Fractional Partial Differential Equation:
Numerical and Computational Issues

Hong Wang

Department of Mathematics, University of South Carolina
IBM Visiting Fellow, Division of Applied Mathematics, Brown University

Division of Applied Mathematics, Brown University
April 19, 2018

Acknowledgements

- Division of Applied Mathematics, Brown University
- The OSD/ARO MURI Grant W911NF-15-1-0562 and the National Science Foundation under Grant DMS-1620194

$$
\begin{aligned}
&\partial_t u - k_+(x,t)\,{}_a^{GL}D_x^{\alpha} u - k_-(x,t)\,{}_x^{GL}D_b^{\alpha} u = f, \quad x \in (a,b),\ t \in (0,T], \\
&u(a,t) = u(b,t) = 0,\ t \in [0,T], \quad u(x,0) = u_0(x),\ x \in [a,b].
\end{aligned} \tag{1}
$$

- $k_{\pm}$ are the left/right *variable* diffusivity coefficients (analytical means fail).
- The left/right Grünwald-Letnikov fractional derivatives of $1 < \alpha < 2$ are

$$
\begin{aligned}
{}_a^{GL}D_x^{\alpha} u(x,t) &:= \lim_{\varepsilon \to 0^+} \frac{1}{\varepsilon^{\alpha}} \sum_{l=0}^{\lfloor (x-a)/\varepsilon \rfloor} g_l^{(\alpha)} u(x - l\varepsilon, t), \\
{}_x^{GL}D_b^{\alpha} u(x,t) &:= \lim_{\varepsilon \to 0^+} \frac{1}{\varepsilon^{\alpha}} \sum_{l=0}^{\lfloor (b-x)/\varepsilon \rfloor} g_l^{(\alpha)} u(x + l\varepsilon, t) \\
{}_a^{GL}D_x^{2} u(x,t) &:= \partial_{xx} u(x,t) =: {}_x^{GL}D_b^{2} u(x,t).
\end{aligned} \tag{2}
$$

- $g_l^{(\alpha)} := (-1)^l \binom{\alpha}{l}$ with $\binom{\alpha}{l}$ being the fractional binomial coefficients.

- FPDEs have different math. & numer. features from integer-order PDEs.
- The implicit FDM obtained by truncating (2) is *unconditionally unstable*!
- The shifted FDM is *unconditionally stable* (Meerschaert & Tadjeran 04)

$$\frac{u_i^m - u_i^{m-1}}{\tau} - \frac{k_i^{+,m}}{h^\alpha}\sum_{l=0}^{i} g_l^{(\alpha)} u_{i-l+1}^m - \frac{k_i^{-,m}}{h^\alpha}\sum_{l=0}^{N-i+1} g_l^{(\alpha)} u_{i+l-1}^m = f_i^m \tag{3}$$

- The matrix form of the FDM

$$(\mathbf{I} + \tau \mathbf{A}^m)\mathbf{u}^m = \mathbf{u}^{m-1} + \tau \mathbf{f}^m, \tag{4}$$

$$a_{i,j}^m := -\frac{1}{h^\alpha}\begin{cases} \big(k_i^{+,m} + k_i^{-,m}\big) g_1^{(\alpha)} > 0, & j = i, \\ \big(k_i^{+,m} g_2^{(\alpha)} + k_i^{-,m} g_0^{(\alpha)}\big) < 0, & j = i-1, \\ \big(k_i^{+,m} g_0^{(\alpha)} + k_i^{-,m} g_2^{(\alpha)}\big) < 0, & j = i+1, \\ k_i^{+,m} g_{i-j+1}^{(\alpha)} < 0, & j < i-1, \\ k_i^{-,m} g_{j-i+1}^{(\alpha)} < 0, & j > i+1 \end{cases} \tag{5}$$

- The matrix A is full and has to be assembled in any traditional scheme.
- Direct solvers have $O(N^3)$ complexity per time step and $O(N^2)$ memory.
- Each time the mesh size and time step are refined by half, the computational work and memory requirement increase
  - 16 times and 4 times, respectively, for one-dimensional problems, or
  - 128 times and 16 times, respectively, for two-dimensional problems, or
  - 1024 times and 64 times, respectively, for three-dimensional problems.

- $g_l^{(\alpha)} := (-1)^l \binom{\alpha}{l}$ have the properties

$$\begin{aligned} & g_1^{(\alpha)} = -\alpha < 0, \quad 1 = g_0^{(\alpha)} > g_2^{(\alpha)} > g_3^{(\alpha)} > \cdots > 0, \\ & \sum_{l=0}^{\infty} g_l^{(\alpha)} = 0, \quad \sum_{l=0}^{m} g_l^{(\alpha)} < 0 \quad (m \geq 1), \\ & g_l^{(\alpha)} = \frac{\Gamma(l-\alpha)}{\Gamma(-\alpha)\Gamma(l+1)} = \frac{1}{\Gamma(-\alpha) l^{\alpha+1}} \Big(1 + O\Big(\frac{1}{l}\Big)\Big) \end{aligned} \tag{6}$$

  - $g_l^{(\alpha)}$, with $1 < \alpha < 2$, are not diagonally dominant, so the FPDE operator (and the direct FDM) does not have maximum principle.
  - Nevertheless, the shifted FDM has

$$\begin{aligned} & \Big(a_{i,i}^m - \sum_{j=1, j\neq i}^{N} |a_{i,j}^m|\Big) h^\alpha \\ & = -(k_i^{+,m} + k_i^{-,m}) g_1^{(\alpha)} - k_i^{+,m} \sum_{l=0, l\neq 1}^{i} g_l^{(\alpha)} - k_i^{-,m} \sum_{l=0, l\neq 1}^{N-i} g_l^{(\alpha)} \\ & > -(k_i^{+,m} + k_i^{-,m}) g_1^{(\alpha)} - (k_i^{+,m} + k_i^{-,m}) \sum_{l=0, l\neq 1}^{\infty} g_l^{(\alpha)} = 0. \end{aligned} \tag{7}$$

- The FDM (3) satisfies maximum principle, which yields stability and error estimate of the FDM in the L^∞ norm, assuming the solution is smooth.
- A heuristic explanation of the stability. Consider (1) with $u_t = 0$, $k_+ = 1$, $k_- = 0$, $f = 0$ and $(a, b) = (0, 1)$ (we use ${}_0^{GL}D_x^\alpha u = {}_0^{RL}D_x^\alpha u$)

$$D^2 {}_0I_x^{2-\alpha} u = 0,\ x \in (0,1), \quad u(0) = 0,\ u(1) = 1, \quad 1 < \alpha < 2, \Longrightarrow$$

$${}_0I_x^{2-\alpha} u = C_1 x + C_0, \Longrightarrow$$

$${}_0I_x u = {}_0I_x^{\alpha-1} {}_0I_x^{2-\alpha} u = {}_0I_x^{\alpha-1}(C_1 x + C_0) = C_1\ x^\alpha \Gamma(\alpha+1) + \frac{C_0\ x^{\alpha-1}}{\Gamma(\alpha)}.$$

where we have used

$${}_0I_x^\gamma x^\mu = \frac{\Gamma(\mu+1)}{\Gamma(\gamma+\mu+1)} x^{\gamma+\mu}, \quad 0 < \gamma < 1,\ \mu > -1 \tag{8}$$

Differentiating the equation and enforcing both the boundary conditions yields

$$u = x^{\alpha-1}, \quad x \in (0,1).$$

- Even the one-sided FDE requires both boundary conditions at $x = 0$ and $x = 1$ to uniquely determine the true solution.
- However, the directly truncated FDM yields a one-sided discretization, which is determined completely by the boundary condition at $x = 0$ and yields the trivial numerical solution $u_i = 0$ for $i = 1, 2, \ldots, N$. This is inconsistent with the FDE.
- The shifted FDM introduces at least one unknown in the other direction and so a two-way coupling, which has to be closed by both the boundary conditions. Hence, the shifted FDM is consistent with the FDE.
- This explains heuristically why the directly truncated FDM is unstable and the shifted FDM is stable.

Theorem

$$A^m = \big(\mathrm{diag}(d_i^{+,m})_{i=1}^N T^{\alpha,N} + \mathrm{diag}(d_i^{-,m})_{i=1}^N (T^{\alpha,N})^T\big)/h^\alpha, \tag{9}$$

$$T^{\alpha,N} := -\begin{bmatrix} g_1^{(\alpha)} & g_0^{(\alpha)} & 0 & \dots & 0 & 0 \\ g_2^{(\alpha)} & g_1^{(\alpha)} & g_0^{(\alpha)} & \ddots & \ddots & 0 \\ \vdots & g_2^{(\alpha)} & g_1^{(\alpha)} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & 0 \\ g_{N-1}^{(\alpha)} & \ddots & \ddots & \ddots & g_1^{(\alpha)} & g_0^{(\alpha)} \\ g_N^{(\alpha)} & g_{N-1}^{(\alpha)} & \dots & \dots & g_2^{(\alpha)} & g_1^{(\alpha)} \end{bmatrix}.$$

- (9) bridges the FPDE and the numerical linear algebra communities.

Theorem

$A^m v$ can be evaluated in $O(N \log N)$ operations in a lossless and matrix-free manner for any vector v, and A^m can be stored in $O(N)$ memory.

