MODULE 3: FINITE ELEMENT METHOD

Practice: OOF2
I. What is OOF2?
OOF2

OOF: Finite Element Analysis of Microstructures

http://www.ctcms.nist.gov/oof/oof2/
Object Oriented Finite Element Analysis Tool

Originally developed in the late 90’s at NIST by Craig Carter, Ed Fuller, Andy Roosen, and Steve Langer

Fundamentally a **two-dimensional** FEM tool

**OOF2** is public domain software created at the National Institute of Standards and Technology (NIST) to investigate the properties of microstructures. The microstructure of a material is the (usually) complex ensemble of polycrystalline grains, second phases, cracks, pores, and other features occurring on length scales large compared to atomic sizes.

At the simplest level, **OOF2** is designed to answer questions like, “I know what this material looks like and what it’s made of, but I wonder what would happen if I pull on it in different ways?”, or “I have a picture of this stuff and I know that different parts expand more than others as the temperature increases -- I wonder where the stresses are greatest?”
- Very user-friendly GUI

- CLI also available
FREE to the end user!
Useability

- Very well documented
  
  http://www.ctcms.nist.gov/~langer/oof2man/

- Many tutorials and guides available
  
  http://www.ctcms.nist.gov/~rlua
  www.ctcms.nist.gov/oof/talks/workshop06/SteveLanger/example.pdf
  http://nanohub.org/resources/4732

- Rollover help within the GUI
Availability

- Free download of source code from
  http://www.ctcms.nist.gov/oof/oof2/#download

- Supported by Linux and Mac OS X, but many pre-reqs

  - Python (2.4 through 2.7)
    - http://www.python.org
  - Magick++
  - gtk+-2.0 (2.6 or later)
    - http://www.gtk.org/download/
  - libgnomecanvas2
    - http://directory.fsf.org/graphics/misc/libgnomecanvas.html
  - pygtk2 (2.6 or later)
    - http://www.pygtk.org
  - swig 1.1 build 883
    - http://www.swig.org/download.html

- Local installation can be tricky

- **EWS**: module load OOF2
But...

OOF: Finite Element Analysis of Microstructures

OOF2

Introduction

OOF2 version 2.1.7 is now available. The major differences between 2.1 and 2.0 are that 2.1 can solve time dependent problems, and has much improved nonlinear solvers. A detailed discussion of the differences and a summary of how to use the new features is included in the What's New in 2.1 page.

OOF2 version 2.1.5 can now be run on the nanoHUB. See the announcement for more details.

OOF2 retains (almost) all of the features of OOF1, although it does not read OOF1 data files. The latest versions of OOF1, however, can write OOF2 data files.

OOF2 is based on a new set of C++ classes for finite elements and material properties, tied together in a Python infrastructure. Python is an easy to use, high-level, object-oriented scripting language.

OOF2 is in an Active Development status.
Availability

Online webtool available at nanoHUB

https://nanohub.org/tools/oof2/

Requires a **free** nanoHUB account for access

(Please register for nanoHUB, Facebook/Google sign-ins can be buggy.)
II. OOF2 basics
All tasks performed in seamlessly and in sequence using OOF2 integrated GUI

Windows based environment
- **OOF2 has no units!**

- Use most convenient units
  - No conversions required

- No error/consistency checking
  - GIGO
  - One error multiplies...
Microstructures

- Fundamental ethos of OOF2 is to process and simulate FEM systems based on experimental microstructures.

- Establish microstructure in OOF2 by **image upload**

- Different phases / materials identified by groups of pixels.

---

silicon carbide micrograph

http://pwatlas.mt.umist.ac.uk/internetmicroscope/micrographs/microstructures/silicon-carbide.html
Materials

- Materials with defined properties created and assigned to pixel groups

- Isotropic or anisotropic properties supported

- Materials properties: Young's modulus, Poisson's ratio, thermal conductivity, coefficient of thermal expansion, dielectric permittivity, viscosity, density, color, ...
A **skeleton** is a partitioning of the micrograph into finite elements, it contains only geometric information.

A **mesh** is derived from the skeleton, containing geometry + equations, fields and boundary conditions.

