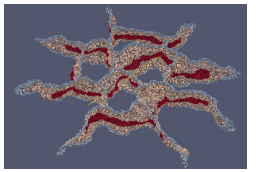
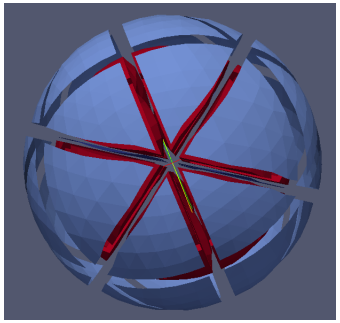
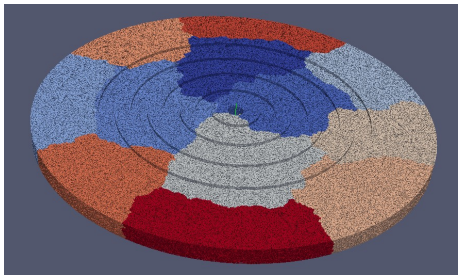
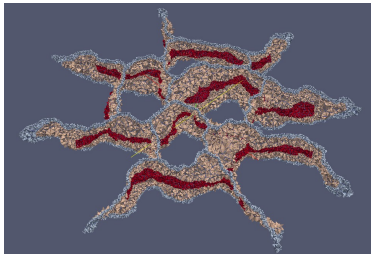
Dune-Curvilinear Grid









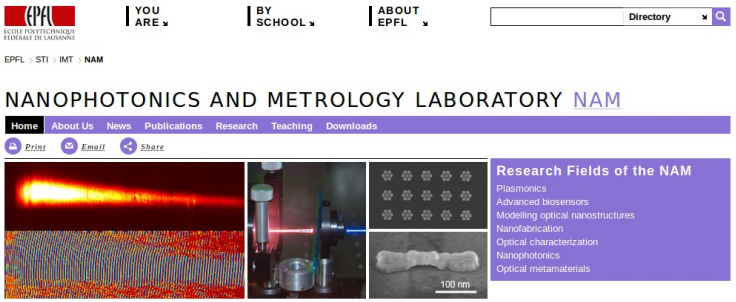
Aleksejs Fomins

and

Benedikt Oswald

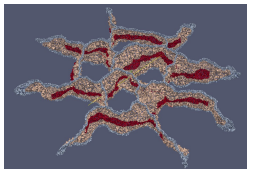
Nanophotonics and Metrology Laboratory, EPFL

