Parallel-in-Time methods for cardiac electrophysiology

Giacomo Rosilho de Souza, Simone Pezzuto, Rolf Krause

Università della Svizzera Italiana, Lugano, Switzerland

Microcard Meeting - Freiburg 2024

Monodomain equation

Spatially discretized monodomain equation:

$$
\begin{aligned}
\mathbf{V}' &= A\mathbf{V} - I_{ion}(\mathbf{V}, \mathbf{z}_a, \mathbf{z}_g),\\
\mathbf{z}_a' &= g_a(\mathbf{V}, \mathbf{z}_a, \mathbf{z}_g),\\
\mathbf{z}_g' &= \Lambda_g(\mathbf{V})(\mathbf{z}_g - \mathbf{z}_{g,\infty}(\mathbf{V}))
\end{aligned}
$$

Monodomain equation

Spatially discretized monodomain equation:

$$\begin{aligned}\mathbf{V}' &= A\mathbf{V} - I_{ion}(\mathbf{V}, \mathbf{z}_a, \mathbf{z}_g),\\ \mathbf{z}_a' &= g_a(\mathbf{V}, \mathbf{z}_a, \mathbf{z}_g),\\ \mathbf{z}_g' &= \Lambda_g(\mathbf{V})(\mathbf{z}_g - \mathbf{z}_{g,\infty}(\mathbf{V}))\end{aligned} \quad \Longrightarrow \quad \begin{pmatrix}\mathbf{V}'\\ \mathbf{z}_a'\\ \mathbf{z}_g'\end{pmatrix} = \begin{pmatrix}A\mathbf{V}\\ 0\\ 0\end{pmatrix} + \begin{pmatrix}-I_{ion}(\mathbf{V}, \mathbf{z}_a, \mathbf{z}_g)\\ g_a(\mathbf{V}, \mathbf{z}_a, \mathbf{z}_g)\\ 0\end{pmatrix} + \begin{pmatrix}0 & 0 & 0\\ 0 & 0 & 0\\ 0 & 0 & \Lambda_g(\mathbf{V})\end{pmatrix}\begin{pmatrix}0\\ 0\\ \mathbf{z}_g - \mathbf{z}_{g,\infty}(\mathbf{V})\end{pmatrix}$$

With $y = (\mathbf{V}, \mathbf{z}_a, \mathbf{z}_g)$ and

$$f_I(y) = \begin{pmatrix}A\mathbf{V}\\ 0\\ 0\end{pmatrix} \qquad f_E(y) = \begin{pmatrix}-I_{ion}(\mathbf{V}, \mathbf{z}_a, \mathbf{z}_g)\\ g_a(\mathbf{V}, \mathbf{z}_a, \mathbf{z}_g)\\ 0\end{pmatrix} \qquad f_g(y) = \Lambda(y)(y - y_\infty(y)) = \begin{pmatrix}0 & 0 & 0\\ 0 & 0 & 0\\ 0 & 0 & \Lambda_g(\mathbf{V})\end{pmatrix}\begin{pmatrix}0\\ 0\\ \mathbf{z}_g - \mathbf{z}_{g,\infty}(\mathbf{V})\end{pmatrix}$$

We get the ODE:

$$y' = f_I(y) + f_E(y) + f_g(y)$$

Hybrid Spectral Deferred Correction

Consider

$$y' = f_I(y) + f_E(y) + f_g(y)$$

with $y(t_n) = y_n$ and $f_g(y) = \Lambda(y)(y - y_\infty(y))$. Then

$$\begin{aligned} y' &= \Lambda(y_n)(y - y_n) - \Lambda(y_n)(y - y_n) + f_I(y) + f_E(y) + f_g(y) \\ &= \Lambda(y_n)(y - y_n) + g(y)\,. \end{aligned}$$

Applying variation of constants:

$$y(t) = y_n + \int_{t_n}^{t} e^{(t-s)\Lambda(y_n)} g(y(s))\mathrm{d}s\,.$$

Replace $g(y(s))$ with interpolating polynomial

$$g(y(s)) \approx \sum_{j=1}^{M} g(y_{n,j})\ell_l(s),$$

with $0 < c_1 < \ldots < c_M = 1$ collocation nodes and

$$y_{n,j} \approx y(t_n + \Delta t c_j),$$

yields system:

$$y_{n,i} = y_n + \Delta t \sum_{j=1}^{M} a_{ij}(\Delta t \Lambda(y_n)) g(y_{n,j}), \quad i = 1, \ldots, M,$$

with

$$a_{ij}(z) = \int_0^{c_i} e^{(c_i - s)z} \ell_j(s) \mathrm{d}s\,.$$

Recall $y = (\mathbf{V}, \mathbf{z}_a, \mathbf{z}_g)$ and

$$\Lambda(y_n) = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & \Lambda_g(\mathbf{V}_n) \end{pmatrix},$$

thus

$$a_{ij}(\Delta t \Lambda(y_n)) = \begin{pmatrix} a_{ij}(0) & 0 & 0 \\ 0 & a_{ij}(0) & 0 \\ 0 & 0 & a_{ij}(\Delta t \Lambda_g(\mathbf{V}_n)) \end{pmatrix}.$$

Hybrid Spectral Deferred Correction

System

$$y_{n,i} = y_n + \Delta t \sum_{j=1}^{M} a_{ij}(\Delta t \Lambda(y_n)) g(y_{n,j}), \quad i = 1, \ldots, M,$$

is compactly written

$$(I - \Delta t \mathbf{A}(\Delta t \Lambda(y_n)\mathbf{G})(\mathbf{y}_n) = \mathbf{1} \otimes y_n,$$
$$\mathbf{C}(\mathbf{y}_n) = \mathbf{1} \otimes y_n,$$

