A novel microfluidic framework introduced for low capillary number moving interface

Mahmoud Aboukhedr

Micro-droplet formation and mobilization is an emerging area of research due to its wide-ranging usage within microfluidics application relevant to tight porous media [1]. Physical understanding of this process at the pore scale is helpful in exploring macroscopic phenomena in oil and gas recovery and also forms the basis of many potential applications such as synthesis of new materials, formulation of products in pharmaceutical, cosmetics and food industries. The standard two-phase Volume Of Fluid (VOF) and/or Level Set (LS) methods for interface tracking between two immiscible fluids, have been used extensively in the literature to perform numerical simulations of droplet formation in different micromodels [2]. These methods are less successful for the study of phenomena when the combined effect of viscous and capillary forces is dominant. This is mainly due to difficulties in predicting the flow in presence of high capillary forces encountered in flows at the pore scale, which can introduce non-physical (parasitic) velocities or instabilities in the numerical results. In the present work, a novel dynamic VOF-based framework for reducing parasitic velocities at low capillary number flows based on [3, 4] work. These frameworks are compared with the commonly used for the prediction of relaxation of static droplets with varying sizes and they are also utilized to study the dynamics of oil droplet trapping and mobilization within pore throats through a T-junction configuration. The overall numerical model development has been conducted within the open-source, CFD Toolbox, OpenFOAM (version 2.3.x). The NavierStokes equations are discretized using a finite volume approach, while the Volume Of Fluid method is used to capture the location of interfaces. The volume fraction () is advected implicitly and additional algorithms for compressing, sharpening and smoothing the interface are implemented in order to guarantee its physical thickness and curvature. Surface tension forces are then computed based on the sharp/smooth () field which maintains a steep transition area for capillary pressure. The influence of the static/equilibrium contact angle imposed at the rigid wall is also considered in order to account for various wettability effects. The developed numerical method targets efficient modelling of multiphase flows in the micro-scale with complex interface motion and irregular solid boundaries. It allows correcting surface forces and permits simulations at low capillary numbers dampening out non-physical velocities. Moreover, the addition of the adaptive grid refinement method and the implicit formulation for the equation of volume fraction, allows larger time steps and thus lower computational cost. Overall, the comparison of computational results with analytical models and experiments shows that the introduced framework improves the accuracy of the original VOF method when the surface tension influence is predominant. Some indicative results for a static droplet case (150 micron) are depicted in Figures 1 and 2 below.

FEM approximation of Two-Phase Navier–Stokes Flow using DUNE-FEM

Marco Agnese[*]

April 18, 2016

We propose a novel fitted finite element method for two-phase Navier–Stokes flow problems that uses piecewise linear finite elements to approximate the moving interface. The meshes describing the discrete interface in general do not deteriorate in time, which means that in numerical simulations a smoothing or a remeshing of the interface mesh is not necessary. We present several numerical experiments for our numerical method, which demonstrate the accuracy and robustness of the proposed algorithm The discretization is implemented within the DUNE-FEM framework.

References

[1] M. Agnese, R. Nürnberg, Internat. J. Numer. Methods Fluids, *Fitted finite element discretization of two–phase Stokes flow*,(2016), to appear.

[*] Imperial College London, m.agnese13@imperial.ac.uk

A Weak Compatibility Condition for Newest Vertex Bisection

Martin Alkämper*

March 15, 2016

Newest Vertex Bisection (NVB) refinement on n-dimensional grids requires a compatibility or matching condition which implies that shared sub-simplices match, when refined from different sides. The condition was introduced by Stevenson in [1]. It has been proven that any 2-dimensional grid can be renumbered such that this condition is fulfilled, but for higher dimension this is an open problem. We introduce a new weaker compatibility condition and show that this weak condition yields termination of the iterative NVB refinement algorithm. Additionally we construct an algorithm based on generalized Kuhn-cubes to renumber the elements of a 3-dimensional grid such that the condition is fulfilled. In contrast to the strong condition with this weak condition not every uniform refinement is conforming, but instead every n-th uniform refinement is conforming. Also a direct bound on the effort of the conforming closure is still an open problem.

References

[1] Stevenson, Rob, he completion of locally refined simplicial partitions created by bisection, *Mathematics of Computation* **77** (2008), 227–241.

*University of Stuttgart, martin.alkaemper@mathematik.uni-stuttgart.de

A Parallel Multigrid Matrix-Free Solver using Schwarz Smoothers

Daniel Arndt[*] Pablo Lucero[*] Julius Witte [*] Patrick Esser [*]
Guido Kanschat[*]

March 15, 2016

We present a memory efficient and parallel framework for a finite element operator application implemented in the generic open-source library deal.II. In particular, the solver uses a geometric multigrid preconditioner with additive Schwarz smoothers. Instead of assembling a sparse matrix, a matrix-free approach is used for the global problem. Only a few patch-based matrices are assembled in order to smooth the residual of our multigrid preconditioner. Our implementation is parallelized on two levels to exploit modern supercomputer architecture in an optimal way: MPI over remote nodes on distributed meshes and thread parallelization with dynamic task scheduling within the nodes. Dedicated data structures are designed for high performance and to keep the memory requirements to a minimum. The framework handles adaptively refined meshes and systems of partial differential equations. We provide performance tests which show that our patch-based implementation is faster than sparse matrix-vector products and uses considerably less memory resources.

[*] Interdisciplinary Center for Scientific Computing, Ruprecht-Karls-Universität Heidelberg,
daniel.arndt@iwr.uni-heidelberg.de
pablo.lucero@iwr.uni-heidelberg.de
julius.witte@iwr.uni-heidelberg.de
patrick.esser@iwr.uni-heidelberg.de
kanschat@uni-heidelberg.de

A Trefftz polynomial space-time discontinuous Galerkin method for the second order wave equation

Lehel Banjai [*] Emmanuil Geogoulis [†] Oluwaseun Lijoka [‡]

April 18, 2016

We present a new time-space discontinuous Galerkin (dG) method for one field wave equation which utilizes special Trefftz-type basis functions. The method is motivated by the class of interior penalty discontinuous Galerkin (IPDG), together with the classical work of Hulbert and Hughes [4]. The existence and uniqueness of the scheme is proved via energy argument together with a special continuity of the bilinear form. We also include the idea of directional Trefftz-type basis functions based on the paper[6] with the purpose of reducing the number of degrees of freedom per element in the approximation. Numerical experiments highlighting the performance of the method compared with standard polynomial spaces are presented.

References

[1] A. Maciąg, and J.Wauer, Solution of the two-dimensional wave equation by using wave polynomials, *Journal of Engineering Mathematics 339–350, 2005.*

[2] S. Petersen, and C. Farhat and R. Tezaur A space-time discontinuous Galerkin method for the solution of the wave equation in the time domain, *International Journal for Numerical Methods in Engineering, 275–295, 2009.*

[3] A. Moiola, R. Hiptmair, and I. Perugia, Plane wave approximation of homogeneous Helmholtz solutions, *Journal of Applied Mathematics and Physics. Journal de Mathématiques et de Physique Appliquées, 809–837, 2011.*

[4] , G. M. Hulbert, and T. J. R. Hughes, Space-time finite element methods for second-order hyperbolic equations, *Computer Methods in Applied Mechanics and Engineering, 327–348, 1990.*

[*] Heriot Watt University, Edinburgh `l.banjai@hw.ac.uk`

[†] University of Leceister, UK., `Emmanuil.Georgoulis@le.ac.uk`

[‡] Heriot Watt University, Edinburgh `ofl1@hw.ac.uk`

[5] P. Monk and G. R. Richter, A discontinuous Galerkin method for linear symmetric hyperbolic systems in inhomogeneous media, *Journal of Scientific Computing, 443–477, 2005.*

[6] O. Cessanat and B. Despres, Application of an ultra weak variational formulation of elliptic PDEs to the two-dimensional Helmholtz problem, *SIAM Journal on Numerical Analysis, 255–299,1998*

Scientific Software and Linux Distributions

Ansgar Burchardt*

April 21, 2016

For scientific software such as frameworks for solving PDEs, the actual implementation is only part of becoming a successful project. Another important aspect is making the software accessible for users.

On Linux distributions, users expect to be able to easily install the software they daily use via the distribution's software management system. However, the installation instructions for many scientific software packages expect the user to install the software from source. While this is the only general method for installation, it is often a hard first task for users, especially when a complex dependency tree is involved.

In this talk, I will give a brief overview how software in Linux distributions is managed, the typical life-cycle of distributions, what distributions can and cannot provide for the scientific software community, and how developers of scientific software can help making packaging their software easier.

The talk will for the most part concentrate on distributions that provide pre-compiled binaries like Debian or Ubuntu, with some considerations for source-based distributions, and on libraries and applications written in C or C++.

*Institut für Numerische Mathematik, TU Dresden, `Ansgar.Burchardt@tu-dresden.de`

A generic implementation of the Virtual Element Method

Andrea Cangiani [*] Andreas Dedner [†] Ganesh Diwan [‡]
Martin Nolte [§]

April 23, 2016

We present the generic implementation approach of the conforming Virtual Element Method (VEM) introduced in [1]. The VEM can be seen as a generalised FEM with its key feature being the ability to construct high order approximation spaces with arbitrary degree of global regularity on meshes made of general polytopes (polygons/polyhedra). The VEM elemental spaces contain a space of (physical frame) polynomials, ultimately responsible for the accuracy of the method, as well as a complementary space of implicitly defined *virtual* functions. In contrast to other existing FEM extensions to polygonal/polyhedral meshes, the virtual functions are never required to be computed explicitly but rather are (partially) accessed through their carefully chosen degrees of freedom. As a result, the implementation complexity of the VEM is comparable to that of FEM. We conclude with some numerical experiments basing our implementation on the DUNE-FEM framework.

References

[1] L. Beirão da Veiga, F. Brezzi, A. Cangiani, G. Manzini, L. D. Marini, A. Russo. ”Basic principles of virtual element methods.” *Mathematical Models and Methods in Applied Sciences*, **23**, no. 01, (2013): 199–214.

