Introduction to Parallel-in-Time methods

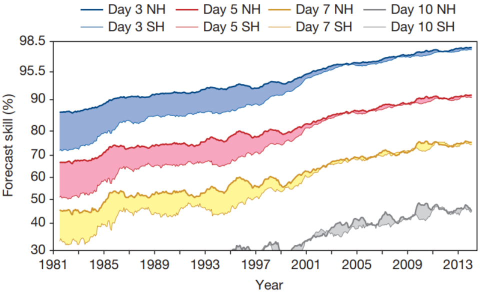
João Guilherme Caldas Steinstraesser

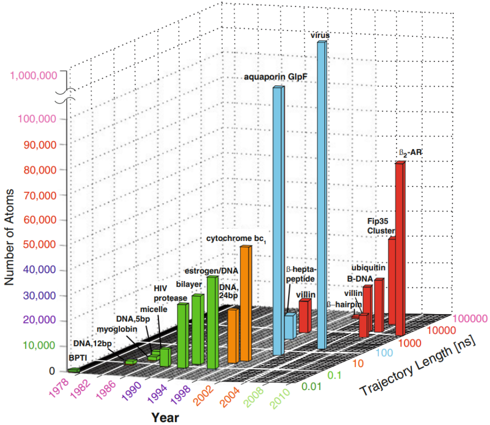
Instituto de Matemática e Estatística - Universidade de São Paulo

October 04, 2024

Table of Contents

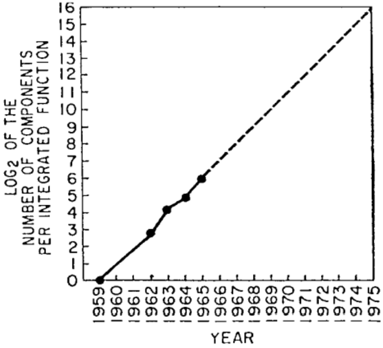
- Trade-off between accuracy, stability and computational cost
- Limited computational resources
- **Need for more efficient computational approaches**

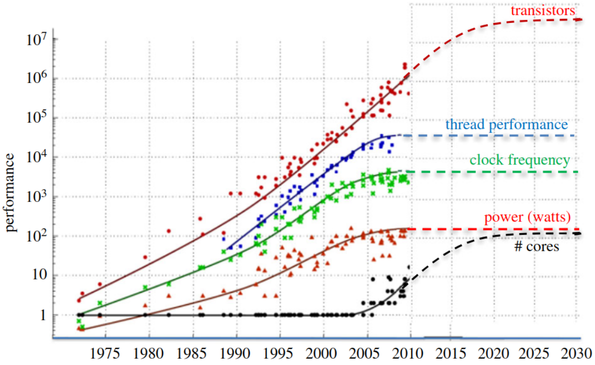




Figures: Bauer, Thorpe, and Brunet (2015) (data from ECMWF); Schlick (2010)

- End of Moore's law [1]? $\implies$ performance gains via parallelism[2]

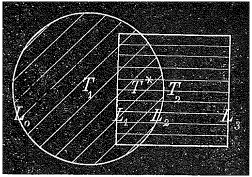


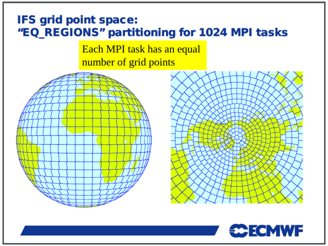


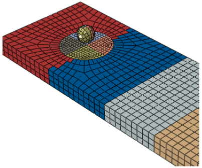
1 Moore, "Cramming more components onto integrated circuits" (1965).
2 Shalf, "The future of computing beyond Moore's Law" (2020).
Figures: Moore (1965), Shalf (2020)

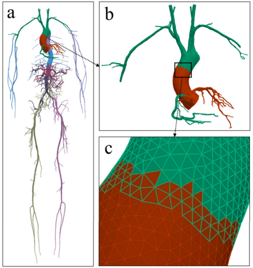
Spatial parallelism

- Schwarz Domain Decomposition methods (DDM)[1]: 1870
- Schwarz methods in parallel computing[2]: 1980s
- Well established methods, operational use



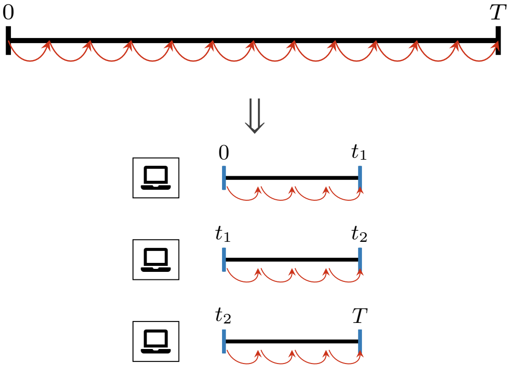






1 Schwarz, "Über einen Grenzübergang durch alternierendes Verfahren" (1870).
2 P. L. Lions, "On the Schwarz Alternating Method I" (1988).
Figures: Schwarz (1870); Mozdzynski (2012); Abaqus (2016); Qin et al. (2023)

- Saturation of parallel gains: how to overcome scalability limit?
- Is it possible to compute time steps in parallel?
- Causality principle?



Parallel-in-Time (PinT) methods

1 Introduction

- Relatively recent development: first ideas in 1964[1]

Parallel Methods for Integrating Ordinary Differential Equations

J. Nievergelt
University of Illinois, Urbana, Illinois

This paper is dedicated to the proposition that, in order to take full advantage for real-time computations of highly parallel computers as can be expected to be available in the near future, much of numerical analysis will have to be recast in a more "parallel" form. By this is meant that serial algorithms ought to be replaced by algorithms which consist of several subtasks which can be computed without knowledge of the results of the other subtasks. As an example, a method is proposed for "parallelizing" the numerical integration of an ordinary differential equation, which process, by all standard methods, is entirely serial.

"... highly parallel computers as can be expected in the near future..."

Introduction

For the last 20 years, one has tried to speed up numerical computation mainly by providing ever faster computers. First, this was achieved by improvements in hardware. Today, as it appears that one is getting closer to the maximal speed of electronic components, the emphasis is put on improving the logical organization of computers, allowing certain operations to be performed in parallel or at least in overlapping time intervals. In order to achieve this, designers are quite willing to introduce considerable redundancy in hardware and to use methods (for instance for performing arithmetic operations) which are inefficient as far as the ratio (time)/(cost) is concerned, as long as the total time is minimized.

"... it appears that one is getting closer to the maximal speed of electronic components..."

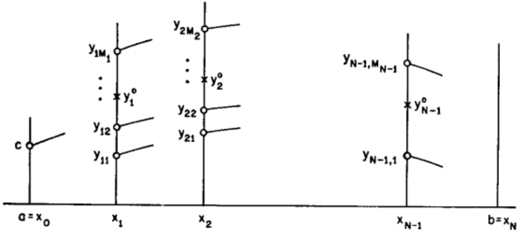


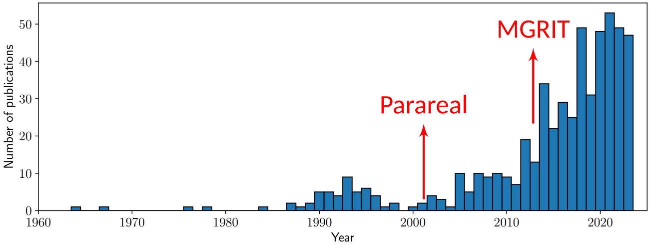
Fig. 1

1 Nievergelt, "Parallel methods for integrating ordinary differential equations" (Dec. 1964).
Figure: Nievergelt (1964)

Parallel-in-Time (PinT) methods

- Increasing interest since 1990s, 2000s
- Three classes of PinT methods[1]: parallelization across
  – the time – the problem – the space
- Review: "50 years of Time Parallel Time Integration" (M. Gander)[2]

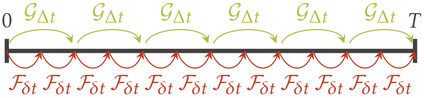
Number of publications on PinT per year [parallel-in-time.org]

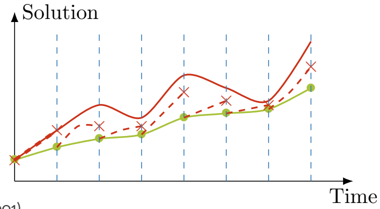


1 Bellen and Zennaro, "Parallel algorithms for initial-value problems for difference and differential equations" (May 1989).
2 M. J. Gander, "50 Years of Time Parallel Time Integration" (2015).

Table of Contents

- Parallelization of the temporal domain
- Iterative predictor-corrector algorithms
  - Fine scheme: accurate and expensive $\implies$ computed in parallel
  - Coarse scheme: less accurate and less expensive $\implies$ computed sequentially
- **Parareal**[1]: two-level with arbitrary schemes
- **MGRIT**[2]: multi-level with arbitrary schemes
- **PFASST**[3]: multi-level using spectral deferred correction (SDC)

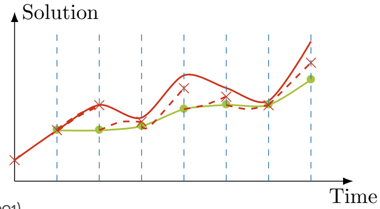




1 J.-L. Lions, Maday, and Turinici, “Résolution d’EDP par un schéma en temps “pararéel”” (Apr. 2001).
2 Friedhoff et al., “A Multigrid-in-Time Algorithm for Solving Evolution Equations in Parallel” (2013).
3 Emmett and M. Minion, “Toward an efficient parallel in time method for partial differential equations” (Mar. 2012).

Parallelization across the time or across the steps

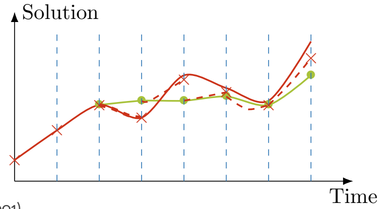
- Parallelization of the temporal domain
- Iterative predictor-corrector algorithms
  - Fine scheme: accurate and expensive $\implies$ computed in parallel
  - Coarse scheme: less accurate and less expensive $\implies$ computed sequentially
- **Parareal**[1]: two-level with arbitrary schemes
- **MGRIT**[2]: multi-level with arbitrary schemes
- **PFASST**[3]: multi-level using spectral deferred correction (SDC)



1 J.-L. Lions, Maday, and Turinici, "Résolution d'EDP par un schéma en temps "pararéel"" (Apr. 2001).
2 Friedhoff et al., "A Multigrid-in-Time Algorithm for Solving Evolution Equations in Parallel" (2013).
3 Emmett and M. Minion, "Toward an efficient parallel in time method for partial differential equations" (Mar. 2012).

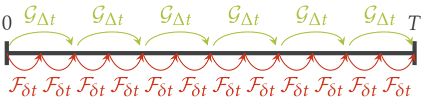
Parallelization across the time or across the steps

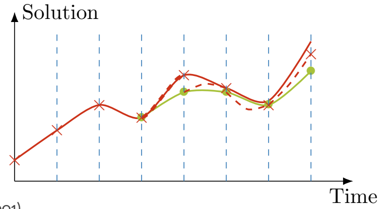
- Parallelization of the temporal domain
- Iterative predictor-corrector algorithms
  - Fine scheme: accurate and expensive $\Longrightarrow$ computed in parallel
  - Coarse scheme: less accurate and less expensive $\Longrightarrow$ computed sequentially
- **Parareal**[1]: two-level with arbitrary schemes
- **MGRIT**[2]: multi-level with arbitrary schemes
- **PFASST**[3]: multi-level using spectral deferred correction (SDC)



1 J.-L. Lions, Maday, and Turinici, “Résolution d’EDP par un schéma en temps “pararéel”” (Apr. 2001).
2 Friedhoff et al., “A Multigrid-in-Time Algorithm for Solving Evolution Equations in Parallel” (2013).
3 Emmett and M. Minion, “Toward an efficient parallel in time method for partial differential equations” (Mar. 2012).

- Parallelization of the temporal domain
- Iterative predictor-corrector algorithms
  - Fine scheme: accurate and expensive $\Longrightarrow$ computed in parallel
  - Coarse scheme: less accurate and less expensive $\Longrightarrow$ computed sequentially
- **Parareal**[1]: two-level with arbitrary schemes
- **MGRIT**[2]: multi-level with arbitrary schemes
- **PFASST**[3]: multi-level using spectral deferred correction (SDC)



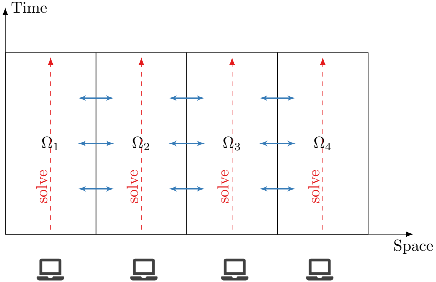


1 J.-L. Lions, Maday, and Turinici, “Résolution d’EDP par un schéma en temps “pararéel”” (Apr. 2001).
2 Friedhoff et al., “A Multigrid-in-Time Algorithm for Solving Evolution Equations in Parallel” (2013).
3 Emmett and M. Minion, “Toward an efficient parallel in time method for partial differential equations” (Mar. 2012).

