# Some Unusual Eigenvalue Problems
## Invited Talk

Zhaojun Bai[1] and Gene H. Golub[2]

[1] University of Kentucky
Lexington, KY 40506, USA,
bai@ms.uky.edu
[2] Stanford University
Stanford, CA 94305, USA
golub@sccm.stanford.edu

**Abstract.** We survey some unusual eigenvalue problems arising in different applications. We show that all these problems can be cast as problems of estimating quadratic forms. Numerical algorithms based on the well-known Gauss-type quadrature rules and Lanczos process are reviewed for computing these quadratic forms. These algorithms reference the matrix in question only through a matrix-vector product operation. Hence it is well suited for large sparse problems. Some selected numerical examples are presented to illustrate the efficiency of such an approach.

## 1 Introduction

Matrix eigenvalue problems play a significant role in many areas of computational science and engineering. It often happens that many eigenvalue problems arising in applications may not appear in a standard form that we usually learn from a textbook and find in software packages for solving eigenvalue problems. In this paper, we described some unusual eigenvalue problems we have encountered. Some of those problems have been studied in literature and some are new. We are particularly interested in solving those associated with large sparse problems. Many existing techniques are only suitable for dense matrix computations and becomes inadequate for large sparse problems.

We will show that all these unusal eigenvalue problems can be converted to the problem of computing a quadratic form $u^T f(A)u$, for a properly defined matrix $A$, a vector $u$ and a function $f$. Numerical techniques for computing the quadratic form to be discussed in this paper will based on the work initially proposed in [6] and further developed in [11,12,2]. In this technique, we first transfer the problem of computing the quadratic form to a Riemann-Stieltjes integral problem, and then use Gauss-type quadrature rules to approximate the integral, which then brings the orthogonal polynomial theory and the underlying Lanczos procedure into the scene. This approach is well suitable for large sparse problems, since it references the matrix $A$ through a user provided subroutine to form the matrix-vector product $Ax$.

J. Palma, J. Dongarra, and V. Hernández (Eds.): VECPAR'98, LNCS 1573, pp. 4–19, 1999.

The basic time-consuming kernels for computing quadratic forms using parallelism are vector inner products, vector updates and matrix-vector products; this is similar to most iterative methods in linear algebra. Vector inner products and updates can be easily parallelized: each processor computes the vector-vector operations of corresponding segments of vectors (local vector operations (LVOs)), and if necessary, the results of LVOs have to sent to other processors to be combined for the global vector-vector operations. For the matrix-vector product, the user can either explore the particular structure of the matrix in question for parallelism, or split the matrix into strips corresponding to the vector segments. Each process then computes the matrix-vector product of one strip. Furthermore, the iterative loop of algorithms can be designed to overlap communication and computation and eliminating some of the synchronization points. The reader may see [8,4] and references therein for further details.

The rest of the paper is organized as follows. Section 2 describes some unusual eigenvalue problems and shows that these problems can be converted to the problem of computing a quadratic form. Section 3 reviews numerical methods for computing a quadratic form. Section 4 shows that how these numerical methods can be applied to those problems described in section 2. Some selected numerical examples are presented in section 5. Concluding remarks are in section 5.

## 2 Some Unusual Matrix Eigenvalue Problems

### 2.1 Constrained eigenvalue problem

Let $A$ be a real symmetric matrix of order $N$, and $c$ a given $N$ vector with $c^T c = 1$. We are interested in the following optimization problem

$$\max_x \quad x^T A x \tag{1}$$

subject to the constraints

$$x^T x = 1 \tag{2}$$

and

$$c^T x = 0. \tag{3}$$

Let

$$\phi(x, \lambda, \mu) = x^T A x - \lambda(x^T x - 1) + 2\mu x^T c, \tag{4}$$

where $\lambda, \mu$ are Lagrange multipliers. Differentiating (4) with respect to $x$, we are led to the equation

$$Ax - \lambda x + \mu c = 0.$$

Then

$$x = -\mu(A - \lambda I)^{-1} c.$$

Using the constraint (3), we have

$$c^T (A - \lambda I)^{-1} c = 0. \tag{5}$$

An equation of such type is referred as a *secular equation*. Now the problem becomes finding the largest $\lambda$ of the above secular equation.

We note that in [10], the problem is cast as computing the largest eigenvalue of the matrix $PAP$, where $P$ is a project matrix $P = I - cc^T$.

### 2.2 Modified eigenvalue problem

Let us consider solving the following eigenvalue problems

$$Ax = \lambda x$$

and

$$(A + cc^T)\bar{x} = \bar{\lambda}\bar{x}$$

where $A$ is a symmetric matrix and $c$ is a vector and without loss of generality, we assume $c^T c = 1$. The second eigenvalue problem can be regarded as a modifed or perturbed eigenvalue problem of the first one. We are interested in obtaining some, not all, of the eigenvalues of both problems. Such computation task often appears in structural dynamic (re-)analysis and other applications [5].

