TUHH
Hamburg University of Technology

SIMILARITIES AND DIFFERENCES BETWEEN PARAREAL, PFASST AND TIME MULTIGRID

Thibaut Lunet & Martin J. Gander (University of Geneva) & Daniel Ruprecht (TUHH) & Robert Speck (Forschungszentrum Jülich GmbH)

Chair Computational Mathematics
Institute of Mathematics (E-10)
Hamburg University of Technology

PinT Workshop 2022

July 11, 2022

Context

- Many parallel-in-time (PinT) ideas over the last two decades
- Most known and used are iterative methods :
  - **Parareal** [Lions, Maday & Turinici, 2001]
    $\rightarrow$ 455 citations since 2001
  - **PFASST** [Emmett & Minion, 2012]
    $\rightarrow$ 237 citations since 2012
  - **MGRIT** [Falgout, Friedhoff et al. 2014 (orig. 2013)]
    $\rightarrow$ 252 citations since 2014
  - **Space-Time Multi-Grid (STMG)** [Gander & Neumüller, 2016; Horton & Vandewalle, 1995; Burmeister & Horton, 1991]
    $\rightarrow$ 127 citations since 2016

$\rightarrow$ *Many others PinT algorithms, but we* **focus** *on those 4*

Introduction

Common factors

- **Iterative methods** solving a (non-)linear system

$$(\text{algo}) : \boldsymbol{u}^k \mapsto \boldsymbol{u}^{k+1}, \quad \text{s.t} \quad \mathbf{A}(\boldsymbol{u}^\infty) = \boldsymbol{f},$$

- Based on a **particular multilevel approach**

Current questions

1. Analysis using **same problem** and **equivalent settings** ?
   e.g PFASST without Spectral Deferred Corrections ... ?
2. **Key differences** between those methods (when there is some ...) ?
3. Generic approach to get **error bounds** ?

$\Rightarrow$ *Toward a* **common framework** *...*

Objectives and outline

- Select a simple common problem (Dahlquist) :

$$\frac{du}{dt} = \lambda u, \quad \lambda \in \mathbb{C}, \quad t \in [0, T]$$

  $\Rightarrow$ linear, ignore any space dimension, no source term
- Restrict (for now) to two level forms
- Focus on Parareal (MGRIT F-relax.), PFASST and (S)TMG
- Only consider linear time integration methods

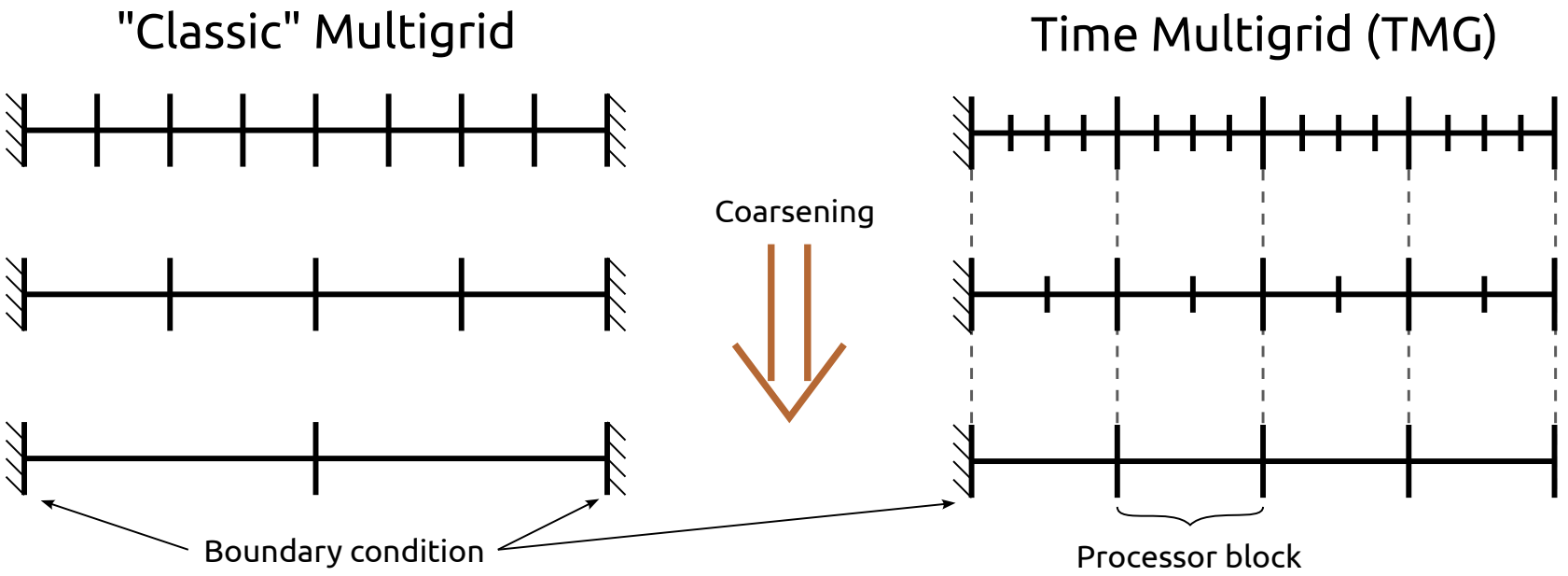
Disclaimer :

