barrier parameter μ to 0.1, its reduction factor θ to 0.1, its increase factor $\tilde{\theta}$ to 3, and the Hessian shift parameter ν to 10^{-14}.

We present the examples one at a time, illustrating various points as we proceed. All figures show eigenvalues as points in the complex plane. When we refer to multiple eigenvalues below, it may be assumed that they are nonderogatory except as noted.

Example OneBlock

Our first example is (2), which we repeat here for convenience. Given n, define $m = n - 1$, and

$$A(x) = \begin{bmatrix} -x_1 & 1 & 0 & \cdot & \cdot & 0 \\ x_1 & 0 & 1 & 0 & \cdot & 0 \\ x_2 & 0 & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & 1 \\ x_{n-1} & 0 & \cdot & \cdot & \cdot & 0 \end{bmatrix}.$$

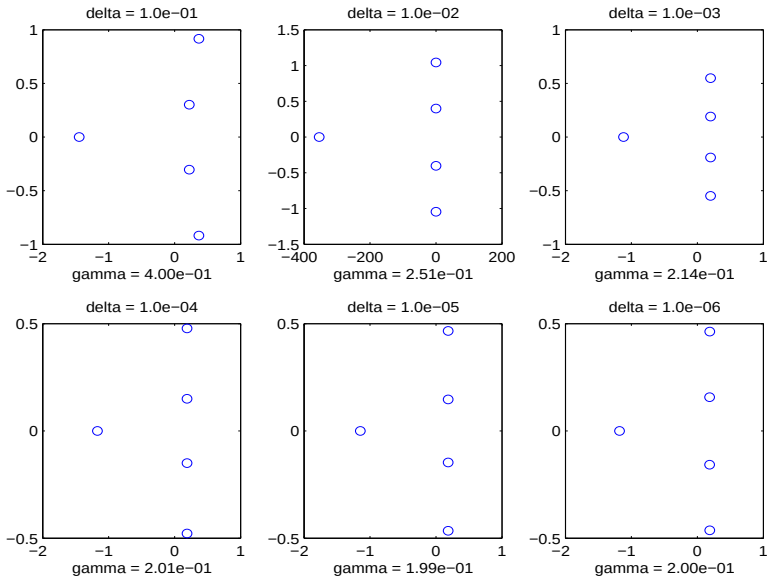

Figure 5: Example OneBlock, Newton Barrier

This example is constructed so that the global minimizer is $x = 0$, with $A(x)$ having an active multiple eigenvalue 0 with multiplicity n [BLO01c]. We used $n = 5$ for our experiments. Figure 4 shows the results of a typical run of Gradient Bundle on this problem. The 6 subplots in Figure 4 show the eigenvalues of $A(x)$ for x generated at the end of *each* of the inner iterations of Gradient Bundle, i.e., for the sampling radius $\epsilon = 10^{-1}$ through $\epsilon = 10^{-6}$, respectively. Each subplot also displays the final value of the spectral abscissa α that was found. Note the steady reduction in the value of α obtained as ϵ is reduced. Note also the reduction in the size of the *imaginary* parts of the eigenvalues, showing that the eigenvalues are moving, with respect to ϵ, towards the limiting multiple zero eigenvalue. The results for Gradient Bundle did not vary much with different runs, although every run starts from a different initial point and generates different bundles of random gradients. Some runs terminated with iterates on the boundary of the feasible region, i.e., with $\|x\|_\infty = 1000$. This indicates the existence of recession directions for α, so that when constrained by $\|x\|_\infty \leq 1000$, there very likely are one or more local minimizers of α on the boundary, although we know there are no unconstrained local minimizers besides 0 and that 0 is a global minimizer. In order to find constrained minimizers, the algorithm would have to be modified, as mentioned earlier.

Figure 5 shows results for 6 runs of Newton Barrier, for values of the robustness parameter δ ranging from 10^{-1} to 10^{-6}, respectively. Each one of these runs requires an outer iteration to vary the barrier parameter μ, and each of these requires an inner iteration to minimize the barrier function, so the amount of work required to generate Figure 5 is at least an order of magnitude more than the work required to generate Figure 4; actually, it is much more than this, because the time required for one Newton step is substantially more than the time required for one bundle step, as more variables are involved. Despite this huge increase in computational work, the information displayed by Figure 5 is rather unimpressive compared with that in Figure 4. In fact, all subplots of Figure 5 show eigenvalues that are strikingly similar to the first plot in Figure 4, which is trivial to compute by comparison.[3] The reason that the results for the smaller values of δ do not show better convergence to the multiple zero eigenvalue is that the barrier functions are so ill-conditioned that the Newton method is unable to minimize them to the specified accuracy. The values of γ shown are the best of the bounds on the robust spectral abscissa computed by Newton Barrier, not necessarily the final bound, as the minimization of the barrier function for the final barrier parameter μ may fail.

