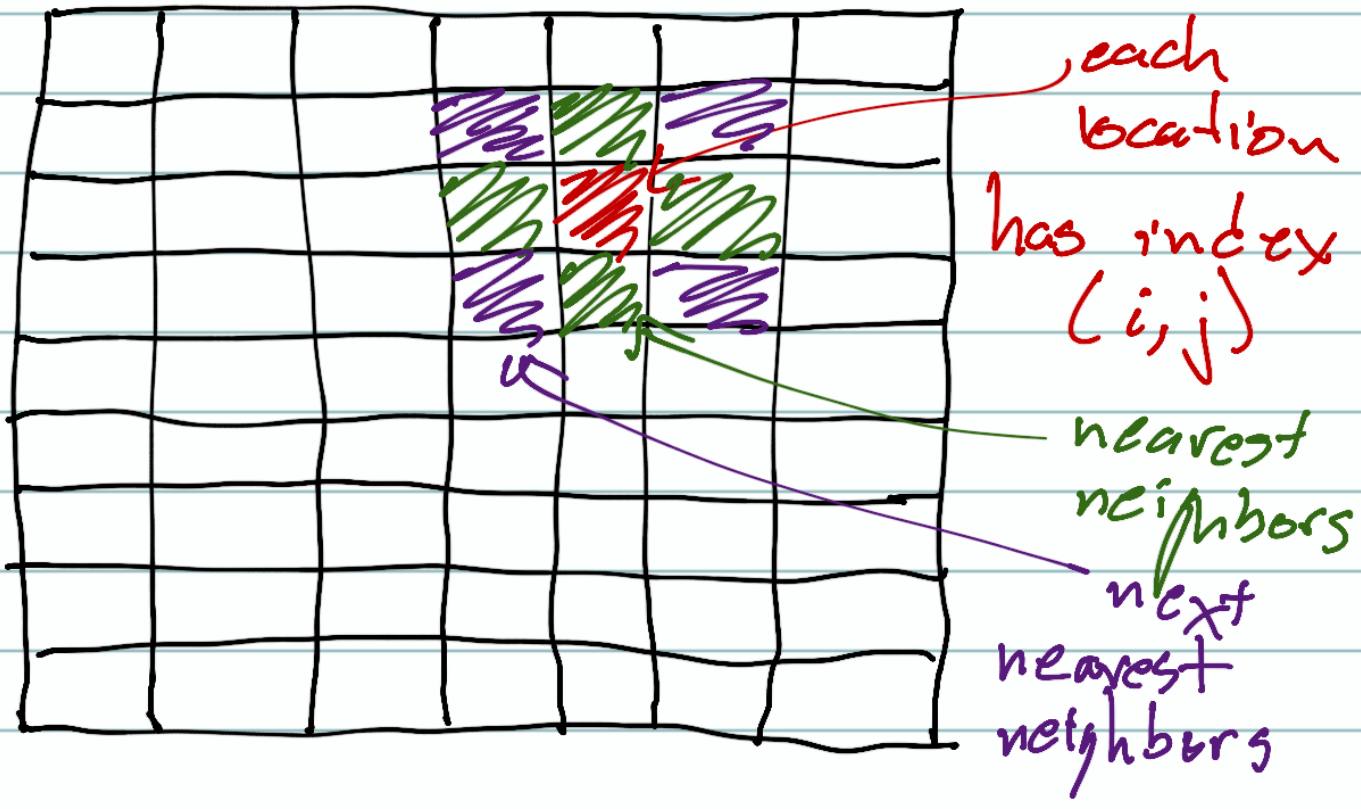
Cellular Automata

-> A type of model without equations, describing how complex behaviors can arise in space and time
  -> Consider locations in a network (commonly a 2D grid where each location has some neighbors)
-> Model evolves through iterations
-> On each iteration each location must follow a simple set of rules for interaction w/ neighboring locations



Pseudocode for a generic CA

Set params

```
a = ...
b = ...
n = ...
niter = ...
```

initialize evolving variable (x) at all locations of network

```
x = zeros(n, n)
```

iterative steps (like time)

```
for k = 1...niter
```

iterate over all locations (this can often be vectorized)

```
    for i = 1...n
        for j = 1...n
```

check rule

```
            if x(i,j) ...
                do something
                to neighbor
                x(i±1, j±1) ->
            else
                .
            end
        end
    end
end
```

The cannonical example

Conway's Game of Life

-> nxn grid where each grid cell can be "alive" or "dead" ($x=0$ or $x=1$)
-> Each cell has 8 neighbors

Rules

1. A live cell with less than 2 live neighbors dies
2. A live cell with 2-3 live neighbors lives
3. A live cell with 4 or more neighbors dies
4. A dead cell with 3 live neighbors becomes live

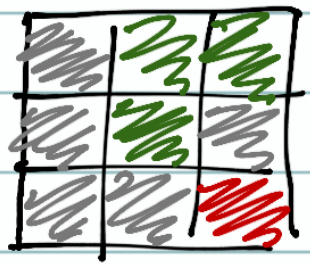
1 -> underpopulation
2 -> co-existence
3 -> overpopulation
4 -> reproduction

-> Draw 1-9 examples

-> Show real example

ESM Activity (and PS6): Forest-Fire Model

Setup: A forest is an $n \times n$ grid of squares of land



States

1: Base
2: Forest
3: Forest on fire

Rules

Lightning 1. Every iteration, each site is struck by lighting w/probability p_l.
$\rightarrow$ If $2 \rightarrow 3$

Fire spread 2. If a site $[i,j]$ is 3, then $[i \pm 1, j], [i, j \pm 1] \rightarrow 3$ w/probability p_f if they are 2.

Tree growth 3. Every iteration each site "grows" with probability p_g. If $1 \rightarrow 2$

Activity steps

1. Make a grid of $n \times n$ size. Randomly populate it w/ 50-50 trees and bare ground.
2. Use pcolor to plot system state
3. Structure code: params, iter loop, logical statements