RIACS

Research Institute for Advanced Computer Science
NASA Ames Research Center

On Adaptive Weighted Polynomial Preconditioning for Hermitian Positive Definite Matrices

Bernd Fischer and Roland W. Freund

RIACS Technical Report 92.09

March 1992

To appear in the Proceedings of the Copper Mountain Conference on Iterative Methods, 1992

On Adaptive Weighted Polynomial Preconditioning for Hermitian Positive Definite Matrices

Bernd Fischer and Roland W. Freund

The Research Institute for Advanced Computer Science is operated by
Universities Space Research Association (USRA),
The American City Building, Suite 311, Columbia, MD 21044, (301)730–2656.

Work reported herein was supported in part by Cooperative Agreement NCC 2–387 between NASA and USRA.

ON ADAPTIVE WEIGHTED POLYNOMIAL PRECONDITIONING FOR HERMITIAN POSITIVE DEFINITE MATRICES

BERND FISCHER* AND ROLAND W. FREUND†

Abstract. The conjugate gradient algorithm for solving Hermitian positive definite linear systems is usually combined with preconditioning in order to speed up convergence. In recent years, there has been a revival of polynomial preconditioning, motivated by the attractive features of the method on modern architectures. Standard techniques for choosing the preconditioning polynomial are based only on bounds for the extreme eigenvalues. Here a different approach is proposed, which aims at adapting the preconditioner to the eigenvalue distribution of the coefficient matrix. The technique is based on the observation that good estimates for the eigenvalue distribution can be derived after only a few steps of the Lanczos process. This information is then used to construct a weight function for a suitable Chebyshev approximation problem. The solution of this problem yields the polynomial preconditioner. In particular, we investigate the use of Bernstein-Szegö weights.

Key words. linear systems, Hermitian positive definite matrices, conjugate gradient algorithm, polynomial preconditioning, Chebyshev approximation problem, Bernstein-Szegö weights

AMS(MOS) subject classifications. 65F10

1. Introduction.

One of the most powerful iterative schemes for solving Hermitian positive definite linear systems

$$Ax = b \tag{1.1}$$

is the conjugate gradient algorithm (CG) of Hestenes and Stiefel [13], especially when it is combined with preconditioning [4]. In recent years, there has been much interest in polynomial preconditioning. The basic idea is as follows: instead of solving the original system (1.1) by CG, the CG iteration is applied either to

$$\psi(A)Ax = \psi(A)b \tag{1.2}$$

(left preconditioning), or to

$$A\psi(A)y = b, \quad x = \psi(A)y \tag{1.3}$$

(right preconditioning). Here ψ is a suitably chosen polynomial of small degree. Moreover, it is required that none of the zeros of ψ coincides with an eigenvalue of A. This guarantees that the preconditioned systems (1.2) and (1.3) are both equivalent to (1.1).

Polynomial preconditioning goes back to the 1950s. It seems that Lanczos [17] was the first to mention the idea; interestingly, his paper is never referenced. Stiefel [23] used polynomial preconditioning techniques to accelerate eigenvalue computations. Rutishauser [20]

* Institut für Angewandte Mathematik, Universität Hamburg, Bundesstrasse 55, D–2000 Hamburg 13, Federal Republic of Germany.

† RIACS, Mail Stop T041–5, NASA Ames Research Center, Moffett Field, CA 94035, The research of this author was supported by Cooperative Agreement NCC 2-387 between the National Aeronautics and Space Administration (NASA) and the Universities Space Research Association (USRA).

proposed an inner-outer iteration process, with CG as the outer iteration and the Chebyshev semi-iterative method [12] as the inner recursion. The motivation for his approach was to reduce roundoff in the classical CG algorithm. In the 1980s, starting with the work of Johnson, Micchelli, and Paul [14], there has been a revival of Rutishauser's method and polynomial preconditioning in general, see [21, 7, 19, 2] and the references given there. The main reason for this renewed interest is that polynomial preconditioning is an attractive technique on vector and parallel computers (see, e.g., [22]). Each CG iteration involves the computation of inner products, which constitutes a bottleneck on many modern architectures. Typically, polynomial preconditioning reduces the total number of inner products, since the multiplication by the preconditioning matrix $\psi(A)$ does not require inner products.

For the Chebyshev iteration in Rutishauser's method, estimates a and b for the smallest and largest eigenvalues of A are needed. Often, good upper bounds b can be obtained easily, using simple techniques such as Gershgorin's theorem [21]. It is far more difficult to estimate the smallest eigenvalue. Saad [21] has proposed a polynomial preconditioning technique that only requires an upper bound for the largest eigenvalue, while the trivial bound $a = 0$ is used for the smallest eigenvalue of the positive definite matrix A. His technique is based on least squares polynomials associated with the family of Jacobi weights [24]. Ideally, one would choose the weight function such that CG for the preconditioned systems (1.2) and (1.3) converges as fast as possible. However, this problem is not addressed in [21].

Another option is to construct polynomial preconditioners via weighted Chebyshev approximation problems. This was proposed by Freund [7] who also suggested a heuristic for adapting the weight function to the eigenvalue distribution of A. The technique exploits the observation that eigenvalue distributions of Hermitian matrices can be surprisingly well estimated, using only a few steps of the Lanczos method. Actually, spectral estimation based on the Lanczos process is a widely used technique in applications (see, e.g., [25]). The obtained estimated eigenvalue distribution is then used to construct a weight function, and finally the polynomial preconditioner is computed by solving the corresponding approximation problem.

In this paper, we further study polynomial preconditioning based on weighted Chebyshev approximation problems. In particular, we investigate the use of Bernstein-Szegő weights [24]. For such weights, the solutions of the associated approximation problems are known explicitly. Therefore, the construction of preconditioning polynomials based on Bernstein-Szegő weights does not involve the numerical solution of an approximation problem.

The remainder of this paper is organized as follows. In § 2, we recall some basic properties of CG, and we discuss Chebyshev polynomial preconditioning. In § 3, we present our approach to polynomial preconditioning based on weighted Chebyshev approximation problems, and we propose a procedure for obtaining a suitable weight function from the Lanczos process. This technique involves the construction of a monotone interpolant. In § 4, we briefly describe a procedure for monotone piecewise cubic interpolation. In § 5, we consider polynomial preconditioners based on Bernstein-Szegő weights. Finally, in § 6, we make some concluding remarks.

Throughout the paper, it is assumed that A in (1.1) is a Hermitian positive definite $N \times N$ matrix, with real or complex entries. As usual, M^H denotes the conjugate transpose

of a matrix M. The vector norm $\|x\| = \sqrt{x^H x}$ is always the Euclidean norm. Finally, we denote by

$$\mathcal{P}_n := \{\varphi(\lambda) \equiv \sigma_0 + \sigma_1\lambda + \cdots + \sigma_n\lambda^n \mid \sigma_0, \sigma_1, \ldots, \sigma_n \in \mathbf{C}\}$$

the set of all complex polynomials of degree at most n.

2. CG and Chebyshev polynomial preconditioning. In this section, we collect some basic facts about CG, and we review Chebyshev polynomial preconditioning.

2.1. The CG algorithm. Let $x_0 \in \mathbf{C}^N$ be any initial guess for the solution of (1.1), and let $r_0 := b - Ax_0$ be the associated residual vector. The CG algorithm generates iterates of the form

$$x_n = x_0 + \chi_n(A)r_0, \quad \text{where} \quad \chi_n \in \mathcal{P}_{n-1}, \quad n = 1, 2, \ldots . \tag{2.1}$$

The corresponding residual vectors are given by

$$r_n = \phi_n(A)r_0, \quad \text{where} \quad \phi_n(\lambda) \equiv 1 - \lambda\chi_n(\lambda). \tag{2.2}$$

In exact arithmetic, the CG algorithm terminates after a finite number of steps with the exact solution $x_L = A^{-1}b$ of (1.1). In the sequel, L always denotes this termination index. We remark that L is just the minimal number of components in any expansion of r_0 into orthonormal eigenvectors v_j of A, and thus we have

$$r_0 = \sum_{j=1}^{L} \sigma_j v_j, \quad \text{where} \quad \sigma_j \neq 0 \quad \text{for all} \quad j. \tag{2.3}$$

In particular, $L \leq N$. In the following, we always assume that the vectors v_j have been scaled such that $\sigma_j > 0$ in (2.3). Furthermore, we denote by λ_j the eigenvalues corresponding to v_j, i.e.,

$$Av_j = \lambda_j v_j. \tag{2.4}$$

Clearly, the λ_j's are distinct, and from now on, we assume that they are numbered in increasing order:

$$\lambda_1 < \lambda_2 < \cdots < \lambda_L.$$

The CG iterates are optimal, in the sense that $r_n^H A^{-1} r_n$ is minimal for all possible iterates of the form (2.1). This minimization property can be shown to be equivalent to the following orthogonality relations:

$$r_n^H r_k = 0 \quad \text{for all} \quad n, k = 0, 1, \ldots, L, \; n \neq k. \tag{2.5}$$

Using (2.2), (2.3), and (2.4), we can rewrite (2.5) in the form

$$\langle \phi_n, \phi_k \rangle = 0 \quad \text{for all} \quad n, k = 0, 1, \ldots, L, \; n \neq k, \tag{2.6}$$

where

$$\begin{aligned}\langle \phi_n, \phi_k \rangle &:= r_0^H \phi_n(A)\phi_k(A)r_0 \\ &= \sum_{j=1}^{L} \sigma_j^2 \phi_n(\lambda_j)\phi_k(\lambda_j) =: \int_{\mathbf{R}} \phi_n(\lambda)\phi_k(\lambda)d\sigma(\lambda).\end{aligned} \tag{2.7}$$

The distribution function $\sigma(\lambda)$ in (2.7) is defined by

$$\sigma(\lambda) \equiv \sum_{j=1}^{L} \sigma_j^2 H(\lambda - \lambda_j), \quad \text{where} \quad H(\lambda) \equiv \begin{cases} 1 & \text{for } \lambda \geq 0, \\ 0 & \text{otherwise.} \end{cases} \tag{2.8}$$

We now turn to polynomial preconditioning. For simplicity, we will focus only on right preconditioning (1.3), but essentially the same statements remain true for left preconditioning (1.2). From now on, it is always assumed that $p > 1$ is a given integer, and we consider preconditioning polynomials $\psi \in \mathcal{P}_{p-1}$ in (1.3).

2.2. Chebyshev polynomial preconditioning. The standard approach for the design of preconditioners is to choose the polynomial ψ in (1.3) such that $A\psi(A)$ is in some sense as close as possible to the identity matrix I. For instance, one could attempt to minimize the Euclidean norm $\|I - A\psi(A)\|$. However, the solution of this problem would require the knowledge of all eigenvalues of A. Therefore, one usually substitutes for the spectrum of A an interval $[a, b]$, where $0 < a \leq \lambda_1$ and $b \geq \lambda_L$. This leads to the Chebyshev approximation problem

$$\min_{\varphi \in \mathcal{P}_p:\, \varphi(0)=1} \; \max_{\lambda \in [a,b]} \; |\varphi(\lambda)|, \tag{2.9}$$

where $\varphi(\lambda) \equiv 1 - \lambda\psi(\lambda)$. It is well known that the optimal solution φ_p of (2.9) is just a suitably shifted and normalized Chebyshev polynomial of the first kind, and we have

$$\varphi_p(\lambda) \equiv \frac{T_p(\ell(\lambda))}{T_p(\xi)}, \quad \text{where} \quad \ell(\lambda) \equiv \frac{b + a - 2\lambda}{b - a} \quad \text{and} \quad \xi = \ell(0). \tag{2.10}$$

Note that the spectrum of the preconditioned matrix $A\psi_p(A)$ is contained in the interval

$$[a_p, b_p], \quad \text{where} \quad a_p := 1 - \frac{1}{T_p(\xi)} \quad \text{and} \quad b_p := 1 + \frac{1}{T_p(\xi)}. \tag{2.11}$$

In view of the optimality properties of the CG algorithm, n steps of CG applied to the preconditioned system (1.3) can at best give the same residual vector as np steps of CG applied to the original system (1.1). As in (2.2), let $r_{np} = \phi_{np}(A)r_0$ denote the residual vector obtained after np steps of CG applied to (1.1). Similarly, let $r_n^{(\mathrm{P})} = \phi_n^{(\mathrm{P})}(A\psi_p(A))r_0$ be the nth residual vector generated by CG applied to (1.3). Obviously, Chebyshev polynomial preconditioning is indeed best possible, provided that the residual polynomials ϕ_{np} and $\phi_n^{(\mathrm{P})}$ satisfy

$$\phi_{np}(\lambda) \equiv \phi_n^{(\mathrm{P})}(\lambda\psi_p(\lambda)). \tag{2.12}$$

Next, we show that (2.12) is fulfilled if the distribution function $\sigma(\lambda)$ in (2.8) corresponds to the worst-case distribution.

The function $\sigma(\lambda)$ is a step function with jumps at the eigenvalues of A. For our discussion, we treat $\sigma(\lambda)$ as a continuous function, and we denote by

$$\delta(\lambda) \equiv \frac{d\sigma(\lambda)}{d\lambda}$$

the corresponding density function. Actually, since the dimension of A (and thus L) is usually large, the step function "looks like" a continuous function. From potential theory, it is known that the worst-case distribution for an interval is the equilibrium distribution, and for the case of the unit interval, this is given by

$$\sigma^{\mathrm{E}}(t) \equiv \arcsin(t), \quad t \in [-1,1]. \tag{2.13}$$

Furthermore, the orthogonal polynomials with respect to the corresponding inner product are the Chebyshev polynomials of the first kind, and we have, for all integers $n,\ k \geq 0, n \neq k$,

$$\int_{-1}^{1} T_n(t)T_k(t)\delta^{\mathrm{E}}(t)dt = 0, \quad \text{where} \quad \delta^{\mathrm{E}}(t) \equiv \frac{1}{\sqrt{1-t^2}}. \tag{2.14}$$

We remark that (2.14) can be evaluated by means of Gaussian quadrature. This gives

$$-\frac{b-a}{2}\int_{-1}^{1} \frac{T_n(t)T_k(t)}{T_n(\xi)T_k(\xi)}\delta^{\mathrm{E}}(t)dt = \sum_{j=1}^{L} \sigma_j^2 \frac{T_n(\ell(\lambda_j))T_k(\ell(\lambda_j))}{T_n(\xi)T_k(\xi)}, \quad n,k < L. \tag{2.15}$$

In other words, for the distribution function defined by (2.15), the CG residuals correspond to Chebyshev polynomials. We remark that in this case the standard error bounds (see, e.g., [11]) for the CG iterates are sharp.

Next, we show that for the worst-case distribution the relation (2.12) is indeed satisfied, and hence Chebyshev polynomial preconditioning is optimal in this case.

LEMMA 2.1. *Let ℓ_p denote the linear mapping that maps $[a_p, b_p]$ (cf. (2.11)) onto the unit interval $[-1,1]$ and let $\xi_p = \ell_p(0)$. Let $\phi_{np}(\lambda) \equiv T_{np}(\ell(\lambda))/T_{np}(\xi)$ and $\phi_n^{(\mathrm{P})}(\lambda) \equiv T_n(\ell_p(\lambda))/T_n(\xi_p)$ be the shifted and normalized Chebyshev polynomial on $[a,b]$ and $[a_p.b_p]$, respectively. Then the identity (2.12) is satisfied.*

Proof. From (2.11) and (2.10) one readily obtains

$$\phi_n^{(\mathrm{P})}(\lambda\psi_p(\lambda)) \equiv \frac{T_n(T_p(\ell(\lambda)))}{T_n(T_p(\xi))}.$$

Equation (2.12) then follows from the well-known identity $T_{np}(t) \equiv T_n(T_p(t))$. □

3. Weighted polynomial preconditioning. As discussed in the previous section, Chebyshev polynomial preconditioning is optimal for matrices A, for which the function $\sigma(\lambda)$ in (2.8) is the worst-case distribution. However, in practice, linear systems, especially those arising in the numerical treatment of partial differential equations, have eigenvalue distributions that are far from the worst case. For those, Chebyshev polynomial preconditioning

is not optimal. Furthermore, it is known [1] that repeated applications of Chebyshev polynomials will transform any given distribution into the worst-case distribution. This behavior is also reflected in Chebyshev polynomial preconditioning. If A has a favorable eigenvalue distribution then the eigenvalue distribution of the preconditioned system is usually much closer to the worst case.

