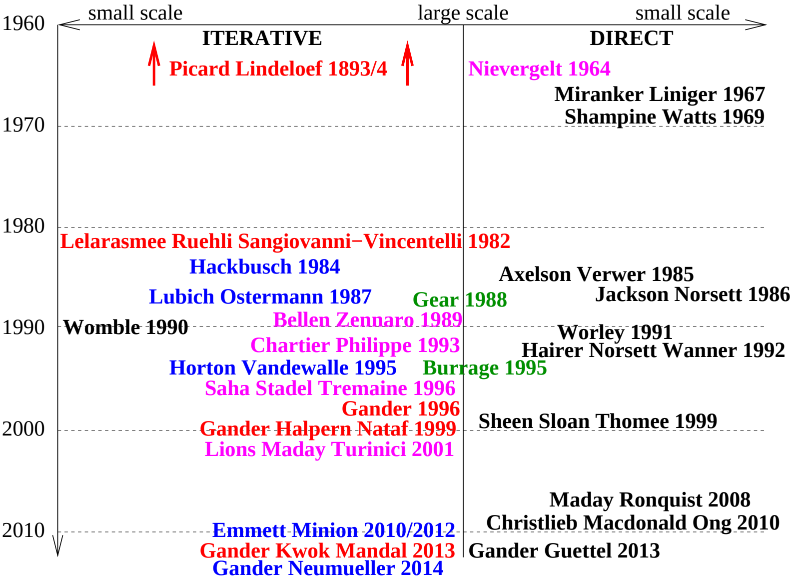
Direct Time Parallel Solvers

Martin J. Gander
martin.gander@unige.ch

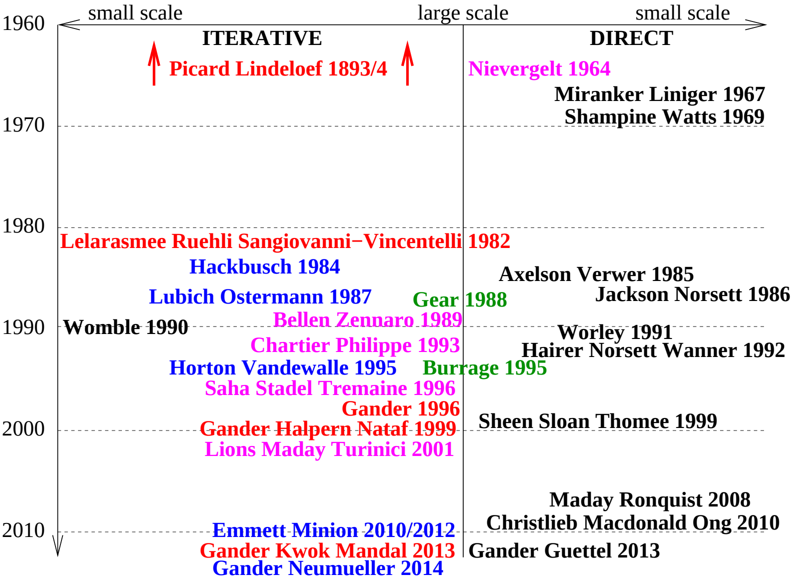
University of Geneva

Toulouse, January 2016

Time Parallel Methods Over the Course of Time



Time Parallel Methods Over the Course of Time

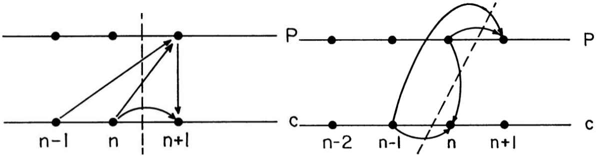


50 Years of time parallel time integration (G, 2015)

Miranker Liniger 1967

Parallel Methods for the Numerical Integration of Ordinary Differential Equations. Math. Comp., Vol 21.

"Let us consider how we might widen the computation front."



For $y' = f(x, y)$, consider the predictor corrector formulas

$$y^p_{n+1} = y^c_n + \frac{h}{2}(3f(y^c_n) - f(y^c_{n-1})), \quad y^c_{n+1} = y^c_n + \frac{h}{2}(f(y^p_{n+1}) + f(y^c_n)).$$

This process is sequential. Consider the modified formulas

$$y^p_{n+1} = y^c_{n-1} + 2hf(y^p_n), \quad y^c_n = y^c_{n-1} + \frac{h}{2}(f(y^p_n) + f(y^c_{n-1})).$$

Those two can be evaluated in parallel.
Results: Methods for $2s$ processors with stability and convergence analysis.

Shampine and Watts 1969

Block Implicit One-Step Methods. Math. of Comp, Vol 23., No. 108.

"A class of one-step methods which obtain a block of r new values at each step are studied."