[2] A. Dedner, R. Klöfkorn, M. Nolte, M. Ohlberger. ”A generic interface for parallel and adaptive discretization schemes: abstraction principles and the DUNE-FEM module.” *Computing*, **90**, no. 3-4 (2010): 165–196.

[*] University of Leicester, andrea.cangiani@le.ac.uk
[†] University of Warwick, a.s.dedner@warwick.ac.uk
[‡] University of Leicester, gcdiwan@gmail.com
[§] University of Freiburg, nolte@mathematik.uni-freiburg.de

Nektar++: An open-source spectral/hp element framework

Chris Cantwell[*] David Moxey[†] Douglas Serson[‡]
Mike Kirby[§] Spencer Sherwin[¶]

Spectral/hp element methods exhibit the numerical convergence properties of spectral methods while retaining the geometric flexibility of conventional finite element methods. They lead to compact data structures and potentially more efficient use of memory bandwidth and processor cache than low-order methods, and therefore can push the boundaries of large-scale complex simulations. However, they are inherently complex to implement which creates a barrier to their adoption by academia and industry.

Nektar++ is a tensor-product based finite element software package designed to allow one to construct both efficient classical low-order h-type solvers and higher-order p-type solvers by encapsulating the complexities of the discretisation. We initially outline the library design, along with our development strategy, pre- and post-processing tools and some examples of the application areas being supported, including Formula 1, flow stability and cardiac electrophysiology. We then discuss some recent code developments, including a formulation for reaction-diffusion problems on manifolds, and performance improvements from using hybrid spectral/hp-Fourier parallelisation and element amalgamation, intended to help address large-scale aerodynamics and biomedical modelling challenges.

[*] Imperial College London, c.cantwell@imperial.ac.uk
[†] Imperial College London, d.moxey@imperial.ac.uk
[‡] Imperial College London, d.serson14@imperial.ac.uk
[§] University of Utah, m.kirby@sci.utah.edu
[¶] Imperial College London, s.sherwin@imperial.ac.uk

Fast solvers for 3D elastodynamic Boundary Element Methods

Stéphanie Chaillat-Loseille[*]

May 13, 2016

The main advantage of the Boundary Element Method (BEM) [1] is that only the domain boundaries are discretized leading to a reduction of the number of degrees of freedom. In traditional BE implementation the dimensional advantage with respect to domain discretization methods is offset by the fully-populated nature of the BEM matrix. This presentation will provide an overview of recent works on improving the efficiency of the BEM to study elastic wave propagation problems in large domains.

In a first part, I will show the principle of the Fast Multipole Method in 3D elastodynamics and visco-elastodynamics to speed up the solution of the BEM [3].

The FM-BEM is intrinsically based on an iterative solver. In 3D elastodynamics, it is efficient but the number of iterations can significantly hinder the overall efficiency. In a second part, I will present an analytic preconditioner: a clever integral representation of the scattered field which naturally incorporates a regularizing operator. When considering Dirichlet boundary value problems, the regularizing operator is a high-frequency approximation to the Dirichlet-to-Neumann operator [2].

Finally, I will show an alternative to the FM-BEM. Using the H-matrix arithmetic and low-rank approximations, we derive a fast direct solver. I will assess the numerical efficiency and accuracy for elastic waves.

References

[1] M. Bonnet, *Boundary integral equations methods in solids and fluids*, Wiley, 1999.

[2] S. Chaillat, M. Darbas, F. Le Louër, Approximate local Dirichlet-to-Neumann map for three-dimensional time-harmonic elastic waves, *Computer Methods in Applied Mechanics and Engineering* (2015), 62-83.

[3] S. Chaillat, M. Bonnet, Recent advances on the fast multipole accelerated boundary element method for 3D time-harmonic elastodynamics, *Wave Motion* (2013), 1090-1104.

[*] Laboratoire POEMS (CNRS-ENSTA-INRIA), Université Paris Saclay, France, stephanie.chaillat@ensta-paristech.fr

A multiscale computational framework for the simulation of calcium cycling

Nagaiah Chamakuri[*] Wilhelm Neubert[†] Stephen Gilbert[‡]
Janine Vierheller[§] Martin Falcke[¶]

April 15, 2016

We develop a multiphysics and multiscale computational framework for the simulation of intracellular calcium dynamics in ventricular cardiac myocytes. Mathematical modeling of excitation-contraction coupling (ECC) is a multiscale problem, and it is therefore difficult to develop spatially detailed simulation tools. The set of reaction diffusion equations describe the behavior of the intracellular calcium concentrations in cytoplasm as well as in the sarcoplasmic reticulum domain. We adopted a detailed calcium release unit (CRU) model to describe the source functions in the PDEs and the dynamics of the membrane potential is based on a physiological ionic model. A novel computational approach to deal with the small time step sizes during the stochastic CRU dynamics is presented.

Using highly unstructured meshes, our method bridges many orders of magnitude to represent accurately the calcium distribution from the single channel to a complete cell with few hundreds of CRUs. The implementation based on different time integration strategies to solve the PDEs, the stochastic events within the CRUs and the physiological membrane models. To solve the PDEs, a conforming finite element method is employed for the spatial discretization and higher order linearly implicit methods, Rosenbrock type methods, for the time integration. The transition times between different states were determined using the Gillespie Algorithm for time dependent transition rates within the CRUs. We employ the Rush-Larsen time step integrator to solve the electrophysiology membrane model. We demonstrate the complete algorithm and present such large scale numerical results for 436 CRU setup using many core parallel machines. Finally, the implementation based on the DUNE software environment is discussed.

[*] Max-Delbrück center for molecular medicine (MDC), Robert-Rössle str-10, 13125 Berlin, Germany. nagaiah.chamakuri@mdc-berlin.de

[†] MDC, 13125 Berlin, Germany. wilhelm.neubert@mdc-berlin.de

[‡] MDC, 13125 Berlin, Germany. stephen.gilbert@mdc-berlin.de

[§] MDC, 13125 Berlin, Germany. Janine.Vierheller@mdc-berlin.de

[¶] MDC, 13125 Berlin, Germany. martin.falcke@mdc-berlin.de

Using horizontal vectorization in the DUNE-ISTL C++ solver library

Christian Engwer[*]

April 22, 2016

Modern hardware architectures exhibit a high level of intra-node concurrency. Exploiting hardware features like SIMD-units poses a challenge for high-level frameworks like DUNE. In order to make use of the SIMD-units and vectorize the code, it sufficient structure in the underlying problem is required. One way to achieve this is by choosing a suitable discretization method, e.g. higher order dG methods. This will allow for vertical vectorization, for example over quadrature points or shape-functions. Still this approach is not always applicable.

An alternative approach is to use horizontal vectorization. Often methods require computing the solution for many right-hand-sides. Examples are inverse problems, reduced basis methods or multi-scale FEM. This opens the route for vectorizing over multiple problems.

We will discuss how to incorporate horizontal vectorization into high-level C++ solver codes at the example of DUNE-ISTL. Most often the auto-vectorizer fails due to the use of high-level interfaces and explicit use of intrinsics is neither portable nor does it fit into high-level interfaces. C++ vectorization data types allow to abstract from the actual hardware and hide the use of intrisics, but still ensure an optimal implementation. The presented techniques allow vectorization through the whole solver stack, starting from the Krylov to strong preconditioners like the AMG. As an example application we present the inverse modelling for source reconstrution from EEG data. Numerical tests show nearly optimal complexity.

[*] Institute for numerical and applied mathematics, University of Münster, christian.engwer@uni-muenster.de

Grid Patches: Cache Efficiency and Vectorization for Low-order Methods in Dune

Jorrit Fahlke[*] Christian Engwer[†]

April 22, 2016

The finite-element-framework Dune contains (interfaces to) a set of grid managers with varying features. Each grid manager has its own unique data structures. Their design is driven largely by the desired features of the particular grid manager, such as MPI-parallelization, unstructured meshes and adaptivity.

This poses a challenge for computing hardware which increasingly relies on exploiting structure in the computation, such as GPUs and the SIMD units of modern CPUs. Another challenge is that algorithms typically have to iterate through the grid linearly, making it difficult to make efficient use of the caches of the CPU.

High order methods can meet these challenges by increasing the order in trade for coarser mesh cells. This allows for better structured computations on single mesh cells, and for reusing a larger part of the loaded data from the caches, increasing the algorithmic intensity.

We want to open up a similar route for low order methods. To this end we present a data structure for on-the-fly refinement of parts of arbitrary grids. This improves data locality and thus cache efficiency for low-order methods, while still allowing for advanced features in grid managers. Since we control the data structures and are not forced to use the data structure of the existing grid manager, we can tune them to allow for easy vectorization or offloading to GPUs.

We will present performance results of a full finite element solver which underline the benefit of such structures and discuss considerations for interface design and implementation.

[*] University of Münster, `jorrit.fahlke@uni-muenster.de`

[†] University of Münster, `christian.engwer@uni-muenster.de`

pyMOR – Generic Model Order Reduction Algorithms for MPI Distributed PDE Solvers

René Milk* Stephan Rave* Felix Schindler*

April 22, 2016

pyMOR [1] is a modern, object-oriented software library of model order reduction algorithms written in the Python programming language. Its unique feature is the easy interoperability with external PDE solver packages which provide the to-be-reduced high-dimensional model, while still allowing a fast interactive development process of new reduction algorithms. This is achieved by expressing every algorithm in pyMOR generically through well-defined interface methods of VectorArray, Operator and Discretization classes. For rapid prototyping of reduction algorithms, instances of these classes may be provided using pyMOR's built-in discretization toolbox, whereas later on, these objects may represent data structures in an external high-performance solver, allowing the very same algorithm to be applied to large-scale, possibly matrix free or nonlinear discretizations.

