Memorandum 1928 (November 2010). ISSN 1874-4850. Available from: http://www.math.utwente.nl/publications
Department of Applied Mathematics, University of Twente, Enschede, The Netherlands

RESIDUAL, RESTARTING AND RICHARDSON ITERATION FOR THE MATRIX EXPONENTIAL

MIKE A. BOTCHEV*

To the memory of my father

Abstract. A well-known problem in computing some matrix functions iteratively is a lack of a clear, commonly accepted residual notion. An important matrix function for which this is the case is the matrix exponential. Assume, the matrix exponential of a given matrix times a given vector has to be computed. We interpret the sought after vector as a value of a vector function satisfying the linear system of ordinary differential equations (ODE), whose coefficients form the given matrix. The residual is then defined with respect to the initial-value problem for this ODE system. The residual introduced in this way can be seen as a backward error. We show how the residual can efficiently be computed within several iterative methods for the matrix exponential. This completely resolves the question of reliable stopping criteria for these methods. Furthermore, we show that the residual concept can be used to construct new residual-based iterative methods. In particular, a variant of the Richardson method for the new residual appears to provide an efficient way to restart Krylov subspace methods for evaluating the matrix exponential.

Key words. matrix exponential, residual, Krylov subspace methods, restarting, Chebyshev polynomials, stopping criterion, Richardson iteration, backward stability, matrix cosine

AMS subject classifications. 65F60, 65F10, 65F30, 65N22, 65L05

1. Introduction. Matrix functions, in particular the matrix exponential, has been an important tool in scientific computations for decades (see e.g. [9, 10, 12, 7, 13]). The lack of a clear notion for a residual for many matrix functions has been a known problem in iterative computation of matrix functions [2, 7, 27]. Although it is possible to define a residual for some matrix functions such as the inverse or the square root, for many important matrix functions including the matrix exponential, sine and cosine, no natural notion for residuals seems to exist.

Assume for given $A \in \mathbb{R}^{n\times n}$, such that $A+A^*$ is positive semidefinite, and $v \in \mathbb{R}^n$ the vector

$$y = \exp(-A)v \tag{1.1}$$

has to be computed. The question is how to evaluate the quality of an approximate solution

$$y_k \approx \exp(-A)v, \tag{1.2}$$

where k refers to the number of steps (iterations) needed to construct y_k. We interpret the vector y as a value of a vector function $y(t)$ at $t = 1$ such that

$$y'(t) = -Ay(t), \quad y(0) = v. \tag{1.3}$$

The exact solution of this initial-value problem (IVP) is given by

$$y(t) = \exp(-tA)v.$$

Assuming now that there is a vector function $y_k(t)$ such that $y_k(1) = y_k$, we define the residual for $y_k(t) \approx y(t)$ as

$$r_k(t) \equiv -Ay_k(t) - y_k'(t). \tag{1.4}$$

*Department of Applied Mathematics, University of Twente, P.O. Box 217, 7500 AE Enschede, the Netherlands, mbotchev@na-net.ornl.gov.

TABLE 1.1

The linear system and matrix exponential residuals. In both cases the sought after vector is $f(A)v$, with either $f(x) = 1/x$ or $f(x) = \exp(-x)$. The error is defined as the exact solution minus the approximate solution.

$f(x)$	$1/x$	$\exp(-x)$
exact solution y	$y = A^{-1}v$	define $y(t) = \exp(-tA)v$, set $y := y(1)$
residual equation	$Ay = v$	$\begin{cases} y'(t) = -Ay(t) \\ y(0) = v \end{cases}$
residual for $y_k \approx y$	$r_k = v - Ay_k$	$r_k(t) = -Ay_k(t) - y_k'(t)$
mapping error $\epsilon_k \to$ residual r_k	$r_k = A\epsilon_k$	$\begin{cases} r_k(t) = \epsilon_k'(t) + A\epsilon_k(t) \\ \epsilon_k(0) = 0 \end{cases}$
perturbed problem (backward stability)	$Ay_k = v - r_k$	$\begin{cases} y_k'(t) = -Ay_k(t) - r_k(t) \\ y_k(0) = v \end{cases}$

The key point in our residual concept is that $y = \exp(-A)v$ is seen not as a problem on its own but rather as the exact solution formula for the problem (1.3). The latter provides the equation where the approximate solution is substituted to yield the residual. We illustrate this in Table 1.1 where the introduced matrix exponential residual is compared against the conventional residual for a linear system $Ay = v$. Note that, as can be seen in the Table, the approximate solution satisfies a perturbed IVP where the perturbation is the residual. Thus, the introduced residual can be seen as a backward error (see Section 4 for residual-based error estimates). If one is interested in computing the matrix exponential $\exp(-A)$ itself then the residual can be defined with respect to the matrix IVP

$$X'(t) = -AX(t), \quad X(0) = I,$$

with the exact solution $X(t) = \exp(-tA)$.

The contribution of this paper is twofold. First, it turns out that the residual (1.4) can efficiently be computed within several iterative methods for matrix exponential evaluation. In this paper, we show how this can be done in several popular Krylov subspace and Chebyshev polynomial methods for computing $\exp(-A)v$. Second, we show how the residual notion leads to new algorithms to compute the matrix exponential. Two basic Richardson-like iterations are proposed and discussed. When combined with Krylov subspace methods, one of them can be seen an efficient way to restart the Krylov subspace methods. Note that our approach for the matrix exponential residual can readily be extended to the sine and cosine matrix functions (see the conclusion section).

The equivalence between the problems (1.2) and (1.3) has been widely used in numerical literature and computations (see e.g. the very first formula in [19] or [12, Section 10.1]). Moreover, methods for solving (1.2) are applied to (1.3) (for instance, exponential time integrators [15, 16]) and vice versa [19, Section 4]. In [27], van den Eshof and Hochbruck represent the error $\epsilon_k(t) \equiv y(t) - y_k(t)$ as the solution of the IVP $\epsilon_k'(t) = -A\epsilon_k(t) + r_k(t)$, $\epsilon_k(0) = 0$ and obtain an explicit, non-computable expression for $\epsilon_k(t)$. This allows them to justify a stopping criterion for their shift-and-invert Lanczos algorithm, based on the stagnation of the approximations. However, neither

in [27] nor anywhere else in the literature, the exponential residual (1.4) seems to be recognized as such.

The paper is organized as follows. Section 2 is devoted to the matrix exponential residual within Krylov subspace methods. In Section 3 we show how the Chebyshev iterations can be modified to adopt the residual control. Section 4 presents some simple residual-based error estimates. Richardson iteration for the matrix exponential is the topic of Section 5. Numerical experiments are discussed in Section 6 and conclusions are drawn in the last section.

Throughout the paper, unless reported otherwise, $\|\cdot\|$ denotes the Euclidean vector 2-norm or the corresponding induced matrix norm.

2. Matrix exponential residual in Krylov subspace methods. The Krylov subspace methods have become an important tool for computing matrix functions (see e.g. [28, 3, 17, 8, 23, 4, 14, 5, 15]). For $A \in \mathbb{R}^{n\times n}$ and $v \in \mathbb{R}^n$ given, Arnoldi process yield, after k steps, vectors $v_1, \ldots, v_{k+1} \in \mathbb{R}^n$ which are orthonormal in exact arithmetic and span the Krylov subspace $\mathcal{K}_k(v, Av, \ldots, A^{k-1}v)$ (see e.g. [10, 24, 29]). If $A = A^*$ Lanczos process is usually used instead of Arnoldi. Together with the basis vectors v_j, Arnoldi or Lanczos process delivers an upper-Hessenberg matrix $\underline{H}_k \in \mathbb{R}^{(k+1)\times k}$, such that the following Arnoldi/Lanczos relation holds [10, 24, 29]:

$$\begin{aligned} &AV_k = V_{k+1}\underline{H}_k, \quad \text{or, equivalently,} \\ &AV_k = V_k H_k + h_{k+1,k} v_{k+1} e_k^T, \end{aligned} \tag{2.1}$$

where $V_s \in \mathbb{R}^{n\times s}$ has columns $v_1, \ldots, v_s$, $H_k \in \mathbb{R}^{k\times k}$ is the matrix $\underline{H}_k$ with the skipped last row $(0, \cdots, 0, h_{k+1,k})$ and $e_k = (0, \cdots, 0, 1)^T \in \mathbb{R}^k$. The first basis vector v_1 is the normalized vector v: $v_1 = v/\|v\|$.

