

Gauss quadrature based finite temperature Lanczos method

Jian Li(李健) and Hai-Qing Lin(林海青)

Citation: Chin. Phys. B, 2022, 31 (**5**): 050203. DOI: 10.1088/1674-1056/ac5986
Journal homepage: http://cpb.iphy.ac.cn; http://iopscience.iop.org/cpb

What follows is a list of articles you may be interested in

Effect of symmetrical frequency chirp on pair production

Kun Wang(王焜), Xuehua Hu(胡学华), Sayipjamal Dulat, and Bai-Song Xie(谢柏松)
Chin. Phys. B, 2021, 30 (**6**): 060204. DOI: 10.1088/1674-1056/abeee8

A local refinement purely meshless scheme for time fractional nonlinear Schrödinger equation in irregular geometry region

Tao Jiang(蒋涛), Rong-Rong Jiang(蒋戎戎), Jin-Jing Huang(黄金晶), Jiu Ding(丁玖), and Jin-Lian Ren(任金莲)
Chin. Phys. B, 2021, 30 (**2**): 020202. DOI: 10.1088/1674-1056/abc0e0

Compact finite difference schemes for the backward fractional Feynman-Kac equation with fractional substantial derivative

Jiahui Hu(胡嘉卉), Jungang Wang(王俊刚), Yufeng Nie(聂玉峰), Yanwei Luo(罗艳伟)
Chin. Phys. B, 2019, 28 (**10**): 100201. DOI: 10.1088/1674-1056/ab3af3

Heteroclinic cycles in a new class of four-dimensional discontinuous piecewise affine systems

Wenjing Xu(徐文静), Wei Xu(徐伟), Li Cai(蔡力)
Chin. Phys. B, 2018, 27 (**11**): 110201. DOI: 10.1088/1674-1056/27/11/110201

Improved reproducing kernel particle method for piezoelectric materials

Ji-Chao Ma(马吉超), Gao-Feng Wei(魏高峰), Dan-Dan Liu(刘丹丹)
Chin. Phys. B, 2018, 27 (**1**): 010201. DOI: 10.1088/1674-1056/27/1/010201

RAPID COMMUNICATION

Gauss quadrature based finite temperature Lanczos method

Jian Li(李健) and Hai-Qing Lin(林海青)†

Beijing Computational Science Research Center, Beijing 100193, *China*

(Received 12 January 2022; revised manuscript received 15 February 2022; accepted manuscript online 2 March 2022)

The finite temperature Lanczos method (FTLM), which is an exact diagonalization method intensively used in quantum many-body calculations, is formulated in the framework of orthogonal polynomials and Gauss quadrature. The main idea is to reduce finite temperature static and dynamic quantities into weighted summations related to one- and two-dimensional Gauss quadratures. Then lower order Gauss quadrature, which is generated from Lanczos iteration, can be applied to approximate the initial weighted summation. This framework fills the conceptual gap between FTLM and kernel polynomial method, and makes it easy to apply orthogonal polynomial techniques in the FTLM calculation.

Keywords: exact diagonalization, Lanczos method, orthogonal polynomials

PACS: 02.60.Dc, 02.60.–x, 75.10.Jm, 75.40.Mg **DOI:** 10.1088/1674-1056/ac5986

1. Introduction

In the study of quantum many-body systems, the exact diagonalization (ED) method is intensively used to calculate static and dynamic quantities.[1–4] Since the dimension of many-body Hilbert space increases exponentially with system size, ED is only appliable on systems of relatively small size. But ED is still an effective method since it can always get unbiased result compared with other variational methods like density matrix renormalization group,[5–7] and does not suffer from sign problem in quantum Monte Carlo simulations.[8–10] To solve the problem of large matrix dimension, ED is often based on algorithms from sparse matrix calculation, especially Lanczos method[11–14] and kernel polynomial method (KPM).[15–18]

The Lanczos method is originally used in the calculation of ground state properties, since only extreme eigenvalues and eigenvectors converge well. With the introduction of finite temperature Lanczos method (FTLM) by Jaklič and Prelovšek,[19,20] finite temperature static and dynamic quantities can be calculated accurately. In FTLM, the true Hamiltonian H is represented by an effective Hamiltonian $\hat{H}$ in the Krylov subspace generated by Lanczos iteration. Then by expanding $\exp(-\beta H)$ into Taylor series, the calculation is reduced to evaluating quadratic forms of the type $\langle n|H^k B H^l A|n\rangle$, which can be achieved using effective Hamiltonian $\hat{H}$.

KPM is another method used in the calculation of finite temperature properties. The main idea behind KPM is using Chebyshev polynomials to expand quantities like density of states, static and dynamic correlation functions. KPM converges very well at high temperature. But when temperature comes to zero, the low lying states, which KPM does not calculate very accurately, contribute an important part in thermodynamic quantities. To overcome this, the KPM should run several Lanczos iterations to get accurate low lying states, and projects these states out in later calculations. Despite the inaccuracy at low lying states, KPM is believed to be simpler and faster than Lanczos method, and does not suffer from the problem of losing orthogonality occurred in high order Lanczos iteration.[15]

Recently there have been numerical experiments benchmarking the accuracy of FTLM and KPM,[21,22] but the relationship between these two classes of ED methods has not been well explored yet. In this paper, we develop and formulate FTLM in the framework of Gauss quadrature and orthogonal polynomials. In this framework, the Lanczos iteration in FTLM is regarded as a procedure to generate a series of orthogonal polynomials by which different functions of Hamiltonian is expanded. These orthogonal polynomials play the same role as that of Chebyshev polynomials in KPM. The combination of Gauss quadrature and Lanczos iteration has been used in the matrix computation community,[23,24] for example, to give error estimate of solution of linear equations, which is related to the quadratic forms $u^\dagger A^{-i} u$ for $i = 1, 2$. Here we generalized this method to the calculation of more general form $u^\dagger f(H) A g(H) v$, which needs the notion of two-dimensional Gauss quadrature and can be applied in the calculation of finite temperature dynamic correlation functions. This Gauss quadrature based framework fills the conceptual gap between FTLM and KPM, which makes it easy to apply orthogonal polynomial techniques commonly used by KPM in the FTLM calculation. The implementation of FTLM is reduced to one- or two-dimensional Gauss quadratures, which is similar to that of KPM, and is simpler than the procedure of Taylor series expansion.

†Corresponding author. E-mail: haiqing0@csrc.ac.cn

© 2022 Chinese Physical Society and IOP Publishing Ltd

2. Fundamental theory

In large scale exact diagonalization of quantum many-body systems, the Hamiltonian H is given as a sparse Hermitian matrix of dimension N. Usually the study of static and dynamic quantities involves calculating trace of $f(H)$, where f is a smooth function. For example, $f(H)=\exp(-\beta H)$ for the calculation of partition function, and $f(H)=\exp(-\mathrm{i}Ht)$ for the calculation of real time evolution. As we will see in Section 3, we can get many static and dynamic quantities by a suitable choice of $f(H)$, and an effective way to calculate following quantities:

1. $u^\dagger f(H)u$,
2. $u^\dagger f(H)v$ where $u\neq v$,
3. $u^\dagger f(H)Ag(H)v$.

