HPSF Conference 2025
Trilinos User-Developer Group (TUG) Meeting | Chicago, IL
May 5-8, 2025

Universität der Bundeswehr München

The Role of Trilinos in 4C: Advancing Coupled Multiphysics Simulations

Matthias Mayr[1,2]
[1]Institute for Mathematics and Computer-Based Simulation | University of the Bundeswehr Munich | Germany
[2]Data Science & Computing Lab | University of the Bundeswehr Munich | Germany

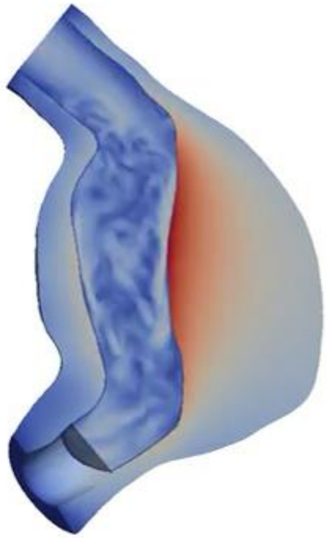
What is Trilinos?

I skip this. Others have already given a way better overview than me.

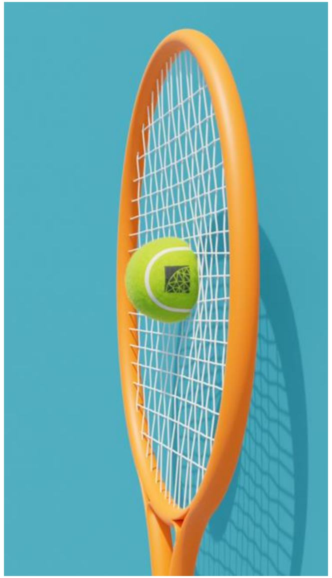
4C: A Comprehensive Multiphysics Simulation Framework

Comprehensive Computational Community Code (4C)

4C is a parallel multiphysics research code to analyze and solve a plethora of physical problems by means of numerical simulation and computational science.



4C
COMPREHENSIVE
COMPUTATIONAL
COMMUNITY
CODE



… enabled by Trilinos.

4C: Research philosophy and areas

Guiding principle

Software as a tool to develop and implement innovative numerical methods for research questions in coupled multiphysics systems driven by the needs of applications

Science

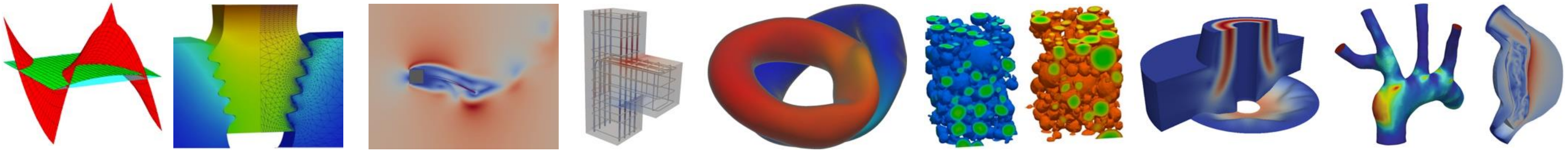
- Numerical methods & algorithms
- Material modeling
- Biopolymer networks
- Fluid/solid interaction
- Mixed-dimensional modeling
- …

Engineering

- Contact & tribology
- Aerospace engineering
- All-solid-state batteries
- Additive manufacturing
- Structural health monitoring of bridges
- ...

Biomedicine

- Respiratory system
- Cardiovascular system
- Musculoskeletal system
- Tumors
- Stomach
- Stents & stent grafts
- Growth & remodeling
- …



