

Basic Galaxy Merger Simulation with Fast Fourier Transform Poisson Solver

Sarah Draves

Goal

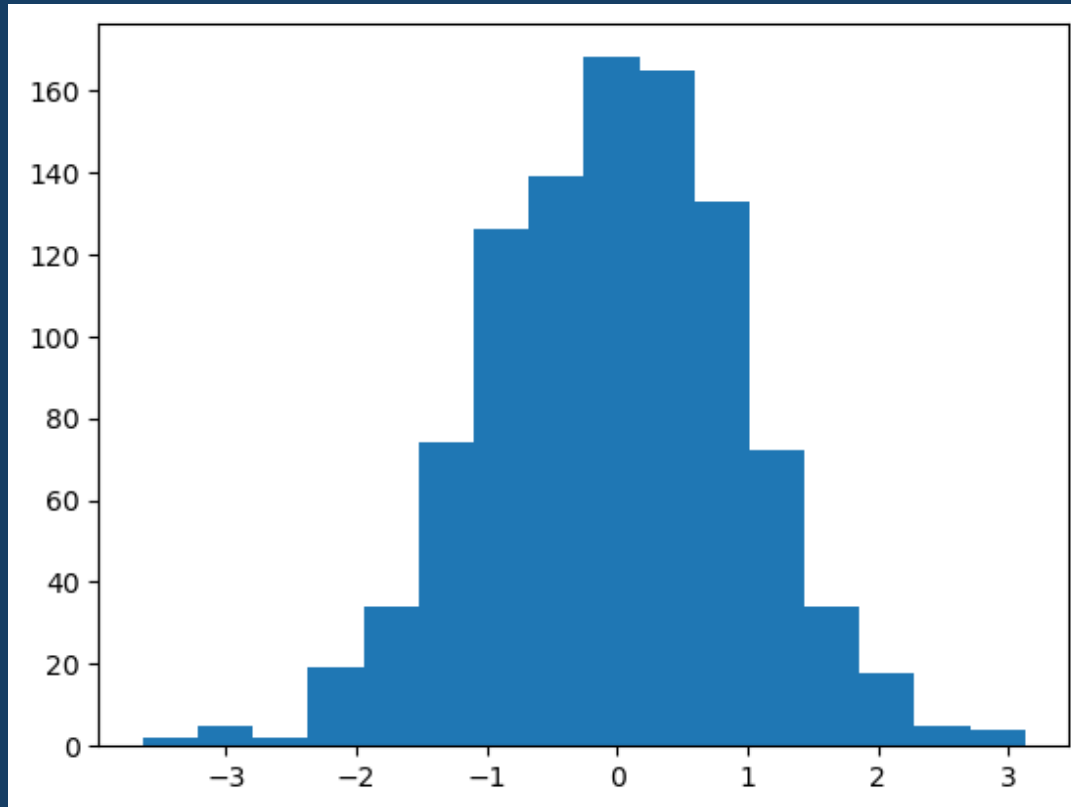
- Learn how to implement FFT with Python packages
- Solve Poisson equation with FFT for finite difference grid
- Have density grid calculated from underlying points and the position of the points updated with each time step

