

Computing Traces of Functions of Matrices

Hongbin Guo*

5/7/2000 final revised

Abstract

For large scale matrix, computing the trace of function of the matrix is expensive, recently an effective algorithm based on Monte Carlo method and Gauss type quadrature rule was developed. In this paper, we analyse condition number of this problem, estimate the error of the algorithm and give a modified algorithm. In our algorithm the relative error is controllable.

Keywords : Quadratic form; Lanczos procedure; Gauss type quadrature rule; Monte Carlo simulation; Condition number.

AMS(MOS) subject classifications : 65F10

1 Introduction

In some applications such as computational physics, one often computes determinant of matrix and trace of function of matrix. For some functions such as $f(x) = 1/x$ or $f(x) = \ln(x)$ computing $tr(f(A))$, i.e. $tr(A^{-1})$ or $\ln(\det(A))$ respectively, may be highly sensitive problems. When the matrix size n is small, we can compute these problems explicitly by using dense matrix computation methods [6]. General speaking, such methods require $\mathcal{O}(n^3)$ floating point operations. However, when n becomes large, it is impractical to compute trace explicitly. By Monte Carlo simulation (MC method) one can reduce these problems to quadratic form $\mathbf{u}^T f(A) \mathbf{u}$, where \mathbf{u} is a n -vectors. Computing the quadratic form was discussed and a practical algorithm was presented in [1]. Although the MC method suffers from low convergency, for a practical low precision requirement, the algorithm saves computational cost significantly compared with conventional algorithms. As for the quadratic form, from views of storage and work cost the algorithm in [1] has obvious advantage over the explicit method. The algorithm is especially suitable for large scale sparse situation. The former mentioned advantages draw us attention to study a more efficient algorithm for these problems.

The rest of this paper is organized as following. In section 2, we review the background and algorithm in [1]. In section 3, we analyse the condition number of these problems. In section 4, we estimate the error of the algorithm. In section 5, we develop a more efficient algorithm, the relative error can be control in the modified algorithm. some numerical examples are presented in section 6. section 7 contains conclusions.

*Institute of Mathematics, Fudan University, Shanghai, P.R. of China 200433.

2 Algorithm

In this section we review some known results and algorithms in[1]. Assume matrix A is real symmetric positive definite, f is a smooth function, we first show how to reduce $tr(f(A))$ to quadratic form by MC simulation , then transform the quadratic form to Riemann-Stieltjes integral and compute the integral by Gauss type quadrature rule.

To introduce the MC method for computing $tr(f(A))$, we list the following theorem:

Theorem 2.1 *Let H be an $n \times n$ symmetric matrix with $tr(H) \neq 0$. Let V be the discrete random variable which takes the values 1 and -1 each with probability 0.5 and let \mathbf{z} be a vector of n independent samples from V . Then $\mathbf{z}^T H \mathbf{z}$ is an unbiased estimator of $tr(H)$, i.e.*

$$E(\mathbf{z}^T H \mathbf{z}) = tr(H)$$

and variance

$$var(\mathbf{z}^T H \mathbf{z}) = 2 \sum_{i \neq j} h_{ij}^2$$

As described in the theorem, we can take m sample vectors \mathbf{z}_i and compute $\mathbf{z}_i^T H \mathbf{z}_i$ to compose the approximation of $tr(H)$ with $m \ll n$.

Next we discuss the quadratic form $\mathbf{u}^T f(A) \mathbf{u}$, where matrix A is a symmetric positive matrix (S.P.D., i.e. $A > 0$). Assume $A = Q^T \Lambda Q$, Q is an orthogonal matrix and Λ is a diagonal matrix with increasingly ordered diagonal elements $\lambda_i \in [a, b]$. Reduce $\mathbf{u}^T f(A) \mathbf{u}$ to the Riemann-Stieltjes integral

$$I[f] = \mathbf{u}^T f(A) \mathbf{u} = \mathbf{u}^T Q^T f(\Lambda) Q \mathbf{u} = \tilde{\mathbf{u}}^T f(\Lambda) \tilde{\mathbf{u}} = \sum_{i=1}^n f(\lambda_i) \tilde{u}_i^2 = \int_a^b f(\lambda) d\mu(\lambda)$$

where

$$\mu(\lambda) = \begin{cases} 0 & a \leq \lambda < \lambda_1. \\ \sum_{j=1}^i \tilde{u}_j^2 & \lambda_i \leq \lambda < \lambda_{i+1} \\ \sum_{j=1}^n \tilde{u}_j^2 & \lambda_n \leq \lambda \leq b \end{cases}$$

using Gauss-type quadrature rules [3, 5],we have

$$I[f] = \sum_{j=1}^N \omega_j f(\theta_j) + \sum_{l=1}^M \nu_l f(\tau_l) + R[f], \quad (2.1)$$

where the weights ω_j and ν_l and the nodes θ_j are unknown and to be determined. The nodes τ_l are prescribed, the error is denoted by $R[f]$. We are only interested in the case $M = 0$ and $M = 1$. If $M = 0$,it is the well-known Gauss rule. If $M = 1$ and $\tau_1 = a$ or $\tau_1 = b$, it is the Gauss-Radau rule.

