Package: multispatialCCM (via r-universe)

September 8, 2024

Title Multispatial Convergent Cross Mapping

Version 1.3

Date 2023-10-22

Author Adam Clark

Maintainer Adam Clark <adam.tclark@gmail.com>

Depends R (>= 3.0.2)

Description The multispatial convergent cross mapping algorithm can be used as a test for causal associations between pairs of processes represented by time series. This is a combination of convergent cross mapping (CCM), described in Sugihara et al., 2012, Science, 338, 496-500, and dew-drop regression, described in Hsieh et al., 2008, American Naturalist, 171, 71–80. The algorithm allows CCM to be implemented on data that are not from a single long time series. Instead, data can come from many short time series, which are stitched together using bootstrapping.

License GPL (>= 2)

Encoding UTF-8

NeedsCompilation yes

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

RemoteUrl https://github.com/adamtclark/multispatialccm

RemoteRef HEAD

RemoteSha 6d318264b9cb20c4fdcbada98c46f600f9cbdaac

Contents

---

multispatialCCM-package

multispatial convergent cross mapping analysis

Description

A package for conducting convergent cross mapping (CCM) tests for causal relations for short, replicated time series. This package contains functions for implementing CCM on replicated data, as well as testing for significance of causal relationships that are detected. This package can also be used to implement state space reconstruction on replicated data (dew-drop regression, Hsieh et al.)

Details

The functions in this package can analyze data from multiple replicated time series. To enter the time series into functions, they should be concatenated back-to-back into a single vector, with gaps between time series indicated by "NA".

The package follows the descriptions in Clark et al. There are two classes of functions: analytical, and diagnostic. Both of these should be used for conducting multispatial CCM.

Analytical: CCM_boot is used to conduct multispatial CCM. SSR_pred_boot is used to find the optimal embedding dimension E for the CCM analysis, and can also be used to implement a state space reconstruction algorithm to predict process dynamics based on another process, or on its own historical dynamics.

Diagnostic: SSR_check_signal is used to test whether dynamics from a time series are sufficiently nonlinear for application of CCM, and tests that they are not too dominated by noise. The ccmtest function is used to test the output from boot_CCM to determine whether it indicates a significant causal link between variables.

Finally, the package also includes the function make_ccm_data, which can be used to make fake data from a simulation of two competing species. This function is used in the examples to show how to utilize the multispatial CCM functions.

Author(s)

Adam Clark

Maintainer: Adam Clark <atclark@umn.edu> ~~ The author and/or maintainer of the package ~~

References

Sugihara, G., R. May, H. Ye, C. Hsieh, E. Deyle, M. Fogarty, and S. Munch. 2012. Detecting Causality in Complex Ecosystems. Science 338.

Hsieh, C., C. Anderson, and G. Sugihara. 2008. Extending Nonlinear analysis to short ecological time series. American Naturalist 171:71–80.

Adam T. Clark, H. Ye, Forest Isbell, Ethan R. Deyle, Jane Cowles, David Tilman, and George Sugihara. 2015. Spatial 'convergent cross mapping' to detect causal relationships from short time-series. Ecology, 96(6):1174–1181.

