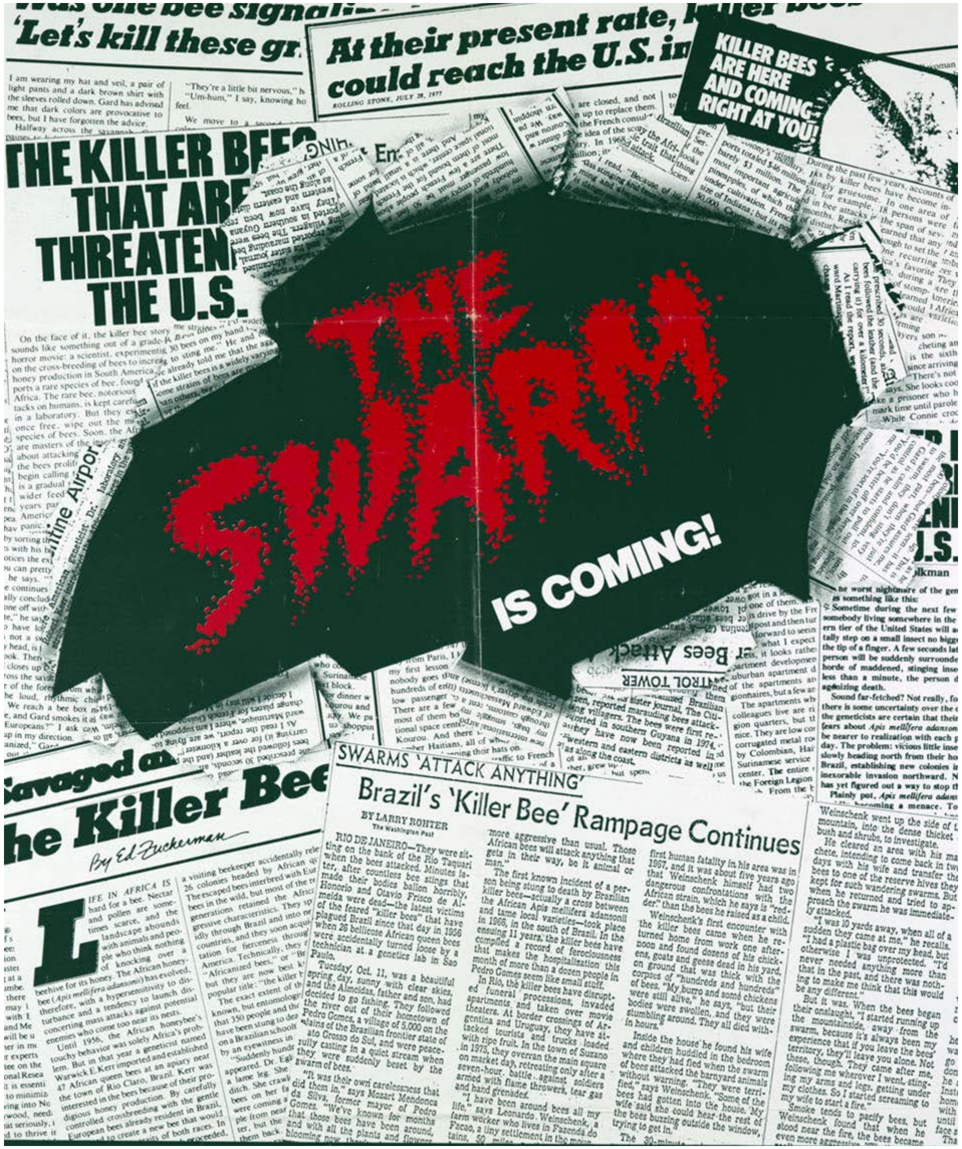
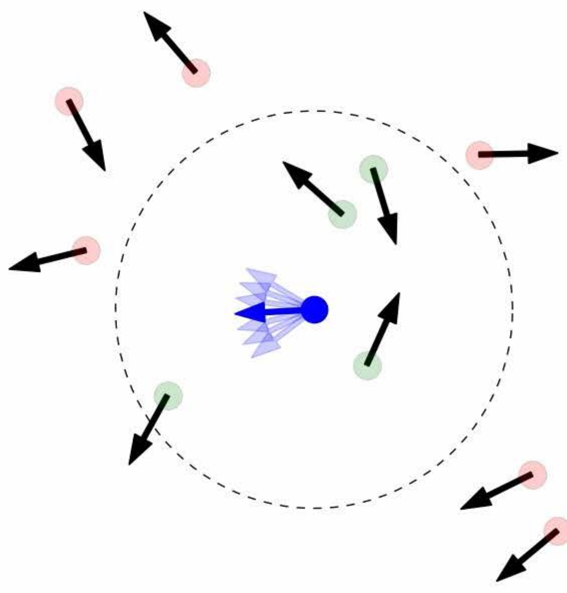
Computer simulations of complex systems



