Package: cellularautomata (via r-universe)

November 22, 2024

Type Package

Title Cellular Automata

Version 0.1.0

Maintainer Vlad Tarko <vladtarko@gmail.com>

Description Create cellular automata from Wolfram rules. Allows the creation of Wolfram style plots, as well as of animations. Easy to create multiple plots, for example the output of a rule with different initial states, or the output of many different rules from the same state. The output of a cellular automaton is given as a matrix, making it easy to try to explore the possibility of predicting its time evolution using various statistical tools available in R.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 7.3.2.9000

VignetteBuilder knitr

Imports gganimate, ggplot2, patchwork, purrr, rlang

Suggests knitr, rmarkdown

Config/pak/sysreqs libgdal-dev gdal-bin libgeos-dev libicu-dev libssl-dev libproj-dev libsqlite3-dev libudunits2-dev

Repository https://vladtarko.r-universe.dev

RemoteUrl https://github.com/vladtarko/cellularautomata

RemoteRef HEAD

RemoteSha f94e285e4f70cd80a3547d79366e4cc26c87881d

Contents

`ca` *Create Cellular Automaton*

Description

Create Cellular Automaton

Usage

```
ca(wolframrule, initialstate, steps = 100, ncols = 101, wrap = TRUE)
```

Arguments

`wolframrule` integer identifying the algorithm according to Wolfram numbering

`initialstate` a vector setting up the initial state

`steps` integer spacifying for how long to run the algorithm

`ncols` how many columns to have. If ‘initialstate‘ is specified, ‘ncols‘ is calculated as ‘length(initialstate)‘. If ‘initialstate‘ is not specified, it is defined as a 1 in the middle of zeros. For instance, with the default ‘ncols = 11‘, the ‘initialstate‘ is a vector of 5 zeros, 1, and another 5 zeros.

`wrap` boolean, default is TRUE. Whether it uses a circular wrap at the end and beginning of lines. If FALSE it puts empty slots on the first and last columns.

Value

an object of class ‘c("cellular_automaton", "matrix")‘

Author(s)

Adapted from code by Nicola Procopio

References

<https://en.wikipedia.org/wiki/Cellular_automaton>

Examples

```
# Wolfram's rule 30
ca(30)

# Wolfram's rule 126 with a random initial state
ca(126,
   initialstate = sample(c(0, 1), size = 100, replace = TRUE),
   steps = 100)
```

plot.cellular_automata *Plot a cellular automaton*

Description

Plot a cellular automaton

Usage

```
## S3 method for class 'cellular_automata'
plot(
  x,
  time_flow = "down",
  circle = FALSE,
  title = paste("Rule: ", attr(x, "wolfram_rule")),
  animate = FALSE,
  ...
)
```

Arguments

`x`	A cellular automaton, usually previously defined by ‘ca()‘.
`time_flow`	String: "down" (default) or "up". Whether time flow is represented as going from top-to-bottom or bottom-to-top.
`circle`	Whether to make the plot circular. Default is FALSE.
`title`	Title of the plot. Use ‘NULL‘ to remove.
`animate`	Whether to return a gganimate object instead of a static ggplot. Default FALSE.
`...`	Not used (included for consistency with the ‘plot‘ generic).

Value

A ggplot of the visual representation of the cellular automaton, or a gganimate object.

Examples

```
ca(30) |> plot()
ca(30, ncols = 100, steps = 100) |> plot()
ca(45, ncols = 100, steps = 100) |> plot()
ca(86, ncols = 100, steps = 100) |> plot()

# use a random initial state
ca(126,
   initialstate = sample(c(0, 1), size = 100, replace = TRUE),
   steps = 100) |>
 plot()
```

`wolfram_rule` *Create the rule for a specific Wolfram number*

Description

Create the rule for a specific Wolfram number

Usage

```
wolfram_rule(rule)
```

Arguments

`rule`	the Wolfram rule

Value

a vector with 8 elements defining the responses to: (111), (110), (101), (100), (011), (010), (001), (000) on the previous row

Examples

```
# get the definition of rule 30
wolfram_rule(30)
```

`wolfram_rule_def` *Plot the definition of a Wolfram rule*

Description

Plot the definition of a Wolfram rule

Usage

```
wolfram_rule_def(rule)
```

Arguments

`rule`	integer, the Wolfram rule

Value

a ggplot object defining the rule

Examples

```
wolfram_rule_def(30)
```

Index