Package ‘bayesRecon’

December 19, 2023

Type Package

Date 2023-12-19

Title Probabilistic Reconciliation via Conditioning

Version 0.2.0

Maintainer Dario Azzimonti <dario.azzimonti@gmail.com>

Description Provides methods for probabilistic reconciliation of hierarchical forecasts of time series. The available methods include analytical Gaussian reconciliation (Corani et al., 2021) <doi:10.1007/978-3-030-67664-3_13>, MCMC reconciliation of count time series (Corani et al., 2022) <doi:10.48550/arXiv.2207.09322>, Bottom-Up Importance Sampling (Zambon et al., 2022) <doi:10.48550/arXiv.2210.02286>.

License LGPL (>= 3)

Depends R (>= 4.1.0)

Imports stats, utils, lpSolve (>= 5.6.18)

Encoding UTF-8

LazyData true

RoxygenNote 7.2.3

Suggests knitr, rmarkdown, forecast, glarma, scoringRules, testthat (>= 3.0.0)

Config/testthat/edition 3

VignetteBuilder knitr

NeedsCompilation no

Author Dario Azzimonti [aut, cre] (<https://orcid.org/0000-0001-5080-3061>),
Nicolò Rubattu [aut] (<https://orcid.org/0000-0002-2703-1005>),
Lorenzo Zambon [aut] (<https://orcid.org/0000-0002-8939-993X>),
Giorgio Corani [aut] (<https://orcid.org/0000-0002-1541-8384>)

Repository CRAN

Date/Publication 2023-12-19 14:00:02 UTC

R topics documented:

`carparts_example` *Example of a time series from carparts*

Description

A monthly time series from the `carparts` dataset, 51 observations, Jan 1998 - Mar 2002.

Usage

```
carparts_example
```

Format

Univariate time series of class ts.

Source

Godahewa, Rakshitha, Bergmeir, Christoph, Webb, Geoff, Hyndman, Rob, & Montero-Manso, Pablo. (2020). Car Parts Dataset (without Missing Values) (Version 2) doi:10.5281/zenodo.4656021

References

Hyndman, R.J., Koehler, A.B., Ord, J.K., and Snyder, R.D., (2008) Forecasting with exponential smoothing: the state space approach, Springer

Godahewa, Rakshitha, Bergmeir, Christoph, Webb, Geoff, Hyndman, Rob, & Montero-Manso, Pablo. (2020). Car Parts Dataset (without Missing Values) (Version 2) doi:10.5281/zenodo.4656021 `extr_mkt_events` *Extreme market events dataset*

Description

Count time series of extreme market events in five economic sectors. The data refer to the trading days between 2004/12/31 and 2018/12/19 (3508 trading days in total).

Usage

```
extr_mkt_events
```

Format

A multivariate time series of class ts.

Details

The counts are computed by considering 29 companies included in the Euro Stoxx 50 index and observing if the value of the CDS spread on a given day exceeds the 90-th percentile of its distribution in the last trading year. The companies are divided in the following sectors: Financial (FIN), Information and Communication Technology (ICT), Manufacturing (MFG), Energy (ENG), and Trade (TRD).

There are 6 time series:

- 5 bottom time series, corresponding to the daily counts for each sector
- 1 upper time series, which is the sum of all the bottom (ALL)

Source

Zambon, L., Agosto, A., Giudici, P., Corani, G. (2023). *Properties of the reconciled distributions for Gaussian and count forecasts*. doi:10.48550/arXiv.2303.15135.

References

Zambon, L., Agosto, A., Giudici, P., Corani, G. (2023). *Properties of the reconciled distributions for Gaussian and count forecasts*. doi:10.48550/arXiv.2303.15135.

Agosto, A. (2022). *Multivariate Score-Driven Models for Count Time Series to Assess Financial Contagion*. doi:10.2139/ssrn.4119895

`extr_mkt_events_basefc` *Base forecasts for the extreme market events dataset*

Description

Base forecasts for the `extr_mkt_events` dataset, computed using the model by Agosto, A. (2022). *Multivariate Score-Driven Models for Count Time Series to Assess Financial Contagion*. doi:10.2139/ssrn.4119895.

Usage

```
extr_mkt_events_basefc
```

Format

A list `extr_mkt_events_basefc` containing

`extr_mkt_events_basefc$mu` data frame of the base forecast means, for each day

`extr_mkt_events_basefc$size` data frame of the static base forecast size parameters

Details

The predictive distribution for the bottom time series is a multivariate negative binomial with a static vector of dispersion parameters and a time-varying vector of location parameters following a score-driven dynamics. The base forecasts for the upper time series are computed using a univariate version of this model. They are in-sample forecasts: for each training instant, they are computed for time t+1 by conditioning on the counts observed up to time t.