This is not a comparison of efficiency or quality of PinT algorithms !

Tentative description on a one-dimensional problem

$$\mathbf{A}\boldsymbol{u} = \boldsymbol{f}$$

Multi-level discretization of $\boldsymbol{u}$:



$\Rightarrow$ TMG combines "classic" multigrid on many blocks (one per processor)

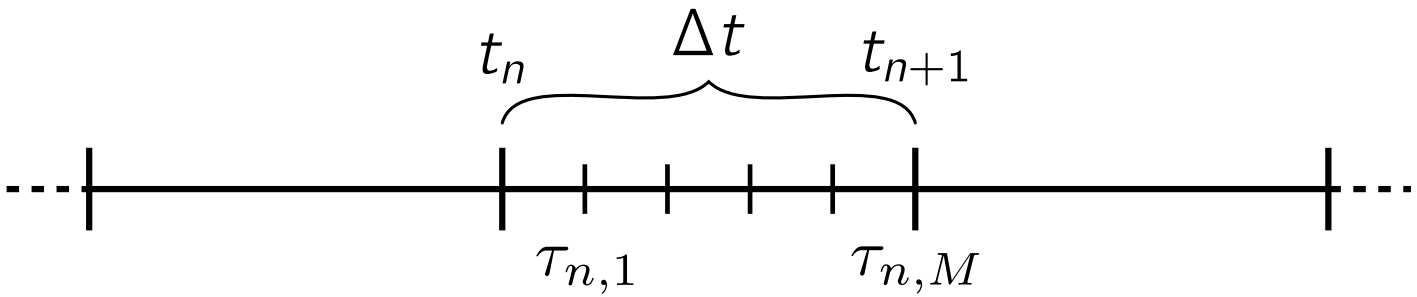
$\Rightarrow$ Write "global" multigrid operators as **block matrices**

Block decomposition

Hamburg University of Technology

Definition of a block

- $[0, T]$ decomposed into N sub-intervals $[t_n, t_{n+1}]$ of size Δt
- *Block* discretization: $\tau_{n,m} = t_n + \Delta t \tau_m, \quad m \in \{1, 2, ..., M\}$



Block variable

Numerical approximation of the solution on one block

$$\boldsymbol{u}_n = [u_{n,1}, u_{n,2}, ..., u_{n,M}]^T$$

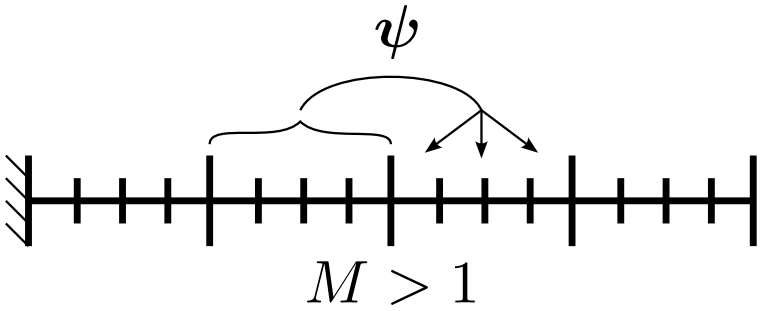
- Maximum block "coarsening" $\Rightarrow M = 1$, $\tau_1 = 1$, $\boldsymbol{u}_n$ is scalar
- Iterative PinT algorithms update block variables $\boldsymbol{u}_n^k$