Example (Clippinger and Dimsdale): for $y' = f(x, y)$,

$$y_{n+1} - \frac{1}{2} y_{n+2} = \frac{1}{2} y_n + \frac{h}{4} f(x_n, y_n) - \frac{h}{4} f(x_{n+2}, y_{n+2}),$$

$$y_{n+2} = y_n + \frac{h}{3} f(x_n, y_n) + \frac{4h}{3} f(x_{n+1}, y_{n+1}) + \frac{h}{3} f(x_{n+2}, y_{n+2})$$

General formulation for r new steps, $\mathbf{y} = (y_{n+1}, \ldots, y_{n+r})$

$$A\mathbf{y} = y_n \mathbf{e} + hf(x_n, y_n)\mathbf{d} + hBF(\mathbf{y}).$$

Solved by fixed point iteration

$$\mathbf{y}^{k+1} = y_n A^{-1} \mathbf{e} + hf(x_n, y_n) A^{-1} \mathbf{d} + hA^{-1} BF(\mathbf{y}^k).$$

Doing just one or a few steps gives a parallel method but reduces stability

Hairer, Nørsett, Wanner 1992

Solving Ordinary Differential Equations I, Springer

Hairer, Nørsett, Wanner 1992

Solving Ordinary Differential Equations I, Springer

... it seems that *explicit* Runge-Kutta methods are not facilitated much by parallelism at the method level (Iserles and Nørsett 1990)

Hairer, Nørsett, Wanner 1992

Solving Ordinary Differential Equations I, Springer

... it seems that *explicit* Runge-Kutta methods are not facilitated much by parallelism at the method level (Iserles and Nørsett 1990)

"Paralysing ODEs" (K. Burrage talk in Helsinki 1990)

Hairer, Nørsett, Wanner 1992

Solving Ordinary Differential Equations I, Springer

... it seems that *explicit* Runge-Kutta methods are not facilitated much by parallelism at the method level (Iserles and Nørsett 1990)

"Paralysing ODEs" (K. Burrage talk in Helsinki 1990)

Parallel Runge-Kutta Methods:

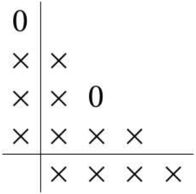


Fig. 11.1. Parallel method

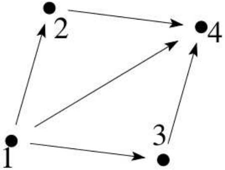


Fig. 11.2. Production graph

Theorem (Jackson and Nørsett 1986): For an explicit RK method with σ sequential stages, the order is at most σ.
$\Longrightarrow$ P-optimal methods.

Result (Hairer, Nørsett and Wanner 1992): Parallel Iterated RK and GBS Extrapolation methods are P-optimal.

Christlieb Macdonald Ong 2010

Parallel High-Order Integrators, SISC, Vol. 32, No. 2.

> *"…we discuss a class of integral defect correction methods which is easily adapted to create parallel time integrators for multicore architectures"*

Review of deferred (difference or defect) correction:

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

Let $\tilde{u}_m$ be an order one approximation (e.g. FE). If $\tilde{u}(t)$ is its interpolant, the error $e(t) := u(t) - \tilde{u}(t)$ satisfies

$$e'(t) = u'(t) - \tilde{u}'(t) = f(u) - \tilde{u}'(t) = f(e + \tilde{u}) - \tilde{u}'(t)$$

a differential equation for $e(t)$ with $e(0) = 0$! Solving it with FE we get e_m:

Theorem (Skeel 1976, see also Fox 1947, Pereyra 1967, Frank and Ueberhuber 1977): The new approximation $\tilde{u}_m + e_m$ is of order two. $\Longrightarrow$ iterated defect correction

Problems: numerical interpolation and differentiation

Dutt, Greengard, Rokhlin 2000

Spectral integral deferred correction:

$$u' = f(u), \quad u(0) = u_0 \quad \Longrightarrow \quad u(t) = u(0) + \int_0^t f(u(\tau))d\tau.$$

Let $\tilde{u}(t)$ be an approximation with residual

$$r(t) := \tilde{u}(0) + \int_0^t f(\tilde{u}(\tau))d\tau - \tilde{u}(t),$$

The error $e(t) := u(t) - \tilde{u}(t)$ satisfies (with $u(0) = \tilde{u}(0)$)

$$\tilde{u}(t) + e(t) = \tilde{u}(0) + \int_0^t f(\tilde{u}(\tau) + e(\tau))d\tau.$$

$$\begin{aligned} \Longrightarrow e(t) &= \tilde{u}(0) + \int_0^t f(\tilde{u}(\tau) + e(\tau))d\tau - \tilde{u}(t) \\ &= r(t) + \int_0^t f(\tilde{u}(\tau) + e(\tau)) - f(\tilde{u}(\tau))d\tau \end{aligned}$$

Using differentiation, we obtain from

$$e(t) = r(t) + \int_0^t f(\tilde{u}(\tau) + e(\tau)) - f(\tilde{u}(\tau))d\tau,$$

the differential equation for the error

$$e'(t) = r'(t) + f(\tilde{u}(t) + e(t)) - f(\tilde{u}(t)).$$

Starting with an order one method, e.g. FE

$$\tilde{u}_{m+1} = \tilde{u}_m + \Delta t f(\tilde{u}_m), \quad \text{for } m = 0, 1, \ldots, M-1.$$

one evaluates with high order quadrature the residual

$$r(t) = \tilde{u}(0) + \int_0^t f(\tilde{u}(\tau))d\tau - \tilde{u}(t),$$

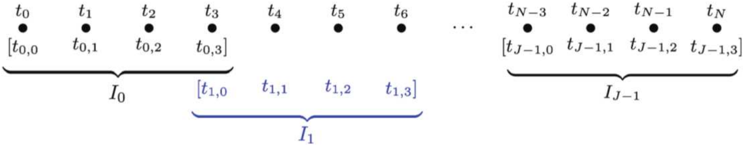
and then computes the approximate error with FE

$$e_{m+1} = e_m + r_{m+1} - r_m + \Delta t(f(\tilde{u}_m + e_m) - f(\tilde{u}_m)).$$

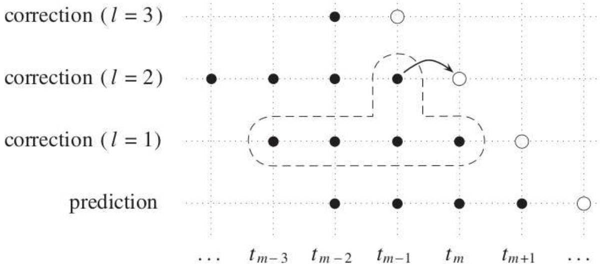
Theorem (Böhmer and Stetter 1984) The new approximation $\tilde{u}_m + e_m$ is of order two.

