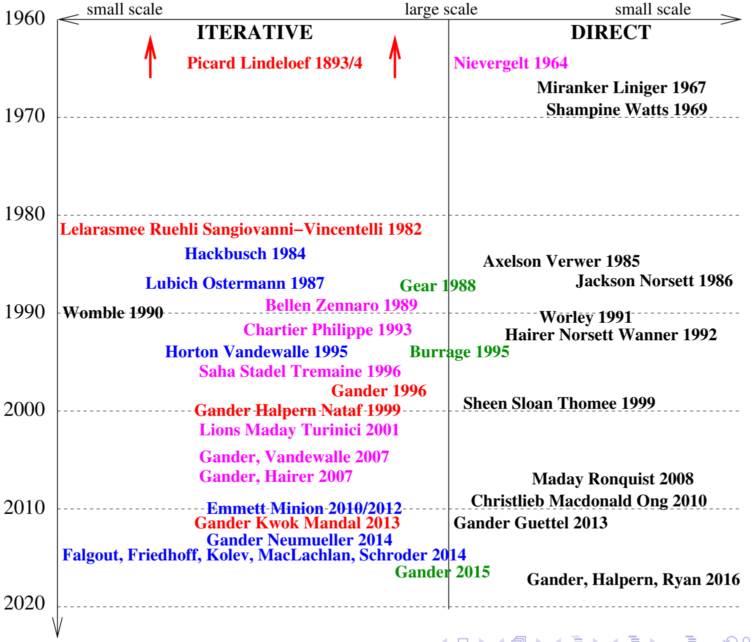
Time Parallel Time Integration
Chapter 2: Methods Based on Multiple Shooting

Martin J. Gander
martin.gander@unige.ch

University of Geneva

Michigan, August 2nd, 2022

Methods Based on Multiple Shooting



Origins of Shooting Methods

Herbert B. Keller (1968): Numerical Methods For Two-Point Boundary-Value Problems, 1968

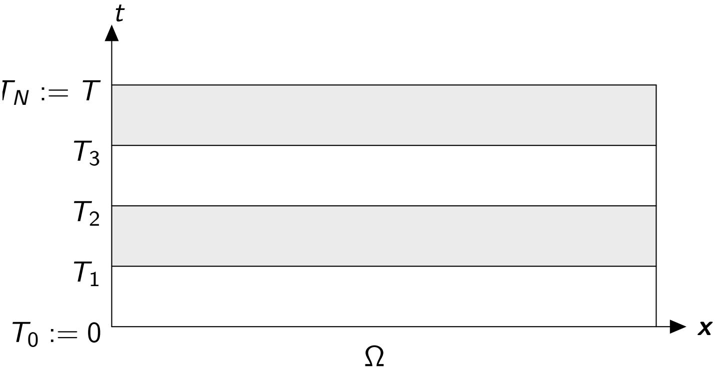
"Initial value methods are seldom advocated in the literature, but we find them extremely practical and theoretically powerful. A modification, called *parallel shooting*, is introduced to treat those 'unstable' cases (with rapidly growing solutions) for which ordinary shooting may be inadequate."

Alfredo Bellen and Marino Zennaro (1989): Parallel Algorithms for Initial-Value Problems for Difference and Differential Equations

"In addition to the two types of parallelism mentioned above, we wish to isolate a third which is analogous to what Gear has more recently called parallelism *across the time*. Here it is more appropriately called parallelism across the steps. In fact, the algorithm we propose is a realization of this kind of parallelism. Without discussing it in detail here, we want to point out that *the idea is indeed that of multiple shooting* and parallelism is introduced at the cost of redundancy of computation."

Decomposition for Shooting Methods in Time

The space-time domain is decomposed in the time direction:

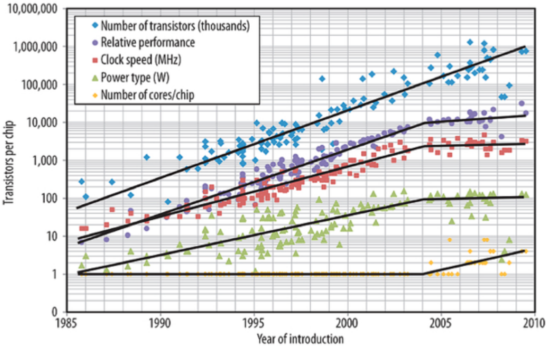


An iteration is then defined, which only uses solutions in the time subdomains, to obtain an approximate solution over the entire time interval $(0, T)$.

First Variant of such a Method

Jörg Nievergelt, (1964): Parallel methods for integrating ordinary differential equations

"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."

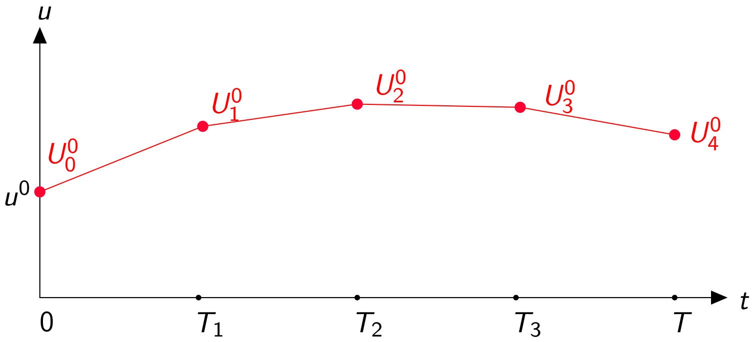


Nievergelts Method

$$\begin{array}{rcl} \partial_t u(t) & = & f(t, u(t)) \quad t \in (0, T], \\ u(0) & = & u^0. \end{array}$$

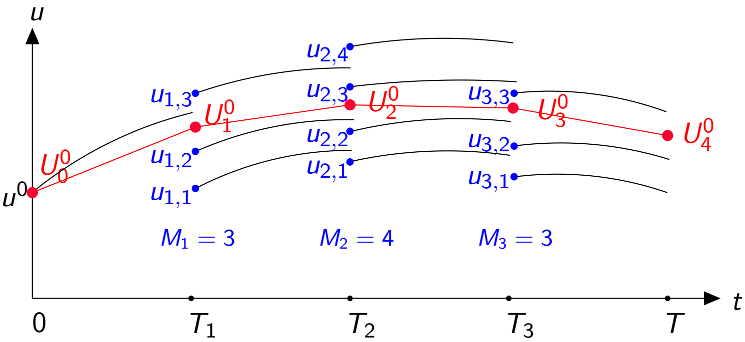
Partition $(0, T]$ into subintervals $(T_{n-1}, T_n]$,
$0 = T_0 < T_1 < T_2 < \ldots < T_N = T$, and then compute

1. A rough approximation U_n^0 (red) using e.g. FE



Nievergelts Method

2. For M_n points $u_{n,j}$ (blue) close to U_n^0, compute accurate trajectories $u_{n,m}(t)$ in parallel



3. Set $U_1^1 := u_{0,1}(T_1)$ and interpolate sequentially by
   - finding the index m such that $U_n^1 \in [u_{n,m}, u_{n,m+1}]$,
   - determining p such that $U_n^1 = pu_{n,m} + (1-p)u_{n,m+1}$,
   - setting the next interpolated value at T_{n+1} to $U_{n+1}^1 := pu_{n,m}(T_{n+1}) + (1-p)u_{n,m+1}(T_{n+1})$.

Result for Linear Problems

Theorem (Nievergelt's Method for Linear Problems)

For scalar linear ordinary differential equations,

$$\begin{aligned} \partial_t u(t) &= au(t) + f(t) \quad t \in (0, T], \\ u(0) &= u^0 \end{aligned}$$

the method of Nievergelt computes the exact solution, $U_n^1 = u(T_n)$ for $n = 1, 2, \ldots, N$, and it suffices to choose $M_n = 2$, i.e. only to compute two trajectories on each time interval to interpolate from.

Proof.

It suffices to prove that if $u_{n,1}(t)$ and $u_{n,2}(t)$ are two solutions on (T_n, T_{n+1}) with initial conditions $u_{n,1}(T_n) = \alpha$ and $u_{n,2}(T_n) = \beta$, then a third solution $v(t)$ with initial condition $v(T_n) = p\alpha + (1-p)\beta$ for any $p \in \mathbb{R}$ is the solution computed by Nievergelts method,

$$v(T_{n+1}) = pu_{n,1}(T_{n+1}) + (1-p)u_{n,2}(T_{n+1}). \qquad (*)$$

By linearity, the linear combination $pu_{n,1}(t) + (1-p)u_{n,2}(t)$ is also solution with initial condition

$$pu_{n,1}(T_n) + (1-p)u_{n,2}(T_n) = p\alpha + (1-p)\beta = v(T_n),$$

and hence by uniqueness we must have

$$v(t) = pu_{n,1}(t) + (1-p)u_{n,2}(t),$$

which implies (*). □

Further Properties

- In the non-linear case, there is additional error
- The method is inefficient compared to any serial method:

  > *"... parallelism is introduced at the expense of redundancy of computation ... it is believed that more general and improved versions of these methods will be of great importance when computers capable of executing many computations in parallel become available."*

- difficult to generalize to systems of ordinary differential equations: one would need many points in a cloud to be sure to cover the accurate trajectory
- The method of Nievergelt is in fact a direct method, but a natural precursor of multiple shooting in time.

Multiple Shooting in the Time Direction

Philippe Chartier and Bernard Philippe (1993): A Parallel Shooting Technique for Solving Dissipative ODEs

> *"We study different modifications of a class of parallel algorithms, initially designed by A. Bellen and M. Zennaro for difference equations and called 'across the steps' methods by their authors, for the purpose of solving initial value problems in ordinary differential equations on a massively parallel computer"*

The idea is to adapt a shooting method, developed for the solution of boundary value problems, to systems of initial value problems

$$\begin{aligned} \partial_t \boldsymbol{u}(t) &= \boldsymbol{f}(t, \boldsymbol{u}(t)) \quad t \in (0, T], \\ \boldsymbol{u}(0) &= \boldsymbol{u}^0, \end{aligned}$$

but there is no target to hit in the case of initial value problems.

