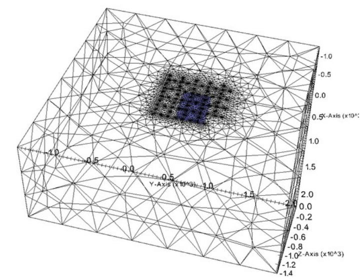
The University of Queensland
School of Earth and Environmental Sciences

esys-escript

Lutz Gross
E-mail : l.gross@uq.edu.au

esys-escript

Geometry



Mathematical Model

$$F = -A \nabla u - B u + X$$
$$\nabla^t F + C \nabla u + D u = Y$$

esys-escript: scripted Model Implementation

```
from esys.escript import *
import esys.escript.unitsSI as U
Q=10*U.W/U.m**3; K=1.7*U.W/(U.m*U.K); rhocp=1.5*U.Mega*U.J/(U.m**3*U.K)
# … time step size:
dt=1.*U.year; n_end=100
# … 40 x 20 grid on 10km x 5km
mydomain=Rectangle(40, 20,l0=10*U.km, l1=5*U.km)
x=mydomain.getX()
# … create PDE and set coefficients:
mypde=LinearPDE(domain)
mypde.setValue(A=K*kronecker(mydomain), D=rhocp/dt, q=whereZero(x[1]-
5*U.km))
# … initial temperature is a vertical, linear profile:
T=0*U.Celsius+30*U.K/U.km*(5*U.km-x[1])
n=0
while n<n_end :
   mypde.setValue(Y=Q+rhocp/dt*T, r=T)
   T=myPDE.getSolution()
    n+=1
    saveVTK("u%s"%n, temperature=T, flux=-K*grad(T) )
    print("Time ",n*dt,": min/max temperature =",inf(T), sup(T))
```

Desktop

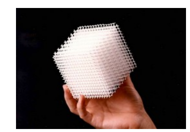
Parallel Supercomputers

esys-escript

- Module for solving partial differential equations (PDEs) in python
- Open source under Apache 2.0 license
- Started back in 2003
- Part of the National Collaborative Research Infrastructure Strategy for Earth Sciences

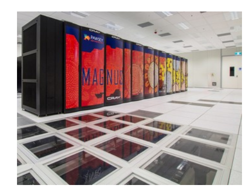
esys-escript (cont.)

- Modeling with PDEs $\rho c_p \frac{\partial T}{\partial t} - \nabla^t K \nabla T = Q$
- Rapid prototyping of new models



- Easy of Use

- “Supercomputing for the masses”