Lab PS I – The Swarm

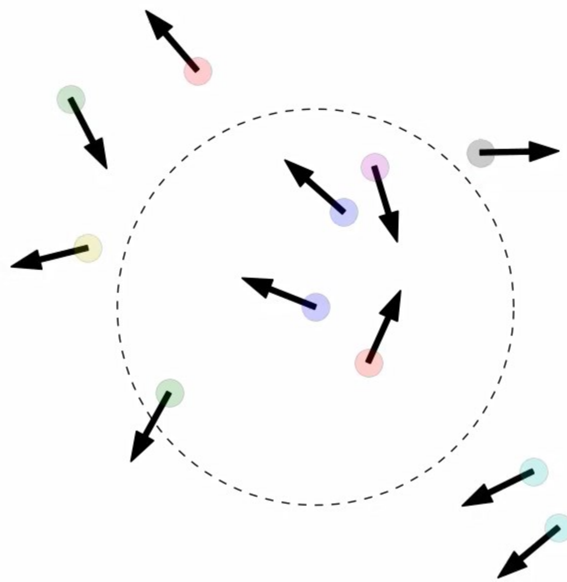
A simple swarming model



Vicsek, T., Czirók, A., Ben-Jacob, E., Cohen, I. and Shochet, O., 1995. Novel type of phase transition in a system of self-driven particles. *Physical review letters*, *75*(6), p.1226.