Revisionist Integral Deferred Correction (RIDC)

Classical progression of integral deferred correction:



Parallelizable version in RIDC:



$\Longrightarrow$ pipelining ideal for multicore architectures

Worley 1991

Parallelizing across time when solving time-dependent partial differential equations, Proc. 5th SIAM Conf. on Parallel Processing for Scientific Computing

> *"The waveform relaxation multigrid algorithm is normally implemented in a fashion* **that is still intrinsically sequential in the time direction**.*"*

$$\begin{pmatrix} a_{11} & & & \\ a_{21} & a_{22} & & \\ & a_{32} & a_{33} & \\ & & a_{43} & a_{44} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{pmatrix}.$$

One step of cyclic reduction:

$$\begin{pmatrix} a_{22} & \\ -\frac{a_{43}}{a_{33}} a_{32} & a_{44} \end{pmatrix} \begin{pmatrix} x_2 \\ x_4 \end{pmatrix} = \begin{pmatrix} f_2 - \frac{a_{21}}{a_{11}} f_1 \\ f_4 - \frac{a_{43}}{a_{33}} f_3 \end{pmatrix},$$

Serial complexity: forward substitution $3n$, cyclic reduction $7n$
Parallel complexity of cyclic reduction is a logarithm in n

Cyclic Reduction in Waveform Relaxation

For a system of ODEs

$$\mathbf{u}_t = A\mathbf{u}, \quad \mathbf{u}(0) = \mathbf{u}_0,$$

Jacobi waveform relaxation is $(A = L + D + U)$

$$\mathbf{u}_t^k = D\mathbf{u}^k + (L + U)\mathbf{u}^{k-1}, \quad \mathbf{u}^k(0) = \mathbf{u}^0.$$

Solving each scalar ODE in this iteration using cyclic reduction, in the context of multigrid waveform relaxation, Worley reached optimal parallel complexity:

Result (Worley 1991): Parallel complexity is $\Theta(\log^2 N_s \log^\gamma N_t)$, $\gamma = \frac{1}{2}\lceil \text{levels} \rceil$ (Multigrid for Laplace has $\Theta(\log^2 N_s)$).

See also Horton, Vandewalle and Worley (SISC 1995) and Simoens and Vandewalle (SISC 2000)

Sheen, Sloan and Thomée 1999

A parallel method for time-discretization of parabolic problems based on contour integral representation and quadrature, Math. of Comp., Vol. 69, No. 1.

> *"These problems are completely independent, and can therefore be computed on separate processors."*

$$\mathbf{u}_t + A\mathbf{u} = 0, \quad u(0) = u_0,$$

Laplace transform with parameter s

$$s\hat{\mathbf{u}} + A\hat{\mathbf{u}} = \mathbf{u}_0 \quad \Longrightarrow \quad \hat{\mathbf{u}} = (sI + A)^{-1}\mathbf{u}_0.$$

Inverse Laplace transform

$$\mathbf{u}(t) = \frac{1}{2\pi i}\int_\Gamma e^{st}\hat{\mathbf{u}}(s)ds.$$

Approximating the integral with a quadrature formula with nodes s_j, one only needs to compute $\hat{\mathbf{u}}(s)$ at $s = s_j$.

More Recent Results

Sheen, Sloan and Thomée (2003): A parallel method for time discretization of parabolic equations based on Laplace transformation and quadrature, IMA Journal of Numerical Analysis, 23.

Thomée (2005): A high order parallel method for time discretization of parabolic type equations based on Laplace transformation and quadrature, Int. J. Numer. Anal. Model, 2.

Lai (2010): On transformation methods and the induced parallel properties for the temporal domain, in Substructuring Techniques and Domain Decomposition Methods, F. Magoulès, ed., Saxe-Coburg Publications.

Douglas, Kim, Lee and Sheen (2011): Higher-order schemes for the Laplace transformation method for parabolic problems, Computing and visualization in science, 14.

Maday and Rønquist 2008

Parallelization in time through tensor-product space-time solvers, CRAS, Vol. 346, No. 1.

> *"Pour briser la nature intrinsèquement séquentielle de cette résolution, on utilise l'algorithme de produit tensoriel rapide."*

Suppose we discretize $u_t = Lu$ using Backward Euler:

$$B\mathbf{u} := \begin{pmatrix} \frac{1}{\Delta t_1} - L & & & \\ -\frac{1}{\Delta t_2} & \frac{1}{\Delta t_2} - L & & \\ & \ddots & \ddots & \\ & & -\frac{1}{\Delta t_N} & \frac{1}{\Delta t_N} - L \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \\ \vdots \\ u_N \end{pmatrix} = \begin{pmatrix} f_1 + \frac{1}{\Delta t_1} u_0 \\ f_2 \\ \vdots \\ f_N \end{pmatrix} = \mathbf{f}$$

If B is diagonalizable, $B = SDS^{-1}$, we can solve in 3 steps:

$$S\mathbf{g} = \mathbf{f}, \quad (\frac{1}{\Delta t_n} - L)w_n = g_n, \quad S^{-1}\mathbf{u} = \mathbf{w}.$$

Problem: B is not diagonalizable if all time steps are equal. How should one choose Δt_j ?

