Package 'phylopomp'

December 3, 2021

R topics documented:

`phylopomp-package` *Phylodynamics for POMP models*

Description

Super cool.

Author(s)

Aaron A. King, Qianying Lin

`diagram` *Genealogy process diagram*

Description

Produces a diagram of the genealogy process state.

Usage

```
diagram(illus, ...)
```

Arguments

`illus`	character; illustrations produced by `getInfo` or one of the `playX` functions.
`...`	graphical parameter settings, suitable for passing to `gpar`.

Value

A list of **grid** graphics objects (`grobs`), invisibly.

Examples

```
library(tidyverse)
playMoran(n=5,mu=10,times=c(0,seq(100,200,by=25)),
  stationary=FALSE,ill=TRUE,tree=TRUE) -> x
plot(x[5,],points=TRUE,diagram=TRUE)

playMoran(n=8,mu=8,times=100:130,sample=FALSE,tree=TRUE,ill=TRUE) |>
  mutate(dg=diagram(illus)) -> x
plot(x,points=TRUE,diagram=TRUE,root_time=NA)

playMoran(n=8,mu=8,times=0:30,sample=FALSE,tree=TRUE,ill=TRUE,
  stationary=FALSE) |>
  mutate(dg=diagram(illus)) -> x
plot(x,points=TRUE,diagram=TRUE,root_time=NA)

library(cowplot)
playMoran(n=5,mu=5,times=0:3,stationary=TRUE,tree=TRUE,ill=TRUE) |>
  mutate(grob=diagram(illus)) -> x
plot_grid(plotlist=c(x$grob,treeplot(x$tree,points=TRUE)),ncol=2,byrow=FALSE)
```

getInfo *Retrieve information from genealogy process simulation*

Description

getInfo

Usage

```
getInfo(data, ...)

## S3 method for class 'gpsim'
getInfo(data, ..., prune = TRUE, compact = FALSE)
```

Arguments

data	gpsim object.
...	arguments passed to specific methods.
prune	logical; prune the tree?
compact	logical; return the tree in compact representation?

Examples

```
library(tidyverse)
library(cowplot)

playMoran(n=5,mu=5,times=0:10,t0=0,tree=TRUE,ill=TRUE) -> x
playMoran(x,times=11:20,tree=TRUE) -> x
plot(x)

playMoran(n=5,mu=10,times=0:10,t0=-3) |>
  getInfo() -> y
plot(y,points=TRUE,diagram=TRUE)

playMoran(n=20,mu=20,times=0:20,stationary=FALSE,tree=TRUE,ill=TRUE) -> x
plot(x,points=TRUE)

y <- getInfo(x,prune=FALSE)
plot(y,points=TRUE)

playMoran(n=5,mu=5,t0=-1,times=0:3,stationary=FALSE,tree=TRUE,ill=TRUE) -> x
plot(x,points=TRUE,diagram=TRUE)

y <- getInfo(x)
plot(y,points=TRUE,diagram=TRUE)

playMoran(n=20,mu=10,times=1:10,sample=TRUE,stationary=FALSE) -> x
x |> getInfo(prune=FALSE,compact=TRUE) -> y
```

```
plot(y,points=TRUE,diagram=TRUE)

x |> getInfo(prune=TRUE,compact=TRUE) -> y
plot(y,points=TRUE,diagram=TRUE)

playMoran(n=8,mu=8,times=0,tree=TRUE,ill=TRUE,sample=FALSE,stationary=TRUE) |>
  playMoranWChain(ntimes=4) |>
  mutate(diag=diagram(illus)) |>
  pull(diag) |>
  {\(x)plot_grid(plotlist=x,ncol=1)}()

playMoran(n=8,mu=8,times=0,tree=TRUE,ill=TRUE,sample=FALSE,stationary=FALSE) |>
  playMoranWChain(ntimes=40) |>
  mutate(diag=diagram(illus)) |>
  slice(35:40) |>
  pull(diag) |>
  {\(x)plot_grid(plotlist=x,ncol=1)}()
```

---

lbdp *Linear birth-death process.*

Description

Simulation and inference based on linear birth-death-sampling processes.

