ODE-based Double-preconditioning for Solving Linear Systems $A^\alpha x = b$ and $f(A)x = b$

Xavier Antoine, Emmanuel Lorin

ODE-BASED DOUBLE-PRECONDITIONING FOR SOLVING LINEAR SYSTEMS $A^\alpha X = B$ AND $F(A)X = B$

XAVIER ANTOINE* AND EMMANUEL LORIN†

Abstract. This paper is devoted to the computation of the solution to *fractional linear algebraic systems* using a differential-based strategy to evaluate matrix-vector products $A^\alpha x$, with $\alpha \in \mathbb{R}_+^*$. More specifically, we propose ODE-based preconditioners for efficiently solving fractional linear systems in combination with traditional sparse linear system preconditioners. Different types of preconditioners are derived (Jacobi, Incomplete LU, Padé) and numerically compared. The extension to systems $f(A)x = b$ is finally considered.

1. Introduction. In the present paper, we are interested in the numerical solution to *Fractional Linear Systems* (FLS) of the form

$$A^\alpha x = b, \tag{1}$$

for some $\alpha \in \mathbb{R}_+^*$, where $A \in \mathbb{C}^{n\times n}$ and $b \in \mathbb{C}^n$ are given, and extensions to equations $f(A)x = b$. We assume that A is a diagonalizable matrix in $\mathbb{R}^+$ or $\mathbb{C}\backslash\mathbb{R}_-$ (to avoid the singularity line of $\arg(z)$) which is sparse or is a (possibly random) perturbation of a sparse matrix, with eigenvalues having positive real parts. Let us recall that A^α is defined by the Cauchy integral formula (see e.g. Theorem 6.2.28 from [10])

$$A^\alpha = (2\pi \mathtt{i})^{-1} A^k \int_{\Gamma_A} z^{\alpha-k}(zI - A)^{-1} dz\,, \tag{2}$$

for $k \geqslant 0$, where Γ_A is a closed contour in the complex plane *enclosing the spectrum of the matrix A*. Even if A is a sparse diagonalizable matrix, A^α is generally dense. The study of this class of systems is largely motivated by the numerical approximation of Fractional Partial Differential Equations (FPDEs), including fractional diffusion and fractional Schrödinger equations, which represents currently a very active research area in computational science [3, 4, 7, 11, 12, 14]. In particular, solving such FPDEs by some real-space approximations leads to FLS. For invertible matrices, we formally have $x = A^{-\alpha}b$.

For $\alpha \in \mathbb{R}$, the n-dimensional differential system

$$y'(\tau) = \alpha(A - I)\big(I + \tau(A - I)\big)^{-1} y(\tau), \qquad y(0) = b\,, \tag{3}$$

admits a solution $y(\tau) = \big(I + \tau(A - I)\big)^\alpha b$ such that $y(1) = A^\alpha b$, where I is the identity matrix in $\mathbb{R}^{n\times n}$ (see [5, 6, 8, 9]). By replacing α by $-\alpha$ in (3), we can compute $x = A^{-\alpha}b$ through an ODE. This approach is an alternative to the techniques based on the Cauchy integral representation (2) as studied e.g. in [1, 2, 3, 11, 14]. Since A^α is unknown, computing $A^\alpha z$, for any vector z (for example in an iterative process), usually requires a non-trivial algorithm such as (3). In this paper, we are most particularly interested in the derivation of efficient and low-cost preconditioners

*INSTITUT ELIE CARTAN DE LORRAINE, UNIVERSITÉ DE LORRAINE, UMR 7502, INRIA NANCY-GRAND EST, F-54506 VANDOEUVRE-LÈS-NANCY CEDEX, FRANCE (XAVIER.ANTOINE@UNIV-LORRAINE.FR).

†SCHOOL OF MATHEMATICS AND STATISTICS, CARLETON UNIVERSITY, OTTAWA, CANADA (ELORIN@MATH.CARLETON.CA); CENTRE DE RECHERCHES MATHÉMATIQUES, UNIVERSITÉ DE MONTRÉAL, MONTRÉAL, CANADA.

P for solving FLS as (1) by a Krylov subspace iterative solver. More precisely, we propose to build some matrices $P \in \mathbb{C}^{n\times n}$ such that

$$PA^{\alpha}x = Pb\,, \tag{4}$$

with $\kappa(A^{\alpha}) \gg \kappa(PA^{\alpha})$, where $\kappa(B)$ denotes the condition number of a given matrix B. This preconditioning procedure will naturally provide a much faster convergence rate of Krylov-type solvers applied to (1). As the construction of products of power matrices with vectors that also include the application of the preconditioner will be performed by using appproximate solutions to (3), preconditioners P will be referred to as ODE-solver preconditioners. It is important to remark that, unlike usual linear system preconditioners, the application of these ODE-solver preconditioners will not be realized by only successively applying P and A^{α} but will rather be computed efficiently by estimating $PA^{\alpha}v$ based on approximate solutions to (1), for any vector v in $\mathbb{C}^{n\times n}$. Different types of ODE-preconditioners (Jacobi or ILU-based preconditioners) will be introduced later and their efficiency will be compared. Let us also notice that the computation of approximate solutions to (1) also requires the numerical solution of several linear systems by using preconditioned GMRES algorithms[1], resulting in an *ODE-based double-preconditioning algorithm*. More precisely, P arising in (4) is a fractional algebraic linear system preconditioner (first preconditioning level), which itself requires the use of preconditioners (second preconditioning level) for solving intermediate linear systems. Since the main objective of this paper is to derive some ODE-solver preconditioners to efficiently solve (1), we propose here to focus on a detailed study for the Crank-Nicolson scheme with uniform time discretization, even if a fourth-order Runge-Kutta method will also be considered to illustrate some possible extensions of our approach. Indeed, we insist on the fact that the purpose of this paper is not to optimize the ODE-solver for a given matrix, but rather to develop a general preconditioning methodology when using the ODE-solver approach. For example, optimized high-order schemes and adaptive time-stepping strategies could also be implemented in combination with an ODE-solver preconditioner but are out of the scope of the present paper and require further studies. Even if the developments are mainly focused on the case of FLS, we also generalize the proposed methodology to systems of the form

$$f(A)x = b, \qquad A \in \mathbb{C}^{n\times n},\, b \in \mathbb{C}^{n}\,, \tag{5}$$

for some analytic functions f. We show that the method can be expected to work well, but many improvements still need to be investigated in details for completeness.

The paper is organized as follows. In Section 2, we consider a direct Crank-Nicolson discretization of the ODE-solver, i.e. without any preconditioner. In Section 3, we present a methodology based on the clustering of the spectrum of the matrix A thanks to linear system preconditioners (called *ODE-solver preconditioners*), leading to a strong reduction of the number of time steps for solving (3). Different types of preconditioners such as the scaling approach, Jacobi method, Incomplete LU (ILU) or Padé-type preconditioners are proposed and compared. The computational complexity is also addressed. In Section 4, we develop an extension to some more general systems of the form $f(A)x = b$. We finally conclude in Section 5.

2. Direct Crank-Nicolson (DCN) method for $A^{\alpha}x = b$. In this section, we derive a Crank-Nicolson algorithm for approximating the solution to (1), by using

[1] In the numerical experiments, we will use the standard GMRES algorithm from `matlab`.

(3) but *without any ODE-solver preconditioner*. This approach will be referred to as the Direct Crank-Nicolson (DCN) method. Let us introduce $\{\tau_k\}_{0\leqslant k\leqslant K_A}$ as a finite sequence of uniformly sampled discrete time steps such that $\tau_k = k\Delta\tau_A$, with $0 \leqslant k \leqslant K_A$, setting $\tau_{K_A} = 1$. For $\alpha \in \mathbb{R}_+^*$, we propose to approximate $A^{-\alpha}b$ iteratively by discretizing (3) with the second-order *Direct Crank-Nicolson* (DCN) scheme

$$\begin{aligned} y_{k+1}^{(A)} &= y_k^{(A)} - \frac{\Delta\tau_A}{2}\alpha(A-I)\big(I+\tau_k(A-I)\big)^{-1}y_k^{(A)} \\ &\quad -\frac{\Delta\tau_A}{2}\alpha(A-I)\big(I+\tau_{k+1}(A-I)\big)^{-1}y_{k+1}^{(A)}\,, \end{aligned}$$

where $y_k^{(A)}$ is an approximation of $y(\tau_k)$. This leads to the 3-steps DCN algorithm

Step 1. Compute $z_k^{(A)}$ solution to $\big(I+\tau_k(A-I)\big)z_k^{(A)} = y_k^{(A)}$.

Step 2. Set $w_k^{(A)} := y_k^{(A)} - \dfrac{\Delta\tau_A}{2}\alpha(I-A)z_k^{(A)}$, and calculate $\omega_{k+1}^{(A)}$ such that

$$\big(I+\tau_{k+1}(A-I)\big)\omega_{k+1}^{(A)} = w_k^{(A)}\,.$$

Step 3. Deduce $y_{k+1}^{(A)}$ solution to

$$(I+\tau_{k+1+\frac{\alpha}{2}}(A-I))y_{k+1}^{(A)} = \omega_{k+1}^{(A)}\,.$$

At each iteration k, $0 \leqslant k \leqslant K_A - 1$, we need to solve three linear systems using traditional *iterative solvers combined with preconditioners*. At the final time $\tau_{K_A} = 1$, we obtain $y_{K_A}^{(A)} \approx A^{-\alpha}b$. More precisely, there exists a positive constant $C > 0$ such that we have the error bound

$$\|y_{K_A}^{(A)} - A^{-\alpha}b\|_2 \leqslant C\Delta\tau_A^2\,,$$

where $\|v\|_2 = \big(\sum_{j=1}^n |v_j|^2\big)^{1/2}$, for any vector $v \in \mathbb{C}^n$. We do not detail the use of other numerical ODE-solvers, which can directly be applied. Nevertheless, for completeness, we consider an explicit Runge-Kutta-4 (RK4) method in Subsection 3.1 as an alternative solver for (3) to show what could be expected when increasing the order of accuracy in time.