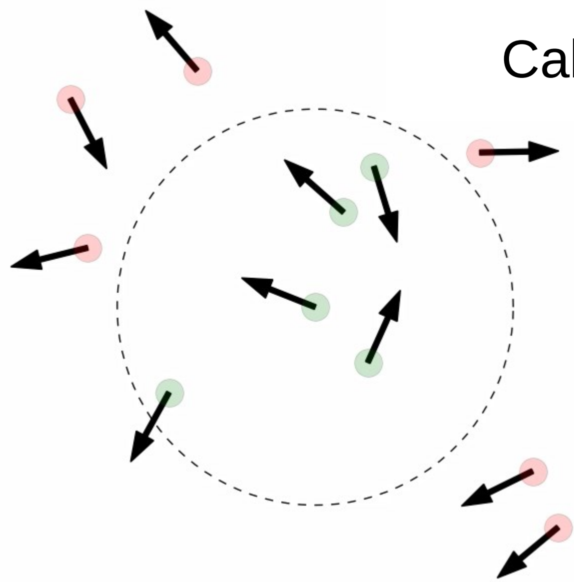
Step I

Find nearest neighbours



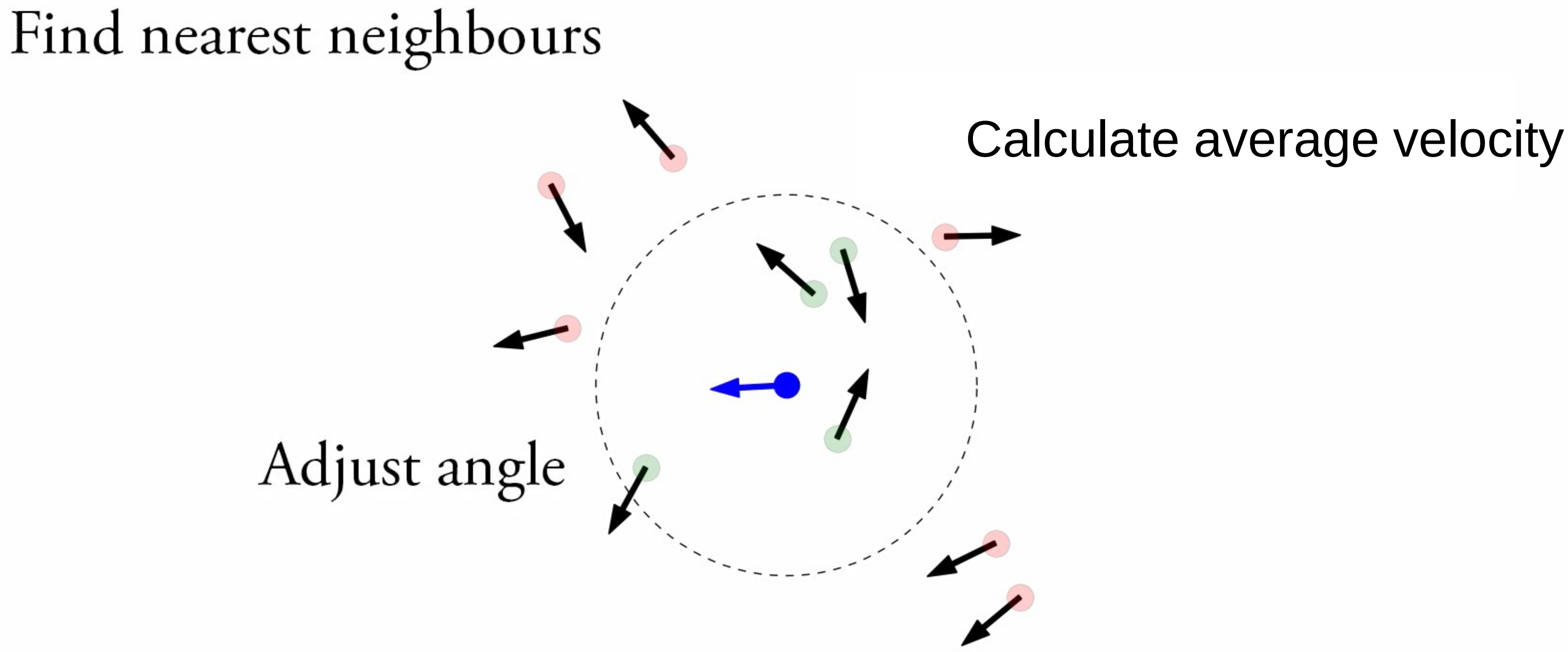
Step II

Find nearest neighbours

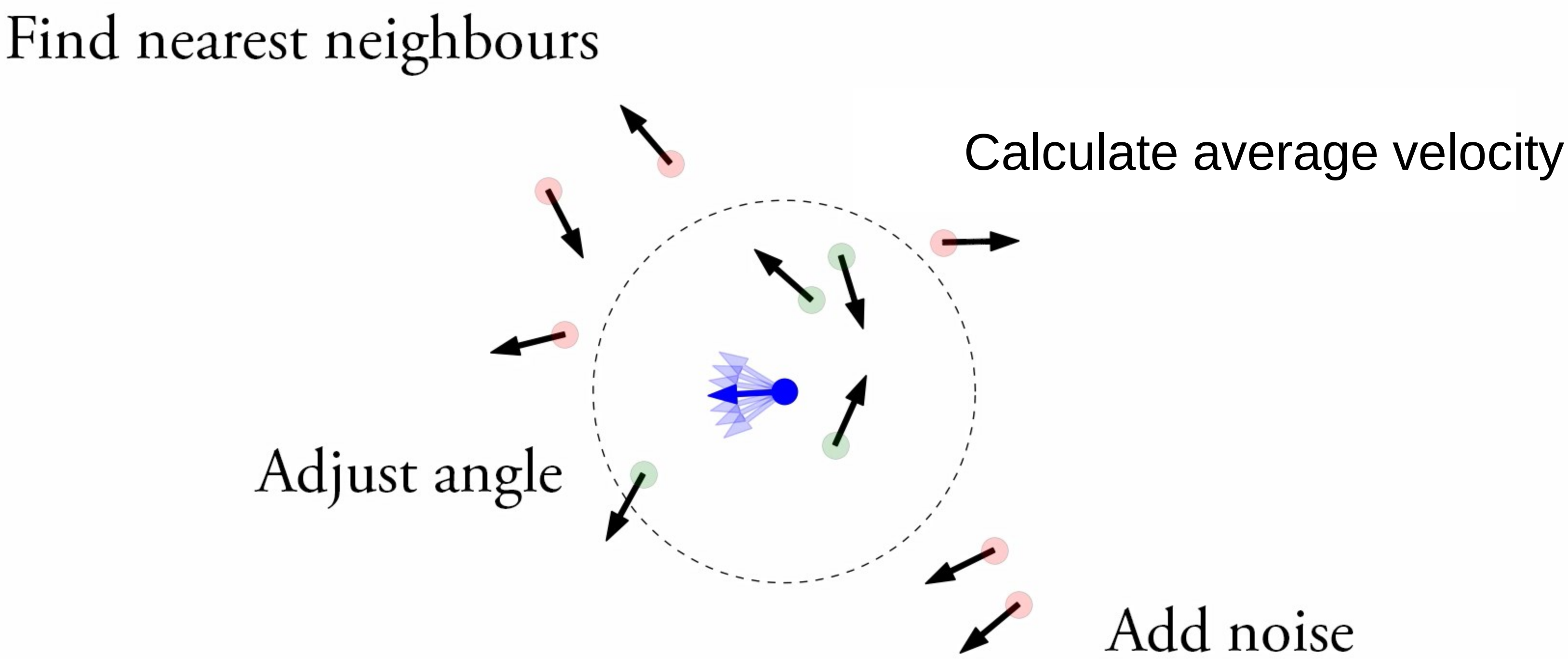


Calculate average velocity

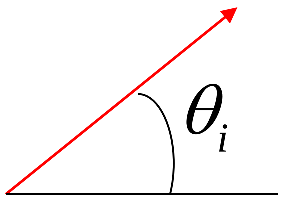
Step III


Find nearest neighbours
Calculate average velocity
Adjust angle

Step IV


Find nearest neighbours
Calculate average velocity
Adjust angle
Add noise

Summary



amplitude

new direction of bird *i*

$$\theta_i = \arg\left(\sum_k e^{i\theta_k}\right) + a\eta_i$$

direction of average velocity of the neighbours

random term

(uniform in the angle)

