Package 'popdemog'

November 25, 2017

Type Package

Title Plot population demographic hisotry

Version 1.1

Date 2017-06-29

Author Ying Zhou

Maintainer Ying Zhou <yz001@uw.edu>

Description Plot demographic graph for single/multiple populations based on the demographic description from scripts of simulation tools.

License GPL-2

RoxygenNote 6.0.1

Suggests knitr, rmarkdown

VignetteBuilder knitr

R topics documented:

`popdemog-package` *Plot population demographic hisotry*

Description

Plot demographic graph for single/multiple populations based on the demographic description from scripts of simulation tools.

Details

This package contains three main functions to visualize the population demographic history from simulation commands. This package can support `ms`, `msa`, `msHot`, `MaCS`, `msprime`, and `cosi`. Please check the tutorial file for the update of supporting list. Users could copy the simulation script (for `ms(a)`, `msHot`, `MaCS`, and `msprime`(need a translation)) into one file, or into a R string, or just use the parameter file (for `Cosi`) as the input command file. Each input should only contain one demographic model.

Once the command file is prepared, function `PlotMS` could be used to generate the demographic graph or to give the parameters to `PlotMMig` plot the multiple migrations. More adjustments on migration graph would be achieved by using function `PlotMig`, which is allowed to add the migration patterns to other plots, such as geological map.

Author(s)

Ying Zhou

Maintainer: Ying Zhou <yz001@uw.edu>

References

ms: Hudson, R. R. "Generating Samples under a Wright-Fisher Neutral Model of Genetic Variation." Bioinformatics 18.2 (2002): 337-38.

msHot: Hellenthal, G., and M. Stephens. "MsHOT: Modifying Hudson's Ms Simulator to Incorporate Crossover and Gene Conversion Hotspots." Bioinformatics 23.4 (2006): 520-21.

MaCS: Chen, G. K., P. Marjoram, and J. D. Wall. "Fast and Flexible Simulation of DNA Sequence Data." Genome Research 19.1 (2008): 136-42.

Cosi: Shlyakhter, Ilya, Pardis C. Sabeti, and Stephen F. Schaffner. "Cosi2: An Efficient Simulator of Exact and Approximate Coalescent with Selection | Bioinformatics | Oxford Academic." OUP Academic. Oxford University Press, 22 Aug. 2014.

msprime: Jerome Kelleher, Alison M Etheridge and Gilean McVean, "Efficient Coalescent Simulation and Genealogical Analysis for Large Sample Sizes", PLoS Comput Biol 12(5): e1004842. doi: 10.1371/journal.pcbi.1004842. 2016.

