MRI Simulator Notes

Liam

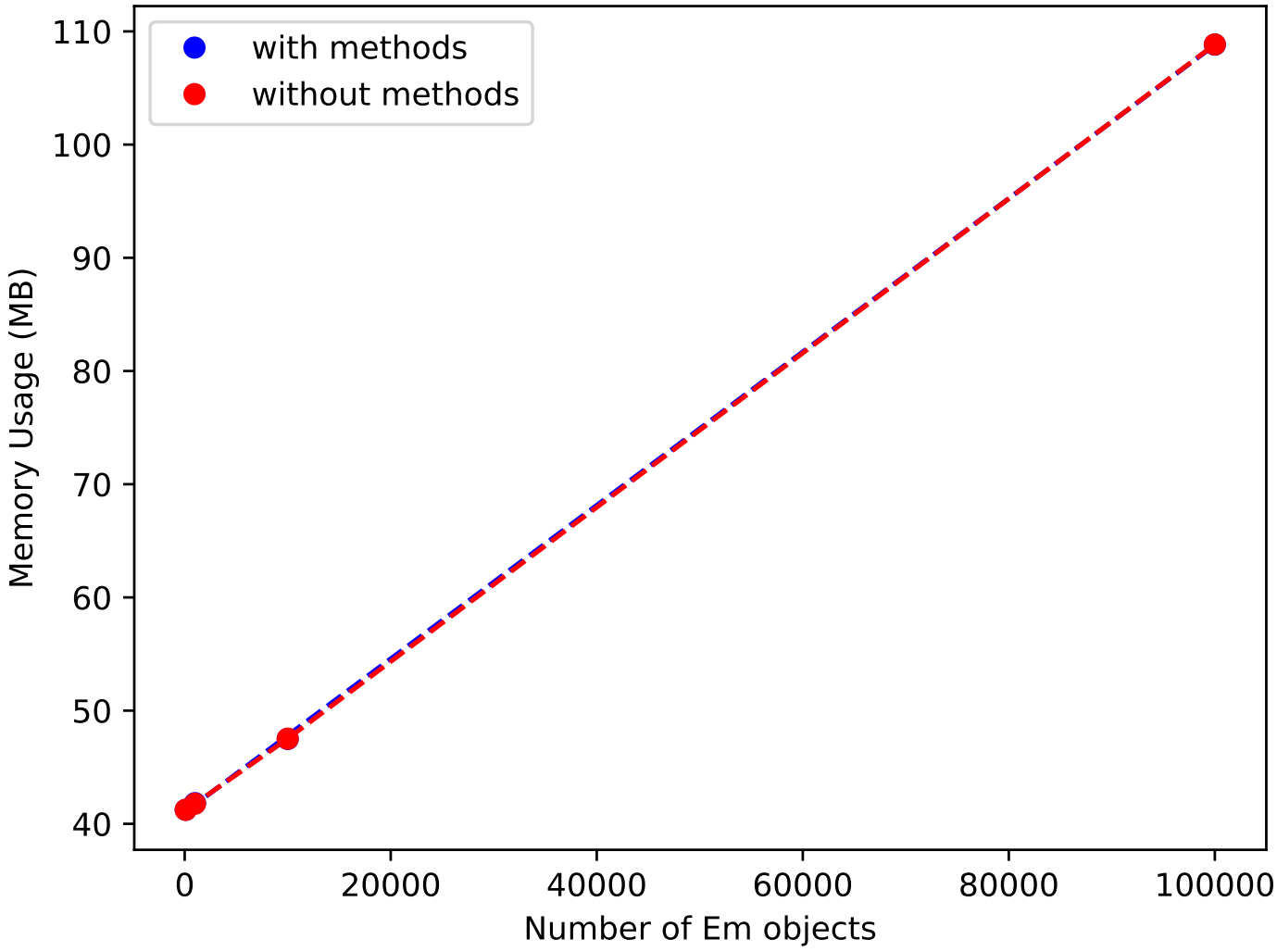
Monday 2nd December, 2019

Tasks

- I looked into memory management in Python.
- I looked further into distributed computing with Ray.
- I wrote code to generate the Pulse objects associated with Cartesian sampling of k-space.

Memory management in Python

- This Stack Overflow post claims that instances of objects do not store their own copies of methods.
- I checked the memory usage of Em objects using resource.getrusage.ru_maxrss, which returns the maximum resident set size used in bytes (reference). I compared the memory usage without and without methods in the class declaration: no difference.



