Analysis of a Time-Parallel Time-Integration Method (and a Historical Overview of Time-Parallel Methods)

Stefan Vandewalle

Department of Computer Science, University of Leuven, Belgium

This is joint work with Martin Gander

Section de Mathématiques, Université de Genève, Switzerland.

Outline

1. **Introduction**
2. **The Parareal Algorithm**
   - *a two-level time-integration method*
   - *potential for parallelism in time*
3. **Relation to Early Papers on Parallel ODE Methods**
   - *a shooting method*
   - *a multigrid-in-time algorithm*
   - *some more equivalences*
4. **Convergence analysis**
   - *scalar ODE problems*
   - *semi-discrete PDE problems*
5. **Concluding Remarks**

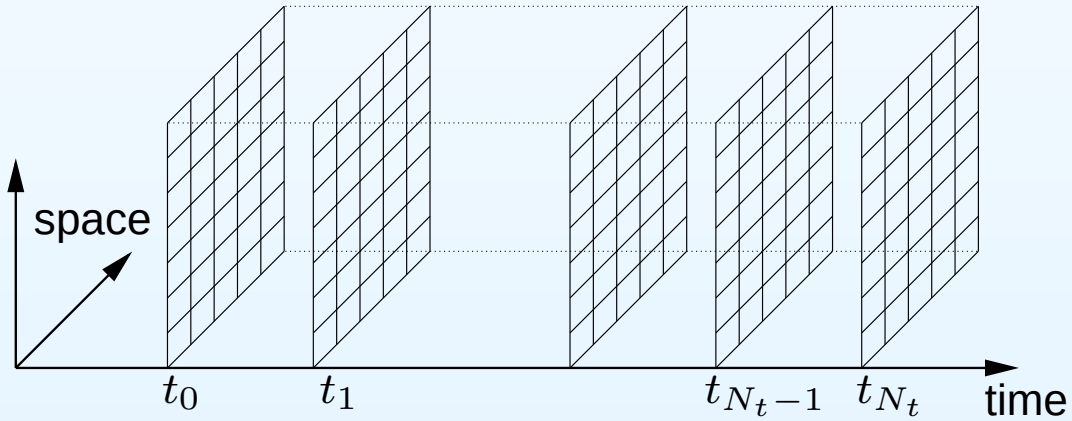
1. Introduction

- **Time-evolution** problems

  $u' = f(u)$ (ordinary differential equations)

  $\dfrac{\partial u}{\partial t} = L(u) + f$ (partial differential equations)

- **Classical methods**: **time-stepping** (explicit/implicit)

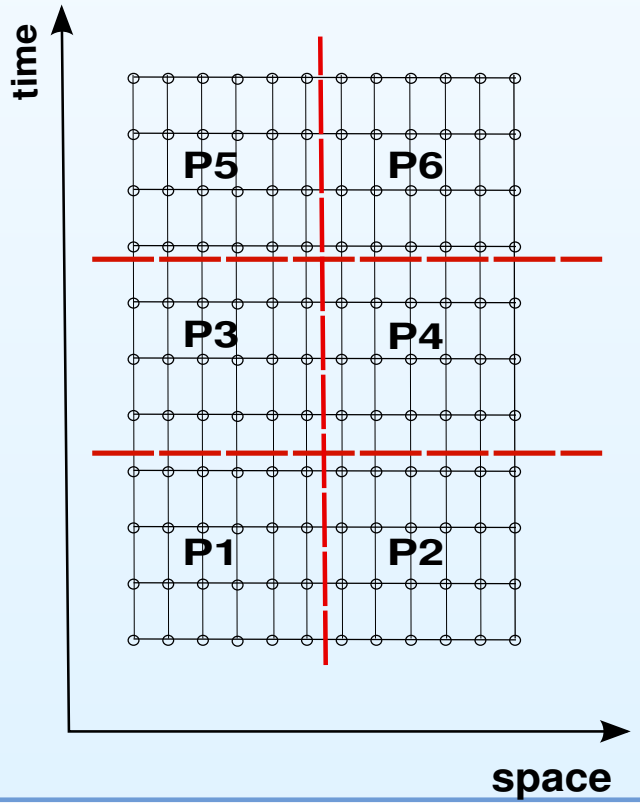


- **Computational complexity**
  - Example: parabolic PDE/Backward Euler/Multigrid
  - Serial: $\mathcal{O}(N_s\, N_t) \Rightarrow$ Parallel: $\mathcal{O}(\log^2(N_s)\, N_t)$

1. Introduction

Can parts of the solution later in time be computed before the solution earlier in time is known ?

Can the processors of a parallel computer (or multi-core chip) be assigned to different time-levels in a time-accurate computation ?



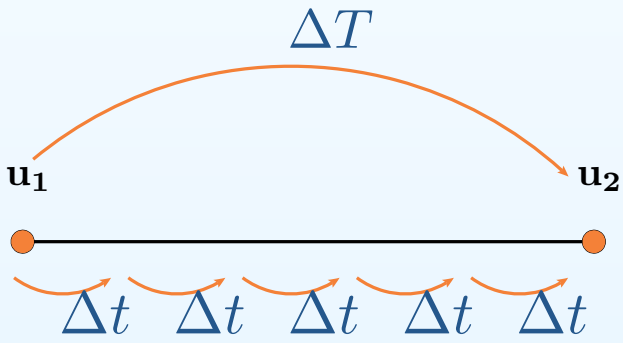
- Many people have investigated this question in 1985-1995, in the ODE literature (book: Burrage)
- E.g.: waveform relaxation, space-time grid iteration, multiple shooting, partitioning,...
- A new algorithm was presented in 2001 in the domain decomposition literature (without reference to earlier work). Since then more than 50 papers have appeared about this method.
- ***Here, we will study this algorithm and see how it relates to earlier work.***

2. The Parareal Algorithm

A “parareal" in time discretization of PDE’s
(Lions, Maday, and Turinici, C.R. Acad.Sci. Paris, t.332, 2001)

- The Parareal Algorithm for the model problem $u' = f(u)$ is defined using two propagation operators:

  1. $\mathbf{G(t_2, t_1, u_1)}$ is a rough approximation to $u(t_2)$ with initial condition $u(t_1) = u_1$,
  2. $\mathbf{F(t_2, t_1, u_1)}$ is a more accurate approximation to $u(t_2)$ with $u(t_1) = u_1$.



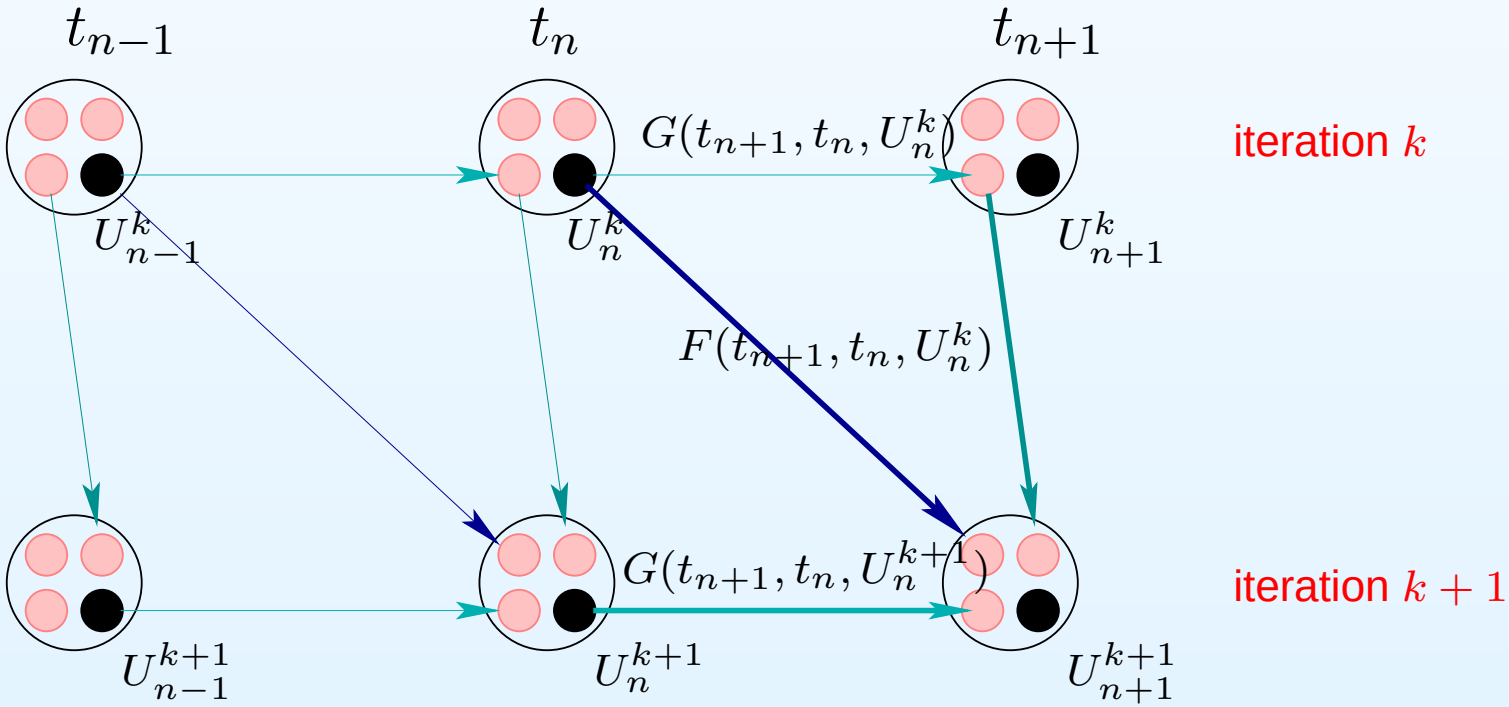
- It starts with a **coarse initial approximation** U_n^0 at the time points $t_1, t_2, \ldots, t_N$, and computes U_n^k for $k = 1, 2, \ldots$ by a **series of correction iterations**.

The Parareal Algorithm

For $k = 0, 1, 2, \ldots$ *(loop over iteration numbers)*

For $n = 0, 1, \ldots, N-1$ *(loop over the time-steps)*

$$U_{n+1}^{k+1} = G(t_{n+1}, t_n, U_n^{k+1}) + F(t_{n+1}, t_n, U_n^k) - G(t_{n+1}, t_n, U_n^k)$$



Remark: Dominant part of computation (F) is parallel (in time).

The Parareal Algorithm

$$U_{n+1}^{k+1} = G(t_{n+1}, t_n, U_n^{k+1}) + F(t_{n+1}, t_n, U_n^k) - G(t_{n+1}, t_n, U_n^k)$$

About the convergence

- Upon convergence: F-propagator accuracy on each t_n.
- Convergence guaranteed after N iterations on $t_0, t_1, \ldots, t_N$.

Two interpretations of Parareal

- solver for the F-equations (if iteration until convergence)
- new time-integrator (if #iterations fixed)

Typical theorem (L.M.T.-2001, for $u' = -au,\ u(0) = u_0$)

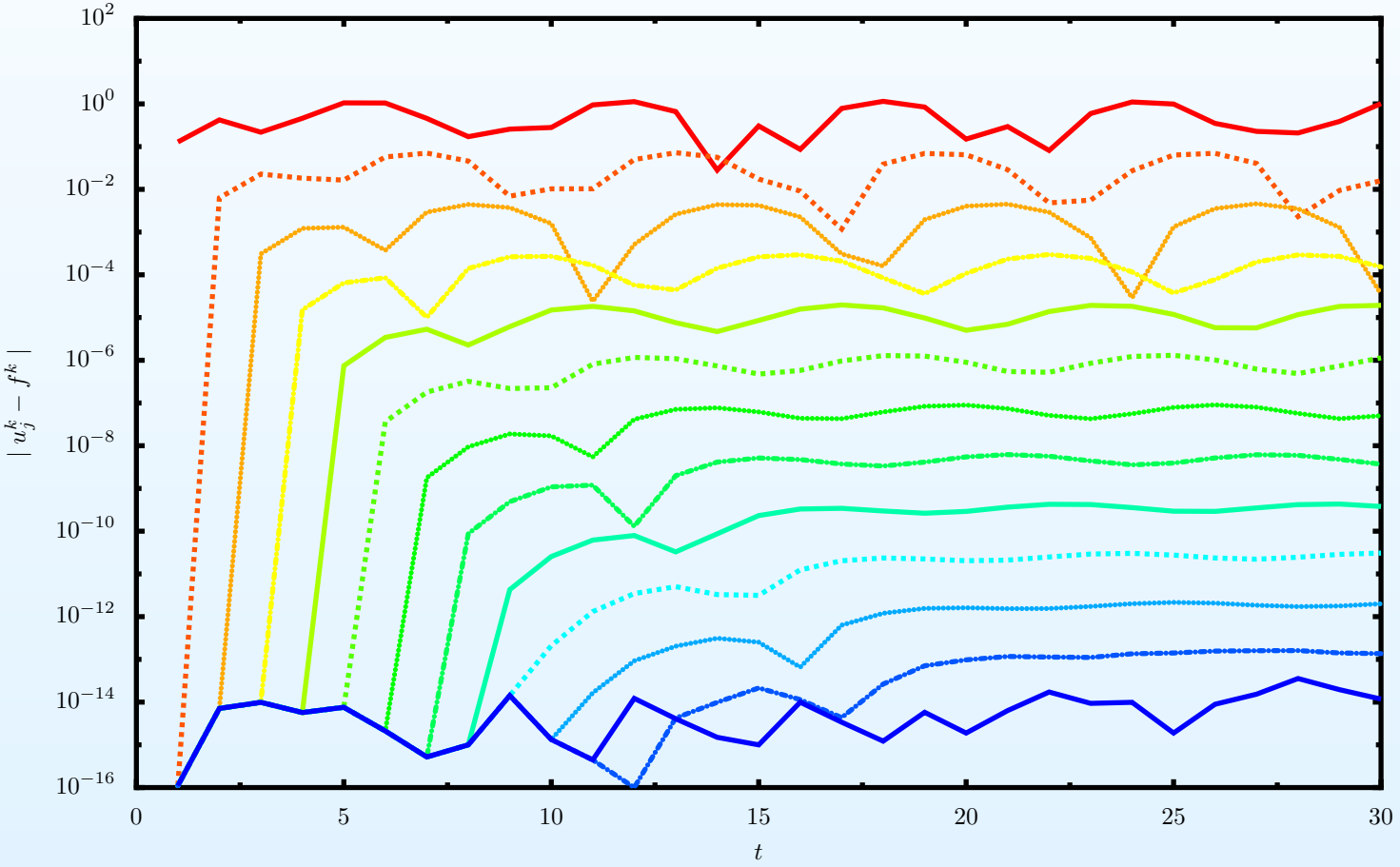
Let $F(t_{n+1}, t_n, U_n^k)$ denote the exact solution at t_{n+1}. Let $G(t_{n+1}, t_n, U_n^k)$ be the backward Euler approximation with time step ΔT. Then,

$$\max_{1 \le n \le N} |u(t_n) - U_n^k| \le C_k \Delta T^{k+1}.$$

The Parareal Algorithm: Example 1

$$u'(t) = -u(t) + \sin(t), \quad u(t_0) = 1.0, \ t \in [0, 30]$$

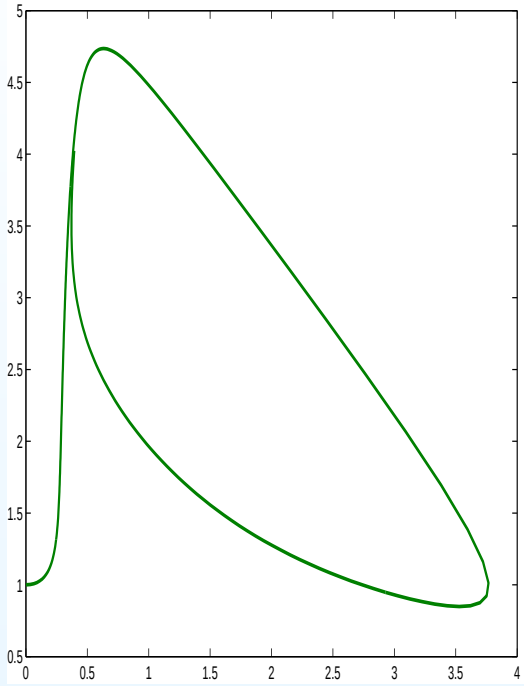
(trapezoidal rule, $\Delta T = 1.0$ and $\Delta t = 0.01$)



The Parareal Algorithm: Example 2

Brusselator

$$\begin{cases} \dot{x} &= A + x^2 y - (B+1)x \\ \dot{y} &= Bx - x^2 y, \end{cases}$$



Parameters: $A = 1$ and $B = 3$.