Examples

```
###Tennessen's standard model
cat("macs 2025 15000000 -i 10 -r 3.0e-04 -t 0.00069 -T -I 4 10 1006 1008 1 0
-n 4 0.205 -n 1 58.00274 -n 2 70.041 -n 3 187.55 -eg 0.9e-10 1 482.46
-eg 1.0e-10 2 570.18 -eg 1.1e-10 3 720.23 -em 1.2e-10 1 2 0.731
-em 1.3e-10 2 1 0.731 -em 1.4e-10 3 1 0.2281 -em 1.5e-10 1 3 0.2281
-em 1.6e-10 2 3 0.9094 -em 1.7e-10 3 2 0.9094 -eg 0.007 1 0
-en 0.007001 1 1.98 -eg 0.007002 2 89.7668 -eg 0.007003 3 113.3896
-eG 0.031456 0 -en 0.031457 2 0.1412 -en 0.031458 3 0.07579
-eM 0.031459 0 -ej 0.03146 3 2 -en 0.0314601 2 0.2546
-em 0.0314602 2 1 4.386 -em 0.0314603 1 2 4.386 -eM 0.0697669 0
-ej 0.069767 2 1 -en 0.0697671 1 1.98 -en 0.2025 1 1 -ej 0.9575923 4 1
-em 0.06765 2 4 32 -em 0.06840 2 4 0", file="std-model-Tennessen.cmd")
#plot the demographic graph
par(mfrow=c(1,2))
PlotMS(input.file="std-model-Tennessen.cmd", type="macs", N4=10000, pop.scale="log",
log.base  =50, inpos = c(1,4,7,9), time.scale = "log10year",
col.pop=c("brown", "blue", "gold3", "forestgreen"),
pops=c("AFR", "EUR", "ASIA", "ARC"),
cex.lab=1, cex.axis = 1, xlab="", length.arrow=0.1)
title("Demographic histoy")
PlotMS(input.file="std-model-Tennessen.cmd", type="macs", N4=10000,
time.scale = "log10year", plot.out=FALSE, demo.out=TRUE)->out;
PlotMig(time_pt=2, demograph_out=out$demograph_out,mig_par=out$mig_par,
col.pop=c("brown", "blue", "gold3", "forestgreen"), pop.scale="topology");
legend("topleft", legend=c("AFR", "EUR", "ASIA", "ARC"),
col=c("brown", "blue", "gold3", "forestgreen"), pch=20, bty="n")
title("Migrations at 200 years ago");unlink("std-model-Tennessen.cmd")


###Archaic introgrssion model
cat("./ms 44 1 -r 20000 50000000 -t 30000 -I 6 20 20 1 1 1 1 -en 0 1 1
-en 0 2 1 -en 0 3 1e-10 -en 0 4 1e-10 -en 0 5 1e-10 -en 0 6 1e-10
-es 0.0125 2 0.97 -en 0.02500025 7 0.25 -en 0.02500025 2 1 -ej 0.05 4 3
-ej 0.05 6 5 -en 0.05000025 3 0.25 -en 0.05000025 5 0.25 -ej 0.0500025 5 3
-en 0.050005 3 0.25 -ej 0.075 2 1 -en 0.0750025 1 1 -ej 0.1 7 3
-en 0.1000025 3 0.25 -ej 0.3 3 1 -en 0.3000025 1 1", file="test.1.ms.cmd")
PlotMS(input.file = "test.1.ms.cmd", type="ms", N4=10000,
time.scale = "kyear", length.arrow=0.1, inpos=c(1,2,5,4.5,5.5,6,3),
col.pop=c("brown", "blue", "forestgreen", rainbow(10)[6:9]));
unlink("test.1.ms.cmd")


###Migration model from ms.
cat("./ms 15 100 -t 3.0 -I 6 0 7 0 0 8 0 -m 1 2 2.5 -m 2 1 2.5 -m 2 3 2.5
-m 3 2 2.5 -m 4 5 2.5 -m 5 4 2.5 -m 5 6 2.5 -m 6 5 2.5 -em 2.0 3 4 2.5
-em 2.0 4 3 2.5", file="test.2.ms.cmd")
PlotMS(input.file = "test.2.ms.cmd", type="ms", N4=10000, col.pop="gray",
col.arrow="black", length.arrow=0.1, lwd.arrow=2);unlink("test.2.ms.cmd")
```

NRuler *add population size ruler*

Description

This function works like the function legend,

Usage

```
NRuler(x, y = NULL, Nsize, Nlab = N.szie, N4, pop.scale,
  linear.scale = 0.2, log.base = 10, ...)
```

Arguments

x, y	Position of the population size ruler, if y does not have numeric value, x will support the keywords inputs from the list "bottomright", "bottom", "bottomleft","left", "topleft", "top", "topright", "right" and "center".
Nsize	The population sizes of the ticks on the ruler
Nlab	The labels on the ticks of the ruler, default as the population sizes
N4	Scalar to scale real population size
pop.scale	A keyword to define the way to reflect population size to lineage width. It supports "log" and "linear".

`linear.scale`	A numeric number used to reflect population size linearly. This value will be applied when `pop.scale`="linear".
`log.base`	The base of logarithm. This value will be applied when `pop.scale`="log".
`...`	Additional arguments can be passed, such as `col`, `lwd`, `lty`.