4C: Core components and capabilities

Single fields

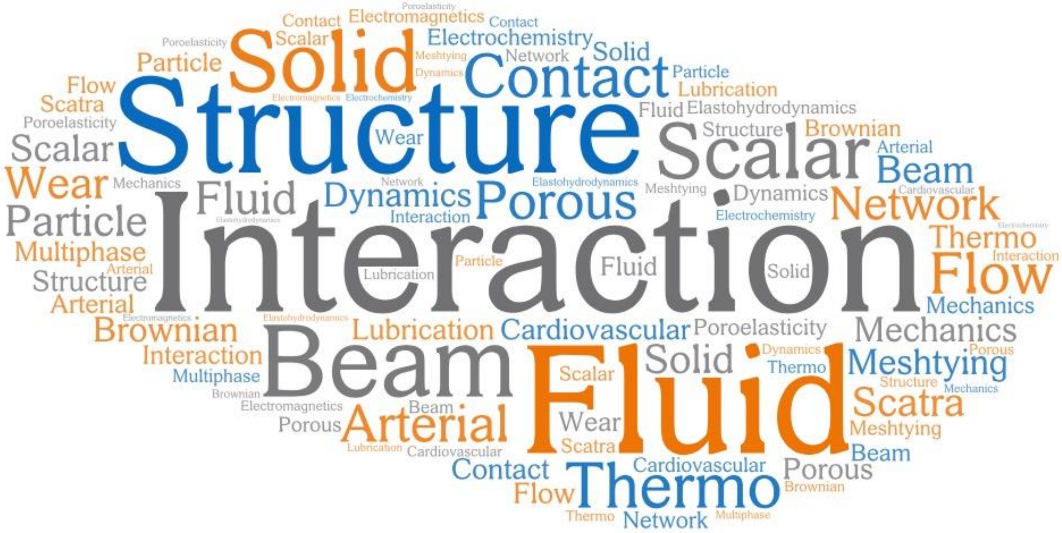
- Solid & structural mechanics
- Fluid flow
- Scalar transport & reactions
- Heat conduction

Coupled multiphysics

- Coupling algorithms (FBI, FSI, SSI, STI, TSI, ...)
- Mortar methods
- Block preconditioning

Numerical methods

- Finite Elements, IGA & CutFEM
- Particle algorithms
- Time integration
- Nonlinear solver
- Linear solvers & preconditioners



4C: Some technical aspects

- leverages the Trilinos project for sparse linear algebra, nonlinear solvers, and linear solvers and preconditioners
- fully implemented in C++
- parallelized with MPI for distributed memory hardware architectures
- open-source development process on GitHub (https://github.com/4C-multiphysics/4C)
  - jointly developed by several groups (TUM, UniBw M, Hereon)

Tool	Use case
Trilinos	sparse linear algebra, (non-)linear solvers, ...
ArborX	search algorithms
CMake & CTest	build & test system
MPI	parallelization
GitHub / git	collaboration, review, CI, version control
Doxygen	source code documentation
sphinx	documentation & user guide

4C: Why do we use Trilinos?

- Our expertise and core interest
  - Numerical modeling of multiphysics systems

Yes, we develop software but not for its own sake, but as a tool to study challenging multiphysics systems arising from applications.

Benefits of Trilinos

- Well-matured scientific software library suitable for
  - finite element discretization of PDEs
  - sparse linear algebra
  - iterative solvers & preconditioners
- Parallelism “for free”
  - on distributed memory architectures
  - using accelerators / modern computer architectures
- Scalability: physics solvers at scale ”out of the box”
- Continuous stream of updates and improvements “for free”

TRILINOS

4C

4C: How do we use Trilinos?

Updates and changes over time

- Linear solvers and preconditioners (Amesos →Amesos2, AztecOO → Belos, ML → MueLu, Teko)
- Update to utilities: Isoropia → Zoltan2
- Finite Element Technology: Intrepid2
- Nonlinear solvers (NOX)

Since the beginning (~2005)

- Sparse linear algebra (Epetra)
- Linear solvers and preconditioners (Amesos, AztecOO, Ifpack, ML)
- Utilities: Teuchos, Zoltan

In progress

- Epetra → Tpetra
- Ifpack → Ifpack2
- Stratimikos

4C: How do we integrate Trilinos?