By simple algebraic derivation, it is easy to show that the eigenvalues $\bar{\lambda}$ of the second problem satisfy the following secular equation

$$1 + c^T(A - \bar{\lambda}I)^{-1}c = 0. \tag{6}$$

### 2.3 Constraint quadratic optimization

Let $A$ be a symmetric positive definite matrix of order $N$ and $c$ a given $N$ vector. The quadratic optimization problem is stated as the following:

$$\min_x \quad x^T Ax - 2c^T x \tag{7}$$

with the constraint

$$x^T x = \alpha^2, \tag{8}$$

where $\alpha$ is a given scalar. Now let

$$\phi(x, \lambda) = x^T Ax - 2c^T x + \lambda(x^T x - \alpha^2) \tag{9}$$

where $\lambda$ is the Lagrange multiplier. Differentiating (9) with respect to $x$, we are led to the equation

$$(A + \lambda I)x - c = 0$$

By the constraint (8), we are led to the problem of determining $\lambda > 0$ such that

$$c^T(A + \lambda I)^{-2}c = \alpha^2. \tag{10}$$

Furthermore, one can show the existence of a unique positive $\lambda^*$ for which the above equation is satisfied. The solution of the original problem (7) and (8) is then $x^* = (A + \lambda^* I)^{-1}c$.

### 2.4 Trace and determinant

The trace and determinant problems are simply to estimate the quantities

$$\mathrm{tr}(A^{-1}) = \sum_{i=1}^{n} e_i^T A^{-1} e_i$$

and

$$\det(A)$$

for a given matrix $A$. For the determinant problem, it can be easily verified that for a symmetric positive definite matrix $A$:

$$\ln(\det(A)) = \mathrm{tr}(\ln(A)) = \sum_{i=1}^{n} e_i^T (\ln(A)) e_i. \tag{11}$$

Therefore, the problem of estimating the determinant is essentially to estimate the trace of the matrix natural logarithm function $\ln(A)$.

### 2.5 Partial eigenvalue sum

The partial eigenvalue sum problem is to compute the sum of all eigenvalues less than a prescribed value $\alpha$ of the generalized eigenvalue problem

$$Ax = \lambda Bx, \tag{12}$$

where $A$ and $B$ are real $N \times N$ symmetric matrices with $B$ positive definite. Specifically, let $\{\lambda_i\}$ be the eigenvalues; one wants to compute the quantity

$$\tau_\alpha = \sum_{\lambda_i < \alpha} \lambda_i$$

for a given scalar $\alpha$.

Let $B = LL^T$ be Cholesky decomposition of $B$, the problem (12) is then equivalent to

$$(L^{-1} A L^{-T}) L^T x = \lambda L^T x.$$

Therefore the partial eigenvalue sum of the matrix pair $(A, B)$ is equal to the partial eigenvalue sum of the matrix $L^{-1} A L^{-T}$, which, in practice, does not need to be formed explicitly.

A number of approaches might be found in literature to solve such problem. Our approach will based on constructing a function $f$ such that the trace of $f(L^{-1} A L^{-T})$ approximates the desired sum $\tau_\alpha$. Specifically, one wants to construct a function $f$ such that

$$f(\lambda_i) = \begin{cases} \lambda_i, & \text{if } \lambda_i < \alpha \\ 0, & \text{if } \lambda_i > \alpha, \end{cases} \tag{13}$$

for $i = 1, 2, \ldots, N$. Then $\mathrm{tr}(f(L^{-1}AL^{-T}))$ is the desired sum $\tau_\alpha$. One of choices is to have the $f$ of the form

$$f(\zeta) = \zeta g(\zeta) \tag{14}$$

where

$$g(\zeta) = \frac{1}{1 + \exp\left(\frac{\zeta - \alpha}{\kappa}\right)},$$

where $\kappa$ is a constant. This function, among other names, is known as the Fermi-Dirac distribution function [15, p. 347]. In the context of a physical system, the usage of this distribution function is motivated by thermodynamics. It directly represents thermal occupancy of electronic states. $\kappa$ is proportional to the temperature of the system, and $\alpha$ is the chemical potential (the highest energy for occupied states).

It is easily seen that $0 < g(\zeta) < 1$ for all $\zeta$ with horizontal asymptotes 0 and 1. $(\alpha, \frac{1}{2})$ is the inflection point of $g$ and the sign of $\kappa$ determines whether $g$ is decreasing $(\kappa > 0)$ or increasing $(\kappa < 0)$. For our application, we want the sum of all eigenvalues less than $\alpha$, so we use $\kappa > 0$. The magnitude of $\kappa$ determines how "close" the function $g$ maps $\zeta < \alpha$ to 1 and $\zeta > \alpha$ to 0. As $\kappa \to 0^+$, the function $g(\zeta)$ rapidly converges to the step function $h(\zeta)$.

$$h(\zeta) = \begin{cases} 1 & \text{if} \quad \zeta < \alpha \\ 0 & \text{if} \quad \zeta > \alpha. \end{cases}$$

The graphs of the function $g(\zeta)$ for $\alpha = 0$ and different values of the parameter $\kappa$ are plotted in Figure 1.