3. Preconditioned Crank-Nicolson (PCN) method for $A^\alpha x = b$. In this Section, we propose some preconditioning techniques for solving the *ODE-based* algorithm. The first approach is actually a simple scaling, introduced in Subsection 3.1. Analytical arguments to justify why the double-preconditioning approach is expected to be efficient are developed in Subsection 3.2. In Subsection 3.3, we consider some more elaborated preconditioners (Jacobi/ILU) for efficiently solving (3), i.e. to reduce the number of time steps to compute the solution for the ODE-solver with a prescribed accuracy. An analysis of the computational cost of the procedure is detailed in Subsection 3.4. Finally, a Padé's approximants preconditioning technique is introduced for comparison in Subsection 3.5. In all the subsections, we propose some simulations to fairly illustrate the behavior of the different approaches.

3.1. Scaling ODE-preconditioning. We discuss now a *Preconditioned Crank-Nicolson* (PCN) method based on the scaling matrix $M = I/\|A\|_2$, where $\|A\|_2$ is the 2-norm of the matrix A. Let us assume here that A is a diagonally dominant matrix.

Since M is a scaling matrix, we trivially have the identity $A^{-\alpha} = M^{\alpha}(MA)^{-\alpha}$. We then compute the finite sequence $\{y_k^{(MA)}\}_{0\leqslant k\leqslant K_{MA}}$ of vector solutions, setting $\tau_{K_{MA}} = \tau_{K_A} = K_{MA}\Delta\tau_{MA} = 1$, following the PCN algorithm

Step 1. Compute $z_k^{(MA)}$ such that: $\big(I + \tau_k(MA - I)\big)z_k^{(MA)} = y_k^{(MA)}$.

Step 2. Define $w_k^{(MA)} := y_k^{(MA)} - \dfrac{\Delta\tau_{MA}}{2}\alpha(I-MA)z_k^{(MA)}$, and obtain $\omega_{k+1}^{(MA)}$ as the solution to the preconditioned linear system

$$\big(I + \tau_{k+1}(MA - I)\big)\omega_{k+1}^{(MA)} = w_k^{(MA)}.$$

Step 3. Deduce $y_{k+1}^{(MA)}$ solution to

$$(I + \tau_{k+1+\frac{\alpha}{2}}(MA - I))y_{k+1}^{(MA)} = \omega_{k+1}^{(MA)}.$$

Each of these linear systems is solved by using an *iterative method and an additional classical preconditioner*. At the final time $\tau_{K_{MA}}$, we get an estimate of $(MA)^{-\alpha}b$, and therefore we deduce $x = M^{\alpha}(MA)^{-\alpha}b$ through

$$y_{K_{MA}}^{(MA)} \quad \approx \quad M^{\alpha}(MA)^{-\alpha}b = A^{-\alpha}b,$$

where there exists a constant $C > 0$ such that: $\|y_{K_A}^{(MA)} - A^{-\alpha}b\|_2 \leqslant C\Delta\tau_{MA}^2$.

Experiment 1. Let us solve the system $A^{\alpha}x = b$, where $A = B + B^T + D \in \mathbb{R}^{n\times n}$, with $n = 400$. The matrix $B = \{b_{ij}\}_{1\leqslant ij\leqslant n}$ is such that $b_{ij} = \texttt{rand}(0,1)$, and $D = \{d_{ij}\}_{1\leqslant ij\leqslant n}$ is the diagonal matrix defined by $d_{ii} = 15 + \texttt{rand}(0,1)$, $1 \leqslant i \leqslant n$. The value of α is fixed to 0.5. The right-hand side b is given in $\mathbb{R}^n$. We report in Fig. 1 (Right) the convergence graph of $\|A^{-\alpha}b - A_h^{-\alpha}b\|_\infty$, where $A^{-\alpha}b$ is the exact solution to the linear system, and $A_h^{-\alpha}b$ (the index h refers to as an approximate value of $A^{-\alpha}$) is computed by solving (3) with a DCN scheme, DRK4 (Direct RK4) scheme, and scaled PCN and PRK4 (Preconditioned RK4) schemes, as described in Subsections (2) and (3.1). In Fig. 1 (Left), we report the spectrum of A and MA. These results show the efficiency of the scaling technique for accelerating the computation of $A^{-\alpha}b$. In particular, we notice the fourth- (resp. second-)order convergence of the RK4 (resp. CN) schemes. The scaling approach is more specifically interesting for efficiently evaluating $A^{-\alpha}b$, but not necessarily useful as a preconditioner as seen below.

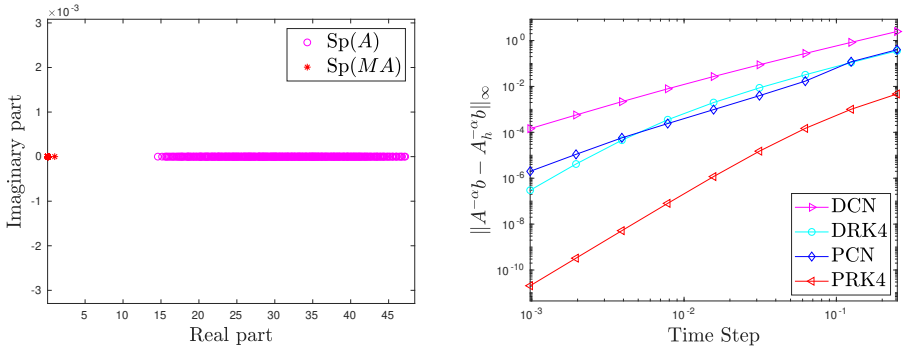

FIG. 1. **Experiment 1.** *(Left) Spectrum of A and MA. (Right) Convergence rate of the DCN/DRK4 and scaled PCN/PRK4 schemes.*

Experiment 1bis. This test is similar to the previous one except that $A = B + D \in \mathbb{R}^{n\times n}$ ($n = 400$), which has a complex spectrum with eigenvalues having positive real parts. The results are reported in Fig. 2, where we again observe an improvement of the convergence of the preconditioned ODE-solvers vs the non-preconditioned ones.

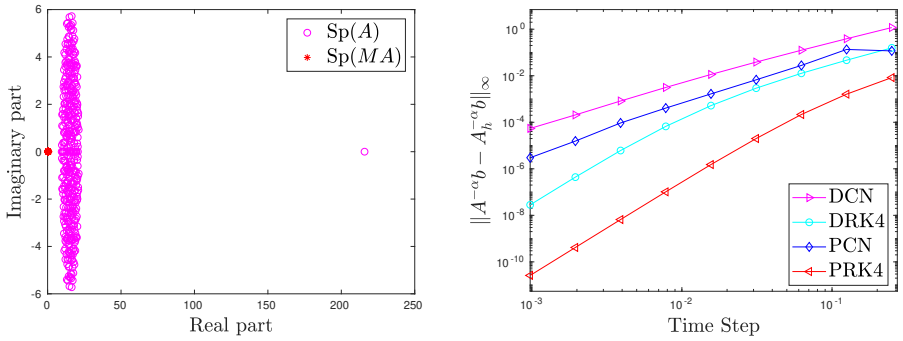

FIG. 2. **Experiment 1bis.** *(Left) Spectrum of A and MA. (Right) Convergence rate of the DCN/DRK4 and scaled PCN/PRK4 schemes.*

3.2. Justification of the ODE-solver preconditioning. To emphasize on the need to include a preconditioning strategy into the ODE-solver methodology, we develop an analysis based on error estimates explaining the improvement in term of convergence rate. More precisely, for a fixed number of basic operations, a PCN method leads to a more accurate evaluation of $A^{-\alpha}b$ compared to the DCN method. By construction, we naturally have $\rho(MA - I) \ll \rho(A - I)$, since $M \approx A^{-1}$. The matrix MA is therefore expected to have a spectrum localized around $1 \in \mathbb{C}$.

The following proposition holds.

PROPOSITION 3.1. *Let us consider the DCN and PCN algorithms described in Subsection 3.1, where we assume that the linear systems are solved exactly. Then, there exists a constant $c > 0$ such that*

$$\|y_{K_A}^{(A)} - A^{-\alpha}b\|_2 \quad \leqslant \quad c\Delta\tau_A^2 \frac{e^{\alpha(\rho(A)-1)} - 1}{\alpha}.$$

Moreover, we have the following estimates:

1. *If we assume that $\rho(MA - I) > 1$, then we have*

$$\|y_{K_{MA}}^{(MA)} - A^{-\alpha}b\|_2 \quad \leqslant \quad c\Delta\tau_{MA}^2 \frac{e^{\alpha(\rho(MA)-1)} - 1}{\alpha}.$$

.

2. *If $1 > \rho(MA - I) > \varepsilon$, for some $\varepsilon \in (0, 1)$, one gets*

$$\|y_{K_{MA}}^{(MA)} - A^{-\alpha}b\|_2 \quad \leqslant \quad c\Delta\tau_{MA}^2 \frac{e^{\alpha(1-\varepsilon)^{-1}} - 1}{\alpha}.$$

From this proposition, we expect a much better error estimate for PCN than for DCN since $\rho(A) \gg 1$ and $\rho(A) \gg \rho(MA)$ by construction of the preconditioner M.

Proof. Let us define the two functions

$$f_A(\alpha, \tau) = -\alpha(A - I)(I + \tau(A - I))^{-1},$$
$$f_{MA}(\alpha, \tau) = -\alpha(MA - I)(I + \tau(MA - I))^{-1}.$$

We then introduce $\rho(\cdot)$ as the spectral radius function and we define

$$\begin{array}{rcccl} \rho_{f_A}(\alpha) & = & \max\limits_{\tau\in[0,1]} \rho\big(f_A(\alpha,\tau)\big) & = & \alpha \max\limits_{\tau\in[0,1]} \rho\big((A-I)(I+\tau(A-I))^{-1}\big)\,, \\ \rho_{f_{MA}}(\alpha) & = & \max\limits_{\tau\in[0,1]} \rho\big(f_{MA}(\alpha,\tau)\big) & = & \alpha \max\limits_{\tau\in[0,1]} \rho\big((MA-I)(I+\tau(MA-I))^{-1}\big)\,. \end{array}$$

Assuming that the linear systems in DCN and PCN are solved exactly, then the overall 2-norm errors at time $T=1$ are such that

$$\|y_{K_A}^{(A)} - A^{-\alpha}b\|_2 \quad \leqslant \quad \mathsf{t}(\Delta\tau_A)\frac{e^{\rho_{f_A}(\alpha)}-1}{\rho_{f_A}(\alpha)},$$

while

$$\|y_{K_{MA}}^{(MA)} - A^{-\alpha}b\|_2 \quad \leqslant \quad \mathsf{t}(\Delta\tau_{MA})\frac{e^{\rho_{f_{MA}}(\alpha)}-1}{\rho_{f_{MA}}(\alpha)},$$

where $\mathsf{t}(\Delta\tau) \to_{\Delta\tau\to 0} 0$ is the truncation error which is second-order for CN, i.e. equal to $c\Delta\tau^2$ for some $c>0$.

