Conway's Game of Life
Computer Algorithms & Visualization

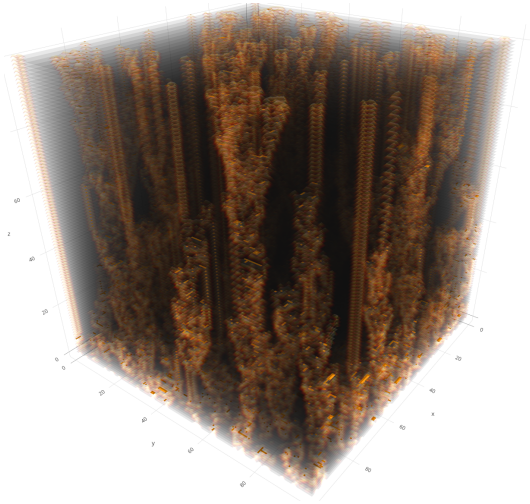
Marcelo Ponce

Labs Linked to Curriculum in STEM, 2024/'25

Department of Computer and Mathematical Sciences - UTSC

Welcome to the *Department of Computer and Mathematical Sciences* @ UTSC!!!

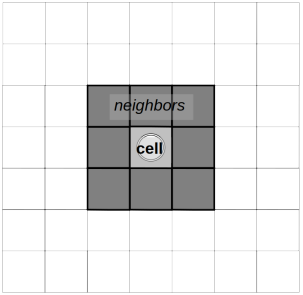
Today's Class



Introduction

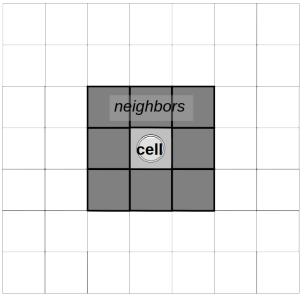
Conway's Game of Life

- Simple "*cellular automaton*" game:
  a regular grid of cells, each with a finite number of possible states



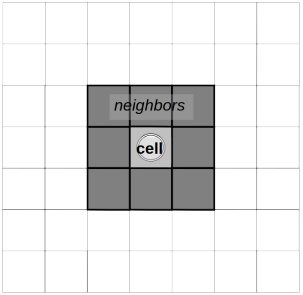
Conway's Game of Life

- Simple "*cellular automaton*" game:
  a regular grid of cells, each with a finite number of possible states
- created by British mathematician *John Horton Conway* in 1970



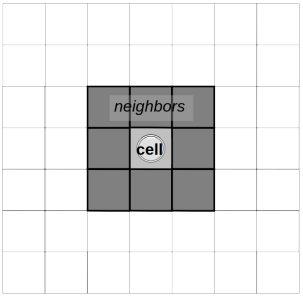
Conway's Game of Life

- Simple "*cellular automaton*" game:
  a regular grid of cells, each with a finite number of possible states
- created by British mathematician *John Horton Conway* in 1970
- simple *set of rules –algorithm–*



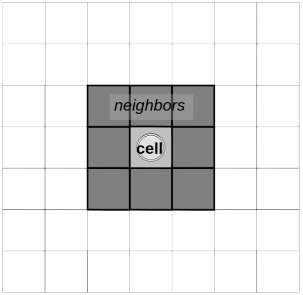
Conway's Game of Life

- Simple "*cellular automaton*" game:
  a regular grid of cells, each with a finite number of possible states
- created by British mathematician *John Horton Conway* in 1970
- simple *set of rules –algorithm–*
- a *new generation* is created advancing "t" by 1, applying the set of rules



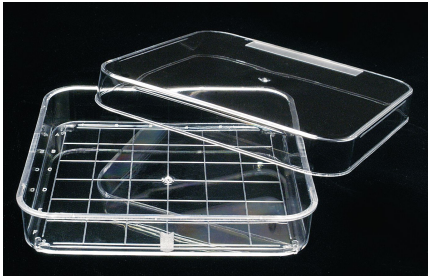
Conway's Game of Life

- Simple "*cellular automaton*" game:
  a regular grid of cells, each with a finite number of possible states
- created by British mathematician *John Horton Conway* in 1970
- simple *set of rules –algorithm–*
- a *new generation* is created advancing "t" by 1, applying the set of rules
- each generation is a pure function of the preceding one: the rules continue to be applied repeatedly to create further generations;
  given an *initial configuration*, the whole future of the system is determined.