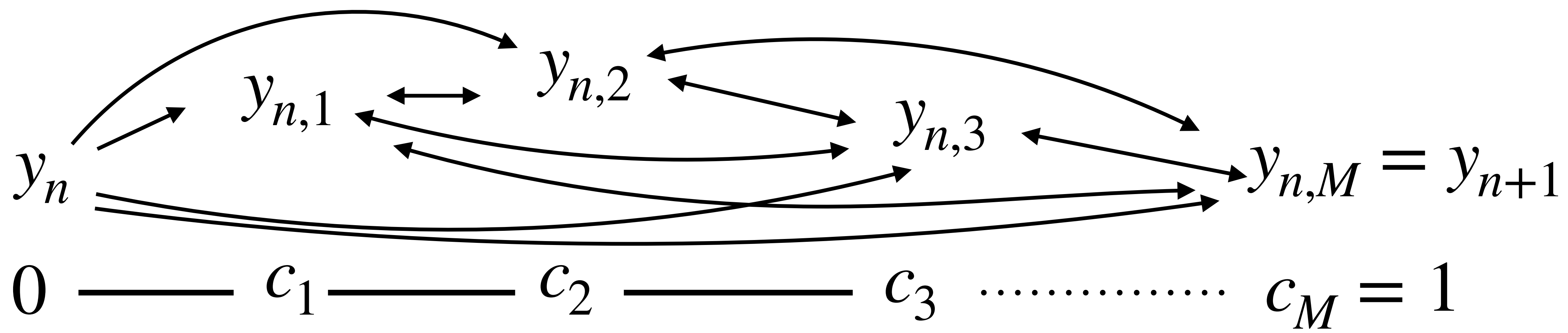
with $\mathbf{y}_n = (y_{n,1}, \ldots, y_{n,M})$, $\mathbf{A}$ matrix of a_{ij}, and $\mathbf{G}$ vector of $g(y_{n,j})$.

Instead of Newton, SDC approach uses preconditioned fixed point iteration:

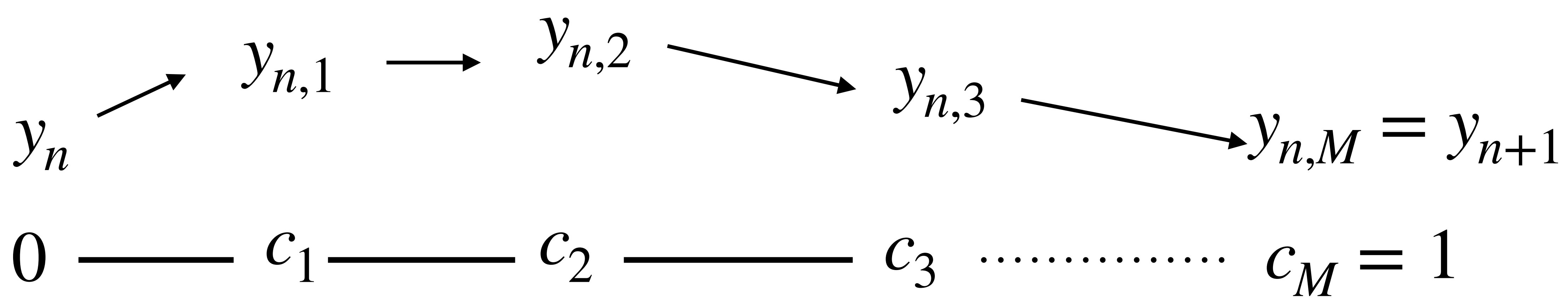
$$\mathbf{P}(\mathbf{y}_n^{k+1}) = \mathbf{P}(\mathbf{y}_n^k) + \mathbf{1} \otimes y_n - \mathbf{C}(\mathbf{y}_n^k),$$

with $\mathbf{P} \approx \mathbf{C}$ but "easy".

$\mathbf{C}$ is defined by an exponential collocation method on the collocation nodes:



We define $\mathbf{P}$ by sequential application of IMEX-Rush-Larsen:



Exact operator $\mathbf{C}$ is hybrid exponential-collocation.

Preconditioner $\mathbf{P}$ is IMEX-Rush-Larsen.

Parallel-in-Time: PFASST Recipe

A sequence of P consecutive steps is given by:

$$t_0 \xrightarrow{\mathbf{C}(\mathbf{y}_0) = \mathbf{1} \otimes y_0,} t_1 \xrightarrow{\mathbf{C}(\mathbf{y}_1) = \mathbf{1} \otimes y_{0,M}} t_2 \xrightarrow{\mathbf{C}(\mathbf{y}_2) = \mathbf{1} \otimes y_{1,M}} t_3 \cdots\cdots t_{P-1} \xrightarrow{\mathbf{C}(\mathbf{y}_{P-1}) = \mathbf{1} \otimes y_{P-2,M}} t_P$$

which is written: $\mathbf{D}(\mathbf{z}) = \mathbf{b}$,

with $\mathbf{z} = (\mathbf{y}_0, \ldots, \mathbf{y}_{P-1})$, $\mathbf{b} = (\mathbf{1} \otimes y_0, 0, \ldots, 0)$ and

$$\mathbf{D} = \mathrm{diag}(\mathbf{C}, \ldots, \mathbf{C}) - \mathbf{H}.$$

Where $\mathbf{H}$ is the matrix taking the last node value of a step to be used as initial value in the next one.