In this section, we propose a heuristic for adapting the preconditioning polynomial to the actual eigenvalue distribution of A.

3.1. Weighted Chebyshev approximation problems. Instead of (2.9), we now consider weighted Chebyshev approximation problems of the form

$$\min_{\varphi\in\mathcal{P}_p:\varphi(0)=1} \max_{\lambda\in[a,b]} |\tilde{w}(\lambda)\varphi(\lambda)|. \tag{3.1}$$

Here $\tilde{w}$ is a continuous weight function on $[a,b]$, and it is always assumed that $\tilde{w}(\lambda) > 0$ on the open interval (a,b). Standard results from approximation theory (see, e.g., [18]) guarantee that there exists a unique optimal polynomial φ_p for (3.1). In general, φ_p is not known explicitly, and one needs to solve (3.1) numerically, for example, using the Remez algorithm (see, e.g., [18, 10]).

If $\tilde{w}(\lambda) \equiv 1$, then the shifted and scaled Chebyshev polynomial (2.10) is the solution of (3.1). The idea now is to use the optimal solution φ_p as a polynomial preconditioner, where the weight function $\tilde{w}$ is chosen based on an estimate for the density δ of the eigenvalue distribution of A. It is more convenient to rewrite (3.1) for the unit interval $[-1,1]$ instead of $[a,b]$. Using the transformation $l(\lambda)$ defined in (2.10), we obtain the approximation problem

$$\min_{\varphi_p\in\mathcal{P}_p:\varphi_p(\xi)=1} \max_{t\in[-1,1]} |w(t)\varphi_p(t)|, \quad \xi \notin [-1,1]. \tag{3.2}$$

It remains to give a heuristic for the choice of the weight function w in (3.2). If the estimated density δ of A happens to be the worst-case density δ^{E} (cf. (2.14)), then the solution of (3.2) should yield the optimal preconditioner for the worst case. As shown in § 2, Chebyshev polynomials are optimal in this case. Clearly, the weight function should be constructed such that $w(\lambda) \equiv 1$ if $\delta(t) \equiv 1/\sqrt{1-t^2}$. Therefore, we suggest the choice

$$w(t) \equiv \sqrt{\delta(t)\sqrt{1-t^2}}. \tag{3.3}$$

3.2. Estimating the distribution function. Usually, a good estimate for the eigenvalue distribution $\sigma(\lambda)$ in (2.8) can be derived from the quantities generated by relatively few steps of the Lanczos process [16]. Actually, since the Lanczos algorithm is equivalent to CG, we can extract all necessary information from a few steps of the CG algorithm applied to the original system (1.1).

Suppose we have run CG for n steps. Then it has generated the entries of the $n \times n$ tridiagonal Lanczos matrix

$$T_n = \begin{bmatrix} \alpha_1 & \beta_2 & 0 & \cdots & 0 \\ \beta_2 & \alpha_2 & \beta_3 & \ddots & \vdots \\ 0 & \beta_3 & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & \beta_n \\ 0 & \cdots & 0 & \beta_n & \alpha_n \end{bmatrix}. \tag{3.4}$$

The eigenvalues of T_n (the so-called Ritz values) are all distinct, and we assume that they are numbered in increasing order:

$$\theta_1 < \theta_2 < \cdots < \theta_n.$$

Let s_i be a set of corresponding orthonormal eigenvectors, i.e.,

$$T_n s_i = \theta_i s_i, \quad \|s_i\| = 1,$$

Moreover, we assume that the s_i's are normalized such that their first components, $\tau_i := (s_i)_1$, are all real and nonnegative.

It is well known that the CG residual polynomials $\phi_0, \phi_1, \ldots, \phi_n$ (cf. (2.2)) are orthogonal with respect to the discrete inner product induced by T_n. More precisely, we have

$$\langle \phi_j, \phi_k \rangle_n = 0 \quad \text{for all} \quad j,k = 0,1,\ldots,n,\ j \neq k, \tag{3.5}$$

where

$$\begin{aligned} \langle \phi_j, \phi_k \rangle_n &:= e_1^H \phi_k(T_n)\phi_j(T_n)e_1 \\ &= \sum_{i=1}^{n} \tau_i^2 \phi_j(\theta_i)\phi_k(\theta_i) =: \int_{\mathbf{R}} \phi_j(\lambda)\phi_k(\lambda)d\tau(\lambda). \end{aligned} \tag{3.6}$$

The distribution function $\tau(\lambda)$ in (3.6) is defined by

$$\tau(\lambda) \equiv \sum_{i=1}^{n} \tau_i^2 H(\lambda - \theta_i),$$

where H is given in (2.8). It can be shown that $\langle \cdot, \cdot \rangle_n$ and the inner product $\langle \cdot, \cdot \rangle$ in (2.7) have the same (modified) moments up to degree $n-1$, i.e., for all real polynomials ϕ of degree at most $n-1$, it holds that

$$\langle 1, \phi \rangle_n = \langle 1, \phi \rangle. \tag{3.7}$$

We can then apply a result by Karlin and Shapley [15, Theorem 22.1], which, roughly speaking, states that the step function $\tau(\lambda)$ has to be close to $\sigma(\lambda)$. More precisely, their theorem states that the condition (3.7) implies that $\sigma(\lambda) - \tau(\lambda)$ has at least $n-1$ sign changes in $[\lambda_{\min}(A), \lambda_{\max}(A)]$. Here, $\lambda_{\min}(A)$ and $\lambda_{\max}(A)$ denote the smallest and largest eigenvalues of A, respectively.

