Efficient and stable Arnoldi restarts for matrix functions based on quadrature

Frommer, Andreas and Güttel, Stefan and Schweitzer, Marcel

2013

MIMS EPrint: **2013.48**

Manchester Institute for Mathematical Sciences
School of Mathematics

The University of Manchester

Reports available from: http://eprints.maths.manchester.ac.uk/
And by contacting: The MIMS Secretary
School of Mathematics
The University of Manchester
Manchester, M13 9PL, UK

ISSN 1749-9097

EFFICIENT AND STABLE ARNOLDI RESTARTS FOR MATRIX FUNCTIONS BASED ON QUADRATURE*

ANDREAS FROMMER†, STEFAN GÜTTEL‡, AND MARCEL SCHWEITZER†

Abstract. When using the Arnoldi method for approximating $f(A)\boldsymbol{b}$, the action of a matrix function on a vector, the maximum number of iterations that can be performed is often limited by the storage requirements of the full Arnoldi basis. As a remedy, different restarting algorithms have been proposed in the literature, none of which was universally applicable, efficient, and stable at the same time. We utilize an integral representation for the error of the iterates in the Arnoldi method which then allows us to develop an efficient quadrature-based restarting algorithm suitable for a large class of functions, including the so-called Stieltjes functions and the exponential function. Our method is applicable for functions of Hermitian and non-Hermitian matrices, requires no a-priori spectral information, and runs with essentially constant computational work per restart cycle. We comment on the relation of this new restarting approach to other existing algorithms and illustrate its efficiency and numerical stability by various numerical experiments.

Key words. matrix function, Krylov subspace approximation, restarted Arnoldi/Lanczos method, deflated restarting, polynomial interpolation, Gaussian quadrature, Padé approximation

AMS subject classifications. 65F60, 65F50, 65F10, 65F30, 41A20

1. Introduction. The computation of $f(A)\boldsymbol{b}$, the action of a matrix function $f(A) \in \mathbb{C}^{N\times N}$ on a vector $\boldsymbol{b} \in \mathbb{C}^N$, is an important task in many areas of science and engineering. Examples include the matrix exponential function $f(z) = e^z$, which is at the heart of exponential integrators for the solution of differential equations [26, 27], the logarithm $f(z) = \log(z)$ used, e.g. in Markov model analysis [39] and identification problems for linear continuous-time multivariable systems [31], fractional powers $f(z) = z^\alpha$ in fractional differential equations [6], and the sign function $f(z) = \mathrm{sign}(z)$ which is often related to spectral projectors and also appears in lattice quantum chromodynamics [5, 41].

In many of these applications, the matrix A is sparse and large so that the explicit computation of the generally dense matrix $f(A)$ by direct methods as in [8, 24, 25] is infeasible. Instead, one seeks to directly approximate the vector $f(A)\boldsymbol{b}$ by iterative methods. By far the most important class of iterative methods for this purpose are *Krylov subspace methods*, see [11, 18, 26, 29, 35]. These methods extract their approximations to $f(A)\boldsymbol{b}$ from Krylov subspaces $\mathcal{K}_m(A, \boldsymbol{b}) = \mathrm{span}\{\boldsymbol{b}, A\boldsymbol{b}, \ldots, A^{m-1}\boldsymbol{b}\}$. Assume that we are given an *Arnoldi decomposition*

$$AV_m = V_m H_m + h_{m+1,m}\boldsymbol{v}_{m+1}\boldsymbol{e}_m^T, \tag{1.1}$$

where the columns of $V_m = [\boldsymbol{v}_1|\boldsymbol{v}_2|\cdots|\boldsymbol{v}_m] \in \mathbb{C}^{N\times m}$ are an orthonormal basis of $\mathcal{K}_m(A, \boldsymbol{b})$ obtained from m steps of the Arnoldi orthogonalization process, $H_m \in \mathbb{C}^{m\times m}$ is an upper Hessenberg matrix, and $\boldsymbol{e}_m \in \mathbb{R}^m$ corresponds to the m-th canonical unit vector (see, e.g., [36, Ch. 6]). Then a popular approach for approximating $f(A)\boldsymbol{b}$ is the *Arnoldi approximation*

$$\boldsymbol{f}_m = V_m f(H_m) V_m^H \boldsymbol{b} = \|\boldsymbol{b}\| V_m f(H_m)\boldsymbol{e}_1. \tag{1.2}$$

*This work was partially supported by Deutsche Forschungsgemeinschaft through Cooperative Research Centre SFB TRR55 "Hadron Physics through Lattice QCD".

†Department of Mathematics, Bergische Universität Wuppertal, 42097 Wuppertal, Germany, {frommer,schweitzer}@math.uni-wuppertal.de

‡School of Mathematics, The University of Manchester, M13 9PL Manchester, United Kingdom, stefan.guettel@manchester.ac.uk

One of the main computational problems associated with the Arnoldi approximation (1.2) is that the full Arnoldi basis V_m needs to be stored. This storage requirement may limit the number of iterations m that can be performed in practice, and thus the accuracy that can be achieved for large problems. A further limiting factor is the growing orthogonalization cost of computing V_m and the cost of evaluating $f(H_m)$ for larger values of m.

There are two popular strategies to overcome these problems. The first one, not further considered in this paper, is to use other subspaces with superior approximation properties, like extended Krylov subspaces [12,30], shift-and-invert Krylov subspaces [32,42], both special cases of general rational Krylov subspaces [4,20–22], with the aim to reach a targeted accuracy within significantly fewer iterations. However, rational Krylov methods typically involve linear system solves with (shifted versions of) the matrix A at each iteration. Thus, when solving linear systems with A is expensive, or in situations where A is not even explicitly available but only implicitly as a routine returning matrix-vector products, rational Krylov methods may be infeasible.

The other possible strategy for circumventing the problems mentioned above, which only requires matrix-vector products with A and will be the subject of this paper, is based on *restarting*, similar to what is often done for the solution of (non-Hermitian) linear systems of equations (the case $f(z) = z^{-1}$). This was already investigated several times [1, 2, 14, 15, 28], but none of the restarting approaches for general matrix functions was completely satisfactory until now. All of these variants solved the storage problem for the Arnoldi basis, but still had to deal with growing cost per restart cycle [14], were numerically unstable [28], or required an accurate rational approximation $r(z) \approx f(z)$ for all z in some spectral region of A [2]. Instead of relying on an error representation involving *divided differences* (see [14, 28]), we propose in this paper a novel algorithm based on the *integral representation* of the error. Our error representation is applicable to a large class of functions and allows for the derivation of a restarting algorithm similar to the one in [28], but without the numerical stability problems and without the restriction to Hermitian matrices.

The remainder of this paper is organized as follows. In section 2 we briefly review the different restarting approaches available in the literature so far. In section 3 we derive a new integral representation for the error of the m-th Arnoldi approximation. In section 4 we then investigate the use of numerical quadrature for evaluating this error representation. The quadrature rule of choice typically depends on the function f and we discuss a selection of quadrature rules specifically tailored to important functions. Numerical experiments illustrating the performance of our method, both for simple model problems and for problems from relevant applications, are presented in section 5. Concluding remarks are given in section 6.

2. The restarted Arnoldi method. We will start by recalling a useful characterization of the Arnoldi approximation (1.2). This result makes clear the relation between the Arnoldi approximation and polynomial interpolation and will be exploited repeatedly in this paper.

LEMMA 2.1 (Ericsson [16] and Saad [35]). *Let $\boldsymbol{f}_m$ be the m-th Arnoldi approximation to $f(A)\boldsymbol{b}$ defined in* (1.2)*. Then*

$$\boldsymbol{f}_m = \widetilde{p}_{m-1}(A)\boldsymbol{b}, \tag{2.1}$$

where $\widetilde{p}_{m-1} \in \mathcal{P}_{m-1}$ is the unique polynomial interpolating f at the eigenvalues of H_m in the Hermite sense (i.e., counting the multiplicity of each eigenvalue as a root in the minimal polynomial of H_m).

When computing the Arnoldi approximation (1.2), one has to deal with two main problems. The first one is that the whole Arnoldi basis V_m needs to be computed and stored for evaluating (1.2), which will be prohibitively expensive when m grows too large. However, even when A is Hermitian and V_m can be computed via the short-recurrence Lanczos method (cf. [36]), this recurrence does generally not translate into a short-recurrence relation for the Arnoldi approximations $\boldsymbol{f}_m$. This is in contrast to the special case of solving a linear system of equations, i.e., when $f(z) = z^{-1}$. A simple strategy to overcome the storage problem is to perform a so-called *two-pass Lanczos method* (see, e.g., [17]), where in a first sweep the tridiagonal matrix H_m is computed using the short recurrence for the Lanczos vectors $\boldsymbol{v}_1, \boldsymbol{v}_2, \dots, \boldsymbol{v}_m$, and then in a second sweep the approximation $\boldsymbol{f}_m$ is formed by generating the Lanczos vectors anew and combining them linearly with the coefficients from the vector $\|\boldsymbol{b}\| f(H_m)\boldsymbol{e}_1$. Although for Hermitian matrices this approach solves the storage issues, it essentially doubles the computational work and is therefore not a satisfactory solution. The other problem is that $f(H_m)\boldsymbol{e}_1$ has to be computed for forming $\boldsymbol{f}_m$, which itself can become prohibitively expensive when m is large.

The same two problems also arise in the iterative solution of non-Hermitian linear systems by Krylov subspace methods, such as FOM or GMRES, cf. [36, 37]. A well-known technique to overcome these problems is *restarting*: after m Arnoldi orthogonalization steps, the approximation $\boldsymbol{f}_m$ is formed, the basis V_m computed so far is discarded, and a second Arnoldi cycle is started to approximate the error $\boldsymbol{d}_m = \boldsymbol{x} - \boldsymbol{f}_m$, where $\boldsymbol{x}$ denotes the sought solution of the linear system $A\boldsymbol{x} = \boldsymbol{b}$. Such a restarting procedure is possible because the error $\boldsymbol{d}_m$ solves the residual equation

$$A\boldsymbol{d}_m = \boldsymbol{r}_m \tag{2.2}$$

and the residual $\boldsymbol{r}_m = \boldsymbol{b} - A\boldsymbol{f}_m$ is easily computable.