Here u and v are N-dimensional vectors representing quantum many-body states. As developed in following sections, the first quantity is related to one-dimensional Gauss quadrature, and the second and third quantities can be calculated by two-dimensional Gauss quadrature.

2.1. Weighted summation and Gauss quadrature

Let H be a Hermitian matrix of dimension N with following eigenvalue decomposition:

$$H=X\Lambda X^\dagger, \tag{1}$$

where $\Lambda=\mathrm{diag}(\lambda_0,\lambda_1,\ldots,\lambda_{N-1})$ is a diagonal matrix with N real eigenvalues, and X is an unitary matrix. Given any smooth function f, the matrix $f(H)$ is defined as

$$f(H)=X\,\mathrm{diag}(f(\lambda_0),f(\lambda_1),\ldots,f(\lambda_{N-1}))X^\dagger=Xf(\Lambda)X^\dagger. \tag{2}$$

Now consider the quadratic form $u^\dagger f(H)u$, where u is a N-dimensional vector representing a quantum many-body state. The quadratic form can be written as

$$\begin{aligned} u^\dagger f(H)u &= (X^\dagger u)^\dagger\cdot f(\Lambda)\cdot(X^\dagger u)\\ &= \sum_{i=0}^{N-1}|(X^\dagger u)_i|^2\cdot f(\lambda_i)\\ &\equiv \sum_{i=0}^{N-1} w_i f(\lambda_i). \end{aligned} \tag{3}$$

We can see that $u^\dagger f(H)u$ can be seen as a weighted summation with weights $w_i=|(X^\dagger u)_i|^2$ and evaluation points λ_i being the eigenvalue of H. This form is exactly the same as that of Gauss quadrature,[25] which is extensively used in numerical calculation of integrals.

To be more specific, Gauss quadrature is an approximation method to calculate integrals numerically. It transforms the integral

$$I(f)=\int_a^b f(x)w(x)\,\mathrm{d}x \tag{4}$$

into a weighted summation

$$I_{\mathrm{G}}(f)=\sum_{i=0}^{n-1} w_i f(x_i), \tag{5}$$

where the nodes x_i and weights w_i are determined by a specific Gauss quadrature rule. For a given n and correctly calculated x_i and w_i, $I(f)=I_{\mathrm{G}}(f)$ for all polynomials f that have degree $2n-1$ or less.

Equation (3) can be seen as a high order Gauss quadrature where the order N is exponentially large. The idea behind FTLM is to find a lower order approximation of this high order quadrature. The nodes and weights of the optimal lower order Gauss quadrature can be generated by Lanczos iteration, which leads to following algorithm:[24,26]

1. With normalized starting vector $q_0=u/||u||$, run Lanczos iteration M steps to get tridiagonal matrix T_M (defined in Eq. (13)).

2. Compute the eigenvalue decomposition

$$S^\dagger T_M S=\mathrm{diag}(\lambda_0,\ldots,\lambda_{M-1}). \tag{6}$$

3. Compute

$$I_{\mathrm{G}}(f)=\sum_{j=0}^{M-1}|S_{0,j}|^2 f(\lambda_j). \tag{7}$$

4. $u^\dagger f(H)u\approx I_{\mathrm{G}}(f)\cdot||u||^2$.

To validate the algorithm of FTLM above, we need to dig into the mathematical principles behind Gauss quadrature, which leads us to the theory of orthogonal polynomials. Furthermore, theory of orthogonal polynomials can give error estimates of FTLM, which is essential in numerical simulation.

2.2. Theory of orthogonal polynomials

The theory of orthogonal polynomials[27] is fundamental in the implementation and analysis of Gauss quadrature. Given interval (a,b), define inner product of any two functions f and g in (a,b) as

$$\langle f,g\rangle:=\int_a^b f(x)g(x)w(x)\,\mathrm{d}x, \tag{8}$$

where w is a fixed positive weight function in (a,b), and without loss of generality we set the following restriction on $w(x)$:

$$\int_a^b w(x)\,\mathrm{d}x=1. \tag{9}$$

Let $\{p_n\}_{n=0}^\infty$ denote a series of orthogonal polynomials in (a,b). Here n denotes the degree of polynomial. We further require that p_n is normalized, namely,

$$\langle p_i,p_j\rangle=\delta_{ij}\ \text{ for }\ i,j\geq 0. \tag{10}$$

In general $\{p_n\}_{n=0}^\infty$ can ge generated by following 3-term recurrence relation with $p_0=1$ and $\beta_0=0$:

$$\beta_{n+1}p_{n+1}(x)=(x-\alpha_n)p_n(x)-\beta_n p_{n-1}(x). \tag{11}$$

Here $\alpha_n = \langle xp_n, p_n\rangle$, and β_{n+1} is introduced to normalize p_{n+1}, namely,

$$\beta_{n+1} = ||(x-\alpha_n)p_n(x) - \beta_n p_{n-1}(x)||. \tag{12}$$

The α_i and β_i for $0 \le i < n$ can be used to construct a $n \times n$ tridiagonal matrix T_n as follows:

$$T_n = \begin{bmatrix} \alpha_0 & \beta_1 & 0 & \dots & 0 \\ \beta_1 & \alpha_1 & \beta_2 & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \alpha_{n-2} & \beta_{n-1} \\ 0 & \dots & 0 & \beta_{n-1} & \alpha_{n-1} \end{bmatrix}. \tag{13}$$

The T_n is important in the study of orthogonal polynomials and Lanczos algorithm. Given the eigenvalue decomposition $T_n = S\Lambda S^\dagger$ where $\Lambda = \mathrm{diag}(x_0, x_1, \dots, x_{n-1})$, it turns out that $x_0, x_1, \dots, x_{n-1}$ are the zeros of $p_n(x)$, and

$$S = \begin{bmatrix} \frac{p_0(x_0)}{a_0} & \frac{p_0(x_1)}{a_1} & \dots & \frac{p_0(x_{n-1})}{a_{n-1}} \\ \frac{p_1(x_0)}{a_0} & \frac{p_1(x_1)}{a_1} & \dots & \frac{p_1(x_{n-1})}{a_{n-1}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{p_{n-1}(x_0)}{a_0} & \frac{p_{n-1}(x_1)}{a_1} & \dots & \frac{p_{n-1}(x_{n-1})}{a_{n-1}} \end{bmatrix}, \tag{14}$$

where

$$a_j = \mathrm{e}^{\mathrm{i}\phi_j}\sqrt{\sum_{i=0}^{n-1} p_i^2(x_j)} \tag{15}$$

and $\mathrm{e}^{\mathrm{i}\phi_j}$ is a phase factor for each eigenvector.

If we define

$$w_i = |S_{0,i}|^2, \tag{16}$$

then for any polynomial $p(x)$ of degree less than $2n$, we have

$$\sum_{i=0}^{n-1} p(x_i)w_i = \langle p(x), p_0\rangle. \tag{17}$$

Equation (17) is the fundamental result of Gauss quadrature, which states that we can choose a set of n nodes and weights to construct a quadrature rule of order $2n-1$. One can find detailed proof of Eqs. (14) and (17) in Appendix A.