Truncation Error Estimates

Study the model problem

$$\frac{du}{dt} + au = 0, \quad t \in (0, T), \quad u(0) = u_0$$

Theorem (G, Halpern, Ryan, Tran 2014)

For a Backward Euler discretization, the error is minimized if all time steps are equal.

To be able to diagonalize, we introduce a geometric mesh $\Delta t_n = (1+\epsilon)^{n-1}\Delta t_1$, $n = 2, \ldots, N$ and associated numerical approximation $u_n(\epsilon)$.

Theorem (G, Halpern, Ryan, Tran 2014)

The difference between the geometric mesh and fixed step mesh approximations satisfies for ε small

$$u_N(\epsilon) - u_N(0) = \alpha(aT, N)u_0\varepsilon^2 + o(\varepsilon^2), \textit{with}$$
$$\alpha(x, N) = \frac{N(N^2-1)}{24}\left(\frac{x/N}{1+x/N}\right)^2 (1+x/N)^{-N}.$$

Roundoff Error Estimates

For a given ϵ, the time parallel algorithm needs to solve $B\mathbf{u} = \mathbf{f}$ by solving $S\mathbf{g} = \mathbf{f}$, $(\frac{1}{\Delta t_n} + a)w_n = g_n$, $S^{-1}\mathbf{u} = \mathbf{w}$.

Theorem (G, Halpern, Ryan, Tran 2014)

Let $\mathbf{u}$ be the exact solution of $B\mathbf{u} = \mathbf{f}$, and $\hat{\mathbf{u}}$ be the computed solution by diagonalization. Then

$$\frac{\|\mathbf{u} - \hat{\mathbf{u}}\|_\infty}{\|\mathbf{u}\|_\infty} \lesssim macheps \, \frac{N^2(2N+1)(N+aT)}{\phi(N)} \, \varepsilon^{-(N-1)},$$

with

$$\phi(N) = \begin{cases} \frac{N}{2}!(\frac{N}{2} - 1)! & \text{if } N \text{ is even,} \\ (\frac{N-1}{2}!)^2 & \text{if } N \text{ is odd.} \end{cases}$$

The error of the direct time parallel solver at time T can be estimated by

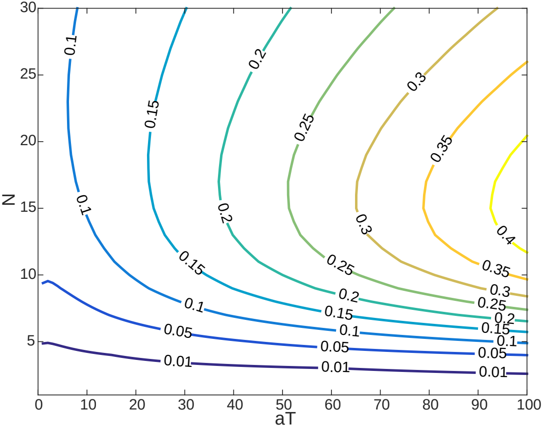
$$\frac{|e^{-aT}u_0 - \hat{u}_N|}{|u_0|} \leq \frac{|e^{-aT}u_0 - u_N(0)|}{|u_0|} + \frac{|u_N(0) - u_N(\epsilon)|}{|u_0|} + \frac{|u_N(\epsilon) - \hat{u}_N|}{|u_0|}.$$

Balancing Roundoff and Truncation Error

Theorem (Optimized geometric time mesh)

Roundoff and Truncation Errors are balanced if

$$\epsilon(aT, N) = \left(macheps \frac{N^2(2N+1)(N+aT)}{\phi(N)\alpha(aT,N)} \right)^{\frac{1}{N+1}} .$$

