A block bi-diagonal Toeplitz preconditioner for block lower triangular Toeplitz system from time-space fractional diffusion equations

Yong-Liang Zhao

Joint work with Xian-Ming Gu

Innsbruck • Austria, 25 Jan., 2020

Outline

Fractional calculus

Marquis de l'Hôspital: "What does $\frac{d^n}{dx^n} f(x)$ mean if $n = 1/2$?"

Fractional calculus

Marquis de l'Hôspital: "What does $\frac{d^n}{dx^n} f(x)$ mean if $n = 1/2$?"

Leibniz: "$\cdots$ Thus it follows that $d^{\frac{1}{2}}x$ will be equal to $x\sqrt{dx : x}$. This is an apparent paradox from which, one day, useful consequences will be drawn." (In response to Marquis de l'Hôpital, 1695)

Fractional calculus

- There exists numerous definitions of fractional derivatives[a],

[a] I. Podlubny. Fractional Differential Equations. Vol. 198, Academic Press, San Diego, CA, 1998.

Fractional calculus

- There exists numerous definitions of fractional derivatives[a],
- In general such definitions are not equivalent,

[a] I. Podlubny. Fractional Differential Equations. Vol. 198, Academic Press, San Diego, CA, 1998.

Fractional calculus

- There exists numerous definitions of fractional derivatives[a],
- In general such definitions are not equivalent,
- We can think of recasting several differential model with classical derivative to their fractional counterpart.

---

[a] I. Podlubny. Fractional Differential Equations. Vol. 198, Academic Press, San Diego, CA, 1998.

Fractional calculus

Applications of fractional calculus[b]:

- Physics: Electrical spectroscopy impedance, Continuous time random walk, $\cdots$;
- Control: Air-based precision positioning system, Active damping of flexible structures, $\cdots$;
- Image processing: Image denoising, Image inpainting, $\cdots$;
- Biology: HIV infection, Morris-Lecar neuron model, $\cdots$;
- Economic: Barrier options pricing, Black-Scholes model, $\cdots$;
- $\cdots$

[b] H. Sun et al. "A new collection of real world applications of fractional calculus in science and engineering". In: Commun. Nonlinear Sci. Numer. Simulat. 64 (2018), pp. 213–231.

Fractional diffusion model

The time-space fractional diffusion equation (TSFDE):

$$\begin{cases} {}_0^C\mathcal{D}_t^{\alpha}\, u(x,t) = e_1\, {}_0\mathcal{D}_x^{\beta}\, u(x,t) + e_2\, {}_x\mathcal{D}_L^{\beta}\, u(x,t) \\ \qquad\qquad + f(x,t), \ 0 < t \le T, \ 0 \le x \le L, \\ u(x,0) = u_0(x), \qquad 0 \le x \le L, \\ u(0,t) = u(L,t) = 0, \qquad 0 \le t \le T, \end{cases} \tag{1}$$

- $\alpha \in (0,1)$, $\beta \in (1,2)$, $e_1, e_2 > 0$.
- Known functions: $u_0(x)$ - initial value; $f(x,t)$ - source term.
- $u(x,t)$, $u_0(x)$ and $f(x,t)$ are sufficiently smooth functions.

Caputo and Riemann-Liouville fractional derivatives

Caputo fractional derivative for $\alpha \in (0,1)$

$$ {}_0^C\mathcal{D}_t^{\alpha}\, u(x,t) = \frac{1}{\Gamma(1-\alpha)} \int_0^t (t-\eta)^{-\alpha} \frac{\partial u(x,\eta)}{\partial \eta} d\eta. $$

Riemann-Liouville fractional derivatives for $\beta \in (1,2)$

$$ {}_0\mathcal{D}_x^{\beta}\, u(x,t) = \frac{1}{\Gamma(2-\beta)} \frac{d^2}{dx^2} \int_0^x \frac{u(\eta,t)}{(x-\eta)^{\beta-1}} d\eta, $$

$$ {}_x\mathcal{D}_L^{\beta}\, u(x,t) = \frac{1}{\Gamma(2-\beta)} \frac{d^2}{dx^2} \int_x^L \frac{u(\eta,t)}{(\eta-x)^{\beta-1}} d\eta $$