`PlotMig` *Plot migration event(s) at a particular time*

Description

This function is used to plot migration events at a particular time point based on the output of `PlotMS` with `demo.out`=T. The parameters of `add` and `map.pos` allow the migration graph to be added to any other background. User can use `time_pt` or `event` to define the time to plot migration(s).

Usage

```
PlotMig(time_pt = NULL, event = 1, mig_par, demograph_out,
  pop.scale = mig_par$pop.scale, linear.scale = mig_par$linear.scale,
  log.base = mig_par$log.base, col.pop = mig_par$col.pop,
  col.arrow = mig_par$col.arrow, xlim = mig_par$xlim, ylim = mig_par$ylim,
  lwd.arrow = mig_par$lwd.arrow, length.arrow = mig_par$length.arrow,
  angle.arrow = mig_par$angle.arrow, topology.scale = 1, add = FALSE,
  map.pos = NULL, m.adjust = 0)
```

Arguments

`time_pt`	A numeric value define at which time to plot migration(s). time_pt should be in the same scale as the setting of `time.scale`. For example, `time_pt` = 3 when `time.scale` = "log10year" gives us the migrations at 10^3 years ago.
`event`	An index to define at which time to plot migration(s) out. Every demographic event has an index in the order of time. Any demographic changes at the same time is defined as the same event and share the same index.
`mig_par`	A list contained all settings for plot demographic graph, refer `PlotMS`.
`demograph_out`	A list contained all demographic information, refer `PlotMS`.
`pop.scale`	A keyword to define the way to adjust the lineage width. "topology" returns only topology structure among simulated populations, ignoring either the population sizes and the time duration difference between any demographic events. "linear" linearly reflects the population sizes to lineage widths, the scale magnitude is defined in variable `linear.scale`; "log" is to use logarithm function to reflect the population sizes, the base is defined in `log.base`.
`linear.scale`	Linear scale magnitude, be applied when `pop.scale`="linear".
`log.base`	The base of logarithm, be applied when `pop.scale`="log".
`col.pop`	Population lineage color.
`col.arrow`	Migration arrow color.
`xlim`	The range of x-axis.
`ylim`	The range of y-axis.
`lwd.arrow`	The line width of arrow, defined by 0.5+migration strength*`lwd.arrow`.

`length.arrow`	The size of arrow end
`angle.arrow`	The angle of the arrow end, between 0 and 90.
`topology.scale`	Used for scale the size of circle when the `pop.scale=` "topology".
`add`	A logical value allowing to add the migrations to other background (2D only). Positions for every population dime should be well defined in `map.pos`.
`map.pos`	A matrix, each row is the position for each population.
`m.adjust`	Migration threshold for plotting the migration events. Migration event with strength higher than `m.adjust` will be shown. Default value is 0.

See Also

`PlotMS`

Examples

```
cat("./ms 15 100 -t 3.0 -I 6 0 7 0 0 8 0 -m 1 2 2.5 -m 2 1 2.5 -m 2 3 2.5
-m 3 2 2.5 -m 4 5 2.5 -m 5 4 2.5 -m 5 6 2.5 -m 6 5 2.5 -em 2.0 3 4 2.5
-em 2.0 4 3 2.5", file = "test.mig.cmd")
out <- PlotMS(input.file = "test.mig.cmd", type = "ms", N4 = 10000, plot.out = FALSE, demo.out = TRUE);
#check all migration events
events <- out$mig_par$events
print(events)
#check the time for those migration events
timeofevents <- out$mig_par$time[events]
print(timeofevents)
#plot event by event
par(mfrow = c(1, 2))
PlotMig(event = 1, demograph_out = out$demograph_out, mig_par = out$mig_par)
title("Event-1");
PlotMig(event = 2, demograph_out = out$demograph_out, mig_par = out$mig_par,
col.pop = 1:6, xlim = c(-5,4))
title("Event-2", cex.main = 3);
legend("topleft", col = 1:6, pch = 20, bty = "n", cex = 2,
legend = c("pop-1", "pop-2", "pop-3", "pop-4", "pop-5", "pop-6"))
unlink("test.mig.cmd")
```

`PlotMMig` *Plot Multiple Migrations*

Description

This function is used to plot all the migration evens. It based on the output of `PlotMS` with `demo.out`=FALSE. Plot settings should be customized in `PlotMS`. Use function `PlotMig` to customize the plot of single migration.