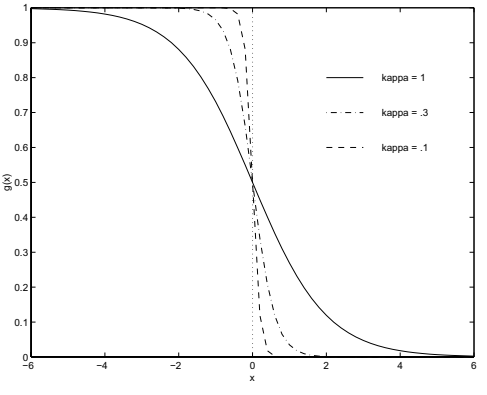


**Fig. 1.** Graphs of $g(\zeta)$ for different values of $\kappa$ where $\alpha = 0$.

With this choice of $f(\zeta)$, we have

$$\tau_\alpha = \sum_{\lambda_i < \alpha} \lambda_i \approx \mathrm{tr}(f(L^{-1}AL^{-T})) = \sum_{i=1}^{n} e_i^T f(L^{-1}AL^{-T}) e_i. \tag{15}$$

In summary, the problem of computing partial eigenvalue sum becomes computing the trace of $f(L^{-1}AL^{-T})$.

## 3 Quadratic Form Computing

As we have seen, all those unusual eigenvalue problems presented in section 2 can be summarized as the problem of computing the quadratic form $u^T f(A)u$, where $A$ is a $N \times N$ real matrix, and $u$ is a vector, and $f$ is a proper defined function. One needs to find an approximate of the quantity $u^T f(A)u$, or give a lower bound $\ell$ and/or an upper bound $\nu$ of it. Without loss of generality, one may assume $u^T u = 1$.

The quadratic form computing problem is first proposed in [6] for bounding the error of CG method for solving linear system of equations. It has been further developed in [11,12,2] and extended to other applications. The main idea is to first transform the problem of the quadratic form computing to a Riemann-Stieltjes integral problem, and then use Gauss-type quadrature rules to approximate the integral, which then brings the orthogonal polynomial theory and the underlying Lanczos procedure into the picture.

Let us go through the main idea. Since $A$ is symmetric, the eigen-decomposition of $A$ is given by $A = Q^T \Lambda Q$, where $Q$ is an orthogonal matrix and $\Lambda$ is a diagonal matrix with increasingly ordered diagonal elements $\lambda_i$. Then we have

$$u^T f(A)u = u^T Q^T f(\Lambda) Q u = \tilde{u}^T f(\Lambda)\tilde{u} = \sum_{i=1}^{N} f(\lambda_i)\tilde{u}_i^2,$$

where $\tilde{u} = (\tilde{u}_i) \equiv Qu$. The last sum can be considered as a Riemann-Stieltjes integral

$$u^T f(A)u = \int_a^b f(\lambda) d\mu(\lambda),$$

where the measure $\mu(\lambda)$ is a piecewise constant function and defined by

$$\mu(\lambda) = \begin{cases} 0, & \text{if } \lambda < a \le \lambda_1, \\ \sum_{j=1}^{i} \tilde{u}_j^2, & \text{if } \lambda_i \le \lambda < \lambda_{i+1} \\ \sum_{j=1}^{N} \tilde{u}_j^2 = 1, & \text{if } b \le \lambda_N \le \lambda \end{cases}$$

and $a$ and $b$ are the lower and upper bounds of the eigenvalues $\lambda_i$.

To obtain an estimate for the Riemann-Stieltjes integral, one can use the Gauss-type quadrature rule [9,7]. The general quadrature formula is of the form

$$I[f] = \sum_{j=1}^{n} \omega_j f(\theta_j) + \sum_{k=1}^{m} \rho_k f(\tau_k), \tag{16}$$

where the weights $\{\omega_j\}$ and $\{\rho_k\}$ and the nodes $\{\theta_j\}$ are unknown and to be determined. The nodes $\{\tau_k\}$ are prescribed. If $m = 0$, then it is the well-known

Gauss rule. If $m = 1$ and $\tau_1 = a$ or $\tau_1 = b$, it is the Gauss-Radau rule. The Gauss-Lobatto rule is for $m = 2$ and $\tau_1 = a$ and $\tau_2 = b$.

The accuracy of the Gauss-type quadrature rules may be obtained by an estimation of the remainder $R[f]$:

$$R[f] = \int_a^b f(\lambda) d\mu(\lambda) - I[f].$$

For example, for the Gauss quadrature rule,

$$R[f] = \frac{f^{(2n)}(\eta)}{(2n)!} \int_a^b \left[ \prod_{i=1}^{n} (\lambda - \theta_i) \right]^2 d\mu(\lambda),$$

where $a < \eta < b$. Similar formulas exist for Gauss-Radau and Gauss-Lobatto rules. If the sign of $R[f]$ is determined, then the quadrature formula $I[f]$ is a lower bound (if $R[f] > 0$) or an upper lower bound (if $R[f] < 0$) of the quantity $u^T f(A)u$.