Source

Agosto, A. (2022). *Multivariate Score-Driven Models for Count Time Series to Assess Financial Contagion*. doi:10.2139/ssrn.4119895

References

Agosto, A. (2022). *Multivariate Score-Driven Models for Count Time Series to Assess Financial Contagion*. doi:10.2139/ssrn.4119895

Zambon, L., Agosto, A., Giudici, P., Corani, G. (2023). *Properties of the reconciled distributions for Gaussian and count forecasts*. doi:10.48550/arXiv.2303.15135.

get_reconc_matrices *Build hierarchy matrices*

Description

Creates the aggregation and summing matrices for a temporal hierarchy of time series from a user-selected list of aggregation levels.

Usage

```
get_reconc_matrices(agg_levels, h)
```

Arguments

agg_levels	user-selected list of aggregation levels.
h	number of steps ahead for the bottom level forecasts.

Value

A list containing the named elements:

- `A` the aggregation matrix;
- `S` the summing matrix.

See Also

`temporal_aggregation()`

Examples

```
library(bayesRecon)

#Create monthly hierarchy
agg_levels <- c(1,2,3,4,6,12)
h <- 12
rec_mat <- get_reconc_matrices(agg_levels, h)
S <- rec_mat$S
A <- rec_mat$A
```

`infantMortality` *Infant Mortality grouped time series dataset*

Description

A yearly grouped time series dataset, from 1901 to 2003, of infant mortality counts (deaths) in Australia; disaggregated by state (see below), and sex (male and female).

Usage

```
infantMortality
```

Format

List of time series of class ts.

Details

States: New South Wales (NSW), Victoria (VIC), Queensland (QLD), South Australia (SA), Western Australia (WA), Northern Territory (NT), Australian Capital Territory (ACT), and Tasmania (TAS).

Source

hts package CRAN

References

R. J. Hyndman, R. A. Ahmed, G. Athanasopoulos and H.L. Shang (2011) Optimal combination forecasts for hierarchical time series. Computational Statistics and Data Analysis, 55(9), 2579-2589.

`M3_example` *Example of a time series from the M3 forecasting competition*

Description

A monthly time series, from the M3 forecasting competition ("N1485").

Usage

```
M3_example
```

Format

List of time series of class ts.

Source

https://forecasters.org/resources/time-series-data/m3-competition/

`reconc_BUIS` *BUIS for Probabilistic Reconciliation of forecasts via conditioning*

Description

Uses the Bottom-Up Importance Sampling algorithm to draw samples from the reconciled forecast distribution, which is obtained via conditioning.

Usage

```
reconc_BUIS(
  S,
  base_forecasts,
  in_type,
  distr,
  num_samples = 20000,
  suppress_warnings = FALSE,
  seed = NULL
)
```

Arguments

`S`	Summing matrix (n x n_bottom).
`base_forecasts`	A list containing the base_forecasts, see details.
`in_type`	A string or a list of length n. If it is a list the i-th element is a string with two possible values: • 'samples' if the i-th base forecasts are in the form of samples; • 'params' if the i-th base forecasts are in the form of estimated parameters. If it `in_type` is a string it is assumed that all base forecasts are of the same type.
`distr`	A string or a list of length n describing the type of base forecasts. If it is a list the i-th element is a string with two possible values: • 'continuous' or 'discrete' if `in_type[[i]]`='samples'; • 'gaussian', 'poisson' or 'nbinom' if `in_type[[i]]`='params'. If `distr` is a string it is assumed that all distributions are of the same type.
`num_samples`	Number of samples drawn from the reconciled distribution.
`suppress_warnings`	Logical. If `TRUE`, no warnings about effective sample size are triggered. If `FALSE`, warnings are generated. Default is `FALSE`. See Details.
`seed`	Seed for reproducibility.

Details

The parameter base_forecast is a list containing n elements where the i-th element depends on the values of in_type[[i]] and distr[[i]].

If in_type[[i]]='samples', then base_forecast[[i]] is a vector containing samples from the base forecast distribution.

If in_type[[i]]='params', then base_forecast[[i]] is a vector containing the estimated:

- mean and sd for the Gaussian base forecast if distr[[i]]='gaussian', see Normal;
- lambda for the Poisson base forecast if distr[[i]]='poisson', see Poisson;
- mu and size for the negative binomial base forecast if distr[[i]]='nbinom', see NegBinomial.