The matrix $T^{\alpha,N}$ is embedded into a $2N \times 2N$ circulant matrix $C^{\alpha,2N}$

$$C^{\alpha,2N} := \begin{bmatrix} T^{\alpha,N} & S^{\alpha,N} \\ S^{\alpha,N} & T^{\alpha,N} \end{bmatrix}, \quad S^{\alpha,N} := \begin{bmatrix} 0 & g_N^{(\alpha)} & \dots & \dots & g_3^{(\alpha)} & g_2^{(\alpha)} \\ 0 & 0 & g_N^{(\alpha)} & \dots & \ddots & g_3^{(\alpha)} \\ 0 & 0 & 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \dots & 0 & \ddots & 0 & g_N^{(\alpha)} \\ g_0^{(\alpha)} & 0 & \dots & 0 & 0 & 0 \end{bmatrix}.$$

- Let $c^{\alpha,2N}$ be the first column of $C^{\alpha,2N}$. Then $C^{\alpha,2N}$ can be decomposed as

$$C^{\alpha,2N} = F_{2N}^{-1} \operatorname{diag}(F_{2N} c^{\alpha,2N}) \, F_{2N} \tag{10}$$

- A fast matrix-vector multiplication $A^m v$ is formulated as follows
  - For any $v \in \mathbb{R}^N$, define v_{2N} by

$$v_{2N} = \begin{bmatrix} v \\ 0 \end{bmatrix}, \quad C^{\alpha,2N} v_{2N} = \begin{bmatrix} T^{\alpha,N} & S^{\alpha,N} \\ S^{\alpha,N} & T^{\alpha,N} \end{bmatrix} \begin{bmatrix} v \\ 0 \end{bmatrix} = \begin{bmatrix} T^{\alpha,N} v \\ S^{\alpha,N} v \end{bmatrix}. \quad (11)$$

  - $F_{2N} v_{2N}$ can be carried out in $O(N \log N)$ operations via FFT, so $C^{\alpha,2N} v_{2N}$ can be evaluated in $O(N \log N)$ operations.
  - The first N entries of $C^{\alpha,2N} v_{2N}$ yields $T^{\alpha,N} v$.
  - Similarly, $(T^{\alpha,N})^T v$ can be evaluated in $O(N \log N)$ operations.
  - $A^m v$ can be evaluated in $O(N \log N)$ operations.
- The fast algorithm is
  - matrix-free as it does not need to store A^m, but needs only to store $(d_i^{\pm,m})_{i=1}^N$ and $T^{\alpha,N}$, i.e.,$(3N+1)$ parameters.
  - exact as no compression is used.
  - non-intrusive.

$$
\begin{aligned}
&\partial_t u - k_{x,+}(x,y,t)\,{}_a^{GL}D_x^{\alpha} u - k_{x,-}(x,y,t)\,{}_x^{GL}D_b^{\alpha} u - k_{y,+}(x,y,t)\,{}_c^{GL}D_y^{\beta} u \\
&\quad - k_{y,-}(x,y,t)\,{}_y^{GL}D_d^{\beta} u = f(x,y,t), \quad (x,y) \in \Omega := \Pi_{i=1}^2 (a_i, b_i), \ t \in (0,T]; \\
&u(x,y,t) = 0, \ (x,y) \in \partial\Omega, \ t \in [0,T], \quad u(x,y,0) = u_o(x,y), \ (x,y) \in \overline{\Omega}.
\end{aligned} \tag{12}
$$

- An FDM for $1 \le i \le N_1$, $1 \le j \le N_2$ and $1 \le m \le N_t$

$$
\begin{aligned}
&\frac{u_{i,j}^m - u_{i,j}^{m-1}}{\tau} - \frac{k_{x,i,j}^{+,m}}{h_1^{\alpha}} \sum_{l=0}^{i} g_l^{(\alpha)} u_{i-l+1,j}^m - \frac{k_{x,i,j}^{-,m}}{h_1^{\alpha}} \sum_{l=0}^{N_1-i+1} g_l^{(\alpha)} u_{i+l-1,j}^m \\
&- \frac{k_{y,i,j}^{+,m}}{h_2^{\beta}} \sum_{l=0}^{j} g_l^{(\beta)} u_{i,j-l+1}^m - \frac{k_{y,i,j}^{-,m}}{h_2^{\beta}} \sum_{l=0}^{N_2-j+1} g_l^{(\alpha)} u_{i,j+l-1}^m = f_{i,j}^m.
\end{aligned} \tag{13}
$$

- Let $N = N_1 N_2$. Introduce N-dimensional vectors u^m and f^m defined by

$$
\begin{aligned}
u^m &:= \left[u_{1,1}^m, \cdots, u_{N_1,1}^m, u_{1,2}^m, \cdots, u_{N_1,2}^m, \cdots, u_{1,N_2}^m, \cdots, u_{N_1,N_2}^m\right]^T, \\
f^m &:= \left[f_{1,1}^m, \cdots, f_{N_1,1}^m, f_{1,2}^m, \cdots, f_{N_1,2}^m, \cdots, f_{1,N_2}^m, \cdots, f_{N_1,N_2}^m\right]^T.
\end{aligned} \tag{14}
$$

- The FDM (13) can be expressed in the matrix form (4).

- An ADI algorithm was developed to solve the FDM (13), first solving the x part as N_2 one-dimensional systems and then solving the y part as N_1 one-dimensional systems. Its computational complexity is $O(N^2)$.
- Solving (13) by the fast 1D FDM with ADI (W. & Wang 11) results in a computational complexity $O(N \log N)$ per matrix-vector multiplication.
- Strength and weakness of ADI
  - $+$ Reduce multidimensional problems to one-dimensional systems.
  - $+$ Easy to implement, avoid multidimensional structure of A^m.
  - $-$ It has proved stability and convergence if the FD operators in the x- and y-directions commute, not satisfied by general coefficients.
  - $-$ It is lossy and has higher regularity requirement.

- $A^{m,x}$ accounts for the coupling of all the nodes in the x direction
  - $A^{m,x}$ is block-diagonal with full diagonal blocks.
  - Each diagonal block $A_j^{m,x}$ is identical to that for a 1D problem
  - $A^{m,x}\mathsf{v}$ can be evaluated in $N_2 O(N_1 \log N_1) = O(N \log N)$ operations.
  - $A^{m,x}$ can be stored in $N_2 O(N_1) = O(N)$ memory.
- $A^{m,y}$ accounts for the coupling of all the nodes in the y direction.
  - $A^{m,y}$ is a full block matrix with sparse matrix blocks.
  - We prove that $A^{m,y}$ is block-Toeplitz-circulant-block

$$
\begin{aligned}
A^{m,y} &= \Big[K_+^{m,y}\big(T^{\beta,N_2} \otimes I_{N_1}\big) + K_-^{m,y}\big((T^{\beta,N_2})^T \otimes I_{N_1}\big)\Big]/h_2^{\beta},\\
K_+^{m,y} &:= \mathrm{diag}\Big(\Big\{\mathrm{diag}\big(\{k_{y,i,j}^{+,m}\}_{i=1}^{N_1}\big)\Big\}_{j=1}^{N_2}\Big),\\
K_-^{m,y} &:= \mathrm{diag}\Big(\Big\{\mathrm{diag}\big(\{k_{y,i,j}^{-,m}\}_{i=1}^{N_1}\big)\Big\}_{j=1}^{N_2}\Big).
\end{aligned}
\tag{15}
$$

  - $A^{m,y}$ can be stored in $O(N)$ memory and $A^{m,y}\mathsf{v}$ can be evaluated in $O(N \log N)$ operations in a lossless and matrix-free manner.

- Let S^{β,N_2} be Toeplitz matrices of order N_2 for T^{β,N_2}, as in 1D. Introduce

$$\begin{aligned} C^{\beta,2N} &:= \begin{bmatrix} T^{\beta,N_2} \otimes I_{N_1} & S^{\beta,N_2} \otimes I_{N_1} \\ S^{\beta,N_2} \otimes I_{N_1} & T^{\beta,N_2} \otimes I_{N_1} \end{bmatrix}, \\ C^{\beta,2N} v_{2N} &= \begin{bmatrix} (T^{\beta,N_2} \otimes I_{N_1}) v \\ (S^{\beta,N_2} \otimes I_{N_1}) v \end{bmatrix}, \quad v_{2N} := \begin{bmatrix} v \\ 0 \end{bmatrix}, \quad \forall v \in \mathbb{R}^N. \end{aligned} \tag{16}$$

- Let $c^{\beta,2N}$ be the first column vector of $C^{\beta,2N}$, $F_{2N_2} \otimes F_{N_1}$ be the 2D Fourier transform matrix, and $\hat{c}^{\beta,2N}$ be the Fourier transform of $c^{\beta,2N}$

$$\begin{aligned} \hat{c}^{\beta,2N} &:= \big(F_{2N_2} \otimes F_{N_1}\big) c^{\beta,2N}, \\ C^{\beta,2N} &= \big(F_{2N_2} \otimes F_{N_1}\big)^{-1} \mathrm{diag}\big(\hat{c}^{\beta,2N}\big) \big(F_{2N_2} \otimes F_{N_1}\big). \end{aligned} \tag{17}$$

    - $\big(F_{2N_2} \otimes F_{N_1}\big) v_{2N}$ can be performed in $O(N \log N)$ operations via FFT.
    - (17) shows that $C^{\beta,2N} v_{2N}$ can be evaluated in $O(N \log N)$ operations.
    - (16) shows that $A^{m,y} v$ can be performed in $O(N \log N)$ operations!