When trying to develop a restarting technique for general matrix functions f, one of the main problems is the lack of a direct analogue of (2.2). However, it is possible to represent the *error* of the restarted Arnoldi approximations based on divided differences (see, e.g., [10]), as the following result from [14] shows. This result is stated for *Arnoldi-like decompositions*, which are decompositions of the form (1.1) but with the requirement dropped that the columns of V_m be orthonormal.

THEOREM 2.2 (Eiermann & Ernst [14]). *Given $A \in \mathbb{C}^{N\times N}$, $\boldsymbol{b} \in \mathbb{C}^N$, let $AV_m = V_m H_m + h_{m+1,m}\boldsymbol{v}_{m+1}\boldsymbol{e}_m^T$ be an Arnoldi-like decomposition and $w_m(z) = (z-\theta_1)\cdots(z-\theta_m)$ be the nodal polynomial associated with the Ritz values $\theta_1, \dots, \theta_m$, i.e., the eigenvalues of H_m. Then the error of $\boldsymbol{f}_m$ defined in* (1.2) *is given as*

$$f(A)\boldsymbol{b} - \boldsymbol{f}_m = \|\boldsymbol{b}\|\gamma_m[D_{w_m}f](A)\boldsymbol{v}_{m+1} =: e_m(A)\boldsymbol{v}_{m+1}, \tag{2.3}$$

where $[D_{w_m}f]$ denotes the m-th divided difference of f with respect to the interpolation nodes $\theta_1, \dots, \theta_m$, and $\gamma_m = \prod_{i=1}^m h_{i+1,i}$.

This representation of the error after m steps of the Arnoldi method (or an Arnoldi-like method) allows to perform restarting similar to the linear system case. Again, the error is represented as a matrix function of A multiplied onto a vector. While in the linear system case only the right-hand side of (2.2) changes at each restart, for a general function f also the *error function* changes from f to a multiple of $[D_{w_m}f]$. Assuming exact arithmetic, a restarting procedure as summarized in Algorithm 1 can be employed to approximate $f(A)\boldsymbol{b}$.

While Algorithm 1 together with the error representation (2.3) allow for restarting of the Arnoldi method in theory, this combination is not feasible for practical

Algorithm 1: Restarted Arnoldi method for $f(A)\boldsymbol{b}$ from [14] (generic version).

Given: A, $\boldsymbol{b}$, f, m
Compute the Arnoldi decomposition $AV_m^{(1)} = V_m^{(1)} H_m^{(1)} + h_{m+1,m}^{(1)} \boldsymbol{v}_{m+1}^{(1)} \boldsymbol{e}_m^T$
with respect to A and $\boldsymbol{b}$.
Set $\boldsymbol{f}_m^{(1)} := \|\boldsymbol{b}\| V_m^{(1)} f(H_m^{(1)}) \boldsymbol{e}_1$.
for $k = 2, 3, \ldots$ until convergence **do**
  Determine the error function $e_m^{(k-1)}(z)$.
  Compute the Arnoldi decomposition $AV_m^{(k)} = V_m^{(k)} H_m^{(k)} + h_{m+1,m}^{(k)} \boldsymbol{v}_{m+1}^{(k)} \boldsymbol{e}_m^T$
  with respect to A and $\boldsymbol{v}_{m+1}^{(k-1)}$.
  Set $\boldsymbol{f}_m^{(k)} := \boldsymbol{f}_m^{(k-1)} + \|\boldsymbol{b}\| V_m^{(k)} e_m^{(k-1)}(H_m^{(k)}) \boldsymbol{e}_1$.

computations due to severe stability problems. These problems can be explained by the well-known fact that the numerical evaluation of high-order divided differences is prone to instabilities, especially when interpolation nodes are close to each other, thereby causing subtractive cancelations and very small denominators in the divided difference table. For Hermitian A it is known that the Ritz values of all restart cycles will asymptotically appear as a two-cyclic sequence [2], so that the interpolation nodes will form $2m$ clusters and the evaluation of the error function using (2.3) will necessarily become unstable.

A different error representation for Hermitian A was investigated in [28].

THEOREM 2.3 (Ilic et al. [28]). *Let A be Hermitian and the assumptions of Theorem 2.2 be fulfilled. Let W_m be the unitary matrix whose columns are the eigenvectors of H_m, and define $\alpha_i = \boldsymbol{e}_m^T W_m \boldsymbol{e}_i$ and $\gamma_i = \boldsymbol{e}_1^T W_m \boldsymbol{e}_i$ $(i = 1, \ldots, m)$. Then*

$$f(A)\boldsymbol{b} - \boldsymbol{f}_m = \|\boldsymbol{b}\| h_{m+1,m} g(A) \boldsymbol{v}_{m+1} \tag{2.4}$$

with

$$g(z) = \sum\nolimits_{i=1}^{m} \alpha_i \gamma_i D_{w_i}(z) \quad \text{where} \quad w_i(z) = (z - \theta_i). \tag{2.5}$$

The error function representation in Theorem 2.3 involves only first-order divided differences, so that one could expect it to be less prone to numerical instabilities than the representation from Theorem 2.2. However, this is only moderately so as was observed in [28], rendering this representation still unstable and therefore not usable in practice, in particular if accuracy and reliability requirements are high.

The original restarting method of [14] is stable. Instead of relying on an error function for restarting, the same function f is used throughout all restart cycles. This is possible because the Arnoldi-like approximations from consecutive restart cycles satisfy the update formula

$$\boldsymbol{f}_m^{(k)} = \boldsymbol{f}_m^{(k-1)} + \|\boldsymbol{b}\| V_m^{(k)} [f(H_{km}) \boldsymbol{e}_1]_{(k-1)m+1:km}, \quad k \geq 2, \tag{2.6}$$

where the subscript of the last term means that only the m trailing entries of the resulting vector are used, and where all the Hessenberg matrices from the previous restart cycles are recursively accumulated in a block-Hessenberg matrix

$$H_{km} = \begin{bmatrix} H_{(k-1)m} & O \\ h_{m+1,m}^{(k-1)} \boldsymbol{e}_1 \boldsymbol{e}_{(k-1)m}^T & H_m^{(k)} \end{bmatrix}. \tag{2.7}$$

Note that only the original function f is needed for updating the approximations via (2.6), so that the stability problems with divided differences are completely avoided. This restarting approach also solves the storage problem for the Arnoldi basis, as only the last Arnoldi basis $V_m^{(k)}$ is required for evaluating (2.6). The price one has to pay with this method, however, is that it is still necessary to evaluate f on a Hessenberg matrix H_{km} of increasing size km. This is a computational cost that grows (often cubically in km) from one restart cycle to the next, which may again result in an unacceptably high cost if many restart cycles are required for converging below a targeted accuracy level. This is especially problematic as the convergence of a restarted Arnoldi method is generally slower than that of the standard (unrestarted) Arnoldi method, so that the dimension of the matrix H_{km} from (2.7) will be larger than the dimension of the matrix H_m from the standard Arnoldi approximation (1.2) achieving a comparable accuracy.

Algorithm 2: Restarted Arnoldi approximation for $f(A)\boldsymbol{b}$ from [2].

Given: A, $\boldsymbol{b}$, m, rational approximation $r \approx f$ of the form (2.8)
Set $\boldsymbol{f}_m^{(0)} = \boldsymbol{0}$ and $\boldsymbol{v}_{m+1}^{(0)} = \boldsymbol{b}$
for $k = 1, 2, \ldots$ until convergence **do**
  Compute the Arnoldi decomposition $AV_m^{(k)} = V_m^{(k)} H_m^{(k)} + h_{m+1,m}^{(k)} \boldsymbol{v}_{m+1}^{(k)} \boldsymbol{e}_m^T$ with respect to A and $\boldsymbol{v}_{m+1}^{(k-1)}$.
  if $k = 1$ **then**
    for $i = 1, \ldots, \ell$ **do**
      Solve $(t_i I - H_m^{(k)})\boldsymbol{r}_{i,1} = \boldsymbol{e}_1$
  else
    for $i = 1, \ldots, \ell$ **do**
      Solve $(t_i I - H_m^{(k)})\boldsymbol{r}_{i,k} = h_{m+1,m}^{(k-1)} (\boldsymbol{e}_m^T \boldsymbol{r}_{i,k-1}) \boldsymbol{e}_1$
  $\boldsymbol{h}_m^{(k)} = \sum_{i=1}^{\ell} \alpha_i \boldsymbol{r}_{i,k}$
  Set $\boldsymbol{f}_m^{(k)} := \boldsymbol{f}_m^{(k-1)} + \|\boldsymbol{b}\| V_m^{(k)} \boldsymbol{h}_m^{(k)}$.

It was shown in [2] that for a rational function in partial fraction form

$$r(z) = \sum\nolimits_{i=1}^{\ell} \frac{\alpha_i}{t_i - z}, \tag{2.8}$$

the evaluation of (2.6) with $f = r$ is possible with constant work per restart cycle, as the block lower triangular structure of H_{km} allows for the evaluation of $(t_i I - H_{km})^{-1} \boldsymbol{e}_1$ via a sequential solution of k shifted linear systems

$$(t_i I - H_m^{(1)})\boldsymbol{r}_{i,1} = \boldsymbol{e}_1, \tag{2.9}$$

$$(t_i I - H_m^{(j)})\boldsymbol{r}_{i,j} = h_{m+1,m}^{(j-1)} (\boldsymbol{e}_m^T \boldsymbol{r}_{i,j-1}) \boldsymbol{e}_1, \quad j = 2, \ldots, k, \tag{2.10}$$

of size m. Exploiting that only the last block of $r(H_{km})\boldsymbol{e}_1$ is required, only ℓ linear systems of size m need to be solved per restart cycle. This allows for efficient restarting for general functions f whenever a sufficiently accurate rational approximation r with $r(A)\boldsymbol{b} \approx f(A)\boldsymbol{b}$ is available. Note that this rational approximation needs to be known a-priori and stays fixed throughout all restart cycles. The resulting method is summarized in Algorithm 2.