See Also

CCM_boot, SSR_pred_boot, SSR_check_signal, ccmtest

Examples

```
#Simulate data to use for multispatial CCM test
#See function for details - A is causally forced by B,
#but the reverse is not true.
ccm_data_out<-make_ccm_data()
Accm<-ccm_data_out$Accm
Bccm<-ccm_data_out$Bccm

#Calculate optimal E
maxE<-5 #Maximum E to test
#Matrix for storing output
Emat<-matrix(nrow=maxE-1, ncol=2); colnames(Emat)<-c("A", "B")

#Loop over potential E values and calculate predictive ability
#of each process for its own dynamics
for(E in 2:maxE) {
  #Uses defaults of looking forward one prediction step (predstep)
  #And using time lag intervals of one time step (tau)
  Emat[E-1,"A"]<-SSR_pred_boot(A=Accm, E=E, predstep=1, tau=1)$rho
  Emat[E-1,"B"]<-SSR_pred_boot(A=Bccm, E=E, predstep=1, tau=1)$rho
}

#Look at plots to find E for each process at which
#predictive ability rho is maximized
matplot(2:maxE, Emat, type="l", col=1:2, lty=1:2,
        xlab="E", ylab="rho", lwd=2)
legend("bottomleft", c("A", "B"), lty=1:2, col=1:2, lwd=2, bty="n")

#Results will vary depending on simulation.
#Using the seed we provide,
#maximum E for A should be 2, and maximum E for B should be 3.
#For the analyses in the paper, we use E=2 for all simulations.

E_A<-2
E_B<-3

#Check data for nonlinear signal that is not dominated by noise
#Checks whether predictive ability of processes declines with
#increasing time distance
#See manuscript and R code for details
signal_A_out<-SSR_check_signal(A=Accm, E=E_A, tau=1,
  predsteplist=1:10)
signal_B_out<-SSR_check_signal(A=Bccm, E=E_B, tau=1,
  predsteplist=1:10)

#Run the CCM test
#E_A and E_B are the embedding dimensions for A and B.
#tau is the length of time steps used (default is 1)
```

```
#iterations is the number of bootsrap iterations (default 100)
# Does A "cause" B?
#Note - increase iterations to 100 for consistant results
CCM_boot_A<-CCM_boot(Accm, Bccm, E_A, tau=1, iterations=10)
# Does B "cause" A?
CCM_boot_B<-CCM_boot(Bccm, Accm, E_B, tau=1, iterations=10)

#Test for significant causal signal
#See R function for details
(CCM_significance_test<-ccmtest(CCM_boot_A,
                   CCM_boot_B))

#Plot results
plotxlimits<-range(c(CCM_boot_A$Lobs, CCM_boot_B$Lobs))

#Plot "A causes B"
plot(CCM_boot_A$Lobs, CCM_boot_A$rho, type="l", col=1, lwd=2,
     xlim=c(plotxlimits[1], plotxlimits[2]), ylim=c(0,1),
     xlab="L", ylab="rho")
#Add +/- 1 standard error
matlines(CCM_boot_A$Lobs,
cbind(CCM_boot_A$rho-CCM_boot_A$sdevrho,
CCM_boot_A$rho+CCM_boot_A$sdevrho),
lty=3, col=1)

#Plot "B causes A"
lines(CCM_boot_B$Lobs, CCM_boot_B$rho, type="l", col=2, lty=2, lwd=2)
#Add +/- 1 standard error
matlines(CCM_boot_B$Lobs,
cbind(CCM_boot_B$rho-CCM_boot_B$sdevrho,
CCM_boot_B$rho+CCM_boot_B$sdevrho),
lty=3, col=2)

legend("topleft",
c("A causes B", "B causes A"),
lty=c(1,2), col=c(1,2), lwd=2, bty="n")
```

---

ccmtest   *Test for significant causal signal*

Description

Tests output from CCM_boot for significant causal signal. This function compares the 95% confidence intervals for esimated rho for the shortest and longest libraries calculated, and uses this to determine whether predictive power has significantly increased.

Usage

```
ccmtest(CCM_boot_A, CCM_boot_B)
```

Arguments

`CCM_boot_A`	Output structure from a CCM test using CCM_boot
`CCM_boot_B`	Output structure from a CCM test using CCM_boot

Value

`res`	Structure containing the p-values for both tests.

Author(s)

Adam Clark

References

Sugihara, G., R. May, H. Ye, C. Hsieh, E. Deyle, M. Fogarty, and S. Munch. 2012. Detecting Causality in Complex Ecosystems. Science 338.

Adam T. Clark, H. Ye, Forest Isbell, Ethan R. Deyle, Jane Cowles, David Tilman, and George Sugihara. 2015. Spatial 'convergent cross mapping' to detect causal relationships from short time-series. Ecology, 96(6):1174–1181.

See Also

CCM_boot, SSR_pred_boot, SSR_check_signal

Examples

```
#Simulate data to use for multispatial CCM test
#See function for details - A is causally forced by B,
#but the reverse is not true.
ccm_data_out<-make_ccm_data()
Accm<-ccm_data_out$Accm
Bccm<-ccm_data_out$Bccm

#Set optimal E - see multispatialCCM for details
E_A<-2
E_B<-3

#Run the CCM test
#E_A and E_B are the embedding dimensions for A and B.
#tau is the length of time steps used (default is 1)
#iterations is the number of bootsrap iterations (default 100)
# Does A "cause" B?
CCM_boot_A<-CCM_boot(Accm, Bccm, E_A, tau=1, iterations=10)
# Does B "cause" A?
CCM_boot_B<-CCM_boot(Bccm, Accm, E_B, tau=1, iterations=10)

(CCM_significance_test<-ccmtest(CCM_boot_A,
                  CCM_boot_B))
```

| CCM_boot | *Run multispatial CCM algorithm on two time series* |

Description

Runs the multispatial convergent cross mapping algorithm on two time series, A and B, to determine whether process A is a forcing process (i.e., causally affects) or process B. A and B do not need to be from single, long time series, but rather can be combinations of many (e.g. spatially-replicated) time series. See "Arguments" for details.

Usage

```
CCM_boot(A, B, E, tau=1,
DesiredL=((tau*(E-1)+(E+1)):length(A)-E+2),
iterations=100)
```

Arguments

A	Time series that is being tested as a forcing process (i.e., the causal process). Should be a single vector. If data come from multiple time series, gaps between these should be marked with an "NA". E.g., c(1,2,3, NA, 1,2,3) implies two time series, each of length 3. Order of plots does not matter (because they will be shuffled during bootstrapping), but should match the order used in B.
B	Time series that is being tested as a response process (i.e., the process being affected by the causal process). Should be a single vector. If data come from multiple time series, gaps between these should be marked with an "NA". E.g., c(1,2,3, NA, 1,2,3) implies two time series, each of length 3. Order of plots does not matter (because they will be shuffled during bootstrapping), but should match the order used in A.
E	Embedding dimension to use for the analysis. Should be based on dimension that provides the best prediction of process A against itself using function "SSR_pred_boot" (state space reconstruction).
tau	Number of time steps to use for lagged components in the attractor space. Defaults to 1.
DesiredL	Desired library lengths for which to compute CCM. Defaults to the maximum possible length ((tau * (E - 1) + (E + 1)):length(A) - E + 2) (though number of resulting predictions may be smaller because of gaps in the time series). Shortening this list (e.g., only predicting every nth element) will reduce run-time for the algorithm, but may also reduce ability to detect causal relations.
iterations	Number of iterations for bootstrapping. Defaults to 100.

Value

A	Input process A
Aest	Estimated values of A from B using CCM, for the last (longest) library length considered

B	Input process B
rho	Pearson correlation coefficient of estimates of A from B for each library length tested
varrho	Variance of rho based on bootstrapping for each library length
sdevrho	Standard error (i.e., sqrt(var/n)) of the mean of rho for each library length
Lobs	Library lengths for which rho was calculated
E	Embedding dimension used for the analysis
tau	Time lag length used for the analysis
FULLinfo	Includes all output from .C call. Much of this is redundant with the variables listed above. Best not to look at this.

Warning

If you do not separate distinct time series with "NA" as described in "Arguments", they will not be treated as such!

Author(s)

Adam Clark

References

Sugihara, G., R. May, H. Ye, C. Hsieh, E. Deyle, M. Fogarty, and S. Munch. 2012. Detecting Causality in Complex Ecosystems. Science 338.

Adam T. Clark, H. Ye, Forest Isbell, Ethan R. Deyle, Jane Cowles, David Tilman, and George Sugihara. 2015. Spatial 'convergent cross mapping' to detect causal relationships from short time-series. Ecology, 96(6):1174–1181.

See Also

SSR_pred_boot, SSR_check_signal, ccmtest

Examples

```
#Simulate data to use for multispatial CCM test
#See function for details - A is causally forced by B,
#but the reverse is not true.
ccm_data_out<-make_ccm_data()
Accm<-ccm_data_out$Accm
Bccm<-ccm_data_out$Bccm

#Set optimal E - see multispatialCCM for details
E_A<-2
E_B<-3

#Run the CCM test
#E_A and E_B are the embedding dimensions for A and B.
#tau is the length of time steps used (default is 1)
```

```
#iterations is the number of bootsrap iterations (default 100)
# Does A "cause" B?
CCM_boot_A<-CCM_boot(Accm, Bccm, E_A, tau=1, iterations=10)
# Does B "cause" A?
CCM_boot_B<-CCM_boot(Bccm, Accm, E_B, tau=1, iterations=10)
```

CCM_bootstrap   *Run multispatial CCM algorithm on two time series*

Description

Internal C function used by the CCM_boot function.