The system is again solved with preconditioned fixed point

$$\mathbf{Q}(\mathbf{z}^{k+1}) = \mathbf{Q}(\mathbf{z}^k) + \mathbf{b} - \mathbf{D}(\mathbf{z}^k)$$

and two preconditioners are available:

$$\mathbf{Q}^{ser} = \mathrm{diag}(\mathbf{P}, \ldots, \mathbf{P}) - \mathbf{H}, \qquad \mathbf{Q}^{par} = \mathrm{diag}(\mathbf{P}, \ldots, \mathbf{P}).$$

Parallel-in-Time: PFASST Recipe

A sequence of P consecutive steps is given by:

$$t_0 \xrightarrow{\mathbf{C}(\mathbf{y}_0) = \mathbf{1} \otimes y_0,} t_1 \xrightarrow{\mathbf{C}(\mathbf{y}_1) = \mathbf{1} \otimes y_{0,M}} t_2 \xrightarrow{\mathbf{C}(\mathbf{y}_2) = \mathbf{1} \otimes y_{1,M}} t_3 \quad \cdots\cdots \quad t_{P-1} \xrightarrow{\mathbf{C}(\mathbf{y}_{P-1}) = \mathbf{1} \otimes y_{P-2,M}} t_P$$

which is written: $\mathbf{D}(\mathbf{z}) = \mathbf{b}$,

with $\mathbf{z} = (\mathbf{y}_0, \ldots, \mathbf{y}_{P-1})$, $\mathbf{b} = (\mathbf{1} \otimes y_0, 0, \ldots, 0)$ and

$$\mathbf{D} = \operatorname{diag}(\mathbf{C}, \ldots, \mathbf{C}) - \mathbf{H}.$$

Where $\mathbf{H}$ is the matrix taking the last node value of a step to be used as initial value in the next one.

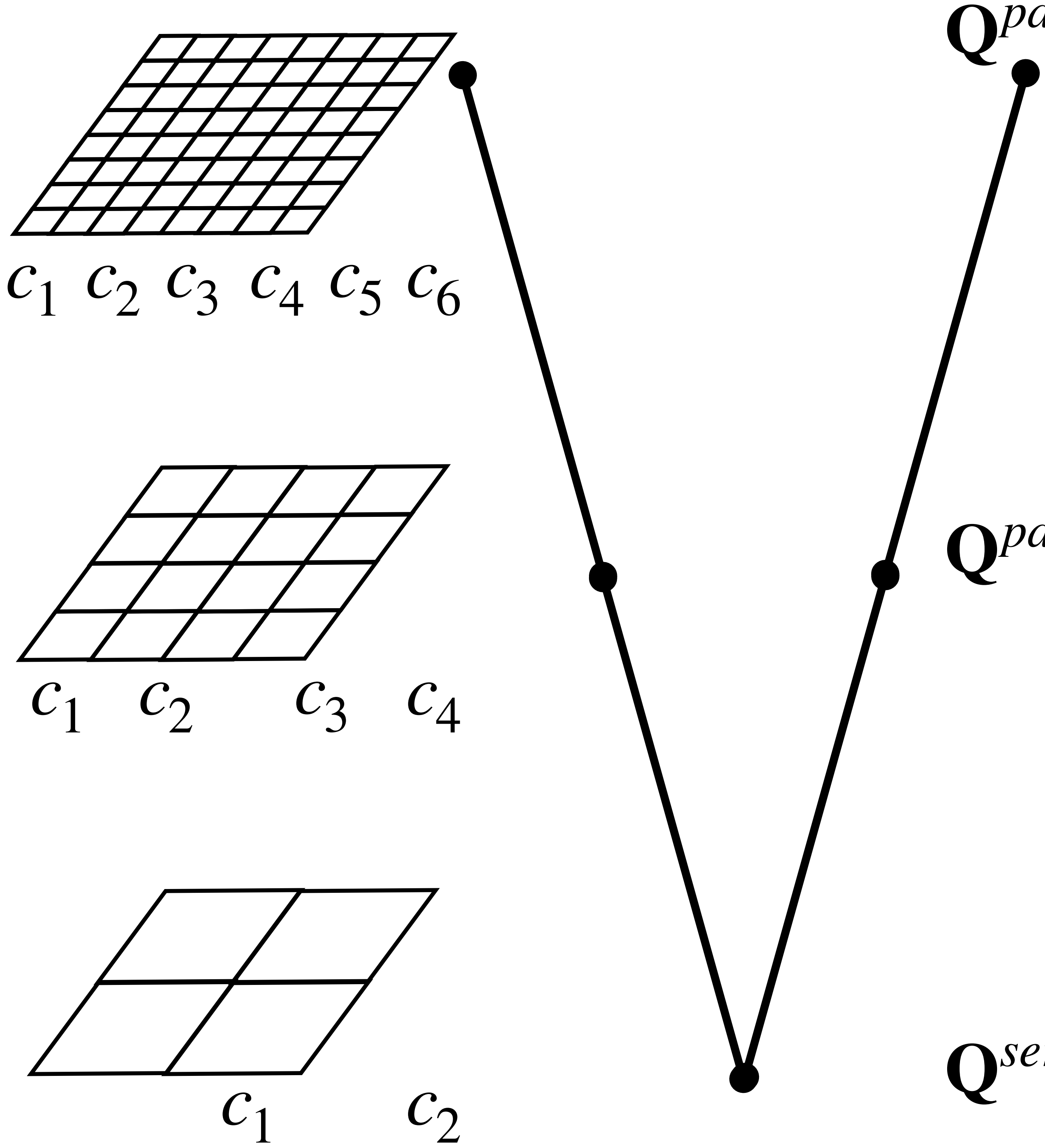
The system is again solved with preconditioned fixed point

$$\mathbf{Q}(\mathbf{z}^{k+1}) = \mathbf{Q}(\mathbf{z}^k) + \mathbf{b} - \mathbf{D}(\mathbf{z}^k)$$

and two preconditioners are available:

$$\mathbf{Q}^{ser} = \operatorname{diag}(\mathbf{P}, \ldots, \mathbf{P}) - \mathbf{H}, \qquad \mathbf{Q}^{par} = \operatorname{diag}(\mathbf{P}, \ldots, \mathbf{P}).$$