Therefore, we use a monotone C^1-interpolant of the step function $\tau(\lambda)$ as our estimation for the eigenvalue distribution σ of A. For simplicity, we again transform the interval $[a,b]$ to the unit interval $[-1,1]$, using the linear map $\ell(\lambda)$ defined in (2.10). We note that we only require that the lower bound a is nonnegative. In particular, the trivial choice $a = 0$ is feasible. Since the largest Ritz value θ_n is usually a good approximation for $\lambda_{\max}(A)$, we set $b := \theta_n$. Finally, we always denote by $f(t)$ the estimate for the eigenvalue distribution $\sigma(\lambda)$ transformed to $[-1,1]$.

The interpolating points are chosen as follows. We set

$$t_i := \begin{cases} 0 & \text{for } i = 0, \\ \ell(\theta_i) & \text{for } i = 1, 2, \ldots, n-1, \\ 1 & \text{for } i = n, \end{cases}$$

and

$$\vartheta_i := \begin{cases} 0 & \text{for } i = 0, \\ \tau_i^2/2 + \sum_{j=1}^{i-1} \tau_j^2 & \text{for } i = 1, 2, \ldots, n-1, \\ 1 & \text{for } i = n. \end{cases}$$

The estimated distribution is then chosen as a monotone function $f \in C^1[-1,1]$ satisfying

$$f(t_i) = \vartheta_i, \quad i = 0, 1, \ldots, n. \tag{3.8}$$

4. Monotone piecewise cubic interpolation. One obvious choice for the interpolating function f is a monotone piecewise cubic interpolant. In this section we briefly describe how to construct such a function. We follow the derivation of Fritsch and Carlson [9] and Fritsch and Butland [8].

Let $-1 = t_0 < t_1 < \ldots < t_n = 1$ and $0 = \vartheta_0 < \vartheta_1 < \ldots < \vartheta_n = 1$ be given. Our goal is to construct a piecewise cubic function $S \in C^1[-1,1]$ such that

$$S(t_i) = \vartheta_i, \quad i = 0, 1, \ldots, n, \tag{4.1}$$

and S is monotone on [-1,1]. To this end, let $h_i = t_{i+1} - t_i$ and $\Delta_i = (\vartheta_{i+1} - \vartheta_i)/h_i$, $i = 0, 1, \ldots, n-1$. The trick is to express S, on any subinterval $[t_i, t_{i+1}]$, in terms of the derivatives $d_i = S'(t_i)$, $i = 0, 1, \ldots, n$, (cf. [9])

$$S(t) \equiv \left[\frac{d_i + d_{i+1} - 2\Delta_i}{h_i^2}\right](t - t_i)^3 + \left[\frac{-2d_i - d_{i+1} + 3\Delta_i}{h_i}\right](t - t_i)^2 + d_i(t - t_i) + \vartheta_i.$$

By construction, any choice of the free parameters d_i leads to a function $S \in C^1[-1,1]$ that fulfills (4.1). The remaining step is to adjust the d_i's such that S is monotone on $[-1,1]$.

In the literature one can find several schemes for computing the (not uniquely determined) d_i. Here we used a formula proposed by Brodlie [3] and Fritsch and Butland [8]:

$$d_i = \begin{cases} \dfrac{\Delta_{i-1}\Delta_i}{\xi_i\Delta_i + (1-\xi_i)\Delta_{i-1}} & \text{for } \Delta_{i-1}\Delta_i > 0, \\ 0 & \text{otherwise,} \end{cases} \quad i = 1, 2, \ldots, n-1, \tag{4.2}$$

where $\xi_i = (h_{i-1} + 2h_i)/(3(h_{i-1} + h_i))$. In addition to (4.2), we still need to choose the boundary conditions d_0 and d_n. Since we have no information about the endpoint derivatives available, we select a (weak) version of the so-called "not-a-knot" condition (cf. De Boor [5,

pp. 54]). Here one chooses d_0 and d_n such that S is twice continuously differentiable on $[t_0, t_2)$ and $(t_{n-2}, t_n]$, respectively. One obtains

$$\begin{aligned} d_0 &= \frac{h_0}{h_1}(3\Delta_1 - (2d_1 + d_2)) + 3\Delta_0 - 2d_1, \\ d_n &= \frac{h_{n-1}}{h_{n-2}}(3\Delta_{n-2} - (2d_{n-1} + d_{n-2})) + 3\Delta_{n-1} - 2d_{n-1}. \end{aligned} \tag{4.3}$$

However, this special choice of d_0 and d_n does not necessarily produce a monotone S on the subintervals $[t_0, t_1]$ and $[t_{n-1}, t_n]$, respectively.

Note that the additional requirements $d_0 \in [0, 3\Delta_0]$ and $d_n \in [0, 3\Delta_{n-1}]$ will lead to a monotone function. Thus, if, e.g., d_0 (computed by (4.3)) turns out to be negative or bigger than $3\Delta_0$, we simply set $d_0 = 0$ or $d_0 = 3\Delta_0$, respectively.