Idea to use shooting in time

Introduce intermediate targets: as for Nievergelt, split the time interval $(0, T]$ into subintervals $(T_{n-1}, T_n]$, $n = 1, \ldots, N$, with $0 = T_0 < T_1 < T_2 < \ldots < T_N = T$, and solve the original problem on the subintervals,

$$\begin{array}{rcl} \partial_t \boldsymbol{u}_n(t) & = & \boldsymbol{f}(t, \boldsymbol{u}_n(t)) \quad t \in (T_n, T_{n+1}], \\ \boldsymbol{u}_n(T_n) & = & \boldsymbol{U}_n. \end{array}$$

This is however only possible if we know the starting point $\boldsymbol{U}_n$ of the solution for each interval, $\boldsymbol{U}_n = \boldsymbol{u}(T_n)$. These values are called shooting parameters, and they must satisfy the system of equations

$$\begin{array}{rcccl} \boldsymbol{U}_0 & = & \boldsymbol{u}_0(0) & = & \boldsymbol{u}^0, \\ \boldsymbol{U}_1 & = & \boldsymbol{u}_0(T_1) & = & \boldsymbol{u}_0(T_1, \boldsymbol{U}_0), \\ \boldsymbol{U}_2 & = & \boldsymbol{u}_1(T_2) & = & \boldsymbol{u}_1(T_2, \boldsymbol{U}_1), \\ & \vdots & & \vdots & \\ \boldsymbol{U}_N & = & \boldsymbol{u}_{N-1}(T_N) & = & \boldsymbol{u}_{N-1}(T_N, \boldsymbol{U}_{N-1}). \end{array}$$

The multiple shooting equations

Collecting the the shooting parameters $\boldsymbol{U}_n$ in the vector of vectors $\boldsymbol{U} := (\boldsymbol{U}_0, \boldsymbol{U}_1, \ldots, \boldsymbol{U}_N)^T$, one can determine their values by solving the non-linear system of equations

$$\boldsymbol{F}(\boldsymbol{U}) := \begin{pmatrix} \boldsymbol{U}_0 - u^0 \\ \boldsymbol{U}_1 - \boldsymbol{u}_0(T_1, \boldsymbol{U}_0) \\ \boldsymbol{U}_2 - \boldsymbol{u}_1(T_2, \boldsymbol{U}_1) \\ \vdots \\ \boldsymbol{U}_N - \boldsymbol{u}_{N-1}(T_N, \boldsymbol{U}_{N-1}) \end{pmatrix} = 0.$$

If we apply Newton's method to this system, like in the classical shooting method, to determine the shooting parameters, we start with an initial guess $\boldsymbol{U}^0$ and then compute for $k = 0, 1, 2, \ldots$

$$\boldsymbol{U}^{k+1} = \boldsymbol{U}^k - \left[\boldsymbol{F}'(\boldsymbol{U}^k)\right]^{-1} \boldsymbol{F}(\boldsymbol{U}^k)$$

Rewriting the Newton Iteration

The Jacobian is given by (I denotes the identity)

$$F'(U^k) = \begin{bmatrix} I & & & \\ -\frac{\partial u_0}{\partial U_0}(T_1, U_0^k) & I & & \\ & -\frac{\partial u_1}{\partial U_1}(T_2, U_1^k) & I & \\ & & \ddots & \ddots \end{bmatrix}$$

Multiplying the Newton iteration by the Jacobian matrix on both sides, we find the recurrence relation

$$\begin{aligned} U_0^{k+1} &= u^0, \\ U_1^{k+1} &= u_0(T_1, U_0^k) + \frac{\partial u_0}{\partial U_0}(T_1, U_0^k)(U_0^{k+1} - U_0^k), \\ U_2^{k+1} &= u_1(T_2, U_1^k) + \frac{\partial u_1}{\partial U_1}(T_2, U_1^k)(U_1^{k+1} - U_1^k), \quad \ldots \end{aligned}$$

or in more compact form for $n = 0, 1, 2, \ldots N-1$

$$\begin{aligned} U_0^{k+1} &= u^0, \\ U_{n+1}^{k+1} &= u_n(T_{n+1}, U_n^k) + \frac{\partial u_n}{\partial U_n}(T_{n+1}, U_n^k)(U_n^{k+1} - U_n^k) \end{aligned}$$

Computing the Jacobian Terms

As for classical shooting methods taking a derivative with respect to $\boldsymbol{U}_n$ of the system of ODEs to be solved on each time interval

$$\begin{aligned}\partial_t \boldsymbol{u}_n(t) &= \boldsymbol{f}(t, \boldsymbol{u}_n(t)) \quad t \in (T_n, T_{n+1}],\\ \boldsymbol{u}_n(T_n) &= \boldsymbol{U}_n,\end{aligned}$$

and denoting the derivative of the trajectory by $V_n(t) := \frac{\partial \boldsymbol{u}_n}{\partial \boldsymbol{U}_n}(t, \boldsymbol{U}_n)$, we obtain for $V_n(t)$ the linear system of ODEs

$$\begin{aligned}\partial_t V_n(t) &= \frac{\partial f}{\partial \boldsymbol{u}_n}(t, \boldsymbol{u}_n(t, \boldsymbol{U}_n))V_n(t) \quad t \in (T_n, T_{n+1}],\\ V_n(T_n) &= I,\end{aligned}$$

where I is the identity matrix.
One thus just has to compute solutions of these two coupled systems to use Newton's method.

Convergence Results

Theorem (Quadratic Convergence)

If the function $\boldsymbol{f}(t,\boldsymbol{u}(t))$ in the system of ODEs is twice continuously differentiable in its second argument, then the multiple shooting method converges locally quadratically, i.e.

$$||\boldsymbol{U}-\boldsymbol{U}^{k+1}|| \leq \frac{1}{2}||[\boldsymbol{F}'(\boldsymbol{U}^k)]^{-1}|| \, ||\boldsymbol{F}''(\boldsymbol{U}^k)|| \, ||\boldsymbol{U}-\boldsymbol{U}^k||^2 + O(||\boldsymbol{U}-\boldsymbol{U}^k||^3)$$

where $\boldsymbol{F}'(\boldsymbol{U})$ denotes the Jacobian of $\boldsymbol{F}(\boldsymbol{U})$ and $\boldsymbol{F}''(\boldsymbol{U})$ the bilinear map representing the second derivative of $\boldsymbol{F}(\boldsymbol{U})$.

Proof. We expand $\boldsymbol{F}(\boldsymbol{U})$ around the iteration $\boldsymbol{U}^k$,

$$\begin{aligned}\boldsymbol{F}(\boldsymbol{U}) &= \boldsymbol{F}(\boldsymbol{U}^k) + \boldsymbol{F}'(\boldsymbol{U}^k)(\boldsymbol{U}-\boldsymbol{U}^k) \\ &\quad + \frac{1}{2}\boldsymbol{F}''(\boldsymbol{U}^k)(\boldsymbol{U}-\boldsymbol{U}^k, \boldsymbol{U}-\boldsymbol{U}^k) + O(||\boldsymbol{U}-\boldsymbol{U}^k||^3)\end{aligned}$$

From Newton's method, we have

$$0 = \boldsymbol{F}(\boldsymbol{U}^k) + \boldsymbol{F}'(\boldsymbol{U}^k)(\boldsymbol{U}^{k+1}-\boldsymbol{U}^k)$$

Proof continued

If $\boldsymbol{U}$ is the solution, we have $\boldsymbol{F}(\boldsymbol{U}) = 0$, and the difference gives

$$\begin{aligned}
0 &= \boldsymbol{F}'(\boldsymbol{U}^k)(\boldsymbol{U}-\boldsymbol{U}^k) - \boldsymbol{F}'(\boldsymbol{U}^k)(\boldsymbol{U}^{k+1}-\boldsymbol{U}^k) \\
&\quad + \frac{1}{2}\boldsymbol{F}''(\boldsymbol{U}^k)(\boldsymbol{U}-\boldsymbol{U}^k, \boldsymbol{U}-\boldsymbol{U}^k) \\
&\quad + O(||\boldsymbol{U}-\boldsymbol{U}^k||^3) \\
&= \boldsymbol{F}'(\boldsymbol{U}^k)(\boldsymbol{U}-\boldsymbol{U}^{k+1}) + \frac{1}{2}\boldsymbol{F}''(\boldsymbol{U}^k)(\boldsymbol{U}-\boldsymbol{U}^k, \boldsymbol{U}-\boldsymbol{U}^k) \\
&\quad + O(||\boldsymbol{U}-\boldsymbol{U}^k||^3).
\end{aligned}$$

Solving for $(\boldsymbol{U}-\boldsymbol{U}^{k+1})$ and taking norms then leads to the quadratic convergence estimate.

A new result in time

Theorem (Finite Step Convergence)

If $\boldsymbol{U}_0^0 = \boldsymbol{u}^0$, then the multiple shooting method has the property that

$$\boldsymbol{U}_n^k = \boldsymbol{u}(T_n) \quad \text{if } k \geq n,$$

i.e. $\boldsymbol{U}_n^k$ coincides with the exact solution from iteration index $k = n$ onward.

Proof. The proof is by induction in the time direction: for $n = 0$, we have by assumption $\boldsymbol{U}_0^0 = \boldsymbol{u}^0$, and $\boldsymbol{U}_0^{k+1} = \boldsymbol{u}^0$ for $k = 0, 1, 2, \ldots$ by the multiple shooting method, so the initial value is always exact. So suppose that $\boldsymbol{U}_n^k = \boldsymbol{u}(T_n)$ for $k \geq n$.

For $n+1$, we have from the multiple shooting method that

$$\boldsymbol{U}_{n+1}^{k+1} = \boldsymbol{u}_n(T_{n+1}, \boldsymbol{U}_n^k) + \frac{\partial \boldsymbol{u}_n}{\partial \boldsymbol{U}_n}(T_{n+1}, \boldsymbol{U}_n^k)(\boldsymbol{U}_n^{k+1} - \boldsymbol{U}_n^k).$$

Proof continued

Now to show that $\boldsymbol{U}_{n+1}^{k+1} = \boldsymbol{u}(T_{n+1})$ for $k+1 \geq n+1$, we note that

$$k+1 \geq n+1 \quad \Longrightarrow \quad k \geq n,$$

and hence we have already by the induction hypothesis that

$$\boldsymbol{U}_n^k = \boldsymbol{u}(T_n), \quad \text{and also} \quad \boldsymbol{U}_n^{k+1} = \boldsymbol{u}(T_n),$$

and introducing this into

$$\boldsymbol{U}_{n+1}^{k+1} = \boldsymbol{u}_n(T_{n+1}, \boldsymbol{U}_n^k) + \frac{\partial \boldsymbol{u}_n}{\partial \boldsymbol{U}_n}(T_{n+1}, \boldsymbol{U}_n^k)(\boldsymbol{U}_n^{k+1} - \boldsymbol{U}_n^k),$$

the second term cancels because of the difference, and we obtain

$$\boldsymbol{U}_{n+1}^{k+1} = \boldsymbol{u}_n(t_{n+1}, \boldsymbol{u}(T_n)) = \boldsymbol{u}(T_{n+1}),$$

which concludes the proof.

But the computation of the Jacobian terms becomes prohibitive for large systems!

Origins of the parareal algorithm

Prasenjit Saha, Joachim Stadel and Scott Tremaine (1997): A parallel integration method for solar system dynamics

> *"We describe how long-term solar system orbit integration could be implemented on a parallel computer. The interesting feature of our algorithm is that* each processor is assigned not to a planet or a pair of planets but to a time-interval*. Thus, the 1st week, 2nd week, . . . , 1000th week of an orbit are computed concurrently. The problem of matching the input to the $(n+1)$-st processor with the output of the n-th processor can be solved efficiently by an iterative procedure. Our work is related to the so-called* **waveform relaxation methods** *. . . "*

They cite Bellen and Zennaro and Nievergelt as sources of inspiration, and waveform relaxation!

The parareal algorithm

Jacques-Louis Lions, Yvon Maday and Gabriel Turinici (2001): Résolution d'EDP par un schéma en temps "pararéel"

"On propose dans cette Note un schéma permettant de profiter d'une architecture parallèle pour la discrétisation en temps d'une équation d'évolution aux dérivées partielles. Cette méthode, basée sur un schéma d'Euler, *combine des résolutions grossières et des résolutions fines et indépendantes en temps en s'inspirant de ce qui est classique en espace. La parallélisation qui en résulte se fait dans la direction temporelle ce qui est en revanche non classique*. Elle a pour principale motivation les problèmes en temps réel, d'où la terminologie proposée de '**pararéel**"

The parareal algorithm is a shooting method
with a coarse Jacobian approximation !

Working with Stefan Vandewalle, November 7, 2002.

Historical notes

- Saha, Stadel and Tremaine introduce a key new feature in the multiple shooting algorithm, namely to approximate the Jacobian terms by certain differences, computed for a simpler model of the planetary system, including only the interaction with the sun
- Lions, Maday and Turinici, independently of previous work, except for the work of Chartier and Philippe, discovered the same algorithm, coming however from a virtual control approach.
- They explain their algorithm on the Dahlquist equation ("Pour commencer, on expose l'idée sur l'exemple simple") using Backward Euler.
- They then generalize their results to the heat equation, and to a semi-linear advection diffusion problem, where a variant of the algorithm is proposed by linearization about the previous iterate

The Parareal Algorithm

For solving the non-linear problem

$$\begin{aligned} \partial_t \boldsymbol{u}(t) &= \boldsymbol{f}(t, \boldsymbol{u}(t)) \quad t \in (0, T], \\ \boldsymbol{u}(0) &= \boldsymbol{u}^0. \end{aligned}$$

parareal needs two propagation operators:

1. $\boldsymbol{G}(t_2, t_1, \boldsymbol{u}_1)$ is a coarse approximation[1] to the solution $\boldsymbol{u}(t_2)$ with initial condition $\boldsymbol{u}(t_1) = \boldsymbol{u}_1$,
2. $\boldsymbol{F}(t_2, t_1, \boldsymbol{u}_1)$ is a more accurate approximation[2] of the solution $u(t_2)$ with initial condition $\boldsymbol{u}(t_1) = \boldsymbol{u}_1$.