Adding nonlinear multigrid:



$$\mathbf{Q}^{par}(\mathbf{z}^{k+1}) = \mathbf{Q}^{par}(\mathbf{z}^k) + \mathbf{b} - \mathbf{D}(\mathbf{z}^k$$

plus τ and coarse grid corrections

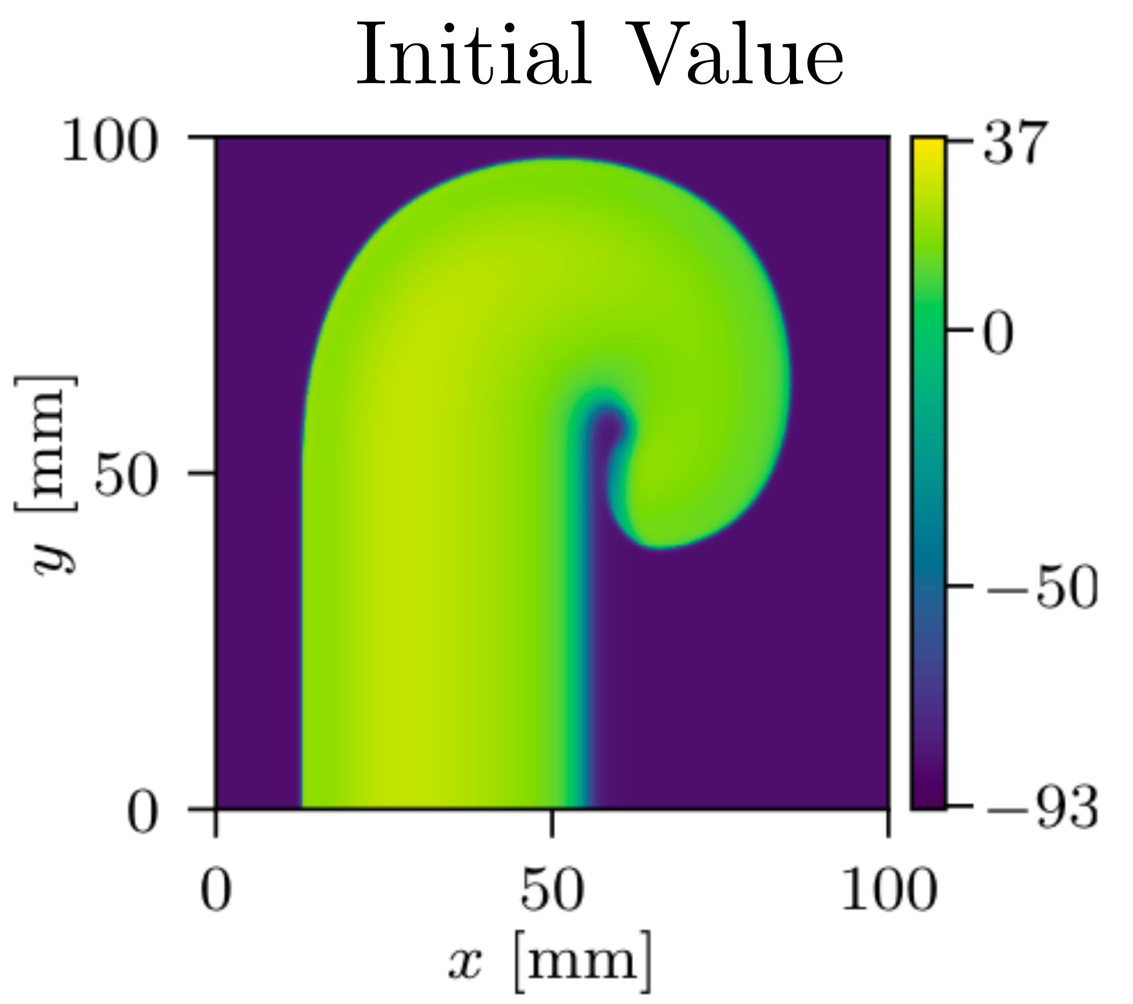
$$\mathbf{Q}^{par}(\mathbf{z}^{k+1}) = \mathbf{Q}^{par}(\mathbf{z}^k) + \mathbf{b} - \mathbf{D}(\mathbf{z}^k$$

$$\mathbf{Q}^{ser}(\mathbf{z}^{k+1}) = \mathbf{Q}^{ser}(\mathbf{z}^k) + \mathbf{b} - \mathbf{D}(\mathbf{z}^k)$$