- In the numerical experiments the data are given as follows
  - $k_{x,\pm}(x,y,z,t) = k_{y,\pm}(x,y,z,t) = k_{z,\pm}(x,y,z,t) = K = 0.005$
  - $f = 0$, $\alpha = \beta = \gamma = 1.8$, $\Omega = (-1,1)^3$, $[0,T] = [0,1]$.
  - The true solution is expressed via the inverse Fourier transform

$$
\begin{aligned}
u(x,y,z,t) = \frac{1}{\pi}\int_0^\infty & e^{-2K|\cos(\frac{\pi\alpha}{2})|(t+0.5)\xi^\alpha}\cos(\xi x)d\xi \\
\times\frac{1}{\pi}\int_0^\infty & e^{-2K|\cos(\frac{\pi\beta}{2})|(t+0.5)\eta^\beta}\cos(\eta y)d\eta \\
\times\frac{1}{\pi}\int_0^\infty & e^{-2K|\cos(\frac{\pi\gamma}{2})|(t+0.5)\zeta^\gamma}\cos(\zeta z)d\zeta.
\end{aligned}
$$

  - The initial condition $u_o(x,y,z)$ is chosen to be $u(x,y,z,0)$.
- The Meerschaert & Tadjeran FDM and the fast FDM implemented in Fortran 90 on a workstation of 120 GB of memory.

Table: The CPU of the FDM and fast FDM

$h = \Delta t$	# of nodes	The FDM	The fast FDM
2^{-3}	4,096	1h 4m 26s	0.58s
2^{-4}	32,768	2 months 25d 9h 12m	5.74s
2^{-5}	262,144	N/A	1m 6s
2^{-6}	2,097,152	N/A	14m 22s
2^{-7}	16,777,216	N/A	3h 49m 56s
2^{-8}	134,217,728	N/A	3days 3h 18m 52s

- It would take the regular FDM at least years of CPU times on state of the art supercomputers to finish the simulation, if the computer has enough memory.
- Parallelization was used in measuring the peak performance of supercomputers. The nonlocal nature of FPDEs makes the communications in the simulations global, which further increases the work clock time of the FDM simulations.

- The fast matrix-vector multiplication is based on (9) (or its multi-D version).
  - The Toeplitz structure of $T^{\alpha,N}$ $\sim$ the translation invariance of the fractional difference operator (3) $\sim$ the translation invariance of FPDE operator (2) $\sim$ stationary increments of underlying Lévy process.
  - The impact of the variable $k_{\pm}(x,t)$ $\sim$ variable volatility in the variable-coefficient SDE, which are not translation invariant, is reflected in the non-Toeplitz diagonal matrices $K_{\pm}^{m}$.
- The FDM (3) has only first-order accuracy in space and time. High-order FDMs, finite element methods (FEMs) and finite volume methods (FVMs) were developed for sFPDEs in the literature and the discrete operators are also translation invariant, so fast solvers can also be developed.
- The FDM, FEM and FVM operators are translation invariant if FPDE operators are discretized on structured (e.g. uniform or graded) meshes.
  - $+$ lossless, matrix-free and $O(N \log N)$ matrix-vector multiplication
  - $-$ restrictive on partitions

- There has been a lot of works in the literature on fast numerical methods for nonlocal problems, including the fast multipole method (FMM) (Greengard & Rokhlin 1987), the hierarchical (H-) matrix method (Hackbusch 1999) and the randomized matrix method (Halko et al 11).
- Many were extended to FPDEs (including but not limited to):
  - Use H-matrix approach to compress the stiffness matrix to arbitraty accuracy by a banded matrix + low-rank matrices, and multigrid to solve the approximate system (Ainsworth et al 17, Zhao et al 17)
    + $O(N \log N)$ computational complexity on general partition.
    − lossy, strongly heterogeneous coefficients with high uncertainty?
  - Use the approximate system as a preconditioner (Li et al, on going)
    + $O(N \log N)$ lossless on general partition. The approximate system seems to be an optimal preconditioner
    − $O(N^2)$ computational complexity on a general partition.
  - A low-rank approximation to off-diagonal blocks coupling different subdomains on a piecewise-structured partition (Jia & W. 15).

$$\begin{aligned}\partial_t u - k_{x,+}(x,y,t)\ {}_{a_1(y)}\mathbb{D}_x^{p_1(\alpha)} u - k_{x,-}(x,y,t)\ {}_x\mathbb{D}_{b_1(y)}^{p_1(\alpha)} u \\ -k_{y,+}(x,y,t)\ {}_{a_2(x)}\mathbb{D}_y^{p_2(\beta)} u - k_{y,-}(x,y,t)\ {}_y\mathbb{D}_{b_2(x)}^{p_2(\beta)} u = f(x,y,t), \\ (x,y) \in \Omega,\ t \times (0,T], \\ u(x,y,0) = u_0(x,y),\ (x,y) \in \Omega, \quad u(x,y,t) = 0, \quad (x,y) \in \partial\Omega,\ t \in [0,T].\end{aligned} \tag{18}$$

- Ω is a bounded convex domain. $a_1(y)$ and $b_1(y)$ refer to the left and right boundary of Ω at given y, and similarly $a_2(x)$ and $b_2(x)$.
- ${}_{a_1(y)}\mathbb{D}_x^{p_1(\alpha)}$ and ${}_x\mathbb{D}_{b_1(y)}^{p_1(\alpha)}$ (and ${}_{a_2(x)}\mathbb{D}_y^{p_2(\beta)}$ and ${}_y\mathbb{D}_{b_2(x)}^{p_2(\beta)}$) are defined by

$$\begin{aligned}{}_{a_1(y)}\mathbb{D}_x^{p_1(\alpha)} u(x,y,t) &:= \int_1^2 p_1(\alpha)\ {}_{a_1(y)}D_x^{\alpha} u(x,y,t) d\alpha, \\ {}_x\mathbb{D}_{b_1(y)}^{p_1(\alpha)} u(x,y,t) &:= \int_1^2 p_1(\alpha)\ {}_xD_{b_1(y)}^{\alpha} u(x,y,t) d\alpha.\end{aligned} \tag{19}$$

- $p_1(\alpha)$ (or $p_2(\beta)$) refers to the PDE counting for the integrated impact of the fractional derivatives in the x (and y) direction with respect to α (or β).

$$
\begin{aligned}
{}_{a_1(y)}D_x^{\alpha}u(x,y,t) &:= \lim_{\varepsilon\to 0^+}\frac{1}{\varepsilon^{\alpha}}\sum_{l=0}^{\lfloor (x-a_1(y))/\varepsilon\rfloor} g_l^{(\alpha)}u(x-l\varepsilon,y,t),\\
{}_xD_{b_1(y)}^{\alpha}u(x,y,t) &:= \lim_{\varepsilon\to 0^+}\frac{1}{\varepsilon^{\alpha}}\sum_{l=0}^{\lfloor (b_1(y)-x)/\varepsilon\rfloor} g_l^{(\alpha)}u(x+l\varepsilon,y,t),
\end{aligned}
\tag{20}
$$

- The lower/upper limits of the fractional derivatives may depend on y (or x).
- For $p_1(\alpha)=\delta(\alpha)$ and $p_2(\beta)=\delta(\beta)$, the distributed order sFPDE (19) reduces to the conventional FPDE in the convex domain Ω.
- For $p_1(\alpha)=\sum_{l=1}^{l_1}\omega_l^x\delta(\alpha_l)$ and $p_2(\beta)=\sum_{l=1}^{l_2}\omega_l^y\delta(\beta_l)$, the distributed order FPDE (19) reduces to a multi-term sFPDE in Ω.
- Subsequently, we focus on the sFPDE in the convex domain Ω.

- Let a_1 (or b_1) be the left (or right) most boundary point of Ω, a_2 and b_2 defined similarly. Then $\Omega \subset [a_1, b_1] \times [a_2, b_2]$.
- Let $\overline{\Omega}_h := \overline{\Omega} \cap \{(x_i, y_j)\}_{0 \le i \le N_1+1; 0 \le j \le N_2+1}$.

$$\begin{aligned} \Pi_h &:= \big\{(i,j) : i_1(j)+1 \le i \le i_2(j)-1,\ 1 \le j \le N_2\big\}; \\ N &:= |\Pi_h| = \sum_{j=1}^{N_2} n_j, \qquad n_j := i_2(j) - i_1(j) - 1. \end{aligned} \tag{21}$$

- An FDM is defined by each node $(i,j) \in \Pi_h$ as follows

$$\begin{aligned} &\frac{u_{i,j}^m - u_{i,j}^{m-1}}{\tau} - \Big[\frac{k_{x,i,j}^{+,m}}{h_1^\alpha} \sum_{l=0}^{i-i_1(j)+1} g_l^{(\alpha)} u_{i-l+1,j}^m + \frac{k_{x,i,j}^{-,m}}{h_1^\alpha} \sum_{l=0}^{i_2(j)-i+1} g_l^{(\alpha)} u_{i+l-1,j}^m\Big] \\ &\quad - \Big[\frac{k_{y,i,j}^{+,m}}{h_2^\beta} \sum_{l=0}^{j-j_1(i)+1} g_l^{(\beta)} u_{i,j-l+1}^m + \frac{k_{y,i,j}^{-,m}}{h_2^\beta} \sum_{l=0}^{j_2(i)-j+1} g_l^{(\beta)} u_{i,j+l-1}^m\Big] = f_{i,j}^m. \end{aligned} \tag{22}$$