As for Nievergelt, $(0, T]$ is partitioned into subintervals $(T_{n-1}, T_n]$. Parareal then starts with an initial coarse approximation $\boldsymbol{U}_n^0$ at $T_0, T_1, \ldots, T_N$, and then computes for $k = 0, 1, \ldots$

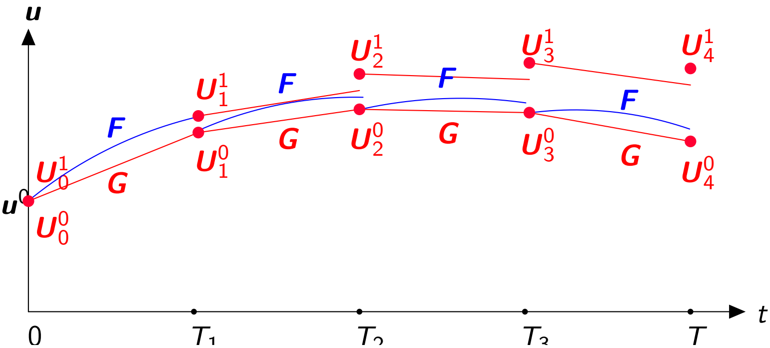
$$\begin{aligned} &\boldsymbol{U}_0^{k+1} := \boldsymbol{u}^0, \\ &\boldsymbol{U}_{n+1}^{k+1} := \boldsymbol{F}(T_{n+1}, T_n, \boldsymbol{U}_n^k) + \boldsymbol{G}(T_{n+1}, T_n, \boldsymbol{U}_n^{k+1}) - \boldsymbol{G}(T_{n+1}, T_n, \boldsymbol{U}_n^k) \end{aligned}$$

[1] $\boldsymbol{G}$ stands for 'grossier', which means coarse in French.

[2] $\boldsymbol{F}$ stands for 'fine'

Illustration of Parareal

$$U_0^{k+1} := u^0,$$
$$U_{n+1}^{k+1} := F(T_{n+1}, T_n, U_n^k) + G(T_{n+1}, T_n, U_n^{k+1}) - G(T_{n+1}, T_n, U_n^k)$$



Initial guess of parareal computed with the coarse propagator G in red, and first parallel fine solutions computed with the fine propagator F in blue.

Finite Step Convergence of Parareal

Theorem (Finite Step Convergence)

Like for multiple shooting, the parareal algorithm has the property that

$$\boldsymbol{U}_n^k = \boldsymbol{F}(T_n, 0, \boldsymbol{u}^0) \quad \textit{if } k \geq n,$$

i.e. $\boldsymbol{U}_n^k$ coincides with the fine approximation from iteration index $k = n$ onward.

Proof. by induction in the time direction: for $n = 0$, we have by

$$\boldsymbol{U}_0^{k+1} := \boldsymbol{u}^0$$

that the initial value is always the same as the one used for the fine solution. So suppose that $\boldsymbol{U}_n^k = \boldsymbol{F}(T_n, 0, \boldsymbol{u}^0)$ for $k \geq n$. For $n + 1$, we have from the parareal algorithm that

$$\boldsymbol{U}_{n+1}^{k+1} = \boldsymbol{F}(T_{n+1}, T_n, \boldsymbol{U}_n^k) + \boldsymbol{G}(T_{n+1}, T_n, \boldsymbol{U}_n^{k+1}) - \boldsymbol{G}(T_{n+1}, T_n, \boldsymbol{U}_n^k).$$

Proof continued

Now to show that

$$\boldsymbol{U}_{n+1}^{k+1} = \boldsymbol{F}(T_{n+1}, 0, \boldsymbol{u}^0)$$

for $k+1 \geq n+1$, since

$$k+1 \geq n+1 \quad \Longrightarrow \quad k \geq n,$$

we have by the induction hypothesis that

$$\boldsymbol{U}_n^{k+1} = \boldsymbol{U}_n^k = \boldsymbol{F}(T_n, 0, \boldsymbol{u}^0),$$

and introducing this into the parareal formula, the second and third term cancel, and we obtain

$$\boldsymbol{U}_{n+1}^{k+1} = \boldsymbol{F}(T_{n+1}, T_n, \boldsymbol{F}(T_n, 0, \boldsymbol{u}^0)) = \boldsymbol{F}(T_{n+1}, 0, \boldsymbol{u}^0),$$

which concludes the proof.

Note that convergence after N iterations is too late to make the algorithm useful, no speedup is possible then!

Parareal Relation to Multiple Shooting

Theorem (Parareal is an Approximate Multiple Shooting Method)

The parareal algorithm with an exact fine solver is a multiple shooting method with an approximation of the derivative terms stemming from the Jacobian in Newton's method by a difference of trajectories computed on a coarser grid.

Proof. Recall the multiple shooting method

$$\boldsymbol{U}_{n+1}^{k+1} = \boldsymbol{u}_n(T_{n+1}, \boldsymbol{U}_n^k) + \frac{\partial \boldsymbol{u}_n}{\partial \boldsymbol{U}_n}(T_{n+1}, \boldsymbol{U}_n^k)(\boldsymbol{U}_n^{k+1} - \boldsymbol{U}_n^k).$$

A Taylor series of $\boldsymbol{u}_n(T_{n+1}, \boldsymbol{U}_n^{k+1})$ about $\boldsymbol{U}_n^k$,

$$\boldsymbol{u}_n(T_{n+1}, \boldsymbol{U}_n^{k+1}) = \boldsymbol{u}_n(T_{n+1}, \boldsymbol{U}_n^k) + \frac{\partial \boldsymbol{u}_n}{\partial \boldsymbol{U}_n}(T_{n+1}, \boldsymbol{U}_n^k)(\boldsymbol{U}_n^{k+1} - \boldsymbol{U}_n^k) + O(||\boldsymbol{U}_n^{k+1} - \boldsymbol{U}_n^k||^2),$$

implies that

$$\frac{\partial \boldsymbol{u}_n}{\partial \boldsymbol{U}_n}(T_{n+1}, \boldsymbol{U}_n^k)(\boldsymbol{U}_n^{k+1} - \boldsymbol{U}_n^k) \approx \boldsymbol{u}_n(T_{n+1}, \boldsymbol{U}_n^{k+1}) - \boldsymbol{u}_n(T_{n+1}, \boldsymbol{U}_n^k)$$

Approximate Shooting Method

Approximating the Jacobian term by the difference gives

$$\tilde{\boldsymbol{U}}_{n+1}^{k+1} = \boldsymbol{u}_n(T_{n+1}, \tilde{\boldsymbol{U}}_n^k) + \boldsymbol{u}_n(T_{n+1}, \tilde{\boldsymbol{U}}_n^{k+1}) - \boldsymbol{u}_n(T_{n+1}, \tilde{\boldsymbol{U}}_n^k).$$

This method is not of much interest, since the first and last term on the right hand side cancel, and one thus sequentially integrates the problem using the middle term.

The parareal algorithm remedies this by replacing the last two terms on the right hand side using a coarse approximation,

$$\boldsymbol{U}_{n+1}^{k+1} := \boldsymbol{F}(T_{n+1}, T_n, \boldsymbol{U}_n^k) + \boldsymbol{G}(T_{n+1}, T_n, \boldsymbol{U}_n^{k+1}) - \boldsymbol{G}(T_{n+1}, T_n, \boldsymbol{U}_n^k)$$

Parareal is thus a multiple shooting method with an approximation of the Jacobian by a difference computed on a coarser mesh (or model).

Superlinear Convergence Estimate for Parareal

Theorem (Parareal Convergence Estimate)

Let $\boldsymbol{F}$ denote the exact solution and let $\boldsymbol{G}$ be an approximate solution of order p with local truncation error bounded by $C_3\Delta T^{p+1}$. If

$$\|\boldsymbol{G}(t+\Delta T,t,\boldsymbol{v})-\boldsymbol{G}(t+\Delta T,t,\boldsymbol{w})\|\leq(1+C_2\Delta T)\|\boldsymbol{v}-\boldsymbol{w}\|,$$

$$\boldsymbol{F}(T_n,T_{n-1},x)-\boldsymbol{G}(T_n,T_{n-1},x)=c_{p+1}(x)\Delta T^{p+1}+c_{p+2}(x)\Delta T^{p+2}+\ldots$$

where the c_j, $j=p+1,p+2,\ldots$ are continuously differentiable, then the parareal algorithm satisfies

$$\begin{aligned}\|\boldsymbol{u}(T_n)-\boldsymbol{U}_n^k\| &\leq \frac{C_3}{C_1}\frac{(C_1\Delta T^{p+1})^{k+1}}{(k+1)!}(1+C_2\Delta T)^{n-k-1}\prod_{\ell=0}^{k}(n-\ell)\\ &\leq \frac{C_3}{C_1}\frac{(C_1T_n)^{k+1}}{(k+1)!}e^{C_2(T_n-T_{k+1})}\Delta T^{p(k+1)},\end{aligned}$$

where C_1 is related to the c_j, see the proof.

Proof.

Since $\boldsymbol{F}$ represents the exact solution, we have the identity

$$\boldsymbol{u}(T_{n+1}) = \boldsymbol{F}(T_{n+1}, T_n, \boldsymbol{u}(T_n)),$$

and subtracting from this the parareal algorithm

$$\boldsymbol{U}_{n+1}^{k+1} := \boldsymbol{F}(T_{n+1}, T_n, \boldsymbol{U}_n^k) + \boldsymbol{G}(T_{n+1}, T_n, \boldsymbol{U}_n^{k+1}) - \boldsymbol{G}(T_{n+1}, T_n, \boldsymbol{U}_n^k)$$

we obtain when adding and subtracting $\boldsymbol{G}(T_{n+1}, T_n, \boldsymbol{u}(T_n))$

$$\begin{aligned}\boldsymbol{u}(T_{n+1}) - \boldsymbol{U}_{n+1}^{k+1} &= \boldsymbol{F}(T_{n+1}, T_n, \boldsymbol{u}(T_n)) - \boldsymbol{G}(T_{n+1}, T_n, \boldsymbol{u}(T_n)) \\ &\quad - \left(\boldsymbol{F}(T_{n+1}, T_n, \boldsymbol{U}_n^k) - \boldsymbol{G}(T_{n+1}, T_n, \boldsymbol{U}_n^k)\right) \\ &\quad + \boldsymbol{G}(T_{n+1}, T_n, \boldsymbol{u}(T_n)) - \boldsymbol{G}(T_{n+1}, T_n, \boldsymbol{U}_n^{k+1}).\end{aligned}$$

Using the expansion for the first two lines, we obtain

$$\begin{aligned}\boldsymbol{u}(T_{n+1}) - \boldsymbol{U}_{n+1}^{k+1} &= c_{p+1}(\boldsymbol{u}(T_n))\Delta T^{p+1} + c_{p+2}(\boldsymbol{u}(T_n))\Delta T^{p+2} + \ldots \\ &\quad - \left(c_{p+1}(\boldsymbol{U}_n^k)\Delta T^{p+1} + c_{p+2}(\boldsymbol{U}_n^k)\Delta T^{p+2} + \ldots\right) \\ &\quad + \boldsymbol{G}(T_{n+1}, T_n, \boldsymbol{u}(T_n)) - \boldsymbol{G}(T_{n+1}, T_n, \boldsymbol{U}_n^{k+1}).\end{aligned}$$

