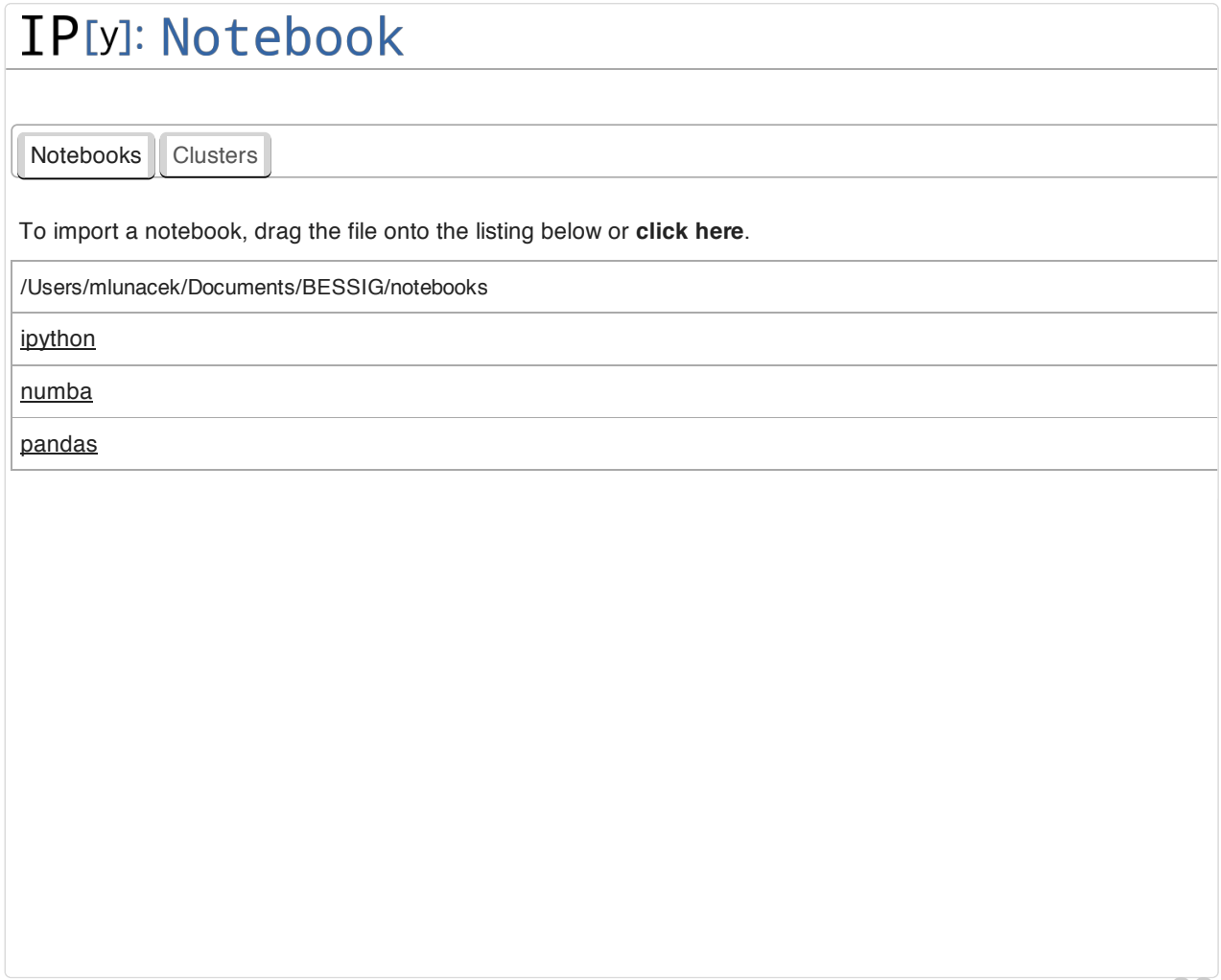
Python: data, speed, and parallel computing

Monte Lunacek

Research Computing, University of Colorado **Boulder**

Data Analysis with Pandas

- http://pandas.pydata.org (http://pandas.pydata.org)
- Efficient data frames
- Reading and writing data
- Data alignment and indexing
- Reshaping
- Merging, joining, group by
- Time-series
- and much more...
- Allows you to stay in Python