Process

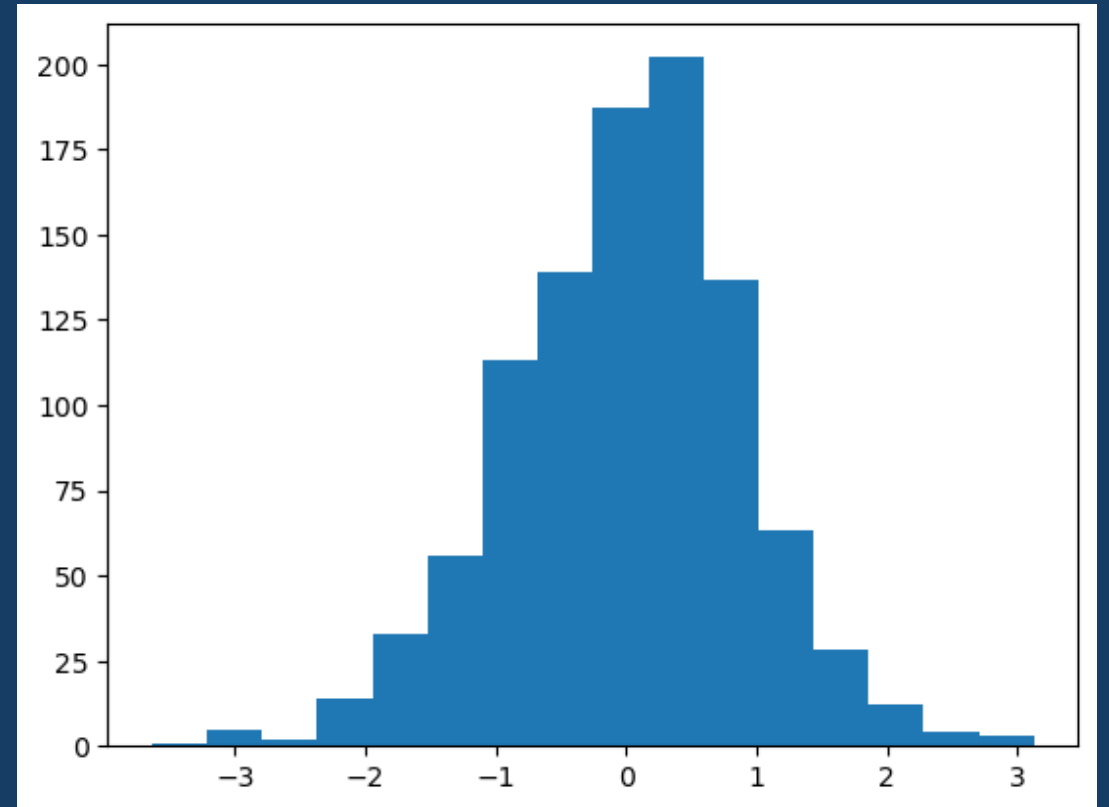
- Tried to dive into coding the whole thing but realized I need to break it down into smaller problems quickly
- Started with one step, one dimensional