Proof continued

Since the c_j, $j = p+1, p+2, \ldots$ are continuously differentiable, there exists a constant C_1 such that

$$\begin{aligned}
||(c_{p+1}(\boldsymbol{u}(T_n)) - c_{p+1}(\boldsymbol{U}_n^k))\Delta T^{p+1} \\
+ (c_{p+2}(\boldsymbol{u}(T_n)) - c_{p+2}(\boldsymbol{U}_n^k))\Delta T^{p+2} + \ldots|| \\
\leq ||(c_{p+1}(\boldsymbol{u}(T_n)) - c_{p+1}(\boldsymbol{U}_n^k))||\Delta T^{p+1} \\
+ ||(c_{p+2}(\boldsymbol{u}(T_n)) - c_{p+2}(\boldsymbol{U}_n^k))||\Delta T^{p+2} + \ldots \\
\leq C_1 \Delta T^{p+1} ||\boldsymbol{u}(T_n) - \boldsymbol{U}_n^k||.
\end{aligned}$$

We can thus take norms in the equation on the previous slide, and using the first (Lipschitz) condition on $\boldsymbol{G}$ yields

$$\|\boldsymbol{u}(T_{n+1}) - \boldsymbol{U}_{n+1}^{k+1}\| \leq C_1 \Delta T^{p+1} \|\boldsymbol{u}(T_n) - \boldsymbol{U}_n^k\| + (1 + C_2 \Delta T)\|\boldsymbol{u}(T_n) - \boldsymbol{U}_n^{k+1}\|$$

To bound $\|\boldsymbol{u}(T_{n+1}) - \boldsymbol{U}_{n+1}^{k+1}\|$, we study

$$e_{n+1}^{k+1} = \alpha e_n^k + \beta e_n^{k+1}, \quad e_{n+1}^0 = \gamma + \beta e_n^0,$$

where we set $\alpha := C_1 \Delta T^{p+1}$, $\beta := 1 + C_2 \Delta T$ and $\gamma := C_3 \Delta T^{p+1}$.

Proof continued

The initialization is obtained from the initial guess for Parareal obtained by $\boldsymbol{G}$ using the Lipschitz condition,

$$\begin{aligned}
&||\boldsymbol{u}(T_{n+1}) - \boldsymbol{U}^0_{n+1}|| = ||\boldsymbol{u}(T_{n+1}) - \boldsymbol{G}(T_{n+1}, T_n, \boldsymbol{U}^0_n)|| \\
&\quad = ||\boldsymbol{u}(T_{n+1}) - \boldsymbol{G}(T_{n+1}, T_n, \boldsymbol{u}(T_n)) \\
&\qquad + \boldsymbol{G}(T_{n+1}, T_n, \boldsymbol{u}(T_n)) - \boldsymbol{G}(T_{n+1}, T_n, \boldsymbol{U}^0_n)|| \\
&\quad \leq C_3 \Delta T^{p+1} + (1 + C_2 \Delta T)||\boldsymbol{u}(T_n) - \boldsymbol{U}^0_n||.
\end{aligned}$$

Generating functions: multiplying the recurrence by ζ^{n+1} and summing over n, we find for the first recurrence on the left

$$\begin{aligned}
\sum_{n=0}^{\infty} e^{k+1}_{n+1} \zeta^{n+1} &= \alpha \sum_{n=0}^{\infty} e^k_n \zeta^{n+1} + \beta \sum_{n=0}^{\infty} e^{k+1}_n \zeta^{n+1} \\
&= \alpha\zeta \sum_{n=0}^{\infty} e^k_n \zeta^n + \beta\zeta \sum_{n=0}^{\infty} e^{k+1}_n \zeta^n.
\end{aligned}$$

Now since $e^k_0 = ||\boldsymbol{u}(0) - \boldsymbol{U}^k_0|| = 0$, we can start summing at $n = 1$ on the right, to get the same sum as on the left.

Proof continued

The **generating function** $\rho_k(\zeta) := \sum_{n=1}^{\infty} e_n^k \zeta^n$ thus satisfies the recurrence relation

$$\rho_{k+1}(\zeta) = \alpha\zeta\rho_k(\zeta) + \beta\zeta\rho_{k+1}(\zeta).$$

This recurrence relation in k can now easily be solved,

$$\rho_{k+1}(\zeta) = \frac{\alpha\zeta}{1-\beta\zeta}\rho_k(\zeta) = \frac{\alpha^{k+1}\zeta^{k+1}}{(1-\beta\zeta)^{k+1}}\rho_0(\zeta).$$

To find $\rho_0(\zeta)$, we multiply also the initialization by ζ^{n+1} and sum in n,

$$\sum_{n=0}^{\infty} e_{n+1}^0 \zeta^{n+1} = \gamma\zeta\sum_{n=0}^{\infty}\zeta^n + \beta\zeta\sum_{n=0}^{\infty} e_n^0\zeta^n.$$

Summing the geometric series represented by the first sum on the right, we thus find

$$\rho_0(\zeta) = \gamma\frac{\zeta}{1-\zeta} + \beta\zeta\rho_0(\zeta) \implies \rho_0(\zeta) = \frac{\gamma\zeta}{(1-\zeta)(1-\beta\zeta)}.$$

Proof continued

Inserting this result into the formula for $\rho_k(\zeta)$ gives

$$\rho_k(\zeta) = \gamma\alpha^k \frac{\zeta^{k+1}}{(1-\zeta)(1-\beta\zeta)^{k+1}},$$

and the power series coefficients of this function bound the error of the parareal algorithm. To simplify we replace $1-\zeta$ in the denominator by $1-\beta\zeta$, which only increases the coefficients in the power series, because $\beta \geq 1$ and

$$\frac{1}{1-\zeta} = 1+\zeta+\zeta^2+\ldots \quad \text{and} \quad \frac{1}{1-\beta\zeta} = 1+\beta\zeta+\beta^2\zeta^2+\ldots,$$

and we thus consider the modified generating function

$$\tilde{\rho}_k(\zeta) = \gamma\alpha^k \frac{\zeta^{k+1}}{(1-\beta\zeta)^{k+2}},$$

We now use the general binomial series formula

$$\frac{1}{(1-z)^{b+1}} = \sum_{j=0}^{\infty} \binom{b+j}{j} z^j,$$

Proof continued

which gives in our case

$$\frac{1}{(1-\beta\zeta)^{k+2}} = \sum_{j=0}^{\infty} \binom{k+1+j}{j} \beta^j \zeta^j.$$

Therefore, the power series expansion of the modified generating function is

$$\begin{aligned} \tilde{\rho}_k(\zeta) &= \gamma\alpha^k \sum_{j=0}^{\infty} \binom{k+1+j}{j} \beta^j \zeta^{k+1+j} \\ &= \gamma\alpha^k \sum_{n=k+1}^{\infty} \binom{n}{n-k-1} \beta^{n-k-1} \zeta^n. \end{aligned}$$

The expansion coefficients for $n \leq k$ are zero, as we have seen in the finite step convergence result! Now using that

$$\binom{n}{n-k-1} = \binom{n}{k+1} = \frac{1}{(k+1)!} \prod_{\ell}^{k} (n-\ell),$$

Proof continued

we obtain

$$\tilde{\rho}_k(\zeta) = \frac{\gamma\alpha^k}{(k+1)!} \sum_{n=k+1}^{\infty} \prod_{\ell}^{k} (n-\ell)\beta^{n-k-1}\zeta^n.$$

The n-th coefficient e_n^k thus satisfies for $n > k$ the bound

$$e_n^k \leq \frac{\gamma\alpha^k}{(k+1)!}\beta^{n-k-1} \prod_{\ell}^{k} (n-\ell),$$

and we see that this bound also contains the zero bound for $n \leq k$ because of the product term which vanishes for $k > n$.

Since the error of parareal is bounded by e_n^k, we finally obtain using the values of α, β and γ that

Proof continued

$$||\boldsymbol{u}(T_n)-\boldsymbol{U}_n^k|| \leq \frac{\gamma\alpha^k}{(k+1)!}\beta^{n-k-1}\prod_{\ell}^{k}(n-\ell)$$

$$= \frac{C_3\Delta T^{p+1}C_1^k\Delta T^{(p+1)k}}{(k+1)!}(1+C_2\Delta T)^{n-k-1}\prod_{\ell=0}^{k}(n-\ell)$$

$$= \frac{C_3}{C_1}\frac{(C_1\Delta T^{p+1})^{k+1}}{(k+1)!}(1+C_2\Delta T)^{n-k-1}\prod_{\ell=0}^{k}(n-\ell),$$

the first result. For the second, Taylor expand the exponential function

$$(1+C_2\Delta T)^{n-k-1} \leq e^{C_2\Delta T(n-k-1)} = e^{C_2(T_n-T_{k+1})},$$

and over-estimate the product term,

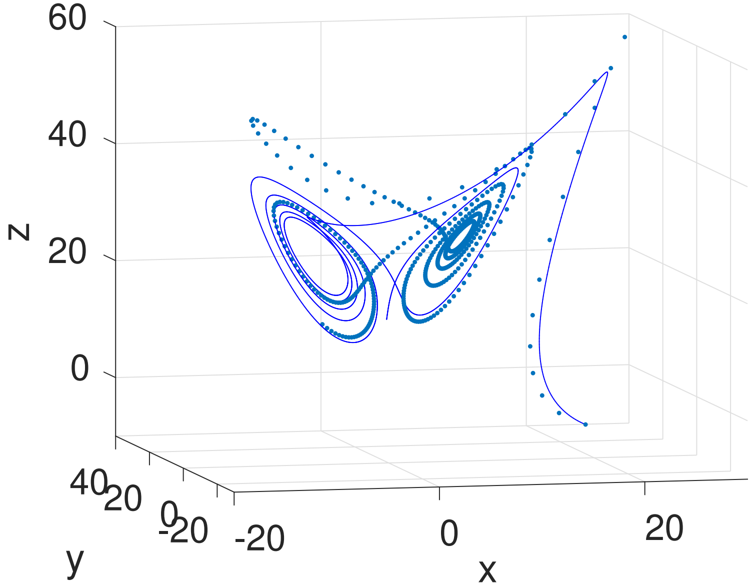
$$(C_1\Delta T^{p+1})^{k+1}\prod_{\ell=0}^{k}(n-\ell) \leq \Delta T^{p(k+1)}(C_1\Delta T)^{k+1}n^{k+1}$$

$$= \Delta T^{p(k+1)}(C_1 T_n)^{k+1}.$$

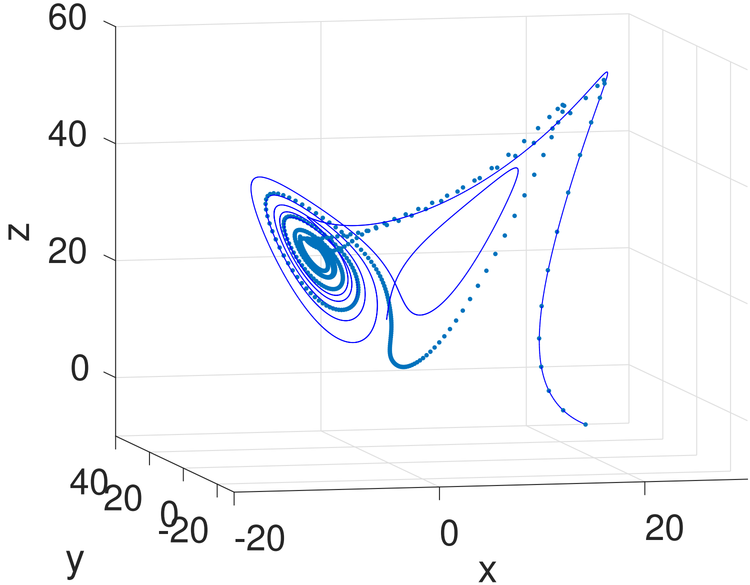
```
function U=Parareal(F,G,T,u0,N,K);
% PARAREAL implementation of the parareal algorithm
%   U=Parareal(F,G,T,u0,N,K); applies the parareal algorithm with fine
%   solver F(t0,t1,ut0) and coarse solver G(t0,t1,ut0) on [0,T] with
%   initial condition u0 at t=0 using N equidistant coarse time points
%   doing K iterations. The output U{k} contains the parareal
%   approximations at the coarse time points for each iteration k.

dT=T/N; TT=0:dT:T;                                  % coarse time mesh
U{1}(1,:)=u0;
for n=1:N                                           % initial guess with G
  Go(n+1,:)=G(TT(n),TT(n+1),U{1}(n,:));
  U{1}(n+1,:)=Go(n+1,:);                            % keep Go for parareal
end;
for k=1:K                                           % parareal iteration
  for n=1:N
    Fn(n+1,:)=F(TT(n),TT(n+1),U{k}(n,:));           % parallel with F
  end;
  U{k+1}(1,:)=u0;
  for n=1:N
    Gn(n+1,:)=G(TT(n),TT(n+1),U{k+1}(n,:));         % sequential with G
    U{k+1}(n+1,:)=Fn(n+1,:)+Gn(n+1,:)-Go(n+1,:); % parareal update
  end;
  Go=Gn;                                            % keep for next iteration
end;
```