IP[y]: Notebook
Notebooks
Clusters
To import a notebook, drag the file onto the listing below or click here.
/Users/mlunacek/Documents/BESSIG/notebooks
ipython
numba
pandas

Speed

Research Computing Tutorial

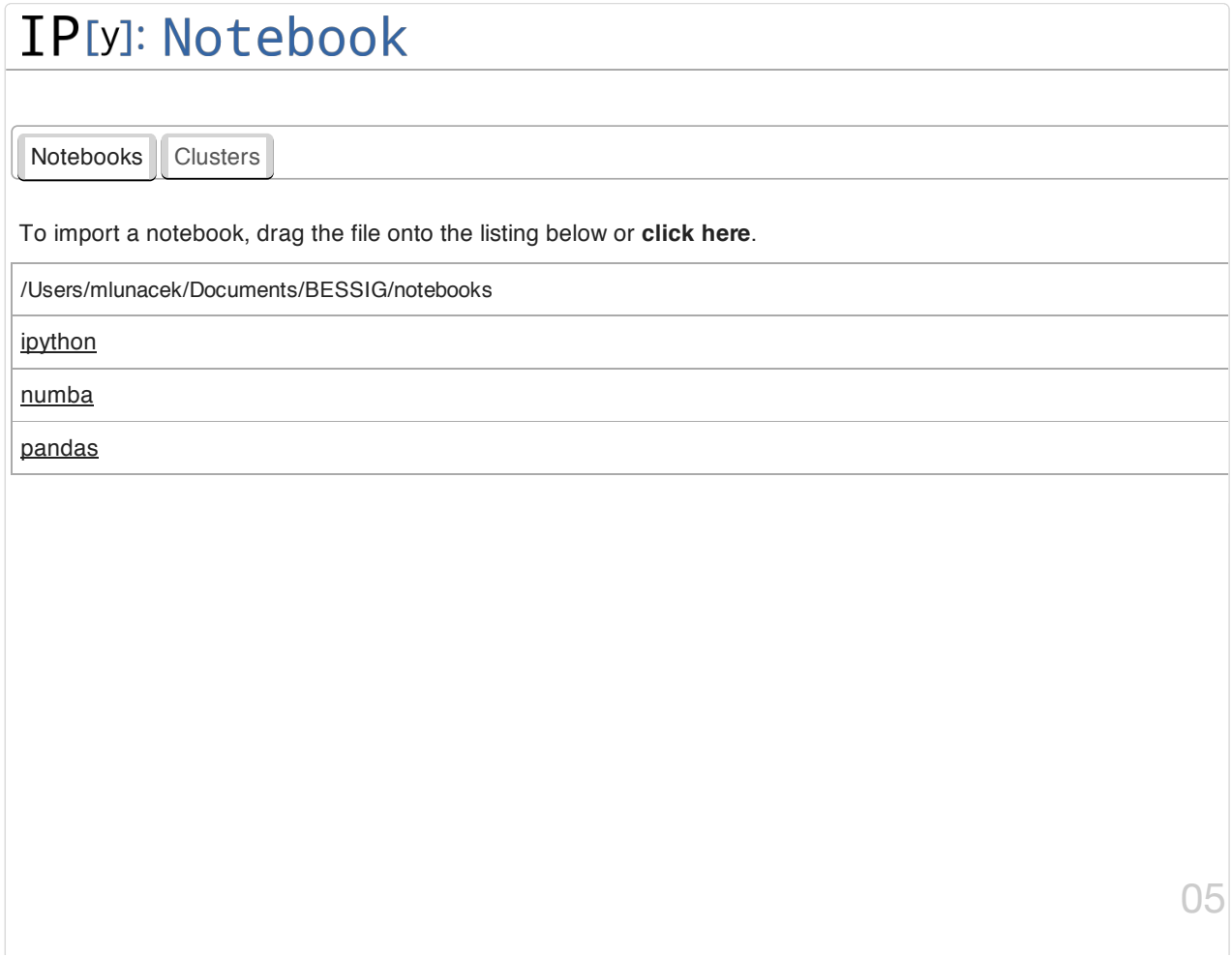
(https://github.com/ResearchComputing/python_hpc/tree/master/10_exte

by Thomas Hauser

- http://www.numpy.org (http://www.numpy.org)
- http://cython.org (http://cython.org)
- http://www.f2py.com (http://www.f2py.com)

