FiPy

A Finite Volume PDE Solver Using Python

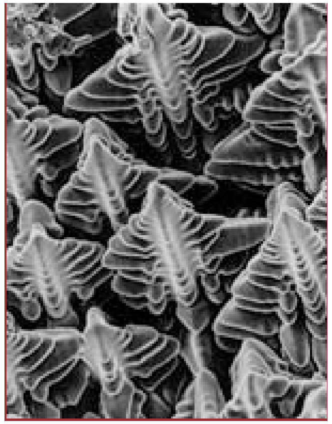
D. Wheeler, J. E. Guyer & J. A. Warren

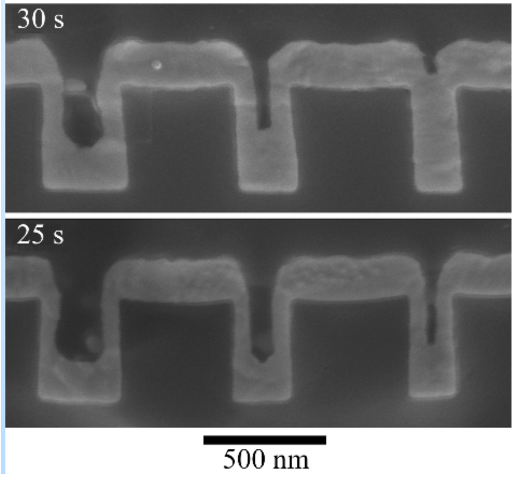
www.ctcms.nist.gov/fipy/

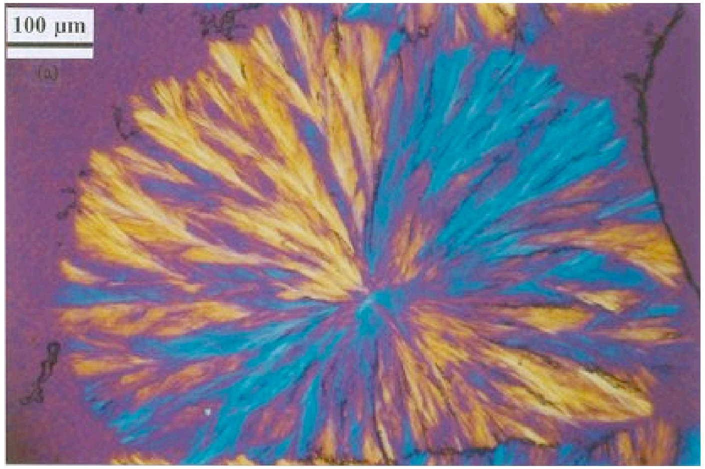
Metallurgy Division &
Center for Theoretical and Computational Materials Science
Materials Science and Engineering Laboratory

Motivation

- PDEs are ubiquitous in Materials Science problems
- Solve PDEs in weird and unique ways
- Easy to pose problems
- Easy to customize
- Don’t care about numerical methods







What is FiPy?

- FiPy is a computer program written in Python to solve partial differential equations (PDEs) using the Finite Volume method
  - Python is a powerful object oriented scripting language with tools for numerics
  - The Finite Volume method is a way to solve a set of PDEs, similar to the Finite Element or Finite Difference methods

Why a common code?

- Many interface motion codes for solving Materials Science problems at NIST.
  - Phase Field for solidification and melting
  - Phase Field for grain boundary motion
  - Phase Field for elasticity
  - Phase Field for electrochemistry
  - Level Set code for electrochemistry
  - etc…

- Need for code homogeneity
  - Institutional memory is lost with constant rewriting of codes
  - Need for preservation and reuse
  - Leverage different skill sets

Design

- Implement interface tracking
  - Phase Field, Level Set, Volume of Fluid, particle tracking
- Object-oriented structure
  - Encapsulation and Inheritance
  - Adapt, extend, reuse
- Test-based development
- Open Source
  - CVS and compressed source archives
  - Bug tracker and mailing lists
- High-level scripting language
- Python programming language

Design: test-based development

- 485 major tests, comprising thousands of low-level tests
- Tests *are* documentation (and vice versa)

298 *Module fipy.variables.variable*

__ge__(*self*, *other*)

Test if a Variable is greater than or equal to another quantity

```
>>> a = Variable(value = 3)
>>> b = (a >= 4)
>>> b
(Variable(value = 3) >= 4)
>>> b()
0
>>> a.setValue(4)
>>> b()
1
>>> a.setValue(5)
>>> b()
1
```

Design: test-based development

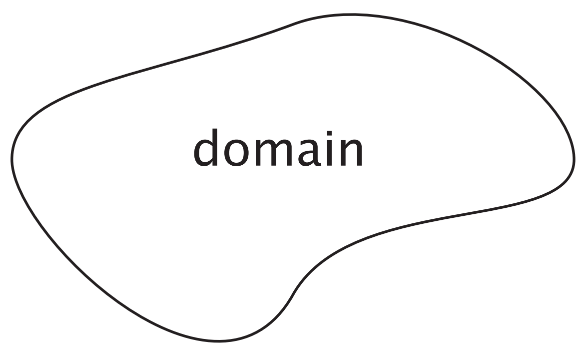
- 485 major tests, comprising thousands of low-level tests
- Tests *are* documentation (and vice versa)

```
Running __main__.Variable.__gt__.__doc__
Trying: a = Variable(value = 3)
Expecting: nothing
ok
Trying: b = (a > 4)
Expecting: nothing
ok
Trying: b
Expecting: (Variable(value = 3) > 4)
ok
Trying: b()
Expecting: 0
ok
Trying: a.setValue(5)
Expecting: nothing
ok
Trying: b()
Expecting: 1
ok
0 of 6 examples failed in __main__.Variable.__gt__.__doc__
```

Finite Volume Method

- Solve a general PDE on a given domain for a field ϕ

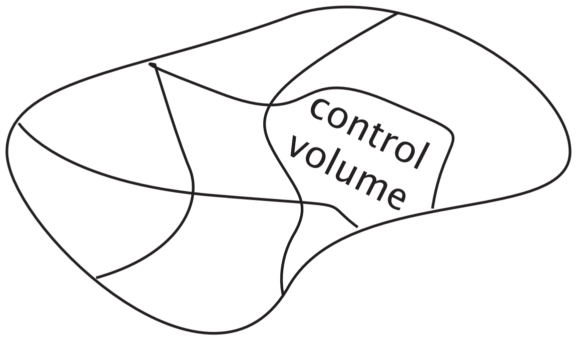
$$\underbrace{\frac{\partial(\rho\phi)}{\partial t}}_{\text{transient}} - \underbrace{\nabla\cdot(\Gamma\nabla\phi)}_{\text{diffusion}} - \underbrace{[\nabla\cdot(\Gamma_i\nabla)]^n\phi}_{n^{\text{th}}\text{ order diffusion}} - \underbrace{\nabla\cdot(\vec{u}\phi)}_{\text{convection}} - \underbrace{S_\phi}_{\text{source}} = 0$$



Finite Volume Method

- Solve a general PDE on a given domain for a field ϕ
- Integrate PDE over arbitrary control volumes

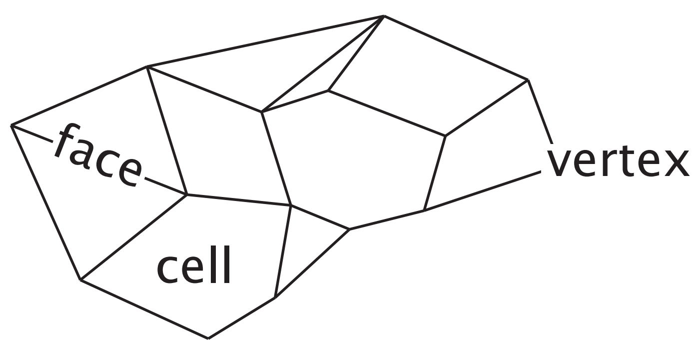
$$\underbrace{\int_V \frac{\partial(\rho\phi)}{\partial t}\,dV}_{\text{transient}} - \underbrace{\int_S \Gamma(\vec{n}\cdot\nabla\phi)\,dS}_{\text{diffusion}} - \underbrace{\int_S \Gamma_n(\vec{n}\cdot\nabla\cdots)\,dS}_{n^{\text{th}}\text{ order diffusion}} - \underbrace{\int_S (\vec{n}\cdot\vec{u})\phi\,dS}_{\text{convection}} - \underbrace{\int_V S_\phi\,dV}_{\text{source}} = 0$$



Finite Volume Method

- Solve a general PDE on a given domain for a field ϕ
- Integrate PDE over arbitrary control volumes
- Evaluate PDE over polyhedral control volumes

$$\underbrace{\frac{\rho\phi V - (\rho\phi V)^{\text{old}}}{\Delta t}}_{\text{transient}} - \underbrace{\sum_{\text{face}} [\Gamma A \vec{n} \cdot \nabla \phi]_{\text{face}}}_{\text{diffusion}} - \underbrace{\sum_{\text{face}} [\Gamma A \vec{n} \cdot \nabla \{\cdots\}]_{\text{face}}}_{n^{\text{th}}\text{ order diffusion}} - \underbrace{\sum_{\text{face}} [(\vec{n} \cdot \vec{u}) A \phi]_{\text{face}}}_{\text{convection}} - \underbrace{V S_\phi}_{\text{source}} = 0$$



Finite Volume Method

- Solve a general PDE on a given domain for a field ϕ
- Integrate PDE over arbitrary control volumes
- Evaluate PDE over polyhedral control volumes
- Obtain a large coupled set of linear equations in ϕ

$$\begin{pmatrix} a_{11} & a_{12} & & \\ a_{21} & a_{22} & \ddots & \\ & \ddots & \ddots & \ddots \\ & & \ddots & a_{nn} \end{pmatrix} \begin{pmatrix} \phi_1 \\ \phi_2 \\ \vdots \\ \phi_n \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix}$$

Diffusion Example

$$\underbrace{\frac{\partial \phi}{\partial t}}_{\text{transient}} - \underbrace{\nabla \cdot (\nabla \phi)}_{\text{diffusion}} = 0$$

- # create a mesh
- # create a field variable
- # create the equation terms
- # create the equation
- # create a viewer
- # solve

Diffusion Example

$$\underbrace{\frac{\partial \phi}{\partial t}}_{\text{transient}} - \underbrace{\nabla \cdot (\nabla \phi)}_{\text{diffusion}} = 0$$

- # create a mesh

```
L = nx * dx
from fipy.meshes.grid2D import Grid2D
mesh = Grid2D(nx = nx, dx = dx)
```

- # create a field variable
- # create the equation
- # create a viewer
- # solve

Diffusion Example

$$\underbrace{\frac{\partial \phi}{\partial t}}_{\text{transient}} - \underbrace{\nabla \cdot (\nabla \phi)}_{\text{diffusion}} = 0$$