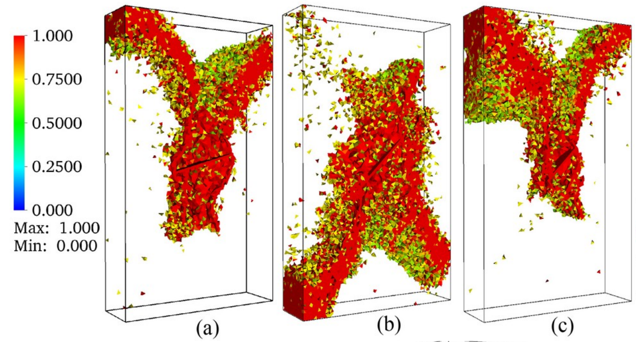


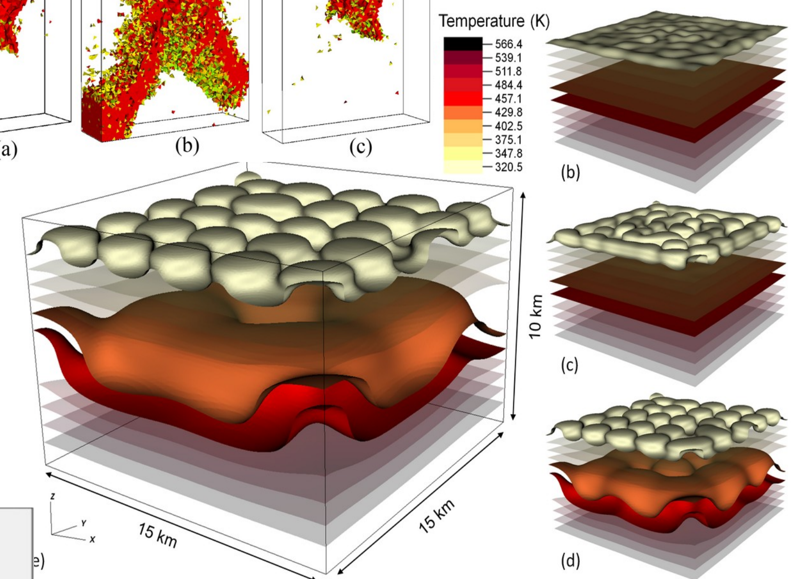
- Open Source
- Programming in python

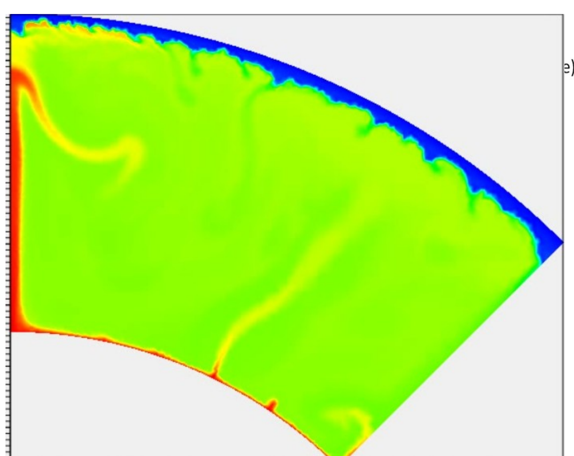
python powered

Some applications

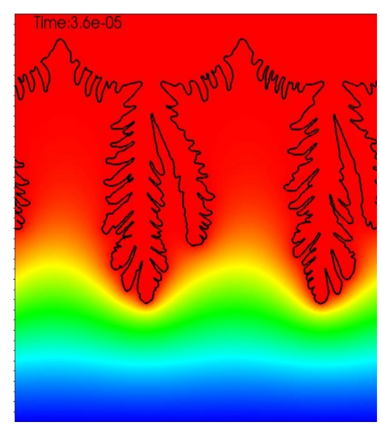
- Mantel Convection
- Pores Media Flow
- Geomechanics
- Geophysical inversion
- Earthquakes
- Volcanoes
- Tsunamis
- ...







esys-escript



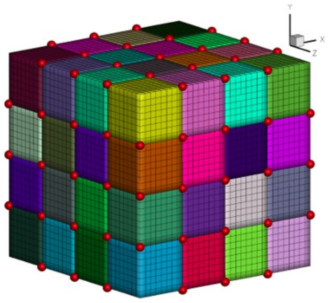
THE UNIVERSITY OF QUEENSLAND AUSTRALIA

Downloads

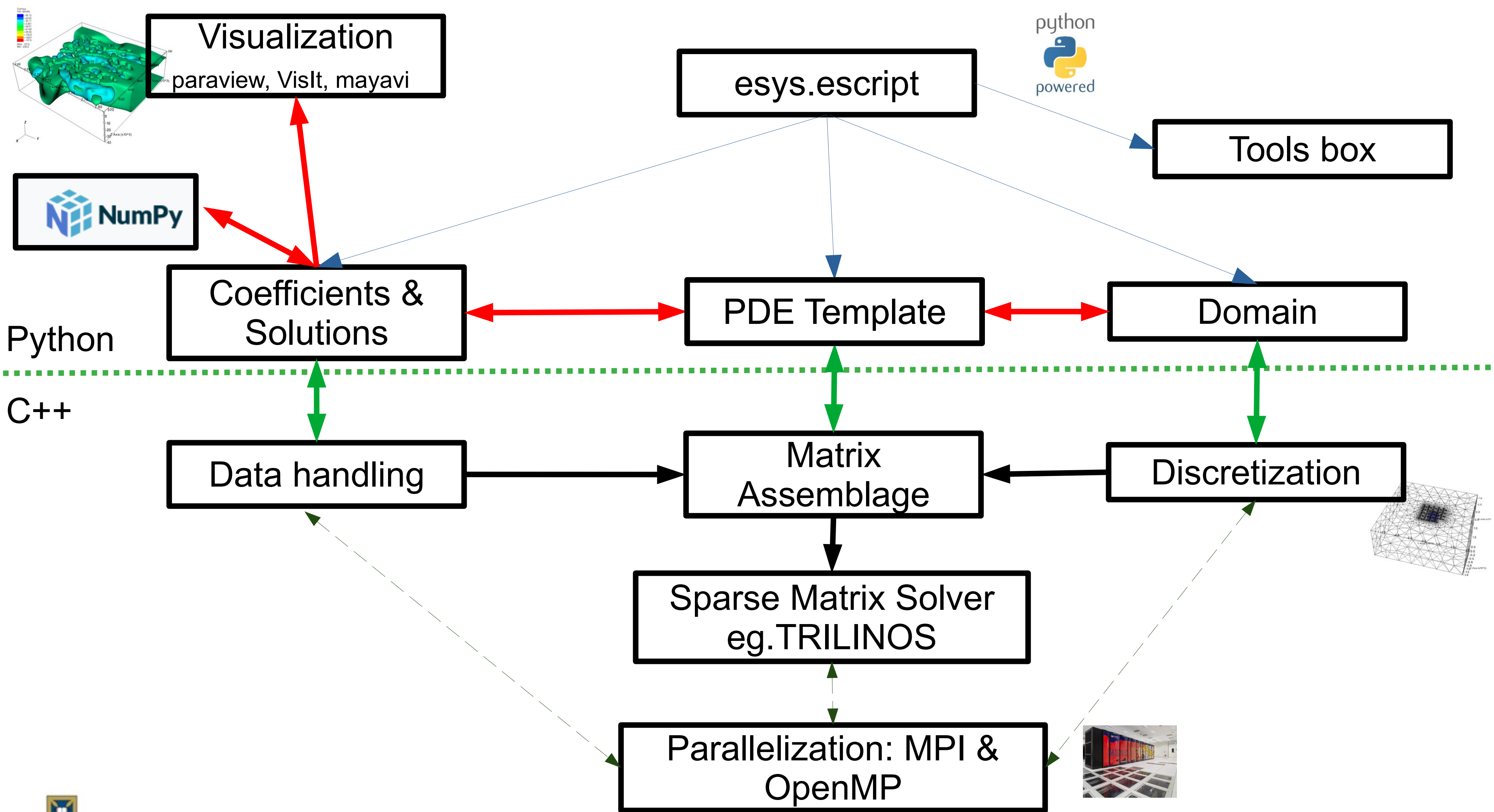
- Software: current version 5.9
  https://github.com/esys-escript/esys-escript.github.io
  - Installation from source
  - Also Q&A, bug reports
- Binary
  - Anaconda: https://anaconda.org/conda-forge/esys-escript
  - Debian Package manager
- Users' guide:
  https://github.com/esys-escript/esys-escript.github.io/blob/master/user.pdf
- API Documentation: https://esys-escript.github.io/api.html