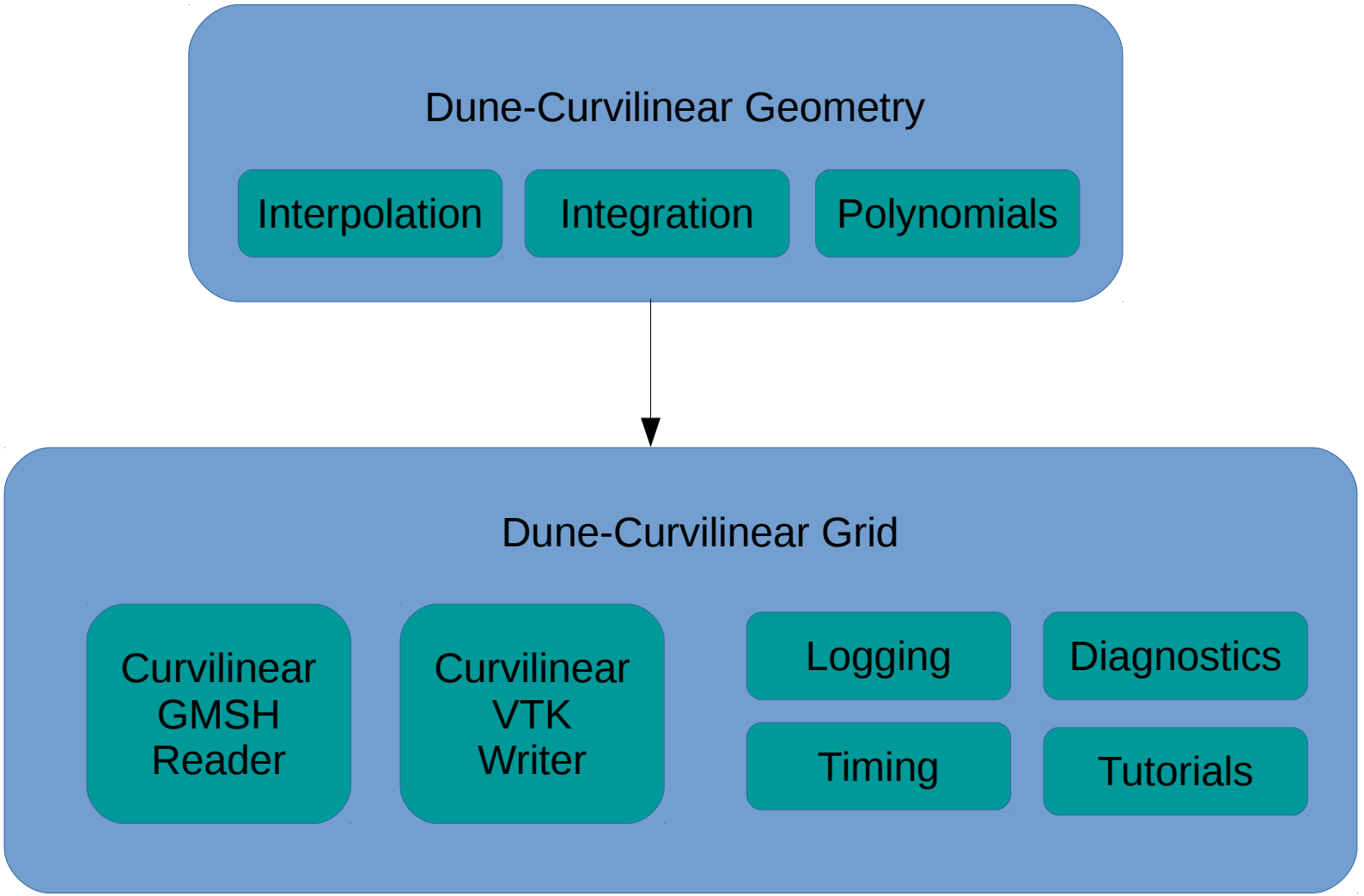
LSPR AG, Grubenstrasse 9, 8045 Zurich



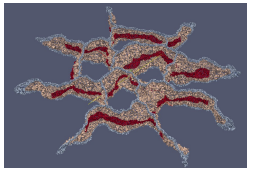
LSPR AG

Overview




Dune-Curvilinear Geometry
Interpolation
Integration
Polynomials
Dune-Curvilinear Grid
Curvilinear GMSH Reader
Curvilinear VTK Writer
Logging
Diagnostics
Timing
Tutorials

Introduction



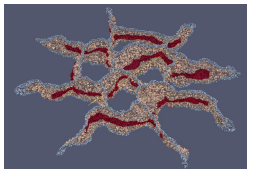
Who

- Aleksejs Fomins and Benedikt Oswald, LSPR AG
- PhD Supervisor: Prof Olivier Martin, EPFL

Why

- PhD in Nanophotonics & Metrology Lab, EPFL
- Accurate and efficient electromagnetic modelling
- Guiding design of commercial biosensors

License Details



Dune-Curvilinear Grid

- property of LSPR AG
- **open source and free for non-commercial use**
- commercial use is negotiable

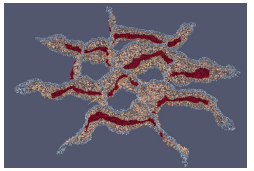
Contributions

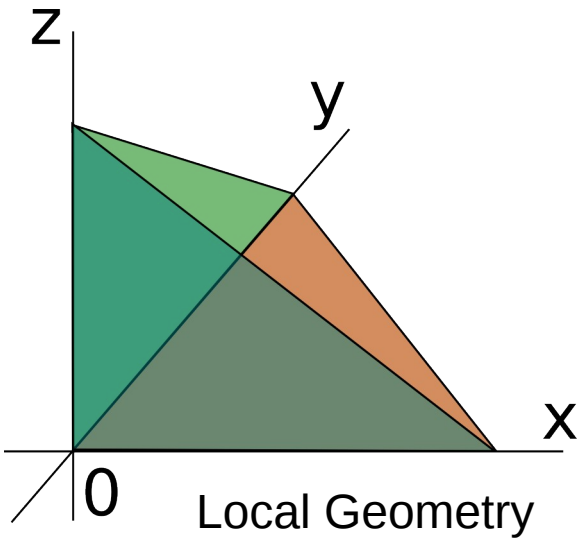
- are welcome
- need to go through LSPR AG

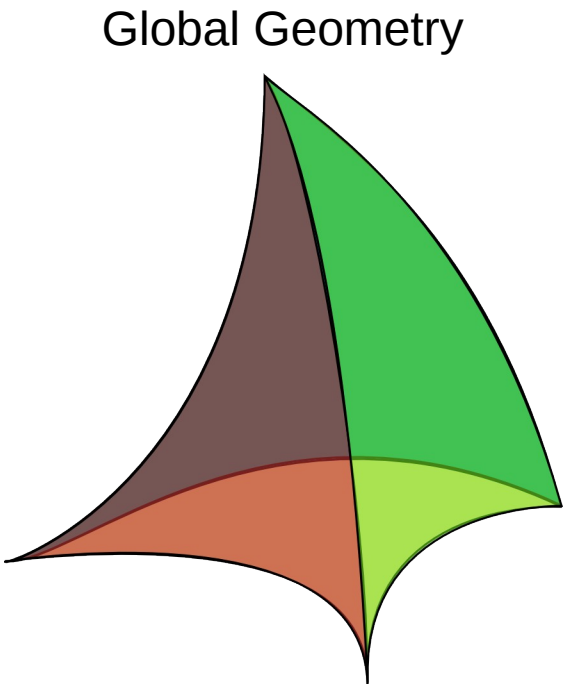
http://www.github.com/lspr-ag/dune-curvilineargeometry

