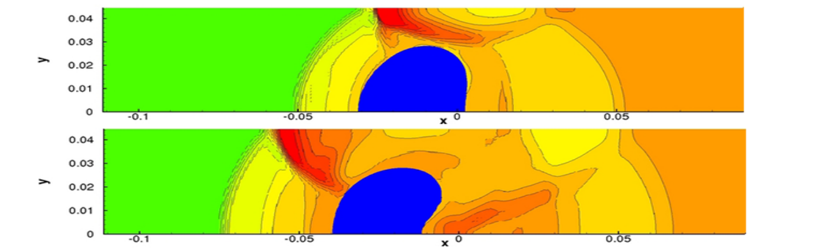
UNIVERSITY OF TWENTE.

MACS MSM



Brief Overview of hpGEM

A. R. Thornton

hpGEM Package

http://hpGEM.org

hpGEM: A software library for DGFEM;
enabling easy and rapid application development.

hpGEM Package

http://hpGEM.org

hpGEM: A software library for DGFEM;
enabling easy and rapid application development.

- Goal: provide general and frequently needed data structures and methods to implement DGFEM algorithms

hpGEM Package

http://hpGEM.org

hpGEM: A software library for DGFEM;
enabling easy and rapid application development.

- Goal: provide general and frequently needed data structures and methods to implement DGFEM algorithms
- Philosophy:
  - Support wide variety of mesh types and problems
  - Provide convenient interfaces to mesh sources and integrators
  - Developed using object-oriented programming technique leading to flexible and reusable software
  - Written using C++ using objects, inheritance and templates.

Basics of DG

We will consider the simple hyperbolic equations:

$$\frac{\partial \boldsymbol{u}}{\partial t} + \nabla \boldsymbol{F}(\boldsymbol{u}) = 0$$

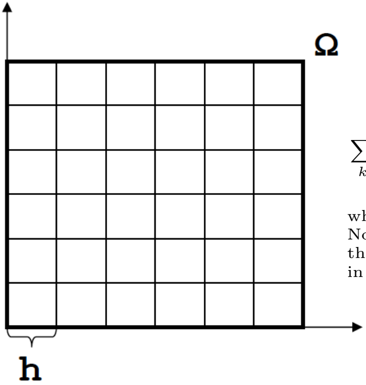
Multiple by a test function V and integrate over a domain Ω

$$\int_\Omega V \frac{\partial \boldsymbol{u}}{\partial t}\,\mathrm{d}\Omega + \int_\Omega V \nabla \boldsymbol{F}(\boldsymbol{u})\,\mathrm{d}\Omega = 0$$

Rearranging the second term (integration by parts) gives

$$\int_\Omega V \frac{\partial \boldsymbol{u}}{\partial t}\,\mathrm{d}\Omega = \int_\Omega \nabla V \boldsymbol{F}\,\mathrm{d}\Omega - \int_{\partial\Omega} V F\,\mathrm{d}S$$

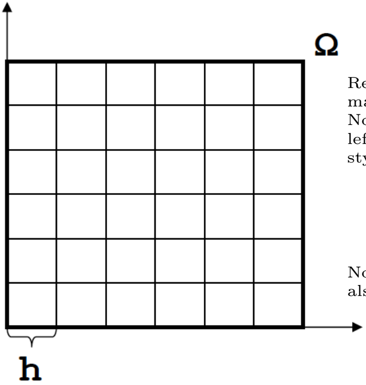
On to a grid



$$\sum_k \int_{\Omega_k} V_j \frac{\partial \boldsymbol{u}}{\partial t} \, \mathrm{d}\Omega = \sum_k \int_{\Omega_k} \nabla V_j \boldsymbol{F} \, \mathrm{d}\Omega - \int_{k_{\partial\Omega}} \boldsymbol{H} \, \mathrm{d}S$$

where $\boldsymbol{H} = V_k \boldsymbol{F}$.
Note, the function is not forced to be continuos across the boundary of each volume and V_k can be different in each volume.