- The "boundary" nodes of the FDM do not necessarily lie on $\partial\Omega$ but their distances from $\partial\Omega$ are less than h_1 or h_2.
- We enforce the Dirichlet BC at the "boundary" nodes, which introduces an error of order $O(h)$ and retains the accuracy of the FDM.
- A^m is dense but is not in a tensor product form of Toeplitz-like matrices.
- We split $A^m = A^{m,x} + A^{m,y}$ and $A^{m,x}$ is still block diagonal but each diagonal block $A_j^{m,x}$ may have different size.
- Note any $\mathsf{v} \in \mathbb{R}^N$ can be expressed in the form

$$\mathsf{v} = \left[\mathsf{v}_1^T, \mathsf{v}_2^T, \cdots, v_{N_2}^T\right]^T, \ \mathsf{v}_j = \left[v_{i_1(j)+1,j}, \ldots, u_{i_2(j)-1,j}\right]^T, \ 1 \le j \le N_2.$$

Then $A^{m,x}\mathsf{v}$ can be evaluated in $O(N \log N)$ via the formula

$$A_\alpha^{m,x}\mathsf{v} = \left[\left(A_1^{m,x}\mathsf{v}_1\right)^T, \left(A_2^{m,x}\mathsf{v}_2\right)^T, \ldots, \left(A_{N_2}^{m,x}\mathsf{v}_{N_2}\right)^T\right]^T.$$

- The tensor-product decomposition of $A^{m,y}$ is no longer true.
- We use the symmetry of the fractional differential operators in the x and y directions and borrow the idea of the relabelling in the ADI. Algorithmically,
  - Let w denote the reindexing of the vector v by labeling the nodes in the y direction first
$$w = Pv \tag{23}$$
    where P represents the permutation matrix that maps v to w.
  - Let $B^{m,y}$ denote the analogue of $A^{m,y}$ that accounts for the spatial coupling by labelling the nodes in the y direction first. Then
$$A^{m,y} = P^T B^{m,y} P. \tag{24}$$
    We combine (23) and (24) to obtain
$$A^{m,y} v = P^T B^{m,y} w. \tag{25}$$

- The key points are as follows:
  - By labling the nodes in the y direction first, the stiffness matrix $B^{m,y}$ is block diagonal like $A^{m,x}$.
  - If we store *v* in the form of *w*, then $B^{m,y}$*w* can be evaluated in $O(N \log N)$ as $A^{m,x}$*v*.
  - In ADI the two labelings were used in solving two different families of subsystems.
  - We borrow the idea of ADI by using the two labelings in the matrix vector multiplication by A^m, but without splitting the scheme.
  - This boils down to storing *v* as a two-dimensional array corresponding to the indexing of the nodes (x_i, y_j).
  - Transforming *v* to *w* in (23) can be carried out simply by letting the index j goes first in the two-dimensional array storing *v* and vice versa.
- In summary, we can evaluate $A^{m,y}$*v* in $O(N \log N)$ operations in a lossless and matrix-free manner, by borrowing the idea of ADI of relabeling but without splitting the numerical scheme that may lead to a lossy evaluation.

- Consider the sFPDE (1) with fractional derivative BC

$$u(a,t)=0,\ \beta u(b,t)+\Big(k_+(x,t)\big)_a^{GL}D_x^{\alpha-1}u+k_-(x,t)_x^{GL}D_b^{\alpha-1}\Big)\Big|_{x=b}=g(t). \quad (26)$$

  - a fractional Neumann BC for $\beta=0$ or a fractional Robin BC for $\beta>0$.

$$\beta u_N^m+\frac{d_{+,N}^m}{h^{\alpha-1}}\sum_{k=0}^{N}g_k^{(\alpha-1)}u_{N-k}^m+\frac{d_{-,N}^m}{h^{\alpha-1}}g_0^{(\alpha-1)}u_N^m=g(t^m). \quad (27)$$

- $g_k^{(\alpha-1)}$ have the properties

$$\begin{gathered} g_0^{(\alpha-1)}=1,\quad -1<1-\alpha=g_1^{(\alpha-1)}<g_2^{(\alpha-1)}<g_3^{(\alpha-1)}<\cdots<0, \\ \sum_{k=0}^{\infty}g_k^{(\alpha-1)}=0,\quad \sum_{k=0}^{m}g_k^{(\alpha-1)}>0,\quad m\geq 1. \end{gathered} \quad (28)$$

- $g_k^{(\alpha-1)}$ have M matrix properties, so the discretization of the fractional BC has maximum principle. No shift!

$$a_{i,j} := \frac{1}{h^\alpha} \begin{cases} -\big(k_{+,i} + k_{-,i}\big) g_1^{(\alpha)}, & 1 \le i = j \le N-1; \\ -\big(k_{+,i} g_2^{(\alpha)} + k_{-,i} g_0^{(\alpha)}\big), & j = i-1, 2 \le i \le N-1; \\ -\big(k_{+,i} g_0^{(\alpha)} + k_{-,i} g_2^{(\alpha)}\big), & j = i+1, 1 \le i \le N-1; \\ -k_{+,i} g_{i-j+1}^{(\alpha)}, & 1 \le j \le i-2, 3 \le i \le N-1; \\ -k_{-,i} g_{j-i+1}^{(\alpha)}, & 3 \le j \le N, 1 \le i \le N-2; \\ \dfrac{k_{-,N} g_{N-j}^{(\alpha-1)} h}{\tau}, & 1 \le j \le N-1, i = N; \\ \dfrac{\beta h^\alpha + \big(k_{-,N} + k_{+,N}\big) g_0^{(\alpha-1)} h}{\tau}, & i = j = N. \end{cases} \tag{29}$$

- The first $N-1$ row are diagonally dominant as they are similar to those in the case of the Dirichlet BC (having one more column)

$$a_{i,i} - \sum_{j=1, j\neq i}^{N} \big|a_{i,j}\big| > 0, \quad 1 \le i \le N-1.$$

- The last row requires extra study as it comes from the discretization of fractional derivative BC and so has a different structure.

$$
\begin{aligned}
&h^{\alpha}\Big[a_{N,N} - \sum_{j=1}^{N-1} \big|a_{N,j}\big|\Big] \\
&= \frac{h}{\tau}\Big[(k_{+,N} + k_{-,N})g_0^{(\alpha-1)} + k_{+,N}\sum_{l=1}^{N-1} g_l^{(\alpha-1)} + \beta h^{\alpha-1}\Big] \\
&\geq \frac{h}{\tau}\Big[\big(k_{+,N} + k_{-,N}\big)\sum_{l=0}^{N-1} g_l^{(\alpha-1)} + \beta h^{\alpha-1}\big) > 0.
\end{aligned}
\tag{30}
$$

- The discretization of the fractional derivative BC is diagonally dominant.
- A is strongly diagonally dominant M-matrix.
- The numerical scheme determines a unique solution (no extra condition needed to enforce the uniquenes of the solution).

- The stiffness matrix A can be expressed in a block form

$$A = \begin{bmatrix} A_{N-1,N-1} & A_{N-1,N} \\ A_{N,N-1}^T & a_{N,N} \end{bmatrix}.$$

  - $A_{N-1,N-1}$ is the stiffness matrix for the interior nodes, hence the decomposition (9) for the Dirichlet BC is still valid.
  - Matrix-vector multiplication by $A_{N-1,N-1}$ is done in $O(N \log N)$.
  - The remaining is at most rank two. Hence, a matrix-free, lossless, fast matrix-vector multiplication by A can be carried out in $O(N \log N)$.

$$-D\big(K(x)\big(\theta\ {}^{C,l}_{a}D_x^{1-\beta}u-(1-\theta)\ {}^{C,r}_{x}D_b^{1-\beta}u\big)\big)=f(x),\quad x\in(a,b),$$
$$u(a)=u_l,\quad u(b)=u_r,\qquad 0<\beta<1,\ \ 0\le\theta\le 1. \tag{31}$$

- derived from a local mass balance + a fractional Fick's law.
- θ is the weight of forward versus backward transition probability.
- The left- and right-fractional integrals, Caputo and Riemann-Liouville fractional derivatives are defined by

$$\begin{aligned}
{}_aI_x^{\beta}u(x) &= {}_aD_x^{-\beta}u(x) := \frac{1}{\Gamma(\beta)}\int_a^x (x-s)^{\beta-1}u(s)ds,\\
{}_xI_b^{\beta}u(x) &= {}_xD_b^{-\beta}u(x) := \frac{1}{\Gamma(\beta)}\int_x^b (s-x)^{\beta-1}u(s)ds,\\
{}^{C}_{a}D_x^{1-\beta}u &:= {}_aI_x^{\beta}Du, \qquad {}^{C}_{x}D_b^{1-\beta}u := -{}_xI_b^{\beta}Du,\\
{}^{RL}_{a}D_x^{1-\beta}u &:= D\ {}_aI_x^{\beta}u, \qquad {}^{RL}_{x}D_b^{1-\beta}u := -D\ {}_xI_b^{\beta}u.
\end{aligned} \tag{32}$$

- Conservative and non-conservative FPDEs are not equivalent for variable diffusivity coefficient problems, as the differentiation of the conservative form yields a fractional derivative of order $0 < 1 - \beta < 1$.
- Numerically, FEM/FVM are suited for conservative FPDEs, FDM is suited for nonconservative FPDEs.
- For many applications, local conservation property is crucial. In this case, FVM is preferred.
- A FEM naturally has second-order accuracy in space, without requiring a Richardson extrapolation as in FDM.

