

AMath 483/583 — Lecture 29 — June 3, 2011

Today:

- Reproducible research
- Binary I/O
- Animation: plots to movies
- Sage
- Parallel IPython
- Course evaluations

Some new examples:

```
$CLASSHG/codes/io
$CLASSHG/codes/graphics
$CLASSHG/codes/python/mectest.py
```

Notes:

ASCII vs. binary output

Often need to write out a large array of floats with full precision.

For example, one solution value on 3d grid ...

```
210      do i=1,n
          do j=1,n
            do k=1,n
              write(21,210) u(i,j,k)
              format(e24.16)
            enddo; enddo; enddo
```

How much disk space does this take?

This writes 24 ASCII characters (1 byte each) for each number, requires 24 bytes to store e.g. $0.40000000000000000000E+01$ so $24n^3$ bytes (24 MB if $n = 100$) for full array.

Note: In memory storing one 8-byte float takes only 8 bytes. (8 MB if $n = 100$.) **ASCII takes 3× the space.**

Also takes additional time to convert to ASCII, **≈ 10× slower to write ASCII than dumping binary.**

Notes:

Binary output in Fortran

Can use **unformatted** write in Fortran:

```
! $CLASSHG/codes/io/binwrite.f90
open(unit=20, file="u.bin", form="unformatted", &
      access="direct", recl=8*m*n)

do j=1,n
  do i=1,m
    u(i,j) = real(m*(j-1) + i, kind=8)
  enddo
enddo

write(20,rec=1) u
close(20)
```

This writes 1 record of length $recl=8*m*n$.

The resulting binary file `u.bin` cannot be edited directly.

But we can read it into Python...

Notes:

Reading binary data files in Python

To recover U array of dimension $m \times n$ in Python:

```
# $CLASSHG/codes/io/binread.py

from scipy.io import numpyio

file = open('u.bin', 'rb')
m = ...
n = ...
u = numpyio.fread(file, m*n, 'd')

# now use Fortran ordering to reshape,
# filling U by columns:
U = u.reshape((m,n), order='F')
```

Notes:

Other options for binary data

Binary formats that contain a lot of [metadata](#)...

[Hierarchical Data Format](#): HDF, HDF4, HDF5

HDF5 file structure includes two major types of object:

- [Datasets](#): multidimensional arrays of a homogenous type
- [Groups](#): container structures for datasets and other groups

[NetCDF](#):

<http://www.unidata.ucar.edu/software/netcdf/>

“a set of software libraries and self-describing, machine-independent data formats that support the creation, access, and sharing of array-oriented scientific data.”

Notes:

5 possible ways to animate...

1. Some graphics packages have animation tools.
2. Create a sequence of images, view them one by one, On the fly, pausing between frames.

Create sequence of image files, e.g.

frame0001.png, frame0002.png, etc.

and then either:

3. Combine into single animated file, movie.gif or movie.mpg, etc.
4. Create html page that loads them one by one to create animation.

[Examples](#): [\\$CLASSHG/codes/graphics/movies](#)

5. Use Sage.

Notes:

On the fly plotting

```
from matplotlib import pyplot as plt
import time

pause_time = 0.4 # seconds between frames

for n in range(nsteps+1):
    plt.clf() # clear frame
    # plot frame n with necessary plot commands
    plt.draw() # make sure screen updated
    time.sleep(pause_time)
```

Disadvantages: Cannot view again without recomputing,
Hard to share with others.

Notes:

Creating a sequence of image files

```
for n in range(nsteps+1):
    plt.clf() # clear frame
    # plot frame n with necessary plot commands
    plt.draw() # make sure screen updated
    fname = "frame%s.png" % str(n).rjust(4,'0')
    plt.savefig(fname)
```

This creates frame0000.png, frame0001.png, etc.

Can combine into a single animated gif via Unix **convert**:

```
$ convert -delay 20 frame*.png movie.gif
```

Other formats also possible. See man page or

<http://www.imagemagick.org/script/convert.php>

Notes:

ImageMagick convert

For documentation see:

<http://www.imagemagick.org/script/convert.php>

Useful for converting single image between file types, e.g.

```
$ convert myplot.png myplot.pdf
```

For resizing image:

```
$ convert myplot.png -resize 50% smallplot.png
```

and many image processing tools (blurring, etc.)

Notes:

Animating a sequence of image files in html

```
$CLASSHG/codes/graphics/movies/html_movie.py

import html_movie
plotfiles = []
for n in range(nsteps+1):
    # plot frame n with necessary plot commands
    fname = "frame%s.png" % str(n).rjust(4,'0')
    plt.savefig(fname)
    plotfiles.append(fname)
html_movie.make_movie(plotfiles, "movie.html")
```

This creates an html file that uses JavaScript to loop through frame0000.png, frame0001.png, etc. in the browser.
Includes buttons to pause movie, change speed, etc.

Notes:

Sage

Sage is an open source math software project.

<http://www.sagemath.org>

Founded by Prof. William Stein of the UW Math Department.

Python-based, includes > 100 packages in all fields of mathematics, symbolic manipulation, etc.

[Sage notebook](#) web-based interface, useful for experimenting and writing up notes.

Try it out on-line: <http://www.sagenb.org>

Many sample worksheets give an idea of what's possible.

Notes:

Parallelization in IPython

There are good instructions on how to do this at:

<http://ipython.scipy.org/doc/rel-0.9.1/html/parallel/>

Example: `$CLASSHG/codes/python/mectest.py`

Notes:

The End

Thanks for participating in this class.

Many thanks to our awesome TA, Grady Lemoine!

Have a good summer.

Notes: