session3

October 21, 2024

0.1 Consigne

Le travail attendu pour ces 3 semaines est à faire dans les fichiers `session1.py`, `session2.py`, `session3.py`.

Les fichiers `session{1|2|3}.py` commenceront par un commentaire avec les noms de leur auteur :

```
# prenom1 nom1
# prenom2 nom2
```

Une attention particulière sera portée à la qualité de la documentation de votre code.

Cette semaine, il s'agit de travaillr dans session3.py

1 Session 3 -Modélisation spatialisé de cultures diversifiées

[1]:
```
import matplotlib.pyplot as plt
import session3 as project
%matplotlib inline
```

On suppose maintenant qu'on cherche à planter des plantes aléatoirement dans un champ.

2 1 Modélisation statique

Dans un premier temps, on modélise des plantes d'espèces différentes comme des des disques ayant des rayons différents. On place ces disques aléatoirement avec une loi uniforme sur la position de sorte qu'elles ne s'intersectent pas.

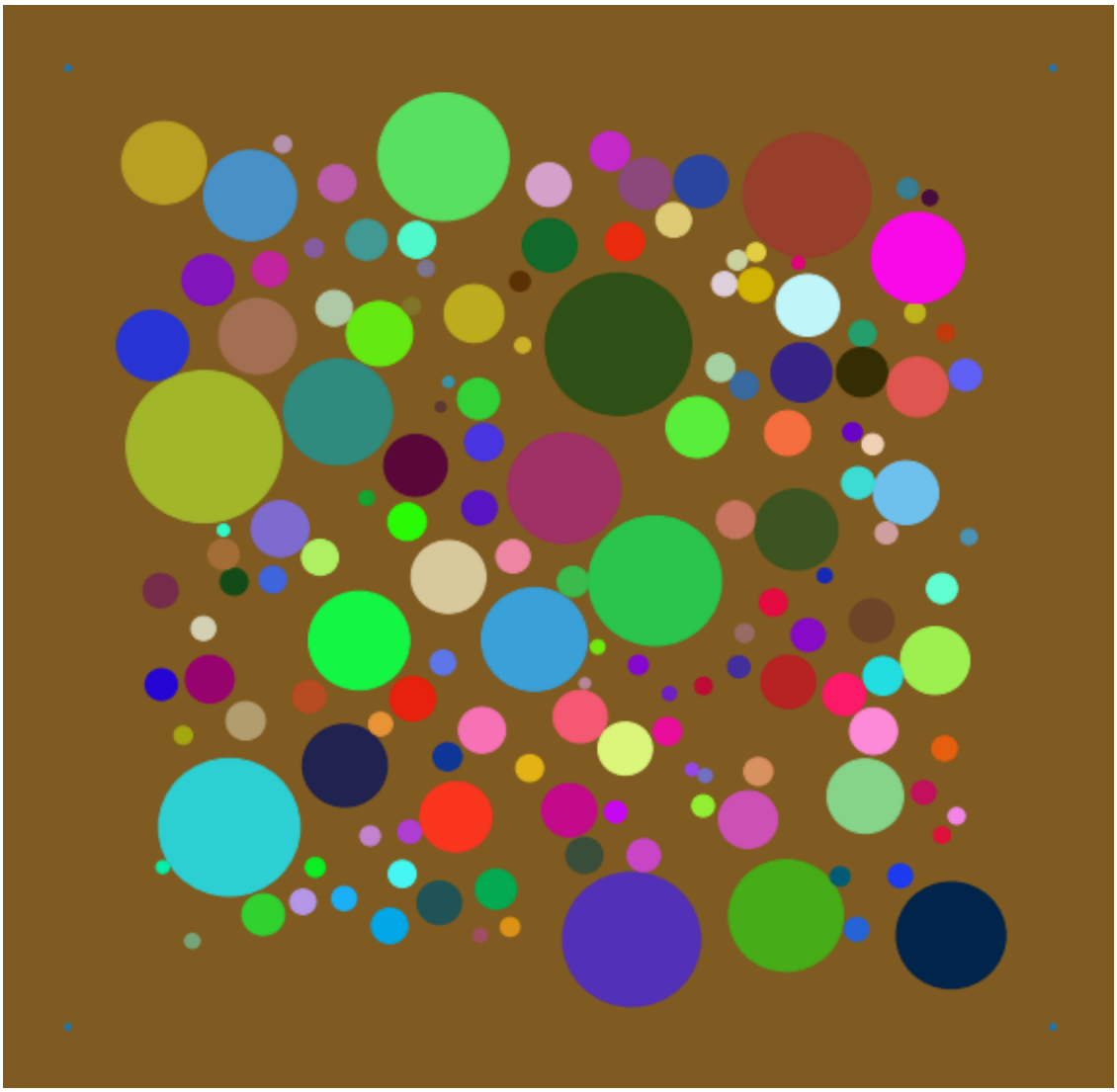
1.1 Implémenter une fonction qui renvoie une liste de N plantes plantées aléatoirement avec une loi uniforme sur un carré de longueur L. Les plantes sont représentées par des disques de rayons aléatoire entre r_{min} et r_{max}.