- # create a mesh
- # create a field variable

```
from fipy.variables.cellVariable import CellVariable
var = CellVariable(mesh = mesh, value = 0)

def centerCells(cell):
    return abs(cell.getCenter()[0] - L/2.) < L/10.

var.setValue(value = 1., cells = mesh.getCells(filter = centerCells))
```

- # create the equation
- # create a viewer
- # solve

Diffusion Example

$$\underbrace{\frac{\partial \phi}{\partial t}}_{\text{transient}} - \underbrace{\nabla \cdot (\nabla \phi)}_{\text{diffusion}} = 0$$

- # create a mesh
- # create a field variable
- # set the initial conditions

```
def centerCells(cell):
    return abs(cell.getCenter()[0] - L/2.) < L/10.

var.setValue(value = 1., cells = mesh.getCells(filter = centerCells))
```

- # create the equation
- # create a viewer
- # solve

Diffusion Example

$$\underbrace{\frac{\partial \phi}{\partial t}}_{\text{transient}} - \underbrace{\nabla \cdot (\nabla \phi)}_{\text{diffusion}} = 0$$

- # create a mesh
- # create a field variable
- # create the equation

```
from fipy.terms.transientTerm import TransientTerm
from fipy.terms.implicitDiffusionTerm import ImplicitDiffusionTerm

## equivalent forms

## eq = (TransientTerm() == ImplicitDiffusionTerm(coeff = 1))

## eq = TransientTerm() - ImplicitDiffusionTerm(coeff = 1)

eq = (TransientTerm() - ImplicitDiffusionTerm(coeff = 1) == 0)
```

- # create a viewer
- # solve

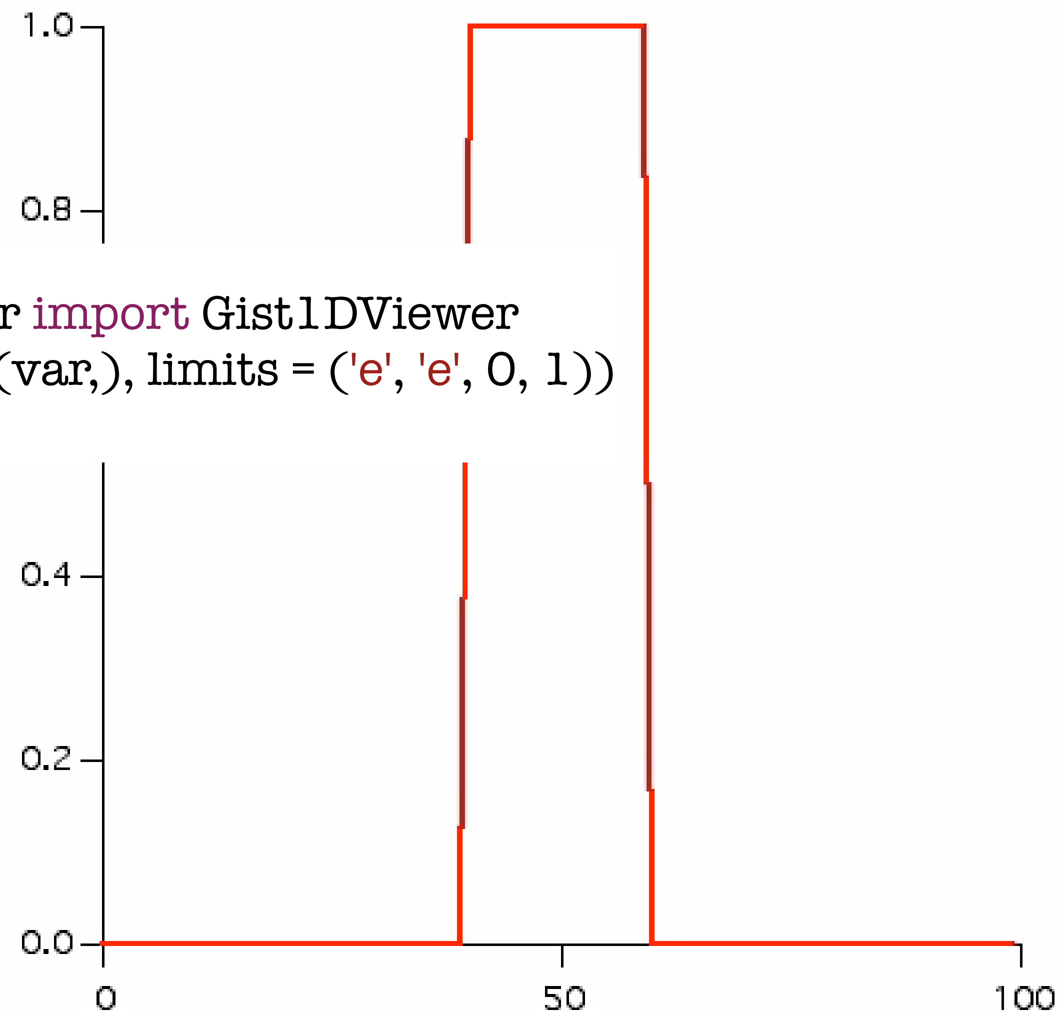
Diffusion Example

$$\underbrace{\frac{\partial \phi}{\partial t}}_{\text{transient}} - \underbrace{\nabla \cdot (\nabla \phi)}_{\text{diffusion}} = 0$$

- # create a mesh
- # create a field variable
- # create the equation terms
- # create the equation
- # create a viewer

```
from fipy.viewers.gist1DViewer import Gist1DViewer
viewer = Gist1DViewer(vars = (var,), limits = ('e', 'e', 0, 1))
viewer.plot()
```

- # solve

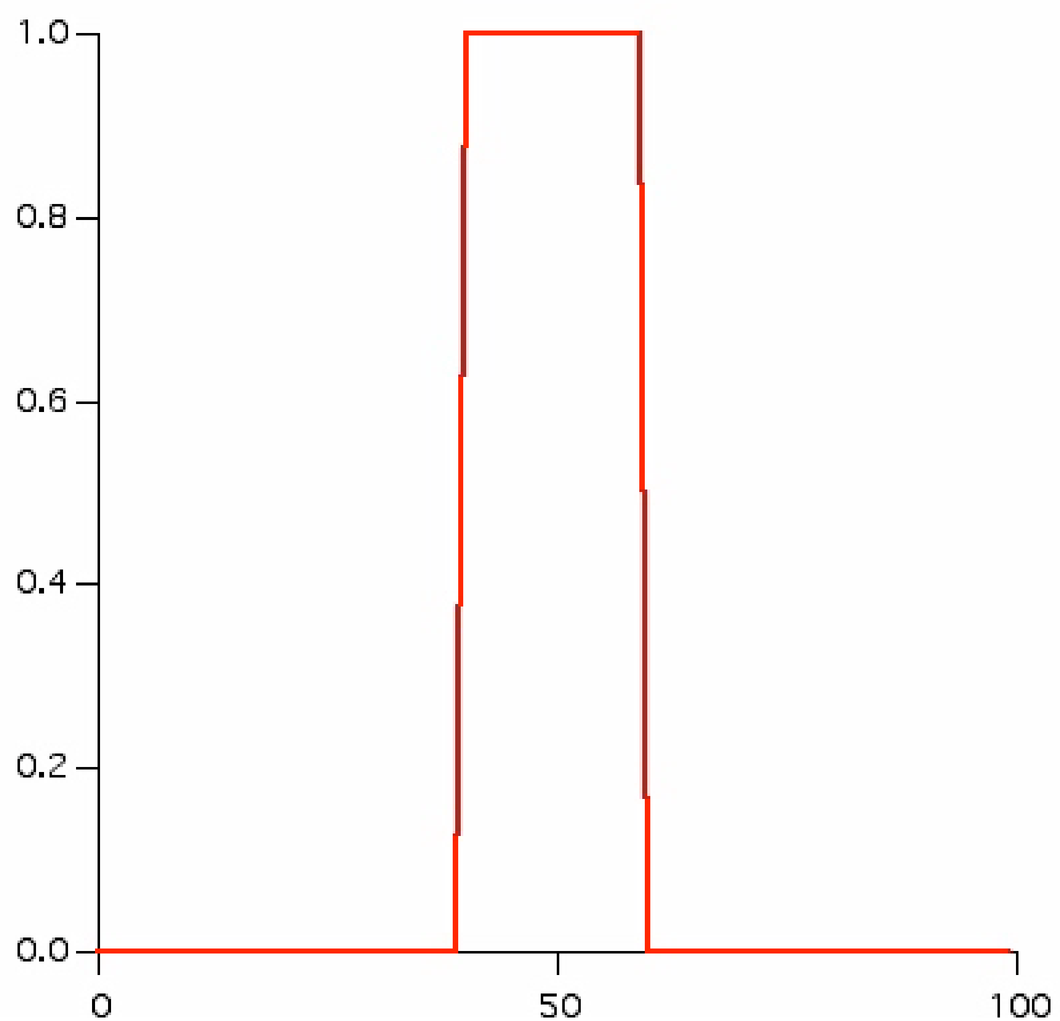


Diffusion Example

$$\underbrace{\frac{\partial \phi}{\partial t}}_{\text{transient}} - \underbrace{\nabla \cdot (\nabla \phi)}_{\text{diffusion}} = 0$$

- # create a mesh
- # create a field variable
- # create the equation terms
- # create the equation
- # create a viewer
- # solve

```
for i in range(steps):
    var.updateOld()
    eq.solve()
    viewer.plot()
```



Convection Example

$$\underbrace{\nabla \cdot (\vec{u}\phi)}_{\text{convection}} + \underbrace{\nabla \cdot (\nabla\phi)}_{\text{diffusion}} = 0$$

$$\phi|_{x=0} = 0 \qquad \phi|_{x=L} = 1$$

- # create a mesh
- # create a field variable
- # create the boundary conditions
- # create the equation
- # create a viewer
- # solve

Convection Example

$$\underbrace{\nabla \cdot (\vec{u}\phi)}_{\text{convection}} + \underbrace{\nabla \cdot (\nabla \phi)}_{\text{diffusion}} = 0$$

$$\phi|_{x=0} = 0 \qquad \phi|_{x=L} = 1$$

- # create a mesh
- # create a field variable
- # create the boundary conditions

```
from fipy.boundaryConditions.fixedValue import FixedValue
bcs = (
    FixedValue(mesh.getFacesLeft(), 0),
    FixedValue(mesh.getFacesRight(), 1),
)
```

- # create the equation
- # create a viewer
- # solve

Convection Example

$$\underbrace{\nabla \cdot (\vec{u}\phi)}_{\text{convection}} + \underbrace{\nabla \cdot (\nabla\phi)}_{\text{diffusion}} = 0$$

$$\phi|_{x=0} = 0 \qquad \phi|_{x=L} = 1$$

- # create a mesh
- # create a field variable
- # create the boundary conditions
- # create the equation

```python
from fipy.terms.implicitDiffusionTerm import ImplicitDiffusionTerm
diffusionTerm = ImplicitDiffusionTerm(coeff = 1)

from fipy.terms.exponentialConvectionTerm import ExponentialConvectionTerm
convectionTerm = ExponentialConvectionTerm(coeff = (10,0),
                                           diffusionTerm = diffusionTerm)

eq = (diffusionTerm + convectionTerm == 0)
```

