Particle Interactions Puzzle

By Duncan McGough

This appears to be the Vicsek model.

In [1]:
```
import particle_interactions_puzzle as pi
```

Problem 1: Small Noise

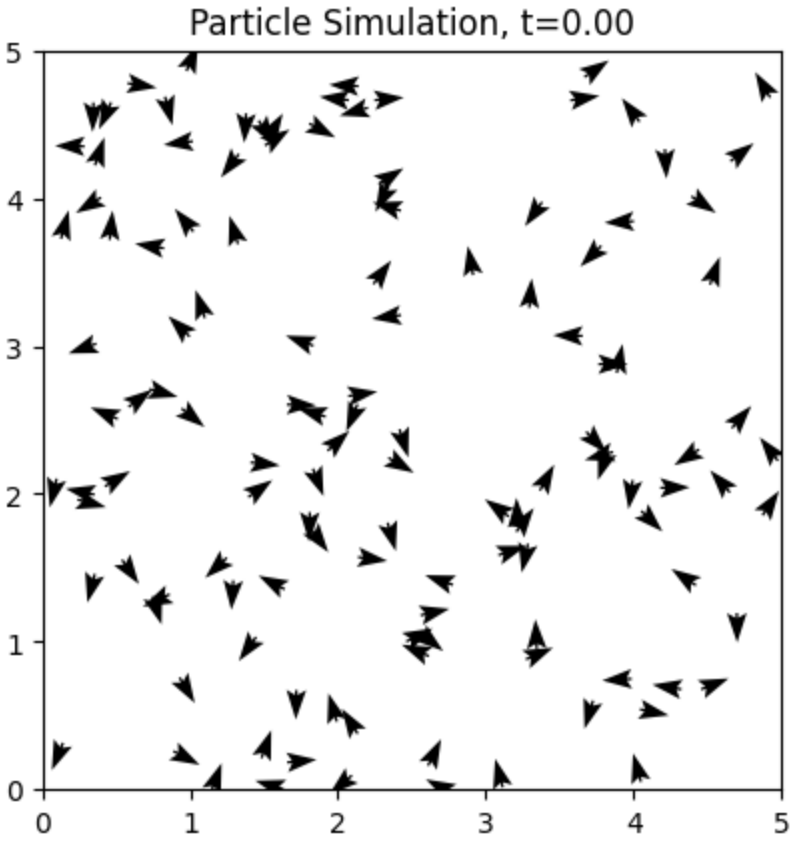
Let's set up our problem with the following parameters, using $\eta = 0.01$:

In [2]:
```
N = 125
L = 5
R = 1
v = 1
dt = 0.25

eta = 0.01

sim = pi.Simulation(N,L,eta,v,dt,R)
sim
```

Out[2]:
```
=================== Simulation ===================
Current time: 0
Particles: 125
Domain size: 5
Timestep: 0.25
Noise: 0.01
Particle speed: 1
```

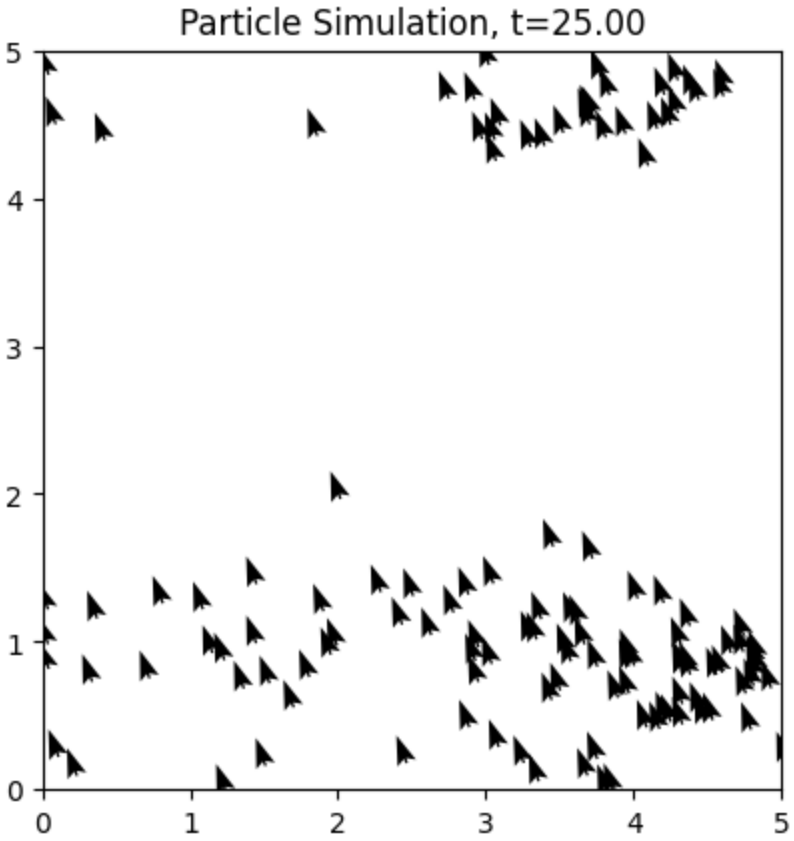
Visualizing the initial flow field:

In [3]:
```
pi.plot_simulation_timestep(sim)
```



And running out the simulation for a while, we can see that with small noise that there exists "flocking" behavior in the particle swarm:

In [4]:
```
for _ in range(100):
    sim = sim.to_timestepped()
pi.plot_simulation_timestep(sim)
```



Problem 1: Large Noise

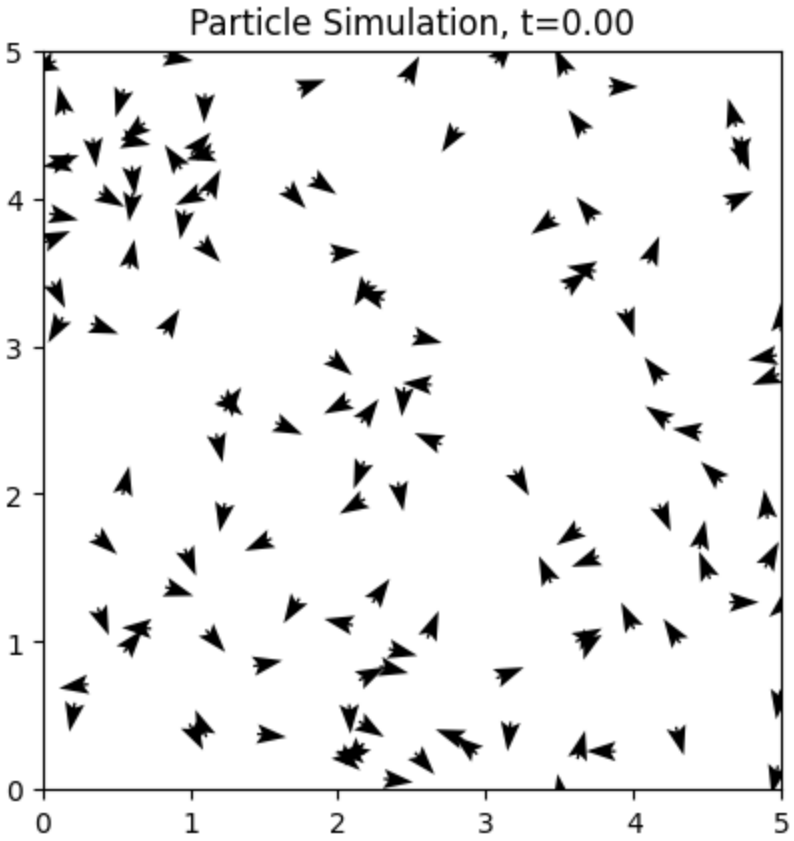
Now, let's set up the same problem now with a larger noise parameter $\eta = 0.99$:

In [5]:
```
eta = 0.99

sim = pi.Simulation(N,L,eta,v,dt,R)
sim
```

Out[5]:
```
=================== Simulation ===================
Current time: 0
Particles: 125
Domain size: 5
Timestep: 0.25
Noise: 0.99
Particle speed: 1
```