[2]:
```
N = 150
L = 10
rmin = .05
rmax = .8
plants = project.multi_intercrop(N, L, rmin, rmax)
print(plants[0])
```

```
{'pos': [2.7583430426283844, 6.402924506174025], 'r': 0.5494378953583844}
```

1.2 Implémenter une fonction qui produit une représentation graphique du champs.

[3]:
```
project.fig_field(plants,L)
```



1.3 Imaginer un algorithme plus rapide que l'algorithme naïf dans le cas ou on a un seul rayon(On pourra utiliser un KD tree pour chercher les Ns plus proches voisins). Comparer les vitesses d'execution des deux fonctions.

[4]:
```
L=100
N=800
r=.4

t, plants = project.monocrop(N,L,r)
t_KD, plants_KD = project.monocrop_KD(N,L,r)
print("Algo naïf: %s, Algo KD tree: %s"%(t,t_KD))
```

```
project.fig_field(plants,L)
project.fig_field(plants_KD, L)
```

```
1 0
2 1
3 1
4 1
5 1
6 1
7 1
8 1
9 1
10 1
11 1
12 1
13 1
14 1
15 1
16 1
17 1
18 1
19 1
20 1
21 1
22 1
23 1
24 1
25 1
26 1
27 1
28 1
29 1
30 1
31 1
32 1
33 1
34 1
35 1
36 1
37 1
38 1
39 1
40 1
41 1
42 1
43 1
44 1
```

45 1
46 1
47 1
48 1
49 1
50 1
51 1
52 1
53 1
54 1
55 1
56 1
57 1
58 1
59 1
60 1
61 1
62 1
63 1
64 1
65 1
66 1
67 1
68 1
69 1
70 1
71 1
72 1
73 1
74 1
75 1
76 1
77 1
78 1
79 1
80 1
81 1
82 1
83 1
84 1
85 1
86 1
87 1
88 1
89 1
90 1
91 1
92 1

93 1
94 1
95 1
96 1
97 1
98 1
99 1
100 1
101 1
102 1
103 1
104 1
105 1
106 1
107 1
108 1
109 1
110 1
111 1
112 1
113 1
114 1
115 1
116 1
117 1
118 1
119 1
120 1
121 1
122 2
123 1
124 1
125 1
126 1
127 1
128 1
129 1
130 1
131 1
132 1
133 1
134 1
135 1
136 1
137 1
138 1
139 1
140 1

141 1
142 1
143 1
144 1
145 1
146 1
147 1
148 1
149 1
150 1
151 1
152 1
153 1
154 1
155 1
156 1
157 1
158 1
159 1
160 1
161 1
162 1
163 1
164 1
165 1
166 1
167 1
168 2
169 1
170 1
171 1
172 1
173 1
174 1
175 1
176 1
177 1
178 1
179 1
180 1
181 1
182 1
183 1
184 1
185 1
186 1
187 1
188 2

189 1
190 1
191 1
192 1
193 1
194 1
195 1
196 1
197 1
198 1
199 1
200 1
201 1
202 1
203 1
204 3
205 1
206 1
207 1
208 1
209 1
210 1
211 1
212 1
213 1
214 1
215 2
216 1
217 1
218 1
219 1
220 1
221 1
222 1
223 1
224 1
225 1
226 1
227 1
228 1
229 1
230 1
231 1
232 1
233 1
234 1
235 1
236 1