Rules

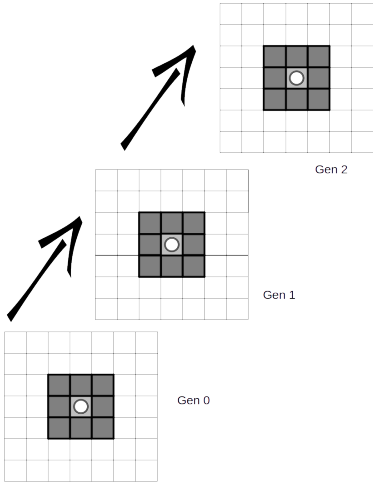
- Every cell interacts with its eight neighbors
- At each step in time, the following transitions occur:
  1. Any live cell with **fewer than two live** neighbours **dies**: *underpopulation*.
  2. Any live cell with **two or three live** neighbours **lives** on to the next generation.
  3. Any live cell with **more than three** live neighbours **dies**: *overpopulation*.
  4. Any dead cell with exactly three live neighbours becomes a live cell: *reproduction*.



Hands-On I

Let's do it on the worksheets/board...

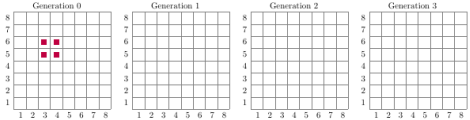
- Every cell interacts with its eight neighbors
- At each step in time, the following transitions occur:
  1. Any live cell with **fewer than two live** neighbours **dies**: *underpopulation*.
  2. Any live cell with **two or three live** neighbours **lives** on to the next generation.
  3. Any live cell with **more than three** live neighbours **dies**: *overpopulation*.
  4. Any dead cell with exactly three live neighbours becomes a live cell: *reproduction*.



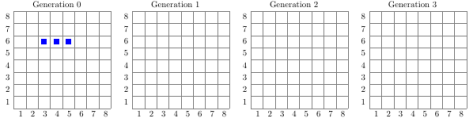
Game of Life – Hands-On

April 22, 2025

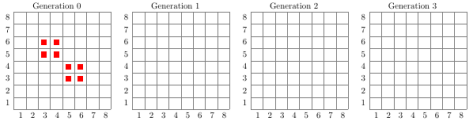
1 The Block



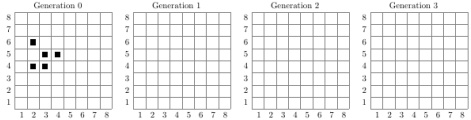
2 Blinker



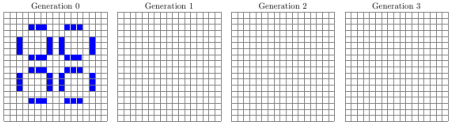
3 Beacon



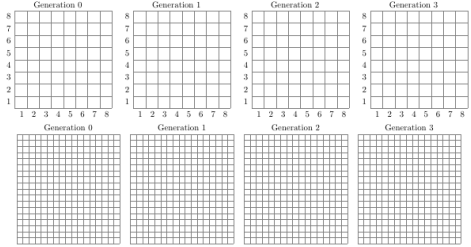
4 Glider



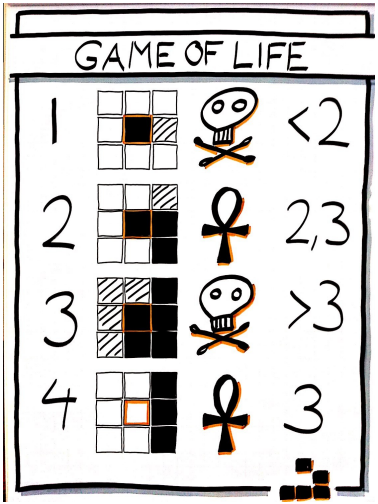
5 Pulsar



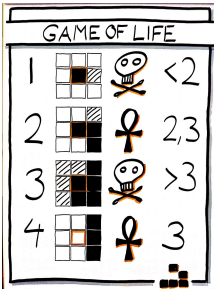
6 Make your own design...