- A conventional derivation of the FVM
  - Let $a =: x_0 < x_1 < \ldots < x_i < \ldots < x_{N+1} := b$ be a (not necessarily uniform) partition and $x_{i-1/2} := (x_{i-1} + x_i)/2$.
  - Let $\{\phi_i\}_{i=1}^N$ be the hat functions with nodes x_i and $u = \sum_{j=1}^N u_j \phi_j$.
  - Let $\boldsymbol{u} := [u_1, u_2, \ldots, u_N]^T$, $\boldsymbol{f} := [f_1, f_2, \ldots, f_N]^T$, and $A := [a_{i,j}]_{i,j=1}^N$.
  - Integrating (**??**) over $(x_{i-1/2}, x_{i+1/2})$ yields

$$
\begin{aligned}
&A\boldsymbol{u} = \boldsymbol{f}, \quad f_i := \int_{x_{i-1/2}}^{x_{i+1/2}} f(x)dx, \quad 1 \le i, j \le N. \\
&a_{i,j} := \Big[K(x) \big(\theta \; {}_a^{C,l}D_x^{1-\beta} \phi_j - (1-\theta) \; {}_x^{C,r}D_b^{1-\beta} \phi_j \big) \Big]_{x=x_{i+1/2}}^{x=x_{i-1/2}}.
\end{aligned}
\tag{33}
$$

- The salient difference of the FVM from its integer-order analogue
  - $\operatorname{supp}\{\phi_j\} = [x_{j-1}, x_{j+1}]$. But ${}_a^{C,l}D_x^{1-\beta} \phi_j|_{x=x_{i+1/2}} \neq 0$ for $j \le i+1$ and ${}_x^{C,r}D_b^{1-\beta} \phi_j|_{x=x_{i-1/2}} \neq 0$ for $j \ge i-1$.
  - The stiffness matrix A is full, which requires $O(N^3)$ of operations to invert and $O(N^2)$ of memory to store.

Theorem

$$A = \big(K_- \; T_L^{\beta,N} + K_+ \; T_R^{\beta,N}\big)/(\Gamma(\beta+1)h^{1-\beta}),$$
$$K_\pm := \mathrm{diag}\big(\{K(x_{i\pm\frac{1}{2}})\}_{i=1}^N\big), \qquad T_L^{\beta,N} = (l_{i-j}), \quad T_R^{\beta,N} = (r_{i-j}) \tag{34}$$

with l_i and r_i being defined in (35). Hence, A can be stored in $O(N)$ memory and Av can be evaluated in $O(N\log N)$ operations in a lossless manner for any $v \in \mathbb{R}^N$.

- In fact, we need only to store $K(x_{i-\frac{1}{2}})$ for $i = 1, \ldots, N+1$, and l_i and r_i for $i = -N, \ldots, -1, 0, 1, \ldots, N$, which are totally $5N+3$ parameters.
- This represents a significant saving over the traditional storage of N^2 entries.

$$
l_i := \begin{cases} (1-\theta)\Big[\Big(-i-\frac{1}{2}\Big)^{\beta} + \Big(-i+\frac{3}{2}\Big)^{\beta} - 2\Big(-i+\frac{1}{2}\Big)^{\beta}\Big], & -N \le i \le -2, \\ (1-\theta)\Big[\Big(\frac{1}{2}\Big)^{\beta} + \Big(\frac{5}{2}\Big)^{\beta} - 2\Big(\frac{3}{2}\Big)^{\beta}\Big], & i = -1; \\ (1-\theta)\Big(\frac{3}{2}\Big)^{\beta} - (2-\theta)\Big(\frac{1}{2}\Big)^{\beta}, & i = 0; \\ (1+\theta)\Big(\frac{1}{2}\Big)^{\beta} - \theta\Big(\frac{3}{2}\Big)^{\beta}, & i = 1; \\ \theta\Big[2\Big(i-\frac{1}{2}\Big)^{\beta} - \Big(i+\frac{1}{2}\Big)^{\beta} - \Big(i-\frac{3}{2}\Big)^{\beta}\Big], & 2 \le i \le N. \end{cases}
$$

$$
r_i := \begin{cases} (1-\theta)\Big[2\Big(-i-\frac{1}{2}\Big)^{\beta} - \Big(-i-\frac{3}{2}\Big)^{\beta} - \Big(-i+\frac{1}{2}\Big)^{\beta}\Big], & -N \le i \le -2; \\ (2-\theta)\Big(\frac{1}{2}\Big)^{\beta} - (1-\theta)\Big(\frac{3}{2}\Big)^{\beta}, & i = -1; \\ \theta\Big(\frac{3}{2}\Big)^{\beta} - (1+\theta)\Big(\frac{1}{2}\Big)^{\beta}, & i = 0; \\ \theta\Big[\Big(\frac{5}{2}\Big)^{\beta} - 2\Big(\frac{3}{2}\Big)^{\beta} + \Big(\frac{1}{2}\Big)^{\beta}\Big], & i = 1; \\ \theta\Big[\Big(i+\frac{3}{2}\Big)^{\beta} - 2\Big(i+\frac{1}{2}\Big)^{\beta} + \Big(i-\frac{1}{2}\Big)^{\beta}\Big], & 2 \le i \le N; \end{cases}
\tag{35}
$$

- By (34), we need only to evaluate $T_L^{\beta,N} v$ (and $T_R^{\beta,N} v$) in a fast manner.
- The matrix $T_L^{\beta,N}$ can be embedded into a $2N \times 2N$ circulant matrix C_{2N}

$$C_{2N} := \begin{bmatrix} T_L^{\beta,N} & * \\ * & T_L^{\beta,N} \end{bmatrix}, \qquad v_{2N} = \begin{bmatrix} v \\ 0 \end{bmatrix}. \tag{36}$$

- A circulant matrix C_{2N} can be decomposed as

$$C_{2N} = F_{2N}^{-1} \operatorname{diag}(F_{2N} c_{2N}) \, F_{2N} \tag{37}$$

  F_{2N} is the Fourier transform matrix and c_{2N} is the first column of C_{2N}.
  - $C_{2N} v_{2N}$ and so Av can be evaluated in $O(N \log N)$ operations in a lossless and matrix-free manner.
- Both mass conservation property and accuracy of the FVM are retained.

- The fast matrix-vector multiplication reduces the computational cost per Krylov subspace iteration from $O(N^2)$ to $O(N \log N)$.
- For the steady-state FDE (**??**), the condition number of the stiffness matrix A is $\kappa(A) = O(h^{-(2-\beta)})$. Hence, the number of Krylov subspace iterations is $O(h^{-(1-\beta/2)}) = O(N^{1-\beta/2})$.
- This leads to an overall computational cost of $O(N^{2-\beta/2} \log N)$ even if a fast Krylov subspace iterative method is used.
- This calls for the development of an effective and efficient preconditioner.

- A superfast direct solver was developed for a symmetric and positive-definite (SPD) Toeplitz system (Ammar and Gragg 1988),
  - which inverts a full SPD Toeplitz system in $O(N \log^2 N)$ computations,
  - which does not always work very effectively especially for ill conditioned SPD Toeplitz systems.
- We developed a superfast preconditioner for the steady-state FDE (**??**) with $\theta = 1/2$ (W. & Du 13)

Theorem

$M := T_L^{\beta,N} + T_R^{\beta,N}$ *is a full SPD, Toeplitz matrix.*

- We just use M as a preconditioner for the FVM (34) as shown below.

- Outline of (a perturbation-based) proof: Let $K_0 := \mathrm{diag}\big(\{K(x_i)\}_{i=1}^N\big)$ and $K_\pm := \mathrm{diag}\Big(\big\{K\big(x_{i\pm\frac{1}{2}}\big)\big\}_{i=1}^N\Big)$. We have

$$\begin{aligned}
&\gamma(\beta)^{-1}K_0^{-1}A \\
&\quad = K_0^{-1}K_-\ T_L^{\beta,N} + K_0^{-1}K_+\ T_R^{\beta,N} \\
&\quad = K_0^{-1}\big[K_0 + (K_- - K_0)\big]T_L^{\beta,N} + K_0^{-1}\big[K_0 + (K_+ - K_0)\big]T_L^{\beta,N} \\
&\quad = M + K_0^{-1}\Big[(K_- - K_0)T_L^{\beta,N} + (K_+ - K_0)T_R^{\beta,N}\Big] \\
&\quad = M + O(h).
\end{aligned} \tag{38}$$

- M is a good preconditioner for the FVM

$$\big(K_0^{-1}K_-\ T_L^{\beta,N} + K_0^{-1}K_+\ T_R^{\beta,N}\big)\boldsymbol{u} = \gamma(\beta)^{-1}K_0^{-1}A\boldsymbol{u} = \gamma(\beta)^{-1}K_0^{-1}\boldsymbol{f}. \tag{39}$$

Numerical experiments by a superfast preconditioned fast CGS (W. & Du 2013)

- The data: $\beta = 0.2$, $\theta = 0.5$, $K(x) = \Gamma(1.2)(1+x)$, $u_l = u_r = 0$, $[a, b] = [0, 1]$,
- The true solution $u(x) = x^2(1-x)^2$, f is computed accordingly