237 1
238 1
239 1
240 1
241 1
242 1
243 1
244 1
245 1
246 1
247 1
248 1
249 1
250 1
251 1
252 1
253 1
254 1
255 1
256 1
257 1
258 1
259 1
260 1
261 1
262 1
263 1
264 1
265 1
266 1
267 1
268 1
269 1
270 1
271 1
272 1
273 1
274 1
275 1
276 1
277 1
278 1
279 2
280 1
281 1
282 1
283 1
284 1

285 1
286 1
287 1
288 2
289 1
290 2
291 1
292 1
293 1
294 1
295 1
296 1
297 1
298 1
299 1
300 1
301 2
302 1
303 1
304 1
305 1
306 1
307 1
308 1
309 1
310 1
311 1
312 1
313 1
314 1
315 1
316 1
317 1
318 1
319 1
320 1
321 1
322 1
323 1
324 1
325 1
326 2
327 1
328 1
329 1
330 1
331 1
332 1

333 2
334 1
335 1
336 1
337 1
338 1
339 1
340 1
341 1
342 1
343 1
344 1
345 1
346 1
347 1
348 2
349 1
350 1
351 1
352 1
353 2
354 1
355 1
356 1
357 1
358 1
359 2
360 1
361 1
362 1
363 1
364 1
365 1
366 1
367 1
368 1
369 1
370 1
371 2
372 1
373 1
374 1
375 1
376 1
377 1
378 1
379 1
380 2

381 1
382 1
383 1
384 1
385 1
386 1
387 1
388 1
389 1
390 1
391 1
392 1
393 1
394 2
395 1
396 1
397 1
398 1
399 1
400 1
401 1
402 1
403 1
404 1
405 1
406 1
407 1
408 2
409 1
410 1
411 2
412 1
413 1
414 1
415 1
416 1
417 2
418 1
419 1
420 1
421 1
422 1
423 1
424 1
425 1
426 1
427 2
428 1

429 1
430 1
431 1
432 1
433 1
434 1
435 1
436 1
437 1
438 1
439 2
440 1
441 1
442 1
443 2
444 1
445 1
446 1
447 1
448 1
449 1
450 1
451 1
452 1
453 1
454 1
455 1
456 1
457 1
458 2
459 1
460 1
461 1
462 1
463 1
464 2
465 1
466 1
467 1
468 1
469 1
470 1
471 1
472 1
473 2
474 1
475 1
476 1

477 1
478 1
479 1
480 1
481 1
482 1
483 1
484 1
485 1
486 1
487 1
488 2
489 1
490 1
491 1
492 2
493 1
494 1
495 1
496 1
497 2
498 1
499 1
500 1
501 1
502 1
503 1
504 1
505 1
506 1
507 3
508 2
509 2
510 2
511 2
512 1
513 1
514 1
515 1
516 1
517 2
518 1
519 1
520 1
521 2
522 1
523 2
524 1

525 1
526 1
527 2
528 1
529 1
530 2
531 2
532 2
533 1
534 1
535 1
536 1
537 4
538 1
539 2
540 1
541 1
542 1
543 1
544 1
545 1
546 1
547 1
548 1
549 1
550 1
551 1
552 2
553 1
554 1
555 1
556 1
557 1
558 1
559 1
560 1
561 1
562 1
563 1
564 2
565 1
566 1
567 1
568 1
569 1
570 1
571 1
572 1