Recall G-o-L Rules...



Life Forms in GoL



Still lifes	
Block	
Bee-hive	
Loaf	
Boat	
Tub	

Oscillators	
Blinker (period 2)	
Toad (period 2)	
Beacon (period 2)	
Pulsar (period 3)	
Penta-decathlon (period 15)	

Spaceships	
Glider	
Light-weight spaceship (LWSS)	
Middle-weight spaceship (MWSS)	
Heavy-weight spaceship (HWSS)	

Game of Life in R

Game of Life in R

We can use a library named `fun`:

- We may need to install the library first, e.g.
  `> install.packages("fun")`
- Then, load the library:
  `> library(fun)`
- Now, let's try the following command and follow the instructions in the screen:
  `> demo("GameOfLife")`

after pressing Enter...

Game of Life in R

We can use a library named `fun`:

- We may need to install the library first, e.g.

  ```
  > install.packages("fun")
  ```

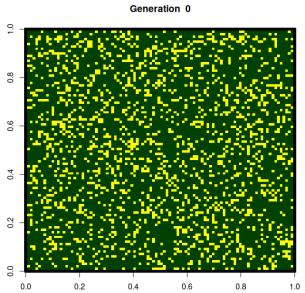
- Then, load the library:

  ```
  > library(fun)
  ```

- Now, let's try the following command and follow the instructions in the screen:

  ```
  > demo("GameOfLife")
  ```

after pressing Enter...



after ≈ hundred times...

Game of Life in R

We can use a library named `fun`:

- We may need to install the library first, e.g.

  ```
  > install.packages("fun")
  ```

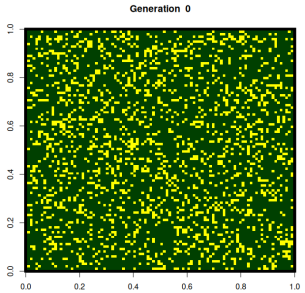
- Then, load the library:

  ```
  > library(fun)
  ```

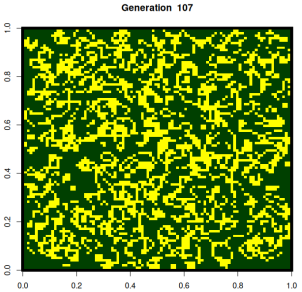
- Now, let's try the following command and follow the instructions in the screen:

  ```
  > demo("GameOfLife")
  ```

after pressing Enter ...



after ≈ hundred times...

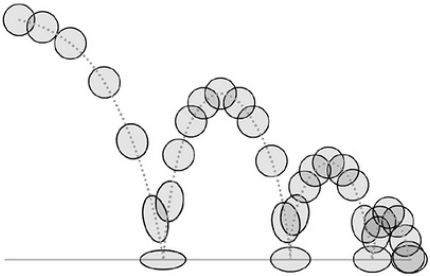


```
> ## Game of Life (https://en.wikipedia.org/wiki/The_Game_of_Life)
> ## code by Linlin Yan <linlin.yan@cos.name>
> ## URL: https://d.cosx.org/d/15402
> row <- 100

> col <- 100

> init_life <- function(p) {
+     m <- matrix(ifelse(runif(row * col) < p, 1, 0), row, col)
+     m[1, ] = -1
+     m[row, ] = -1
+     m[, 1] = -1
+     m[, col] = -1
+     m
+ }

> count_life <- function(m) {
+     sum(ifelse(m < 0, 0, m))
+ }

> next_life <- function(x) {
+     t <- matrix(-1, row, col)
+     for (i in 2:(row - 1)) {
+         for (j in 2:(col - 1)) {
+             c <- count_life(x[(i - 1):(i + 1), (j - 1):(j + 1)])
+             t[i, j] <- ifelse((c >= 3) && (c <= 4), 1, 0)
+         }
+     }
+     t
+ }

> life_color <- c("#000000", "#004000", "#FFFF00")

> draw_life <- function(m, generation) {
+     image(m, col = life_color)
+     title(paste("Generation ", generation))
+ }

> generation <- 0

> lives <- init_life(0.2)

> while (TRUE) {
+     draw_life(lives, generation)
+     lives <- next_life(lives)
+     generation <- generation + 1
+ }
```