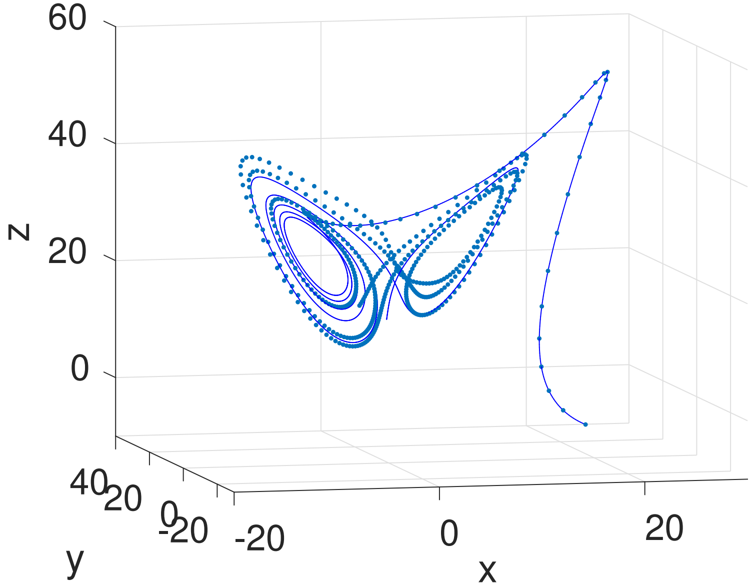
Parareal for the Lorenz equations: k=1



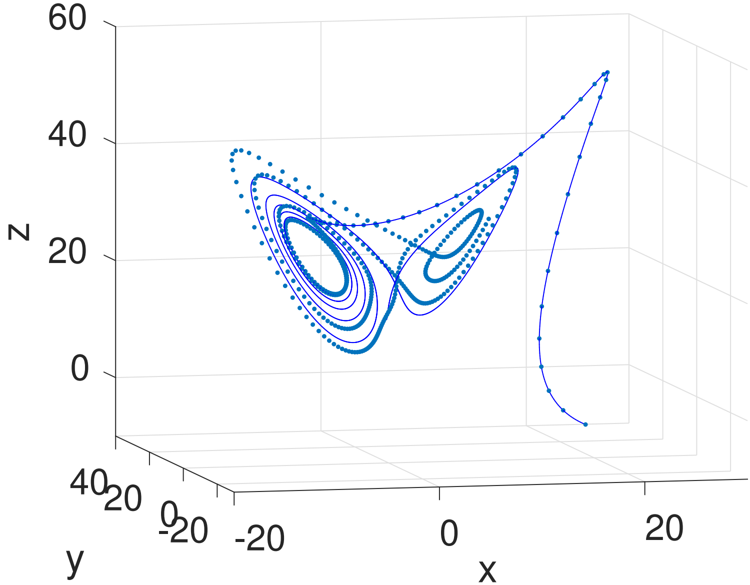
Parareal for the Lorenz equations: k=2



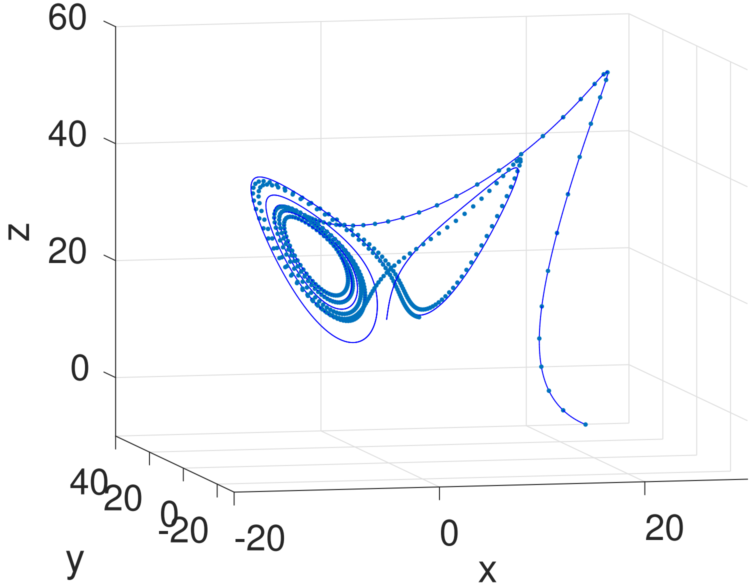
Parareal for the Lorenz equations: k=3



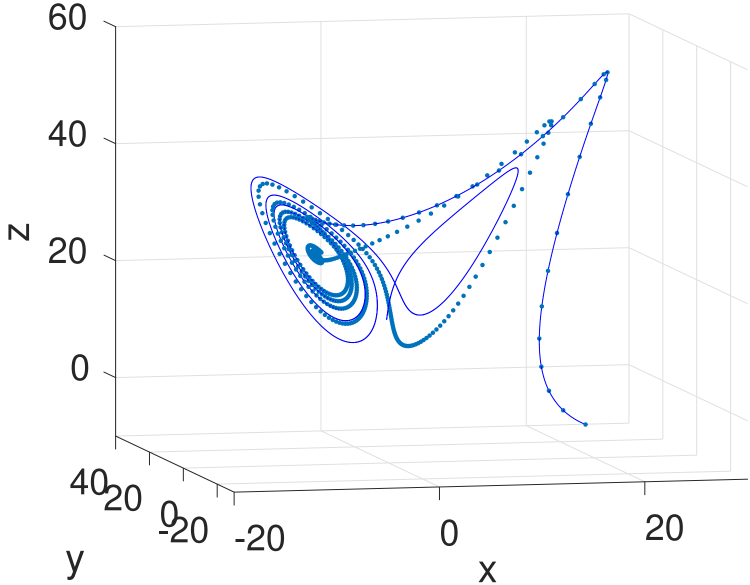
Parareal for the Lorenz equations: k=4



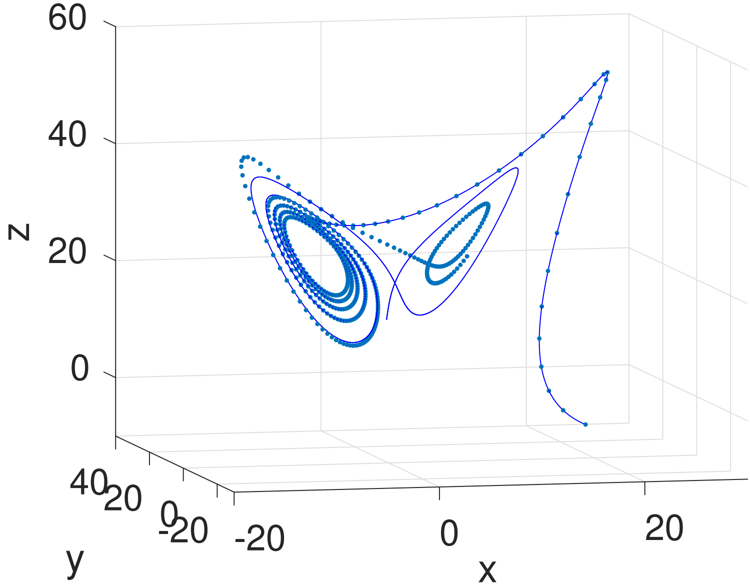
Parareal for the Lorenz equations: k=5



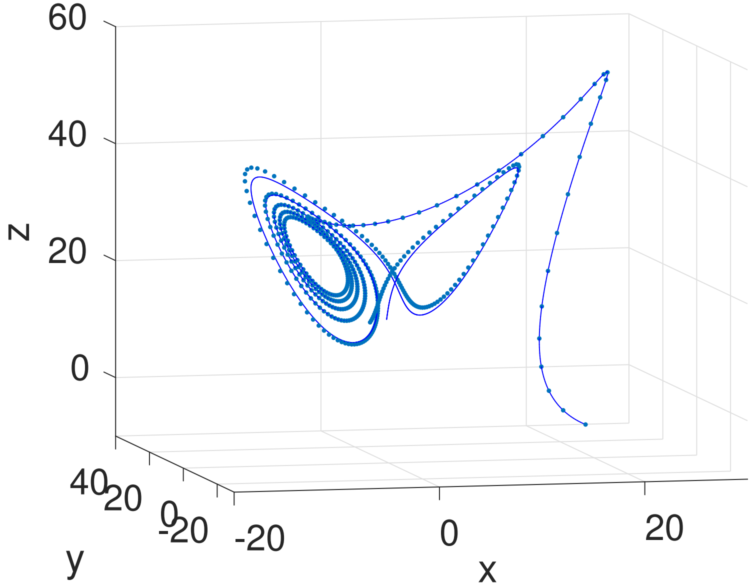
Parareal for the Lorenz equations: k=6



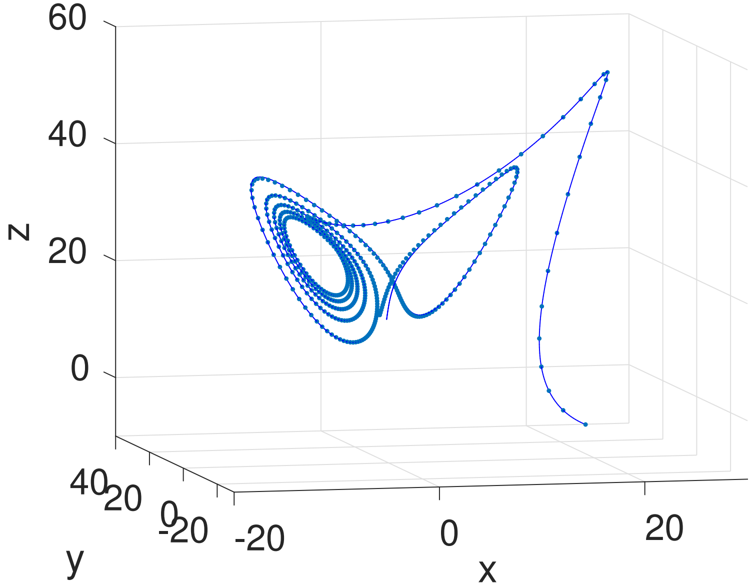
Parareal for the Lorenz equations: k=7



Parareal for the Lorenz equations: k=8

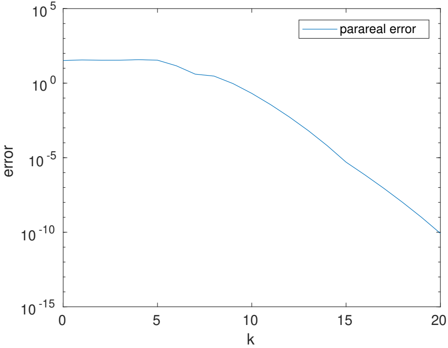


Parareal for the Lorenz equations: k=9



Do we achieve any speedup?