Modifying the boundary conditions, material properties, or equations to be solved requires rebuilding of the mesh but not the skeleton.

[Image of a skeleton and mesh diagram]

A good skeleton is highly **homogeneous** (homogeneity index \( \sim 0.99 \)) - each element contains a single phase.

![Homogeneity Diagram]

- **Homogeneous** (\( HI = 1.0 \))
- **Heterogeneous** (\( HI \sim 0.3 \))
- **Heterogeneous** (\( HI \sim 0.5 \))

A good skeleton also contains very few high-aspect ratio elements, yielding better FEM numerical solutions.

A number of skeleton refinement tools exist to minimize the effective energy:

\[ E = \alpha E_{\text{homog}} + (1 - \alpha) E_{\text{shape}} \]

- annealing
- edge swapping
- smoothing
- refinement (element partitioning)
The **mesh** is constructed from the **skeleton**.

The mesh elements in OOF2 support **linear** and **quadratic** interpolation functions.

These functions interpolate values from the mesh nodes to the element interiors.
OOF2 supports solution of the following equations:

- heat equation
- force balance
- Coulomb equation
- plane stress
- in-plane polarization
- plane heat flux

Complex boundary conditions supported (Dirichlet, Neumann, periodic, generalized force)

Complex fields possible (T, displacement, voltage)
A number of advanced numerical solvers are available.

User-specified tolerance and maximum iterations.

Problems are typically large and sparse.

**direct:** slow
  high accuracy
  unsuitable for large problems (hi RAM rqmts)

**iterative:** fast
  lower accuracy
  only choice for large problems
Visualization

- Micrograph, skeleton, and mesh visualized in Graphics pane.

- Also used to visualize solutions (stresses, temperature) computed over the terminal mesh.
The solution of scalar and vector fields over the terminal mesh can be outputted using the Analysis pane.

Data can be dumped to file for offline analysis.
The Activity window indicates the current tasks occupying OOF2

Very useful for monitoring long processes (solving, skeleton refinement, visualization)
The Messages window provides detailed task information, and communications to the user.

```
OOF2 Messages 1
File  Windows

Save...  Error

OOF.Graphics_1.Layer.Select(n=6)
OOF.Graphics_1.Layer.Show(n=7)
OOF.Graphics_1.Layer.Select(n=7)
OOF.Graphics_1.Layer.Hide(n=7)
OOF.Graphics_1.Layer.Show(n=2)
OOF.Graphics_1.Layer.Select(n=2)
OOF.Graphics_1.Layer.Show(n=1)
OOF.Graphics_1.Layer.Select(n=1)
OOF.Graphics_1.Layer.Show(n=0)
OOF.Graphics_1.Layer.Select(n=0)
OOF.Skeleton.Modify(skeleton='nanocomposite.tif:skeleton', modifier=Anneal(target=(), criterion=AverageEnergy(alpha=0.8499999999999999), T=0.0, delta=0.005000000000000001, iteration=FixedItems(iterations=500))
Iteration 1: E = 8.3007e+01, deltaE=-5.7525e-03 ( 0.007%), Acceptance Rate = 45.
Iteration 2: E = 8.2974e+01, deltaE=-5.5392e-03 ( 0.007%), Acceptance Rate = 43.
Iteration 3: E = 8.2943e+01, deltaE=-5.5299e-03 ( 0.007%), Acceptance Rate = 45.
Iteration 4: E = 8.2910e+01, deltaE=-5.8175e-03 ( 0.007%), Acceptance Rate = 46.
Iteration 5: E = 8.2876e+01, deltaE=-5.8012e-03 ( 0.007%), Acceptance Rate = 45.
Iteration 6: E = 8.2842e+01, deltaE=-6.0846e-03 ( 0.007%), Acceptance Rate = 46.
Iteration 7: E = 8.2807e+01, deltaE=-6.2262e-03 ( 0.008%), Acceptance Rate = 43.
Iteration 8: E = 8.2769e+01, deltaE=-5.5983e-03 ( 0.007%), Acceptance Rate = 44.
Iteration 9: E = 8.2735e+01, deltaE=-4.8661e-03 ( 0.006%), Acceptance Rate = 44.
```