One Step, One Dimensional

Before Step



After Step

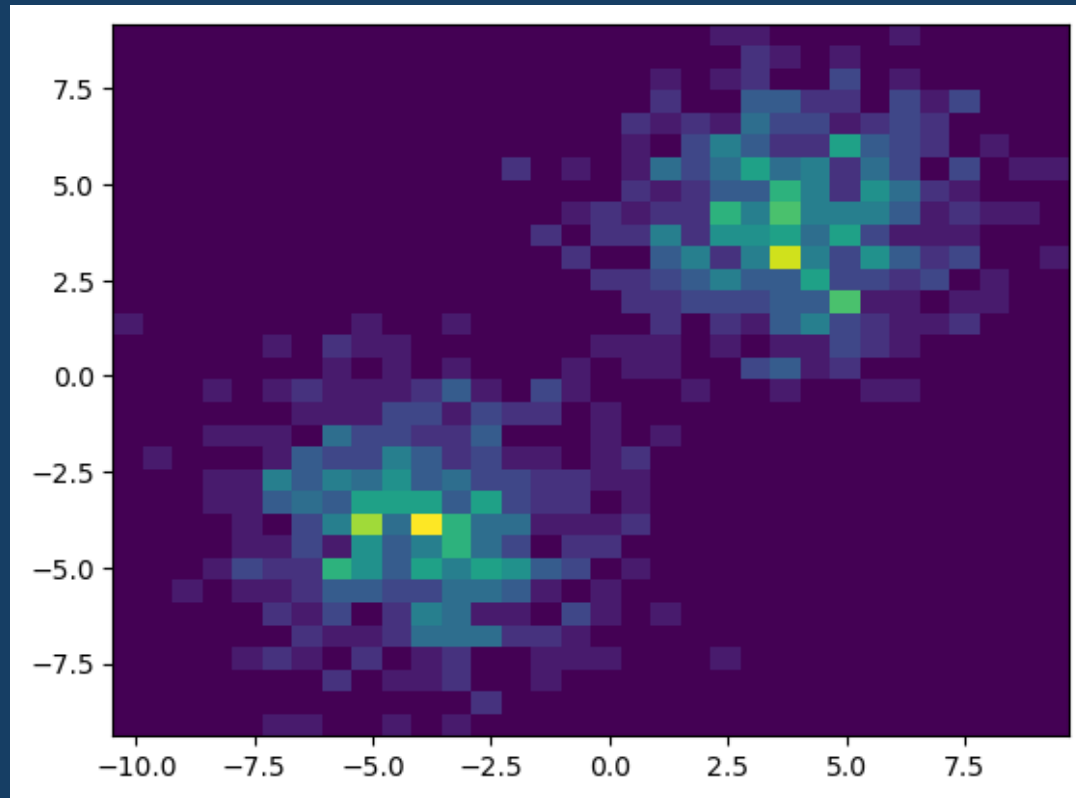


Process

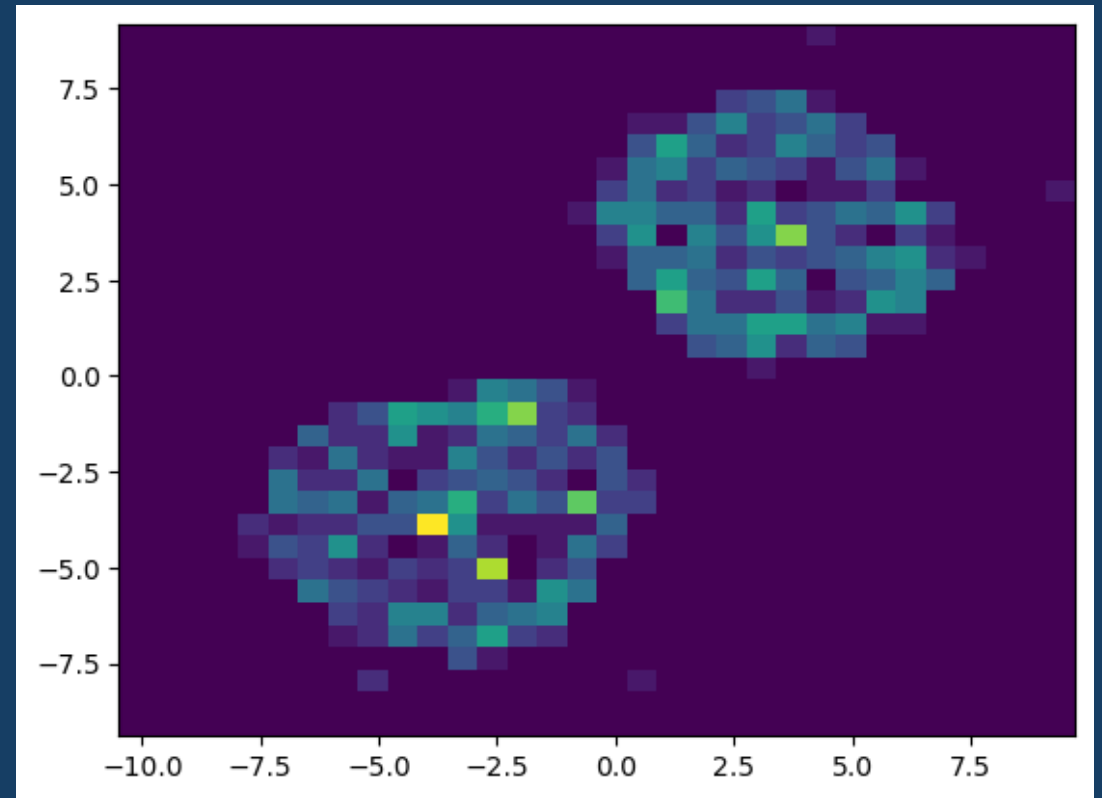
- Tried to dive into coding the whole thing but realized I need to break it down into smaller problems quickly
- Started with one step, one dimensional
- Went back to two dimensional, still one step