	Gauss			CGS		
N	$\|u - u_G\|_{L^\infty}$	CPU(s)		$\|u - u_C\|_{L^\infty}$	CPU(s)	Itr. #
2^5	2.018×10^{-4}	0.000		2.018×10^{-4}	0.000	32
2^6	5.157×10^{-5}	0.000		5.157×10^{-5}	0.000	65
2^7	1.294×10^{-5}	0.000		1.294×10^{-5}	0.016	128
2^8	3.214×10^{-6}	0.047		3.214×10^{-6}	0.141	217
2^9	7.893×10^{-7}	0.500		7.893×10^{-7}	3.359	599
2^{10}	1.887×10^{-7}	7.797		1.886×10^{-7}	2 m 2 s	1,110
2^{11}	4.030×10^{-8}	2 m 38 s		4.047×10^{-8}	21 m 13 s	2,624
2^{12}	6.227×10^{-9}	24 m 29 s		7.468×10^{-8}	4 h 19 m	7,576
2^{13}	5.783×10^{-9}	3 h 27 m		N/A	$>$ 2 days	$>$ 20,000
	FCGS			PFCGS		
	$\|u - u_F\|_{L^\infty}$	CPU(s)	Itr. #	$\|u - u_S\|_{L^\infty}$	CPU(s)	Itr. #
2^5	2.018×10^{-4}	0.000	32	2.018×10^{-4}	0.000	6
2^6	5.157×10^{-5}	0.016	63	5.157×10^{-5}	0.000	5
2^7	1.294×10^{-5}	0.031	128	1.294×10^{-5}	0.000	5
2^8	3.214×10^{-6}	0.125	248	3.214×10^{-6}	0.006	5
2^9	7.893×10^{-7}	0.578	576	7.893×10^{-7}	0.016	5
2^{10}	1.886×10^{-7}	2.281	1,078	1.887×10^{-7}	0.047	5
2^{11}	4.037×10^{-8}	9.953	1,997	4.038×10^{-8}	0.078	5
2^{12}	1.587×10^{-8}	57.27	5,130	6.194×10^{-9}	0.188	5
2^{13}	2.372×10^{-8}	2 m 52 s	7,410	4.345×10^{-9}	0.391	5

- Use the numerical solutions by Gaussian elimination as a benchmark:
    - The conjugate gradient squared (CGS) method diverges, due to significant amount of round-off errors.
    - The fast CGS (FCGS) reduced the CPU time significantly, as the operations for each iteration is reduced from $O(N^2)$ to $O(N \log N)$.
        - The number of iterations is still $O(N^{1-\beta/2})$,
        - It is less accurate than Gaussian at fine meshes due to round-off errors.
    - The preconditioner M is optimal, so the preconditioned FCGS (PFCGS) has an overall computational cost of $O(N \log^2 N)$.
        - It significantly reduces round-off errors.
        - It generates more accurate solutions than Gaussian elimination.
        - It further reduces CPU time.
        - Although the superfast Toeplitz solver might have potential problems for ill-conditioned SPD Toeplitz systems as a direct solver, it seems to perform very well as a preconditioner for the FVM (34).

- Solutions to FDEs with smooth data and domain may have boundary layers, a numerical method that is discretized on a uniform mesh is not effective.
  - FDM is out of the question, as Grünwald-Letnikov derivative is inherently defined on a uniform mesh.
  - Riemann-Liouville and Caputo derivatives offer such flexibilities.
- Bebause of the nonlocal nature of FDEs, a numerical scheme discretized on an arbitrarily adaptively refined mesh
  - offers great flexbility and effective approximation property
  - offers possible advantage on its theoretical analysis
  - but destroys the structure of its stiffness matrix and so efficiency.
- Motivation: balancing flexibility and efficiency.
  - Wherever a refinement is needed, try to use a structured refinement.

Theorem

$$A = \Big[\mathrm{diag}(K^-)T_- + \mathrm{diag}(K^+)T_+\Big]\mathrm{diag}\big(\{h_i^{\beta-1}\}_{i=1}^m\big), \quad T_-, T_+ \ \text{Toeplitz}.$$

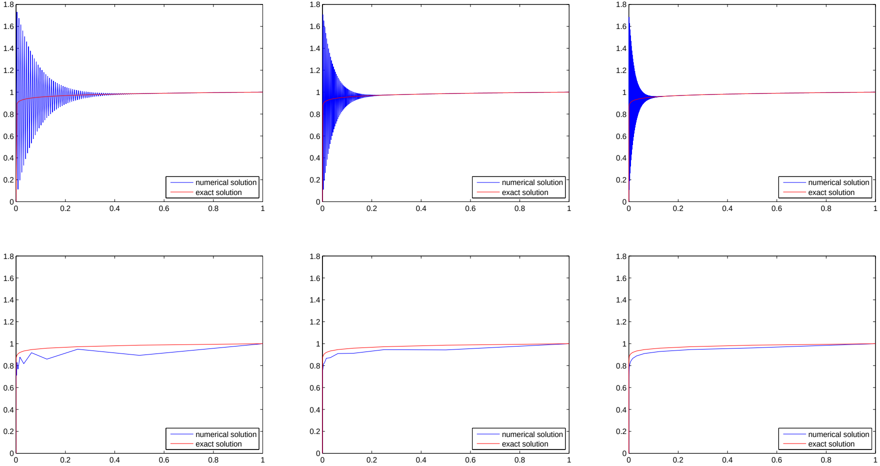
Av can be evaluated in $O(N \log N)$ computations in a lossless and matrix free manner, A can be stored in $O(N)$ memory.

$$\begin{aligned} D\big({}_0I_x^{\beta} Du\big) &= 0, \quad x \in (0,1), \\ u(0) = 0, \quad u(1) &= 1 \end{aligned}$$

Its solution $u(x) = x^{1-\beta}$ for $x \in (0,1)$.

	N	CPU	#of iterations
Gauss	256	0.640s	
	512	5.567s	
	1024	59s	
CGS	256	2.978s	256
	512	29s	512
	1024	403s	1024
FCGS	256	0.073s	256
	512	0.139s	512
	1024	0.391s	1024

Figure: First row: numerical solutions on a uniform mesh of $n = 256, 512, 1024$; Second row: numerical solutions on a geometrically refined mesh $n = 48, 64, 96$.



- Solutions to linear elliptic/parabolic FPDEs with smooth data and domain may have boundary layers, a uniform mesh is not effective.
  - FDM is out of the question, as Grünwald-Letnikov derivatives are inherently defined on uniform meshes.
  - Riemann-Liouville and Caputo derivatives offer such flexibilities.
- Bebause of the nonlocal nature of FDEs, a numerical scheme discretized on an arbitrarily adaptively refined mesh
  - offers great flexbility and effective approximation property
  - offers possible advantage on its theoretical analysis
  - destroys the structure of its stiffness matrix and so efficiency.
- Motivation: balancing flexibility and efficiency.
- A purely gridded mesh does not work as effectively.
- We propose to use a composite mesh that consists of
  - gridded mesh near the boundary,
  - a uniform mesh in most of the domain.

Outline of the structure of the stiffness matrix form

- We assume only a boundary layer near the left endpoint for simplicity.
- We begin by a uniform mesh of size h, and then use a gridded mesh on $[0, h]$ with $m+1$ nodes.
- Then A can be expressed in the following 3×3 matrix form

$$A = \begin{bmatrix} a_{1,1} & A_{1,l} & A_{1,r} \\ A_{l,1} & A_{l,l} & A_{l,r} \\ A_{r,1} & A_{r,l} & A_{r,r} \end{bmatrix}. \tag{40}$$

  - $A_{1,l}$, $A_{1,r}$, $A_{l,1}$, and $A_{r,1}$ are (row or column) vectors
  - The southeast 2×2 blocks require careful analysis.

Structure of the stiffness matrix

Theorem

The submatrices $A_{l,l}$ and $A_{r,r}$ can be decomposed as

$$\begin{aligned}
A_{l,l} &= \frac{1}{\Gamma(\beta+1)}\Big[\mathrm{diag}(K_l^-)\big(\gamma Q_l + (1-\gamma)Q_r\big) \\
&\qquad -\mathrm{diag}(K_l^+)\big(\gamma P_l + (1-\gamma)P_r\big)\Big]\mathrm{diag}\big(\{h_i^{\beta-1}\}_{i=1}^m\big), \\
A_{r,r} &= \frac{h^{\beta-1}}{\Gamma(\beta+1)}\Big[\mathrm{diag}(K_r^-)\big(\gamma S + (1-\gamma)R^T\big) - \mathrm{diag}(K_r^+)\big(\gamma R + (1-\gamma)S^T\big)\Big].
\end{aligned}$$

- P_l, P_r, Q_l, Q_r, R, and S are Toeplitz
- $A_{r,r}$ has the same form as before, since it is for a uniform mesh
- $A_{l,l}$ corresponds to a gridded mesh, and has an additional diagonal matrix (reflecting the impact of the mesh) multiplier on the right.