To get an estimate of $\rho_{f_A}(\alpha)$, we assume that $\rho(A) \gg 1$ and $\rho(MA-I) > 1$. Therefore, we obtain

$$\alpha \leqslant \rho_{f_A}(\alpha) \leqslant \alpha(\rho(A)-1), \qquad \alpha \leqslant \rho_{f_{MA}}(\alpha) \leqslant \alpha(\rho(MA)-1).$$

For some $c>0$, we then have

$$\|y_{K_A}^{(A)} - A^{-\alpha}b\|_2 \quad \leqslant \quad c\Delta\tau_A^2\frac{e^{\alpha(\rho(A)-1)}-1}{\alpha}$$

and

$$\|y_{K_{MA}}^{(MA)} - A^{-\alpha}b\|_2 \quad \leqslant \quad c\Delta\tau_{MA}^2\frac{e^{\alpha(\rho(MA)-1)}-1}{\alpha}.$$

If we assume now that $1 > \rho(MA-I) > \varepsilon > 0$, we obtain

$$\alpha \leqslant \rho_{f_{MA}}(\alpha) \leqslant \alpha(1-\varepsilon)^{-1}\,,$$

resulting in the estimate

$$\|y_{K_{MA}}^{(MA)} - A^{-\alpha}b\|_2 \quad \leqslant \quad c\Delta\tau_{MA}^2\frac{e^{\alpha(1-\varepsilon)^{-1}}-1}{\alpha},$$

ending hence the proof. □

Based on the above proposition and since $\rho(A) \gg \rho(MA)$, it is reasonable to choose $\Delta\tau_{MA} \gg \Delta\tau_A$ in PCN. This implies that $K_{MA} \ll K_A$, thus leading to a much faster algorithm. More specifically, for a given error and assuming for instance that $\rho(MA-I) > 1$, we can estimate that the time step for PCN is roughly such that

$$\Delta\tau_{MA} \approx \Delta\tau_A\frac{e^{\alpha(\rho(A)-1)}-1}{e^{\alpha(\rho(MA)-1)}-1}.$$

Equivalently, for a fixed error, and since CN is a second-order scheme, the number of iterations (linearly related to the computational complexity) for PCN is reduced by a factor

$$\sqrt{(e^{\alpha(\rho(A)-1)}-1)/(e^{\alpha(\rho(MA)-1)}-1)}$$

compared to DCN. In other words, the matrix scaling does not increase the convergence order, but provides a better precision. After $K_A = \lfloor 1/\Delta\tau_A \rfloor$ iterations, we get a fixed error E for a number of iterations

$$K_{MA} = \lfloor 1/\Delta\tau_{MA} \rfloor \approx K_A \Big\lfloor \sqrt{\frac{e^{\alpha(\rho(MA)-1)}-1}{e^{\alpha(\rho(A)-1)}-1}} \Big\rfloor. \tag{6}$$

If we are now in the situation where $1 > \rho(MA - I) > \varepsilon$, then we conclude that

$$K_{MA} = \lfloor 1/\Delta\tau_{MA} \rfloor \approx K_A \Big\lfloor \sqrt{\frac{e^{\alpha(1-\varepsilon)^{-1}}-1}{e^{\alpha(\rho(A)-1)}-1}} \Big\rfloor. \tag{7}$$

Let us remark that from the parallel computing point of view, the ODE approach is expected to be less efficient than the Cauchy integral approach based on (2) which is obviously embarrassingly parallel. However, a parareal approach could certainly be adapted [13].

3.3. Jacobi/ILU ODE-preconditioning. We propose now to use a Jacobi ODE-preconditioner to solve $A^\alpha x = b$. The main difference with the scaling approach is that in general $[M, A] \neq 0$, implying that $M^\alpha (MA)^{-\alpha} \neq A^{-\alpha}$. However, we expect that $M^\alpha (MA)^{-\alpha} \approx A^{-\alpha}$. As a consequence, we propose to use $P = M^\alpha (MA)^{-\alpha}$ as an ODE-solver preconditioner in (4). Rather than solving (3), we consider

$$M^\alpha (MA)^{-\alpha} A^\alpha x = M^\alpha (MA)^{-\alpha} b\,. \tag{8}$$

The resulting algorithm is then

- **Step 1.** Compute $(MA)^{-\alpha} b$ as above, and then multiply by M^α which is a diagonal matrix, to get the new right-hand side vector.
- **Step 2.** To solve (8), use an iterative linear solver, requiring hence the computation of sequences $v_{k+1} = M^\alpha (MA)^{-\alpha} A^\alpha v_k$, for $v_k \in \mathbb{C}^n$ given, as follows
  1. Compute $w_k = A^\alpha v_k$, for A^α given.
  2. Compute $z_k = (MA)^{-\alpha} w_k$ by using the ODE algorithm (a matrix scaling could be used here to accelerate this step).
  3. Deduce $v_{k+1} = M^\alpha z_k$.

An ILU ODE-preconditioner can be used as well to efficiently solve (1). First, we implement an ILU ODE-preconditioner M for A, i.e. $A \approx \widetilde{L}\widetilde{U}$. We formally denote by M the inverse of $\widetilde{L}\widetilde{U}$. We then compute $(MA)^{-\alpha} b$. To this end, we numerically solve

$$\widetilde{L}\widetilde{U} y'(\tau) = -\alpha (A - \widetilde{L}\widetilde{U}) \big(\widetilde{L}\widetilde{U} + \tau (A - \widetilde{L}\widetilde{U})\big)^{-1} y(\tau), \qquad y(0) = b, \tag{9}$$

by a PCN and next calculate $M^\alpha (MA)^{-\alpha} b$ from

$$\widetilde{L}\widetilde{U} y'(\tau) = \alpha (I - \widetilde{L}\widetilde{U}) (\widetilde{L}\widetilde{U} + \tau (I - \widetilde{L}\widetilde{U}))^{-1} y(\tau), \qquad y(0) = (MA)^{-\alpha} b\,. \tag{10}$$

The above system is solved iteratively, which requires the computation of sequences $\{v_{k+1}\}_k$ defined by $v_{k+1} = M^\alpha (MA)^{-\alpha} A^\alpha v_k$, similarly to the Jacobi preconditioner.

Experiment 2. We consider the linear system $A^\alpha x = b$, for $A = B + \delta B + D \in \mathbb{R}^{n \times n}$ ($n = 150$). We define $B = \{b_{ij}\}_{1 \leqslant ij \leqslant n}$, with $b_{ij} = \texttt{rand}(0, 1)$. In addition, $D = \{d_{ij}\}_{ij}$ is a diagonal matrix such that $d_{ii} = 2.5 + \texttt{rand}(0, 1)$. The parameter

δ is fixed to $\delta = 0, 0.5, 1$ in the following examples, and $\alpha = 0.5$. The vector b is given in $\mathbb{R}^n$ and we take $\Delta\tau_A = 10^{-2}$ ($K_{A,MA} = 100$). For $\delta = 0$, we report in Fig. 3 (Bottom-Right) the residual history vs the iteration number, for the GMRES algorithm with a restart parameter equal to 20, respectively for: the DCN scheme and for the PCN method with scaling, Jacobi and ILU (for a tolerance equal to 10^{-3}) ODE-preconditioners. In Fig. 3 (Bottom/Top-Left, Top-Right), we report the spectrum of A and MA for $\delta = 0$, where M is chosen as a scaling matrix, Jacobi or ILU ODE-preconditioner. We conclude that the ILU ODE-preconditioner provides the best convergence rate, although Jacobi is also quite efficient. The scaling matrix is clearly not a good preconditioner and *should rather be used for a fast and direct evaluation of* $A^{-\alpha}b$. The same tests as above are repeated but with $\delta = 1$ (resp. $\delta = 0.5$). The results are available in Fig. 4 (Left) (resp. (Right)), with the same conclusion.

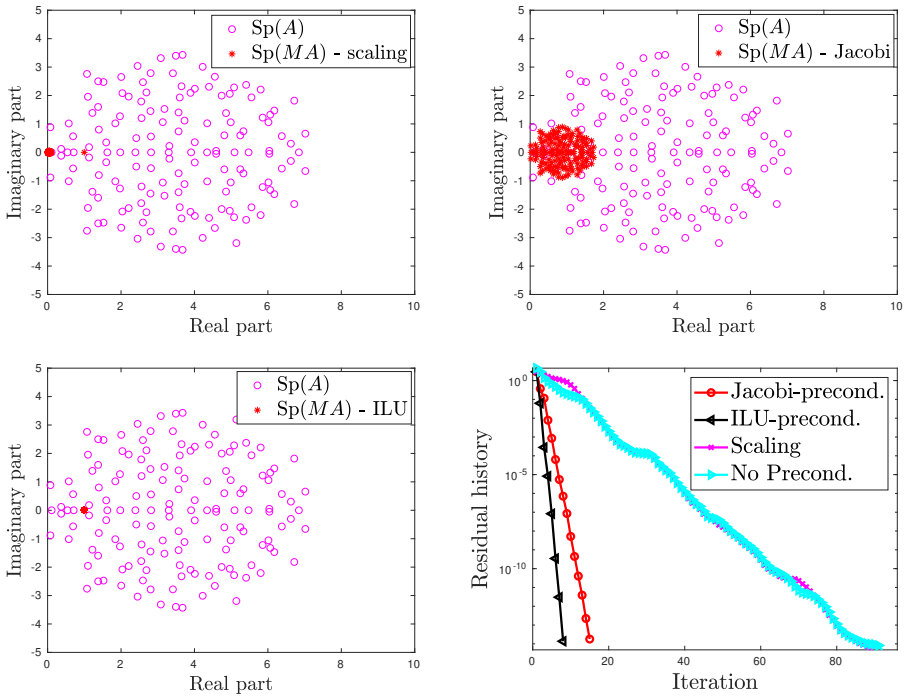

FIG. 3. **Experiment 2.** *For $\delta = 0$: Spectrum of A and MA with scaling matrix (Top-Left), Jacobi (Top-Right) and ILU ODE-preconditioner (Botton-Left). (Bottom-Right) Convergence history of GMRES without/with various preconditioners.*