Let us briefly recall how the weights and the nodes in the quadrature formula are obtained. First, we know that a sequence of polynomials $p_0(\lambda), p_1(\lambda), p_2(\lambda), \ldots$ can be defined such that they are orthonormal with respect to the measure $\mu(\lambda)$:

$$\int_a^b p_i(\lambda) p_j(\lambda) d\mu(\lambda) = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases}$$

where it is assumed that the normalization condition $\int d\mu = 1$ (i.e., $u^T u = 1$). The sequence of orthonormal polynomials $\pi_j(\lambda)$ satisfies a three-term recurrence

$$\gamma_j p_j(\lambda) = (\lambda - \alpha_j) p_{j-1}(\lambda) - \gamma_{j-1} p_{j-2}(\lambda),$$

for $j = 1, 2, \ldots, n$ with $p_{-1}(\lambda) \equiv 0$ and $p_0(\lambda) \equiv 1$. Writing the recurrence in matrix form, we have

$$\lambda p(\lambda) = T_n p(\lambda) + \gamma_n p_n(\lambda) e_n$$

where

$$p(\lambda)^T = [p_0(\lambda), p_1(\lambda), \ldots, p_{n-1}(\lambda)], \qquad e_n^T = [0, 0, \ldots, 1]$$

and

$$T_n = \begin{pmatrix} \alpha_1 & \gamma_1 & & & & \\ \gamma_1 & \alpha_2 & \gamma_2 & & & \\ & \gamma_2 & \alpha_3 & \ddots & & \\ & & \ddots & \ddots & \ddots & \\ & & & \ddots & \ddots & \gamma_{n-1} \\ & & & & \gamma_{n-1} & \alpha_n \end{pmatrix}.$$

Then for the Gauss quadrature rule, the eigenvalues of $T_n$ (which are the zeros of $p_n(\lambda)$) are the nodes $\theta_j$. The weights $\omega_j$ are the squares of the first elements of the normalized (i.e., unit norm) eigenvectors of $T_n$.

For the Gauss-Radau and Gauss-Lobatto rules, the nodes $\{\theta_j\}$, $\{\tau_k\}$ and weights $\{\omega_j\}, \{\rho_j\}$ come from eigenvalues and the squares of the first elements of the normalized eigenvectors of an adjusted tridiagonal matrices of $\tilde{T}_{n+1}$, which has the prescribed eigenvalues $a$ and/or $b$.

To this end, we recall that the classical Lanczos procedure is an elegant way to compute the orthonormal polynomials $\{p_j(\lambda)\}$ [16,11]. We have the following algorithm in summary form. We refer it as the Gauss-Lanczos (GL) algorithm.

**GL algorithm:** Let $A$ be a $N \times N$ real symmetric matrix, $u$ a real $N$ vector with $u^T u = 1$. $f$ is a given smooth function. Then the following algorithm computes an estimation $I_n$ of the quantity $u^T f(A)u$ by using the Gauss rule with $n$ nodes.

- Let $x_0 = u$, and $x_{-1} = 0$ and $\gamma_0 = 0$
- For $j = 1, 2, \ldots, n$,
  1. $\alpha_j = x_{j-1}^T A x_{j-1}$
  2. $v_j = A x_{j-1} - \alpha_j x_{j-1} - \gamma_{j-1} x_{j-2}$
  3. $\gamma_j = \|v_j\|_2$
  4. $x_j = r_j / \gamma_j$
- Compute eigenvalues $\theta_k$ and the first elements $\omega_k$ of eigenvectors of $T_n$
- Compute $I_n = \sum_{k=1}^{n} \omega_k^2 f(\theta_k)$

We note that the "For" loop in the above algorithm is an iteration step of the standard symmetric Lanczos procedure [16]. The matrix $A$ in question is only referenced here in the form of the matrix-vector product. The Lanczos procedure can be implemented with only 3 $n$-vectors in the fast memory. This is the major storage requirement for the algorithm and is an attractive feature for large scale problems.

On the return of the algorithm, from the expression of $R[f]$, we may estimate the error of the approximation $I_n$. For example, if $f^{(2n)}(\eta) > 0$ for any $n$ and $\eta$, $a < \eta < b$, then $I_n$ is a lower bound $\ell$ of the quantity $u^T f(A)u$.

**Gauss-Radau-Lanczos (GRL) algorithm:** To implement the Gauss-Radau rule with the prescribed node $\tau_1 = a$ or $\tau_1 = b$, the above GL algorithm just needs to be slightly modified. For example, with $\tau_1 = a$, we need to extend the matrix $T_n$ to

$$\tilde{T}_{n+1} = \begin{bmatrix} T_n & \gamma_n e_n \\ \gamma_n e_n^T & \phi \end{bmatrix}.$$

Here the parameter $\phi$ is chosen such that $\tau_1 = a$ is an eigenvalue of $\tilde{T}_{n+1}$. From [10], it is known that

$$\phi = a + \delta_n,$$

where $\delta_n$ is the last component of the solution $\delta$ of the tridiagonal system

$$(T_n - aI)\delta = \gamma_n^2 e_n.$$

Then the eigenvalues and the first components of eigenvectors of $\tilde{T}_{n+1}$ gives the nodes and weight of the Gauss-Radau rule to compute an estimation $\tilde{I}_n$ of $u^T f(A)u$.

Furthermore, if $f^{(2n+1)}(\eta) < 0$ for any $n$ and $\eta$, $a < \eta < b$, then $\tilde{I}_n$ (with $b$ as a prescribed eigenvalue of $\tilde{T}_{n+1}$) is a lower bound $\ell$ of the quantity $u^T f(A)u$. $\tilde{I}_n$ (with $a$ as a prescribed eigenvalue of $\tilde{T}_{n+1}$) is an upper bound $\nu$.