2.1. Ritz-Galerkin approximation. An approximation y_k to the matrix exponential $y = \exp(-A)v$ is usually computed as $y_k(1)$, with

$$y_k(t) = V_k \exp(-tH_k)(\beta e_1), \tag{2.2}$$

where $\beta = \|v\|$ and $e_1 = (1, 0, \cdots, 0) \in \mathbb{R}^k$. An important property of the Krylov subspace is its scaling invariance: application of the Arnoldi process to tA results in the upper-Hessenberg matrix of the form $t\underline{H}_k$ and the basis vectors $v_1, \ldots, v_{k+1}$ independent of t. It is convenient for us to write

$$y_k(t) = V_k u_k(t), \quad u_k(t) \equiv \exp(-tH_k)(\beta e_1), \tag{2.2$'$}$$

with $u_k(t) : \mathbb{R} \to \mathbb{R}^k$ being the solution of the projected IVP

$$u_k'(t) = -H_k u_k(t), \quad u_k(0) = \beta e_1. \tag{2.3}$$

The residual notion (1.4) allows us to see the approximation (2.2) as the Ritz-Galerkin approximation: the residual vector $r_k(t)$ is orthogonal, for any $t \geqslant 0$, to the search space $\operatorname{span}(v_1, \ldots, v_k)$:

$$V_k^* r_k(t) = V_k^*(-Ay_k(t) - y_k'(t)) = V_k^*(-AV_k u_k(t) - V_k u_k'(t)) = -H_k u_k(t) - u_k'(t) = 0, \tag{2.4}$$

where we used the relation $V_k^* A V_k = H_k$, which follows from (2.1).

Note that the Krylov subspace approximation (2.2) satisfies the initial condition $y_k(0) = v$ by construction:

$$y_k(0) = V_k u_k(0) = V_k(\beta e_1) = \beta v_1 = v.$$

Thus, there is no danger that residual $r_k(t) = -Ay_k(t) - y_k(t)'$ is small in norm for some $y_k(t)$ approaching a solution of the ODE system $y' = Ay$ with another initial data. The following simple Lemma provides an explicit expression for the residual:

LEMMA 2.1. *Let $y_k(t) \approx y(t) = \exp(-tA)v$ be the Krylov subspace approximation given by (2.2). Then for any $t \geqslant 0$ the residual $r_k(t)$ for $y_k(t) \approx y(t)$ is*

$$\begin{aligned} r_k(t) &= -\beta h_{k+1,k} e_k^T \exp(-tH_k) e_1 v_{k+1}, \\ \|r_k(t)\| &= |\beta h_{k+1,k} e_k^T \exp(-tH_k) e_1| = |h_{k+1,k}[u_k(t)]_k|, \end{aligned}$$

where $[u_k(t)]_k$ is the last entry of the vector function $u_k(t)$ defined in (2.2′).

Proof. It follows from (2.2) that $y_k'(t) = -V_k H_k \exp(-tH_k)(\beta e_1)$. From the Arnoldi relation (2.1) we have

$$Ay_k(t) = AV_k \exp(-tH_k)(\beta e_1) = (V_k H_k + h_{k+1,k} v_{k+1} e_k^T) \exp(-tH_k)(\beta e_1),$$

which yields the result:

$$r_k(t) = -Ay_k(t) - y_k'(t) = -h_{k+1,k} v_{k+1} e_k^T \exp(-tH_k)(\beta e_1).$$

□

The residual $r_k(t)$ turns out to be closely related to the so-called *generalized residual* $\rho_k(t)$ [15]. Following [15] (see also [23]), we can write

$$\begin{aligned} y_k(t) = \beta V_k \exp(-tH_k) e_1 \qquad &= \frac{1}{2\pi i} \oint_\Gamma e^\lambda (\lambda I + tH_k)^{-1} \beta e_1 \mathrm{d}\lambda, \\ y(t) = \exp(-tA) v \qquad &= \frac{1}{2\pi i} \oint_\Gamma e^\lambda (\lambda I + tA)^{-1} v \mathrm{d}\lambda, \end{aligned}$$

where Γ is a closed contour in $\mathbb{C}$ encircling the spectrum of A. Thus, $y_k(t)$ is an approximation to $y(t)$ where the resolvent inverse $(\lambda I + tA)^{-1} v$ is approximated by k steps of the fully orthogonal method (FOM):

$$\epsilon_k = y(t) - y_k(t) = \frac{1}{2\pi i} \oint_\Gamma e^\lambda \mathrm{error}_k^{\mathrm{FOM}} \mathrm{d}\lambda.$$

Since the FOM error is unknown, the authors of [15] replace it by the known FOM residual, which is $\beta(-th_{k+1,k}) v_{k+1} e_k^T (\lambda I + tH_k)^{-1} e_1$. This leads to the generalized residual

$$\begin{aligned} \rho_k(t) &\equiv \frac{1}{2\pi i} \oint_\Gamma e^\lambda \beta(-th_{k+1,k}) v_{k+1} e_k^T (\lambda I + tH_k)^{-1} e_1 \mathrm{d}\lambda \\ &= -\beta t h_{k+1,k} e_k^T \exp(-tH_k) e_1 \, v_{k+1}, \end{aligned}$$

which coincides, up to a factor t, with our matrix exponential residual $r_k(t)$.

2.2. Shift-and-invert Arnoldi/Lanczos approximations. In the shift-and-invert (SaI) Arnoldi/Lanczos approximations [20, 27] the Krylov subspace is built up with respect to the matrix $(I + \gamma A)^{-1}$, with $\gamma > 0$ being a parameter, so that the Krylov basis matrix $V_{k+1} \in \mathbb{R}^{n \times (k+1)}$ and an upper-Hessenberg matrix $\underline{\tilde{H}}_k \in \mathbb{R}^{(k+1) \times k}$ are built such that (cf. (2.1))

$$\begin{aligned} &(I + \gamma A)^{-1} V_k = V_{k+1} \underline{\tilde{H}}_k, \quad \text{or, equivalently,} \\ &(I + \gamma A)^{-1} V_k = V_k \tilde{H}_k + \tilde{h}_{k+1,k} v_{k+1} e_k^T, \end{aligned} \tag{2.5}$$

where $\tilde{H}_k \in \mathbb{R}^{k\times k}$ is formed by the first k rows of $\underline{\tilde{H}}_k$. The approximation $y_k(t) \approx \exp(-tA)v$ is then computed as given by (2.2), with H_k defined as [27]

$$H_k = \frac{1}{\gamma}(\tilde{H}_k^{-1} - I). \tag{2.6}$$

Relation (2.5) can be rewritten as (cf. formula (4.1) in [27])

$$AV_k = V_k H_k - \frac{\tilde{h}_{k+1,k}}{\gamma}(I+\gamma A)v_{k+1}e_k^T\tilde{H}_k^{-1}, \tag{2.7}$$

which leads to the following lemma:

LEMMA 2.2. *Let $y_k(t) \approx y(t) = \exp(-tA)v$ be the SaI Krylov subspace approximation (2.2), with H_k defined in (2.6). Then for any $t \geqslant 0$ the residual $r_k(t)$ for $y_k(t) \approx y(t)$ is*

$$\begin{aligned} r(t) &= \beta\,\frac{\tilde{h}_{k+1,k}}{\gamma}\,e_k^T\tilde{H}_k^{-1}\exp(-tH_k)e_1(I+\gamma A)v_{k+1},\\ \|r(t)\| &\leqslant |\beta|\left|\frac{\tilde{h}_{k+1,k}}{\gamma}\right||e_k^T\tilde{H}_k^{-1}\exp(-tH_k)e_1|(1+\gamma\|A\|). \end{aligned}$$

Proof. The proof is very similar to that of Lemma 2.1. Instead of the conventional Arnoldi relation (2.1), relation (2.7) should be used. □

2.3. Error estimation in Krylov subpace methods. If $y_k(t)$ is a Krylov subspace approximation to $y(t) = \exp(-tA)v$ then the error function $\epsilon_k(t) \equiv y(t) - y_k(t)$ satisfies the IVP

$$\epsilon_k'(t) = -A\epsilon_k(t) + r_k(t), \quad \epsilon_k(0) = 0. \tag{2.8}$$

To estimate the error, this equation can approximately be solved by any suitable time integration scheme, for example, by Krylov exponential schemes as discussed e.g. in [8, Section 4] or [15]. The time integration process for solving (2.8) can further be optimized to take into account that the residual function $r_k(t)$ depends on time as $r_k(t) = \psi_k(t)v_{k+1}$ with $v_{k+1} = \text{const}(t)$ and $\psi_k(t)$ being a scalar function of t (see Lemma 2.1):

$$\psi_k(t) \equiv -\beta h_{k+1,k}e_k^T\exp(-tH_k)e_1.$$

Van den Eshof and Hochbruck [27] propose to get an error estimate by replacing in $\epsilon_k(t) \equiv y(t) - y_k(t)$ the exact solution $y(t)$ with the same continued Krylov process approximation $y_{k+m}(t)$:

$$\begin{aligned} \epsilon_k(t) &\approx y_{k+m}(t) - y_k(t) = V_{k+m}u_{k+m}(t) - V_k u_k(t) = V_{k+m}\tilde{\epsilon}_k(t),\\ \|\epsilon_k(t)\| &\approx \|\tilde{\epsilon}_k(t)\| = \|u_{k+m}(t) - \tilde{u}_k(t)\|, \end{aligned} \tag{2.9}$$

where

$$V_k u_k(t) = V_{k+m}\tilde{u}_k(t), \quad \tilde{u}_k(t) = [\,(u_k(t))^T, \underbrace{0,\ldots,0}_{m}]^T$$

and $u_k(t)$ and $u_{k+m}(t)$ are the solutions of the projected IVP (2.3) obtained with respectively k and $k+m$ Krylov steps. It is not difficult to see that in this case