Example TwoBlock

Given n_1 and n_2, define $m = n_1 + n_2 - 1$, $n = n_1 + n_2$, and

$$A(x) = \begin{bmatrix} -x_1 & 1 & 0 & \cdot & \cdot & 0 & x_{n_1} & x_{n_1+1} & \cdot & \cdot & \cdot & x_{n_1+n_2-1} \\ x_1 & 0 & 1 & 0 & \cdot & 0 & 0 & 0 & \cdot & \cdot & \cdot & 0 \\ x_2 & 0 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & 0 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & 1 & 0 & \cdot & \cdot & \cdot & \cdot & 0 \\ x_{n_1-1} & 0 & \cdot & \cdot & \cdot & 0 & 0 & \cdot & \cdot & \cdot & \cdot & 0 \\ x_{n_1} & 0 & \cdot & \cdot & \cdot & 0 & -1 & 0 & \cdot & \cdot & \cdot & 0 \\ x_{n_1+1} & 0 & \cdot & \cdot & \cdot & 0 & 0 & -1 & \cdot & \cdot & \cdot & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 0 \\ x_{n_1+n_2-1} & 0 & \cdot & \cdot & \cdot & 0 & 0 & 0 & \cdot & \cdot & 0 & -1 \end{bmatrix}.$$

This example is constructed so that the global minimizer is $x = 0$, with $A(0)$ having an active (nonderogatory) multiple eigenvalue 0, with multiplicity n_1, and an inactive semisimple eigenvalue -1, with multiplicity n_2.

[3] There is one exception; the anomalous subplot for $\delta = 10^{-2}$ corresponds to an iterate x with large norm, again demonstrating the probable presence of a constrained local minimizer at the boundary of the feasible region.

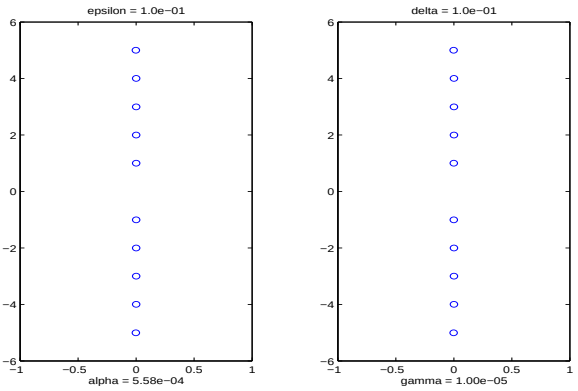

Figure 6: Example SimpleEigs, Bundle and Barrier

Example OneBlock is the special case $n_2 = 0$. We used $n_1 = 5$, $n_2 = 5$ for our experiments. The results are very similar to the results for Example OneBlock, except for the presence of the additional eigenvalue -1, so we do not display them.

Example SimpleEigs

Given an even integer n, let $m = n/2 - 1$, with

$$A(x) = \begin{bmatrix} x_1 & 1 & 0 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 0 \\ -1 & x_1 & 0 & 0 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & x_2 & 2 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & 0 & -2 & x_2 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & x_m & m & 0 & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & -m & x_m & 0 & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 0 & 0 & y & m+1 \\ 0 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 0 & -m-1 & y \end{bmatrix}.$$

where $y = -\sum_{k=1}^{m} x_k$. This example is constructed so that the global minimizer is $x = 0$, with $A(0)$ having $m + 1$ simple complex conjugate pairs of eigenvalues, $\pm k\, i$, $k = 1, \ldots, m$, all of them active, and no real eigenvalues. We used $n = 10$. Because all the active eigenvalues are simple, this problem is very easily solved to high accuracy. Figure 6 shows the eigenvalues computed in just one inner iteration of Gradient Bundle with sampling radius $\epsilon = 0.1$ and one run of Newton Barrier (6 inner iterations) with $\delta = 0.1$. Even at these course levels of accuracy, the results accurately identify the optimal eigenvalues.

