Chapter 1

Matrices, Moments and Quadrature: Applications to Time-Dependent Partial Differential Equations

James V. Lambers, Alexandru Cibotarica and Elisabeth M. Palchak

Additional information is available at the end of the chapter

http://dx.doi.org/10.5772/62247

Abstract

The numerical solution of a time-dependent PDE generally involves the solution of a stiff system of ODEs arising from spatial discretization of the PDE. There are many methods in the literature for solving such systems, such as exponential propagation iterative (EPI) methods, that rely on Krylov projection to compute matrix function-vector products. Unfortunately, as spatial resolution increases, these products require an increasing number of Krylov projection steps, thus drastically increasing computational expense.

This paper describes a modification of EPI methods that uses Krylov subspace spectral (KSS) methods, to compute these matrix function-vector products. KSS methods represent a balance between the efficiency of explicit methods and the stability of implicit methods. This balance is achieved by approximating the matrix exponential with different polynomials for each Fourier coefficient of the solution. These polynomials arise from techniques due to Golub and Meurant for computing bilinear forms involving matrix functions by treating them as Riemann-Stieltjes integrals, which are then approximated using Gaussian quadrature rules.

This paper describes how the nodes for the quadrature rules required by KSS methods can be estimated very rapidly through asymptotic analysis of block Lanczos iteration, thus drastically reducing computational expense without sacrificing accuracy. Numerical experiments demonstrate that this modification causes the number of Krylov projection steps to become bounded independently of the grid size, thus dramatically improving efficiency and scalability.

Keywords: exponential propagation iterative methods, Krylov subspace spectral methods, stiffness, Gaussian quadrature, Lanczos iteration

1. Introduction

Consider an autonomous system of ordinary differential equations (ODEs) of the form

$$\mathbf{y}' = F(\mathbf{y}), \quad \mathbf{y}(t_0) = \mathbf{y}_0, \tag{1}$$

such as one that would naturally arise from spatial discretization of a partial differential equation (PDE). Such systems are generally stiff when the underlying PDE has significant diffusive or advective terms, and this stiffness is exacerbated by increasing the spatial resolution. Stiffness leads to significantly increased computational expense for both explicit and implicit time-stepping methods. For explicit methods, the time step is severely restricted by the CFL condition, while implicit methods require the solution of ill-conditioned systems of linear equations during each time step. Such systems are generally sparse and therefore best suited for iterative methods, but for these stiff systems of ODEs, iterative methods require many iterations or a specially developed preconditioner [1].

Exponential propagation iterative (EPI) methods, introduced by Tokman et al. [1, 2], are Runge-Kutta like time-stepping methods that are designed to minimize the number of matrix function-vector products of the form $\mathbf{w} = \varphi(A\tau)\mathbf{b}$, where φ is a smooth function, A is a square matrix, τ is a parameter determined by the time step, and $\mathbf{b}$ is a column vector. The approach used by EPI methods to compute $\mathbf{w}$ for a given symmetric matrix A is to apply the Lanczos algorithm to A with the initial vector $\mathbf{b}$, until we obtain a matrix X_j with orthonormal columns and a tridiagonal matrix T_j such that $X_j^T A X_j = T_j$. Then, we can compute the approximation

$$\mathbf{w}_j = \| \mathbf{b} \|_2 \, X_j \varphi(T_j \tau)\mathbf{e}_1, \tag{2}$$

where $\mathbf{e}_1 = [1 \quad 0 \quad \cdots \quad 0]^T$. Since each column $\mathbf{x}_k$ of the matrix X_j is of the form $\mathbf{x}_k = p_{k-1}(A)\mathbf{b}$, where $p_n(A)$ is a polynomial of degree n in A, $\mathbf{w}_j$ is the product of a polynomial in A of degree $j - 1$ and $\mathbf{b}$. Since the matrix A arises from a stiff PDE, the eigenvalues of A are distributed over several orders of magnitude. As it is generally not possible to accurately approximate $\varphi(\lambda)$ on such a large interval with a low-degree polynomial, a large number of Lanczos iterations is generally necessary in order to obtain a sufficiently accurate of $\mathbf{w}$.

The difficulty that time-stepping methods have with stiffness stems from the fact that low- and high-frequency components of the solution, which change at widely varying speeds, are coupled and therefore must be evolved using a common time step. While the low-frequency components generally make the most significant contribution to the solution, the high-frequency components change most rapidly, and therefore constrain the time step. The greater

http://dx.doi.org/10.5772/62247

the spatial resolution, the greater the bandwidth of the solution, thus constraining the time step even further.

This coupling, however, is not the only cause of the greater computational expense incurred by time-stepping methods applied to stiff systems. A key contributing factor is the use of the same polynomial or rational function to approximate all of these components of $\varphi(A\tau)\mathbf{b}$, when such a function cannot effectively approximate $\varphi(\lambda\tau)$ on a large interval except at high degree, which increases computational expense. An alternative is to use Krylov subspace spectral (KSS) methods [3, 4], which employ a component-wise approach to these matrix function-vector products. In KSS methods, which are explicit, each Fourier coefficient of the solution is computed using an interpolating polynomial with frequency-dependent interpolation points. This individualized approach to computing each Fourier component yields high-order accuracy and stability like that of implicit methods, even though KSS methods themselves are explicit.

To date, KSS methods have been applied mostly to linear PDEs on d-dimensional boxes, for $d = 1, 2, 3$, with either periodic or homogeneous Dirichlet or Neumann boundary conditions. A first-order KSS method was applied to nonlinear diffusion equations for image processing by Guidotti et al. [5], but for that problem, a straightforward linearization was used during each time step. In order to compute solutions of nonlinear PDEs with higher-order accuracy in time, it is necessary to treat the nonlinearity more carefully. This can be accomplished by combining KSS methods with another high-order time-stepping method, such as EPI methods.

In this paper, this combination is presented, for the purpose of solving systems of ODEs of the form (1) that are obtained through spatial discretization of nonlinear PDEs defined on rectangular domains with periodic, homogeneous Dirichlet, or homogeneous Neumann boundary conditions. The proposed algorithm, first described in [6, 7], uses the component-wise approach of KSS methods to more efficiently compute matrix function-vector products of the form $\mathbf{y} = \varphi(A\tau)\mathbf{b}$. The following features distinguish the combined approach from previous work on either EPI or KSS methods:

- Instead of applying Krylov projection (e.g., see [8–10]) with initial vector $\mathbf{b}$ to compute $\mathbf{y}$, it is applied only to a low-frequency projection of $\mathbf{b}$, in order to avoid the larger number of iterations that Krylov projection typically incurs at higher spatial resolution. Furthermore, for advection-dominated problems, denoising is applied to the Krylov subspace basis vectors, to eliminate spurious high-frequency oscillations in these vectors which slow convergence.

- For the high-frequency portion of $\mathbf{b}$, a KSS method, as described in [11], is used to apply $\varphi(A\tau)$. In this latest version of KSS methods, each Fourier component of the output vector $\mathbf{y}$ is still approximated using its own block quadrature rule, as before. However, now the quadrature nodes (i.e., frequency-dependent interpolation points for φ) are obtained through high-frequency analysis of block Lanczos iteration, which yields simple formulas for the nodes, instead of having to obtain them by solving an eigenvalue problem [3]. This greatly improves efficiency over the approach used in [3, 4], in which the nodes are obtained

by explicitly applying block Lanczos iteration to A for *each* Fourier component, without sacrificing accuracy [11].

- Once the frequency-dependent interpolation points are obtained, it is necessary to construct and apply frequency-dependent interpolating polynomial approximations $p_\omega(\lambda)$ of φ to A, for each wave number ω, and then the Fourier coefficients of the solution are inner products of the form $\hat{e}_\omega^H\, p_\omega(A\tau)\mathbf{b}$, where $\hat{e}_\omega$ is a discretization of an appropriate Fourier basis function. This paper provides implementation details for this task, and explains how it can be accomplished with as few Fourier transforms as possible [6, 7].

The outline of the paper is as follows. Section 2 gives an overview of KSS methods. Section 3 reviews their acceleration based on asymptotic analysis of recursion coefficients, first presented in [11], and discusses extension to non-self-adjoint operators. Section 4 provides a brief overview of EPI methods and demonstrates the spurious high-frequency oscillations that can occur when using standard Krylov projection within an EPI method. Section 5 describes how KSS and EPI methods are combined. Numerical results are presented in Section 6. Concluding remarks and directions for future work are given in Section 7.

2. Matrices, moments, quadrature, and PDE

To provide context for the latest advancements with KSS methods, we begin with an overview of KSS methods as described in [3], applied to the 1-D parabolic PDE

$$u_t + Lu = 0, \quad 0 < x < 2\pi, \quad t > 0, \quad Lu = -(p(x)u_x)_x + q(x)u, \tag{3}$$

$$u(x,0) = f(x), \quad 0 < x < 2\pi, \tag{4}$$

$$u(0,t) = u(2\pi,t), \quad t > 0. \tag{5}$$

where $p(x) > 0$ and $q(x) \geq 0$ on $[0, 2\pi]$, and $q(x)$ is not identically zero. In KSS methods, each Fourier coefficient of the solution $\tilde{u}(x, t_{n+1})$ is obtained by applying the exact solution operator of the PDE to $\tilde{u}(x, t_n)$. For a given wave number ω, such a Fourier coefficient is given by

$$\hat{u}(\omega,t_{n+1}) = \left\langle \frac{1}{\sqrt{2\pi}} e^{i\omega x}, e^{-L\Delta t}\tilde{u}(x,t_n) \right\rangle, \tag{6}$$

where

http://dx.doi.org/10.5772/62247

$$\langle f, g \rangle = \int_0^{2\pi} \overline{f(x)} g(x) dx$$

is the standard inner product on [0, 2π] and $e^{-L\Delta t}$ is the solution operator of the PDE (3).