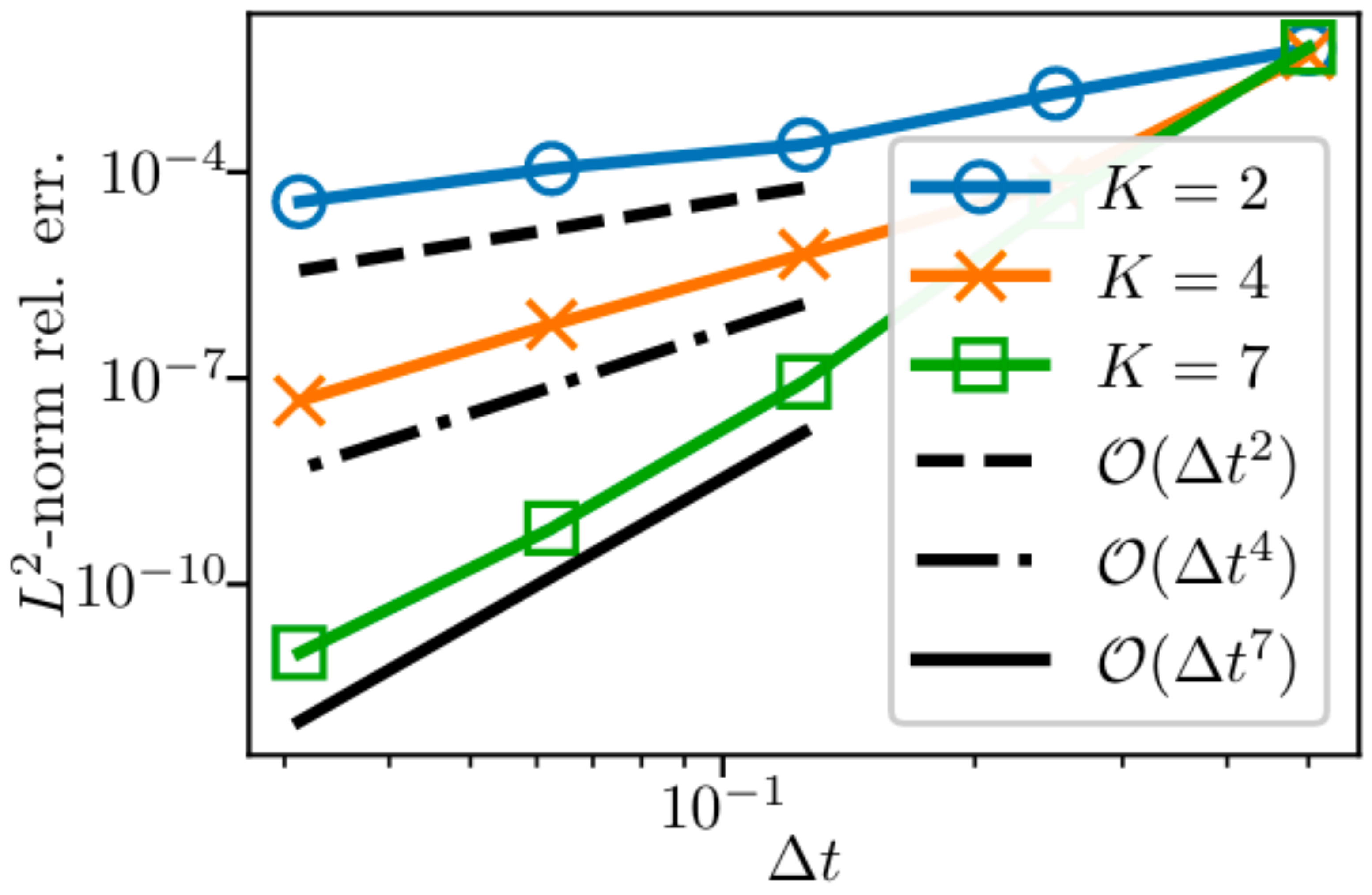
Convergence experiments: Serial setting

Solve Monodomain equation with $\Omega = [0,100] \times [0,100]\text{mm}^2$, $T = 1\text{ms}$, $\Delta x = 0.2\text{mm}$,

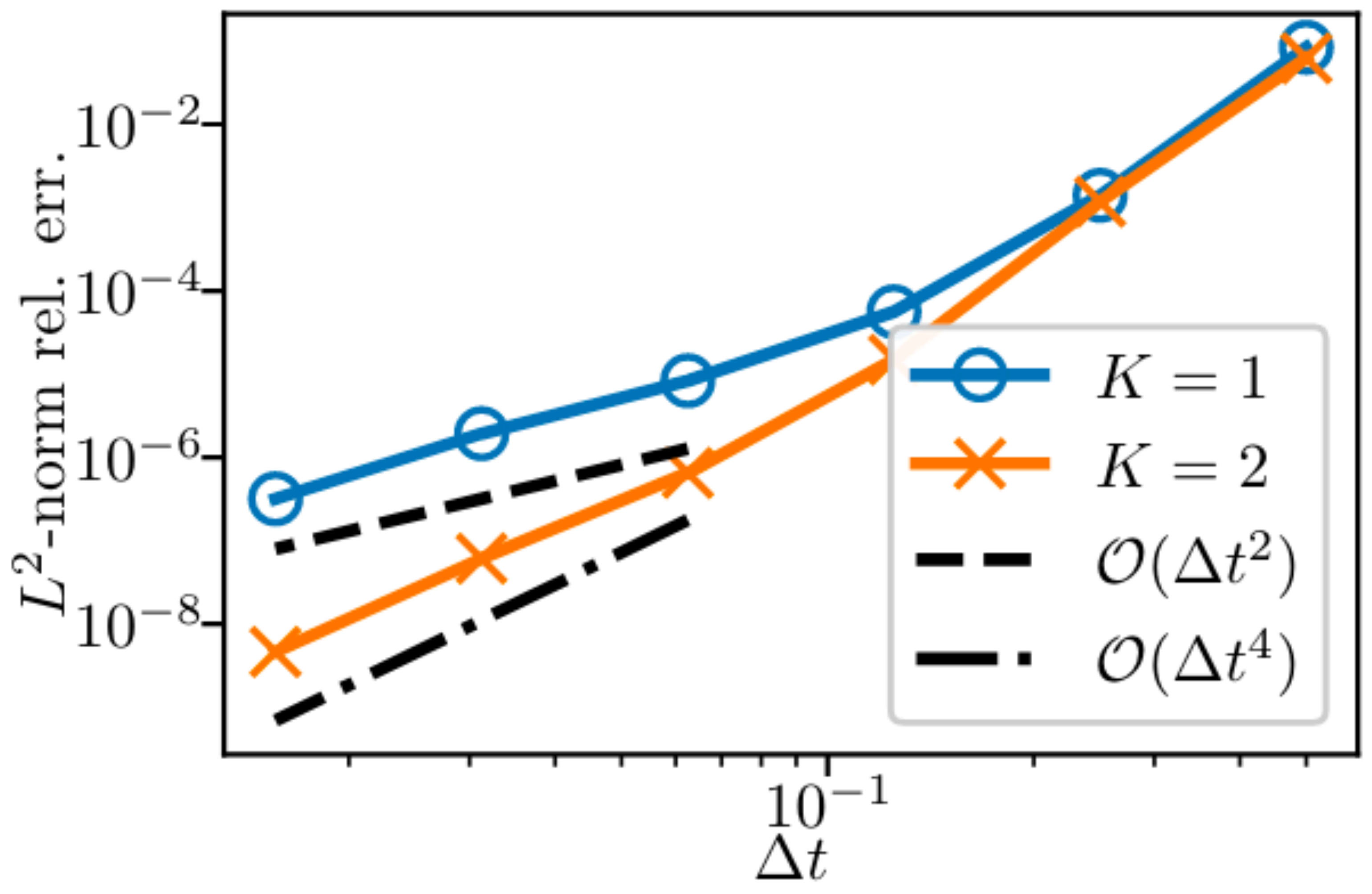
✦ ten Tusscher-Panfilov (smoothed), ✦ Coarsening in time only.