http://www.github.com/lspr-ag/dune-curvilineargrid

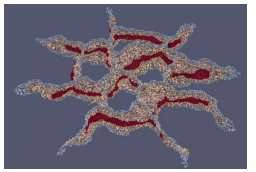
Curvilinear Geometry







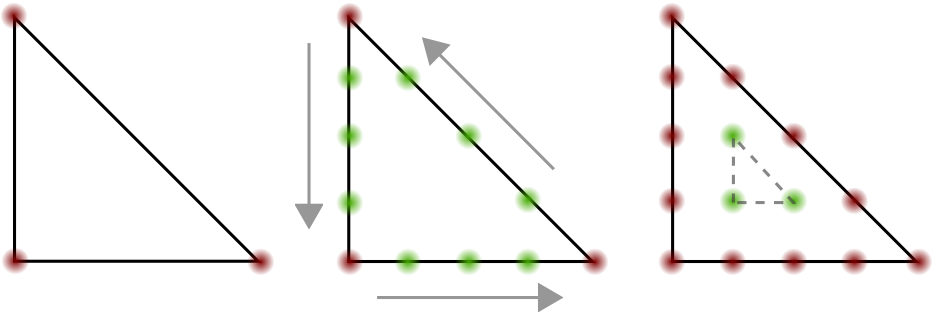
Curvilinear Geometry (1)



Gmsh curvilinear vertex numbering

Vertices → Edges → Faces → Internal recursive

(AKA very complicated)



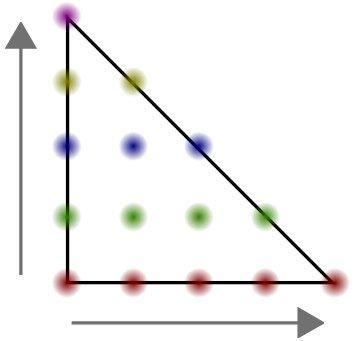
- Dune curvilinear vertex numbering

for(z = 0 to order)

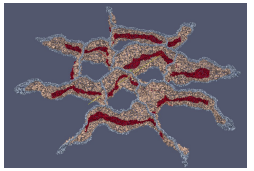
for(y = 0 to order - z)

for(x = 0 to order - z - y)

makepoint(x, y, z)

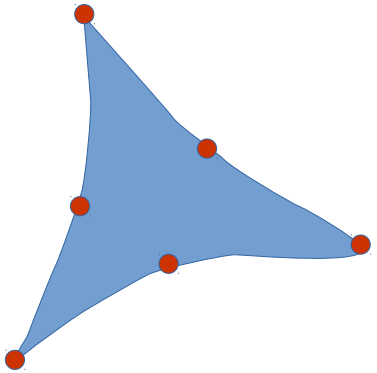


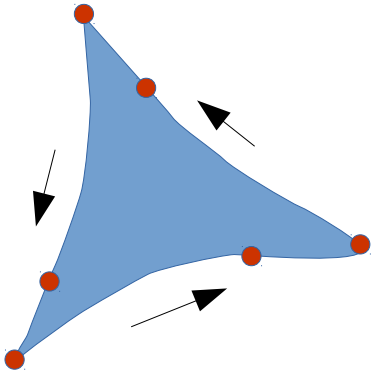
Curvilinear Geometry (2)



Gmsh vertex selection strategy

[1] A. Johnen, J.-F. Remacle, and C. Geuzaine. Geometrical validity of curvilinear finite elements. Journal of Computational Physics, 233(0):359 – 372, 2013. ISSN 0021-9991

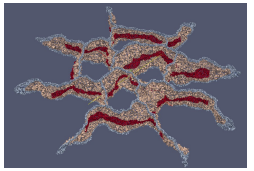




Optimal vertex selection strategy

[2] M. Lenoir. Optimal isoparametric finite elements and error estimates for domains involving curved boundaries. Journal on Numerical Analysis, 23(3):pp. 562–580, 1986. ISSN 00361429.

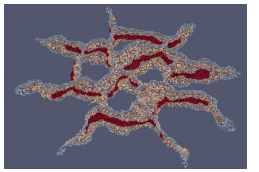
Curvilinear Geometry (3)



Features:

- Arbitrary order geometries via Lagrange Polynomials
- Polynomial manipulation
  - Local2Global map, J, detJ, derivatives, integrals
- Global2Local via Newton method
- Recursive numerical integration
- Vector and matrix simultaneous integration
- Cached Geometries
- Extensive tests

Curvilinear Geometry (4)



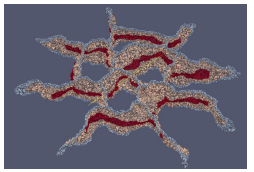
Interface:

```
CurvilinearGeometry ( Dune::GeometryType gt,
                      const std::vector<GlobalCoordinate> &vertices,
                      InterpolatoryOrderType order)
```

```
CurvilinearGeometry ( const ReferenceElement &refElement,
                      const std::vector<GlobalCoordinate> &vertices,
                      InterpolatoryOrderType order)
```

```
CurvilinearGeometry ( const ElementInterpolator & elemInterp)
```

Curvilinear Geometry (5)