Caputo and Riemann-Liouville fractional derivatives

Caputo fractional derivative for $\alpha \in (0,1)$

$$ {}_0^C\mathcal{D}_t^{\alpha}\, u(x,t) = \frac{1}{\Gamma(1-\alpha)} \int_0^t (t-\eta)^{-\alpha} \frac{\partial u(x,\eta)}{\partial \eta} d\eta. $$

Riemann-Liouville fractional derivatives for $\beta \in (1,2)$

$$ {}_0\mathcal{D}_x^{\beta}\, u(x,t) = \frac{1}{\Gamma(2-\beta)} \frac{d^2}{dx^2} \int_0^x \frac{u(\eta,t)}{(x-\eta)^{\beta-1}} d\eta, $$

$$ {}_x\mathcal{D}_L^{\beta}\, u(x,t) = \frac{1}{\Gamma(2-\beta)} \frac{d^2}{dx^2} \int_x^L \frac{u(\eta,t)}{(\eta-x)^{\beta-1}} d\eta $$

The definition of fractional-order derivatives can be found in

✓ I. Podlubny, Fractional Differential Equations, Vol. 198, Academic Press, San-Diego, CA (1999)

✓ M. D. Ortigueira, J. A. Machado Tenreiro, What is a fractional derivative?, J. Comput. Phys., 293 (2015): 4-13.

Outline

Time-marching scheme

The $L2\text{-}1_\sigma$ formula[c] and weighted and shifted Grünwald difference[d] (WSGD) method are employed. Then the time-marching scheme is

$$\begin{cases} h^\beta \sum_{s=0}^{j} c_{j-s}^{(\alpha,\sigma)} \left(\mathbf{u}^{s+1} - \mathbf{u}^{s}\right) = K_N \mathbf{u}^{j+\sigma} + h^\beta \mathbf{f}^{j+\sigma}, \\ u_i^0 = u_0(x_i). \end{cases}$$

[c] A. A. Alikhanov. "A new difference scheme for the time fractional diffusion equation". In: J. Comput. Phys. 280 (2015), pp. 424–438.

[d] W. Tian, H. Zhou, and W. Deng. "A class of second order difference approximations for solving space fractional diffusion equations". In: Math. Comp. 84 (2015), pp. 1703–1727.

Time-marching scheme

The $L2\text{-}1_\sigma$ formula[c] and weighted and shifted Grünwald difference[d] (WSGD) method are employed. Then the time-marching scheme is

$$\begin{cases} h^\beta \sum_{s=0}^{j} c_{j-s}^{(\alpha,\sigma)} \left(\mathbf{u}^{s+1} - \mathbf{u}^s\right) = K_N \mathbf{u}^{j+\sigma} + h^\beta \mathbf{f}^{j+\sigma}, \\ u_i^0 = u_0(x_i). \end{cases}$$

The stability and convergence of the proposed second-order accurate numerical scheme are discussed in

- ✓ Y.-L. Zhao, T.-Z. Huang, et al., A fast second-order implicit difference method for time-space fractional advection-diffusion equation, Numer. Func. Anal. Opt., 41 (2019) 257-293.

[c] A. A. Alikhanov. "A new difference scheme for the time fractional diffusion equation". In: J. Comput. Phys. 280 (2015), pp. 424–438.

[d] W. Tian, H. Zhou, and W. Deng. "A class of second order difference approximations for solving space fractional diffusion equations". In: Math. Comp. 84 (2015), pp. 1703–1727.

Time-marching scheme

$K_N = e_1 G_\beta + e_2 G_\beta^T$ and the Toeplitz matrix G_β is given

$$G_\beta = \begin{bmatrix} \omega_1^{(\beta)} & \omega_0^{(\beta)} & 0 & \cdots & 0 & 0 \\ \omega_2^{(\beta)} & \omega_1^{(\beta)} & \omega_0^{(\beta)} & 0 & \cdots & 0 \\ \vdots & \omega_2^{(\beta)} & \omega_1^{(\beta)} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & 0 \\ \omega_{N-2}^{(\beta)} & \ddots & \ddots & \ddots & \omega_1^{(\beta)} & \omega_0^{(\beta)} \\ \omega_{N-1}^{(\beta)} & \omega_{N-2}^{(\beta)} & \cdots & \cdots & \omega_2^{(\beta)} & \omega_1^{(\beta)} \end{bmatrix} \in \mathbb{R}^{(N-1)\times(N-1)}.$$

