Math 480a: Sage, Spring 2011, Homework 4

William Stein

Due April 27, 2011

Instructions: Do the following three problems, and turn them in by the beginning of class on Wednesday, April 27, 2011. If you get help from a classmate or friend on any homework problem, please explicitly thank them in your write up. These problems will mostly help you develop your knowledge of Cython and Numpy.

1. Create a straightforward naive implementation of Gauss elimination (reduced row echelon form) for arbitrary 2d numpy arrays.

   ```
   INPUT: A 2-dimensional numpy array
   OUTPUT: The reduced row echelon form of that array
   ```

   Don't worry about issues of roundoff error, deciding whether something is 0 or small, etc. If you have never taken a class on matrix algebra (e.g., Math 308) or can't remember what Gauss elimination is, just look up a description of the Gauss elimination algorithm yourself (e.g., type "Gauss elimination" into Google). For example,

   ```
   sage: import numpy
   sage: a = numpy.array([[1,2,3], [4,5,6], [7,8,9]], dtype=float)
   sage: a
   array([[ 1.,  2.,  3.],
          [ 4.,  5.,  6.],
          [ 7.,  8.,  9.]])
   sage: a.shape
   (3, 3)
   sage: npgauss(a)
   array([[ 1.,  0., -1.],
          [-0.,  1.,  2.],
          [ 0.,  0.,  0.]])
   sage: a    # original array should be unchanged
   array([[ 1.,  2.,  3.],
          [ 4.,  5.,  6.],
          [ 7.,  8.,  9.]])
   ```

2. Rewrite your Gauss elimination implementation from Problem 1 using Cython. Don't use any special annotations, datatypes or other speedups. Compare timings with various sizes of arrays and dtypes. Be creative about what array sizes and dtypes you use. Don't just test your implementation on square matrices.

3. Rewrite your implementation from Problem 2 using any Cython trickery you can think of to try to make it fast. E.g., see `http://docs.cython.org/src/tutorial/numpy.html`. Compare timings with various sizes of arrays and dtypes. How did you do? Did you get a big speedup or not? Why or why not?

Contest. Try your best implementation on the following input

```
sage: a = numpy.random.rand(500,600)
sage: time v = npgauss(a)
Time: CPU xxx s, Wall: xxx s
```

We will see who in the class can get the best CPU time.