- # create a viewer
- # solve

Convection Example

$$\underbrace{\nabla \cdot (\vec{u}\phi)}_{\text{convection}} + \underbrace{\nabla \cdot (\nabla \phi)}_{\text{diffusion}} = 0$$

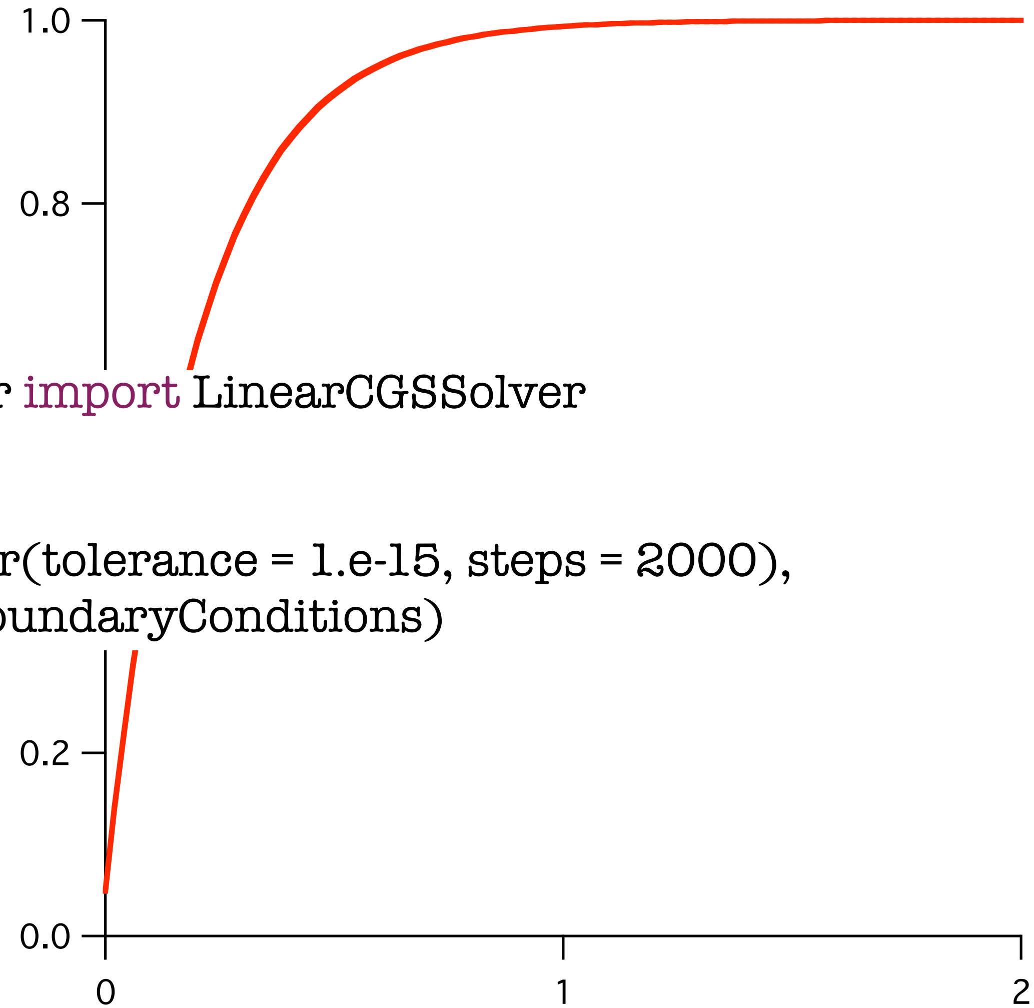
$$\phi|_{x=0} = 0 \qquad \phi|_{x=L} = 1$$

- # create a mesh
- # create a field variable
- # create the boundary conditions
- # create the equation
- # create a viewer
- # solve

```
from fipy.solvers.linearCGSSolver import LinearCGSSolver

eq.solve(var = var,
         solver = LinearCGSSolver(tolerance = 1.e-15, steps = 2000),
         boundaryConditions = boundaryConditions)

viewer.plot()
```



Phase Field Dendrite Example

after J. A. Warren, R. Kobayashi, A. E. Lobkovsky, and W. C. Carter, *Acta Materialia* **51** (20), (2003) 6035–6058

$$\tau_\phi \frac{\partial \phi}{\partial t} - \alpha^2 \nabla^2 \phi - \phi(1-\phi) m_2(\phi, T) = 0$$

$$\underbrace{\frac{\partial T}{\partial t}}_{\text{transient}} - \underbrace{D_T \nabla^2 T}_{\text{diffusion}} - \underbrace{\frac{\partial \phi}{\partial t} = 0}_{\text{source}}$$

$$m_2(\phi, T) = \phi - \frac{1}{2} - \frac{\kappa_1}{\pi} \arctan(\kappa_2 T)$$

- # create a mesh
- # create the field variables
- # create the phase equation
- # create the temperature equation
- # create a viewer
- # solve

Phase Field Dendrite Example

after J.A. Warren, R. Kobayashi, A. E. Lobkovsky, and W. C. Carter, *Acta Materialia* **51** (20), (2003) 6035–6058

$$\tau_\phi \frac{\partial \phi}{\partial t} - \alpha^2 \nabla^2 \phi - \phi(1-\phi) m_2(\phi, T) = 0$$

$$\underbrace{\frac{\partial T}{\partial t}}_{\text{transient}} - \underbrace{D_T \nabla^2 T}_{\text{diffusion}} - \underbrace{\frac{\partial \phi}{\partial t} = 0}_{\text{source}}$$

$$m_2(\phi, T) = \phi - \frac{1}{2} - \frac{\kappa_1}{\pi} \arctan(\kappa_2 T)$$

- # create a mesh
- # create the field variables
- # create the phase equation

```
m2 = phase - 0.5 - kappa1 * arctan(kappa2 * temperatue)
phaseEq = (TransientTerm(coeff = tau) == \
           ImplicitDiffusionTerm(coeff = alpha**2) + \
           ImplicitSourceTerm(coeff = m2 * ((m2 < 0) - phase)) + \
           (m2 > 0) * m2 * phase)
```

- # create the temperature equation
- # create an iterator
- # create a viewer
- # solve

National Institute of Standards and Technology
Technology Administration, U.S. Department of Commerce

Phase Field Dendrite Example

after J. A. Warren, R. Kobayashi, A. E. Lobkovsky, and W. C. Carter, *Acta Materialia* **51** (20), (2003) 6035–6058

$$\tau_\phi \frac{\partial \phi}{\partial t} - \alpha^2 \nabla^2 \phi - \phi(1-\phi) m_2(\phi, T) = 0$$

$$\underbrace{\frac{\partial T}{\partial t}}_{\text{transient}} - \underbrace{D_T \nabla^2 T}_{\text{diffusion}} \underbrace{- \frac{\partial \phi}{\partial t} = 0}_{\text{source}}$$

$$m_2(\phi, T) = \phi - \frac{1}{2} - \frac{\kappa_1}{\pi} \arctan(\kappa_2 T)$$

- # create a mesh
- # create the field variables
- # create the phase equation
- # create the temperature equation

```
temperatureEq = (TransientTerm() == \
                 ImplicitDiffusionTerm(coeff = tempDiffCoeff) + \
                 (phase - phase.getOld()) / timeStepDuration)
```

- # create a viewer
- # solve

National Institute of Standards and Technology
Technology Administration, U.S. Department of Commerce

Phase Field Dendrite Example

after J. A. Warren, R. Kobayashi, A. E. Lobkovsky, and W. C. Carter, *Acta Materialia* **51** (20), (2003) 6035–6058

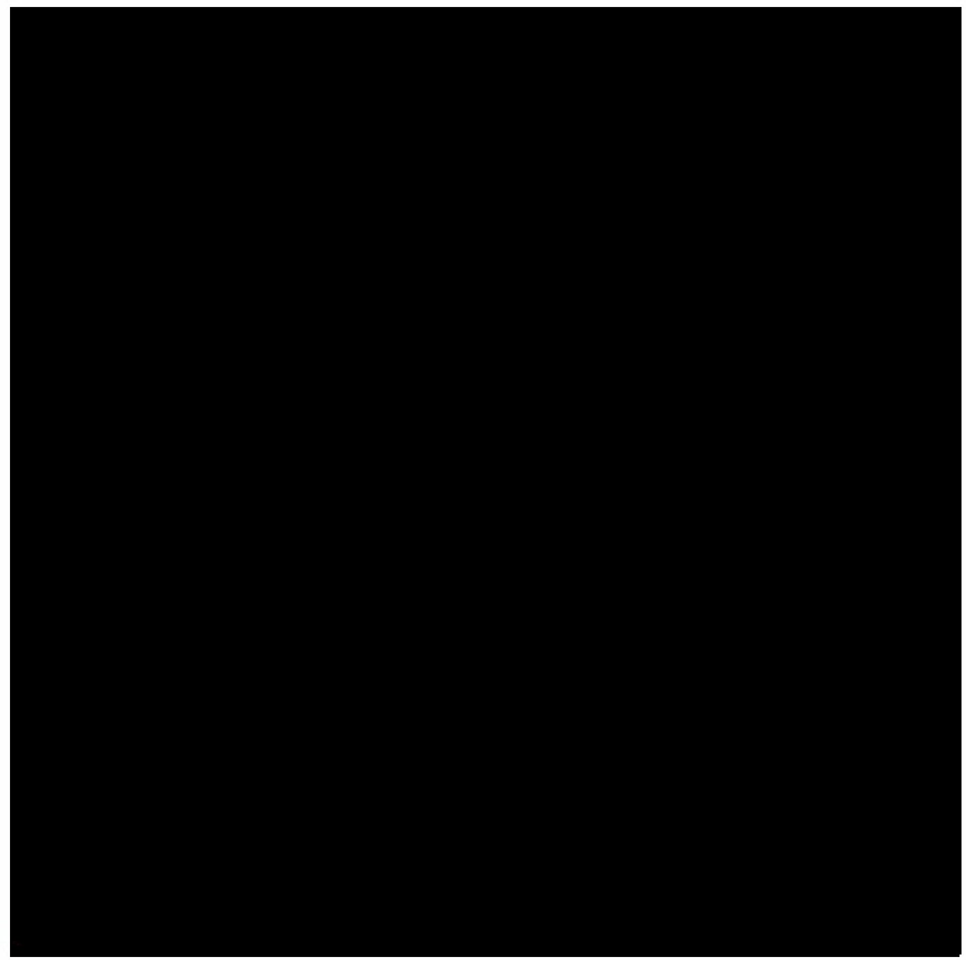
$$\tau_\phi \frac{\partial \phi}{\partial t} - \alpha^2 \nabla^2 \phi - \phi(1-\phi) m_2(\phi, T) = 0$$

$$\underbrace{\frac{\partial T}{\partial t}}_{\text{transient}} - \underbrace{D_T \nabla^2 T}_{\text{diffusion}} - \underbrace{\frac{\partial \phi}{\partial t} = 0}_{\text{source}}$$

$$m_2(\phi, T) = \phi - \frac{1}{2} - \frac{\kappa_1}{\pi} \arctan(\kappa_2 T)$$

- # create a mesh
- # create the field variables
- # create the phase equation
- # create the temperature equation
- # create an iterator
- # create a viewer
- # solve

```
for i for range(steps):
    phase.updateOld()
    temperature.updateOld()
    phaseEq.solve(phase, dt = time
    temperatureEq.solve(temperat
    if i%frameRate == 0:
        phaseViewer.plot()
        temperatureViewer.plot()
```