- We used $N = 500$ coarse time intervals
- We could have used 500 'processors'
- We needed eight iterations
- We could have computed the equivalent of eight fine approximations sequentially (neglecting the coarse propagator and communication)
- This would correspond to a speedup of $500/8 \approx 60$.



Performance for Dahlquist's equation

$$\partial_t u = \lambda u, \quad u(0) = u^0, \quad \lambda \in \mathbb{C}.$$

In that case, one can obtain two different convergence estimates, depending on if solutions are growing or not.

Theorem (Superlinear convergence)

For Dahlquist's equation with exact fine solver $F(T_{n+1}, T_n, v) = ve^{\lambda \Delta T}$ *and coarse solver* $G(T_{n+1}, T_n, v) = vR_G(\lambda \Delta T)$ *where* R_G *is the stability function of* G*, and arbitrary initialization* U_n^0 *satisfying* $U_0^0 = u^0$*, we have the estimate*

$$\begin{aligned}\max_{1\le n\le N} |u(T_n) - U_n^k| \quad &\le \quad \frac{|e^{\lambda\Delta T} - R_G(\lambda\Delta T)|^k}{k!} R_0^{n-k-1} \\ &\quad \times \prod_{\ell=1}^{k} (N-\ell) \max_{1\le n\le N} |u(T_n) - U_n^0|,\end{aligned}$$

with $R_0 := |R_G(\lambda\Delta T)|$ *if* $|R_G(\lambda\Delta T)| > 1$*, and else* $R_0 = 1$*.*

Linear convergence for decaying solutions

Theorem (Linear convergence)

If $\Re(\lambda) \leq 0$, and ΔT is such that G is in its region of absolute stability, $|R_G(\lambda\Delta T)| < 1$, then the parareal algorithm applied to the Dahlquist test equation satisfies for all time the estimate

$$\sup_{n>0} |u(T_n) - U_n^k| \leq \left(\frac{|e^{\lambda\Delta T} - R_G(\lambda\Delta T)|}{1 - |R_G(\lambda\Delta T|} \right)^k \sup_{n>0} |u(T_n) - U_n^0|.$$

Proof. As in the earlier proof we arrive at

$$\begin{aligned} \left|u(T_{n+1}) - U_{n+1}^{k+1}\right| &\leq \left|e^{\lambda\Delta T} - R_G(\lambda\Delta T)\right| \left|u(T_n) - U_n^k\right| \\ &\quad + |R_G(\lambda\Delta T)| \left|u(T_n) - U_n^{k+1}\right| \\ &\leq \left|e^{\lambda\Delta T} - R_G(\lambda\Delta T)\right| \sup_{n>0} \left|u(T_n) - U_n^k\right| \\ &\quad + |R_G(\lambda\Delta T)| \sup_{n>0} \left|u(T_n) - U_n^{k+1}\right| \end{aligned}$$

Proof continued

Note that we used that solutions remain bounded to take the sup on the right.

We can thus also take the sup on the left, and using that

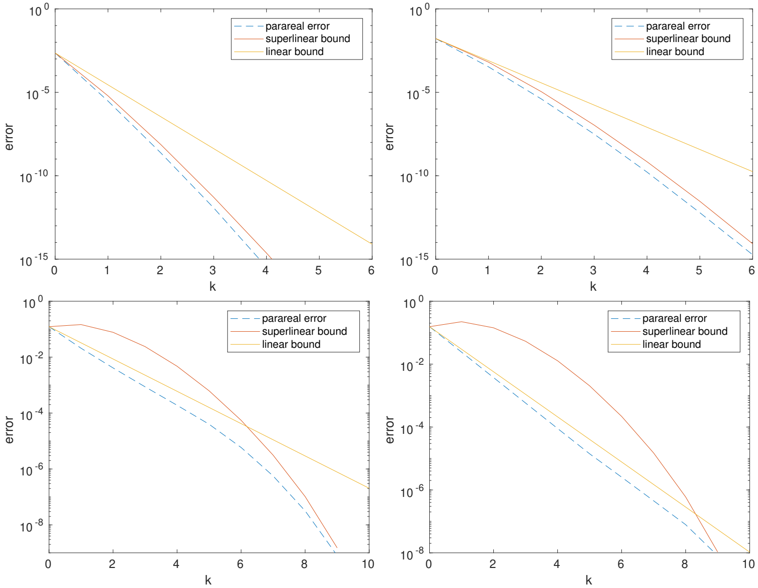
$$U_0^{k+1} = u(T_0) := \boldsymbol{u}^0$$

by the definition of the parareal algorithm, we can include the index zero on the left as well. We thus obtain

$$\begin{aligned}(1 - |R_G(\lambda\Delta T)|) \sup_{n>0} \left|u(T_n) - U_n^{k+1}\right| \\ \leq \left|e^{\lambda\Delta T} - R_G(\lambda\Delta T)\right| \sup_{n>0} \left|u(T_n) - U_n^k\right|,\end{aligned}$$

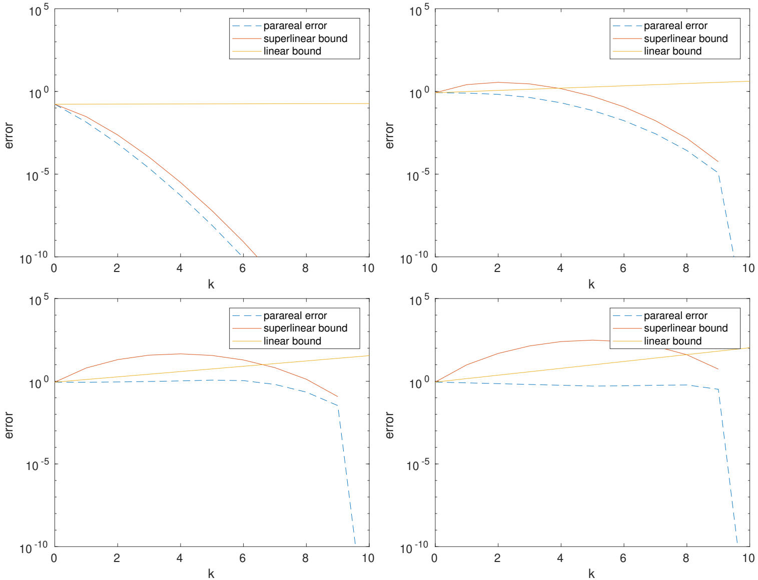
which implies the result by induction.

Numerical Experiment for Dahlquist's equation



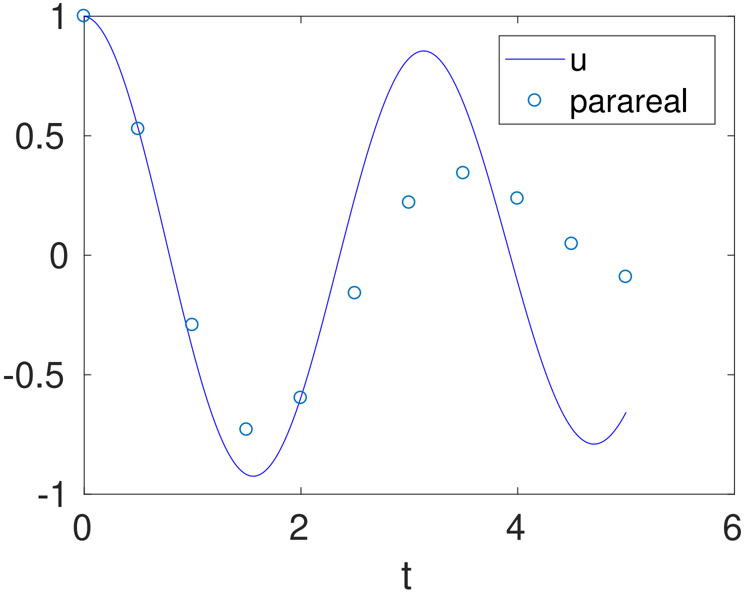
Errors and bounds with $\lambda = -1$ for $T = 0.25, 1, 10, 50$.

Numerical Experiment for Dahlquist's equation

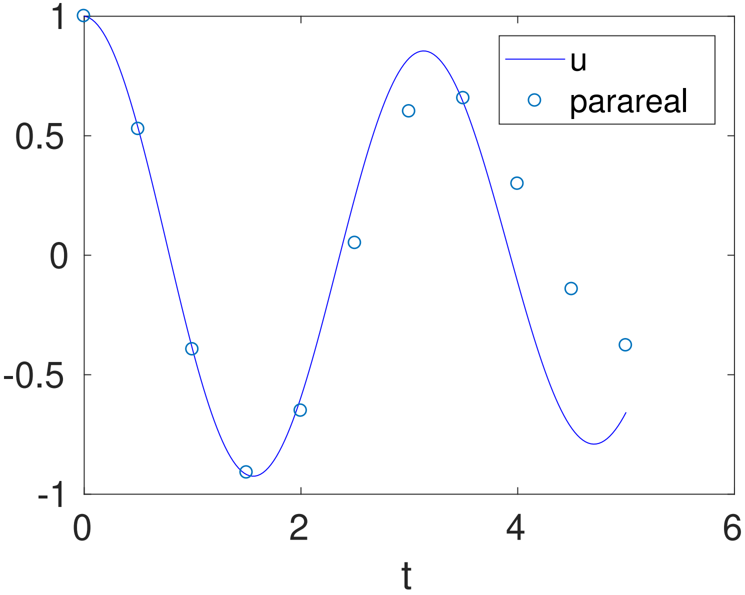


Errors and bounds with $\lambda = 2i$ for $T = 1, 5, 10, 20$

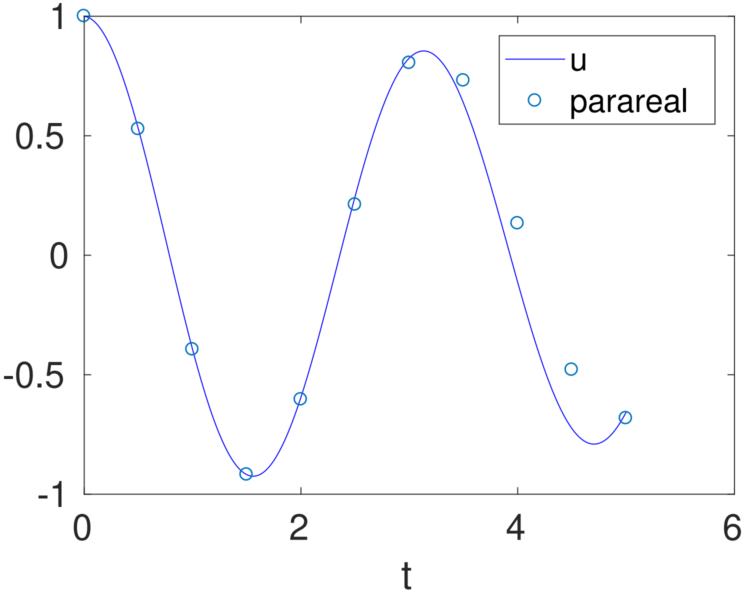
Snapshots of the solution, $T = 5$, $k = 1$