See the description of the parameters in_type and distr for more details.

The order of the base_forecast list is given by the order of the time series in the summing matrix.

Warnings are triggered from the Importance Sampling step if:

- weights are all zeros, then the upper is ignored during reconciliation;
- the effective sample size is < 200;
- the effective sample size is < 1% of the sample size (num_samples if in_type is 'params' or the size of the base forecast if if in_type is 'samples').

Note that warnings are an indication that the base forecasts might have issues. Please check the base forecasts in case of warnings.

Value

A list containing the reconciled forecasts. The list has the following named elements:

- bottom_reconciled_samples: a matrix (n_bottom x num_samples) containing the reconciled samples for the bottom time series;
- upper_reconciled_samples: a matrix (n_upper x num_samples) containing the reconciled samples for the upper time series;
- reconciled_samples: a matrix (n x num_samples) containing the reconciled samples for all time series.

References

Zambon, L., Azzimonti, D. & Corani, G. (2024). *Efficient probabilistic reconciliation of forecasts for real-valued and count time series*. doi:10.1007/s1122202310343y.

See Also

reconc_gaussian()

Examples

```
library(bayesRecon)

# Create a minimal hierarchy with 2 bottom and 1 upper variable
rec_mat <- get_reconc_matrices(agg_levels=c(1,2), h=2)
S <- rec_mat$S


#1) Gaussian base forecasts

#Set the parameters of the Gaussian base forecast distributions
mu1 <- 2
mu2 <- 4
muY <- 9
mus <- c(muY,mu1,mu2)

sigma1 <- 2
sigma2 <- 2
sigmaY <- 3
sigmas <- c(sigmaY,sigma1,sigma2)

base_forecasts = list()
for (i in 1:nrow(S)) {
base_forecasts[[i]] = c(mus[[i]], sigmas[[i]])
}


#Sample from the reconciled forecast distribution using the BUIS algorithm
buis <- reconc_BUIS(S, base_forecasts, in_type="params",
                 distr="gaussian", num_samples=100000, seed=42)

samples_buis <- buis$reconciled_samples

#In the Gaussian case, the reconciled distribution is still Gaussian and can be
#computed in closed form
Sigma <- diag(sigmas^2)  #transform into covariance matrix
analytic_rec <- reconc_gaussian(S, base_forecasts.mu = mus,
                                base_forecasts.Sigma = Sigma)

#Compare the reconciled means obtained analytically and via BUIS
print(c(S %*% analytic_rec$bottom_reconciled_mean))
print(rowMeans(samples_buis))


#2) Poisson base forecasts

#Set the parameters of the Poisson base forecast distributions
lambda1 <- 2
lambda2 <- 4
lambdaY <- 9
lambdas <- c(lambdaY,lambda1,lambda2)
```

```
base_forecasts <- list()
for (i in 1:nrow(S)) {
 base_forecasts[[i]] = lambdas[i]
}

#Sample from the reconciled forecast distribution using the BUIS algorithm
buis <- reconc_BUIS(S, base_forecasts, in_type="params",
                          distr="poisson", num_samples=100000, seed=42)
samples_buis <- buis$reconciled_samples

#Print the reconciled means
print(rowMeans(samples_buis))
```

reconc_gaussian — *Analytical reconciliation of Gaussian base forecasts*

Description

Closed form computation of the reconciled forecasts in case of Gaussian base forecasts.

Usage

```
reconc_gaussian(S, base_forecasts.mu, base_forecasts.Sigma)
```

Arguments

`S`	summing matrix (n x n_bottom).
`base_forecasts.mu`	a vector containing the means of the base forecasts.
`base_forecasts.Sigma`	a matrix containing the covariance matrix of the base forecasts.

Details

The order of the base forecast means and covariance is given by the order of the time series in the summing matrix.

The function returns only the reconciled parameters of the bottom variables. The reconciled upper parameters and the reconciled samples for the entire hierarchy can be obtained from the reconciled bottom parameters. See the example section.

Value

A list containing the bottom reconciled forecasts. The list has the following named elements:

- `bottom_reconciled_mean`: reconciled mean for the bottom forecasts;
- `bottom_reconciled_covariance`: reconciled covariance for the bottom forecasts.