2.1. Matrices, moments, and quadrature

The spatial discretization of (6) yields the bilinear form

$$\mathbf{u}^H \varphi(A) \mathbf{v}, \tag{7}$$

where $\mathbf{u} = \frac{1}{\sqrt{2\pi}} e^{i\omega x}$ and $\mathbf{v} = \tilde{u}(x, t_n)$ are N-vectors, $A = L_N$, where L_N is a spectral discretization of L, and $\varphi(\lambda) = e^{-\lambda t}$.

Because the Sturm-Liouville operator L is self-adjoint and positive definite, it follows that the matrix A is symmetric positive definite. As such, it has real eigenvalues $b = \lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_N = a > 0$, and corresponding orthonormal eigenvectors $\mathbf{q}_j$, $j = 1, \ldots, N$. From the spectral decomposition of A, we obtain the following representation of (6):

$$\mathbf{u}^H \varphi(A) \mathbf{v} = \sum_{j=1}^{N} \varphi(\lambda_j) \mathbf{u}^H \mathbf{q}_j \mathbf{q}_j^H \mathbf{v}. \tag{8}$$

As described by Golub and Meurant in [12], (8) can be viewed as a Riemann-Stieltjes integral

$$\mathbf{u}^H \varphi(A) \mathbf{v} = \int_a^b \varphi(\lambda) d\alpha(\lambda), \tag{9}$$

where the measure $\alpha(\lambda)$ is defined by

$$\alpha(\lambda) = \begin{cases} 0 & \text{if } \lambda < a \\ \sum_{j=i}^{N} u_j v_j, & \text{if } \lambda_i \leq \lambda < \lambda_{i-1}, \quad u_j = \mathbf{u}^H \mathbf{q}_j, \quad v_j = \mathbf{q}_j^H \mathbf{v}. \\ \sum_{j=1}^{N} u_j v_j, & \text{if } b \leq \lambda \end{cases}$$

This allows approximation of (6) using Gaussian quadrature rules, where the nodes and weights are obtained by applying the Lanczos algorithm to A with initial vectors $\mathbf{u}$ and $\mathbf{v}$ [12].

Figure 1 demonstrates why integrals of the form (8) can be approximated accurately with a small number of nodes in the case where A is a discretization of a differential operator and the

vector **u** is used to extract a particular Fourier coefficient of $f(A)\mathbf{v}$. We examine the distribution $d\alpha(\lambda)$ in the case where $\mathbf{u} = \mathbf{v} = e^{i\omega x}$ for small and large values of ω, and for A discretizing a differential operator of the form $-\partial_x\, a(x)\partial_x$, with $a(x) > 0$ being a smooth function or a piecewise constant function. In either case, $d\alpha(\lambda)$ is mostly concentrated within a portion of the interval of integration $[a, b]$. Gaussian quadrature rules for such integrals naturally target these relevant portions [3, 13].

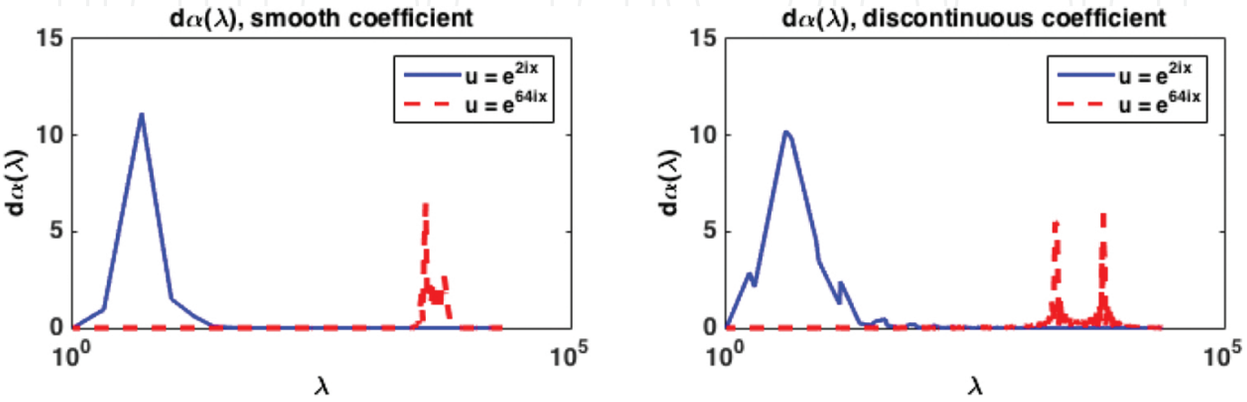

Figure 1. The distribution $d\alpha(\lambda)$ from (8) where the matrix A represents a spectral discretization of a 1-D, second-order differential operator with smooth leading coefficient (top plot) and discontinuous leading coefficient (bottom plot), where **u** = **v** is a discretization of e^{2ix} (solid curve) or e^{64ix} (dashed curve).

Block Lanczos algorithm

$X_0 = 0$, $R_0 = [\mathbf{u} \quad \mathbf{v}]$, $R_0 = X_1 B_0$ (QR factorization)

for $n = 1, 2, \ldots, K$

$V = AX_n$

$M_n = X_n^H V$

$R_n = V - X_{n-1} B_{n-1}^H - X_n M_n$

$R_n = X_{n+1} B_n$ (QR factorization)

end

In the case where $\mathbf{u} \neq \mathbf{v}$, the presence of a negative weight would destabilize the quadrature rule [14]. Alternatively, we consider the approximation of the 2×2 matrix integral

$$[\mathbf{u} \quad \mathbf{v}]^H \varphi(A)[\mathbf{u} \quad \mathbf{v}]. \tag{10}$$

We use the most general K-node quadrature formula, as described in [12], to get an approximation for (9) of the form

http://dx.doi.org/10.5772/62247

$$\int_a^b \varphi(\lambda) d\mu(\lambda) = \sum_{j=1}^{2K} \varphi(\lambda_j) \mathbf{v}_j \mathbf{v}_j^H + error, \tag{11}$$

where, for each j, λ_j is a scalar and $\mathbf{v}_j$ is a 2-vector. Each node λ_j is an eigenvalue of the matrix

$$T_K = \begin{bmatrix} M_1 & B_1^H & & \\ B_1 & M_2 & B_2^H & \\ & \ddots & \ddots & \ddots \\ & & B_{K-1} & M_K \end{bmatrix}, \tag{12}$$

which is a block-tridiagonal matrix of order $2K$. The vector $\mathbf{v}_j$ consists of the first two elements of the corresponding normalized eigenvector. The matrices M_j and B_j are computed using the block Lanczos algorithm [15], described below.

2.2. Krylov subspace spectral methods

The block KSS method [3, 4] for (3) begins by defining

$$R_0 = \begin{bmatrix} \hat{\mathbf{e}}_\omega & \mathbf{u}^n \end{bmatrix}, \tag{13}$$

where $\hat{\mathbf{e}}_\omega$ is a discretization of $\frac{1}{\sqrt{2\pi}} e^{i\omega x}$ and $\mathbf{u}^n$ is a discretization of the approximate solution $u(x, t)$ at time $t_n = n\Delta t$. Next, we compute the QR factorization of R_0,

$$R_0 = X_1(\omega) B_0(\omega)$$

which outputs

$$X_1(\omega) = \begin{bmatrix} \hat{\mathbf{e}}_\omega & \dfrac{\mathbf{u}_\omega^n}{\| \mathbf{u}_\omega^n \|_2} \end{bmatrix} \tag{14}$$

and

$$B_0(\omega) = \begin{bmatrix} 1 & \hat{\mathbf{e}}_\omega^H \mathbf{u}^n \\ 0 & \| \mathbf{u}_\omega^n \|_2 \end{bmatrix}, \tag{15}$$

where

$$\mathbf{u}_{\omega}^{n} = \mathbf{u}^{n} - \hat{\mathbf{e}}_{\omega}\hat{\mathbf{e}}_{\omega}^{H}\mathbf{u}^{n} = \mathbf{u}^{n} - \hat{\mathbf{e}}_{\omega}\hat{u}(\omega, t_n). \tag{16}$$

Then we apply the block Lanczos algorithm [15] to the matrix L_N, which comes from the discretization of L, with initial block $X_1(\omega)$. This produces a block tridiagonal matrix T_K of the form (12), where every entry of T_K is a function of ω. Then, at time t_{n+1}, each Fourier coefficient of the solution is

$$[\hat{\mathbf{u}}^{n+1}]_{\omega} = [B_0(\omega)^H E_{12}^H \exp[-\mathcal{T}_K(\omega)\Delta t]E_{12}B_0(\omega)]_{12}, \quad E_{12} = \begin{bmatrix}\mathbf{e}_1 & \mathbf{e}_2\end{bmatrix}. \tag{17}$$

An inverse FFT applied to the vector of Fourier coefficients $\hat{\mathbf{u}}^{n+1}$ yields the vector $\hat{\mathbf{u}}^{n+1}$, which consists of the values of the solution $u(x, t_{n+1})$ at the grid points.