Outline

The BLTT system

From another point of view, if all time steps are stacked in a vector, we obtain the BLTT system:

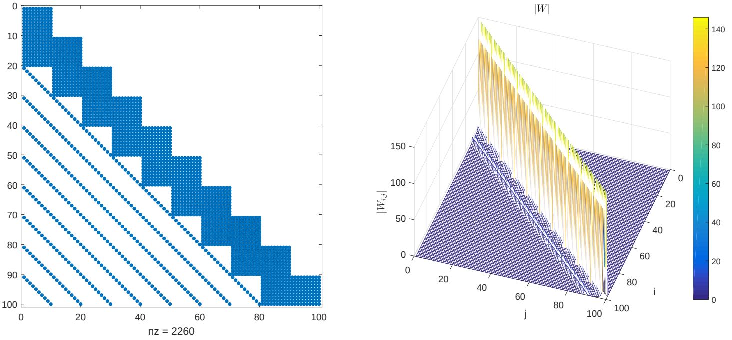
$$\begin{cases} A\mathbf{u}^1 = y_0, & (2a) \\ W\mathbf{u} = \mathbf{y}, & (2b) \end{cases}$$

where

$$\mathbf{u} = \begin{bmatrix} \mathbf{u}^2 \\ \mathbf{u}^3 \\ \vdots \\ \mathbf{u}^M \end{bmatrix}, \quad W = \begin{bmatrix} A_0 & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ A_1 & A_0 & \mathbf{0} & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ A_{M-3} & \ddots & \ddots & \ddots & \mathbf{0} \\ A_{M-2} & A_{M-3} & \cdots & \cdots & A_0 \end{bmatrix}.$$

Outline

Two preliminary information of W



(a) (b)

Figure: The sparsity pattern (Left) and decay elements (Right) of matrix $W \in \mathbb{R}^{100\times 100}$, when $M = N = 11$.

The B2T preconditioner

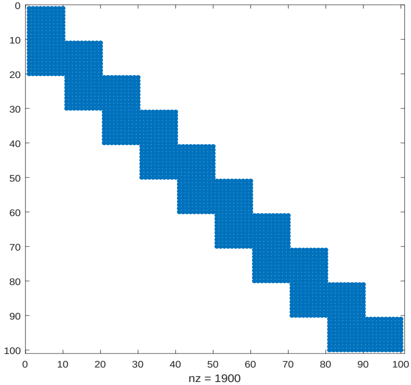
It implies that the main information of W clustered in the first two nonzero block diagonals. Inspired by this idea, a B2T preconditioner $P_W = \mathrm{tridiag}(A_1, A_0, \mathbf{0})$ is developed for the system (2b).

$$A_0 = h^{\beta} c_0^{(\alpha,\sigma)} I - \sigma K_N, \quad A_1 = h^{\beta}\left(c_1^{(\alpha,\sigma)} - c_0^{(\alpha,\sigma)}\right) I - (1-\sigma) K_N.$$

The B2T preconditioner

It implies that the main information of W clustered in the first two nonzero block diagonals. Inspired by this idea, a B2T preconditioner $P_W = \mathrm{tridiag}(A_1, A_0, \mathbf{0})$ is developed for the system (2b).

$$A_0 = h^\beta c_0^{(\alpha,\sigma)} I - \sigma K_N, \quad A_1 = h^\beta \left(c_1^{(\alpha,\sigma)} - c_0^{(\alpha,\sigma)} \right) I - (1-\sigma) K_N.$$



Several properties of P_W

Remark

P_W is a block Toeplitz matrix with Toeplitz block (BTTB), thus the storage requirement is of $\mathcal{O}(N)$.

Several properties of P_W

Remark

P_W is a block Toeplitz matrix with Toeplitz block (BTTB), thus the storage requirement is of $\mathcal{O}(N)$.

Theorem

P_W is nonsingular.