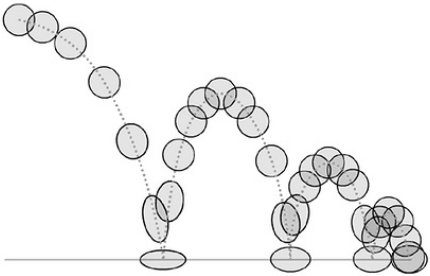
Visualizations

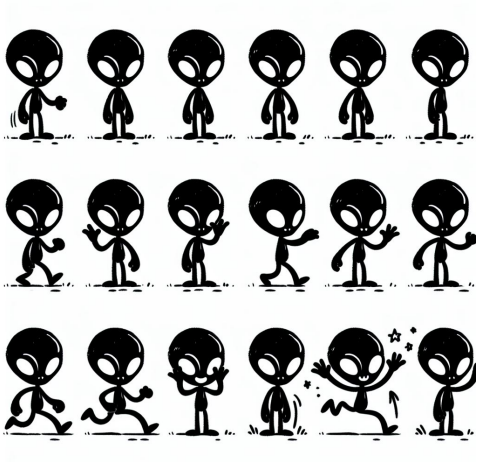
How are animations generated?

How are animations generated?



How are animations generated?





How are animations generated?

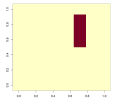
- *Frame by frame*
- Each "frame" corresponds to one instant in time
- With enough frames and *scrolling* through them fast enough, it gives the appearance of a "natural motion"

How are animations generated?

- *Frame by frame*
- Each "frame" corresponds to one instant in time
- With enough frames and *scrolling* through them fast enough, it gives the appearance of a "natural motion"
- In *Game of Life*, the "frames" would be each of the *generations*

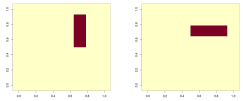
How are animations generated?

- *Frame by frame*
- Each "frame" corresponds to one instant in time
- With enough frames and *scrolling* through them fast enough, it gives the appearance of a "natural motion"
- In *Game of Life*, the "frames" would be each of the *generations*



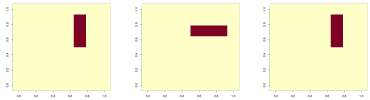
How are animations generated?

- *Frame by frame*
- Each "frame" corresponds to one instant in time
- With enough frames and *scrolling* through them fast enough, it gives the appearance of a "natural motion"
- In *Game of Life*, the "frames" would be each of the *generations*



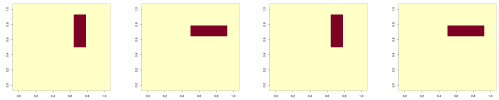
How are animations generated?

- *Frame by frame*
- Each "frame" corresponds to one instant in time
- With enough frames and *scrolling* through them fast enough, it gives the appearance of a "natural motion"
- In *Game of Life*, the "frames" would be each of the *generations*



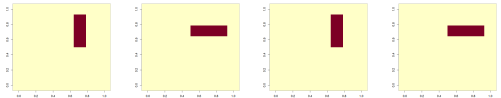
How are animations generated?

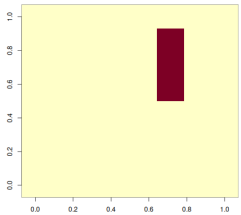
- *Frame by frame*
- Each "frame" corresponds to one instant in time
- With enough frames and *scrolling* through them fast enough, it gives the appearance of a "natural motion"
- In *Game of Life*, the "frames" would be each of the *generations*



How are animations generated?