Usage

```
playLBDP(
  data = NULL,
  lambda,
  mu,
  psi,
  n0 = 1,
  t0 = 0,
  times,
  tree = FALSE,
  ill = FALSE
)

lbdp_exact(data, lambda, mu, psi)

lbdp_pomp(data, lambda, mu, psi, n0 = 1, t0 = 0)
```

Arguments

data	data frame containing the genealogy event times and event codes.
lambda	birth rate
mu	death rate

psi	sampling rate
n0	initial population size
t0	initial time
times	times at which output is requested.
tree	logical; represent the genealogical tree in Newick format?
ill	logical; return an illustration?

Details

playLBDP is a simulator of the genealogy process induced by a linear birth-death-sampling process.

lbdp_exact gives the exact likelihood of a linear birth-death process, conditioned on $n_0 = 0$ (Stadler, 2010, Thm 3.5). The derivation is also given in comments in the code.

The data argument should in the format returned by newick2df.

lbdp_pomp constructs a **pomp** object containing a given set of data and a linear birth-death-sampling process.

It is assumed that data is in the format returned by newick2df.

Value

playLBDP returns an object of class gpsim. Available methods for such objects include getInfo and plot.

lbdp_exact returns the log likelihood of the genealogy. Note that the time since the most recent sample is informative.

References

T. Stadler. Sampling-through-time in birth-death trees. *Journal of Theoretical Biology* **267**, 396–404, 2010.

See Also

Other Genealogy processes: leventhal, moran, sirs, sirws

Examples

```
playLBDP(lambda=1,mu=0.5,psi=1,n0=3,times=seq(0,4,by=0.2),tree=TRUE) -> x
plot(x,points=TRUE)

y <- getInfo(x,prune=FALSE)
plot(y,points=TRUE)

library(ggplot2)
y$lineages |>
  ggplot(aes(x=time,y=lineages))+
  geom_step()+
  geom_vline(xintercept=y$etimes,alpha=0.1)
```

```
y$cumhaz |>
  ggplot(aes(x=exp(-Lambda)))+
  stat_ecdf()+
  geom_abline(slope=1)

playLBDP(lambda=2,mu=1,psi=0.5,n0=1,times=5) |>
  getInfo() -> x
plot(getInfo(x,compact=TRUE),points=TRUE)

library(pomp)
x$tree |>
  newick2df(time=5) |>
  lbdp_pomp(psi=0.5,lambda=2,mu=1,n0=1) |>
  pfilter(Np=2000) |>
  logLik()

x$tree |>
  newick2df(time=5) |>
  lbdp_exact(psi=0.5,lambda=2,mu=1)
```

`leventhal` *Leventhal (2014) SI model*

Description

Simulation and inference for Leventhal's model

Usage

```
leventhal_pomp(data, beta, gamma, psi, N, I0 = 1, t0 = 0)
```

Arguments

`data`	optional data frame; output from `playLeventhal`.
`beta`	contact rate
`gamma`	recovery rate
`psi`	sampling rate
`N`	population size
`I0`	initial number of infections
`t0`	initial time

Details

`leventhal_pomp` constructs a **pomp** object containing a given set of data and a Leventhal-type SIS infection.

It is assumed that `data` is in the format returned by `newick2df`.

See Also

Other Genealogy processes: `lbdp`, `moran`, `sirs`, `sirws`

`moran` *Moran genealogy process.*

Description

Run the MGP simulator.

Usage

```
playMoran(
  data = NULL,
  n,
  mu,
  t0 = 0,
  times,
  sample = TRUE,
  tree = FALSE,
  ill = FALSE,
  stationary = TRUE
)

playMoranWChain(
  data = NULL,
  n,
  mu,
  t0 = 0,
  ntimes,
  tree = TRUE,
  ill = TRUE,
  stationary = TRUE
)
```

Arguments

`data`	optional data frame; output from `playMoran` or `playMoranWChain`
`n`	population size
`mu`	Moran event rate.

t0	initial time
times	times at which output is requested.
sample	logical; if sample=TRUE, a sample is taken at each of the specified times.
tree	logical; represent the genealogical tree in Newick format?
ill	logical; return an illustration?
stationary	logical; should the initial genealogy be drawn from the stationary distribution?
ntimes	integer; number of timesteps to advance the chain.

Value

A tibble with state attribute.