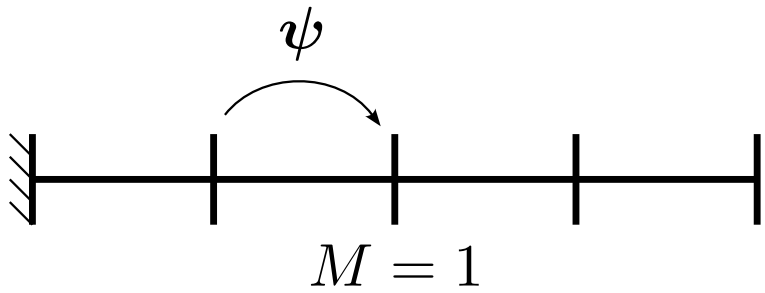
Block operators

Linear operators mapping two consecutive block variables

$$\phi(\boldsymbol{u}_{n+1}) = \boldsymbol{f}_n = \chi(\boldsymbol{u}_n) \quad \Leftrightarrow \quad \boldsymbol{u}_{n+1} = \phi^{-1}\chi\boldsymbol{u}_n := \psi(\boldsymbol{u}_n)$$

- ϕ : linear system solved on each block (implicit view)
- χ : produce right-hand-side from solution of previous block
- ψ : equivalent to a time propagator





Example of block operators

Runge-Kutta time integration

Stability function $R(z) \approx e^z$, ℓ equidistant time steps per block.

- *interface formulation* (natural approach): $M := 1$

$$\phi := R(\lambda \Delta t/\ell)^{-\ell}, \quad \chi := 1$$

- *volume formulation*: $M := \ell$, $\tau_m := m/\ell$, $r := R(\lambda \Delta t/\ell)^{-1}$

$$\phi := \begin{pmatrix} r & & \\ -1 & r & \\ & \ddots & \ddots \end{pmatrix}, \quad \chi := \begin{pmatrix} 0 & \dots & 0 & 1 \\ \vdots & & \vdots & 0 \\ \vdots & & \vdots & \vdots \end{pmatrix}$$

- Other volume formulations are possible
- Similar ideas for other time discretizations (collocation, multistep, ...)

Block coarsening

Two different approaches to define a coarse level

1. **Geometric coarsening** : coarser block discretization with $\tilde{M} < M$

$$\tilde{\phi}(\tilde{\boldsymbol{u}}_{n+1}) = \tilde{\chi}(\tilde{\boldsymbol{u}}_n), \quad \mathbf{T}_F^C : \mathbb{R}^M \mapsto \mathbb{R}^{\tilde{M}}, \quad \mathbf{T}_C^F : \mathbb{R}^{\tilde{M}} \mapsto \mathbb{R}^M$$

   $\Rightarrow$ explicit through block discretization

2. π-**coarsening** : block operator with less accuracy (cheaper)

$$\phi_{\boldsymbol{\Delta}}(\boldsymbol{u}_{n+1}) = \chi(\boldsymbol{u}_n)$$

   $\Rightarrow$ implicit through the block operator, **same block discretization !**

Important aspects

- π-coarsening can "hide" geometric coarsening

$$\phi := R(\lambda \Delta t/\ell)^{-\ell}, \quad \phi_{\mathbf{\Delta}} := R(\lambda \Delta t)^{-1}$$

$\rightarrow$ no need of transfer operators !

- Spectral Deferred Correction (SDC) is based on π-coarsening

$$\phi := I - \lambda \Delta t \mathbf{Q}, \quad \phi_{\mathbf{\Delta}} := I - \lambda \Delta t \mathbf{Q}_{\mathbf{\Delta}}$$

with $\mathbf{Q}$ the collocation matrix on M points
and $\mathbf{Q}_{\mathbf{\Delta}}$ a lower order approximation (Backward Euler, LU, ...)

- Both types of coarsening can be combined : $\tilde{\phi}_{\mathbf{\Delta}}$
e.g : PFASST uses a $\mathbf{Q}_{\mathbf{\Delta}}$ matrix on fewer nodes : $\tilde{\mathbf{Q}}_{\mathbf{\Delta}}$

Global problem for the numerical solution

Write the global system by combining the blocks (implicit form)

$$\mathbf{A}\boldsymbol{u} := \begin{pmatrix} \phi & & & \\ -\chi & \phi & & \\ & \ddots & \ddots & \\ & & -\chi & \phi \end{pmatrix} \begin{bmatrix} \boldsymbol{u}_1 \\ \boldsymbol{u}_2 \\ \vdots \\ \boldsymbol{u}_N \end{bmatrix} = \begin{bmatrix} \chi(u_0 \mathbb{1}) \\ 0 \\ \vdots \\ 0 \end{bmatrix} =: \boldsymbol{f}$$

$\rightarrow$ Use propagator ψ rather than ϕ (explicit form)

$$\phi^{-1}\mathbf{A}\boldsymbol{u} = \begin{pmatrix} \mathbf{I} & & & \\ -\psi & \mathbf{I} & & \\ & \ddots & \ddots & \\ & & -\psi & \mathbf{I} \end{pmatrix} \begin{bmatrix} \boldsymbol{u}_1 \\ \boldsymbol{u}_2 \\ \vdots \\ \boldsymbol{u}_N \end{bmatrix} = \begin{bmatrix} \psi(u_0 \mathbb{1}) \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

Those are different systems solving the same solution !