One Step, Two Dimensional

Before Step



After Step

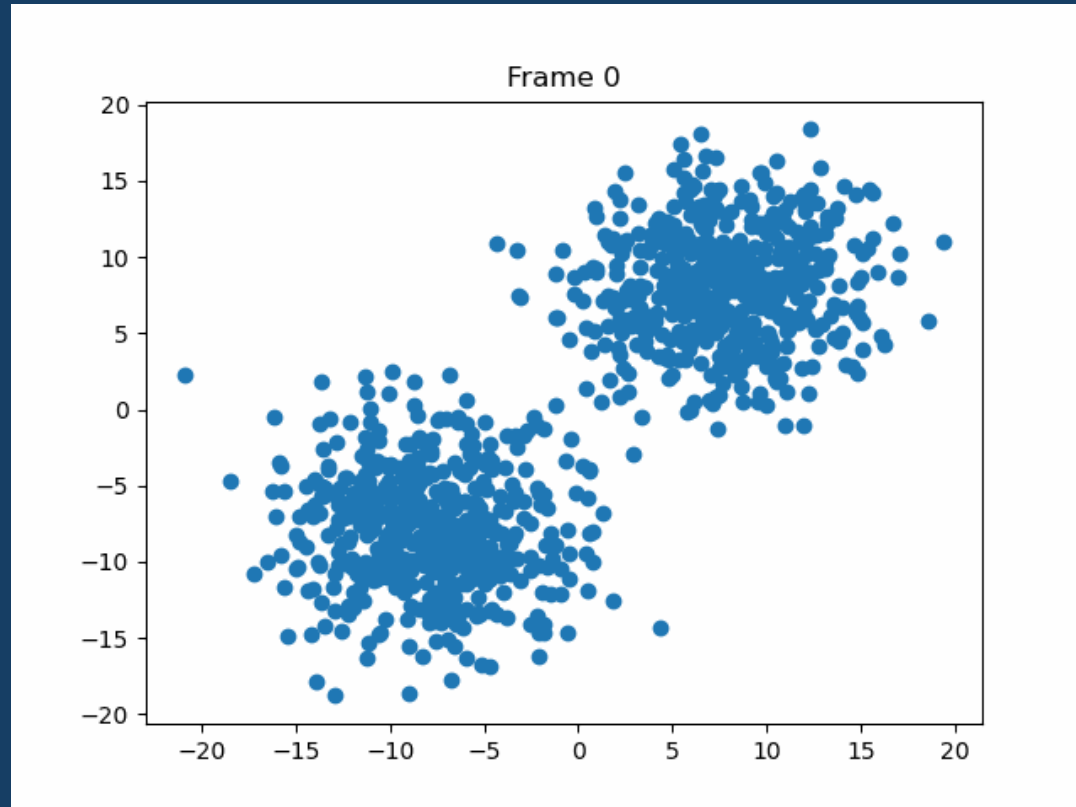


Process

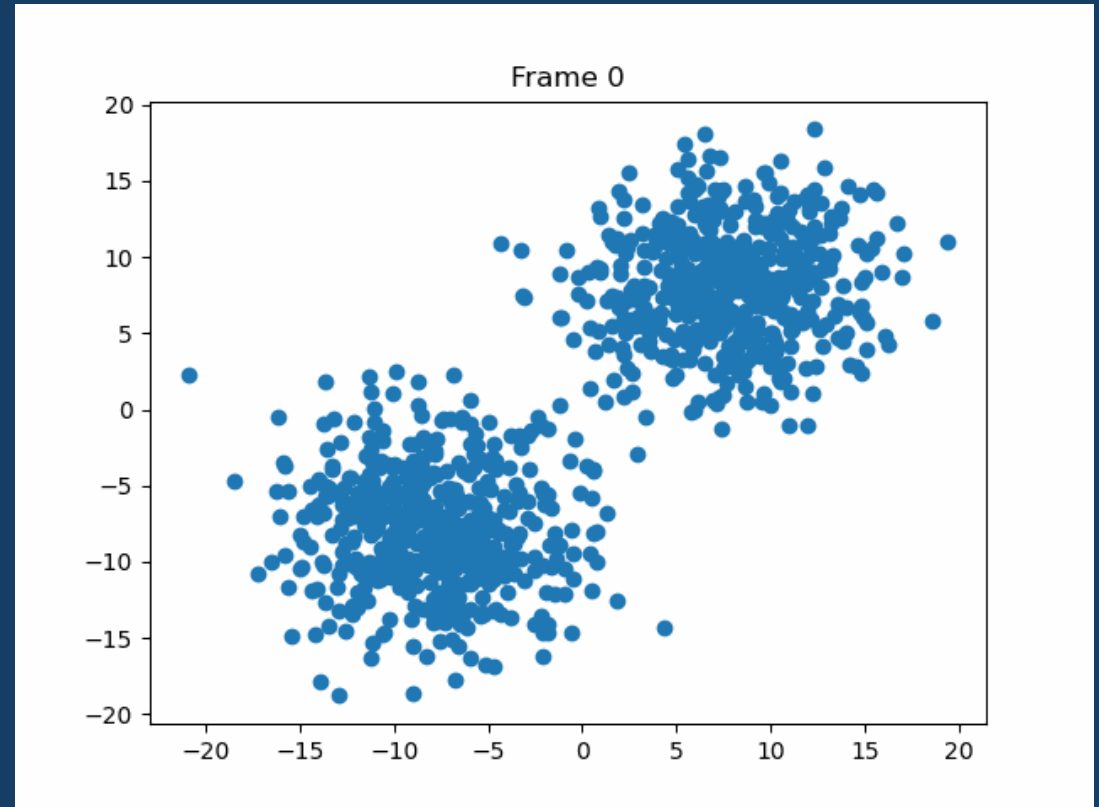
- Tried to dive into coding the whole thing but realized I need to break it down into smaller problems quickly
- Started with one step, one dimensional
- Went back to two dimensional, still one step
- Ready to iterate in two dimensions