In this contribution we give a short overview on pyMOR's general architecture and then focus on recently developed tools for the integration of pyMOR with MPI parallel solvers. Our approach is based on a simple event-loop mechanism which enables straightforward, parallelization agnostic usage of MPI distributed discretizations from rank 0. This not only makes it possible to use pyMOR's (sequential) reduction algorithms without change for MPI parallel discretizations. Our approach also enables the user to control and debug large simulations on supercomputers easily from within an interactive Python shell. We will showcase pyMOR's design by presenting numerical examples using models provided by the deal.II, DUNE, FEniCS and NGSolve PDE solver packages.

References

[1] R. Milk, S. Rave, F. Schindler, *pyMOR - Generic Algorithms and Interfaces for Model Order Reduction*, accepted for publication in SIAM J. Sci. Comput. (2016), URL: http://arxiv.org/abs/1506.07094.

*University of Münster, (rene.milk|stephan.rave|felix.schindler)@uni-muenster.de

Computational methods and technologies of the Basic System of Modelling (BSM)

V.S.Gladkikh[*] Y.L.Gurieva[†] V.P.Il'in[‡] A.V.Moskalev[§] A.V.Petukhov[¶] I.N.Skopin[||]

April 7, 2016

A general structure and main computational components of Basic System of Modelling (BSM) are presented. BSM is an integrated computational environment for support of all stages of numerical solution of interdisciplinary direct and inverse mathematical problems which describe various real processes and phenomena and which have to be solved on a multi-processor computer system. Different tools of BSM are responsible for construction of geometric and functional models, generation of adaptive non-structured grids, approximation of the original differential and/or integral equations, solving of large algebraic systems, optimization approaches, post-processing, etc.

Each block of tools is the extendable autonomous Open Source code. Communications (information transfer) between different parts of BSM are based on some set of its data structures: geometric and functional data structures which include all information about the problem to be solved, grid and algebraic ones, etc. These data structures support also the interfaces with the end users and with external software available such as CAD/CAE products, for example. The functionality of BSM is supposed to be a redundant collection of the advanced computational methods and technologies which provide a fast and flexible assembling of efficient applied packages for particular classes of the problems and/or particular needs. The architecture of BSM is based on the cross-platform and multi-language implementation as well as on the Component Object Management. BSM is designed to be efficiently used on a wide range of computer systems.

[*] The Institute of Computational Mathematics and Mathematical Geophysics SB RAS, gladvs_ru@mail.ru

[†] The Institute of Computational Mathematics and Mathematical Geophysics SB RAS, yana@lapasrv.sscc.ru

[‡] The Institute of Computational Mathematics and Mathematical Geophysics SB RAS and Novosibirsk State University, Novosibirsk, Russia, ilin@sscc.ru

[§] The Institute of Computational Mathematics and Mathematical Geophysics SB RAS

[¶] The Institute of Computational Mathematics and Mathematical Geophysics SB RAS, petukhov@lapasrv.sscc.ru

[||] The Institute of Computational Mathematics and Mathematical Geophysics SB RAS and Novosibirsk State University, Novosibirsk, Russia, iskopin@gmail.com

The prototype version of BSM is illustrated by some preliminary results for several 3D applications: a heat transfer, electromagnetic problems in a frequency domain, multi-scale diffusion-convection mixed boundary value problems, a scalable parallel implementation of the Domain Decomposition Methods in the Krylov subspaces to solve very large sparse SLAEs.

`Dune::ACFem` – Adaptive Convenient Finite Elements

Claus-Justus Heine[*]

March 23, 2016

We present the software package `Dune::ACFem` which – on top of `Dune::Fem` – implements an easy to use finite element toolbox for conforming finite element methods. In `Dune::ACFem` weak formulations for PDEs are be glued together from an easily extendable zoo of predefined standard PDE-models using natural notation. Predefined standard algorithms for adaptive methods lead to successful implementations with only a few lines of own source code where even tedious computations of "right hand sides" for test-problems can be carried out automatically. `Dune::ACFem` works with `C++` expression templates in order to achieve convenient notation without sacrificing efficiency. Features provided by the `Dune` core-modules and `Dune::Fem` like parallelization, finite element spaces, solvers are inherited from these "parent" packages.

References

[1] A. Dedner, R. Klöfkorn, M. Nolte, M. Ohlberger, A generic interface for parallel and adaptive scientific computing: Abstraction principles and the DUNE-FEM module. *Computing*, **90(3)**, 2011, 165–196

[2] C.-J. Heine, `Dune::ACFem`, `http://users.dune-project.org/projects/dune-acfem/`

[*] IANS, University of Stuttgart `Claus-Justus.Heine@IANS.Uni-Stuttgart.DE`

Regression Testing and Continuous Integration for Scientific Codes

Timo Heister

This talk will discuss motivations, tools, best practices, and past experiences with automated testing for scientific software. Because information about testing software in general is easy to be found, focus will be on issues specific to scientific codes.

After an introduction to testing and the discussion and demonstration of unit testing frameworks and other tools (like ctest and numdiff), I will explain how testing is done in the various projects I am involved in. I am the co-author of several different open source scientific software projects with different audiences, project sizes, and types. This includes the large c++ finite element library deal.II, the mantle convection code ASPECT for the geoscience community, and a smaller mineral physics code BurnMan written in Python, among others.

Finally, I will discuss the topic of continuous integration based on the tools discussed above and give general advice.

NGS-Py: Efficient element matrix calculation and application

Matthias Hochsteger[*] Joachim Schöberl[†]

April 22, 2016

In this talk we present details about the calculation of element matrices and matrix free operator application from expression trees in NGS-Py. We discuss the differences between run-time evaluation and just in time compilation. Finite element shape-function evaluation and transpose evaluation on CPUs using vectorization, and on GPUs is discussed. In particular, a pipelined reduction algorithm is presented.

[*] TU Wien, matthias.hochsteger@tuwien.ac.at

[†] TU Wien, joachim.schoeberl@tuwien.ac.at

TSFC: a structure-preserving form compiler

Miklós Homolya[*] Lawrence Mitchell[†] Fabio Luporini[‡]
Andrew T.T. McRae[§] David A. Ham[¶]

Firedrake is an automated system for the solution of PDEs using the finite element method. The high level interface uses the Unified Form Language (UFL) of the FEniCS project to allow expression of complex variational forms in a language that is close to that of the underlying mathematics. A core component of the software is the form compiler, that takes this high level description and produces the low-level code for finite element assembly. In this talk, I will present the core ideas and the design of TSFC, Firedrake's new form compiler.

The core concept of TSFC is to maintain the structure of the input expression as long as possible. This facilitates the application of optimisations at the highest possible level of abstraction.

TSFC is built on two core ideas:

1. A novel way of handling test and trial functions within discontinuous Galerkin methods, exploiting the linearity of the form.
2. Compilation happens in two stages, using a tensor algebra language as an intermediate representation. This creates better separation of concerns between the implementation of finite element objects and the optimal scheduling of the resulting tensor algebra operations.

TSFC achieves good compilation time even for complicated forms, such as those featuring non-affine elements. Its preservation of the algebraic structure of the input expressions will facilitate optimisations that exploit structure in the finite elements themselves as well, such as the sum factorisation of tensor product elements. The information about the structure of finite elements will be provided by FInAT, an abstract, smarter library of finite elements.

[*] Imperial College London, m.homolya14@imperial.ac.uk
[†] Imperial College London, lawrence.mitchell@imperial.ac.uk
[‡] Imperial College London, f.luporini12@imperial.ac.uk
[§] University of Bath, a.t.t.mcrae@bath.ac.uk
[¶] Imperial College London, david.ham@imperial.ac.uk

Uncertainty quantification and numerical homogenisation for porous media problems with OpenFOAM and other opensource software

Matteo Icardi*

April 23, 2016

In this talk we will present our recent efforts to develop a unified and automatic workflow to study flow and transport in porous materials. The main software used is OpenFOAM, an opensource CFD solver written in C++ and based on finite volume discretisations. However also other FEM codes, such as GetDP, Fenics and Dune, are tested and used for specific problems. We will present results related to diffusion, convection-diffusion, Stokes, and Navier-Stokes equations, solved in complex two and three-dimensional geometries.

To automatise the entire simulation workflow, first random heterogeneous materials are generated with deposition algorithms implemented in Bullet Physics and Blender, as well, as random packing algorithm developed in Python. The meshing step is then performed by Gmsh or by snappyHexMesh. Many random realisations are generated and then discretised and solved.

All the steps are implemented with openMPI parallelism and in a multi-resolution fashion to allow the usage of our Multilevel Monte Carlo uncertainty quantification code, entirely developed in Python. Python is also used to link all the various steps of the simulations and data analysis.

In this talk, we will show the potential of this fully opensource approach for the multiscale simulation and uncertainty quantification of several realistic problems in porous media. Some theoretical aspects related to homogenisation, upscaling, and spectral properties of the differential operators will also be given.

References

[1] Icardi, Matteo, Gianluca Boccardo, and Raul Tempone. ”On the predictivity of pore-scale simulations: estimating uncertainties with multilevel Monte Carlo.” *Advances in Water Resources* (2016)..

*University of Warwick, matteo.icardi@warwick.ac.uk

Implementation in MoFEM of hierarchical and heterogenous approximations bases

Lukasz Kaczmarczyk and Chris Pearce,
University of Glasgow,
lukasz.kaczmarczyk@glasgow.ac.uk
chris.pearce@glasgow.ac.uk

April 6, 2016

MoFEM [1] (Mesh Oriented Finite Element Method) is a C++ library supporting the solution of finite elements problems. It is developed to provide a free and open finite element code for engineers, students and academics. It is tailored for the solution of multi-physics problems with arbitrary levels of approximation, different levels of mesh refinement and optimised for high-performance computing. It is also designed to be able manage complexities related to heterogeneous order of approximations for L2,H1,H-div and H-curl spaces, see [2].

MoFEM is the blend of the Boost MultiIndex containers, MOAB (Mesh Oriented Database) [4] and PETSc (Portable, Extensible Toolkit for Scientific Computation) [3]. It is open-source software under the GNU Lesser General Public License. MoFEM can read and write a number of different mesh file formats using functionality provided by MoAB. The current version of MoFEM has full support for CUBIT/TRELIS, TetGEN and NetGEN for pre-processing and ParaView for post-processing.

