asQ - ParaDiag methods with Firedrake and PETSc

Josh Hope-Collins

July 7, 2025

Outline

Parallel-in-time motivation

ParaDiag methods

asQ

Outline

Parallel-in-time motivation

ParaDiag methods

asQ

Time-dependent O/PDEs

$$\boldsymbol{M}\partial_t u + \boldsymbol{K} u = b(t)$$

$$\boldsymbol{M}\left(\frac{u^{n+1} - u^n}{\Delta t}\right) + \theta \boldsymbol{K} u^{n+1} + (1-\theta)\,\boldsymbol{K} u^n = b^{n+1}$$

$$\left(\frac{\boldsymbol{M}}{\Delta t} + \theta \boldsymbol{K}\right) u^{n+1} = b^{n+1} - \left(\frac{-\boldsymbol{M}}{\Delta t} + (1-\theta)\,\boldsymbol{K}\right) u^n$$

Performance of serial method

$$\left(\frac{\boldsymbol{M}}{\Delta t} + \theta \boldsymbol{K}\right) u^{n+1} = \tilde{b}$$

Work: $W_s = K_s M_s N_x^q N_t \sim N_x^q N_t$

Processors: $P_s \sim N_x$

Time: $T_s = \dfrac{W_s}{P_s} = K_s M_s N_t N_x^{q-1} \sim N_t N_x^{q-1}$

Outline

Parallel-in-time motivation

ParaDiag methods

asQ

IMPERIAL

All-at-once system

$$\left(\frac{-\boldsymbol{M}}{\Delta t} + (1-\theta)\,\boldsymbol{K}\right) u^{n} + \left(\frac{\boldsymbol{M}}{\Delta t} + \theta\boldsymbol{K}\right) u^{n+1} = b^{n+1}$$

All-at-once system

$$\boldsymbol{A}_0 u^n + \boldsymbol{A}_1 u^{n+1} = b^{n+1}$$

All-at-once system

$$A_0 u^n + A_1 u^{n+1} = b^{n+1}$$

$$A_0 u^0 + \quad A_1 u^1 \qquad = b^1$$

All-at-once system

$$\boldsymbol{A}_0 u^n + \boldsymbol{A}_1 u^{n+1} = b^{n+1}$$

$$\begin{aligned}
\boldsymbol{A}_0 u^0 + \quad \boldsymbol{A}_1 u^1 & & &= b^1 \\
\boldsymbol{A}_0 u^1 + \quad \boldsymbol{A}_1 u^2 & & &= b^2
\end{aligned}$$

All-at-once system

$$\boldsymbol{A}_0 u^n + \boldsymbol{A}_1 u^{n+1} = b^{n+1}$$

$$\begin{aligned}
\boldsymbol{A}_0 u^0 + \quad \boldsymbol{A}_1 u^1 \qquad\qquad\qquad\qquad &= b^1 \\
\boldsymbol{A}_0 u^1 + \quad \boldsymbol{A}_1 u^2 \qquad\qquad &= b^2 \\
\boldsymbol{A}_0 u^2 + \quad \boldsymbol{A}_1 u^3 &= b^3
\end{aligned}$$

IMPERIAL

All-at-once system

$$\boldsymbol{A}_0 u^n + \boldsymbol{A}_1 u^{n+1} = b^{n+1}$$

$$\begin{aligned}
\boldsymbol{A}_0 u^0 + \quad \boldsymbol{A}_1 u^1 \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad &= b^1 \\
\boldsymbol{A}_0 u^1 + \quad \boldsymbol{A}_1 u^2 \qquad\qquad\qquad\qquad\qquad\qquad &= b^2 \\
\boldsymbol{A}_0 u^2 + \quad \boldsymbol{A}_1 u^3 \qquad\qquad\qquad &= b^3 \\
\boldsymbol{A}_0 u^3 + \quad \boldsymbol{A}_1 u^4 \qquad &= b^4
\end{aligned}$$