NIST
National Institute of Standards and Technology
Technology Administration, U.S. Department of Commerce

Cahn-Hilliard Example

- # create a mesh
- # create the field variable
- # create the equation
- # create the boundary conditions
- # create a viewer
- # solve

$$\frac{\partial \phi}{\partial t} - \nabla \cdot D \nabla \left(\frac{\partial f}{\partial \phi} - \epsilon^2 \nabla^2 \phi \right) = 0$$

$$f = \frac{a^2}{2} \phi^2 (1 - \phi)^2$$

$$\vec{n} \cdot \nabla \phi = 0 \quad \text{on all boundaries}$$

$$\vec{n} \cdot \nabla^3 \phi = 0 \quad \text{on all boundaries}$$

Cahn-Hilliard Example

$$\underbrace{\frac{\partial \phi}{\partial t}}_{\text{transient}} - \underbrace{\nabla \cdot D a^2 \left[1 - 6\phi\left(1 - \phi\right)\right] \nabla \phi}_{2^{\text{nd}} \text{ order diffusion}} + \underbrace{\nabla \cdot D \nabla \epsilon^2 \nabla^2 \phi}_{4^{\text{th}} \text{ order diffusion}} = 0$$

$$\vec{n} \cdot \nabla \phi = 0 \quad \text{on all boundaries}$$

$$\vec{n} \cdot \nabla^3 \phi = 0 \quad \text{on all boundaries}$$

- # create a mesh
- # create the field variable
- # create the equation
- # create the boundary conditions
- # create a viewer
- # solve

Cahn-Hilliard Example

$$\underbrace{\frac{\partial \phi}{\partial t}}_{\text{transient}} - \underbrace{\nabla \cdot D a^2 \left[1 - 6\phi\left(1-\phi\right)\right] \nabla \phi}_{2^{\text{nd}}\ \text{order diffusion}} + \underbrace{\nabla \cdot D \nabla \epsilon^2 \nabla^2 \phi}_{4^{\text{th}}\ \text{order diffusion}} = 0$$

$$\vec{n} \cdot \nabla \phi = 0 \quad \text{on all boundaries}$$

$$\vec{n} \cdot \nabla^3 \phi = 0 \quad \text{on all boundaries}$$

- # create a mesh
- # create the field variable
- # create the equation

```
faceVar = var.getArithmeticFaceValue()
doubleWellDerivative = a**2 * ( 1 - 6 * faceVar * (1 - faceVar))

from fipy.terms.nthOrderDiffusionTerm import NthOrderDiffusionTerm
from fipy.terms.transientTerm import TransientTerm
eq = (TransientTerm() == \
   NthOrderDiffusionTerm(coeffs = (diffusionCoeff * doubleWellDerivative,)) -\
   NthOrderDiffusionTerm(coeffs = (diffusionCoeff, epsilon**2)))
```

- # create the boundary conditions
- # create a viewer
- # solve

Cahn-Hilliard Example

$$\underbrace{\frac{\partial \phi}{\partial t}}_{\text{transient}} - \underbrace{\nabla \cdot D a^2 \left[1 - 6\phi\left(1-\phi\right)\right] \nabla \phi}_{2^{\text{nd}} \text{ order diffusion}} + \underbrace{\nabla \cdot D \nabla \epsilon^2 \nabla^2 \phi}_{4^{\text{th}} \text{ order diffusion}} = 0$$

- # create a mesh
- # create the field variable
- # create the equation
- # create the boundary conditions

$$\vec{n} \cdot \nabla \phi = 0 \quad \text{on all boundaries}$$

$$\vec{n} \cdot \nabla^3 \phi = 0 \quad \text{on all boundaries}$$

```
from fipy.boundaryConditions.nthOrderBoundaryCondition \
   import NthOrderBoundaryCondition

BCs = (NthOrderBoundaryCondition(mesh.getExteriorFaces(), 0, 3),)
```

- # create a viewer
- # solve

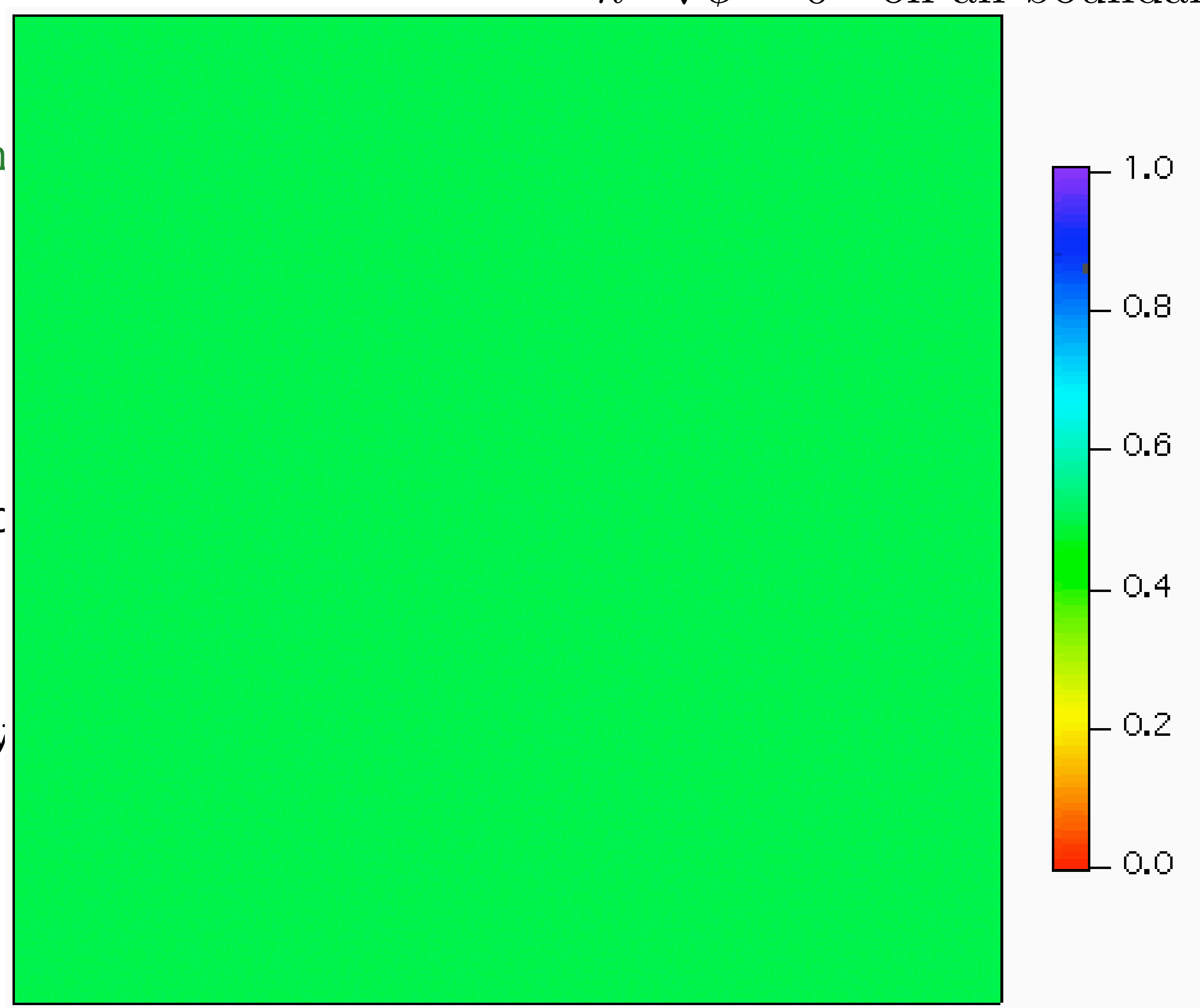
Cahn-Hilliard Example

$$\underbrace{\frac{\partial \phi}{\partial t}}_{\text{transient}} - \underbrace{\nabla \cdot D a^2 \left[1 - 6\phi\left(1-\phi\right)\right] \nabla \phi}_{2^{\text{nd}} \text{ order diffusion}} + \underbrace{\nabla \cdot D \nabla \epsilon^2 \nabla^2 \phi}_{4^{\text{th}} \text{ order diffusion}} = 0$$

$$\vec{n} \cdot \nabla \phi = 0 \quad \text{on all boundaries}$$

- # create a mesh
- # create the field variable
- # create the equation
- # create the boundary con
- # create a viewer
- # solve

```
dexp = 0.01
for step in range(steps):
    dt = Numeric.exp(dexp
    dt = min(100, dt)
    dexp += 0.01
    var.updateOld()
    eq.solve(var, boundary
    viewer.plot()
```



ies

CEAC Example

- Copper electodeposition in submicron features
- electrolyte additives influence deposition rate
- CEAC - Curvature Enhanced Accelerator Coverage.

$$\frac{d\theta_a}{dt} = \kappa|\vec{v}|\theta_a$$

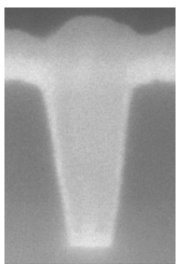
Accelerator coverage

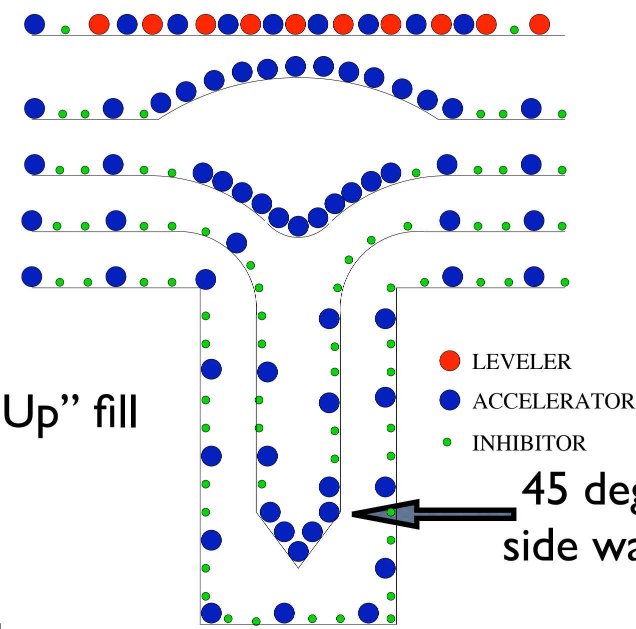
Curvature

Interface speed

fixed point moving with the interface

"Momentum plating"