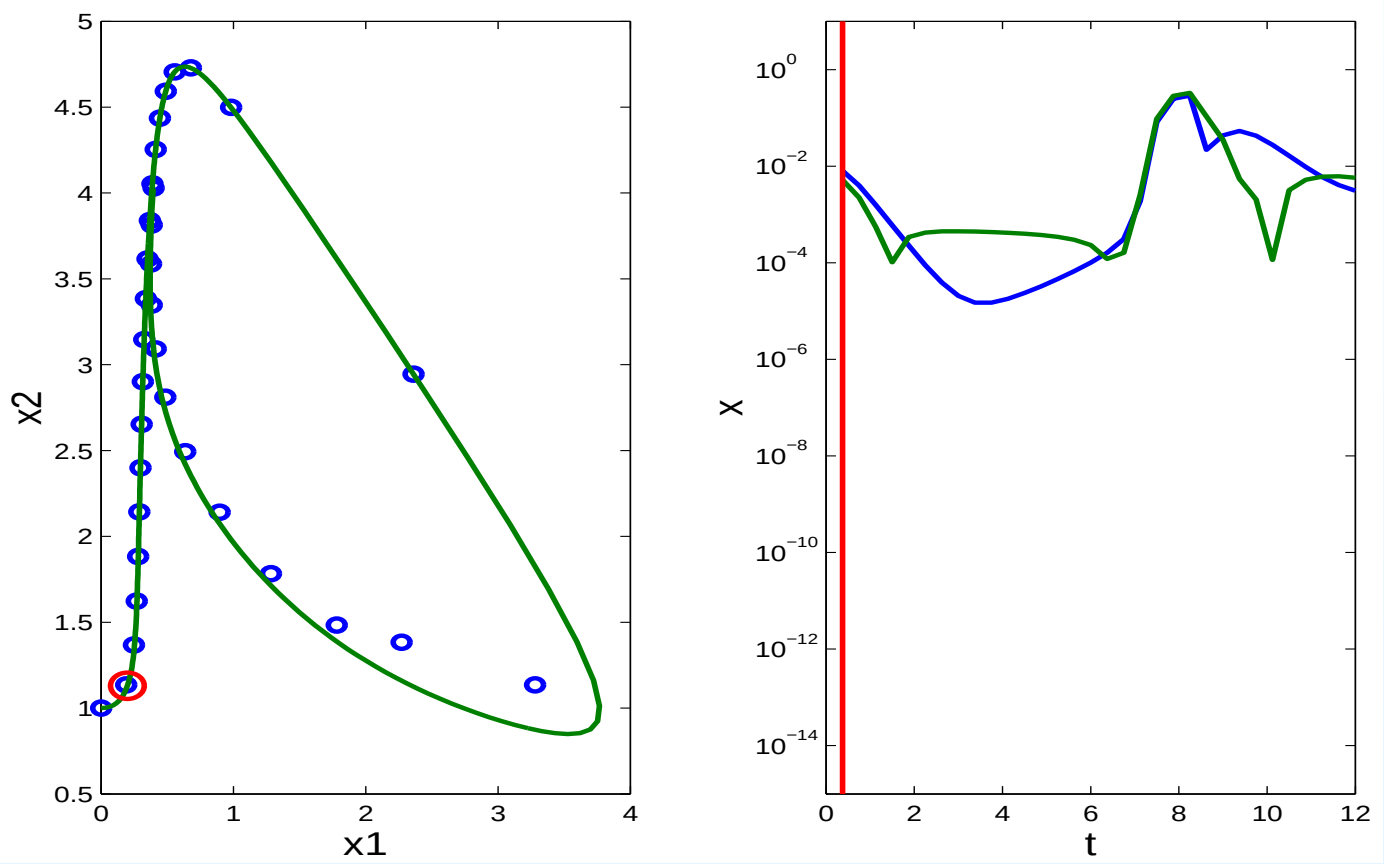
Initial conditions: $x(0) = 0$, $y(0) = 1$.

Simulation time: $t \in [0, T = 12]$

Discretization: fourth order Runge Kutta, $\Delta T = \frac{T}{32}$, $\Delta t = \frac{T}{320}$