3. Integral representation of the error function. The main problem causing instability in the implementations of Algorithm 1 considered in the literature so far is the numerical evaluation of divided differences. We now derive an alternative integral representation of the error function. As a first step, we give a formula for interpolating polynomials of functions representable as a Cauchy-type integral.

LEMMA 3.1. *Let $\Omega \subset \mathbb{C}$ be a region and let $f : \Omega \to \mathbb{C}$ be analytic with the integral representation*

$$f(z) = \int_\Gamma \frac{g(t)}{t-z}\,\mathrm{d}t, \quad z \in \Omega, \tag{3.1}$$

with a set $\Gamma \subset \mathbb{C} \setminus \Omega$ and a function $g : \Gamma \to \mathbb{C}$. The interpolating polynomial p_{m-1} of f with interpolation nodes $\{\theta_1, \ldots, \theta_m\} \subset \Omega$ is given as

$$p_{m-1}(z) = \int_\Gamma \left(1 - \frac{w_m(z)}{w_m(t)}\right) \frac{g(t)}{t-z}\,\mathrm{d}t, \tag{3.2}$$

where $w_m(z) = (z-\theta_1)\cdots(z-\theta_m)$.

Proof. Observe that $1 - w_m(z)/w_m(t)$ is a polynomial of degree m in z with a root at t. Therefore it contains a linear factor $t-z$, showing that $(1-w_m(z)/w_m(t))/(t-z)$ is a polynomial of degree $m-1$ in z, and so is the whole right-hand side of (3.2). By definition of w_m we have

$$\int_\Gamma \left(1 - \frac{w_m(\theta_i)}{w_m(t)}\right) \frac{g(t)}{t-\theta_i}\,\mathrm{d}t = \int_\Gamma \frac{g(t)}{t-\theta_i}\,\mathrm{d}t = f(\theta_i) \quad \text{for} \quad i = 1, \ldots, m, \tag{3.3}$$

showing that the interpolation conditions are satisfied. Interpolation conditions for derivatives of f can be checked in the same way after differentiating the right-hand side of (3.2) with respect to z. ∎

With Lemma 3.1 we are now prepared to give an integral representation for the error of the Arnoldi approximation to $f(A)\boldsymbol{b}$.

THEOREM 3.2. *Let f have an integral representation as in Lemma 3.1, and let $A \in \mathbb{C}^{N\times N}$ with $\mathrm{spec}(A) \subset \Omega$ and $\boldsymbol{b} \in \mathbb{C}^N$ be given. Denote by $\boldsymbol{f}_m$ the m-th Arnoldi approximation (1.2) to $f(A)\boldsymbol{b}$ with $\mathrm{spec}(H_m) = \{\theta_1, \ldots, \theta_m\} \subset \Omega$. Then*

$$f(A)\boldsymbol{b} - \boldsymbol{f}_m = \gamma_m \int_\Gamma \frac{g(t)}{w_m(t)} (tI - A)^{-1} \boldsymbol{v}_{m+1}\,\mathrm{d}t =: e_m(A)\boldsymbol{v}_{m+1}, \tag{3.4}$$

where $\gamma_m = \prod_{i=1}^m h_{i+1,i}$ and $w_m(z) = (z-\theta_1)\cdots(z-\theta_m)$.

Proof. Let p_{m-1} denote the interpolating polynomial of f with respect to the interpolation nodes $\theta_1, \ldots, \theta_m$. By subtracting p_{m-1} from f and using the representations (3.1) and (3.2) we have

$$f(z) - p_{m-1}(z) = \int_\Gamma \frac{w_m(z)}{w_m(t)} \frac{g(t)}{t-z}\,\mathrm{d}t. \tag{3.5}$$

Substituting A for z in (3.5), post-multiplying by $\boldsymbol{b}$, and noting that $p_{m-1}(A)\boldsymbol{b} = \boldsymbol{f}_m$ by Lemma 2.1 then leads to

$$f(A)\boldsymbol{b} - \boldsymbol{f}_m = \int_\Gamma \frac{g(t)}{w_m(t)} (tI - A)^{-1} w_m(A)\boldsymbol{b}\,\mathrm{d}t. \tag{3.6}$$

The assertion then follows from the fact that $w_m(A)\boldsymbol{b} = \gamma_m \boldsymbol{v}_{m+1}$, see [34, Cor. 1]. ∎

Theorem 3.2 shows that the error of the Arnoldi approximation $\boldsymbol{f}_m$ to $f(A)\boldsymbol{b}$ can again be interpreted as an error matrix function $e_m(A)$ applied to a vector. In contrast to (2.3), the error function is now represented via an integral instead of divided differences, and this new representation can be used in Algorithm 1. A similar integral representation also holds for the error in all subsequent restart cycles, since due to the general Cauchy-type integral used in Lemma 3.1 and Theorem 3.2, these results also apply when f is replaced by e_m.

COROLLARY 3.3. *Let the assumptions of Theorem 3.2 hold, and let $\boldsymbol{f}_m^{(k)}$ be the restarted Arnoldi approximation to $f(A)\boldsymbol{b}$ after k restart cycles of Algorithm 1. Then the error of $\boldsymbol{f}_m^{(k)}$ satisfies*

$$f(A)\boldsymbol{b} - \boldsymbol{f}_m^{(k)} = \gamma_m^{(1)} \cdots \gamma_m^{(k)} \int_\Gamma \frac{g(t)}{w_m^{(1)}(t) \cdots w_m^{(k)}(t)} (tI - A)^{-1} \boldsymbol{v}_{m+1}^{(k)} \, \mathrm{d}t =: e_m^{(k)}(A) \boldsymbol{v}_{m+1}^{(k)}. \tag{3.7}$$

In principle, the results of Theorem 3.2 and Corollary 3.3 can be applied to any function f analytic in a neighborhood of $\mathrm{spec}(A)$, because then by the Cauchy integral formula we have

$$f(A)\boldsymbol{b} = \frac{1}{2\pi i} \int_\Gamma f(t) \, (tI - A)^{-1} \boldsymbol{b} \, \mathrm{d}t, \tag{3.8}$$

where Γ is a closed contour (or a union of closed contours) winding around each eigenvalue of A exactly once. However, this representation is not always useful from a computational point of view, as it requires information about the spectral region of A, which in general is not available.

Therefore we will now focus on a class of functions where the path $\Gamma = (-\infty, 0]$ is fixed and does not depend on $\mathrm{spec}(A)$, the so-called *Stieltjes functions* (see, e.g., [23]). Important examples of Stieltjes functions include

$$f(z) = z^{-\alpha} = \frac{\sin((\alpha - 1)\pi)}{\pi} \int_{-\infty}^{0} \frac{t^{-\alpha}}{t - z} \, \mathrm{d}t \quad \text{for } \alpha \in (0, 1) \tag{3.9}$$

and

$$f(z) = \frac{\log(1 + z)}{z} = \int_{-\infty}^{-1} \frac{t^{-1}}{t - z} \, \mathrm{d}t. \tag{3.10}$$

For further examples of Stieltjes functions we refer to [23]. In addition to being independent of $\mathrm{spec}(A)$, the path Γ is a real interval, which often allows one to find elegant integral transformations leading to finite integration intervals. Such transformations will be considered in more detail in section 4.

Some interesting functions like $\widetilde{f}(z) = z^\alpha$ for $\alpha \in (0, 1)$, including the square root as the most important special case, or $\widetilde{f}(z) = \log(1 + z)$, do not belong to the class of Stieltjes functions but can be written as $\widetilde{f}(z) = zf(z)$, where f is a Stieltjes function. In this case, the results of Theorem 3.2 and Corollary 3.3 do not apply directly, at least not using the very favorable Stieltjes integral representation. One possibility to overcome this problem and still use the Stieltjes representation in a restarting algorithm is to first compute $\widetilde{\boldsymbol{b}} = A\boldsymbol{b}$ and then approximate $f(A)\widetilde{\boldsymbol{b}}$. While this is theoretically feasible, it should be avoided in computations because $\|\widetilde{\boldsymbol{b}}\|$ may be significantly larger than $\|\boldsymbol{b}\|$, resulting in larger absolute errors of the Arnoldi

approximations. It is therefore desirable to be able to work with $\widetilde{f}$ directly, and Theorem 3.2 can be modified easily to accommodate for such functions.

COROLLARY 3.4. *Let the assumptions of Theorem 3.2 hold and let* $\widetilde{f}(z) = zf(z)$. *Denote by* $\widetilde{\boldsymbol{f}}_m$ *the* m*-th Arnoldi approximation* (1.2) *to* $\widetilde{f}(A)\boldsymbol{b}$. *Then*

$$\widetilde{f}(A)\boldsymbol{b} - \widetilde{\boldsymbol{f}}_m = \gamma_m A \int_\Gamma \frac{g(t)}{w_m(t)} (tI - A)^{-1} \boldsymbol{v}_{m+1} \, \mathrm{d}t - h_{m+1,m} \left(\boldsymbol{e}_m^T f(H_m) \boldsymbol{e}_1 \right) \boldsymbol{v}_{m+1}. \quad (3.11)$$

Proof. By (1.2) we have

$$\widetilde{\boldsymbol{f}}_m = V_m H_m f(H_m) \boldsymbol{e}_1. \quad (3.12)$$

Inserting the Arnoldi decomposition (1.1) gives

$$\widetilde{\boldsymbol{f}}_m = A V_m f(H_m) \boldsymbol{e}_1 - h_{m+1,m} \left(\boldsymbol{e}_m^T f(H_m) \boldsymbol{e}_1 \right) \boldsymbol{v}_{m+1}. \quad (3.13)$$

By subtracting (3.13) from $\widetilde{f}(A)\boldsymbol{b}$ we arrive at

$$\widetilde{f}(A)\boldsymbol{b} - \boldsymbol{f}_m = A \left(f(A)\boldsymbol{b} - V_m f(H_m) \boldsymbol{e}_1 \right) - h_{m+1,m} \left(\boldsymbol{e}_m^T f(H_m) \boldsymbol{e}_1 \right) \boldsymbol{v}_{m+1}. \quad (3.14)$$