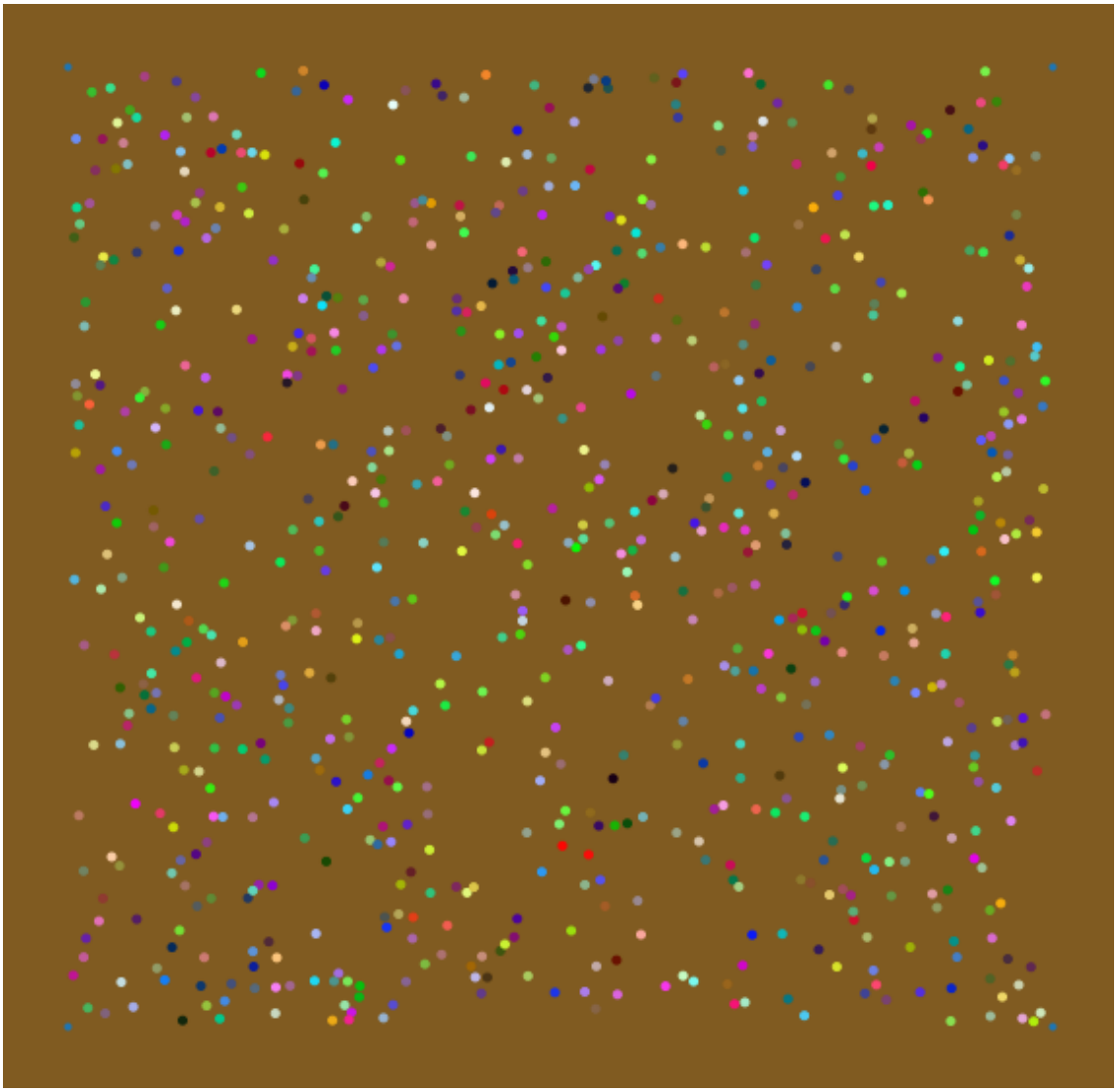
573 1
574 2
575 1
576 1
577 1
578 1
579 2
580 2
581 2
582 1
583 1
584 1
585 1
586 1
587 1
588 1
589 1
590 1
591 2
592 1
593 2
594 1
595 1
596 1
597 3
598 1
599 1
600 1
601 1
602 1
603 1
604 1
605 1
606 1
607 1
608 1
609 1
610 1
611 1
612 1
613 1
614 1
615 1
616 1
617 1
618 3
619 1
620 1