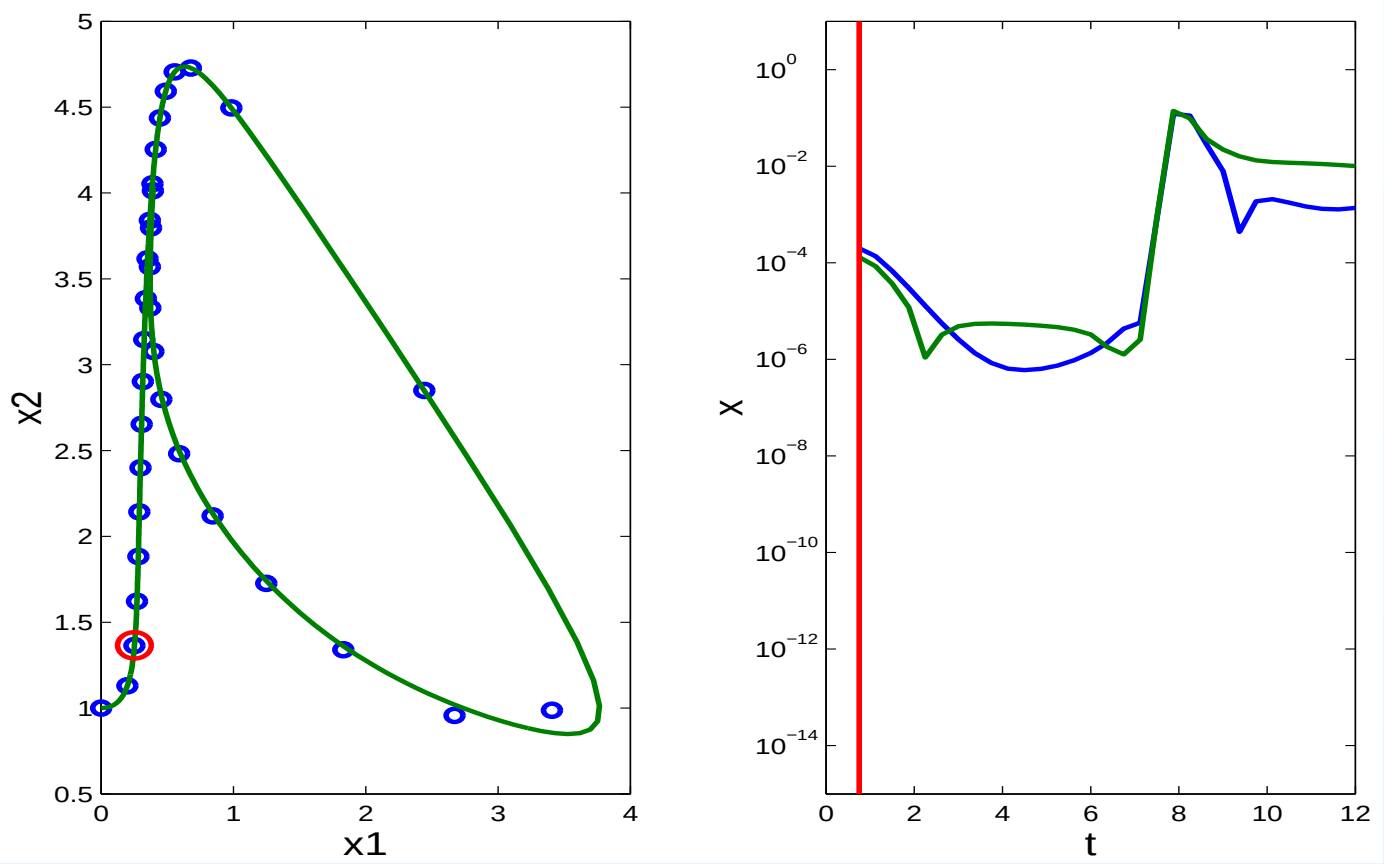
The Parareal Algorithm: Example 2

Brusselator



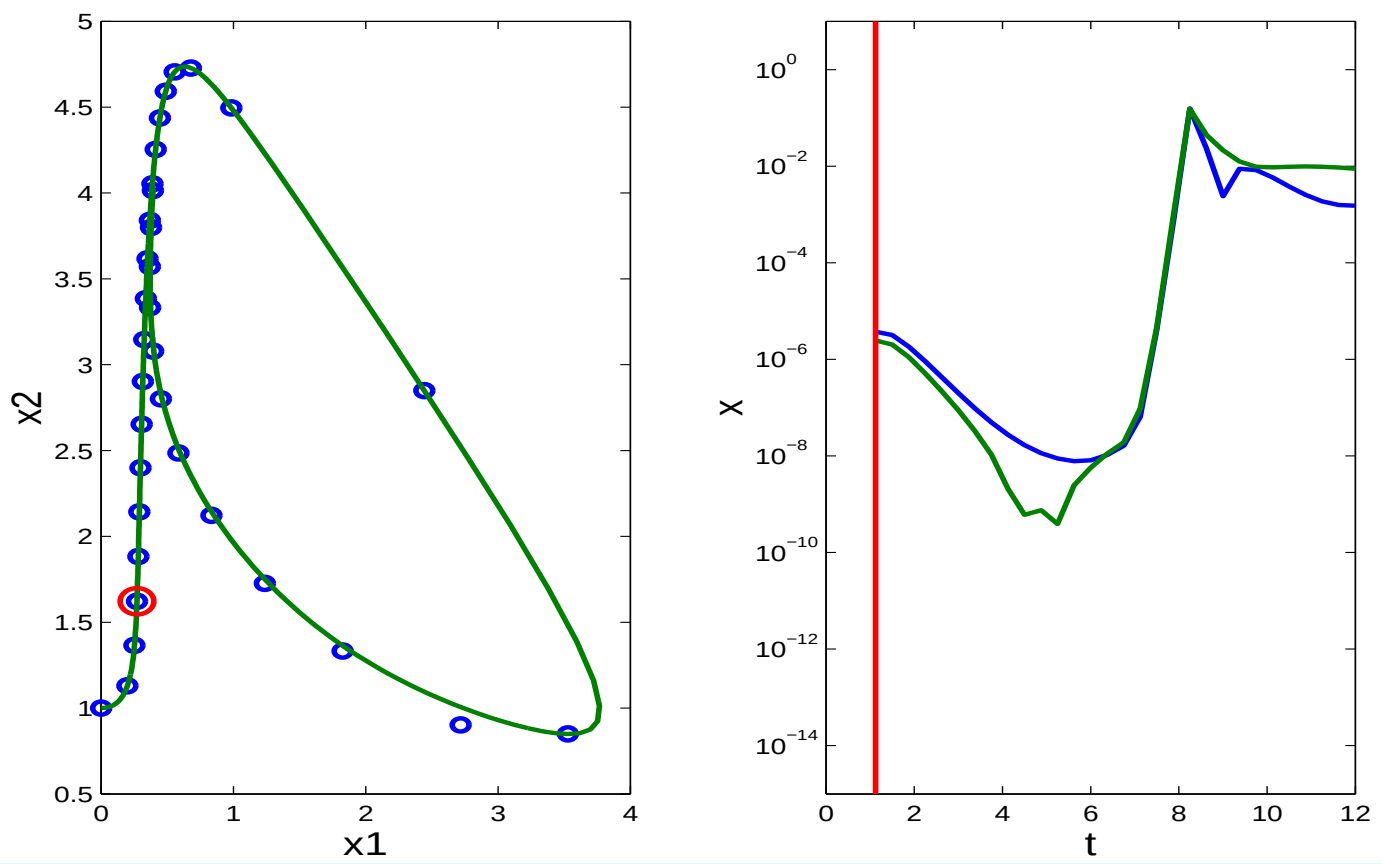
The Parareal Algorithm: Example 2

Brusselator



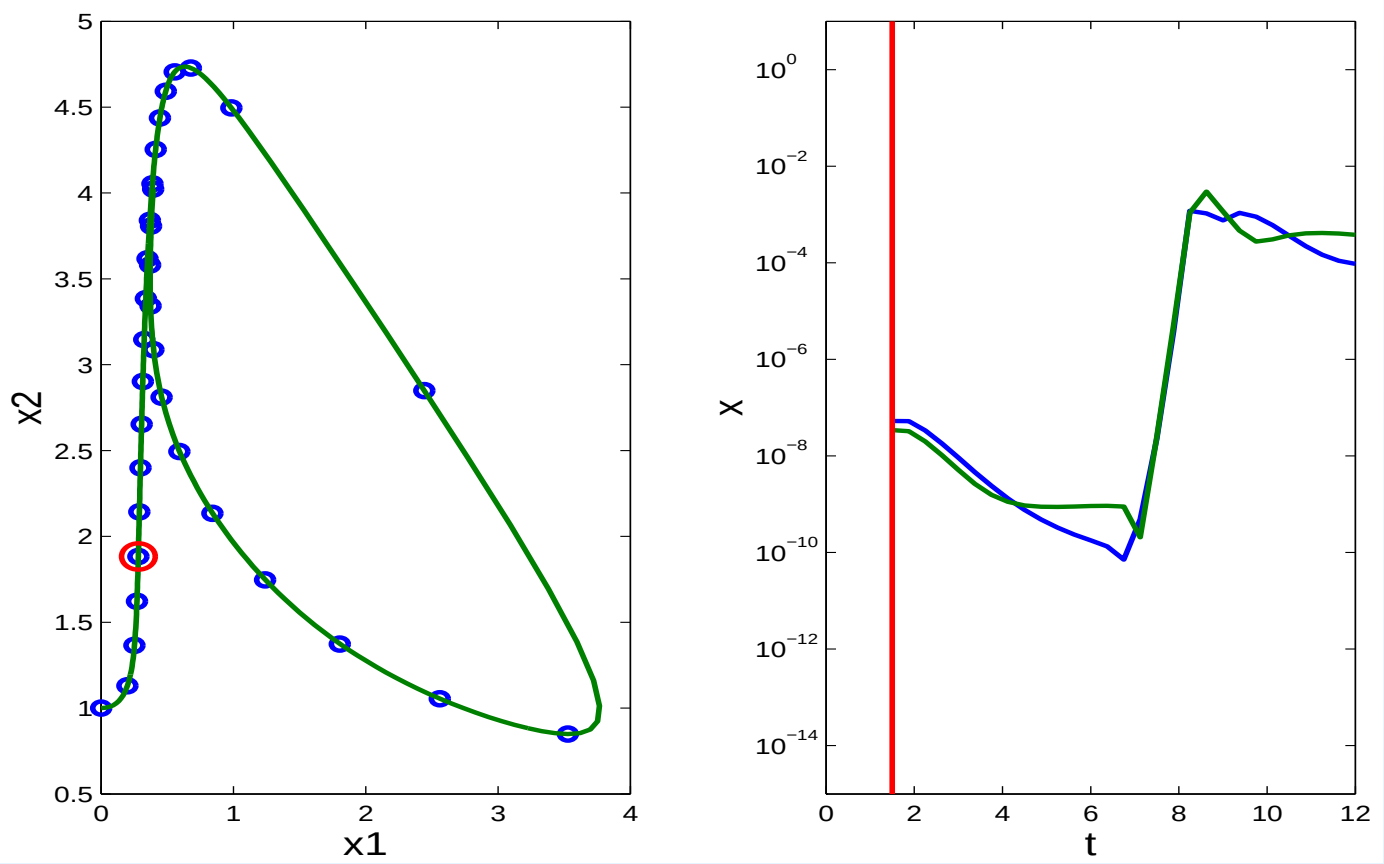
The Parareal Algorithm: Example 2

Brusselator



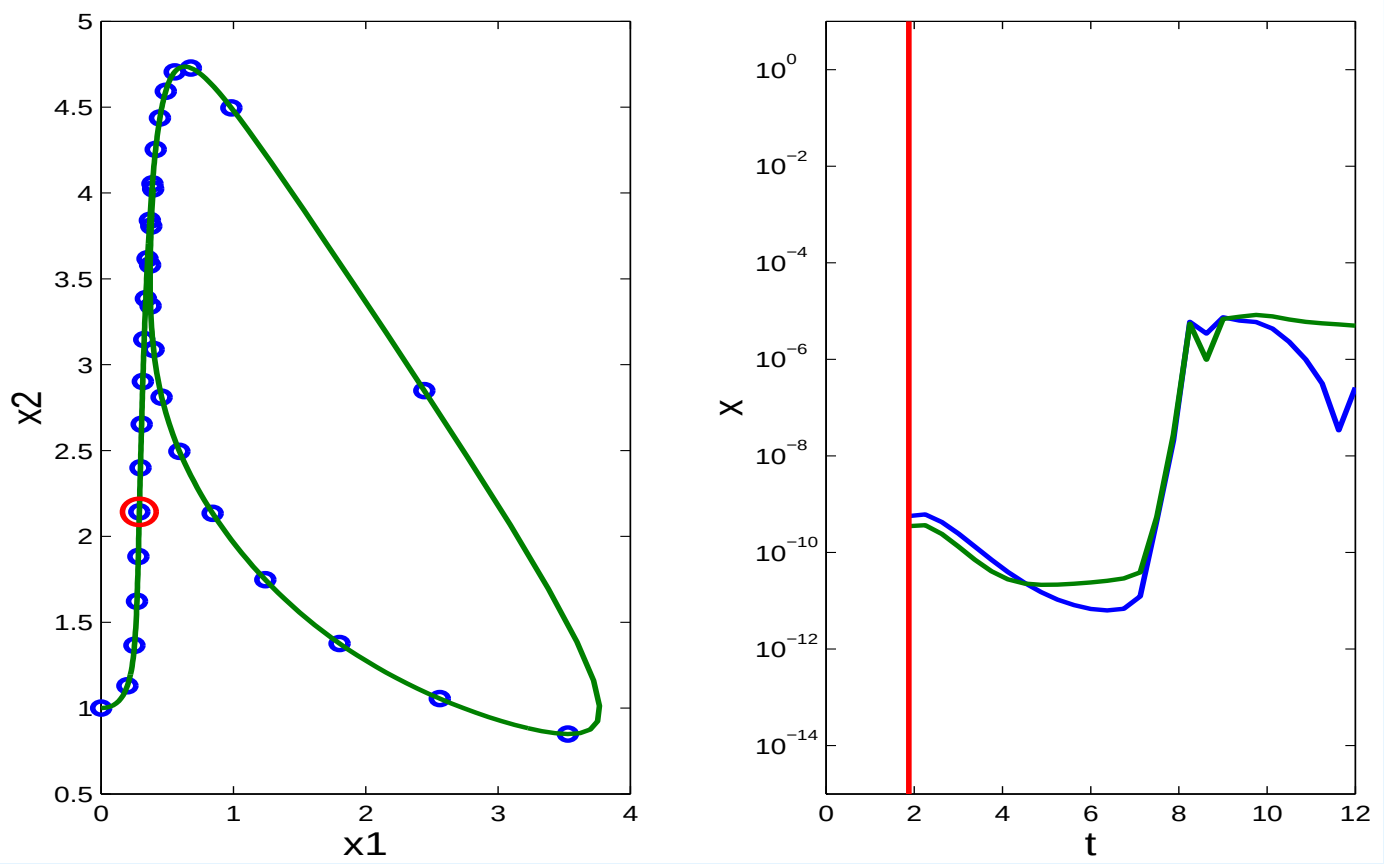
The Parareal Algorithm: Example 2

Brusselator



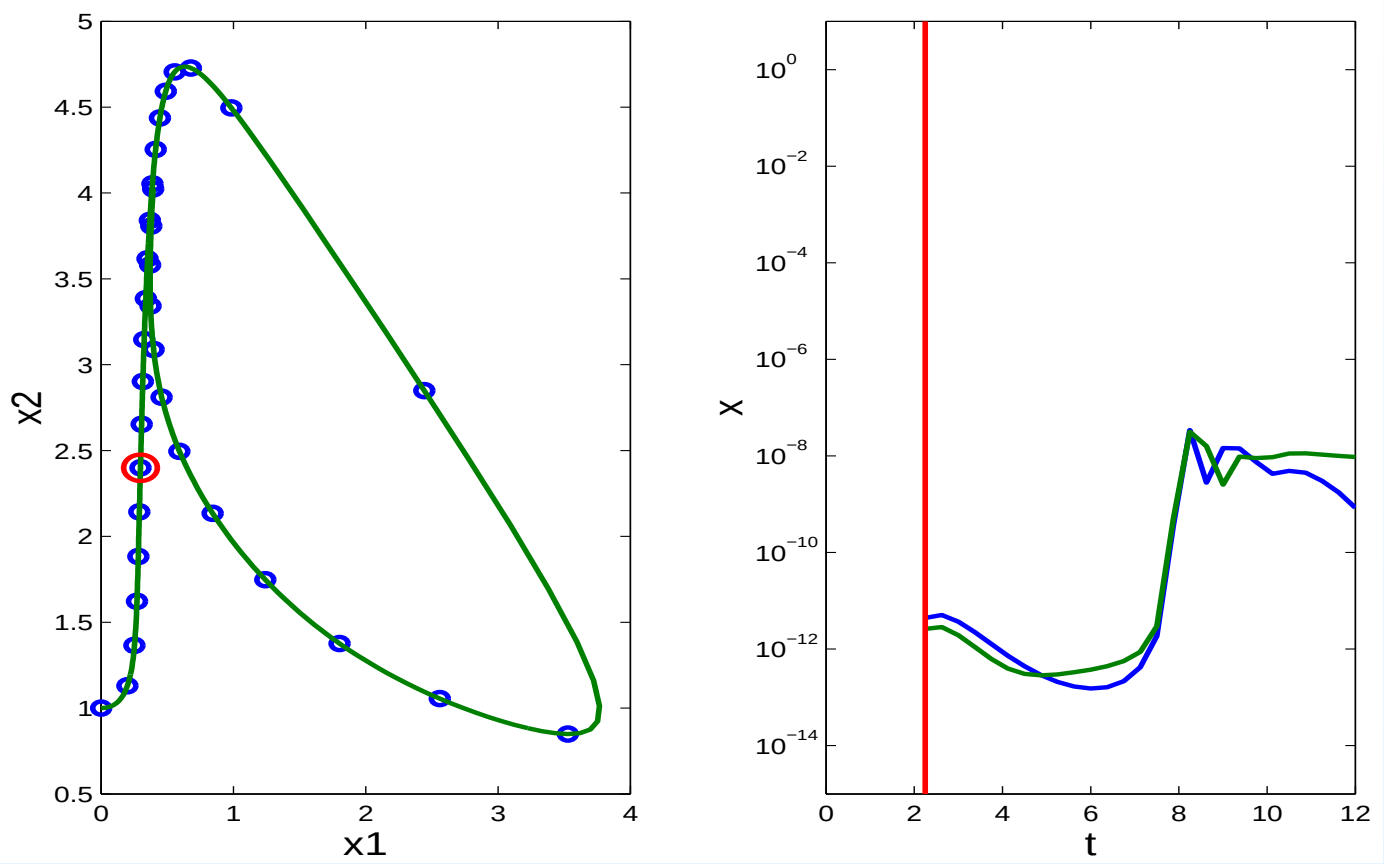
The Parareal Algorithm: Example 2

Brusselator



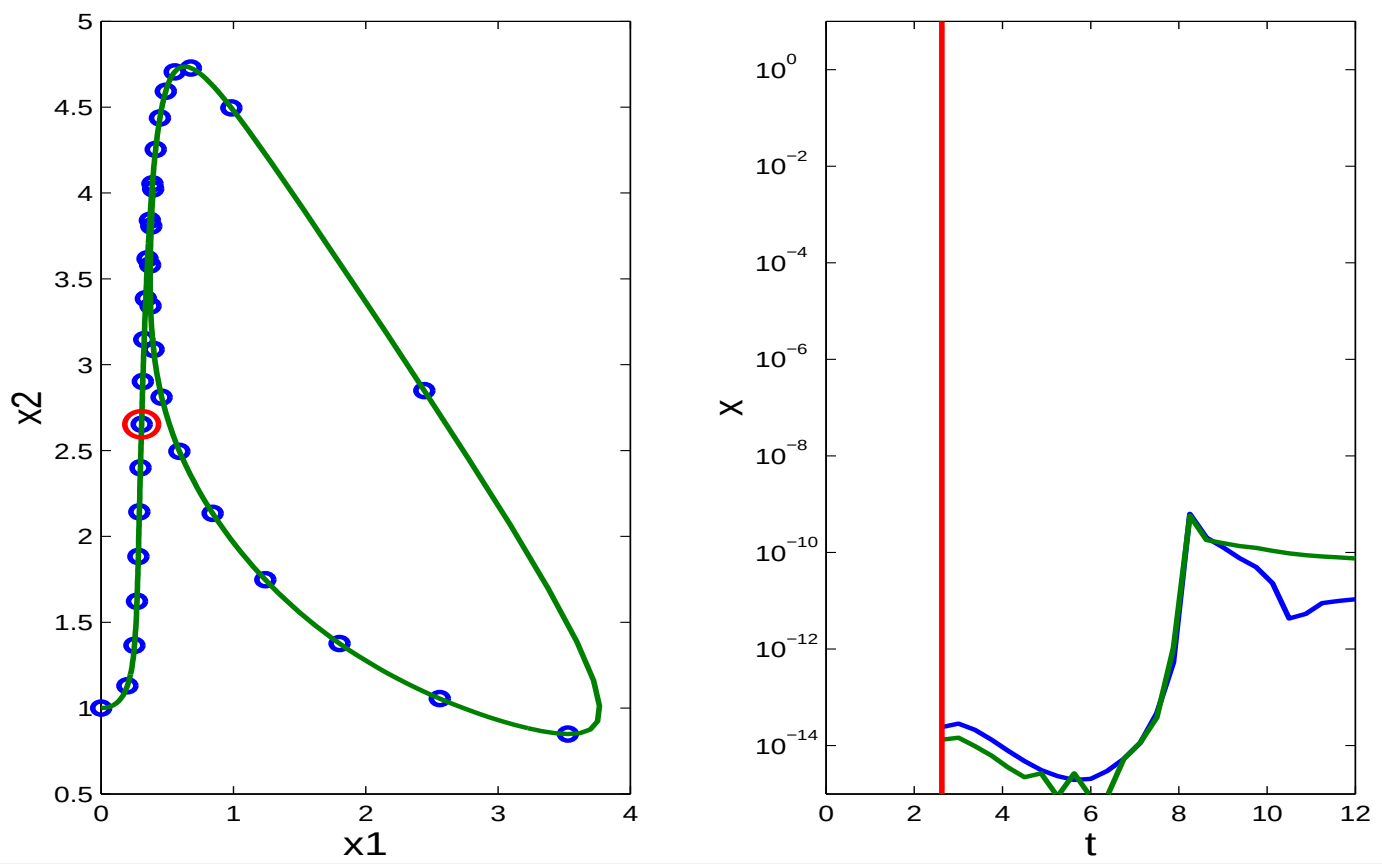
The Parareal Algorithm: Example 2

Brusselator



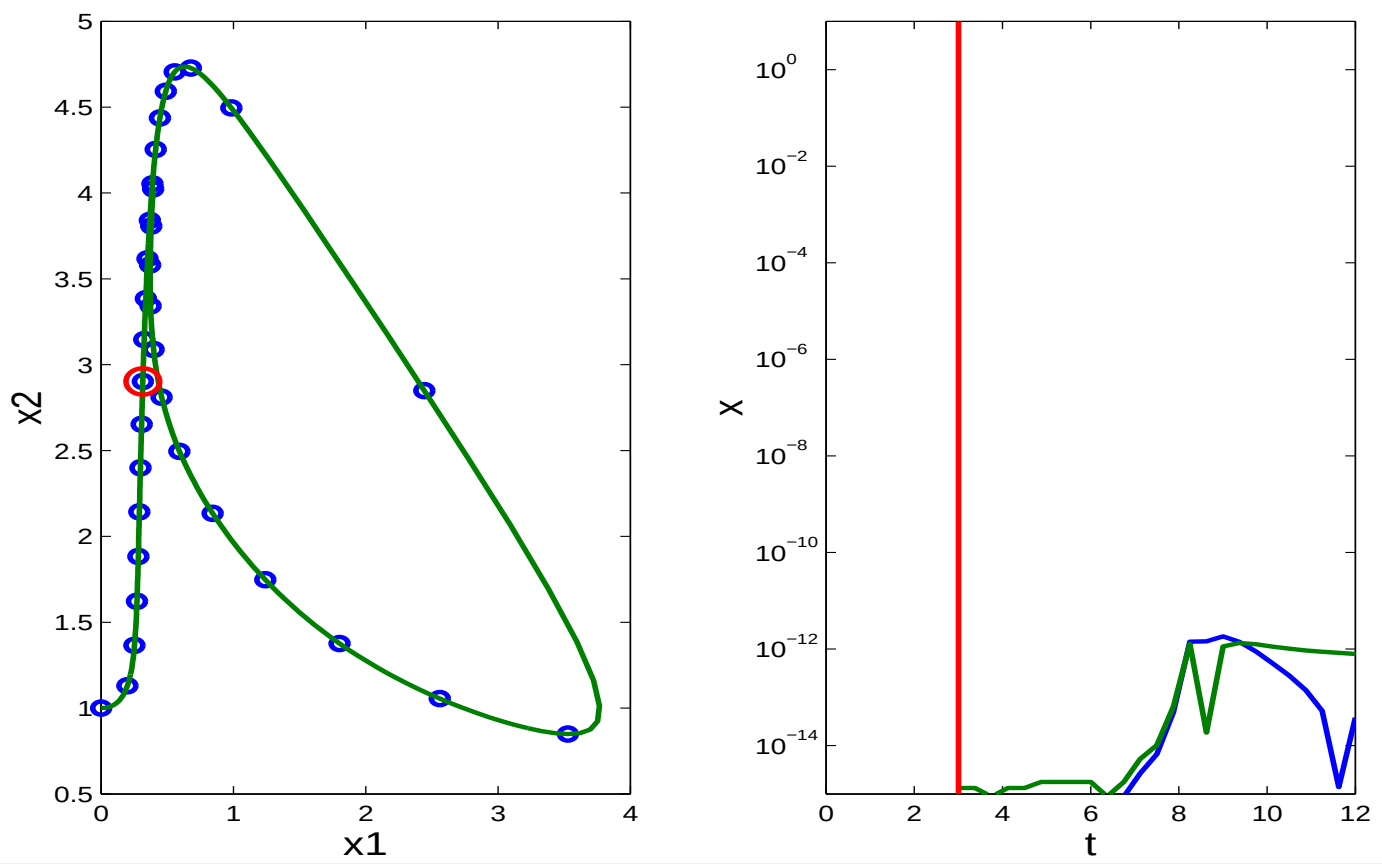
The Parareal Algorithm: Example 2