Preconditioned fixed-point iteration using π-coarsening

Classical representation for PARAREAL (explicit form)

$$\boldsymbol{u}^{k+1} = \boldsymbol{u}^k + \mathbf{M_E}^{-1}(\phi^{-1}\boldsymbol{f} - \phi^{-1}\mathbf{A}\boldsymbol{u}^k), \quad \mathbf{M_E} := \begin{pmatrix} \mathbf{I} & & \\ -\psi_{\mathbf{\Delta}} & \mathbf{I} & \\ & \ddots & \ddots \end{pmatrix}$$

Using the implicit form

$$\boldsymbol{u}^{k+1} = \boldsymbol{u}^k + \mathbf{M}^{-1}(\boldsymbol{f} - \mathbf{A}\boldsymbol{u}^k), \quad \mathbf{M} := \begin{pmatrix} \phi & & \\ -\phi\phi_{\mathbf{\Delta}}^{-1}\chi & \phi & \\ & \ddots & \ddots \end{pmatrix}$$

Update on one block :

$$\boldsymbol{u}_{n+1}^{k+1} = (\phi^{-1} - \phi_{\mathbf{\Delta}}^{-1})\chi\boldsymbol{u}_n^k + \phi_{\mathbf{\Delta}}^{-1}\chi\boldsymbol{u}_n^{k+1} = \mathcal{F}\boldsymbol{u}_n^k + \mathcal{G}\boldsymbol{u}_n^{k+1} - \mathcal{G}\boldsymbol{u}_n^k$$

Block iteration

Generic formula for a block solution update through iteration

$$\boldsymbol{u}_{n+1}^{k+1} = \mathbf{B}_1^0(\boldsymbol{u}_{n+1}^k) + \mathbf{B}_0^1\left(\boldsymbol{u}_n^{k+1}\right) + \mathbf{B}_0^0\left(\boldsymbol{u}_n^k\right) + \mathbf{B}_{-1}^0(\boldsymbol{u}_{n-1}^k) + \dots$$

e.g : PARAREAL only got $\mathbf{B}_0^1$ and $\mathbf{B}_0^0$

- All (generic) block operators (ϕ, χ, $\tilde{\phi}$, $\phi_{\boldsymbol{\Delta}}$, ...) are hidden in the **block iteration terms** $\mathbf{B}_j^i$
- Only requires a "consistency" condition
- **Primary block iteration** : block terms with $i \geq 0$ and $j \geq 0$
- Generic method to determine associated error bound
  $\rightarrow$ *cf. talk of Ausra*

$\Rightarrow$ **common framework** that could be used for other iterative methods

Two-level Time Multigrid

TUHH
Hamburg University of Technology

Apply the same approach from "classic" multigrid

- Define a coarse block level using geometric coarsening
- Smoothing operator on the fine level
- "Direct" solver on the coarse level

Generic two-level algorithm

1. ν_1 pre-relaxation steps with the smoother
2. restrict residuum to the coarse level
3. a coarse correction using with exact coarse solver
4. interpolate coarse solve to add correction on the fine level
5. ν_2 post-relaxation steps with the smoother

TMG smoothers

Block Jacobi relaxation

$$\boldsymbol{u}^{k+1} = \boldsymbol{u}^k + \omega \mathbf{D}^{-1}(\boldsymbol{f} - \mathbf{A}\boldsymbol{u}^k), \quad \mathbf{D} = \begin{pmatrix} \phi & & \\ & \phi & \\ & & \ddots \end{pmatrix}$$

$$\Rightarrow \quad \boldsymbol{u}_{n+1}^{k+1} = (1-\omega)\boldsymbol{u}_{n+1}^{k} + \omega \phi^{-1} \boldsymbol{\chi} \boldsymbol{u}_n^k$$