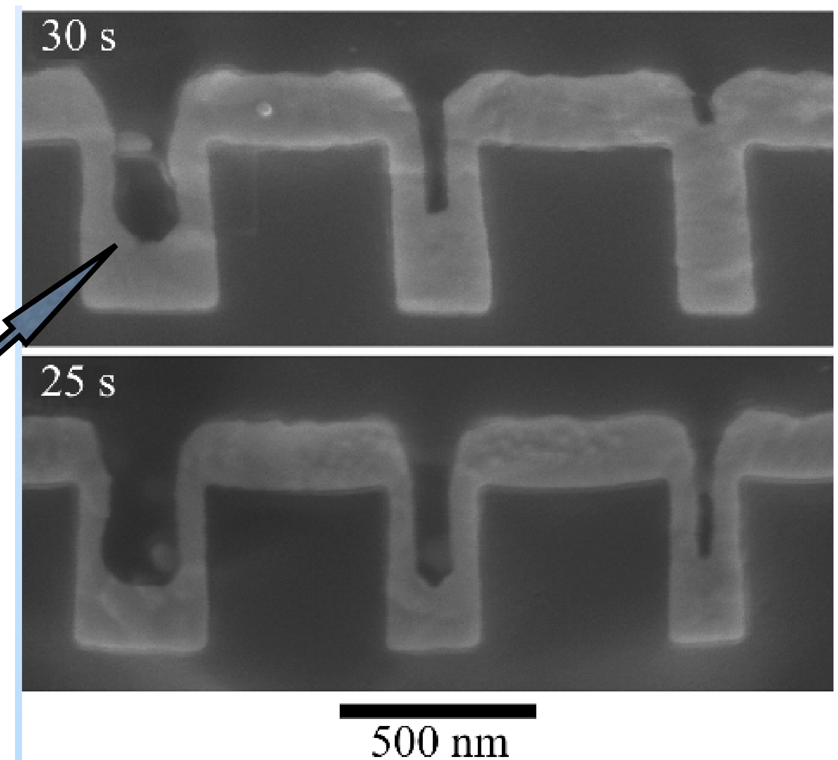




"Bottom Up" fill

45 degree side wall tilt

Experimental sequence



National Institute of Standards and Technology
Technology Administration, U.S. Department of Commerce

CEAC Example

after D. Josell, D. Wheeler, W. H. Huber, and T. P. Moffat, *Physical Review Letters*, **87**(1), (2001) 016102

- # create a mesh
- # create the field variables
- # create the governing equations
- # create the boundary conditions
- # solve

$$v = \frac{\Omega}{nF}(b_0 + b_1\theta)\frac{c_m^i}{c_m^\infty}\exp\left(\frac{-\alpha F}{RT}\eta\right)$$

$$\frac{\partial \phi}{\partial t} + v_\text{ext}|\nabla\phi| = 0$$

$$\frac{\partial \theta}{\partial t} - Jv\theta - k_\theta c_\theta^i(1-\theta) = 0$$

$$\frac{\partial c_m}{\partial t} - \nabla\cdot D_m \nabla c_m = 0$$

$$D_m \hat{n}\cdot\nabla c_m = \frac{v}{\Omega} \quad \text{on } \phi = 0$$

$$\frac{\partial c_\theta}{\partial t} - \nabla\cdot D_\theta \nabla c_\theta = 0$$

$$D_\theta \hat{n}\cdot\nabla c_\theta = -k_\theta c_\theta \Gamma(1-\theta) \quad \text{on } \phi = 0$$

$$|\nabla\phi| = 1$$

CEAC Example

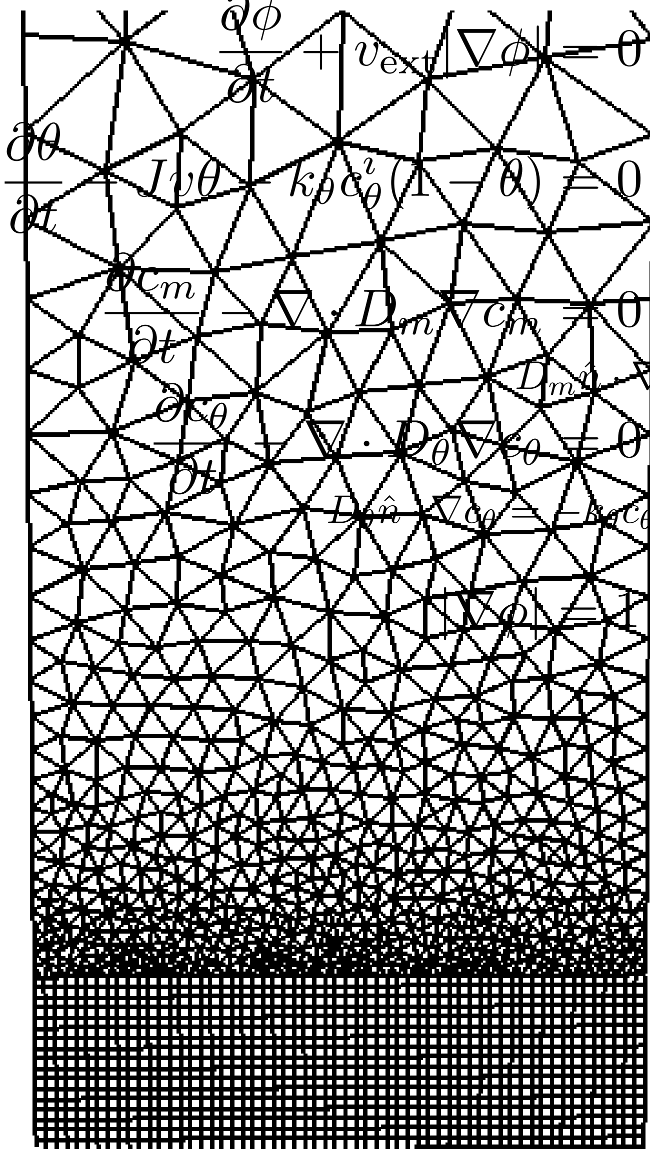
after D. Josell, D. Wheeler, W. H. Huber, and T. P. Moffat,
Physical Review Letters, **87**(1), (2001) 016102

$$v = \frac{\Omega}{nF}(b_0 + b_1\theta)\frac{c_m^i}{c_m^\infty}\exp\left(\frac{-\alpha F}{RT}\eta\right)$$

- # create a mesh

```
from gapFillMesh import TrenchMesh
mesh = TrenchMesh(cellSize = 0.002e-6,
                  trenchSpacing = 1e-6,
                  trenchDepth = 0.5e-6,
                  boundaryLayerDepth = 50e-6,
                  aspectRatio = 2.)
```

- # create the field variables
- # create the governing equations
- # create the boundary conditions
- # solve



CEAC Example

after D. Josell, D. Wheeler, W. H. Huber, and T. P. Moffat, *Physical Review Letters*, **87**(1), (2001) 016102

$$v = \frac{\Omega}{nF}(b_0 + b_1\theta)\frac{c_m^i}{c_m^\infty}\exp\left(\frac{-\alpha F}{RT}\eta\right)$$

$$\frac{\partial \phi}{\partial t} + v_{\text{ext}}|\nabla\phi| = 0$$

$$\frac{\partial \theta}{\partial t} - Jv\theta - k_\theta c_\theta^i(1-\theta) = 0$$

$$\frac{\partial c_m}{\partial t} - \nabla\cdot D_m \nabla c_m = 0$$

$$D_m \hat{n}\cdot\nabla c_m = \frac{v}{\Omega} \quad \text{on } \phi = 0$$

$$\frac{\partial c_\theta}{\partial t} - \nabla\cdot D_\theta \nabla c_\theta = 0$$

$$D_\theta \hat{n}\cdot\nabla c_\theta = -k_\theta c_\theta \Gamma(1-\theta) \quad \text{on } \phi = 0$$

$$|\nabla\phi| = 1$$

- # create a mesh
- # create the field variables
  - # distance variable
  - # surface accelerator concentration
  - # bulk accelerator concentration
  - # metal ion concentration
  - # interfacial velocity

```
exchangeCurrentDensity = constantCurrentDensity \
    + acceleratorDependenceCurrentDensity * acceleratorVar.getInterfaceVar()

currentDensity = exchangeCurrentDensity * metalVar / bulkMetalConcentration
    * Numeric.exp(-transferCoefficient * faradaysConstant * overpotential \
                  / gasConstant / temperature)

depositionRateVariable = currentDensity * atomicVolume / charge / faradaysC
```

CEAC Example

after D. Josell, D. Wheeler, W. H. Huber, and T. P. Moffat, *Physical Review Letters*, **87**(1), (2001) 016102

$$v = \frac{\Omega}{nF}(b_0 + b_1\theta)\frac{c_m^i}{c_m^\infty}\exp\left(\frac{-\alpha F}{RT}\eta\right)$$

$$\frac{\partial \phi}{\partial t} + v_{\text{ext}}|\nabla \phi| = 0$$

$$\frac{\partial \theta}{\partial t} - Jv\theta - k_\theta c_\theta^i (1-\theta) = 0$$

$$\frac{\partial c_m}{\partial t} - \nabla \cdot D_m \nabla c_m = 0$$

$$D \; \hat{n} \cdot \nabla c \quad = -\frac{v}{\Omega} \quad \text{on } \phi = 0$$

$$\theta) \quad \text{on } \phi = 0$$

- # create a mesh
- # create the field variables
- # create the governing equations
  - # advection equation

```
from fipy.terms.transientTerm import TransientTerm
from fipy.models.levelSet.advection.higherOrderAdvectionTerm \
        import HigherOrderAdvectionTerm

advectionEquation = TransientTerm() +
        HigherOrderAdvectionTerm(coeff = extensionVelocityVariable)
```

  - # surfactant equation
  - # metal equation
  - # bulk accelerator equation
- # create the boundary conditions

CEAC Example

after D. Josell, D. Wheeler, W. H. Huber, and T. P. Moffat, *Physical Review Letters*, **87**(1), (2001) 016102

$$v = \frac{\Omega}{nF}(b_0 + b_1\theta)\frac{c_m^i}{c_m^\infty}\exp\left(\frac{-\alpha F}{RT}\eta\right)$$

$$\frac{\partial \phi}{\partial t} + v_{\text{ext}}|\nabla\phi| = 0$$

$$\frac{\partial \theta}{\partial t} - Jv\theta - k_\theta c_\theta^i(1-\theta) = 0$$

$$\frac{\partial c_m}{\partial t} - \nabla\cdot D_m\nabla c_m = 0$$

$$D_m\hat{n}\cdot\nabla c_m = \frac{v}{\Omega} \quad \text{on } \phi = 0$$

$$\frac{\partial c_\theta}{\partial t} - \nabla\cdot D_\theta\nabla c_\theta = 0$$

$$D_\theta\hat{n}\cdot\nabla c_\theta = -k_\theta c_\theta\Gamma(1-\theta) \quad \text{on } \phi = 0$$

$$|\nabla\phi| = 1$$

- # create a mesh
- # create the field variables
- # create the governing equations
- # create the boundary conditions
- # solve