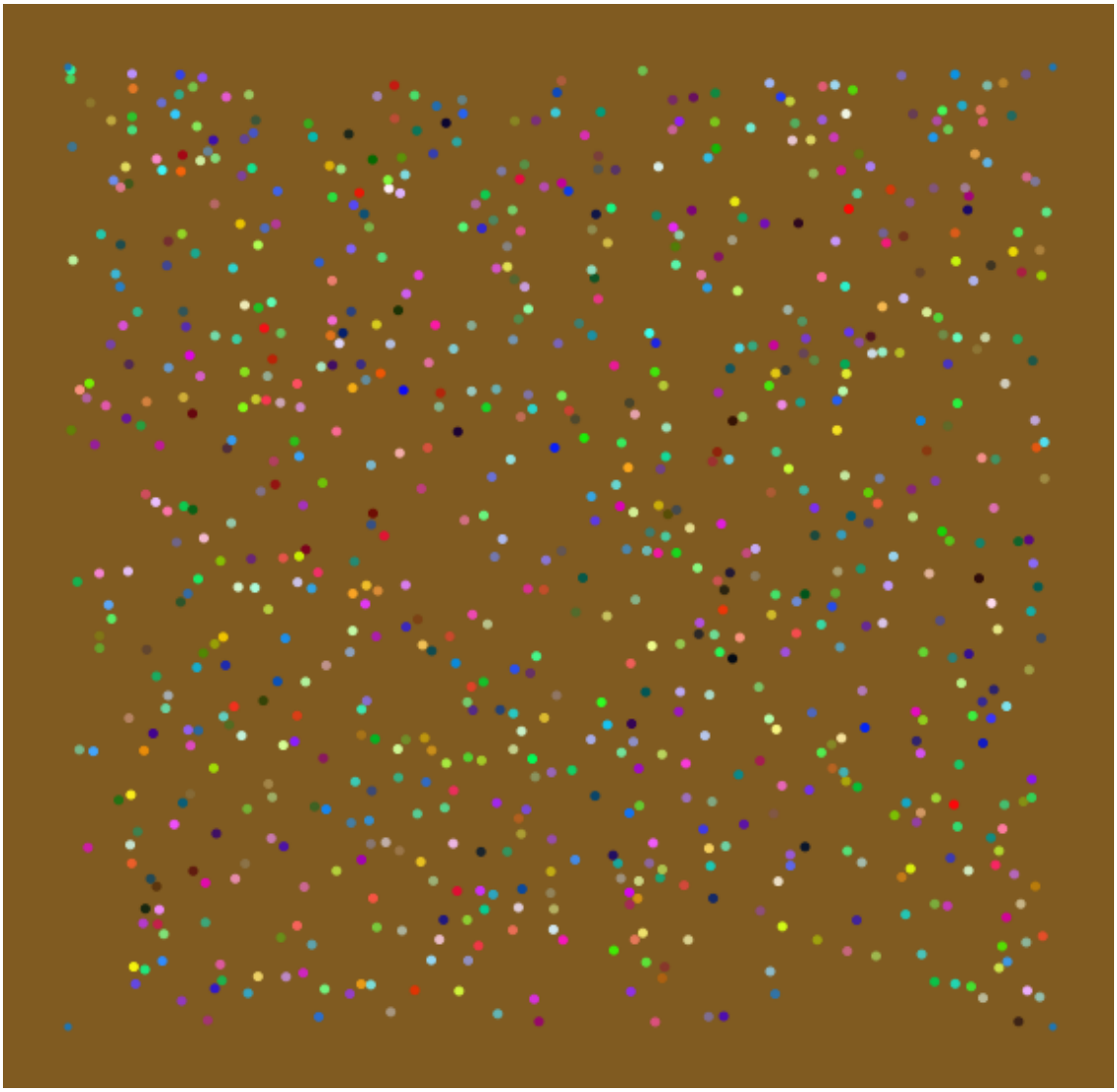
621 2
622 1
623 2
624 1
625 1
626 2
627 1
628 1
629 1
630 1
631 1
632 1
633 1
634 1
635 1
636 1
637 1
638 1
639 1
640 1
641 1
642 1
643 1
644 1
645 2
646 1
647 2
648 1
649 1
650 1
651 3
652 2
653 1
654 1
655 1
656 1
657 1
658 1
659 1
660 1
661 1
662 1
663 1
664 1
665 1
666 3
667 1
668 2