Brusselator



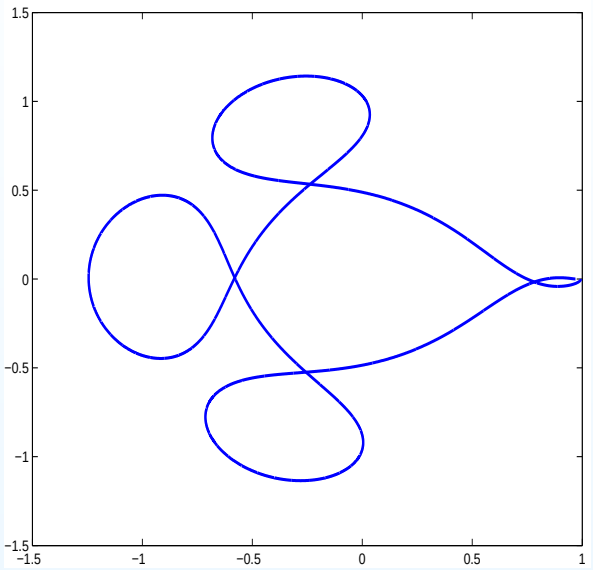
The Parareal Algorithm: Example 2

Brusselator



The Parareal Algorithm: Example 3 (Arenstorf orbit)

$$\begin{aligned}
\ddot{x} &= x + 2\dot{y} - b\frac{x+a}{D_1} - a\frac{x-b}{D_2} \\
\ddot{y} &= y - 2\dot{x} - b\frac{y}{D_1} - a\frac{y}{D_2},
\end{aligned}$$



Parameters: $a = 0.01227$, $b = 1 - a$.

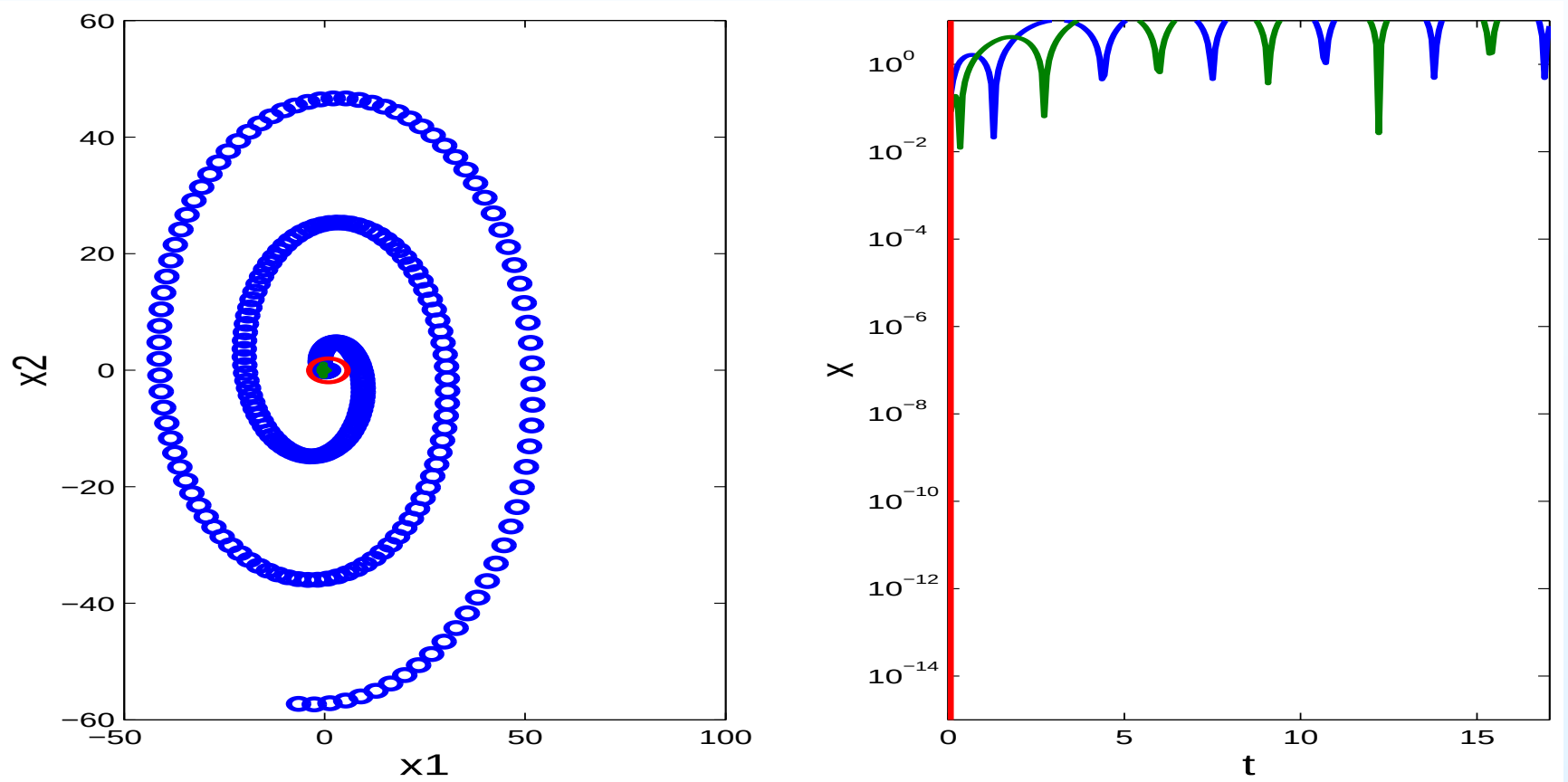
Initial conditions: $x(0) = 0.994$, $\dot{x} = 0$, $y(0) = 0$, $\dot{y}(0) = -2.00158$

Simulation time: $t \in [0, T = 17.06]$

Discretization: Fourth order Runge Kutta, $\Delta T = \frac{T}{250}$, $\Delta t = \frac{T}{10000}$.