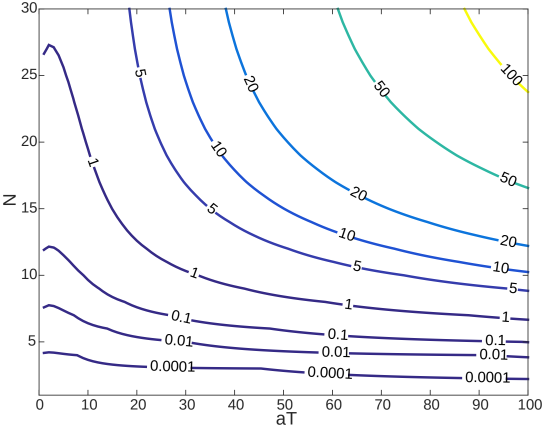


Potential for Parallelization

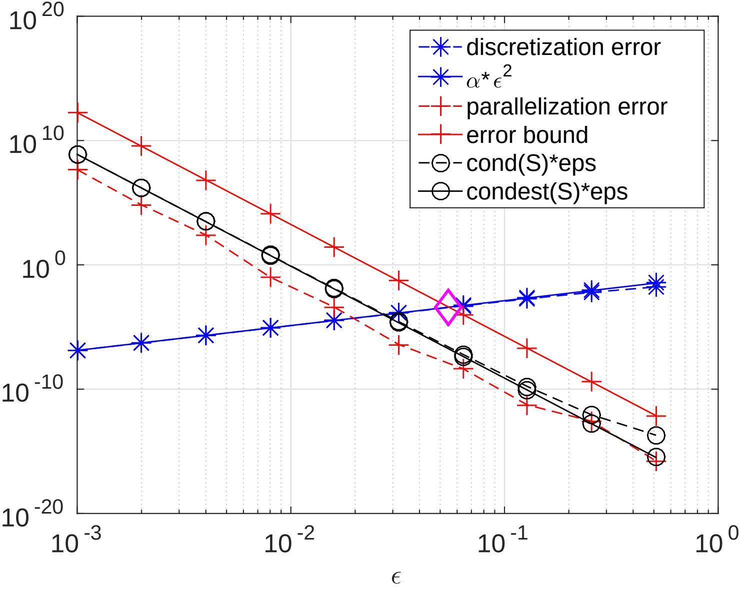
Using the optimized ϵ, solving

$$\frac{du}{dt} + au = 0, \quad t \in (0, T), \quad u(0) = u_0$$

with Backward Euler in parallel using N processors will increase the error by the factor:



ODE Numerical Experiment



Diagonalization for PDEs by Tensorisation

For example for the discretized heat equation

$$\frac{1}{\Delta t_n}(\mathbf{u}^n - \mathbf{u}^{n-1}) - \Delta_h \mathbf{u}^n = \mathbf{f}^n.$$

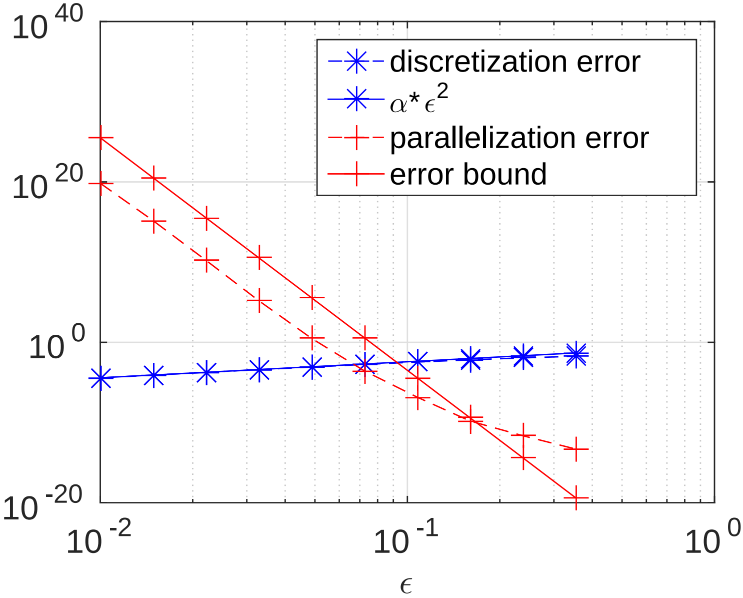
Setting $\mathbf{u} := (\mathbf{u}^1, \ldots, \mathbf{u}^N)$, $\mathbf{f} := (\mathbf{f}^1 + \frac{1}{\Delta t_1}\mathbf{u}_0, \mathbf{f}^2, \ldots, \mathbf{f}^N)$ and using the Kronecker symbol

$$(B \otimes I_x - I_t \otimes \Delta_h)\mathbf{u} = \mathbf{f}, \quad B := \begin{pmatrix} \frac{1}{\Delta t_1} & & & 0 \\ -\frac{1}{\Delta t_2} & \frac{1}{\Delta t_2} & & \\ 0 & \ddots & \ddots & \\ & & -\frac{1}{\Delta t_N} & \frac{1}{\Delta t_N} \end{pmatrix}.$$

If B is diagonalizable, $B = SDS^{-1}$, one can solve in 3 steps:

$$\begin{array}{lrcl} (a) & (S \otimes I_x)\mathbf{g} & = & \mathbf{f}, \\ (b) & (\frac{1}{\Delta t_n} - \Delta_h)\mathbf{w}^n & = & \mathbf{g}^n, \quad 1 \le n \le N, \\ (c) & (S^{-1} \otimes I_x)\mathbf{u} & = & \mathbf{w}. \end{array}$$

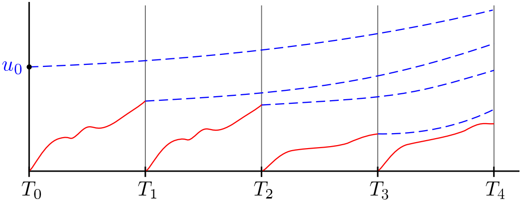
Heat Equation Numerical Experiment



G and Güttel 2012

For linear problems $\mathbf{u}'(t) = A\mathbf{u}(t) + \mathbf{g}(t), \quad \mathbf{u}(0) = \mathbf{u}_0$

ParaExp: use overlapping decomposition



Solve first non-overlapping inhomogeneous problems

$$\mathbf{v}_j'(t) = A\mathbf{v}_j(t) + \mathbf{g}(t), \quad \mathbf{v}_j(T_{j-1}) = \mathbf{0}, \quad t \in [T_{j-1}, T_j],$$

and then overlapping homogeneous problems

$$\mathbf{w}_j'(t) = A\mathbf{w}_j(t), \quad \mathbf{w}_j(T_{j-1}) = \mathbf{v}_{j-1}(T_{j-1}), \quad t \in [T_{j-1}, T]$$

The solution is then obtained by summation:

$$\mathbf{u}(t) = \mathbf{v}_k(t) + \sum_{j=1}^{k} \mathbf{w}_j(t) \quad \text{with } k \text{ such that} \quad t \in [T_{k-1}, T_k].$$