All-at-once system

$$\boldsymbol{A}_0 u^n + \boldsymbol{A}_1 u^{n+1} = b^{n+1}$$

$$\begin{pmatrix} \boldsymbol{A}_1 & & & \\ \boldsymbol{A}_0 & \boldsymbol{A}_1 & & \\ & \boldsymbol{A}_0 & \boldsymbol{A}_1 & \\ & & \boldsymbol{A}_0 & \boldsymbol{A}_1 \end{pmatrix} \begin{pmatrix} u^1 \\ u^2 \\ u^3 \\ u^4 \end{pmatrix} = \begin{pmatrix} b^1 - \boldsymbol{A}_0 u^0 \\ b^2 \\ b^3 \\ b^4 \end{pmatrix}$$

$$\boldsymbol{A}\boldsymbol{u} = \boldsymbol{b}$$

Kronecker products

$$\boldsymbol{A} \in \mathbb{R}^{n\times n}, \quad \boldsymbol{B} \in \mathbb{R}^{m\times m}, \quad \boldsymbol{A}\otimes\boldsymbol{B} \in \mathbb{R}^{nm\times nm}$$

$$\boldsymbol{A}\otimes\boldsymbol{B} = \begin{pmatrix} a_{00}\boldsymbol{B} & a_{01}\boldsymbol{B} & a_{02}\boldsymbol{B} \\ a_{10}\boldsymbol{B} & a_{11}\boldsymbol{B} & a_{12}\boldsymbol{B} \\ a_{20}\boldsymbol{B} & a_{21}\boldsymbol{B} & a_{22}\boldsymbol{B} \end{pmatrix}$$

$$(\boldsymbol{AC})\otimes(\boldsymbol{BD}) = (\boldsymbol{A}\otimes\boldsymbol{B})(\boldsymbol{C}\otimes\boldsymbol{D})$$

$$\boldsymbol{A}\otimes\boldsymbol{B} = (\boldsymbol{A}\otimes\boldsymbol{I})(\boldsymbol{I}\otimes\boldsymbol{B}) = (\boldsymbol{I}\otimes\boldsymbol{B})(\boldsymbol{A}\otimes\boldsymbol{I})$$

θ-method all-at-once sytem

$$\boldsymbol{M}\left(\frac{u^{n+1}-u^{n}}{\Delta t}\right)+(1-\theta)\,\boldsymbol{K}u^{n}+\theta\boldsymbol{K}u^{n+1}=b^{n+1}$$

$$\boldsymbol{A}=\begin{pmatrix} \frac{1}{\Delta t} & & & \\ \frac{-1}{\Delta t} & \frac{1}{\Delta t} & & \\ & \frac{-1}{\Delta t} & \frac{1}{\Delta t} & \\ & & \frac{-1}{\Delta t} & \frac{1}{\Delta t} \end{pmatrix}\otimes\boldsymbol{M}+\begin{pmatrix} \theta & & & \\ (1-\theta) & \theta & & \\ & (1-\theta) & \theta & \\ & & (1-\theta) & \theta \end{pmatrix}\otimes\boldsymbol{K}$$

$$\boldsymbol{A}=\boldsymbol{B}_1\otimes\boldsymbol{M}+\boldsymbol{B}_2\otimes\boldsymbol{K}$$

ParaDiag-II: Circulant preconditioners

$$\begin{pmatrix} \frac{1}{\Delta t} & & & \frac{-1}{\Delta t} \\ \frac{-1}{\Delta t} & \frac{1}{\Delta t} & & \\ & \frac{-1}{\Delta t} & \frac{1}{\Delta t} & \\ & & \frac{-1}{\Delta t} & \frac{1}{\Delta t} \end{pmatrix} \otimes \boldsymbol{M} + \begin{pmatrix} \theta & & & (1-\theta) \\ (1-\theta) & \theta & & \\ & (1-\theta) & \theta & \\ & & (1-\theta) & \theta \end{pmatrix} \otimes \boldsymbol{K}$$

$$\boldsymbol{P} = \boldsymbol{C}_1 \otimes \boldsymbol{M} + \boldsymbol{C}_2 \otimes \boldsymbol{K} \approx \boldsymbol{A}$$

$$\boldsymbol{P} = (\boldsymbol{V} \otimes \boldsymbol{I}_x)(\boldsymbol{D}_1 \otimes \boldsymbol{M} + \boldsymbol{D}_2 \otimes \boldsymbol{K})(\boldsymbol{V}^{-1} \otimes \boldsymbol{I}_x)$$

