2014-04-18.sagews

April 18, 2014

Contents

1 Math 480b Sage Course

1.1 April 18, 2014

Screencast: `http://youtu.be/c6st7_3VmDU`

Plan

- Questions?
- Homework reminder: everything due tonight at 6pm.
- Python class: the field of rational numbers
- Example decorators: @interact, @parallel, @cached_function, @disk_cached_function
- Something different: some 2D (and maybe 3D?) graphics in Sage

```
f = open('myfile','w')   # 'r', 'a'
f
```

```
<open file 'myfile', mode 'w' at 0x7acc390>
```

```
f.write("hi")
```

```
f.flush()
```

```
f.close()
```

1.2 Another Python Class: The Field of Rational Numbers

- quick review: look at rational.sage
- goal: make a class that models The Field of Rational Numbers

```
class RationalField:
    def __init__(self):
        pass

    def __repr__(self):
        return "The Rational Numbers"
    def cardinality(self):
        return "really, really big (infinite!)"
    def __iter__(self):  # wrong
        i = 0
        while True:
            yield i
            i += 1
```

```
Q = RationalField()
Q
```

```
The Rational Numbers
```

```
QQ
```

```
Rational Field
```

```
j = 0
for n in QQ:
    j += 1
    print n,
    if j > 1000:
        break
```

```
%load rational.sage
class RationalField:
    def __init__(self):
        pass
    def __repr__(self):
        return "The Rational Numbers"
    def cardinality(self):
        return "really, really big (infinite!)"
    def __iter__(self):  # wrong
        i = 0
        while True:
            yield i
            i += 1
    def __call__(self, n, d):
        return RationalNumber(n, d)
```

```
Q = RationalField()
Q
```

```
The Rational Numbers
```

```
Q(2,3)
```
```
2/3
```

```
Q(-5,1)
```
```
-5/1
```

```
QQ.zero_ideal()
```
```
Principal ideal (0) of Rational Field
```

```
random_matrix(QQ,10)
```
```
[  -1    0    2    0  1/2    1    0   -2   -1    0]
[   1    0   -1    2    0    0    0    0   -1    0]
[  -2    0    0    0    0    1   -1    0    0   -1]
[   0    0   -1 -1/2    1   -1    1    1    1    0]
[ 1/2    1    0    1    0  1/2    0   -1    0    2]
[   1    0   -2    0    0    0    2   -2    0  1/2]
[   0    1    2    1    1 -1/2   -2    1    0   -2]
[ 1/2    0    2   -2   -1  1/2    0   -2    1  1/2]
[   0   -2   -1    2    2   -2    0    2  1/2   -2]
[   0  1/2    0   -2    2   -2   -2    0   -1   -1]
```

```
show(QQ)
```
$$Q$$

```
latex(QQ)
```
```
\Bold{Q}
```

1.3 Example Decorators

(these are sage-specific)

1.3.1 @cached_function

```
@foo
def f(...):
    ...

THE SAME AS

def f(...):
    ....
f = foo(f)
```

```
cached_function
```

```
def stupid(n):
    print "uhhhhh..."
    sys.stdout.flush()
```

```
    sleep(2)
    return n*n
```

```
stupid(17)
```

```
uhhhhh...
289
```

```
@cached_function
def stupid(n):
    print "uhhhhh..."
    sys.stdout.flush()
    sleep(2)
    return n*n
```

```
stupid(17)
```

```
uhhhhh...
289
```

```
stupid(17)
```

```
289
```

```
get_memory_usage()
```

```
1060.99609375
```

```
stupid(178)
```

```
uhhhhh...
31684
```

```
get_memory_usage()
```

```
1060.99609375
```

1.3.2 @disk_cached_function

```
disk_cached_function(
```

```
@disk_cached_function('stupid')
def stupid(n):
    print "uhhhhh..."
    sys.stdout.flush()
    sleep(2)
    return n*n
```

```
stupid(1457)
```

```
uhhhhh...
2122849
```

```
%time stupid(1457)
```

```
2122849
CPU time: 0.00 s, Wall time: 0.00 s
```

```
# don't do the above millions of times.
USE sqlite
```

```
import sqlite3
```