The MoFEM library itself is designed to be small, modular and extendable. Users can implement modules as independent projects, located in its own repository, which can be either public or private, with their own copyright and license. Such flexibility, allowing for privacy and different licensing parts of the code, is designed to accommodate the needs of both academics and industry. Since each user module is implemented in MoFEM using its internal database, data between modules can be exchanged without prior knowledge of how each module is developed.

We will present various implementations of problems involving hierarchical approximations spaces focussing attention on implementation and general philosophy of the code.

References

[1] MoFEM finte element code. http://mofem.eng.gla.ac.uk.

[2] Mark Ainsworth. Essential boundary conditions and multi-point constraints in finite element analysis. *Computer Methods in Applied Mechanics and Engineering*, 190(48):6323–6339, 2001.

[3] Satish Balay, Shrirang Abhyankar, Mark F. Adams, Jed Brown, Peter Brune, Kris Buschelman, Lisandro Dalcin, Victor Eijkhout, William D. Gropp, Dinesh Kaushik, Matthew G. Knepley, Lois Curfman McInnes, Karl Rupp, Barry F. Smith, Stefano Zampini, and Hong Zhang. PETSc Web page. http://www.mcs.anl.gov/petsc, 2015.

[4] Timothy J Tautges. Canonical numbering systems for finite-element codes. *International Journal for Numerical Methods in Biomedical Engineering*, 26(12):1559–1572, 2010.

Adaptive Higher Order Discontinuous Galerkin Methods for Two-phase Flow in Porous Media with DUNE-FEM

Birane Kane[*]

We present higher order Discontinuous Galerkin discretizations schemes [1], [4] of two-phase flow model arising in the modelisation of subsurface flows in strongly heterogeneous porous media. The flow in the domain is immiscible and incompressible with no mass transfer between phases. We consider a fully implicit, locally conservative, higher order discretization on adaptively generated meshes. The implementation is based on the Open-Source PDE software framework DUNE [2] and the finite element module DUNE-FEM [3].

References

[1] D. N. ARNOLD, F. BREZZI, , B. COCKBURN, D. MARINI, *Unified analysis of discontinuous Galerkin methods for elliptic problems*, SIAM J. Numer. Anal. 39 (2002), 1749– 1779.

[2] P. BASTIAN, M. BLATT , A. DEDNER, C. ENGWER, R. KLÖFKORN, R. KORNHUBER, M. OHLBERGER, O. SANDER, *A Generic Grid Interface for Parallel and Adaptive Scientific Computing. Part II: Implementation and Tests in DUNE.*, Computing, 82(2-3), (2008), 121–138.

[3] A. DEDNER, R. KLÖFKORN, M. NOLTE, M. OHLBERGER, *A generic interface for parallel and adaptive scientific computing: Abstraction principles and the DUNE-FEM module*, Computing 90:3 (2011), 165–196.

[4] B. KANE, *Using DUNE-FEM for Adaptive Higher Order Discontinuous Galerkin Methods for Two-phase Flow in Porous Media*, Archive of Numerical Software, submitted.

[*] Institute of Applied Analysis and Numerical Simulation, University of Stuttgart, birane.kane@mathematik.uni-stuttgart.de

Steady State modelling of the FLAME D in OpenFOAM using the Eddy Dissipation Combustion Model.

George Kanellis[*] Antti Oksanen[†]

April 22, 2016

A custom-build, steady state solver that simulates turbulent reactive gas flows is implemented in the OpenFOAM framework. The turbulence is modelled using the Reynolds-averaged k-Epsilon model. The effective reaction rate is obtained as the minimum between the Eddy Dissipation Combustion rate [1] and the kinetic one. The chemistry is modelled using several global reaction mechanisms. The solver is veryfied and tested on the Sandia Flame D case [2] presented at the TNF workshop, which is a piloted CH4/air flame with its central jet Reynolds number being equal to 22400. The solver convergence was satisfactory and an acceptable match with the experimental results is achieved. Finally, the results are compared with the experimental data and also with OpenFOAM implementations of the Partially Stirred Reactor model [3] and the Eddy Dissipation Concept [4]. The long term target is the application of this solver at the modelling of a large scale, biomass-fired boiler.

References

[1] B. F. Magnusen and B. H. Hjertager. On Mathematical Modeling of Turbulent Combustion with special emphasis on Soot Formation and Combustion, *16th Symposium (International) on Combustion, the Combustion Institute, Pittsburgh*, (1976).

[2] R. S. Barlow, J. H. Frank. Effects of turbulence on species mass fractions in methane/air jet flames. *Proc. Combust. Inst. 27, 1087–1095*, (1998).

[3] J. Chomiak J, A. Karlsson. Flame liftoff in diesel sprays, *26th Symposium (International) on Combustion , pp. 2557– 2564*, (1996).

[*] Tampere university of Technology, georgios.kanellis@tut.fi

[†] Tampere university of Technology, antti.oksanen@tut.fi

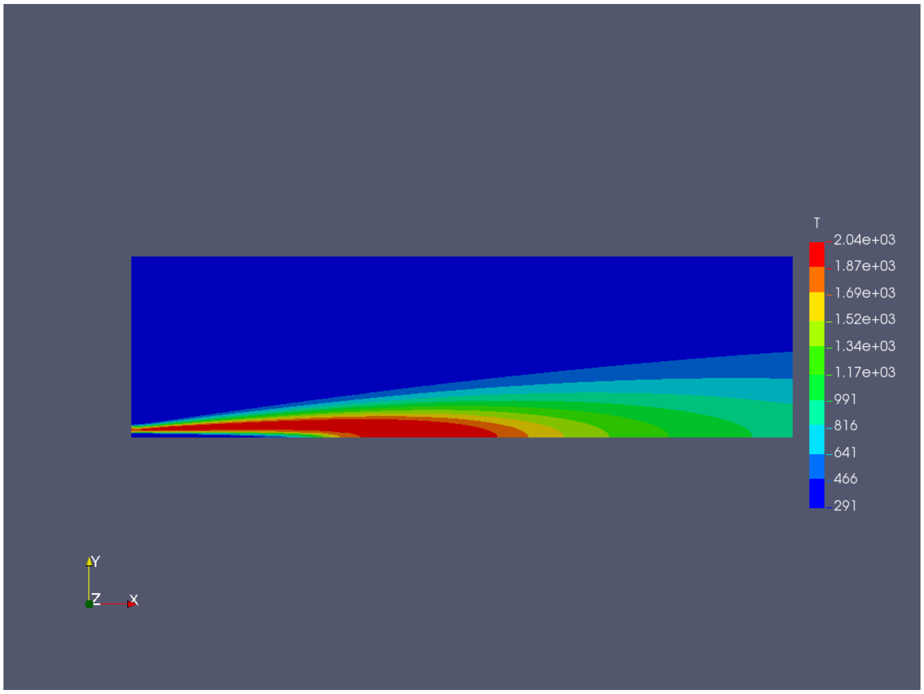


Figure 1: FLAME - D, Temperature Field.

[4] D. A. Lysenko, I. S. Ertesvåg and K. E. Rian. Numerical Simulations of the Sandia Flame D Using the Eddy Dissipation Concept, *Flow, Turbulence and Combustion 93(4), 665-687*, (2014).

Automated systemtesting in scientific numerical software frameworks using the example of Dune

Dominic Kempf[*] Timo Koch[†] Bernd Flemisch[‡]
Peter Bastian[§]

Testing is acknowledged as indispensible support for scientific software development and assurance of software quality to produce trustworthy simulation results. Most of the time, testing in software frameworks developed at research facilities is restricted to either unit testing or simple benchmark programs. However, in a modern numerical software framework, such as deal.II, FEniCS, or Dune, the number of possible feature combinations constituting a program is vast. Only system testing, meaning testing within a possible end user environment also emulating variability, can assess software quality and reproducibility of numerical results.

We provide dune-testools, an easy-to-use interface taking workload off developers and administrators in open-source scientific numerical software framework projects. In our approach, the large number of possible combinations is reduced using the scientific expert knowledge of developers to identify the practically relevant combinations. The variability of one such systemtest can be easily modelled through an extended INI-file syntax, that allows to describe sets of configurations. Both runtime and compiletime variants of systemtests are supported. To implement the latter, dune-testtools interacts with Dune's CMake-based build system. Furthermore, dune-testtools provides an interface and sample implementations for quality measures tailored to numerical frameworks for the solution of PDEs.

References

[1] D. Kempf, T. Koch, System testing in scientific numerical software frameworks using the example of Dune, *Proceedings of the Dune user meeting 2015*, (submitted)

[*] Heidelberg University, dominic.kempf@iwr.uni-heidelberg.de
[†] University of Stuttgart, timo.koch@iws.uni-stuttgart.de
[‡] University of Stuttgart, bernd.flemisch@iws.uni-stuttgart.de
[§] Heidelberg University, peter.bastian@iwr.uni-heidelberg.de

Describe, Don't Implement:
Declarative Programming in PDE Solvers

Andreas Klöckner[*] Matt Wala [†]

May 19, 2016

High-performance PDE solvers, as a software matter, are stretched between many, often contradictory demands: keeping up with the fast pace of research in numerical methods, scaling to serve demanding, high-fidelity engineering applications, and adapting to the ever-changing landscape of parallel, high-performance hardware. These tensions contribute considerably to making PDE solver software difficult to write and hard to maintain. Domain-specific mini-languages ('DSLs') and code generation are a proven way to ease the tension and maintain separation of concerns.

In this talk, I will present a set of loosely coupled, Python-based tools that each address a distinct challenge in PDE solver construction with the help of a mini-DSL. 'Loopy' helps translate arithmetic on arrays into high-performance parallel code on CPUs and GPUs. 'Leap' and 'Dagrt' enable multi-rate, multi-physics, multi-language, and multi-method time integration while laying the foundation for advanced cross-operator optimizations. Grudge, lastly, allows the efficient semidiscretization of discontinuous Galerkin finite element operators. Time permitting I will also discuss common, underlying infrastructure that enables and underpins these tools.