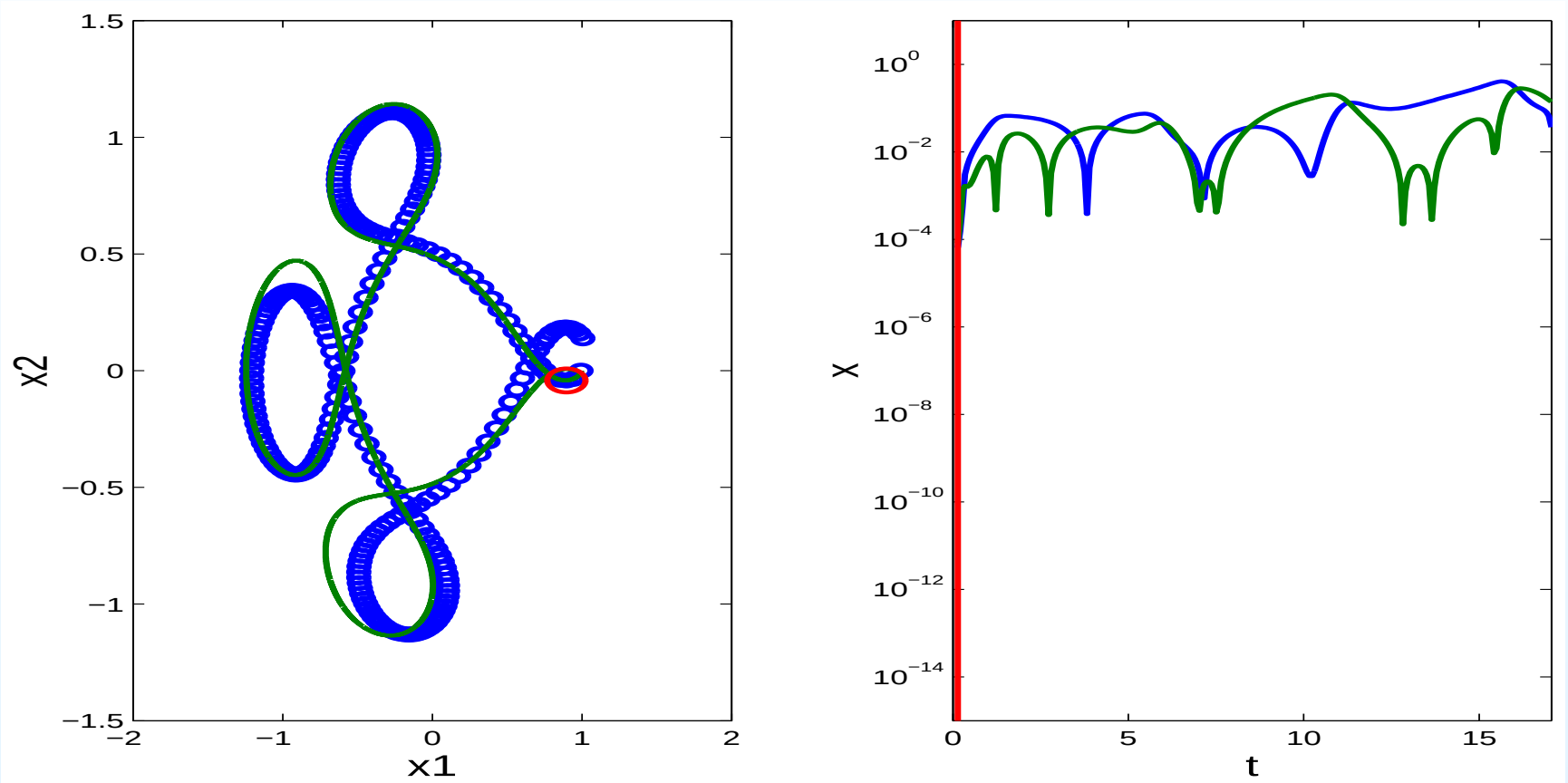
The Parareal Algorithm: Example 3

Arenstorf orbit



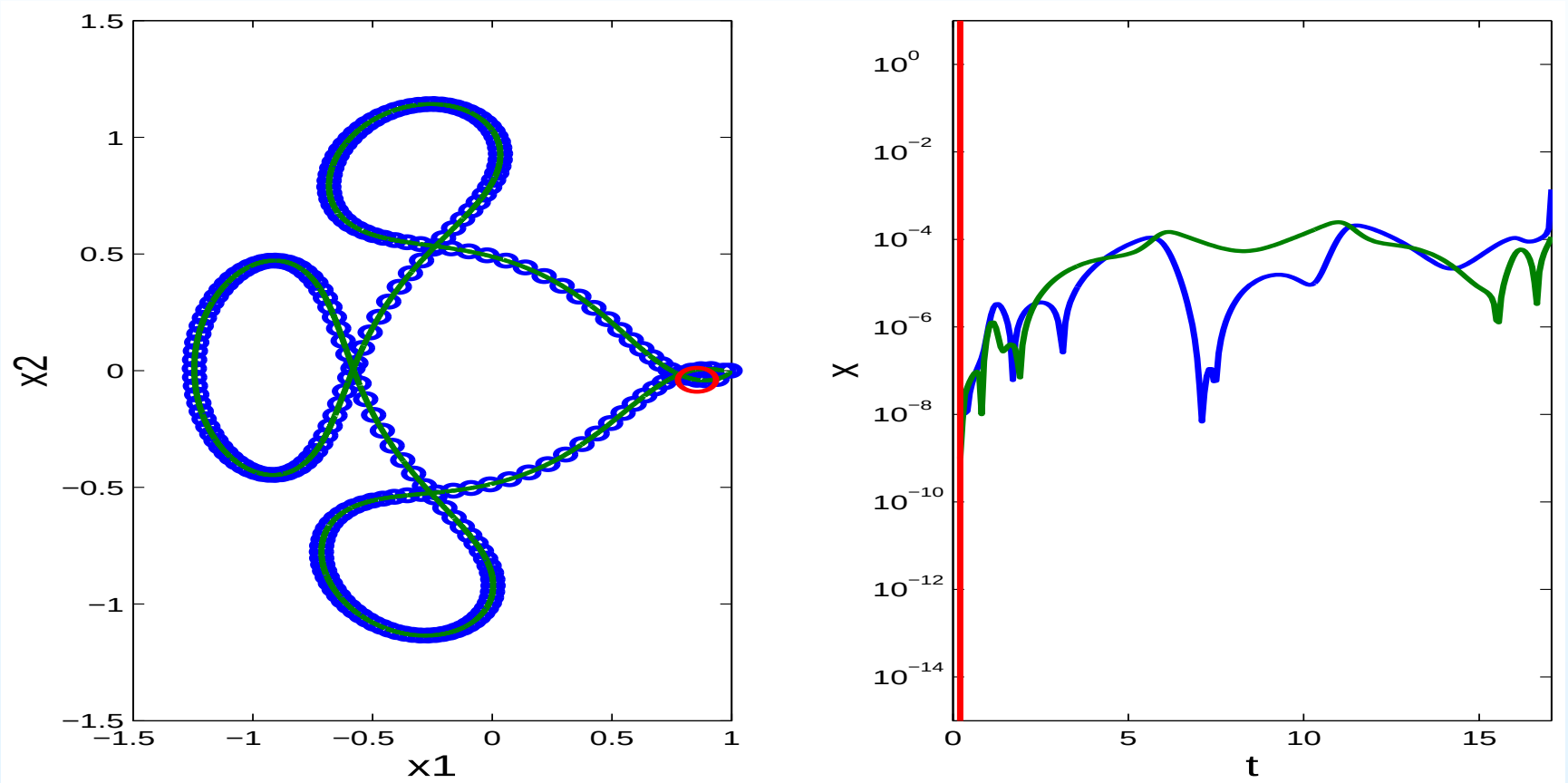
The Parareal Algorithm: Example 3

Arenstorf orbit



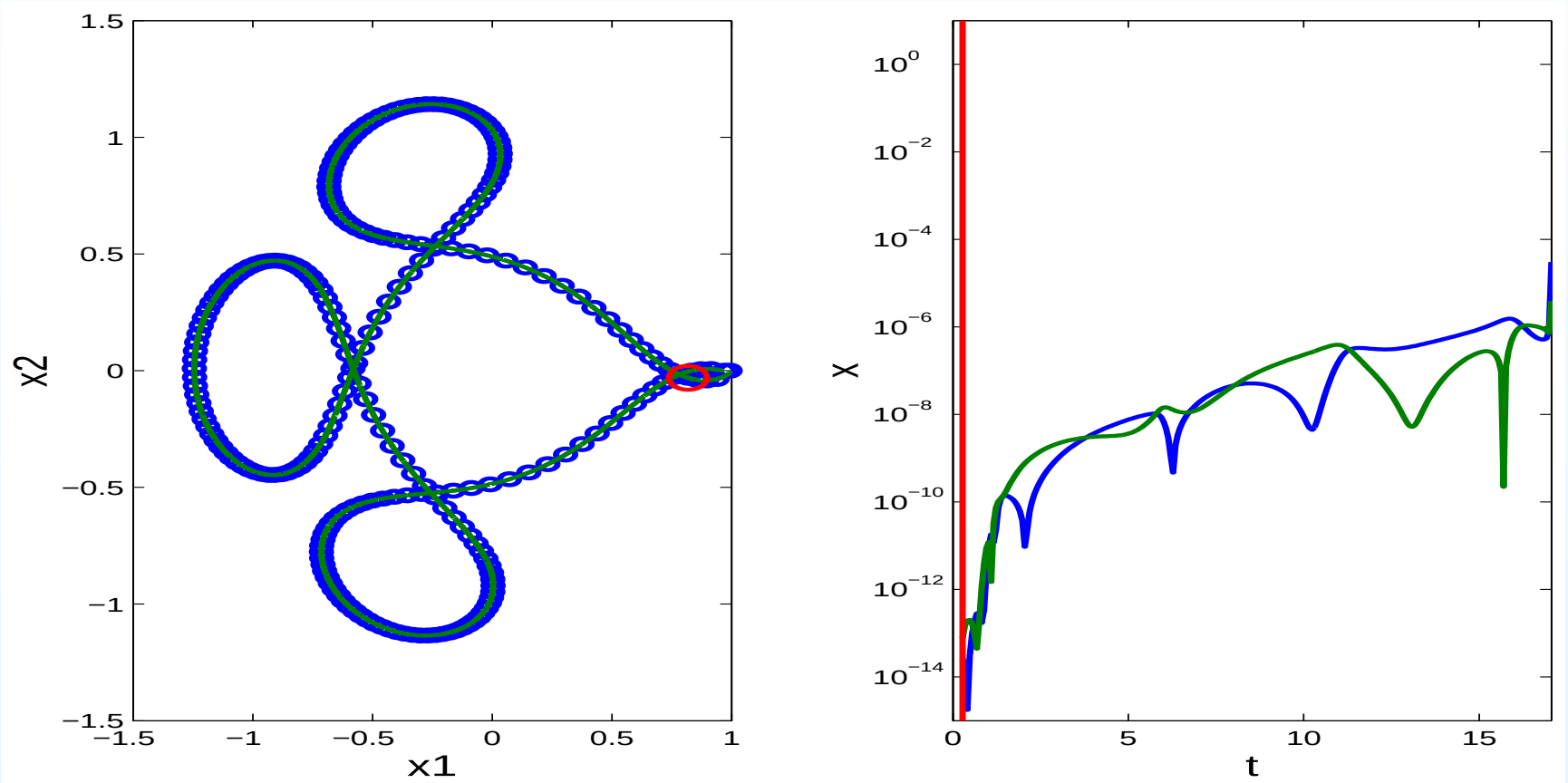
The Parareal Algorithm: Example 3

Arenstorf orbit



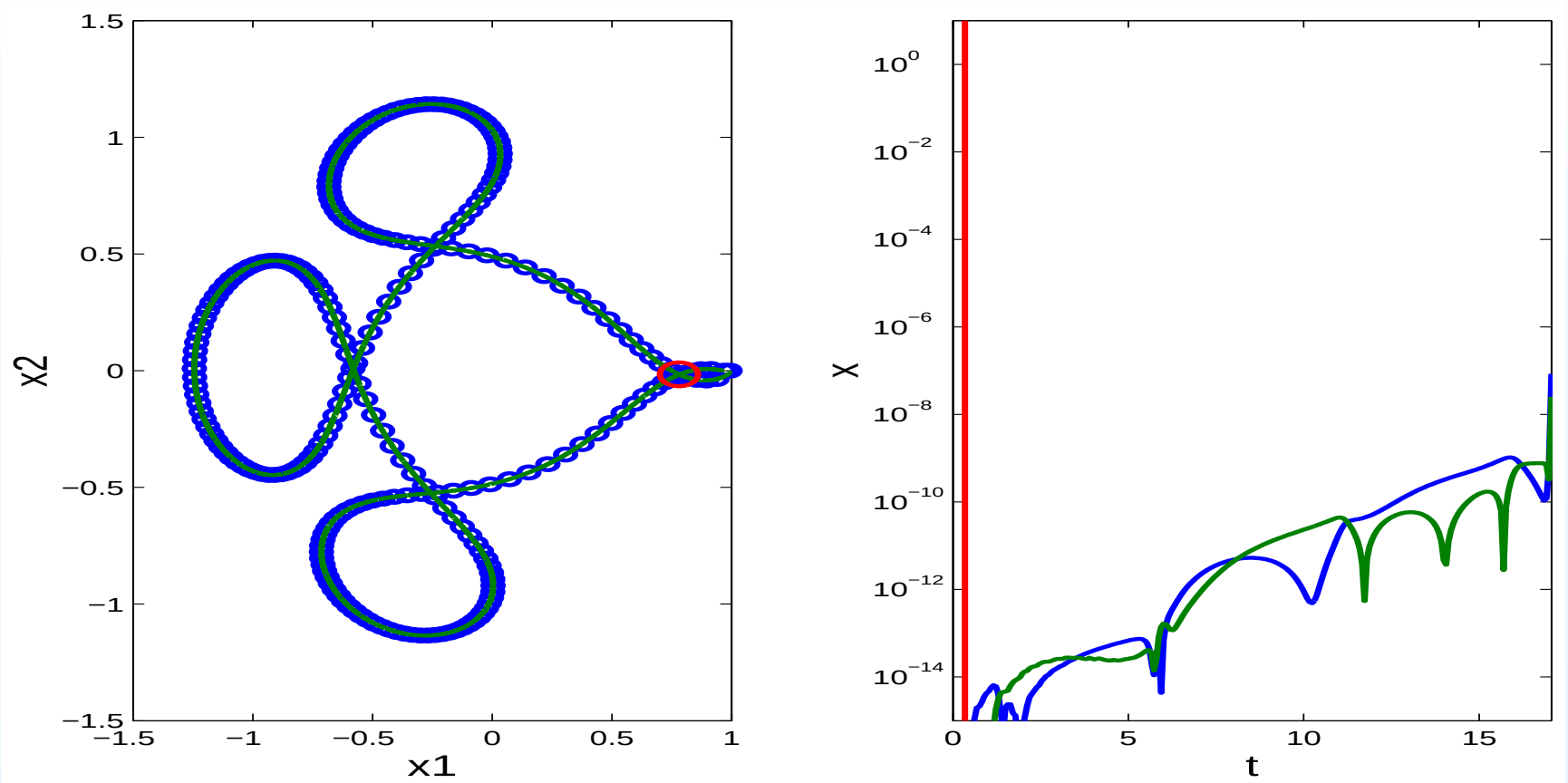
The Parareal Algorithm: Example 3

Arenstorf orbit



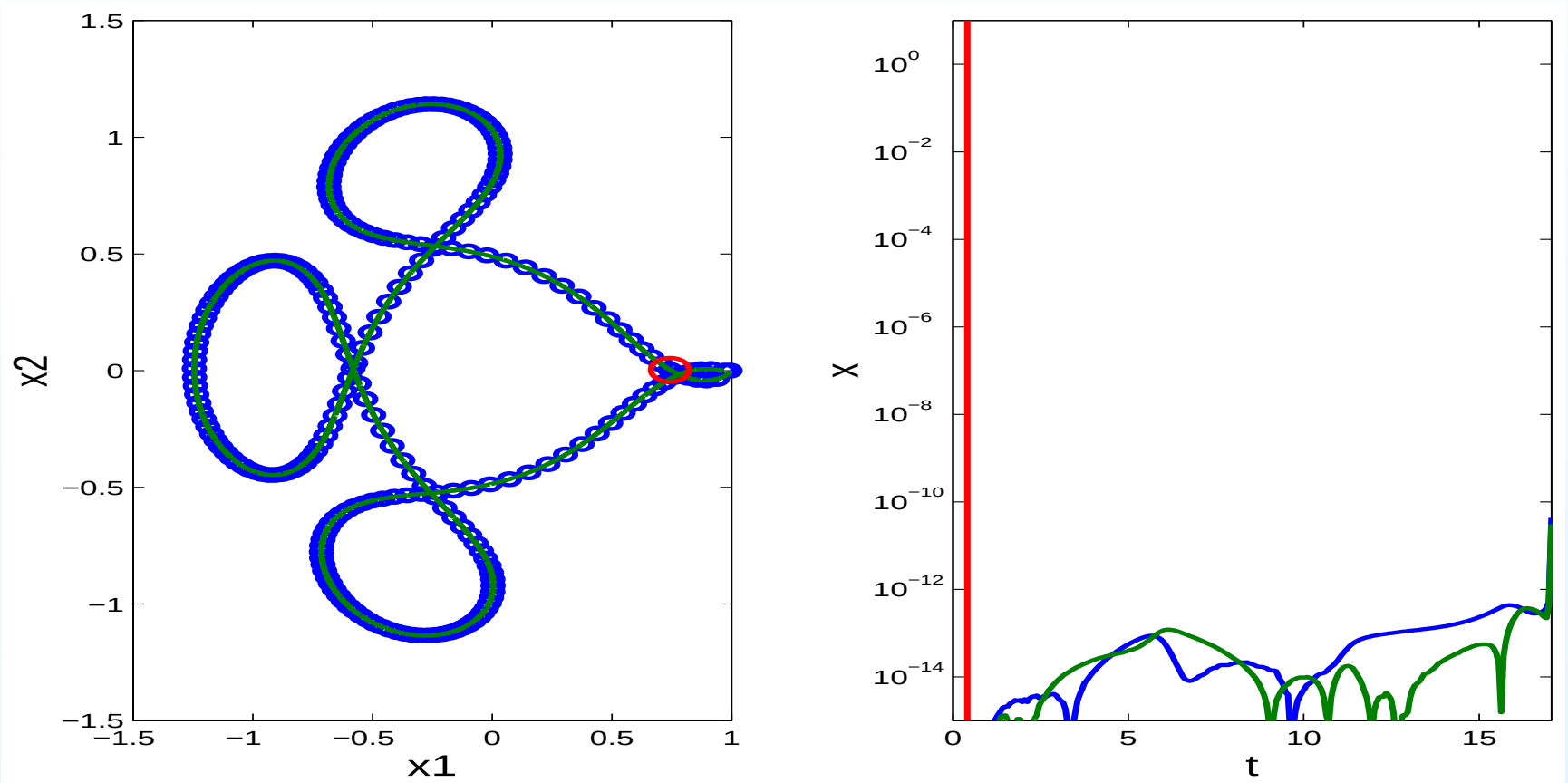
The Parareal Algorithm: Example 3

Arenstorf orbit



The Parareal Algorithm: Example 3

Arenstorf orbit



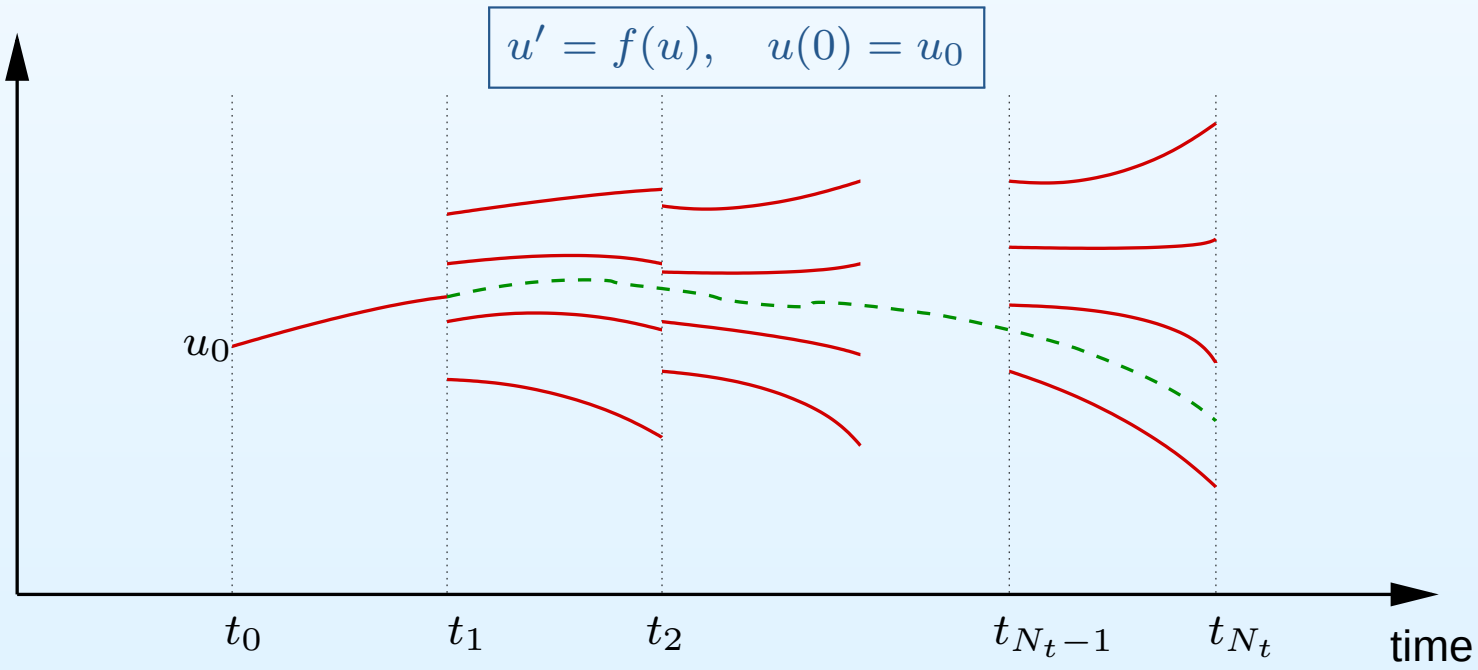
The Parareal Algorithm

- **Speedup:** $S = \frac{T_S}{T_P} = \frac{Nt_F}{Nt_G + K(Nt_G + \frac{N}{P}t_F)} \approx P/K$
  - P #processors – K #iterations – N #time-steps
  - t_F (t_G): cost of 1 step of F-propagator (G-propagator)
  - Very limited speedup if many iterations !
  - Not useful on single processor
- **Many extensions**
  - 'Multilevel' in time (Garrido, Lee, Fladmark, and Espedal)
  - 'Multilevel' in space (Fisher, Hecht, and Maday)
  - 'Domain decomposition' in space (Maday and Turinici)
  - 'Subspace filtering' (Farhat and Cortial)
- **Many applications**:
  - Fluid, structure, molecular dynamics, control, finance,...