We can restate Gauss quadrature in the language of discrete inner product. Given the nodes x_i and weights w_i defined above, we can define a discrete inner product in $[a,b]$ and its associated norm by

$$(u,v)_{n,w} = \sum_{j=0}^{n-1} u(x_j)v(x_j)w_j, \tag{18}$$

$$||u||_{n,w} = \sqrt{(u,u)_{n,w}}. \tag{19}$$

And according to Eq. (17) we have

$$(u,v)_{n,w} = \langle u, v\rangle, \quad (p_i,p_j)_{n,w} = \delta_{ij} \tag{20}$$

for all degree(u) + degree$(v) \le 2n-1$ and $i,j \le n-1$.

2.3. Relationship with Lanczos iteration

Lanczos iteration is a standard method to transform a Hermitian matrix into tridiagonal form by an unitary transformation. For a given matrix H and a starting normalized vector u, the Lanczos iteration is given by

$$\beta_{n+1}q_{n+1} = (H-\alpha_n)q_n - \beta_n q_{n-1} \tag{21}$$

with $q_0 = u, \beta_0 = 0$, and

$$\begin{aligned} \alpha_n &= q_n^\dagger H q_n, \\ \beta_{n+1} &= ||(H-\alpha_n)q_n - \beta_n q_{n-1}||. \end{aligned} \tag{22}$$

The transformed tridiagonal matrix is exactly the same as T_n defined in Eq. (13).

We can see many similarities between Eqs. (11) and (21). Actually Lanczos iteration does its transformation according to a sequence of orthogonal polynomials.[24] To see this, let us define $q_i = p_i(H)q_0$, where p_i is a polynomial of degree i. According to Eq. (21), we have

$$\beta_{n+1}p_{n+1}(H)q_0 = (H-\alpha_n)p_n(H)q_0 - \beta_n p_{n-1}(H)q_0. \tag{23}$$

So $p_n(x)$ satisfies the recurrence

$$\beta_{n+1}p_{n+1}(x) = (x-\alpha_n)p_n(x) - \beta_n p_{n-1}(x), \tag{24}$$

which is the same recurrence relation of p_n in Eq. (11).

Here for simplicity we assume that for the Hermitian matrix H of dimension N, we can transform it into a tridiagonal matrix T_N of the same dimension by Lanczos iteration with a suitable normalized starting vector u. Note that Lanczos tridiagonalization procedure is generally an unitary transformation, namely,

$$H = UT_N U^\dagger, \tag{25}$$

where

$$U = [q_0 = u, q_1, \dots, q_{N-1}], \quad q_j = p_j(H)q_0. \tag{26}$$

Given the eigenvalue decomposition $H = X\Lambda X^\dagger$ and $T_N = S\Lambda S^\dagger$, we get

$$\begin{aligned} S &= U^\dagger X, \\ |S_{0,i}|^2 &= |S_{i,0}^\dagger|^2 = |(X^\dagger u)_i|^2. \end{aligned}$$

Thus in the weighted summation (see Eq. (3))

$$u^\dagger f(H)u = \sum_{i=0}^{N-1} w_i f(\lambda_i)$$

we have

$$w_i = |S_{0,i}|^2,$$

which is the same weight as that of Gauss quadrature in Eq. (16).

Now we can explain more explicitly the algorithm of FTLM in Subsection 2.1 using the language of discrete inner product. Suppose f is a smooth function and u is a normalized vector, by definition, we have

$$u^\dagger f(H)u = (f,p_0)_{N,w}. \tag{27}$$

Then the idea is to approximate $(f,p_0)_{N,w}$ by a lower order discrete inner product (or equivalently, lower order Gauss quadrature), namely $(f,p_0)_{M,\tilde{w}}$, where $M \ll N$ is the number of Lanczos iterations we run. We need to give an error estimate of this method. To be more specific, the error is defined as

$$r = (f,p_0)_{N,w} - (f,p_0)_{M,\tilde{w}}. \tag{28}$$

The f can be expand by orthogonal polynomials

$$f = \sum_{i=0}^{\infty} c_i p_i, \quad c_i = \langle f, p_i \rangle . \tag{29}$$

Considering N is very large, and also p_i for $i \geq N$ is not well defined, the expansion of f can be cut off at $N-1$, which leads to[28]

$$f = \sum_{i=0}^{N-1} c_i p_i, \quad c_i = (f,p_i)_{N,w}. \tag{30}$$

Then $(f,p_0)_{N,w} = c_0$, and

$$(f,p_0)_{M,\tilde{w}} = \sum_{i=0}^{N-1} c_i (p_i,p_0)_{M,\tilde{w}} = c_0 + \sum_{i=2M}^{N-1} c_i (p_i,p_0)_{M,\tilde{w}}. \tag{31}$$

So

$$r = -\sum_{i=2M}^{N-1} c_i (p_i,p_0)_{M,\tilde{w}} = -\sum_{i=2M}^{N-1} (f,p_i)_{N,w} (p_i,p_0)_{M,\tilde{w}}, \tag{32}$$

which means that M Lanczos iterations will give approximation up to order of $2M$.

2.4. Two-dimensional Gauss quadrature

Here comes to the question of how to calculate the following quantities:

1. $u^\dagger f(H)v$ where $u \neq v$,
2. $u^\dagger f(H)Ag(H)v$.

We can see that the second quantity is a general form of the first one given $A = g(H) = 1$. As for the first case, if u, v are real vectors and H is a real symmetric matrix, we can use the following identity[23] to calculate $u^T f(H)v$:

$$\begin{aligned} u^T f(H)v \;=\; & \frac{1}{4}[(u+v)^T f(H)(u+v) \\ & -(u-v)^T f(H)(u-v)]. \end{aligned} \tag{33}$$

Then we can use the method talked before to calculate $u^T f(H)v$, the only difference is that we need to run Lanczos iteration twice.

But for the general case, namely, $u^\dagger f(H)Ag(H)v$, we need a different method, which needs the notion of two-dimensional Gauss quadrature, as will be discussed below.

Two-dimensional Gauss quadrature, and also two-dimensional orthogonal polynomials, can be easily constructed from a tensor product of two one-dimensional Gauss quadratures and orthogonal polynomials respectively. Formally, given two weighted Hilbert space $L^2_w(a,b)$ and $L^2_{\hat{w}}(a,b)$, we can construct a Hilbert space on $(a,b)\times(a,b)$ by defining the following inner product:

$$\langle f,g \rangle_2 := \iint f(x,y)g(x,y)w(x)\hat{w}(y)\,\mathrm{d}x\mathrm{d}y. \tag{34}$$

Given two sequences of orthogonal polynomials $\{p_m\}_{m=0}^{\infty}$ and $\{q_n\}_{n=0}^{\infty}$ with weight functions w and $\hat{w}$ in (a,b), it can be verified that $p_m(x)q_n(y)$ forms a orthogonal and complete basis in $(a,b)\times(a,b)$, with the property

$$\langle p_m(x)q_i(y), p_n(x)q_j(y) \rangle_2 = \delta_{mn}\delta_{ij}. \tag{35}$$

Same as one-dimensional discrete inner product, we can define two-dimensional discrete inner product as

$$(f,g)^{n,\hat{w}}_{m,w} = \sum_{i=0}^{m-1}\sum_{j=0}^{n-1} w_i \hat{w}_j f(x_i,y_j) g(x_i,y_j), \tag{36}$$

where (x_i,w_i), $0 \leq i \leq m-1$ and $(y_j,\hat{w}_j), 0 \leq j \leq n-1$ are nodes and weights for p_m and q_n, respectively.