The assertion now follows by applying Theorem 3.2 to the first term. ∎

Corollary 3.4 can easily be generalized to functions of the form $\widetilde{f}(z) = z^\ell f(z)$ by repeated application of (1.1). We just stated the result for $zf(z)$ for the sake of notational simplicity and because it appears to be the most important case in practice. Ignoring for a moment the term

$$-h_{m+1,m} \left(\boldsymbol{e}_m^T f(H_m) \boldsymbol{e}_1 \right) \boldsymbol{v}_{m+1} \quad (3.15)$$

in (3.11), we observe that also in this case the error function $\widetilde{e}_m(z)$ is of a similar form as the original function $\widetilde{f}(z) = zf(z)$, in the sense that it is of the form $ze_m(z)$, where $e_m(z)$ denotes the error function for $f(z)$ from (3.4). The remaining term (3.15) in the error representation does not hamper our restarting approach, as it can be evaluated along with $\widetilde{\boldsymbol{f}}_m$ from (3.12) at almost no cost. A slightly modified restarting procedure for functions of the form $zf(z)$ thus simply involves evaluating (3.15), subtracting it from the current iterate and then proceeding as before by approximating $\widetilde{e}_m(A)\boldsymbol{v}_{m+1}$ by a new Arnoldi cycle. Corollary 3.3 can also be straightforwardly transferred to this modified situation, allowing all restarts to be performed in the same way.

Overall, we are now in a position to use Algorithm 1 for a broad class of functions with an error function representation based on integrals instead of divided differences. While mathematically equivalent, this seems favorable from a computational point of view since the numerical evaluation of integrals is typically more stable than the evaluation of difference formulas (see, e.g., [13] for a discussion of this topic in the context of solving differential equations). The numerical experiments in section 5 indeed demonstrate that our restarting method based on the integral representation of the error is more stable than algorithms based on the divided difference formulas from [14] or [28]. In our approach only the evaluation of $w_m(t)$ may appear prone to under- or overflow, as it is a polynomial of possibly high degree m. However, note that

$$\frac{\gamma_m}{w_m(t)} = h_{m+1,m} \boldsymbol{e}_m^T \left(tI_m - H_m \right)^{-1} \boldsymbol{e}_1, \quad (3.16)$$

see, e.g., [36], so that the necessary scalar quantities can be computed by solving a shifted linear system of dimension m. Another technique for reliably evaluating $w_m(t)$ in factored form is to use a suitable reordering of its zeros while computing the product. In our implementation and numerical tests reported in section 5 we always used the shifted linear system approach based on (3.16).

4. Evaluation of the error function by numerical quadrature. The presented restarting method relies on the ability to approximate the action of the error function, $e_m(A)\boldsymbol{b}$, which in turns requires the approximation of the integral in (3.4) by numerical quadrature. An arbitrary quadrature formula for $e_m(z)$ from (3.4) is of the form

$$\widehat{e}_m(z) = \gamma_m \sum\nolimits_{i=1}^{\ell} \omega_i \frac{g(t_i)}{w_m(t_i)} \frac{1}{t_i - z} \tag{4.1}$$

for quadrature nodes $t_i \in \Gamma$ and weights ω_i. Clearly, (4.1) is a rational approximation of type $(\ell - 1, \ell)$ to $e_m(z)$ with poles $t_1, \ldots, t_\ell$. The restarting approach based on quadrature of $e_m(z)$ hence is similar in spirit to the method of [2] (see also Algorithm 2), where a rational approximation of $f(z)$ is used to allow for restarts with constant work per restart cycle. Indeed, the following result states that under certain assumptions both approaches are mathematically equivalent.

LEMMA 4.1. *Let the quadrature nodes and weights in* (4.1) *be fixed throughout all restart cycles in Algorithm 1. Let Algorithm 2 utilize a rational approximation of the form* (2.8) *with poles* t_i *and weights* $\alpha_i = \omega_i g(t_i)$. *Assume that this quadrature formula is also used to evaluate* f *in the first restart cycle of Algorithm 1. Then both algorithms produce the same approximations* $\boldsymbol{f}_m^{(k)}$ *at each restart cycle* $k \geq 1$.

Proof. From (2.9) and (4.1) (with $w_m(t_i) = 1$ in the first restart cycle) it immediately follows that both algorithms produce the same first Arnoldi approximation

$$\boldsymbol{f}_m^{(1)} = \|\boldsymbol{b}\| V_m^{(1)} \sum\nolimits_{i=1}^{\ell} \omega_i g(t_i)(t_i I - H_m^{(1)})^{-1} \boldsymbol{e}_1. \tag{4.2}$$

In subsequent restart cycles $k \geq 2$ of Algorithm 1, using the error function representation (4.1), the approximations are computed as

$$\boldsymbol{f}_m^{(k)} = \boldsymbol{f}_m^{(k-1)} + \|\boldsymbol{b}\| V_m^{(k)} \sum\nolimits_{i=1}^{\ell} \frac{\omega_i \gamma_m^{(1)} \cdots \gamma_m^{(k-1)} g(t_i)}{w_m^{(1)}(t_i) \cdots w_m^{(k-1)}(t_i)} (t_i I - H_m^{(k)})^{-1} \boldsymbol{e}_1. \tag{4.3}$$

From (2.10) we get $\boldsymbol{r}_{i,k} = h_{m+1,m}^{(k-1)} (\boldsymbol{e}_m^T \boldsymbol{r}_{i,k-1})(t_i I - H_m^{(k)})^{-1} \boldsymbol{e}_1$. Repeated application of (3.16) yields

$$h_{m+1,m}^{(k-1)} (\boldsymbol{e}_m^T \boldsymbol{r}_{i,k-1}) = \frac{\gamma_m^{(1)} \cdots \gamma_m^{(k-1)}}{w_m^{(1)}(t_i) \cdots w_m^{(k-1)}(t_i)},$$

so that (4.3) is equivalent to

$$\boldsymbol{f}_m^{(k)} = \boldsymbol{f}_m^{(k-1)} + \|\boldsymbol{b}\| V_m^{(k)} \sum\nolimits_{i=1}^{\ell} \omega_i g(t_i) \boldsymbol{r}_{i,k},$$

which is precisely the update formula of Algorithm 2 when $\alpha_i = \omega_i g(t_i)$. ∎

There are three main reasons why a restarting approach based on quadrature is potentially superior to the method of [2] (Algorithm 2) based on a *fixed* rational

approximant. First of all, the construction of a fixed rational approximant r such that $r(A)\boldsymbol{b} \approx f(A)\boldsymbol{b}$ requires some a-priori information about a spectral region of A, which may be difficult or impossible to obtain, in particular, for non-Hermitian A. Our integral representation of the error (3.7) allows for the automated construction of rational approximations without using any spectral information, in particular, if the path Γ does not depend on $\mathrm{spec}(A)$ (as is the case for Stieltjes functions).

Secondly, in Algorithm 2, the same rational approximation has to be used in every restart cycle because the vectors $\boldsymbol{r}_{i,k}$ in (2.9) and (2.10) have to be stored and updated separately for each of the underlying shifted linear systems. In our new algorithm, however, there is no reason why the quadrature rule (4.1) needs to be fixed throughout all restart cycles. In fact, this quadrature rule can be adapted dynamically so that at each restart cycle k the required quantity $e_m^{(k-1)}(H_m^{(k)})\boldsymbol{e}_1$ is computed with sufficient accuracy. We will find that for later restart cycles the number of quadrature nodes ℓ needed for a fixed absolute target accuracy can typically be decreased simply because the integrand in (3.7) becomes uniformly smaller in magnitude. Even when the path Γ must depend on A, as is the case for restarting the exponential function of non-Hermitian A, we can cheaply use Ritz information available in our restarting algorithm to adaptively choose Γ (see sections 4.3 and 5.3).

Thirdly, quadrature naturally allows for adaptivity and error control. In our implementation given in Algorithm 3 we use a very simple form of adaptive quadrature. At each restart cycle we approximate the integral of the error function (3.4) with different numbers of quadrature nodes $\widetilde{\ell}$ and ℓ with $\widetilde{\ell} < \ell$. In our implementation we initialize $\widetilde{\ell} = 8$ and use $\ell = \mathrm{round}(\sqrt{2} \cdot \widetilde{\ell})$. If the norm of the difference between the resulting quadrature approximations is larger than a prescribed error tolerance `tol`, we further increase the number of quadrature nodes by a factor $\sqrt{2}$ until the desired accuracy is reached. If the number of quadrature nodes did not increase in a given restart cycle, we start the next restart cycle with a reduced number of quadrature nodes, see Algorithm 3.

REMARK 4.2. *In* [15] *an extension of Algorithm 2 with deflated restarting was proposed. Such a deflation procedure can be straightforwardly adapted to Algorithm 3: after each restart cycle k a reordered Schur decomposition of $H_m^{(k)}$ is used to restart the Arnoldi process with a set of d target Ritz vectors. The analysis in* [15], *in particular Theorem 3.2, describes how the nodal polynomial $w_m(t)$ in* (3.4) *needs to be modified with deflated restarting.*

A crucial aspect for achieving a robust and efficient restarting algorithm is the choice of a proper quadrature rule (4.1). In principle, any convergent quadrature rule may be used. In our case, adaptive variants such as Gauss–Kronrod quadrature (see [9]) seem appropriate, although for some special functions one can exploit the structure of the integrand in the error function (3.4). We will therefore take a closer look at some particularly important functions.

4.1. $f(z) = z^{-\alpha}$ for $\alpha \in (0,1)$. The inverse fractional powers $f(z) = z^{-\alpha}$ for $\alpha \in (0,1)$ are Stieltjes functions, see (3.9). When working with Stieltjes functions in general, one has the advantage that the path Γ is always explicitly known and independent of $\mathrm{spec}(A)$, but on the other hand one has to deal with an infinite integration interval. Although there exist Gaussian quadrature rules for infinite (or half-infinite) integration intervals (see, e.g., [19]), we will pursue a different approach here by applying a variable substitution for transforming the infinite integral in (3.9) into a finite one. A similar kind of integral transformation was also used in [7] when working with integral representations for the matrix p-th root.