[*] University of Illinois at Urbana-Champaign, Urbana, IL, USA, andreask@illinois.edu

[†] University of Illinois at Urbana-Champaign, Urbana, IL, USA, wala1@illinois.edu

Multiphysics with deal.II - Poroelasticity Solver

Uwe Köcher[*] Markus Bause[†]

February 24, 2016

In this work we present variational design and implementation aspects of a multiphysics solver for poroelasticity problems based on the deal.II library. Such problems appear for instance in geomechanical engineering and consist of a slightly compressible fluid flow within the pores of a porous media which is coupled with an elastic deformation of the surrounding solid. The coupled problem can be solved iteratively using an operator splitting; cf. [1, 2]. This enables us to reuse our implementations for distributed parallel and high-order simulations as well as our solver and preconditioning technology; cf. [3, 4]. Moreover, the deal.II library facilitate us to implement the framework with a mixed-multiphysics discretisation in space on a single triangulation. The treatment of the multiphysics boundary conditions is adressed further. We illustrate the performance of our framework by some three-dimensional simulations.

References

[1] U. Köcher, M. Bause: Poroelasticity: Flexible high-order variational space-time discretisations, in preparation, 2016.

[2] M. Bause, U. Köcher: Iterative coupling of variational space-time methods for Biot's system of poroelasticity. Lect. Notes Comput. Sci. and Engrg., accepted, 2016.

[3] U. Köcher: Variational space-time methods for the elastic wave equation and the diffusion equation, Ph.D. thesis, Helmut-Schmidt-University, Univ. of the German Federal Armed Forces, [urn:nbn:de:gbv:705-opus-31129], 2015.

[4] M. Bause, U. Köcher: Variational time discretization for mixed finite element approximations of nonstationary diffusion problems. J. Comput. Appl. Math. 289:208-224, [doi:10.1016/j.cam.2015.02.015], 2015.

[*] Department of Mechanical Engineering, Chair of Numerical Mathematics, Helmut-Schmidt-University Hamburg, Germany, koecher@hsu-hamburg.de

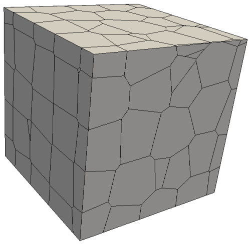
[†] Department of Mechanical Engineering, Chair of Numerical Mathematics, Helmut-Schmidt-University Hamburg, Germany, bause@hsu-hamburg.de

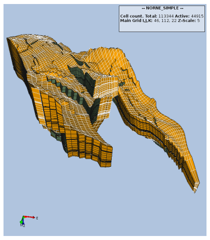
Higher Order Finite Volume Methods on Polyhedral Grids in DUNE

Anna Kvashchuk[1], Robert Klöfkorn[2], and Martin Nolte[3]

April 22, 2016

In this presentation we discuss a new grid implementation within the DUNE software framework DUNE [1](www.dune-project.org). This grid implementation covers arbitrary polyhedral grids including so called Corner-point grids that are typically used in reservoir simulation. The new DUNE grid is called `PolyhedralGrid` and part of the Open Porous Media Initiative (www.opm-project.org) which is based on the DUNE core modules.




-- NORNE_SIMPLE --
Cell count. Total: 113344 Active: 44915
Main Grid I,J,K: 46, 112, 22 Z-Scale: 5

For such grids, where a reference element is not available, we discuss typical pitfalls and necessary changes to the DUNE grid interface. The effectiveness of our implementation is demonstrated with explicit and implicit higher order Finite Volume schemes with an application to reactive advection-diffusion problems. The implementation of the numerical scheme is based on the DUNE-FEM framework [2].

References

[1] P. Bastian, M. Blatt, A. Dedner, C. Engwer, R. Klöfkorn, R. Kornhuber, M. Ohlberger, and O. Sander. A Generic Grid Interface for Parallel and Adaptive

[1] University of Stavanger, a.kvashchuk@stud.uis.no
[2] International Research Institute of Stavanger, robert.kloefkorn@iris.no
[3] University of Freiburg, nolte@mathematik.uni-freiburg.de

Scientific Computing. Part II: Implementation and Tests in DUNE. *Computing*, 82(2–3):121–138, 2008. [doi].

[2] A. Dedner, R. Klöfkorn, M. Nolte, and M. Ohlberger. A Generic Interface for Parallel and Adaptive Scientific Computing: Abstraction Principles and the DUNE-FEM Module. *Computing*, 90(3–4):165–196, 2010. [doi].

A compressible solver for modelling cavitation in OpenFOAM

Nikolaos Kyriazis* Phoevos Koukouvinis† Manolis Gavaises‡

April 22, 2016

An explicit compressible solver implemented in OpenFOAM [1] for cavitating flows is presented. The current work is based on a homogeneous mixture method, which means that the vapor is predicted through the variation of the density. A two step barotropic equation for the pressure, the Tait equation for the liquid and the Egerer formula for the mixture is used:

$$p(\rho) = \begin{cases} B[(\frac{\rho}{\rho_{sat,L}})^n - 1] + p_{sat}, & \rho \geq \rho_{sat,L} \\ p_{sat} + C(\frac{1}{\rho_{sat,L}} - \frac{1}{\rho}), & \rho < \rho_{sat,L} \end{cases} \tag{1}$$

where $B = 293.5 \cdot 10^6 Pa$, $n = 7.15$, $\rho_{sat,L} = 998.2\frac{kg}{m^3}$, $C = 1450\frac{Pa \cdot kg}{m^3}$, and $p_{sat} = 2339Pa$. In order for the numerical scheme to be applicable to a wide range of Mach numbers, the hybrid flux of Schmidt et al. [2] is used in conjuction with Runge-Kutta method of fourth order in time.

Validation for the shock tube test case is shown in Fig. (1). The unsteady shedding mechanism around a hydrofoil is depicted in Fig. (2). The inlet condition is $u_{in} = 12m/s$, the outlet is $p_{out} = 0.742bar$, the time step calculated for $CFL = 1.3$ is $dt = 6 \cdot 10^{-8}s$ and the final simulation time was $t = 2s$.

References

[1] OpenFOAM Foundation, OpenFOAM user guide, downloaded: (21-5-2015). URL `http://www.openfoam.org/docs/user`.

[2] S. Schmidt, I. Sezal, G. Schnerr, M. Talhamer, Riemann Techniques for the Simulation of Compressible Liquid Flows with Phase-transition at all Mach numbers-Shock and Wave Dynamics in Cavitating 3-D Micro and Macro Systems, *AIAA Aerospace Sciences Meeting and Exhibit* **46** (2008), 2008–1238.

*City University London, `Nikolaos.Kyriazis@city.ac.uk`

†City University London, `Foivos.Koukouvinis.1@city.ac.uk`

‡City University London, `M.Gavaises@city.ac.uk`

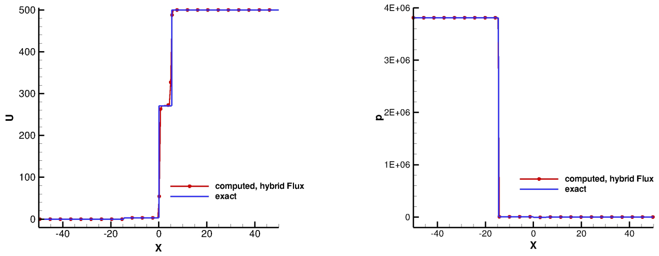
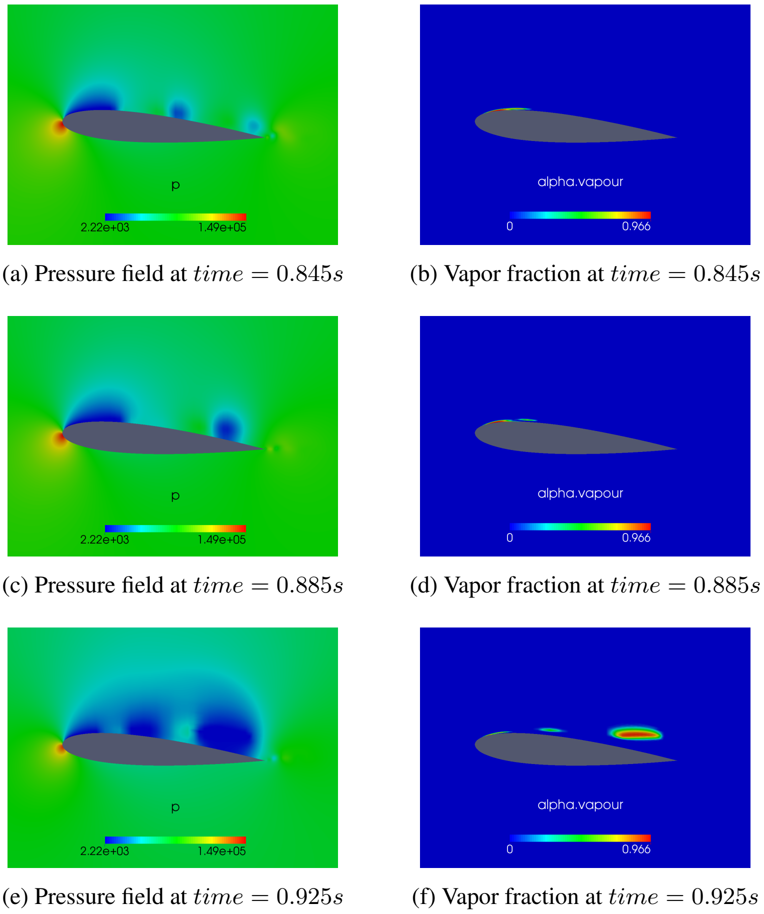


Figure 1: Comparison of the exact and numerical solutions for the shock tube test case. The left and right states are $\rho_L = 1000kg/m^3$, $u_L = 0m/s$ and $\rho_R = 1kg/m^3$, $u_R = 500m/s$ respectively.