It can be verified that

$$\begin{aligned} (p_m(x)q_i(y), p_n(x)q_j(y))^{N,\hat{w}}_{M,w} \;&=\; \langle p_m(x)q_i(y), p_n(x)q_j(y) \rangle_2 \\ &=\; \delta_{mn}\delta_{ij} \end{aligned} \tag{37}$$

for all $m+n \leq 2M-1$ and $i+j \leq 2N-1$.

Now we can transform $u^\dagger f(H)Ag(H)v$ to a two-dimensional discrete inner product, and similar to the one-dimensional case, we can use a lower order discrete inner product to approximate the high order one.

As before, given $H = X\Lambda X^\dagger$, we have

$$\begin{aligned} & u^\dagger f(H)Ag(H)v \\ =\; & u^\dagger X f(\Lambda) X^\dagger A X g(\Lambda) X^\dagger v \\ =\; & \sum_{i,j} (X^\dagger u)_i^\dagger f(\lambda_i)(X^\dagger A X)_{ij} g(\lambda_j)(X^\dagger v)_j \\ =\; & \sum_{i,j} |(X^\dagger u)_i|^2 f(\lambda_i) \frac{(X^\dagger A X)_{ij}}{(X^\dagger u)_i (X^\dagger v)_j^\dagger} g(\lambda_j) |(X^\dagger v)_j|^2 \\ \equiv\; & \sum_{i,j} |(X^\dagger u)_i|^2 f(\lambda_i) C(\lambda_i,\lambda_j) g(\lambda_j) |(X^\dagger v)_j|^2 \\ =\; & (f(x)C(x,y),g(y))^{N,\hat{w}}_{N,w}, \end{aligned} \tag{38}$$

in which we have introduced an auxiliary function $C(x,y)$. It is only defined at some discrete points as follows:

$$C(\lambda_i,\lambda_j)=\frac{(X^\dagger AX)_{ij}}{(X^\dagger u)_i(X^\dagger v)_j^\dagger}. \tag{39}$$

It is now clear that $u^\dagger f(H)Ag(H)v$ is a two-dimensional discrete inner product. The next step is to expand $C(x,y)$ as summation of orthogonal polynomials up to order $N\times N$,[29]

$$C(x,y)=\sum_{m=0}^{N-1}\sum_{n=0}^{N-1}\mu_{mn}p_m(x)q_n(y). \tag{40}$$

The expansion coefficients μ_{mn} can be calculated as

$$\begin{aligned}\mu_{mn} &= \langle C(x,y),p_m(x)q_n(y)\rangle_2\\ &= (C(x,y),p_m(x)q_n(y))_{N,w}^{N,\hat{w}}\\ &= \sum_{i,j}|(X^\dagger u)_i|^2 p_m(\lambda_i)C(\lambda_i,\lambda_j)q_n(\lambda_j)|(X^\dagger v)_j|^2\\ &= \sum_{i,j}(X^\dagger u)_i^\dagger p_m(\lambda_i)(X^\dagger AX)_{ij}q_n(\lambda_j)(X^\dagger v)_j\\ &= u^\dagger p_m(H)Aq_n(H)v\\ &= u_m^\dagger Av_n,\end{aligned} \tag{41}$$

where $u_m=p_m(H)u$ and $v_n=q_n(H)v$ are vectors generated by Lanczos iteration with starting vector u and v, respectively.

The last step is to use a lower order discrete inner product to approximate $u^\dagger f(H)Ag(H)v$. Suppose we run two Lanczos iterations with starting vector u for M_1 steps and starting vector v for M_2 steps, which generates tridiagonal matrix T_{M_1} and $\hat{T}_{M_2}$, respectively. The nodes and weights associated with T_{M_1} and $\hat{T}_{M_2}$ are (λ_i,w_i) and $(\hat{\lambda}_j,\hat{w}_j)$ respectively, then we have

$$\begin{aligned}&u^\dagger f(H)Ag(H)v\\ \approx&\sum_{i,j}w_i f(\lambda_i)C(\lambda_i,\hat{\lambda}_j)g(\hat{\lambda}_j)\hat{w}_j\\ =&\sum_{m,n}\mu_{mn}\sum_{i,j}w_i f(\lambda_i)p_m(\lambda_i)q_n(\hat{\lambda}_j)g(\hat{\lambda}_j)\hat{w}_j\\ =&\sum_{m,n}\mu_{mn}\Big(\sum_i w_i f(\lambda_i)p_m(\lambda_i)\Big)\Big(\sum_j\hat{w}_j g(\hat{\lambda}_j)q_n(\hat{\lambda}_j)\Big)\\ =&\sum_{m,n}\mu_{mn}(f,p_m)_{M_1,w}(g,q_n)_{M_2,\hat{w}}.\end{aligned} \tag{42}$$

Here we need to calculate $p_m(\lambda_i)$ (and also $q_n(\hat{\lambda}_j)$) for each m and i. This can be achieved by using the eigenvectors of T_{M_1}. From Eq. (14) we can see that each row of matrix S divided by the first row of S respectively just gives $p_m(\lambda_i)$.

For a special case, when $g(x)=1=q_0$, we have $(g,q_n)_{M_2,\hat{w}}=\delta_{n0}$, then

$$\begin{aligned}u^\dagger f(H)Ag(H)v &= u^\dagger f(H)Av\\ &= \sum_{m,n}\mu_{mn}(f,p_m)_{M_1,w}\delta_{n0}\\ &= \sum_m\mu_{m0}(f,p_m)_{M_1,w},\end{aligned} \tag{43}$$

which means that we only need to run one Lanczos iteration for M_1 steps. In general case, the numerical effort for the calculation of μ_{mn} ranges between $N(M_1+M_2)$ and NM_1M_2 operations, depending on whether memory is available for saving M_1 (or M_2) vectors of dimension N.

3. Formulas for static and dynamic quantities

Until now we have not discussed how to calculate static and dynamic quantities for a real quantum system. Actually the routines in FTLM share many similarities as in KPM,[15,17] thus can be expressed in a unified form.

In this section and later Dirac bra–ket notations will be used to denote matrix vector multiplication, this notation is inconvenient in previous sections but more suitable when it comes to physical applications.

3.1. Stochastic evaluation of traces

Although we have the method to calculate $\langle u|f(H)|u\rangle$, in many cases we need to evaluate the trace of a given operator. For example, the partition function is given by a trace

$$Z=\mathrm{tr}[\mathrm{e}^{-\beta H}]=\sum_i\langle i|\,\mathrm{e}^{-\beta H}|i\rangle, \tag{44}$$

where $\{|i\rangle\}$ is a complete set of basis.