Theorem

The eigenvalues of the preconditioned matrix $P_W^{-1}W$ are all equal to 1. (PS: if P_W^{-1} can be exactly carried out.)

Computational aspects (1)

The efficient preconditioners can be used to improve the convergence of Krylov subspace solvers (KSSs). The main idea of preconditioning can be described as follows,

$$\begin{cases} P_W^{-1} W \mathbf{u} = P_W^{-1} \mathbf{y}, & \texttt{BiCGSTAB}, \\ W (P_W^{-1})_j \mathbf{s} = \mathbf{y}, \quad \mathbf{s} = P_W \mathbf{u}, & \texttt{FGMRES}. \end{cases}$$

At the moment, it knows that

- Most eigenvalues of $P_W^{-1} W$ or $W(P_W^{-1})_j$ will be clustered at 1.
- This spectral information makes the KSSs converge very fast when they are applied to solve the resultant linear systems.

Computational aspects (2)

As we know, when the preconditioned KSSs (such as BiCGSTAB and FGMRES; *refer to Saad's book, 2003*) are employed, it should solve the sub-systems in each iteration step:

$$\begin{cases} P_W \mathbf{z} = \mathbf{v}, & \texttt{BiCGSTAB}, \\ (P_W)_j \mathbf{z} = \mathbf{v}, & \texttt{FGMRES}. \end{cases}$$

Computational aspects (2)

As we know, when the preconditioned KSSs (such as BiCGSTAB and FGMRES; *refer to Saad's book, 2003*) are employed, it should solve the sub-systems in each iteration step:

$$\begin{cases} P_W \mathbf{z} = \mathbf{v}, & \texttt{BiCGSTAB}, \\ (P_W)_j \mathbf{z} = \mathbf{v}, & \texttt{FGMRES}. \end{cases}$$

Algorithm 1 Compute $\mathbf{z} = P_W^{-1}\mathbf{v}$

1: Reshape $\mathbf{v}$ into an $(N-1)\times M$ matrix $\check{V}$
2: Compute $\hat{\mathbf{b}}_1 = A_0^{-1}\check{V}(:,1)$
3: **for** $k = 2,\cdots,M$ **do**
4: $\quad \varphi = \check{V}(:,k) - A_1\hat{\mathbf{b}}_{k-1}$
5: $\quad \hat{\mathbf{b}}_k = A_0^{-1}\varphi$
6: **end for**
7: Stack $\hat{\mathbf{b}}_k$ $(k = 1,\cdots,M)$ in a vector $\mathbf{z}$

Outline

The Toeplitz inversion formula

When we compute $P_W^{-1}v$, it only needs to calculate $A_0^{-1}\tilde{v}$ (via the Toeplitz inversion formula) and $A_1\tilde{v}$ (via FFTs).

The Toeplitz inversion formula[e] requires to solve two Toeplitz systems at first:

$$\begin{cases} A_0\boldsymbol{\xi} = q_1, \\ A_0\boldsymbol{\eta} = q_{N-1}, \end{cases} \tag{3}$$

where q_1, q_{N-1} are the first and last columns of $(N-1)\times(N-1)$ identity matrix,

$$\boldsymbol{\xi} = [\xi_1, \cdots, \xi_{N-1}]^T \quad \text{and} \quad \boldsymbol{\eta} = [\eta_1, \cdots, \eta_{N-1}]^T.$$

[e] S. Lee, H.-K. Pang, and H.-W. Sun. "Shift-invert Arnoldi approximation to the Toeplitz matrix exponential". In: SIAM J. Sci. Comput. 32 (2010), pp. 774–792.