Experiments 3. We next compare the convergence of the GMRES algorithm with Jacobi and ILU ODE-preconditioners, and without preconditioner, for different values of the fractional exponent $\alpha = 0.3, 0.6, 0.9$. We solve the system $A^\alpha x = b$, where $A = B + D \in \mathbb{R}^{n\times n}$ ($n = 150$), for $B = \{b_{ij}\}_{1\leqslant ij\leqslant n}$ such that $b_{ij} = \texttt{rand}(0, 1)$, and where $D = \{d_{ij}\}_{1\leqslant ij\leqslant n}$ is a diagonal matrix such that $d_{ii} = 3 + \texttt{rand}(0, 1)$. We take $K_A = K_{MA} = 100$, i.e. $\Delta\tau_{A,MA} := 10^{-2}$. We report the convergence graphs in Fig. 5, showing that the ILU ODE-preconditioner is by far, the most efficient preconditioner when the drop tolerance is 10^{-3}. In addition, it is not α-dependent. In Fig. 6, we report the convergence graph for the ILU ODE-preconditioner with $\alpha = 0.5$, and

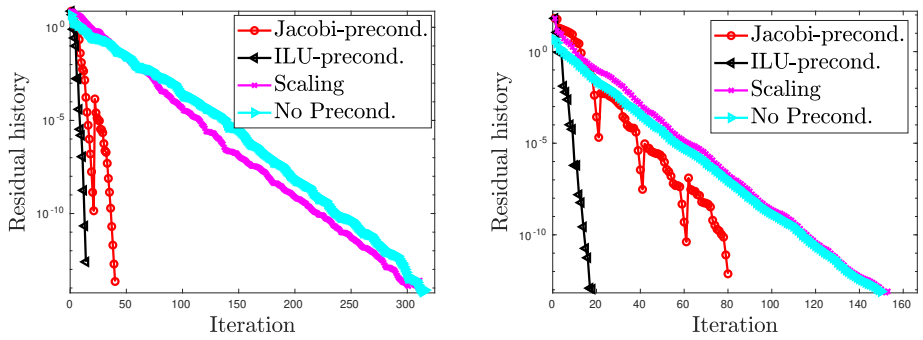

FIG. 4. **Experiment 2.** *Convergence history of GMRES without/with various preconditioners: (Left)* $\delta = 1$ *and (Right)* $\delta = 0.5$.

a drop tolerance of 10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}. As expected, the smaller the tolerance, the faster the GMRES algorithm.

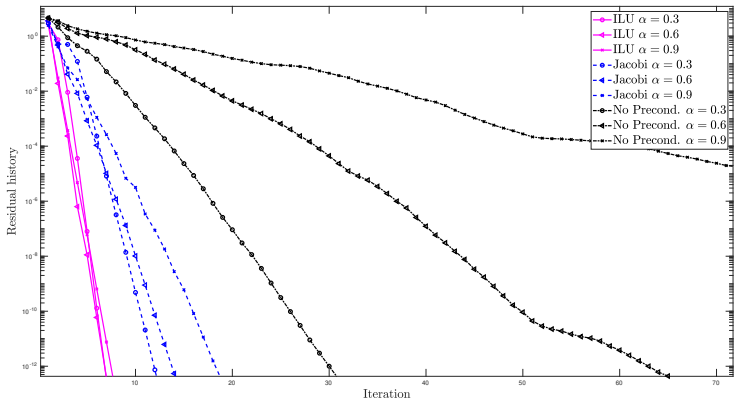

FIG. 5. **Experiment 3.** *Convergence history of GMRES without/with various preconditioners, for* $\alpha = 0.3, 0.6, 0.9$.

Experiment 4. Let $A^\alpha x = b$, where A is a *sparse* random matrix $A \in \mathbb{R}^{n\times n}$ with $n = 1000$ and $\alpha = 0.75$ (resp. $n = 2500$ and $\alpha = 0.5$) (see the sparsity pattern of A in Fig. 8 (Top-Left) (resp. Fig. 9 (Left))), with spectrum reported in Fig. 8 (Top-Right). We choose $K_A = K_{MA} = 16$ and the drop tolerance in the ILU preconditioner is fixed to 5×10^{-2}. We plot the different convergence rates in Fig. 8 (Bottom) (resp. Fig. 9 (Right)). As expected, we first observe that the convergence is much faster than for the non-sparse matrices, and yet the ILU-ODE preconditioner is the most efficient.

3.4. About the computational complexity. The key question addressed now is the computational cost for solving a fractional linear system, either directly or by using a differential-based preconditioning. We assume that $A \in \mathbb{R}^{n\times n}$ is a sparse

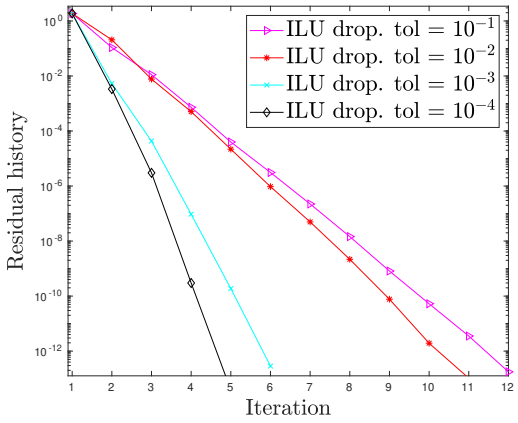

FIG. 6. **Experiment 3.** *Convergence history for the ILU ODE-preconditioned solver with drop tolerance equal to* 10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}.

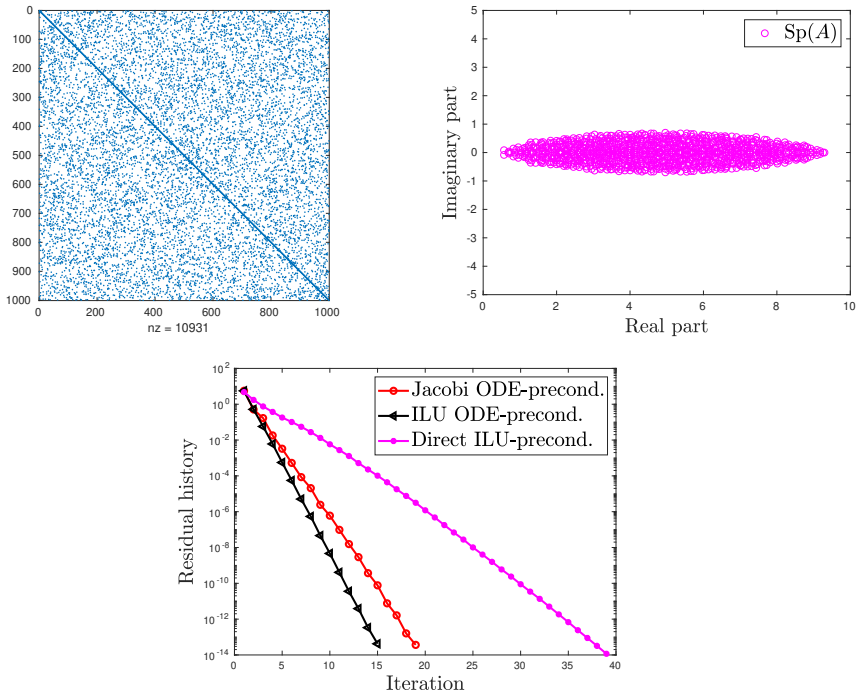

FIG. 7. **Experiment 4.** *For* $A \in \mathbb{R}^{1000\times 1000}$ *and* $\alpha = 0.75$ *(Top-Left) Sparse matrix structure A. (Top-Right) Spectrum of A. (Bottom) Convergence history.*

matrix or the perturbation of a sparse matrix, with eigenvalues having a positive real part, while A^α is a full matrix, where n is assumed to be large and $\alpha \in \mathbb{R}$. We alternatively consider that A^α is either known or unknown. In both cases, we have to compute $A^\alpha x$ for any vector x.

Direct and preconditioned solver. If A^α is given, any iterative (preconditioned or not)

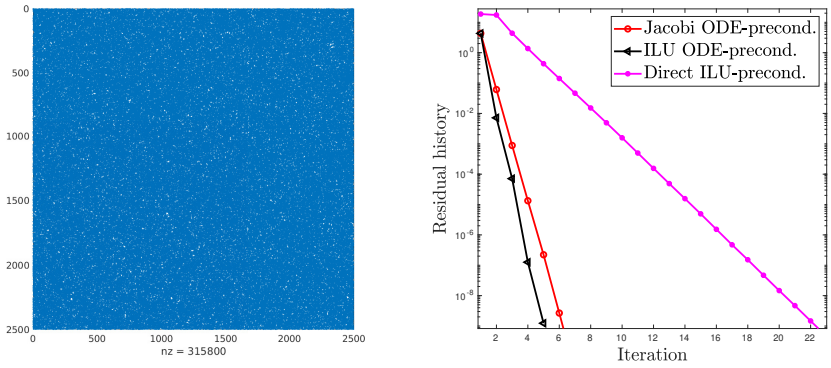

FIG. 8. **Experiment 4.** *For $A \in \mathbb{R}^{2500\times 2500}$ and $\alpha = 0.5$ (Top-Left) Sparse matrix structure A. (Top-Right) Spectrum of A. (Bottom) Convergence history.*

solver applied to $A^\alpha x = b$ would naturally require $c_0 n^3$ operations, with possibly a large prefactor if A^α is ill-conditioned. Moreover, deriving a parallel scalable algorithm is not *a priori* straightforward. If A^α is not given, we then need to compute $A^{-\alpha}b$ by using (3). Let us denote by $K_A^{(p)}$ the number of time iterations for computing $A^{-\alpha}x$, where p represents the order of the ODE-solver (the larger p, the smaller $K_A^{(p)}$). Each time iteration requires the computation of several preconditioned linear systems for a large (sparse) matrix. Hence, the computational complexity is given by $O(K_A^{(p)} n^\gamma)$, with $1 < \gamma < 3$, and γ close to 1 for a well-conditioned matrix A.