We would like to mention that FORTRAN versions for the described procedures are available in NETLIB (PCHIP package).

Finally, we set $\delta(t) \equiv S'(t)$, where S is the computed monotone interpolant, and then define the weight function $w(t) \equiv w(t; S)$ by (3.3). The desired polynomial preconditioner is then obtained by (numerically) solving the resulting weighted Chebyshev approximation problem (3.2).

5. Bernstein-Szegö weight functions. In the preceding section we first approximated the distribution function and then solved the resulting approximation problem (3.2) numerically. Here we follow a different approach which will eliminate the latter approximation process. The idea is to restrict the weight functions to a class, for which the solution of the Chebyshev problem (3.2) is explicitly known. We consider three classes of weight function that fulfill this requirement.

Let ρ_k be a real polynomial of degree k with $\rho_k(t) > 0$ on $[-1, 1]$, and define

$$s_0(t) \equiv 1, \quad s_{1/2}(t) \equiv \sqrt{1+t}, \quad s_1(t) \equiv \sqrt{1-t^2}. \tag{5.1}$$

Then (3.2) can be solved explicitly for the so-called Bernstein-Szegö weight functions

$$w_j(t) \equiv \frac{s_j(t)}{\sqrt{\rho_k(t)}}, \quad j \in \{0, 1/2, 1\}; \tag{5.2}$$

see Szegö [24], Freund [6], and the references given therein. More precisely, the solution φ_p of (3.2) with respect to w_j, $j \in \{0, 1/2, 1\}$, is explicitly known for $p \geq p_j$, where

$$p_j = \begin{cases} 0 & \text{if } j = 1 \text{ and } k = 0, \\ \lfloor (k+1)/2 - j \rfloor & \text{otherwise.} \end{cases}$$

For convenience, in the sequel, we allow ρ_k to have (simple) zeros at the endpoints ± 1, i.e., the cases $j = 0, 1/2$ are now included in the case $j = 1$. Therefore, we will only investigate weight functions of the form

$$w(t) \equiv w_1(t) \equiv \frac{\sqrt{1-t^2}}{\sqrt{\rho_k(t)}}. \tag{5.3}$$

In view of (3.3), we have to adjust ρ_k such that

$$f(\lambda) \equiv \int_{-1}^{\lambda} \frac{\sqrt{1-t^2}}{\rho_k(t)} dt \tag{5.4}$$

fulfills the interpolatory conditions $f(t_j) = \theta_j$ (cf. (3.8)). Note that f, defined by (5.4), is "automatically" monotone on $[-1,1]$, and that $f(-1) = 0$ for every ρ_k.

It turns out to be advantageous to express ρ_k in terms of its zeros

$$\rho_k(t) \equiv a_{m+1} \prod_{j=1}^{m} (t - a_j) \prod_{j=1}^{l} (t - z_j)(t - \overline{z}_j), \quad k = m + 2l. \tag{5.5}$$

Here, a_j, $j = 1, 2, \ldots, m$, are the real zeros and $z_j = x_j + iy_j$, $y_j > 0$, $j = 1, 2, \ldots, l$, are the complex zeros in the upper half plane. The partial fractions expansion of (5.4) reads

$$\begin{aligned} f(\lambda) &\equiv \frac{1}{a_{m+1}} \left(\sum_{j=1}^{m} A_j \int_{-1}^{\lambda} \frac{\sqrt{1-t^2}}{t - a_j} dt + \sum_{j=1}^{l} \int_{-1}^{\lambda} \frac{(B_j t + C_j)\sqrt{1-t^2}}{(t - z_j)(t - \overline{z}_j)} dt \right) \\ &\equiv F(\lambda; a_1, \ldots, a_{m+1}, x_1, \ldots, x_l, y_1, \ldots, y_l), \end{aligned} \tag{5.6}$$

where

$$A_j = \frac{1}{\rho_k'(a_j)}, \quad j = 1, 2, \ldots, m,$$

$$B_j t + C_j \equiv \frac{\rho_k'(\overline{z}_j)(t - \overline{z}_j) + \rho_k'(z_j)(t - z_j)}{\rho_k'(z_j)\rho_k'(\overline{z}_j)}, \quad j = 1, 2, \ldots, l.$$

It is readily verified that all integrals in (5.6) have an explicit antiderivative. We omit these routine, but somewhat lengthy, calculations.

Now the interpolatory conditions $f(t_i) = \vartheta_i$ (cf. (3.8)) and the positivity constraints $\rho_k(t) > 0$, $t \in (-1, 1)$ lead to the following nonlinear constraint interpolation problem:

find real numbers $a_1, \ldots, a_{m+1}, x_1, \ldots, x_l, y_1, \ldots, y_l$ subject to

$$\begin{aligned} F(t_j; a_1, \ldots, a_{m+1}, x_1, \ldots, x_l, y_1, \ldots, y_l) &= \vartheta_j, \quad j = 1, 2, \ldots, n, \\ |a_j| &\geq 1, \quad j = 1, 2, \ldots, m, \\ \operatorname{sgn}(a_{m+1}) &= (-1)^m \prod_{j=1}^{m} \operatorname{sgn}(a_j), \\ y_j &> 0, \quad j = 1, 2, \ldots, l. \end{aligned} \tag{5.7}$$

Apart from the sign conditions, the problem (5.7) can be viewed as a nonlinear system of n equations in $k + 1 = m + 1 + 2l$ unknowns. Therefore, a natural choice of the polynomial degree is $k = n - 1$. Also, we would like to mention that (5.7) does not address the problem of (possible) multiple zeros at ± 1. However, this problem never occurred in our experiments.