Wrappers

- Currently mix of wrappers and direct use
- Transition to wrap EVERYTHING to
  - control the interface
  - ease transition to Tpetra

Releases

- Internal versioning scheme <year.counter> for tested git hashes to
  - decouple from Trilinos release schedule (esp. for our contributions to Trilinos)
  - allow asynchronous updates of Trilinos for each developer
- Mapping of git hash to <year.counter> version
- Forward-looking CI pipeline to prepare for upstream changes in develop

Installation

- Install other required software via spack
- Trilinos: manual build & installation w/ our selection of packages
- CMake integration into 4C

Software and development challenges

Challenges

- CMake / TriBITS configuration of Trilinos not straightforward for new users (PhD students)
- Documentation & tutorials sometimes spotty and outdated
- Contributing to Trilinos
  - We cannot view PR testing results
  - Testing requires pre-inspection

We are grateful for

- All existing packages and functionalities
- Structured deprecations and removals
- Help and comments on GitHub
- Personal contacts to Trilinos developers at Sandia

Co-development of 4C and Trilinos

Some characteristics of multiphysics systems

- Block matrices
- Properties vary across fields
- Ill-conditioned linear systems

Co-development

- Block preconditioning (Teko, MueLu)
- Algebraic multigrid methods (MueLu)

Utilities for block matrices

From 4C to Trilinos

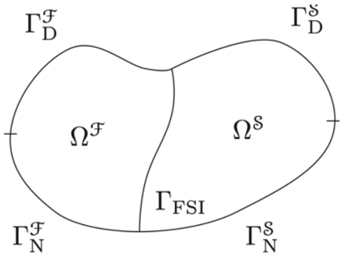
- Xpetra::Blocked[...]
- Xpetra::MapExtractor
- Some MueLu factories

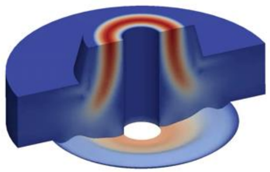
Use cases

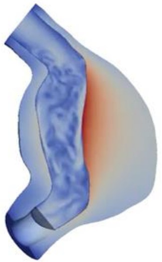
Fluid/solid interaction	Contact mechanics	Beam/solid interaction	...

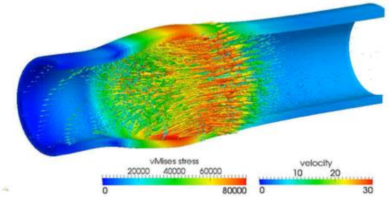
Case study: fluid/solid interaction (FSI)

- Deformable solid bodies interacting with fluid flow
  - Solid governed: elastodynamics
  - Fluid flow: incompressible Navier-Stokes equations
- Non-matching grids at FSI interface
  - Mortar discretization
  - Adaptive time stepping w/ a-posteriori error control





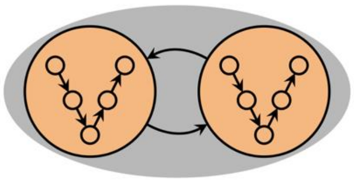


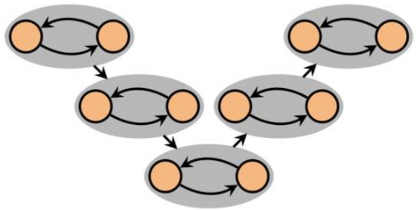


- Iterative solver with multilevel AMG preconditioning
  - 3x3 block system

$$\begin{bmatrix} \mathcal{S} & & \mathcal{C}^{\mathcal{S}\mathcal{F}} \\ & \mathcal{A} & \mathcal{C}^{\mathcal{G}\mathcal{F}} \\ \mathcal{C}^{\mathcal{F}\mathcal{S}} & \mathcal{C}^{\mathcal{F}\mathcal{G}} & \mathcal{F} \end{bmatrix} \begin{bmatrix} \Delta\mathbf{x}^{\mathcal{S}} \\ \Delta\mathbf{x}^{\mathcal{G}} \\ \Delta\mathbf{x}^{\mathcal{F}} \end{bmatrix} = - \begin{bmatrix} \mathbf{r}^{\mathcal{S}} \\ \mathbf{r}^{\mathcal{G}} \\ \mathbf{r}^{\mathcal{F}} \end{bmatrix}$$

  - Block-iterative vs. fully coupled AMG



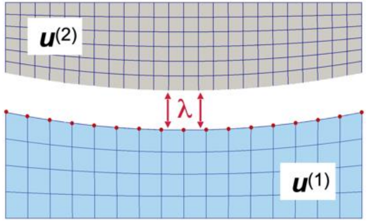


Mayr M, Klöppel T, Wall WA, Gee MW: A Temporal Consistent Monolithic Approach to Fluid--Structure Interaction Enabling Single Field Predictors, SIAM J. Sci. Comput., 37(1):B30-B59, 2015