$\tilde{\epsilon}_k(t) \equiv u_{k+m}(t) - \tilde{u}_k(t)$ is the Galerkin solution of (2.8) with respect to the subspace colspanV_{k+m}. Indeed, we have

$$\begin{aligned} y'_{k+m} &= -Ay_{k+m} - r_{k+m}(t), & y_{k+m}(t) &= V_{k+m}u_{k+m}(t), \\ y'_k &= -Ay_k - r_k(t), & y_k(t) &= V_{k+m}\tilde{u}_k(t). \end{aligned}$$

Subtracting y'_k from y'_{k+m} and multiplying the result from the left by V^*_{k+m} we obtain

$$(u_{k+m}(t)-\tilde{u}_k(t))' = -H_{k+m}(u_{k+m}(t)-\tilde{u}_k(t))+V^*_{k+m}r_k(t), \quad V^*_{k+m}r_k(t) = \psi_k(t)e_{k+1},$$

and we arrive at the projected IVP

$$\tilde{\epsilon}'_k(t) = -H_{k+m}\tilde{\epsilon}_k(t) + \psi_k(t)e_{k+1}, \tag{2.10}$$

where e_{k+1} is the $(k+1)$th basis vector in $\mathbb{R}^{k+m}$. This shows that error estimation by the same continued Krylov process is a better option than solving the correction equation (2.8) by a new Krylov process: the latter would mean that we neglect the built up subspace. In fact, solving IVP (2.8) by another process and then correcting the approximate solution $y_k(t)$ can be seen as a restarting. We will further explore this approach in Section 5.

3. Matrix exponential residual for Chebyshev approximations. A well-known method to compute $y_m(t) \approx \exp(-tA)v$ is based on the Chebyshev polynomial expansion (see for instance [26, 22]):

$$y_m(t) = P_m(-tA)v = \left[\sum_{k=1}^{m} c_k T_k(-tA) + \frac{c_0}{2}I\right] v. \tag{3.1}$$

Here we assume that the matrix tA can be transformed to have its eigenvalues within the interval $[-1,1] \subset \mathbb{R}$ (for example, A can be a Hermitian or a skew-Hermitian matrix). Here, T_k are the Chebyshev polynomials of the first kind, whose actions on the given vector v can be computed by the Chebyshev recursion

$$T_0(x) = 1, \quad T_1(x) = x, \quad T_{k+1}(x) = 2xT_k(x) - T_{k-1}(x), \quad k = 1, 2, \ldots, \tag{3.2}$$

and the coefficients c_k can be computed, for a large M, as

$$c_k = \frac{2}{M}\sum_{j=1}^{M} \exp(\cos\theta_j))\cos(k\theta_j), \quad k = 0, 1, \ldots, m, \quad \theta_j = \frac{\pi(j-\frac{1}{2})}{M}, \tag{3.3}$$

which means interpolating $\exp(x)$ at the Chebyshev polynomial roots (see e.g. [22, Section 3.2.3]). This Chebyshev polynomial approximation is used for evaluating different matrix functions in [2].

The recursive algorithm (3.1)–(3.3) can be modified to provide, along with $y_m(t)$, vectors $y'_m(t)$ and $Ay_m(t)$, so that the exponential residual $r_m(t) \equiv -Ay_m(t) - y'_m(t)$ can be controlled in the course of the iterations. To do this, we use the well-known relations

$$T'_k(x) = kU_{k-1}(x), \tag{3.4}$$

$$xT_k(x) = \frac{1}{2}(T_{k+1}(x) + T_{k-1}(x)), \tag{3.5}$$

$$xU_k(x) = \frac{1}{2}(U_{k+1}(x) + U_{k-1}(x)), \tag{3.6}$$

$$T_k(x) = \frac{1}{2}(U_k(x) - U_{k-2}(x)), \tag{3.7}$$

where $k = 1, 2, \ldots$ and U_k are the Chebyshev polynomials of the second kind:

$$U_0(x) = 1, \quad U_1(x) = 2x, \quad U_{k+1}(x) = 2xU_k(x) - U_{k-1}(x), \quad k = 1, 2, \ldots. \tag{3.8}$$

For (3.7) to hold true for $k = 1$ we denote $U_{-1}(x) = 0$. From (3.1) and (3.4),(3.6) it follows that

$$\begin{aligned} y'_m(t) &= \left[\sum_{k=1}^{m} \frac{c_k}{t}(-tA)T'_k(-tA)\right] v \\ &= \left[\sum_{k=1}^{m} \frac{c_k k}{2t}(U_k(-tA) + U_{k-2}(-tA))\right] v, \quad m = 1, 2, \ldots. \end{aligned} \tag{3.9}$$

Similarly, from (3.1), (3.5) and (3.7), we obtain

$$\begin{aligned} -Ay_m(t) &= \left[\sum_{k=1}^{m} \frac{c_k}{2t}(T_{k+1}(-tA) + T_{k-1}(-tA)) - \frac{c_0}{2}A\right] v \\ &= \left[\sum_{k=1}^{m} \frac{c_k}{2t}(U_{k+1}(-tA) - U_{k-3}(-tA)) - \frac{c_0}{2}A\right] v, \quad m = 1, 2, \ldots, \end{aligned} \tag{3.10}$$

where we define $U_{-2}(x) = -1$.

The obtained recursions can be used to formulate an algorithm for computing $y_m(t) \approx \exp(-tA)v$ which control the residual $r_m(t) = -Ay_m(t) - y'_m(t)$, see Figure 3.1. Just as the original Chebyshev recursion algorithm for matrix exponential, it requires one action of the matrix A per iteration. To be able to control the residual, more vectors have to be stored than in the conventional algorithm: eight instead of four.

4. Residual-based error estimates. By definition of the residual (1.4), the approximate solution $y_k(t) \approx \exp(-tA)v$ is the exact solution of the problem

$$y'_k(t) = -Ay_k(t) - r_k(t), \quad y(0) = v, \tag{4.1}$$

which is a perturbation of the original problem (1.3). Therefore the residual $r_k(t)$ can be seen as the backward error for $y_k(t)$. From (4.1) and (1.3) it is easy to see that the error $\epsilon_k(t)$ satisfies the initial-value problem

$$\epsilon'_k(t) = -A\epsilon_k(t) + r_k(t), \quad \epsilon_k(0) = 0, \tag{4.2}$$

with the exact solution

$$\epsilon_k(t) = \int_0^t \exp((s-t)A)r_k(s)\mathrm{d}s. \tag{4.3}$$

This formula can be used to obtain error bounds in terms of the norms of the matrix exponential and the residual [30]:

LEMMA 4.1. *Let $|A|$ denote a matrix whose entries are absolute values of the entries of A. Let $\bar{r}_k(t) : \mathbb{R} \to \mathbb{R}^n$ be a vector-function with the entries $[\bar{r}_k(t)]_i$ defined as*

$$[\bar{r}_k(t)]_i = \max_{s\in[0,t]} |[r_k(s)]_i|, \quad i = 1, \ldots, n.$$

$u_{-2} := -v, \quad u_{-1} := 0, \quad u_0 := v, \quad u_1 := (-2 * t) * (A * v)$
compute c_0
$y := (0.5 * c_0) * u_0, \quad y' := 0, \quad \texttt{minusAy} := (c_0/t/4) * u_1$

for $k = 1, \ldots, N_{\max}$
$u_2 := 2 * (-t) * (A * u_1) - u_0$
compute c_k
$y := y + (c_k/2) * (u_1 - u_{-1})$
$y' := y' + (c_k * k/2/t) * (u_1 + u_{-1})$
$\texttt{minusAy} := \texttt{minusAy} + (c_k/4/t) * (u_2 - u_{-2})$
$u_{-2} := u_{-1}$
$u_1 := u_0$
$u_0 := u_1$
$u_1 := u_2$
$\texttt{resnorm} := \|\texttt{minusAy} - y'\|$
if $\texttt{resnorm} < \texttt{toler}$
return
end
end

FIG. 3.1. *Chebyshev expansion algorithm to compute the vector* $y_{N_{\max}}(t) \approx \exp(-tA)v$. *The input parameters are* $A \in \mathbb{R}^{n\times n}$, $v \in \mathbb{R}^n$, $t > 0$ *and* $\texttt{toler} > 0$. *It is assumed that for the eigenvalues* λ *of* tA *holds* $-1 \leqslant \lambda \leqslant 1$.