Algorithm 3: Quadrature-based restarted Arnoldi approximation for $f(A)\boldsymbol{b}$.

Given: A, $\boldsymbol{b}$, f, m, `tol`
Compute the Arnoldi decomposition $AV_m^{(1)} = V_m^{(1)} H_m^{(1)} + h_{m+1,m}^{(1)} \boldsymbol{v}_{m+1}^{(1)} \boldsymbol{e}_m^T$
  with respect to A and $\boldsymbol{b}$.
Set $\boldsymbol{f}_m^{(1)} := \|\boldsymbol{b}\| V_m^{(1)} f(H_m^{(1)}) \boldsymbol{e}_1$.
Set $\widetilde{\ell} := 8$ and $\ell := \mathrm{round}(\sqrt{2} \cdot \widetilde{\ell})$.
for $k = 2, 3, \ldots$ until convergence **do**
  Compute the Arnoldi decomposition $AV_m^{(k)} = V_m^{(k)} H_m^{(k)} + h_{m+1,m}^{(k)} \boldsymbol{v}_{m+1}^{(k)} \boldsymbol{e}_m^T$.
    with respect to A and $\boldsymbol{v}_{m+1}^{(k-1)}$.
  Choose sets $(\widetilde{t}_i, \omega_i)_{i=1,\ldots,\widetilde{\ell}}$ and $(t_i, \omega_i)_{i=1,\ldots,\ell}$ of quadrature nodes/weights.
  Set `accurate` := `false` and `refined` := `false`.
  while `accurate` = `false` **do**
    Compute $\widetilde{\boldsymbol{h}}_m^{(k)} = e_m^{(k-1)}(H_m^{(k)}) \boldsymbol{e}_1$ by quadrature of order $\widetilde{\ell}$.
    Compute $\boldsymbol{h}_m^{(k)} = e_m^{(k-1)}(H_m^{(k)}) \boldsymbol{e}_1$ by quadrature of order ℓ.
    if $\|\boldsymbol{h}_m^{(k)} - \widetilde{\boldsymbol{h}}_m^{(k)}\| <$ `tol` **then**
      `accurate` := `true`.
    else
      Set $\widetilde{\ell} := \ell$ and $\ell := \mathrm{round}(\sqrt{2} \cdot \widetilde{\ell})$.
      Set `refined` := `true`.
  Set $\boldsymbol{f}_m^{(k)} := \boldsymbol{f}_m^{(k-1)} + \|\boldsymbol{b}\| V_m^{(k)} \boldsymbol{h}_m^{(k)}$.
  if `refined` = `false` **then**
    Set $\ell := \widetilde{\ell}$ and $\widetilde{\ell} := \mathrm{round}(\ell/\sqrt{2})$.

LEMMA 4.3. *Let $z \in \mathbb{C} \setminus \mathbb{R}^-$. Then for all $\beta > 0$*

$$z^{-\alpha} = \frac{2\sin((\alpha+1)\pi)\beta^{1-\alpha}}{\pi} \int_{-1}^{1} \frac{(x-1)^{-\alpha}(x+1)^{\alpha-1}}{-\beta(1-x) - z(1+x)} \,\mathrm{d}x. \tag{4.4}$$

Proof. This follows by applying the Cayley transform $t = -\beta\frac{1-x}{1+x}$ to (3.9). ∎

The representation (4.4) is particularly convenient for our purpose, because it can be very efficiently dealt with by quadrature. To this end, observe that the numerator of the integrand in (4.4) is exactly the *Jacobi weight function* $\omega(x) = (x-1)^{-\alpha}(x+1)^{\alpha-1}$. The Jacobi weight function can be resolved exactly by using *Gauss–Jacobi quadrature*, cf. [9], despite its singularities at both endpoints of the interval of integration. The remaining integrand $\frac{1}{-\beta(1-x)-z(1+x)}$ does not have any singularities as long as z stays away from the negative real axis. The following result shows that Gauss–Jacobi quadrature for (4.4) corresponds to a Padé approximant (cf. [3]), and it also gives a hint on how to choose the transformation parameter β. The connection between Padé approximation and quadrature is classical, but we include the following lemma because we could not find it in this explicit form in the literature.

LEMMA 4.4. *Let $\beta > 0$ and let x_i and ω_i $(i = 1, \ldots, \ell)$ be the nodes and weights of the ℓ-node Gauss–Jacobi quadrature rule on $[-1, 1]$. Then*

$$r_{\ell-1,\ell}(z) = \frac{2\sin((\alpha+1)\pi)\beta^{1-\alpha}}{\pi} \sum\nolimits_{i=1}^{\ell} \frac{\omega_i}{-\beta(1-x_i) - z(1+x_i)} \tag{4.5}$$

is the $(\ell-1,\ell)$-Padé approximant for $z^{-\alpha}$ with expansion point β.

Proof. Note that (4.5) clearly is a rational function of type $(\ell-1,\ell)$ in partial fraction form. Therefore we only have to verify the Padé matching conditions

$$\frac{\mathrm{d}^j}{\mathrm{d}z^j} z^{-\alpha}\Big|_{z=\beta} = \frac{\mathrm{d}^j}{\mathrm{d}z^j} r_{\ell-1,\ell}(z)\Big|_{z=\beta} \quad \text{for } j=0,\dots,2\ell-1. \tag{4.6}$$

The derivatives of $r_{\ell-1,\ell}(z)$ are given by

$$\frac{\mathrm{d}^j}{\mathrm{d}z^j} r_{\ell-1,\ell}(z) = \frac{2\sin((\alpha+1)\pi)\beta^{1-\alpha}}{\pi} \sum\nolimits_{i=1}^{\ell} (-1)^j \frac{j!\cdot(1+x_i)^j\cdot\omega_i}{(-\beta(1-x_i)-z(1+x_i))^{j+1}}. \tag{4.7}$$

For $z=\beta$ all denominators in (4.7) become independent of x_i and we arrive at

$$\frac{\mathrm{d}^j}{\mathrm{d}z^j} r_{\ell-1,\ell}(z)\Big|_{z=\beta} = \frac{2\sin((\alpha+1)\pi)\beta^{1-\alpha}}{\pi} \sum\nolimits_{i=1}^{\ell} (-1)^j \frac{j!\cdot(1+x_i)^j\cdot\omega_i}{(-2\beta)^{j+1}}. \tag{4.8}$$

As Gauss–Jacobi quadrature with ℓ nodes is exact for polynomials up to degree $2\ell-1$, we have for $j=0,\dots,2\ell-1$ the relation

$$\frac{\mathrm{d}^j}{\mathrm{d}z^j} r_{\ell-1,\ell}(z)\Big|_{z=\beta} = \frac{2j!\cdot\sin((\alpha+1)\pi)\beta^{1-\alpha}}{(-2\beta)^{j+1}\pi}(-1)^j \int_{-1}^{1} (1+x)^j(1-x)^{-\alpha}(1+x)^{1-\alpha}\,\mathrm{d}x.$$

Differentiating the right-hand side of (4.4) and evaluating at β gives the same result, which completes the proof. ∎

Lemma 4.4 suggests that the rational approximation (4.5) is particularly well suited for approximating $A^{-\alpha}$ when the spectrum of A is clustered around β. A reasonable choice of the transformation parameter therefore is $\beta = \mathrm{trace}(A)/n$, the arithmetic mean of the eigenvalues of A. We only note that in our context of using quadrature for evaluating the error function in a restarted Arnoldi algorithm, more sophisticated choices of β are certainly possible, for example based on the Ritz values $\mathrm{spec}(H_m^{(k)})$ which can be explicitly computed when m is small. Our numerical experiments suggest, however, that the method is not very sensitive to the choice of β: although an unfortunate choice of β may well increase the number of required quadrature nodes ℓ, the computational cost of evaluating the quadrature rule is typically negligible compared to the matrix-vector products and orthogonalizations in the Arnoldi process.

So far our analysis of the quadrature formula was carried out for the original function $f(z)=z^{-\alpha}$, but we will rather use quadrature to approximate the error function $e_m(z)$ in our algorithm. The situation gets slightly more difficult in this case: for example, it is generally not excluded that the integrand in (3.4) will have singularities on the interval of integration. Applying the Cayley transform $t=-\beta\frac{1-x}{1+x}$ to (3.4) and using the integral representation (3.9) of $z^{-\alpha}$, the integral to be approximated when evaluating the error function becomes

$$\frac{2\sin((\alpha+1)\pi)\beta^{1-\alpha}}{\pi} \int_{-1}^{1} \frac{1}{w_m(-\beta\frac{1-x}{1+x})} \frac{(x-1)^{-\alpha}(x+1)^{\alpha-1}}{-\beta(1-x)-z(1+x)}\,\mathrm{d}x. \tag{4.9}$$

While the singularities at the endpoints of the interval $[-1,1]$ can still be handled by Gauss–Jacobi quadrature, the term $1/w_m(-\beta\frac{1-x}{1+x})$, being the reciprocal of a polynomial of degree m, introduces m additional singularities in the integrand. Recalling

the definition $w_m(z) = (z - \theta_1) \cdots (z - \theta_m)$ of the nodal polynomial, one easily sees that the singularities of the non-transformed integrand are exactly the Ritz values, which can in general lie anywhere in the field of values of A, in or outside the interval of integration. Hence we can only guarantee that there are no singularities on the interval of integration if the field of values of A is disjoint from the negative real axis. The most important special case in which this is known to be satisfied is when A is Hermitian positive definite. In this case, the field of values of A reduces to the interval $[\lambda_{\min}, \lambda_{\max}]$, where $\lambda_{\min}, \lambda_{\max} > 0$ denote the smallest and largest eigenvalue of A, respectively, so that all Ritz values are positive and bounded away from zero. For non-Hermitian A, this can in most cases not be guaranteed, and it may occasionally happen that a Ritz value appears on the negative real axis. We emphasize that this is not a problem specific to our quadrature-based restarting approach but rather to the nature of the functions $f(z) = z^{-\alpha}$ and the respective error functions, which are not defined for matrices with eigenvalues on the negative real axis. All other restarting algorithms can potentially produce Ritz values on the negative real axis, too, if the field of values of A is not disjoint from this set.