This algorithm has temporal accuracy $O(\Delta t^{2K-1})$ for parabolic problems [3]. Even higher-order accuracy, $O(\Delta t^{4K-2})$, is obtained for the second-order wave equation [4], due to the second-order time derivative. Furthermore, under appropriate assumptions on the coefficients of the PDE, the 1-node KSS method is unconditionally stable [3, 4]. More generally, the temporal order of accuracy is $O(\Delta t^{(2K-1)d})$, where d is the highest order of a time derivative in the PDE; this order of accuracy has been observed with the Schrödinger equation [16] and Maxwell's equations [18].

3. Asymptotic analysis of block Lanczos iteration

For each Fourier coefficient of the solution, KSS methods use a different quadrature rule, because the measure $\alpha(\lambda)$ in (8) is defined in terms of the frequency. It follows that if $S(L; \Delta t)$ is the solution operator of the PDE (e.g. $S(L; \Delta t) = e^{-L\Delta t}$ for the PDE (3)), then the function $S(\lambda; \Delta t)$ is approximated by a polynomial of degree $2K$ that interpolates $S(\lambda; \Delta t)$ at different points for each frequency, which are the block Gaussian quadrature nodes λ_j in (11). Taking all Fourier coefficients together, the computed solution at time t_{n+1}, $\mathbf{u}^{n+1}$ can be expressed as

$$\mathbf{u}^{n+1} = \sum_{j=0}^{2K-1} D_j(\Delta t) L_N^j \mathbf{u}^n,$$

where L_N is a discretization of L on an N-point grid and $D_j(\Delta t)$ is a matrix that is diagonal in discrete Fourier space. The diagonal entries are the coefficients of these frequency-dependent interpolating polynomials in power form, with each row corresponding to a different frequency.

In the block KSS method described in [3, 4] and reviewed in the previous section, these interpolation points are the eigenvalues of the block tridiagonal matrix T_K from (12) that is produced by block Lanczos iteration. Although each such matrix is small, computing the eigenvalues and eigenvectors for *each* frequency is computationally expensive. Therefore, in

http://dx.doi.org/10.5772/62247

this section, we describe a much faster approach to obtaining estimates of these nodes, at least for high frequencies. This approach was first presented in [11] and generalized in [6, 7]. The basic idea is to examine the entries of T_K as $|\omega| \to \infty$, where ω is the wave number.

3.1. The block case

As in the previous section, let $\mathbf{u}^n$ be a discretization of the approximate solution $u(x, t)$ at time $t_n = n\Delta t$ on a uniform N-point grid. Then, KSS methods use the initial block $R_0 = \left[\hat{\mathbf{e}}_\omega \quad \mathbf{u}^n\right]$, for each $\omega = -N/2 + 1, \ldots, N/2$. We start the first iteration of the block Lanczos algorithm by finding the QR-factorization of R_0:

$$R_0 = X_1 B_0,$$

where

$$\mathrm{X}_1 = \begin{bmatrix} \hat{\mathbf{e}}_\omega & \dfrac{\mathbf{u}_\omega^n}{\mathsf{P}\mathbf{u}_\omega^n \mathsf{P}_2} \end{bmatrix} \text{ and } B_0 = \begin{bmatrix} 1 & \hat{u}(\omega, t_n) \\ 0 & \mathsf{P}\mathbf{u}_\omega^n \mathsf{P}_2 \end{bmatrix}. \tag{18}$$

with $\mathbf{u}_\omega^n$ is defined as in (16). We note that if the solution u is continuous, then as $|\omega| \to \infty$, $|\hat{u}^n(\omega)| \to 0$, so that in the limit B_0 is diagonal.

The next step is to compute

$$M_1 = X_1^H L_N X_1, \tag{19}$$

where the matrix L_N is a spectral discretization of the operator L defined by $Lu = pu_{xx} + q(x)u$, with p being a constant. Substituting the value of X_1 from (18) into (19) yields

$$M_1 = \begin{bmatrix} \omega^2 p + \bar{q} & \dfrac{\widehat{L_N \mathbf{u}_\omega^n}(\omega)}{\| \mathbf{u}_\omega^n \|_2} \\ \overline{\dfrac{\widehat{L_N \mathbf{u}_\omega^n}(\omega)}{\| \mathbf{u}_\omega^n \|_2}} & R(L_N, \mathbf{u}_\omega^n) \end{bmatrix},$$

where $\bar{q}$ is the mean of $q(x)$ on $(0, 2\pi)$, $\widehat{L_N \mathbf{u}_\omega^n}(\omega) = \hat{\mathbf{e}}_\omega^H L_N \mathbf{u}_\omega^n$ is the Fourier coefficient of the grid function $L_N \mathbf{u}^n$ corresponding to the wave number ω, and $R\left(L_N, \mathbf{u}_\omega^n\right) = \dfrac{\left\langle \mathbf{u}_\omega^n, L_N \mathbf{u}_\omega^n \right\rangle}{\left\langle \mathbf{u}_\omega^n, \mathbf{u}_\omega^n \right\rangle}$ is the Rayleigh quotient of L_N and $\mathbf{u}_\omega^n$. As $|\omega|$ increases, the Fourier coefficients of a continuous

function decay to zero; therefore, as long as the solution is sufficiently regular, the non-diagonal entries of M_1 become negligible, that is,

$$M_1 \approx \begin{bmatrix} \omega^2 p + \bar{q} & 0 \\ 0 & R(L_N, \mathbf{u}^n) \end{bmatrix}.$$

Proceeding with the iteration, and neglecting any terms that are Fourier coefficients or are of lower order in ω, we obtain

$$R_1 = L_N X_1 - X_1 M_1 \approx \begin{bmatrix} \tilde{\mathbf{q}}\hat{\mathbf{e}}_\omega & \dfrac{L_N \mathbf{u}_\omega^n}{\| \mathbf{u}_\omega^n \|_2} - R(L_N, \mathbf{u}_\omega^n) \dfrac{\mathbf{u}_\omega^n}{\| \mathbf{u}_\omega^n \|_2} \end{bmatrix},$$

where $\mathbf{q}$ is a vector consisting of the value of $q(x)$ at the grid points, $\tilde{\mathbf{q}} = \mathbf{q} - \bar{q}$, and multiplication of vectors is component-wise.

To obtain X_2, we perform the QR-factorization $R_1 = X_2 B_1$. We note that the (1, 2) entry of B_1, modulo lower-order terms, is the Fourier coefficient $\hat{v}_1(\omega)$, where

$$\mathbf{v}_1 = \tilde{\mathbf{q}} \left(\frac{L_N \mathbf{u}_\omega^n}{\| \mathbf{u}_\omega^n \|_2} - R(L_N, \mathbf{u}_\omega^n) \frac{\mathbf{u}_\omega^n}{\| \mathbf{u}_\omega^n \|_2} \right).$$

It follows that if the coefficient $q(x)$ and solution $u(x, t)$ are sufficiently regular, then B_1 approaches a diagonal matrix as $|\omega| \to \infty$, just as B_0 does. Continuing this process, it can be shown that given sufficient regularity of the solution and coefficients of L, each block M_j or B_j of T_K from (12) becomes approximately diagonal at high frequencies.

Because $[\mathsf{T}_K]_{ij} \approx 0$ when $i + j$ is odd in the high-frequency case, it follows that if the rows and columns of T_K are permuted so that odd-numbered and even-numbered rows and columns are grouped together and in order, then the eigenvalue problem for T_K approximately *decouples*. Therefore, approximate eigenvalues of T_K can be obtained by computing the eigenvalues of the tridiagonal matrices obtained by performing "non-block" Lanczos on L_N with initial vectors equal to the two columns of R_0 separately, rather than applying block Lanczos with these two columns together in the initial block. In [11], it is shown that this decoupling also takes place if the leading coefficient $p(x)$ of L is *not* constant.

http://dx.doi.org/10.5772/62247

3.2. The non-block case

The decoupling observed in the preceding discussion reveals that we can obtain approximations of half of the block Gaussian quadrature nodes for all Fourier coefficients by applying "non-block" Lanczos iteration to the matrix L_N with initial vector $\mathbf{u}^n$, the computed solution, as is done in Krylov projection methods such as those described in [8–10]. These nodes, denoted by $\lambda_1, \ldots, \lambda_m$ where m is the number of iterations, will be referred to as *frequency-independent nodes*. Because this iteration does not depend on the wave number ω, the frequency-independent nodes need only be computed once for each vector $\mathbf{u}$ for which an expression of the form $\varphi(L_N \tau)\mathbf{u}^n$ is required. The other half of the nodes can be estimated through an asymptotic analysis of Lanczos iteration applied to L_N with initial vector $\hat{\mathbf{e}}_\omega$ [11]; these are called *frequency-dependent nodes* and will be denoted by $\lambda_{1,\omega}, \ldots, \lambda_{m,\omega}$.