Combination with π-coarsening, $\omega = 1$

$\Rightarrow$ **Approximate Block Jacobi relaxation**

$$\boldsymbol{u}^{k+1} = \boldsymbol{u}^k + \mathbf{P}_{Jac}^{-1}(\boldsymbol{f} - \mathbf{A}\boldsymbol{u}^k), \quad \mathbf{P}_{Jac} = \begin{pmatrix} \phi_{\boldsymbol{\Delta}} & & \\ & \phi_{\boldsymbol{\Delta}} & \\ & & \ddots \end{pmatrix}$$

$$\Rightarrow \quad \boldsymbol{u}_{n+1}^{k+1} = \left[\mathbf{I} - \phi_{\boldsymbol{\Delta}}^{-1}\phi\right] \boldsymbol{u}_{n+1}^{k} + \phi_{\boldsymbol{\Delta}}^{-1} \boldsymbol{\chi} \boldsymbol{u}_n^k$$

Coarse grid problem

Use a coarse block discretization with $\tilde{M} < M$ and define

$$\tilde{\mathbf{A}}\tilde{\boldsymbol{u}} := \begin{pmatrix} \tilde{\phi} & & & \\ -\tilde{\chi} & \tilde{\phi} & & \\ & \ddots & \ddots & \\ & & \tilde{\chi} & \tilde{\phi} \end{pmatrix} \begin{bmatrix} \tilde{\boldsymbol{u}}_1 \\ \tilde{\boldsymbol{u}}_2 \\ \vdots \\ \tilde{\boldsymbol{u}}_N \end{bmatrix} = \begin{bmatrix} \mathbf{T}_F^C \chi(u_0 \mathbb{1}) \\ 0 \\ \vdots \\ 0 \end{bmatrix} =: \tilde{\boldsymbol{f}}$$

- $\mathbf{T}_F^C$: restriction operator
- $\mathbf{T}_C^F$: prolongation operator
- Global update formula :

$$\boldsymbol{u}^{k+1} = \boldsymbol{u}^k + \bar{\mathbf{T}}_C^F \tilde{\mathbf{A}}^{-1} \bar{\mathbf{T}}_F^C (\boldsymbol{f} - \mathbf{A}\boldsymbol{u}^k)$$

Block iteration for the coarse grid correction

- Assumption 1: $\mathbf{T}_F^C \mathbf{T}_C^F = \mathbf{I}$
- Assumption 2: $\boldsymbol{\Delta}_\chi = \mathbf{T}_F^C \chi - \tilde{\chi} \mathbf{T}_F^C = 0$
- Block iteration formula:

$$\boldsymbol{u}_{n+1}^{k+1} = (\mathbf{I} - \mathbf{T}_C^F \tilde{\phi}^{-1} \mathbf{T}_F^C \phi) \boldsymbol{u}_{n+1}^{k} + \mathbf{T}_C^F \tilde{\phi}^{-1} \mathbf{T}_F^C \chi \boldsymbol{u}_n^{k+1}$$

- Combines with π-coarsening $\Rightarrow$ **Approximate Coarse Correction** :

$$\boldsymbol{u}_{n+1}^{k+1} = (\mathbf{I} - \mathbf{T}_C^F \tilde{\phi}_{\boldsymbol{\Delta}}^{-1} \mathbf{T}_F^C \phi) \boldsymbol{u}_{n+1}^{k} + \mathbf{T}_C^F \tilde{\phi}_{\boldsymbol{\Delta}}^{-1} \mathbf{T}_F^C \chi \boldsymbol{u}_n^{k+1}$$

Writing STMG as a block iteration

Generic Time Multi-Grid (TMG)

Original algorithm uses discontinuous Galerkin in time

1. Relaxation with **block Jacobi smoother**
2. Coarse correction with **exact coarse grid operators**

Simplified TMG

- Assumption 1 and 2
- $\omega = 1$, $\nu_1 = 1$, $\nu_2 = 0$
- For two-level, the block iteration is

$$\boldsymbol{u}_{n+1}^{k+1} = \left(\phi^{-1}\chi - \mathbf{T}_C^F \tilde{\phi}^{-1} \mathbf{T}_F^C \chi\right) \boldsymbol{u}_n^k + \mathbf{T}_C^F \tilde{\phi}^{-1} \tilde{\chi} \mathbf{T}_F^C \boldsymbol{u}_n^{k+1}$$

$\Rightarrow$ definition of PARAREAL with geometric coarsening

Writing PFASST as a block iteration

TUHH
Hamburg University of Technology

Generic PFASST (only time dimension)

Original algorithm uses collocation as base integrator
Uses low-order RK method as approximate time integrator (SDC)

1. Relaxation with **approximate block Jacobi smoother** : $\phi_{\mathbf{\Delta}}$
2. Coarse correction with **approximate coarse grid operators** $\tilde{\phi}_{\mathbf{\Delta}}$