(a) Pressure field at $time = 0.845s$

(b) Vapor fraction at $time = 0.845s$

(c) Pressure field at $time = 0.885s$

(d) Vapor fraction at $time = 0.885s$

(e) Pressure field at $time = 0.925s$

(f) Vapor fraction at $time = 0.925s$

Figure 2: Unsteady vortex shedding in NACA 0015 hydrofoil at angle of attack of 6 degrees.

Efficient mesh management in Firedrake using PETSc DMPlex

M. Lange[*] M. Knepley[†] L. Mitchell[‡] G. Gorman[§]

April 21, 2016

The use of composable abstractions allows the application of new and established algorithms to a wide range of problems while automatically inheriting the benefits of well-known performance optimisations [1]. In this talk we give a detailed overview of the PETSc DMPlex domain topology abstraction and its underlying dimension-independent API, as well as highlighting its natural composition with the Firedrake automated finite element system to create a PDE solving environment which combines expressiveness, flexibility and high performance. We describe the parallel mesh and data distribution capabilities provided by DMPlex, including arbitrary parallel overlap generation and mesh redistribution for load balance [2], and demonstrate how Firedrake achieves a cache-oblivious data layout through PETSc that increases performance of the finite element assembly.

References

[1] M. Lange, L. Mitchell, M. Knepley, G. Gorman, Efficient mesh management in Firedrake using PETSc-DMPlex, *Accepted in SISC Special Issue*, preprint: *arXiv:1506.07749*, 2016.

[2] M. Knepley, M. Lange, G. Gorman, Unstructured Overlapping Mesh Distribution in Parallel *Submitted to ACM Transactions on Mathematical Software*, preprint: *arXiv:1506.06194*, 2015.

[*] Department of Earth Science and Engineering, Imperial College London, UK, michael.lange@imperial.ac.uk

[†] Department of Computational and Applied Mathematics, Rice University, USA, knepley@gmail.com

[‡] Department of Computing and Department of Mathematics, Imperial College London, UK, lawrence.mitchell@imperial.ac.uk

[§] Department of Earth Science and Engineering, Imperial College London, UK, g.gorman@imperial.ac.uk

Domain decomposition methods for reaction-diffusion systems

Daniel Loghin[*]

April 22, 2016

The finite element formulation of systems of coupled reaction-diffusion problems yields large sparse linear systems which are often intractable via standard solution methods. In this talk I will discuss a non-overlapping domain decomposition approach that allows for the fast resolution of such problems. The method relies on an interface Schur complement preconditioner which incorporates information about the problem parameters and the underlying PDE operators. Numerical examples will demonstrate the efficiency and robustness of the proposed method, in particular, independence of geometric and problem parameters, such as diffusion coefficients and reaction rates.

[*] University of Birmingham, d.loghin@bham.ac.uk

Linear Operator—a generic, high-level expression syntax for deal.II

Matthias Maier[*], Mauro Bardelloni[†], Luca Heltai[†]

April 6, 2016

In this talk an *expression syntax* for the evaluation of matrix-matrix, matrix-vector and vector-vector operations will be presented. The implementation is similar to the well-known general concept of *expression templates* as used, for example, in the C++ linear-algebra libraries `Eigen` and `Blaze`. The novelty of the approach that is discussed here lies in the use of new C++11 features like *lambda expressions* and `std::function` objects that avoids the majority of the implementational complexity that usually comes with a pure template solution.

The expression syntax has been developed within `deal.II`, but it is fairly generic: the implementation only requires a minimal vector and matrix interface, that all of `deal.II`'s concrete vector and matrix types adhere to. This makes the interface fully transparent with respect to the concrete implementation, in particular to the storage strategy (full matrix, sparse structure), and memory strategy (local, shared, distributed).

The talk concludes with a number of performance comparisons and examples that demonstrate that the framework results in efficient, short and concise code. The performance comparisons show that the overhead introduced by `std::function` objects is negligible for moderately sized matrices, even when compared to native expression-template implementations.

References

[1] W. Bangerth, T. Heister, L. Heltai, G. Kanschat, M. Kronbichler, M. Maier, B. Turcksin, The deal.II library, Version 8.3, *Archive of Numerical Software* **4** (2016).

[2] M. Maier, M. Bardelloni, L. Heltai, Linear Operator— a generic, high-level expression syntax, *Computers & Mathematics with Applications* (in review).

[*] School of Mathematics, University of Minnesota, 127 Vincent Hall, 206 Church Street SE, Minneapolis, MN 55455, USA, (msmaier@umn.edu).

[†] SISSA - International School for Advanced Studies, Via Bonomea 265, 34136 Trieste, Italy, (mauro.bardelloni@sissa.it, luca.heltai@sissa.it).

IGATOOLS: a general purpose C++14 library for Isogeometric Analysis

Massimiliano Martinelli[*] Pablo Antolin[†]

M. Sebastián Pauletti[‡]

April 7, 2016

We present the design and the implementation of IGATOOLS [1] (`http://www.igatools.org`), a C++14 general purpose library for solving PDEs using the isogeometric analysis framework [2].

In the IGATOOLS design, the mathematical concepts of the isogeometric method and their relationships are directly mapped into classes and their interactions. This encapsulation gives flexibility to use the library in a wide range of scientific areas and applications. We provide a precise framework for a lot of loose, available information regarding the implementation of the isogeometric method, and also discuss the similarities and differences between this and the finite element method.

The library uses advanced object oriented and generic programming techniques to ensure reusability, reliability, and maintainability of the source code. Among other capabilities, the library supports the development of dimension independent code (including manifolds and tensor-valued spaces), implements mutlithreaded methods and takes full advantage of the underlying tensor-product structure of the problem at hand (if any). The library also provides a plugin for interfacing with ParaView [3] in order to help the user to visualize the results.

We finally present a number of code examples to illustrate the flexibility and power of the library.

References

[1] M. S. Pauletti, M. Martinelli, N. Cavallini, and P. Antolin, "Igatools: An Isogeometric Analysis Library", *SIAM J. Sci. Comput.* (2015), **37(4)**, 465–496.

[*] IMATI "E. Magenes" - CNR, Pavia `martinelli@imati.cnr.it`

[†] Dipartimento di Matematica "F. Casorati", Università degli Studi di Pavia, `pablo.antolinsanchez@unipv.it`

[‡] Instituto de Matemática Aplicada del Litoral (IMAL), Consejo Nacional de Investigaciones científicas y técnicas (CONICET) and Universidad Nacional del Litoral, Santa Fe, Argentina, `pauletti@santafe-conicet.gov.ar`

[2] T. J. R. Hughes, J. A. Cottrell, Y. Bazilevs, “Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement”, *Comput. Methods Appl. Mech. Engrg.* (2005), **194**:39-41, 4135–4195.

[3] www.paraview.org

Extending DUNE: The `dune-xt` modules

Tobias Leibner[*] René Milk[*] Mario Ohlberger[*]
Felix Schindler[*]

April 7, 2016

In this contribution we introduce the `dune-xt`[1] family of modules, our effort to complement the DUNE core with a well tested and structured collection of utilities and concepts that is open source licensed, welcoming contributions and freely available on GitHub:

`https://github.com/dune-community/`

We will give a broad overview of the constituent modules `dune-xt-common`, `dune-xt-grid`, `dune-xt-la` and `dune-xt-functions`, as well as concrete usage examples of convenience helpers and abstractions in `dune-multiscale` and `dune-gdt`.

We will also show our how testing setup leverages combined features of GitHub, travis.ci[1], the Google C++ Testing Framework[2] and the DUNE testtools module[2] to achieve automated, convenient, continuous testing with very little maintenance effort.

References

[1] T. Leibner, R. Milk, F. Schindler, Extending DUNE: The dune-xt modules, *ArXiv e-prints [cs.MS] (1602.08991)*, 2016.

[2] D. Kempf and T. Koch, System testing in scientific numerical software frameworks using the example of dune, *submitted to the Proceedings of the DUNE User Meeting 2015, Sep. 28-29, 2015, Heidelberg, Germany*, 2016.

[*] Institute of Applied Mathematics, University of Münster, Einsteinstr. 62, D-48161 Münster, Germany, {`tobias.leibner,rene.milk,mario.ohlberger,felix.schindler`}`@wwu.de`.

[1] `https://travis-ci.org`

[2] `https://github.com/google/googletest`

pyMOR – Generic Model Order Reduction Algorithms for MPI Distributed PDE Solvers

René Milk* Stephan Rave* Felix Schindler*

April 22, 2016

pyMOR [1] is a modern, object-oriented software library of model order reduction algorithms written in the Python programming language. Its unique feature is the easy interoperability with external PDE solver packages which provide the to-be-reduced high-dimensional model, while still allowing a fast interactive development process of new reduction algorithms. This is achieved by expressing every algorithm in pyMOR generically through well-defined interface methods of VectorArray, Operator and Discretization classes. For rapid prototyping of reduction algorithms, instances of these classes may be provided using pyMOR's built-in discretization toolbox, whereas later on, these objects may represent data structures in an external high-performance solver, allowing the very same algorithm to be applied to large-scale, possibly matrix free or nonlinear discretizations.

In this contribution we give a short overview on pyMOR's general architecture and then focus on recently developed tools for the integration of pyMOR with MPI parallel solvers. Our approach is based on a simple event-loop mechanism which enables straightforward, parallelization agnostic usage of MPI distributed discretizations from rank 0. This not only makes it possible to use pyMOR's (sequential) reduction algorithms without change for MPI parallel discretizations. Our approach also enables the user to control and debug large simulations on supercomputers easily from within an interactive Python shell. We will showcase pyMOR's design by presenting numerical examples using models provided by the deal.II, DUNE, FEniCS and NGSolve PDE solver packages.

References

[1] R. Milk, S. Rave, F. Schindler, *pyMOR - Generic Algorithms and Interfaces for Model Order Reduction*, accepted for publication in SIAM J. Sci. Comput. (2016), URL: `http://arxiv.org/abs/1506.07094`.