1.3.3 @interact

```
@interact
def f(n=[1..12], m=(1..100), c=Color('red')):
    print "%s * %s = %s"%(n,m, n*m)
    print "c = ", c
```

```
f.m = 50
f.n = 4
f.c = 'green'
```

```
f(4,5)
```

```
4 * 5 = 20
```

```
f(17,18)
```

```
17 * 18 = 306
```

1.3.4 @parallel

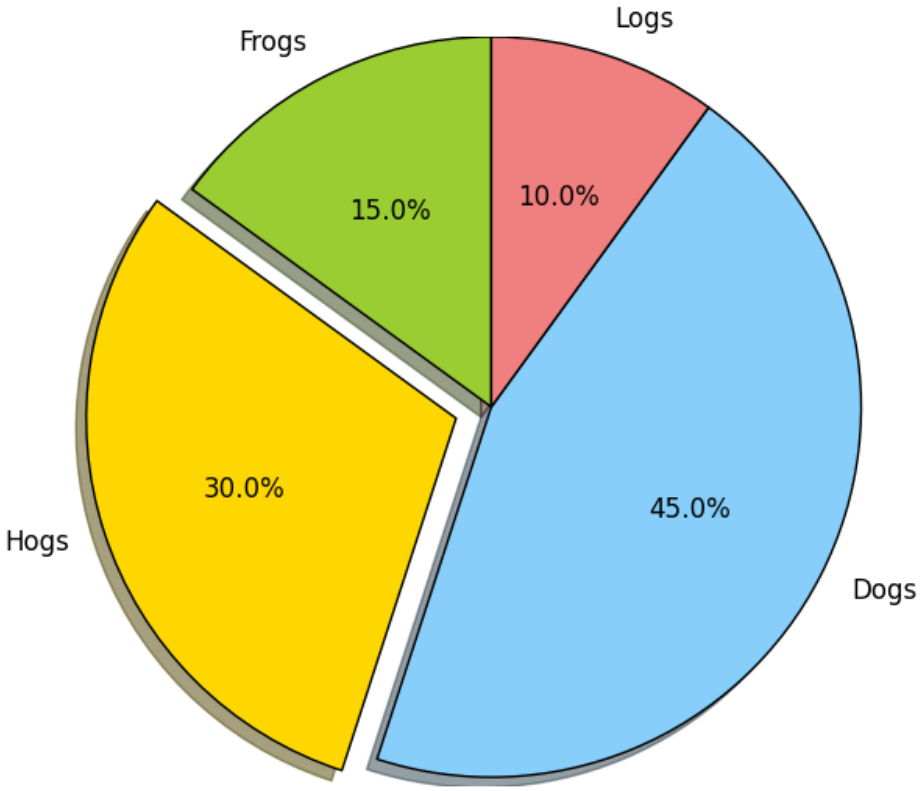
1.4 Some graphics in Sage

- How do I plot a function?
- How do I plot a polygon?
- Plot a random walk?
- Matplotlib plotting: http://matplotlib.org/gallery.html
- Some regular polytopes, random sphere, function of two variables.

```
import matplotlib.pyplot as plt


# The slices will be ordered and plotted counter-clockwise.
labels = 'Frogs', 'Hogs', 'Dogs', 'Logs'
sizes = [15, 30, 45, 10]
colors = ['yellowgreen', 'gold', 'lightskyblue', 'lightcoral']
explode = (0, 0.1, 0, 0) # only "explode" the 2nd slice (i.e. 'Hogs')

plt.pie(sizes, explode=explode, labels=labels, colors=colors,
        autopct='%1.1f%%', shadow=True, startangle=90)
```