**Gauss-Lobatto-Lanczos (GLL) algorithm:** To implement the Gauss-Lobatto rule, $T_n$ computed in the GL algorithm is updated to

$$\hat{T}_{n+1} = \begin{bmatrix} T_n & \psi e_n \\ \psi e_n^T & \phi \end{bmatrix}.$$

Here the parameters $\phi$ and $\psi$ are chosen so that $a$ and $b$ are eigenvalues of $\hat{T}_{n+1}$. Again, from [10], it is known that

$$\phi = \frac{\delta_n b - \mu_n a}{\delta_n - \mu_n} \qquad \text{and} \qquad \psi^2 = \frac{b+a}{\delta_n - \mu_n},$$

where $\delta_n$ and $\mu_n$ are the last components of the solutions $\delta$ and $\mu$ of the tridiagonal systems

$$(T_n - aI)\delta = e_n \qquad \text{and} \qquad (T_n - bI)\mu = e_n.$$

The eigenvalues and the first components of eigenvectors of $\hat{T}_{n+1}$ gives the nodes and weight of the Gauss-Lobatto rule to compute an estimation $\hat{I}_n$ of $u^T f(A)u$. Moreover, if $f^{(2n)}(\eta) > 0$ for any $\eta$, $a < \eta < b$, then $\hat{I}_n$ is an upper bound $\nu$ of the quantity $u^T f(A)u$.

Finally, we note that we need not always compute the eigenvalues and the first components of eigenvectors of the tridiagonal matrix $T_n$ or its modifications $\tilde{T}_{n+1}$ or $\hat{T}_{n+1}$ for obtaining the estimation $I_n$ or $\tilde{I}_n$, $\hat{I}_n$. We have following proposition.

**Proposition 1.** *For Gaussian rule:*

$$I_n = \sum_{k=1}^{n} \omega_k^2 f(\theta_k) = e_1^T f(T_n) e_1. \tag{17}$$

*For Gauss-Radau rule:*

$$\tilde{I}_n = \sum_{k=1}^{n} \omega_k^2 f(\theta_k) + \rho_1 f(\tau_1) = e_1^T f(\tilde{T}_{n+1}) e_1. \tag{18}$$

*For Gauss-Lobatto rule:*

$$\hat{I}_n = \sum_{k=1}^{n} \omega_k^2 f(\theta_k) + \rho_1 f(\tau_1) + \rho_2 f(\tau_2) = e_1^T f(\hat{T}_{n+1}) e_1. \tag{19}$$

Therefore, if the (1,1) entry of $f(T_n)$, $f(\tilde{T}_{n+1})$ or $f(\hat{T}_{n+1})$ can be easily computed, for example, $f(\lambda) = 1/\lambda$, we do not need to compute the eigenvalues and eigenvectors.

## 4 Solving the UEPs by Quadratic Form Computing

In this section, we use the GL, GRL and GLL algorithms for solving those unusual eigenvalue problems discussed in section 2.

**Constraint eigenvalue problem** Using the GL algorithm with the matrix $A$ and the vector $c$, we have

$$c_1^T(A-\lambda I)^{-1}c = e_1^T(T_n-\lambda I)^{-1}e_1 + R,$$

where $R$ is the remainder. Now we may solve reduced-order secular equation

$$e_1^T(T_n-\lambda I)^{-1}e_1 = 0$$

to find the largest $\lambda$ as the approximate solution of the problem. This secular equation can be solved using the method discussed in [17] and its implementation available in LAPACK [1].

**Modified eigenvalue problem** Again, using the GL algorithm with the matrix $A$ and the vector $c$, we have

$$1 + c^T(A-\bar{\lambda}I)^{-1}c = 1 + e_1^T(T_n-\bar{\lambda}I)^{-1}e_1 + R,$$

where $R$ is the remainder. Then we may solve the eigenvalue problem of $T_n$ to approximate some eigenvalues of $A$, and then solve reduced-order secular equation

$$1 + e_1^T(T_n-\bar{\lambda}I)^{-1}e_1 = 0$$

for $\bar{\lambda}$ to find some approximate eigenvalues of the modified eigenvalue problem.

**Constraint quadratic programming** By using the GRL algorithm with the prescribed node $\tau_1 = b$ for the matrix $A$ and vector $c$, it can be shown that

$$c^T(A+\lambda I)^{-2}c \geq e_1^T(\tilde{T}_{n+1}+\lambda I)^{-2}e_1$$

for all $\lambda > 0$. Then by solving the reduced-order secular equation

$$e_1^T(\tilde{T}_{n+1}+\lambda I)^{-2}e_1 = \alpha^2$$

for $\lambda$, we obtain $\underline{\lambda}_n$, which is a lower bound of the solution $\lambda^*$: $\underline{\lambda}_n \leq \lambda^*$

On the other hand, using the GRL algorithm with the prescribed node $\tau_1 = a$, we have

$$c^T(A+\lambda I)^{-2}c \leq e_1^T(\tilde{T}_{n+1}+\lambda I)^{-2}e_1$$

for all $\lambda > 0$. Then by solving the reduced-order secular equation

$$e_1^T(\tilde{T}_{n+1}+\lambda I)^{-2}e_1 = \alpha^2$$

for $\lambda$. We have an upper bound $\bar{\lambda}_n$ of the solution $\lambda^*$: $\bar{\lambda}_n \geq \lambda^*$.