Iterative Two Dimensional

Five Steps, $dt=0.1$



Twenty Steps $dt=0.01$

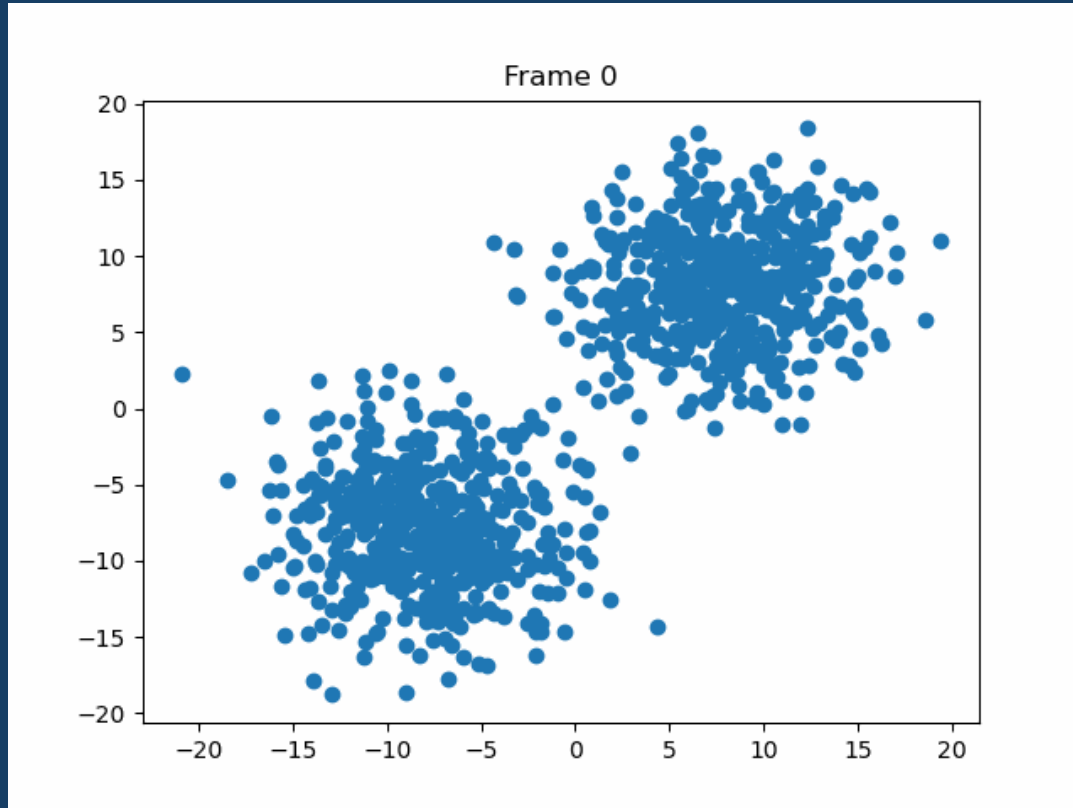


Process

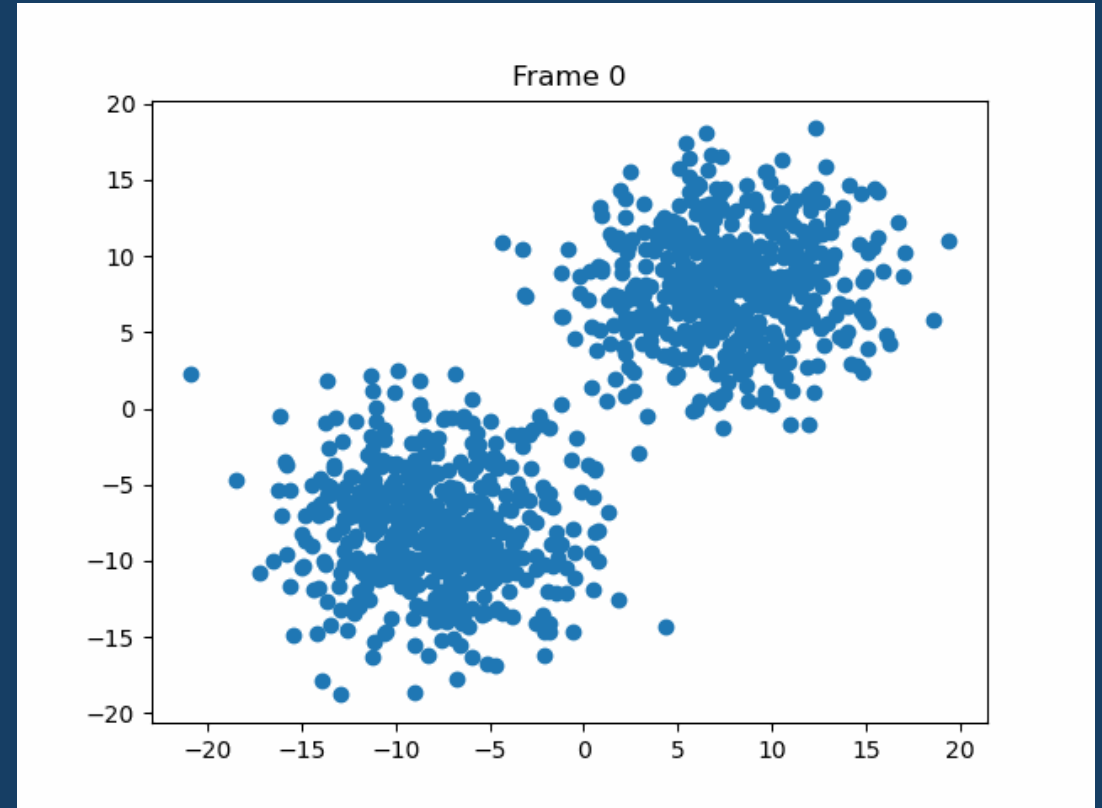
- Tried to dive into coding the whole thing but realized I need to break it down into smaller problems quickly
- Started with one step, one dimensional
- Went back to two dimensional, still one step
- Ready to iterate in two dimensions
- Improved velocity updates with leapfrog method