- $P = 1$ serial steps,
- $L = 1$ multigrid levels,
- $M = 6$ collocation nodes.



- $P = 1$ serial steps,
- $L = 2$ multigrid levels,
- $M = 4,2$ collocation nodes.

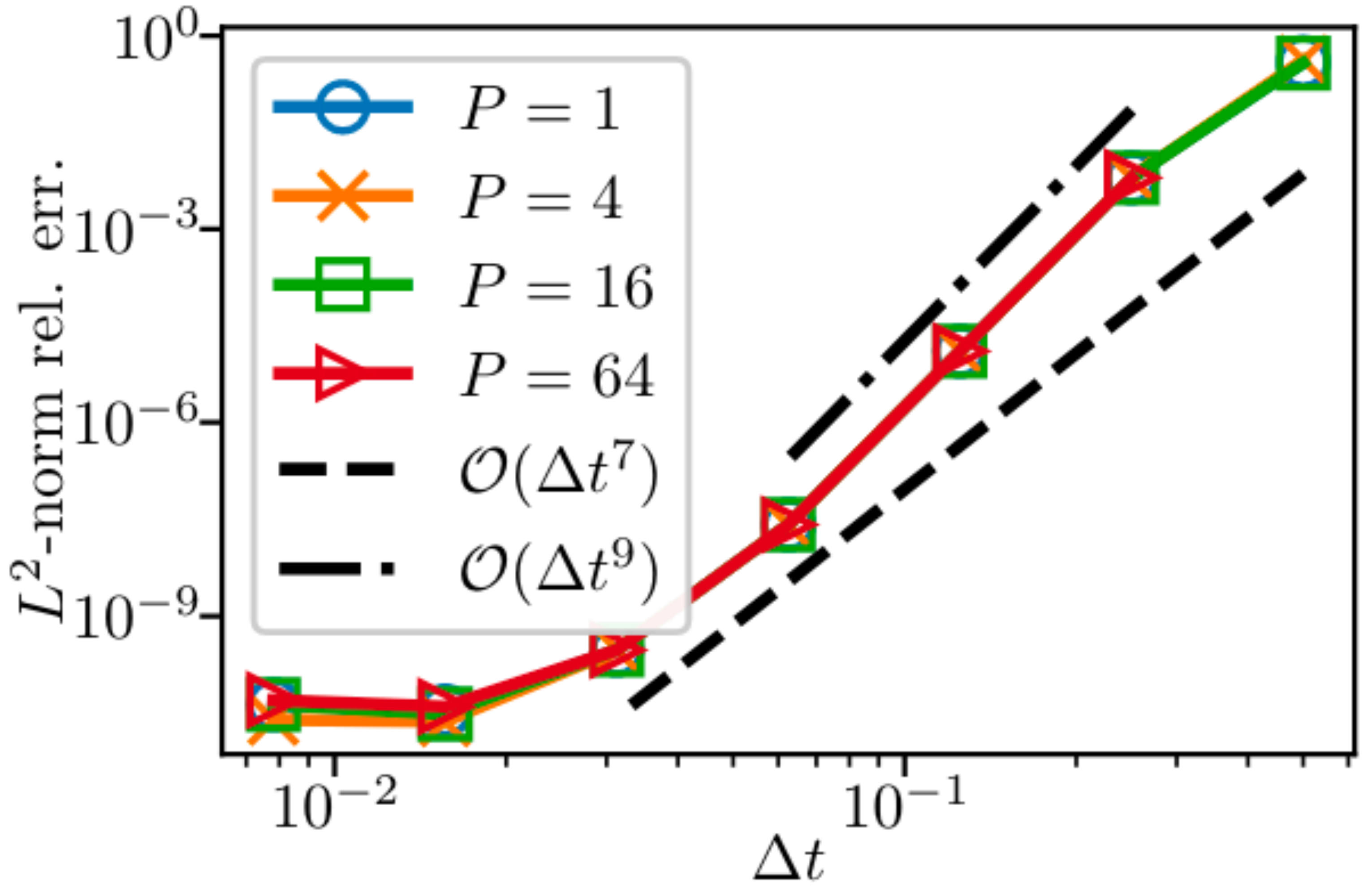


[1]: Radau has order $2M - 1$, but exponential collocation has order $M + 1$.