The Toeplitz inversion formula

Then the inverse of A_0 can be expressed as

$$A_0^{-1} = \frac{1}{2\xi_1}\left(C_1 S_1 + C_2 S_2\right), \tag{4}$$

where the first columns of (skew-)circulant matrices C_1, S_1, C_2 and S_2 are separately given

$$\boldsymbol{\xi}, \quad [\eta_{N-1}, -\eta_1, \cdots, -\eta_{N-2}]^T, \quad [\eta_{N-1}, \eta_1, \cdots, \eta_{N-2}]^T, \quad \boldsymbol{\xi}.$$

The Toeplitz inversion formula

Then the inverse of A_0 can be expressed as

$$A_0^{-1} = \frac{1}{2\xi_1}\left(C_1 S_1 + C_2 S_2\right), \tag{4}$$

where the first columns of (skew-)circulant matrices C_1, S_1, C_2 and S_2 are separately given

$$\boldsymbol{\xi}, \quad [\eta_{N-1}, -\eta_1, \cdots, -\eta_{N-2}]^T, \quad [\eta_{N-1}, \eta_1, \cdots, \eta_{N-2}]^T, \quad \boldsymbol{\xi}.$$

Eq. (4) has following decomposition

$$A_0^{-1} = \frac{1}{2\xi_1} F^* \left(\Lambda^{(1)} F \Omega^* F^* \Lambda^{(2)} + \Lambda^{(3)} F \Omega^* F^* \Lambda^{(4)}\right) F \Omega,$$

here $\Omega = \mathrm{diag}\left(1, (-1)^{-\frac{1}{N-1}}, \cdots, (-1)^{-\frac{N-2}{N-1}}\right)$.

The skew-circulant preconditioner

A skew-circulant preconditioner P_{sk} is designed to fast solve (3).

The skew-circulant preconditioner

A skew-circulant preconditioner P_{sk} is designed to fast solve (3).

$$P_{sk} = h^{\beta} c_0^{(\alpha,\sigma)} I - \sigma sk(K_N),$$

where $sk(K_N) = e_1 sk(G_\beta) + e_2 sk(G_\beta)^T$. The first column and row of $sk(G_\beta)$ are respectively:

$$\left[\omega_1^{(\beta)}, \omega_2^{(\beta)}, \cdots, \omega_{N-2}^{(\beta)}, -\omega_0^{(\beta)}\right]^T, \quad \left[\omega_1^{(\beta)}, \omega_0^{(\beta)}, -\omega_{N-2}^{(\beta)}, \cdots, -\omega_2^{(\beta)}\right].$$

The skew-circulant preconditioner

Theorem

The matrix P_{sk} is invertible.

Theorem

The generating function of the sequence $\{K_N\}_{N=2}^{\infty}$ is in the Wiener class.

Theorem

Suppose $0 < \hat{v} < h^{\beta} c_0^{(\alpha,\sigma)}$. Then for any $\varepsilon > 0$, there exists an $N' > 0$, such that for all $N - 1 > N'$, $P_{sk}^{-1} A_0 - I = U + V$, where $\mathrm{rank}(U) < 2N'$ and $\| V \|_2 < \varepsilon$.

The algorithms

All in all, the following two algorithms are integrated to evaluate $P_W^{-1}v$.

Algorithm 2 Compute $z = P_W^{-1}v$

1: Reshape v into an $(N-1)\times M$ matrix $\check{V}$
2: Compute $\hat{b}_1 = A_0^{-1}\check{V}(:,1)$ via Algorithm 3
3: **for** $k = 2, \cdots, M$ **do**
4: $\quad \varphi = \check{V}(:,k) - A_1\hat{b}_{k-1}$
5: $\quad \hat{b}_k = A_0^{-1}\varphi$ via Algorithm 3
6: **end for**
7: Stack $\hat{b}_k$ $(k = 1, \cdots, M)$ in a vector z

The algorithms

Algorithm 3 Compute $\tilde{z} = A_0^{-1} v$

1: Solve $A_0 \boldsymbol{\xi} = q_1$ via FGMRES/PBiCGSTAB with P_{sk}
Solve $A_0 \boldsymbol{\eta} = q_{N-1}$ via FGMRES/PBiCGSTAB with P_{sk}
2: $s_1 = [\eta_{N-1}, -\eta_1, \cdots, -\eta_{N-2}]^T$, $s_2 = [\eta_{N-1}, \eta_1, \cdots, \eta_{N-2}]^T$
3: $\Lambda^{(1)} = \texttt{fft}(\xi)$, $\Lambda^{(2)} = \hat{\Omega}^* .* \texttt{fft}(s_1)$,
$\Lambda^{(3)} = \texttt{fft}(s_2)$, $\Lambda^{(4)} = \hat{\Omega}^* .* \texttt{fft}(\xi)$
4: $\tilde{v} = \texttt{fft}(\hat{\Omega} .* v)$
5: $z_1 = \hat{\Omega}^* .* \texttt{ifft}(\Lambda^{(2)} .* \tilde{v})$, $z_2 = \hat{\Omega}^* .* \texttt{ifft}(\Lambda^{(4)} .* \tilde{v})$,
$z_3 = \Lambda^{(1)} .* \texttt{fft}(z_1)$, $z_4 = \Lambda^{(3)} .* \texttt{fft}(z_2)$
6: $\tilde{z} = \frac{1}{2\xi_1} \texttt{ifft}(z_3 + z_4)$