Parallelization across the problem or across the system

- Parallelization of the spatial domain
- Iterative methods: Schwarz Waveform Relaxation Methods (SWR)[1]:
  - Independent integrations in each subdomain
  - Communication between subdomains in time windows
- Derivation and approximation of optimal interface conditions[2]
- Length of time windows and overlap between subdomains

$$
\begin{cases}
(u_i^k)_t + L(u_i^k) = f, & \boldsymbol{x} \in \Omega_i, 0 < t \leq T \\
u_i^k(0, \boldsymbol{x}) = u_0(\boldsymbol{x}), & \boldsymbol{x} \in \Omega_i \\
\mathcal{B}_{ij}(u_i^k) = \mathcal{B}_{ij}(u_j^{k-1}), & \boldsymbol{x} \in \Gamma_{ij}, 0 < t \leq T
\end{cases}
$$



1 Lelarasmee, Ruehli, and Sangiovanni-Vincentelli, “The Waveform Relaxation Method for Time-Domain Analysis of Large Scale Integrated Circuits” (July 1982).
2 M. Gander, Halpern, and F., “Optimal convergence for overlapping and non-overlapping Schwarz waveform relaxation” (1999).

Parallelization across the method

- Direct (non-iterative) solvers
- Example: Nievergelt (1964)[1]
  - Compute $y_0^0 = c, y_1^0, \dots, y_N^0$ using a coarse solver;
  - Define starting points $y_{i,1}, \dots, y_{i,M_i}$, $M_0 = 1$, around each y_i^0 and compute accurate trajectories $y_{i,1}(x), \dots, y_{i,M_i}(x)$ along $[x_i, x_{i+1}]$ (in parallel)
  - Starting from $Y_1 = y_{0,1}(x_1)$, find $y_{i,j}$ and $y_{i,j+1}$ containing Y_i and linearly interpolate their respective trajectories to obtain Y_{i+1}.

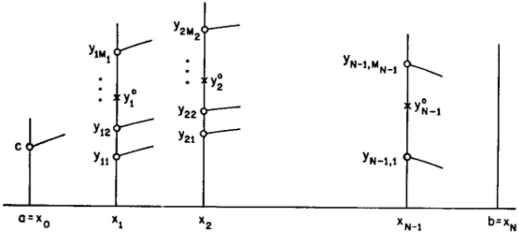


FIG. 1

1 Nievergelt, "Parallel methods for integrating ordinary differential equations" (Dec. 1964).
Figure: Nievergelt (1964)

Parallelization across the method

- Direct (non-iterative) solvers
- Example: ParaDiag[1]
  - Diagonalization of the "all-at-once" time-stepping matrix B:

$$\frac{\boldsymbol{u}_{n+1} - \boldsymbol{u}_n}{\Delta t_n} = L\boldsymbol{u}_n + \boldsymbol{f}_n, \qquad n = 0, \dots, N_{\Delta t} - 1, \qquad \boldsymbol{u}_n \in \mathbb{R}^{N_{\Delta x}}$$

can be written as $(B \otimes I_{N_{\Delta x}} - I_{N_{\Delta t}} \otimes L)\boldsymbol{U} = (I_{N_{\Delta x}} \otimes I_{N_{\Delta t}})\boldsymbol{f}, \qquad \boldsymbol{U} \in \mathbb{R}^{N_{\Delta t} \times N_{\Delta x}}$

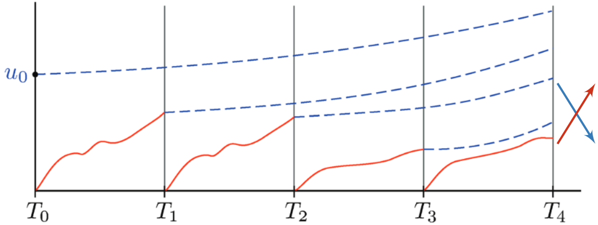
$$B = \begin{pmatrix} \frac{1}{\Delta t_0} & & & \\ -\frac{1}{\Delta t_1} & \frac{1}{\Delta t_1} & & \\ & \ddots & \ddots & \\ & & -\frac{1}{\Delta t_{N_{\Delta t}-1}} & \frac{1}{\Delta t_{N_{\Delta t}-1}} \end{pmatrix}$$

  - If B is **diagonalizable**, then we have $N_{\Delta t}$ independent linear systems.
  - All time step sizes must be different!

1 Maday and Rønquist, "Parallelization in time through tensor-product space–time solvers" (Dec. 2007).

Parallelization across the method

- Direct (non-iterative) solvers
- Example: ParaExp[1]
  - For linear problems $\boldsymbol{u}'(t) = A\boldsymbol{u}(t) + \boldsymbol{g}(t)$
  - Based on the efficient and accurate computation of the matrix exponential e^{tA} ($\star$)
  - Decompose the problem into inhomog. + homog.



Solve $\boldsymbol{v}_n'(t) = A\boldsymbol{v}_n(t) + \boldsymbol{g}(t)$ in parallel

Solve $\boldsymbol{w}_n'(t) = A\boldsymbol{w}_n(t)$ using ($\star$)

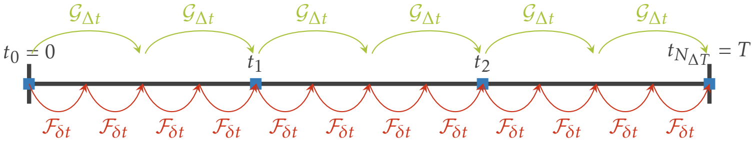
$$u(t) = v_k(t) + \sum_{j=1}^{k} w_j(t), t \in [T_{k-1}, T_k]$$

1 M. J. Gander and Güttel, "PARAEXP: A Parallel Integrator for Linear Initial-Value Problems" (Jan. 2013).
Figure: M. J. Gander and Güttel (2013)

Table of Contents

The Parareal method

- Initially formulated for linear problems[1]
- Rewritten to a simpler form and extended to nonlinear problems[2]
- Solve $\frac{du}{dt}(t) = F(u(t)), \;\; u(0) = u_0$
- Divide the temporal domain into $N_{\Delta T}$ time slices $[t_0, t_1], \ldots, [t_{N_{\Delta t}-1}, t_{N_{\Delta t}}]$
- Iteratively compute approximations U_n^k to $u(t_n)$
- Define:
  - a coarse solver (propagator) $\mathcal{G}_{\Delta t}(U, t_n, t_{n+1})$ (*e.g.*, using $\Delta t \leq \Delta T$);
  - a fine solver (propagator) $\mathcal{F}_{\delta t}(U, t_n, t_{n+1})$ (*e.g.*, using $\delta t < \Delta t$)



1 J.-L. Lions, Maday, and Turinici, "Résolution d'EDP par un schéma en temps "pararéel"" (Apr. 2001).
2 Baffico et al., "Parallel-in-time molecular-dynamics simulations" (Nov. 2002).

- Initialization of the algorithm: serial integration of $\mathcal{G}_{\Delta t}$:

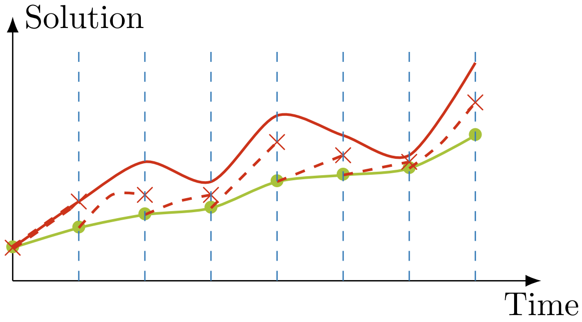
$$U_{n+1}^0 = \mathcal{G}_{\Delta t}(U_n^0, t_n, t_{n+1}), \qquad n = 0, \dots, N_{\Delta T} - 1$$

- In each iteration:

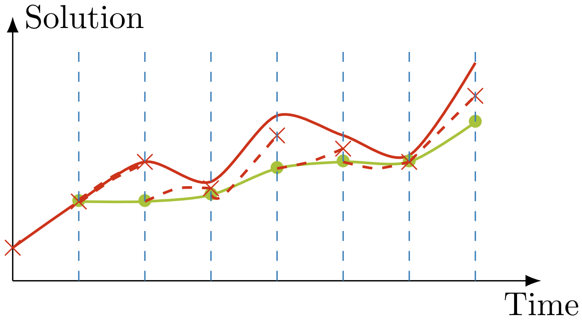
$$U_{n+1}^{k+1} = \underbrace{\mathcal{G}_{\Delta t}(U_n^{k+1}, t_n, t_{n+1})}_{\text{prediction}} + \underbrace{\mathcal{F}_{\delta t}(U_n^k, t_n, t_{n+1}) - \mathcal{G}_{\Delta t}(U_n^k, t_n, t_{n+1})}_{\text{correction}}, \qquad n = 0, \dots, N_{\Delta T} - 1$$

  — Serial coarse prediction (depends on $k+1$)
  — **Parallel** fine corrections (depend only on k)

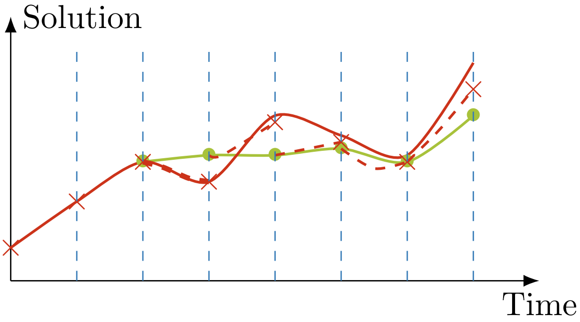
The Parareal method



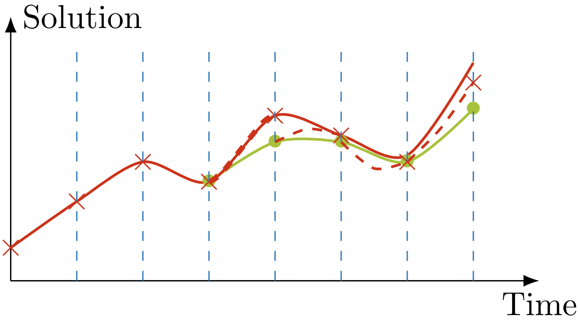
The Parareal method



The Parareal method



The Parareal method

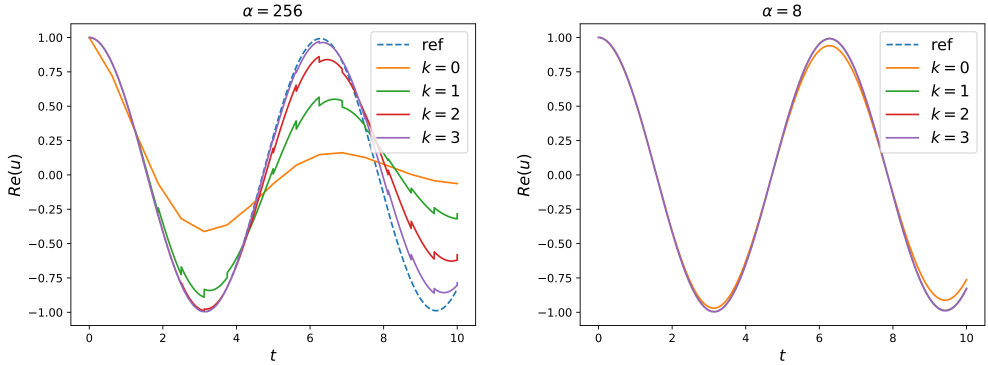


Example of application of Parareal[1]

$$\begin{cases} u'(t) = -iu(t), & 0 < t \leq T = 10 \\ u(0) = 1 \end{cases}$$

- $\mathcal{F}_{\delta t}, \mathcal{G}_{\Delta t}$: backward Euler
- $\Delta T = \frac{T}{16}$, $\delta t = \frac{\Delta T}{256}$, $\Delta t = \alpha \delta t$, $\alpha \geq 2$

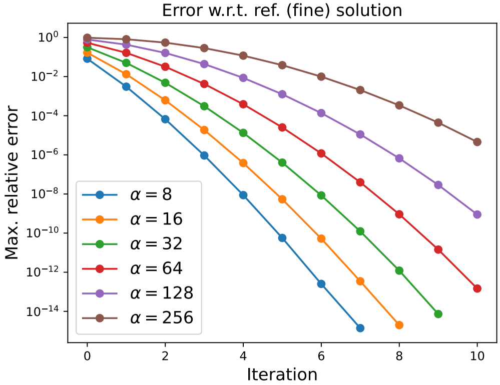
Solution at $t = T$:

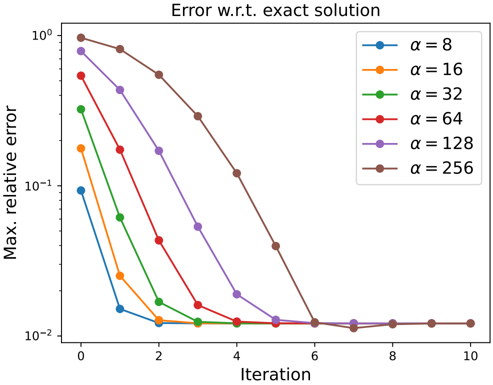


1 Simulations produced using the Python library pyMGRIT: https://pymgrit.github.io/pymgrit/

Example of application of Parareal[1]

- The convergence speed depends on: ΔT, Δt, δt; the problem; $\mathcal{F}_{\delta t}$, $\mathcal{G}_{\Delta t}$, ...
- The accuracy of Parareal is limited by $\mathcal{F}_{\delta t}$