Usage

```
PlotMMig(demograph_out, mig_par, m.adjust = 0)
```

Arguments

`demograph_out`	A list of all demographic information, more descriptions will be found in the return value description of `PlotMS`.
`mig_par`	A list of all settings for plotting demographic graph, more descriptions will be found in the return value description of `PlotMS`.
`m.adjust`	Migration threshold for plotting the migration events. Migration event with strength higher than `m.adjust` will be shown. Default value is 0.

See Also

`PlotMS`

Examples

```
cat("./ms 15 100 -t 3.0 -I 6 0 7 0 0 8 0 -m 1 2 2.5 -m 2 1 2.5 -m 2 3 2.5
-m 3 2 2.5 -m 4 5 2.5 -m 5 4 2.5 -m 5 6 2.5 -m 6 5 2.5 -em 2.0 3 4 2.5
-em 2.0 4 3 2.5", file = "test.mig.cmd")
out<-PlotMS(input.file = "test.mig.cmd", type = "ms", N4 = 10000,
plot.out = FALSE, demo.out = TRUE, col.pop=1:6, cex.lab=1.5);
PlotMMig(demograph_out = out$demograph_out, mig_par = out$mig_par)
unlink("test.mig.cmd")
```

`PlotMS` *Plot population demographic graph and generate demographic parameters*

Description

This is the main function to plot demographic graph for single/multiple populations, which is named after Hudson's `ms`. It can read the simulation script from `ms`, `msa`, `msHot`, `MaCS`, `scrm`, and `cosi`.

The `input.file` or `imput.cmd` and command `type` are required to give a quick plot of demographic history. Principally, every component in the output graph could be directly customized according to need.

In the demographic graph, each population has a lineage vertically stretch back in time (`inpos` for positions, `col.pop` for the colors, `pops` for the names, and `time.scale` for the time axis). The width of the lineage reflects the relative population size (`pop.scale` for lineage widths). Population splits and migrations are in arrows (`col.arrow`, `length.arrow`, `lwd.arrow`, `angle.arrow`).

Usage

```
PlotMS(input.cmd = NULL, input.file = NULL, type, inpos = NULL,
  N4 = 10000, pop.scale = "linear", linear.scale = 0.2, log.base = 10,
  time.scale = "4Ne", gen = 25, m.adjust = 0, col.pop = "gray45",
  col.arrow = col.pop, length.arrow = 0.15, lwd.arrow = 1,
  angle.arrow = 15, pops = NULL, xlab = "Population",
  ylab = paste("Time before present (", time.scale, ")", sep = ""),
  xlim = NULL, ylim = NULL, plot.out = T, demo.out = F, cex.lab = 1,
  cex.axis = 1, axes = T)
```