where $\hat{\Omega} = \left[1, (-1)^{-\frac{1}{N-1}}, \cdots, (-1)^{-\frac{N-2}{N-1}}\right]^T$.

Numerical examples

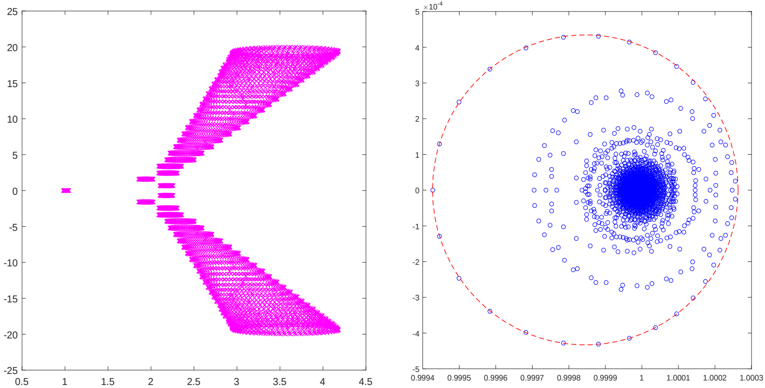
Example X Considering the equation (1) with diffusion coefficients $e_1 = 20$, $e_2 = 0.02$ and the exact solution is $u(x,t) = e^{2t}x^2(1-x)^2$.

Numerical examples

Table: Results of different iterative methods for Example X.

$(\alpha,\ \beta)$	$M = N$	BS Time	BFSM Time	SK2-PBiCGSTAB (Iter, Iter3)	SK2-PBiCGSTAB Time	S2-PBiCGSTAB (Iter, Iter3)	S2-PBiCGSTAB Time	SK2-FGMRES (Iter, Iter3)	SK2-FGMRES Time	S2-FGMRES (Iter, Iter3)	S2-FGMRES Time
(0.1, 1.1)	2^6	0.213	0.007	(4+2, 5)	0.014	(5+2, 5)	0.015	(6+5, 5)	0.020	(6+5, 6)	0.021
	2^7	3.469	0.044	(4+2, 5)	0.056	(5+2, 5)	0.057	(6+5, 5)	0.077	(6+6, 5)	0.092
	2^8	$>$ 20 min	0.234	(5+2, 5)	0.142	(5+2, 5)	0.144	(6+6, 5)	0.234	(6+7, 5)	0.273
	2^9	†	1.839	(5+2, 5)	0.995	(5+2, 5)	0.998	(6+7, 5)	1.912	(6+8, 5)	2.185
	2^{10}	†	19.839	(5+2, 5)	2.635	(5+2, 6)	2.672	(6+9, 5)	6.672	(6+10, 5)	7.480
(0.4, 1.7)	2^6	0.185	0.009	(4+2, 5)	0.014	(6+2, 6)	0.015	(6+5, 7)	0.021	(7+5, 6)	0.022
	2^7	2.993	0.043	(4+2, 5)	0.057	(6+2, 5)	0.058	(6+6, 6)	0.090	(7+5, 6)	0.078
	2^8	$>$ 20 min	0.235	(6+2, 5)	0.140	(6+2, 5)	0.141	(6+7, 6)	0.268	(7+6, 5)	0.233
	2^9	†	1.840	(6+3, 5)	1.486	(6+3, 5)	1.485	(6+7, 5)	1.906	(7+6, 5)	1.664
	2^{10}	†	19.838	(6+3, 5)	3.887	(6+3, 5)	3.878	(6+8, 5)	5.983	(7+7, 5)	5.248
(0.7, 1.4)	2^6	0.183	0.009	(4+3, 5)	0.020	(5+3, 5)	0.020	(6+6, 7)	0.024	(6+6, 8)	0.025
	2^7	2.969	0.040	(5+3, 5)	0.081	(5+3, 5)	0.083	(6+7, 6)	0.104	(7+8, 6)	0.119
	2^8	$>$ 20 min	0.238	(5+4, 5)	0.279	(5+4, 5)	0.279	(6+8, 6)	0.300	(7+9, 6)	0.342
	2^9	†	1.842	(5+4, 5)	1.975	(5+4, 5)	1.988	(6+10, 5)	2.688	(7+11, 6)	2.971
	2^{10}	†	19.847	(5+5, 5)	6.429	(5+5, 5)	6.526	(6+11, 5)	8.174	(7+14, 5)	10.540
(0.9, 1.9)	2^6	0.176	0.009	(4+2, 5)	0.015	(6+2, 5)	0.016	(5+5, 5)	0.200	(6+5, 5)	0.021
	2^7	2.950	0.043	(6+3, 5)	0.081	(6+3, 5)	0.082	(6+6, 5)	0.091	(6+6, 5)	0.092
	2^8	$>$ 20 min	0.209	(6+3, 5)	0.209	(6+3, 5)	0.214	(6+7, 5)	0.267	(6+7, 5)	0.271
	2^9	†	1.837	(6+4, 5)	1.968	(6+4, 5)	1.986	(6+8, 5)	2.164	(6+8, 5)	2.182
	2^{10}	†	19.853	(6+4, 5)	5.211	(6+4, 5)	5.276	(6+10, 5)	7.505	(6+10, 5)	7.447