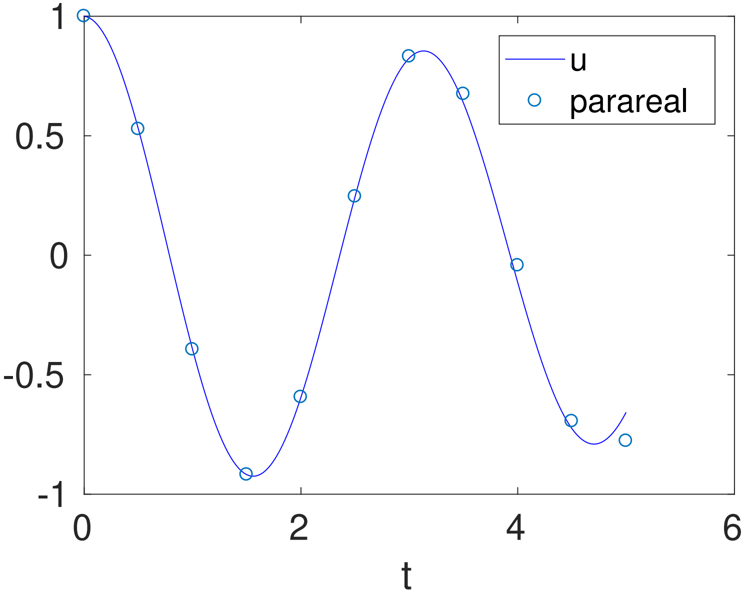
Snapshots of the solution, $T = 5$, $k = 2$



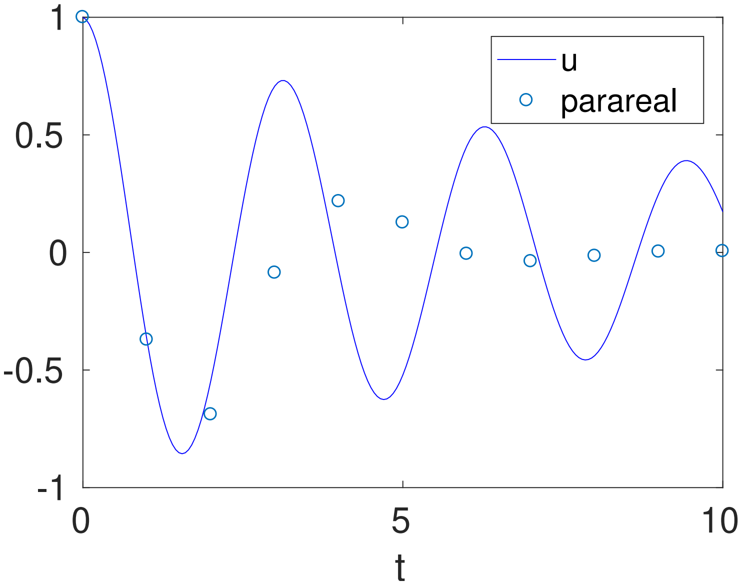
Snapshots of the solution, $T = 5$, $k = 3$



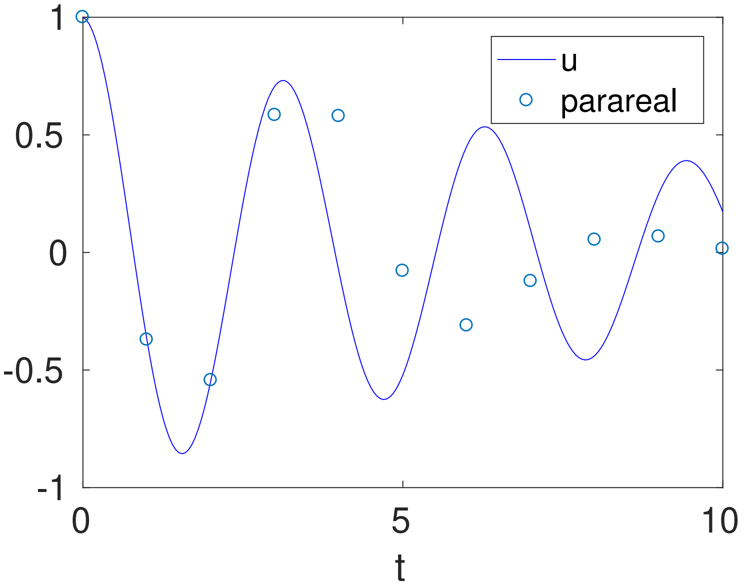
Snapshots of the solution, $T = 5$, $k = 4$



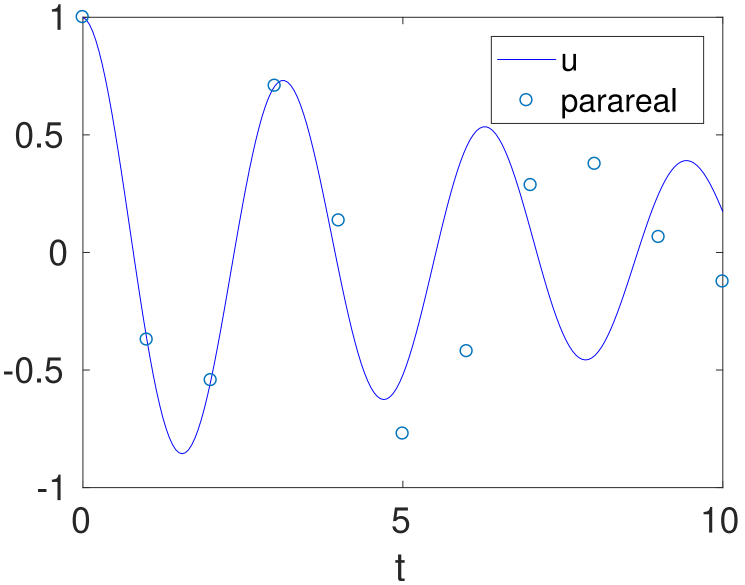
Snapshots of the solution, $T = 10$, $k = 1$



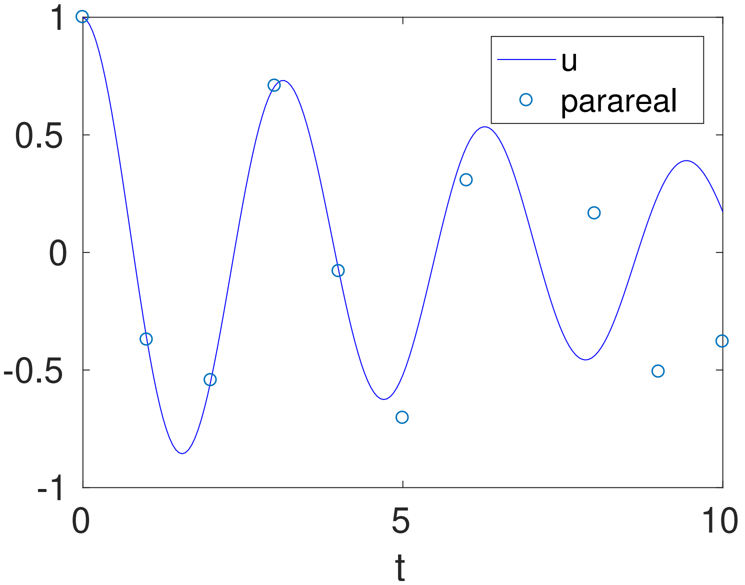
Snapshots of the solution, $T = 10$, $k = 2$



Snapshots of the solution, $T = 10$, $k = 3$



Snapshots of the solution, $T = 10$, $k = 4$



Heat Equation

$$\begin{array}{rcll} \partial_t u(x,t) & = & \partial_{xx} u(x,t) & \text{in } (0,\pi)\times(0,T], \\ u(x,0) & = & u_0(x) & \text{in } (0,\pi), \\ u(0,t) & = & 0 & \text{in } (0,T], \\ u(\pi,t) & = & 0 & \text{in } (0,T]. \end{array}$$

A Fourier sine series expansion

$$u(x,t) = \sum_{\omega=1}^{\infty} \hat{u}(\omega,t)\sin(\omega x)$$

leads to

$$\partial_t u = \sum_{\omega=1}^{\infty} \partial_t \hat{u}(\omega,t)\sin(\omega x) = \partial_{xx} u = -\sum_{\omega=1}^{\infty} \omega^2 \hat{u}(\omega,t)\sin(\omega x).$$

The Fourier coefficients must thus satisfy the equation

$$\partial_t \hat{u}(\omega,t) = -\omega^2 \hat{u}(\omega,t),$$

a special case of the Dahlquist test equation

Parareal Convergence for the Heat Equation

Theorem

Let F be exact, G have stability function R_G with $sup_{x<0}|e^x - R_G(x)| = \rho_s$ finite. Then

$$\max_{1\leq n\leq N} ||u(t_n) - U_n^k||_2 \leq \frac{\rho_s^k}{k!}\prod_{\ell=1}^{k}(N-\ell)C_0^N,$$

where $||\cdot||_2$ denotes the spectral norm in space, and $C_0^N := \sqrt{\sum_{\omega=1}^{\infty} \max_{1\leq n\leq N}|\hat{u}(T_n) - \hat{U}_n^0|^2}$.

If the negative real axis is in the region of absolute stability of G and $\lim_{x\to-\infty}|R_G(x)| < 1$ (A_0-stability), then

$$\sup_{n>0} ||u(t_n) - U_n^k||_2 \leq \rho_l^k C_0^\infty,$$

with constant ρ_l depending only on G.

Proof.

The numerator in the superlinear bound becomes

$$|e^{-\omega^2\Delta T} - R_G(-\omega^2\Delta T)|^k \leq \rho_s^k,$$

because G is A_0-stable. We can thus square

$$|u(T_n) - U_n^k| \leq \frac{|e^{\lambda\Delta T} - R_G(\lambda\Delta T)|^k}{k!}|R_G(\lambda\Delta T)|^{n-k-1} \prod_{\ell=1}^{k}(n-\ell) \max_{1\leq n\leq N}|u(T_n) - U_n^0|,$$

in the proof of the Dahlquist case and sum over ω,

$$\sum_{\omega=1}^{\infty}|\hat{u}(T_n)-\hat{U}_n^k|^2 \leq \left(\frac{\rho_s^k}{k!}\prod_{\ell=1}^{k}(n-\ell)\right)^2 \sum_{\omega=1}^{\infty}\max_{1\leq n\leq N}|\hat{u}(T_n)-\hat{U}_n^0|^2,$$

where we used that $|R_G(-\omega^2\Delta T)|^{n-k-1} \leq 1$. We can now use Parseval-Plancherel on the left to conclude.

Proof continued

Similarly we define from the linear bound in the Dahlquist case

$$\rho_l := \sup_{\omega\in\mathbb{R}} \frac{|e^{-\omega^2\Delta T} - R_G(-\omega^2\Delta T)|}{1-|R_G(-\omega^2\Delta T|},$$

and obtain for the Fourier coefficients

$$\sup_{n>0} |\hat{u}(T_n) - \hat{U}_n^k|^2 \leq \rho_l^{2k} \sup_{n>0} |\hat{u}(T_n) - \hat{U}_n^0|^2.$$

Summing the squares of the Fourier coefficients gives

$$\begin{aligned}
\sup_{n>0}\sum_{\omega=1}^{\infty} |\hat{u}(T_n) - \hat{U}_n^k|^2 &\leq \sum_{\omega=1}^{\infty}\sup_{n>0} |\hat{u}(T_n) - \hat{U}_n^k|^2 \\
&\leq \rho_l^{2k}\sum_{\omega=1}^{\infty}\sup_{n>0} |\hat{u}(T_n) - \hat{U}_n^0|^2,
\end{aligned}$$

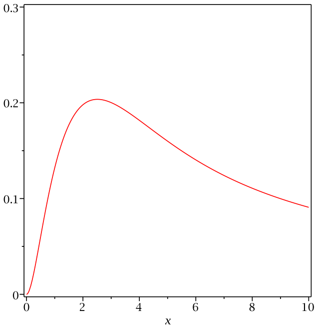
and using again Parseval-Plancherel on the left gives the result.