Using such two-sided approximation as illustrated in Figure 2, the iteration can be adaptively proceeded until the estimations $\underline{\lambda}_n$ and $\bar{\lambda}_n$ are sufficiently close, we then obtain an approximation

$$\lambda^* \approx \frac{1}{2}(\underline{\lambda}_n + \bar{\lambda}_n)$$

of the desired solution $\lambda^*$.

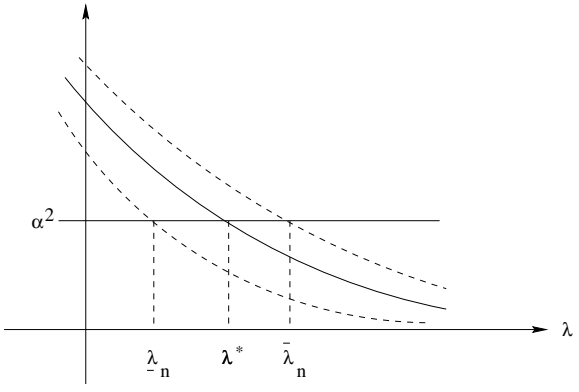


**Fig. 2.** Two-sided approximation approximation of the solution $\lambda^*$ for the constraint quadratic programming problem (7) and (8).

**Trace, determinant and partial eigenvalue sum** As shown in sections 2.4 and 2.5, the problems of computing trace of the inverse of a matrix $A$, determinant of a matrix $A$ and partial eigenvalue sum of a symmetric positive definite pair $(A, B)$ can be summarized as the problem of computing the trace of a corresponding matrix function $f(H)$, where $H = A$ or $H = L^{-1}AL^{-T}$ and $f(\lambda) = 1/\lambda$, $\ln(\lambda)$ or $\lambda/\left(1+\exp(\frac{\lambda-\alpha}{\kappa})\right)$. To efficiently compute the trace of $f(H)$, instead of applying GR algorithm or its variations $N$ times for each diagonal element of $f(H)$, we may use a Monte Carlo approach which only applies the GR algorithm $m$ times to obtain an unbiased estimation of $\mathrm{tr}(f(H))$. For practical purposes, $m$ can be chosen much smaller than $N$. The saving in computational costs could be significant. Such a Monte Carlo approach is based on the following lemma due to Hutchinson [14].

**Proposition 2.** *Let $C = (c_{ij})$ be an $N \times N$ symmetric matrix with $tr(C) \neq 0$. Let $\mathcal{V}$ be the discrete random variable which takes the values 1 and $-1$ each with probability 0.5 and let $z$ be a vector of $n$ independent samples from $\mathcal{V}$. Then $z^T Cz$ is an unbiased estimator of $tr(C)$, i.e.,*

$$E(z^T Cz) = tr(C),$$

*and*

$$var(z^T Cz) = 2\sum_{i\neq j} c_{ij}^2.$$

To use the above proposition in practice, one takes $m$ such sample vectors $z_i$, and then uses GR algorithm or its variations to obtain an estimation $I_n^{(i)}$, a lower bound $\ell_n^{(i)}$ and/or an upper bound $\nu_n^{(i)}$ of the quantity $z_i^T f(H) z_i$:

$$\ell_n^{(i)} \leq z_i^T f(H) z_i \leq \nu_n^{(i)}.$$

Then by taking the mean of the $m$ computed estimation $I_n^{(i)}$ or lower and upper bounds $\ell_n^{(i)}$ and $\nu_n^{(i)}$, we have

$$\mathrm{tr}(f(H)) \approx \frac{1}{m}\sum_{i=1}^{m} I_n^{(i)}$$

or

$$\frac{1}{m}\sum_{i=1}^{m} \ell_n^{(i)} \leq \frac{1}{m}\sum_{i=1}^{m} z_i^T f(H) z_i \leq \frac{1}{m}\sum_{i=1}^{m} \nu_n^{(i)}.$$

It is natural to expect that with a suitable sample size $m$, the mean of the computed bounds yields a satisfactory estimation of the quantity $\mathrm{tr}(f(H))$. To assess the quality of such estimation, one can also obtain probabilistic bounds of the approximate value [2].

## 5 Numerical Examples

In this section, we present some numerical examples to illustrate our quadratic form based algorithms for solving some of the unusual eigenvalue problems discussed in section 2.