As an example, we consider the case where the matrix A comes from a spectral discretization of the operator $Lu = -pu_{xx} + q(x)u$, where p is a constant, and assuming periodic boundary conditions [11]. Carrying out three iterations, which corresponds to a fifth-order accurate KSS method for a parabolic PDE, yields the following recursion coefficients as functions of the wave number ω, after neglecting lower-order terms:

$$\begin{bmatrix} \alpha_1 & \overline{\beta_1} & 0 \\ \beta_1 & \alpha_2 & \overline{\beta_2} \\ 0 & \beta_2 & \alpha_3 \end{bmatrix} \approx \begin{bmatrix} p\omega^2 & \| \tilde{\mathbf{q}} \|_2 & 0 \\ \| \tilde{\mathbf{q}} \|_2 & p\omega^2 & 2p \mid \omega \mid \| \mathbf{q}_x \|_2 / \| \tilde{\mathbf{q}} \|_2 \\ 0 & 2p \mid \omega \mid \| \mathbf{q}_x \|_2 / \| \tilde{\mathbf{q}} \|_2 & p\omega^2 \end{bmatrix}.$$

It follows that the frequency-dependent nodes can easily be estimated as

$$\lambda_{1,\omega} = p\omega^2, \quad \lambda_{i,\omega} = p\omega^2 \pm \sqrt{\beta_1^2 + \beta_2^2}, \quad i = 2,3 \tag{20}$$

whereas for a third-order KSS method, the frequency-dependent nodes are

$$\lambda_{1,\omega} = p\omega^2 + \beta_1, \quad \lambda_{2,\omega} = p\omega^2 - \beta_1.$$

In [6, 7], similar formulas for the nodes are derived for a PDE with homogeneous Neumann boundary conditions, and for a 2-D PDE with periodic boundary conditions.

When the matrix A is a finite-difference representation of the underlying differential operator, the block Gaussian quadrature nodes can be represented more accurately if formulas for the eigenvalues of symmetric Toeplitz matrices are used for the leading-order terms in the nodes. For example, in (20), $p\omega^2$ is replaced by $2p(N/\pi)^2(1 - \cos(\pi\omega/N))$, where N is the number of grid points.

3.3. Non-self-adjoint operators

The theory developed in [12] applies to symmetric positive definite matrices, but this property is not essential [17, 18]. That said, care must be taken with nonsymmetric matrices, as a straightforward use of unsymmetric Lanczos to treat bilinear forms as Riemann-Stiejtjes integrals, as described in [19], can suffer from "serious breakdown" [20]. Therefore, a more robust approach for applying KSS methods to linear PDE with non-self-adjoint spatial differential operators is to use Arnoldi iteration to approximate $\varphi(A\tau)\mathbf{b}$. The algorithm for Arnoldi iteration, applied to a matrix A and initial vector $\mathbf{z}_0$, is as follows:

Arnoldi iteration

$v_1 = \mathbf{z}_0 / \| \mathbf{z}_0 \|_2$
for $j = 1, 2, \ldots$
  $\mathbf{z}_j = A\mathbf{v}_j$
  for $k = 1, 2, \ldots, j$
    $h_{kj} = \mathbf{v}_k^H \mathbf{z}_j$
    $\mathbf{z}_j = \mathbf{z}_j - h_{kj}\mathbf{v}_k$
  end
  $h_{j+1,j} = \| \mathbf{z}_j \|_2$
  $\mathbf{v}_{j+1} = \mathbf{z}_j / h_{j+1,j}$
end

The output of Arnoldi iteration is an upper Hessenberg matrix H_m, and a matrix V_m with orthonormal columns, such that

$$AV_m = V_m H_m + h_{m+1,m}\mathbf{z}_{m+1}\mathbf{e}_m^H. \tag{21}$$

By analogy with (17), to approximate $\mathbf{u}^{n+1} = \varphi(A\tau)\mathbf{u}^n$, we can compute each Fourier coefficient $[\hat{\mathbf{u}}^{n+1}]_\omega$ of $\mathbf{u}^{n+1}$ by applying *block* Arnoldi iteration [21] to A, with initial block $R_0(\omega)$ as defined in (13). After m iterations, this iteration produces a block upper Hessenberg matrix $H_m(\omega)$, from which we obtain

$$[\hat{\mathbf{u}}^{n+1}]_\omega = \left[B_0(\omega)^H E_{12}^H \varphi(H_m(\omega)\tau) E_{12} B_0(\omega) \right]_{12}, \tag{22}$$

where $B_0(\omega)$ and E_{12} are as defined in (15) and (17), respectively. As shown in [6, 7], like with block Lanczos, the eigenvalue problem for $H_m(\omega)$ approximately decouples for high frequen-

http://dx.doi.org/10.5772/62247

cies, due to the decay of the Fourier coefficients of $\mathbf{u}^n$. It follows that we can easily estimate the frequency-dependent eigenvalues of $H_m(\omega)$, which are used as interpolation points for a polynomial approximation of $\varphi(\lambda)$, by applying "non-block" Arnoldi iteration, as described in the above algorithm, with an initial vector $\mathbf{z}_0$ chosen to be a discretization of an appropriate Fourier basis function. Additional details can be found in [6, 7].

4. EPI methods

In this section, we give an overview of EPI methods, as developed by Tokman et al. in [1, 2]. To solve an autonomous system of ODEs of the form (1), a time step from time t_n to time $t_{n+1} = t_n + \Delta t$ is taken as follows. First, the time derivative $F(\mathbf{y})$ is expressed in terms of its Taylor expansion around $\mathbf{y}(t_n)$, which yields

$$\frac{d\mathbf{y}}{dt} = F(\mathbf{y}(t_n)) + A_n(\mathbf{y}(t) - \mathbf{y}(t_n)) + R(\mathbf{y}(t)), \tag{23}$$

where A_n is the Jacobian matrix of $\mathbf{F}(\mathbf{y})$ evaluated at $\mathbf{y}(t_n)$, and $R(\mathbf{y}(t))$ is the Taylor remainder. Next, we multiply both sides by an integrating factor $e^{-A_n t}$ and then integrate from t_n to t_{n+1} to obtain

$$\mathbf{y}(t_{n+1}) = \mathbf{y}(t_n) + [e^{A_n \Delta t} - I]A_n^{-1}F(\mathbf{y}(t_n)) + \int_{t_n}^{t_{n+1}} e^{A_n(t_{n+1}-\tau)}R(\mathbf{y}(\tau))d\tau. \tag{24}$$

The integral on the right side is then approximated using a quadrature rule. This requires the evaluation of matrix function-vector products of the form $\varphi(A\tau)\mathbf{b}$, with $A = A_n$, for various "φ-functions" and various choices of the vectors $\mathbf{b}$ and scaling factors τ, which are determined so as to satisfy order conditions.

Any such matrix function-vector product is computed using Krylov projection. Arnoldi iteration is applied to A (or Lanczos iteration, if A is symmetric) with initial vector $\mathbf{b}$. After m iterations, we obtain (21), from which we obtain an approximation

$$\varphi(A\tau)\mathbf{b} \approx \| \mathbf{b} \|_2 \, V_m \varphi(H_m \tau)\mathbf{e}_1, \tag{25}$$

where, as before, H_m is upper Hessenberg (or tridiagonal of A is symmetric), and the columns of V_m form an orthonormal basis for the Krylov subspace

$$\mathcal{K}(A, \mathbf{b}, m) = \operatorname{span}\{\mathbf{b}, Ab, A^2\mathbf{b}, \ldots, A^{m-1}\mathbf{b}\}.$$

The accuracy of this approximation is discussed in [9]. In the case where A is ill-conditioned, the number of iterations m needed for convergence of (25) can be quite large, and this is exacerbated by increasing the spatial resolution in the discretization of the underlying PDE from which (1) arises.

When the number of iterations is large, an additional issue, particularly for advection-dominated problems, is the appearance of spurious high-frequency oscillations in the columns of V_m, even if the initial vector **b** represents a smooth function. As can be seen in **Figure 2**, even after relatively few iterations, these oscillations can occur and reduce the columns of V_m to essentially noise.

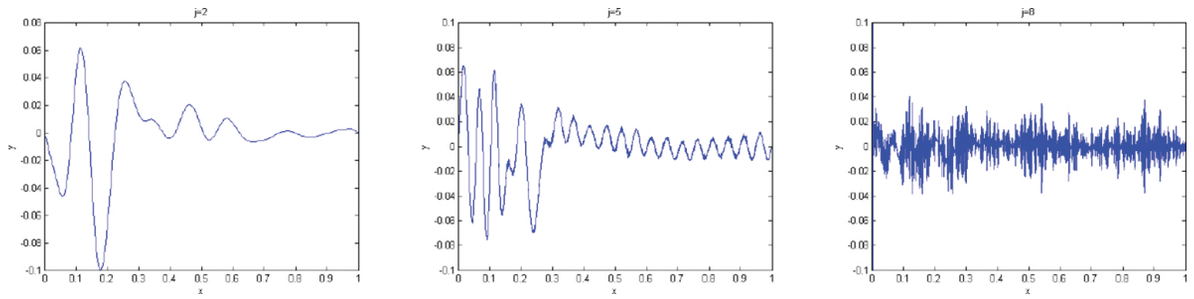

Figure 2. Columns of V_m from (25) generated by Arnoldi iteration applied to the matrix from Burgers' equation (see Section 6.2).