Arguments

`input.cmd`	A string input for demographic events. User can paste simulation scripts for the convenient of debugging.
`input.file`	A file for demographic events. It can be simulation scripts (for `ms`, `msa`, `MaCS`) or parameter files (for `cosi`).
`type`	A keyword indicates the type of simulation command, currently supports "ms" for `ms`, "msa" for `msa`, "macs" for `MaCS`, "scrm" for `scrm`, "cosi" for `cosi`, "msprime" for msprime. Please check tutorial file to see more support simulators.
`inpos`	Population positions at time 0.
`N4`	A numeric for the effective population size of 4Ne, exact the same definition as 4N0 in the well-known simulation tool `ms`.
`pop.scale`	A keyword to define the way to adjust the lineage width. "topology" returns only topology structure among simulated populations, ignoring either the population sizes and the time duration difference between any demographic events. "linear" linearly reflects the population sizes to lineage widths, the scale magnitude is defined in variable `linear.scale`; "log" is to use logarithm function to reflect the population sizes, the base is defined in `log.base`.
`linear.scale`	Linear scale magnitude, be applied when `pop.scale`="linear".
`log.base`	The base of logarithm, be applied when `pop.scale`="log".
`time.scale`	A keyword to define the time unit used in the plot, it can be "4Ne", "generation", "year", "kyear", and "log10year". When the `pop.scale`="topology", this parameter will be ignored.
`gen`	Years per generation. Default value is 25.
`m.adjust`	Threshold for migration strength. Only arrows with migration strength bigger than `m.adjust` are plotted. This value should be stay between 0 and 1.
`col.pop`	Color for each population.
`col.arrow`	Color for each migration arrow
`length.arrow`	Size of arrow ends
`lwd.arrow`	Arrow width, which is determined as 0.5+migration strength*`lwd.arrow`.
`angle.arrow`	Arrow end angle, between 0 and 90.
`pops`	Population name labels. Default as 1:total population number.
`xlab`	Title for the x-axis
`ylab`	Title for the y-axis
`xlim`	Range of x-axis
`ylim`	Range of y-axis
`plot.out`	A logical variable to give out the demographic plot or not. If TRUE, the demographic plot will be given.
`demo.out`	A logical variable to output the demographic parameters. If TRUE,all demographic parameters that used for the graph will output as a list.
`cex.lab`	The magnification to be used for x and y labels relative to the current setting of `cex`.
`cex.axis`	The magnification to be used for axis annotation relative to the current setting of `cex`.
`axes`	A logical value to plot the axes or not.

Value

if the parameter `plot=F/FALSE`, following three lists will be returned:

`demograph_out`	This list contains all demographic details from the input command file: `time.series` is a vector of time; `Pos` is a numeric matrix of positions for each population at every demographic event; `N` is a numeric matrix of population size for each population at every demographic event; `m` is 3-D numeric matrix of migration rates between populations at every demographic event; `survive` is a matrix recording the begin and end for each population according to the demographic events; `g.rate` is a matrix of exponential growth rates at every demographic event; `pop.pos` is a numeric vector of the population positions at time 0; `pop.lab` is a vector of population names; `mscmd` is the `ms` command for demographic plot. All simulation script will turn to ms command for extract the demographic information; `present.pop.num` is the number of populations at present; `total.pop.num` is the number of total populations exist in the plot; `N4` is 4Ne, please refers to Hudson's `ms` for more description; `gen` is the years per generation.
`evo_par`	This list contains all parameters used to draw the demographic graph, including: `pop.scale` define the way to scale population size; `linear.scale` is used for scale population size linearly; `log.base` is the base of logarithm of the population size; `time.scale` defines the time unit used in the plot; `time` is the scaled time vector according to the `time.scale`; `col.pop` specifies the plotting color for each population; `col.arrow` specifies the plotting color for each migration arrow; `length.arrow` is the size of arrow ends; `lwd.arrow` is the width of arrow; `angle.arrow` is the angle of the arrow end; `lab.pop` is a vector of population names; `lab.pos` is a vector of positions to plot the population names; `xlim` is the range of x-axis; `ylim` is the range of y-axis; `xlab` is the title of x-axis; `ylab` it the title of y-axis; `cex.lab` is the magnification to be used for x and y labels; `cex.axis` is the magnification to be used for axis annotation; `axes` is a logical value to plot the axes or not.
`mig_par`	This list contains all parameters used to draw the migrations, including: `pop.scale` defines the way to scale population size; `linear.scale` is used for scaling population size linearly; `log.base` is the base of logarithm of the population size; `time.scale` defines the time unit used in the plot; `time` is the scaled time vector according to the `time.scale`; `lab.pop` is a vector of population names; `col.pop` specifies the plotting color for each population; `col.arrow` specifies the plotting color for each migration arrow; `length.arrow` is the size of arrow ends; `lwd.arrow` is the width of arrow; `angle.arrow` is the angle of the arrow end; `xlim` specifies the range of x-axis; `ylim` specifies the range of y-axis; `mfrow` sets the layout of multiple migrations in one plot; `events` records the events when migration state changes; `cex.lab` is the magnification to be used for x and y labels;