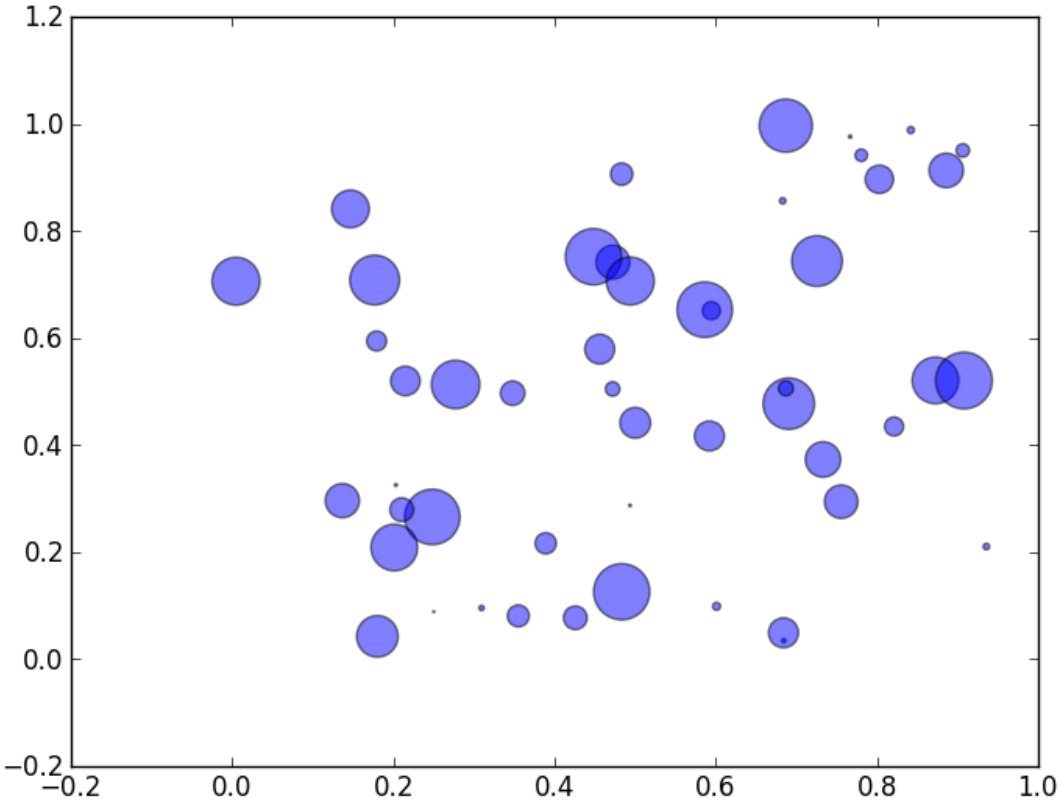
```
# Set aspect ratio to be equal so that pie is drawn as a circle.
plt.axis('equal')
plt.show()
```

```
([<matplotlib.patches.Wedge object at 0x7a98510>, <matplotlib.patches.Wedge object at
0x7895f10>, <matplotlib.patches.Wedge object at 0x7895cd0>, <matplotlib.patches.Wedge
object at 0x7b9b610>], [<matplotlib.text.Text object at 0x7a988d0>, <matplotlib.text.Text
object at 0x7895e50>, <matplotlib.text.Text object at 0x7b9bc50>, <matplotlib.text.Text
object at 0x7874050>], [<matplotlib.text.Text object at 0x7a98e10>, <matplotlib.text.Text
object at 0x7895d50>, <matplotlib.text.Text object at 0x7b9bf90>, <matplotlib.text.Text
object at 0x7874690>])

(-1.1074797737419919, 1.027940084990864, -1.0246160299958926, 1.0000000049123423)
```




```
plt
```

```
<module 'matplotlib.pyplot' from '/usr/local/sage/sage-6.2/local/lib/python2.7/site-
packages/matplotlib/pyplot.py'>

'\nSimple demo of a scatter plot.\n'
<matplotlib.collections.PathCollection object at 0x7848450>
```


1.2
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.2
0.0
0.2
0.4
0.6
0.8
1.0