Advanced (Sol) - Hari

May 24, 2020

[1]:
```
import numpy as np
from scipy.ndimage import convolve
```

[2]:
```
def run_game(number_of_iterations):
    # Convolving on this kernel does a count of cells around the center
    kernel = np.array([[1, 1, 1],
                       [1, 0, 1],
                       [1, 1, 1]], dtype=np.uint8)

    board = [[0,0,1,0,1],
    [0,1,0,0,1],
    [0,1,0,0,1],
    [0,0,1,0,1]]
    board = np.array(board)

    print(board)

    count = 0
    cutoff = 1
    while count < number_of_iterations:

        count += 1

        # Run a single 2D convolutional filter over the board with constant 0␣
↪padding
        convolved_board = convolve(board, kernel, mode="constant", cval=0)

        # The kernel we used finds the sum of the 8 cells around a given cell
        # So we can do a bit of fancy numpy work to get the next board
        next_board = (
            ((board == 1) & (convolved_board > 1) & (convolved_board < 4))
            | ((board == 0) & (convolved_board == 3))
        ).astype(np.uint8)

        print(next_board)

        print(
```

```
            f"count: {count}"
        )

        board = next_board
```

[3]:
```
run_game(20)
```

```
[[0 0 1 0 1]
 [0 1 0 0 1]
 [0 1 0 0 1]
 [0 0 1 0 1]]
[[0 0 0 1 0]
 [0 1 1 0 1]
 [0 1 1 0 1]
 [0 0 0 1 0]]
count: 1
[[0 0 1 1 0]
 [0 1 0 0 1]
 [0 1 0 0 1]
 [0 0 1 1 0]]
count: 2
[[0 0 1 1 0]
 [0 1 0 0 1]
 [0 1 0 0 1]
 [0 0 1 1 0]]
count: 3
[[0 0 1 1 0]
 [0 1 0 0 1]
 [0 1 0 0 1]
 [0 0 1 1 0]]
count: 4
[[0 0 1 1 0]
 [0 1 0 0 1]
 [0 1 0 0 1]
 [0 0 1 1 0]]
count: 5
[[0 0 1 1 0]
 [0 1 0 0 1]
 [0 1 0 0 1]
 [0 0 1 1 0]]
count: 6
[[0 0 1 1 0]
 [0 1 0 0 1]
 [0 1 0 0 1]
 [0 0 1 1 0]]
count: 7
[[0 0 1 1 0]
```

```
 [0 1 0 0 1]
 [0 1 0 0 1]
 [0 0 1 1 0]]
count: 8
[[0 0 1 1 0]
 [0 1 0 0 1]
 [0 1 0 0 1]
 [0 0 1 1 0]]
count: 9
[[0 0 1 1 0]
 [0 1 0 0 1]
 [0 1 0 0 1]
 [0 0 1 1 0]]
count: 10
[[0 0 1 1 0]
 [0 1 0 0 1]
 [0 1 0 0 1]
 [0 0 1 1 0]]
count: 11
[[0 0 1 1 0]
 [0 1 0 0 1]
 [0 1 0 0 1]
 [0 0 1 1 0]]
count: 12
[[0 0 1 1 0]
 [0 1 0 0 1]
 [0 1 0 0 1]
 [0 0 1 1 0]]
count: 13
[[0 0 1 1 0]
 [0 1 0 0 1]
 [0 1 0 0 1]
 [0 0 1 1 0]]
count: 14
[[0 0 1 1 0]
 [0 1 0 0 1]
 [0 1 0 0 1]
 [0 0 1 1 0]]
count: 15
[[0 0 1 1 0]
 [0 1 0 0 1]
 [0 1 0 0 1]
 [0 0 1 1 0]]
count: 16
[[0 0 1 1 0]
 [0 1 0 0 1]
 [0 1 0 0 1]
 [0 0 1 1 0]]
```

```
count: 17
[[0 0 1 1 0]
 [0 1 0 0 1]
 [0 1 0 0 1]
 [0 0 1 1 0]]
count: 18
[[0 0 1 1 0]
 [0 1 0 0 1]
 [0 1 0 0 1]
 [0 0 1 1 0]]
count: 19
[[0 0 1 1 0]
 [0 1 0 0 1]
 [0 1 0 0 1]
 [0 0 1 1 0]]
count: 20
```