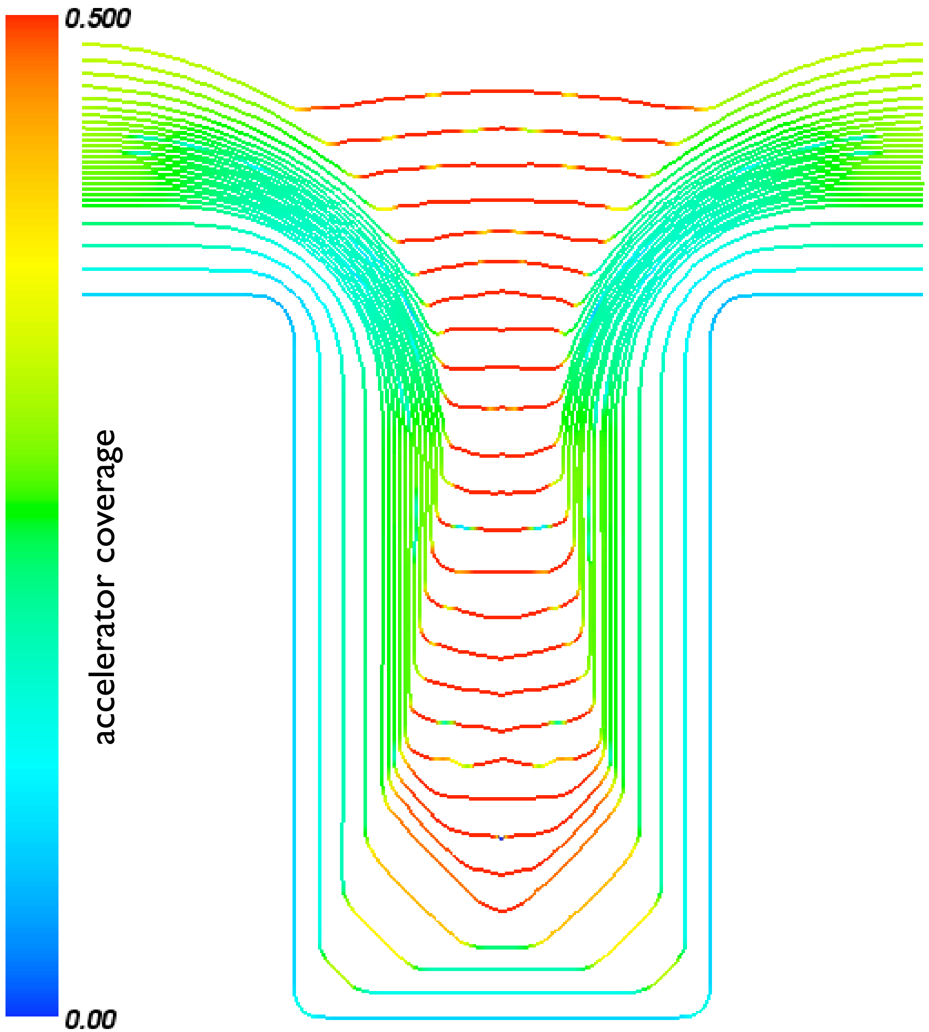
```
for step in range(numberOfSteps):
    if step % levelSetUpdateFrequency == 0:
        distanceVar.calcDistanceFunction()
    for var in (distanceVar, acceleratorVar, metalVar, bulAcceleratorVar):
        var.updateOld()
    dt = cflNumber * cellSize / max(extensionVelocityVariable)
    distanceVar.extendVariable(extensionVelocityVariable)

    advectionEquation.solve(distanceVar, dt = dt)
    surfactantEquation.solve(acceleratorVar, dt = dt)
    metalEquation.solve(metalVar, dt = dt, boundaryConditions = metalEquationBCs)
    bulkAcceleratorEquation.solve(bulkAcceleratorVar, dt = dt,
                                  boundaryConditions = acceleratorBCs)
```

CEAC Example

after D. Josell, D. Wheeler, W. H. Huber, and T. P. Moffat,
Physical Review Letters, **87**(1), (2001) 016102

no “leveler”

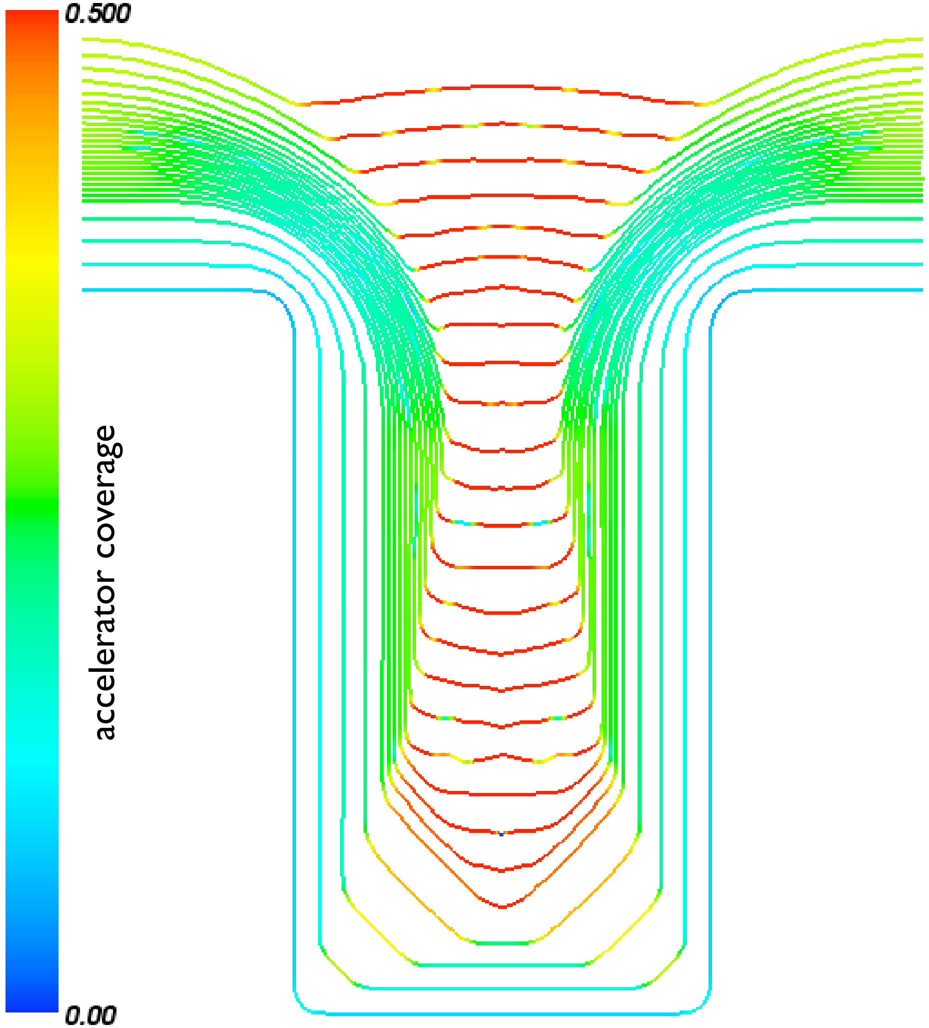


National Institute of Standards and Technology
Technology Administration, U.S. Department of Commerce

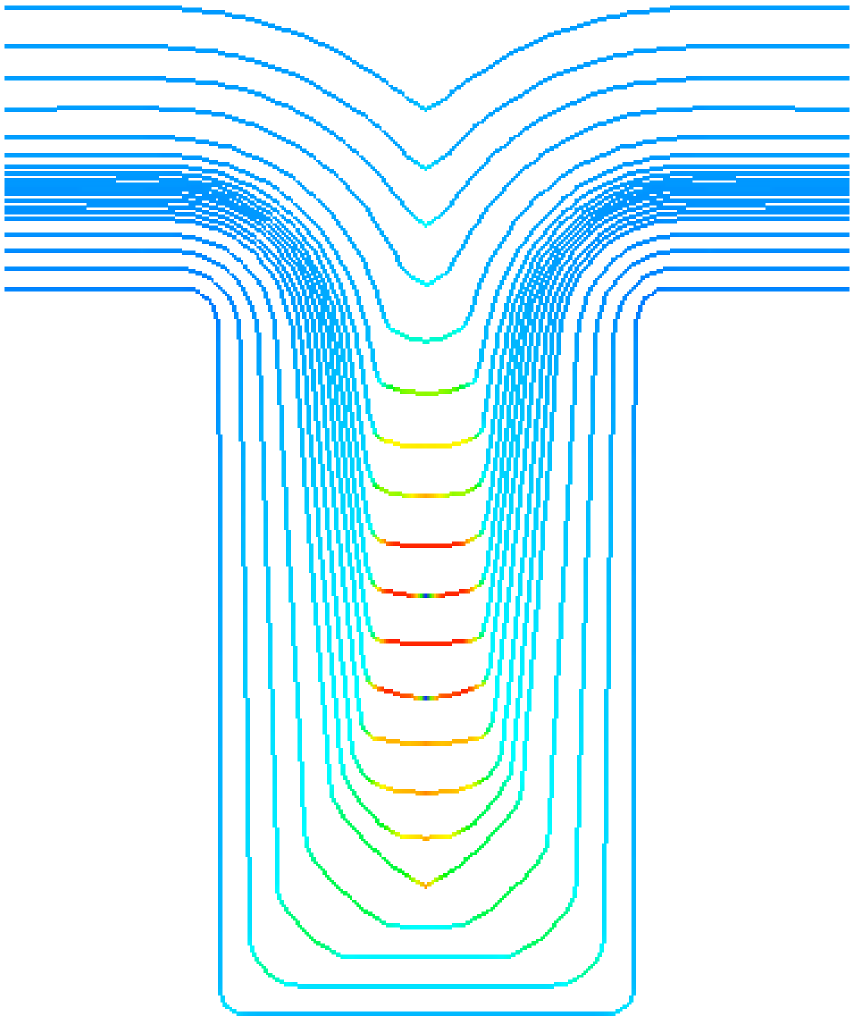
CEAC Example

after D. Josell, D. Wheeler, W. H. Huber, and T. P. Moffat, *Physical Review Letters*, **87**(1), (2001) 016102

no “leveler”