Case 1: $M = 0$

To determine ω_j, ν_l and θ_j , we construct a sequence of orthogonal polynomials $\{p_k(\lambda)\}_{k=0}^N$.

$$\int_a^b d\mu = 1, \quad (2.2)$$

$$\int_a^b p_i(\lambda) p_j(\lambda) d\mu(\lambda) = \delta_{ij}, \quad (2.3)$$

$$\gamma_k p_k(\lambda) = (\lambda - \alpha_k) p_{k-1}(\lambda) - \gamma_{k-1} p_{k-2}(\lambda) \quad k = 1, 2, \dots, N. \quad (2.4)$$

with $p_{-1}(\lambda) \equiv 0$ and $p_0 \equiv 1$. Writing recurrence (2.4) in matrix form, we have:

$$\lambda \mathbf{p}(\lambda) = T_N \mathbf{p}(\lambda) + \gamma_N p_N \mathbf{e}_N$$

where

$$\mathbf{p}^T(\lambda) = [p_0(\lambda), p_1(\lambda), \dots, p_{N-1}(\lambda)], \quad \mathbf{e}_N^T = [0, 0, \dots, 1]$$

and

$$T_N = \begin{pmatrix} \alpha_1 & \gamma_1 & & & \\ \gamma_1 & \alpha_2 & \gamma_2 & & \\ & \ddots & \ddots & \ddots & \\ & & & \gamma_{N-2} & \alpha_{N-1} & \gamma_{N-1} \\ & & & & \gamma_{N-1} & \alpha_N \end{pmatrix} \quad (2.5)$$

The matrix T_N can also be derived by Lanczos procedure of matrix A with initial vector $\mathbf{x}_1 = \mathbf{u} / \|\mathbf{u}\|_2$. In Gauss quadrature rule, the nodes $\{\theta_j\}$ are the eigenvalues of T_N , i.e. the zeros of $p_N(\lambda)$. The weights $\{\omega_j\}$ are the squares of the first element of the normalized eigenvectors of T_N [3].

Case 2: $M = 1$ and $\tau_1 = a$ or b

Gauss-Radau rule, we need to construct the polynomial $p_{N+1}(\lambda)$ such that $p_{N+1}(a) = 0$ or $p_{N+1}(b) = 0$ respectively, i.e. we need to extend the matrix T_N to

$$\tilde{T}_{N+1} = \begin{bmatrix} T_N & \gamma_N \mathbf{e}_N \\ \gamma_N \mathbf{e}_N^T & \phi \end{bmatrix}$$

where the parameter ϕ is chosen such that a or b is an eigenvalue of \tilde{T}_{N+1} . From [5] we know that $\phi = a + \delta_N$ or $\phi = b + \delta_N$, where δ_N is the last component of the solution δ of the tridiagonal system

$$(T_N - aI)\delta = \gamma_N^2 \mathbf{e}_N \quad \text{or} \quad (T_N - bI)\delta = \gamma_N^2 \mathbf{e}_N.$$

Algorithm 1 Let $A \in R^{n \times n}$, $A > 0$, $\lambda(A) \subset [a, b]$, f be a smooth function on $[a, b]$, $\mathbf{u} \in R^n$. ε is the convergence tolerance. This algorithm computes $\mathbf{u}^T f(A) \mathbf{u}$ and if the sign of $f^{(2N+1)}(x)$ does not change on $x \in [a, b]$ we estimate the lower bound L and upper bound U of $\mathbf{u}^T f(A) \mathbf{u}$.