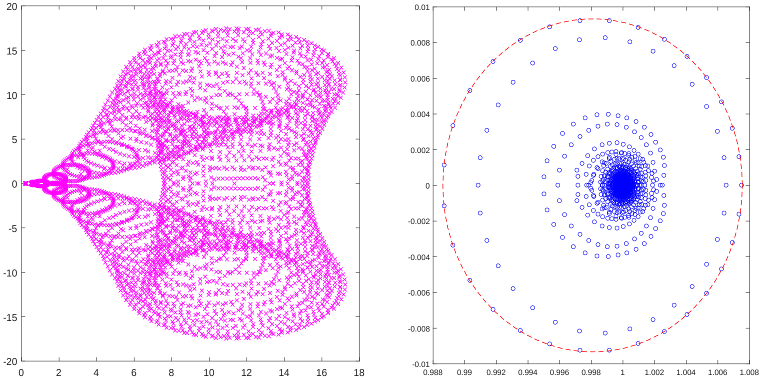
Spectra of W and $P_W^{-1}W$



(a) Eigenvalues of W (b) Eigenvalues of $P_W^{-1}W$

Figure: Spectra of W and $P_W^{-1}W$, when $M = N = 2^6$ and $(\alpha, \beta) = (0.1, 1.1)$ in Example X.

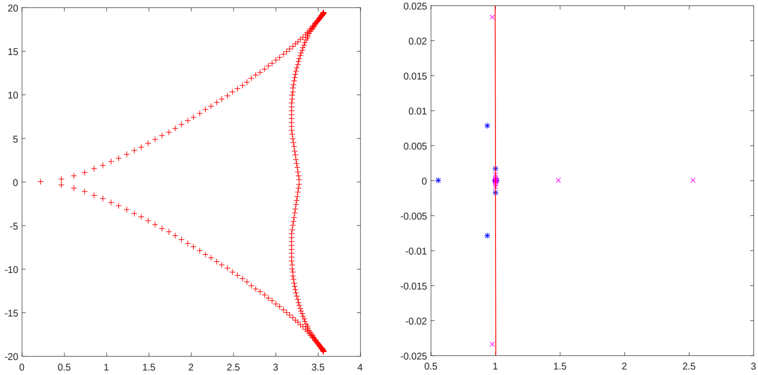
Spectra of W and $P_W^{-1}W$



(a) Eigenvalues of W (b) Eigenvalues of $P_W^{-1}W$

Figure: Spectra of W and $P_W^{-1}W$, when $M = N = 2^6$ and $(\alpha, \beta) = (0.7, 1.4)$ in Example X.