Leapfrog Two Dimensional

Five Steps, $dt=0.1$



Twenty Steps $dt=0.01$

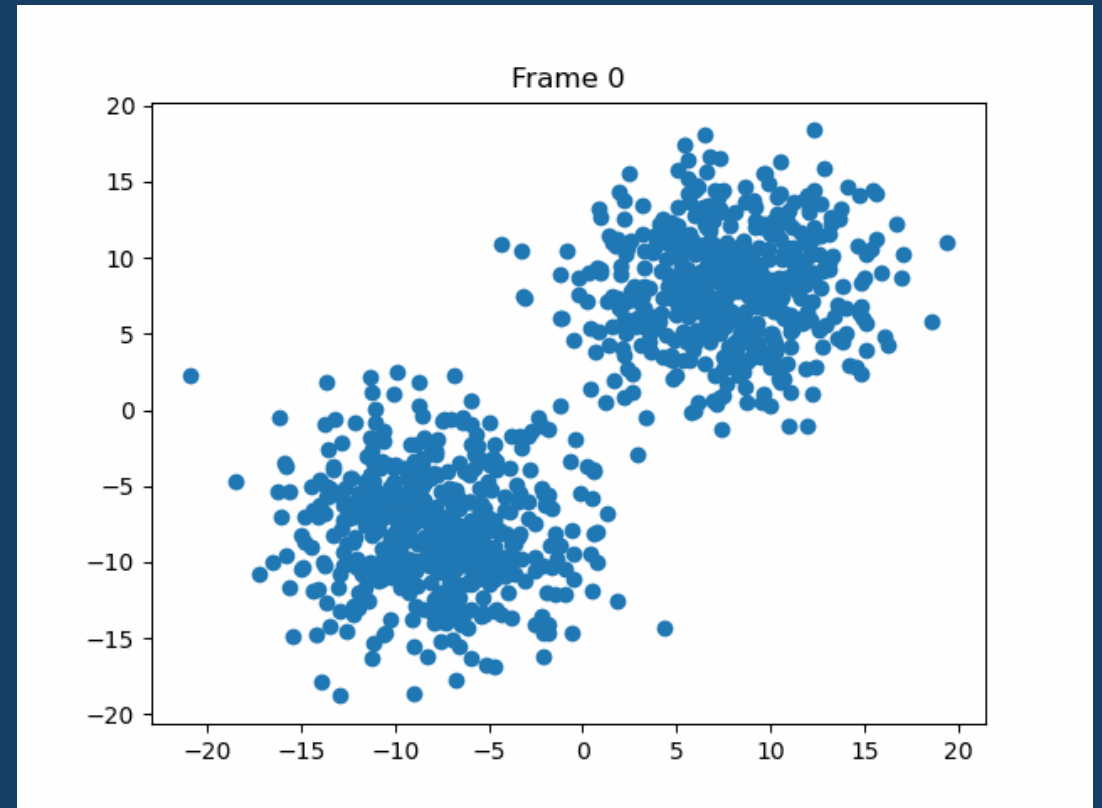
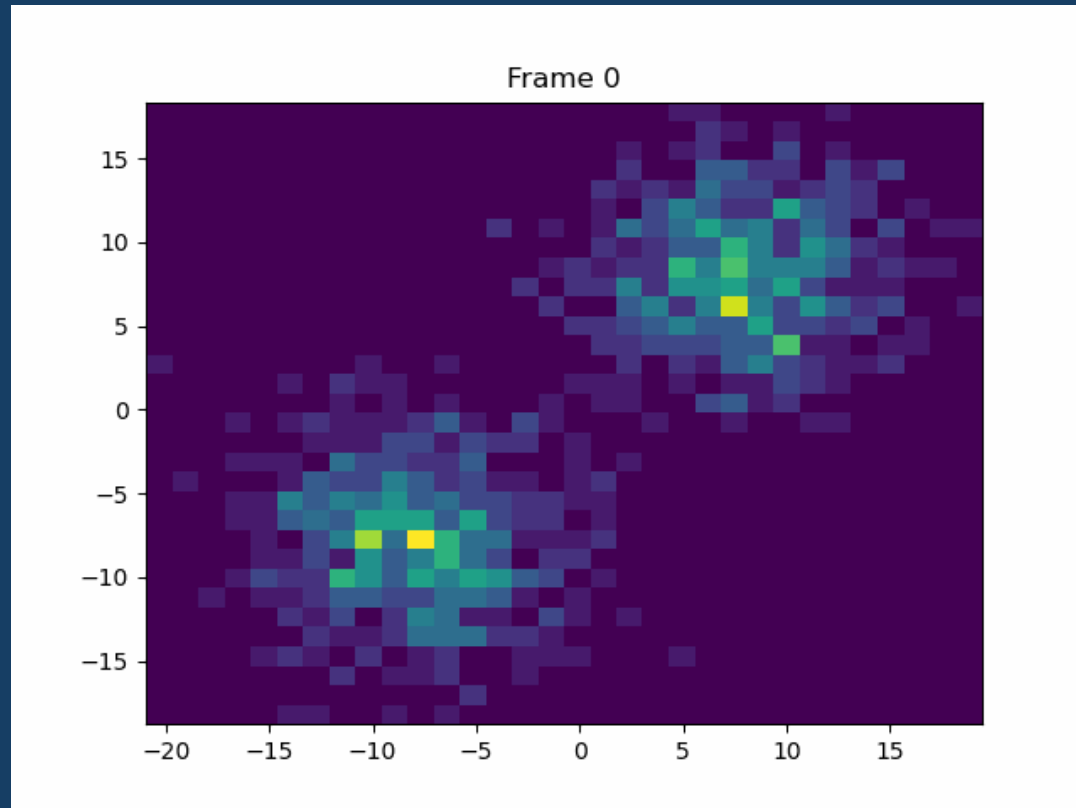


Process

- Tried to dive into coding the whole thing but realized I need to break it down into smaller problems quickly
- Started with one step, one dimensional
- Went back to two dimensional, still one step
- Ready to iterate in two dimensions
- Improved velocity updates with leapfrog method
- Made the grid resize with each iteration

Leapfrog Flex Two Dimensional

300 Steps, $dt=0.1$



Future Improvements

- Give particles initial velocity and acceleration
- Have particles of different mass
- Add particle to particle interaction within each grid square
- Vectorize loops for efficiency