By using the representation $z^\alpha = z z^{\alpha-1}$ for $\alpha \in (0, 1)$, the presented techniques can be extended directly to positive fractional powers with the error representation from Corollary 3.4.

4.2. $f(z) = \log(1+z)/z$**.** The function $f(z) = \log(1+z)/z$ also belongs to the class of Stieltjes functions, see (3.10). Therefore the techniques and ideas are similar to those presented in section 4.1. Again, the infinite interval of integration can be easily transformed into a finite interval.

LEMMA 4.5. *Let $z \in \mathbb{C} \setminus (-\infty, -1]$. Then*

$$\frac{\log(1+z)}{z} = \int_{-1}^{1} \frac{1}{z(1+x)+2} \, \mathrm{d}x. \tag{4.10}$$

Proof. This follows by applying the transformation $t = -2/(1+x)$ to (3.10). ∎

The integrand in (4.10) is analytic in a neighborhood of $[-1, 1]$ as long as z stays away from $(-\infty, -1]$, so that Gauss–Legendre quadrature is an obvious choice for approximating this integral. In this case we can again make a connection to Padé approximants.

LEMMA 4.6. *Let $\beta > 0$ and let x_i and ω_i $(i = 1, \ldots, \ell)$ be the nodes and weights of the ℓ-node Gauss–Legendre quadrature rule on $[-1, 1]$. Then*

$$r_{\ell-1,\ell}(z) = \sum\nolimits_{i=1}^{\ell} \frac{\omega_i}{-z(1+x_i)+2} \tag{4.11}$$

is the $(\ell-1, \ell)$-Padé approximant for $\log(1+z)/z$ with expansion point 0.

Proof. The proof proceeds analogously to the proof of Lemma 4.4 by noting that ℓ-node Gauss–Legendre quadrature is exact for polynomials of degree up to $2\ell - 1$, and using the formula

$$\frac{\mathrm{d}^j}{\mathrm{d}z^j} r_{\ell-1,\ell}(z) = \sum\nolimits_{i=1}^{\ell} (-1)^j \frac{j! \cdot (1+x_i)^j \cdot \omega_i}{(z(1+x_i)+2)^{j+1}} \tag{4.12}$$

for the derivatives of $r_{\ell-1,\ell}(z)$. ∎

Note that in contrast to the situation with $z^{-\alpha}$, the expansion point for the Padé approximant (4.11) is fixed to be 0, and good approximation properties can

be expected especially when A has eigenvalues near the origin. Otherwise, using the technique suggested by Corollary 3.4 and properties of the logarithm, one can instead approximate $\log(I + (\beta^{-1}A - I))\boldsymbol{b} = \log(A)\boldsymbol{b} - \log(\beta)\boldsymbol{b}$ with the matrix $(\beta^{-1}A - I)$ having a "shrinked" spectrum.

Concerning the singularities of the integrand in (3.4), the analysis from the previous section exactly carries over since the nodal polynomial $w_m(z)$ depends only on the Ritz values (and thus A and $\boldsymbol{b}$) but not on the function f to be approximated.

4.3. $f(z) = e^z$**.** Clearly, the exponential is not a Stieltjes function, but it can be represented via the Cauchy integral formula as

$$e^z = \frac{1}{2\pi i} \int_\Gamma \frac{e^t}{t - z} \, \mathrm{d}t, \tag{4.13}$$

so that our quadrature-based algorithm can be applied. In the case where A is negative semi-definite, the trapezoidal rule on parabolic, hyperbolic or cotangent Hankel contours Γ is well suited for approximating e^z via quadrature, see [40,43,44]. However, the contours discussed in these papers depend on the number of quadrature nodes ℓ and bend towards the imaginary axis as ℓ becomes larger, which causes oscillations in the integrand. Here we are primarily interested in non-Hermitian matrices and the contour thus needs to depend on A and the Ritz values computed in all restart cycles. In numerical experiments not reported here we hoped to achieve a small approximation error in a larger neighborhood of the negative real axis by using the quadrature rules of [40, 43, 44] with a slightly increased number of quadrature nodes. However, with this straightforward approach we observed numerical instabilities and degrading accuracies for the resulting scalar approximations of e^z even on the negative real axis $z \leq 0$ (for which the Hankel contours have been optimized). We therefore decided to use a possibly non-optimal but fixed parabolic contour Γ parameterized as

$$\gamma(\zeta) = a + i\zeta - c\zeta^2, \quad \zeta \in \mathbb{R}, \tag{4.14}$$

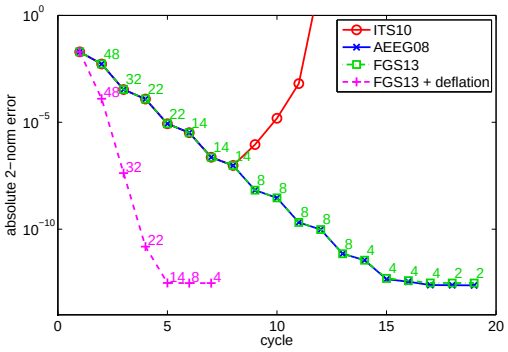
where the parameters $a, c > 0$ can be used to shift the contour and to widen or narrow the region enclosed by the contour. We choose these parameters dynamically after each restart such that all Ritz values are enclosed by the contour with some positive distance. More precisely, if Θ denotes the set of all Ritz values accumulated until a certain restart, we choose $a = \max\big[\mathrm{real}(\Theta + 1) \cup \{1\}\big]$ and $c = \min\big[\{0.25\} \cup \sqrt{a + i \cdot \mathrm{imag}(\Theta) - \Theta}\big]$, where all operations on the set Θ are performed element-wise. To truncate the infinite interval of integration for ζ up to a given error tolerance `tol`, we define a truncation parameter $\zeta_t = \sqrt{1 - \log(\texttt{tol})/c}$ such that $|e^{\gamma(\pm\zeta_t)}| = \texttt{tol}$. We then approximate (4.13) as

$$e^z \approx \frac{1}{2\pi i} \int_{-\zeta_t}^{\zeta_t} \frac{e^{\gamma(\zeta)}\gamma'(\zeta)}{\gamma(\zeta) - z} \, \mathrm{d}\zeta \approx \frac{2\zeta_t}{\ell} \sum\nolimits_{j=1}^{\ell} \frac{e^{\gamma(\zeta^{(j)})}\gamma'(\zeta^{(j)})}{\gamma(\zeta^{(j)}) - z}, \quad \zeta^{(j)} = \zeta_t\Big(\frac{2j-1}{\ell} - 1\Big),$$

which corresponds to the application of the ℓ-node midpoint rule on $[-\zeta_t, \zeta_t]$.

For the quadrature approximation of error functions in subsequent restart cycles we use the same contour parametrization (4.14), with a and c possibly adapted to enclose the union of all Ritz values, and the same error tolerance `tol`.

5. Numerical experiments. In this section we demonstrate the stability and efficiency of the proposed restarting approach, Algorithm 3, by applying it to various model problems and problems from relevant applications. All computations were



method	time
divided differences (ITS10)	0.34
rational approx. (AEEG08)	0.72
quadrature (FGS13)	0.77
quadrature + deflation	0.35

FIG. 5.1. *Approximating $A^{-1/2}\boldsymbol{b}$: Convergence history (left) and running times (right) for the different restarting algorithms. The numbers next to the curves for the new quadrature-based methods indicate the number of quadrature nodes used for evaluating the error function in the corresponding restart cycle. The restart length is $m = 50$ in all cases, and in the variant with deflation a number of $d = 5$ Ritz vectors was used.*

performed in MATLAB 7. Since a part of MATLAB code is interpreted, MATLAB implementations are not always best suited for comparing running times of algorithms, but they are certainly appropriate to assess stability. Moreover, since all algorithms spend most of their time in sparse matrix-vector multiplications, which are calls to pre-compiled routines in MATLAB, larger differences in running times can be trusted to be significant.

5.1. 2D Laplacian, $f(z) = z^{-1/2}$. In this first example we compute $A^{-1/2}\boldsymbol{b}$, where A is the real, symmetric positive definite matrix arising from the finite difference discretization of the negative two-dimensional Laplace operator with $N = 100$ grid points in both spatial dimensions, so that $A \in \mathbb{R}^{10^4 \times 10^4}$. According to (3.9), the function f under consideration belongs to the class of Stieltjes functions and we will use the techniques discussed in section 4.1 in our quadrature-based restarting algorithm. We compare the quadrature-based approach to the restarting algorithm from [28] based on divided differences, as well as the restarting algorithm from [2] using as the rational function the best relative Zolotarev approximation of order 24 on the spectral interval of A [45]. In addition, the behavior of the quadrature-based deflated restarting method [15] with $d = 5$ target eigenvalues is reported. The left part of Figure 5.1 shows the convergence history of the different methods for restart length $m = 50$. We observe that the divided difference-based method (denoted as ITS10) becomes highly unstable after the eighth restart, while the quadrature-based approach (denoted as FGS13) and the approach using Zolotarev's rational approximation (denoted as AEEG08) show a non-distinguishable convergence behavior. The table on the right-hand side of Figure 5.1 reports the running time of the different methods for $m = 50$. We observe that the method using divided differences has the fastest running time (but is unstable), while the approach using a rational approximation and our quadrature-based restarting approach are slightly slower and require almost the same running time. Note, however, that the time needed for estimating the smallest and largest eigenvalue of A using the MATLAB routine `eigs` when constructing the Zolotarev approximation to $z^{-1/2}$ is not included in the reported timings and takes about one second. In contrast, the new quadrature-based method works as a black-box and does not require any spectral information or additional computations, so that in total it needs less than half the time of the method from [2] if $f(A)\boldsymbol{b}$ is

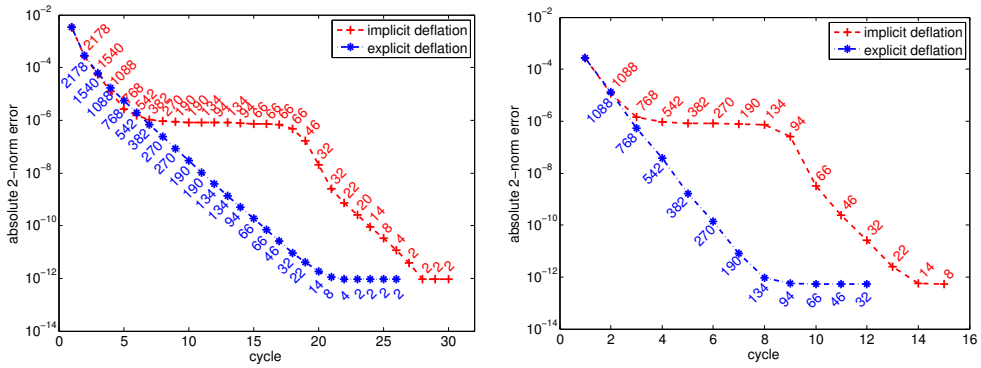


FIG. 5.2. *Approximating* $\mathrm{sign}(Q)\boldsymbol{b}$*: Convergence history for the quadrature-based method with implicit and explicit deflation of eigenvectors corresponding to the* $d = 15$ *smallest eigenvalues of* Q *with restart length* $m = 20$ *(left) and* $m = 40$ *(right).*

computed once for a single vector $\boldsymbol{b}$.

The numbers next to the convergence curves of the quadrature-based restarting method correspond to the numbers of quadrature nodes required to reach the desired target accuracy of $\texttt{tol} = 10^{-13}$ at each restart cycle. We observe that the numbers decrease in later restart cycles, with no more than 8 quadrature nodes required in the last eleven restart cycles of the method without deflation. This can be explained in two ways. On the one hand, the *relative* accuracy needed to reach a certain prescribed *absolute* error tolerance is lower in later restart cycles simply because the norm of the error becomes smaller. On the other hand, the integrand in the representation of the error function decays more rapidly when t goes towards $-\infty$ for Stieltjes functions and positive definite matrices in later cycles (because all Ritz values are on the positive real axis), so that the integral becomes easier to handle numerically.

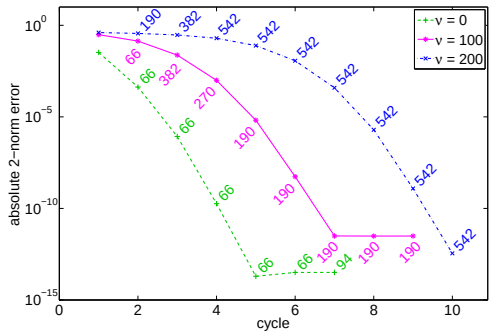
5.2. Overlap Dirac operator, $f(z) = \mathrm{sign}(z)$. In *quantum chromodynamics* (QCD), an area of Theoretical Physics, the strong interaction between quarks is studied. In *lattice quantum chromodynamics*, this theory is simulated on a four-dimensional space–time lattice with 12 variables, corresponding to all possible combinations of three colors and four spins, at each lattice point. The simulation of *overlap fermions*, which preserve the so-called *chiral symmetry* on the lattice, requires the solution of linear systems involving the overlap Dirac operator [33]

$$N_{\mathrm{ovl}} := \rho I + \Gamma_5 \,\mathrm{sign}(Q), \tag{5.1}$$

where $\rho > 1$ is a mass parameter, Q represents a periodic nearest-neighbor coupling on the lattice, and Γ_5 is a permutation which permutes the spins on each lattice point in an identical manner. The matrix Q is very large (of size $n = 10^6$ or greater for realistic grid sizes), sparse and complex. Depending on the so-called *chemical potential*, Q is Hermitian (zero chemical potential) or non-Hermitian (nonzero chemical potential). In the following example, we will only investigate the Hermitian case.

As it is not feasible to explicitly compute $\mathrm{sign}(Q)$, the preferred technique for solving linear systems with (5.1) is using an iterative method which only performs matrix-vector products with $\mathrm{sign}(Q)$, and hence Krylov subspace techniques are the methods of choice. At each outer Krylov iteration one therefore has to compute

$$\mathrm{sign}(Q)\boldsymbol{b} \tag{5.2}$$



	$\nu = 0$	$\nu = 100$	$\nu = 200$
a	1	1	1
c	0.25	0.04	0.004
ζ_t	11.12	29.48	87.89

FIG. 5.3. *Approximating $e^{sA}\boldsymbol{b}$: Convergence history and number of quadrature nodes for varying convection parameter $\nu = 0, 100, 200$ (left) and parameters determining the Hankel contour (right). The restart length is $m = 70$ in all cases.*

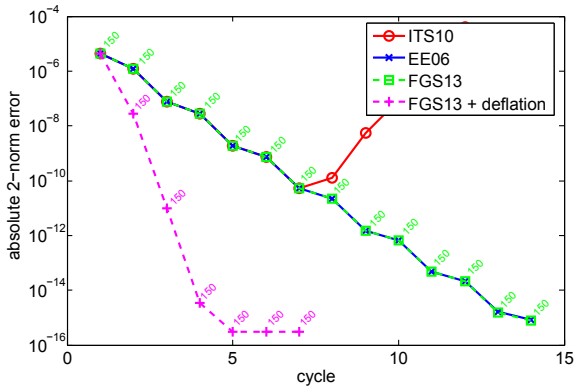
with the vector $\boldsymbol{b}$ changing from one iteration to the next. As (5.2) needs to be evaluated many times in a single simulation, it is common practice to explicitly deflate the eigenvectors corresponding to the d smallest eigenvalues of Q *once*, thereby speeding up convergence in *all* iterations.

We report on the results of our quadrature-based restarting method for a simulation on a lattice with 16 points in each space–time direction, resulting in a matrix $Q \in \mathbb{C}^{12\cdot 16^4 \times 12\cdot 16^4}$. The sign function is typically computed via the relation

$$\operatorname{sign}(Q)\boldsymbol{b} = (Q^2)^{-1/2} Q\boldsymbol{b}, \tag{5.3}$$

so that we can use our techniques developed in section 4.1 for $z^{-\alpha}$ with $\alpha = 1/2$. The convergence history in case of implicit and explicit deflation is shown in Figure 5.2 for restart length $m = 20$ (left) and $m = 40$ (right). The deflation parameter is $d = 15$ in all cases. We observe that, after an initial phase of slow convergence, the restarting method with implicit deflation shows the same convergence slope as the method with explicit deflation. This is in agreement with the analysis from [15] which states that both methods exhibit the same asymptotic behavior. The running time of the method needed to reach an accuracy of 10^{-10} is approximately 240 seconds with explicit deflation and 350 seconds with implicit deflation. Note, however, that the time needed for the computation of the d smallest eigenvectors of Q is not included in the running time of the algorithm with explicit deflation. Using the MATLAB function `eigs`, the explicit computation of 15 eigenvectors of Q takes about 4 hours, so that this technique should only be used if $\operatorname{sign}(Q)\boldsymbol{b}$ has to be computed for a very large number of vectors $\boldsymbol{b}$.

5.3. 2D convection–diffusion, $f(z) = e^{sz}$. In this paragraph we present results for the matrix exponential function applied to symmetric and non-symmetric stable matrices A using the techniques from section 4.3. The matrices A correspond to the standard finite difference discretization of a 2D convection–diffusion equation on the unit square with constant convection field and different convection parameters ν. The case $\nu = 0$ corresponds to a symmetric problem and for increasing ν the non-normal matrix A has eigenvalues with larger imaginary parts. We choose the scaling parameter $s = 2 \cdot 10^{-3}$ and use 500 discretization points in both spatial dimensions, resulting in a matrix $A \in \mathbb{R}^{500^2 \times 500^2}$. The left part of Figure 5.3 shows the convergence history of our quadrature-based restarting method with restart length



method	time
divided differences (ITS10)	0.28
accumulated H_{km} (EE06)	17.49
quadrature (FGS13)	1.24
quadrature + deflation	0.64

FIG. 5.4. *Approximating* $(e^{-s\sqrt{A}} - I)A^{-1}\boldsymbol{b}$*: Convergence history (left) and running times (right) for the different restarting procedures. The restart length is* $m = 50$ *in all cases, and in the variant with deflation a number of* $d = 5$ *Ritz vectors was used.*

$m = 70$ and three different values of ν. Again, the numbers next to the convergence curves indicate the number of quadrature nodes used in each restart cycle. Note that the choice of the integration path Γ must also be adaptive for the exponential function, and the table on the right part of Figure 5.3 reports the maximum values of the parameters a and ζ_t and the minimum values of the parameter c determining the parabolic Hankel contour (4.14). We observe that with increasing convection ν the number of required quadrature nodes increases. This seems reasonable because the complex region on which the error function needs to be approximated becomes larger. This is also reflected in the values of the parameter c reported in the table on the right hand-side of Figure 5.3, which show that the contour becomes wider with increasing convection ν.

5.4. 2D Laplacian, $f(z) = (e^{-s\sqrt{z}} - 1)/z$**.** We consider the Stieltjes function

$$f(z) = \frac{e^{-s\sqrt{z}} - 1}{z} = \frac{1}{\pi} \int_{-\infty}^{0} \frac{\sin(st^{-1}\sqrt{-t})}{t - z} \, \mathrm{d}t. \tag{5.4}$$

This function has important applications for solving wave equations, because certain solutions may be written as rational functions of $f(z)$ from (5.4) and $g(z) = z^{-\alpha}$. Standard adaptive quadrature rules suitable for infinite intervals like, e.g., Gauss–Kronrod quadrature can be used to apply our restarting approach with this function.