new position of bird *i*

$$r_i(t+dt) = r_i(t) + v_0 \begin{pmatrix} \cos\theta_i \\ \sin\theta_i \end{pmatrix} dt$$

Your task

- Simulate swarming model for 5000 birds (0.5p)
- Add a bird of prey (0.5p)

- follows the closest bird
- all birds in a radius r_b run away from it

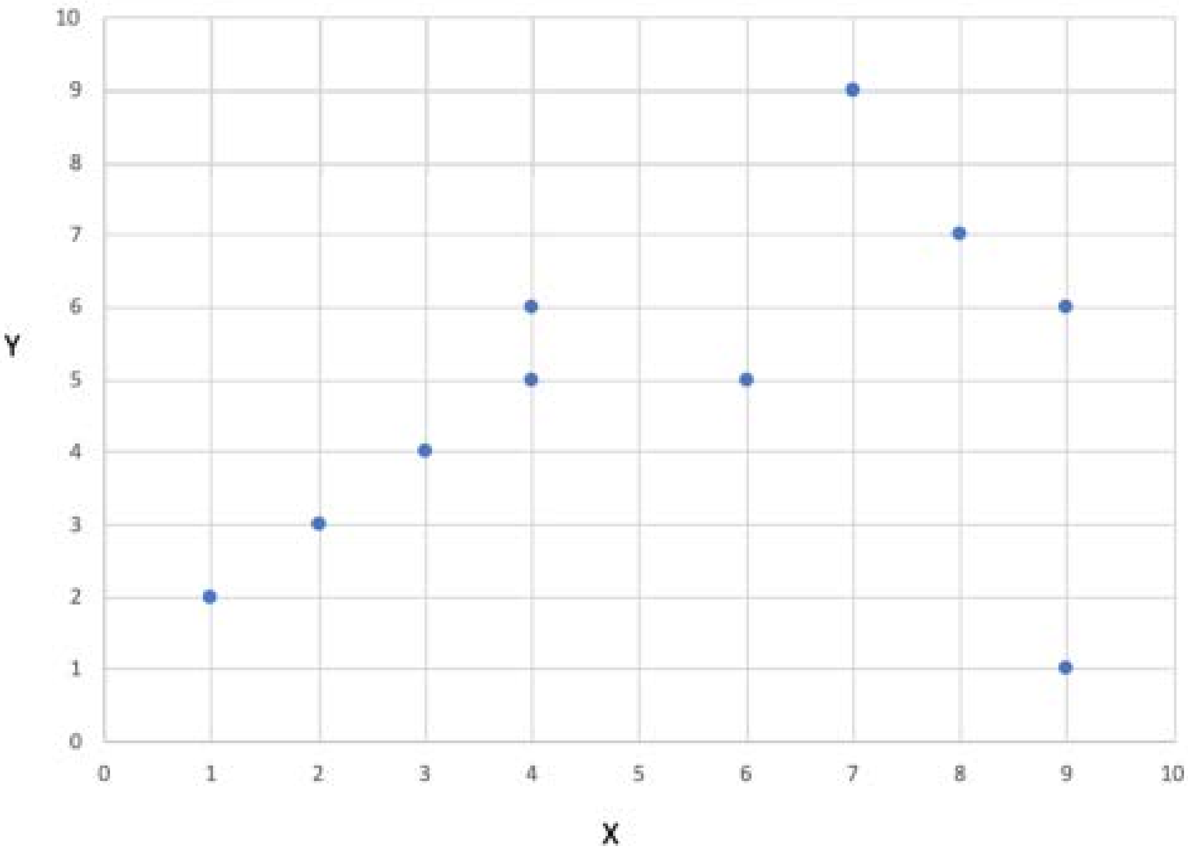
The (devil in the) details

- you need a fast way of getting the neighbours of bird *i*
- try to explore tree algorithms:

https://towardsdatascience.com/tree-algorithms-explained-ball-tree-algorithm-vs-kd-tree-vs-brute-force-9746debcd940