1. $\mathbf{x}_1 = \mathbf{u} / \|\mathbf{u}\|_2$, $\gamma_0 \mathbf{x}_0 = 0$
2. for $j = 1, 2, \dots$,
 - (a) Compute T_j by Lanczos procedure on A .
 - (b) Compute eigenvalues θ_i and the first elements ζ_i of eigenvectors of T_j , form $I_j = \sum_{i=1}^j \zeta_i^2 f(\theta_i)$.
 - (c) If $\gamma_j = 0$ or $|I_j - I_{j-1}| < \varepsilon |I_j|$, break.
- end for
3. Denote $N=j$, adjust T_N to \tilde{T}_{N+1} for a and b respectively.
4. Compute eigenvalues θ'_i and the first elements ζ'_i of eigenvectors of \tilde{T}_{N+1} for a, b respectively. Using Gauss-Radau rule we obtain lower bound L_N and upper bound U_N .

$$5. L = \|\mathbf{u}\|_2^2 L_N; \quad U = \|\mathbf{u}\|_2^2 U_N; \quad I = \|\mathbf{u}\|_2^2 I_N.$$

Remark 2.1 Estimate the remainder of Gauss-type quadrature rule , we have:

$$\begin{aligned} R[f] &= \frac{f^{(2N)}(\eta)}{(2N)!} \int_a^b \left[\prod_{i=1}^N (\lambda - \theta_i) \right]^2 d\mu(\lambda) && \text{for } M = 0, \\ R[f] &= \frac{f^{(2N+1)}(\eta)}{(2N+1)!} \int_a^b (\lambda - \tau_1) \left[\prod_{i=1}^N (\lambda - \theta_i) \right]^2 d\mu(\lambda) && \text{for } M = 1, \\ R[f] &= \frac{f^{(2N+M)}(\eta)}{(2N+M)!} \int_a^b \prod_{l=1}^M (\lambda - \tau_l) \left[\prod_{i=1}^N (\lambda - \theta_i) \right]^2 d\mu(\lambda) && \text{for } M > 1. \end{aligned}$$

For $M = 1$, examine the sign of $R[f]$,we know it is determined by the sign of $f^{(2N+1)}$ and $(\lambda - \tau_1)$.

Remark 2.2 The matrix A is reference only in the form matrix-vector product.it requires only $3n$ fast memory. So we say this algorithm is attractive for large-scale problems.

Algorithm 2 (BFG) Let $A \in R^{n \times n}$, $A > 0$, f be a smooth function . m is the given sample number of MC simulation. This algorithm approximates $tr(f(A))$ and estimates upper bound, lower bound.

1. for $j = 1, 2 \dots, m$

- (a) Generate n -vector \mathbf{z}_j whose element are uniformly distributed in the interval $(0,1)$.
- (b) for $i = 1 : n$, if $\mathbf{z}_j(i) < 0.5$, then $\mathbf{z}_j(i) = -1$, otherwise, $\mathbf{z}_j(i) = 1$.
- (c) Apply Algorithm 1 to obtain a lower bound L_j and an upper bound U_j of $\mathbf{z}_j^T f(A) \mathbf{z}_j$
- (d) $I_j = (L_j + U_j)/2$

end for.

2. $I = \sum_{i=1}^m I_i/m$; $L = \sum_{i=1}^m L_i/m$; $U = \sum_{i=1}^m U_i/m$.

3 Perturbation analysis

In this section, we first play first-order perturbation analysis of $tr(f(A))$ for general matrices and functions, then give results on two special functions which we are interested in.

Theorem 3.1 Let $A, E \in C^{n \times n}$ be Hermitian matrices, E is a perturbation matrix, denote by λ_i the eigenvalues of A , by $\mu_i = \lambda_i + \varepsilon_i$ the corresponding eigenvalues of $A + E$, f is differentiable at λ_i so that $f(x) = f(\lambda_i) + f^{(1)}(\lambda_i)(x - \lambda_i) + \mathcal{O}((x - \lambda_i)^2)$ and $f(A), f^{(1)}(A)$ exist, then:

$$|tr(f(A + E) - f(A))| \leq tr(B) \|E\|_2 + \mathcal{O}(\|E\|_2^2)$$

where $B = \text{diag} (|f^{(1)}(\lambda_1)|, |f^{(1)}(\lambda_2)|, \dots, |f^{(1)}(\lambda_n)|)$.

Proof: Because A and E are Hermitian matrices, we have

$$|\varepsilon_i| = |\mu_i - \lambda_i| \leq \|E\|_2$$

$$|f(\mu_i) - f(\lambda_i)| = |f^{(1)}(\lambda_i)\varepsilon_i| + \mathcal{O}(\varepsilon_i^2) \leq |f^{(1)}(\lambda_i)| \|E\|_2 + \mathcal{O}(\|E\|_2^2)$$

by the definition of function of matrix, we know the eigenvalues of $f(A + E)$ and $f(A)$ are $f(\mu_i), f(\lambda_i)$ respectively, thus

$$\begin{aligned} |tr(f(A + E) - f(A))| &= \left| \sum_{i=1}^n (f(\mu_i) - f(\lambda_i)) \right| \\ &\leq \sum_{i=1}^n |f^{(1)}(\lambda_i) \varepsilon_i| + \mathcal{O}(\|E\|_2^2) \\ &\leq tr(B) \|E\|_2 + \mathcal{O}(\|E\|_2^2) \end{aligned}$$

□

From theorem 3.1 we can derive the condition number for $tr(A^{-1})$ and $tr(\ln(A))$.