On to a grid



Replace the face integral with a numerical approximation, $\mathcal{H}$.
Now for the internal faces, the values obtained on the left and right must be the same i.e. Lax-Friedrichs style flux

$$\mathcal{H} = \frac{1}{2}\left(\boldsymbol{H}(\text{left}) + \boldsymbol{H}(\text{right})\right) + \Phi$$

Normally take the basis functions V_k to be polynomials of order p.

Why Discontinuous Galerkin FEM?

- Well suited to handle complicated geometries.
- Ability of local refinement.
- Preservation of local conservation properties.
- Order of accuracy depends on the degree of polynomial approximation.
- Easily handle adaptivity strategies: h- and p-adaptivity.
- Highly parallelisable.
- Can deal with shocks (in the solution and properties).

History of hpGEM

- 2003 : Project started
- 2007 : First internal alpha released
- 2010 : First public release
- 2012 : Version 2 kernel conceived
- 2013 : Work started on new kernel
- 2014 : Beta release of hpGEM 2 kernel

Main features of hpGEM 2.0 I

- Unstructured mixed mesh support
  - 1D mesh support : lines
  - 2D hybrid mesh support : triangles and quadrilaterals
  - 3D hybrid mesh support : tetrahedra, pyramids, prisms and hexahedra
  - 4D cubic mesh support
- Centaur mesh readers for 2D and 3D
- Rectangular and tetrahedral mesh generators (in 1D, 2D and 3D)
- Basic moving mesh support
- Space and space-time support
- Gauss integration rules upto, at least, seventh order (most 11^{th}) for all supported polytopes (n-dimensional shapes)

Main features of hpGEM 2.0 II

- Tecplot discontinuous output routines
- Global algebraic system assembly
- Element data caching
  - By default face normals, physical gradient of basis functions, Jacobian of the mapping,
  - Extra user definition possible
- Predefined sets of basis functions.
  - H1 conforming (order 1-5)
  - DG-H1 conforming (order 1-5)
  - Monomials
  - DG-Curl conforming (Tetahedra only)
- P-refinement (i.e. different basis functions can be used in each element)

Main features of hpGEM 2.0 III

- H-refinement
- Multigrid
- Easy cross platform building via CMake
- Complete doxygen documentation (can be build or viewed via the website)
- Self test suite
    - UnitTests: Test individual features
    - ProblemTests: Test the features together
- Series of tutorials
    - Simple DG
    - Simple application
    - Use of advanced features

Main features of hpGEM 2.0 IV

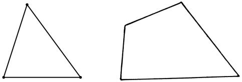
  - Kernel walk through (for the very advanced user who wants to extend the kernel)
- Web builder tool, allows simple point and click generation of code for a series of predefined applications
- Simple API kernel interfaces so applications can be created with minimal user code.
  - hpGEMUI : Flexible.
  - hpGEMUISimplfied : Quick and easy for standard DG-applications
- Able to couple with open-source particle solver MercuryDPM (http://MercuryDPM.org)