---

make_ccm_data   *Makes fake data for other functions*

Description

Builds a fake data set of two interacting processes, based on the model in the Sugihara et al. publication below, and based on a two-species discrete-time competition model. In the model, process A is causally affected by process B, but process B is not influenced by process A.

Usage

```
make_ccm_data(sp_sd=0.125, obs_sd=0.025,
Sstr=0.375, times=10, burnin=100,
number_of_chains=20, seednum=2718)
```

Arguments

sp_sd	Standard deviation used to add process noise. If you are simulating multiple plots, this adds normally distributed noise with mean=0 to the growth rates of each species in different plots.
obs_sd	Standard deviation used to add observation error. Observation error is added to process X as a lognormal variable (X*rlnorm), with mean=0 on the lognormal scale.
Sstr	Forcing strength defining the effect of process B on process A.
times	Number of sequential observations desired for the time series in each plot (i.e., length of each independent time series)?
burnin	Burnin time before starting the experiment. This can be used to remove correlation among plots that occurs because of starting conditions.
number_of_chains	
	Total number of time series (i.e., how many replicates will be assembled into a single long time series?)
seednum	Random seed used for simulation.

Value

Accm	Time series for process A. Gaps between time series are indicated by a "NA" entry.
Bccm	Time series for process B. Gaps between time series are indicated by a "NA" entry.
time_ccm	Time indices corresponding to process A and B.

Author(s)

Adam Clark

References

Sugihara, G., R. May, H. Ye, C. Hsieh, E. Deyle, M. Fogarty, and S. Munch. 2012. Detecting Causality in Complex Ecosystems. Science 338.

Adam T. Clark, H. Ye, Forest Isbell, Ethan R. Deyle, Jane Cowles, David Tilman, and George Sugihara. 2015. Spatial 'convergent cross mapping' to detect causal relationships from short time-series. Ecology, 96(6):1174–1181.

See Also

CCM_boot, SSR_pred_boot, SSR_check_signal, ccmtest

Examples

```
#Simulate data to use for multispatial CCM test
#See function for details - A is causally forced by B,
#but the reverse is not true.
ccm_data_out<-make_ccm_data()
Accm<-ccm_data_out$Accm
Bccm<-ccm_data_out$Bccm
```

SSR_check_signal	*Test process for auto-predictability.*

Description

Predict elements of a process based historical observations of that process using cross-validation. Tests whether past observations are able to make good estimates of future elements of the time series.

Usage

```
SSR_check_signal(A, E, tau = 1,
predsteplist = 1:10, matchSugi = 0)
```

Arguments

`A`	Process to be predicted. Should be a single vector. If data come from multiple time series, gaps between these should be marked with an "NA".
`E`	Embedding dimension to use for the analysis. Should be based on dimension that provides the best prediction of process A against itself using function "SSR_pred_boot" (state space reconstruction).
`tau`	Number of time steps to use for lagged components in the attractor space. Defaults to 1.
`predsteplist`	Vector of time step lengths for prediction.
`matchSugi`	Set to 1 to match results in Sugihara et al. publication described below, which removes only point i in cross validation - if 0, then removes all points within X(t-(E-1)):X(t+1)

Value

predatout=predatout, rho_pre_slope=rho_pre_slope, rho_predmaxCI=rho_predmaxCI

`predatout`	Vector of rho values describing predictive ability of process against itself for each prediction time step length
`rho_pre_slope`	Slope of rho values as a function of prediction distance
`rho_predmaxCI`	95% confidence interval for rho value corresponding to the longest prediction interval tested

Author(s)

Adam Clark

References

Sugihara, G., R. May, H. Ye, C. Hsieh, E. Deyle, M. Fogarty, and S. Munch. 2012. Detecting Causality in Complex Ecosystems. Science 338.

Adam T. Clark, H. Ye, Forest Isbell, Ethan R. Deyle, Jane Cowles, David Tilman, and George Sugihara. 2015. Spatial 'convergent cross mapping' to detect causal relationships from short time-series. Ecology, 96(6):1174–1181.