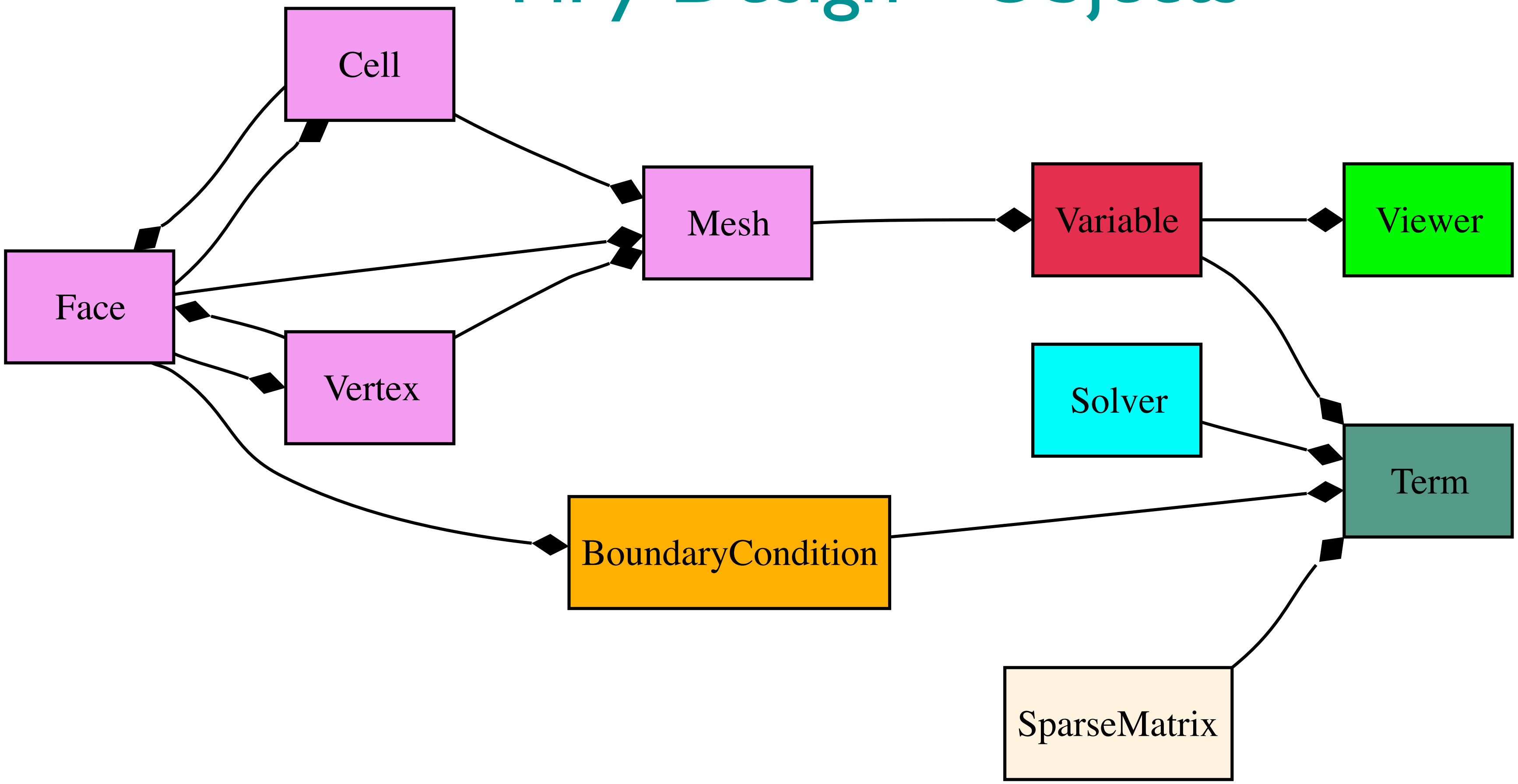


with “leveler”
(just add another surfactant equation set)

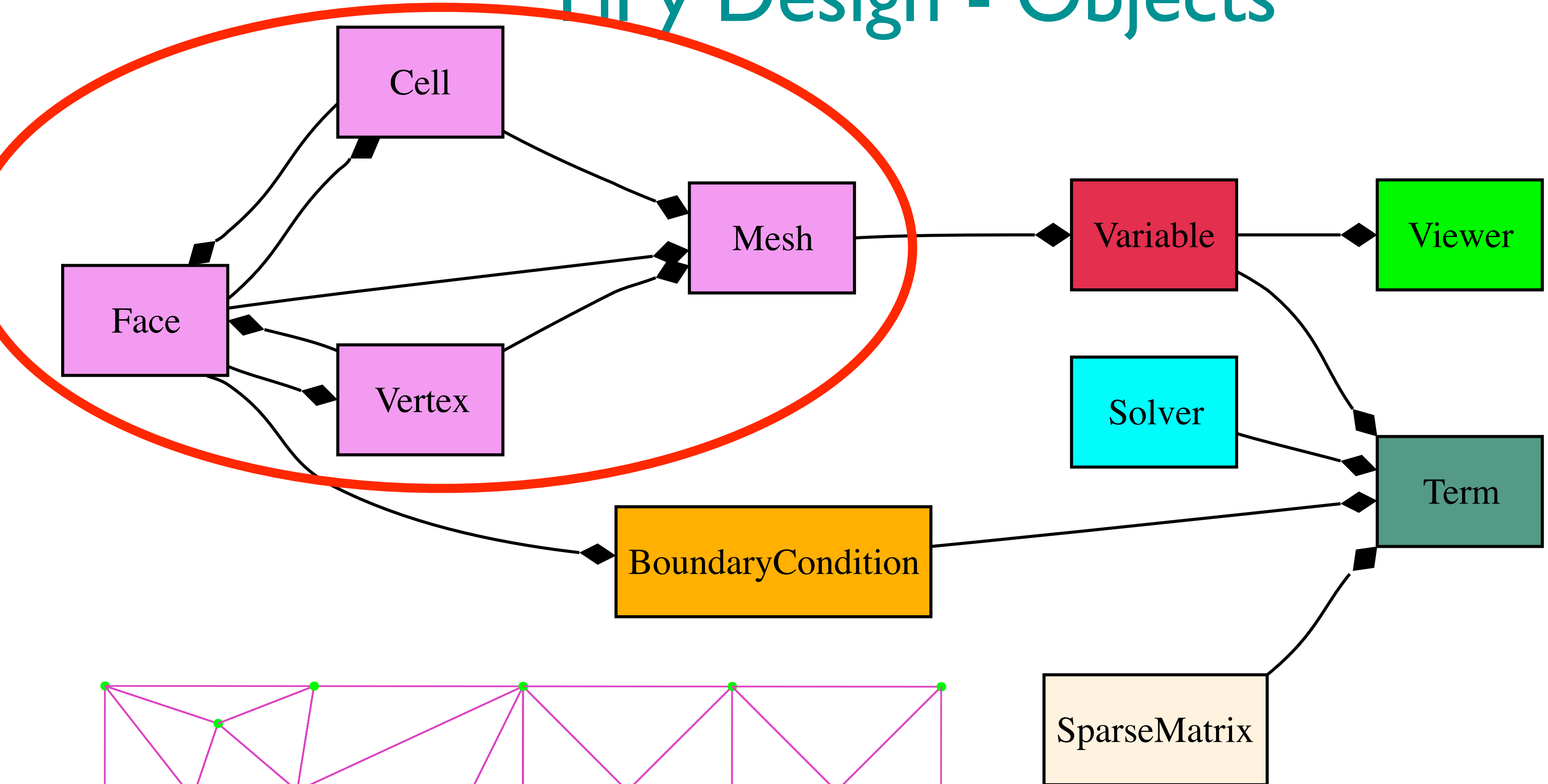


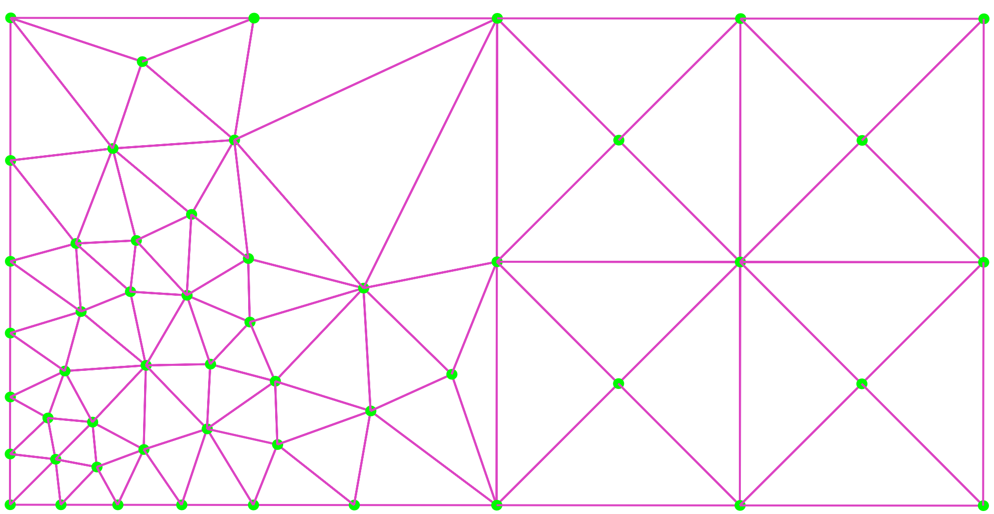
NIST
National Institute of Standards and Technology
Technology Administration, U.S. Department of Commerce