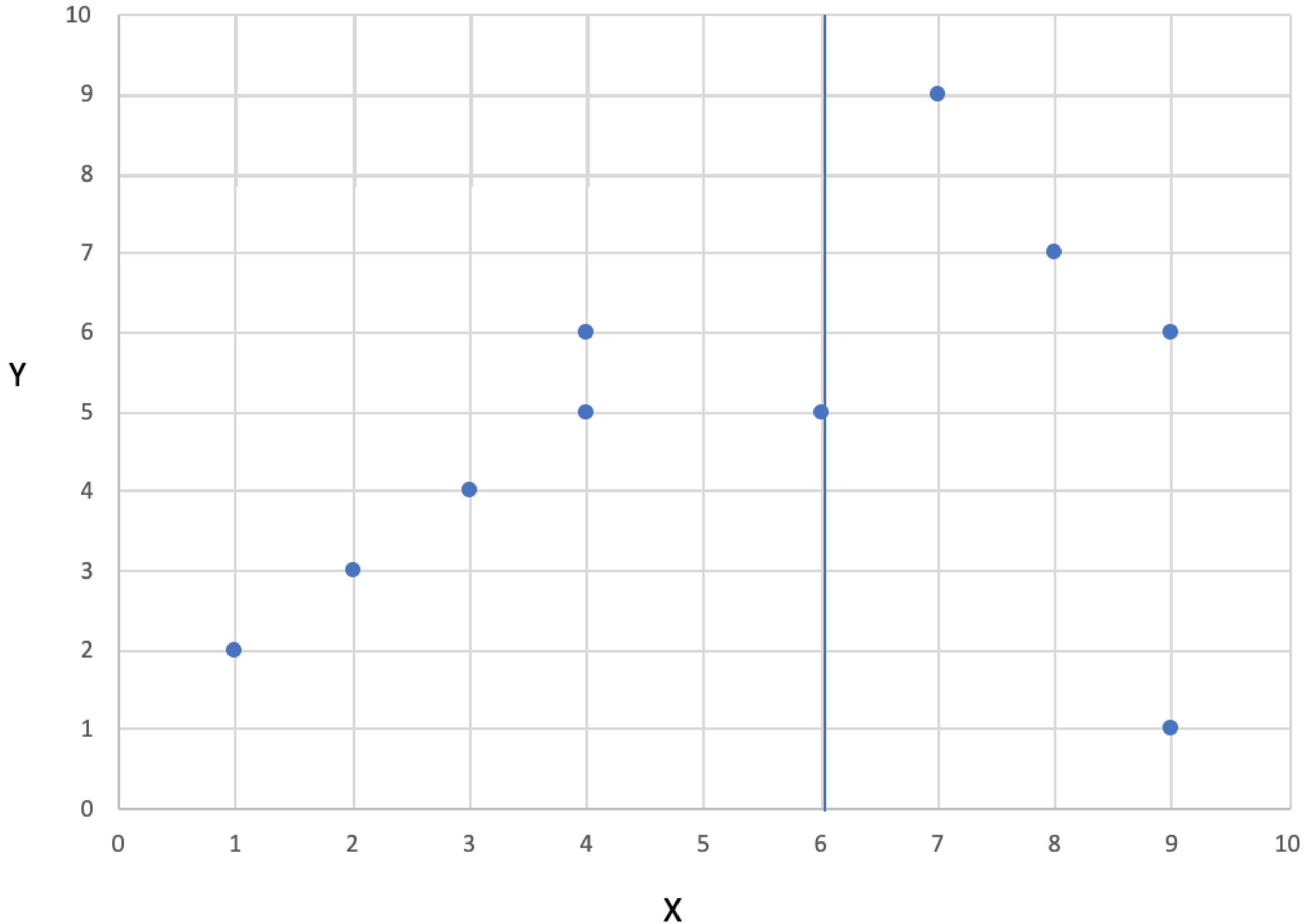
- one possibility is to use a kd-tree:

KDTree - example

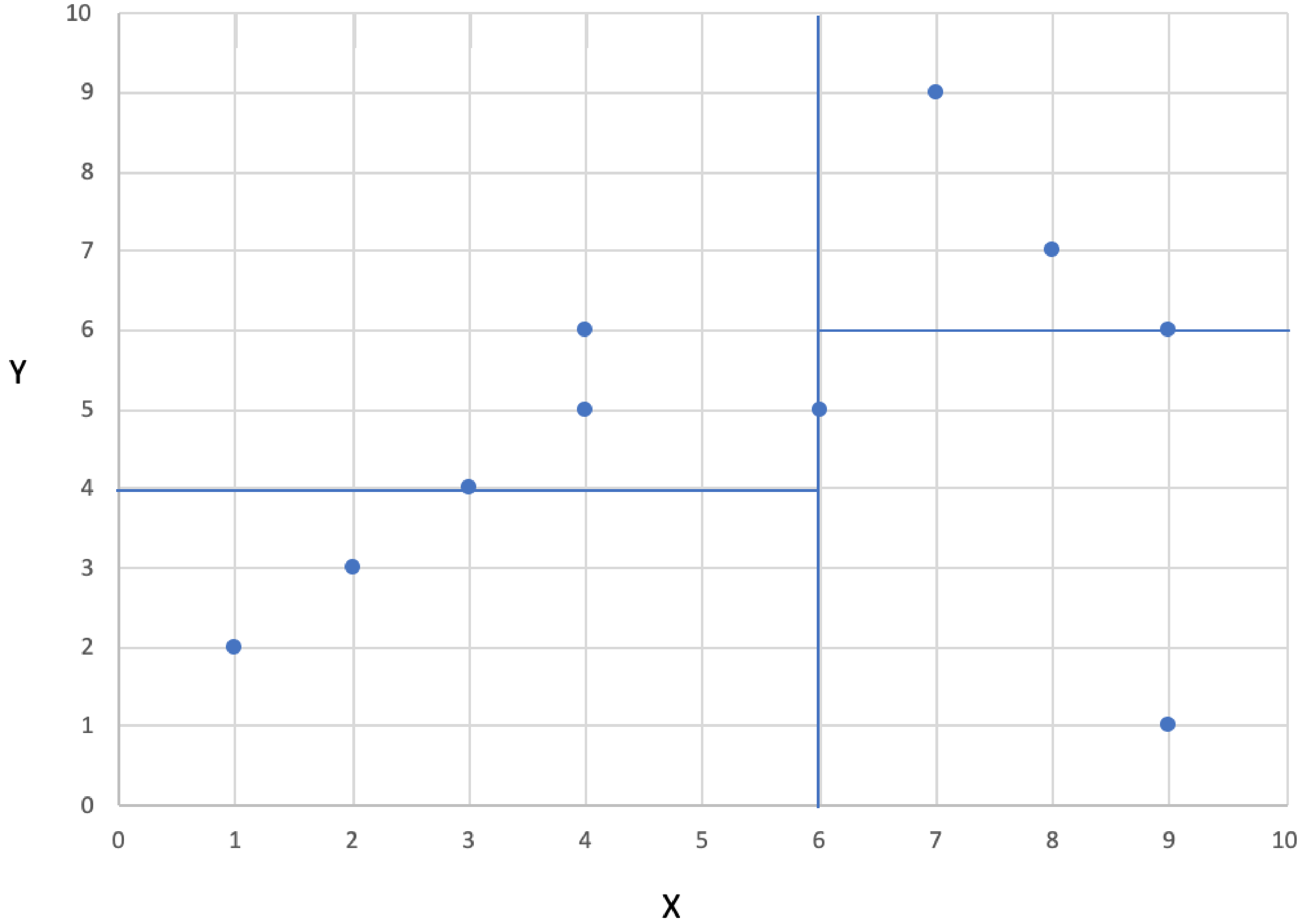


X-Werte	Y-Werte
9	1
3	4
4	6
6	5
2	3
8	7
7	9
9	6
1	2
4	5

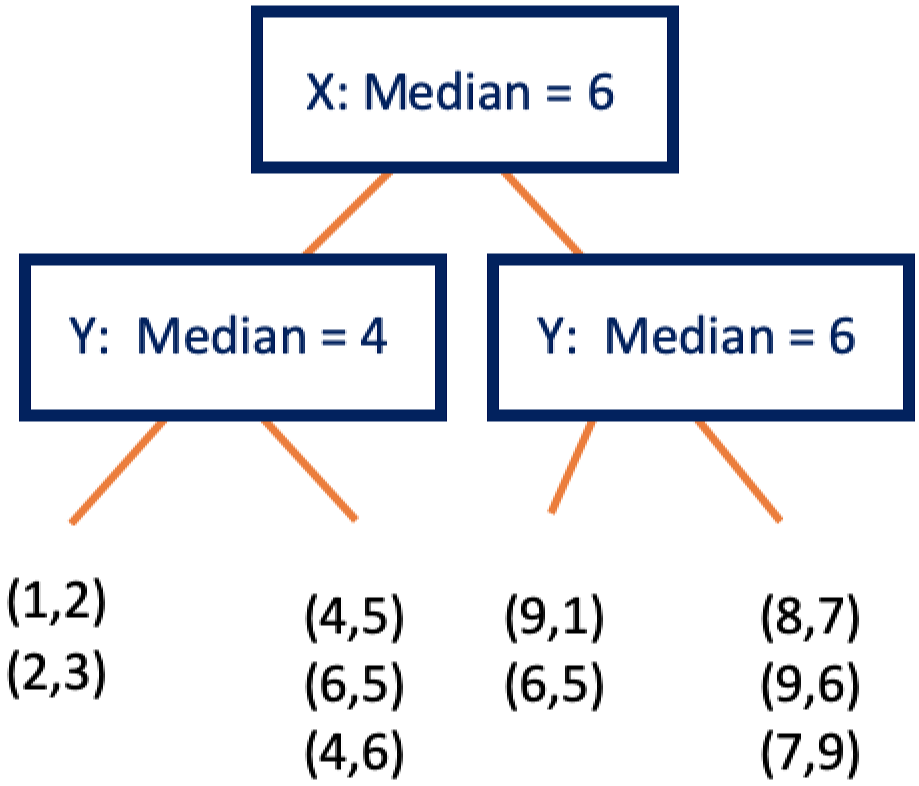
Division at the median along x



Division along y



Tree structure



Birds tree

from scipy.spatial import cKDTree

periodic boundary conditions

birds_tree = cKDTree(positions,boxsize=[L,L])

dist = tree.sparse_distance_matrix(birds_tree,max_distance=r,output_type='coo_matrix')

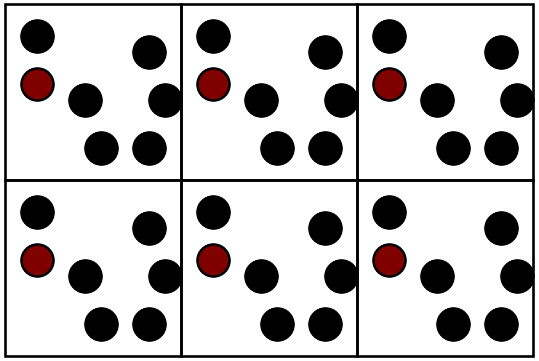
this produces a (sparse) matrix with distances
between birds (if they are smaller than r)

Other methods in cKDTree

bird_tree.query(x, k)	find k nearest neighbours to point x