**Table 1.** Numerical Results of estimating $\mathrm{tr}(A^{-1})$

| Matrix | $N$ | "Exact" | Iter | Estimated | Rel.err |
|---|---|---|---|---|---|
| Poisson | 900 | $5.126e+02$ | 30–50 | $5.020e+02$ | 2.0% |
| VFH | 625 | $5.383e+02$ | 12–21 | $5.366e+02$ | 0.3% |
| Wathen | 481 | $2.681e+01$ | 33–58 | $2.667e+01$ | 0.5% |
| Lehmer | 200 | $2.000e+04$ | 38–70 | $2.017e+04$ | 0.8% |

**Table 2.** Numerical results of estimating $\ln(\det(A)) = \mathrm{tr}(\ln A)$

| Matrix | $N$ | "Exact" | Iter | Estimated | Rel.err |
|---|---|---|---|---|---|
| Poisson | 900 | $1.065e+03$ | 11–29 | $1.060e+03$ | 0.4% |
| VFH | 625 | $3.677e+02$ | 10–14 | $3.661e+02$ | 0.4% |
| Heat Flow | 900 | $5.643e+01$ | 4 | $5.669e+01$ | 0.4% |
| Pei | 300 | $5.707e+00$ | 2–3 | $5.240e+00$ | 8.2% |

### 5.1 Trace and determinant

Numerical results for a set of test matrices presented in Tables 1 and 2 are first reported in [2]. Some of these test matrices are model problems and some are from practical applications. For example, VFH matrix is from the analysis of transverse vibration of a Vicsek fractal. These numerical experiments are carried out on an Sun Sparc workstation. The so-called "exact" value is computed by using the standard methods for dense matrices. The numbers in the "Iter"-column are the number of iterations $n$ required for the estimation $I_n^{(i)}$ to reach stationary value within the given tolerance value $tol = 10^{-4}$, namely,

$$|I_n - I_{n-1}| \leq tol * |I_n|.$$

The number of random sample vector $z_i$ is $m = 20$. For those test matrices, the relative accuracy of the new approach within 0.3% to 8.2% may be sufficient for practical purposes.

### 5.2 Partial eigenvalue sum

Here we present a numerical example from the computation of the total energy of an electronic structure. Total energy calculation of a solid state system is necessary in simulating real materials of technological importance [18]. Figure 3 shows a carbon cluster that forms part of a "knee" structure connecting nanotubes of different diameters and the distribution of eigenvalues such carbon structure with 240 atoms. One is interested in computing the sum of all these eigenvalues less than zero. Comparing the performance of our method with dense methods, namely symmetric QR algorithm and bisection method in LAPACK, our method achieved up to a factor of 20 speedup for large system on an Convex Exemplar SPP-1200 (see Table 3). Because of large memory requirements, we were not able to use LAPACK divide-and-conquer symmetric eigenroutines. Furthermore, algorithms for solving large-sparse eigenvalue problems, such as Lanczos method or implicitly restarted methods for computing some eigenvalues are found inadequate due to large number of eigenvalues required. Since the

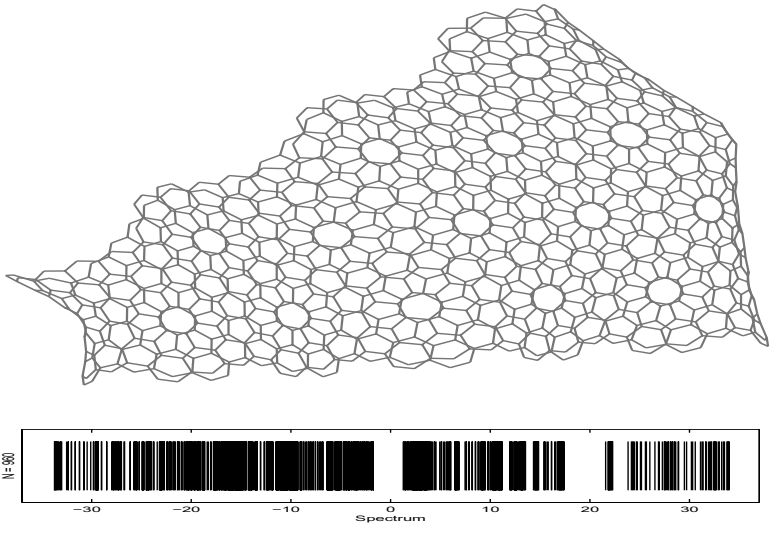


**Fig. 3.** A carbon cluster that forms part of a "knee" structure, and the corresponding spectrum

**Table 3.** Performance of our method vs. dense methods on Convex Exemplar SPP-1200. Here, 10 Monte Carlo samples were used to obtain estimates for each systems size.

| | | Dense methods | | | GR Algorithm | | % Relative |
|---|---|---|---|---|---|---|---|
| $n$ | $m$ | Partial Sum | QR Time | BI Time | Estimate | Time | Error |
| 480 | 349 | -4849.8 | 7.4 | 7.6 | -4850.2 | 2.8 | 0.01 |
| 960 | 648 | -9497.6 | 61.9 | 51.8 | -9569.6 | 18.5 | 0.7 |
| 1000 | 675 | -9893.3 | 80.1 | 58.6 | -10114.1 | 22.4 | 2.2 |
| 1500 | 987 | -14733.1 | 253.6 | 185.6 | -14791.8 | 46.4 | 0.4 |
| 1920 | 1249 | -18798.5 | 548.3 | 387.7 | -19070.8 | 72.6 | 1.4 |
| 2000 | 1299 | -19572.9 | 616.9 | 431.8 | -19434.7 | 78.5 | 0.7 |
| 2500 | 1660 | -24607.6 | 1182.2 | 844.6 | -24739.6 | 117.2 | 0.5 |
| 3000 | 1976 | -29471.3 | 1966.4 | 1499.7 | -29750.9 | 143.5 | 0.9 |
| 3500 | 2276 | -34259.5 | 3205.9 | 2317.4 | -33738.5 | 294.0 | 1.5 |
| 4000 | 2571 | -39028.9 | 4944.3 | 3553.2 | -39318.0 | 306.0 | 0.7 |
| 4244 | 2701 | -41299.2 | 5915.4 | 4188.0 | -41389.8 | 339.8 | 0.2 |