*University of Münster, (rene.milk|stephan.rave|felix.schindler)@uni-muenster.de

Firedrake: composable abstractions for finite element simulation

Lawrence Mitchell[*] David A. Ham[†]

Gheorghe-Teodor Bercea[‡] Miklós Homolya[‡] Fabio Luporini[‡] Paul H. J. Kelly[‡]

Florian Rathgeber[§] Andrew T. T. McRae[¶]

April 6, 2016

The complexity inherent in the application of advanced numerics on modern hardware to coupled physical systems presents a critical barrier to simulation development. To overcome this, we must create simulation software which embodies the abstraction and composability of the underlying mathematics. In this way, a system is created in which mathematicians, computer scientists, and application specialists can each deploy their own expertise, benefiting from the expertise of the others. Critically, this approach minimises the extent to which individuals must become polymaths to share in these advances.

In this talk I will present Firedrake and PyOP2, a composition of new and existing abstractions which creates a particularly complete separation of concerns. A key observation is that by careful abstraction design, we are able to teach computers how to reason about the mathematical structure of the problem, rather than working out everything on paper. This enables the creation of high performance, sophisticated finite element models from a very high level mathematical specification and has enabled advances in computer science and numerics, while also facilitating the creation of simulation systems for a variety of applications.

[*] Department of Computing and Department of Mathematics, Imperial College London, lawrence.mitchell@imperial.ac.uk

[†] Department of Mathematics, Imperial College London

[‡] Department of Computing, Imperial College London

[§] European Centre for Medium-range Weather Forecasts

[¶] Department of Mathematical Sciences, University of Bath

Architecture and application of the data filtering library ”CPPPO” to transport phenomena in dense gas-particle flows

Federico Municchi[*] Stefan Radl [†]

June 14, 2016

Most of the phenomena occurring in physics and engineering are modelled by means of a multi-scale approach. This is due to the broad range of temporal and spatial scales at which relevant phenomena take place [1]. The multi-scale approach typically consists of making use of data available from ”well-resolved” simulations, to derive closures for models on larger scales. Establishing a proper functional form for such closures is not trivial [2, 3] since it requires e processing of a huge amount of data from several parameter configurations.

In this work, we present the core architecture and some applications of CPPPO [4], a universal open-source library for developing such closure models by filtering data from resolved multiphase flow simulations.

The library is designed to process data ”on-the-fly” from massively parallel numerical simulations performing filtering sampling and binning operations on Eulerian and Lagrangian fields. CPPPO comes with an interface class to the widely-used finite volume library OpenFOAM®. A general interface to read csv data files is also provided. Furthermore, the library can be used to process data from both structured and unstructured grids. A scalability analysis conducted at the Vienna Scientific Cluster VSC-3 showed excellent parallel efficiency when executing ”CPPPO” on up to 1,024 cores. This is due to the use of novel filtering algorithms that take advantage from the fine graining of the computational domain.

The library has been applied to study the heat transfer in mono- and bi-disperse particle beds by mean of Particle-Resolved Direct Numerical Simulation (PR-DNS) using CFDEMCoupling® [5]. Some implications for the development of closure relations for heat transfer in bi-disperse gas-particle beds are discussed.

[*] Institute of Process and Particle engineering, TU Graz, Graz, Austria fmunicchi@tugraz.at

[†] Institute of Process and Particle engineering, TU Graz, Graz, Austria sradl@tugraz.at

References

[1] J. Van der Hoef, M.A., van Sint Annaland, M., Deen, N.G., Kuipers,Numerical simulation of dense gassolid fluidized beds: a multiscale modeling strategy, Annu. Rev. Fluid Mech. **40**,47-70 (2008) .

[2] S. Radl, S. Sundaresan, A drag model for filtered Euler Lagrange simulations of clustered gas particle suspensions,Chemical Engineering Science. **117**, 416−425 (2014) .

[3] Sankaran Sundaresan, Stefan Radl, Christian C. Milioli, Coarse-Grained Models for Momentum, Energy and Species Transport in Gas-Particle Flows,14th international conference on fluidization, 2013.

[4] Federico Municchi, Stefan Radl, Christoph Goniva, Highly efficient spatial data filtering in parallel using the opensource library CPPPO, Computer Physics Communications, 2016, IN PRESS.

[5] Kloss, C., Goniva, C., Hager, A., Amberger, S., and Pirker,S. Models, algorithms and validation for opensource DEM and CFD-DEM. Progress in Computational Fluid Dynamics, 12:140–152, 2012.

Rapid Prototyping in DUNE Through Python Scripting

Martin Nolte[*], Andreas Dedner[†]

April 22, 2016

The DUNE discretization library provides abstract concepts required for implementing a large class of grid-based numerical methods. At its core is a flexible interface for parallel, adatiptive computational grids. Performance is retained by static polymorphism making use of C++ templates. This results in a steep learning curve, especially for beginners to C++ programming.

The Python language, on the other hand, is very beginner-friendly and is becoming increasingly popular in the scientific computing community. We describe a new module, dune-fempy, closing the gap between the efficiency of the C++ implementation and Python's accessibility. Apart from providing bindings to the most important DUNE interface classes, the module helps developers to easily export their algorithms to Python users.

Two difficulties arise in this context. Typically, the template arguments to DUNE classes are only known upon object construction, preventing Python bindings to be compiled at setup time. Moreover, Python automatically performs type erasure and simply passing back Python objects to C++ will result in a significant performance overhead. Both problems can be overcome by delaying compilation of the Python bindings.

This approach allows for rapid prototyping of numerical programs using small Python scripts without sacrificing flexibility or performance. This will be illustrated by a few examples.

[*] University of Freiburg, nolte@mathematik.uni-freiburg.de
[†] University of Warwick, a.s.dedner@warwick.ac.uk

A discontinuous Galerkin based incompressible Navier-Stokes solver

Marian Piatkowski[*] Peter Bastian[†]

The simulation of the transient incompressible Navier-Stokes equations provides not only a challenge towards high-performance computing, but is also a challenge for itself. The system is decoupled by the projection methods which is one of the few methods capable of solving large 3D problems in a reasonable computational time.

Based on this temporal discretization we present a new approach using discontinuous Galerkin approximation for both velocity and pressure in space. With this approximation the projection step guarantees local mass conservation. The parabolic equation for the tentative velocity is solved with diagonal implicit Runge-Kutta methods. For the solution of the elliptic equation for the pressure a hybrid AMG preconditioner based on a correction in the conforming subspace is employed. The performance of the implementation is based on the sum factorization approach and comparisons to the naive implementation are presented.

[*] IWR, Heidelberg University, marian.piatkowski@iwr.uni-heidelberg.de

[†] IWR, Heidelberg University, peter.bastian@iwr.uni-heidelberg.de

A generic discretization toolbox: `dune-gdt`

René Milk* Mario Ohlberger* Felix Schindler*

We present the generic discretization toolbox `dune-gdt` for the efficient and reliable grid-based approximation of partial differential equations (PDE). The mathematical framework behind `dune-gdt` relies on a splitting of localized quantities associated with a PDE into their respective building blocks, such as numerical quadratures and integrands or analytical and numerical fluxes. `dune-gdt` provides interfaces and implementations of these building blocks, which allows for a quick development of new numerical schemes by a combination of provided and user-defined building blocks.

These building blocks are defined in terms of sets of local functions [1], which are provided by data functions, local discrete functions and bases of discrete function spaces. Thus, for instance, the very same implementation of a local L^2–operator (formed by a volume integral operator and a product integrand) is used in a global L^2–operator which can be assembled into a matrix or can act on any pair of functions (discrete or not) to compute a product, and local and global L^2–projections and prolongations.

For these building blocks `dune-gdt` provides unified local assembly based on several parallelization paradigms, and was already used in HPC [3] and model reduction contexts [2]. `dune-gdt` is realized as a module within DUNE (see [1] and the references therein) and can make use of other discretization modules. It is an open source software package which is freely available on GitHub:

`https://github.com/dune-community/dune-gdt`

References

[1] T. Leibner, R. Milk, and F. Schindler. Extending DUNE: The dune-xt modules. *ArXiv e-prints [cs.MS]* (1602.08991), 2016.

[2] R. Milk, S. Rave, and F. Schindler. pymor - generic algorithms and interfaces for model order reduction. *arXiv e-prints [cs.MS]* (1506.07094), 2015.

[3] J. Mohring, R. Milk, A. Ngo, O. Klein, O. Iliev, M. Ohlberger, and P. Bastian. Uncertainty Quantification for Porous Media Flow Using Multilevel Monte Carlo, *Large-Scale Scientific Computing: 10th International Conference, Sozopol, Bulgaria*, 145–152. Springer International Publishing, Cham, 2015.

*Applied Mathematics, University of Münster, Einsteinstr. 62, D-48149 Münster, Germany, {rene.milk,mario.ohlberger,felix.schindler}@uni-muenster.de.

pySDC: A Python framework for spectral deferred corrections and PFASST

Daniel Ruprecht[*] Robert Speck[†]

April 20, 2016

The talk will present PYSDC [1], a Python framework implementing spectral deferred corrections (SDC) and their parallel-in-time multi-level variant PFASST (for "parallel full approximation scheme in space and time") [2]. PYSDC can be used for rapid prototyping of new developments and provides an easy way to try out ideas.

I will discuss two recent algorithmic studies that we have conducted using PYSDC. The first is an analysis of *fast-wave slow-wave* SDC [3], a splitting method of arbitrary order for compressible flow. It integrates acoustic waves implicitly to avoid stability limits while slow dynamics like advection are treated explicitly to save computational cost and minimise numerical dispersion.

The second study investigates PFASST's potential for *algorithm-based fault tolerance* [4]. As all parallel-across-the-step methods, PFASST stores multiple copies of the solution at different time steps – naturally, in case one of these is lost due to node failure, one can try to reconstruct it using information from neighbouring steps.