References

4Ne: `http://home.uchicago.edu/rhudson1/source/mksamples.html`

Examples

```
#example 1
cat("./ms 44 1 -r 20000 50000000 -t 30000 -I 6 20 20 1 1 1 1 -en 0 1 1
-en 0 2 1 -en 0 3 1e-10 -en 0 4 1e-10 -en 0 5 1e-10 -en 0 6 1e-10
-es 0.0125 2 0.97 -en 0.02500025 7 0.25 -en 0.02500025 2 1 -ej 0.05 4 3
-ej 0.05 6 5 -en 0.05000025 3 0.25 -en 0.05000025 5 0.25 -ej 0.0500025 5 3
-en 0.050005 3 0.25 -ej 0.075 2 1 -en 0.0750025 1 1 -ej 0.1 7 3
-en 0.1000025 3 0.25 -ej 0.3 3 1 -en 0.3000025 1 1", file="test.1.ms.cmd")
PlotMS(input.file = "test.1.ms.cmd", type="ms", N4=10000, time.scale = "kyear")
#adjust the population position
PlotMS(input.file = "test.1.ms.cmd", type="ms", N4=10000, time.scale = "kyear",
inpos=c(1,2,5,4,6,7,3))
#add color for each population
PlotMS(input.file = "test.1.ms.cmd", type="ms", N4=10000, time.scale = "kyear",
inpos=c(1,2,5,4,6,7,3), col.pop=rainbow(10)[3:9])
#add population names
unlink("test.1.ms.cmd")

#example 2
cat("./ms 15 100 -t 3.0 -I 6 0 7 0 0 8 0 -m 1 2 2.5 -m 2 1 2.5 -m 2 3 2.5
-m 3 2 2.5 -m 4 5 2.5 -m 5 4 2.5 -m 5 6 2.5 -m 6 5 2.5 -em 2.0 3 4 2.5
-em 2.0 4 3 2.5", file="test.2.ms.cmd")
PlotMS(input.file = "test.2.ms.cmd", type="ms", N4=10000, col.pop=3,
col.arrow=1, lwd.arrow=1.5)
unlink("test.2.ms.cmd")

#example 3
cat("./ms 1 1 -t 1.0 -I 3 10 10 10 -n 1 1.682020 -n 2 3.736830
-n 3 7.292050 -eg 0 2 116.010723 -eg 0 3 160.246047
-ma 0 0.881098 0.561966 0.881098 0 2.797460 0.561966 2.797460 0
-ej 0.028985 3 2 -en 0.028985 2 0.287184
-ema 0.028985 3 0 7.293140 0 7.293140 0 0 0 0 0 -ej 0.197963 2 1
-en 0.303501 1 1", file="Ryan2009.cmd")
PlotMS(input.file = "Ryan2009.cmd", type="ms", N4=10000, pop.scale="log",
log.base=200, pops=c("AFR", "EUR", "ESA"),
col.pop=c("brown", "blue", "yellow"));
unlink("Ryan2009.cmd")

#example 4
cat("ms 4 1 -t 7156.0000000 -r 2000.0000 10000000 -eN 0 5
-eG 0.000582262 1318.18 -eG 0.00232905 -329.546 -eG 0.00931619 82.3865
-eG 0.0372648 -20.5966 -eG 0.149059 5.14916 -eN 0.596236 0.5 -T",
file="zigzag.cmd")
par(mfrow=c(1,2))
PlotMS(input.file = "zigzag.cmd", type="ms", N4=10000)
#change the time unit
PlotMS(input.file = "zigzag.cmd", type="ms", N4=10000, time.scale="log10year")
unlink("zigzag.cmd")
```

Index