See Also

Other Genealogy processes: lbdp, leventhal, sirs, sirws

Examples

```
library(tidyverse)
library(cowplot)

playMoran(n=5,mu=5,times=0:10,t0=0,tree=TRUE,ill=TRUE) -> x
playMoran(x,times=11:20,tree=TRUE) -> x
plot(x)

playMoran(n=5,mu=10,times=0:10,t0=-3) |>
  getInfo() -> y
plot(y,points=TRUE,diagram=TRUE)

playMoran(n=20,mu=20,times=0:20,stationary=FALSE,tree=TRUE,ill=TRUE) -> x
plot(x,points=TRUE)

y <- getInfo(x,prune=FALSE)
plot(y,points=TRUE)

playMoran(n=5,mu=5,t0=-1,times=0:3,stationary=FALSE,tree=TRUE,ill=TRUE) -> x
plot(x,points=TRUE,diagram=TRUE)

y <- getInfo(x)
plot(y,points=TRUE,diagram=TRUE)

playMoran(n=20,mu=10,times=1:10,sample=TRUE,stationary=FALSE) -> x
x |> getInfo(prune=FALSE,compact=TRUE) -> y
plot(y,points=TRUE,diagram=TRUE)

x |> getInfo(prune=TRUE,compact=TRUE) -> y
plot(y,points=TRUE,diagram=TRUE)

playMoran(n=8,mu=8,times=0,tree=TRUE,ill=TRUE,sample=FALSE,stationary=TRUE) |>
  playMoranWChain(ntimes=4) |>
```

```
  mutate(diag=diagram(illus)) |>
  pull(diag) |>
  {\(x)plot_grid(plotlist=x,ncol=1)}()

playMoran(n=8,mu=8,times=0,tree=TRUE,ill=TRUE,sample=FALSE,stationary=FALSE) |>
  playMoranWChain(ntimes=40) |>
  mutate(diag=diagram(illus)) |>
  slice(35:40) |>
  pull(diag) |>
  {\(x)plot_grid(plotlist=x,ncol=1)}()
```

`newick2df` *Convert a tree in Newick format to data frame*

Description

Convert a genealogical tree in Newick format to a data frame suitable for use with **pomp**.

Usage

```
newick2df(tree, time = NA, root_time = 0)
```

Arguments

`tree`	tree data in Newick format.
`time`	time of the genealogy.
`root_time`	time of the root.

Details

If `time` is furnished, it is assumed that the absence of samples between the latest leaf and `time` is informative.

Invisible nodes (labeled 'X_' for any X) are dropped.

Value

A data frame suitable for use as `pomp` input, containing three columns:

time numeric; time of the genealogy event.

lineages integer; the value of the lineage-count function at the specified time. Note that this function is right-continuous with left limits, and constant on the inter-event intervals.

code integer; a code describing the nature of the event. 1 indicates a coalescence; 0 indicates a dead sample; -1 indicates a live sample; 2 indicates the root.

Examples

```
playSIRwS(Beta=2,gamma=1,psi=2,S0=100,I0=2,R0=0,times=c(0,5),t0=0,tree=TRUE) -> x
y <- getInfo(x)
newick2df(y$tree) -> z

# compare to y$lineages
y$lineages |>
  all.equal(tail(z,-1)[,1:2],tolerance=1e-5)
```

`print` *Special print functions*

Description

Print functions for 'gpsim' objects.

Usage

```
## S3 method for class 'gpsim'
print(x, ...)
```

Arguments

`x`	`gpsim` object.
`...`	arguments passed to print.

`reexports` *Objects exported from other packages*

Description

These objects are imported from other packages. Follow the links below to see their documentation.

foreach `%dopar%`, `foreach`, `registerDoSEQ`

sirs *SIRS with sampling simulator.*

Description

Run the simulator.

Usage

```
playSIRS(
  data = NULL,
  Beta,
  gamma,
  psi,
  Delta,
  S0,
  I0,
  R0,
  t0 = 0,
  times,
  tree = FALSE,
  ill = FALSE
)

sirs_pomp(data, Beta, gamma, psi, Delta, S0, I0, R0, t0 = 0)
```

Arguments

data	data frame containing the genealogy in the format returned by newick2df.
Beta	transmission rate.
gamma	recovery rate.
psi	sampling rate.
Delta	waning rate of immunity.
S0	initial size of susceptible population.
I0	initial size of infected population.
R0	initial size of recovered population.
t0	initial time
times	times at which output is requested.
tree	logical; represent the genealogical tree in Newick format?
ill	logical; return an illustration?