- $\boldsymbol{C}_{1,2} = \boldsymbol{V}\boldsymbol{D}_{1,2}\boldsymbol{V}^{-1}$ are *simultaneously* diagonalisable

ParaDiag-II: Circulant preconditioners

$$\begin{pmatrix} \frac{1}{\Delta t} & & & \frac{-\alpha}{\Delta t} \\ \frac{-1}{\Delta t} & \frac{1}{\Delta t} & & \\ & \frac{-1}{\Delta t} & \frac{1}{\Delta t} & \\ & & \frac{-1}{\Delta t} & \frac{1}{\Delta t} \end{pmatrix} \otimes \boldsymbol{M} + \begin{pmatrix} \theta & & & \alpha\,(1-\theta) \\ (1-\theta) & \theta & & \\ & (1-\theta) & \theta & \\ & & (1-\theta) & \theta \end{pmatrix} \otimes \boldsymbol{K}$$

$$\boldsymbol{P}^{(\alpha)} = \boldsymbol{C}_1^{(\alpha)} \otimes \boldsymbol{M} + \boldsymbol{C}_2^{(\alpha)} \otimes \boldsymbol{K} \approx \boldsymbol{A}$$

$$\boldsymbol{P}^{(\alpha)} = (\boldsymbol{V} \otimes \boldsymbol{I}_x)\,(\boldsymbol{D}_1 \otimes \boldsymbol{M} + \boldsymbol{D}_2 \otimes \boldsymbol{K})\left(\boldsymbol{V}^{-1} \otimes \boldsymbol{I}_x\right)$$

- $\boldsymbol{C}_{1,2}^{(\alpha)} = \boldsymbol{V}\boldsymbol{D}_{1,2}\boldsymbol{V}^{-1}$ are *simultaneously* diagonalisable
- $\alpha \in (0,1]$, and in practice can be very small ($\approx 10^{-4}$)

Circulant diagonalisation

$$\boldsymbol{P} = \boldsymbol{C}_1 \otimes \boldsymbol{M} + \boldsymbol{C}_2 \otimes \boldsymbol{K}, \qquad \boldsymbol{C}_j = \boldsymbol{V}\boldsymbol{D}_j\boldsymbol{V}^{-1},$$

$$\boldsymbol{P} = \left(\boldsymbol{V} \otimes \boldsymbol{I}_x\right)\left(\boldsymbol{D}_1 \otimes \boldsymbol{M} + \boldsymbol{D}_2 \otimes \boldsymbol{K}\right)\left(\boldsymbol{V}^{-1} \otimes \boldsymbol{I}_x\right)$$

$$\boldsymbol{\Lambda}_k = \left(\lambda_{1,k}\boldsymbol{M} + \lambda_{2,k}\boldsymbol{K}\right) \quad \forall k \in [1, N_t]$$

$$\boldsymbol{V} = \boldsymbol{\Gamma}_\alpha^{-1}\boldsymbol{\mathcal{F}}^{-1}, \quad \boldsymbol{D}_j = \mathtt{diag}\left(\boldsymbol{\mathcal{F}}\boldsymbol{\Gamma}_\alpha \boldsymbol{c}_j\right), \quad \boldsymbol{\Gamma}_\alpha = \mathtt{diag}\left(\alpha^{\frac{k-1}{N_t}}\right) \forall k \in [1, N_t]$$

Circulant linear solves

$$\boldsymbol{P} = \left(\boldsymbol{V} \otimes \boldsymbol{I}_x\right)\left(\boldsymbol{D}_1 \otimes \boldsymbol{M} + \boldsymbol{D}_2 \otimes \boldsymbol{K}\right)\left(\boldsymbol{V}^{-1} \otimes \boldsymbol{I}_x\right)$$

$$\boldsymbol{P}\boldsymbol{x} = \boldsymbol{b}$$

Circulant linear solves

$$P = \color{red}{(V \otimes I_x)} (D_1 \otimes M + D_2 \otimes K) \left(V^{-1} \otimes I_x\right)$$

$$Px = b$$

- Step-(a) $y_1 = \left(V^{-1} \otimes I_x\right) b$

Circulant linear solves

$$\boldsymbol{P} = \left(\boldsymbol{V} \otimes \boldsymbol{I}_x\right) \left(\boldsymbol{D}_1 \otimes \boldsymbol{M} + \boldsymbol{D}_2 \otimes \boldsymbol{K}\right) \left(\boldsymbol{V}^{-1} \otimes \boldsymbol{I}_x\right)$$

