Discrete Adjoint Approaches for CHT Applications in OpenFOAM
Euro AD 2020

Dr. Markus Towara

Virtual Argonne, August 11th 2020
Outline

OpenFOAM intro

Discrete adjoint OpenFOAM

CHT Application

WIP / Summary, Outlook

Summary & Outlook
OpenFOAM Softwarepackage

- Open-Source CFD software package (GPLv3)
- OpenFOAM.com / OpenFOAM.org / foam-extend
- Finite volume discretization on 3D unstructured meshes
- Two major releases a year
- Cell centered physical quantities (mostly)
- Highly complex C++ code, heavily relying on inheritance and templates
- Code: \( \sim \) 1M LOC, 9k files
- Minimal external dependencies
- Parallelization using MPI
Motivation: Topology Optimization

- Augment NS momentum equations by source term $\alpha u$: [1, 2]

$$ (u \otimes \nabla) u = \nu \nabla^2 u - \frac{1}{\rho} \nabla p - \alpha u $$

- Parameter $\alpha$ allows to penalize cells / regions of the geometry to redirect flow
- Penalty term can be interpreted as porosity, according to Darcy’s law [3]

$$ \frac{\Delta p}{\Delta x} = -\left(\frac{\mu}{\kappa}\right)u $$
How to find appropriate penalty field $\alpha$

- Define cost function $J$, e.g. total power loss between inlet and outlet:
  $$J = - \int_{\Gamma} \left( p + \frac{1}{2} \| u \|^2 \right) u \cdot n \, d\Gamma$$

- Calculate sensitivity of the cost function w.r.t. parameters $\alpha_i$
  $$\frac{dJ}{d\alpha_i} = ???$$

- Calculate an updated porosity field $\alpha^{n+1}$, e.g. using gradient descent:
  $$\alpha_i^{n+1} = \alpha_i^n - \lambda \cdot \frac{dJ^n}{d\alpha_i^n}, \text{ with constraints } 0 \leq \alpha_i \leq \alpha_{\text{max}}$$

- Loop until $\alpha$ converged...
Features of Discrete Adjoint OpenFOAM

Groundworks:
▶ Currently based on OpenFOAM v2006
▶ Typedef approach allows to differentiate the whole simulation code (Black-Box)
▶ For practical applications not feasible, further optimizations are needed
▶ A partial list of optimizations and features enabled by them are listed below

Algorithmic optimizations:
▶ Checkpointing [4]
▶ Reverse accumulation and Piggy-backing [5, 6]
▶ Symbolic differentiation of embedded linear solvers (SDLS) [7, 8]
▶ Adjoints of MPI parallelism by Adjoint-MPI [9, 8]
Our Goal: Optimization of cooling geometries for HPC infrastructure, utilizing excess heat for e.g. office heating.

Playground for:
- Topology optimization
- Shape optimization
- Parameter studies

Possible (water) cooling solution for a server blade
CHT Equations

Fluid domain:

\[(\mathbf{u} \otimes \nabla) \mathbf{u} = \nu \nabla^2 \mathbf{u} - \frac{1}{\rho} \nabla p + \mathbf{b} \]
\[\nabla \cdot \mathbf{u} = 0\]

\[\nabla \cdot (\rho c_p \mathbf{u} T) = \nabla \cdot (k \nabla T) + \dot{q}_F\]

Solid domain:

\[k \nabla^2 T = - \dot{q}_S\]

In OF: Solve fluid and solid domain separately (repeatedly), implicitly coupled through boundary temperatures and heat fluxes\(^1\).

\(^1\)fully coupled solver available in Foam-extend
Requirements for solver, exceeding previously implemented functionality:

- Discrete adjoint solver based on `chtMultiRegionSimpleFoam`
- Implement checkpointing for multiple mesh regions
- Identified multiple issues with old checkpointing approach due to hidden members in boundary conditions
- Manually add members to checkpoint depending on boundary patch type (`Foam::isA<T>`):
  - `gradient_` in `Foam::fixedGradientFvPatchScalarField`
  - `valueFraction_` in `Foam::mixedFvPatchField<T>`
- Need solid tools to identify such issues before going into production
- Alternatively use a semi-automatic checkpointing approach [10]
Case study: topology optimization with heat transfer

- Simple case, based on a tutorial case of foam-extend ported to OpenFOAM
- Fixed temperature profile at top of solid (400K from 25% to 50% of length, 323.5K else)
- Convergence of primal highly dependent on relaxation factor of solid temperature
- Cost function a blend of pressure loss and average outlet temperature
- Want to increase convection ($Pr$) near wall
- Run piggyback optimization
Optimization result
- Allow Temperature diffusion within porous medium
- Coupled solvers for better convergence
- Use new optimization framework included with OpenFOAM v2006 (used for continuous adjoints in vanilla OpenFOAM)
Summary

▶ Algorithmic differentiation of highly complex codebase yields flexible discrete adjoint toolkit.
▶ Efficiency optimizations introduced to make real world applications feasible.
▶ Adjoint workflow natively applicable to complex CHT applications.
▶ Discrete adjoint OpenFOAM is available under the GPL-v3 https://stce.rwth-aachen.de/foam.

Acknowledgement
This work is part of ZIM project Entwicklung optimierter Kühlgeometrien mittels adjungierter Simulationsmethoden für die Direkt-Heißwasserkühlung von Rechenzentren (Development of optimized cooling geometries for hot watercooling of data centers using adjoint simulation methods). It is supported by the Federal Ministry for Economical Affairs and Energy (BMWi), on the basis of a decision by the German Bundestag.
T. Borrvall and J. Petersson.
Topology optimization of fluids in Stokes flow.

C. Othmer.
A continuous adjoint formulation for the computation of topological and surface sensitivities of ducted flows.

S. Whitaker.
Flow in porous media i: A theoretical derivation of Darcy’s law.

A. Griewank and A. Walther.

B. Christianson.
Reverse accumulation and attractive fixed points.
Andreas Griewank and Christèle Faure.  
Piggyback differentiation and optimization.  

M. B. Giles.  
Collected matrix derivative results for forward and reverse mode algorithmic differentiation.  

MPI-parallel discrete adjoint OpenFOAM.  

M. Schanen.  
*Semantics Driven Adoints of the Message Passing Interface*.  
Dissertation, RWTH Aachen University, 2014.
M. Towara, J. Lotz, and U. Naumann.
Discrete adjoint approaches for cht applications in OpenFOAM.