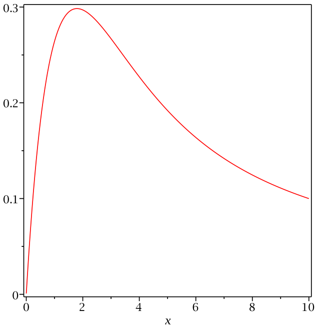
Backward Euler Example

If G is Backward Euler, $R_G(z) = \frac{1}{1-z}$, and we get

$$\rho_s = \sup_{x<0} \left| e^x - \frac{1}{1-x} \right| = \sup_{x>0} \left| e^{-x} - \frac{1}{1+x} \right|$$

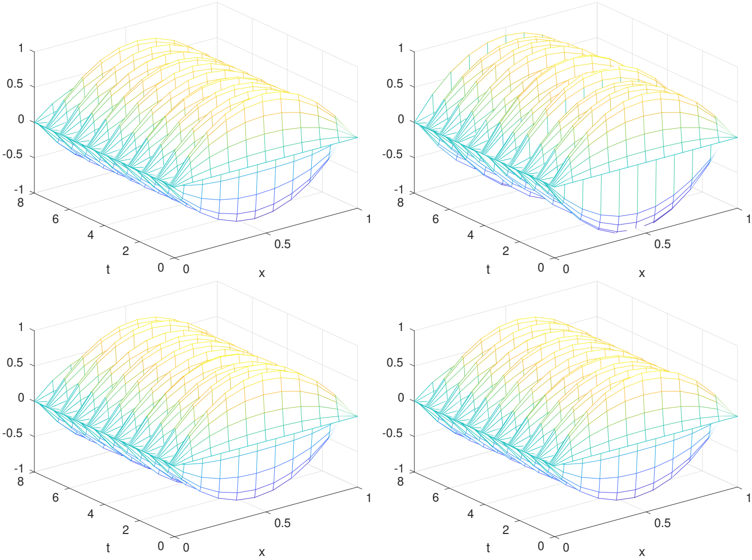
$$\rho_l = \sup_{\omega \in \mathbb{R}} \frac{\left| e^{-\omega^2 \Delta T} - \frac{1}{1+\omega^2 \Delta T} \right|}{1 - \left| \frac{1}{1+\omega^2 \Delta T} \right|} = \sup_{x>0} \frac{\left| e^{-x} - \frac{1}{1+x} \right|}{1 - \left| \frac{1}{1+x} \right|} = \sup_{x>0} \frac{\left| (1+x)e^{-x} - 1 \right|}{x}$$





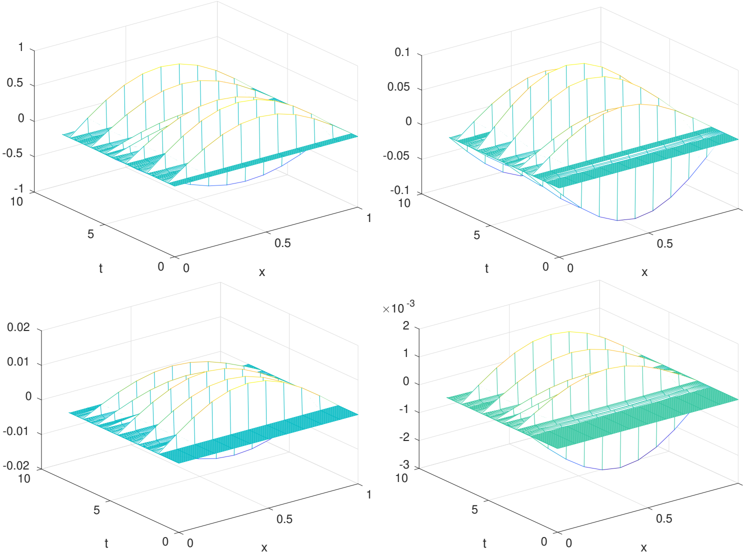
$\rho_s = 0.2036321888$ $\rho_l = 0.2984256075$

Numerical Experiment



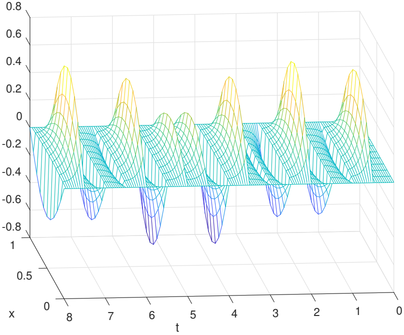
Solution of the heat equation, initial coarse approximation and the first two parareal iterations

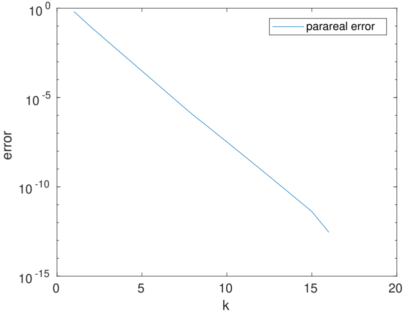
Error



Error in the initial coarse approximation and the first three parareal iterations when solving the heat equation example

Intuition why Parareal works so well





Left: error turned after the first parareal iteration

Right: Decay of the error in the maximum norm in time and L^2 norm in space

The transport or advection equation

$$\begin{aligned}\partial_t u(x,t) + a\partial_x u(x,t) &= 0 \quad \text{in } \mathbb{R}\times(0,T],\\ u(x,0) &= u_0(x) \quad \text{in } \mathbb{R}.\end{aligned}$$

A Fourier transform in space

$$\begin{aligned}u(x,t) &= \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\infty} e^{i\omega x}\hat{u}(\omega,t)d\omega,\\ \hat{u}(x,t) &= \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\infty} e^{-i\omega x}u(x,t)dx,\end{aligned}$$

shows that each Fourier mode $\hat{u}(\omega,t)$ satisfies

$$\partial_t \hat{u}(\omega,t) + i\omega a\hat{u}(\omega,t) = 0, \quad \hat{u}(\omega,0) = \hat{u}_0(\omega),$$

again a special case of the Dahlquist test equation. Note that for the hyperbolic transport equation, the corresponding λ lies indeed on the imaginary axis.

Convergence estimate

Theorem

Let F be exact and G have stability function R_G such that $sup_{x\in\mathbb{R}}|e^{ix} - R_G(ix)| = \rho_s$ is bounded. Then

$$\max_{1\le n\le N} ||u(t_n) - U_n^k||_2 \le \frac{\rho_s^k}{k!}\prod_{\ell=1}^{k}(N-\ell)C_0^N,$$

where $||\cdot||_2$ denotes the spectral norm in space, and

$$C_0^N := \sqrt{\sum_{\omega=1}^{\infty}\max_{1\le n\le N}|\hat{u}(T_n) - \hat{U}_n^0|^2}.$$

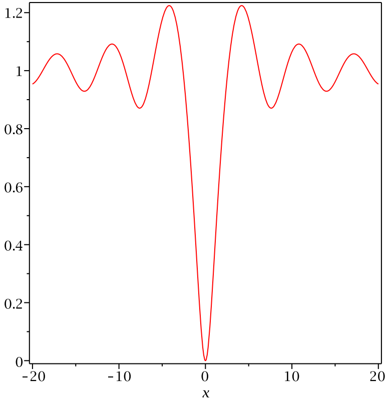
Proof.

The proof is as in the case of the heat equation. $\square$

Example for Backward Euler

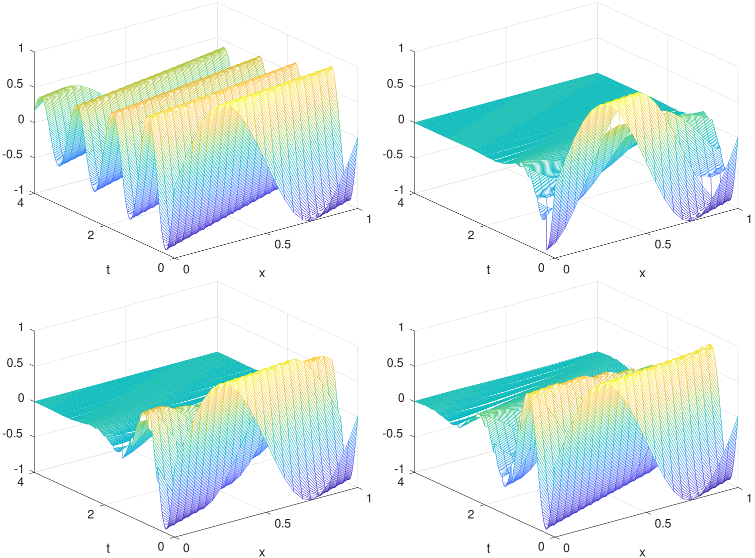
For G Backward Euler, the stability function is $R_G(z) = \frac{1}{1-z}$, and thus

$$\rho_s = \sup_{x\in\mathbb{R}} \left|e^{ix} - \frac{1}{1-ix}\right|.$$



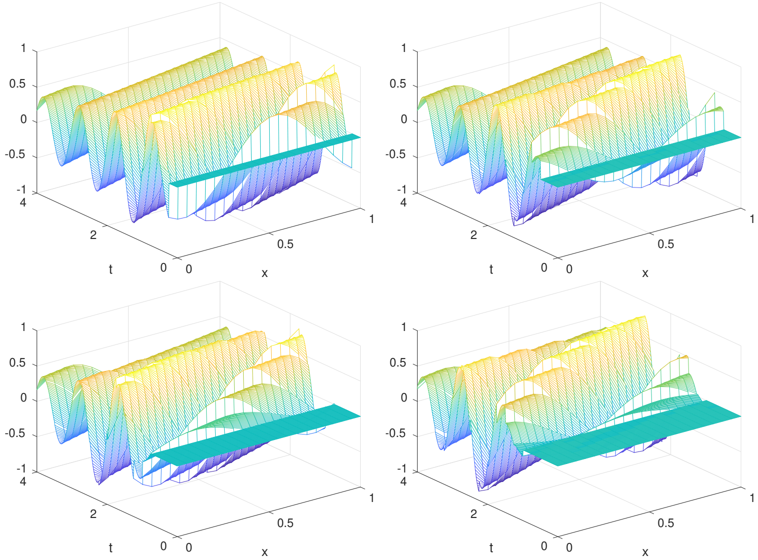
$$\rho_s = 1.224353426$$

Numerical Example



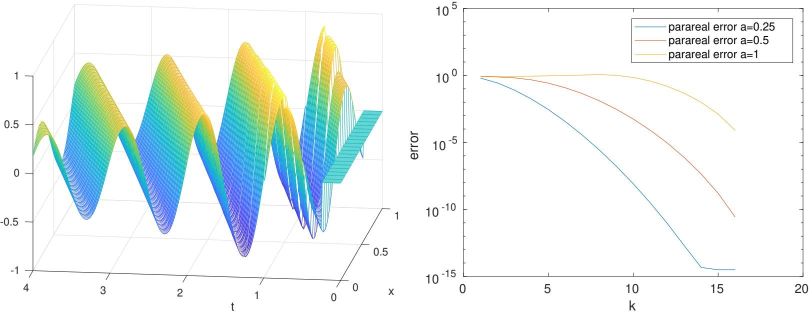
Solution of the transport equation, initial coarse approximation and the first two parareal iterations

Plot of the error



Error in the initial coarse approximation and the first three parareal iterations for the transport equation example

Intuition why this does not work well



Left: error after the first parareal iteration with the view turned

Right: Decay of the error in the maximum norm of the parareal iterates when solving the transport equation for three different transport speeds: $a = 0.25, 0.5, 1$.