At first glance it seems impossible to evaluate since the Lanczos iteration needs to be repeated for all N states of a given basis, which makes the total computational effort proportional to N^2. It turns out that extremely good approximation of the trace can be obtained with a much simpler approach: stochastic evaluation of trace, in which estimate of trace is based on the average over a small number $R\ll N$ of randomly chosen vectors[15,30]

$$\mathrm{tr}[f(H)]\approx\frac{N}{R}\sum_{r=0}^{R-1}\langle r|f(H)|r\rangle, \tag{45}$$

where $\{|r\rangle\}$ is a set of random vectors with the i-th element being ξ_{ri}, and $\xi_{ri}\in\mathbb{C}$ are a set of independent identically distributed random variables fulfill

$$\langle\xi_{ri}\rangle=0, \tag{46}$$

$$\langle\xi_{ri}\xi_{r'j}\rangle=0, \tag{47}$$

$$\langle\xi_{ri}^*\xi_{r'j}\rangle=\delta_{rr'}\delta_{ij}, \tag{48}$$

here we use the notation $\langle\cdot\rangle$ to denote the statistical average.

It can be proved that equation (45) is an unbiased estimate of $\mathrm{tr}[f(H)]$ and the relative error of the estimate is of order $O(1/\sqrt{RN})$. So for large matrix where $N\gg1$, R can be kept small.

The ξ_{ri} can also be real and fulfill

$$\langle\xi_{ri}\rangle=0, \tag{49}$$

$$\langle \xi_{ri}\xi_{r'j}\rangle = \delta_{rr'}\delta_{ij}. \tag{50}$$

Typical chosen of ξ_{ri} can be Gaussian distribution with average 0 and standard deviation 1.

3.2. Thermal average and density of states

Given partition function $Z = \mathrm{tr}[\mathrm{e}^{-\beta H}]$, the thermal average of operator A is

$$\langle A\rangle = \frac{1}{Z}\mathrm{tr}[\mathrm{e}^{-\beta H}A]. \tag{51}$$

We can use stochastic evaluation of traces to calculate these quantities. From Eq. (43) we can see that only one Gauss quadrature rule is need to calculate both $\langle r|\mathrm{e}^{-\beta H}|r\rangle$ and $\langle r|\mathrm{e}^{-\beta H}A|r\rangle$ for each given random vector $|r\rangle$, so only one Lanczos iteration is need for the given random vector. This can be generalized to many operators if we want to calculate thermal average of these operators at the same time.

Here we consider two limiting cases to illustrate the accuracy of FTLM.

1. $\beta \to 0$. This is the high temperature limit, where

$$\mathrm{e}^{-\beta x} \to 1 = p_0$$

and according to Eq. (32), few Lanczos iteration will give accurate result.

2. $\beta \to \infty$. This is the low temperature limit, $\mathrm{e}^{-\beta x}$ will be sharply dominated at $x = E_{\min}$. From the theory of orthogonal polynomial expansion, many high order expansion will contribute to this nearly discontinuous function. Furthermore, Gibbs oscillation[17,31] will creep into the expanded function, which introduces numerical instability. In this case, low temperature Lanczos method[32] have been proposed to address this problem. One can also use ground state Lanczos method in this super low temperature regime, which is generally more accurate.

As for density of states, it is defined as

$$\rho(E) = \frac{1}{N}\sum_{k=0}^{N-1}\delta(E-E_k). \tag{52}$$

The Dirac $\delta(x)$ function above is difficult for numerical integration, we can approximate it by a Lorentz function, namely,

$$\delta(x) = \lim_{\varepsilon\to 0^+}\frac{1}{\pi}\frac{\varepsilon}{\varepsilon^2+x^2} \equiv \lim_{\varepsilon\to 0^+} l(x;\varepsilon). \tag{53}$$

Then we set ε as a small positive number and rewrite $\rho(E)$ as

$$\begin{aligned}\rho(E) &\approx \frac{1}{N}\sum_{k=0}^{N-1} l(E-E_k;\varepsilon)\\ &= \frac{1}{N}\sum_{k=0}^{N-1}\langle k|l(E-H;\varepsilon)|k\rangle\\ &= \frac{1}{N}\mathrm{tr}[l(E-H;\varepsilon)]\\ &\approx \frac{1}{R}\sum_{r=0}^{R-1}\langle r|l(E-H;\varepsilon)|r\rangle,\end{aligned} \tag{54}$$

where $|k\rangle$ is the k-th eigenvector of H and $|r\rangle$ is a random vector.

It is numerically more stable to calculate cumulative density of states $c(E)$ instead of $\rho(E)$. $c(E)$ is defined as

$$c(E) = \int_{E_0}^{E}\rho(E')\mathrm{d}E' = \frac{1}{N}\sum_{k=0}^{N-1}h(E-E_k), \tag{55}$$

where $h(x)$ is a step function defined as

$$\begin{aligned}h(x) &= \begin{cases}1, & x\geq 0,\\ 0, & x<0\end{cases}\\ &= \lim_{\varepsilon\to 0^+}\frac{1}{1+\mathrm{e}^{-x/\varepsilon}}\\ &\equiv \lim_{\varepsilon\to 0^+} g(x;\varepsilon).\end{aligned} \tag{56}$$

Similar to the calculation of $\rho(E)$, we set ε as a small positive number and rewrite $c(E)$ as

$$\begin{aligned}c(E) &\approx \frac{1}{N}\sum_{k=0}^{N-1} g(E-E_k;\varepsilon)\\ &= \frac{1}{N}\sum_{k=0}^{N-1}\langle k|g(E-H;\varepsilon)|k\rangle\\ &= \frac{1}{N}\mathrm{tr}[g(E-H;\varepsilon)]\\ &\approx \frac{1}{R}\sum_{r=0}^{R-1}\langle r|g(E-H;\varepsilon)|r\rangle.\end{aligned} \tag{57}$$

3.3. Real time evolution

As a studying case, here we consider real time evolution. Specifically, we are interested in the quantity

$$I = \langle u|\mathrm{e}^{-\mathrm{i}Ht}|v\rangle. \tag{58}$$

From Eq. (43) we can see that

$$\begin{aligned}I &= \sum_m \langle u|p_m(H)|v\rangle(\mathrm{e}^{-\mathrm{i}tx}, p_m(x))_{M,w}\\ &= \sum_m \langle u|p_m(H)|v\rangle[(\cos(tx), p_m(x))_{M,w}\\ &\quad -\mathrm{i}(\sin(tx), p_m(x))_{M,w}].\end{aligned} \tag{59}$$

Here we consider two limiting cases:

1. $t \to 0$. In this case

$$\sin(tx)\to 0, \quad \cos(tx)\to 1 = p_0(x),$$

equation (59) is accurate for very few Lanczos iteration.

2. $t\to\infty$. In this case both $\sin(tx)$ and $\cos(tx)$ will oscillate badly in the integration interval $[E_{\min}, E_{\max}]$, and Gauss quadrature based integration rule will fail to converge.