Theorem

The submatrices $A_{l,r}$ and $A_{r,l}$ can be decomposed as

$$\begin{aligned} A_{l,r} &= \frac{(1-\gamma)h^{\beta-1}}{\Gamma(\beta+1)}\big(\mathrm{diag}(K_l^-)E - \mathrm{diag}(K_l^+)D\big), \\ A_{r,l} &= \frac{\gamma}{\Gamma(\beta+1)}(\mathrm{diag}(K_r^-)H - \mathrm{diag}(K_r^+)G)\mathrm{diag}(\{h_i^{\beta-1}\}_{i=1}^m). \end{aligned}$$

- D, E, G, and H are non-Toeplitz full matrices. Their typical entries are of the form

$$\begin{aligned} d_{i,j} &= 2(j+1-3\cdot 2^{i-m-1})^\beta - (j-3\cdot 2^{i-m-1})^\beta - (j+2-3\cdot 2^{i-m-1})^\beta, \\ g_{i,j} &= \Big[2^{m-j+1}\Big(i+\frac{3}{2}\Big)-1\Big]^\beta - \frac{3}{2}\Big[2^{m-j+1}\Big(i+\frac{3}{2}\Big)-2\Big]^\beta \\ &\quad + \frac{1}{2}\Big[2^{m-j+1}(i+\frac{3}{2})-4\Big]^\beta. \end{aligned}$$

- Use a fractional binomial expansion, we have

$$
\begin{aligned}
D \approx & -2\binom{\beta}{2}[1,1,\dots,1]^T\left[\frac{1}{2^{2-\beta}},\frac{1}{3^{2-\beta}},\dots,\frac{1}{(n-1)^{2-\beta}}\right] \\
& -2\binom{\beta}{4}[1,1,\dots,1]^T\left[\frac{1}{2^{4-\beta}},\frac{1}{3^{4-\beta}},\dots,\frac{1}{(n-1)^{4-\beta}}\right] \\
& +18\binom{\beta}{3}[2^{-m},2^{-m+1},\dots,2^{-1}]^T\left[\frac{1}{2^{3-\beta}},\frac{1}{3^{3-\beta}},\dots,\frac{1}{(n-1)^{3-\beta}}\right] \\
& -108\binom{\beta}{4}[2^{-2m},2^{-2m+2},\dots,2^{-2}]^T\left[\frac{1}{2^{4-\beta}},\frac{1}{3^{4-\beta}},\dots,\frac{1}{(n-1)^{4-\beta}}\right].
\end{aligned}
$$

  - The matrices can be approximated by a finite sum of low-rank matrices.
  - The matrix-vector multiplication can be performed in $O(N)$ operations.

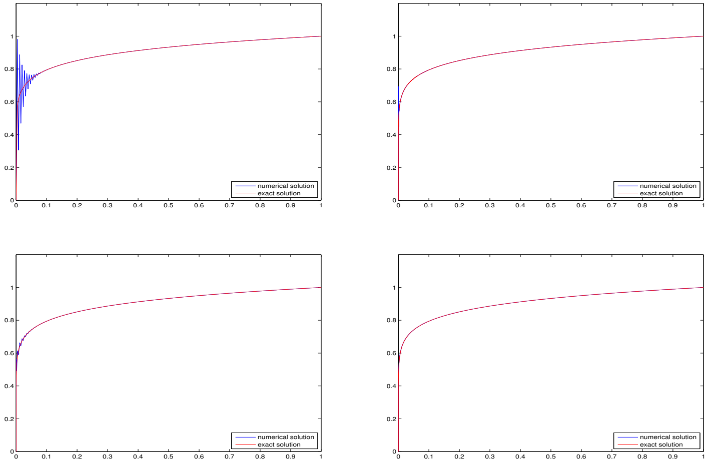
- Consider (**??**) with $K = 1$, $f = 0$, $\theta = 1$, $\beta = 0.9$, $u_l = 0$, $u_r = 1$, i.e.,

$$\begin{aligned} D\big({}_0I_x^\beta Du\big) &= 0, \quad x \in (0,1), \\ u(0) = 0, \quad u(1) &= 1 \end{aligned}$$

Its solution $u(x) = x^{1-\beta}$ for $x \in (0,1)$.

n	$\|u_n - u\|$	$\|u_{n,m} - u\|$	$\|u_{n,m} - u\|$
128	4.3546×10^{-1}	2.6805×10^{-1}, $m = 7$	2.0315×10^{-1}, $m = 11$
256	4.0630×10^{-1}	2.3336×10^{-1}, $m = 8$	1.3403×10^{-1}, $m = 16$
512	3.7909×10^{-1}	2.0315×10^{-1}, $m = 9$	8.2504×10^{-2}, $m = 22$
1024	3.5370×10^{-1}	1.7685×10^{-1}, $m = 10$	3.8488×10^{-2}, $m = 32$
8192	2.8730×10^{-1}	1.6668×10^{-1}, $m = 13$	N/A

Figure: First row: numerical solutions on a uniform mesh of n=256, 8192; Second row: numer. solns. on a composite mesh with $n = 256$ and $m = 8$, 16.



- Consider (**??**) with $K = 1$, $\theta = 0.5$, $\beta =$ 0.95, $u_l = 0$, $u_r = 1$,

$$f(x) = \frac{(1-\gamma)(1-\beta)}{\Gamma(\beta)x(1-x)^{1-\beta}}, \qquad u(x) = x^{1-\beta}, \quad x \in (0,1).$$

	m	n	Error	Iterations
Gauss	2^3	2^8	1.4379×10^{-1}	
	2^4	2^9	1.0491×10^{-1}	
	2^5	2^{10}	5.8194×10^{-2}	
CGS	2^3	2^8	1.4379×10^{-2}	48
	2^4	2^9	1.0491×10^{-1}	77
	2^5	2^{10}	5.8194×10^{-2}	142
FCGS	2^3	2^8	1.4379×10^{-1}	48
	2^4	2^9	1.0491×10^{-1}	78
	2^5	2^{10}	5.8194×10^{-2}	150
PFCGS	2^3	2^8	1.4379×10^{-1}	9
	2^4	2^9	1.0491×10^{-1}	13
	2^5	2^{10}	5.8194×10^{-2}	16

Table: Numerical results on a uniform mesh

	n	Error	Iterations	CPUs
Gauss	2^8	1.8827×10^{-1}		0.01s
	2^9	1.8206×10^{-1}		0.01s
	2^{10}	1.7596×10^{-1}		0.05s
	2^{11}	1.7002×10^{-1}		0.25s
	2^{12}	1.6425×10^{-1}		1.25s
	2^{13}	1.5867×10^{-1}		9.76s
	2^{14}	1.5327×10^{-1}		97s
CGS	2^8	1.8827×10^{-1}	46	0.01s
	2^9	1.8206×10^{-1}	66	0.01s
	2^{10}	1.7596×10^{-1}	94	0.18s
	2^{11}	1.7002×10^{-1}	133	0.86s
	2^{12}	1.6425×10^{-1}	188	4.94s
	2^{13}	1.5867×10^{-1}	266	30.78s
	2^{14}	1.5327×10^{-1}	379	187s
FCGS	2^8	1.8827×10^{-1}	46	0.05s
	2^9	1.8206×10^{-1}	66	0.16s
	2^{10}	1.7596×10^{-1}	94	0.29s
	2^{11}	1.7002×10^{-1}	133	1.16s
	2^{12}	1.6425×10^{-1}	188	2.00s
	2^{13}	1.5867×10^{-1}	266	12s
	2^{14}	1.5327×10^{-1}	379	27s
PFCGS	2^8	1.8827×10^{-1}	8	0.02s
	2^9	1.8206×10^{-1}	8	0.02s
	2^{10}	1.7596×10^{-1}	9	0.05s
	2^{11}	1.7002×10^{-1}	10	0.09s
	2^{12}	1.6425×10^{-1}	10	0.14s
	2^{13}	1.5867×10^{-1}	10	0.66s
	2^{14}	1.5327×10^{-1}	11	1.00s

$$
\begin{aligned}
-\int_0^{2\pi} \big(D_\theta \ K \ I_\theta^\beta D_\theta u(x,y)\big) P(d\theta) = f(x,y), \quad & \text{in } \Omega \subset \mathbb{R}^2, \\
u = 0, \quad & \text{on } \partial\Omega.
\end{aligned}
\tag{41}
$$

- $P(d\theta)$ is a probability measure on $[0, 2\pi)$,
- D_θ is the differential operator in the direction of θ

$$
D_\theta u(x,y) := \Big(\cos\theta \frac{\partial}{\partial x} + \sin\theta \frac{\partial}{\partial y}\Big) u(x,y),
$$

  and I_θ^β, with $0 < \beta < 1$, represents the βth order fractional integral operator in the direction of θ given by

$$
I_\theta^\beta u(x,y) := \int_0^\infty \frac{s^{\beta-1}}{\Gamma(\beta)} u(x - s\cos\theta, y - s\sin\theta) ds.
$$

- If $P(d\theta)$ is atomic with atoms $\{0, \pi/2, \pi, 3\pi/2\}$, then (41) reduces to the usual coordinate form.