It holds for any $t \geqslant 0$

$$\|\epsilon_k(t)\|_* \leqslant \||t\varphi(-tA)|\,\bar{r}_k(t)\|_* \leqslant \||t\varphi(-tA)|\|_*\|\bar{r}_k(t)\|_*, \tag{4.4}$$

with $\|\cdot\|_*$ *being any consistent matrix (vector) norm and* $\varphi(x) = (\exp(x) - 1)/x$. *Note that, for any* $B \in \mathbb{R}^{n\times n}$, $\||\varphi(B)|\|_* = \|\varphi(B)\|_*$ *for the 1- and the maximum norms.*

Proof. For simplicity of notation, throughout the proof we omit the subindex $\cdot_k$ and write $\epsilon_k(t) = \epsilon(t)$ and $r_k(t) = r(t)$. The entry i of $\epsilon(t)$ can be bounded as

$$\sum_{j=1}^{n} \int_0^t [\exp((s-t)A)]_{ij} \mathrm{d}s\, \check{r}_j \leqslant [\epsilon(t)]_i \leqslant \sum_{j=1}^{n} \int_0^t [\exp((s-t)A)]_{ij} \mathrm{d}s\, \hat{r}_j, \tag{4.5}$$

$$\check{r}_j = \min_{s\in[0,t]} [r(s)]_j, \quad \hat{r}_j = \max_{s\in[0,t]} [r(s)]_j, \tag{4.6}$$

where $[\exp((s-t)A)]_{ij}$ is the (i,j) entry of $\exp((s-t)A)$. Integrating the left and right parts of this inequality, we obtain

$$[(I - \exp(-tA))A^{-1}\check{r}]_i \leqslant [\epsilon(t)]_i \leqslant [(I - \exp(-tA))A^{-1}\hat{r}]_i, \tag{4.7}$$

$$|\epsilon(t)|_i \leqslant \max\{|(I - \exp(-tA))A^{-1}\check{r}|_i, |(I - \exp(-tA))A^{-1}\hat{r}|_i\}. \tag{4.8}$$

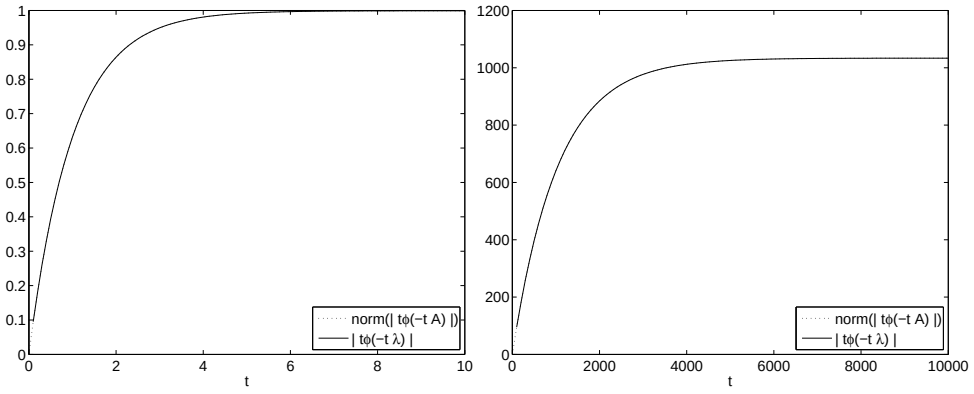

FIG. 4.1. *The values of* $\|\|t\varphi(-tA)\|\|$ *(dotted) and* $|t\varphi(-t\lambda_{\min})|$ *(solid) against* t *for* $A = \text{tridiag}(-1,3,-1)$ *(left) and* $A = \text{tridiag}(-1,2,-1)$ *(right). In both cases* $A \in \mathbb{R}^{100\times 100}$. *Note different scale of the* t *axes. The horizontal asymptote is* $1/\lambda_{\min}$.

We have

$$|(I-\exp(-tA))A^{-1}\check{r}|_i \leqslant \sum_{i=1}^{n} |(I-\exp(-tA))A^{-1}|_{ij}|\check{r}|_j,$$

and a similar estimate holds for $|(I-\exp(-tA))A^{-1}\hat{r}|_i$. Therefore

$$|\epsilon(t)|_i \leqslant \sum_{j=1}^{n} |(I-\exp(-tA))A^{-1}|_{ij} \underbrace{\max\{|\check{r}|_j, |\hat{r}|_j\}}_{[\bar{r}]_j} = [|\underbrace{(I-\exp(-tA))A^{-1}}_{-t\varphi(-tA)}|\,\bar{r}]_i,$$

which ends the proof. □

The estimates provided by the last lemma can further be specified if more information on A is available. For example if A is symmetric positive definite with eigenvalues lying in the interval $[\lambda_{\min}, \lambda_{\max}]$ then (in the 2-norm)

$$\|\|t\varphi(-tA)|\|\| = \|\|t\varphi(-t\Lambda)|\|\| \leqslant \frac{1-\exp(-t\lambda_{\max})}{\lambda_{\min}} \leqslant \frac{1}{\lambda_{\min}}$$

with Λ being a diagonal matrix with the eigenvalues of A as its entries. Thus, the magnitude of the smallest eigenvalues of A is decisive in this case. To illustrate this, in Figure 4.1 we have plotted the values $\|\|t\varphi(-tA)|\|\|$ against t for two different matrices A of size 100×100. Note that before evaluating the matrix exponential we can always shift A such that its smallest eigenvalue exceeds one:

$$\exp(-tA)v = \exp(-t(A+I))\exp(t)v.$$

Hence, $\|\|t\varphi(-tA)|\|\| \leqslant 1/\lambda_{\min} \leqslant 1$ and we have in (4.4)

$$\|\epsilon_k(t)\| \leqslant \|\bar{r}_k(t)\|, \quad t \geqslant 0.$$

5. Richardson iteration for the matrix exponential. The notion of the residual allows us to introduce a Richardson method for the matrix exponential.

5.1. Preconditioned Richardson iteration. Consider the preconditioned Richardson iterative method

$$x_{k+1} = x_k + M^{-1} r_k \tag{5.1}$$

for solving a linear system $Ax = b$, with the preconditioner $M \approx A$ and residual $r_k = b - Ax_k$. Note that $M^{-1}r_k$ is an approximation to the unknown error $A^{-1}r_k$. By analogy with (5.1), we formulate the Richardson method for the matrix exponential as

$$y_{k+1}(t) = y_k(t) + \tilde{\epsilon}_k(t), \tag{5.2}$$

where $\tilde{\epsilon}_k \approx \epsilon_k$ is the approximate solution of the IVP (2.8). One option, which we follow here, is to choose a suitable $M \approx A$ and let $\tilde{\epsilon}_k$ be the solution of the IVP

$$\tilde{\epsilon}_k'(t) = -M\tilde{\epsilon}_k(t) + r_k(t), \quad \tilde{\epsilon}_k(0) = 0. \tag{5.3}$$

Just as when solving linear systems, M has to compromise between the approximation quality $M \approx A$ and the ease of solving (5.3).

Residual $r_k(t)$ of the Richardson iteration (5.2),(5.3) can be shown to satisfy the following recursion. From (5.2) and (5.3) we have

$$-y_{k+1}'(t) = -y_k'(t) + M\tilde{\epsilon}_k(t) - r_k(t).$$

Subtracting relation $Ay_{k+1}(t) = Ay_k(t) + A\tilde{\epsilon}_k(t)$ from this equation, we get

$$r_{k+1}(t) = -y_k(t) + M\tilde{\epsilon}_k(t) - r_k(t) - Ay_k(t) - A\tilde{\epsilon}_k(t) = (M - A)\tilde{\epsilon}_k(t). \tag{5.4}$$

Taking into account that

$$\tilde{\epsilon}_k(t) = \int_0^t \exp((s-t)M) r_k(s) \mathrm{d}s,$$

we obtain

$$r_{k+1}(t) = (M - A)\tilde{\epsilon}_k(t) = (M - A) \int_0^t \exp((s-t)M) r_k(s) \mathrm{d}s. \tag{5.5}$$

Using relation (4.4), we arrive at the following result:

LEMMA 5.1. *Let $|A|$ and $\bar{r}_k(t)$ be as defined in Lemma 4.1. The residual $r_k(t) = -y_k'(t) - Ay_k(t)$ in the exponential Richardson method* (5.2) *satisfies for any $t \geqslant 0$*

$$\begin{aligned} \|r_{k+1}(t)\|_* &\leqslant \| |t(M-A)\varphi(-tM)|\, \bar{r}_k(t)\|_* \\ &\leqslant \| |t(M-A)\varphi(-tM)| \|_* \|\bar{r}_k(t)\|_*, \\ \textit{so that} \quad \max_{s\in[0,t]} \|r_{k+1}(s)\|_* &\leqslant \max_{s\in[0,t]} \| |s(M-A)\varphi(-sM)| \|_* \max_{s\in[0,t]} \|r_k(s)\|_*, \end{aligned}$$

with $\|\cdot\|_*$ being any consistent matrix (vector) norm and $\varphi(x) = (\exp(x) - 1)/x$.