Theorem 3.2 *Let A and $A + E$ be S.P.D. matrices, $E = \varepsilon F, \|F\|_2 = \|A\|_2$, then:*

$$\begin{aligned} \left| \frac{tr(\ln(A+E) - \ln(A))}{tr(\ln(A))} \right| &\leq \kappa_{tr\ln} \frac{\|E\|_2}{\|A\|_2} + \mathcal{O}(\varepsilon^2) & \det(A) \neq 1 \\ tr(\ln(A + E) - \ln(A)) &\leq tr(A^{-1}) \|E\|_2 + \mathcal{O}(\|E\|_2^2) & \det(A) = 1 \end{aligned}$$

where $\kappa_{tr\ln} = \frac{tr(A^{-1})\|A\|_2}{|tr(\ln(A))|}$

Proof: By theorem 3.1 we have

$$\begin{aligned} tr(\ln(A + E) - \ln(A)) &\leq \sum_{i=1}^n \frac{1}{\lambda_i} \|E\|_2 + \mathcal{O}(\|E\|_2^2) \\ &= tr(A^{-1}) \|E\|_2 + \mathcal{O}(\|E\|_2^2) \end{aligned}$$

So the second result is proved. dividing two hands with $tr(\ln(A))$ we complete the proof of the first result. □

Next we derive the condition number of $tr(A^{-1})$ under more general condition, A and E may not be Hermitian.

Lemma 3.3 $|tr(A^T B)| \leq \|A\|_F \|B\|_F$.

Lemma 3.4 $\|AB\|_F^2 \leq \|A\|_F^2 \|B\|_2^2$.

Theorem 3.5 *Let A and $A + E$ be nonsingular matrices, $tr(A^{-1}) \gg \varepsilon, E = \varepsilon F, \|F\|_2 = \|A\|_2$, then:*

$$\left| \frac{tr((A + E)^{-1} - A^{-1})}{tr(A^{-1})} \right| \leq \kappa_{trinv} \frac{\|E\|_2}{\|A\|_2} + \mathcal{O}(\varepsilon^2)$$

where $\kappa_{trinv} = \frac{\|A^{-1}\|_F^2 \|A\|_2}{tr(A^{-1})}$

Proof: Define matrix $P(\varepsilon)$ and analyse its Taylor expansion at point A :

$$P(\varepsilon) = (A + \varepsilon F)^{-1},$$

$$P(\varepsilon) = P(0) + P'(0)\varepsilon + \mathcal{O}(\varepsilon^2).$$

so $(A+E)^{-1}-A^{-1} = P'(0)\varepsilon+\mathcal{O}(\varepsilon^2)$, we want to know $P'(0)$. Differentiating $(A+\varepsilon F)P(\varepsilon) = I$ and then set $\varepsilon = 0$, we get *Fréchet* differential quotient of $f = 1/x$ at point A in the direction F .

$$\begin{aligned} P'(0) &= -A^{-1}FA^{-1}, \\ \text{tr}((A+E)^{-1}-A^{-1}) &= -\text{tr}(-A^{-1}FA^{-1})\varepsilon + \mathcal{O}(\varepsilon^2). \end{aligned}$$

by Lemma 3.3 and Lemma 3.4, we have

$$\begin{aligned} \left| \text{tr}(A^{-1}FA^{-1}) \right| &\leq \|A^{-1}F\|_F \|A^{-1}\|_F \\ &\leq \|A^{-1}\|_F^2 \|F\|_2. \end{aligned}$$

because $\|F\|_2 = \|A\|_2$, so

$$\begin{aligned} \left| \frac{\text{tr}((A+E)^{-1}-A^{-1})}{\text{tr}(A^{-1})} \right| &\leq \varepsilon \frac{\|F\|_2 \|A^{-1}\|_F^2}{\text{tr}(A^{-1})} + \mathcal{O}(\varepsilon^2) \\ &= \frac{\|A\|_2 \|A^{-1}\|_F^2 \|E\|_2}{\text{tr}(A^{-1}) \|A\|_2} + \mathcal{O}(\varepsilon^2) \\ &= \kappa_{\text{trinv}} \frac{\|E\|_2}{\|A\|_2} + \mathcal{O}(\varepsilon^2). \end{aligned}$$

□

4 Error estimation

In this section, we first give the error estimation for the Gauss-type quadrature rule used for computing quadratic form $\mathbf{u}^T f(A)\mathbf{u}$, then analyse the confidence interval of MC simulation.