Other methods

- ctypes (http://docs.python.org/2/library/ctypes.html)
- fwrap (http://fwrap.sourceforge.net)
- numba (http://numba.pydata.org)



Parallel Python

Multicore wiki

(http://wiki.python.org/moin/ParallelProcessing)

mpi4py (http://mpi4py.scipy.org/)

Scoop (https://code.google.com/p/scoop/)

IPython Parallel (http://ipython.org/ipython-doc/dev/parallel/)

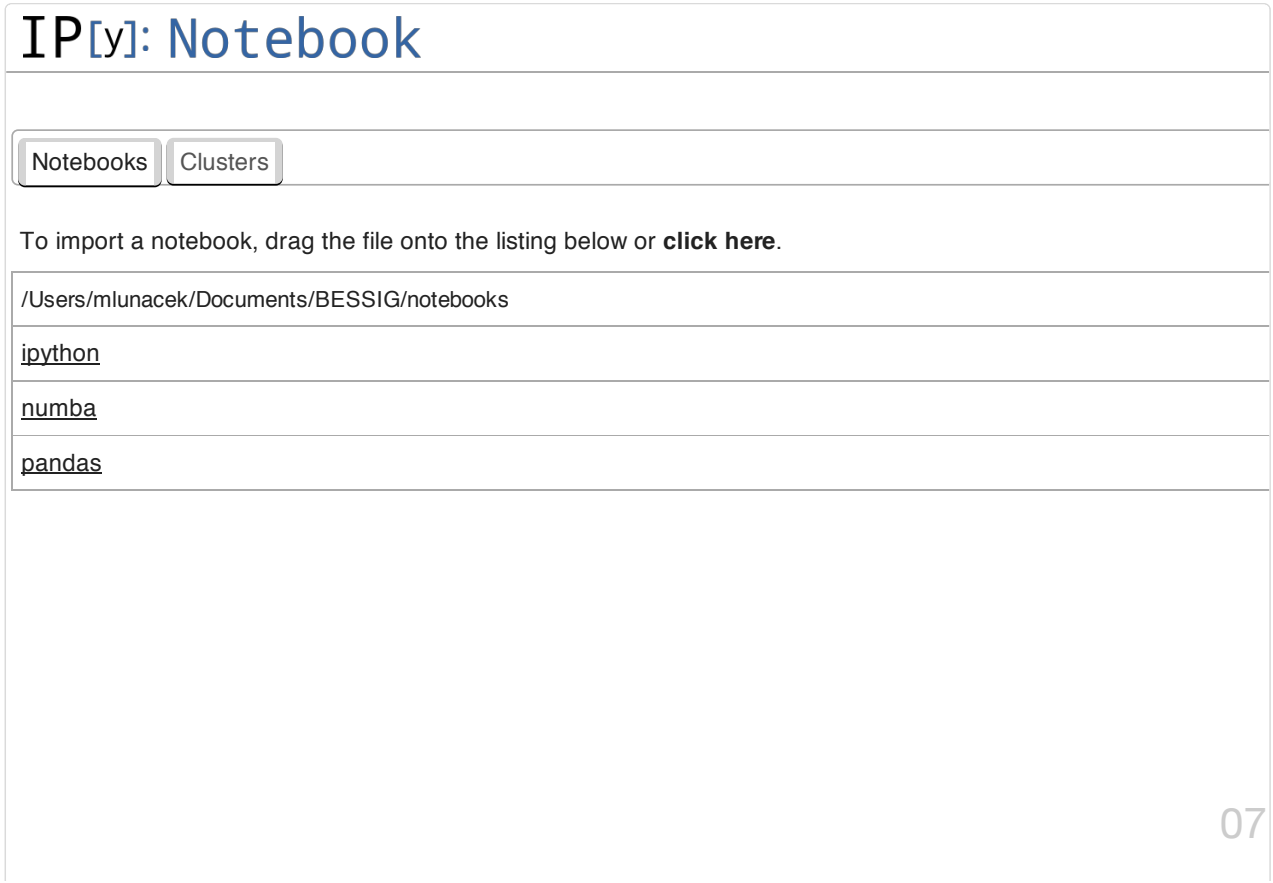
(ZeroMQ)