Jacobi ODE-preconditioner. Let $K_{MA}^{(p)}$ be the number of time iterations for computing $M^\alpha (MA)^{-\alpha} x$ with an order-p ODE-solver. Using a matrix scaling allows to decrease $K_{MA}^{(p)}$ as well (see Subsection 3.1), i.e. $K_{MA}^{(p)} \ll K_A^{(p)}$. Each time iteration of the differential-based solver requires the computation of several preconditioned sparse linear systems, for a total of $O(K_{MA}^{(p)} n^\gamma)$ operations. The latter represents the costs for preconditioning the linear system $A^\alpha x = b$ into $M^\alpha (MA)^{-\alpha} A^\alpha x = M^\alpha (MA)^{-\alpha} b$. This is a full linear system, which however is potentially very well preconditioned, and hence requires much less iterations than a direct method. We expect a total cost of about $c_1 K_{MA}^{(p)} n^\gamma + c_2 n^3$ operations, with $c_2 \ll c_0$.

ILU ODE-preconditioner. The ILU ODE-preconditioned algorithm is a little bit more computationally complex than for the Jacobi preconditioned method. The main difference comes from the need for solving sparse triangular linear systems to compute $v_{k+1} = M^\alpha z_k$, for all k. Again, this cost is relatively small and naturally leads to a drastic reduction of the number of GMRES iterations. Globally, we have $c_3 K_{MA}^{(p)} n^\gamma + c_4 n^3$ operations, with $c_1 < c_3$. However, compared to the Jacobi approach, less GMRES iterations are expected, i.e. $c_4 \ll c_2 \ll c_0$.

Experiment 5. To illustrate the computational complexity of the proposed preconditioning technique, we consider $A^{1/2}x = b$, with $A = B + B^T + D \in \mathbb{R}^{n\times n}$, where $B = \{b_{ij}\}_{1\leqslant ij\leqslant n}$ is such that $b_{ij} = \texttt{rand}(0, 0.1)$, and $D = \{d_{ij}\}_{1\leqslant ij\leqslant n}$ is the diagonal matrix $d_{ii} = 5 + \texttt{rand}(0, 1)$. We report in Fig. 7 the convergence rate of DCN method, and PCN scheme with scaling, Jacobi and ILU ODE-preconditioning techniques, with a drop tolerance equal to $10^{-1}, 10^{-2}, 10^{-3}$, for $n = 50, 100, 200$ (from Top-Left to Bottom). To compare the performance of the different approaches, we

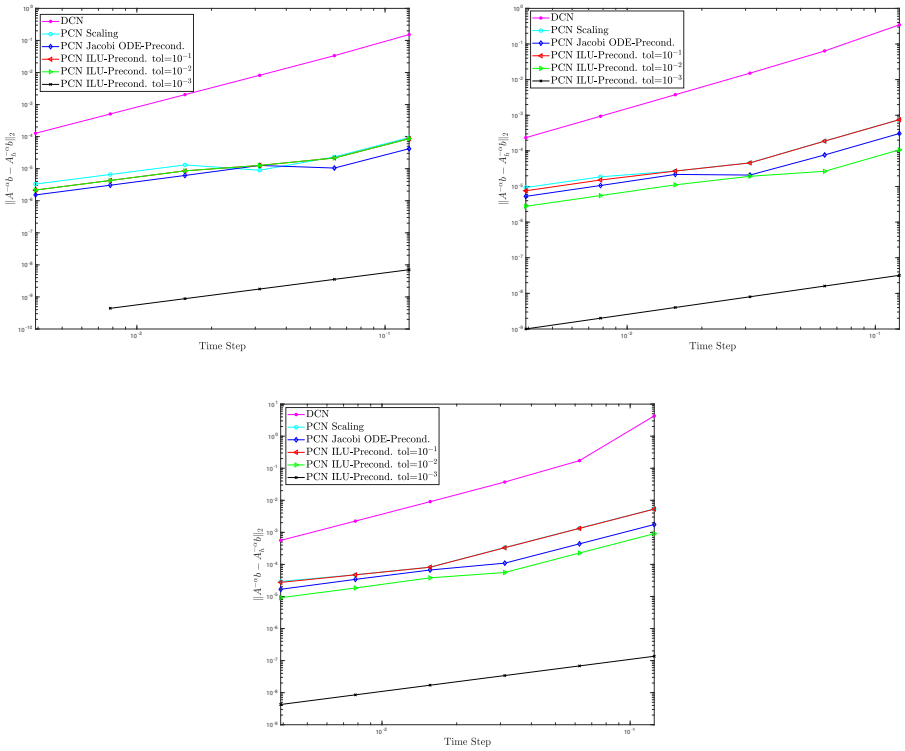

FIG. 9. **Experiment 5.** 2-*norm error as a function of the time step for DCN, PCN with scaling, Jacobi ODE-preconditioning and ILU ODE-preconditioning with a drop tolerance equal to* $10^{-1}, 10^{-2}, 10^{-3}$. *(Top-Left)* $n = 50$, *(Top-Right)* $n = 100$, *(Bottom)* $n = 200$.

need to fix the error and estimate for each preconditioning a corresponding time step (e.g. the number of iterations). More specifically, let us recall that $\Delta\tau_A$ is the time step of the DCN method, that is without ODE-preconditioning. Based on the above tests, we see that the time steps for both the scaling, Jacobi and ILU (with drop tolerance 10^{-1}) preconditioned algorithms can be taken roughly 64 times larger than for DCN, since the error for PCN with $\Delta\tau_{MA} = 1/2^3$ is about the same as for DCN but where $\Delta\tau_A = 1/2^9$ for this latter solver. Hence for the same error, it corresponds to 64 times less iterations. Let us remark that this is consistent with the analysis proposed in Subsection 3.2, where we numerically find: $K_{MA}/K_A \approx 59$ from (6) for the Jacobi preconditioning which is in relatively good agreement. Regarding the ILU ODE-preconditioning, the gain is even much higher, since very few iterations lead to a high precision.

3.5. Comparison with Padé's preconditioners. As an alternative to ODE-preconditioners, it is natural to use Padé's approximants to construct a so-called *Padé-preconditioner*, when A^α is known. These preconditioners can then be seen as an alternative to the ODE-preconditioner $P = M^\alpha(MA)^{-\alpha}$ in (4). More concretely, we introduce a mth-order Padé approximation $p_m^{(\alpha)}$, for $z \in \mathbb{C}$, such that $z^\alpha \approx p_m^{(\alpha)}(z)$.

For instance if $\alpha = 1/2$, the θ-rotating Padé approximants are defined by

$$\sqrt{z} \approx p_m^{(1/2)}(z) = \sum_{k=0}^{m} a_k^{(m)} - \sum_{k=1}^{m} \frac{a_k^{(m)} d_k^{(m)}}{z + d_k^{(m)}},$$

where the coefficients are given, for $\theta \in [0, \pi/2)$, by

$$(11)\; a_0^{(m)} = 0, \qquad a_k^{(m)} = \frac{e^{\mathtt{i}\theta}}{m \cos^2\Big(\dfrac{(2k+1)\pi}{4m}\Big)}, \qquad d_k^{(m)} = e^{\mathtt{i}\theta} \tan^2\Big(\frac{(2k+1)\pi}{4m}\Big).$$

A natural preconditioner in (4) for $A^{1/2}$ is then

$$M_{\text{Padé}} := \Big(\sum_{k=0}^{m} a_k^{(m)} I - \sum_{k=1}^{m} a_k^{(m)} d_k^{(m)} (A + d_k^{(m)} I)^{-1}\Big) A^{-1},$$

which is an approximation of $A^{-1/2}$. Hence, if $A^{1/2}$ is given, we solve

$$\Big(\sum_{k=0}^{m} a_k^{(m)} I - \sum_{k=1}^{m} a_k^{(m)} d_k^{(m)} (A + d_k^{(m)} I)^{-1}\Big) A^{-1} A^{\alpha} x = \Big(\sum_{k=0}^{m} a_k^{(m)} I - \sum_{k=1}^{m} a_k^{(m)} d_k^{(m)} (A + d_k^{(m)} I)^{-1}\Big) A^{-1} b\,.$$

From a practical point of view, intermediate sparse linear systems have to be solved for each k, $0 \leqslant k \leqslant m$, which also require a second linear system preconditioning.

Experiment 6. We solve $A^{1/2}x = b$, with $A = B + B^T/2 + D \in \mathbb{R}^{n\times n}$ ($n = 200$), where $B = \{b_{ij}\}_{1\leqslant ij\leqslant n}$ is such that $b_{ij} = \texttt{rand}(0,1)$. The diagonal matrix $D = \{d_{ij}\}_{1\leqslant ij\leqslant n}$ is defined by $d_{ii} = 3 + \texttt{rand}(0,1)$. We compute in logscale

$$\|M_{\text{Padé}} A^{\alpha} - I\|_2 = \sup_{x\neq 0} \frac{\|M_{\text{Padé}} A^{\alpha} x - x\|_2}{\|x\|_2}$$

as a function of the number of Padé terms, from $m = 4$ to $m = 2048$ and for $\theta = -\pi/6, -\pi/3, 0, \pi/6, \pi/3, \pi/2$. We observe on Fig. 10 (Left) a convergence in logscale as well as θ-independence, except for some very small values of m in Fig. 10 (Right) corresponding to realistic values for a practical computation.

The interest of the Padé's preconditioner is that no ODE-system needs to be solved. However, Padé's preconditioners still require the solution to linear systems, and therefore its efficiency is asymptotically similar to ODE-preconditioners. We compare the convergence rate of ILU ODE-preconditioners and Padé's preconditioners. Essentially, Padé's preconditioners lead to the solution to $O(m)$ linear systems.