4.1 Error estimation for Algorithm 1

In our applications of computing the quadratic form, we don't satisfy the remainder estimation of Gauss-type quadrature rule in remark 2.1. We explore the remainder analysis in matrix language. Examine the procedure (2.2) -(2.4), we obtain the corresponding Lanczos procedure for matrix A with initial vector $\mathbf{x}_1 = \mathbf{u}/\|\mathbf{u}\|_2$, we can find Lanczos vector $\mathbf{x}_i = p_{i-1}(A)\mathbf{u}/\|\mathbf{u}\|_2$. (for p_i see (2.2)-(2.4)).

Proposition 4.1 *If tridiagonal matrix T_m (refer (2.5)) is generated by Lanczos procedure on symmetric matrix A with normalized initial vector \mathbf{x}_1 , denote by \mathbf{x}_i the Lanczos vector, then we have:*

$$e_m^T T_m^i e_1 = 0, \text{ for } i \leq m-2, \quad (4.6)$$

$$e_m^T T_m^{m-1} e_1 = \gamma_1 \gamma_2 \cdots \gamma_{m-1}, \quad (4.7)$$

$$\mathbf{x}_1^T A^m \mathbf{x}_{m+1} = \gamma_1 \gamma_2 \cdots \gamma_m. \quad (4.8)$$

Proof. The former two results can be verified easily. To prove the third equality, we only need to notice $A^m \mathbf{x}_1 = \gamma_1 \gamma_2 \cdots \gamma_m \mathbf{x}_{m+1} + \sum_{i=1}^m c_i \mathbf{x}_i$. □

Proposition 4.2 Let \mathbf{x}_1 be a normalized vector, run m -steps Lanczos procedure and use Algorithm 1 to compute $\mathbf{x}_1^T f(A) \mathbf{x}_1$, then :

$$\sum_{j=1}^m \omega_j f(\theta_j) = \mathbf{e}_1^T f(T_m) \mathbf{e}_1$$

$$\sum_{j=1}^{m+1} \omega'_j f(\theta'_j) = \mathbf{e}_1^T f(\tilde{T}_{m+1}) \mathbf{e}_1$$

Proof. Recall (2.4), analyse the eigenpair of $T_m, \tilde{T}_{m+1}, f(T_m)$ and $f(\tilde{T}_{m+1})$. noticing T_m and \tilde{T}_{m+1} are normal matrices, we can prove the proposition. \square

In the rest parts we give propositions and theorems only for the case $M = 0$, similar results exist for $M > 0$.

Proposition 4.3 Assume symmetric matrix $A \in R^{n \times n}$, $\mathbf{x}_1 \in R^n$ is a normalized vector, run m -steps Lanczos procedure and use Algorithm 1 to compute $\mathbf{x}_1^T f(A) \mathbf{x}_1$ ($M = 0$), denote by \mathcal{P}_k the set of all degree- k polynomial, we have:

1. if $f \in \mathcal{P}_{2m}$, $f(x) = \sum_{i=0}^{2m} b_i x^i$, then the remainder $R[f] = b_{2m}(\gamma_1 \gamma_2 \cdots \gamma_m)^2$.
2. if $f \in \mathcal{P}_{2m-1}$, then $R[f] = 0$.
3. if the degree of the minimal polynomial of A with respect to \mathbf{x}_1 is $m-1$, then $R[f] = 0$.

Proof. Recall the Lanczos procedure, denote $X_m = (\mathbf{x}_1, \mathbf{x}_2 \cdots \mathbf{x}_m)$, we have:

$$AX_m = X_m T_m + \gamma_m \mathbf{x}_{m+1} \mathbf{e}_m^T$$

$$A^k X_m = X_m T_m^k + \gamma_m \sum_{i=0}^{k-1} A^i \mathbf{x}_{m+1} \mathbf{e}_m^T T_m^{k-i-1} \quad k = 1, 2, \dots$$

noticing proposition 4.1 and $\mathbf{x}_1^T A^i \mathbf{x}_{m+1} = 0$ for $i < m$, we have

$$\begin{aligned} \mathbf{x}_1^T A^{2m} X_m \mathbf{e}_1 - \mathbf{e}_1^T T_m^{2m} \mathbf{e}_1 &= \gamma_m \sum_{i=0}^{2m-1} \mathbf{x}_1^T A^i \mathbf{x}_{m+1} \mathbf{e}_m^T T_m^{2m-i-1} \mathbf{e}_1 \\ &= \gamma_m \sum_{i=m}^{2m-1} \mathbf{x}_1^T A^i \mathbf{x}_{m+1} \mathbf{e}_m^T T_m^{2m-i-1} \mathbf{e}_1 \\ &= \gamma_m \mathbf{x}_1^T A^m \mathbf{x}_{m+1} \gamma_1 \gamma_2 \cdots \gamma_{m-1} \\ &= \gamma_m (\gamma_1 \gamma_2 \cdots \gamma_{m-1})^2 \gamma_m. \end{aligned}$$