The success of any nonlinear solver applied to (5.7) depends strongly on good starting values. We will now describe a (linear) method for obtaining such values. In view of (5.4) we have

$$\rho_k(\lambda)f'(\lambda) \equiv \sqrt{1-\lambda^2},$$

and consequently

$$(\rho_k(\lambda)f(\lambda))' \equiv \rho_k(\lambda)'f(\lambda) + \sqrt{1-\lambda^2}. \tag{5.8}$$

Hence, by integrating (4.8) and using $f(-1) = 0$, we obtain the identity

$$\rho_k(\lambda)f(\lambda) \equiv \int_{-1}^{\lambda} \rho_k(t)'f(t)dt + g(\lambda), \tag{5.9}$$

where $g(\lambda) \equiv \int_{-1}^{\lambda} \sqrt{1-t^2}dt$. Setting

$$\begin{aligned} \rho_k(\lambda) &\equiv \sum_{j=0}^{k} b_j\lambda^j, \quad b_j \in \mathbb{R}, \\ \gamma_j(\lambda; f) &\equiv \int_{-1}^{\lambda} t^{j-1}f(t)dt \end{aligned} \tag{5.10}$$

in equation (5.9) gives

$$\sum_{j=0}^{k} b_j \left(\lambda^j f(\lambda) - j\gamma_j(\lambda; f)\right) \equiv g(\lambda). \tag{5.11}$$

Then we end up with a semiinfinite linear problem:

find real numbers b_j, $j = 0, 1, \ldots, k$, subject to

$$\begin{aligned} G(t_i; b_0, \ldots, b_k) &= 0, \quad i = 1, 2, \ldots, n, \\ \sum_{j=0}^{k} b_j t^j &> 0, \quad t \in (-1, 1), \end{aligned} \tag{5.12}$$

where

$$G(t_i; b_0, \ldots, b_k) = \sum_{j=0}^{k} b_j \left(t_i^j\theta_i - j\gamma_j(t_i; f)\right) - g(t_i).$$

The price paid for the linearity of the problem are the infinitely many constraints. In addition, we have to evaluate the integrals γ_j which involve the unknown function f (cf. (5.10)). To overcome this problem, we approximate f by a monotone piecewise cubic function S, as described in § 4. Clearly, the resulting integrals $\gamma_j(\lambda; S)$ are easy to compute.

A straightforward approach for attacking semiinfinite problems is to replace the infinitely many constraints by a finite subset. Finally, after relaxing the interpolatory conditions slightly, we obtain the following linear programming problem:

find real numbers b_j, $j = 0, 1, \ldots, k$, subject to

$$\begin{aligned} &\sum_{i=1}^{n} G(t_i; b_0, \ldots, b_k) = \min!, \\ &G(t_i; b_0, \ldots, b_k) \geq 0 \quad i = 1, 2, \ldots, n, \\ &\sum_{j=0}^{k} b_j \hat{t}_i^j > 0, \quad \hat{t}_i \in (-1, 1),\ i = 1, 2, \ldots, M. \end{aligned} \tag{5.13}$$

It is not hard to check that the feasible set of (5.13) is not empty, and hence (5.13) always has a solution $\rho_k^*(t; M) \equiv \sum_{j=0}^{k} b_j^* t^j$. Here the second parameter M indicates that ρ_k^* depends on the number of positivity constraints. Unfortunately, it is possible that $\rho_k^*(t; M)$ has zeros in $(-1, 1)$. Here, one basically has to distinguish between the cases that $\rho_k^*(t; M)$ has one zero μ in the first (last) interval $(-1, \hat{t}_1)$ $((\hat{t}_M, 1))$ or two zeros μ_1, μ_2 in $(\hat{t}_i, \hat{t}_{i+1})$, $i \in \{0, 1, \ldots, M\}$, where $\hat{t}_0 := -1$ and $\hat{t}_{M+1} := 1$. Such zeros can not be used as starting values for the nonlinear problem (5.7). Therefore, we replace μ by -1 or 1 and the real zeros μ_1, μ_2 by the complex zeros $z = (\mu_1 + \mu_2)/2 + i\varepsilon$, $\overline{z}$, where ε is some small positive number. Notice that

$$(t - \mu_1)(t - \mu_2) - (t - z)(t - \overline{z}) = \frac{1}{4}(\mu_1 - \mu_2)^2 + \varepsilon^2 < |\hat{t}_i - \hat{t}_{i+1}|^2 + \varepsilon^2.$$

For sufficiently large M and sufficiently small ε, this "zero-substitution" only slightly perturbs $\rho_k^*(t; M)$.

In our experiments, we always chose the $\hat{t}_i$'s as Chebyshev knots with $M = 200$. The corresponding linear solution $\rho_k^*(t; M)$ always served as a good starting guess for the nonlinear solver applied to (5.7).

6. Concluding remarks. On modern architectures, it is attractive to combine the conjugate gradient algorithm with polynomial preconditioning. In this paper, we have presented an approach for adapting polynomial preconditioners to the actual eigenvalue distribution of the coefficient matrix of the linear system. Our technique is based on the observation that good estimates for the eigenvalue distribution can be derived after only a few steps of the Lanczos process. We then use this information to construct a weight function for a suitable Chebyshev approximation problem. The solution of this problem yields the polynomial preconditioner. We have explored the use of Bernstein-Szegö weights.