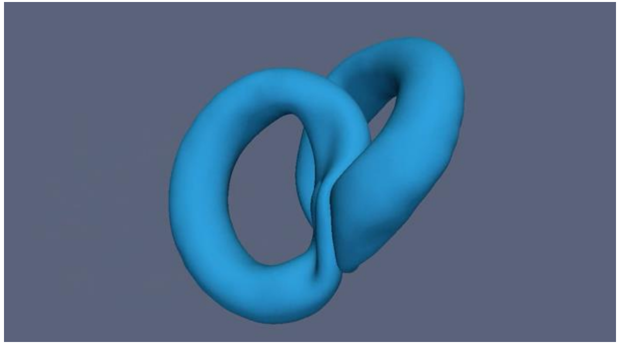
Mayr M, Wall WA, Gee MW: Adaptive time stepping for fluid-structure interaction solvers, Fin. Elements Anal. Design, 141:55-69, 2018

Case study: computational contact mechanics

- Contact of solid bodies
  - Solid governed by elastodynamics
  - Contact: Hertz-Signorini-Moreau conditions



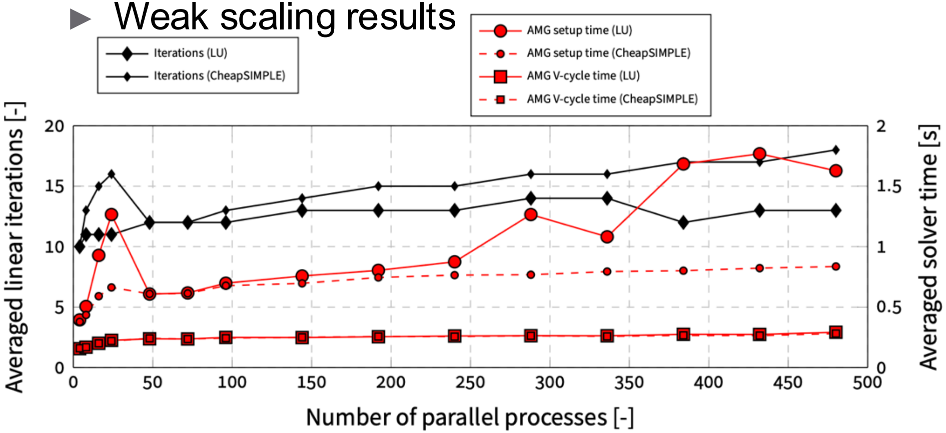
- Mortar discretization of Lagrange multipliers
  - Standard or dual shape functions
  - Dynamic load balancing (Zoltan2)



- Iterative solver with multilevel AMG preconditioning
  - 2x2 block system

$$\begin{pmatrix} \mathsf{K} & \mathsf{C}_1^\mathsf{T} \\ \mathsf{C}_2 & -\mathsf{Z} \end{pmatrix} \begin{bmatrix} \Delta \boldsymbol{u} \\ \Delta \boldsymbol{\lambda} \end{bmatrix} = - \begin{bmatrix} \mathbf{r}^{\boldsymbol{u}} \\ \mathbf{r}^{\boldsymbol{\lambda}} \end{bmatrix}$$

  - Schur complement preconditioner tailored to specifics of saddle point system
  - Weak scaling results

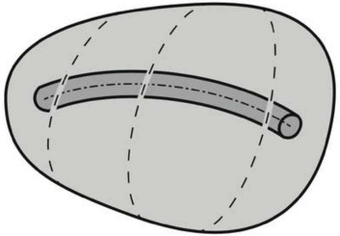


Mayr M, Popp A: Scalable computational kernels for mortar finite element methods, Engng. Comput., 39(5):3691-3720, 2023