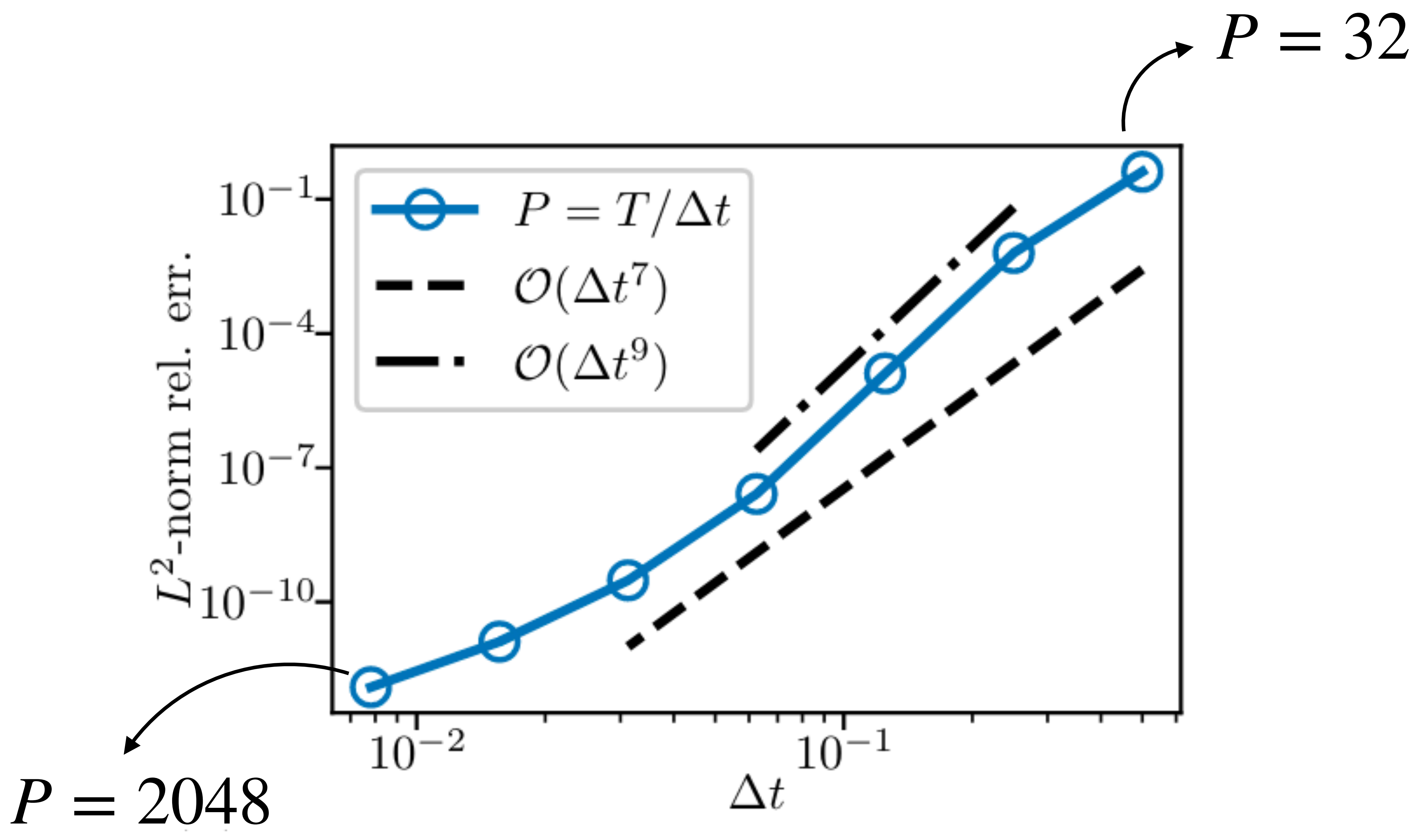
Convergence experiments: Parallel setting

Similar problem as before, but $\Omega = [0{,}100]$mm and $T = 16$ms.

- $P = 1{,}4{,}16{,}64$ parallel steps,
- $L = 2$ multigrid levels,
- $M = 6{,}3$ collocation nodes.



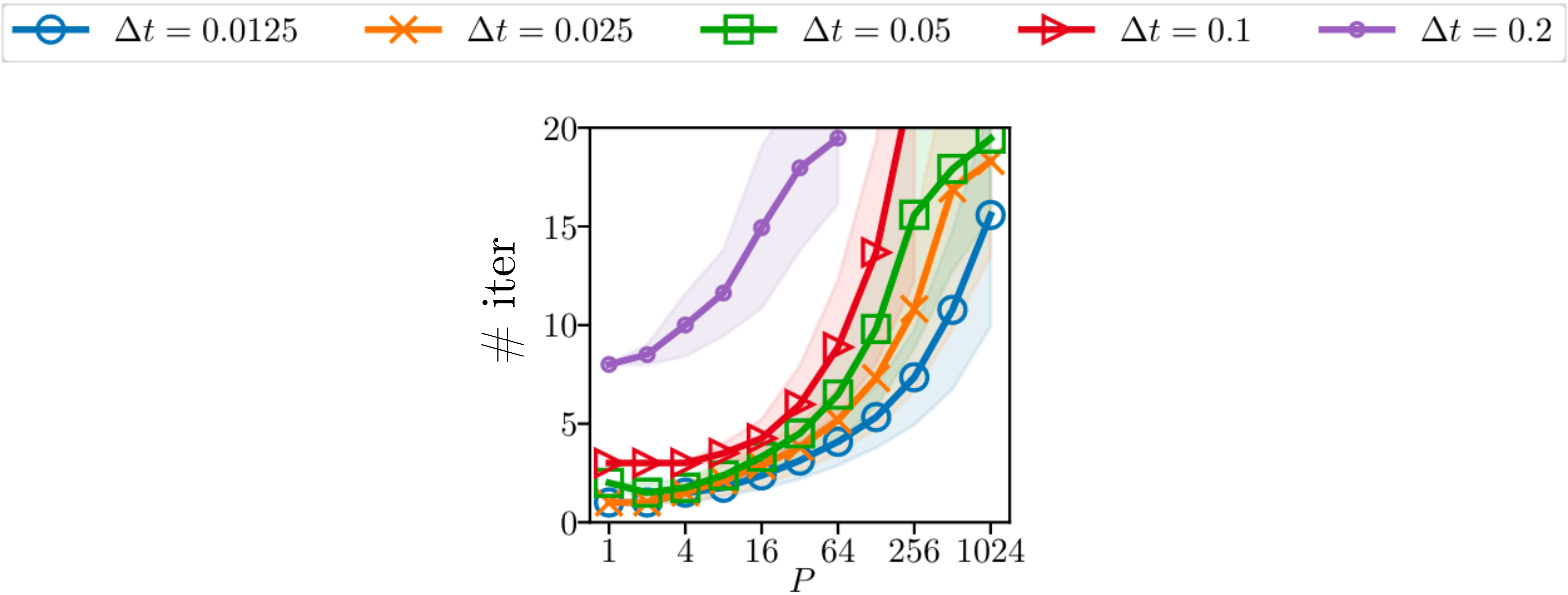
- $P = T/\Delta t$ (whole interval in parallel),
- $L = 2$ multigrid levels,
- $M = 6{,}3$ collocation nodes.