This manuscript is still a preliminary and incomplete version. In the final paper, we will also include numerical results.

Acknowledgement. We would like to thank Youcef Saad for bringing reference [17] to our attention.

REFERENCES

[1] R. L. ADLER AND T. J. RIVLIN, *Ergodic and mixing properties of Chebyshev polynomials*, Proc. Amer. Math. Soc., 15 (1964), pp. 794–796.

[2] S. F. ASHBY, T. A. MANTEUFFEL, AND J. S. OTTO, *A comparison of adaptive Chebyshev and least squares polynomial preconditioning for Hermitian positive definite linear systems*, SIAM J. Sci. Stat. Comput., 13 (1992), pp. 1–29.

[3] K. W. BRODLIE, *A review of methods for curve and function drawing*, in Mathematical Methods in Computer Graphics and Design, K. W. Brodlie, ed., Academic Press, London, 1980, pp. 1–37.

[4] P. CONCUS, G. H. GOLUB, AND D. P. O'LEARY, *A generalized conjugate gradient method for the numerical solution of elliptic partial differential equations*, in Sparse Matrix Computations, J. R. Bunch and D. J. Rose, eds., Academic Press, New York, 1976, pp. 309–332.

[5] C. DE BOOR, *A Practical Guide to Splines*, Springer, New York, 1978.

[6] R. W. FREUND, *On some approximation problems for complex polynomials*, Constr. Approx., 4 (1988), pp. 111–121.

[7] R. W. FREUND, *Polynomial preconditioners for Hermitian and certain non-Hermitian matrices*, Minisymposium Presentation, 1989 SIAM Annual Meeting, San Diego, July 1989.

[8] F. N. FRITSCH AND J. BUTLAND, *A method for constructing local monotone piecewise cubic interpolation*, SIAM J. Sci. Stat. Comput., 5 (1984), pp. 300–304.

[9] F. N. FRITSCH AND R. E. CARLSON, *Monotone piecewise cubic interpolation*, SIAM J. Numer. Anal., 17 (1980), pp. 238–246.

[10] G. H. GOLUB AND L. B. SMITH, *Chebyshev approximation of continuous functions by a Chebyshev system of functions*, Collect. Algorithms CACM, 414 (1971), pp. P1–P10.

[11] G. H. GOLUB AND C. F. VAN LOAN, *Matrix Computations*, Second Edition, The Johns Hopkins University Press, Baltimore, 1989.

[12] G. H. GOLUB AND R. S. VARGA, *Chebyshev semi-iterative methods, successive overrelaxation iterative methods, and second order Richardson iterative methods*, Numer. Math., 3 (1961), pp. 147–168.

[13] M. R. HESTENES AND E. STIEFEL, *Methods of conjugate gradients for solving linear systems*, J. Res. Nat. Bur. Stand., 49 (1952), pp. 409–436.

[14] O. G. JOHNSON, C. A. MICCHELLI, AND G. PAUL, *Polynomial preconditioners for conjugate gradient calculations*, SIAM J. Numer. Anal., 20 (1983), pp. 362–376.

[15] S. KARLIN AND L. S. SHAPLEY, *Geometry of moment spaces, Memoirs of the American Mathematical Society*, 12, AMS, Providence, 1953.

[16] C. LANCZOS, *An iteration method for the solution of the eigenvalue problem of linear differential and integral operators*, J. Res. Natl. Bur. Stand., 45 (1950), pp. 255–282.

[17] C. LANCZOS, *Chebyshev polynomials in the solution of large-scale linear systems*, in Proceedings of the Association for Computing Machinery, Toronto, Sauls Lithograph Co., Washington, D.C., 1953, pp. 124–133.

[18] G. MEINARDUS, *Approximation of functions: Theory and numerical methods*, Springer, New York, 1967.

[19] D. P. O'LEARY, *Yet another polynomial preconditioner for the conjugate gradient algorithm*, Linear Alg. Appl., 154-156 (1991), pp. 377–388.

[20] H. RUTISHAUSER, *Theory of gradient methods, in Refined Iterative Methods for Computation of the Solution and the Eigenvalues of Self-Adjoint Boundary Value Problems*, M. Engeli, Th. Ginsburg, H. Rutishauser, and E. Stiefel, eds., Birkhäuser, Basel (1959), pp. 24–49.

[21] Y. SAAD, *Practical use of polynomial preconditionings for the conjugate gradient method*, SIAM J. Sci. Stat. Comput., 6 (1985), pp. 865–881.

[22] Y. SAAD, *Krylov subspace methods on supercomputers*, SIAM J. Sci. Stat. Comput., 10 (1989), pp. 1200–1232.

[23] E. L. STIEFEL, *Kernel polynomials in linear algebra and their numerical applications*, U.S. National Bureau of Standards, Applied Mathematics Series, 49 (1958), pp. 1–22.

[24] G. SZEGŐ, *Orthogonal Polynomials*, Fourth Edition, Amer. Math. Soc., Providence, 1975.

[25] R. R. WHITEHEAD, *Moment methods and Lanczos methods*, in Theory and Applications of Moment Methods in Many-Fermion Systems, B. J. Dalton, S. M. Grimes, J. P. Vary, and S. A. Williams, eds., Plenum Press, New York, 1980.

RIACS
Mail Stop 230-5
NASA Ames Research Center
Moffett Field, CA 94035