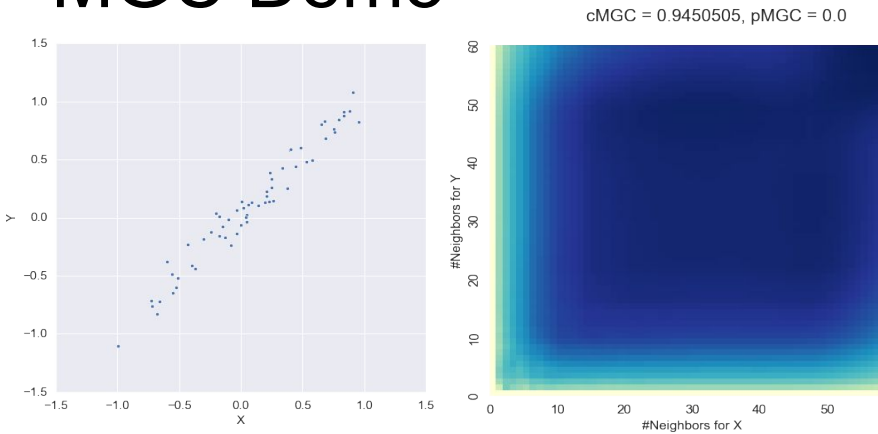
Task 2 (Satish): Implement MGC into the package

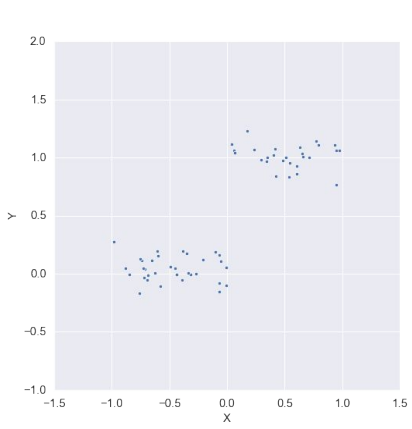
Last Week Accomplishments:

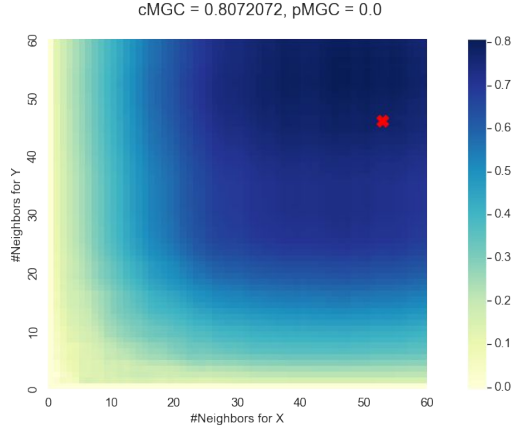
- Extract Bear's permutation test from DCorr into a common utility and use that to compute p-value in MGC and write tests.
  - For now, have just copied it, when Bear merges to development, will call that.
- Compare Python MGC with R version and report the performance.
- Write a master function in main.py that can call the different independence tests.
- Fix bug in MGC test statistic computation and a few bugs in simulations.
- Push mgcpy to PyPi.
- Add a demo notebook for MGC

