Package 'evolCCM'

September 29, 2021

Type Package

Title R package for Community Coevolution Model

Version 0.1.0

Author Chaoyue Liu

Maintainer Chaoyue Liu <Chaoyue.Liu@dal.ca>

Description This is the R implementation for CCM (Community Coevolution Model) from the manuscript ``The Community Coevolution Model with Application to the Study of Evolutionary Relationships between Genes based on Phylogenetic Profiles". It includes the main functions for profiles simulation, profiles visualization and CCM estimation.

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.1.2

R topics documented:

`EstimateCCM` *Estimate the parameters in CCM*

Description

The function for estimating the parameters in CCM.

Usage

```
EstimateCCM = function(profiles, phytree, ip=0.1, pen=0.5, ... )
```

Arguments

profiles	a matrix containing the profiles. Columns are profiles and rows are species.
phytree	a phylogenetic tree.
ip	the initial values for optimizer. For a better convergence, a good set of starting values could be the estimates by a large tuning parameter.
pen	the tuning parameter λ. Default value is 0.5. Value of 0 means no regularization.
...	control parameters available to optimizer 'nlminb' such as 'trace', 'rel.tol', ...Example See '?nlminb', 'control' argument .

Value

a list with following elements

- alpha: estimated intrinsic rates.
- B: estimated association matrix.
- nlm.par: all estimated parameters in the order as 'c(alpha, diag(B), B[upper.tri(B)])'.
- nlm.converge: convergence message. See '?nlminb'.
- nlm.hessian: estimated Hessian matrix used for calculating standard errors.

References

Paradis E, Schliep K (2019). "ape 5.0: an environment for modern phylogenetics and evolutionary analyses in R." Bioinformatics, 35, 526-528.

David M. Gay (1990), Usage summary for selected optimization routines. Computing Science Technical Report 153, AT&T Bell Laboratories, Murray Hill.

Examples

```
set.seed(123)
# generate a ranom 200-tip tree.
# Larger tree contains more information and tends to give better MLEs.
t <- rtree(200)
# setting arbatrunderlying parameters for a pair
n <- 2
alpha <- c(0.1, 0.1)
B <- matrix(0, n, n)
B[1,2] <- B[2,1] <- 1
diag(B) <- c(-0.3, 0.3)

# using the same set of parameter to simulate 20 pairs
# and estimate with CCM.

trueP = c(alpha, diag(B), B[upper.tri(B)]) # true parameters
estP = matrix(NA, nrow=20, ncol=length(trueP))
for (i in 1:20){
    simDF <- SimulateProfiles(t, alpha, B)
    aE <- EstimateCCM(profiles = simDF, phytree=t)
    estP[i,] = c(aE$alpha, diag(aE$B), aE$B[upper.tri(aE$B)])
    # or estP[i,] = aE$par same order as above
    print(i)
}
```

```
# plot the estimates
boxplot(estP)
points(1:length(trueP), trueP, pch=8, col="red")
```

ProcessAE *Process the output of 'EstimateCCM'*

Description

Calculate the convergence rate and standard errors.

Usage

```
ProcessAE(aE)
```

Arguments

aE	a list returned by 'EstimateCCM()'.

Value

a list with following elements:

- rate: rate of converge. If value is too large (e.g. >300), consider increasing the tuning parameter λ.
- hessianSE: estimated standard errors using Hessian matrix.

Examples

```
set.seed(123)
t <- rtree(100)
d <- TreeToDend(t)

# setting random parameters for a pair without interaction
n <- 2
alpha <- runif(n, -0.1, 0.1)
B <- matrix(0, n, n)
diag(B) <- runif(n, -0.1, 0.1)
B[1,2] <- B[2,1] <- 0 # independent pair

simDF <- SimulateProfiles(t, alpha, B)
ProfilePlot(simDF, d) # plot the profiles
aE <- EstimateCCM(profiles = simDF, phytree=t)
estSE <- ProcessAE(aE)$hessianSE
# testing if there is significant interaction
# p value for Ha: \eqn{\beta != 0}
sigScore <- aE$nlm.par[5] / estSE[5]
print(2*(1 - pnorm(abs(sigScore))))

# simulate a pair with interaction
B[1,2]<-B[2,1] <- 0.5 # moderate strength
simDF <- SimulateProfiles(t, alpha, B)
ProfilePlot(simDF, d)
```

```
aE <- EstimateCCM(profiles = simDF, phytree=t)
estSE <- ProcessAE(aE)$hessianSE
# testing if there is significant interaction
# p value for Ha: \eqn{\beta != 0}
sigScore <- aE$nlm.par[5] / estSE[5]
print(2*(1 - pnorm(abs(sigScore))))
```

`ProfilePlot` *Plot the phylogenetic profiles*

Description

Plot the phylogenetic profiles along with the phylogenetic tree. This function is based on 'heatmap.2()' from 'gplots' package.

Usage

```
ProfilePlot(profile, dend, ...)
```

Arguments

`profile`	a data frame or matrix contains the phylogenetic profiles. Each column is a profile. The row names must match the tip labels of the tree.
`dend`	the dendrogram that will be plotted on the left side. See 'TreeToDen()'.
`...`	some other parameters available for 'heatmap.2()', such as 'main', 'xlab', 'labRow'...

Examples

```
d = TreeToDend(rtree(50))
p = matrix(sample(0:1,100, replace=T), nrow=50)
rownames(p) = labels(d)
ProfilePlot(p, d, main="Plot of two profiles")
```

`SimulateProfiles` *Simulate profiles*

Description

Simulate binary profiles based on the Community Coevolution Model using the input tree and user-defined parameters.

Usage

```
SimulateProfiles(phytree, alpha, B, root=F)
```

Arguments

phytree	a phylogenetic tree.
alpha	a vector of the intrinsic rates (α) in CCM. The length of the vector decides how many profiles to simulate.
B	a symmetric association matrix (B) in CCM. If the input matrix is not symmetric, the upper triangle of the matrix will be used.
root	a vector of states at root node. 'root=F' as default (random states will be assigned at root).

Value

a matrix containing simulated profiles.

Examples

```
set.seed(123)
# generate a random 50-tip tree
t <- rtree(50)
# simulate 5 profiles
n <- 5
# assigning parameter values.
# The parameters should be in a reasonable scale otherwise the simulated profiles may have all 0s or 1s.
alpha =runif(n, -0.5, 2) # assign n random intrinsic rates.
association matrix B
B <- matrix(0,n,n)
B[1:3,1:3] <- 1 # first 3 genes are correlated. The other 2 have no interaction with others
diag(B) <- runif(n, -0.5,0.5) # assign half the difference between gain and loss rates for each gene
# simulate 5 random profiles
simDF <- SimulateProfiles(t, alpha, B)
# plot the profiles
d <- TreeToDend(t) # convert the tree to dendrogram
ProfilePlot(simDF, d)
```

TreeToDend *Convert tree to dendrogram*

Description

Convert the non-ultrametric tree to a dendrogram for 'ProfilePlot()' to plot the profiles using the 'chronos()' function from 'ape' package.

Usage

```
TreeToDend(phytree)
```

Arguments

tree	an object of class "phylo"

Value

a dendrogram as an input for 'ProfilePlot()' function

Examples

```
#library(ape)
# create a random 50-tip tree
t = rtree(50)
plot(t)
d = TreeToDend(t)
plot(d)
```

Index