3. Relation to Early Papers on Parallel ODE Methods

J. Nievergelt, **Parallel Methods for Integrating Ordinary Differential Equations**. Comm. of the ACM, Vol 7(12), 1964.

"For the last 20 years, one has tried to speed up numerical computation mainly by providing ever faster computers. ***Today, as it appears that one is getting closer to the maximal speed of electronic components****, emphasis is put on allowing operations to be performed in parallel. In the near future, much of numerical analysis will have to be recast in a more "parallel" form."*



Multiple Shooting

For the model problem

$$u' = f(u), \quad u(0) = u_0, \quad t \in [0, 1]$$

one splits the time interval into subintervals and then solves

$$\begin{array}{rclrclrcl} u_0' & = & f(u_0), & u_1' & = & f(u_1), & u_2' & = & f(u_2), \\ u_0(0) & = & U_0, & u_1(\frac{1}{3}) & = & U_1, & u_2(\frac{2}{3}) & = & U_2, \end{array}$$

together with the conditions

$$U_0 - u_0 = 0, \quad U_1 - u_0(\frac{1}{3}, U_0) = 0, \quad U_2 - u_1(\frac{2}{3}, U_1) = 0$$

$$\Longleftrightarrow \quad F(\boldsymbol{U}) = 0, \quad \boldsymbol{U} = (U_0, U_1, U_2)^T.$$

Multiple Shooting

Solving $F(\boldsymbol{U}) = 0$ with Newton-method leads to

$$\begin{pmatrix} U_0^{k+1} \\ U_1^{k+1} \\ U_2^{k+1} \end{pmatrix} = \begin{pmatrix} U_0^{k} \\ U_1^{k} \\ U_2^{k} \end{pmatrix} - \begin{bmatrix} 1 & & \\ -\frac{\partial u_0}{\partial U_0}(\frac{1}{3}, U_0^k) & 1 & \\ & -\frac{\partial u_1}{\partial U_1}(\frac{2}{3}, U_1^k) & 1 \end{bmatrix}^{-1} \begin{pmatrix} U_0^k - u_0 \\ U_1^k - u_1(\frac{1}{3}, U_0^k) \\ U_2^k - u_1(\frac{2}{3}, U_1^k) \end{pmatrix}$$

Multiplying through by the matrix, we find the recurrence

$$\begin{aligned} U_0^{k+1} &= u_0, \\ U_1^{k+1} &= u_0(\tfrac{1}{3}, U_0^k) + \tfrac{\partial u_0}{\partial U_0}(\tfrac{1}{3}, U_0^k)(U_0^{k+1} - U_0^k), \\ U_2^{k+1} &= u_1(\tfrac{2}{3}, U_1^k) + \tfrac{\partial u_1}{\partial U_1}(\tfrac{2}{3}, U_1^k)(U_1^{k+1} - U_1^k). \end{aligned}$$

General multiple shooting with N intervals

$$U_{n+1}^{k+1} = u_n(t_{n+1}, U_n^k) + \frac{\partial u_n}{\partial U_n}(t_{n+1}, U_n^k)(U_n^{k+1} - U_n^k).$$

Parareal is a Multiple Shooting Method

Theorem: If in the multiple shooting method:

$$u_n(t_{n+1}, U_n^k) \approx F(t_{n+1}, t_n, U_n^k)$$

$$\frac{\partial u_n}{\partial U_n}(t_{n+1}, U_n^k)(U_n^{k+1} - U_n^k) \approx G(t_{n+1}, t_n, U_n^{k+1}) - G(t_{n+1}, t_n, U_n^k)$$

then the multiple shooting and parareal algorithms coincide.

Proof: By observation. Compare:

$$U_{n+1}^{k+1} = u_n(t_{n+1}, U_n^k) + \frac{\partial u_n}{\partial U_n}(t_{n+1}, U_n^k)(U_n^{k+1} - U_n^k),$$

$$U_{n+1}^{k+1} = F(t_{n+1}, t_n, U_n^k) + G(t_{n+1}, t_n, U_n^{k+1}) - G(t_{n+1}, t_n, U_n^k).$$

Parareal=multiple shooting with a coarse approximate Jacobian

Parareal is a Multiple Shooting Method

Different approximations for $\frac{\partial u_n}{\partial U_n}(t_{n+1}, U_n^k)(U_n^{k+1} - U_n^k)$ lead to different time-parallel algorithms.

- **A. Bellen and M. Zennaro**: (J.C.A.M. 1989) *time-parallel algorithm for difference equations* Approximate each column of Jacobian $\frac{\partial u_n}{\partial U_n}$ in a finite difference way.
- **C. Farhat and M. Chandesris**: (Int.J.Num.Meth.Engr. 2003) *time-decomposed parallel time-integrator* Compute Jacobian $\frac{\partial u_n}{\partial U_n}$ approximately by solving the variational equations on the coarse mesh.

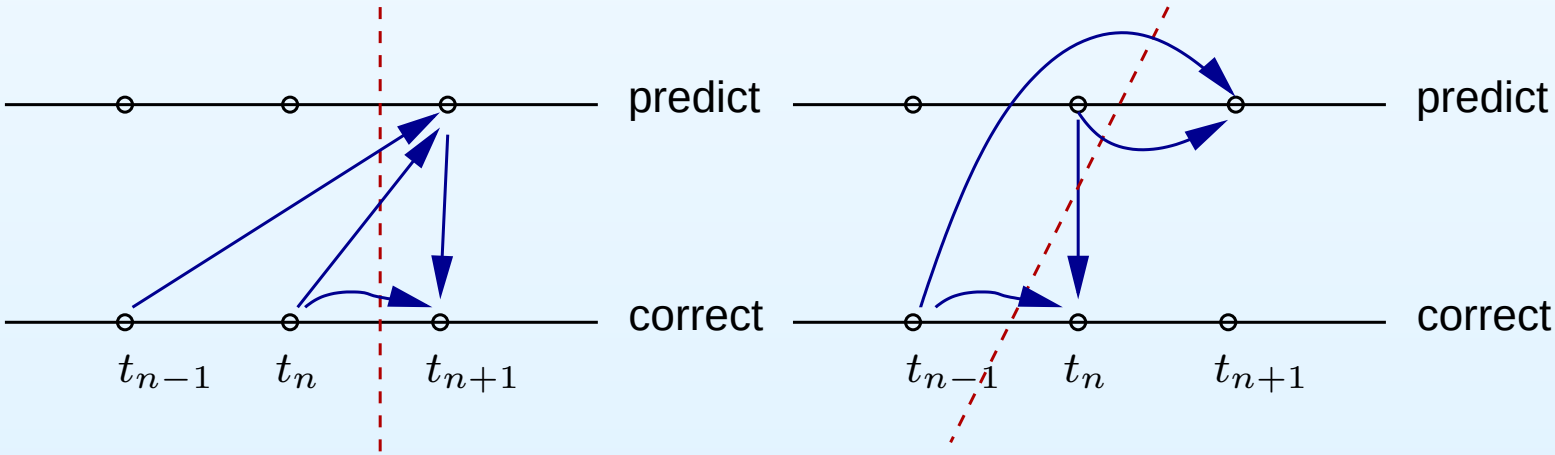
$$\left(\frac{\partial \boldsymbol{u}_n}{\partial \boldsymbol{U}_n}\right)' = \boldsymbol{f}'(\boldsymbol{u}_n)\frac{\partial \boldsymbol{u}_n}{\partial \boldsymbol{U}_n}, \qquad \frac{\partial \boldsymbol{u}_n}{\partial \boldsymbol{U}_n}(t_n) = I$$

Relation to Early Papers on Parallel ODE Methods

W. Miranker and W. Liniger, **Parallel Methods for the Numerical Integration of Ordinary Differential Equations**. Math. Comp., Vol 21, 1967.

*"It appears at first sight that the sequential nature of the numerical methods do not permit a parallel computation on all of the processors to be performed. We say that **the front of computation** is too narrow to take advantage of more than one processor... Let us consider how we might widen the computation front."*

$$u' = f(u), \quad u(0) = u_0$$



predictor-corrector method for time-integration

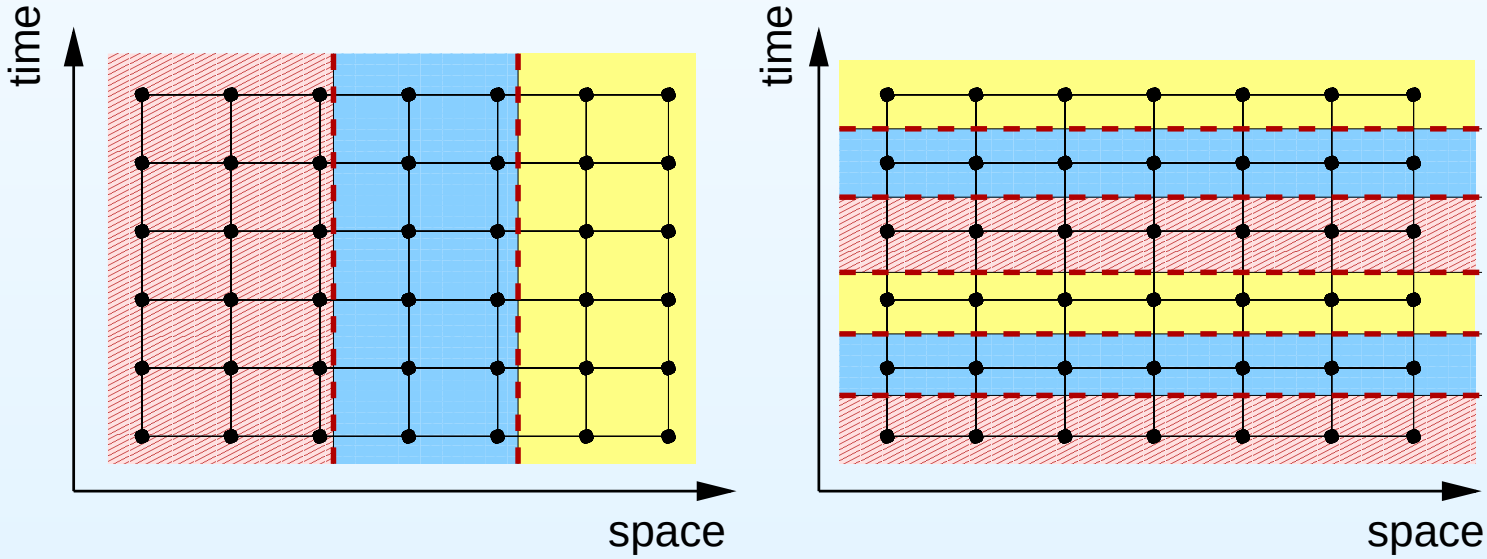
Space-Time Iterative Methods

- **Windowed Relaxation**

  J. Saltz, V. Naik, D. Nicol. Reduction of the effects of communication delays in scientific algorithms on message passing MIMD architectures. SISC 8(1), 1987.

- **Parallel Time-stepping**

  D. Womble. A time-stepping algorithm for parallel computers. SISC 11(5), 1990.



Start the iteration on a new time-level, before the iteration on the previous time-level has converged.

A Negative Result

Deshpande, Malhotra, Douglas, Schultz. A rigorous analysis of time domain parallelism. Parallel Alg. and Appl. (6) 1995.

- if a good solver is used on each time step, no parallel speedup is possible.
- if a very slow solver is used on each time step, a small parallel speedup can be achieved.

Quote from the technical report 1993:

"We show that this approach is not normally useful".

Parabolic Multigrid Methods

- Apply **a multigrid algorithm** on the **space-time mesh**
- Provide rapid information propagation forward in time, by means of a (set of) coarse meshes.
- Three basic variants:
  - **Time-parallel multigrid** (Hackbusch, 1984; Bastian, Burmeiser, Horton, 1990; Horton 1992; Oosterlee, 1992;...)
  - **Multigrid waveform relaxation** (Lubich and Ostermann, 1987; Vdw and Piessens,1988;...)
  - **Space-time multigrid** (Horton and Vdw, 1995)
- They differ w.r.t
  - smoother (pointwise/line-wise; Jacobi/GS;...)
  - coarsening (semi-coarsening in space/in time)

Parareal is a (Space-) Time Multigrid Method

Multigrid Full Approximation Scheme

Given: $M(\boldsymbol{u}) = \boldsymbol{b}$ (fine grid) and $M_H(\boldsymbol{U}) = \boldsymbol{b}_H$ (coarse grid)

$$\begin{cases} \tilde{\boldsymbol{u}}^k & = & S(\boldsymbol{u}^k, \boldsymbol{b}) \\ M_H(\boldsymbol{U}^{k+1}) & = & I_h^H(\boldsymbol{b} - M(\tilde{\boldsymbol{u}}^k)) + M_H(I_h^H \tilde{\boldsymbol{u}}^k) \\ \boldsymbol{u}^{k+1} & = & \tilde{\boldsymbol{u}}^k + I_H^h(\boldsymbol{U}^{k+1} - I_h^H \tilde{\boldsymbol{u}}^k) \end{cases}$$

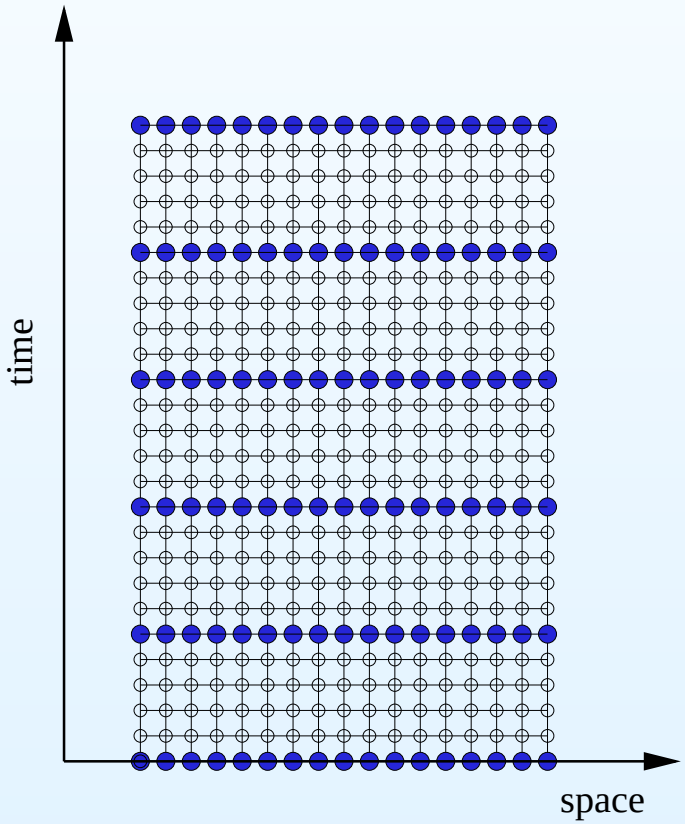
Theorem: Parareal is a two-level full approximation multigrid scheme with aggressive semi-coarsening in the time dimension and with $S(\cdot,\cdot)$: block Jacobi "F-smoother", I_h^H: injection, and I_H^h: trivial inclusion.