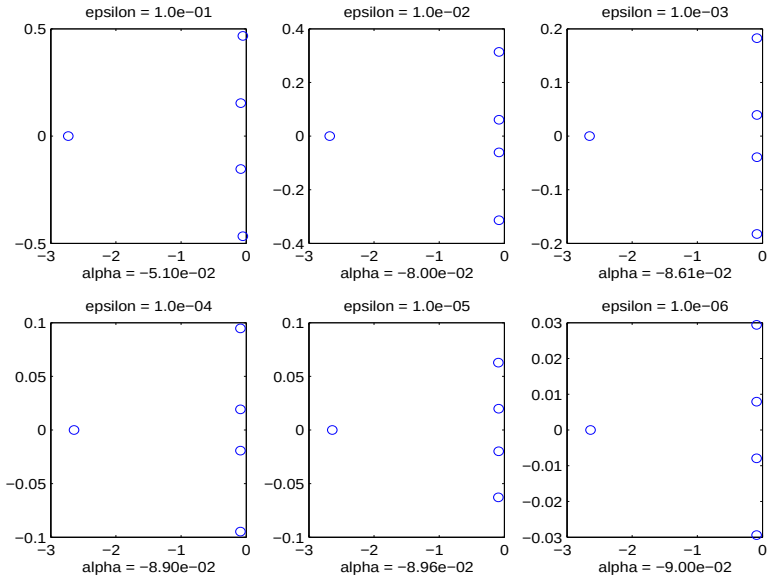

Figure 7: Example PolShc(b), Gradient Bundle

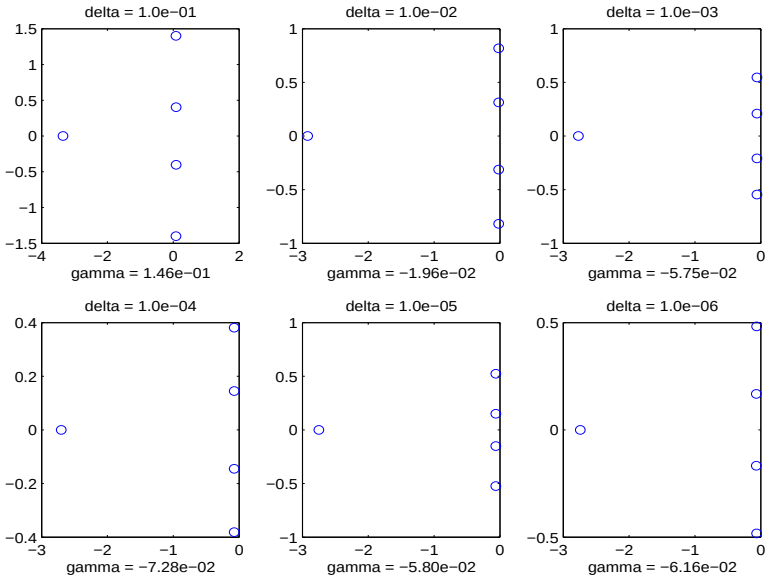

Figure 8: Example PolShc(b), Newton Barrier

Example PolShc

This consists of three specific small examples from [PS99].

(a) $n = 3$, $m = 2$, with

$$A(x) = \begin{bmatrix} -x_1 & 1 & 0 \\ 13 + 5x_1 - x_2 & 0 & 1 \\ -x_2 & 0 & 0 \end{bmatrix}.$$

(b) $n = 5$, $m = 3$, with

$$A(x) = \begin{bmatrix} -3 & 1 & 0 & 0 & 0 \\ -1 & 0 & 1 & 0 & 0 \\ 3 - x_2 & 0 & 0 & 1 & 0 \\ 2 - x_1 & 0 & 0 & 0 & 1 \\ -x_3 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

(c) $n = 3$, $m = 3$, with

$$A(x) = \begin{bmatrix} 2 & 2 & 0 \\ -1 & 0 & 0 \\ -1 & 0 & 2 \end{bmatrix} + x_1 \begin{bmatrix} -.4582 & .4027 & .9691 \\ .737 & -.4511 & -.3452 \\ -.7406 & .816 & .7331 \end{bmatrix}$$

$$+x_2 \begin{bmatrix} -.589 & .5471 & -.6725 \\ .1761 & .5744 & .4972 \\ -.7262 & -.4928 & -.8219 \end{bmatrix} + x_3 \begin{bmatrix} -.9571 & -.3868 & -.1505 \\ .4578 & -.656 & .3161 \\ -.1786 & .4769 & .5364 \end{bmatrix}.$$

Apparently Examples PolShc(a) and PolShc(b) have only one local minimizer of $\alpha(A(x))$. In the case of Example PolShc(a), many runs of Gradient Bundle from different starting points led to the same stable minimizer, approximately $x = [17.73,\ 206.4]^T$ with $\alpha(A(x)) = -5.909$, corresponding to an active triple eigenvalue (and no inactive eigenvalues). Many runs also led to the boundary, indicating the probable presence of local constrained minimizers with higher objective values (see the discussion for Example OneBlock).

In the case of Example PolShc(b), all runs of Gradient Bundle led to the same stable minimizer, approximately $x = [2.008,\ 3.135,\ 0.0002]^T$, with $\alpha(A(x)) = -0.0900$, corresponding to an active quadruple eigenvalue (and one inactive simple eigenvalue). Figures 7 and 8 show the results for Example PolShc(b) for Gradient Bundle (at the end of each inner iteration) and

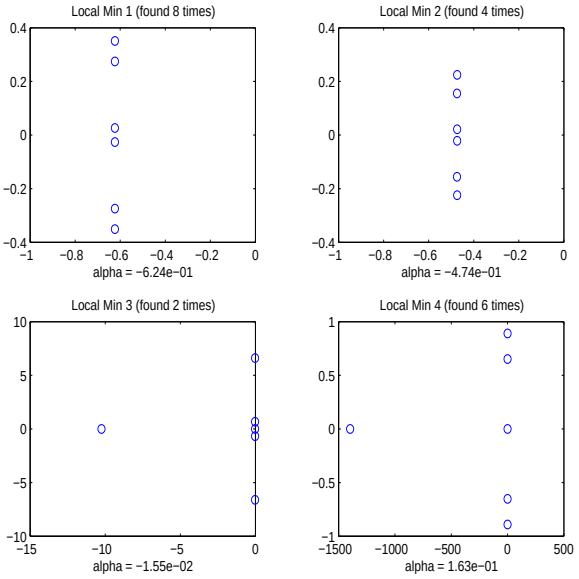

Figure 9: Example ABKC0, Gradient Bundle

Newton Barrier (for various choices of the robustness parameter δ), respectively (see the discussion for Figure 4 and 5). As in Example OneBlock, ill-conditioning of the barrier functions prevented Newton Barrier from obtaining accurate robust spectral abscissa bounds γ for the two smallest values of δ.