Proof. The proof follows the lines of the proof of Lemma 4.1. □

The estimate provided by the lemma shows that, at least for some matrices A and M and not too large $t \geqslant 0$, the exponential Richardson iteration converges faster than the Richardson iteration for linear system solution. Indeed, since

$$t(M - A)\varphi(-tM) = (M - A)M^{-1}(I - \exp(-tM)),$$

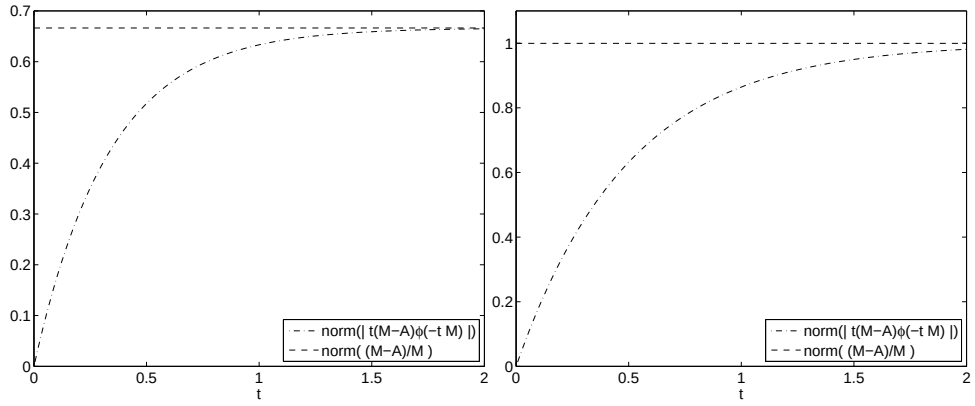

FIG. 5.1. *Upper bounds for residual reduction in Richardson iteration for the linear system (dashed) and matrix exponential (dash-dotted).* $A = \mathrm{tridiag}(-1, 3, -1)$ *(left) and* $A = \mathrm{tridiag}(-1, 2, -1)$ *(right). In both cases* $A \in \mathbb{R}^{100\times 100}$*,* $M = \mathrm{diag}(A)$*.*

the upper bounds for the residual reduction are

$$\text{linear system Richardson:} \quad \frac{\|r_{k+1}\|_*}{\|r_k\|_*} \leqslant \|(M-A)M^{-1}\|_*,$$

$$\text{exponential Richardson:} \quad \frac{\|r_{k+1}(t)\|_*}{\|\bar{r}_k(t)\|_*} \leqslant \|\,|(M-A)M^{-1}(I-\exp(-tM))|\,\|_*,$$

with $t \geqslant 0$ in the second inequality. For general matrices A and M it is hard to prove that

$$\|\,|(M-A)M^{-1}(I-\exp(-tM))|\,\| \leqslant \|(M-A)M^{-1}\|, \qquad t \geqslant 0.$$

This inequality holds in the 2-norm, for instance, if A is an M-matrix and M is its diagonal part (in this case the matrices $M-A$, M^{-1} and $I-\exp(-tM)$ are elementwise nonnegative and we can get rid of the absolute value sign). As can be seen in Figure 5.1, exponential Richardson can converge reasonably well even when $\|(M-A)M^{-1}\|$ is hopelessly close to one.

An important practical issue hindering the use of the exponential Richardson iteration is the necessity to store the vectors $r_k(t)$ for different t. To achieve a good accuracy, sufficiently many samples of $r_k(t)$ have to be stored. Our limited experience indicates the exponential Richardson iteration can be of interest if the accuracy requirements are relatively low, say upto 10^{-5}. In the experiments described in Section 6.3 just 20 samples were sufficient to get the error below tolerance 10^{-4} with $n = 10^4$.

5.2. Krylov restarting via Richardson iteration. In the exponential Richardson iteration (5.2) the error $\tilde{\epsilon}_k(t)$ does not have to satisfy (5.3), which is just one possible choice for $\tilde{\epsilon}_k(t)$. Another choice is to take $\tilde{\epsilon}_k(t)$ to be the Krylov approximate solution of the IVP

$$\tilde{\epsilon}_k' = -A\tilde{\epsilon}_k + r_k(t), \quad \tilde{\epsilon}_k(0) = 0. \tag{5.6}$$

If the approximate solution $y_k(t)$ is also obtained by a Krylov process, then the Richardson iteration (5.2),(5.6) can be seen as a restarted Arnoldi/Lanczos method

for computing $\exp(-tA)v$. Indeed, assume, the IVP (5.6) is solved approximately by m Arnoldi or Lanczos steps, so that the next Richardson approximation is

$$y_{k+m}(t) = y_k(t) + \tilde{\epsilon}_k(t). \tag{5.7}$$

Assume $y_k(t)$ is the Krylov or SaI Krylov approximation to $\exp(-tA)v$, given by (2.2), (2.1) or by (2.2), (2.6), respectively. To derive an expression for $r_{k+m}(t)$, we first notice that

$$r_k(t) = \psi_k(t) w_k, \quad \psi_k : \mathbb{R} \to \mathbb{R}, \quad w_k = \text{const}(t) \in \mathbb{R}^n, \tag{5.8}$$

with a scalar function $\psi_k(t)$ and a constant vector w_k. These are given by

$$\psi_k(t) = -\beta h_{k+1,k} e_k^T \exp(-tH_k) e_1, \quad w_k = v_{k+1}$$

for the regular Krylov approximation (see Lemma 2.1) and by

$$\psi_k(t) = \beta \, \frac{\tilde{h}_{k+1,k}}{\gamma} \, e_k^T \tilde{H}_k^{-1} \exp(-tH_k) e_1 (I + \gamma A) v_{k+1}, \quad w_k = (I + \gamma A) v_{k+1}$$

for the shift-and-invert Krylov approximation (see Lemma 2.2). The error

$$\epsilon_k(t) = y(t) - y_k(t) = \int_0^t \exp((s-t)A) r_k(s) \mathrm{d}s = \int_0^t \psi_k(s) \exp((s-t)A) w_k \mathrm{d}s$$

is approximated by the m step Krylov solution $\tilde{\epsilon}_k(t)$ of (5.6):

$$\begin{aligned} \tilde{\epsilon}_k(t) &= \int_0^t \psi_k(s) \widehat{V}_m \exp((s-t)\widehat{H}_m) \|w_k\| e_1 \mathrm{d}s \\ &= \widehat{V}_m \underbrace{\int_0^t \exp((s-t)\widehat{H}_m) \psi_k(s) \|w_k\| e_1 \mathrm{d}s}_{\widehat{u}(t)}, \end{aligned} \tag{5.9}$$

where $e_1 = (1, 0, \ldots, 0)^T \in \mathbb{R}^m$ and $\widehat{V}_m \in \mathbb{R}^{n \times m}$ and $\widehat{H}_m \in \mathbb{R}^{m \times m}$ result from m steps of the Arnoldi/Lanczos process for the matrix A and the vector w_k. It is not difficult to see that $\widehat{u}(t)$ is the solution of the IVP

$$\widehat{u}'(t) = -\widehat{H}_m \widehat{u}(t) + \psi_k(t) \|w_k\| e_1, \quad \widehat{u}(0) = 0. \tag{5.10}$$

From (5.9) and (5.10), we have

$$\begin{aligned} r_{k+m}(t) &= -y'_{k+m}(t) - A y_{k+m}(t) = -y'_k(t) - \tilde{\epsilon}'_k(t) - A y_k(t) - A\tilde{\epsilon}_k(t) \\ &= r_k(t) - \widehat{V}_m \widehat{u}'(t) - A\widehat{V}_m \widehat{u}(t) = r_k(t) - \widehat{V}_m(-\widehat{H}_m \widehat{u}(t) + \psi_k(t)\|w_k\| e_1) - A\widehat{V}_m \widehat{u}(t) \\ &= r_k(t) + \widehat{V}_m \widehat{H}_m \widehat{u}(t) - \underbrace{\psi_k(t)\|w_k\| \widehat{V} e_1}_{r_k(t)} - A\widehat{V}_m \widehat{u}(t) = (\widehat{V}_m \widehat{H}_m - A\widehat{V}_m)\widehat{u}(t). \end{aligned} \tag{5.11}$$