bird_tree,query_ball_point(x, r)	Find all points within distance r of point(s) x.

- do not use any loops (except over time) - use numpy.sum to sum over the columns/rows of a matrix

Periodic boundary conditions

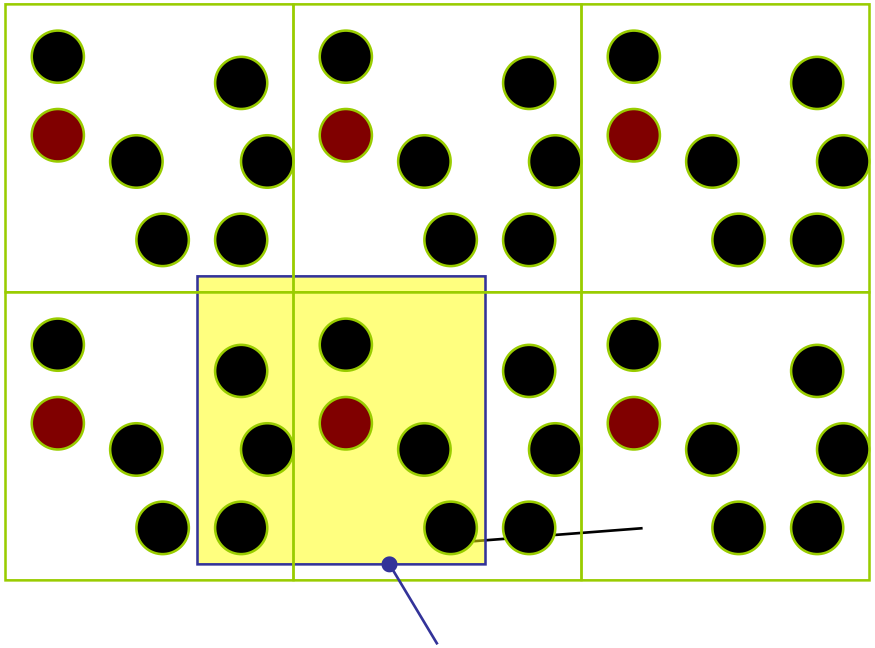


```
if newX > L:
    newX -= L
if newX < 0:
    newX += L

if newY >L:
    newY -= L
if newY < 0:
    newY += L
```

size of the box

Closest periodic image



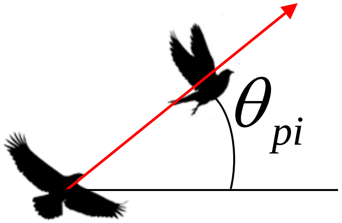
vector joining particle (i) and the closest periodic image of (j)

dr=np.remainder(ri - rj + L/2., L) - L/2

Birds

- all fly with the same velocity v_0
- their initial distribution is uniform (both in positions and in orientations)