In Example PolShc(c), the spectral abscissa is apparently unbounded below; almost all runs of Gradient Bundle generated stable iterates with $\|x\|_\infty = \Xi$ and every run of Newton Barrier generated stable iterates with $\|x\|_2$ nearly equal to Ξ. Some runs of Gradient Bundle converged to an unstable local minimizer, $x = [-1.45\ 0.42\ 1.75]^T$, with $\alpha(A(x)) = 0.672$, corresponding to an active triple eigenvalue.

In the results reported in [PS99], no attempt was made to optimize stability. The technique discussed there generates marginally stable matrices, e.g., for Example PolShc(b), with $\alpha(A(x)) = -0.01$.

Example ABKC0

Given an $n \times n$ matrix $\bar{A}$, an $n \times r$ matrix $\bar{B}$ and an $s \times n$ matrix $\bar{C}$, let $m = rs - 1$ and define the affine family $A(x)$ by $\bar{A} + \bar{B}\,\mathrm{mat}(x)\,\bar{C}$, where mat is the natural isometry from $\mathbf{R}^m$ to the space of r by s matrices with lower right entry fixed to zero. We fix one entry to zero because, without this restriction, we find that randomly generated matrices $\bar{A}$, $\bar{B}$ and $\bar{C}$ lead

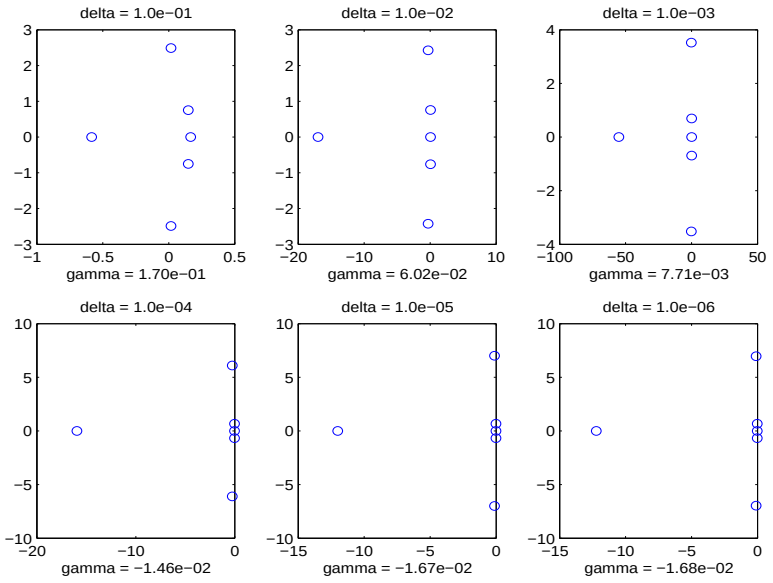

Figure 10: Example ABKC0, Newton Barrier

invariably to problems where the first step of Gradient Bundle generates an iterate x on the boundary of the feasible region. Though the restriction is artificial, it serves the purpose of generating some problems with interior local minimizers and therefore allowing us to keep the discussion of Gradient Bundle as simple as possible. We set $n = 6$, $r = 3$ and $s = 2$, and generated the matrices $\bar{A}$, $\bar{B}$ and $\bar{C}$ randomly; the matrix data are available from [Dat].

Figure 9 is different from any of the figures shown up to this point. It shows the eigenvalues associated with four different local minimizers of $\alpha(A(x))$ for this particular affine family, obtained by running Gradient Bundle from various different starting points. We made 20 runs of Gradient Bundle (6 inner iterations for each run). The subplots are sorted by the spectral abscissa value found and indicate how many times each local minimizer was found. Although Gradient Bundle found four different local minimizers of $\alpha(A(x))$, Newton Barrier, started from the same 20 starting points, found only one local minimizer of $\alpha_\delta(A(x))$, this minimizer depending of course on the value of the robustness parameter δ. Figure 10 shows this minimizer for each δ. In contrast to the previous results, Figure 10 shows steady reduction in the robust spectral abscissa bound γ as δ is decreased. Most likely this is because the active optimal eigenvalues consist of one triple real eigenvalue and one simple conjugate pair, in comparison with multiplicity