Simplified PFASST

- Assumption 1 and 2
- For two-level, the block iteration is

$$\begin{aligned}\boldsymbol{u}_{n+1}^{k+1} = &[\mathbf{I} - \mathbf{T}_C^F \tilde{\phi}_{\mathbf{\Delta}}^{-1} \mathbf{T}_F^C \phi](\mathbf{I} - \phi_{\mathbf{\Delta}}^{-1}\phi)\boldsymbol{u}_{n+1}^k \\ &+ (\mathbf{I} - \mathbf{T}_C^F \tilde{\phi}_{\mathbf{\Delta}}^{-1} \mathbf{T}_F^C \phi)\phi_{\mathbf{\Delta}}^{-\mathbf{1}} \chi \boldsymbol{u}_n^k + \mathbf{T}_C^F \tilde{\phi}_{\mathbf{\Delta}}^{-\mathbf{1}} \mathbf{T}_F^C \chi \boldsymbol{u}_n^{k+1}.\end{aligned}$$

$\Rightarrow$ primary block iteration with only non-zero terms

Elemental two-level TMG

Hamburg University of Technology

Correction \ Relaxation	Exact ϕ	Approximate ϕ_{Δ}
Exact $\tilde{\phi}$	TMG ($\omega = 1$)	TMG$_f$
Approximate $\tilde{\phi}_{\Delta}$	TMG$_c$	PFASST

- TMG ($\omega = 1$): $\sim$ Parareal written with geometric coarsening
- TMG$_c$: $\sim$ Parareal combining geometric **and** π-coarsening

$$\boldsymbol{u}_{n+1}^{k+1} = \left(\phi^{-1}\chi - \mathbf{T}_C^F \tilde{\phi}_{\Delta}^{-1} \mathbf{T}_F^C \chi\right) \boldsymbol{u}_n^k + \mathbf{T}_C^F \tilde{\phi}_{\Delta}^{-1} \tilde{\chi} \mathbf{T}_F^C \boldsymbol{u}_n^{k+1}$$

- TMG$_f$: something new, inspired by PFASST

Elemental two-level TMG

Hamburg University of Technology

Correction \ Relaxation	Exact ϕ	Approximate ϕ_{Δ}
Exact $\tilde{\phi}$	TMG ($\omega = 1$)	TMG_f
Approximate $\tilde{\phi}_{\Delta}$	TMG_c	PFASST

Collocation as base integrator

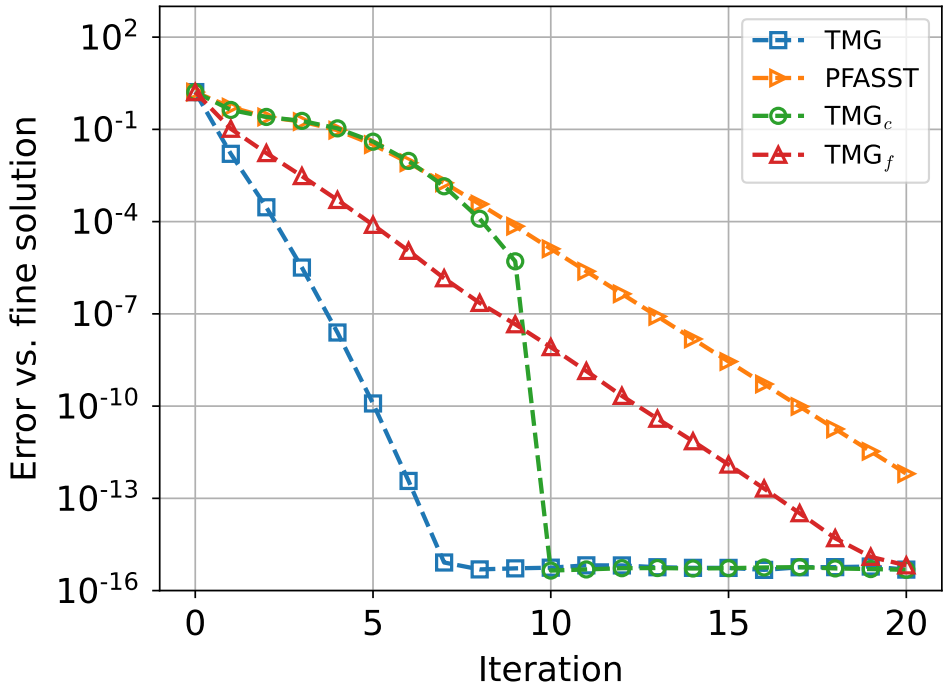
ϕ_{Δ} with Backward Euler

$M = 5,\ \tilde{M} = 3$

$\lambda = 2i - 0.2$

$T = 2\pi$

$N = 10$



Elemental two-level TMG

Hamburg University of Technology

Correction \ Relaxation	Exact ϕ	Approximate ϕ_{Δ}
Exact $\tilde{\phi}$	TMG ($\omega = 1$)	TMG_f
Approximate $\tilde{\phi}_{\Delta}$	TMG_c	PFASST