It can be found from the former steps that $\mathbf{x}_1^T A^i \mathbf{x}_1 - \mathbf{e}_1^T T_m^i \mathbf{e}_1 = 0$ for $i < 2m$, thus we complete the proof of result 1 and 2. By assumptions of result 3, it is easy to prove $\gamma_m = 0$, so $R[f] = 0$. \square

Theorem 4.4 Assume $A > 0$, $\lambda(A) \in (a, b)$, run m -steps Lanczos procedure on A with normalized initial vector \mathbf{x}_1 , $f(x) = \sum_{i=0}^{\infty} b_i x^i$ converge on $x \in (a, b)$, and series $\sum_{l=2}^{\infty} |b_{2m+l-1}| l \|A\|^l$ converges to a constant ρ . Compute $\mathbf{x}_1^T f(A) \mathbf{x}_1$ by Algorithm 1 with $M = 0$. then:

$$|R[f]| \leq |b_{2m}| (\gamma_1 \gamma_2 \cdots \gamma_m)^2 + |\gamma_m| \|A\|^{2m-2} \rho$$

Proof. Using the denotations of proposition 4.3, we have:

$$AX_m = X_m T_m + \gamma_m \mathbf{x}_{m+1} e_m^T$$

$$A^k X_m = X_m T_m^k + \gamma_m \sum_{i=0}^{k-1} A^i \mathbf{x}_{m+1} e_m^T T_m^{k-i-1} \quad k = 1, 2, \dots$$

let $S(k) = \mathbf{x}_1^T A^k X_m e_1$, similar to the reducing of proposition 4.3, we have:

$$\begin{aligned} S(k) &= \mathbf{x}_1^T (X_m T_m^k + \gamma_m \sum_{i=0}^{k-1} A^i \mathbf{x}_{m+1} e_m^T T_m^{k-i-1}) e_1 \\ \sum_{k=0}^{\infty} b_k S(k) &= e_1^T f(T_m) e_1 + \gamma_m \sum_{k=2m}^{\infty} b_k \sum_{i=0}^{k-1} \mathbf{x}_1^T A^i \mathbf{x}_{m+1} e_m^T T_m^{k-i-1} e_1 \\ &= e_1^T f(T_m) e_1 + \gamma_m \sum_{k=2m}^{\infty} b_k \sum_{i=m}^{k-m} \mathbf{x}_1^T A^i \mathbf{x}_{m+1} e_m^T T_m^{k-i-1} e_1 \\ &= e_1^T f(T_m) e_1 + b_{2m} (\gamma_1 \gamma_2 \cdots \gamma_m)^2 \\ &\quad + \gamma_m \sum_{l=1}^{\infty} b_{2m+l} \sum_{i=m}^{l+m} \mathbf{x}_1^T A^i \mathbf{x}_{m+1} e_m^T T_m^{l+2m-i-1} e_1 \end{aligned}$$

thus

$$\begin{aligned} \mathbf{x}_1^T f(A) \mathbf{x}_1 - e_1^T f(T_m) e_1 &= b_{2m} (\gamma_1 \gamma_2 \cdots \gamma_m)^2 \\ &\quad + \gamma_m \sum_{l=1}^{\infty} b_{2m+l} \sum_{i=m}^{l+m} \mathbf{x}_1^T A^i \mathbf{x}_{m+1} e_m^T T_m^{l+2m-i-1} e_1 \end{aligned}$$

$$|R[f]| \leq |b_{2m}| (\gamma_1 \gamma_2 \cdots \gamma_m)^2 + |\gamma_m| \sum_{l=1}^{\infty} \sum_{i=m}^{l+m} |b_{2m+l}| \|A\|_2^i \|T_m\|_2^{l+2m-i-1}$$

since $\|T_m\|_2 \leq \|A\|_2$, we have:

$$\begin{aligned} |R[f]| &\leq |b_{2m}| (\gamma_1 \gamma_2 \cdots \gamma_m)^2 + |\gamma_m| \sum_{l=1}^{\infty} |b_{2m+l}| (l+1) \|A\|_2^{l+2m-1} \\ &= |b_{2m}| (\gamma_1 \gamma_2 \cdots \gamma_m)^2 + |\gamma_m| \|A\|_2^{2m-2} \rho \end{aligned}$$