This can be alleviated by filtering out high-frequency components of the columns of V_m after each matrix-vector multiplication. As can be seen in **Figure 3**, the spurious high-frequency oscillations are reduced, resulting in more meaningful data in V_m. Future work will include the automatic, adaptive selection of an appropriate threshold for filtering high-frequency components.

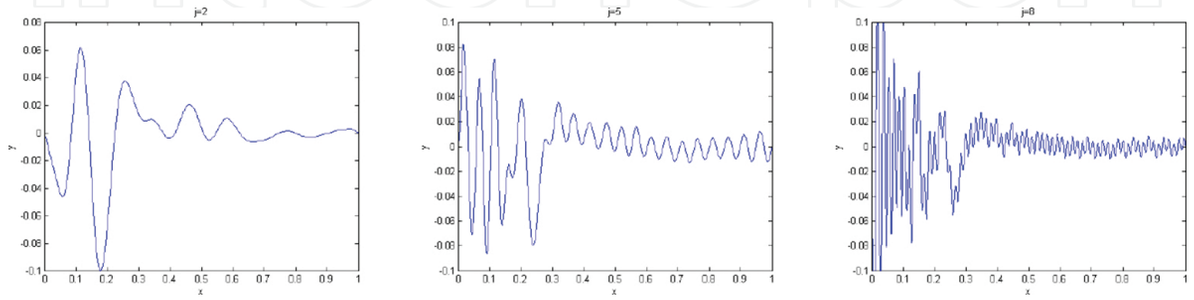

Figure 3. Columns of V_m from (25) generated by Arnoldi iteration, with denoising, applied to the matrix from Burgers' equation (see Section 6.2).

http://dx.doi.org/10.5772/62247

The behavior of the unfiltered Krylov vectors is not surprising, as similar behavior is displayed by the unsmoothed Fourier method applied to hyperbolic PDEs [22]. In that work, the proposed remedies were to either use filtering, or increase the number of grid points; the former remedy serves as the motivation for denoising in this context. It is worth noting that in both the unfiltered and filtered cases, no loss of orthogonality was observed in the Krylov vectors; that is, $1/\sqrt{m}\,\|V_m^T V_m - I_m\|_F$ was negligibly small (that is, on the order of 10^{-10} or smaller) in both cases.

5. KSS-EPI methods

We now describe the combination of KSS and EPI methods. The EPI method itself is not changed; what is modified is the approach to computing any matrix function-vector product of the form $\varphi(A\tau)\mathbf{b}$. The main steps in the new approach are as follows:

1. First, it is necessary to determine a cutoff frequency N_c. In the numerical experiments presented in this paper, the value of N_c has been determined by experimentation; it is demonstrated in [7] that the performance is not unduly sensitive to the choice of N_c. In future work, an adaptive approach to choosing N_c will be developed.

2. Using an FFT, $\mathbf{b}$ is decomposed into low-frequency and high-frequency components $\mathbf{b}_L$ and $\mathbf{b}_H$. Specifically, $\mathbf{b} = \mathbf{b}_L + \mathbf{b}_H$, where each Fourier coefficient of $\mathbf{b}_L$ with wave number $\vec{\omega}$ is zero if $\|\vec{\omega}\|_\infty \geq N_c$.

3. $\varphi(A\tau)\mathbf{b}_L$ is computed using Krylov projection, as is normally done in EPI methods.

4. $\varphi(A\tau)\mathbf{b}_H$ is computed using a KSS method, as described in Section 3.

5. The results of the preceding two steps are added to obtain an approximation of $\varphi(A\tau)\mathbf{b}$.

We now provide details on how step 4 can be performed efficiently, by minimizing the number of FFTs. To simplify the exposition, we consider the 1-D case, with periodic boundary conditions. For each wave number ω, we obtain the frequency-dependent nodes $\{\lambda_{1,\omega}, \lambda_{2,\omega}, \ldots, \lambda_{K,\omega}\}$, as described in Section 3, by approximating the entries of the tridiagonal matrix $\mathsf{T}_K(\omega)$ produced by Lanczos iteration with initial vector $\hat{\mathbf{e}}_\omega$. As demonstrated in Section 3, this is readily accomplished using the coefficients of the spatial differential operator. Next, we use Krylov projection with initial vector $\mathbf{b}_H$, as in Section 4, and compute the eigenvalues of the resulting tridiagonal (or Hessenberg, if A is nonsymmetric) matrix. These are the frequency-independent nodes $\{\lambda_1, \lambda_2, \ldots, \lambda_K\}$.

The Fourier coefficient of $\varphi(A\tau)\mathbf{b}_H$ corresponding to the wave number ω is obtained by computing the same Fourier coefficient of $p_{2K-1,\omega}(A\tau)\mathbf{b}_H$, where $p_{2K-1,\omega}$ is the polynomial interpolant of $\varphi(\lambda)$ with interpolation points $\{\lambda_i, \lambda_{i,\omega}\}_{i=1}^K$. Expressing this interpolant in Newton form, we have

$$p_{2K-1,\omega}(\lambda) = \sum_{j=1}^{K} \varphi[\lambda_1,\ldots,\lambda_j] \prod_{i=1}^{j-1}(\lambda-\lambda_i) + \sum_{j=1}^{K} \varphi[\lambda_1,\ldots,\lambda_K,\lambda_{1,\omega},\ldots,\lambda_{j,\omega}] \left(\prod_{i=1}^{j-1}(\lambda-\lambda_{i,\omega}) \right) \prod_{k=1}^{K}(\lambda-\lambda_k). \tag{26}$$

Arranging the interpolation points in the order indicated above allows us to reduce the number of FFTs needed. Using the relation from Lanczos iteration,

$$AX_K = X_K T_K + \mathbf{r}_K \mathbf{e}_K^T, \tag{27}$$

we define

$$\begin{aligned} \mathbf{v} &= p_{K-1}(A)\mathbf{b}_H = \{\varphi[\lambda_1] + \varphi[\lambda_1,\lambda_2](A-\lambda_1 I) + \cdots \\ &\quad + \varphi[\lambda_1,\lambda_2,\ldots,\lambda_K](A-\lambda_1 I)\cdots(A-\lambda_{K-1} I)\}\mathbf{b}_H \\ &= \| \mathbf{b}_H \|_2 \, X_K p_{K-1}(A)\mathbf{e}_1, \\ \mathbf{w} &= q_K(A)\mathbf{b}_H = (A-\lambda_1 I)\cdots(A-\lambda_K I)\mathbf{b}_H \\ &= \beta_1\beta_2\ldots\beta_{K-1}\mathbf{r}_K \\ &= \beta_1\ldots\beta_K X_{K+1}\mathbf{e}_{K+1}, \end{aligned}$$

and

$$\begin{aligned} \tilde{p}_{K-1,\omega}(\lambda) &= \varphi[\lambda_1,\ldots,\lambda_K,\lambda_{1,\omega}] + \varphi[\lambda_1,\ldots,\lambda_{2,\omega}](\lambda-\lambda_{1,\omega}) + \cdots \\ &\quad + \varphi[\lambda_1,\ldots,\lambda_{K,\omega}](\lambda-\lambda_{1,\omega})\cdots(\lambda-\lambda_{K-1,\omega}) \\ &= C_{K-1}^{\omega}\lambda^{K-1} + \cdots + C_1^{\omega}\lambda + C_0^{\omega}, \end{aligned}$$

where C_j^{ω}, for $j = 0, 1, \ldots, K-1$, are the coefficients of $\tilde{p}_{K-1,\omega}$ in power form, which can easily be computed by repeated application of nested multiplication to the last K terms of the Newton form of $p_{2K-1,\omega}$.

Finally, using F to denote the discrete Fourier transform, we have

$$\varphi(A\tau)\mathbf{b}_H \approx p_{2K-1,\omega}(A\tau)\mathbf{b}_H = \mathbf{v} + \tilde{p}_{K-1,\omega}(A)\mathbf{w} = v + \mathsf{F}^{-1}\sum_{j=0}^{K-1}[C_j^{\omega}]\mathcal{F}A^j\mathbf{w}, \tag{28}$$

and it can easily be seen that the solution at each time step requires K FFTs and one inverse FFT. It is worth noting that once the Krylov subspace vectors $A^j\,\mathbf{w}$, $j = 0, 1, \ldots, K-1$ are computed, the K FFTs can be performed in parallel.

http://dx.doi.org/10.5772/62247

6. Numerical results

In this section, we compare several versions of EPI methods, as applied to two test problems; additional test problems are featured in [6, 7]. The versions differ in the way in which they compute matrix function-vector products of the form $\varphi(A\tau)\mathbf{b}$:

- standard Krylov projection, as in (25), hereafter referred to as "Krylov-EPI", either with or without denoising as in Section 4;
- using the KSS approach, as described in Section 5, hereafter referred to as "KSS-EPI";
- Newton interpolation using Leja points [23], hereafter referred to as "LEJA"; and
- adaptive Krylov projection [24], hereafter referred to as "AKP".