Experiment 6bis. In this experiment, we compare below the use of a Padé preconditioner with m equal to the number of time steps in the ODE-solver $K_{A,MA}$. We consider $A^{1/2}x = b$, where $A = B + 0.5B^T + D \in \mathbb{R}^{100} \times \mathbb{R}^{100}$, with $b_{ij} = \texttt{rand}(0,1)$ and $D = \{d_{ij}\}_{ij}$ is a diagonal matrix such that $d_{ii} = 3 + \texttt{rand}(0,1)$. We compare the convergence rate when using ILU ODE-preconditioners (for a drop tolerance equal to 10^{-4}) with K_{MA} time iterations, and Padé's preconditioner with $m = K_{MA}$ and $\theta = \pi/3$. We observe that the Padé's preconditioner leads to a good approximation of $A^{-1/2}$, but then the rate of convergence of the GMRES solver is the same as without preconditioner. For the same matrix A, we compare the convergence rate for

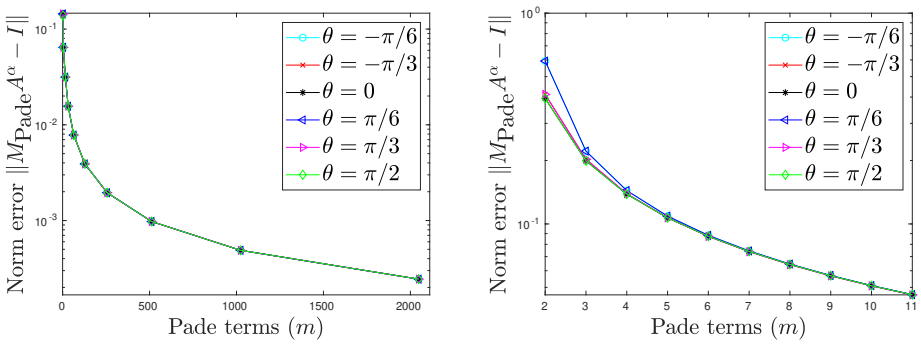

FIG. 10. **Experiment 6.** *2-norm error $\|M_{\alpha}A^{\alpha} - I\|_2$ as a function of the number of Padé terms, for $\theta = -\pi/6, -\pi/3, 0, \pi/6, \pi/3, \pi/2$. (Left) $m = 4$ to $m = 2048$. (Right) $m = 2$ to 11.*

$K_{MA} = m = 4, 16, 64, 256$ in Fig. 11. We observe a faster convergence with ILU ODE-preconditioner than with Padé's preconditioner, although the latter provides faster convergence during the first GMRES iterations, but with a smaller slope than ILU. The ODE-solver is also proved to be not so much sensitive to K_{MA}, which is a strength of the method. Running a similar test (see Fig. 12) with the drop tolerance 5×10^{-2}, we observe that the convergence rate of the GMRES solver using the ILU ODE-preconditioner is less sensitive to K_{MA} than the Padé preconditioner to m.

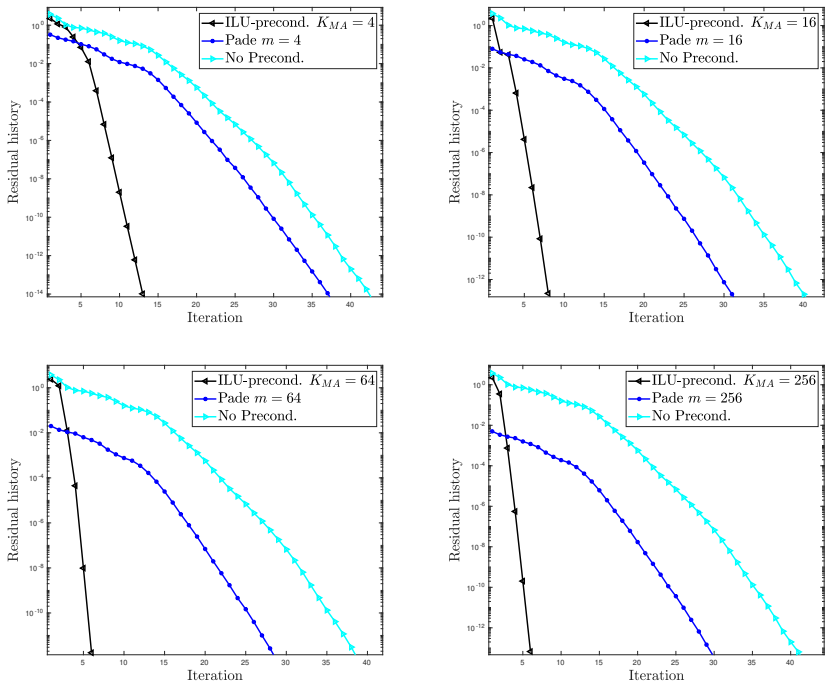

FIG. 11. **Experiment 6bis.** *convergence rate of the GMRES: ILU (with drop tolerance 10^{-4}), Padé preconditioner with (Top-Left) $m = 4$, (Top-Right) $m = 16$, (Bottom-Left) $m = 64$ and (Bottom-Right) $m = 256$.*

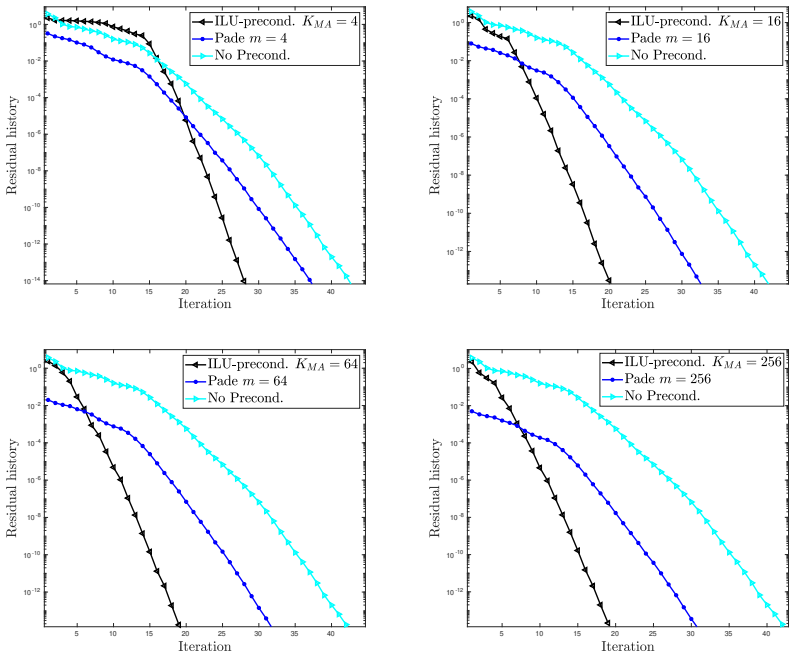

FIG. 12. **Experiment 6bis.** *GMRES convergence comparison: ILU (drop tolerance* 5×10^{-2}*), Padé preconditioner with (Top-Left)* $m = 4$*, (Top-Right)* $m = 16$*, (Bottom-Left)* $m = 64$ *and (Bottom-Right)* $m = 256$*.*

4. Extension to $f(A)x = b$. The methodology proposed above can be extended to some more general systems of the form

$$f(A)x = b, \tag{12}$$

where f is an analytic function of A. We still assume that A is a large sparse matrix, or the perturbation of a sparse matrix, which is diagonalizable in $\mathbb{C}$ with eigenvalues having non-negative real parts. We propose a derivation of ODE-solver preconditioners for (12) following a similar strategy as for the FLS.

4.1. ODE formulation and direct discretization. To extend the above ideas and derive the ODE system, we suppose that the solution to our system is of the form $y(\tau) = f(I+\tau(A-I))b$, with $y(0) = f(I)b$. Since $y(1) = f(A)b$, we want to determine a first-order system of ODEs which has $y : \tau \mapsto y(\tau)$ as a solution. To this end, we assume that $z(t) = f(t)$ in $\mathbb{R}$ is solution to a first-order ODE: $f'(t) = G(f(t))$, where G must be built. Formally, we have $G(s) = f'(f^{-1}(s))$. Thus assuming that $y(\tau) = f(I + \tau(A - I))b$, one gets

$$y'(\tau) = (A - I)f'(I + \tau(A - I))b = (A - I)G\big(f(I + \tau(A - I))\big)b\,.$$

Hence, we consider the following system of ODEs

$$y'(\tau) \quad = \quad (A - I)f'\big(I + \tau(A - I)\big)f(I + \tau(A - I))^{-1}y(\tau), \quad y(0) = f(I)b\,.$$

We first need to check when this system is well-posed. Basically, this requires f to be analytic (which is a sufficient condition) and $f'(f^{-1})$ to be (locally) Lipschitz continuous. Then, the following proposition holds.

PROPOSITION 4.1. *Let us assume that f is an analytic diffeomorphism in $\mathbb{C}\backslash\mathbb{R}_-$ and that A is diagonalizable with a spectrum in $\mathbb{C}\backslash\mathbb{R}_-$. Then, the solution y to the following system of ODEs*

$$y'(\tau) = (A-I)f'\big(I+\tau(A-I)\big)f(I+\tau(A-I))^{-1}y(\tau), \quad y(0)=f(I)b\,, \tag{13}$$

is such that $y(1)=f(A)b$.

Applying the above result, we directly deduce the

COROLLARY 4.1. *Consider $f(z)=z^\alpha$ in Proposition 4.1, then (13) degenerates into (3).*

Proof. First, we formally have $f'(A)=\alpha A^{\alpha-1}$. Hence, we obtain

$$y'(\tau) = \alpha(A-I)\big(I+\tau(A-I)\big)^{\alpha-1}\big(I+\tau(A-I)\big)^{-\alpha}y(\tau),$$

which also writes

$$y'(\tau) = \alpha(A-I)\big(I+\tau(A-I)\big)^{-1}y(\tau), \quad y(0)=b.$$

This is finally consistent with (3). □

From now on, in order to compute $f(A)b$, we simply have to numerically solve (13). In this paper, we are however interested in the solution to linear systems. In other words, we want to formally compute $x=f(A)^{-1}b$ as the solution to $f(A)x=b$, and more specifically to build a preconditioner for $f(A)$ based on an approximation of $f(A)^{-1}$. We then have the following

COROLLARY 4.2. *Let us suppose that f is an analytic diffeomorphism and that A is a diagonalizable matrix with a spectrum in $\mathbb{C}\backslash\mathbb{R}_-$. Setting $\mathcal{F}:=1/f$, the solution y to the ODE system*

$$y'(\tau) = (A-I)\mathcal{F}'\big(I+\tau(A-I)\big)f\big(I+\tau(A-I)\big)y(\tau), \quad y(0)=\mathcal{F}(I)b, \tag{14}$$

satisfies $y(1)=f(A)^{-1}b$.

From (14), we can then construct an ODE-preconditioner to efficiently solve $f(A)x=b$. More precisely, we have the following result.