Manage 1D - 4D problems

- 1D: line

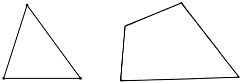
Manage 1D - 4D problems

- 1D: line
- 2D: triangle, quadrilateral

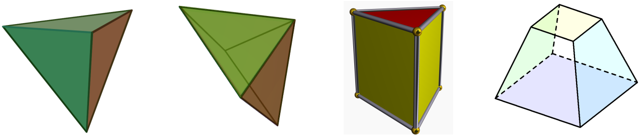


Manage 1D - 4D problems

- 1D: line
- 2D: triangle, quadrilateral

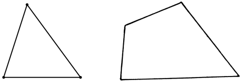


- 3D: tetrahedra, pyramids, triangular prisms, hexahedra

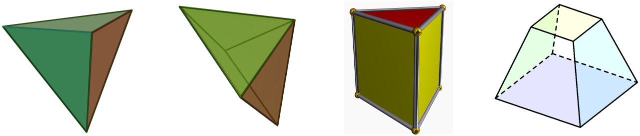


Manage 1D - 4D problems

- 1D: line
- 2D: triangle, quadrilateral

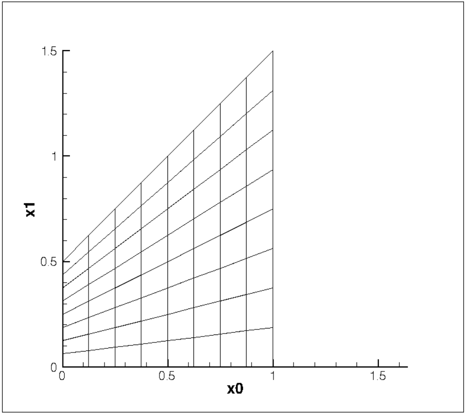


- 3D: tetrahedra, pyramids, triangular prisms, hexahedra



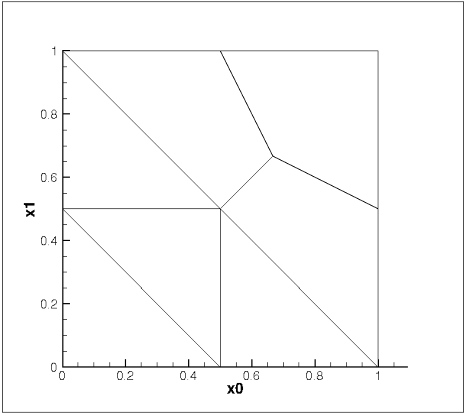
- 4D: space + time : Limited 4-cube support

Structured



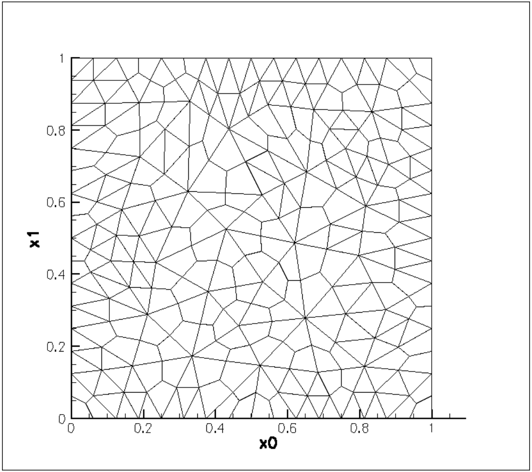
- Created with rectangular mesh generator
- Mesh mover used to move the nodes