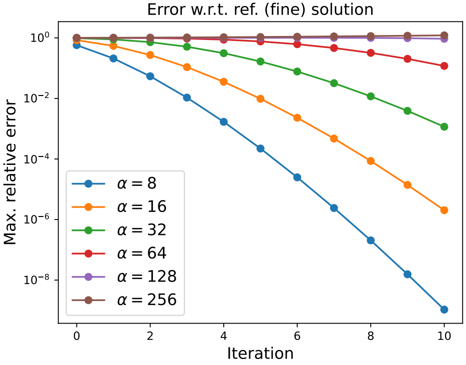


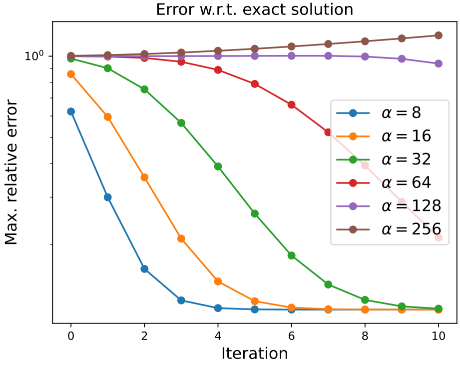
1 Simulations produced using the Python library pyMGRIT: https://pymgrit.github.io/pymgrit/

Example of application of Parareal: what if T is larger?[1]

- $T \leftarrow 10T$, same time step sizes

Relative errors:



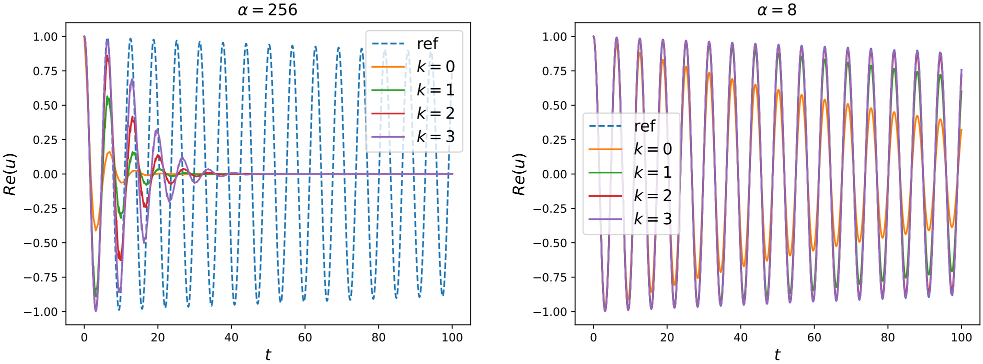


1 Simulations produced using the Python library pyMGRIT: https://pymgrit.github.io/pymgrit/

Example of application of Parareal: what if T is larger?[1]

- $T \leftarrow 10T$, same time step sizes

Solution at $t = T$:



1 Simulations produced using the Python library pyMGRIT: https://pymgrit.github.io/pymgrit/

What is a coarse propagator?

- $\Delta t > \delta t$ is not mandatory!
- The coarse propagator $\mathcal{G}_{\Delta t}$ can be defined by:
  - A larger time step size
  - and/or a coarser spectral resolution
  - and/or a lower-order method
  - and/or a simplified model
  - ...
- **Key points:** the coarse propagator should be:
  - substantially cheaper than the fine propagator
  - still relatively accurate
  - stable

Pros and cons of Parareal (and other predictor-corrector iterative PinT methods)

- Pros:
  - Non-intrusive and flexible: $\mathcal{F}_{\delta t}$ and $\mathcal{G}_{\Delta t}$ are arbitrary;
  - Simple formulation with several interpretations;
  - Relatively simple implementation;
  - Easy-to-use open-source libraries: XBraid[1], PyMGRIT[2], LibPFASST[3], PyPFASST[4].
- Cons:
  - Accuracy is limited by $\mathcal{F}_{\delta t}$;
  - Fast convergence is required for parallel efficiency;
  - Complex task scheduling may be required for parallel efficiency;
  - Unsolved issues when applied to some problems (hyperbolic ones).

1 https://github.com/XBraid/xbraid
2 https://github.com/pymgrit/pymgrit
3 https://github.com/libpfasst/LibPFASST
4 https://github.com/memmett/PyPFASST

Interpretation of Parareal as an iterative linear system solver[1]

- If $\mathcal{F}_{\delta t}$ is linear, then

$$U_0 = u_0$$
$$U_{n+1} = \mathcal{F}_{\delta t}(U_n, t_n, t_{n+1}), \qquad n = 0, \dots N_{\Delta T} - 1$$

can be written as

$$A_f \boldsymbol{U} := \begin{pmatrix} 1 & & & \\ -\mathcal{F}_{\delta t} & 1 & & \\ & \ddots & \ddots & \\ & & -\mathcal{F}_{\delta t} & 1 \end{pmatrix} \begin{pmatrix} U_0 \\ \vdots \\ U_{N_{\Delta T}} \end{pmatrix} = \begin{pmatrix} u_0 \\ 0 \\ \vdots \\ 0 \end{pmatrix} =: \boldsymbol{f}$$

and similarly to $\mathcal{G}_{\Delta t} \implies A_c \boldsymbol{U} = \boldsymbol{f}$.

1 Ruprecht, "Wave propagation characteristics of Parareal" (June 2018).

Interpretation of Parareal as an iterative linear system solver[1]

- The fine system can be solved iteratively using a fixed point method

$$A_f \boldsymbol{U} = \boldsymbol{f} \implies B\boldsymbol{U}^{k+1} = C\boldsymbol{U}^k + \boldsymbol{f}, \qquad A_f = B - C$$

- Defining

$$B = A_c, \qquad C = A_c - A_f$$

  we have

$$A_c\boldsymbol{U}^{k+1} = (A_c - A_f)\boldsymbol{U}^k + \boldsymbol{f} \implies \boldsymbol{U}^{k+1} = (I - A_c^{-1}A_f)\boldsymbol{U}^k + A_c^{-1}\boldsymbol{f}$$

  which is equivalent to the Parareal iteration.
- The convergence depends on the spectral radius of $(I - A_c^{-1}A_f)$.

1 Ruprecht, "Wave propagation characteristics of Parareal" (June 2018).

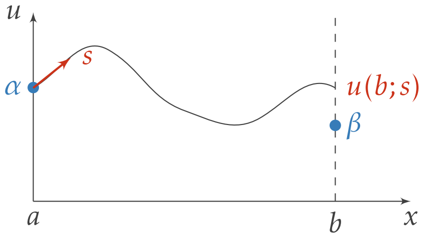
Interpretation of Parareal as a multiple shooting method in time

The shooting method in space:

- Solve a BVP by replacing it by an IVP with unknown initial condition $u'(a) = s$:

$$\begin{cases} u'' = f(x,u,u'), & a < x < b \\ u(a) = \alpha \\ u(b) = \beta \end{cases} \implies \begin{cases} u'' = f(x,u,u'), & a < x < b \\ u(a) = \alpha \\ u'(a) = s \end{cases} \tag{1}$$

- Find the zero of $F(s) := u(x=b;s) - \beta$ using Newton: $s^{(k+1)} = s^{(k)} - \frac{F(s^{(k)})}{F'(s^{(k)})}$

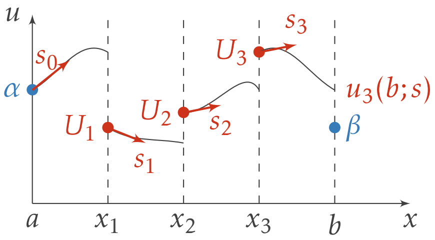


The multiple shooting method in space:

- Divide $[a, b]$ in N intervals and define in each $[x_i, x_{i+1}]$ the IVP

$$\begin{cases} u_i'' = f(x, u_i, u_i'), & x_i < x < x_{i+1} \\ u_i(x_i) = U_i \\ u'(x_i) = s_i \end{cases}$$

- Unknowns: $\boldsymbol{U} := (U_0, \ldots, U_{N-1})^T, \qquad \boldsymbol{s} := (s_0, \ldots, s_{N-1})^T$



The multiple shooting method in space:

- We want to find the zeros of

$$
\begin{aligned}
F_{i+1}(\boldsymbol{U},\boldsymbol{s}) &:= u_i(x_{i+1};U_i,s_i) - U_{i+1}, \qquad i = 0,\dots,N-2 \\
F_N(\boldsymbol{U},\boldsymbol{s}) &:= u_{n-1}(b;U_{N-1},s_{N-1}) - \beta, \\
\hat{F}_{i+1}(\boldsymbol{U},\boldsymbol{s}) &:= u_i'(x_{i+1};U_i,s_i) - s_{i+1}, \qquad i = 0,\dots,N-2
\end{aligned}
$$

- Newtons's method applied to $\boldsymbol{F}(\boldsymbol{U},\boldsymbol{s}) := (F_1,\dots,F_N,\hat{F}_1,\dots,\hat{F}_{N-1})^T = \mathbf{0}$:

$$
\begin{pmatrix} \boldsymbol{U}^{(k+1)} \\ \boldsymbol{s}^{(k+1)} \end{pmatrix} = \begin{pmatrix} \boldsymbol{U}^{(k)} \\ \boldsymbol{s}^{(k)} \end{pmatrix} - [J_{\boldsymbol{F}}(\boldsymbol{U}^{(k)},\boldsymbol{s}^{(k)})]^{-1}\boldsymbol{F}(\boldsymbol{U}^{(k)},\boldsymbol{s}^{(k)})
$$

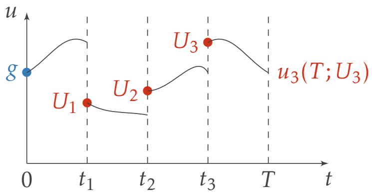
The multiple shooting method in time[1]

- Divide $[0, T]$ in $N_{\Delta T}$ intervals and define smaller IVPs:

$$\begin{cases} u' = f(u), & 0 < t < T \\ u(0) = g \end{cases} \implies \begin{cases} u_n' = f(u_n), & t_n < t < t_{n+1} \\ u_n(t_n) = U_n \end{cases},$$

for $n = 0, \ldots, N_{\Delta T} - 1$, with U_n unknown.

- Let $u_n(t; U_n)$ be the solution of the IVP in $[t_n, t_{n+1}]$ and $\boldsymbol{U} := (U_0, \ldots, U_{N_{\Delta T}-1})^T$.



1 M. J. Gander and Vandewalle, “Analysis of the Parareal Time-Parallel Time-Integration Method” (Jan. 2007).

Interpretation of Parareal as a multiple shooting method in time

- We want to find the zeros of

$$\begin{aligned} F_{n+1}(\boldsymbol{U}) &:= u_n(t_{n+1}; U_n) - U_{n+1}, \qquad n = 0, \dots, N_{\Delta T} - 1 \\ F_0(\boldsymbol{U}) &:= g - U_0 \end{aligned}$$

- Newtons's method applied to $\boldsymbol{F}(\boldsymbol{U}) := (F_0, \dots, F_{N_{\Delta T}})^T$: $\boldsymbol{U}^{k+1} = \boldsymbol{U}^k - J_{\boldsymbol{F}}^{-1}(\boldsymbol{U}^k)\boldsymbol{F}(\boldsymbol{U}^k)$

$$\implies \begin{cases} U_0^{k+1} = g \\ U_{n+1}^{k+1} = u_n(t_{n+1}; U_{n+1}^k) + \frac{\partial u_n}{\partial U_n}(t_{n+1}, U_n^k)(U^{k+1} - U_n^k) \end{cases}$$

- Considering the approximations

$$\begin{aligned} u_n(t_{n+1}; U_{n+1}^k) &= \mathcal{F}_{\delta t}(U_k^n, t_n, t_{n+1}), \\ \frac{\partial u_n}{\partial U_n}(t_{n+1}; U_n^k) &= \frac{\mathcal{G}_{\Delta t}(U_n^{k+1}, t_n, t_{n+1}) - \mathcal{G}_{\Delta t}(U_n^k, t_n, t_{n+1})}{U_n^{k+1} - U_n^k} \end{aligned}$$

we obtain the Parareal method.

A first convergence result of Parareal

$$U_{n+1}^{k+1} = \underbrace{\mathcal{G}_{\Delta t}(U_n^{k+1}, t_n, t_{n+1})}_{\text{prediction}} + \underbrace{\mathcal{F}_{\delta t}(U_n^k, t_n, t_{n+1}) - \mathcal{G}_{\Delta t}(U_n^k, t_n, t_{n+1})}_{\text{correction}}, \qquad n = 0, \dots, N_{\Delta T} - 1$$

- **Theorem**[1]**:**
  Let $u_{\text{ref},n} = \mathcal{F}_{\delta t}(u_0, 0, t_n)$ be the fine serial solution at $t = t_n$. Then, the Parareal solution satisfies $u_n^k = u_{\text{ref},n}\ \forall n \leq k$, *i.e.*, each iteration yields exact convergence at one time slice (at least).

- **Corollary:**
  If there are $N_{\Delta T}$ time slices, the Parareal method converges to the fine serial solution **in at most $N_{\Delta T}$ iterations.**

$\Longrightarrow$ **Parareal converges to the fine solution in a finite number of iterations.**

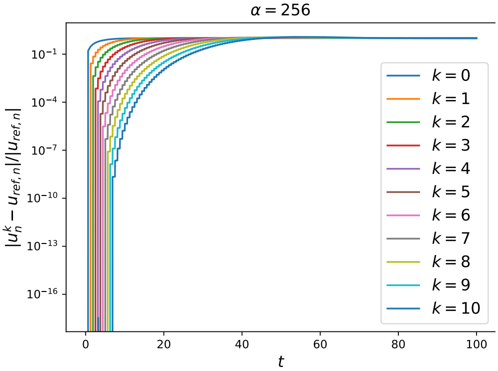
1 M. J. Gander, “Analysis of the Parareal Algorithm Applied to Hyperbolic Problems Using Characteristics” (2008).