problem is required to be solved repeatly, we are now able to solve previously intractable large scale problems. The relative accuracy of new approach within 0.4% to 1.5% is satisfactory for the application [3].

## 6 Concluding Remarks

In this paper, we have surveyed numerical techniques based on computing quadratic forms for solving some unusual eigenvalue problems. Although there exist some numerical methods for solving these problems (see [13] and references therein), most of these can be applied only for small and/or dense problems. The techniques presented here reference the matrix in question only through a matrix-vector product operation. Hence, they are more suitable for large sparse problems.

The new approach deserves further study; in particular, for error estimation and convergence of the methods. An extensive comparative study of the trade-offs in accuracy and computational costs between the new approach and other existing methods should be conducted.

**Acknowledgement** Z. B. was supported in part by an NSF grant ASC-9313958, an DOE grant DE-FG03-94ER25219.

## References

1. Anderson, E., Bai, Z., Bischof, C., Demmel, J., Dongarra, J., Du Croz, J., Greenbaum, A., Hammarling, S., McKenney, A., Ostrouchov, S., Sorensen, D.:, LAPACK Users' Guide (second edition), SIAM, Philadelphia, 1995. 13
2. Bai, Z., Fahey, M., Golub, G.: Some large-scale matrix computation problems. J. Comp. Appl. Math. **74** (1996) 71–89. 4, 9, 15, 16
3. Bai, Z., Fahey, M., Golub, G., Menon, M., Richter, E.: Computing partial eigenvalue sum in electronic structure calculations, Scientific Computing and Computational Mathematics Program, Computer Science Dept., Stanford University, SCCM-98-03, 1998. 18
4. Barrett. R., Berry. M., Chan. F., Demmel. J., Donato. J., Dongarra. J., Eijkhout. V., Pozo. R., Romine. C., van der Vorst., H.: Templates for the solution of linear systems: Building blocks for iterative methods. SIAM, Philadelphia, 1994. 5
5. Carey, C., Golub, G., Law, K.: A Lanczos-based method for structural dynamic reanalysis problems. Inter. J. Numer. Methods in Engineer., 37 (1994) 2857–2883. 6
6. Dahlquist, G., Eisenstat, S., Golub, G.: Bounds for the error of linear systems of equations using the theory of moments. J. Math. Anal. Appl. 37 (1972) 151–166. 4, 9
7. Davis. P., Rabinowitz. P.: Methods of Numerical Integration. Academic Press, New York, 1984. 9
8. Demmel, J., Heath. M., van der Vorst., H.: Parallel linear algebra, in Acta Numerica, Vol.2, Cambridge Press, New York, 1993 5

9. Gautschi. W., A survey of Gauss-Christoffel quadrature formulae. In P. L Bultzer and F. Feher, editors, *E. B. Christoffel – the Influence of His Work on on Mathematics and the Physical Sciences*, pages 73–157. Birkhauser, Boston, 1981. 9
10. Golub, G.: Some modified matrix eigenvalue problems. SIAM Review, 15 (1973) 318–334. 6, 11, 12
11. Golub, G., Meurant, G.: Matrices, moments and quadrature, in Proceedings of the 15th Dundee Conference, June 1993, D. F. Griffiths and G. A. Watson, eds., Longman Scientific & Technical, 1994. 4, 9, 11
12. Golub, G., Strakoš, Z.: Estimates in quadratic formulas, Numerical Algorithms, 8 (1994) 241–268. 4, 9
13. Golub. G., Van Loan. C.: Matrix Computations. Johns Hopkins University Press, Baltimore, MD, third edition, 1996. 18
14. Hutchinson, M.: A stochastic estimator of the trace of the influence matrix for Laplacian smoothing splines, Commun. Statist. Simula., 18 (1989) 1059–1076. 14
15. Kerstin. K., Dorman. K. R.: A Course in Statistical Thermodynamics, Academic Press, New York, 1971. 8
16. Lanczos, C.: An iteration method for the solution of the eigenvalue problem of linear differential and integral operators, J. Res. Natl. Bur. Stand, 45 (1950) 225–280. 11, 11
17. Li., R.-C.: Solving secular equations stably and efficiently, Computer Science Division, Department of EECS, University of California at Berkeley, Technical Report UCB//CSD-94-851,1994 13
18. Menon. M., Richter. E., Subbaswamy. K. R.: Structural and vibrational properties of fullerenes and nanotubes in a nonorthogonal tight-binding scheme. J. Chem. Phys., 104 (1996) 5875–5882. 16