669 1
670 1
671 1
672 1
673 1
674 4
675 1
676 1
677 1
678 1
679 1
680 1
681 1
682 1
683 1
684 1
685 1
686 1
687 1
688 1
689 1
690 2
691 1
692 1
693 1
694 1
695 1
696 2
697 1
698 3
699 2
700 1
701 2
702 1
703 2
704 1
705 2
706 1
707 1
708 1
709 1
710 1
711 2
712 1
713 2
714 1
715 1
716 1

717 1
718 1
719 1
720 2
721 1
722 1
723 1
724 1
725 1
726 1
727 1
728 1
729 1
730 1
731 1
732 2
733 1
734 1
735 2
736 1
737 1
738 1
739 1
740 1
741 1
742 1
743 2
744 1
745 1
746 3
747 1
748 1
749 1
750 1
751 1
752 1
753 1
754 1
755 2
756 1
757 2
758 1
759 1
760 1
761 2
762 1
763 2
764 1

```
765 1
766 2
767 1
768 1
769 1
770 1
771 1
772 1
773 1
774 1
775 1
776 1
777 2
778 1
779 2
780 1
781 2
782 1
783 1
784 1
785 1
786 1
787 1
788 1
789 1
790 1
791 1
792 1
793 1
794 1
795 1
796 1
797 2
798 1
799 1
Algo naïf: 0.4413485527038574, Algo KD tree: 0.04983162879943848
```





1.4 Quand on considère une espèce unique (toutes les plantes ont le même rayon), quelle est la densité maximale possible de plantes en considérant une seule espèce sur un réseau triangulaire

1.5 Estimer la densité moyenne maximale qu'on peut atteindre avec un plantation aléatoire et comparer la à la densité maximale (avec une espèce)

3 2 Modélisation dynamique

On considère maintenant la dynamique de croissance des plantes, on suppose que les plantes sont plantées à des temps aléatoires suivant une loi de Poisson. Le rayon de la plante croit suivant $r(t) = \alpha t$ jusqu'à ce que les plantes soient enlevées du champs après un temps t_h. On vérifiera

avant de planter chaque plante que celle-ci n'entrera pas en collision avec ses voisines au cours de leur croissance.

2.1 Simuler la dynamique du champs. La fonction renverra une liste de dictionnaires (un par plante) contenant la position et le temps de plantation de celle-ci. Implémenter un fonction qui permet de visualiser l'état du champs au temps t (un snapshot et une animation).

[5]:
```
planting_rate=1
Rmax=.4
th=30
plants, ps = project.dynamic_random_planting(planting_rate, Rmax, th)
project.fig_dynamic(plants, "random_planting.png", "random_planting.mp4")
print(ps)
```

```
/home/kodda/Dropbox/p2pflab/kaku/L3INFO/TME_agri/all_sessions/session3.py:103:
RuntimeWarning: More than 20 figures have been opened. Figures created through
the pyplot interface (`matplotlib.pyplot.figure`) are retained until explicitly
closed and may consume too much memory. (To control this warning, see the
rcParam `figure.max_open_warning`).
  fig=pl.figure(figsize=(10,4))
IMAGEIO FFMPEG_WRITER WARNING: input image is not divisible by
macro_block_size=16, resizing from (572, 231) to (576, 240) to ensure video
compatibility with most codecs and players. To prevent resizing, make your input
image divisible by the macro_block_size or set the macro_block_size to 1
(risking incompatibility).
```


```
[{'pos': [4.668048842289379, 1.014139719211888], 't': 65}, {'pos':
[2.1664818975608275, 1.5917205716611718], 't': 69}, {'pos': [5.908851342793739,
-0.26851403690643005], 't': 69}, {'pos': [3.3409368812902107,
0.5624180750079493], 't': 69}, {'pos': [2.7744207036029223, 1.117325175956231],
't': 70}, {'pos': [3.5261671365763285, 0.1519857571392902], 't': 72}, {'pos':
[8.779162420568102, 0.7319397795242479], 't': 72}, {'pos': [2.4914027264544703,
-1.3268745111409428], 't': 74}, {'pos': [7.386074777821885, 1.5723004133950549],
't': 74}, {'pos': [1.1151562383304183, 1.196253128693074], 't': 75}, {'pos':
[4.338517512739314, 1.4247063863938485], 't': 75}, {'pos': [7.733674833854234,
-1.5094280181321844], 't': 75}, {'pos': [1.6772722172993255,
0.7784589752882152], 't': 76}, {'pos': [5.169151083438382, 0.4476232564203322],
't': 77}, {'pos': [9.329932414175262, -1.277682208486419], 't': 77}, {'pos':
[0.49978271247025197, -1.09338354019558], 't': 77}, {'pos': [4.971401166784223,
-0.47756542973693183], 't': 77}, {'pos': [6.6108415138777685,
1.0705635114051644], 't': 80}, {'pos': [0.7451610725043711, 1.5265802994385025],
't': 80}, {'pos': [1.2713437959497864, -0.09944967009812333], 't': 82}, {'pos':
[3.7432292381183374, 1.4241992277275952], 't': 82}, {'pos': [3.0061853694055984,
0.18992514725964038], 't': 82}, {'pos': [0.8338331107825068,
0.7136325006058319], 't': 82}, {'pos': [1.9886242093252444,
-1.0465406451428372], 't': 82}, {'pos': [7.984546941368776, 0.8042622138224429],
't': 83}, {'pos': [5.418372203328483, 0.11537849234310116], 't': 84}, {'pos':
[2.933978606208198, 0.6828620693456293], 't': 84}, {'pos': [6.097669458298238,
```