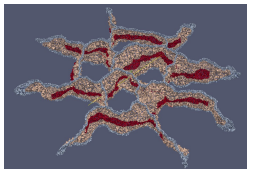
Element Interpolation:

- Up to 5th order simplex polynomials hard-coded
  - Via monomial sums for speed
- Arbitrary order available analytically

One of 56 tetrahedral polynomials of order 5

```
-625.0/24*M[35] - 3125.0/24*M[36] - 3125.0/12*M[38] - 3125.0/12*M[41] - 3125.0/24*M[45] -
625.0/24*M[50] - 3125.0/24*M[37] - 3125.0/6*M[39] - 3125.0/4*M[42] - 3125.0/6*M[46] - 3125.0/24*M[51] -
3125.0/12*M[40] - 3125.0/4*M[43] - 3125.0/4*M[47] - 3125.0/12*M[52] - 3125.0/12*M[44] - 3125.0/6*M[48]
- 3125.0/12*M[53] - 3125.0/24*M[49] - 3125.0/24*M[54] - 625.0/24*M[55] + 625.0/8*M[20] + 625.0/2*M[21]
+ 1875.0/4*M[23] + 625.0/2*M[26] + 625.0/8*M[30] + 625.0/2*M[22] + 1875.0/2*M[24] + 1875.0/2*M[27] +
625.0/2*M[31] + 1875.0/4*M[25] + 1875.0/2*M[28] + 1875.0/4*M[32] + 625.0/2*M[29] + 625.0/2*M[33] +
625.0/8*M[34] - 2125.0/24*M[10] - 2125.0/8*M[11] - 2125.0/8*M[13] - 2125.0/24*M[16] - 2125.0/8*M[12] -
2125.0/4*M[14] - 2125.0/8*M[17] - 2125.0/8*M[15] - 2125.0/8*M[18] - 2125.0/24*M[19] + 375.0/8*M[4] +
375.0/4*M[5] + 375.0/8*M[7] + 375.0/4*M[6] + 375.0/4*M[8] + 375.0/8*M[9] - 137.0/12*M[1] -
137.0/12*M[2] - 137.0/12*M[3] + 1
```

Curvilinear Geometry (5)



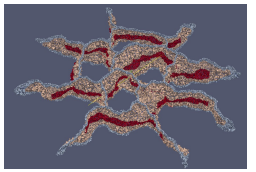
Local Method:

- Returns false if point not inside element
  - (subject to change after this meeting)
- *mydim = cdim:* Via Newton Method
- *mydim < cdim:* Forbidden

```
GlobalCoordinate global ( const LocalCoordinate &local ) const
```

```
bool local  ( const GlobalCoordinate & globalC,
                    LocalCoordinate  & localC) const
```

Curvilinear Geometry (6)



Normals:

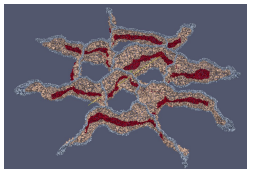
- Now part of Geometry Interface
- Still using inverse Jacobian as in linear case

```
GlobalCoordinate subentityNormal(
    InternalIndexType indexInInside,
    const LocalCoordinate &local) const
```

```
GlobalCoordinate subentityUnitNormal(
    InternalIndexType indexInInside,
    const LocalCoordinate &local) const
```

```
GlobalCoordinate subentityIntegrationNormal(
     InternalIndexType indexInInside,
     const LocalCoordinate &local) const
```

Curvilinear Geometry (7)

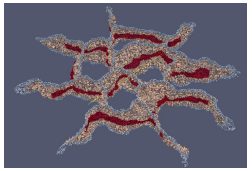


Subentity Geometries:

- Now part of Geometry Interface

```
template<int subdim>
CurvilinearGeometry< ctype, subdim, cdim> subentityGeometry(
   InternalIndexType subentityIndex)
```

Curvilinear Geometry (8)

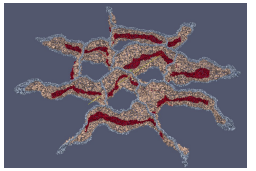


Polynomial Class:

- Stores polynomials as set of Monomials
- Arbitrary order and dimension
- Arithmetics, derivatives, integral over ref. elem.
- Tolerance for clean-up of small terms

```
Polynomial<double, 3> poly3D_test;

poly3D_test += Monomial(2.0, 0, 0, 0);
poly3D_test += Monomial(-3.0, 1, 0, 0);

Polynomial<double, 3> poly3D_test3_1 = poly3D_test.derivative(0);
Polynomial<double, 3> poly3D_test3_2 = poly3D_test.derivative(1);
Polynomial<double, 3> poly3D_test3_3 = poly3D_test.derivative(2);

poly3D_test += Monomial(3,1,2,3);
poly3D_test = (poly3D_test * poly3D_test * 2 - poly3D_test * 5 + 16.5) - 3;
poly3D_test.compactify();


std::cout << poly3D_test.to_string() << std::endl;
std::cout << poly3D_test.evaluate(testEval3D) << std::endl;
std::cout << poly3D_test.integrateRefSimplex() << std::endl;
```