If $\widehat{V}$ and $\widehat{H}$ result from the conventional Arnoldi/Lanczos process, then (cf. (2.1)) $\widehat{V}_m \widehat{H}_m - A\widehat{V}_m = -\widehat{h}_{m+1,m} \widehat{v}_{m+1} e_m^T$, so that

$$r_{k+m}(t) = -\widehat{h}_{m+1,m} [\widehat{u}(t)]_m \widehat{v}_{m+1}, \tag{5.12}$$

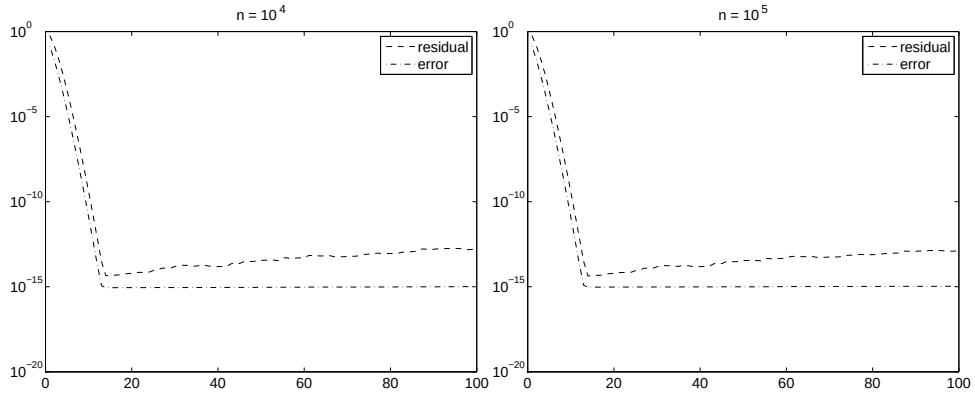

FIG. 6.1. *The residual and the true error norms in the Chebyshev algorithm to compute* $y_m \approx \exp(-A)v$ *against iteration number* m. *Normal matrix. Left: the matrix size* $n = 10^4$, *right:* $n = 10^5$.

with $[\widehat{u}(t)]_m$ being the last component of $\widehat{u}(t)$. If $\widehat{V}$ and $\widehat{H}$ are obtained with the SaI Arnoldi/Lanczos process then (cf. (2.7))

$$\widehat{V}_m \widehat{H}_m - A\widehat{V}_m = \widehat{\widehat{h}}_{m+1,m}\gamma^{-1}(I+\gamma A)\widehat{v}_{m+1}e_m^T \widehat{\widetilde{H}}_m^{-1},$$

with all the quantities defined by (2.5),(2.6) (replacing the subindices $\cdot_k$ by $\cdot_m$ and adding the $\widehat{\cdot}$ sign). This yields

$$r_{k+m}(t) = \widehat{\widehat{h}}_{m+1,m}\gamma^{-1}[\widehat{\widetilde{H}}_m^{-1}\widehat{u}(t)]_m(I+\gamma A)\widehat{v}_{m+1} \tag{5.13}$$

From (5.12) and (5.13) we see that the residual $r_{k+m}(t)$ is, just as in (5.8), a scalar time-dependent function times a constant vector. This shows that the derivation for $r_{k+m}(t)$ remains valid for all Krylov-Richardson iterations (formally, we can set $y_k(t) := y_{k+m}(t)$ and repeat the iteration (5.7)).

6. Numerical experiments. All the numerical experiments have been carried out with Matlab on a Linux PC.

6.1. Residual in Chebyshev iteration. The following simple tests are carried out for the Chebyshev iterative method with incorporated residual control (see Figure 3.1). We compute $\exp(-A)v$ where $v \in \mathbb{R}^n$ is a random vector with mean zero and standard deviation one. In the first test the matrix $A \in \mathbb{R}^{n\times n}$ is diagonal with diagonal entries evenly distributed between -1 and 1. In the second test, we fill the first superdiagonal of A with ones, so that A becomes ill-conditioned. The plots of the error and residual norms are presented in Figures 6.1 and 6.2.

As can be expected, for nonnormal A the error is accumulated during the iteration, so that it is important to know when to stop the iteration. Too many iterations may yield a completely wrong answer. The residual sharply reflects the error behavior, thus providing a reliable error estimate.

6.2. A convection-diffusion problem. In the next several numerical experiments the matrix A is taken to be the standard five-point central difference discretization of the following convection-diffusion operator acting on functions defined in the

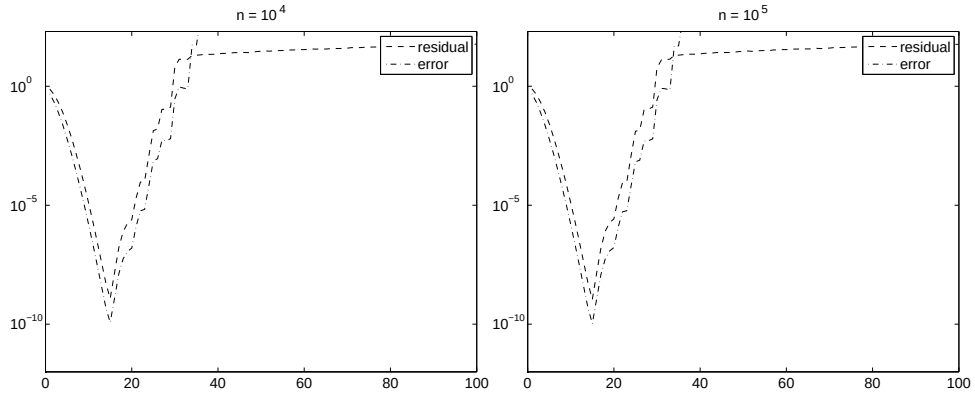

FIG. 6.2. *The residual and the true error norms in the Chebyshev algorithm to compute* $y_m \approx \exp(-A)v$ *against iteration number* m. *Nonnormal matrix. Left: the matrix size* $n = 10^4$, *right:* $n = 10^5$.

domain $(x, y) \in [0, 1]^2$:

$$L[u] = -(D_1 u_x)_x - (D_2 u_y)_y + Pe(v_1 u_x + v_2 u_y),$$

$$D_1(x,y) = \begin{cases} 10^3, & (x,y) \in [0.25, 0.75]^2, \\ 1, & \text{otherwise}, \end{cases} \qquad D_2(x,y) = \frac{1}{2} D_1(x,y),$$

$$v_1(x,y) = x + y, \qquad v_2(x,y) = x - y.$$

To guarantee that the convection terms yield exactly a skew-symmetric matrix, before discretizing, we rewrite the convection terms in the form [18]:

$$v_1 u_x + v_2 u_y = \frac{1}{2}(v_1 u_x + v_2 u_y) + \frac{1}{2}((v_1 u)_x + (v_2 u)_y).$$

This is possible since the velocity field (v_1, v_2) is divergence free. The operator L is set to satisfy the homogeneous Dirichlet boundary conditions. In all cases the discretization is carried out on a 102×102 uniform mesh, producing a $n \times n$ matrix A of size $n = 10^4$. The Peclet number has got values $Pe = 0$ (no convection, $A = A^T$), $Pe = 10$ ($\|A - A^T\|_1 / \|A + A^T\| \approx 3.3 \cdot 10^{-5}$) and $Pe = 100$ ($\|A - A^T\|_1 / \|A + A^T\| \approx 3.3 \cdot 10^{-4}$).

6.3. Exponential Richardson iteration. In this section we apply the exponential Richardson iteration (5.2), (5.3) to compute the vector $\exp(-A)v$ for the convection-diffusion matrices A described in Section 6.2. The vector v is taken to be the normalized vector with equal entries. As discussed above, to be able to update the residual and solve the IVP (5.3), we need to store the values of $r_k(t)$ for different t spanning the time interval of interest. Too few samples may result in an accuracy loss in the interpolation stage. On the other hand, it can be prohibitively expensive to store many samples. Therefore, in its current form, the method does not seem to suit if a high accuracy is needed. On the other hand, it turns out that with a relatively small number of samples (≈ 20) a moderate accuracy upto 10^{-5} can be reached.