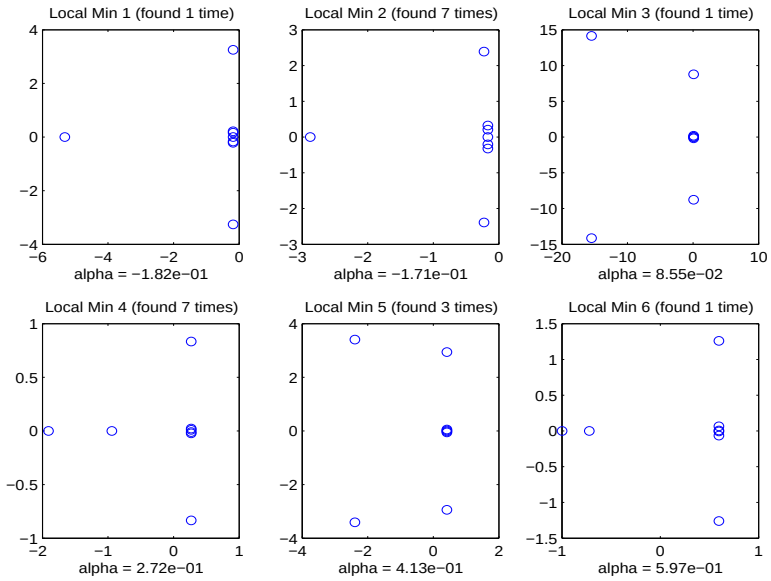

Figure 11: Example Random(a), Gradient Bundle

5 and 4 in Examples OneBlock and PolShc(b) respectively. Comparison of the eigenvalues and the final iterates x shows that the Newton Barrier minimizer converges, as δ decreases, to Local Minimizer 3 found by Gradient Bundle. Somewhat surprisingly, Newton Barrier found a slightly better bound on the robust spectral abscissa than Gradient Bundle did for this run. However, Gradient Bundle found other much better, i.e., more stable, minimizers (Local Minimizers 1 and 2).

Example Random

Given n and m, we randomly generated the matrices A_k, $k = 0, \ldots, m$. The randomly generated data may be found at [Dat].

(a) The first of our two random examples has $m = 5$ and $n = 8$; the randomly generated matrices are dense. As in the previous example, Figure 11 shows various different minimizers found by 20 different runs of Gradient Bundle. This time, 6 different local minimizers were found. Again as in the previous example, Newton Barrier converged to the same minimizer for every one of the 20 starting points. As the robustness parameter δ decreases, this minimizer converges to Gradient Bundle's Local Minimizer 4, with an active quadruple real eigenvalue and an active simple conjugate pair. However, this minimizer is unsta-

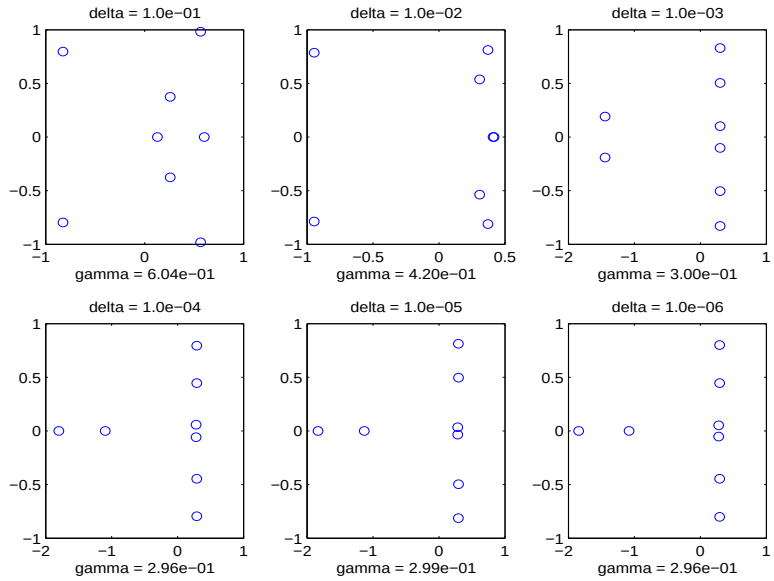

Figure 12: Example Random(a), Newton Barrier

ble; Newton Barrier was unable to find a stable matrix. In contrast, Gradient Bundle found two stable local minimizers.

(b) The second of our two random examples has $n = m = 25$; the randomly generated matrices are sparse. Figure 13 shows the eigenvalues generated by 20 different runs of Gradient Bundle. The subplots are ordered by spectral abscissa; the first seven minimizers are stable. We made no attempt to determine whether there are 20 distinct associated local minimizers, but given the multitude of solutions for the much smaller example Random(a), this seems likely. This example is much too large for Newton Barrier to run in a reasonable amount of time, but Gradient Bundle had no difficulties with it.