P and Δt VS Iterations

Check how number of iterations is affected by:

- number of processors P,
- step size Δt.



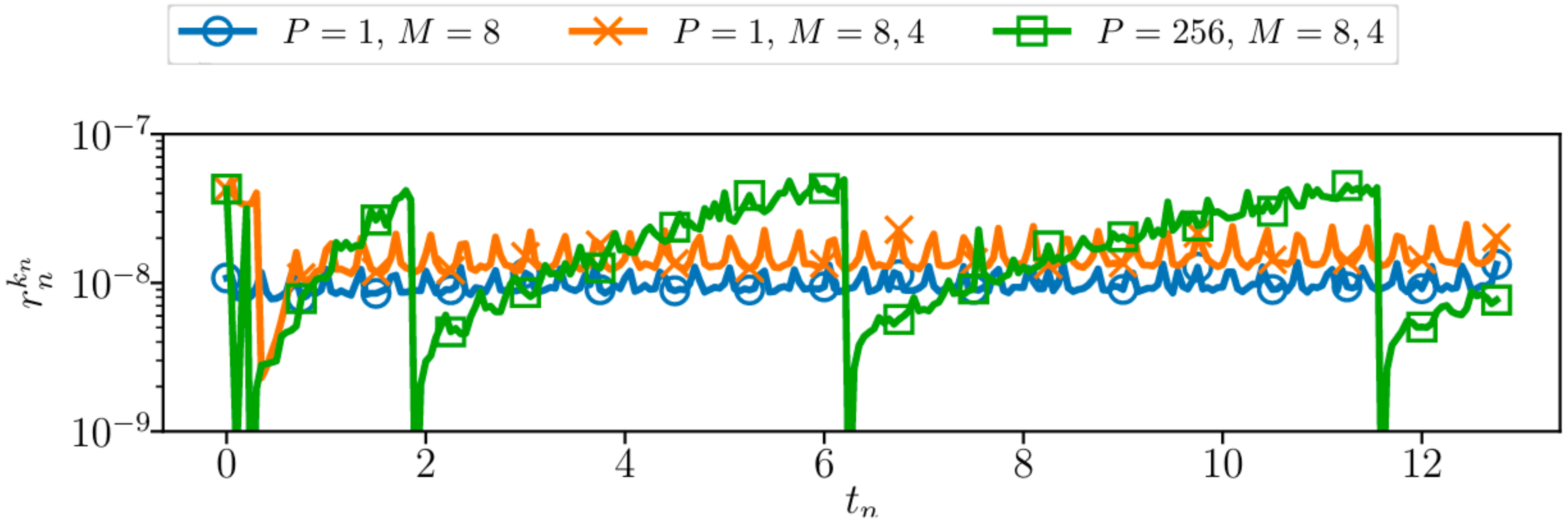
$L = 3$, with $M = 8, 4, 2$.

Average number of iterations versus number of processors, for different step sizes. Shaded areas represent standard deviation.

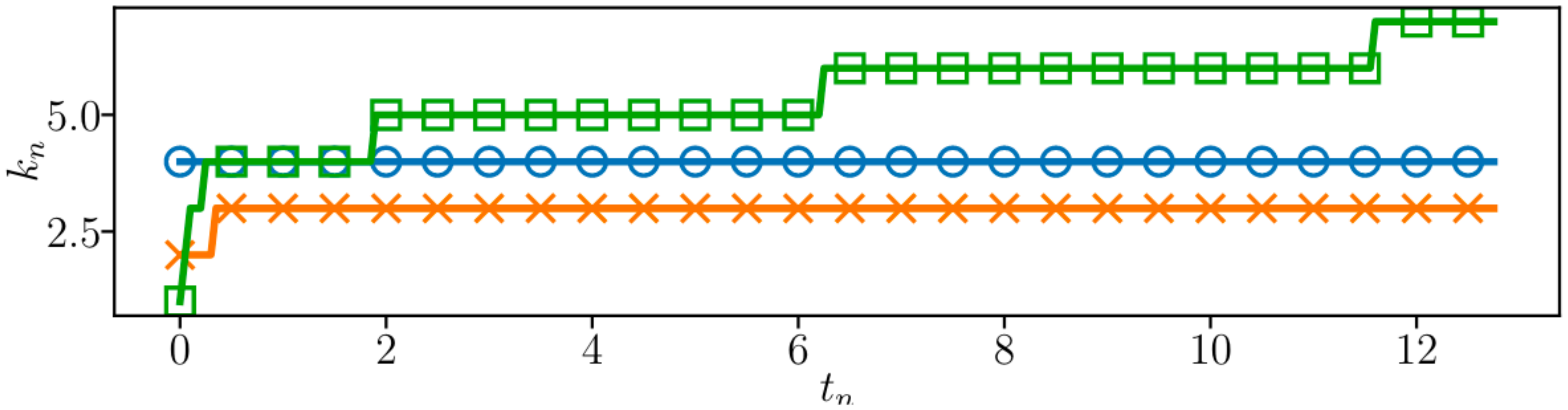
Iterations and residuals over time

Monodomain with $\Delta t = 0.05$ms, up to $T = 256\Delta t = 12.8$ms.

Compare the residuals and iterations of serial single-level and multilevel-methods and a parallel method.



(a) Relative residuals at last iteration



(b) Iterations needed for convergence over time.

The end

Thank you for your attention 😁

Funding: This project has received funding from the European High-Performance Computing Joint Undertaking (JU) under grant agreements No 955495 (MICROCARD) and No 955701 (TIME-X). The JU receives support from the European Union's Horizon 2020 research and innovation programme and Belgium, France, Germany, Switzerland.