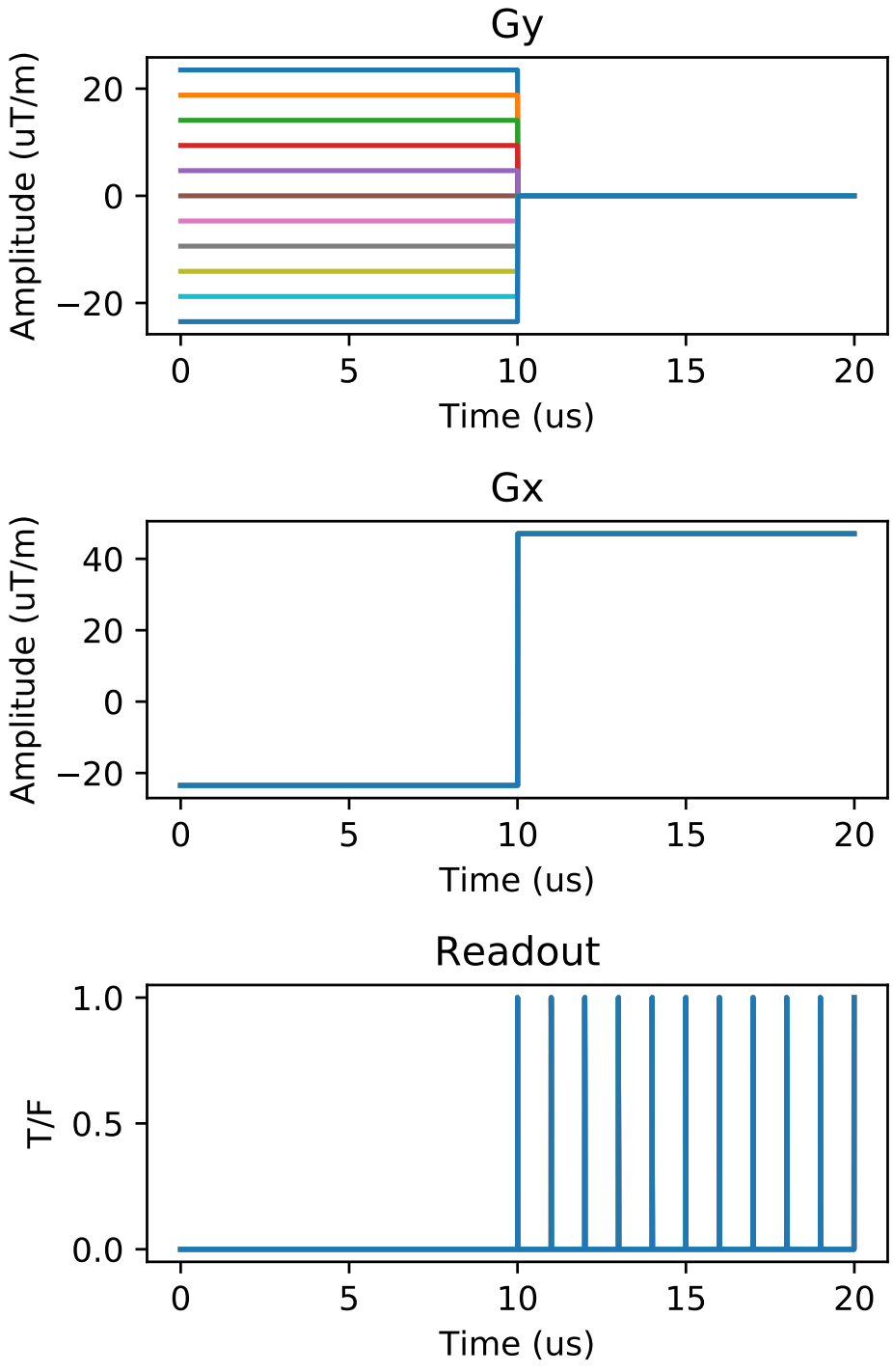
Memory usage per em with/without methods: 0.68 kB/0.68 kB

Distributed computing in Python with Ray

- I looked up further information about distributed computing in Python using Ray.
- This post and the Ray README on GitHub seems to confirm that you can write a Python function/class to run on a single machine and then add a decorator and a few lines prior to the function call/class instantiation call to run the function asynchronously over a cluster.

Cartesian sampling pulses

- I set the number of samples in the phase-encoding and frequency encoding directions as 11.
- I set $k_{x,max} = k_{y,max} = 1.0\ \mathrm{cm}$.
- I used the gyromagnetic ratio of H-1.
- See next slide for the pulses.


Gy
Amplitude (uT/m)
20
0
−20
0
5
10
15
20
Time (us)
Gx
Amplitude (uT/m)
40
20
0
−20
0
5
10
15
20
Time (us)
Readout
T/F
1.0
0.5
0.0
0
5
10
15
20
Time (us)

- I wrote a function to compute the k_x and k_y samples from such a set of pulses.
- This function returns the expected values $k_x = [-1 : 0.2 : 1]\ \mathrm{cm}$ and $ky = [1 : -.2 : -1]\ \mathrm{cm}$ when called with the pulse sequence shown above.