RK4 as base integrator

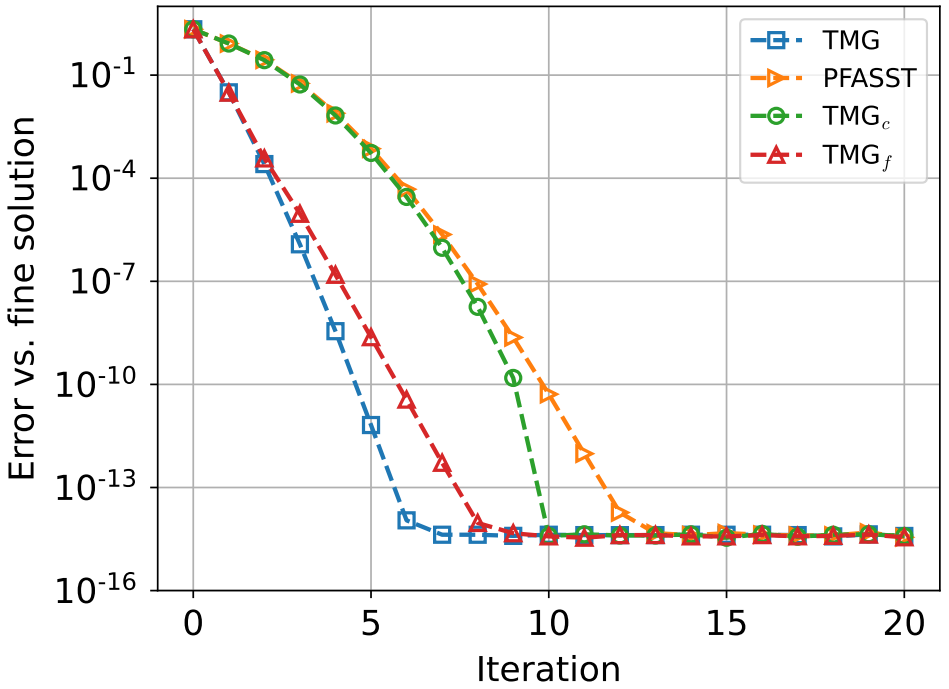
ϕ_{Δ} with RK2

$M = 5,\ \tilde{M} = 3$

$\lambda = 2i - 0.2$

$T = 2\pi$

$N = 10$



Conclusion

TUHH
Hamburg University of Technology

Warnings

- Those are not rightful comparison (performance → talk of Jens)
- Not limited to primary block iterations (three level, MGRIT-FCF, ...)
- Extensible to space-time, but issues with error bound accuracy ...

Main contributions

1. Common framework to describe iterative PinT algorithms
2. Generic method to obtain convergence bounds
3. Identification of differences and similarities between methods
4. Comparison with **common discretization and time integration**

→ small demonstration PYTHON code available online

Outlook

- Finish revision of the submitted manuscript[1]
- Combine error bounds with theoretical speedup analysis
  $\Rightarrow$ generic framework to compare iterative PinT
  (collaboration with Jens, Matthias and Stephanie @Wuppertal)
- Extension of the framework to more complex algorithms/problems
- Use to investigate new PinT algorithms with block iterations

[1] `https://arxiv.org/pdf/2203.16069.pdf`

Thanks for your attention

TUHH
Hamburg University of Technology

EuroHPC
Joint Undertaking

SPONSORED BY THE

Federal Ministry
of Education
and Research

This project has received funding from the European High-Performance Computing Joint Undertaking (JU) under grant agreement No 955701.
The JU receives support from the European Union's Horizon 2020 research and innovation programme and Belgium, France, Germany, and Switzerland.
This project also received funding from the German Federal Ministry of Education and Research (BMBF) grant 16HPC048.

