PencilCode and Python
Why Python?

Highly expressive:

friends = ['john', 'pat', 'gary', 'michael']
for i, name in enumerate(friends):
    print("iteration {iteration} is {name}".format(iteration=i, name=name))

Object oriented

Ease:
Easy to learn, change from IDL or Matlab.

Very active community

Extendability:
libraries for numerics, data analysis, plotting, financing, ...

Interoperability:
Import IDL and Matlab code. “But all my routines are written in IDL.”
Why Python?

“Python is everywhere, it is all around us, even now in this very room.”

3ds Max
Maya
Blender
Cinema 4D
...

Civilization IV, Battlefield 2, World of Tanks, ...

Paraview, Visit, Vapor, ...
Why Python?
Why Python?
Libraries

Matplotlib Gallery

Scipy Functionalities

Tutorial

Tutorials with worked examples and background information for most SciPy submodules.

- SciPy Tutorial
  - Introduction
  - Basic functions
  - Special functions (scipy.special)
  - Integration (scipy.integrate)
  - Optimization (scipy.optimize)
  - Interpolation (scipy.interpolate)
  - Fourier Transforms (scipy.fftpack)
  - Signal Processing (scipy.signal)
  - Linear Algebra (scipy.linalg)
  - Sparse Eigenvalue Problems with ARPACK
  - Compressed Sparse Graph Routines (scipy.sparse.csgraph)
  - Spatial data structures and algorithms (scipy.spatial)
  - Statistics (scipy.stats)
  - Multidimensional image processing (scipy.ndimage)
  - File IO (scipy.io)
  - Weave (scipy.weave)
Libraries

**Python Data Analysis Library**

**pandas**

\[ y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it} \]

http://pandas.pydata.org

**mpmath**

floating-point arithmetic with arbitrary precision

http://mpmath.org

**Hierarchical Data Format**

**vtk Data Format**
Libraries

SunPy
http://sunpy.org

astropy
http://www.astropy.org
import pylab as plt
import pencil as pc

ts = pc.read_ts()
plt.plot(ts.t, ts.urms)
1. Documentation not consequent:

```python
read_ts(*args, **kwargs)
    Read Pencil Code time series data.
    params:
    string: filename   = 'time_series.dat'
    string: datadir    = 'data'
    logical: plot_data = False
    logical: quiet     = False
```

Assemble a 2D animation from a 3D array. *data* has to be a 3D array who's time index has the same dimension as *t*. The time index of *data* as well as its x and y indices can be changed via *dimOrder*.

Keyword arguments:

*dimOrder*: [ (i,j,k) ]
    Ordering of the dimensions in the data array (t,x,y).

*fps*:
Frames per second of the animation.
PencilOld Issues

2. Not much object oriented:

```python
slices, t = pc.read_slices()
```

3. High refraction potential:

```
xyaver.py  xzaver.py  yaver.py  yzaver.py  zaverages.py
zaver.py
```

4. Python3 compatibility issues.

5. Some reading routines are slow (slides).

6. Parallelizability potential.

7. Tried to use IDL structure which is not suited for Python.
```python
slices = pn.read.slices(field='uu1', extension='xy')
slices.t
slices.xy.uu1

# Read multiple slices and extensions.
slices = pn.read.slices(field=['uu1', 'bb2'], extension=['xy', 'yz'])
slices.xy.bb2
slices.yz.uu1
```
Sim Objects

Sim objects are very powerful simulation objects. Capabilities:

1. Read the simulation structure.
2. Modify simulation parameter (*.in files).
3. Modify the cparam.local.
4. Compile.
5. Start, stop and submit (SLURM) a simulation.
6. Create a new simulation and copy it.

```python
import pencilnew as pn
sim = pn.sim.get()
ts = sim.get_ts()
sim.change_value_in_file('start.in', 'ampl', 2.0)
sim.change_value_in_file('cparam.local', 'nxgrid', 128)
sim.compile()
sim.bash('start.csh')
```
import pencilnew as pn
sim = pcn.get_sim('./DUMMY')
for vari in [1, 2, 3]:
    new_sim = sim.copy('new_sim' + str(vari))
    new_sim.change_value_in_file('start.in', 'variable', vari)
    new_sim.compile()
    new_sim.bash('mkdir data; pc_run')