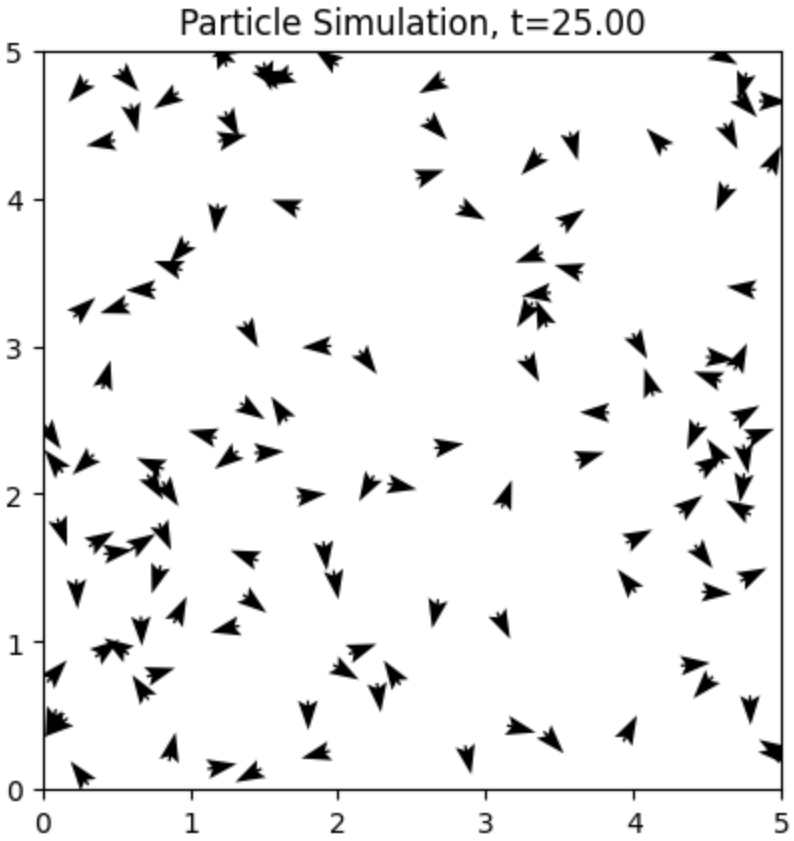
Visualizing again the initial flowfield:

In [6]:
```
pi.plot_simulation_timestep(sim)
```



And running out the simulation for a while, we can see that with large noise the flocking is no longer seen, with consistent more random motion:

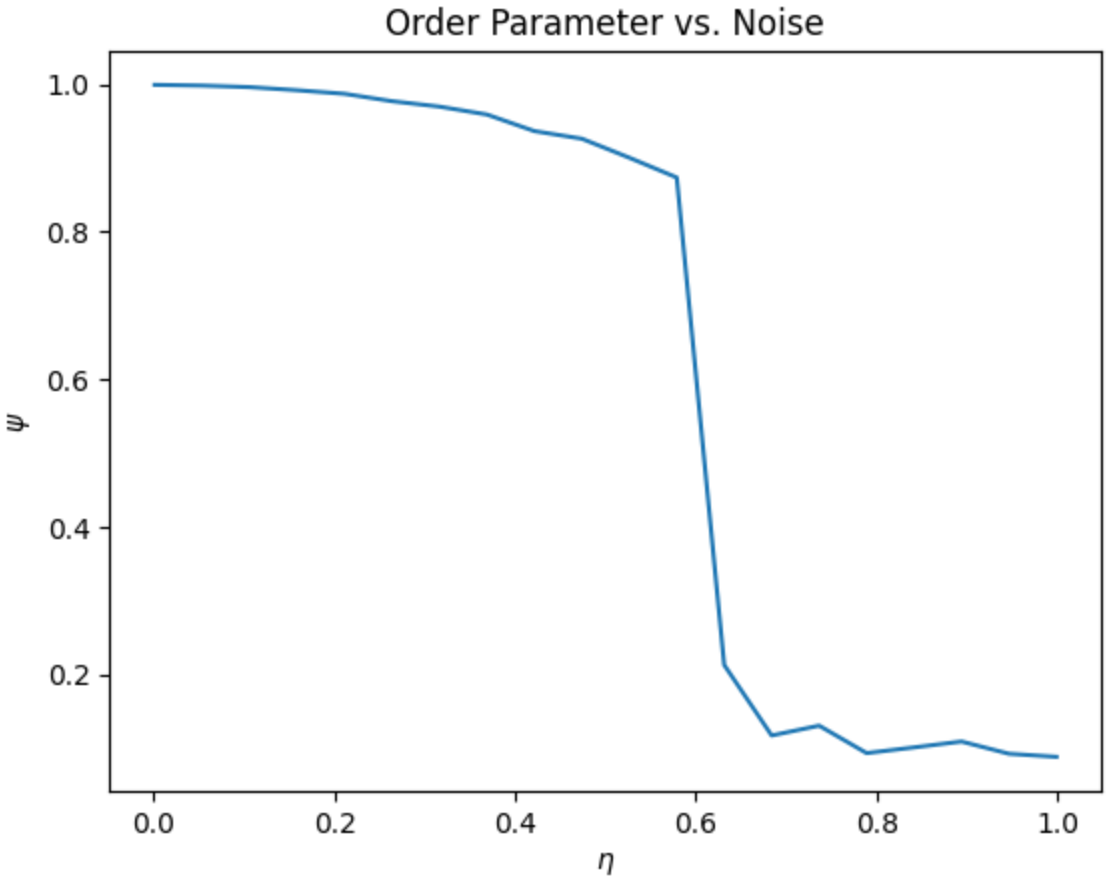
In [7]:
```
for _ in range(100):
    sim = sim.to_timestepped()
pi.plot_simulation_timestep(sim)
```