5 Conclusions

We have presented two methods for optimizing matrix stability. Gradient Bundle is a novel algorithm, inspired by bundle methods for convex optimization, but applicable to non-Lipschitz problems. We believe it offers great versatility and reliability at a very reasonable cost. It demonstrates beyond doubt that spectral abscissa minimizers are typically associated with

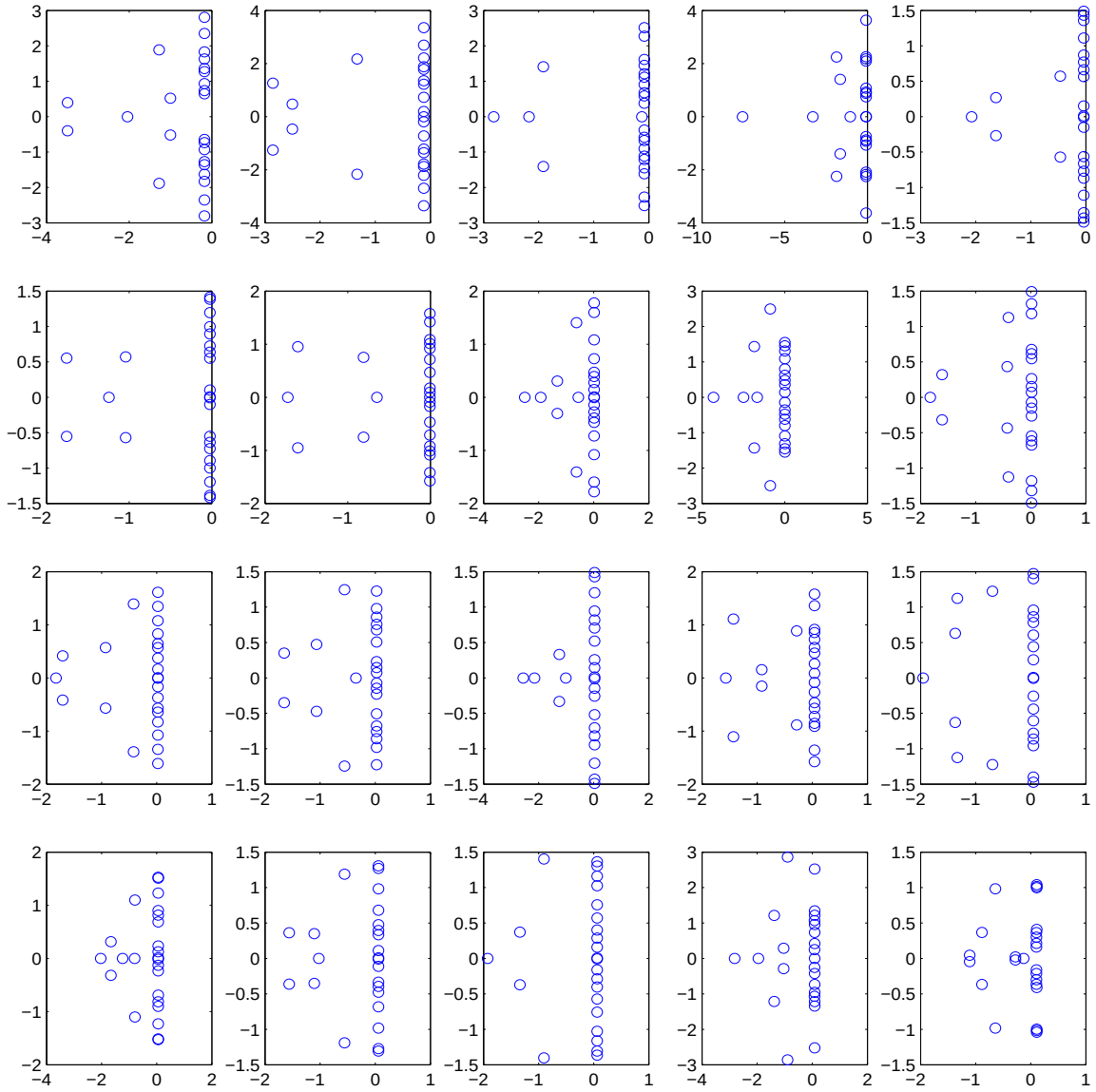

Figure 13: Example Random(b), Gradient Bundle

multiple eigenvalues, and that they can nonetheless be approximated quite accurately. Furthermore, there are advantages to *not* carrying out the minimization too accurately, as then the algorithm will most likely return a solution without unacceptable associated transient peaks. We hope that Gradient Bundle will become a useful tool.

Newton Barrier is a well known approach to smooth, nonconvex optimization with inequality constraints, including nonlinear semidefinite constraints. We used Newton Barrier to carry out robust spectral abscissa minimization, demonstrating that the minimizers indeed, as expected, converge to spectral abscissa minimizers as the robustness parameter δ is decreased. The motivation is that by ensuring that δ is not too small, unacceptable transient peaks can be avoided. Although an interesting concept, the approach is too expensive to be practical for any but the smallest problems. Furthermore, if δ is chosen too large, it may be impossible to achieve asymptotic stability, while if δ is chosen too small, ill conditioning may prevent convergence. We think Gradient Bundle offers the best of both worlds, as one may continue to reduce its sampling radius ϵ until asymptotic stability is achieved, but avoid making it so small that unacceptable transient peaks are incurred.

Although optimizing stability is a very attractive approach to stabilization, inherent difficulties remain. The greatest stumbling block is that there may be so many local minimizers that finding any that are stable remains very difficult.