$$\boldsymbol{P}\boldsymbol{x} = \boldsymbol{b}$$

- Step-(a) $\boldsymbol{y}_1 = \left(\boldsymbol{V}^{-1} \otimes \boldsymbol{I}_x\right) \boldsymbol{b}$
- Step-(b) $\left(\lambda_{1,j}\boldsymbol{M} + \lambda_{2,j}\boldsymbol{K}\right) \boldsymbol{y}_{2,n} = \boldsymbol{y}_{1,n} \quad \forall j \in [1, N_t]$

Circulant linear solves

$$\boldsymbol{P} = (\boldsymbol{V} \otimes \boldsymbol{I}_x)(\boldsymbol{D}_1 \otimes \boldsymbol{M} + \boldsymbol{D}_2 \otimes \boldsymbol{K})\left(\boldsymbol{V}^{-1} \otimes \boldsymbol{I}_x\right)$$

$$\boldsymbol{P}\boldsymbol{x} = \boldsymbol{b}$$

- Step-(a) $\boldsymbol{y}_1 = \left(\boldsymbol{V}^{-1} \otimes \boldsymbol{I}_x\right)\boldsymbol{b}$
- Step-(b) $\left(\lambda_{1,j}\boldsymbol{M} + \lambda_{2,j}\boldsymbol{K}\right)\boldsymbol{y}_{2,n} = \boldsymbol{y}_{1,n} \quad \forall j \in [1, N_t]$
- Step-(c) $\boldsymbol{x} = (\boldsymbol{V} \otimes \boldsymbol{I}_x)\,\boldsymbol{y}_2$

Circulant linear solves

$$\boldsymbol{P} = (\boldsymbol{V} \otimes \boldsymbol{I}_x)(\boldsymbol{D}_1 \otimes \boldsymbol{M} + \boldsymbol{D}_2 \otimes \boldsymbol{K})\left(\boldsymbol{V}^{-1} \otimes \boldsymbol{I}_x\right)$$

$$\boldsymbol{P}\boldsymbol{x} = \boldsymbol{b}$$

- Step-(a) $\boldsymbol{y}_1 = \left(\boldsymbol{V}^{-1} \otimes \boldsymbol{I}_x\right)\boldsymbol{b}$
- Step-(b) $\left(\lambda_{1,j}\boldsymbol{M} + \lambda_{2,j}\boldsymbol{K}\right)\boldsymbol{y}_{2,n} = \boldsymbol{y}_{1,n} \quad \forall j \in [1, N_t]$
- Step-(c) $\boldsymbol{x} = (\boldsymbol{V} \otimes \boldsymbol{I}_x)\,\boldsymbol{y}_2$

All-at-once solution strategies

- Preconditioned Krylov method: $\boldsymbol{P}^{-1}\boldsymbol{A}\boldsymbol{x} = \boldsymbol{P}^{-1}\boldsymbol{b}$
  $\boldsymbol{P}^{-1}\boldsymbol{A}$ has N_x non-unit eigenvalues
- Richardson iteration: $\boldsymbol{P}\boldsymbol{x}^{k+1} = (\boldsymbol{P} - \boldsymbol{A})\,\boldsymbol{x}^{k} + \boldsymbol{b}$
  Convergence rate bounded by $\alpha/(1-\alpha)$
- Roundoff error: $\mathcal{O}(\epsilon\alpha^{-1})$ if ϵ is machine precision

Performance model vs time-serial method

$$(\beta_1 \boldsymbol{M} + \beta_2 \boldsymbol{K})\, x = \tilde{b}$$

Serial: $(\beta_1 = 1/\Delta t,\ \beta_2 = \theta)$

$$W_s = K_s\, M_s\, (N_x^q\, N_t)$$

$$P_s \sim N_x$$

$$T_s \sim \frac{W_s}{P_s} = K_s\, M_s\, N_t\, N_x^{q-1}$$

Parallel: $(\beta_1 = \lambda_1,\ \beta_2 = \lambda_2)$

$$W_p = 2\, K_p\, M_p\, (N_x^q\, N_t)$$

$$P_p \sim 2 N_x\, N_t$$

$$T_p \sim \frac{W_p}{P_p} = K_p\, M_p\, N_x^{q-1} + T_c$$

Ideal speedup bound

Speedup: $S = \dfrac{T_s}{T_p} = \left(\dfrac{N_t}{\gamma\omega}\right)\dfrac{1}{1 + T_c/T_b}$