- Galerkin formulation: given $f \in H^{-(1-\beta/2)}(\Omega)$, seek $u \in H_0^{1-\beta/2}(\Omega)$

$$B(u,v) := \int_0^{2\pi} \Big[\int_\Omega K\ I_\theta^\beta D_\theta u\ D_\theta v dx dy \Big] P(d\theta) = \langle f, v \rangle, \qquad \forall\ v \in H_0^{1-\beta/2}(\Omega). \tag{42}$$

Theorem

$B(\cdot,\cdot)$ is coercive and continuous on $H_0^{1-\beta/2}(\Omega) \times H_0^{1-\beta/2}(\Omega)$. Hence, the Galerkin weak formulation (42) has a unique solution. Moreover,

$$\|u\|_{H^{1-\beta/2}(\Omega)} \le C \|f\|_{H^{-(1-\beta/2)}(\Omega)}.$$

- Let $h_1 := 1/(N_1+1)$, $h_2 := 1/(N_2+1)$, $x_i := ih_1$, and $y_j := jh_2$.
- Let $\psi(\xi) = 1 - |\xi|$ for $\xi \in [-1,1]$ and 0 elsewhere. Let

$$\begin{aligned} \phi_{i,j}(x,y) &:= \psi\left(\frac{x-x_i}{h_1}\right)\psi\left(\frac{y-y_j}{h_2}\right), \quad 1 \le i \le N_1,\ 2 \le j \le N_2, \\ u_h(x,y) &= \sum_{j'=1}^{N_2}\sum_{i'=1}^{N_1} u_{i',j'}\phi_{i',j'}(x,y), \quad (x,y) \in \Omega. \end{aligned}$$

- A bilinear finite element scheme for $i = 1,\ldots,N_1$ and $j = 1,\ldots,N_2$

$$\sum_{j'=1}^{N_2}\sum_{i'=1}^{N_1} B\big(\phi_{i',j'},\phi_{i,j}\big)u_{i',j'} = \big(f,\phi_{i,j}\big)_{L^2} =: f_{i,j}. \tag{43}$$

- Let $N := N_1 N_2$, $A = \big[a_{m,n}\big]_{m,n=1}^{N}$, and

$$\begin{aligned} u &:= \big[u_{1,1},\ldots,u_{N_1,1},u_{1,2},\ldots,u_{N_1,2},\ldots,u_{1,N_2},\ldots,u_{N_1,N_2}\big]^T,\\ f &:= \big[f_{1,1},\ldots,f_{N_1,1},f_{1,2},\ldots,f_{N_1,2},\ldots,f_{1,N_2},\ldots,f_{N_1,N_2}\big]^T \end{aligned}$$

- Let $a_{m,n} := B\big(\phi_{i',j'},\phi_{i,j}\big)$ with

$$\begin{aligned} m &= (j-1)N_1 + i, \qquad 1 \le i \le N_1, \quad 1 \le j \le N_2,\\ n &= (j'-1)N_1 + i', \qquad 1 \le i' \le N_1, \quad 1 \le j' \le N_2. \end{aligned} \tag{44}$$

- The FEM (43) can be expressed in a matrix form

$$Au = f. \tag{45}$$

- Features of numerical methods for coordinate-form FPDEs
  - A is dense, the number of nonzero entries at each row $= O(N_1 + N_2)$, which $\to \infty$ as $N \to \infty$.
  - The number of nonzero entries at each row divided by the total number of the entries at the same row $= O((N_1 + N_2)/N) = O(N^{-1/2})$.
  - A has a tensor produce structure.
- Features of the finite element method for full FPDEs
  - A is full.
  - A has a complicated structure, as it couples the nodes in all the directions!
  - It does not seem feasible to explore a tensor-produce structure of A.
  - We instead explore the translation invariance property of A.

Theorem

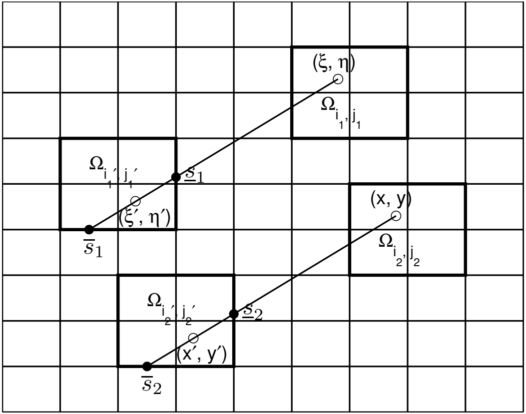
Let the indices (i_1, j_1), (i_1', j_1'), (i_2, j_2), *and* (i_2', j_2') *be related by*

$$i_1' - i_1 = i_2' - i_2, \qquad j_1' - j_1 = j_2' - j_2. \tag{46}$$

Then the following translation-invariance property holds

$$\begin{aligned}
&\int_0^{2\pi} \Big[\int_\Omega K \; D_\theta^{-\beta} D_\theta \phi_{i_1', j_1'}(x,y) D_\theta \phi_{i_1, j_1}(x,y) dxdy \Big] P(d\theta) \\
&\quad = \int_0^{2\pi} \Big[\int_\Omega K \; D_\theta^{-\beta} D_\theta \phi_{i_2', j_2'}(x,y) D_\theta \phi_{i_2, j_2}(x,y) dxdy \Big] P(d\theta).
\end{aligned} \tag{47}$$

Figure: Illustration of the translation invariance



Theorem

The stiffness matrix A is an N_2-by-N_2 block-Toeplitz matrix

$$A = \begin{pmatrix} T_0 & T_1 & \dots & T_{N_2-2} & T_{N_2-1} \\ T_{-1} & T_0 & T_1 & \ddots & T_{N_2-2} \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ T_{2-N_2} & \ddots & T_{-1} & T_0 & T_1 \\ T_{1-N_2} & T_{2-N_2} & \dots & T_{-1} & T_0 \end{pmatrix}, \tag{48}$$

Each block T_j is an N_1-by-N_1 Toeplitz matrix

$$T_j = \begin{pmatrix} t_{0,j} & t_{1,j} & \dots & t_{N_1-2,j} & t_{N_1-1,j} \\ t_{-1,j} & t_{0,j} & t_{1,j} & \ddots & t_{N_1-2,j} \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ t_{2-N_1,j} & \ddots & t_{-1,j} & t_{0,j} & t_{1,j} \\ t_{1-N_1,j} & t_{2-N_1,j} & \dots & t_{-1,j} & t_{0,j} \end{pmatrix}. \tag{49}$$

A is symmetric if the probability measure $P(d\theta)$ is periodic with a period π.

- Av can be evaluated in $O(N \log N)$ operations, by embedded into a $4N$-by-$4N$ block-circulant-circulant-block matrix.
  - For coordinate FPDEs, A^y is block-Toeplitz-circulant-block that can be embedded into a $2N$-by-$2N$ block-circulant-circulant-block matrix.
- A is generated by $O(N)$ parameters.
  - A requires only $O(N)$ memory to store.
  - Unlike FDM, the evaluation of A is very expensive.
  - Only $O(N)$ (in contrast to N^2) entries of A need to be evaluated, a significant reduction of CPU time.
- A block-circulant-circulant-block preconditioner can be developed.

- A 4-point (2 points in x or y) Gauss-Legendre quadrature is used to evaluate entries of A and the right-hand side
- The finite element scheme is solved by the fast congugate gradient squared (FCGS), the preconditioned fast CGS (PFCGS), and Gaussian elimination (Gauss) solvers.
- These solvers were implemented using Compaq Visual Fortran 6.6 on a ThinkPad T410 Laptop.

- $\beta = 0.5$, $K_i := 1 + \sin 2\theta_i$ for $i = 1, 2, 3, 4$.
- $u = x^2(1-x)^2y^2(1-y)^2$, f is calculated accordingly.

Table: The convergence rates of the Gauss, FCGS, and PFCGS solutions

	Gauss	FCGS	PFCGS	
$N_1 = N_2$	$\lVert u - u_h \rVert_{L^2(\Omega)}$	$\lVert u - u_h \rVert_{L^2(\Omega)}$	$\lVert u - u_h \rVert_{L^2(\Omega)}$	Conv. Rate
2^3	3.487×10^{-5}	3.487×10^{-5}	3.487×10^{-5}	
2^4	8.876×10^{-6}	8.876×10^{-6}	8.876×10^{-6}	1.97
2^5	2.097×10^{-6}	2.097×10^{-6}	2.097×10^{-6}	2.08
2^6	4.759×10^{-7}	4.759×10^{-7}	4.759×10^{-7}	2.14
2^7	N/A	1.055×10^{-7}	1.056×10^{-7}	2.17
2^8	N/A	2.307×10^{-8}	2.311×10^{-8}	2.19
2^9	N/A	4.999×10^{-9}	5.003×10^{-9}	2.21
2^{10}	N/A	1.079×10^{-9}	1.078×10^{-9}	2.21

Table: The CPU time of the FCGS, PFCGS, and Gauss

	full A	$O(N)$ entries	Gauss	FCGS		PFCGS	
$N_1 = N_2$	CPU	CPU	CPU	CPU	Itr. #	CPU	Itr. #
2^3	0.91s	0.05s	0.00s	0.00s	5	0.00s	4
2^4	14s	0.20s	0.05s	0.00s	9	0.00s	6
2^5	3m47s	0.83s	19s	0.05s	15	0.05s	7
2^6	1h2m	3.48s	25m6s	0.45s	28	0.19s	10
2^7	N/A	14s	N/A	3.44s	52	0.94s	11
2^8	N/A	55s	N/A	35s	94	6.73s	15
2^9	N/A	3m37s	N/A	4m49s	170	44s	21
2^{10}	N/A	14m39s	N/A	35m43s	300	4m13s	29

- A similar strategy can be used for
  - high-order finite element methods
  - discontinuous Galerkin methods

  where the stiffness matrices would be in block Toeplitz-like form in the context of uniform meshes.
- The development of an efficient and effective preconditioner can be significantly more difficult and challenging.

Thank You

for Your Attention!