0.6366298678215037], 't': 84}, {'pos': [1.170124408764405, -1.5219853779758177],
't': 85}, {'pos': [0.9243464319552304, 0.3170279219387713], 't': 85}, {'pos':
[7.770453154496921, 0.08967110223507357], 't': 85}, {'pos': [8.525445944041259,
-0.9771641739819537], 't': 89}, {'pos': [8.337398005494922,
0.21731974244821717], 't': 89}, {'pos': [1.7497731788896704,
-0.03912367942537753], 't': 89}, {'pos': [8.61335712340029,
-0.15462071784738107], 't': 91}, {'pos': [6.719877001901092,
-0.09445909241913997], 't': 92}, {'pos': [6.15553587453016,
-1.3521041711928357], 't': 92}, {'pos': [5.670265768550136, 1.4044909469891098],
't': 92}, {'pos': [4.218485701055547, 0.5352147452961331], 't': 92}, {'pos':
[2.105786851333455, -0.3278633465914045], 't': 94}, {'pos': [7.155005807880472,
-1.2610170074671712], 't': 96}, {'pos': [0.6219491542398236,
-0.6720361542992009], 't': 96}, {'pos': [4.994498164822371,
-1.4781771365999028], 't': 96}, {'pos': [3.319323798746438, 1.2504306316374656],
't': 97}, {'pos': [7.545549042281875, -0.7554825510074192], 't': 97}, {'pos':
[6.436331257953165, -0.691625946470182], 't': 97}, {'pos': [3.428755431296804,
-0.3140928040552875], 't': 97}, {'pos': [9.145196462165558,
-0.22792336002642832], 't': 98}, {'pos': [8.533945320283612,
-1.4409804173862328], 't': 98}]

<Figure size 720x288 with 0 Axes>

<Figure size 720x288 with 0 Axes>

<Figure size 720x288 with 0 Axes>

<Figure size 720x288 with 0 Axes>

<Figure size 720x288 with 0 Axes>

<Figure size 720x288 with 0 Axes>

<Figure size 720x288 with 0 Axes>

<Figure size 720x288 with 0 Axes>

<Figure size 720x288 with 0 Axes>

<Figure size 720x288 with 0 Axes>

<Figure size 720x288 with 0 Axes>

<Figure size 720x288 with 0 Axes>

<Figure size 720x288 with 0 Axes>

<Figure size 720x288 with 0 Axes>

<Figure size 720x288 with 0 Axes>

<Figure size 720x288 with 0 Axes>

<Figure size 720x288 with 0 Axes>

```
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
```

```
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
```

```
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
<Figure size 720x288 with 0 Axes>
```

[6]:
```
from IPython.display import Video

Video("random_planting.mp4")
```

[6]:
```
<IPython.core.display.Video object>
```

2.2 Trouver une façon plus rapide de faire les simulations (Bonus)