We organize the computations in the method as follows. The residual vector function $r_k(t)$ is stored as 20 samples. At each iteration, the IVP (5.3) is solved by the Matlab `ode15s` ODE solver, the values of the right hand side function $-M\tilde{\epsilon}_k(t) + r_k(t)$

TABLE 6.1

Performance of the exponential Richardson method for the convection-diffusion test problem, `toler` $= 10^{-4}$, $M = \mathrm{tridiag}(A)$

	total flops$\times n$	matvecs A / steps	LU $I + \alpha M$	lin.systems $I + \alpha M$	matvecs M
		$Pe = 0$			
EXPOKIT	4590	918 matvecs	—	—	—
exp. Richardson	2192	8 steps	24	176	192
		$Pe = 10$			
EXPOKIT	4590	918 matvecs	—	—	—
exp. Richardson	2202	8 steps	29	176	192
		$Pe = 100$			
EXPOKIT	4590	918 matvecs	—	—	—
exp. Richardson	2492	9 steps	31	200	

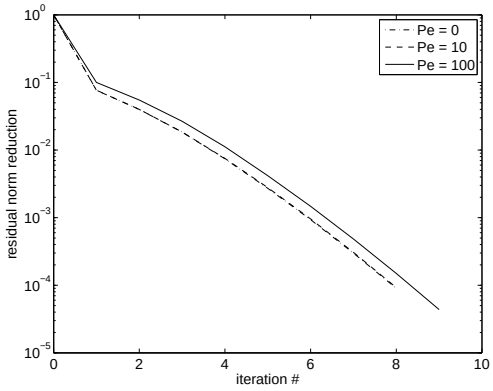

FIG. 6.3. *Convergence history of the exponential Richardson iteration*

are interpolated using the stored samples. The `ode15s` solver is run with tolerances determined by the final required accuracy and produces the solution $\tilde{\epsilon}_k(t)$ in the form of its twenty samples. Then, the solution and residual are updated according to (5.2) and (5.5), respectively.

We have chosen M to be the tridiagonal part $\mathrm{tridiag}(A)$ of the matrix A. Table 6.1 and Figure 6.3 contains results of the test runs. Except the Richardson method, as a reference we use the EXPOKIT code [25] with the maximal Krylov dimension 100. It is rather difficult to compare the total computational work of the EXPOKIT and Richardson method. We restrict ourselves to the matrix-vector part of the work. In the Richardson method this work consists of the matrix-vector multiplication (matvec) with $M - A$ in (5.5) and the work done by the `ode15s` solver. The matvec with bidiagonal $M - A$ costs about $3n$ flops times 20 samples, in total $60n$ flops[1]. The linear algebra work in `ode15s` is essentially tridiagonal matvecs, LU factorizations and back/forward substitutions with (possibly shifted and scaled) M. According to [10, Section 4.3.1], tridiagonal LU factorization, back- and forward substitution require each about $2n$ flops. A matvec with tridiagonal M is $5n$ flops. Thus, in overal

[1] We use definition of flop from [10, Section 1.2.4].

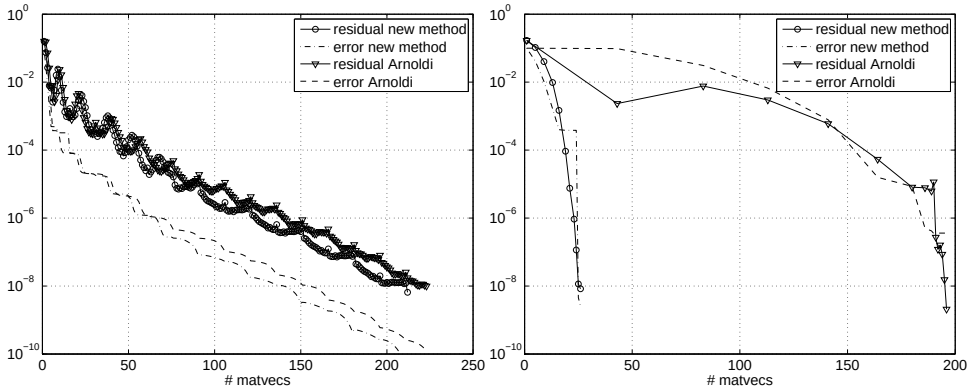

FIG. 6.4. *Convergence plots of the Arnoldi/Lanczos and the new Krylov-Richardson methods. Left: Pe = 0, restart is 15, right: Pe = 100, SaI strategy. The plateau in the Krylov-Richardson error on the right plot is because the solution is not accurately updated until the restart or the end of the iterations.*

exponential Richardson costs $60n$ flops times the number of iterations plus $2n$ flops times the number of LU factorizations and back/forward substitutions plus $5n$ flops times the total number of the ODE solver steps. The matvec work in EXPOKIT consists of matvecs with pentadiagonal A, which is about $9n$ flops.

From Table 6.1 we see that exponential Richardson is approximately twice cheaper than EXPOKIT. As expected from the convergence estimates, exponential Richardson converges much faster than the conventional Richardson iteration for solving a linear system $Ax = v$ would do. For these A and v, 8-9 iterations of the conventional Richardson would only give a residual reduction by a factor of ≈ 0.99.

6.4. Experiments with Krylov-Richardson iteration. In this section we present some numerical comparisons of the Krylov-Richardson method with conventional Arnoldi/Lanczos methods for evaluating the matrix exponential. We again compute the vector $y = \exp(-A)v$ for the convection-diffusion matrices A and v being a vector of a unit 2-norm with equal entries. The iterations are stopped as soon as the residual gets below 10^{-8}.

Except the classical Arnoldi/Lanczos method [8, 23, 4, 14], we have been tested the SaI method of Van den Eshof and Hochbruck [27]. We have implemented the method exactly as described in their paper, with a single modification. In particularly, in all the tests the shift parameter γ is set to $0.1t_{\text{end}}$ and the relaxed stopping criterion strategy for the inner iterative SaI solvers is employed. The only thing we have changed is the stopping criterion of the outer Krylov process. To be able to exactly compare the computational work, we replaced the stopping criterion of Van den Eshof and Hochbruck (based on iterant stagnation) by our residual stopping criterion (see Lemma 2.2). Without this modification, the results are biased towards our new schemes since the stopping criterion of Van den Eshof and Hochbruck tends to be slightly pessimistic (the iterations are usually stopped later than with the residual criterion). Note that the relaxed strategy for the inner SaI solver is then also based on the residual norm and not on the error estimate.

Since the Krylov-Richardson method is essentially a restarting technique, it has to be compared with another existing restarting technique. To compare with, we have chosen the restarting method described in [21]. This choice is motivated by the

fact that the method from [21] is algorithmically very close to our Krylov-Richardson method. In fact, the only essential difference is handling of the projected problem. In the method [21] the projected matrix H_k built up in every restart is appended to a larger matrix $\widetilde{H}_{*+k}$. There, the projected matrices from each restart are accumulated. Thus, if 10 restarts of 20 steps are done, we have to compute the matrix exponential of a 200×200 matrix. In our method, the projected matrices are not accumulated, at every restart we deal with a 20×20 matrix. The price to pay is, however, the solution of the small IVP (5.10).

In our implementation, at each Krylov-Richardson iteration the IVP (5.10) is solved by the `ode15s` ODE solver from Matlab. To save the computational work, it is essential that the solver most of the time is called with a relaxed tolerance parameter (in our code we set tolerance to the 1% of the current residual norm). This is sufficient to accurately estimate the residual. Only when the actual solution update takes place (see formula (5.7)) we solve the projected IVP to a full accuracy. This happens at the end of each restart or when the stopping criterion is satisfied.

Since the residual time dependence in Krylov-Richardson is given by a scalar function, little storage is needed for the look up table. Based on the required accuracy, the `ode15s` solver automatically determines how many samples need to be stored (in our experiments this usually did not exceed 300).

Note that a number of efficient restarting strategies for have recently been developed [1, 11, 6]. These, too, can surely be combined with our residual Richardson strategy. We plan to explore this possibility in the future.

Table 6.2 and Figures 6.4 contain the results of the test runs. The first observation we make is that the convergence of the Krylov-Richardson iteration is essentially the same as of the classical Arnoldi/Lanczos method. This is not influenced by the restart value or by the SaI strategy. Theoretically, this is to be expected: the former method applies Krylov for the φ function, the latter for the exponential; for both functions similar convergence estimates hold, though slightly more favorable for the φ function [14].

If no SaI strategy is applied, the gain we have with the Krylov-Richardson is twofold. First, a projected problem of a much smaller size has to be solved. Second, we have some freedom of choosing the initial guess vector (in standard Arnoldi/Lanczos we always must to start with v). This freedom is not complete, since the residual of the initial guess has to have the scalar dependence on time. Several variants for choosing the initial guess vector exist, and we will explore these possibilities in the future.

A significant gain in total computational work is achieved when Krylov-Richardson is combined with the SaI strategy. The gain is due the reduction in the number of the inner iterations (the number of outer iterative steps is approximately the same). Currently, we do not completely understand this behavior. Apparently, the Krylov subspace vectors built in the Krylov-Richardson method constitute more favorable right-hand sides for the inner SaI solvers to converge. It is rather difficult to analyze this phenomenon, and we will try to do this in the near future.