FiPy Design - Objects



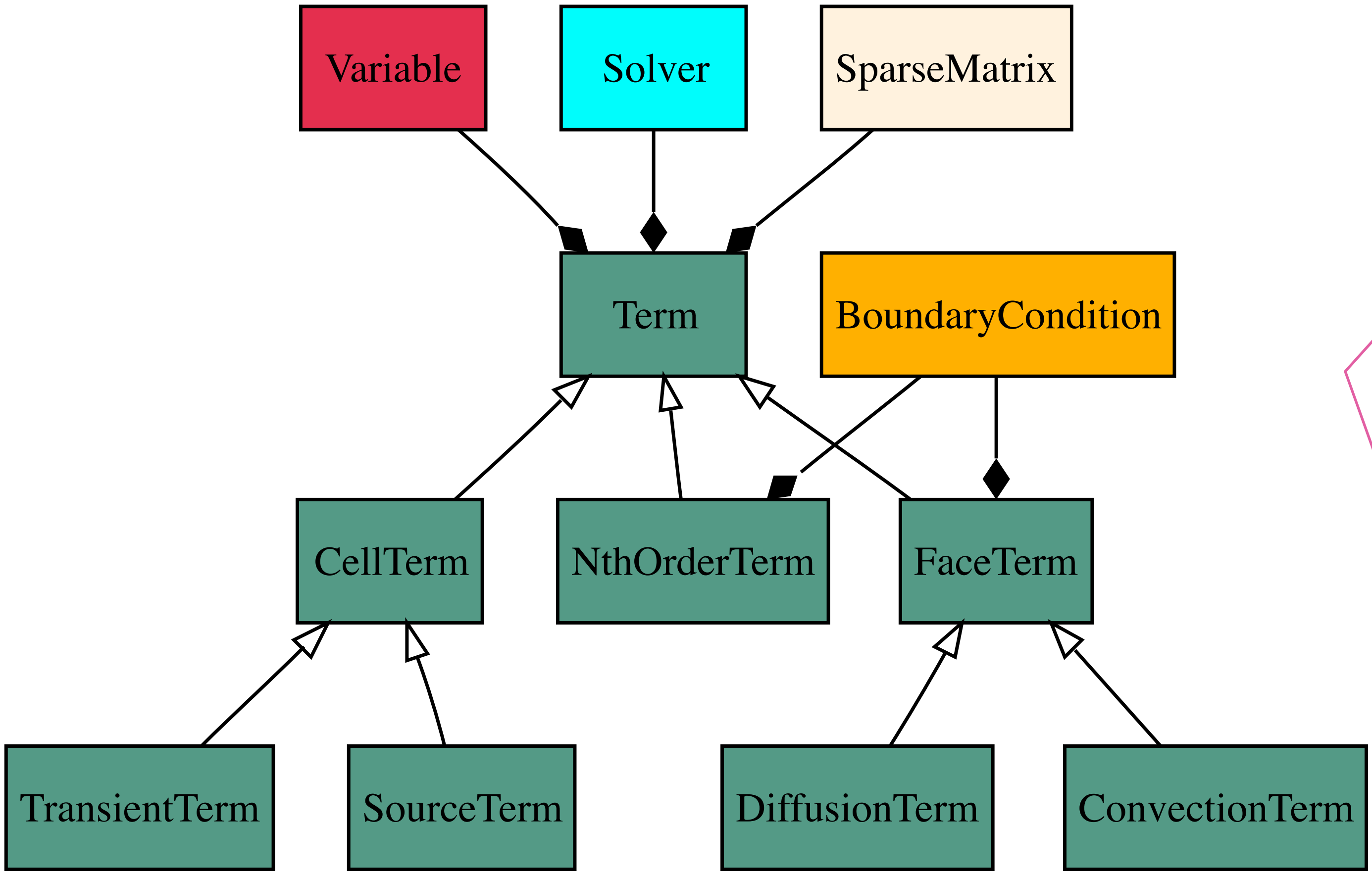
FiPy Design - Objects

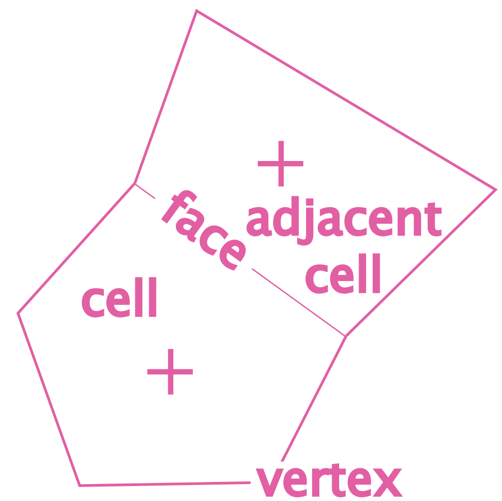


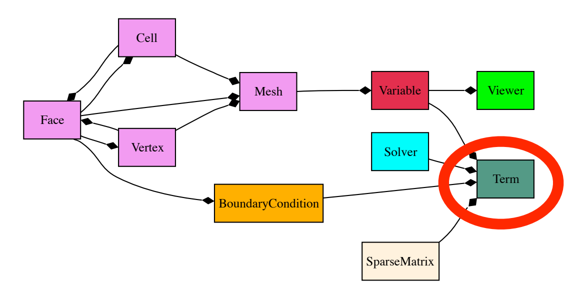


FiPy Design - Terms

$$\underbrace{\frac{\rho\phi V - (\rho\phi V)^{\text{old}}}{\Delta t}}_{\text{transient}} - \underbrace{\sum_{\text{face}} [\Gamma A \vec{n} \cdot \nabla \phi]_{\text{face}}}_{\text{diffusion}} - \underbrace{\sum_{\text{face}} [\Gamma A \vec{n} \cdot \nabla \{\cdots\}]_{\text{face}}}_{n^{\text{th}} \text{ order diffusion}} - \underbrace{\sum_{\text{face}} [(\vec{n} \cdot \vec{u}) A \phi]_{\text{face}}}_{\text{convection}} - \underbrace{V S_\phi}_{\text{source}} = 0$$



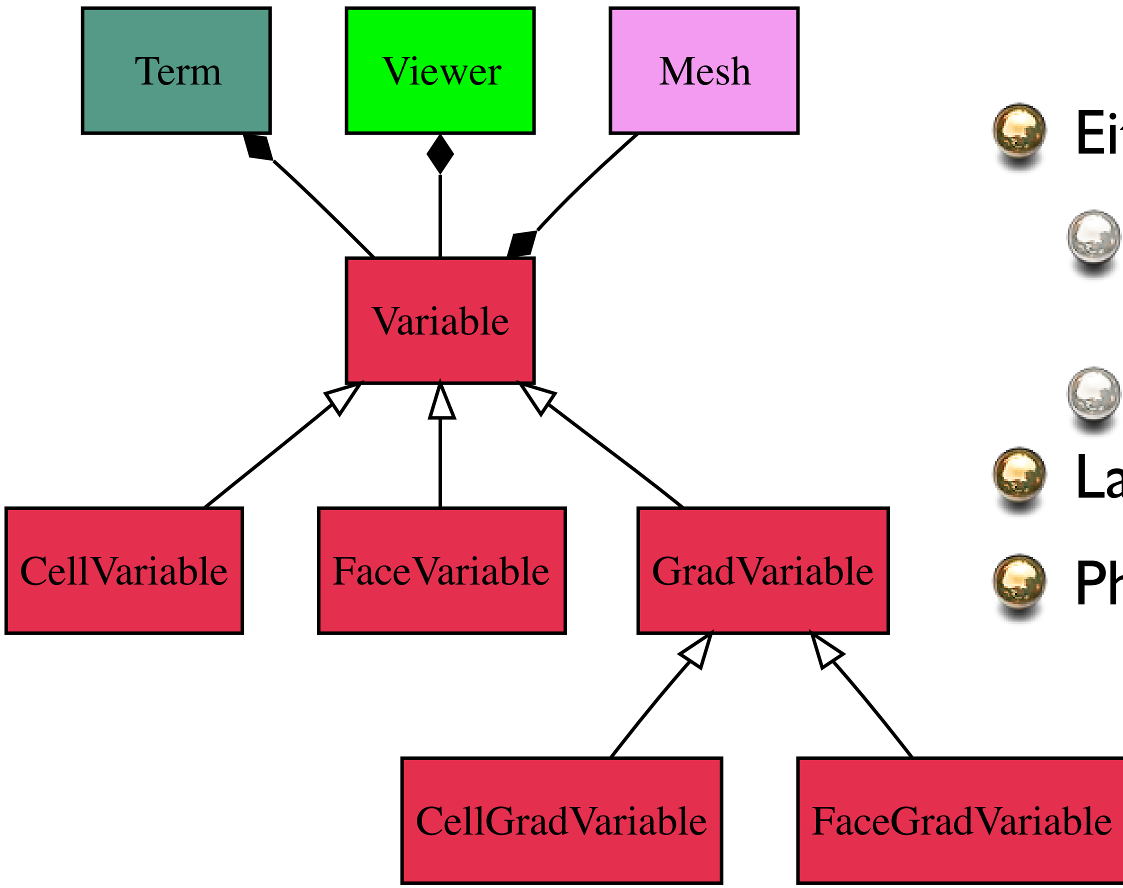




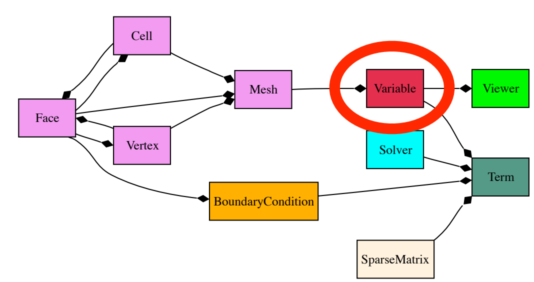
National Institute of Standards and Technology
Technology Administration, U.S. Department of Commerce

FiPy Design - Variables

$$\underbrace{\frac{\rho\phi V - (\rho\phi V)^{\text{old}}}{\Delta t}}_{\text{transient}} - \underbrace{\sum_{\text{face}}[\Gamma A \vec{n} \cdot \nabla \phi]_{\text{face}}}_{\text{diffusion}} - \underbrace{\sum_{\text{face}}[\Gamma A \vec{n} \cdot \nabla \{\cdots\}]_{\text{face}}}_{n^{\text{th}} \text{ order diffusion}} - \underbrace{\sum_{\text{face}}[(\vec{n} \cdot \vec{u}) A \phi]_{\text{face}}}_{\text{convection}} - \underbrace{V S_\phi}_{\text{source}} = 0$$



- Either:
  - solution variables (evaluated by `Term`)
  - set by intermediate calculation
- Lazy evaluation
- Physical dimensions



Efficiency Issues

- Tested efficiency against Ryo Kobayashi's FORTRAN code specifically written to solve grain impingement problem.
- Naive all-Python FiPy was 200X as slow
- "Smarter" all-Python FiPy is 30X as slow
- After optimization and judicious C-inline, FiPy is now 7X as slow
- Development time is reduced considerably
  - FORTRAN requires ≈1800 lines of single-use code
  - Python "smart" requires ≈ 100 lines
  - Python "inline" requires ≈ 300 lines

Future Work

- Adaptive meshes
- Multigrid
- Cell-centered finite volume
- Repair/improve support for physical dimensions
- Export (formatted text, HDF, etc.)
- Viewers (refactor and add more, e.g., OpenDX (Dan Lewis?))
- More examples:
  - Fluid flow
  - Elasticity
  - Electrochemistry Phase Field (implemented, but vexing)
- ???

Summary

- Cross-platform, Open Source code for solving phase transformation problems
- Capable of solving multivariate, coupled, non-linear PDEs
- Extensive documentation, dozens of examples, hundreds of tests
- Python syntax both easy to learn and powerful
- Object-oriented structure easy to adapt to unique problems
- Slower to run than hand-tailored FORTRAN or C…
- …but *much* faster to write

www.ctcms.nist.gov/fipy/

Acknowledgements

- Alex Mont – Montgomery Blair High School
- John Dukovic – Applied Materials
- Daniel Josell – NIST Metallurgy Division
- Tom Moffat – NIST Metallurgy Division
- Steve Langer – NIST Information Technology Laboratory
- Andrew Reid – NIST Materials Science and Engineering Laboratory
- Edwin García – NIST Materials Science and Engineering Laboratory
- Daniel Lewis – GE Ceramic and Metallurgy Technologies