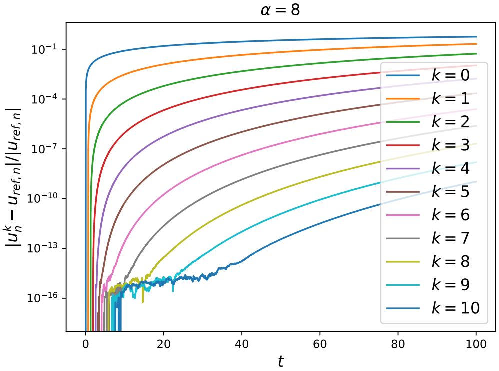
A first convergence result of Parareal[1]

$$\begin{cases} u'(t) = -iu(t), \quad 0 < t \leq T = 100 \\ u(0) = 1 \end{cases}$$

- $\mathcal{F}_{\delta t}, \mathcal{G}_{\Delta t}$: backward Euler
- $\Delta T = \frac{T}{160}$, $\delta t = \frac{\Delta T}{256}$, $\Delta t = \alpha \delta t$, $\alpha \geq 2$

Error vs time at each iteration:





1 Simulations produced using the Python library pyMGRIT: https://pymgrit.github.io/pymgrit/

A first convergence result of Parareal

- Parareal converges in a finite number of iterations ($N_{\Delta T}$)
- However:

$$\text{Cost (time) of an iteration} \geq \text{Cost}(\mathcal{G}_{\Delta t}) + \frac{\text{Cost}(\mathcal{F}_{\delta t})}{N_{\Delta T}}$$

$$\implies \text{Cost of } N_{\Delta T} \text{ iterations} > \text{Cost}(\mathcal{F}_{\delta t})$$

- **Therefore: convergence should be obtained in $\ll N_{\Delta T}$ iterations (at least approximately) in order to**

$$\text{Speedup} := \frac{T_{\text{ref}}}{T_{\text{parareal}}} > 1$$

(using $N_{\Delta T}$ parallel processors)

- Number of fine and coarse time steps: $N_{\delta t} = T/\delta t$, $N_{\Delta t} = T/\Delta t$
- Cost of one fine or coarse time step: τ_f, τ_c
- Cost of the fine serial simulation: $T_{\text{ref}} = N_{\delta t}\tau_f$
- Cost of the coarse serial prediction: $T_c = N_{\Delta t}\tau_c$
- Cost of the fine parallel prediction: $T_f = \frac{N_{\delta t}}{N_{\Delta T}}\tau_f$
- **Speedup** after iteration k:

$$S = \frac{T_{\text{ref}}}{T_c + k(T_c + T_f)} = \frac{1}{(k+1)\frac{N_{\Delta t}}{N_{\delta t}}\frac{\tau_c}{\tau_f} + \frac{k}{N_{\Delta T}}}$$

Goal: $S > 1$ (ideally: $S \gg 1$)

- Speedup:

$$S = \frac{1}{(k+1)\frac{N_{\Delta t}}{N_{\delta t}}\frac{\tau_c}{\tau_f} + \frac{k}{N_{\Delta T}}}$$

- Parallel efficiency:

$$\mathcal{E} = \frac{S}{N_{\text{proc}}} \xRightarrow{N_{\text{proc}}=N_{\Delta T}} \mathcal{E} = \frac{1}{(k+1)N_{\Delta T}\frac{N_{\Delta t}}{N_{\delta t}}\frac{\tau_c}{\tau_f} + k}$$

Bounds on S and $\mathcal{E}$:

1. $S < \frac{N_{\Delta T}}{k}, \mathcal{E} < \frac{1}{k} \implies k$ should be smaller than $N_{\Delta T}$
2. $S < \frac{1}{k+1}\frac{N_{\delta t}}{N_{\Delta t}}\frac{\tau_f}{\tau_c}, \mathcal{E} < \frac{1}{k+1}\frac{1}{N_{\Delta T}}\frac{N_{\delta t}}{N_{\Delta t}}\frac{\tau_f}{\tau_c} \implies \mathcal{G}_{\Delta t}$ should be much less expensive than $\mathcal{F}_{\delta t}$

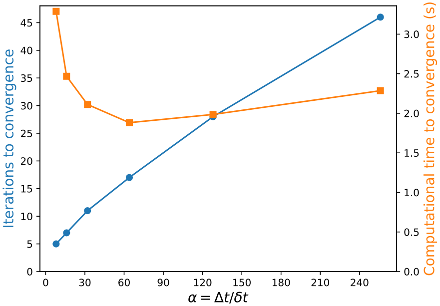
These bounds are conflicting: accuracy of $\mathcal{G}_{\Delta t}$ *vs* convergence speed

Example of application of Parareal: convergence vs computational time[1]

$$\begin{cases} u'(t) = -iu(t), & 0 < t \leq T = 100 \\ u(0) = 1 \end{cases}$$

- $\mathcal{F}_{\delta t}, \mathcal{G}_{\Delta t}$: backward Euler
- $\Delta T = \frac{T}{160}$, $\delta t = \frac{\Delta T}{256}$, $\Delta t = \alpha \delta t$, $\alpha \geq 2$, $N_{\text{proc}} = 16$

- Ref. simulation (serial $\mathcal{F}_{\delta t}$) ≈ 0.75s (it's a very small problem $\implies$ no speedup!).



1 Simulations produced using the Python library pyMGRIT: https://pymgrit.github.io/pymgrit/

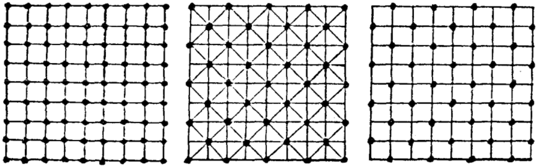
Table of Contents

4 Multigrid Reduction in Time (MGRIT)

- The MultiGrid Reduction In Time (MGRIT) is a multilevel PinT approach inspired on spatial multigrid (MGR)

MGR[1]:

- Iterative solution of linear systems arising from the discretization of spatial-dependent problems.
- Hierarchy of grids:
  - Fine grids: few iterations of a chosen linear system solver (*e.g.*, Gauss-Seidel)
  - Coarse grids: computation of correction terms



1 M. Ries and U. Trottenberg, “MGR-ein blitzschneller elliptischer löser” (1979).
Figure: Manfred Ries, Ulrich Trottenberg, and Winter (1983)

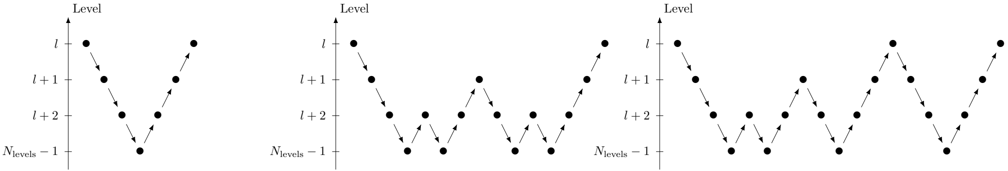
Multigrid Reduction (in space)

$$\begin{cases} \Delta u(\boldsymbol{x}) = f(\boldsymbol{x}), & \boldsymbol{x} \in \Omega \\ u(\boldsymbol{x}) = g(\boldsymbol{x}), & \boldsymbol{x} \in \partial\Omega \end{cases} \implies A\boldsymbol{U} = \boldsymbol{f}$$

- Consider two grids:
  - Fine grid, with mesh size $h \implies A_f \boldsymbol{U}_f = \boldsymbol{f}_f$
  - Coarse grid, with mesh size ph, $p > 1 \implies A_c \boldsymbol{U}_c = \boldsymbol{f}_c$
- Steps of the **Full Approximation Scheme (FAS):**
  1. Relaxation or smoothing on the fine grid: do a few iterations on $A_f \boldsymbol{U}_f = \boldsymbol{f}_f$ and compute the residual $\boldsymbol{r}_f := \boldsymbol{f}_f - A_f \tilde{\boldsymbol{U}}_f$
  2. Restrict the residual to the coarse grid: $\boldsymbol{r}_c := R\boldsymbol{r}_f$
  3. Solve (approximately) $A_c \boldsymbol{e}_c = \boldsymbol{r}_c$ to obtain the coarse grid error $\boldsymbol{e}_c$:
  4. Prolong the error to the fine grid: $\boldsymbol{e}_f := P\boldsymbol{e}_c$
  5. Correct the fine grid solution: $\boldsymbol{U}_f \leftarrow \boldsymbol{U}_f + \boldsymbol{e}_f$ (initial guess for Step 1).
- Using a hierarchy of grids, **Step 3 can be done recursively.**

Parameters defining the spatial MGR

- Number of levels;
- Mesh coarsening;
- Amount of relaxation on each level (number of iterations of the solver)
- Relaxation strategy: V-cycles, W-cycles, F-cycles



Multigrid Reduction In Time (MGRIT)[1]

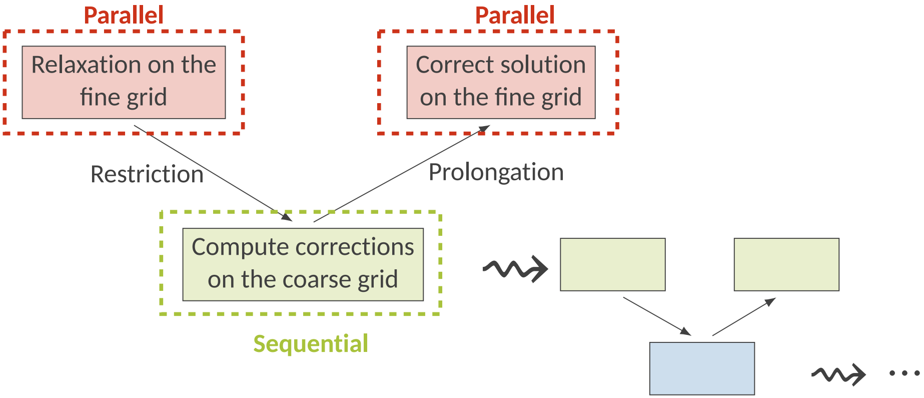
Concepts and vocabulary from MGR are transposed to the spatial domain:

- Grids: temporal grids
- Relaxation: time integration using an approximate scheme
- Amount of relaxation: number of time steps to be computed
- Restriction/interpolation between temporal grids
- V-cycles, W-cycles, F-cycles

1 Friedhoff et al., "A Multigrid-in-Time Algorithm for Solving Evolution Equations in Parallel" (2013).

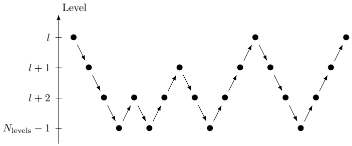
Multigrid Reduction in Time (MGRIT)

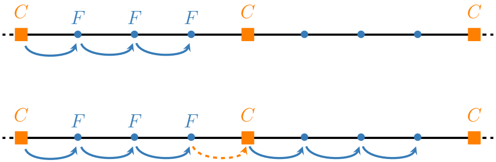
- The solution is iteratively improved using the Full Approximation Scheme (FAS):



Multigrid Reduction in Time (MGRIT)

- Several parameters to be set, *e.g.*, :
  - Number of levels;
  - Time-stepping scheme in each level: $\phi_0, \phi_1, \ldots, \phi_{N_{\text{levels}}-1}$;
  - Coarsening factor between levels: $\Delta t_{l+1} = m_c \Delta t_l$;
  - Relaxation strategy (cycle type);
  - Amount of relaxation: F, FCF, FCFCF, ..., F(CF)$^{N_{\text{relax}}}$



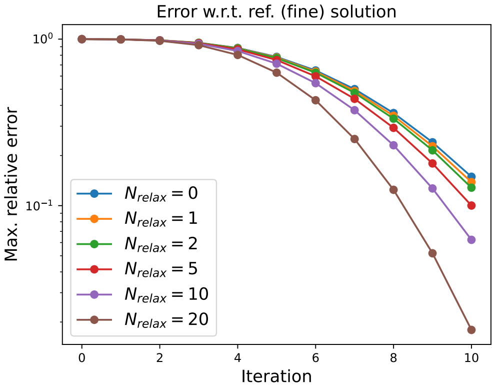


- **MGRIT with 2 levels and $N_{\text{relax}} = 0 \implies$ Parareal**

Example of application of MGRIT[1]: $N_{\textbf{relax}}$

$$\begin{cases} u'(t) = -iu(t), & 0 < t \le T = 100 \\ u(0) = 1 \end{cases}$$

- $\mathcal{F}_{\delta t}, \mathcal{G}_{\Delta t}$: backward Euler
- $N_{\text{levels}} = 2$, $\Delta t_0 = \frac{T}{40960}$, $\Delta t_0 = 64\Delta t_1$

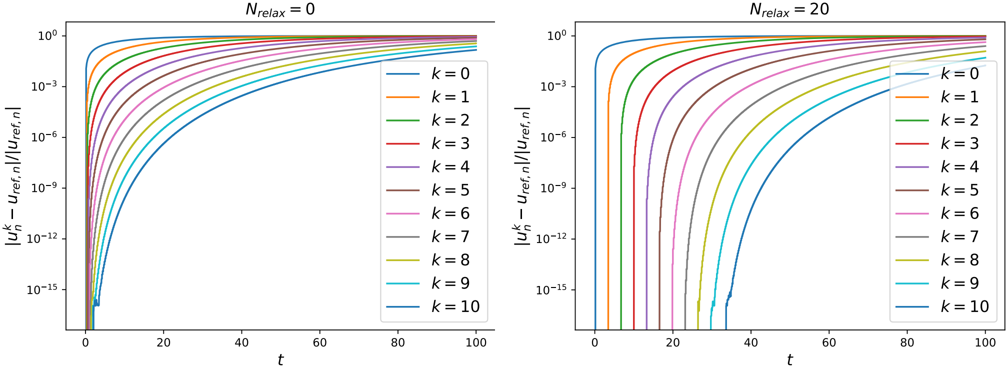


1 Simulations produced using the Python library pyMGRIT: https://pymgrit.github.io/pymgrit/

Example of application of MGRIT[1]: N_{relax}

$$\begin{cases} u'(t) = -iu(t), \quad 0 < t \leq T = 100 \\ u(0) = 1 \end{cases}$$