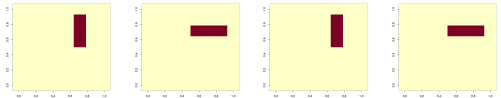
- *Frame by frame*
- Each "frame" corresponds to one instant in time
- With enough frames and *scrolling* through them fast enough, it gives the appearance of a "natural motion"
- In *Game of Life*, the "frames" would be each of the *generations*

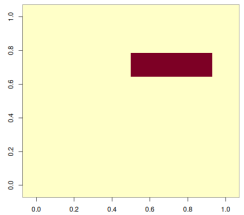




How are animations generated?

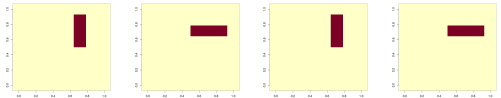
- *Frame by frame*
- Each "frame" corresponds to one instant in time
- With enough frames and *scrolling* through them fast enough, it gives the appearance of a "natural motion"
- In *Game of Life*, the "frames" would be each of the *generations*

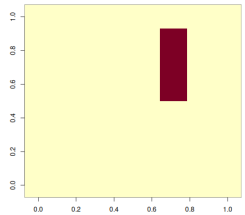




How are animations generated?

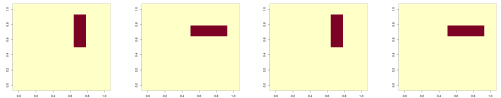
- *Frame by frame*
- Each "frame" corresponds to one instant in time
- With enough frames and *scrolling* through them fast enough, it gives the appearance of a "natural motion"
- In *Game of Life*, the "frames" would be each of the *generations*

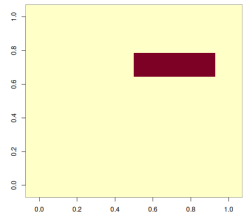




How are animations generated?

- *Frame by frame*
- Each "frame" corresponds to one instant in time
- With enough frames and *scrolling* through them fast enough, it gives the appearance of a "natural motion"
- In *Game of Life*, the "frames" would be each of the *generations*



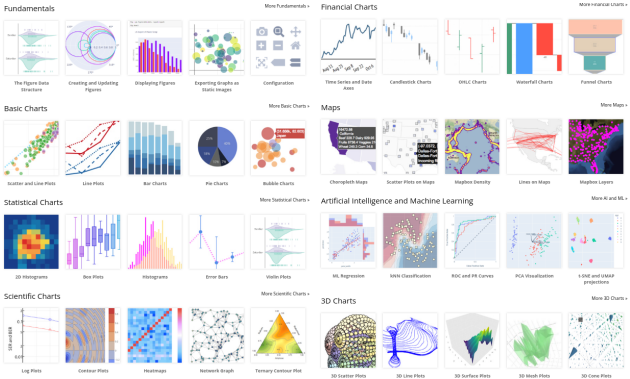


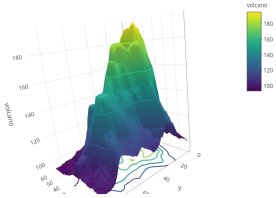
Simple Movie generation...

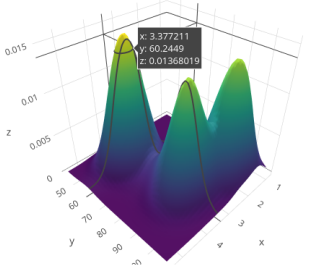
```
library(animation)

animation::saveVideo(
  # exactly same loop as before
  for (i in 1:10){
    draw_life(lives, generation)
    lives <- next_life(lives)
    generation <- generation + 1
  },
  # specify name of the movie...
  video.name = "Animation_GoL.avi")
```

Interactive Visualizations in R, with `plotly`

- `https://plotly.com/r/`
- free, open-source tool to create **interactive** plots
- high-quality visualizations