□

4.2 Error estimation for MC simulation

Let $w_i = z_i^T H z_i - \text{tr}(H)$ be samples of random variable W , then $E(W) = 0$. To estimate confidence interval of MC simulation, Z. Bai etc [1] use the Hoeffding's inequality [9], but using this inequality one need to compute $\text{range}(W)$, i.e. $[\min_i(w_i), \max_i(w_i)]$. It may be a more difficult problem. Thanks to the center limit theorem we know that $\bar{w}_N = (\sum_{i=1}^N w_i)/N$ asymptotically obey normal distribution .

$$P(|\bar{w}_N - E(W)| < \frac{\lambda_\alpha \sigma}{\sqrt{N}}) \approx \frac{2}{\sqrt{2\pi}} \int_0^{\lambda_\alpha} e^{-\frac{1}{2}t^2} dt = 1 - \alpha \quad (4.9)$$

When N is large enough, the approximation to distribution will be good. σ , Standard variance of random variable W , can be estimated in the MC procedure, if given a confidence probability $p = 1 - \alpha$ and a accuracy $|\bar{w}_N - E(W)| < \delta$, by (4.9) we can compute the number N , i.e. how many samples we need .

5 Improvement of algorithm

Examine (4.9), for given argument $p = 1 - \alpha$ and δ , to guarantee $\bar{w} \in [E(W) - \delta, E(W) + \delta]$ with probability p , the argument σ and N should satisfy:

$$\frac{\sigma}{\sqrt{N}} \leq \frac{\delta}{\lambda_\alpha}.$$

If δ is the upper bound of relative accuracy, $|\bar{I}_N - \text{tr}(H)| / |\text{tr}(H)| = |\bar{w}_N - E(W)| / |\text{tr}(H)| < \delta$, then N should satisfy

$$N > \left(\frac{\lambda_\alpha}{\delta} \right)^2 \left(\frac{\sigma}{\text{tr}(H)} \right)^2 \quad (5.10)$$

In practice we replace $\text{tr}(H)$ and σ^2 with unbiased estimation $\bar{I}_N = \sum_{j=1}^N I_j / N$ and $\bar{\sigma}_N^2 = \sum_{j=1}^N (I_j - \bar{I})^2 / (N - 1) = (\sum_{j=1}^N I_j^2 - N\bar{I}^2) / (N - 1)$ respectively. \bar{I}_N and $\bar{\sigma}_N$ are updated as the number of samples increasing. Because variance σ^2 is unknown, we can not prescribe the number of samples, N should be decided according to the computed quantity.

Algorithm 3 (Modified) *Given confidence probability p and relative accuracy δ , compute unbiased estimation I_p of the quantity $\text{tr}(f(A))$ by MC simulation. confidence interval will be $(I_p - \delta I_p, I_p + \delta I_p)$.*

1. for $j = 1, 2 \dots$

(a) Generate n -vector \mathbf{z}_j whose element are uniformly distributed in the interval $(0, 1)$.

(b) for $i = 1 : n$, if $\mathbf{z}_j(i) < 0.5$, then $\mathbf{z}_j(i) = -1$, otherwise, $\mathbf{z}_j(i) = 1$.

(c) Apply Algorithm 1 to compute $I_j = \mathbf{z}_j^T f(A) \mathbf{z}_j$.

(d) update \bar{I}_j , $\bar{\sigma}_j^2$ and verify condition (5.10), if it is satisfied , break.

end for.

2. $I_p = \bar{I}_j$.

Remark 5.1 *Algorithm 3 has an important improvement compared to Algorithm 2. Using Algorithm 3 we can control relative error with a given probability. Algorithm 2 computes m samples but we don't know what accuracy it has attained.*

Remark 5.2 *In Algorithm 2 the number of samples is given before computing. In Algorithm 3, N depends on $\bar{\sigma}_j^2$, N and $\bar{\sigma}_j^2$ are updated simultaneously. If σ is small, We only need few samples otherwise we'll compute a lot of samples.*

6 Numerical examples

In this section, we test four examples for $f(A) = A^{-1}$ and two examples for $f(A) = \ln(A)$. All examples are the same as used in [1], *Poisson matrix*, *Wathen matrix* and *Lehmer matrix* just come from Higham test set [7], *Linear heat flow matrix* is a block tridiagonal matrix. Because partial elements of *Wathen matrix* are random, we can not get the *Wathen matrix* exactly equal to that used in [1].