- $\mathcal{F}_{\delta t}, \mathcal{G}_{\Delta t}$: backward Euler
- $N_{\text{levels}} = 2$, $\Delta t_0 = \frac{T}{40960}$, $\Delta t_0 = 64\Delta t_1$

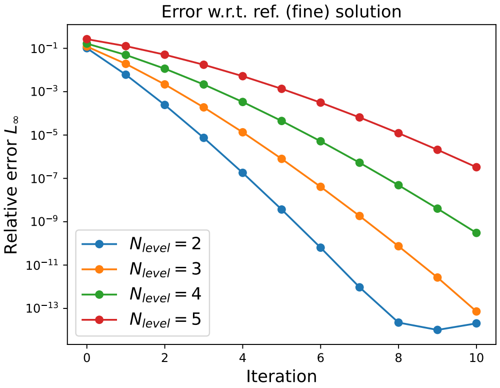


1 Simulations produced using the Python library pyMGRIT: https://pymgrit.github.io/pymgrit/

Example of application of MGRIT[1]: $N_{\textbf{levels}}$

$$\begin{cases} u'(t) = -iu(t), & 0 < t \leq T = 100 \\ u(0) = 1 \end{cases}$$

- ϕ_l, $l = 0, \ldots, N_{\text{levels}} - 1$: backward Euler
- $\Delta t_0 = \frac{T}{40960}$, $\Delta t_l = 4^l \delta t_0$



1 Simulations produced using the Python library pyMGRIT: https://pymgrit.github.io/pymgrit/

Parallel Full Approximation Scheme in Space and Time (PFASST)[1]

Three main ingredients:

- Time-stepping scheme on each level: Spectral Deferred Correction (SDC)
  - Iterative method (on each time step) based on Gaussian quadrature
  - Increasing order across iterations
- Multilevel structure based on the Full Approximation Scheme (FAS)
  - Solve the problem on coarser, cheaper levels
- Predictor-corrector parallel-in-time algorithm:
  - Apply the SDC iterative procedure in several time steps simultaneously

1 Emmett and M. Minion, “Toward an efficient parallel in time method for partial differential equations” (Mar. 2012).

Spectral Deferred Corrections (SDC)[1]

- Solve $u_t = f(u)$ in $[t_n, t_{n+1}]$ using Gauss-Lobato quadrature:
  - Define $t_n = t_{n,0} < \cdots < t_{n,M} = t_{n+1}$
  - Approximate

$$u(t_{n,m+1}) = u(t_{n,m}) + \int_{t_{n,m}}^{t_{n,m+1}} f(u(s))ds \implies U_{n,m+1} = U_{n,m} + \sum_{j=1}^{M} q_{m,j} f(U_{n,j})$$

  - u is unknown at the quadrature nodes $\implies$ start from an initial guess and apply corrections iteratively:

$$U_{n,m+1}^{k+1} = U_{n,m}^{k+1} + \sum_{j=1}^{M} q_{m,j} f(U_{n,j}^{k}) + \Delta t_m [f(U_{n,m+1}^{k+1}) - f(U_{n,m+1}^{k})]$$

  - After k iterations, the solution has accuracy $\mathcal{O}(\Delta t^k)$

1 Dutt, Greengard, and Rokhlin, (2000).

Temporal parallelization of SDC

- SDC is serial across the time steps: the iterations are performed in each time step $[t_n, t_{n+1}]$ before moving to the next one.
- PFASST computes the SDCs simultaneously in several time steps: a few SDC iterations are performed and the (not yet converged) solution at t_n is used as initial solution for $[t_n, t_{n+1}]$.
- PFASST uses multiple levels as in MGRIT, with a correction procedure based on FAS.
- Good parallel efficiency results (*e.g.*, for the heat equation):

2nd-/2nd-order			2nd-/4th-order			4th-/4th-order		
N_p	Speedup	Efficiency	N_p	Speedup	Efficiency	N_p	Speedup	Efficiency
1	0.8	–	1	0.9	–	1	0.7	–
2	1.2	60.0 %	2	1.4	70.0 %	2	1.0	–
4	2.1	53.5 %	4	2.6	65.0 %	4	1.9	47.5 %
8	3.5	43.5 %	8	4.6	57.5 %	8	3.2	40.0 %
16	5.6	35.0 %	12	5.9	49.2 %	12	4.6	38.3 %
32	8.6	26.9 %	24	7.2	30.0 %	24	6.6	27.5 %

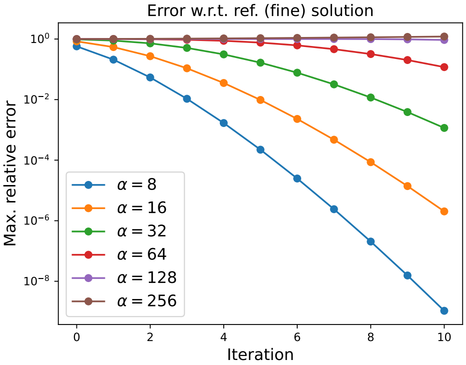
1 Emmett and M. Minion, "Toward an efficient parallel in time method for partial differential equations" (Mar. 2012).
Table: M. L. Minion et al. (2015)

Warmup: Parareal for a real / complex ODE[1]

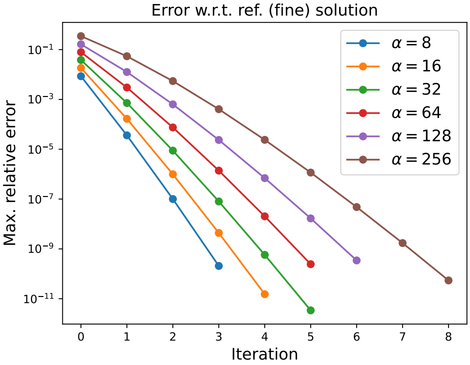
$$\begin{cases} u'(t) = au(t), \quad 0 < t \leq T = 100 \\ u(0) = 1 \end{cases}$$

- $\mathcal{F}_{\delta t}, \mathcal{G}_{\Delta t}$: backward Euler
- $\Delta T = \frac{T}{160}, \delta t = \frac{\Delta T}{2560}, \Delta t = \alpha \delta t, \alpha \geq 2$

$a = -i$

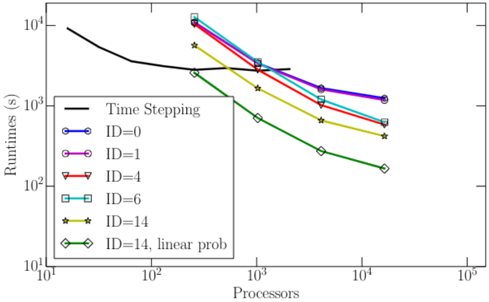


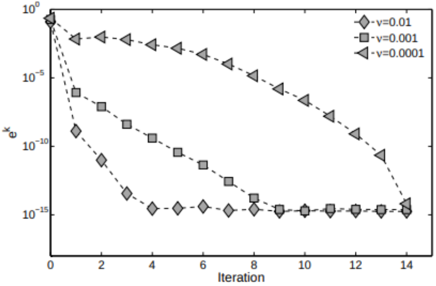
$a = -0.1$



1 Simulations produced using the Python library pyMGRIT: https://pymgrit.github.io/pymgrit/

- Predictor-corrector PinT methods are successfully applied to parabolic, diffusive problems
  - Convergence + stability + speedup
- Examples: thermodynamics, chemical diffusion, Navier-Stokes with small Re, finance,...

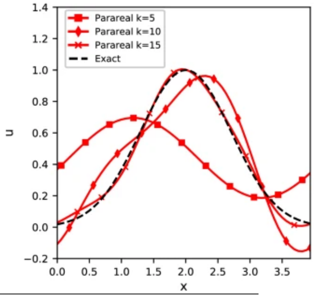


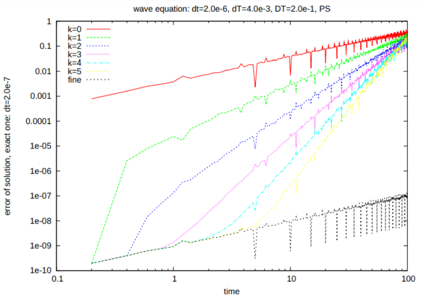


Figures: Falgout et al. (2017); Steiner et al. (2014)

PinT for hyperbolic problems

- Stability and convergence issues when applied to simple hyperbolic problems
- Discouraged application to more complex advection-dominated problems
- Why?[1]
  - Mismatch of discrete phase speeds
  - Mainly on high wavenumbers

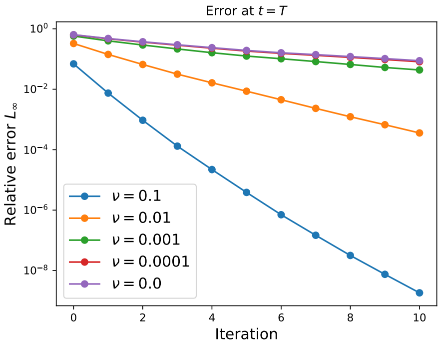




1 Ruprecht, "Wave propagation characteristics of Parareal" (June 2018).
Figures: Ruprecht (2018); Dai and Maday (2013)

Example: convergence of Parareal for hyperbolic vs parabolic[1]

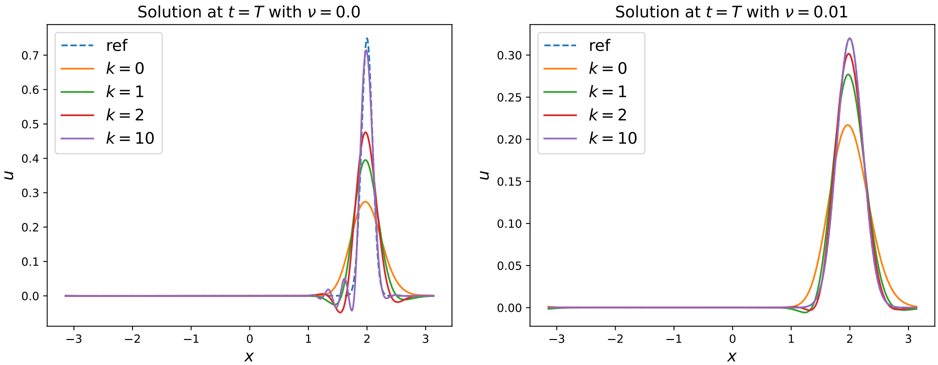
- 1D advection-diffusion equation: $u_t + u_x = \nu u_{xx}$
- $\mathcal{F}_{\delta t}, \mathcal{G}_{\Delta t}$: backward Euler + spectral method; $T = 2$; $\delta t = T/1024$; $\Delta t = \Delta T = T/64$



1 Simulations produced using the Python library pyMGRIT: https://pymgrit.github.io/pymgrit/

Example: convergence of Parareal for hyperbolic vs parabolic[1]

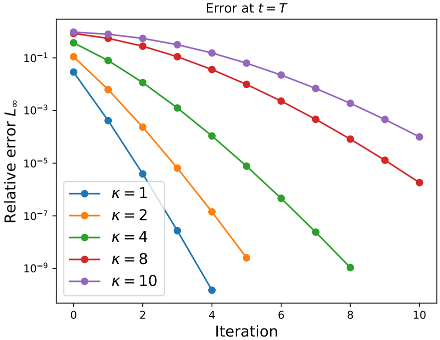
- 1D advection-diffusion equation: $u_t + u_x = \nu u_{xx}$
- $\mathcal{F}_{\delta t}, \mathcal{G}_{\Delta t}$: backward Euler + spectral method; $T = 2$; $\delta t = T/1024$; $\Delta t = \Delta T = T/64$



1 Simulations produced using the Python library pyMGRIT: https://pymgrit.github.io/pymgrit/

Example: convergence of Parareal depending on the wavenumber[1]

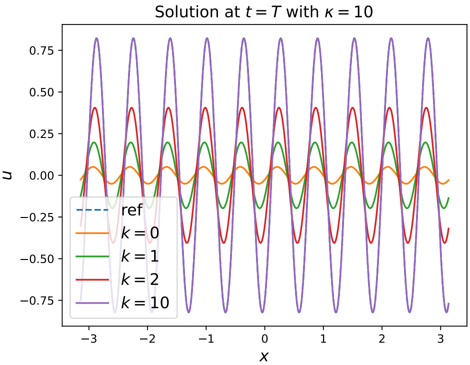
- 1D Advection equation: $u_t + u_x = 0$, $u(0,x) = \sin(\kappa x)$
- $\mathcal{F}_{\delta t}, \mathcal{G}_{\Delta t}$: backward Euler + spectral method; $T = 2$; $\delta t = T/1024$; $\Delta t = \Delta T = T/64$

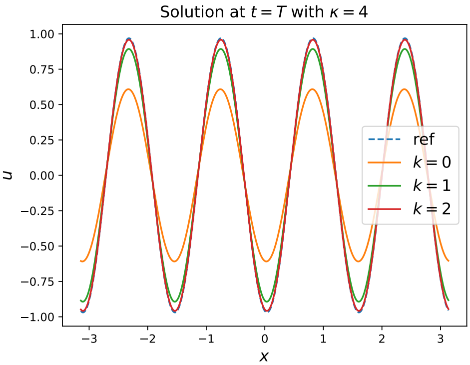


1 Simulations produced using the Python library pyMGRIT: https://pymgrit.github.io/pymgrit/

Example: convergence of Parareal depending on the wavenumber[1]