PROPOSITION 4.2. *We suppose that f is an analytic diffeomorphism in $\mathbb{C}\backslash\mathbb{R}_-$, and that $A\in\mathbb{R}^{n\times n}$ is a diagonalizable matrix with a spectrum in $\mathbb{C}\backslash\mathbb{R}^-$. We define $\mathcal{F}:=1/f$ and assume that $b\in\mathbb{R}^n$. Then, the following numerical DCN scheme*

$$\begin{aligned} y_{k+1}^{(A)} &= y_k^{(A)} + \frac{\Delta\tau_A}{2}(A-I)\mathcal{F}'\big(I+\tau_{k+1}(A-I)\big)f\big(I+\tau_{k+1}(A-I)\big)y_{k+1}^{(A)} \\ &\quad + \frac{\Delta\tau_A}{2}(A-I)\mathcal{F}'\big(I+\tau_k(A-I)\big)f\big(I+\tau_k(A-I)\big)y_k^{(A)}\,, \end{aligned}$$

with $y_0^{(A)}=b$ and $K_A\tau_A=1$, is such that there exists a positive real-valued constant $C=C(f,A)>0$ so that

$$\|y_{K_A}^{(A)}-f(A)^{-1}b\|_2 \leqslant C\Delta\tau_A^2\,.$$

Example 1. We consider the following system

$$\exp(A)x=b\,, \tag{15}$$

where A is a real-valued diagonalizable matrix with positive eigenvalues. We define $\mathcal{F}(z) = e^{-z}$. In this case, the ODE system (14) writes

$$y'(\tau) = -(A-I)\exp\big(-(I+\tau(A-I))\big)\exp\big(I+\tau(A-I)\big)y(\tau), \quad y(0) = e^{-I}b\,.$$

Hence, we have

$$y'(\tau) = (I-A)y(\tau), \quad y(0) = e^{-I}b\,. \tag{16}$$

As expected, the solution to (16) is $\exp(-A)b$, which is the solution to (15).

4.2. Preconditioning for $f(A)x = b$. Unlike Section 3, it is not usually possible to easily determine $f(A)$ from $f(MA)$, and additional conditions are then required. The extension of the strategy developed for $A^\alpha x = b$ to systems of the form $f(A)x = b$ requires a relation between $f(A)^{-1}$ and $f(MA)^{-1}$. In the following, we assume that there exists a function $\mathcal{R}$ such that

$$f(A)^{-1}b = \mathcal{R}(M, f(MA)^{-1})b\,. \tag{17}$$

When this hypothesis is satisfied, we can expect that the preconditioning method is efficient. In practice, (17) is rarely exactly satisfied (except for $f(z) = z^\alpha$, where $\mathcal{R}(M, f(MA)^{-1}) = M^\alpha(MA)^{-\alpha}$). We can then select a preconditioner such that $f(A)^{-1}b \approx \mathcal{R}(M, f(MA)^{-1})b$.

4.2.1. Scaling ODE-preconditioning. We first discuss the scaling of the CN solver, referred to as *Preconditioned Crank-Nicolson* (PCN) method for the scaling matrix $M = I/\|A\|_2$ and where A is diagonally dominant. Unlike Section 3, computing $f(A)^{-1}$ can be sometimes impossible. Because M is a scaling matrix, we have: $f(A)^{-1}b = \mathcal{R}(M, f(MA)^{-1})b$. We therefore compute the finite sequence $\{y_k^{(MA)}\}_{0\leqslant k\leqslant K_{MA}}$, with $\tau_{MA} = K_{MA}\Delta\tau_{MA} = 1$, i.e. we solve

$$\begin{aligned} y_{k+1}^{(MA)} = y_k^{(MA)} &+ \frac{\Delta\tau_{MA}}{2}(MA-I)\mathcal{F}'\big(I+\tau_{k+1}(MA-I)\big)f\big(I+\tau_{k+1}(MA-I)\big)y_{k+1}^{(MA)} \\ &+ \frac{\Delta\tau_{MA}}{2}(MA-I)\mathcal{F}'\big(I+\tau_k(MA-I)\big)f\big(I+\tau_k(MA-I)\big)y_k^{(MA)}\,. \end{aligned}$$

At time τ_{MA}, $y_{K_{MA}}^{(MA)}$ is an estimate of $f(MA)^{-1}b$. We then deduce that

$$y_{K_{MA}}^{(MA)} \quad \approx \quad \mathcal{R}\big(M, f(MA)^{-1}\big)b \quad = \quad f(A)^{-1}b\,.$$

Experiment 7. For $b \in \mathbb{R}^n$, we solve $\log(A)x = b$, where $A = B + B^T + D \in \mathbb{R}^{n\times n}$, for $n = 50$. The matrix $B = \{b_{ij}\}_{1\leqslant ij\leqslant n}$ is defined by $b_{ij} = \texttt{rand}(0,1)$, and D is the diagonal matrix with elements : $d_{ii} = 26 + \texttt{rand}(0,1)$. For $M = I/\|A\|_2$, we have: $\log(MA)b = \log(A)b + \log(I/\|A\|_2)$. We report in Fig. 13 (Right), the convergence history of the 2-norm error $\|f(A)^{-1}b - f(A_h)^{-1}b\|_2$, where $\log(A_h)$ is computed by the DCN and the scaled PCN schemes. In Fig. 13 (Left), we report the spectrum of A and MA.

Experiment 7bis. We repeat the same computations but with $A = B + D \in \mathbb{R}^{n\times n}$ which has a complex spectrum with eigenvalues having positive real parts. The results are reported in Fig. 14, showing an improvement of the convergence of PCN over DCN.

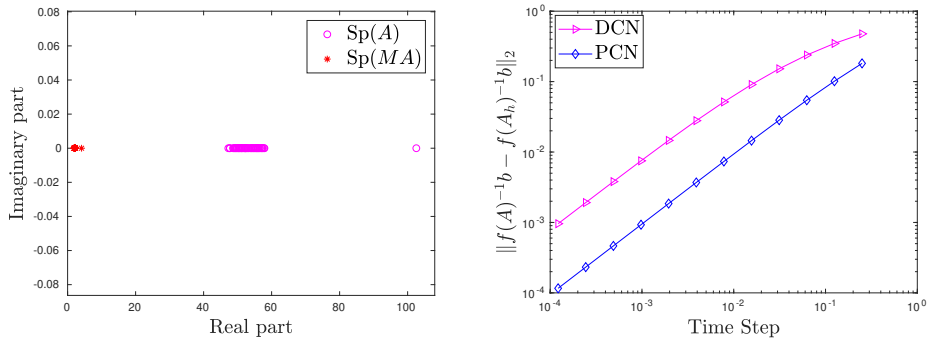

FIG. 13. **Experiment 7.** *(Left) Spectrum of A and MA. (Right) Convergence rate of the DCN and scaled PCN schemes.*

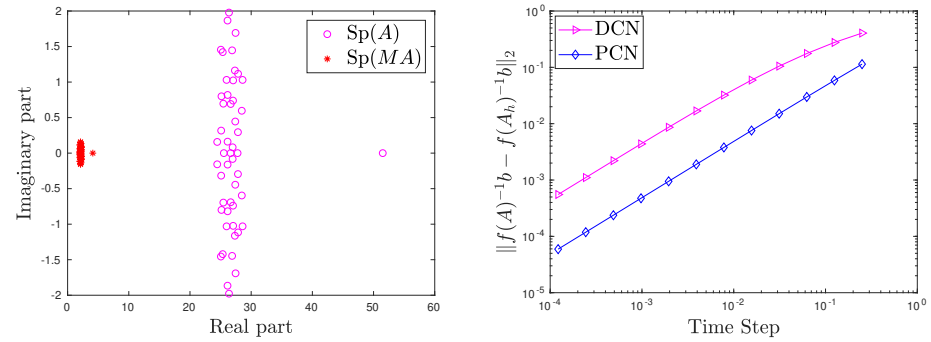

FIG. 14. **Experiment 7bis.** *(Left) Spectrum of A and MA. (Right) Convergence rate of the DCN and scaled PCN schemes.*

4.2.2. Jacobi and ILU ODE-preconditioning. We assume that f is an analytic diffeomorphism and define $\mathcal{F} := 1/f$. The preconditioning of the linear system $f(A)x = b$ requires additional analytical steps compared to $A^\alpha x = b$. Indeed, let us denote by M a preconditioner for A such that $MA \approx I$. As a consequence, $\|D_{MA}\|$ is close to 0 (where $\|\cdot\|$ is a matrix norm), with $D_{MA} = MA - I$. Formally, we then have, for $\tau \in (0,1]$ and $\|D_{MA}\|$ small enough,

$$\begin{array}{rcl} \mathcal{F}'(I+\tau(MA-I)) & = & \mathcal{F}'(I) + \tau D_{MA}\mathcal{F}''(I) + O(\tau^2 D_{MA}^3)\,, \\ f(I+\tau(MA-I)) & = & f(I) + \tau D_{MA} f'(I) + O(\tau^2 D_{MA}^2)\,. \end{array} \tag{18}$$

Rather than computing exactly $f(MA)^{-1}b$ $(= \mathcal{F}(MA)b)$ which is usually impossible (where MA has a spectrum clusterized around $1 \in \mathbb{C}$) through the ODEs system

$$y'_{MA}(\tau) \quad = \quad D_{MA}\mathcal{F}'\big(I+\tau D_{MA}\big) f(I+\tau D_{MA}) y_{MA}(\tau), \quad y_{MA}(0) = \mathcal{F}(I)b\,,$$

with $\widetilde{y}_{MA}(0) = \mathcal{F}(I)b$, we propose to numerically solve, for $\tau \in (0,1]$,

$$\widetilde{y}'_{MA}(\tau) = D_{MA}(\mathcal{F}'(I)f(I) + \tau\big(D_{MA}\mathcal{F}''(I)f(I) + \mathcal{F}'(I)D_{MA}f'(I)\big))\widetilde{y}_{MA}(\tau). \tag{19}$$

We denote by $\tau \mapsto \mathcal{G}_{MA}(\tau)$ the semi-group associated to (19), such that $\widetilde{y}_{MA}(1) = \mathcal{G}_{MA}(1)b$. We then have: $\mathcal{G}_{MA}(1)b = \widetilde{y}_{MA}(1) \approx f^{-1}(MA)b$. For preconditioning

$f(A)x = b$, we compute

$$f(M)\mathcal{G}_{MA}(1)f(A)x = f(M)\mathcal{G}_{MA}(1)b\,.$$

Let us remark that if M is a Jacobi preconditioner, then $f(M)$ is trivial to compute.