Computations are carried out on PC pentium MMX200 using Matlab 5.0. We compute quadratic form by Algorithm 1 with Gauss-rule (i.e. $M = 0$). The convergence tolerance of Algorithm 1 is set to be 10^{-4} , confidence probability is set to 0.95 as in [1]. The control relative errors are set to 2% for $tr(A^{-1})$ and 1% for $\ln(A)$. In table 1, κ is the condition

Table 1: Estimate $tr(f(A))$ by MC simulation

Problems	$tr(A^{-1})$				$\ln(A)$	
	Lehmer	Wathen	Poisson	H flow	Poisson	H flow
Matrix						
n	200	481	900	900	900	100
κ	23094.1	498.40	1063.8	34.24	64.37	19.01
Var/ I^2	0.0066	0.002	0.0287	1.29e-4	9.77e-4	0.003
MC-iter	66	4	243	5	22	85
In-iter	31-57	44-49	23-41	5	13-22	4
I	2.0e+4	32.47	512.6	526.8	1065.0	56.43
I_p	19870	33.03	516.6	528.6	1072.0	56.92
Con-err	2%	2%	2%	2%	1%	1%
Real-err	0.677%	1.7%	0.77%	0.34%	0.703%	0.860%
Interval	(19470,20260)	(32.37,33.69)	(506.3,526.9)	(518.0,539.2)	(1062,1083)	(56.35,57.49)
Err-BFG	0.8%	0.5%	2.0%	—	0.4%	0.4%
Inte-BFG	(18600,21600)	(25.8,27.5)	(428,575)	—	(1030,1080)	(54.2,59.2)

number of $tr(f(A))$ analysed in section 3, ‘Var’ is the true variance of random variable W , in practice we only use its estimation $\bar{\sigma}^2$ in our algorithm. We denote by ‘I’ the exact values of $tr(f(A))$, by ‘MC-iter’ the number of samples, by ‘In-iter’ the iteration steps of Algorithm 1, by ‘ I_p ’ the estimation to ‘I’, by ‘Con-err’ the given control upper of relative error, by ‘Real-err’ the real computation relative error, by ‘Interval’ the confidence interval. Denote by ‘Err-BFG’ the real relative error and by ‘Inte-BFG’ the confidence interval computed by Algorithm 2 in [1], where the number of samples was set to 50 for all examples.

Because *Wathen matrix* is random, we can not compare the quantities between two algorithms, but relative accuracy of confidence interval can still be compared.

All examples illustrate that the Modified Algorithm produces a more accurate and useful confidence interval than BFG Algorithm. Relative errors are under the control of given accuracy δ and probability p . Apart from examples listed here, a lot of other examples indicate the same properties.

7 Conclusion

In this paper, we examine the problem of computing $tr(f(A))$ and a fast algorithm. Theoretically, condition number and computation error are estimated. An important improvement of algorithm is developed too. The Modified Algorithm make the accuracy under control, solve

the stop criterion problem and give more accurate confidence interval than BFG Algorithm.

Acknowledgment The author thanks prof. Erxiong Jiang for carefully check and prof. Zhaojun Bai for introducing his new papers to Fudan University.

References

- [1] Zhaojun Bai, Mark Fahey and Gene Golub, Some large-scale matrix computation problems, *J. comput. appl. math.* 74 (1996) 71-89.
- [2] Zhaojun Bai and Gene H. Golub, Bounds for the trace of the inverse and the determinant of symmetric positive definite matrices. *Annals of numer. math.* 4(1997) 29-38.
- [3] Jiang Erxiong, Zhao Fengguang. Numerical approximation (Chinese edition) 1996, Fudan university press.
- [4] Jiang Erxiong, The symmetric matrices computation. (Chinese edition) 1984, Shanghai science and Technology press.
- [5] G. Golub, Some modified matrix eigenvalue problems, *SIAM Rev.* 15(1973)318-334.
- [6] G. Golub and C. Van Loan, *Matrix Computation*, (3rd edn., 1996)
- [7] N.J. Higham, The test matrix toolbox for Matlab. Numerical Analysis Report No.276. University of Manchester. England. Dec.1995.
- [8] B.N. Parlett, The symmetric eigenvalue problem. (SIAM edition) 1997.
- [9] D. Pollard, *Convergence of Stochastic Processes* (Springer, New York, 1984).
- [10] J.H. Wilkinson, *Algebraic eigenvalue problems*. Clarendon Press, 1965.