- 1D Advection equation: $u_t + u_x = 0$, $u(0, x) = \sin(\kappa x)$
- $\mathcal{F}_{\delta t}, \mathcal{G}_{\Delta t}$: backward Euler + spectral method; $T = 2$; $\delta t = T/1024$; $\Delta t = \Delta T = T/64$





1 Simulations produced using the Python library pyMGRIT: https://pymgrit.github.io/pymgrit/

Table of Contents

- Consider the Dahlquist's test equation:

$$u'(t) = au(t), \qquad u(0) = u_0, \qquad a \in \mathbb{C} \tag{2}$$

- The stability function $A = A(a\Delta t)$ of a given time-stepping scheme defines the stability region $\{a\Delta t \text{ s.t. } |A(a\Delta t)| \leq 1\}$
- We want to express u_n^k as

$$u_n^k = A_{\text{parareal}} u_0$$

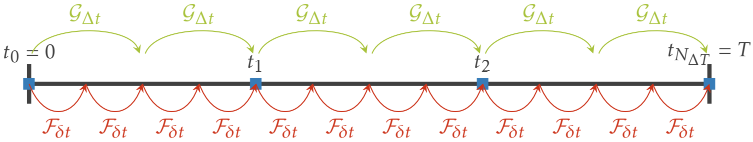
where $A_{\text{parareal}} = A_{\text{parareal}}(a, n, k, \mathcal{F}_{\delta t}, \mathcal{G}_{\Delta t}, \delta t, \Delta t, \Delta T)$ is the **stability function of Parareal**

Linear stability analysis of Parareal[1]

- Assume $\Delta T/\delta t, \Delta T/\Delta t \in \mathbb{N}$
- Define the stability functions:
  - $A_f = A_f(a\delta t)$: stability function of $\mathcal{F}_{\delta t}$ (for advancing one fine time step);
  - $A_c = A_c(a\Delta t)$: stability function of $\mathcal{G}_{\Delta t}$ (for advancing one coarse time step);

  e.g., (forward Euler, backward Euler): $A_f = (1 + a\delta t)$, $A_f = (1 - a\delta t)^{-1}$
- Define $\tilde{A}_f = A_f^{\Delta T/\delta t}$ and $\tilde{A}_c = A_c^{\Delta T/\delta t}$

  such that $u_{n+1} = \tilde{A}_f u_n$ and $u_{n+1} = \tilde{A}_c u_n$



1 Staff and Rønquist, “Stability of the Parareal Algorithm” (2005).

Linear stability analysis of Parareal[1]

- Replacing in the Parareal iteration:

$$u_n^k = \tilde{A}_c u_{n-1}^k + (\tilde{A}_f - \tilde{A}_c) u_{n-1}^{k-1}$$

- The recurrence relation leads to

$$A_{\text{parareal}} = \sum_{i=0}^{k} \begin{pmatrix} n \\ i \end{pmatrix} (\tilde{A}_f - \tilde{A}_c)^i \tilde{A}_c^{n-i}$$

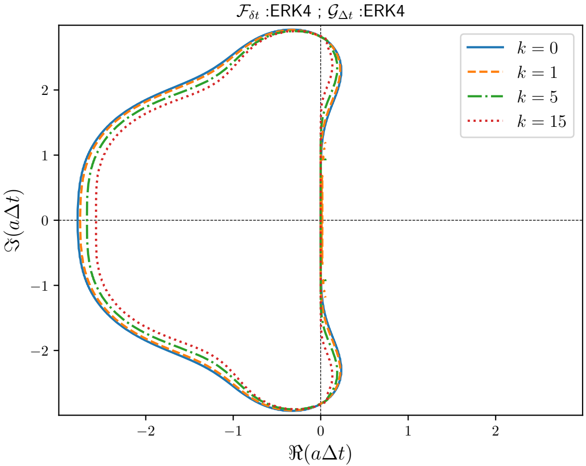
- This stability analysis can be extended to (linearized) nonlinear problems:

$$u'(t) = Lu + N(u) \implies u'(t) = Lu + \tilde{N}u, \qquad L, \tilde{N} \in \mathbb{C}$$

1 Staff and Rønquist, “Stability of the Parareal Algorithm” (2005).

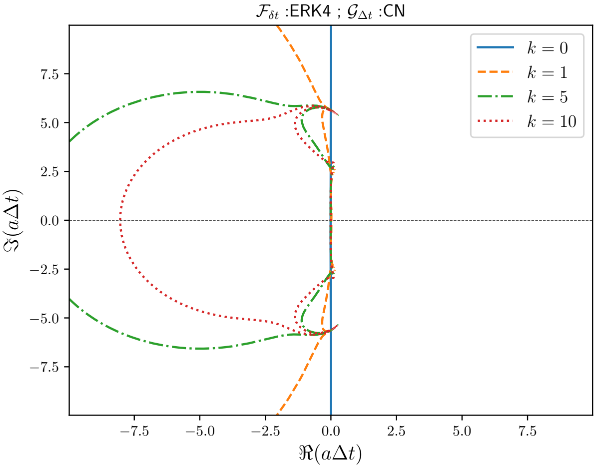
Linear stability analysis of Parareal for $u'(t) = au$

- Example: $n = 100$; $\Delta t = 2\delta t$; $\mathcal{F}_{\delta t} = \mathcal{G}_{\Delta t} =$ ERK4



Linear stability analysis of Parareal for $u'(t) = au$

- Example: $n = 100$; $\Delta t = 2\delta t$; $\mathcal{F}_{\delta t}$ = ERK4; $\mathcal{G}_{\Delta t}$ = Crank-Nicolson



- This stability study can be extended to the two-level MGRIT with arbitrary relaxation N_{relax}[1]:
- Two-level MGRIT with $N_{\text{relax}} \equiv$ Parareal with $N_{\text{relax}} + 1$ time slices computed by $\mathcal{F}_{\delta t}$ per iteration[2]:

$$u_{n+1}^{k+1} = \mathcal{G}_{\Delta t}(U_n^{k+1}) + \mathcal{F}_{\delta t}^{N_{\text{relax}}+1}(U_{n-N_{\text{relax}}}^k) - \mathcal{G}_{\Delta t}\mathcal{F}_{\delta t}^{N_{\text{relax}}}(U_{n-N_{\text{relax}}}^k)$$

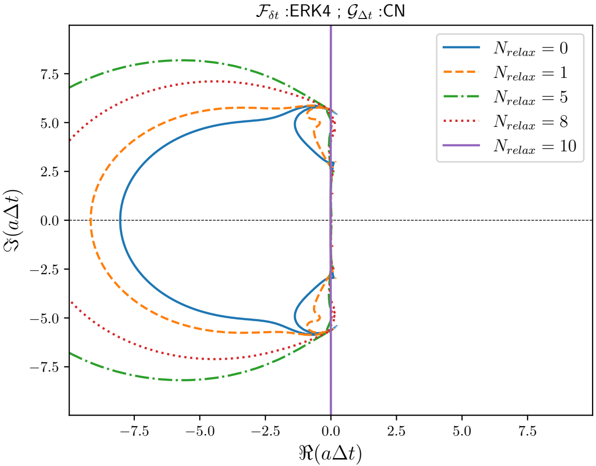
- Recurrence relation:

$$A_{\text{MGRIT}} = \sum_{i=0}^{\lfloor k/(N_{\text{relax}}+1)\rfloor} \binom{n - iN_{\text{relax}}}{i} (\tilde{A}_f^{N_{\text{relax}}}(\tilde{A}_f - \tilde{A}_c))^i \tilde{A}_c^{n-i(N_{\text{relax}}+1)}$$

1 Caldas Steinstraesser, Peixoto, and Schreiber, “Parallel-in-time integration of the shallow water equations on the rotating sphere using Parareal and MGRIT” (Jan. 2024).
2 M. J. Gander, Kwok, and Zhang, “Multigrid interpretations of the parareal algorithm leading to an overlapping variant and MGRIT” (June 2018).

Linear stability analysis of the two-level MGRIT for $u'(t) = au$

- Example: $n = 100$; $\Delta t = 2\delta t$; $\mathcal{F}_{\delta t}$ = ERK4; $\mathcal{G}_{\Delta t}$ = CN; $k = 10$



Convergence of the Parareal method

- Consider the Dahlquist test equation: $u'(t) = au(t), u(0) = u_0$
- $\mathcal{F}_{\delta t}$ is the exact propagator: $\mathcal{F}_{\delta t}(u, 0, t) = u_0 e^{at}$

- **Theorem**[1]**:** if $\mathcal{G}_{\Delta t}$ is the explicit or the implicit Euler method, then

$$|U_n^k - u(t_n)| \leq c_k \Delta t^k$$

  i.e., **at iteration k, the Parareal method has order k.**

- **Theorem**[2]**:** Let $\mathcal{G}_{\Delta t}$ be a $p-$th order method in its region of absolute stability and $T < \infty$. Then

$$\max_{1 \leq n \leq N_{\Delta T}} |u_n^k - u(t_n)| \leq \frac{(CT)^k}{k!} \Delta t^{pk} \max_{1 \leq n \leq N_{\Delta T}} |u_n^0 - u(t_n)|$$

  i.e., **at iteration k, Parareal has order pk.**

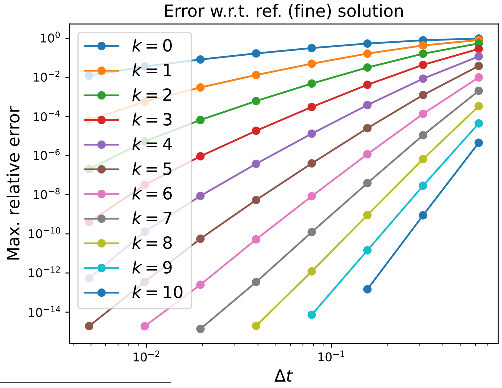
1 J.-L. Lions, Maday, and Turinici, “Résolution d’EDP par un schéma en temps “pararéel”” (Apr. 2001).
2 M. J. Gander and Vandewalle, “Analysis of the Parareal Time-Parallel Time-Integration Method” (Jan. 2007).

Convergence of the Parareal method[1]

$$\begin{cases} u'(t) = -iu(t), & 0 < t \leq T = 10 \\ u(0) = 1 \end{cases}$$

- $\mathcal{F}_{\delta t}, \mathcal{G}_{\Delta t}$: backward Euler
- $\Delta T = \frac{T}{16}$, $\delta t = \frac{\Delta T}{256}$, $\Delta t = \alpha \delta t$, $\alpha \geq 2$

Error at $t = T$:



1 Simulations produced using the Python library pyMGRIT: https://pymgrit.github.io/pymgrit/

- **Theorem**[1]**:** Consider the heat or the advection equation in one spatial dimension:

$$u_t = u_{xx}, \qquad u_t = u_x, \qquad x \in \Omega = \mathbb{R}, T < \infty, \qquad u(0,x) \in L^2(\Omega)$$

$\mathcal{F}_{\delta t}$ is the exact propagator and $\mathcal{G}_{\Delta t}$ is a stable one-step method. Then:

$$\max_{1 \le n \le N_{\Delta T}} \left\| u(t_n) - U_n^k \right\|_2 \le \frac{\beta_s^k}{k!} \prod_{j=1}^{k} (N_{\Delta T} - j) \max_{1 \le n \le N_{\Delta T}} \left\| u(t_n) - U_n^0 \right\|_2$$

where β_s depends only on $\mathcal{G}_{\Delta t}$.

Heat equation ($\beta_s = \gamma_s$)

	Backward Euler	Trapezoidal rule	SDIRK	Radau IIA
γ_s	0.2036321888	1	0.1717941220	0.0634592650
γ_l	0.2984256075	∞	0.2338191487	0.0677592165

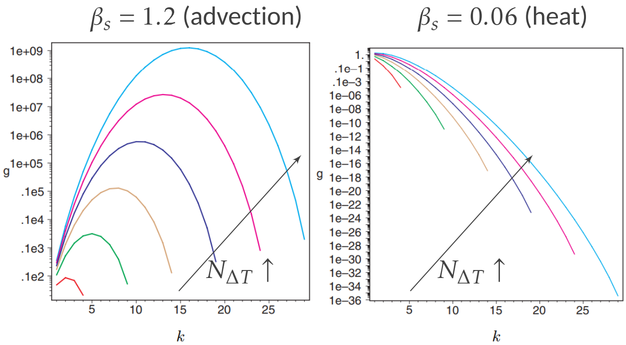
Advection equation ($\beta_s = \alpha_s$)

	Backward Euler	Trapezoidal rule	SDIRK	Radau IIA
α_s	1.224353426	2	1.185652097	1.362526017
α_l	1.632645559	∞	∞	2.231320732

1 M. J. Gander and Vandewalle, "Analysis of the Parareal Time-Parallel Time-Integration Method" (Jan. 2007).
Tables: M. J. Gander and Vandewalle (2007)

Order of convergence of Parareal

- The convergence factor is bounded by the function $g(\beta, k, N) := \frac{\beta^k}{k!} \prod_{j=1}^{k} (N_{\Delta T} - j)$



- Similar conclusions in the MGRIT framework with arbitrary N_{levels}, N_{relax}, cycling strategy[1].