So the real time evaluation is different from imaginary time evaluation in the sense that real time evaluation will fail to converge in the $t\to\infty$ limit, while imaginary time evaluation will admit accurate results in both $\beta\to 0$ and $\beta\to\infty$ limits.

3.4. Dynamic correlation function

Before we dig into the calculation of finite temperature correlation function, we may first give a glance for the zero temperature case. For zero temperature, the dynamic correlation function for two operator A and B is

$$\langle A;B\rangle_{\omega}^{\pm}=\lim_{\varepsilon\to 0^{+}}\langle 0|A\frac{1}{\omega+\mathrm{i}\varepsilon\mp H}B|0\rangle. \tag{60}$$

From the viewpoint of FTLM, equation (60) is in the form of $\langle u|f(H)|v\rangle$ that can be handled by Gauss quadrature, and the computational complexity is the same as standard Lanczos and KPM.

The finite temperature response function is given by

$$\begin{aligned} C(t) &= \langle A(t)A\rangle \\ &= \frac{1}{Z}\mathrm{tr}[\mathrm{e}^{(-\beta+\mathrm{i}t)H}A\,\mathrm{e}^{-\mathrm{i}Ht}A] \\ &\approx \frac{N}{ZR}\sum_{r=0}^{R-1}\langle r|\mathrm{e}^{(-\beta+\mathrm{i}t)H}A\,\mathrm{e}^{-\mathrm{i}Ht}A|r\rangle \end{aligned} \tag{61}$$

in which the term $\langle r|\mathrm{e}^{(-\beta+\mathrm{i}t)H}A\,\mathrm{e}^{-\mathrm{i}Ht}A|r\rangle$ follows the general form $\langle u|f(H)Ag(H)|v\rangle$ with $f(x;t)=\mathrm{e}^{(-\beta+\mathrm{i}t)x}$ and $g(x;t)=\mathrm{e}^{-\mathrm{i}tx}$, and needs a two-dimensional Gauss quadrature to calculate.

As mentioned in real time evolution, Gauss quadrature based FTLM is not accurate for evolution time t being large, this is also true in Eq. (61). In this case it is better to calculate the Fourier transform of $C(t)$

$$D(\omega)=\frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\infty}\mathrm{e}^{\mathrm{i}\omega t}C(t)\,\mathrm{d}t. \tag{62}$$

In the language of two-dimensional discrete inner product, we can write $C(t)$ as

$$\begin{aligned} C(t) &\approx \frac{N}{ZR}\sum_{r=0}^{R-1}\langle r|f(H)Ag(H)A|r\rangle \\ &= \frac{N}{ZR}\sum_{r=0}^{R-1}(f(x;t)C(x,y),g(y;t))_{N,w}^{N,\hat{w}}. \end{aligned} \tag{63}$$

So its Fourier transform is

$$D(\omega)=\frac{N}{ZR}\sum_{r=0}^{R-1}(C(x,y),G(x,y;\omega))_{N,w}^{N,\hat{w}}, \tag{64}$$

where

$$\begin{aligned} G(x,y;\omega) &= \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\infty}\mathrm{e}^{\mathrm{i}\omega t}f(x;t)g(y;t)\,\mathrm{d}t \\ &= \sqrt{2\pi}\mathrm{e}^{-\beta x}\delta(\omega-(y-x)) \\ &\approx \sqrt{2\pi}\mathrm{e}^{-\beta x}l(\omega-(y-x);\varepsilon), \end{aligned} \tag{65}$$

and l is the Lorentz function defined in Eq. (53).

In general case, the two-dimensional Gauss quadrature $(C(x,y),G(x,y))_{N,w}^{N,\hat{w}}$ in Eq. (64) can be calculated as follows:

$$\begin{aligned} (C(x,y),G(x,y))_{N,w}^{N,\hat{w}} &= \sum_{m,n}\mu_{mn}(p_m(x)G(x,y),q_n(y))_{N,w}^{N,\hat{w}} \\ &\equiv \sum_{m,n}\mu_{mn}G_{mn}, \end{aligned} \tag{66}$$

where

$$\begin{aligned} G_{mn} &= (p_m(x)G(x,y),q_n(y))_{N,w}^{N,\hat{w}} \\ &= \sum_{i,j}w_i p_m(\lambda_i)G(\lambda_i,\hat{\lambda}_j)q_n(\hat{\lambda}_j)\hat{w}_j. \end{aligned} \tag{67}$$

4. Numerical results of 1D *XY* model

In this section we give numerical results to illustrate the idea of Gauss quadrature based finite temperature Lanczos method. The numerical calculation is carried out on the one-dimensional XY model.

The XY model is introduced by Lieb, Schultz and Mattis in 1961,[33] they considered a chain of N 1/2-spins, governed by the Hamiltonian

$$H=\sum_{j}\left[(1+\gamma)S_j^xS_{j+1}^x+(1-\gamma)S_j^yS_{j+1}^y-hS_j^z\right]. \tag{68}$$

This model can be exactly solved by Jordan–Wigner transformation. To make the exact solution simpler, here we consider open boundary condition, and calculate the exact solution for finite lattice, then compare with numerical results come from FTLM. All the calculations are carried out on a lattice of 20 sites ($N=20$), with $\gamma=0.2$ and $h=0.6$.

First we consider specific heat. Specific heat is defined as

$$C_V=\frac{\beta^2}{N}\left(\langle H^2\rangle-\langle H\rangle^2\right). \tag{69}$$

Both $\langle H^2\rangle$ and $\langle H\rangle$ can be calculated by a symmetrical Gauss quadrature of the form $\langle u|f(H)|u\rangle$. The numerical results are shown in Fig. 1(a).

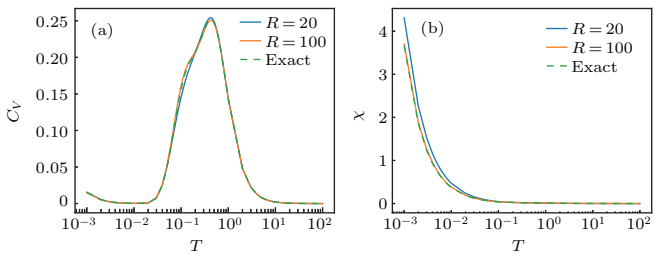


Fig. 1. (a) specific heat and (b) magnetic susceptibility of 1D XY model. In both figures the Lanczos iteration steps is set to 100, and the number of random vectors (denoted by R) is set to 20 and 100 respectively.

The second quantity we consider is magnetic susceptibility, which is defined as

$$\chi=\beta\left(\langle m_z^2\rangle-\langle m_z\rangle^2\right), \tag{70}$$

where m_z is the average magnetization in z direction

$$m_z=\frac{1}{N}\sum_{j}S_j^z. \tag{71}$$

Both $\langle m_z^2 \rangle$ and $\langle m_z \rangle$ can be calculated by an unsymmetrical Gauss quadrature of the form $\langle u|f(H)|v\rangle$. The numerical results are shown in Fig. 1(b).