"Difficulty" measures: $\gamma = \dfrac{K_p}{K_s}, \quad \omega = \dfrac{M_p}{M_s}$

Block solve time: $T_b = K_p N_x^{q-1}$

Communication time: T_c

Efficiency: $E = \dfrac{S}{P_p/P_s} = \dfrac{1}{2\gamma\omega}\dfrac{1}{1 + T_c/T_b}$

Nonlinear all-at-once system

$$\boldsymbol{M}\partial_t \boldsymbol{u} + \boldsymbol{f}(\boldsymbol{u}) = 0$$

$$\begin{pmatrix} \frac{1}{\Delta t} & & & \\ \frac{-1}{\Delta t} & \frac{1}{\Delta t} & & \\ & \frac{-1}{\Delta t} & \frac{1}{\Delta t} & \\ & & \frac{-1}{\Delta t} & \frac{1}{\Delta t} \end{pmatrix} \otimes \boldsymbol{M} + \left(\begin{pmatrix} \theta & & & \\ (1-\theta) & \theta & & \\ & (1-\theta) & \theta & \\ & & (1-\theta) & \theta \end{pmatrix} \otimes \boldsymbol{I}_x \right) \begin{pmatrix} \boldsymbol{f}(\boldsymbol{u}^1) \\ \boldsymbol{f}(\boldsymbol{u}^2) \\ \boldsymbol{f}(\boldsymbol{u}^3) \\ \boldsymbol{f}(\boldsymbol{u}^4) \end{pmatrix}$$

IMPERIAL

Nonlinear all-at-once Jacobian

$$\boldsymbol{M}\partial_t \boldsymbol{u} + \boldsymbol{f}(\boldsymbol{u}) = 0$$

$$\begin{pmatrix} \theta\nabla\boldsymbol{f}(\boldsymbol{u}^1) & & & \\ (1-\theta)\nabla\boldsymbol{f}(\boldsymbol{u}^1) & \theta\nabla\boldsymbol{f}(\boldsymbol{u}^2) & & \\ & (1-\theta)\nabla\boldsymbol{f}(\boldsymbol{u}^2) & \theta\nabla\boldsymbol{f}(\boldsymbol{u}^3) & \\ & & (1-\theta)\nabla\boldsymbol{f}(\boldsymbol{u}^3) & \theta\nabla\boldsymbol{f}(\boldsymbol{u}^4) \end{pmatrix}$$

Nonlinear all-at-once preconditioner

$$\boldsymbol{M}\partial_t \boldsymbol{u} + \boldsymbol{f}(\boldsymbol{u}) = 0$$

$$\begin{pmatrix} \frac{1}{\Delta t} & & & \frac{-\alpha}{\Delta t} \\ \frac{-1}{\Delta t} & \frac{1}{\Delta t} & & \\ & \frac{-1}{\Delta t} & \frac{1}{\Delta t} & \\ & & \frac{-1}{\Delta t} & \frac{1}{\Delta t} \end{pmatrix} \otimes \boldsymbol{M} + \begin{pmatrix} \theta & & & \alpha(1-\theta) \\ (1-\theta) & \theta & & \\ & (1-\theta) & \theta & \\ & & (1-\theta) & \theta \end{pmatrix} \otimes \overline{\nabla \boldsymbol{f}(\boldsymbol{u})}$$

$$\overline{\nabla \boldsymbol{f}(\boldsymbol{u})} = \sum_n^{N_t} \frac{\nabla \boldsymbol{f}(\boldsymbol{u}^n)}{N_t} \quad \text{or} \quad \nabla \boldsymbol{f}(\overline{\boldsymbol{u}}) = \nabla \boldsymbol{f}\left(\sum_n^{N_t} \frac{\boldsymbol{u}^n}{N_t}\right)$$