1 Hessenthaler et al., "Multilevel Convergence Analysis of Multigrid-Reduction-in-Time" (Jan. 2020).
Figure: M. J. Gander and Vandewalle (2007)

Main conclusions

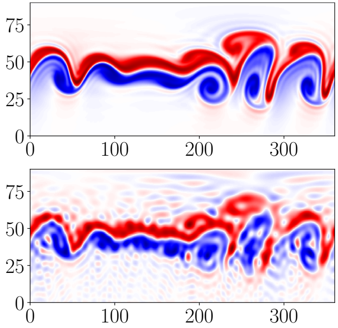
- Through iterations:
  - The convergence order of Parareal increases
  - but its stability degrades!
- Stability and convergence are more critical for purely hyperbolic problems:
  - Monotone convergence cannot be ensured in general
  - The stability regions move away from the imaginary axis
- Stability and convergence can be improved by:
  - Adding diffusion $\implies$ affects accuracy
  - Increasing N_{relax}, reducing Δt, ... $\implies$ affects speedup

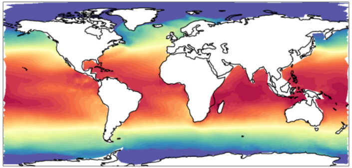
PinT methods for hyperbolic / advection-dominated problems

- How to overcome stability and convergence issues?
  - Adding diffusion?

> "Twelve Ways To Fool The Masses When Giving Parallel-In-Time Results"[1]
>
> **Way 4.** *If you are not completely thrilled about the speedup because the number of iterations K is too high, try adding more diffusion. You might have to experiment a little to find just the right amount.*

  - Parametrization
  - Choice of time-stepping schemes
- Towards the temporal parallelization of operational models?

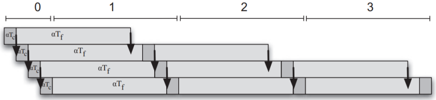




1 Götschel et al., "Twelve Ways to Fool the Masses When Giving Parallel-in-Time Results" (2021).
Figures:Caldas Steinstraesser, Peixoto, and Schreiber (2024), Philippi and Slawig (2024)

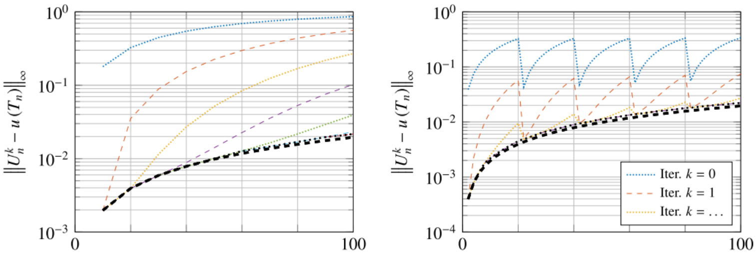
Towards better parallel performance

- Improved parallel task scheduling, reducing idleness [1]



$$S = \frac{1}{(k+1)\frac{N_{\Delta t}}{N_{\delta t}}\frac{\tau_c}{\tau_f} + \frac{k}{N_{\Delta T}}} \Longrightarrow S = \frac{1}{\left(\frac{k}{N_{\Delta t}}+1\right)\frac{N_{\Delta t}}{N_{\delta t}}\frac{\tau_c}{\tau_f} + \frac{k}{N_{\Delta T}}}$$

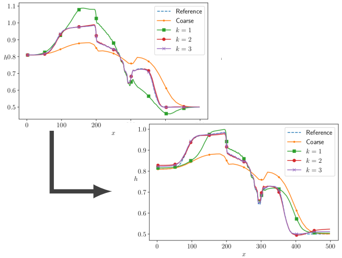
- Windowing: improve convergence speed vs achievable speedup[2]

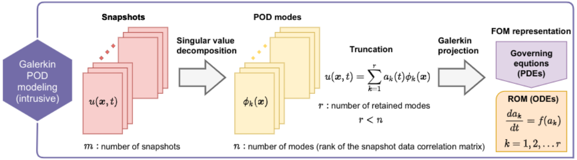


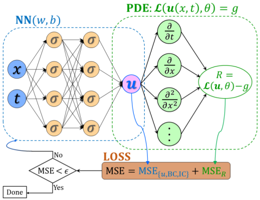
1 Aubanel, "Scheduling of tasks in the parareal algorithm" (Mar. 2011).
2 Nielsen, Brunner, and Hesthaven, "Communication-aware adaptive Parareal with application to a nonlinear hyperbolic system of partial differential equations" (Oct. 2018).
Figures: Aubanel (2011); Nielsen, Brunner, and Hesthaven (2018)

Model reduction / Machine Learning in PinT

- The fine dynamics cannot be represented by coarsening traditional schemes
- Introduce the fine dynamics in $\mathcal{G}_{\Delta t}$ using:
  - Reduced-order models (POD, EIM, RB, ...)[1]
  - Machine Learning techniques (PINN)[2]
- Challenges:
  - Construction of efficient models
  - Efficient computation



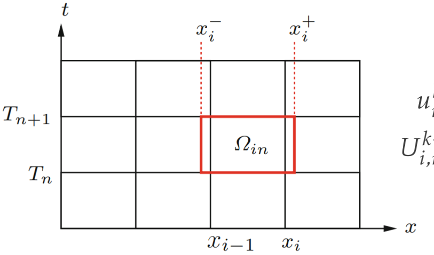




1 Chen, Hesthaven, and Zhu, “On the Use of Reduced Basis Methods to Accelerate and Stabilize the Parareal Method” (2014).
2 Ibrahim, Götschel, and Ruprecht, “Parareal with a Physics-Informed Neural Network as Coarse Propagator” (2023).
Figures: Caldas Steinstraesser (2021); Ahmed et al. (2021); Meng et al. (2020)

Space-time parallelization

- The temporal parallelization is seen as a complement to the spatial one
- PinT + DDM
  - Parareal with $\mathcal{F}_{\delta t}, \mathcal{G}_{\Delta t}$ = Schwarz WR[1]
  - Schwarz embedded in the Parareal iteration: Parareal Schwarz WR[2]
  - Convergence depends on T, interface conditions, overlapping size.



$$
\begin{aligned}
u_{i,n}^{k+1} &= \mathcal{F}_{\delta t}(U_{i,n}^k, t_n, t, \mathcal{B}_i^{\pm} u_{i\pm1,n}^k) \\
U_{i,n+1}^{k+1} &= u_{i,n}^{k+1}(\cdot, t_{n+1}) + \mathcal{G}_{\Delta t}(U_{i,n}^{k+1}, t_n, t_{n+1}, \mathcal{B}_i^{\pm} u_{i\pm1,n}^{k+1}) \\
&\quad - \mathcal{G}_{\Delta t}(U_{i,n}^k, t_n, t_{n+1}, \mathcal{B}_i^{\pm} u_{i\pm1,n}^k)
\end{aligned}
$$

1 Maday and Turinici, "The Parareal in Time Iterative Solver: a Further Direction to Parallel Implementation" (2005).
2 M. J. Gander, Jiang, and Li, "Parareal Schwarz Waveform Relaxation Methods" (2013).
Figure: M. J. Gander, Jiang, and Li (2013)

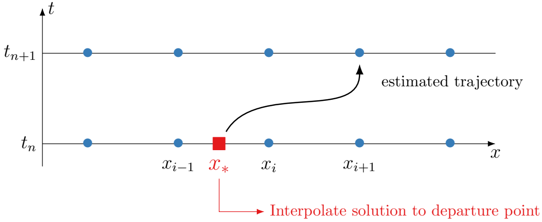
PinT for atmospheric circulation models[1]

- Parallelization of weather/climate models?
- Are **well-established time stepping schemes** in atmospheric modeling suitable as coarse schemes in PinT (Parareal and MGRIT)?
  - Eulerian implicit-explicit **(IMEX)** (MPAS/NCAR, NUMA/NPS/NRL)
  - Semi-Lagrangian semi-implicit **(SL-SI-SETTLS)** (IFS/ECMWF, GFS/NCEP/NOAA)
- How much artificial (hyper-)viscosity ∇^q?
- Shallow water equations on the rotating sphere

$$\boldsymbol{U}^{n+1} = \boldsymbol{F}_I^{\Delta t/2}\left(\boldsymbol{F}_E^{\Delta t}\left(\boldsymbol{F}_I^{\Delta t/2}\left(\boldsymbol{U}^n\right)\right)\right)$$

Linear terms: implicit

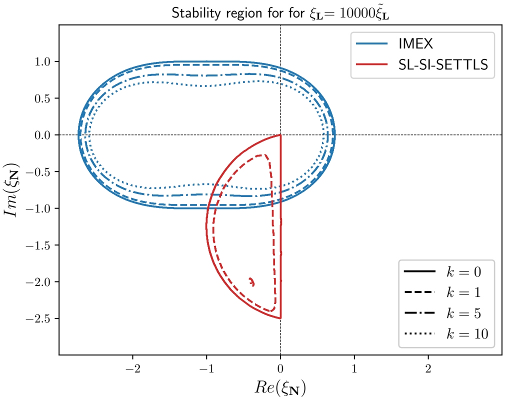
Nonlinear terms: explicit



2 Caldas Steinstraesser, Peixoto, and Schreiber, “Parallel-in-time integration of the shallow water equations on the rotating sphere using Parareal and MGRIT” (Jan. 2024).

PinT for atmospheric circulation models[1]

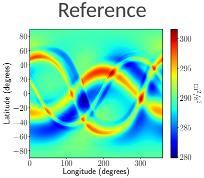
- **Stability of Parareal along iterations**
  Fine scheme: IMEX; Coarse scheme: IMEX , SL-SI-SETTLS ; $n = 100, m_c = 2$

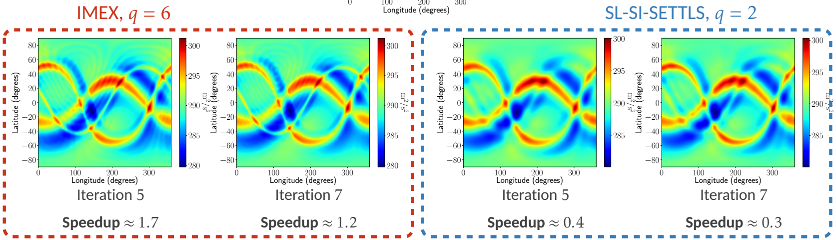


1 Caldas Steinstraesser, Peixoto, and Schreiber, "Parallel-in-time integration of the shallow water equations on the rotating sphere using Parareal and MGRIT" (Jan. 2024).

- **Numerical simulations: Parareal/MGRIT using as corse scheme:**
  - **SL-SI-SETTLS**: strong instabilities; requires large, low-order viscosity
  - **IMEX**: high-order viscosity only on the coarsest levels is sufficient; speedup > 1

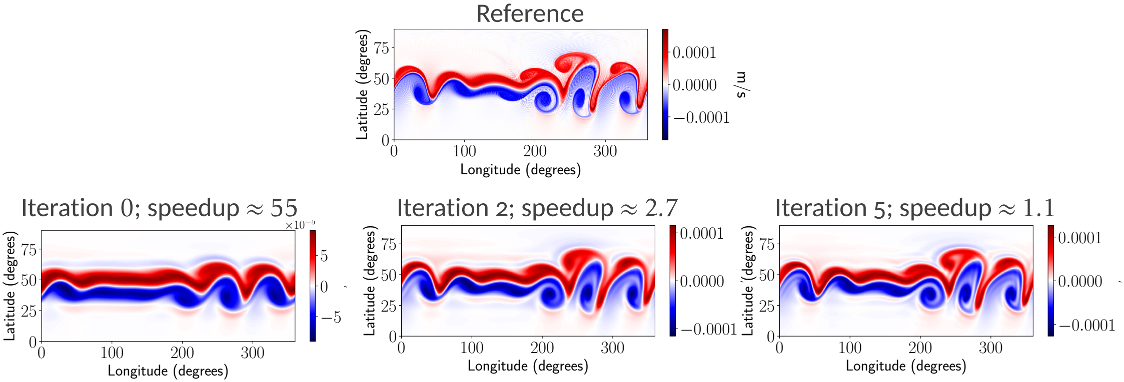
Using $N_{\text{proc}} = 64$:





1 Caldas Steinstraesser, Peixoto, and Schreiber, "Parallel-in-time integration of the shallow water equations on the rotating sphere using Parareal and MGRIT" (Jan. 2024).

The unstable jet test case[1] using Parareal, $\mathcal{F}_{\delta t} = \mathcal{G}_{\Delta t} =$IMEX, ∇^4 only in $\mathcal{G}_{\Delta t}$, $N_{\textbf{proc}} = 64$



1 Galewsky, Scott, and Polvani, "An initial-value problem for testing numerical models of the global shallow-water equations" (Jan. 2004).
1 Caldas Steinstraesser, Peixoto, and Schreiber, "Parallel-in-time integration of the shallow water equations on the rotating sphere using Parareal and MGRIT" (Jan. 2024).

Thank you!

joao.steinstraesser@usp.br

www.ime.usp.br/∼joao.steinstraesser

The scripts for the simulations using pyMGRIT are available in www.ime.usp.br/∼joao.steinstraesser/files/tests_pymgrit.zip

Abaqus. *Abaqus Analysis User's Guide*. English. United States: Dassault Systèmes Simulia Corp, 2016.

Ahmed, Shady E. et al. "Nonlinear proper orthogonal decomposition for convection-dominated flows". In: *Physics of Fluids* 33.12 (Dec. 2021). ISSN: 1089-7666. DOI: 10.1063/5.0074310. URL: http://dx.doi.org/10.1063/5.0074310.

Aubanel, Eric. "Scheduling of tasks in the parareal algorithm". In: *Parallel Computing* 37.3 (Mar. 2011), pp. 172–182. ISSN: 0167-8191. DOI: 10.1016/j.parco.2010.10.004. URL: http://dx.doi.org/10.1016/j.parco.2010.10.004.

Baffico, L. et al. "Parallel-in-time molecular-dynamics simulations". In: *Physical Review E* 66.5 (Nov. 2002). ISSN: 1095-3787. DOI: 10.1103/physreve.66.057701. URL: http://dx.doi.org/10.1103/PhysRevE.66.057701.