7. Concluding remarks and an outlook to further research. The proposed residual notion appears to provide a reliable stopping criterion in the iterative methods for computing the matrix exponential. This is confirmed by the numerical tests and analysis. Furthermore, the residual concept seems to set up a whole framework for a new class of the methods for evaluating the matrix exponential. Some basic methods of this class are proposed in this paper. Many new research questions arise. One of

TABLE 6.2

Results of the test runs of the Krylov-Richardson, conventional Arnoldi/Lanczos and EXPOKIT methods

	SaI solver	total matvecs
	$Pe = 0$	
EXPOKIT, restart 15	—	1190
Lanczos, restart 15	—	229 ($15 \times 15 + 3$ steps)
new method, restart 15	—	212 ($15 \times 14 + 1$ steps)
EXPOKIT, restart 100	—	1020
Lanczos, restart 100	—	137 ($100 + 36$ steps)
new method, restart 100	—	150 ($100 + 49$ steps)
Lanczos, SaI	CG[a]	292 (11 steps)
new method, SaI	CG[a]	20 (9 steps)
Lanczos, SaI	sparse LU	9 (8 steps)
new method, SaI	sparse LU	10 (9 steps)
	$Pe = 10$	
EXPOKIT, restart 15	—	1173
Arnoldi, restart 15	—	229 ($15 \times 14 + 11$ steps)
new method, restart 15	—	220 ($15 \times 14 + 9$ steps)
EXPOKIT, restart 100	—	1020
Arnoldi, restart 100	—	142 ($100 + 41$ steps)
new method, restart 100	—	155 ($100 + 54$ steps)
Arnoldi, SaI	GMRES[b]	186 (10 steps)
new method, SaI	GMRES[b]	20 (9 steps)
Arnoldi, SaI	sparse LU	9 (8 steps)
new method, SaI	sparse LU	10 (9 steps)
	$Pe = 100$	
EXPOKIT, restart 15	—	1343
Arnoldi, restart 15	—	242 ($15 \times 16 + 1$ steps)
new method, restart 15	—	247 ($15 \times 16 + 6$ steps)
EXPOKIT, restart 100	—	1020
Arnoldi, restart 100	—	166 ($100 + 65$ steps)
new method, restart 100	—	188 ($100 + 87$ steps)
Arnoldi, SaI	GMRES[b]	196 (15 steps)
new method, SaI	GMRES[b]	26 (10 steps)
Arnoldi, SaI	sparse LU	11 (10 steps)
new method, SaI	sparse LU	11 (10 steps)

[a] CG with the IC(0) preconditioner
[b] GMRES(100) with the ILU(0) preconditioner

them is a comprehensive convergence analysis of the new exponential Richardson and Krylov-Richardson methods. Another interesting research direction is development of other residual-based iterative methods.

Finally, one may ask whether the proposed residual notion can not be extended to other matrix functions. This is possible once a residual equation can be identified, i.e. an equation such that the matrix function satisfies this equation (see Table 1.1). For example, if we are interested in computing the vector

$$u = \cos(A)v,$$

for given $A \in \mathbb{R}^{n\times n}$ and $v \in \mathbb{R}^n$, then we may consider a vector function $u(t) = \cos(tA)v$ which is a solution of the IVP

$$u''(t) = -A^2u, \quad u(0) = v, \quad u'(0) = 0.$$

Thus, for an approximate solution $u_k(t) \approx u(t)$ satisfying the initial conditions, residual can be introduced as

$$r_k(t) \equiv -A^2u_k(t) - u_k''(t).$$

Acknowledgments. The author would like to thank Jan Verwer, Marlis Hochbruck and Julien Langou for fruitful discussions.

REFERENCES

[1] M. Afanasjew, M. Eiermann, O. G. Ernst, and S. Güttel. Implementation of a restarted Krylov subspace method for the evaluation of matrix functions. *Linear Algebra Appl.*, 429:2293–2314, 2008.

[2] M. Benzi and N. Razouk. Decay bounds and $O(n)$ algorithms for approximating functions of sparse matrices. *Electron. Trans. Numer. Anal.*, 28:16–39, 2007.

[3] V. L. Druskin and L. A. Knizhnerman. Two polynomial methods of calculating functions of symmetric matrices. *U.S.S.R. Comput. Maths. Math. Phys.*, 29(6):112–121, 1989.

[4] V. L. Druskin and L. A. Knizhnerman. Krylov subspace approximations of eigenpairs and matrix functions in exact and computer arithmetic. *Numer. Lin. Alg. Appl.*, 2:205–217, 1995.

[5] V. L. Druskin and L. A. Knizhnerman. Extended Krylov subspaces: approximation of the matrix square root and related functions. *SIAM J. Matrix Anal. Appl.*, 19(3):755–771 (electronic), 1998.

[6] M. Eiermann, O. G. Ernst, and S. Güttel. Deflated restarting for matrix functions. Submitted, October 2009.

[7] A. Frommer and V. Simoncini. Matrix functions. In W. H. A. Schilders, H. A. van der Vorst, and J. Rommes, editors, *Model Order Reduction: Theory, Research Aspects and Applications*, pages 275–304. Springer, 2008.

[8] E. Gallopoulos and Y. Saad. Efficient solution of parabolic equations by Krylov approximation methods. *SIAM J. Sci. Statist. Comput.*, 13(5):1236–1264, 1992.

[9] F. R. Gantmacher. *The theory of matrices. Vol. 1.* AMS Chelsea Publishing, Providence, RI, 1998. Translated from the Russian by K. A. Hirsch, Reprint of the 1959 translation.

[10] G. H. Golub and C. F. Van Loan. *Matrix Computations*. The Johns Hopkins University Press, Baltimore and London, third edition, 1996.

[11] S. Güttel. *Rational Krylov Methods for Operator Functions*. PhD thesis, Technischen Universität Bergakademie Freiberg, March 2010. `www.guettel.com`.

[12] N. J. Higham. *Functions of Matrices: Theory and Computation*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2008.

[13] N. J. Higham and A. H. Al-Mohy. Computing matrix functions. *Acta Numer.*, 19:159–208, 2010.

[14] M. Hochbruck and C. Lubich. On Krylov subspace approximations to the matrix exponential operator. *SIAM J. Numer. Anal.*, 34(5):1911–1925, Oct. 1997.

[15] M. Hochbruck, C. Lubich, and H. Selhofer. Exponential integrators for large systems of differential equations. *SIAM J. Sci. Comput.*, 19(5):1552–1574, 1998.

[16] M. Hochbruck and A. Ostermann. Exponential integrators. *Acta Numer.*, 19:209–286, 2010.

[17] L. A. Knizhnerman. Calculation of functions of unsymmetric matrices using Arnoldi's method. *U.S.S.R. Comput. Maths. Math. Phys.*, 31(1):1–9, 1991.

[18] L. A. Krukier. Implicit difference schemes and an iterative method for solving them for a certain class of systems of quasi-linear equations. *Sov. Math.*, 23(7):43–55, 1979. Translation from Izv. Vyssh. Uchebn. Zaved., Mat. 1979, No. 7(206), 41–52 (1979).

[19] C. Moler and C. Van Loan. Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later. *SIAM Rev.*, 45(1):3–49, 2003.

[20] I. Moret and P. Novati. RD rational approximations of the matrix exponential. *BIT*, 44:595–615, 2004.

[21] J. Niehoff. *Projektionsverfahren zur Approximation von Matrixfunktionen mit Anwendungen auf die Implementierung exponentieller Integratoren*. PhD thesis, Mathematisch-Naturwissenschaftlichen Fakultät der Heinrich-Heine-Universität Düsseldorf, December 2006.

[22] V. S. Ryaben′kii and S. V. Tsynkov. *A theoretical introduction to Numerical Analysis*. Chapman & Hall/CRC, Boca Raton, FL, 2007.

[23] Y. Saad. Analysis of some Krylov subspace approximations to the matrix exponential operator. *SIAM J. Numer. Anal.*, 29(1):209–228, 1992.

[24] Y. Saad. Iterative methods for sparse linear systems. Book out of print, 2000. `www-users.cs.umn.edu/~saad/books.html`.

[25] R. B. Sidje. EXPOKIT. A software package for computing matrix exponentials. *ACM Trans. Math. Softw.*, 24(1):130–156, 1998. `www.maths.uq.edu.au/expokit/`.

[26] H. Tal-Ezer. Spectral methods in time for parabolic problems. *SIAM J. Numer. Anal.*, 26(1):1–11, 1989.

[27] J. van den Eshof and M. Hochbruck. Preconditioning Lanczos approximations to the matrix exponential. *SIAM J. Sci. Comput.*, 27(4):1438–1457, 2006.

[28] H. A. van der Vorst. An iterative solution method for solving $f(A)x = b$, using using Krylov subspace information obtained for the symmetric positive definite matrix A. *J. Comput. Appl. Math.*, 18:249–263, 1987.

[29] H. A. van der Vorst. *Iterative Krylov methods for large linear systems*. Cambridge University Press, 2003.

[30] J. L. M. van Dorsselaer, J. F. B. M. Kraaijevanger, and M. N. Spijker. Linear stability analysis in the numerical solution of initial value problems. In *Acta numerica, 1993*, Acta Numer., pages 199–237. Cambridge Univ. Press, Cambridge, 1993.