See Also

CCM_boot, SSR_pred_boot, ccmtest

Examples

```
#Simulate data to use for multispatial CCM test
#See function for details - A is causally forced by B,
#but the reverse is not true.
ccm_data_out<-make_ccm_data()
Accm<-ccm_data_out$Accm
Bccm<-ccm_data_out$Bccm
```

```
#Set optimal E - see multispatialCCM for details
E_A<-2
E_B<-3

#Check data for nonlinear signal that is not dominated by noise
#Checks whether predictive ability of processes declines with
#increasing time distance
#See manuscript and R code for details
signal_A_out<-SSR_check_signal(A=Accm, E=E_A, tau=1,
  predsteplist=1:10)
signal_B_out<-SSR_check_signal(A=Bccm, E=E_B, tau=1,
  predsteplist=1:10)
```

SSR_pred_boot   *State space reconstruction function*

Description

Predict elements of A using B using state space reconstruction. If A=B, then the algorithm uses cross validation to assess the ability of historical portions of the A time series to predict future components of the time series. This function can be used to find the embedding dimension E that maximizes predictive ability.

Usage

```
SSR_pred_boot(A, B = A, E, tau = 1, predstep = 1, matchSugi = 0)
```

Arguments

A	Process to be compared to B, or to itself. Should be a single vector. If data come from multiple time series, gaps between these should be marked with an "NA".
B	Process to be compared to A. If left empty, algorithm defaults to A=B.
E	Embedding dimension to use for the analysis. Should be based on dimension that provides the best prediction of process A against itself using function "SSR_pred_boot" (state space reconstruction).
tau	Number of time steps to use for lagged components in the attractor space. Defaults to 1.
predstep	Number of time steps into the future to make predictions from past observations.
matchSugi	Set to 1 to match results in Sugihara et al. publication described below, which removes only point i in cross validation - if 0, then removes all points within X(t-(E-1)):X(t+1)

Value

A	Returns variable from input
Aest	Estimated values for A
B	Returns variable from input
E	Returns variable from input
tau	Returns variable from input
pAlength	Length of A from input
pBlength	Length of B from input
predstep	Returns variable from input
prepvec	Returns 1 if A and B were treated as same process
pmatchSugi	Returns variable from input
acceptablelib	List of library lengths that were used for the analysis, adjusting for ends and gaps in the library
plengthacceptablelib	
	Length of library that was used for the analysis
rho	Pearson correlation coefficient describing predictive ability of A against B or against itself

Author(s)

Adam Clark

References

Sugihara, G., R. May, H. Ye, C. Hsieh, E. Deyle, M. Fogarty, and S. Munch. 2012. Detecting Causality in Complex Ecosystems. Science 338.

Adam T. Clark, H. Ye, Forest Isbell, Ethan R. Deyle, Jane Cowles, David Tilman, and George Sugihara. 2015. Spatial 'convergent cross mapping' to detect causal relationships from short time-series. Ecology, 96(6):1174–1181.

See Also

CCM_boot, SSR_check_signal, ccmtest

Examples

```
#Simulate data to use for multispatial CCM test
#See function for details - A is causally forced by B,
#but the reverse is not true.
ccm_data_out<-make_ccm_data()
Accm<-ccm_data_out$Accm
Bccm<-ccm_data_out$Bccm

#Calculate optimal E
maxE<-5 #Maximum E to test
#Matrix for storing output
```

```
Emat<-matrix(nrow=maxE-1, ncol=2); colnames(Emat)<-c("A", "B")

#Loop over potential E values and calculate predictive ability
#of each process for its own dynamics
for(E in 2:maxE) {
  #Uses defaults of looking forward one prediction step (predstep)
  #And using time lag intervals of one time step (tau)
  Emat[E-1,"A"]<-SSR_pred_boot(A=Accm, E=E, predstep=1, tau=1)$rho
  Emat[E-1,"B"]<-SSR_pred_boot(A=Bccm, E=E, predstep=1, tau=1)$rho
}

#Look at plots to find E for each process at which
#predictive ability rho is maximized
matplot(2:maxE, Emat, type="l", col=1:2, lty=1:2,
        xlab="E", ylab="rho", lwd=2)
legend("bottomleft", c("A", "B"), lty=1:2, col=1:2, lwd=2, bty="n")

#Results will vary depending on simulation.
#Using the seed we provide,
#maximum E for A should be 2, and maximum E for B should be 3.
#For the analyses in the paper, we use E=2 for all simulations.
```

Index