Bauer, Peter, Alan Thorpe, and Gilbert Brunet. "The quiet revolution of numerical weather prediction". In: *Nature* 525.7567 (Sept. 2015), pp. 47–55. ISSN: 1476-4687. DOI: 10.1038/nature14956. URL: http://dx.doi.org/10.1038/nature14956.

Bellen, Alfredo and Marino Zennaro. "Parallel algorithms for initial-value problems for difference and differential equations". In: *Journal of Computational and Applied Mathematics* 25.3 (May 1989), pp. 341–350. ISSN: 0377-0427. DOI: 10.1016/0377-0427(89)90037-x. URL: http://dx.doi.org/10.1016/0377-0427(89)90037-X.

Caldas Steinstraesser, J. G. "Coupling large and small scale shallow water models with porosity in the presence of anisotropy". Theses. Université Montpellier, Oct. 2021. URL: https://tel.archives-ouvertes.fr/tel-03435394.

Caldas Steinstraesser, J. G., Pedro da Silva Peixoto, and Martin Schreiber. "Parallel-in-time integration of the shallow water equations on the rotating sphere using Parareal and MGRIT". In: *Journal of Computational Physics* 496 (Jan. 2024), p. 112591. ISSN: 0021-9991. DOI: 10.1016/j.jcp.2023.112591. URL: http://dx.doi.org/10.1016/j.jcp.2023.112591.

Chen, Feng, Jan S. Hesthaven, and Xueyu Zhu. “On the Use of Reduced Basis Methods to Accelerate and Stabilize the Parareal Method”. In: *Reduced Order Methods for Modeling and Computational Reduction*. Ed. by Alfio Quarteroni and Gianluigi Rozza. Cham: Springer International Publishing, 2014, pp. 187–214. ISBN: 978-3-319-02090-7. DOI: 10.1007/978-3-319-02090-7_7. URL: https://doi.org/10.1007/978-3-319-02090-7_7.

Dai, Xiaoying and Yvon Maday. “Stable Parareal in Time Method for First- and Second-Order Hyperbolic Systems”. In: *SIAM Journal on Scientific Computing* 35.1 (Jan. 2013), A52–A78. ISSN: 1095-7197. DOI: 10.1137/110861002. URL: http://dx.doi.org/10.1137/110861002.

Dutt, Alok, Leslie Greengard, and Vladimir Rokhlin. In: *Bit Numerical Mathematics* 40.2 (2000), pp. 241–266. ISSN: 0006-3835. DOI: 10.1023/a:1022338906936. URL: http://dx.doi.org/10.1023/A:1022338906936.

Emmett, Matthew and Michael Minion. “Toward an efficient parallel in time method for partial differential equations”. In: *Communications in Applied Mathematics and Computational Science* 7.1 (Mar. 2012), pp. 105–132. DOI: 10.2140/camcos.2012.7.105. URL: https://doi.org/10.2140/camcos.2012.7.105.

Falgout, R. D. et al. “Multigrid Reduction in Time for Nonlinear Parabolic Problems: A Case Study”. In: *SIAM Journal on Scientific Computing* 39.5 (Jan. 2017), S298–S322. ISSN: 1095-7197. DOI: 10.1137/16m1082330. URL: http://dx.doi.org/10.1137/16M1082330.

Friedhoff, S. et al. “A Multigrid-in-Time Algorithm for Solving Evolution Equations in Parallel”. In: *Presented at: Sixteenth Copper Mountain Conference on Multigrid Methods*. Copper Mountain, CO, United States, Mar 17 - Mar 22, 2013, 2013. URL: http://www.osti.gov/scitech/servlets/purl/1073108.

Galewsky, Joseph, Richard K. Scott, and Lorenzo M. Polvani. “An initial-value problem for testing numerical models of the global shallow-water equations”. In: *Tellus A: Dynamic Meteorology and Oceanography* 56.5 (Jan. 2004), pp. 429–440. DOI: 10.3402/tellusa.v56i5.14436. URL: https://doi.org/10.3402/tellusa.v56i5.14436.

Gander, M., J. Halpern, and Nataf F. “Optimal convergence for overlapping and non-overlapping Schwarz waveform relaxation”. In: *Eleventh International Conference on Domain Decomposition Methods*. 1999, pp. 27–36.

Gander, Martin J. “50 Years of Time Parallel Time Integration”. In: *Contributions in Mathematical and Computational Sciences*. Springer International Publishing, 2015, pp. 69–113. DOI: 10.1007/978-3-319-23321-5_3. URL: https://doi.org/10.1007/978-3-319-23321-5_3.

— .“Analysis of the Parareal Algorithm Applied to Hyperbolic Problems Using Characteristics”. eng. In: *Boletín de la Sociedad Española de Matemática Aplicada* 42 (2008). ID: unige:6268, pp. 21–35. URL: https://archive-ouverte.unige.ch/unige:6268.

Gander, Martin J. and Stefan Güttel. “PARAEXP: A Parallel Integrator for Linear Initial-Value Problems”. In: *SIAM Journal on Scientific Computing* 35.2 (Jan. 2013), pp. C123–C142. ISSN: 1095-7197. DOI: 10.1137/110856137. URL: http://dx.doi.org/10.1137/110856137.

Gander, Martin J., Yao-Lin Jiang, and Rong-Jian Li. “Parareal Schwarz Waveform Relaxation Methods”. In: *Domain Decomposition Methods in Science and Engineering XX*. Springer Berlin Heidelberg, 2013, pp. 451–458. ISBN: 9783642352751. DOI: 10.1007/978-3-642-35275-1_53. URL: http://dx.doi.org/10.1007/978-3-642-35275-1_53.

Gander, Martin J., Felix Kwok, and Hui Zhang. “Multigrid interpretations of the parareal algorithm leading to an overlapping variant and MGRIT”. In: *Computing and Visualization in Science* 19.3-4 (June 2018), pp. 59–74. DOI: 10.1007/s00791-018-0297-y. URL: https://doi.org/10.1007/s00791-018-0297-y.

Gander, Martin J. and Stefan Vandewalle. “Analysis of the Parareal Time-Parallel Time-Integration Method”. In: *SIAM Journal on Scientific Computing* 29.2 (Jan. 2007), pp. 556–578. ISSN: 1095-7197. DOI: 10.1137/05064607x. URL: http://dx.doi.org/10.1137/05064607X.

Götschel, Sebastian et al. “Twelve Ways to Fool the Masses When Giving Parallel-in-Time Results”. In: *Parallel-in-Time Integration Methods*. Springer International Publishing, 2021, pp. 81–94. ISBN: 9783030759339. DOI: 10.1007/978-3-030-75933-9_4. URL: http://dx.doi.org/10.1007/978-3-030-75933-9_4.

Hessenthaler, Andreas et al. “Multilevel Convergence Analysis of Multigrid-Reduction-in-Time”. In: *SIAM Journal on Scientific Computing* 42.2 (Jan. 2020), A771–A796. ISSN: 1095-7197. DOI: 10.1137/19m1238812. URL: http://dx.doi.org/10.1137/19M1238812.

Ibrahim, Abdul Qadir, Sebastian Götschel, and Daniel Ruprecht. "Parareal with a Physics-Informed Neural Network as Coarse Propagator". In: *Euro-Par 2023: Parallel Processing*. Springer Nature Switzerland, 2023, pp. 649–663. ISBN: 9783031396984. DOI: 10.1007/978-3-031-39698-4_44. URL: http://dx.doi.org/10.1007/978-3-031-39698-4_44.

Lelarasmee, E., A.E. Ruehli, and A.L. Sangiovanni-Vincentelli. "The Waveform Relaxation Method for Time-Domain Analysis of Large Scale Integrated Circuits". In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 1.3 (July 1982), pp. 131–145. ISSN: 0278-0070. DOI: 10.1109/tcad.1982.1270004. URL: http://dx.doi.org/10.1109/TCAD.1982.1270004.

Lions, Jacques-Louis, Yvon Maday, and Gabriel Turinici. "Résolution d'EDP par un schéma en temps "pararéel"". In: *Comptes Rendus de l'Académie des Sciences - Series I - Mathematics* 332.7 (Apr. 2001), pp. 661–668.

Lions, P. L. "On the Schwarz Alternating Method I". In: *First International Symposium on Domain Decomposition Methods for Partial Differential Equations*. Ed. by R. Glowinski et al. Society for Industrial and Applied Mathematics, 1988, pp. 1–42.

Maday, Yvon and Einar M. Rønquist. "Parallelization in time through tensor-product space–time solvers". In: *Comptes Rendus. Mathématique* 346.1–2 (Dec. 2007), pp. 113–118. ISSN: 1778-3569. DOI: 10.1016/j.crma.2007.09.012. URL: http://dx.doi.org/10.1016/j.crma.2007.09.012.

Maday, Yvon and Gabriel Turinici. "The Parareal in Time Iterative Solver: a Further Direction to Parallel Implementation". In: *Domain Decomposition Methods in Science and Engineering*. Springer-Verlag, 2005, pp. 441–448. ISBN: 3540225234. DOI: 10.1007/3-540-26825-1_45. URL: http://dx.doi.org/10.1007/3-540-26825-1_45.

Meng, Xuhui et al. "PPINN: Parareal physics-informed neural network for time-dependent PDEs". In: *Computer Methods in Applied Mechanics and Engineering* 370 (Oct. 2020), p. 113250. ISSN: 0045-7825. DOI: 10.1016/j.cma.2020.113250. URL: http://dx.doi.org/10.1016/j.cma.2020.113250.

Minion, M. L. et al. “Interweaving PFASST and Parallel Multigrid”. In: *SIAM Journal on Scientific Computing* 37.5 (Jan. 2015), S244–S263. ISSN: 1095-7197. DOI: 10.1137/14097536x. URL: http://dx.doi.org/10.1137/14097536X.

Moore, Gordon E. “Cramming more components onto integrated circuits”. In: *Electronics* 38.8 (1965).

Mozdzynski, G. *A PGAS implementation by co-design of the ECMWF Integrated Forecasting System*. eng. 2012 2012.

Nielsen, Allan S., Gilles Brunner, and Jan S. Hesthaven. “Communication-aware adaptive Parareal with application to a nonlinear hyperbolic system of partial differential equations”. In: *Journal of Computational Physics* 371 (Oct. 2018), pp. 483–505. ISSN: 0021-9991. DOI: 10.1016/j.jcp.2018.04.056. URL: http://dx.doi.org/10.1016/j.jcp.2018.04.056.

Nievergelt, J. “Parallel methods for integrating ordinary differential equations”. In: *Communications of the ACM* 7.12 (Dec. 1964), pp. 731–733. ISSN: 1557-7317. DOI: 10.1145/355588.365137. URL: http://dx.doi.org/10.1145/355588.365137.

Philippi, B. and T. Slawig. *A Micro-Macro Parareal Implementation for the Ocean-Circulation Model FESOM2*. 2024. arXiv: 2306.17269 [math.NA]. URL: https://arxiv.org/abs/2306.17269.

Qin, Shanlin et al. “A highly parallel fully implicit domain decomposition method for the simulation of the hemodynamics of a patient-specific artery at the full-body scale”. In: *Journal of Computational Physics* 472 (Jan. 2023), p. 111730. ISSN: 0021-9991. DOI: 10.1016/j.jcp.2022.111730. URL: http://dx.doi.org/10.1016/j.jcp.2022.111730.

Ries, M. and U. Trottenberg. “MGR-ein blitzschneller elliptischer löser”. In: *Tech. Rep. Preprint* 277.SFB 72 (1979).

Ries, Manfred, Ulrich Trottenberg, and Gerd Winter. “A note on MGR methods”. In: *Linear Algebra and its Applications* 49 (1983), pp. 1–26. ISSN: 0024-3795. DOI: https://doi.org/10.1016/0024-3795(83)90091-5. URL: https://www.sciencedirect.com/science/article/pii/0024379583900915.

Ruprecht, Daniel. "Wave propagation characteristics of Parareal". In: *Computing and Visualization in Science* 19 (1 June 2018), pp. 1–17. DOI: 10.1007/s00791-018-0296-z.

Schlick, Tamar. "Biomolecular Structure and Modeling: Historical Perspective". In: *Molecular Modeling and Simulation: An Interdisciplinary Guide*. Springer New York, 2010, pp. 1–40. ISBN: 9781441963512. DOI: 10.1007/978-1-4419-6351-2_1. URL: http://dx.doi.org/10.1007/978-1-4419-6351-2_1.

Schwarz, H. A. "Über einen Grenzübergang durch alternierendes Verfahren". In: *Vierteljahrsschrift der Naturforschenden Gesellschaft in Zürich* (1870).

Shalf, John. "The future of computing beyond Moore's Law". In: *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* 378.2166 (2020), p. 20190061. DOI: 10.1098/rsta.2019.0061. eprint: https://royalsocietypublishing.org/doi/pdf/10.1098/rsta.2019.0061. URL: https://royalsocietypublishing.org/doi/abs/10.1098/rsta.2019.0061.

Staff, Gunnar Andreas and Einar M. Rønquist. "Stability of the Parareal Algorithm". In: *Lecture Notes in Computational Science and Engineering*. Springer-Verlag, 2005, pp. 449–456. DOI: 10.1007/3-540-26825-1_46. URL: https://doi.org/10.1007/3-540-26825-1_46.

Steiner, Johannes et al. "Convergence of Parareal for the Navier-Stokes Equations Depending on the Reynolds Number". In: *Numerical Mathematics and Advanced Applications - ENUMATH 2013*. Springer International Publishing, Oct. 2014, pp. 195–202. ISBN: 9783319107059. DOI: 10.1007/978-3-319-10705-9_19. URL: http://dx.doi.org/10.1007/978-3-319-10705-9_19.