We again choose the test matrix A as the finite difference discretization of the negative Laplace operator in two spatial dimensions with 100 discretization points in each spatial direction, and the parameter $s = 10^{-3}$. Although this matrix is symmetric positive definite, the method from [2] cannot be used straightforwardly, as rational approximations for f seem to be difficult to construct. We therefore compare our quadrature-based restarting approach with the method from [14] (denoted as EE06) in which f is evaluated on a Hessenberg matrix H_{km} of growing size.

The plot on the left-hand side of Figure 5.4 shows the convergence curves for the different restart procedures, timings are again given in the table on the right-hand side. The restart length has been chosen as $m = 50$ in all cases, and in the variant with deflation a number of $d = 5$ Ritz vectors was used. In this situation where no rational approximation can be computed a priori, the running times clearly show the superiority of the new quadrature-based algorithm over the original method of [14],

even though we simply used the MATLAB routine `quadgk` [38] to evaluate the error function and therefore did not exploit update formulas for the values $w_m(t_i)$, resulting in many superfluous computations. Nevertheless, our quadrature-based method is faster by a factor of about fifteen. The routine `quadgk` by default applies a variable transformation to the integrand, partitions the integration interval into 10 subintervals, then applies the Gauss(7)–Kronrod(15) rule on each subinterval, and refines if necessary. In this example no refinements were necessary, resulting in a total number of 150 function evaluations at each restart cycle (see the left of Figure 5.4).

6. Conclusions. We derived a quadrature-based Arnoldi restarting method for computing $f(A)\boldsymbol{b}$ based on an integral representation of the error. Our approach allows for restarting with essentially constant work per cycle for a large class of functions. We have shown that our method is similar in spirit to the one from [2] but has larger potential for adaptivity as the underlying rational approximation is not fixed throughout all restarts and can be chosen dynamically at each cycle. Moreover, we have shown that for some special functions the canonical quadrature rules correspond to Padé approximants, and that adaptive quadrature allows for efficient control of the approximation error. We compared our method to other existing restarting approaches for a number of problems and illustrated its favorable numerical stability and efficiency. The extension of our method to problems where no suitable integration path is directly available and the tuning and optimization of the variant for the exponential function will be subject of future research.

REFERENCES

[1] M. Afanasjew, M. Eiermann, O. G. Ernst, and S. Güttel, *A generalization of the steepest descent method for matrix functions*, Electron. Trans. Numer. Anal., 28 (2008), pp. 206–222.

[2] ———, *Implementation of a restarted Krylov subspace method for the evaluation of matrix functions*, Linear Algebra Appl., 429 (2008), pp. 229–314.

[3] G. A. Baker and P. Graves-Morris, *Padé Approximants*, Cambridge University Press, 1996.

[4] B. Beckermann and L. Reichel, *Error estimation and evaluation of matrix functions via the Faber transform*, SIAM J. Numer. Anal., 47 (2009), pp. 3849–3883.

[5] J. Bloch, A. Frommer, B. Lang, and T. Wettig, *An iterative method to compute the sign function of a non-Hermitian matrix and its application to the overlap Dirac operator at nonzero chemical potential*, Comput. Phys. Commun., 177 (2007), pp. 933–943.

[6] K. Burrage, N. Hale, and D. Kay, *An efficient implicit FEM scheme for fractional-in-space reaction–diffusion equations*, SIAM J. Sci. Comput., 34 (2012), pp. A2145–A2172.

[7] J. R. Cardoso, *Computation of the matrix pth root and its Fréchet derivative by integrals*, Electron. Trans. Numer. Anal., 39 (2012), pp. 414–436.

[8] P. Davies and N. Higham, *A Schur–Parlett algorithm for computing matrix functions*, SIAM J. Matrix Anal. Appl., 25 (2003), pp. 464–485.

[9] P. J. Davis and P. Rabinowitz, *Methods of Numerical Integration*, Academic Press, New York, 1975.

[10] C. de Boor, *Divided differences*, Surv. Approximation Theory, 1 (2005), pp. 46–69.

[11] V. Druskin and L. Knizhnerman, *Two polynomial methods of calculating functions of symmetric matrices*, U.S.S.R. Computational Mathematics and Mathematical Physics, 29 (1989), pp. 112–121.

[12] V. Druskin and L. Knizhnerman, *Extended Krylov subspaces: Approximation of the matrix square root and related functions*, SIAM J. Matrix Anal. Appl., 19 (1998), pp. 775–771.

[13] A. Dutt, L. Greengard, and V. Rokhlin, *Spectral deferred correction methods for ordinary differential equations*, BIT, 40 (2000), pp. 241–266.

[14] M. Eiermann and O. G. Ernst, *A restarted Krylov subspace method for the evaluation of matrix functions*, SIAM J. Numer. Anal., 44 (2006), pp. 2481–2504.

[15] M. Eiermann, O. G. Ernst, and S. Güttel, *Deflated restarting for matrix functions*, SIAM J. Matrix Anal. Appl., 32 (2011), pp. 621–641.

[16] T. Ericsson, *Computing functions of matrices using Krylov subspace methods*, Technical Report, Chalmers University of Technology, Göteborg, Sweden, 1990.
[17] A. Frommer and V. Simoncini, *Marix functions*, in Model Order Reduction: Theory, Research Aspects and Applications, H. A. van der Vorst, W. H. A. Schilders, and J. Rommes, eds., Mathematics in Industry, Springer-Verlag, Berlin/Heidelberg, 2008.
[18] E. Gallopoulos and Y. Saad, *Efficient solution of parabolic equations by Krylov approximation methods*, SIAM J. Sci. Stat. Comput., 13 (1992), pp. 1236–1264.
[19] W. Gautschi, *Quadrature formulae on half-infinite intervals*, BIT, 31 (1991), pp. 437–446.
[20] S. Güttel, *Rational Krylov Methods for Operator Functions*, PhD thesis, Fakultät für Mathematik und Informatik der Technischen Universität Bergakademie Freiberg, 2010.
[21] S. Güttel, *Rational Krylov approximation of matrix functions: Numerical methods and optimal pole selection*, GAMM Mitteilungen, 36 (2013), pp. 8–31.
[22] S. Güttel and L. Knizhnerman, *A black-box rational Arnoldi variant for Cauchy–Stieltjes matrix functions*, BIT Numer. Math., (2013).
[23] P. Henrici, *Applied and Computational Complex Analysis Vol. 2*, John Wiley & Sons, 1977.
[24] N. J. Higham, *Functions of Matrices: Theory and Computation*, SIAM, 2008.
[25] N. J. Higham and A. H. Al-Mohy, *Computing matrix functions*, Acta Numer., 19 (2010), pp. 159–208.
[26] M. Hochbruck and C. Lubich, *On Krylov subspace approximations to the matrix exponential operator*, SIAM J. Numer. Anal., 34 (1997), pp. 1911–1925.
[27] M. Hochbruck and A. Ostermann, *Exponential integrators*, Acta Numer., 19 (2010), pp. 209–286.
[28] M. Ilić, I. W. Turner, and D. P. Simpson, *A restarted Lanczos approximation to functions of a symmetric matrix*, IMA J. Numer. Anal., 30 (2010), pp. 1044–1061.
[29] L. Knizhnerman, *Calculation of functions of unsymmetric matrices using Arnoldi's method*, Comput. Math. Math. Phys., 31 (1991), pp. 1–9.
[30] L. Knizhnerman and V. Simoncini, *A new investigation of the extended Krylov subspace method for matrix function evaluations*, Numer. Linear Algebra Appl., 17 (2010), pp. 615–638.
[31] G. Lastman and N. Sinha, *Infinite series for logarithm of matrix, applied to identification of linear continuous-time multivariable systems from discrete-time models*, Electronics Letters, 27 (1991), pp. 1468–1470.
[32] I. Moret and P. Novati, *RD-rational approximations of the matrix exponential.*, BIT, 44 (2004), pp. 595–615.
[33] H. Neuberger, *Exactly massless quarks on the lattice*, Phys. Lett. B, 417 (1998), pp. 141–144.
[34] C. C. Paige, B. N. Parlett, and H. A. Van der Vorst, *Approximate solutions and eigenvalue bounds from Krylov subspaces*, Numer. Linear Algebra Appl., 1 (1993), pp. 1–7.
[35] Y. Saad, *Analysis of some Krylov subspace approximations to the exponential operator*, SIAM J. Numer. Anal., 29 (1992), pp. 209–228.
[36] ———, *Iterative Methods for Sparse Linear Systems, 2nd Edition*, SIAM, 2003.
[37] Y. Saad and M. Schultz, *GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems*, SIAM J. Sci. Stat. Comput., 7 (1986), pp. 856–869.
[38] L. F. Shampine, *Vectorized adaptive quadrature in Matlab*, Journal of Computational and Applied Mathematics, 211 (2008), pp. 131–140.
[39] B. Singer and S. Spilerman, *The representation of social processes by Markov models*, American Journal of Sociology, (1976), pp. 1–54.
[40] L. N. Trefethen, J. A. C. Weideman, and T. Schmelzer, *Talbot quadratures and rational approximations*, BIT, 46 (2006), pp. 653–670.
[41] J. van den Eshof, A. Frommer, T. Lippert, K. Schilling, and H. A. van der Vorst, *Numerical methods for the QCD overlap operator, I. Sign-function and error bounds*, Comput. Phys. Commun., 146 (2002), pp. 203–224.
[42] J. van den Eshof and M. Hochbruck, *Preconditioning Lanczos approximations to the matrix exponential*, SIAM J. Sci. Comput., 27 (2006), pp. 1438–1457.
[43] J. A. C. Weideman, *Optimizing Talbot's contours for the inversion of the Laplace transform*, SIAM J. Numer. Anal., 44 (2006), pp. 2342–2362.
[44] J. A. C. Weideman and L. N. Trefethen, *Parabolic and hyperbolic contours for computing the Bromwich integral*, Math. Comp., 76 (2007), pp. 1341–1356.
[45] E. I. Zolotarev, *Application of elliptic functions to the question of functions deviating least and most from zero*, Zap. Imp. Akad. Nauk. St. Petersburg, 30 (1877), pp. 1–59.