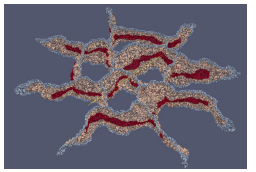
Curvilinear Geometry (10)



Polynomial Class - Purpose:

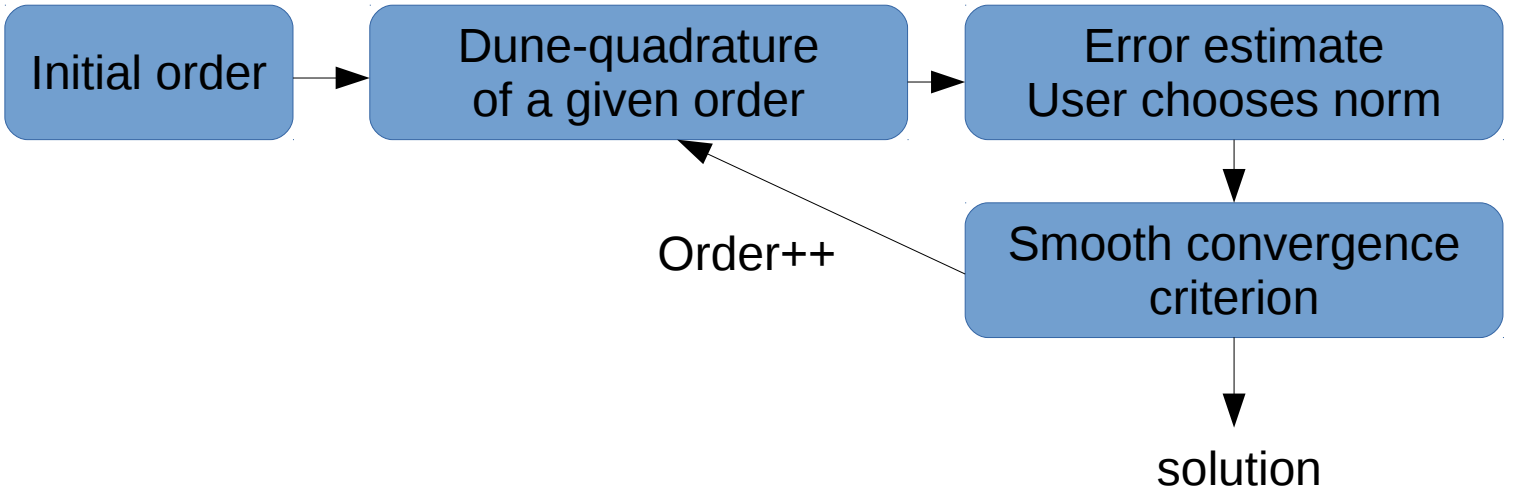
- Element Interpolator – analytical
- Exact integration of polynomial integrands
- DifferentialHelper – Analytic expressions
  - Local2Global Map
  - Det(J)
  - Surface integration normal

Curvilinear Geometry (11)

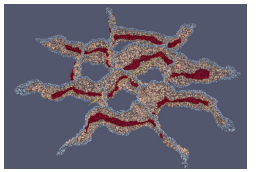


Recursive Integration:

- Reason – non-polynomial Volume
- Integrates Functors of fixed interface
- Can integrate Vectors and Matrices



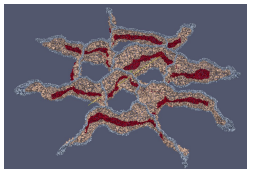
Curvilinear Geometry (12)

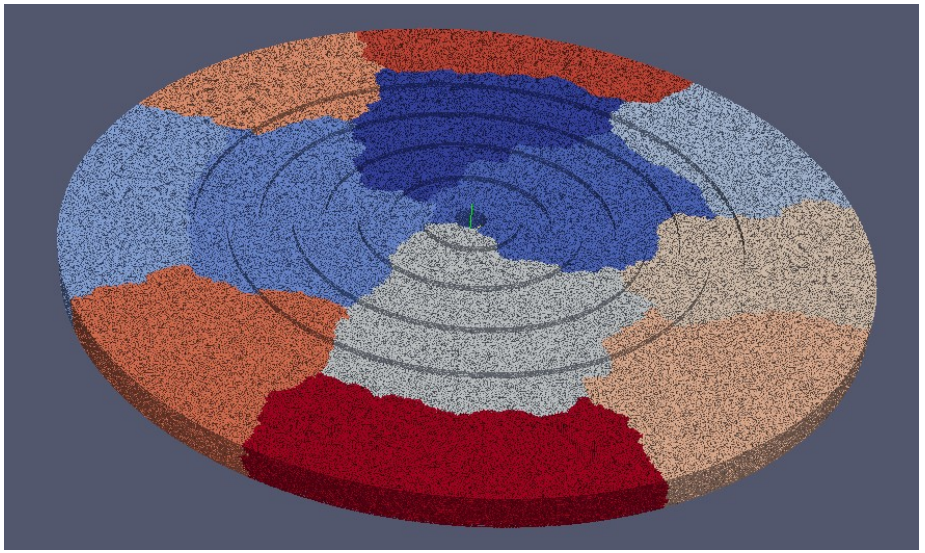


Cached Curvilinear Geometry:

- Pre-computed analytically:
  - Global2Local map
  - Integration Element
- Faster than Non-Cached

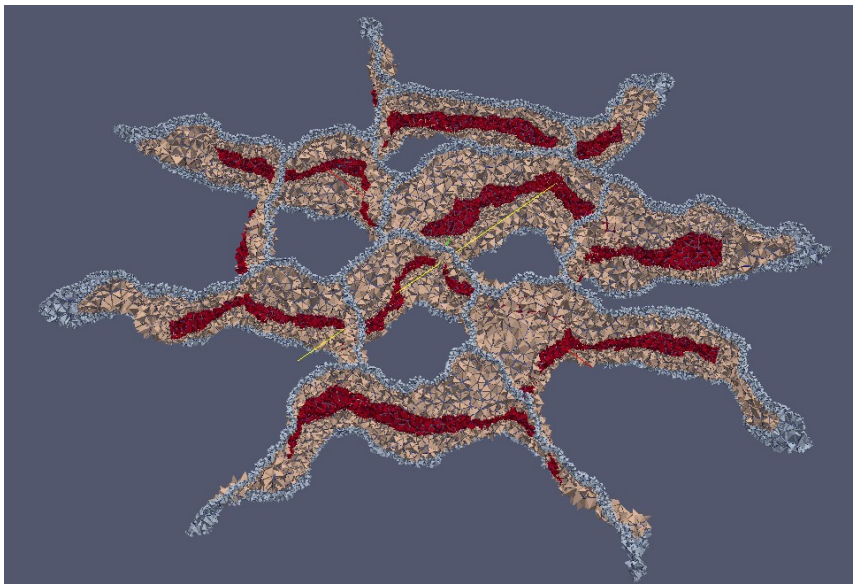
Curvilinear GMSH Reader



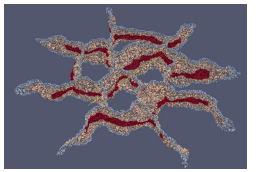


Bullseye lens mesh
4.4 million tetrahedra mesh
Read in parallel
Colorured by process rank...

...Ghost elements
Color-coded by partition type



Curvilinear GMSH Reader (1)

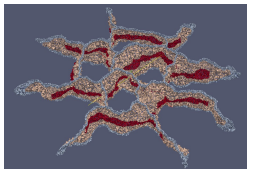


Features:

- Parallel non-bottleneck reading of .msh
- Simplex meshes up to order 5
- Linear meshes work just like they did
- Partitioning via ParMetis [1]
- Optional Parallel VTK output for diagnostics

[1] http://glaros.dtc.umn.edu/gkhome/metis/parmetis/overview

Curvilinear GMSH Reader (2)



Interface-Reader:

```
template <typename FactoryType>
    static void read (FactoryType & factory,
                      const std::string& fileName,
                      MPIHelper &mpihelper,
                      bool useGmshElementIndex = true,
                      bool partitionMesh = true)
```

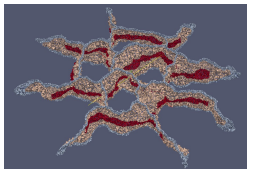
Interface-Factory:

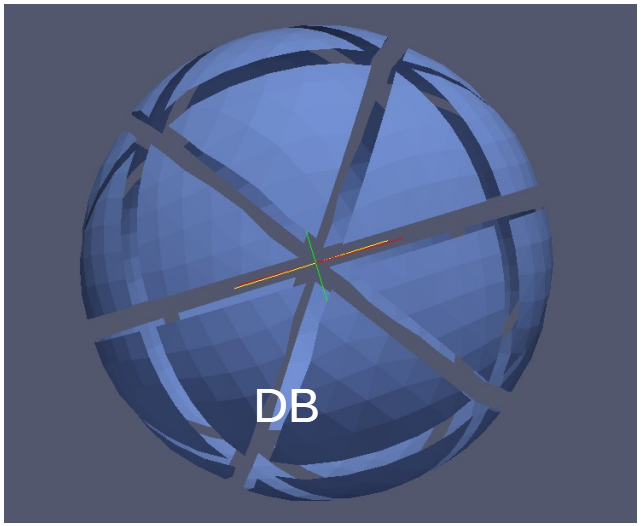
```
void insertVertex ( const VertexCoordinate &pos, const GlobalIndexType globalIndex )
```

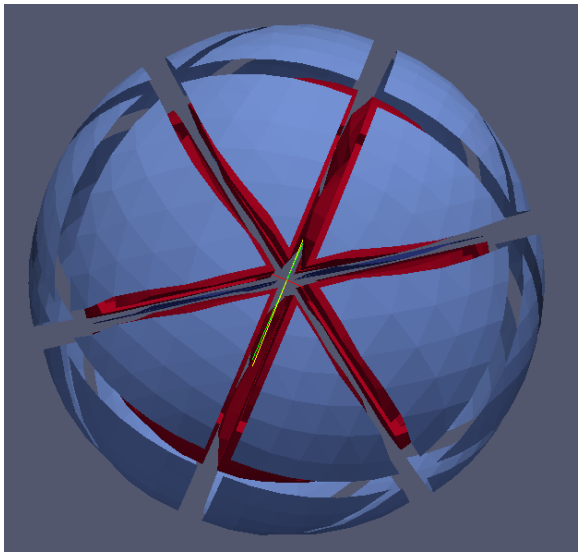
```
void insertElement(
   GeometryType &geometry,
   const LocalIndexVector &vertexIndexSet,
   const GlobalIndexType globalIndex,
   const int elemOrder,
   const int physicalTag)
```