A The Barrier Gradient and Hessian

In addition to our function $A(\cdot)$, we define functions of the variable

$$\begin{pmatrix} \gamma \\ x \\ P \end{pmatrix} \in \mathbf{R} \times \mathbf{R}^m \times \mathbf{S}^n$$

(which we write for convenience (γ, x, P)) by

$$\begin{aligned} \Lambda &= (I - P)^{-1} \\ \Delta &= (P - \delta I)^{-1} \\ Q &= (2\gamma P - PA - A^T P)^{-1} \\ \beta &= (\Xi^2 - x^T x)^{-1}, \end{aligned}$$

so the barrier function is

$$B_\mu = \gamma - \mu(\log\det\Lambda^{-1} + \log\det\Delta^{-1} + \log\det Q^{-1} + \log\det\beta^{-1}).$$

We also define functions, for $k = 1, 2, \ldots, m$,

$$\begin{aligned} P_k &= PA_k + A_k^T P \\ Q_k &= A_k Q + Q A_k^T. \end{aligned}$$

A straightforward application of the chain rule and the fact that the function $X \in \mathbf{S}^n \mapsto \log\det X$ has gradient X^{-1} shows that the gradient of the barrier function is given by

$$\nabla B_\mu = \begin{pmatrix} D_\gamma B_\mu \\ \nabla_x B_\mu \\ \nabla_P B_\mu \end{pmatrix} = \begin{pmatrix} 1 - 2\mu \mathrm{tr}(PQ) \\ \mu(\mathrm{tr}(P_k Q))_1^m + 2\mu\beta x \\ \mu(\Lambda - \Delta - 2\gamma Q + AQ + QA^T) \end{pmatrix}$$

Consider a base point $(\bar\gamma, \bar x, \bar P)$: we indicate the value of any of the above functions at this point by a bar. In particular, the gradient of the barrier at this point is just

$$\begin{pmatrix} 1 - 2\mu \mathrm{tr}(\bar P \bar Q) \\ \mu(\mathrm{tr}(\bar P_k \bar Q))_1^m + 2\mu\bar\beta \bar x \\ \mu(\bar\Lambda - \bar\Delta - 2\bar\gamma \bar Q + \bar A \bar Q + \bar Q \bar A^T) \end{pmatrix}$$

Now consider a nearby point $(\bar\gamma, \bar x, \bar P) + (d\gamma, dx, dP)$. Expanding up to first order shows that at this point the various functions above have values

$$\begin{aligned} \Lambda &= \bar\Lambda + \bar\Lambda dP\, \bar\Lambda + o(\cdot) \\ \Delta &= \bar\Delta - \bar\Delta dP\, \bar\Delta + o(\cdot) \\ Q &= \bar Q - \mathcal{D} + o(\cdot) \\ \beta &= \bar\beta + 2\bar\beta^2 \bar x^T dx + o(\cdot) \end{aligned}$$

where $o(\cdot)$ denotes terms of small order with respect to the size of the perturbation $(d\gamma, dx, dP)$, and

$$\mathcal{D} = \bar Q \Big(2\bar\gamma dP + 2 d\gamma\, \bar P - dP\, \bar A - \bar A^T dP - \sum_k \bar P_k dx_k \Big) \bar Q.$$

The Hessian of the barrier function B_μ at the point $(\bar\gamma, \bar x, \bar P)$ is the self-adjoint linear operator

$$\bar H : \mathbf{R} \times \mathbf{R}^m \times \mathbf{S}^n \to \mathbf{R} \times \mathbf{R}^m \times \mathbf{S}^n$$

with the property that

$$\bar H \begin{pmatrix} d\gamma \\ dx \\ dP \end{pmatrix} = \nabla B_\mu \left(\begin{pmatrix} \bar\gamma \\ \bar x \\ \bar P \end{pmatrix} + \begin{pmatrix} d\gamma \\ dx \\ dP \end{pmatrix} \right) - \nabla B_\mu \begin{pmatrix} \bar\gamma \\ \bar x \\ \bar P \end{pmatrix} + o(\cdot).$$

It is then a routine calculation to check that the Hessian is defined by

$$\bar{H}\begin{pmatrix} d\gamma \\ dx \\ dP \end{pmatrix} = \mu \begin{pmatrix} -2\mathrm{tr}(\bar{P}\mathcal{D} + \bar{Q}dP) \\ (\mathrm{tr}(\bar{P}_k\mathcal{D} + \bar{Q}_k dP))_1^m + 2\bar{\beta}\,dx + 4\bar{\beta}^3\bar{x}\bar{x}^T dx \\ \bar{\Lambda}dP\,\bar{\Lambda} + \bar{\Delta}dP\,\bar{\Delta} - 2\bar{Q}d\gamma - 2\bar{\gamma}\mathcal{D} - \sum_k \bar{Q}_k dx_k - \bar{A}\mathcal{D} - \mathcal{D}\bar{A}^T \end{pmatrix}.$$

References

[AHO98] F. Alizadeh, J.-P. A. Haeberly, and M. L. Overton. Primal-dual interior-point methods for semidefinite programming: convergence rates, stability, and numerical results. *SIAM Journal on Optimization*, 8:746–768, 1998.