Wave Equation Experiment

$$\partial_{tt} u(t,x) = \alpha^2 \partial_{xx} u(t,x) + \text{hat}(x)\sin(2\pi f t) \qquad x,t \in (0,1)$$
$$u(t,0) = u(t,1) = u(0,x) = u'(0,x) = 0$$

α^2	f	serial		parallel			efficiency
		τ_0	error	$\max(\tau_1)$	$\max(\tau_2)$	error	
0.1	1	2.54e−01	3.64e−04	4.04e−02	1.48e−02	2.64e−04	58 %
0.1	5	1.20e+00	1.31e−04	1.99e−01	1.39e−02	1.47e−04	71 %
0.1	25	6.03e+00	4.70e−05	9.83e−01	1.38e−02	7.61e−05	76 %
1	1	7.30e−01	1.56e−04	1.19e−01	2.70e−02	1.02e−04	63 %
1	5	1.21e+00	4.09e−04	1.97e−01	2.70e−02	3.33e−04	68 %
1	25	6.08e+00	1.76e−04	9.85e−01	2.68e−02	1.15e−04	75 %
10	1	2.34e+00	6.12e−05	3.75e−01	6.31e−02	2.57e−05	67 %
10	5	2.31e+00	4.27e−04	3.73e−01	6.29e−02	2.40e−04	66 %
10	25	6.09e+00	4.98e−04	9.82e−01	6.22e−02	3.01e−04	73 %

$\Delta x = \frac{1}{101}$, $\Delta t_0 = \min\{5 \cdot 10^{-4}/\alpha, 1.5 \cdot 10^{-3}/f\}$, RK45 and Chebyshev exponential integrator, 8 processors

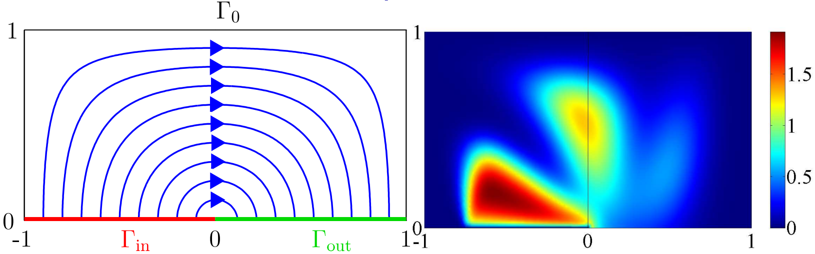
Heat Equation Experiment

$$\partial_t u(t,x) = \alpha \partial_{xx} u(t,x) + \text{hat}(x)\sin(2\pi f t) \qquad x,t \in (0,1)$$
$$u(t,0) = u(t,1) = 0$$
$$u(0,x) = 4x(1-x)$$

α	f	serial		parallel			effi-
		τ_0	error	$\max(\tau_1)$	$\max(\tau_2)$	error	ciency
0.01	1	4.97e−02	3.01e−04	1.58e−02	9.30e−03	2.17e−04	50 %
0.01	10	2.43e−01	4.14e−04	7.27e−02	9.28e−03	1.94e−04	74 %
0.01	100	2.43e+00	1.73e−04	7.19e−01	9.26e−03	5.68e−05	83 %
0.1	1	4.85e−01	2.24e−05	1.45e−01	9.31e−03	5.34e−06	79 %
0.1	10	4.86e−01	1.03e−04	1.45e−01	9.32e−03	9.68e−05	79 %
0.1	100	2.42e+00	1.29e−04	7.21e−01	9.24e−03	7.66e−05	83 %
1	1	4.86e+00	7.65e−08	1.45e+00	9.34e−03	1.78e−08	83 %
1	10	4.85e+00	8.15e−06	1.45e+00	9.33e−03	5.40e−07	83 %
1	100	4.85e+00	3.26e−05	1.44e+00	9.34e−03	2.02e−05	84 %

$\Delta x = \frac{1}{101}$, $\Delta t_0 = \min\{5 \cdot 10^{-4}/\alpha, 1.5 \cdot 10^{-3}/f\}$, RK45 and Chebyshev exponential integrator, 4 processors

Advection-Diffusion Popular Benchmark Problem



	equispaced time	with load balancing
τ_0	24.1 s	(23.7 + 7) s
serial error	1.2e−03	8.3e−04
$\min(\tau_1)$	2.6 s	2.6 s
$\max(\tau_1)$	7.7 s	4.9 s
$\mathrm{mean}(\tau_2)$	0.3 s	0.3 s
parallel err.	4.7e−04	3.1e−04
efficiency	36.9 %	58.3 %

8 processors, `ode15s`, restricted-denominator Arnoldi method (+7 for optimized time grid)

Applying ParaExp to a Nonlinear Problem ?

$$\mathbf{u}'(t) = A\mathbf{u}(t) + B(\mathbf{u}(t)) + \mathbf{g}(t), \quad t \in [0, T], \quad \mathbf{u}(0) = \mathbf{u}_0,$$

Step 1: Parallel nonlinear solves with zero initial data

$$\mathbf{v}_n'(t) = A\mathbf{v}_n(t) + B(\mathbf{v}_n(t)) + \mathbf{g}(t),\ t \in [T_{n-1}, T_n],\ \mathbf{v}_n(T_{n-1}) = 0.$$

Step 2: Parallel linear non homogeneous solves

$$\mathbf{w}_n'(t) = A\mathbf{w}_n(t), \quad t \in [T_{n-1}, T], \quad \mathbf{w}_n(T_{n-1}) = \mathbf{v}_{n-1}(T_{n-1}),$$

where $\mathbf{v}_0(T_0) = \mathbf{u}_0$.