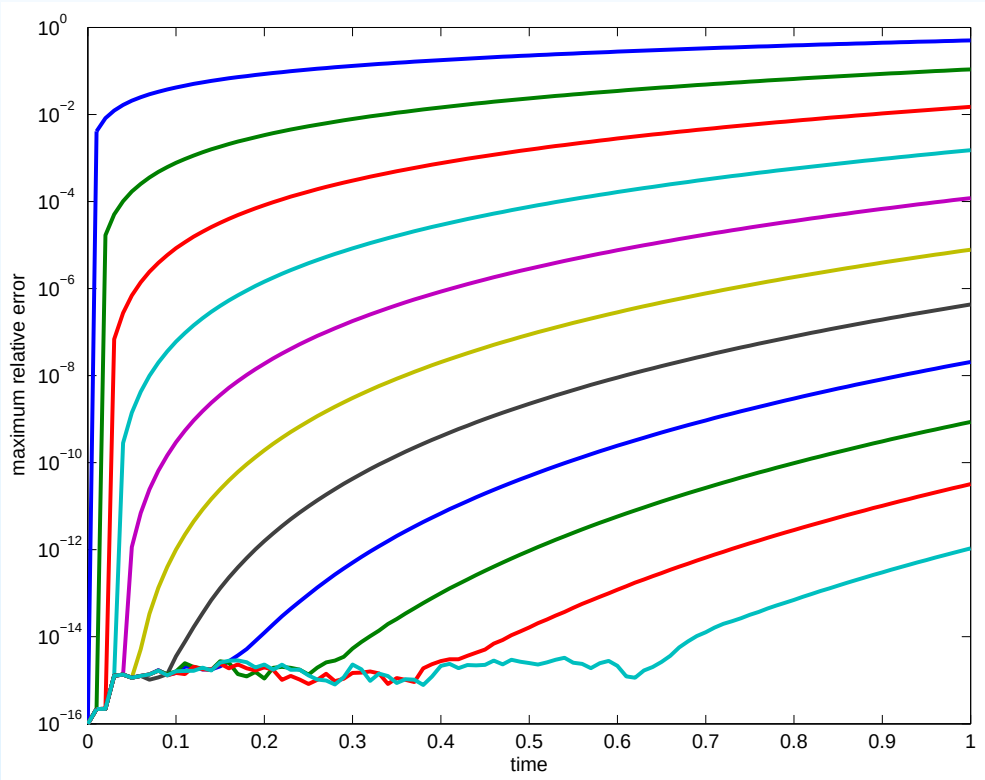
***Multilevel* extension** possible !

Parareal is a (Space-) Time Multigrid Method

An example: $u_t = u_{xx}, (x,t) \in [0,1] \times [0,1]$

(Implicit Euler, $\Delta T = 0.01$, $\Delta t = 0.001$ $\Delta x = 0.01$)

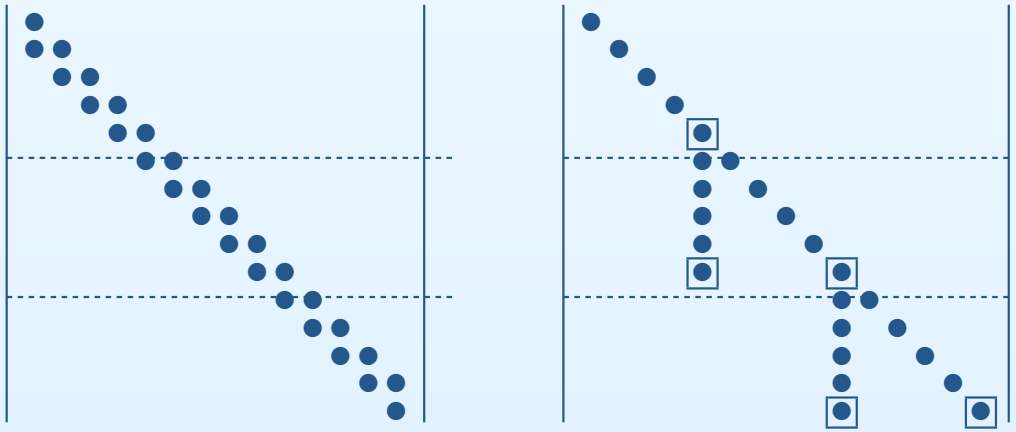




Another equivalence: *partition method of Wang*

- S. Chen and D. Kuck, "*Time and parallel processor bounds for linear recurrences*", IEEE Tr. Comp. 1975.
- H. Wang, ACM TOMS 1981.
- Collect $y_{n+1} = A_n y_n + b_n$ $(\mathbb{R}^d)$ $n = 0, 1, 2, \ldots$
- Solve bidiagonal (block) system in three steps

local elimination $\rightarrow$ global solve $\rightarrow$ local backsubstitution



Another equivalence: *partition method of Wang*

- For initial value problems: Boundary Value Method
  - Amodio and Brugnano (LAA 1992); Brugnano, Mazzia, and Trigiante (APNUM 1993), ...
- Coarse system (*Schur complement*)
  - ***small systems of ODEs***: direct solvers (cyclic reduction, recursive doubling, pipe-lined Gaussian elimination)
  - ***large (sparse) systems of ODEs***: Schur complement is dense and cannot be constructed
- Parareal is a partition method, with an iterative, preconditioned Schur complement solver.

4. Convergence Analysis

For the linear Dahlquist model problem

$$\boxed{u' = -au, \quad u(0) = u_0, \qquad \Re(a) \geq 0.}$$

Let $F(t_{n+1}, t_n, U_n^k)$ denote the exact solution at t_{n+1}.
Let $G(t_{n+1}, t_n, U_n^k) = \beta U_n^k$ be a one step method with ΔT such that it is in its region of absolute stability, i.e., $|\beta| \leq 1$.
Let the local truncation error be bounded by $C\Delta T^{p+1}$.

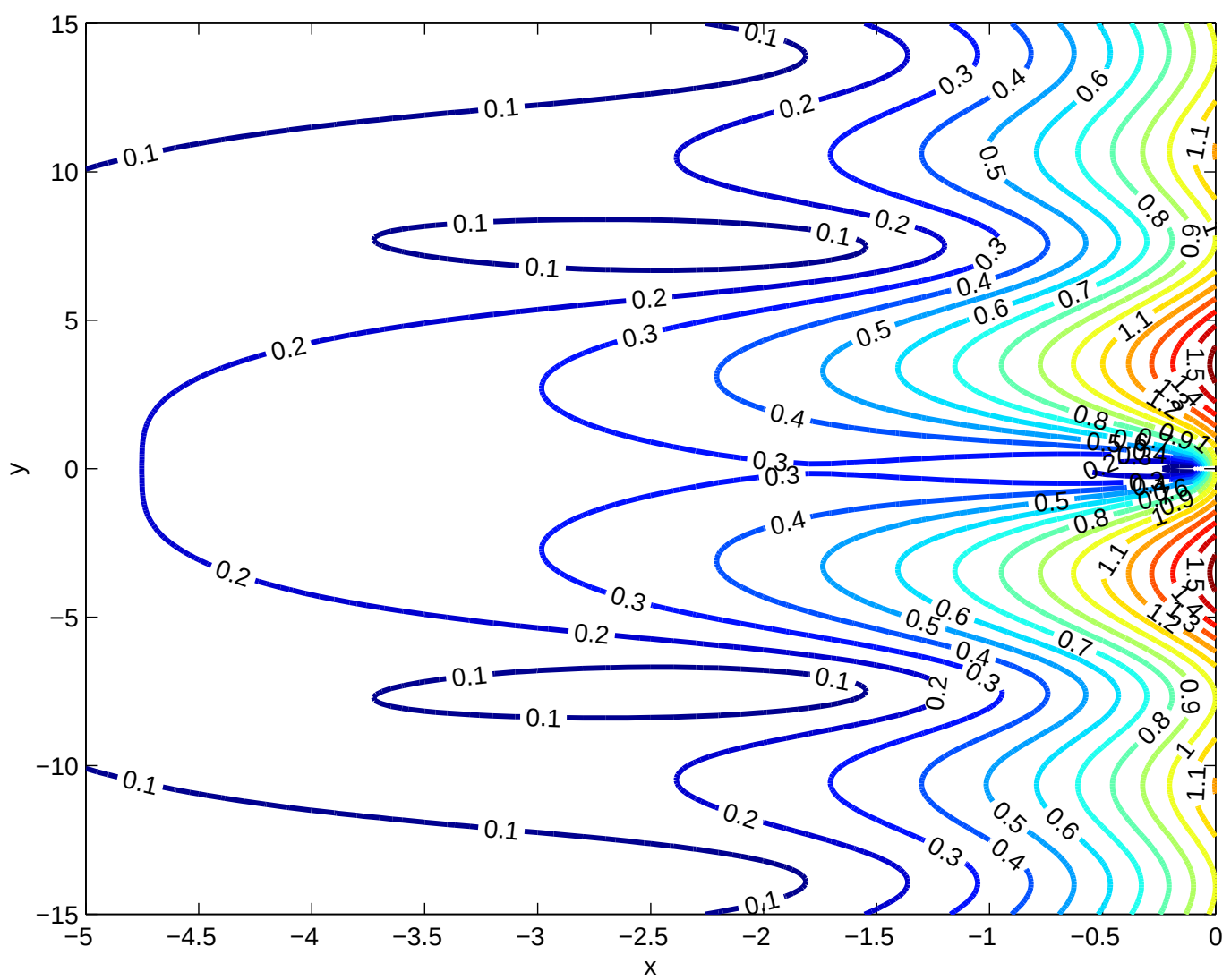
Theorem: Superlinear Convergence on Finite Time windows

$$\max_{1 \leq n \leq N} |u(t_n) - U_n^k| \leq \frac{(CT)^k}{k!} \Delta T^{pk} \max_{1 \leq n \leq N} |u(t_n) - U_n^0|.$$

Theorem: Linear Convergence on Infinite Time windows

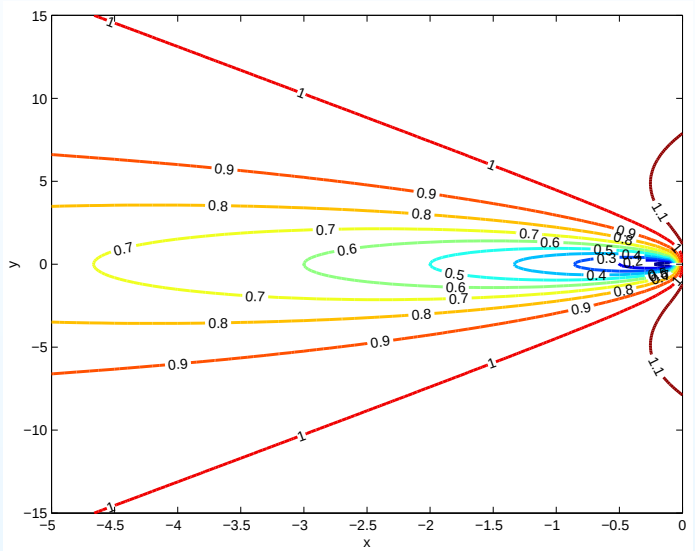
$$\sup_{n>0} |u(t_n) - U_n^k| \leq \left(\frac{C\Delta T^p}{\Re(a) + O(\Delta T)} \right)^k \sup_{n>0} |u(t_n) - U_n^0|.$$

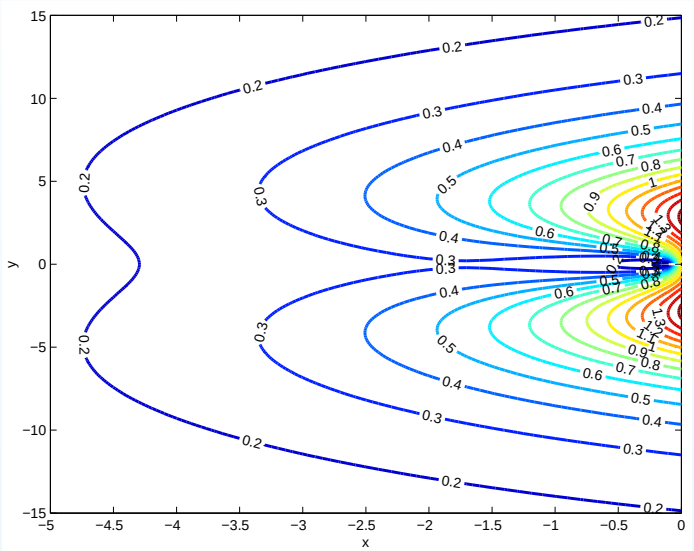
Convergence: $u' = -au$, $-a\Delta T \in \mathbb{C}^-$ ($G =$ B.Euler)

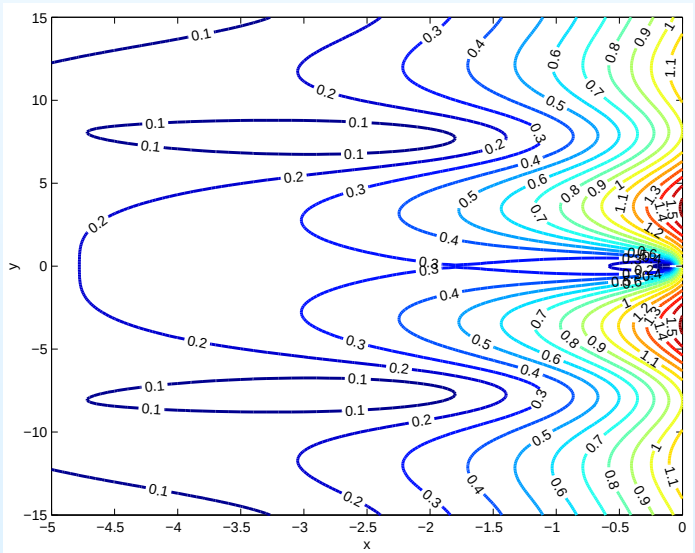


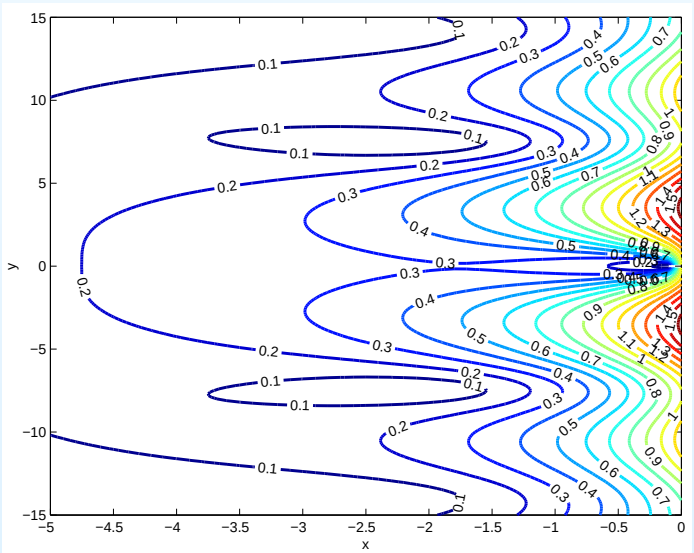
Convergence factor for $u' = -au$ (discrete F and G)