Celery (http://www.celeryproject.org) (RabbitMQ)

Multiprocessing

(http://docs.python.org/2/library/multiprocessing.html)

GPU


IP[y]: Notebook
Notebooks
Clusters
To import a notebook, drag the file onto the listing below or click here.
/Users/mlunacek/Documents/BESSIG/notebooks
ipython
numba
pandas

Tenacious Robustness Test

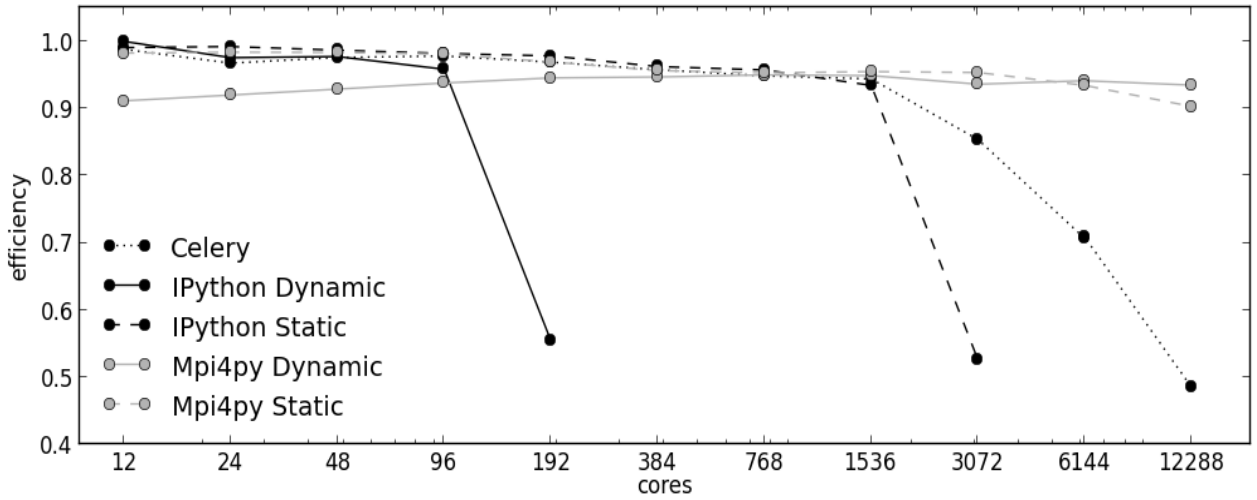
```
@require('time','socket','random',
'IPython.parallel.error.KernelError')
def simulation(x):

   time.sleep(5)
   if random.random() < 0.3:
      raise KernelError
   return {'task':x,
           'host' :socket.gethostname()}
```

Launch 10 nodes

Run **several** tasks

At some point, **kill a node**

Scaling Scaling



[27,33] seconds per job

Average of 10 jobs per core

Conclusions

Have you seen some aspects of Python that make you **excited**?

Combine libraries to achieve the task at hand.

- Notebook

- IPython terminal for developing
- Pandas
- Options for speed
- In **parallel**

References

- Python Scripting for Computational Science (http://www.springer.com/mathematics/computational+science+%26+enc 3-540-73915-9)
- Python Snakes Its Way Into HPC

(http://www.hpcwire.com/hpcwire/2010-11-17/python_snakes_its_way_into_hpc.html)

- Andy Terrel: Getting Started with Python in HPC (http://andy.terrel.us/blog/2012/09/27/starting-with-python/)
- Data Analysis with Python (http://shop.oreilly.com/product/0636920023784.do)
- Optimization with DEAP (https://code.google.com/p/deap/)