Problem 2: Polarization and Order Parameters

Plotting the stationary order parameter, we can clearly see the sharp transition when the particle swarm swaps between ordered flow and unordered flow, sweeping on noise:

In [8]:
```
pi.plot_stationary_order_parameter(N,L,R,v,dt)
```



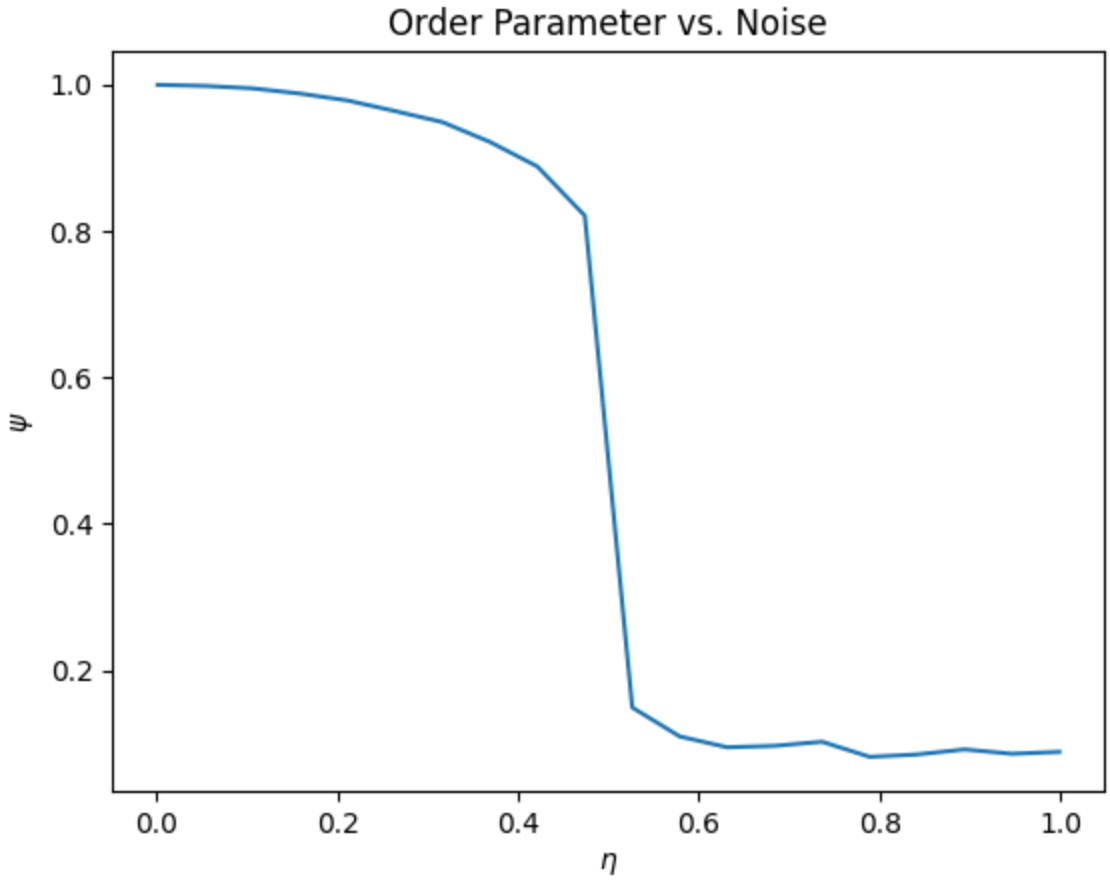
Problem 3

Now, considering η such that a critical η_c exists where phase transition occurs, we can leverage minimizers to iteratively drive a residual to zero where $\eta_c = 0.5$:

In [9]:
```
nc = 0.5
R_opt, v_opt = pi.optimize_for_critical_noise(N, L, dt, nc)
```

Plotting this using the optimized values, we can see it indeed lands on $\eta_c = 0.5$:

In [10]:
```
pi.plot_stationary_order_parameter(N,L,R_opt,v_opt,dt)
```


Order Parameter vs. Noise
1.0
0.8
0.6
0.4
0.2
ψ
0.0
0.2
0.4
0.6
0.8
1.0
η