Discretization Approach

- Finite Element Method (FEM)
  - Grids and unstructured meshes in 2D and 3D
- approximate solution PDEs in weak form
  - by continuous, piecewise linear approximation
  - Solve discrete problem with sparse matrix
    - With direct, iterative, multi-grid solver
- Parallelization on large number of cores (>25000)
  - Use Domain decomposition
  - Hidden from then user

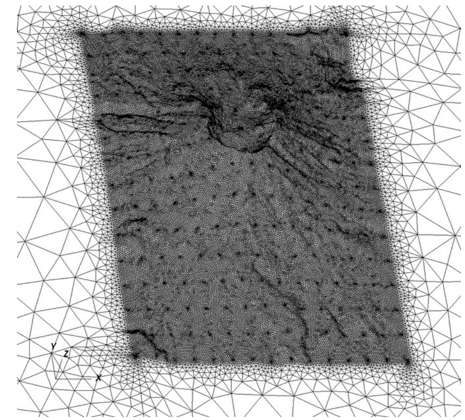


Structure

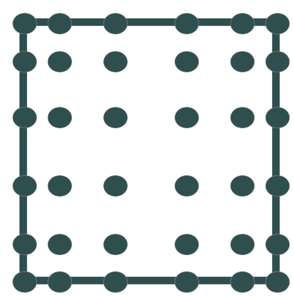


The University Of Queensland Australia

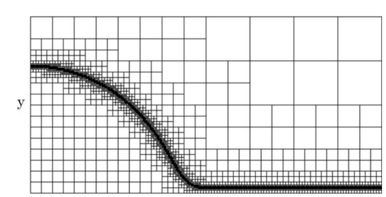
Discretization Models



- Available through different modules:
  - `esys.finley`: General FEM discretization:
    - Unstructured meshes 2D and 3D
    - hexahedron, tetrahedron, triangles, rectangles,
    - Piecewise linear or quadratic
  - `esys.ripley`: rectangular grids in 2D and 3D
  - `esys.dudley`: piecewise linear tetrahedron & triangles
  - `esys.specley`: Spectral Element Method (SEM)
    - for wave propagation
    - Higher order polynomials



  - Under construction: `esys.oxley`:
    - hierarchical grids with local refinements



PDE Template

general linear PDE for unknown solution *u* :

'generalized' flux: $\boldsymbol{F} = -\boldsymbol{A}\nabla u + \boldsymbol{X}$

conservation equation:

$$\nabla^t \boldsymbol{F} + D u = \underbrace{Y}_{\text{volume source}} + \underbrace{y_{dirac}}_{\text{point source}}$$

Python Script Structure

(a) Definition of the domain

(b) Definition of PDE coefficients

(c) Obtain Solution

(d) Post-processing

- Get solution at points → recordings
- Write to file → visualization

Non-linear, time-dependent or inversion