All of these approaches are used in the context of two EPI methods. The first is a third-order, two-stage EPI method [1]

$$\begin{aligned} Y_1 &= \mathbf{y}_n + \frac{1}{3}ha_{11}\varphi_1\left(\frac{1}{3}hA\right)F(\mathbf{y}_n), \\ \mathbf{y}_{n+1} &= \mathbf{y}_n + h\varphi_1(hA)F(\mathbf{y}_n) + 3hb_1\varphi_2(hA)[F(Y_1) - F(\mathbf{y}_n) - A(Y_1 - \mathbf{y}_n)], \end{aligned} \tag{29}$$

where $a_{11} = 9/4$ and $b_1 = 32/81$, and

$$R(Y_1) = F(Y_1) - F(\mathbf{y}_n) - A(Y_1 - \mathbf{y}_n).$$

For this method,

$$\varphi_1(\lambda) = \frac{e^\lambda - 1}{\lambda}, \quad \varphi_2(\lambda) = \frac{e^\lambda - \lambda - 1}{\lambda^2}, \quad \varphi_3(\lambda) = \frac{e^\lambda(6-\lambda) - (6 + 5\lambda + 2\lambda^2)}{\lambda^3}.$$

The second is a fifth-order, three-stage EPI method [25]

$$\begin{aligned} Y_1 &= \mathbf{y}_n + ha_{11}\psi_1(g_{11}hA)F(\mathbf{y}_n), \\ Y_2 &= \mathbf{y}_n + ha_{21}\psi_1(g_{21}hA)F(\mathbf{y}_n) + ha_{22}\psi_2(g_{22}hA)R(Y_1), \\ \mathbf{y}_{n+1} &= \mathbf{y}_n + hb_1\psi_1(g_{31}hA)F(\mathbf{y}_n) + hb_2\psi_2(g_{32}hA)R(Y_1) + \\ &\quad hb_3\psi_3(g_{33}hA)[-2R(Y_1) + R(Y_2)], \end{aligned} \tag{30}$$

where

$$\psi_i(z) = \sum_{j=1}^{j} p_{ij}\varphi_j(z), \quad i = 1,2,3,$$

and the coefficients g_{ij}, a_{ij}, b_j, and p_{ij} are obtained from the description of the EPIRK5s3 method in [25].

For all methods used to compute matrix function-vector products, efficiency will be measured in two ways: (1) total execution time required to integrate over the entire time interval, and (2) the average number of matrix-vector products (plus the average number of FFTs, in the case of KSS-EPI) performed for each evaluation of a matrix function-vector product of the form $\varphi(A\tau)\mathbf{b}$. The phrase "number of iterations" is used throughout this section to refer to this quantity. Throughout this section, for the purpose of discussing performance as a function of spatial resolution, N refers to the number of grid points per dimension.

6.1. Diffusive problem

For our first test problem, we choose a diffusion-dominated PDE: the 2-D Allen-Cahn equation,

$$u_t = \alpha\nabla^2 u + u - u^3, \quad x, y \in [0,1], \quad t \in [0,0.2] \tag{31}$$

with α = 0.1. We impose homogeneous Neumann boundary conditions, and the initial condition

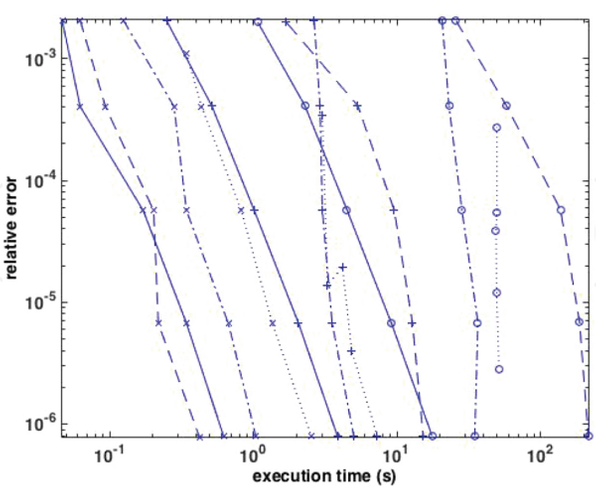

Figure 4. Relative error plotted against execution time for solving the Allen-Cahn equation (31) using the third-order EPI method (29). Matrix function-vector products are computed using KSS-EPI with denoising (solid curves), Krylov-EPI (dashed curves), AKP (dashed-dotted curves), and LEJA (dotted curves), on grids with N = 50 ('+' markers), 150 ('x' markers) and 300 ('o' markers) points per dimension. Time steps used are $\Delta t = (0.2)2^{-p}$, for p = 0, 1, 2, 3, 4.

http://dx.doi.org/10.5772/62247

$$u_0(x,y) = 0.4 + 0.1\cos(2\pi x)\cos(2\pi y).$$

The Laplacian is discretized using a centered finite difference. For KSS-EPI, the low-frequency portion $\mathbf{b}_L$ consists of all components with wave numbers $\omega_i \leq 7$, $i = 1, 2$. That is, for this problem, the value of N_σ, as defined in the previous section, is 7. The low value of this threshold is due to the smoothness of the initial data.

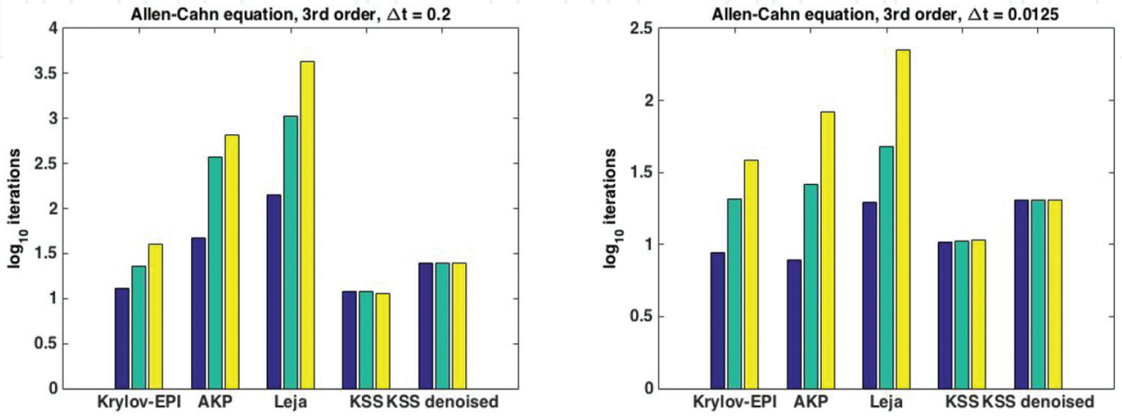

Figure 5. Average number of matrix-vector products, shown on a logarithmic scale, per matrix function-vector product evaluation for each method when solving the Allen-Cahn equation (31) using the third-order EPI method (29). For KSS and KSS denoised, FFTs are also included. For each method, bars correspond to grid sizes of $N = 50, 150, 300$ points per dimension, from left to right. Left plot: $\Delta t = 0.2$. Right plot: $\Delta t = 0.0125$.

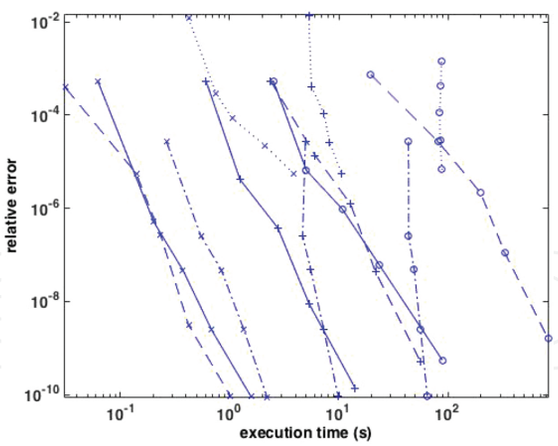

Figure 6. Relative error plotted against execution time for solving the Allen-Cahn equation (31) using the fifth-order EPI method (30). Matrix function-vector products are computed using KSS-EPI with denoising (solid curves), Krylov-EPI (dashed curves), AKP (dashed-dotted curves), and LEJA (dotted curves), on grids with $N = 50$ ('+' markers), 150 ('x' markers), and 300 ('o' markers) points per dimension. Time steps used are $\Delta t = (0.2)2^{-p}$, for $p = 0, 1, 2, 3, 4$.

The results are shown in **Figures 4** and **5** for the third-order EPI method (29), and in **Figures 6** and **7** for the firth-order EPI method (30). It can be seen from **Figures 4** and **6** that for both EPI methods, the accuracy of all four approaches to computing matrix function-vector

products are comparable. However, the efficiency and scalability of these four approaches are very different. In particular, these figures show that for KSS-EPI methods, the growth in execution time as a function of the number of grid points per dimension is much less. For example, note that for the coarsest grid, with $N = 50$, the speed of KSS-EPI is similar to that of Krylov-EPI, but for $N = 150$ and $N = 300$, KSS-EPI is much faster. Furthermore, as N increases, this gap in performance grows. This is due to the fact that while both methods use Krylov projection, KSS-EPI applies it to an initial vector with only low-frequency components, thus significantly reducing the number of iterations needed for convergence.