Spectra of A_0, $P_{sk}^{-1}A_0$ and $P_s^{-1}A_0$

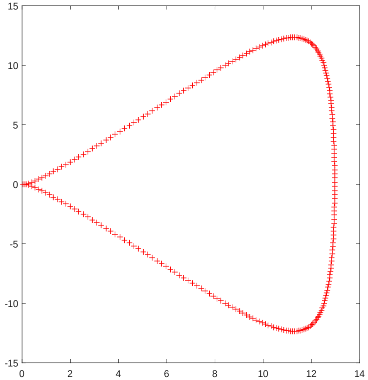


(a) Eigenvalues of A_0

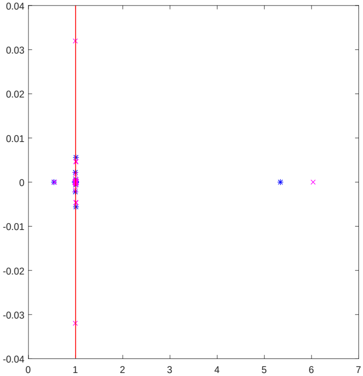
(b) Eigenvalues of $P_{sk}^{-1}A_0$ $(*)$ and $P_s^{-1}A_0$ $(\times)$

Figure: Spectra of A_0, $P_{sk}^{-1}A_0$ and $P_s^{-1}A_0$, when $M = N = 2^8$ and $(\alpha, \beta) = (0.1, 1.1)$ in Example X.

Spectra of A_0, $P_{sk}^{-1}A_0$ and $P_s^{-1}A_0$



(a) Eigenvalues of A_0



(b) Eigenvalues of $P_{sk}^{-1}A_0$ $(*)$ and $P_s^{-1}A_0$ $(\times)$

Figure: Spectra of A_0, $P_{sk}^{-1}A_0$ and $P_s^{-1}A_0$, when $M = N = 2^8$ and $(\alpha, \beta) = (0.7, 1.4)$ in Example X.

Summary

Conclusions

- A B2T preconditioner (P_W), whose storage is of $\mathcal{O}(N)$, is developed to solve the BLTT system.
- A new skew-circulant preconditioner (P_{sk}) is designed to efficiently compute $P_W^{-1}v$.
- Numerical experiments indicate that our skew-circulant preconditioner (P_{sk}) is slightly better than the Strang's circulant preconditioner (P_s).

Work(s) in progress

Work(s) in progress/ideas:

- Notice that the preconditioner P_W only compresses the temporal component. Hence, it is valuable to develop a preconditioner which compresses both the temporal and spatial components.
- P_W is not suitable for parallel computing. Thus, it is interesting to design an efficient and parallelizable preconditioner.
- Some other applications of P_{sk} are worth considering.

Our recent work about FDEs

1) Y.-L. Zhao, P.-Y. Zhu, X.-M. Gu, et al., A limited-memory block bi-diagonal Toeplitz preconditioner for block lower triangular Toeplitz system from time-space fractional diffusion equation, *J. Comput. Appl. Math.* 362 (2019) 99-115.
2) Y.-L. Zhao, P.-Y. Zhu, X.-M. Gu, et al., A preconditioning technique for all-at-once system from the nonlinear tempered fractional diffusion equation, to appear in *J. Sci. Comput.*, 2020. Available online at `https://arxiv.org/abs/1901.00635`
3) M. Li, C. Huang, Y.-L. Zhao, Fast conservative numerical algorithm for the coupled fractional Klein-Gordon-Schrödinger equation, *Numer. Algorithms*, 2019.
4) H.-Y. Jian, T.-Z. Huang, X.-M. Gu, et al., Fast implicit integration factor method for nonlinear space Riesz fractional reaction-diffusion equation, submitted to *J. Comput. Appl. Math.*, in revision, 18 Oct., 2019.

Questions or comments?

Questions or comments?

Many thanks for your attention!