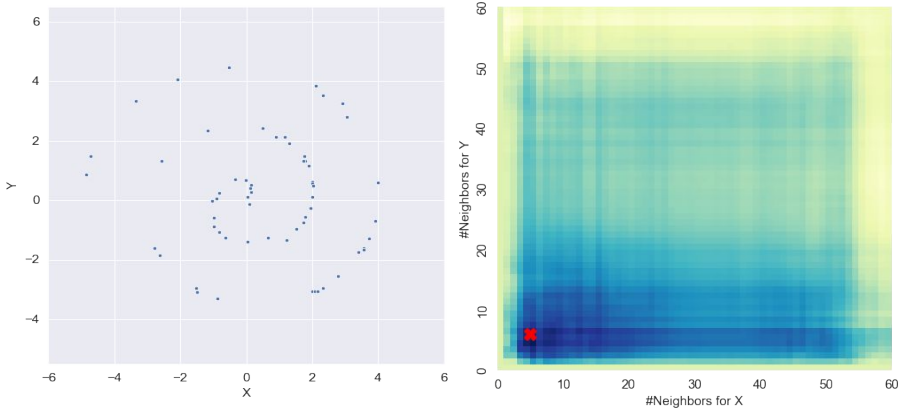
```
pip3 install mgcpy
```

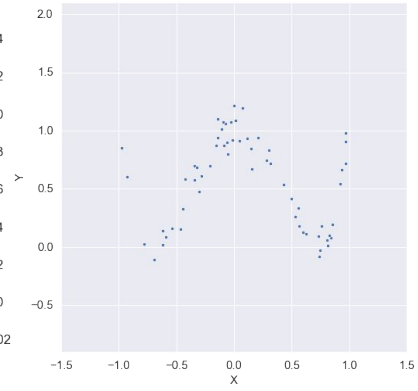
MGC Demo

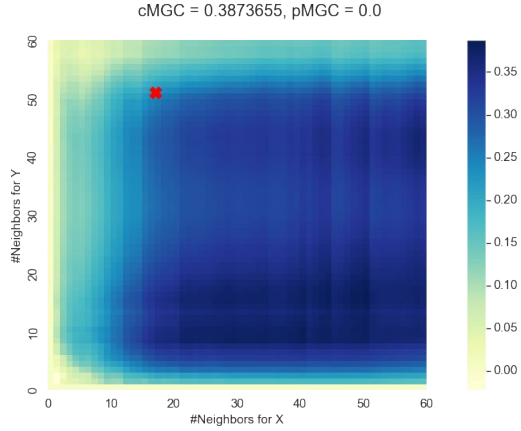








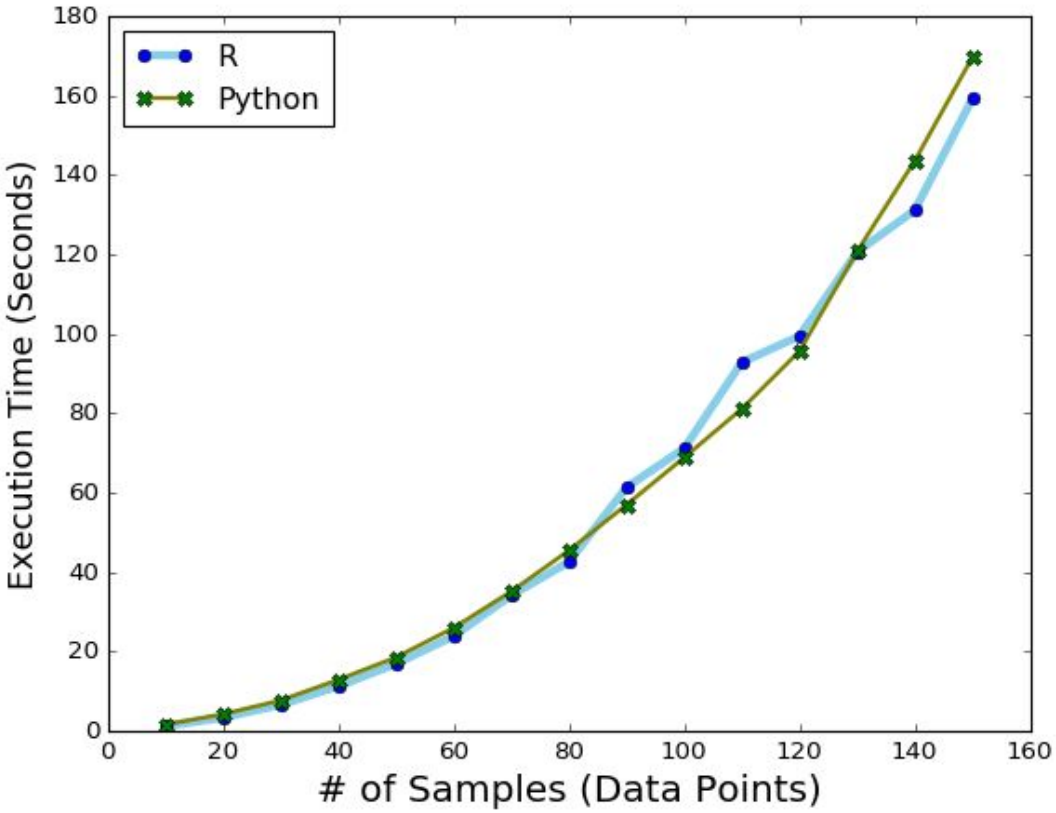




Note: 60 samples, 1 dimension each, with noise of 0.1 (notebook)

MGC performance comparison: Python vs R (notebook)

- **Data:**
  - Linear data (simulated)
  - # of Samples: 10 to 150; steps: 10
  - 1 D with noise: 0.1
- **Execution Time:**
  - **Modules used:** *timeit* for Python and *microbenchmark* for R
  - Best of 5 trials
- **Result:**
  - R version of MGC is mostly faster than the Python version
  - Effect is more pronounced as # of samples increases
- **Specs:**
  - **OS:** macOS 10.14
  - **CPU:** 2.3 GHz (i5, dual core)
  - **RAM:** 8GB, 2133MHz



Stuff to do this week

- Look where Python MGC is slowing down and try to improve its performance
- Study about *fastMGC* and start exploring the MATLAB code, and port some initial code into Python
    - Get link to *fastMGC* paper from jovo?