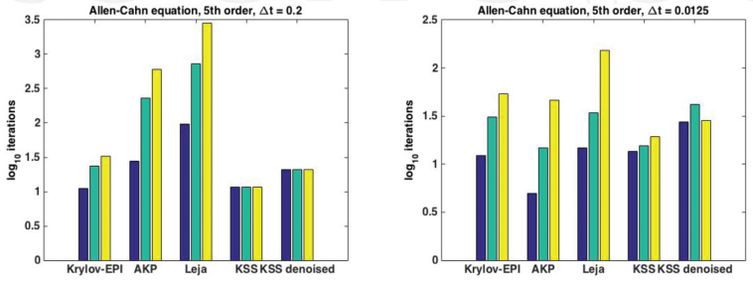

Figure 7. Average number of matrix-vector products, shown on a logarithmic scale, per matrix function-vector product evaluation for each method when solving the Allen-Cahn equation (31) using the fifth-order EPI method (29). For KSS and KSS denoised, FFTs are also included. For each method, bars correspond to grid sizes of $N = 50$, 150, 300 points per dimension, from left to right. Left plot: $\Delta t = 0.2$. Right plot: $\Delta t = 0.0125$.

The difference in scalability is more clearly illustrated in **Figures 5** and **7**. It can be seen that for KSS-EPI, the number of overall iterations (matrix-vector multiplications + FFTs) shows almost no sensitivity to the grid size, compared to Krylov-EPI, AKP and Leja interpolation, all of which exhibit substantial growth as the number of grid points increases. It can also be seen that denoising results in a slightly higher number of overall iterations, so it is not beneficial for this problem. This is not surprising, as this is a diffusive problem that has a smooth solution and is therefore less susceptible to high-frequency oscillations during Lanczos iteration. Although AKP is slightly more accurate than KSS-EPI for fifth order, this is more than offset by the superior efficiency of KSS-EPI with denoising, particularly at larger grid sizes and larger time steps.

6.2. Advective problem

For our second test problem, we choose an advection-dominated PDE: a 1-D Burgers' equation

$$u_t + uu_x = \nu u_{xx}, \quad x \in [0,1], \quad t \in [0,1] \tag{32}$$

with $\nu = 0.03$. We impose homogeneous Dirichlet boundary conditions, and the initial condition

$$u_0(x) = \sin^3(3\pi x)(1-x)^{3/2}.$$

http://dx.doi.org/10.5772/62247

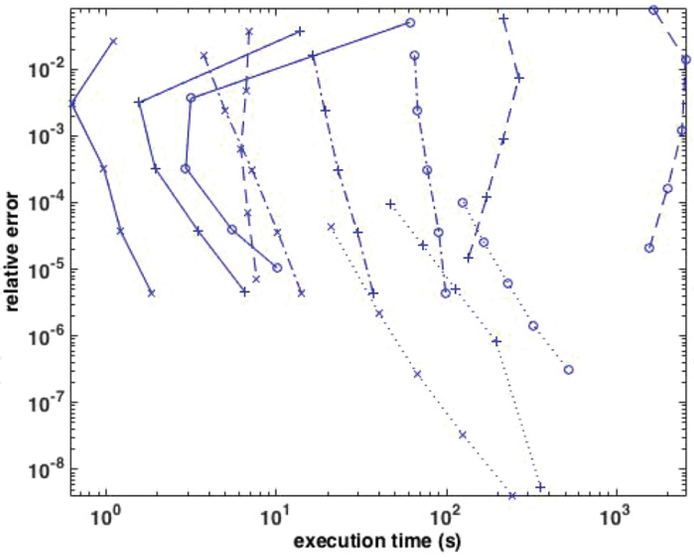

Figure 8. Relative error plotted against execution time for solving Burgers' equation (32) using the third-order EPI method (29). Matrix function-vector products are computed using KSS-EPI with denoising (solid curves), Krylov-EPI (dashed curves), AKP (dashed-dotted curves), and LEJA (dotted curves), on grids with $N = 500$ ('+' markers), 1500 ('x' markers), and 3000 ('o' markers) points. Time steps used are $\Delta t = (0.01)2^{-p}$, for $p = 0, 1, 2, 3, 4$.

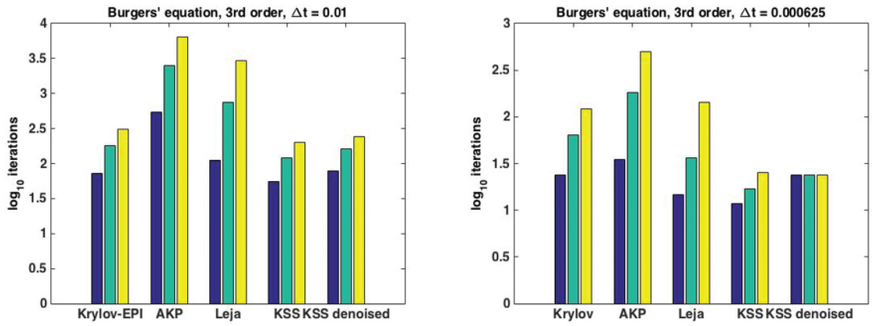

Figure 9. Average number of matrix-vector products, shown on a logarithmic scale, per matrix function-vector product evaluation for each method when solving Burgers' equation (32) using the third-order EPI method (29). For KSS and KSS denoised, FFTs are also included. For each method, bars correspond to grid sizes of $N = 500$, 1500, 3000 points, from left to right. Left plot: $\Delta t = 0.01$. Right plot: $\Delta t = 0.000625$.

For KSS-EPI, the low-frequency portion $\mathbf{b}_L$ consists of all components with wave numbers $\omega \leq N_c = 40$. The higher threshold for this problem, compared to the Allen-Cahn equation (31), is due to the fact that the initial data is less smooth.

The results are shown in **Figures 8** and **9** for the third-order EPI method (29), and in **Figures 10** and **11** for the fifth-order EPI method (30). From **Figures 8** and **10**, it can be seen that Krylov-EPI and KSS-EPI have comparable accuracy, but even for the coarsest grid with $N = 500$ points, KSS-EPI is much faster, and as with the Allen-Cahn equation, this gap in performance only grows with N. As shown in **Figures 9** and **11**, this is again due to the increasing number of Krylov projection steps needed for Krylov-EPI. While Leja interpolation is more accurate than KSS-EPI for both the third- and fifth-order EPI methods, and AKP is also more accurate

in the fifth-order case, both of these methods are still significantly slower than KSS-EPI, and a similar scalability gap can also be observed.

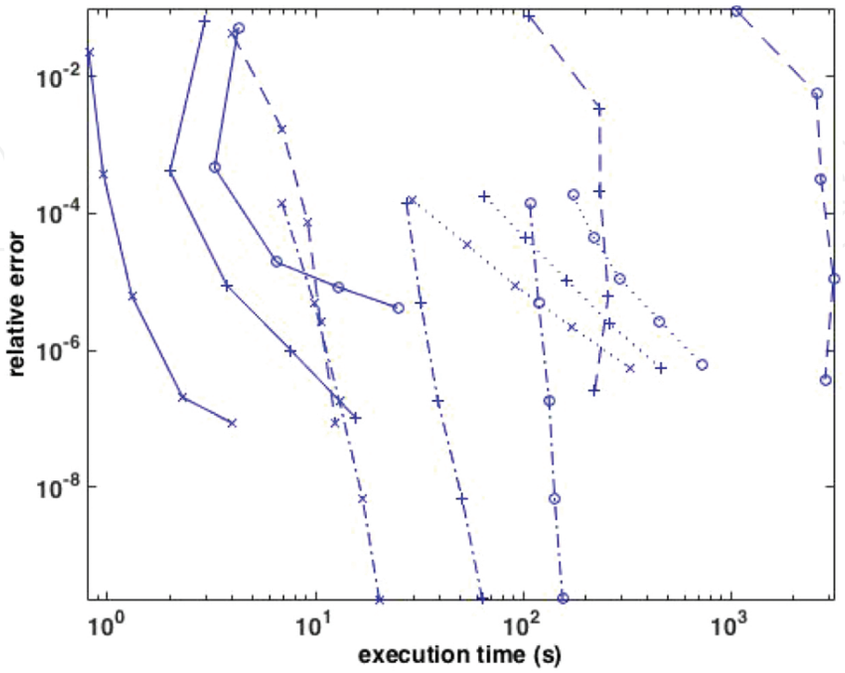

Figure 10. Relative error plotted against execution time for solving Burgers' equation (32) using the fifth-order EPI method (30). Matrix function-vector products are computed using KSS-EPI with denoising (solid curves), Krylov-EPI (dashed curves), AKP (dashed-dotted curves), and LEJA (dotted curves), on grids with $N = 500$ ('+' markers), 1500 ('x' markers), and 3000 ('o' markers) points. Time steps used are $\Delta t = (0.01)2^{-p}$, for $p = 0, 1, 2, 3, 4$.

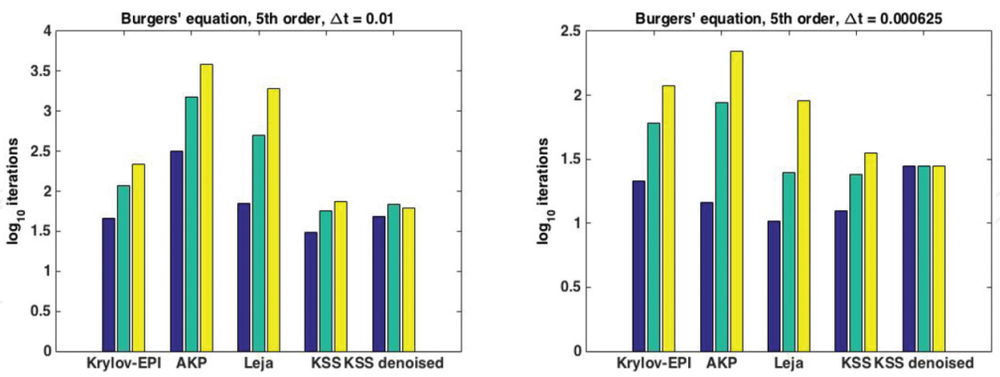

Figure 11. Average number of matrix-vector products, shown on a logarithmic scale, per matrix function-vector product evaluation for each method when solving Burgers' equation (32) using the fifth-order EPI method (30). For KSS and KSS denoised, FFTs are also included. For each method, bars correspond to grid sizes of $N = 500$, 1500, 3000 points, from left to right. Left plot: $\Delta t = 0.01$. Right plot: $\Delta t = 0.000625$.

The difference in scalability among the four approaches to computing matrix function-vector products is more apparent in **Figures 9** and **11**. Denoising applied to KSS-EPI is advantageous for this problem, unlike with the Allen-Cahn equation. As can be seen in these figures, for KSS-

EPI without denoising, the number of iterations is increasing with the number of grid points, though not nearly as rapidly as with the other methods. However, with denoising included, the insensitivity of the number of iterations to the grid size is restored.

6.3. Discussion of efficiency

The major components of the computational cost of KSS-EPI stem from Krylov projection that is applied to low-frequency parts, and FFTs that are applied to both the low- and high-frequency parts. Specifically, suppose the EPI method is of order p, and q Krylov projection steps are needed for convergence of the low-frequency part. Then, the task of evaluating $\varphi(A\tau)b$ requires $q + p$ matrix-vector multiplications, $(p + 3)/2$ FFTs and 2 inverse FFTs if denoising is not used, and $q + p$ matrix-vector multiplications, $q + (p + 3)/2$ FFTs and $q + 2$ inverse FFTs if denoising is used. Denoising, therefore, is only worthwhile if the value of q can be substantially reduced, as in the case of Burgers' equation.

7. Conclusions and future work

We have demonstrated that when solving stiff systems of nonlinear ODEs derived from PDEs, the growth in the computational cost that results from an increase in the number of grid points can be significantly reduced by performing a relatively low and grid-insensitive number of Krylov projection steps and FFTs on low- and high-frequency portions of the solution separately, instead of a number of Krylov projection steps on the entire solution that grows substantially with the number of grid points. The component-wise approach employed by KSS methods, in which each Fourier coefficient of a matrix function-vector product $\varphi(\tau A)\mathbf{b}$ is computed using a frequency-dependent interpolant of φ, allows the Krylov subspace dimension to be bounded independently of the grid size and instead determined by the desired temporal order of accuracy (for the high-frequency part) and by the number of Krylov projection steps required on a coarse grid (for the low-frequency part).

Future work on KSS-EPI methods will focus on the computation of low-frequency Fourier components of the solution, as this step accounts for the most significant part of the running time. This work requires an efficient approach to automatically selecting the threshold N_c that is used to distinguish low-frequency from high-frequency components. Such adaptivity can be based on the smoothness of the solution and the performance of Krylov projection during previous time steps. In addition, computing low-frequency components with methods other than Krylov projection, including Leja interpolation and adaptive Krylov projection, will be investigated. Such combination requires examination of error estimation and stopping criteria for these methods, to determine whether their convergence can be accelerated if it is known that the initial vector represents a bandlimited function.

It is important to note that the decomposition of the block Gaussian quadrature nodes into frequency-dependent and frequency-independent nodes is not limited to cases in which Fourier decomposition is used. In [26], the same idea is used for PDEs defined on a disk, in which a Legendre polynomial expansion is used for the radial part of the solution. The

decoupling of the eigenvalue problem for T_K first observed in [11], and further exploited in [6, 7], occurs due to the rapid decay of the coefficients in whatever series expansion is used to represent the solution. Future work will include taking advantage of this behavior in order to generalize KSS methods to PDEs defined on non-rectangular domains.

Author details

James V. Lambers[1*], Alexandru Cibotarica[2] and Elisabeth M. Palchak[3]

*Address all correspondence to: james.lambers@usm.edu

1 Department of Mathematics, The University of Southern Mississippi, Hattiesburg, MS, USA

2 Ivy Tech Community College, Indianapolis, IN, USA

3 Pearl River Community College, Hattiesburg, MS, USA

References

[1] Tokman, M.: Efficient integration of large stiff systems of ODEs with exponential propagation iterative (EPI) methods. *J. Comp. Phys.* 213 (2006) 748–776.

[2] Loffeld, J., Tokman, M.: Comparative performance of exponential, implicit and explicit integrators for stiff systems of ODEs. Submitted.

[3] Lambers, J. V.: Enhancement of Krylov subspace spectral methods by block Lanczos iteration. *Electron. T. Numer. Ana.* 31 (2008) 86–109.

[4] Lambers, J. V.: An explicit, stable, high-order spectral method for the wave equation based on block Gaussian quadrature. *IAENG Journal of Applied Mathematics* 38 (2008) 333–348.

[5] Guidotti, P., Kim, Y., Lambers, J. V.: Image restoration with a new class of forward-backward diffusion equations of Perona-Malik type. *SIAM Journal on Imaging Sciences* 6(3) (2013) 1416–1444.

[6] Cibotarica, A.: Solution of Nonlinear Time-Dependent PDE Through Componentwise Approximation of Matrix Functions. Ph.D. Dissertation, The University of Southern Mississippi, 2015.

[7] Cibotarica, A., Lambers, J. V., Palchak, E. M.: Solution of nonlinear time-dependent PDE through componentwise approximation of matrix functions. Submitted.

[8] Hochbruck, M., Lubich, C.: A Gautschi-type method for oscillatory second-order differential equations. *Numer. Math.* 83 (1999) 403–426.

[9] Hochbruck, M., Lubich, C.: On Krylov subspace approximations to the matrix exponential operator. *SIAM J. Numer. Anal.* 34 (1996) 1911–1925.

[10] Hochbruck, M., Lubich, C., Selhofer, H.: Exponential integrators for large systems of differential equations. *SIAM J. Sci. Comput.* 19 (1998) 1552–1574.

[11] Palchak, E. M., Cibotarica, A., Lambers, J. V.: Solution of time-dependent PDE through rapid estimation of block Gaussian quadrature nodes. *Lin. Alg. Appl.* 468 (2015) 233–259.

[12] Golub, G. H., Meurant, G.: Matrices, moments and quadrature. *Proceedings of the 15th Dundee Conference*, June-July 1993, Griffiths, D. F., Watson, G. A. (eds.), Longman Scientific & Technical (1994)

[13] Lambers, J. V.: Explicit high-order time-stepping based on componentwise application of asymptotic block Lanczos iteration, *Numerical Linear Algebra with Applications* 19(6) (2012) 970–991.

[14] Atkinson, K.: *An Introduction to Numerical Analysis, 2nd Ed.* Wiley (1989)

[15] Golub, G. H., Underwood, R.: The block Lanczos method for computing eigenvalues. *Mathematical Software III*, J. Rice Ed., (1977) 361–377.

[16] Lambers, J. V. Krylov subspace spectral methods for the time-dependent Schrödinger equation with non-smooth potentials. *Numerical Algorithms* 51 (2009) 239–280.

[17] Lambers, J. V.: Krylov subspace methods for variable-coefficient initial-boundary value problems. Ph.D. Thesis, Stanford University SC/CM program, 2003.

[18] Lambers, J. V.: A spectral time-domain method for computational electrodynamics. *Advances in Applied Mathematics and Mechanics* 1(6) (2009) 781–798.

[19] Saylor, P. E., Smolarski, D. C.: "Why Gaussian quadrature in the complex plane?" *Numerical Algorithms* 26 (2001) 251–280.

[20] Golub, G. H., van Loan, C. F.: *Matrix Computations*, 3rd Ed., Johns Hopkins University Press (1996).

[21] Saad, Y.: *Numerical Methods for Large Eigenvalue Problems*. Halsted Press, New York (1992).

[22] Goodman, J., Hou, T., Tadmor, E.: On the stability of the unsmoothed Fourier method for hyperbolic equations. *Numer. Math.* 67(1) (1994) 93–129.

[23] Bergamaschi, L., Caliari, M., Martnez, A., Vianello, M.: Comparing Leja and Krylov approximations of large scale matrix exponentials. *Lecture Notes in Comput. Sci., 6th International Conference, Reading, UK, May 28–31, 2006, Proceedings, Part IV*, Alexandrov, V. N., van Albada, G. D., Sloot, P. M. A., Dongarra, J. (eds.), Springer (2006) 685–692.

[24] Rainwater, G., Tokman, M.: A new class of split exponential propagation iterative methods of Runge-Kutta type (sEPIRK) for semilinear systems of ODEs. *J. Comp. Phys.* 269 (2014) 40–60.

[25] Tokman, M.: A new class of exponential propagation iterative methods of Runge-Kutta type (EPIRK). *J. Comp. Phys.* 230 (2011) 8762–8778.

[26] Richardson, M., Lambers, J. V.: Krylov subspace spectral methods for polar and cylindrical geometries. In preparation.