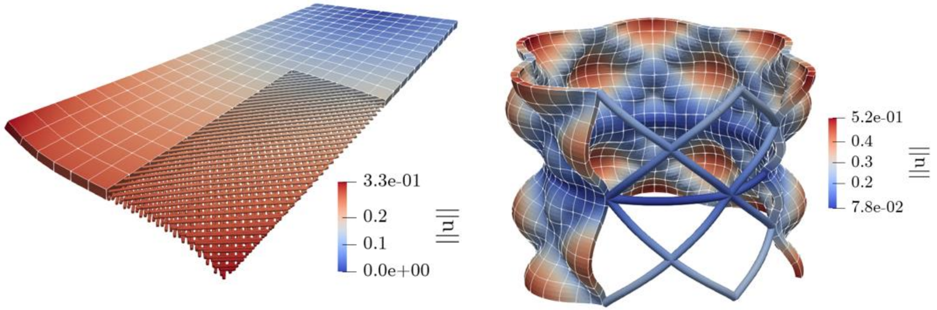
Wiesner TA, Mayr M, Popp A, Gee MW, Wall WA: Algebraic multigrid methods for saddle point systems arising from mortar contact formulations, Int. J. Numer. Meth. Engng., 122(15):3749-3779, 2021

Case study: modeling and solvers for fiber/solid coupling

- Slender fibers embedded in solid continua
  - Solid governed by elastodynamics
  - Fibers modeled as geometrically exact beams



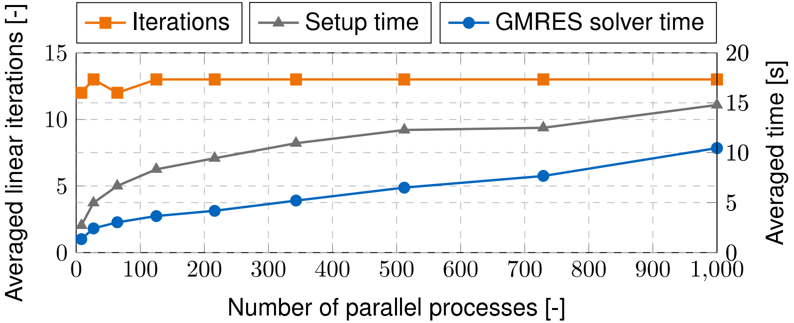
- Mixed-dimensional coupling along beam centerline
  - Mortar-like discretization
  - Penalty regularization



- Iterative solver with multilevel AMG preconditioning
  - 2x2 block system

$$\begin{bmatrix} \mathbf{K}_{BB} + \epsilon \mathbf{D}^T \kappa^{-1} \mathbf{D} & -\epsilon \mathbf{D}^T \kappa^{-1} \mathbf{M} \\ -\epsilon \mathbf{M}^T \kappa^{-1} \mathbf{D} & \mathbf{K}_{SS} + \epsilon \mathbf{M}^T \kappa^{-1} \mathbf{M} \end{bmatrix} \begin{bmatrix} \Delta \mathbf{d}_B \\ \Delta \mathbf{d}_S \end{bmatrix} = \begin{bmatrix} -\mathbf{f}_B^{\text{int}}(\mathbf{d}_B) + \mathbf{f}_B^{\text{ext}} - \mathbf{f}_B^{\text{c}}(\mathbf{d}_S, \mathbf{d}_B) \\ -\mathbf{f}_S^{\text{int}}(\mathbf{d}_S) + \mathbf{f}_S^{\text{ext}} - \mathbf{f}_S^{\text{c}}(\mathbf{d}_S, \mathbf{d}_B) \end{bmatrix}$$

  - Approximate block-LU preconditioner tailored to specifics of beam/solid coupling
  - Weak scaling results



Firmbach M, Steinbrecher I, Popp A, Mayr M: An approximate block factorization preconditioner for mixed-dimensional beam-solid interaction, Comp. Meth. Appl. Mech. Engng., 431: 117256, 2024

Concluding remarks

- Trilinos ...
  - ... comes with some challenges for us
  - ... facilitates parallel computing in 4C
  - ... brings AMG preconditioners to 4C
  - ... enables research at the intersection of coupled multiphysics and preconditioning

- 4C ...
  - ... has used Trilinos from its inception
  - ... is in the process of migrating to Tpetra
  - ... will remain a Trilinos user & contributor
  - ... is only possible due to Trilinos

Funding

dtec.bw - Digitalization and Technology Research Center of the Bundeswehr. dtec.bw is funded by the European Union – NextGenerationEU.

Zentrum für Digitalisierungs- und Technologieforschung der Bundeswehr

Finanziert von der Europäischen Union
NextGenerationEU

Contact

- matthias.mayr@unibw.de
- https://www.unibw.de/imcs-en
- GitHub: @mayrmt

Thank you.