Bird of prey



- moves with the same velocity v_0 as the birds

- follows the closest bird

$$\theta_p = \theta_{pi} + a\eta_p$$

- all the birds within a range of r_b fly away from it, ignoring other birds:

$$\theta_i = \theta_{pi} + a\eta_i$$

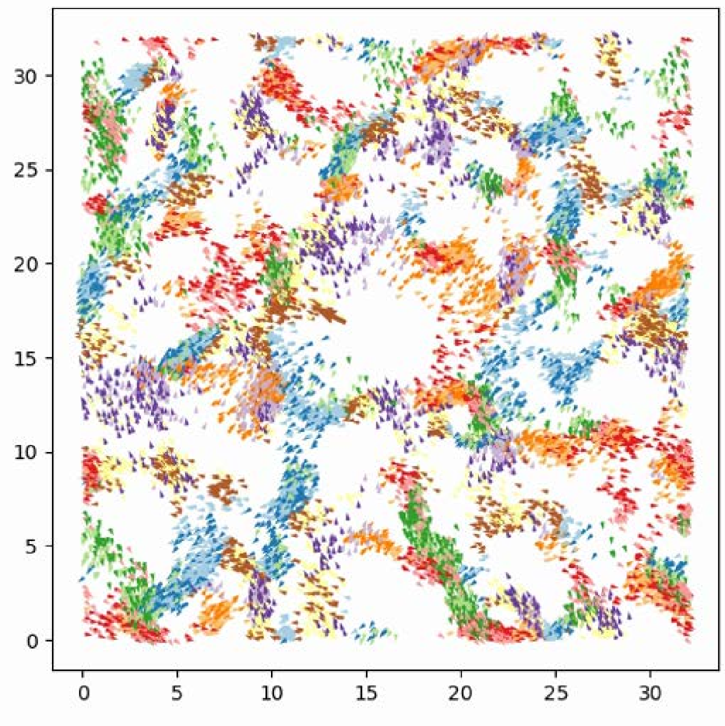
Predator interaction radius

Visualization

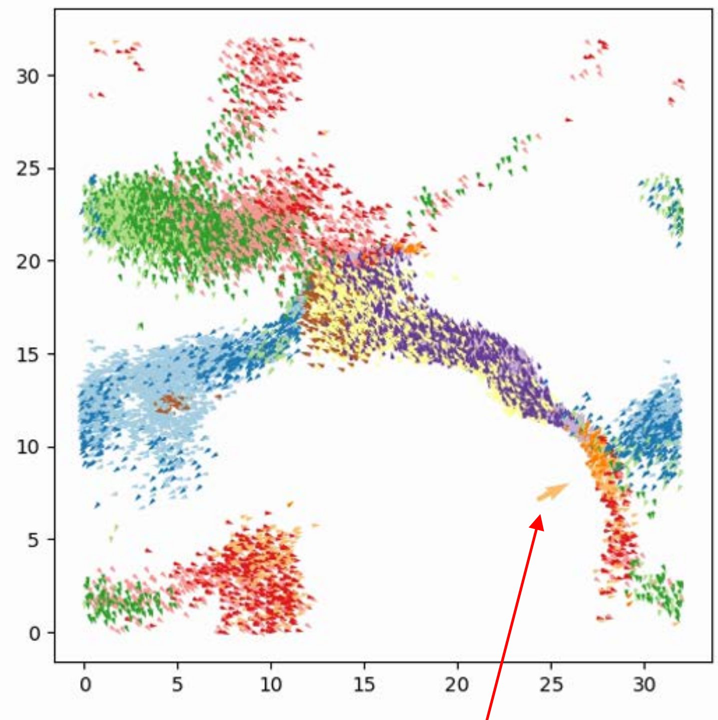
matplotlib.pyplot.quiver([X, Y], U, V, [C])

- plots a 2D field of arrows
- X, Y define the arrow locations, U, V define the arrow directions, and C optionally sets the color

use θ here



(here colors are related to orientations which helps in identifying clusters)



bird of prey

The details

- take e.g.:

  $L=32$, $N=5000\text{-}10000$, $r=1$, $r_b=4$, $v_0=2$, $a=0.15$

- make a movie of several hundred frames
- for task one (w/o bird of prey) check if the system self-organizes after sufficiently long time (all birds flying in the same direction)
- how does the presence of the bird of prey impacts such a self-organization process?

Extra task

- look at the phase transition at the intensity of a noise is changed between 0 and 1

order parameter

$$\chi = \frac{1}{N v_0} \left| \sum_i \vec{v}_i \right|$$

- plot and analyze $\chi(a)$


It's more than speculation-
It's a prediction!
THE SWARM