Trap(1)-BE, RadIIA(1)-BE, Trap(10)-BE, RadIIA(10)-BE









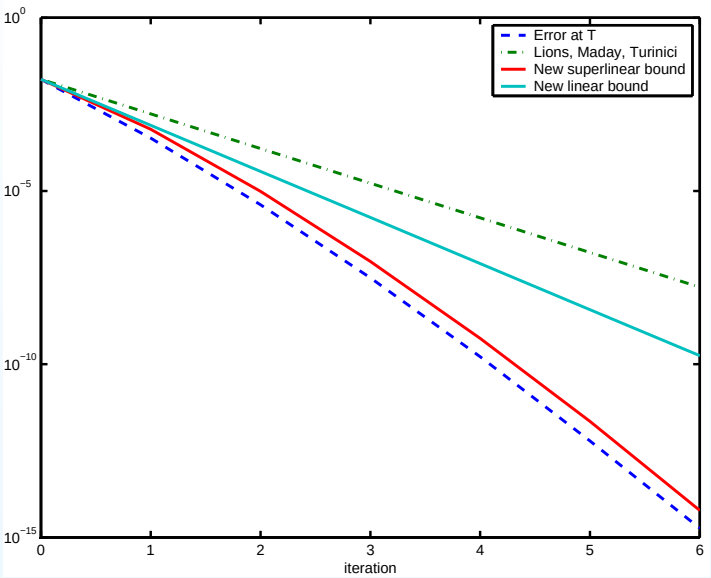
A numerical experiment: $u' = -u, u(0) = 1, t \in [0, T]$

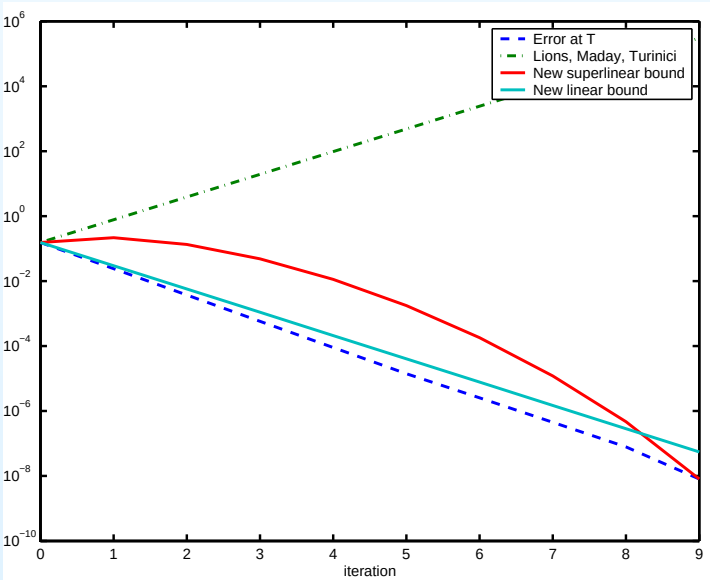
short window: $T = 1$

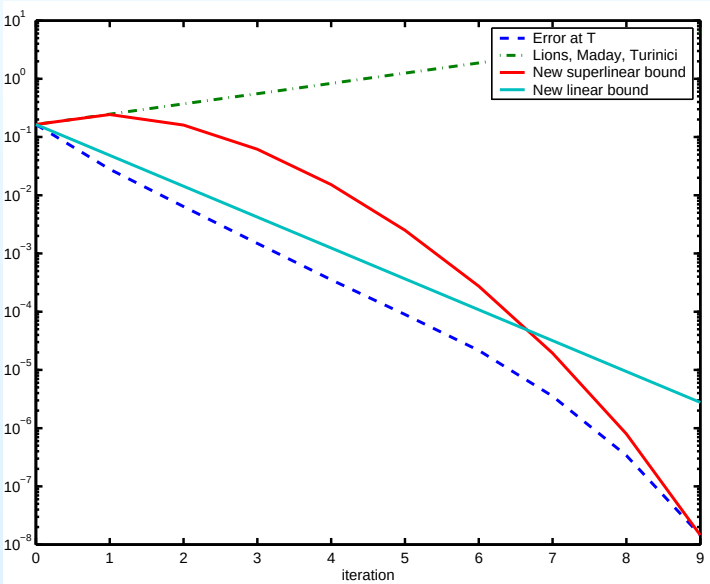
medium window: $T = 15$

long window: $T = 50$

with $\Delta T = T/10$
and $\Delta t = T/40$

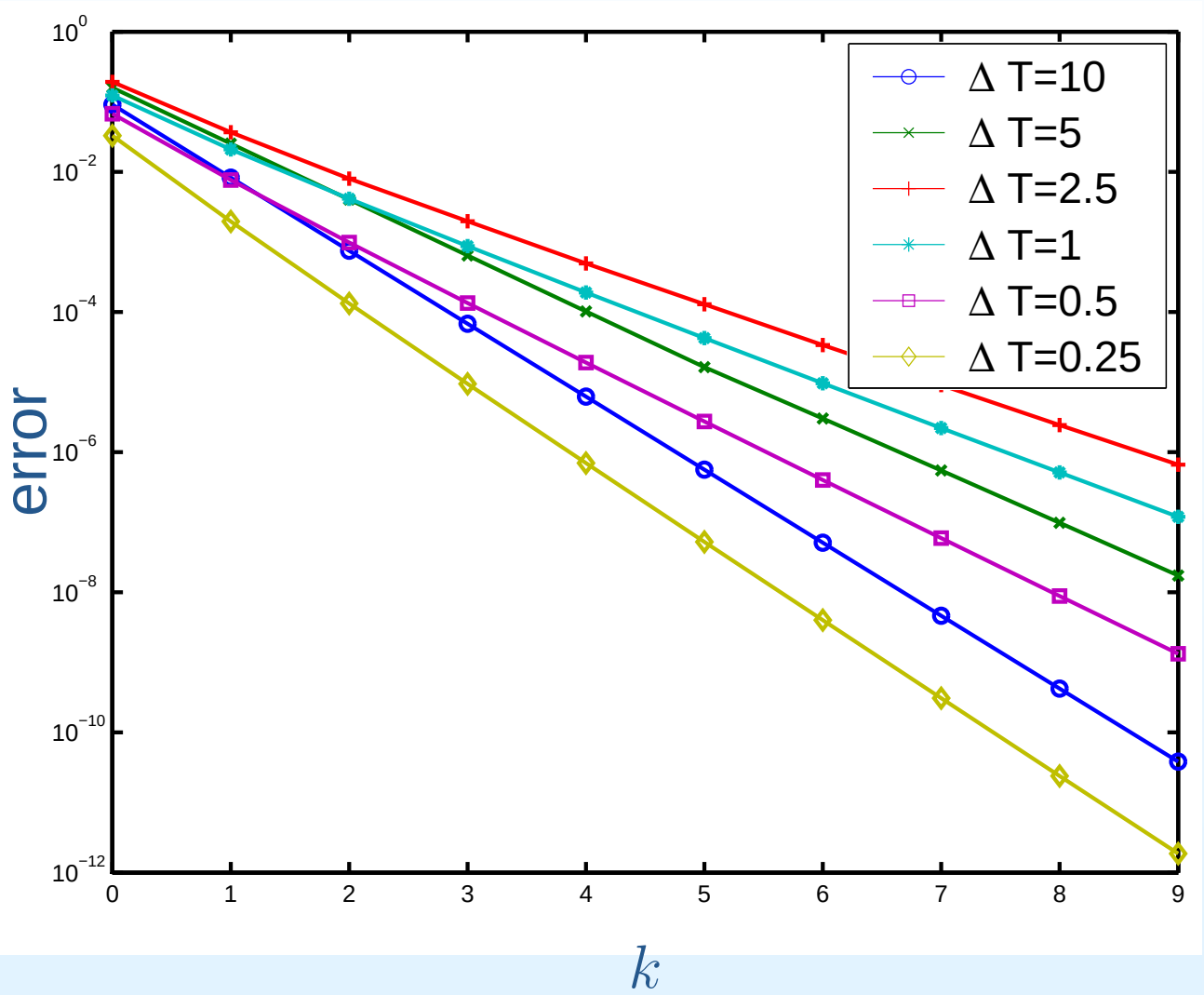






A numerical experiment: $u' = -u, u(0) = 1, t \in [0, T]$

Convergence as a function of ΔT, for fixed $\Delta t = 0.05$.



Convergence for the Heat Equation

Theorem: The parareal algorithm applied to the **heat equation** $\boxed{u_t = \Delta u}$ discretized with an L-stable method in time converges superlinearly on bounded time intervals,

$$\max_{1\leq n\leq N} ||u(t_n) - U_n^k||_2 \leq \frac{\gamma_s^k}{k!} \prod_{j=1}^{k} (N-j) \max_{1\leq n\leq N} ||u(t_n) - U_n^0||_2$$

The convergence is linear on on unbounded time intervals,

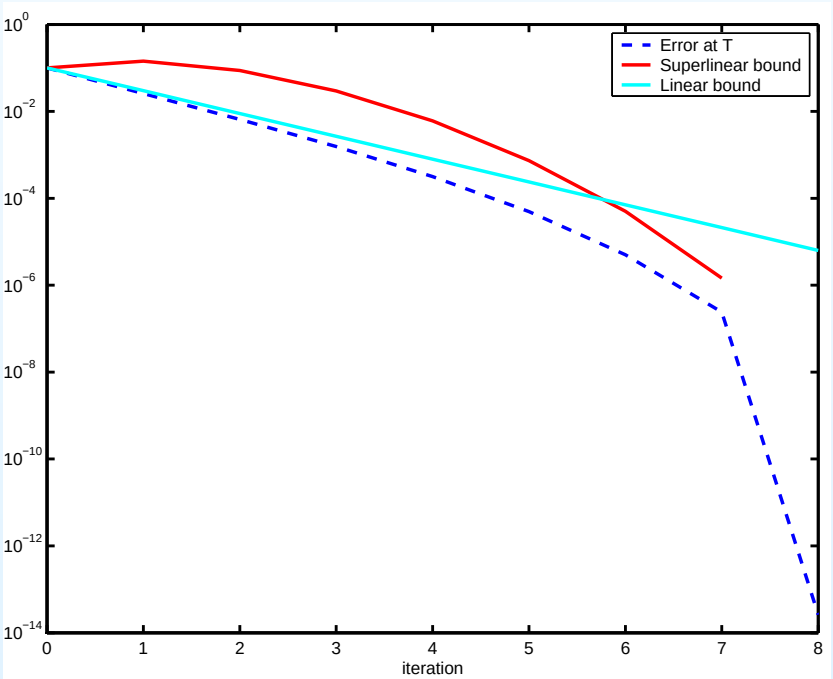
$$\sup_{n>0} ||u(t_n) - U_n^k||_2 \leq \gamma_l^k \sup_{n>0} ||u(t_n) - U_n^0||_2$$

method	order	γ_s	γ_l
BE	1	0.2036	0.2984
SDIRK	3	0.2073	0.1718
Radau IIA	5	0.0634	0.0677

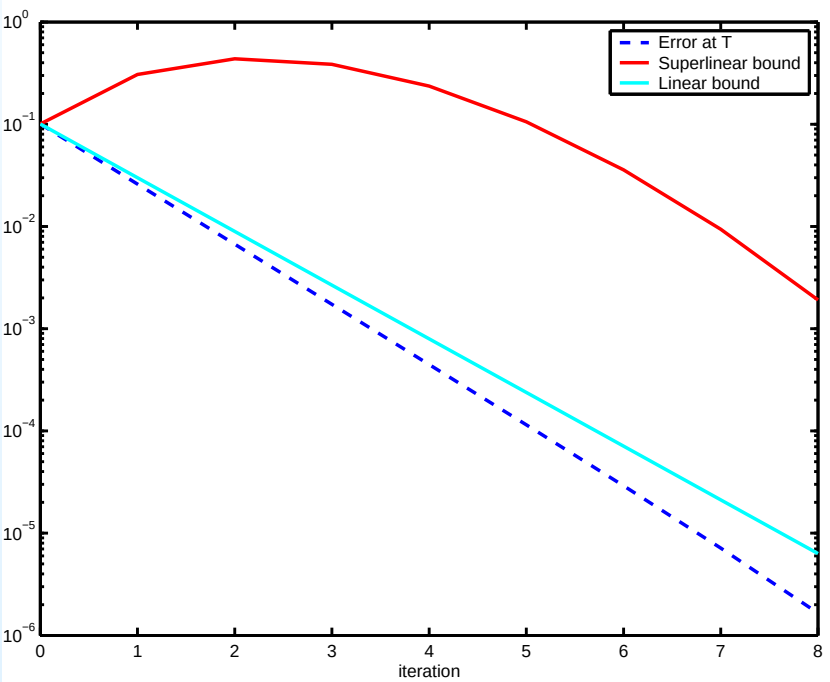
A numerical experiment: $u_t = u_{xx} + u_{yy}$ for $t \in [0, T]$

with $\Delta T = 0.5$, $\Delta t = 0.05$ and $\Delta x = \Delta y = 0.02$, $\Omega = [0, 1] \times [0, 1]$

short window: $T = 4$



medium window: $T = 8$



Convergence for pure Advection Problems

Theorem: The parareal algorithm applied to the **advection equation** $\boxed{u_t = u_x}$ with backward Euler in time converges superlinearly on bounded time intervals,

$$\max_{1\leq n\leq N} ||u(t_n) - U_n^k||_2 \leq \frac{\alpha_s^k}{k!} \prod_{j=1}^{k} (N-j) \max_{1\leq n\leq N} ||u(t_n) - U_n^0||_2,$$

where the constant $\alpha_s = 1.224$.

Remark:

- The superlinear convergence regime sets in only after many iterations.
- There is linear divergence for unbounded time intervals in the case of pure advection, with $\alpha_l = 1.632$.

5. Concluding Remarks

- Parareal can speed up a computation by exploiting time-parallelism. It is not effective as a serial algorithm.
- It is a special type of multiple shooting method or space-time multigrid method.
- It is also an iterative Schur complement solver, a deferred correction method, a predictor-corrector method, ...
- The convergence for linear ODE and PDE model problems
  - superlinear on bounded intervals
  - linear on unbounded intervals
- Implementation is straightforward (5 lines of code), given a fast (inaccurate) code and a slow (accurate) code.

Reference: Gander M. and Vandewalle S., Analysis of the Parareal Time-Parallel Time-Integration Method, SIAM J. Sci. Comp., Vol. 29(2), pp. 556-578, 2007.

5. Concluding Remarks

- Parareal can speed up a computation by exploiting time-parallelism. It is not effective as a serial algorithm.
- It is a special type of multiple shooting method or space-time multigrid method.
- It is also an iterative Schur complement solver, a deferred correction method, a predictor-corrector method, ...
- The convergence for linear ODE and PDE model problems
  - superlinear on bounded intervals
  - linear on unbounded intervals
- Implementation is straightforward (5 lines of code), given a fast (inaccurate) code and a slow (accurate) code.

Reference: Gander M. and Vandewalle S., Analysis of the Parareal Time-Parallel Time-Integration Method, SIAM J. Sci. Comp., Vol. 29(2), pp. 556-578, 2007.