Unstructured mixed mesh support

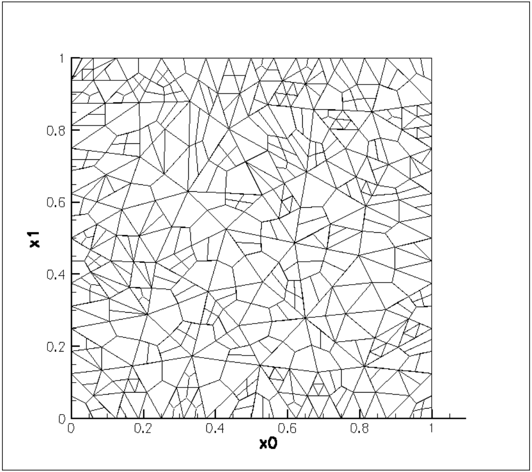


- Created in centaur
- Read and outputted in tecplot format by hpGEM

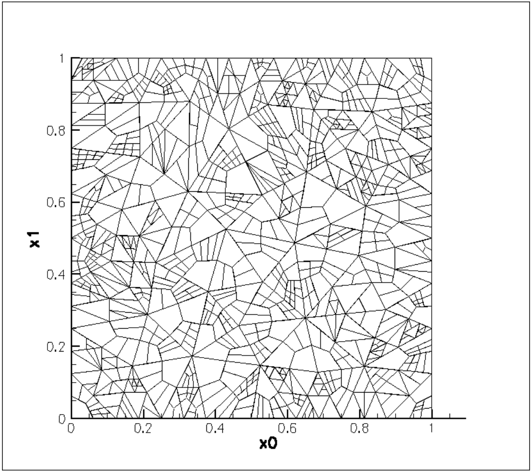
Random h-refinement



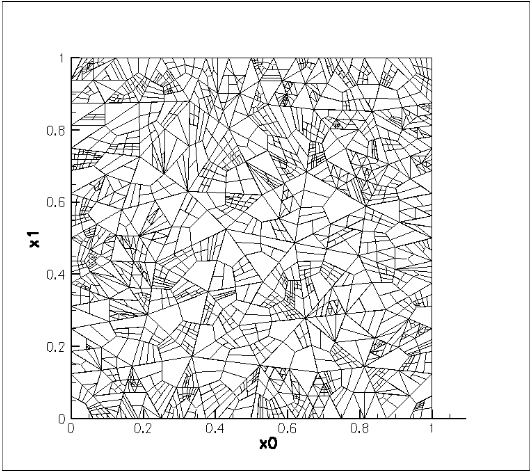
Random h-refinement



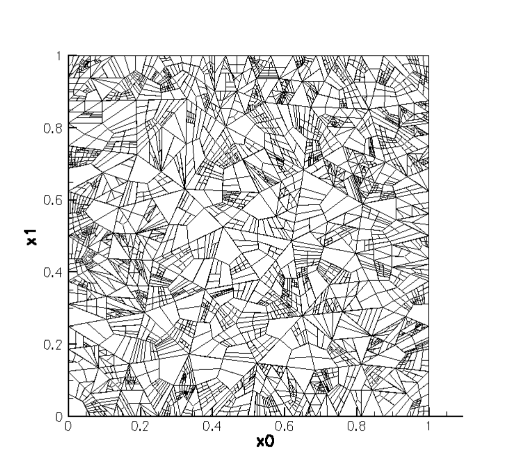
Random h-refinement



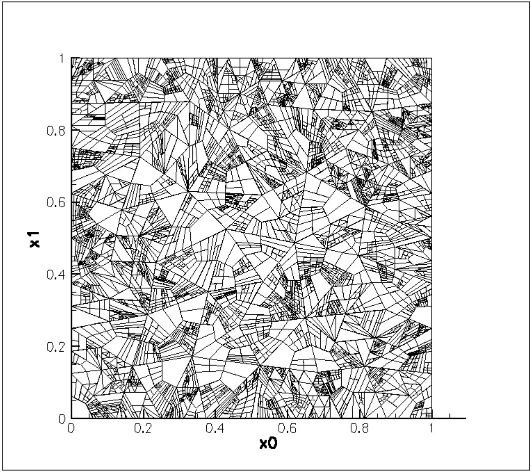
Random h-refinement



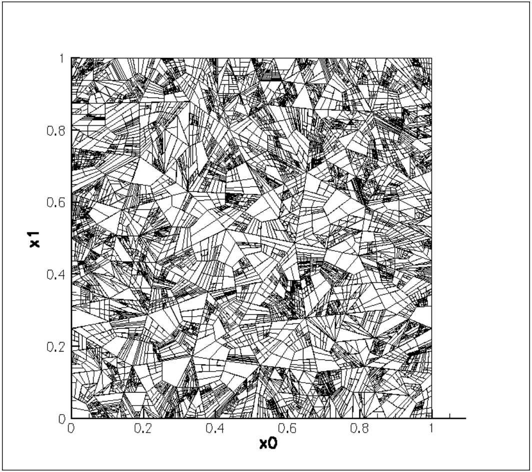
Random h-refinement



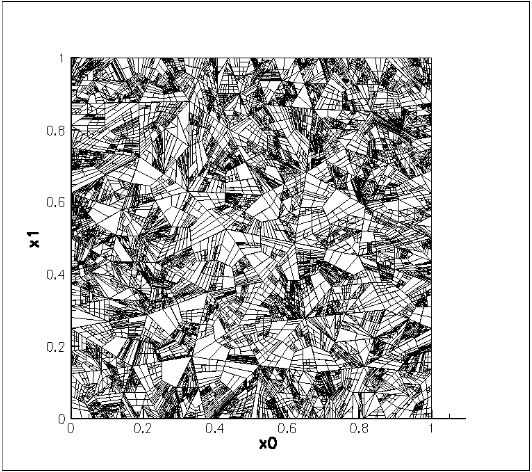
Random h-refinement



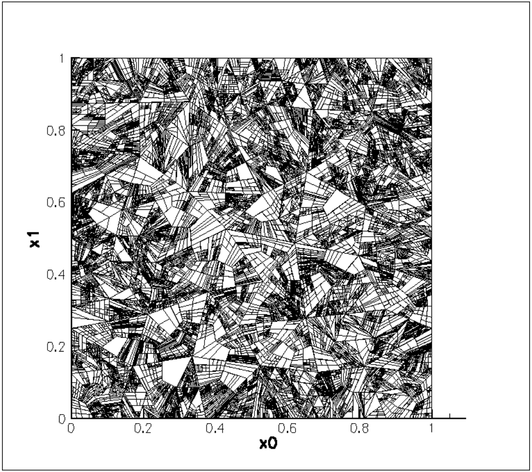
Random h-refinement



Random h-refinement



Random h-refinement



Simple implementation

Creating a mesh

```
RectangularMeshDescriptorT description(DIM);
for(int i=0;i<DIM;++i)
{
    description.bottomLeft_[i]=0;
    description.topRight_[i]=1;
    description.numElementsInDIM_[i]=100;
    description.boundaryConditions_[i]=RectangularMeshDescriptorT::SOLID_WALL;
}
addMesh(description,Base::TRIANGULAR,1,1,1,1);