Computing the linear combination

$$\mathbf{u}(t) = \mathbf{v}_n(t) + \sum_{j=1}^{n} \mathbf{w}_j(t), \quad t \in [T_{n-1}, T_n),$$

Does not lead to the solution of the nonlinear problem!

Borrow an idea from nonlinear PDE analysis

Separation of the solution of the nonlinear problem into

$$\mathbf{u}(t) = \mathbf{v}(t) + \mathbf{w}(t),$$

with $\mathbf{w}$ the solution of the linear problem with initial data $\mathbf{u}_0$

$$\mathbf{w}'(t) = A\mathbf{w}(t), \quad \mathbf{w}(t) = \mathbf{u}_0,$$

and $\mathbf{v}$, the solution of the nonlinear remaining part:

$$\mathbf{v}'(t) = A\mathbf{v}(t) + B(\mathbf{v}(t) + \mathbf{w}(t)) + \mathbf{g}(t), \quad \mathbf{v}(0) = 0.$$

Construction of a nonlinear ParaExp Algorithm

Start with some initial guesses $\mathbf{v}_n^0(T_n)$:

Iteration k: Initial conditions for the linear problems

$$\mathbf{W}_n^k = \begin{cases} \mathbf{u}_0, & n = 0, \\ \mathbf{v}_n^{k-1}(T_n), & n \geq 1. \end{cases}$$

Solve in parallel the linear problems

$$(\mathbf{w}_n^k)'(t) = A\mathbf{w}_n^k(t), \quad t \in [T_{n-1}, T], \quad \mathbf{w}_n^k(T_{n-1}) = \mathbf{W}_{n-1}^k.$$

Solve in parallel the nonlinear problems ($t \in [T_{n-1}, T_n]$)

$$(\mathbf{v}_n^k)'(t) = A\mathbf{v}_n^k(t) + B(\mathbf{v}_n^k(t) + \sum_{j=1}^{n} \mathbf{w}_j^k(t)) + \mathbf{g}(t), \quad \mathbf{v}_n^k(T_{n-1}) = 0.$$

The new approximate solution is then defined by

$$\mathbf{u}^k(t) = \mathbf{v}_n^k(t) + \sum_{j=1}^{n} \mathbf{w}_j^k(t), \quad t \in [T_{n-1}, T_n),$$

which satisfies the nonlinear equation on each time interval $[T_{n-1}, T_n)$ and $\mathbf{u}^k(0) = \mathbf{u}_0$.

Nonlinear ParaExp

Starting with $\mathbf{u}_n^0(T_n) = \mathbf{w}_j^0(T_n) = \mathbf{0}$,
Iteration k: Initial data for the linear problem:

$$\mathbf{W}_n^k = \begin{cases} \mathbf{u}_0, & n = 0, \\ \mathbf{u}_n^{k-1}(T_n) - \sum_{j=1}^n \mathbf{w}_j^{k-1}(T_n), & n \geq 1. \end{cases}$$

Solve in parallel the linear problems

$$(\mathbf{w}_n^k)'(t) = A\mathbf{w}_n^k(t), \quad t \in [T_{n-1}, T], \quad \mathbf{w}_n^k(T_{n-1}) = \mathbf{W}_{n-1}^k.$$

Solve in parallel the nonlinear problems $(t \in [T_{n-1}, T_n])$

$$(\mathbf{u}_n^k)'(t) = A\mathbf{u}_n^k(t) + B(\mathbf{u}_n^k(t)) + \mathbf{g}(t), \; \mathbf{u}_n^k(T_{n-1}) = \sum_{j=1}^n \mathbf{w}^k(T_{n-1})$$

Define the new approximate solution

$$\mathbf{u}^k(t) = \mathbf{u}_{n-1}^k(t), \quad t \in [T_{n-1}, T_n).$$

Convergence behavior of non-linear ParaExp

Theorem (G, Petcu 2015)

The approximate solution $\mathbf{u}^k$ *coincides with the exact solution* $\mathbf{u}$ *on the time interval* $[T_0, T_{k+1}]$.

Theorem (G, Güttel 2015)

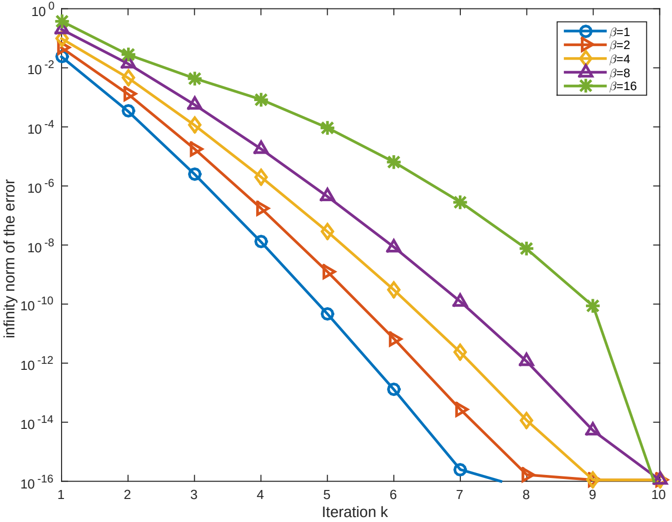
Nonlinear ParaExp is a parareal algorithm with coarse propagator given by a rational Krylov approximation of the linear part over long time intervals.