UNIVERSITÉ
DE GENÈVE
FACULTÉ DES SCIENCES
Section de mathématiques

JÜLICH
Forschungszentrum

Error bound for primary block iteration

$$\boldsymbol{u}_{n+1}^{k+1} = \mathbf{B}_0^1(\boldsymbol{u}_{n+1}^k) + \mathbf{B}_1^0\left(\boldsymbol{u}_n^{k+1}\right) + \mathbf{B}_1^1\left(\boldsymbol{u}_n^k\right)$$

Generating function

- Double recurrence for the error $e_{n+1}^k := \left\|\boldsymbol{u}_{n+1}^k - \boldsymbol{u}_{n+1}\right\|$

$$e_{n+1}^{k+1} \leq \gamma e_{n+1}^k + \beta e_n^{k+1} + \alpha e_n^k$$

  with $\alpha := \left\|\mathbf{B}_1^1\right\|$, $\beta := \left\|\mathbf{B}_1^0\right\|$, $\gamma := \left\|\mathbf{B}_0^1\right\|$
- Solve it using the generating function $\rho_k(\zeta) = \sum_{n=0}^{\infty} e_{n+1}^k \zeta^{n+1}$
- Obtain a bound of the form

$$e_{n+1}^k \leq \theta_{n+1}^k(\alpha, \beta, \gamma)$$

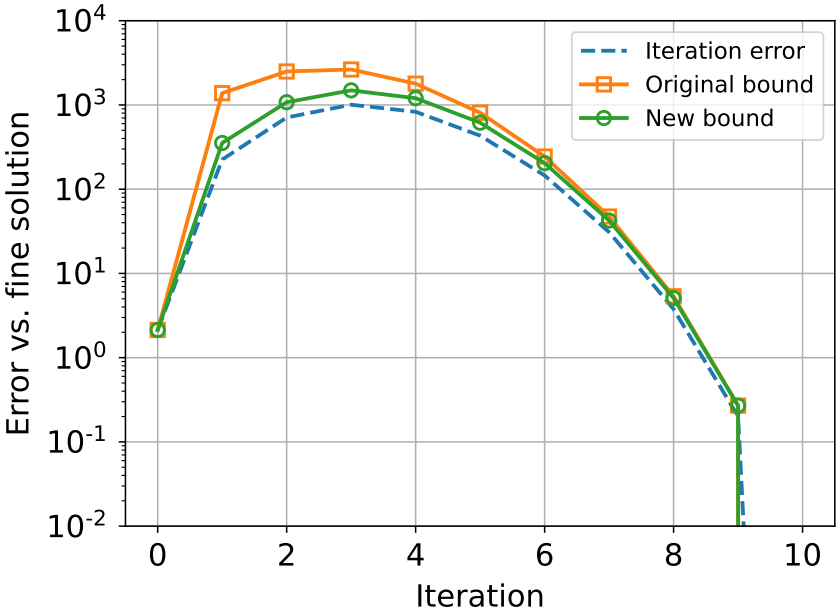
Error bound for primary block iteration

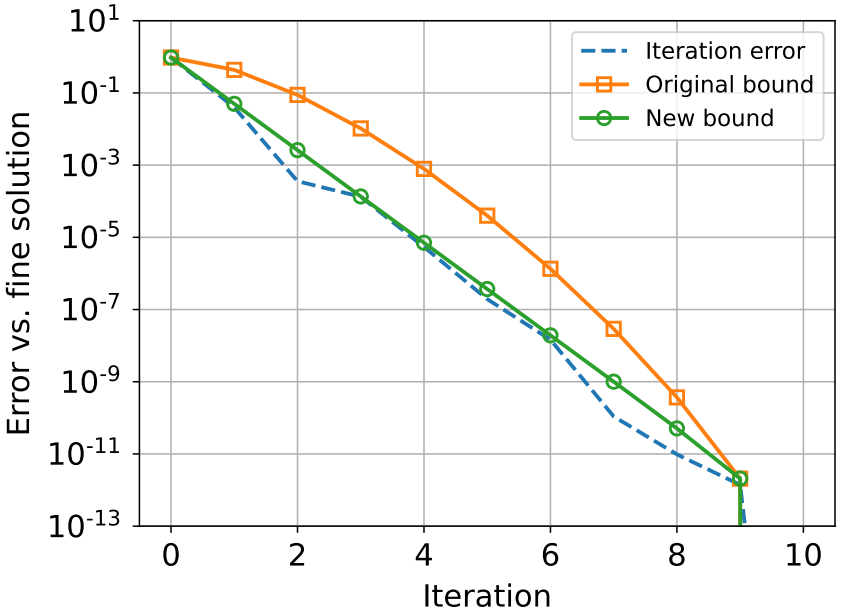
TUHH
Hamburg University of Technology

Example with Parareal

Comparison with the original bound from [Gander & Hairer, 2008]

- Applications with $\lambda \in \{4i, -4\}$





- Can show both linear and super-linear convergence
- Very close to numerical error when block operators are scalars