Details

sirs_pomp constructs a **pomp** object containing a given set of data and a SIR model.

Value

A `tibble` with `state` attribute.

See Also

Other Genealogy processes: `lbdp`, `leventhal`, `moran`, `sirws`

`sirws` *SIR with sampling simulator.*

Description

Run the simulator.

Usage

```
playSIRwS(
  data = NULL,
  Beta,
  gamma,
  psi,
  S0,
  I0,
  R0,
  t0 = 0,
  times,
  tree = FALSE,
  ill = FALSE
)

sir_pomp(data, Beta, gamma, psi, S0, I0, R0, t0 = 0)
```

Arguments

`data`	data frame containing the genealogy in the format returned by `newick2df`.
`Beta`	transmission rate.
`gamma`	recovery rate.
`psi`	sampling rate.
`S0`	initial size of susceptible population.
`I0`	initial size of infected population.
`R0`	initial size of recovered population.
`t0`	initial time
`times`	times at which output is requested.
`tree`	logical; represent the genealogical tree in Newick format?
`ill`	logical; return an illustration?

Details

sir_pomp constructs a **pomp** object containing a given set of data and a SIR model.

Value

A tibble with state attribute.

See Also

Other Genealogy processes: lbdp, leventhal, moran, sirs

Examples

```
playSIRwS(Beta=2,gamma=1,psi=2,S0=1000,I0=5,R0=0,times=0:5,t0=0,tree=TRUE) -> x
playSIRwS(x,times=6:10,psi=1,tree=TRUE) -> x
plot(x)

playSIRwS(Beta=3,gamma=1,psi=2,S0=10,I0=5,R0=0,times=0:5,t0=-1,tree=TRUE) -> x
plot(x,points=TRUE)

y <- getInfo(x)
plot(y,points=TRUE)

library(ggplot2)
y$lineages |>
  ggplot(aes(x=time,y=lineages))+
  geom_step()
```

---

tableauGrob *Diagramming internals*

Description

Facilities to produce diagrammatic representations of genealogy process states.

Usage

```
tableauGrob(data, n, vp = NULL)

playerGrob(name, ballA, ballAcol, ballB, ballBcol, slate, vp = NULL)

resizingTextGrob(y, ..., vp = NULL)

## S3 method for class 'resizingTextGrob'
drawDetails(x, recording = TRUE)

## S3 method for class 'resizingTextGrob'
preDrawDetails(x)
```

```
## S3 method for class 'resizingTextGrob'
postDrawDetails(x)

ballGrob(y, label, color, ..., vp = NULL)

## S3 method for class 'ballGrob'
drawDetails(x, recording = TRUE)

## S3 method for class 'ballGrob'
preDrawDetails(x)

## S3 method for class 'ballGrob'
postDrawDetails(x)
```

Arguments

data	illustration vector
n	length of longest tableau
vp	a viewport object (or NULL).
name	player name
ballA, ballB	ball names
ballAcol, ballBcol	ball colors
slate	time
y	height
...	arguments to be passed to textGrob.
x	An R object.
recording	A logical value indicating whether a grob is being added to the display list or redrawn from the display list.
label	ball name
color	ball color

Details

Code for the resizing text adapted from a blog post by Mark Heckmann (https://ryouready.wordpress.com/2012/08/01/creating
a-text-grob-that-automatically-adjusts-to-viewport-size/).

`treeplot` *Fancy tree plotter*

Description

Plots a genealogical tree.

Usage

```
treeplot(
  tree,
  time = NULL,
  illus = NULL,
  root_time = 0,
  ladderize = TRUE,
  points = FALSE,
  diagram = FALSE
)
```

Arguments

`tree`	character; tree representation in Newick format.
`time`	numeric; times of the genealogies.
`illus`	character; genealogy process diagram information.
`root_time`	numeric; time of the root.
`ladderize`	Ladderize?
`points`	Show nodes and tips?
`diagram`	Show a diagram?

Value

A printable `ggtree` object.

Index