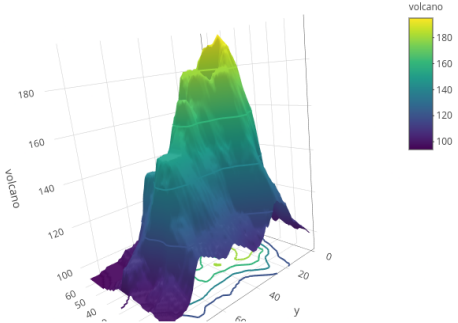






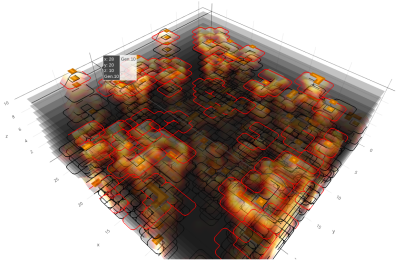
Simple `plotly` example...

```r
library(plotly)

# volcano is a numeric matrix
# containing topographical
# data, included in R
fig <- plot_ly(z = ~volcano)
fig <- fig %>% add_surface()

fig
```



Interactive Visualization of Game of Life

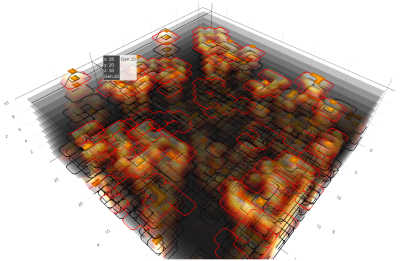
```
# load some pre-defined functions
> source("conway_life.R")

# run a default case
> run_sim()
```



```
# or other cases
> lives <- run_sim(75,75,,10)
> plot_gens(lives, wLegend=FALSE,
    wContours=FALSE, wSurf=TRUE)
```

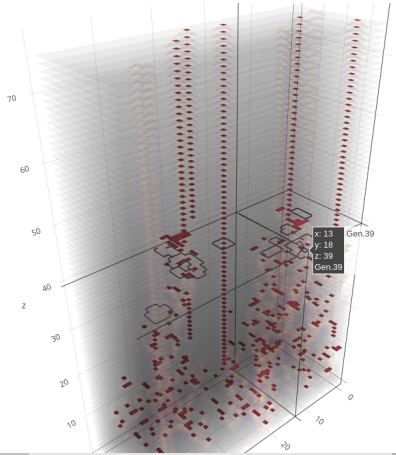
Interactive Visualization of Game of Life

```
# load some pre-defined functions
> source("conway_life.R")

# run a default case
> run_sim()
```



```
# or other cases
> lives <- run_sim(75,75,,10)
> plot_gens(lives, wLegend=FALSE,
    wContours=FALSE, wSurf=TRUE)
```

```
> gen2 <- run_sim(20,50,,75)
> plot_gens(gen2, wSurf=TRUE, colorMap=
    "Electric", opac=0.05)
```

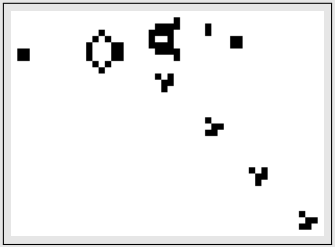


Conclusions

Discussion & Final Remarks

- Conway's *Game of Life* has applications to several theoretical areas of science: mathematics, computer science, physics and even biology
- synthetic biology, epigenetics, ...
- it represents the emergence of complex dynamics from a very simple set of rules – complex system
- it allows us to "experiment" in a different way than we usually do in a "wet"-lab

A single Gosper's *glider gun* creating *gliders*



SRC: wikipedia

Conclusions

What have we discussed today:

- Conway's Game of Life
- Algorithms and R scripts
- Visualization: animations
- Movies generation in R
- Interactive Visualizations

References

References

- https://conwaylife.com/wiki/Conway's_Game_of_Life
- https://playgameoflife.com/
- https://nicholas.carlini.com/writing/2020/digital-logic-game-of-life.html