The computational effort to calculate C_V and χ are approximately same, since only one Lanczos iteration is needed to calculate the symmetric Gauss quadrature $\langle u|f(H)|u\rangle$ and the unsymmetrical Gauss quadrature $\langle u|f(H)|v\rangle$. In both case the Lanczos iteration steps is set to 100, and the number of random vectors (denoted by R) is set to 20 and 100 respectively. From Fig. 1 we can see that FTLM is accurate at high temperature, while the accuracy at low temperature is influenced by statistical fluctuations from random vectors.

The third quantity we calculate is the dynamic correlation function of the average magnetization in z direction,[34] which is defined as

$$\chi(t) = \langle m_z(t) m_z \rangle = \langle \mathrm{e}^{\mathrm{i}Ht} m_z \mathrm{e}^{-\mathrm{i}Ht} m_z \rangle. \tag{72}$$

This quantity can be calculated by a two-dimensional Gauss quadrature of the form $\langle u|f(H)Ag(H)|v\rangle$. The numerical results are shown in Fig. 2(a).

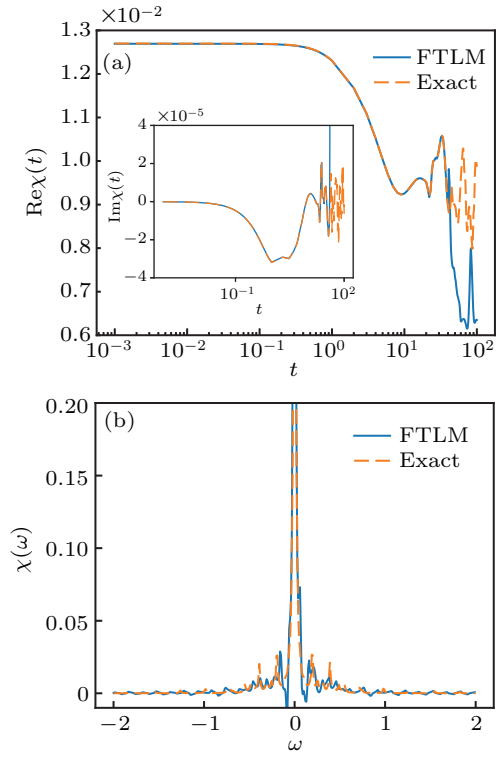


Fig. 2. (a) Dynamic correlation function $\chi(t)$. The main plot shows the real part of $\chi(t)$ while the inset shows the imaginary part of $\chi(t)$. (b) Fourier transform of $\chi(t)$, denoted as $\chi(\omega)$. In both figures the temperature T in the calculation is set to 10. Lanczos iteration steps is set to 300, and the number of random vectors is set to 100.

We can see that numerical result agrees well with exact result when $t < 10$. But for lager t, the numerical result is very inaccurate. As discussed in real time evolution (see Subsection 3.3), Gauss quadrature based integration will fail to converge for highly oscillate functions such as $\mathrm{e}^{\mathrm{i}tH}$ for large t.

It is usually more convenient to study the Fourier transform of $\chi(t)$ defined as follows:

$$\chi(\omega) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} \mathrm{e}^{\mathrm{i}\omega t} \chi(t) \mathrm{d}t. \tag{73}$$

The numerical results are shown in Fig. 2(b).

The calculation of $\chi(\omega)$ also involves two-dimensional Gauss quadrature in which the Dirac δ functions are replaced by Lorentz functions (see Eq. (65)). Since the time scale that FTLM can accurately calculate can not be large, the resolution in ω space, which is represented by the parameter ε in Lorentz function, is also limited due to the time-energy uncertainty principle. As shown in Fig. 2(b), the parameter ε is set to 0.01. Smaller ε will lead to negative $\chi(\omega)$ values, which indicates the failure of convergence.

5. Conclusion

This paper has shown the tight relationship between Lanczos algorithm and orthogonal polynomials, and developed finite temperature Lanczos method in the framework of Gauss quadrature. The Lanczos algorithm can be regarded as a procedure to generate a series of orthogonal polynomials by which different functions of Hamiltonian H are expanded. These orthogonal polynomials also define Gauss quadrature rules. The nodes and weights of Gauss quadrature are given by the eigenvalues and eigenvectors of tridiagonal matrix which is generated by the Lanczos iteration. Given the Gauss quadrature rule, the calculation of quadratic form $u^\dagger f(H)u$, which is the main part of finite temperature Lanczos method, can be reduced to a one-dimensional Gauss quadrature. The calculation of more general form $u^\dagger f(H)Ag(H)v$ can be reduced to a two-dimensional Gauss quadrature. Then we showed that many finite temperature static and dynamic quantities can be calculated by one- or two-dimensional Gauss quadratures.

Our development of FTLM is not to improve numerically the original FTLM introduced by Jaklič and Prelovšek, since both methods admit same numerical results. The advantage of this Gauss quadrature based framework is that it fills the conceptual gap between FTLM and KPM, and makes it easy to apply orthogonal polynomial techniques commonly used by KPM in the FTLM calculation. One unexplored extension of this framework is applying different kernels in FTLM to reduce Gibbs oscillation in the expansion of incontinuous functions. We believe that after this development, FTLM will find more applications in the calculations of quantum many-body systems.

Appendix A: Two theorems on orthogonal polynomials

Theorem 1 The zeros of $p_n(x)$ are the eigenvalues of the tridiagonal matrix

$$T_n = \begin{bmatrix} \alpha_0 & \beta_1 & 0 & \dots & 0 \\ \beta_1 & \alpha_1 & \beta_2 & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \alpha_{n-2} & \beta_{n-1} \\ 0 & \dots & 0 & \beta_{n-1} & \alpha_{n-1} \end{bmatrix}. \tag{A1}$$

Proof Define two vectors of length n

$$\boldsymbol{P}(x) = (p_0(x), p_1(x), \dots, p_{n-1}(x))^{\mathrm{T}},$$
$$\boldsymbol{E}_n = (0, 0, \dots, 0, 1)^{\mathrm{T}},$$

then the recurrence relation

$$\beta_{n+1} p_{n+1}(x) = (x - \alpha_n) p_n(x) - \beta_n p_{n-1}(x) \tag{A2}$$

can be written in a vector form

$$x\boldsymbol{P}(x) = T_n \boldsymbol{P}(x) + \beta_n p_n(x) \boldsymbol{E}_n. \tag{A3}$$

Since $p_n(x_j) = 0$ for all $0 \le j \le n-1$, equation (A3) at $x = x_j$ becomes

$$x_j \boldsymbol{P}(x_j) = T_n \boldsymbol{P}(x_j), \quad 0 \le j \le n-1. \tag{A4}$$

Hence the zeros of $p_n(x)$ are the eigenvalues of T_n.

Theorem 2 Let $x_0, x_1, \dots, x_{n-1}$ be the zeros of p_n, then there exists $w_j > 0$ for $j = 0, 1, \dots, n-1$ such that

$$\sum_{j=0}^{n-1} p(x_j) w_j = \langle p(x), p_0 \rangle \tag{A5}$$

for all degree$(p) \le 2n-1$.