meshes_[0]->setDefaultBasisFunctionSet(Utilities::createDGBasisFunctionSet2DH1Triangle(3));
```

Simple implementation

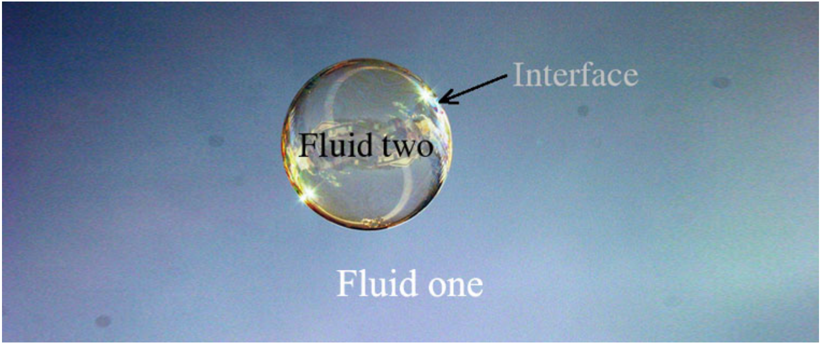
Defining the element integral

```
void elementIntegrand( const ElementT* element, const PointReferenceT& p, LinearAlgebra::Matrix& ret)
 {
    int n=element->getNrOfBasisFunctions();
    LinearAlgebra::NumericalVector phiDerivI(DIM),phiDerivJ(DIM);
    ret.resize(n,n);
    for(int i=0;i<n;++i)
    {
        element->basisFunctionDeriv(i,p,phiDerivI);
        for(int j=0;j<=i;++j)
        {
            element->basisFunctionDeriv(j,p,phiDerivJ);
            ret(j,i)=phiDerivI*phiDerivJ;
            ret(i,j)=ret(j,i);
        }
    }
}
```

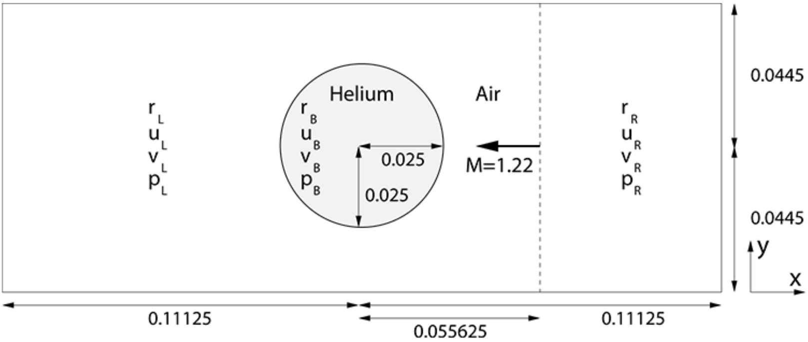
Developed applications

- Two phase flow (Space-Time)
- Maxwell
- Pitman-Li model (Fluid saturated grains)
- Shallow-Water
- Navier-Stokes
- Laplace
- Hamiltonian-Euler
- Linear Potential Flow (available shortly)
- Non-linear Potential Flow
- Granular segregation
- Shallow granular flows