[Arn71] V.I. Arnold. On matrices depending on parameters. *Russian Mathematical Surveys*, 26:29–43, 1971.

[BGFB94] S. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan. *Linear Matrix Inequalities in System and Control Theory*. SIAM, 1994.

[BGL95] V. Blondel, M. Gevers, and A. Lindquist. Survey on the state of systems and control. *European Journal of Control*, 1:5–23, 1995.

[BLO01a] J.V. Burke, A.S. Lewis, and M.L. Overton. Approximating subdifferentials by random sampling of gradients. 2001. Submitted to Math. Oper. Res.

[BLO01b] J.V. Burke, A.S. Lewis, and M.L. Overton. Optimal stability and eigenvalue multiplicity. *Foundations of Computational Mathematics*, 1, 2001. DOI: 10.1007/s102080010008.

[BLO01c] J.V. Burke, A.S. Lewis, and M.L. Overton. Optimizing matrix stability. *Proceedings of the American Mathematical Society*, 2001. To appear.

[BO01] J.V. Burke and M.L. Overton. Variational analysis of non-Lipschitz spectral functions. *Mathematical Programming*, 2001. To appear.

[BT97] V. Blondel and J.N. Tsitsiklis. NP-Hardness of some linear control design problems. *SIAM Journal on Control and Optimization*, 35:2118–2127, 1997.

[Dat] Data for the example problems. To be placed on the web.

[FM68] A.V. Fiacco and G.P. McCormick. *Nonlinear Programming: Sequential Unconstrained Minimization Techniques*. John Wiley, New York, 1968. Republished by SIAM (Philadelphia), 1990.

[GMW81] P.E. Gill, W. Murray, and M.H. Wright. *Practical Optimization*. Academic Press, New York and London, 1981.

[GN00] L. El Ghaoui and S.-I. Niculescu, editors. *Recent Advances in Linear Matrix Inequality Methods for Control*. SIAM, Philadelphia, 2000.

[HHB98] A. Hassibi, J. P. How, and S. P. Boyd. Low-authority controller design via convex optimization. In *Proceedings of IEEE Conference on Decision and Control*, Tampa, FL, December 1998.

[HUL93] J.B. Hiriart-Urruty and C. Lemarechal. *Convex analysis and minimization algorithms*. Springer-Verlag, New York, 1993. Two volumes.

[Kiw85] K.C. Kiwiel. *Methods of descent for nondifferentiable optimization*. Lecture notes in mathematics 1133. Springer-Verlag, Berlin and New York, 1985.

[LM00] F. Leibfritz and E.M.E. Mostafa. An interior point constrained trust region method for a special class of nonlinear semi-definite programming problems. Technical report, Universität Trier, Germany, July 2000.

[LO96] A.S. Lewis and M.L. Overton. Eigenvalue optimization. *Acta Numerica*, pages 149–190, 1996.

[MBO97] J. Moro, J.V. Burke, and M.L. Overton. On the Lidskii-Vishik-Lyusternik perturbation theory for eigenvalues of matrices with arbitrary Jordan structure. *SIAM Journal on Matrix Analysis and Applications*, 18:793–817, 1997.

[Mos01] Mosek optimization software, 2001. http://www.mosek.com/.

[Nem93] A. Nemirovskii. Several NP-hard problems arising in robust stability analysis. *Math. Control Signals Systems*, 6:99–105, 1993.

[NN94] Y. Nesterov and A. Nemirovskii. *Interior Point Polynomial Algorithms in Convex Programming*. SIAM, Philadelphia, 1994.

[PS99] B.T. Polyak and P.S. Shcherbakov. Numerical search of stable or unstable element in matrix or polynomial families: a unified approach to robustness analysis and stabilization. In A.Garulli andd A.Tesi and A.Vicino, editors, *Robustness in Identification and Control*, Lect. Notes Control and Inform. Sci. 245, pages 344–358. Springer, London, 1999.

[Rin96] U.T. Ringertz. Eigenvalues in optimum structural design. In A.R. Conn, L.T. Biegler, T. F. Coleman, and F. Santosa, editors, *Proceedings of an IMA workshop on Large-Scale Optimization*, 1996.

[RW98] R.T. Rockafellar and R.J.B. Wets. *Variational Analysis*. Springer-Verlag, New York, 1998.

[Tod01] M.J. Todd. Semidefinite programming. *Acta Numerica*, 10, 2001. to appear.

[Tre97] L.N. Trefethen. Pseudospectra of linear operators. *SIAM Review*, 39:383–406, 1997.

[TT99] K.-C. Toh and L.N. Trefethen. The Kreiss matrix theorem on a general complex domain. *SIAM J. Matrix Anal. Appl.*, 21:145–165, 1999.

[TTT98] M.J. Todd, K.C. Toh, and R.H. Tütüncü. On the Nesterov-Todd direction in semidefinite programming. *SIAM Journal on Optimization*, 8:769–796, 1998.

[Wri92] M.H. Wright. Interior methods for constrained optimization. *Acta Numerica*, 1:341–407, 1992.