Future improvements of the framework will be sketched, e.g. a proper parallelisation to go beyond convergence studies or integration of other Python packages to provide more sophisticated spatial solvers and discretisations.

References

[1] `https://github.com/Parallel-in-Time/pySDC`

[2] M. Emmett, M. Minion, *Communications in Applied Mathematics and Computational Science*, **7**, 105–132, 2012

[3] D. Ruprecht, R. Speck, arXiv:1602.01626, 2016

[4] R. Speck, D. Ruprecht, arXiv:1510.08334, 2016

[*] School of Mechanical Engineering, University of Leeds, Woodhouse Lane, Leeds LS2 9JT, UK d.ruprecht@leeds.ac.uk

[†] Juelich Supercomputing Centre, Forschungszentrum Juelich GmbH, Germany r.speck@fz-juelich.de

π-DoMUS: Parallel Deal.II MUlti-physics Solver

Alberto Sartori[*] Mauro Bardelloni[†] Luca Heltai[‡]

April 7, 2016

We present the general-purpose parallel finite element multi-physics solver π-DoMUS. It is an open-source software framework, oriented towards high performance computing, featuring a high-level interface (from the user viewpoint) to solve systems of partial differential equations using the `deal.II` library [1]. It is based on the deal2lkit library, and it is designed to minimize the time to market of high performance PDE solvers, exploiting automatic differentiations, and allowing the users to focus on the mathematical formulation of the problem and/or on the design of a preconditioner suited for the specific application.

The code is structured to handle systems of time dependent, non-linear, algebraic-differential equations given in the form $F(y, \dot{y}, t) = 0$.

π-DoMUS features interfaces for advanced non-linear solvers and time steppers, provided by the `Sundials` library [4]. Jacobian matrices are assembled using automatic differentiation tools provided by the `Sacado` package of the `Trilinos` library [2], and block preconditioners rely on the new features of the `deal.II` library to build up an expression syntax for matrix and vector operations through the `LinearOperator` class [3]. Several example applications are included.

The solver is distributed under the free GNU Lesser General Public License (LGPL) and is available from the π-DoMUS homepage at `http://github.com/mathLab/pi-DoMUS`.

References

[1] W. Bangerth et al., The deal.II library, version 8.3, *ACM Trans. Math. Softw.* **38** (2015), 24/1–24/27.

[2] M.A. Heroux et al., An overview of the trilinos project, *ACM Trans. Math. Softw.* **31** (2005), 397–423.

[3] M. Maier et al., LinearOperator - a generic, high-level expression syntax for linear algebra, submitted.

[4] A.C. Hindmarsh et al., Sundials: Suite of nonlinear and differential/algebraic equation solvers, *ACM Trans. Math. Softw.* **31** (2005), 363–369.

[*] SISSA - International School for Advanced Studies, `alberto.sartori@sissa.it`

[†] SISSA - International School for Advanced Studies, `mauro.bardelloni@sissa.it`

[‡] SISSA - International School for Advanced Studies, `luca.heltai@sissa.it`

Comparative studies of population balance methods in liquid-liquid flows

Robert Sawko [*] Marcin Traczyk Albert Yiamakis

April 17, 2016

The talk will introduce OpenPBE, an extensible library solving population balance equations (PBE) within OpenFOAM. We discuss the implementation and integration with `twoPhaseEulerFoam` as well the development of a test suite focused on liquid-liquid flows. Several different formulations are presented and compared such as method of classes, method of moments with functional assumptions and preliminary implementation of a quadrature method.

The library is a generalisation of `IATE` diameter model which appeared in the two-phase model implementation starting from version 2.3. It aims to be solver agnostic by minimizing interactions with other components. Effort has been made to incorporate good software engineering practices such as modularisation, automated build and testing.

The set of test cases consists of calculations based on analytic solutions to PBE problems e.g. [**?**]. We then move to comparisons with experiments in oil-water flows where population balance can be used to quantify the size change of droplets due to breakup and coalescence processes. We present comparisons with a breakup dominated case taken from [**?**] and follow up with [**?**] where both coalescence and breakup are appreciable. Moreover, the study incorporates accuracy against performance metrics.

References

[1] R M Ziff and E D McGrady. The kinetics of cluster fragmentation and depolymerisation. *Journal of Physics A: Mathematical and General*, 18(15):3027, 1985.

[2] S Galinat, O Masbernat, P Guiraud, C Dalmazzone, C Noı, et al. Drop break-up in turbulent pipe flow downstream of a restriction. *Chemical engineering science*, 60(23):6511–6528, 2005.

[3] MJH Simmons and BJ Azzopardi. Drop size distributions in dispersed liquid–liquid pipe flow. *International journal of multiphase flow*, 27(5):843–859, 2001.

[*] Cranfield University, `r.sawko@cranfield.ac.uk`

NGS-Py: Netgen/NGSolve with a new Python front-end

Joachim Schoeberl and Matthias Hochsteger

In this talk we present the new Python steering for Netgen/NGSolve. NG-Solve provides the mathematical objects, finite-element space (with hp-spaces for scalar, vectorial and tensorial function spaces), forms and preconditions. Netgen includes geometric modeling (from Python) and mesh generation. Differential forms are parsed from Python and represented as C++ expression trees. We show examples including non-linear problems, space-time problems via tent-pitching, and present the overall performance.

Coupling The Finite And Boundary Element Methods With FEniCS And BEM++

Matthew Scroggs[*] Timo Betcke[†] Erik Burman[‡]

April 14, 2016

Boundary element simulations are frequently used within engineering to simulate problems in homogeneous unbounded domains, for example acous- tic scattering from a car. However, in many realistic applications small localised inhomogeneities can appear. In these situations, it is necessary to couple finite element method (FEM) and boundary element method (BEM) subproblems efficiently.

In recent years, we have developed at UCL a fast boundary element package called BEM++ [4] which provides a simple Python interface based on a fast C++ computational kernel. In this talk we discuss interfacing BEM++ with FEniCS [1], a popular FEM library, for coupled FEM/BEM simulations. The goal is to provide a simple but powerful Python based framework to setup and solve challenging FEM/BEM coupled problems in a variety of applications.

References

[1] M. Alnæs, J. Blechta, J. Hake, A. Johansson, B. Kehlet and A. Logg, C. Richardson, J. Ring, M. Rognes, G. Wells, The FEniCS project version 1.5, *Archive of numerical software* **3:100** (2015).

[2] R. Hiptmair, P. Meury, Stabilized FEM-BEM coupling for Helmholtz transmission problems, *SIAM journal on numerical analysis* **44(5)** (2006), 2107–2130.

[3] C. Johnson, J.C. Nédélec, On the coupling of boudary integral and finite-element methods, *Mathematics of computation* **35(152)** (1980), 1063–1079.

[4] W. Śmigaj, T. Betcke, S. Arridge, J. Phillips, M. Schweiger, Solving boundary integral problems with BEM++, *ACM Transactions on mathematical software* **41(2)** (2015).

[*] University College London, `matthew.scroggs.14@ucl.ac.uk`
[†] University College London, `t.betcke@ucl.ac.uk`
[‡] University College London, `e.burman@ucl.ac.uk`

Icepack: Finite Element Modeling of Glaciers and Ice Sheets

Daniel Shapero[*] Ian Joughin[†]

March 28, 2016

In this talk, I will describe my work on the glacier modeling library *icepack*, developed as part of my dissertation work. The flow of ice sheets and glaciers is governed by a nonlinear system of partial differential equations for the ice thickness, velocity, temperature, and possibly also the subglacial hydrology. Many of these quantities are not readily observable through satellite or airborne remote sensing. Consequently, we must turn to techniques from data assimilation to infer the current state of a glacier before making predictions of future ice extent. These data assimilation problems are naturally formulated in terms of PDE-constrained optimization.

In addition to solving differential equations and optimization problems, a library for modeling glacier flow must have an API that glaciologists can use without years of experience programming in C++. Satisfying the (at times mutually-exclusive) constraints of usability and flexibility has dictated many of the design choices in *icepack*, such that it differs in many respects from other software packages for modeling ice sheets. Finally, *icepack* is a work-in-progress; I will describe future directions for *icepack*, and features of the general-purpose finite element library deal.II that have been or will be useful.

Icepack is open-source software, available at www.github.com/danshapero/icepack.git.

[*] Applied Mathematics, University of Washington, `shapero@uw.edu`

[†] Applied Physics Lab, University of Washington, `ian@apl.washington.edu`

OpenPhase - the open source phase field simulation library

Oleg Shchyglo[*]

April 22, 2016

We present an open source phase field simulation library "OpenPhase" which is based on a multi-phase field multi-component model of I. Steinbach [1]. The library is intended for high quality quantitative simulations of processes that involve structural and phase transformations. It contains modules that allow simultaneously solving the multi-phase field equations for interface motion, the equations for advection and diffusion of different chemical components, Lattice-Boltzmann solver for liquid phase flow including the interaction with solid structures, and large strain elasto-plastic problem solvers. The modular structure of the OpenPhase library allows easy extension and modification by adding new modules. Code structure and simulation examples related to metallurgical processes will be presented.

The OpenPhase library is distributed in the form of an open source code which is available at www.OpenPhase.de under the GNU GPL v3.

References

[1] I. Steinbach, Phase-field models in materials science, Modelling Simul. Mater. Sci. Eng. **17** (2009) 073001.

[*] Interdisciplinary Centre for Advanced Materials Simulation (ICAMS), Ruhr University Bochum, Germany, `oleg.shchyglo@rub.de`

Partitioned Fluid Structure Interactions for Stokes Flow Problems

David Wells, Jeff Banks and Fengyan Li

Fluid structure interactions, where a fluid equation and a structural equation are coupled at an interface, are a frequent type of problem encountered in various fields like geophysics, flows in biological media, and detonation modeling. These problems frequently require complex mesh data structures, mesh moving, and sophisticated linear solvers.

In this talk we discuss some new results for coupling, with a partitioned scheme, an elastic beam equation to a fluid governed by the Stokes equations. Both domains are discretized with the finite element method.