Experiment 8. We consider $A^\alpha \cos(\varepsilon A)x = b$, with $\varepsilon \geqslant 0$. We set $f(z) = z^\alpha \cos(\varepsilon z)$, and $\mathcal{F}(z) = A^{-\alpha}\sec(\varepsilon z)$. We assume that $A = B + 0.75B^T + D \in \mathbb{R}^{n\times n}$ $(n = 100)$. The coefficients of the matrix B are given by $b_{ij} = \texttt{rand}(0,1)$, while D is a diagonal matrix with coefficients $d_{ii} = 2 + \texttt{rand}(0,1)$. The vector b is given in $\mathbb{R}^n$. We fix $\alpha = 0.5$, $\varepsilon = 0.1$ and $K_A = K_{MA} = 100$. In Fig. 15, we plot the spectrum of A and MA, for M chosen as a Jacobi or ILU ODE-preconditioner (Top). We report in Fig. 15 (Bottom) the residual history of GMRES with a restart parameter fixed to 50 iterations, for the DCN approach and for the PCN scheme with a scaling, Jacobi and ILU ODE-preconditioners (the drop tolerance is 10^{-3}) (the time-step is fixed to 10^{-2}). Again, the best convergence is obtained for the ILU ODE-preconditioner but Jacobi preconditioning is also efficient. Unlike these two preconditioners, the scaling strategy does not work well. We next perform the same test but with $\varepsilon = 0.01$ and $\varepsilon = 0.2$ showing that the convergence rate is strongly dependent on the value of ε (see Fig. 16). This is explained by the fact that the smaller ε, the better the approximation of $1/f(A)$ for the proposed ODE-preconditioners.

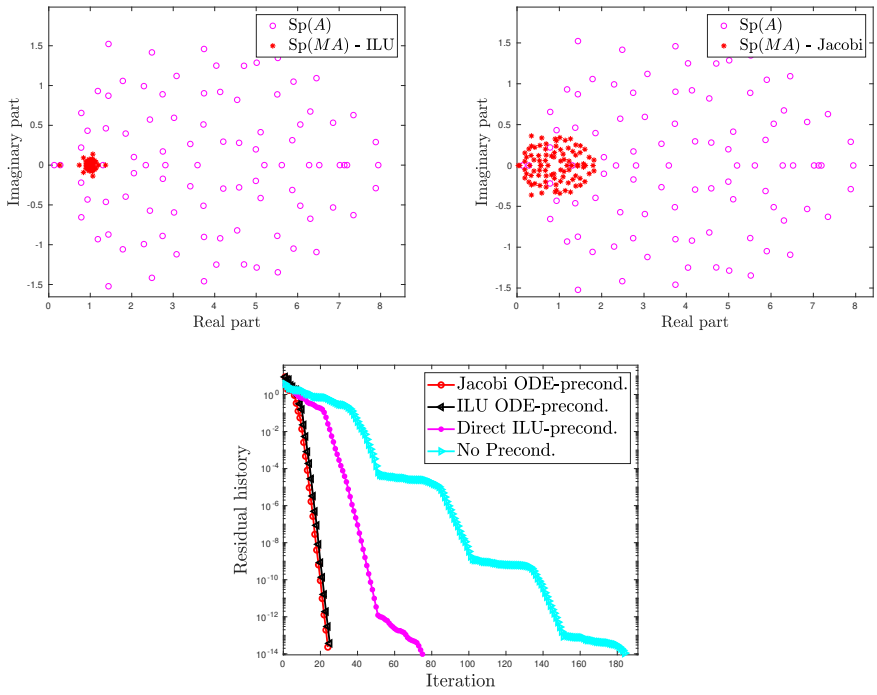

FIG. 15. **Experiment 8.** $\varepsilon = 0.1$. *(Top-Left) Spectrum of A and MA with ILU ODE preconditioning. (Top-Right) Spectrum of A and MA with Jacobi ODE-preconditioner. (Bottom) Convergence history for Jacobi and ILU ODE-preconditioned solvers, scaling preconditioner, and without preconditioner.*

Experiment 8bis. We end by performing the same test as above with $f(z) =$

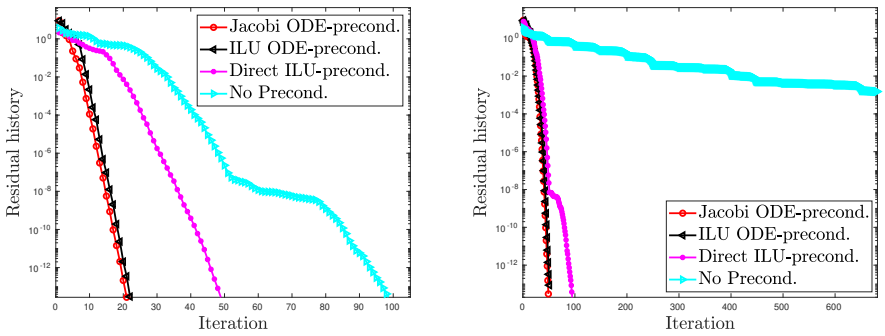

FIG. 16. **Experiment 8.** *Convergence history for Jacobi and ILU ODE-preconditioned solvers, scaling preconditioner, and without preconditioner (Left)* $\varepsilon = 10^{-2}$. *(Right)* $\varepsilon = 5 \times 10^{-1}$.

$z^\alpha \exp(-\varepsilon z)$ (for $\varepsilon = 10^{-1}$ and $\alpha = 0.75$) and $f(z) = z^\alpha \cos(\varepsilon z)$ (with $\varepsilon = 10^{-1}$). The results are reported in Fig. 17.

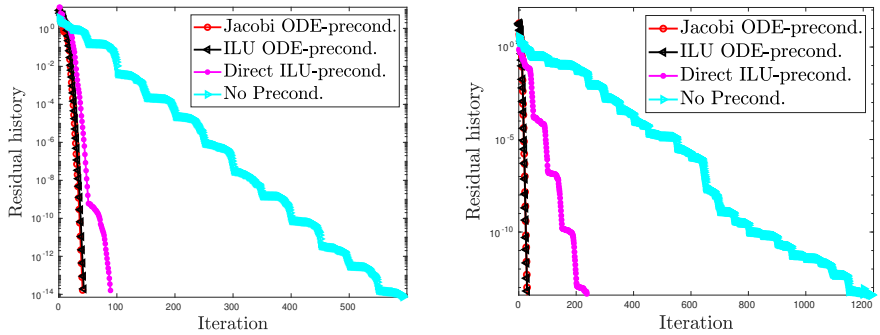

FIG. 17. **Experiment 8bis.** *Convergence history for Jacobi and ILU ODE-preconditioned solvers, scaling preconditioner, and without preconditioner, for* $\varepsilon = 10^{-1}$. *(Left)* $f(z) = z^\alpha \exp(-\varepsilon z)$. *(Right)* $f(z) = z^\alpha \cos(\varepsilon z)$.

5. Conclusion. In this paper, we have proposed a general preconditioning methodology for solving fractional linear systems $A^\alpha x = b$ and more generally $f(A)x = b$, by using i) the solution to a differential system $\mathcal{S}$, and ii) classical preconditioners allowing for a concentration of the spectrum of A, hence reducing the number of iterations for solving $\mathcal{S}$. Several numerical experiments and analytical arguments show the relevance of the derived approach. More elaborated ODE-based preconditioners will be developed in a forthcoming paper, as well as some applications to FPDEs.

REFERENCES

[1] L. Aceto and P. Novati. Efficient implementation of rational approximations to fractional differential operators. *J. Sci. Comput.*, 76(1):651–671, 2018.

[2] X. Antoine and E. Lorin. Double-preconditioning for fractional linear systems. Application to fractional Poisson equations. *Submitted*, 2019.

[3] X. Antoine and E. Lorin. Towards Perfectly Matched Layers for time-dependent space fractional PDEs. *J. Comp. Phys.*, 391:59–90, 2019.

[4] U. Biccari, M. Warma, and E. Zuazua. Local elliptic regularity for the Dirichlet fractional Laplacian. *Adv. Nonlinear Stud.*, 17(2):387–409, 2017.

[5] E. Carson and N. J. Higham. Accelerating the solution of linear systems by iterative refinement in three precisions. *SIAM J. Sci. Comput.*, 40(2):A817–A847, 2018.

[6] P. I. Davies and N. J. Higham. Computing $f(A)b$ for matrix functions f. In *QCD and Numerical Analysis III*, volume 47 of *Lect. Notes Comput. Sci. Eng.*, pages 15–24. Springer, Berlin, 2005.

[7] E. Di Nezza, G. Palatucci, and E. Valdinoci. Hitchhiker's guide to the fractional Sobolev spaces. *Bulletin des Sciences Mathématiques*, 136(5):521–573, 2012.

[8] N. Hale, N.J. Higham, and L. N. Trefethen. Computing $\mathbf{A}^\alpha$, $\log(\mathbf{A})$, and related matrix functions by contour integrals. *SIAM J. Numer. Anal.*, 46(5):2505–2523, 2008.

[9] N.J. Higham. Evaluating Padé approximants of the matrix logarithm. *SIAM Journal on Matrix Analysis and Applications*, 22(4):1126–1135, 2001.

[10] R. A. Horn and C. R. Johnson. *Topics in Matrix Analysis*. Cambridge University Press, Cambridge, 1991.

[11] Y. Huang and A. Oberman. Numerical methods for the fractional Laplacian: a finite difference–quadrature approach. *SIAM J. Numer. Anal.*, 52(6):3056–3084, 2014.

[12] X. Li and C. Xu. Existence and uniqueness of the weak solution of the space-time fractional diffusion equation and a spectral method approximation. *Commun. Comput. Phys.*, 8(5):1016–1051, 2010.

[13] J.-L. Lions, Y. Maday, and G. Turinici. Résolution d'EDP par un schéma en temps "pararéel". *C. R. Acad. Sci. Paris Sér. I Math.*, 332(7):661–668, 2001.

[14] A. Lischke, G. Pang, M. Gulian, F. Song, C. Glusa, X. Zheng, Z. Mao, W. Cai, M. Meerschaert, M. Ainsworth, and G. E. Karniadakis. What is the fractional laplacian? *arXiv:1801.09767v2*, 2018.