Proof Since T_n is a symmetric matrix, it has the following eigenvalue decomposition:

$$T_n = S \Lambda S^{\dagger}, \tag{A6}$$

where $\Lambda = \mathrm{diag}(x_0, x_1, \dots, x_{n-1})$ and S is a unitary matrix, having the form

$$S = \begin{bmatrix} \frac{p_0(x_0)}{a_0} & \frac{p_0(x_1)}{a_1} & \dots & \frac{p_0(x_{n-1})}{a_{n-1}} \\ \frac{p_1(x_0)}{a_0} & \frac{p_1(x_1)}{a_1} & \dots & \frac{p_1(x_{n-1})}{a_{n-1}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{p_{n-1}(x_0)}{a_0} & \frac{p_{n-1}(x_1)}{a_1} & \dots & \frac{p_{n-1}(x_{n-1})}{a_{n-1}} \end{bmatrix}. \tag{A7}$$

Here

$$a_j = \mathrm{e}^{\mathrm{i}\phi_j} \sqrt{\sum_{i=0}^{n-1} p_i^2(x_j)} \tag{A8}$$

and $\mathrm{e}^{\mathrm{i}\phi_j}$ is a phase factor for each eigenvector.

If we define w_j as

$$w_j = \frac{p_0(x_j)}{|a_j|^2} = \frac{1}{|a_j|^2} = |S_{0,j}|^2, \tag{A9}$$

it can be varified that for $0 \le k \le n-1$, we have

$$\sum_{j=0}^{n-1} p_k(x_j) w_j = \sum_{j=0}^{n-1} \frac{p_k(x_j) p_0(x_j)}{|a_j|^2} = \sum_{j=0}^{n-1} S_{k,j}^* S_{0,j} = \delta_{k0}, \tag{A10}$$

which indicates that equation (A5) holds true for degree$(p) \le n-1$. Now suppose $n \le$ degree$(p) \le 2n-1$, according to polynomial remainder theorem p can be written as

$$p = p_n q_m + r_l, \tag{A11}$$

where

$$\mathrm{degree}(q_m) = m \le n-1,$$
$$\mathrm{degree}(r_l) = l < m, \tag{A12}$$

then since p_n is orthogonal to polynomials of degree less than n, we have

$$\langle p, p_0 \rangle = \langle p_n q_m, p_0 \rangle + \langle r_l, p_0 \rangle = \langle p_n, q_m p_0 \rangle + \langle r_l, p_0 \rangle = \langle r_l, p_0 \rangle. \tag{A13}$$

On the other hand, since x_j is the root of p_n, we have

$$\sum_{j=0}^{n-1} p(x_j) w_j = \sum_{j=0}^{n-1} \left[p_n(x_j) q_m(x_j) + r_l(x_j) \right] w_j = \sum_{j=0}^{n-1} r_l(x_j) w_j = \langle r_l, p_0 \rangle = \langle p, p_0 \rangle, \tag{A14}$$

which completes the proof of Eq. (A5).

Acknowledgements

This work is supported by the National Natural Science Foundation of China (Grant Nos. 11734002 and U1930402). All numerical computations were carried out on the Tianhe-2JK at the Beijing Computational Science Research Center (CSRC).

References

[1] Lin H Q 1990 *Phys. Rev. B* **42** 6561
[2] Lin H Q and Gubernatis J 1993 *Comput. Phys.* **7** 400
[3] Zhang J M and Dong R X 2010 *European Journal of Physics* **31** 591
[4] Nataf P and Mila F 2014 *Phys. Rev. Lett.* **113** 127204
[5] Schollwöck U 2005 *Rev. Mod. Phys.* **77** 259
[6] Schollwöck U 2011 *Annals of Physics* **326** 96
[7] White S R 1992 *Phys. Rev. Lett.* **69** 2863
[8] Gubernatis J, Kawashima N and Werner P 2016 *Quantum Monte Carlo Methods: Algorithms for Lattice Models* (Cambridge University Press)

[9] Loh E Y, Gubernatis J E, Scalettar R T, White S R, Scalapino D J and Sugar R L 1990 *Phys. Rev. B* **41** 9301

[10] Mak C H and Chandler D 1990 *Phys. Rev. A* **41** 5709

[11] Lanczos C 1950 *Journal of Research of the National Bureau of Standards* **45** 255

[12] PAIGE C C 1972 *IMA Journal of Applied Mathematics* **10** 373

[13] Golub G and Underwood R 1977 *The block lanczos method for computing eigenvalues Mathematical Software* ed Rice J R (Academic Press) pp. 361–377

[14] Komzsik L 2003 *The Lanczos method: evolution and application (SIAM)*

[15] Weiße A, Wellein G, Alvermann A and Fehske H 2006 *Rev. Mod. Phys.* **78** 275

[16] Weiße A 2009 *Phys. Rev. Lett.* **102** 150604

[17] Silver R N and Röder H 1997 *Phys. Rev. E* **56** 4822

[18] Silver R and Röder H 1994 *International Journal of Modern Physics C* **5** 735

[19] Prelovšek P and Bonča J 1994 *Phys. Rev. B* **49** 5065

[20] Prelovšek P and Bonča J 2013 *Ground State and Finite Temperature Lanczos Methods* (Berlin, Heidelberg: Springer Berlin Heidelberg) pp. 1–30

[21] Schlüter H, Gayk F, Schmidt H J, Honecker A and Schnack J 2021 *Zeitschrift fúr Naturforschung A* **76** 823

[22] Schnack J, Richter J and Steinigeweg R 2020 *Phys. Rev. Res.* **2** 013186

[23] Golub G H and Meurant G A 2010 *Matrices, moments, and quadrature with applications Princeton series in applied mathematics* (Princeton, N.J: Princeton University Press)

[24] Golub G H and Van Loan C F 2013 *Matrix computations* 4th ed Johns Hopkins studies in mathematical sciences (Baltimore, Md: Johns Hopkins Univ. Press)

[25] Stoer J and Bulirsch R 2002 *Introduction to Numerical Analysis* (New York, NY: Springer New York)

[26] Golub G H and Welsch J H 1969 *Mathematics of Computation* **23** 221

[27] Jie S and Tao T 2006 *Spectral and High-Order Methods with Applications Mathematics Monograph* Series 3 (Beijing: Science Press)

[28] It can be stated more rigorously that higher order expansion of f does not contribute to the result of $u^\dagger f(H)u$. All functions $\hat{f}$ with same expansion up to order $N-1$ form a equivalent class as f.

[29] Same as Gauss quadrature in one-dimensional case, higher order expansion of $C(x,y)$ does not contribute to $u^\dagger f(H)Ag(H)v$.

[30] Skilling J (ed) 1989 *Maximum Entropy and Bayesian Methods* (Springer Netherlands)

[31] Hewitt E and Hewitt R E 1979 *Archive for History of Exact Sciences* **21** 129

[32] Aichhorn M, Daghofer M, Evertz H G and von der Linden W 2003 *Phys. Rev. B* **67** 161103

[33] Lieb E, Schultz T and Mattis D 1961 *Annals of Physics* **16** 407

[34] Niemeijer T 1967 *Physica* **36** 377