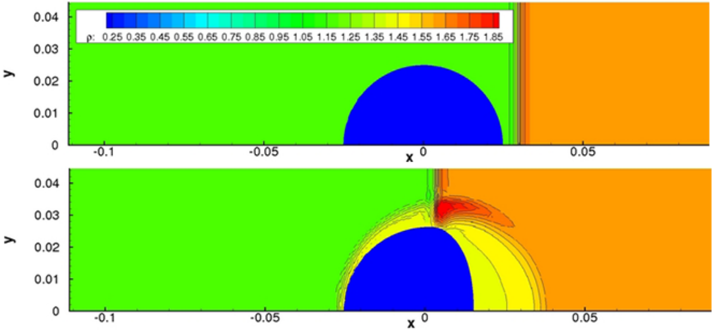
Two Fluid Model



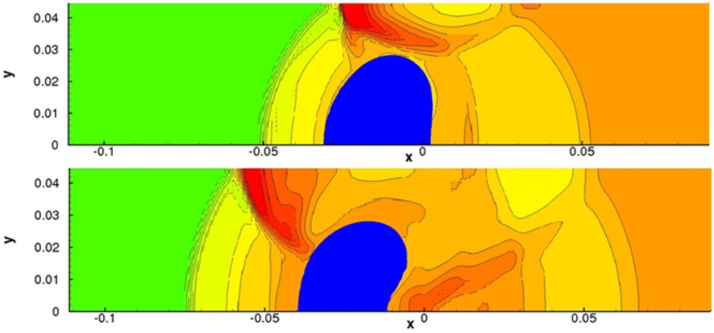
Two Fluid Model



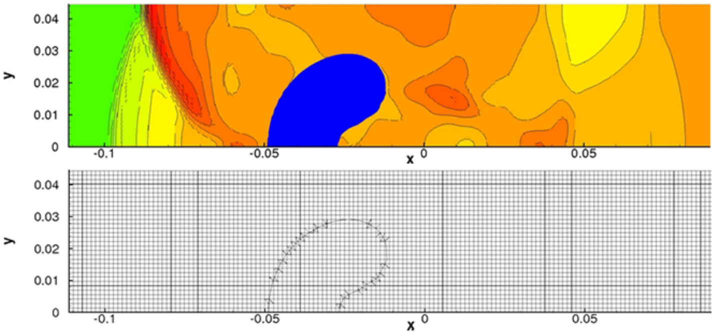
Two Fluid Model



Two Fluid Model



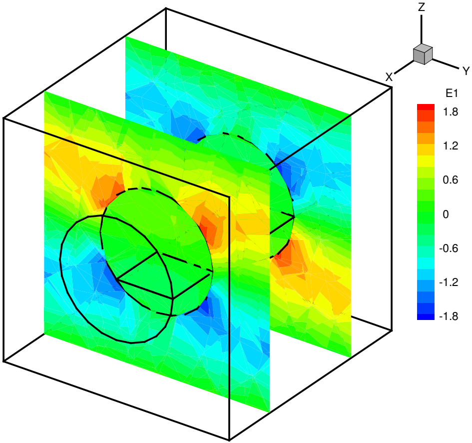
Two Fluid Model



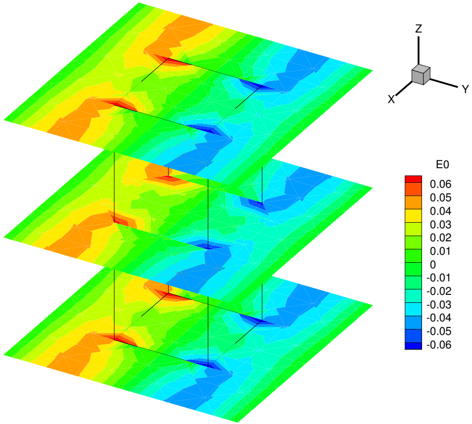
Non-linear Potential Flow

Movie loading please wait

Maxwell



Maxwell



Meet the team

http://hpgem.org/about-the-code/team

Future of hpGEM

- Parallelisation of kernel
- Simpler wrapper tool for specialised equations
- Integration with open-source plotters and mesh generators
- Coupling with Mercury-DPM