Nonlinear problems

Nonlinear system: $\boldsymbol{M}\partial_t u + \boldsymbol{f}(u)$

All-at-once system: $(\boldsymbol{B}_1 \otimes \boldsymbol{M})\,\boldsymbol{u} + (\boldsymbol{B}_2 \otimes \boldsymbol{I}_x)\,\boldsymbol{F}(\boldsymbol{u})$

All-at-once Jacobian: $(\boldsymbol{B}_1 \otimes \boldsymbol{M}) + (\boldsymbol{B}_2 \otimes \boldsymbol{I}_x)\,\nabla \boldsymbol{F}(\boldsymbol{u})$

ParaDiag Jacobian: $(\boldsymbol{C}_1 \otimes \boldsymbol{M}) + (\boldsymbol{C}_2 \otimes \nabla \boldsymbol{f}(\overline{\boldsymbol{u}}))$

Time average: $\overline{\boldsymbol{u}} = \sum_n^{N_t} \boldsymbol{u}^n / N_t$

Outline

Parallel-in-time motivation

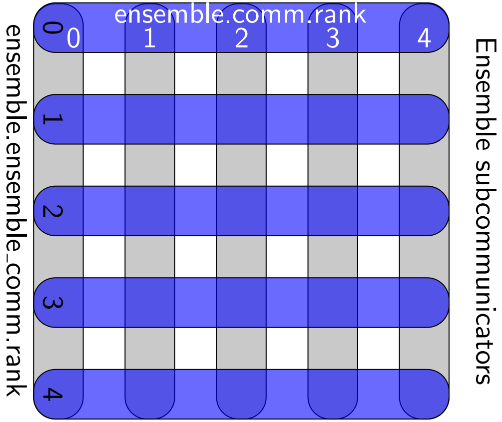
ParaDiag methods

asQ

asQ (github.com/firedrakeproject/asQ)

- A library for implementing ParaDiag methods using Firedrake and PETSc
- General to any time-dependent PDE expressable in UFL
- Provides linear and nonlinear ParaDiag methods
- Parallelised in both time and space
- Aims to be flexible enough to be a sandbox, but performant enough to run “real” cases

Space-time parallelism: `firedrake.Ensemble`



Spatial subcommunicators

github.com/firedrakeproject/asQ

```
### === Solve the heat equation using ParaDiag === ###
from firedrake import *
import asQ

# Space-time partition and all-at-once function
partition = [2, 2, 2, 2]
ensemble = asQ.create_ensemble(partition)

mesh = UnitSquareMesh(32, 32, comm=ensemble.comm)
x, y = SpatialCoordinate(mesh)

V = FunctionSpace(mesh, "CG", degree=1)
ics = Function(V).project(sin(2*pi*x)*cos(pi*y))

aaofunc = asQ.AllAtOnceFunction(ensemble,partition, V)
aaofunc.assign(ics)
```

github.com/firedrakeproject/asQ

```
# Finite element forms

def form_mass(u, v): # M
    return inner(u, v)*dx

def form_function(u, v, t): # K
    return inner(grad(u), grad(v))*dx

bcs = [DirichletBC(V, 0, subdomain=1)]

dt = 0.1
theta = 1

aaoform = asQ.AllAtOnceForm(aaofunc, dt, theta,
                            form_mass, form_function,
                            bcs=bcs)
```

github.com/firedrakeproject/asQ

```
# ParaDiag solver and windowing

solver = asQ.AllAtOnceSolver(
    aaoform, aaofunc, jacobian_form=aaoform,
    solver_parameters=solver_parameters)

u = Function(V)
ofile = VTKFile(f"output{rank}.vtk", mesh.comm)
for i in range(nwindows):
    solver.solve(rhs=None)

    ofile.write(aaofunc[0])

    aaofunc.bcast_field(-1, u)
    aaofunc.assign(u)
```

github.com/firedrakeproject/asQ

```
solver_parameters = {
    "snes_type": "ksponly",
    "ksp_monitor": None,
    "ksp_type": "gmres",
    "ksp_rtol": 1e-8,
    "pc_type": "python",
    "pc_python_type": "asQ.CirculantPC",
    "circulant": {
        "alpha": 1e-4,
        "state": "initial",
        "block": {
            "mat_type": "aij",
            "ksp_type": "chebyshev",
            "ksp_max_it": 3,
            "pc_type": "ilu"}},
    "aaos_jacobian_state": "current"}
```