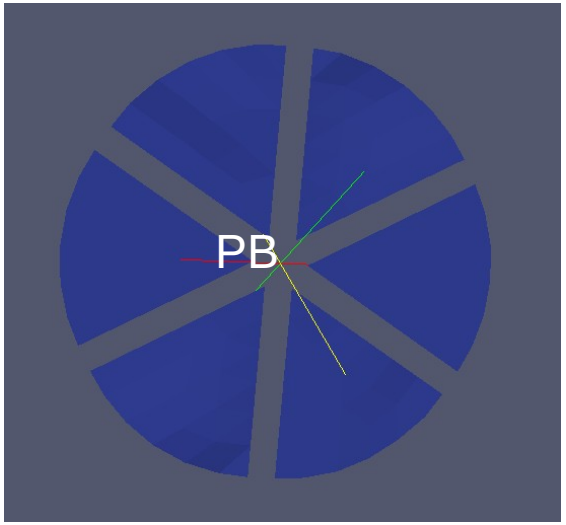
```
void insertBoundarySegment(
   GeometryType &geometry,
   const LocalIndexVector &vertexIndexSet,
   const int elemOrder,
   const LocalIndexType associatedElementIndex,
   const int physicalTag)
```

Curvilinear VTK Writer



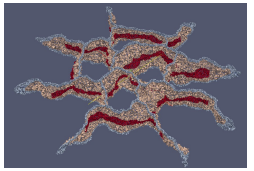






Visualisation of mesh color-coded by partition type

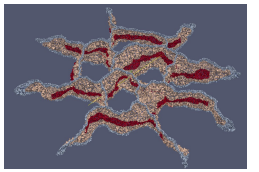
Curvilinear VTK Writer (1)



Features:

- Writes curvilinear simplex meshes
- Writes VTK, VTU and PVTU
- Adjustable virtual refinement
- Scalar: process rank, partition type, physical tag
- Arbitrary number of vector fields
- Extension – CurvVTKGridWriter writes the entire grid to PVTU with a single command

Curvilinear VTK Writer (2)

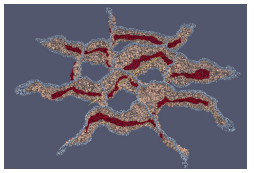


Interface:

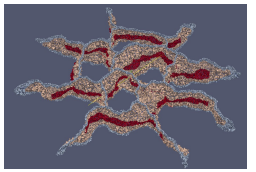
```
void addCurvilinearElement(
            const Dune::GeometryType & geomtype,
            const std::vector<GlobalVector> & nodeSet,
            std::vector<int> & tagSet,
            int elementOrder,
            int nDiscretizationPoint,
            bool interpolate,
            bool explode,
            std::vector<bool> writeCodim)
```

```
template <class VTKElementaryFunction>
    void addField(std::string fieldname, VTKElementaryFunction & vtkfunction)
```

Curvilinear Grid



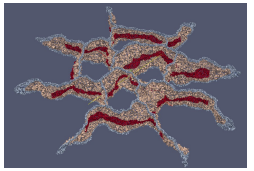
Curvilinear Grid (1)



Features:

- Independent Grid Manager
- Global Index
- Ghost Elements
- DataHandle for all codim
- Tutorials
- Utilities:
  - Grid Diagnostics, LoggingMessage, LoggingTimer
  - Neighbor communication, ParallelDataWriter

Curvilinear Grid (2)



Interface: (Extension wrt dune-grid)

```
GridType * CurvilinearGridFactory::createGrid()
```

```
size_t numBoundarySegments ()
size_t numProcessBoundaries ()
size_t numInternal(int codim)
```

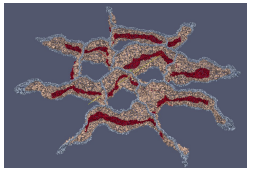
```
template<int codim>
  GlobalIndexType entityGlobalIndex(
    const typename Traits::template Codim< codim >::Entity &entity)
```

```
template<int codim>
  PhysicalTagType entityPhysicalTag(
    const typename Traits::template Codim< codim >::Entity &entity)
```

```
template<int codim>
  InterpolatoryOrderType entityInterpolationOrder(
    const typename Traits::template Codim< codim >::Entity &entity)
```

```
template<int codim>
  typename Codim<codim>::EntityGeometryMappingImpl entityBaseGeometry(
     const typename Traits::template Codim< codim >::Entity &entity)
```

Curvilinear Grid (3)



LoggingMessage:

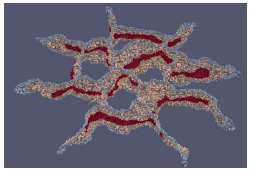
- Parallel Logging
- Verbosity adjustable at compile-time

```
[100%] Communicating complicated face keys to neighbour processors...
[100%] Determining whether the received complicated face candidates exist...
[100%] Storing correct complicated face neighbours...
[100%] Determining faces with missing global indices...
[100%] Sending missing face global indices...
[100%] Marking received face global indices...
[100%] Generating unique corner indices...
[100%] Generating corner iterator list...
[100%] Generating edge iterator list...
[100%] Generating face iterator list...
[100%] Generating element iterator list...
[100%] Generating communication maps...
[100%] Communicating neighbour ranks for different communication protocols...
[100%]  Computing process bounding box...
Fri Sep 25 13:30:17 2015 ::: ...CurvilinearGridBase: Finished generating curvilinear mesh]]
called grid.communicate()
called grid.communicate()
called grid.communicate()
called grid.communicate()
called grid.communicate()
called grid.communicate()
called grid.communicate()
```

LoggingTimer:

- Parallel timing of parts of code
- Sums up several calls of the same snippet
- Statistics of time taken over all processes

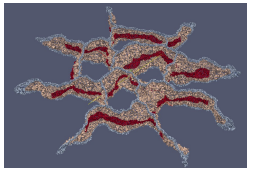
Curvilinear Grid (4)



GridDiagnostics:

- Checks elements for self-intersections
- Statistics on element size, curvature and distribution among processes
- Statistics on grid construction time split between different stages
- Writes output to text file

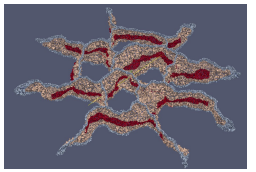
Curvilinear Grid (5)



Tutorials:

- Getting Started
- Traversal
- Visualisation
- Integration
- Communication
- Parallel Data Writing

Known Issues



Curvilinear Geometry

- Global2Local not defined for points outside element
- Global2Local frequently fails on the boundary
- Recursive integral sometimes exceeds order 60

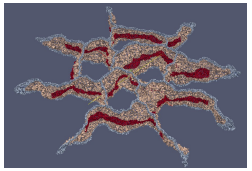
Curvilinear VTK Writer

- Strange holes between curvilinear elements

Curvilinear Grid

- Some DataHandle interfaces (e.g. vertex G→G) have not been tested due to no available tests
- Fails certain gridcheck tests, e.g. local test

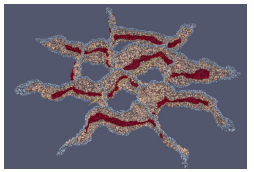
TODO (1/3)



Curvilinear Geometry:

- Changing Geometric Curvature [Faces + Vertex update]
- Hard-code $\partial_i L$ [2-3 days]
- Optimize tests [pass/fail, a week]
- Improve local() [Better Algorithm]
- Exact 2D, 3D quadrature [Exists approx order 20]
- Improve Integration [Basis, Sparse Grids?]
- Other Geometry Types [Subentity maps]

TODO (2/3)



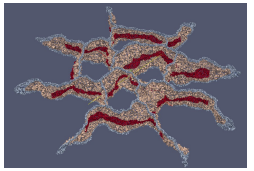
Curvilinear GMSH Reader

- Process Tags for pre-partitioned meshes [1-2 days]
- Other geometry types [GMSH2Dune Mapper]
- Interior and periodic boundaries [Design?]
- Other partition managers

Curvilinear VTK Writer

- Improve performance for meshes with high order
  - Can software (e.g. ParaView) visualize curved elements?
- Other geometry types [Easy given geometry]

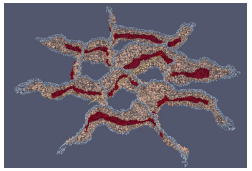
TODO (3/3)



Curvilinear Grid:

- Memory Logging [???]
- Custom communication interfaces [Design?]
- Point Location (OCTree) [Already started...]
- Periodic and interior boundaries [~ a month]
- Load Balancing [Hard]
- Refinement
- Non-conformal meshes
- Parallel Scalability
- Non-simplex geometries

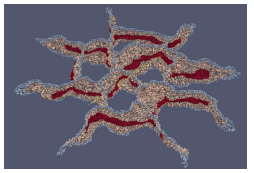
Acknowledgements



- Peter Bastian
- Markus Blatt
- Andreas Dedner
- Jorrit ’Hippi Joe’ Fahlke
- Dominic Kempf
- Robert Kloefkorn
- Martin Nolte
- Oliver Sander
- Christoph Grüniger
- Christian Engwer
- Steffen Müthing
- Olivier Martin
- Jörg Waldvogel
- Steven G. Johnson
- Xiaoye Sherry Li
- George Karypis
- Ralf Hiptmair

NON-EXHAUSTIVE LIST!!!

Features Summary



- Curvilinear Geometry
- Analytical Expressions
- Recursive Vectorized Integration
- Curvilinear Parallel GMSH Reader
- Curvilinear VTK Writer
- Self-consistent Curvilinear Grid

curvilineargrid@lspr.ch