Further Results:

- Convergence analysis gives one additional power of ΔT provided one solves the linearized problem in the coarse propagator.
- Nonlinear ParaExp still converges if one only propagates the linear differential operator in the coarse propagator

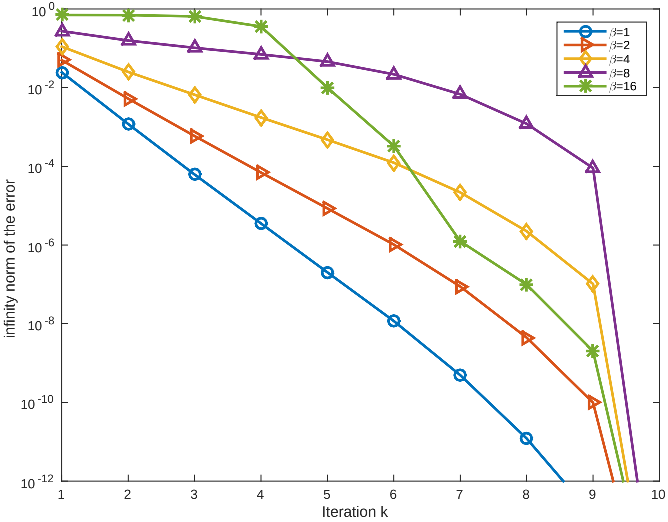
Numerical Experiments: Heat Equation

$u_t = \Delta u + \beta(u - u^3)$, $x \in (0, 1)$, $t \in (0, 0.1)$, 10 coarse time intervals



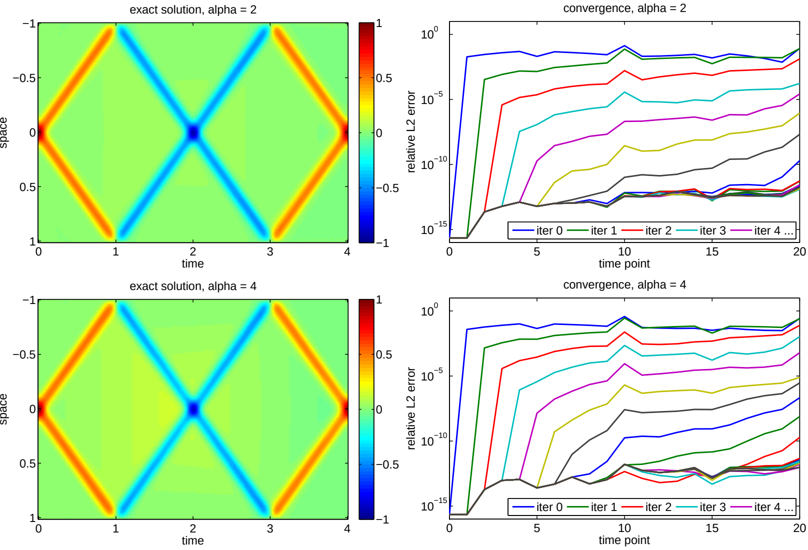
Numerical Experiments: Heat Equation

$u_t = \Delta u + \beta(u - u^3)$, $x \in (0,1)$, $t \in (0,1)$, 10 coarse time intervals



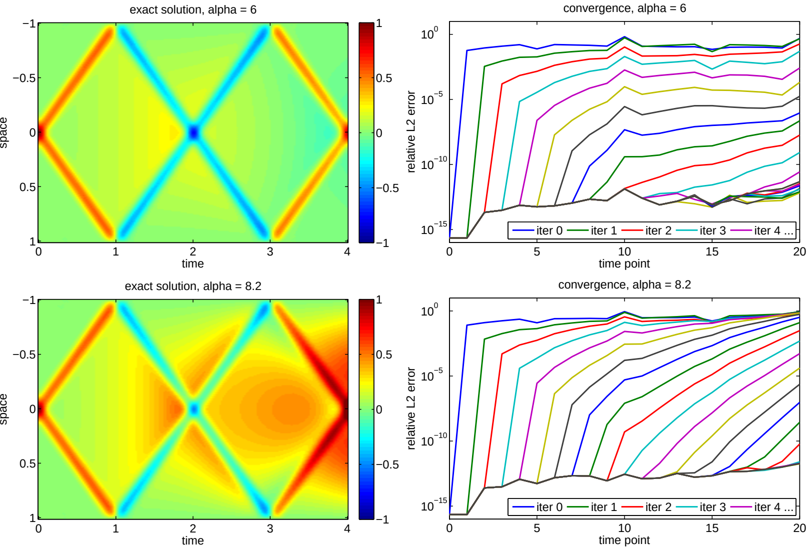
Numerical Experiments: Wave Equation

$u_{tt} = \Delta u + \alpha u^2$, $x \in (0,1)$, $t \in (0,4)$, 20 coarse time intervals



Numerical Experiments: Wave Equation

$u_{tt} = \Delta u + \alpha u^2$, $x \in (0,1)$, $t \in (0,4)$, 20 coarse time intervals



Summary

We have seen five Time Parallel Direct Solver Strategies:

- Small scale methods: Predictor Corrector, Block Methods, Parallel RK and RIDC
- Cyclic Reduction, also together with Waveform Relaxation
- Laplace Transform Methods
- Tensor based methods
- ParaExp based on rational Krylov propagation, and an iterative variant for non-linear problems

Preprints are available at www.unige.ch/∼gander