References

Corani, G., Azzimonti, D., Augusto, J.P.S.C., Zaffalon, M. (2021). *Probabilistic Reconciliation of Hierarchical Forecast via Bayes' Rule*. In: Hutter, F., Kersting, K., Lijffijt, J., Valera, I. (eds) Machine Learning and Knowledge Discovery in Databases. ECML PKDD 2020. Lecture Notes in Computer Science(), vol 12459. Springer, Cham. doi:10.1007/9783030676643_13.

Zambon, L., Agosto, A., Giudici, P., Corani, G. (2023). *Properties of the reconciled distributions for Gaussian and count forecasts*. doi:10.48550/arXiv.2303.15135.

See Also

reconc_BUIS()

Examples

```
library(bayesRecon)

# Create a minimal hierarchy with 2 bottom and 1 upper variable
rec_mat <- get_reconc_matrices(agg_levels=c(1,2), h=2)
S <- rec_mat$S
A <- rec_mat$A

#Set the parameters of the Gaussian base forecast distributions
mu1 <- 2
mu2 <- 4
muY <- 9
mus <- c(muY,mu1,mu2)

sigma1 <- 2
sigma2 <- 2
sigmaY <- 3
sigmas <- c(sigmaY,sigma1,sigma2)

Sigma <- diag(sigmas^2)  #need to transform into covariance matrix
analytic_rec <- reconc_gaussian(S, base_forecasts.mu = mus,
                               base_forecasts.Sigma = Sigma)

bottom_mu_reconc <- analytic_rec$bottom_reconciled_mean
bottom_Sigma_reconc <- analytic_rec$bottom_reconciled_covariance

# Obtain reconciled mu and Sigma for the upper variable
upper_mu_reconc <- A %*% bottom_mu_reconc
upper_Sigma_reconc <- A %*% bottom_Sigma_reconc %*% t(A)

# Obtain reconciled mu and Sigma for the entire hierarchy
Y_mu_reconc <- S %*% bottom_mu_reconc
Y_Sigma_reconc <- S %*% bottom_Sigma_reconc %*% t(S)  # note: singular matrix

# Obtain reconciled samples for the entire hierarchy:
# i.e., sample from the reconciled bottoms and multiply by S
chol_decomp = chol(bottom_Sigma_reconc) # Compute the Cholesky Decomposition
```

```
Z = matrix(rnorm(n = 2000), nrow = 2) # Sample from standard normal
B = chol_decomp %*% Z + matrix(rep(bottom_mu_reconc, 1000), nrow=2) # Apply the transformation

U = S %*% B
Y_reconc = rbind(U, B)
```

---

reconc_MCMC *MCMC for Probabilistic Reconciliation of forecasts via conditioning*

Description

Uses Markov Chain Monte Carlo algorithm to draw samples from the reconciled forecast distribution, which is obtained via conditioning.

This is a bare-bones implementation of the Metropolis-Hastings algorithm, we suggest the usage of tools to check the convergence. The function only works with Poisson or Negative Binomial base forecasts.

The function `reconc_BUIS()` is generally faster on most hierarchies.

Usage

```
reconc_MCMC(
  S,
  base_forecasts,
  distr,
  num_samples = 10000,
  tuning_int = 100,
  init_scale = 1,
  burn_in = 1000,
  seed = NULL
)
```

Arguments

`S`	summing matrix (n x n_bottom).
`base_forecasts`	list of the parameters of the base forecast distributions, see details.
`distr`	a string describing the type of predictive distribution.
`num_samples`	number of samples to draw using MCMC.
`tuning_int`	number of iterations between scale updates of the proposal.
`init_scale`	initial scale of the proposal.
`burn_in`	number of initial samples to be discarded.
`seed`	seed for reproducibility.

Details

The parameter base_forecast is a list containing n elements. Each element is a vector containing the estimated:

- mean and sd for the Gaussian base forecast, see Normal, if distr='gaussian';
- lambda for the Poisson base forecast, see Poisson, if distr='poisson';
- mu and size for the negative binomial base forecast, see NegBinomial, if distr='nbinom'.

The order of the base_forecast list is given by the order of the time series in the summing matrix.

Value

A list containing the reconciled forecasts. The list has the following named elements:

- bottom_reconciled_samples: a matrix (n_bottom x num_samples) containing reconciled samples for the bottom time series;
- upper_reconciled_samples: a matrix (n_upper x num_samples) containing reconciled samples for the upper time series;
- reconciled_samples: a matrix (n x num_samples) containing the reconciled samples for all time series.

References

Corani, G., Azzimonti, D., Rubattu, N. (2023). *Probabilistic reconciliation of count time series*. doi:10.1016/j.ijforecast.2023.04.003.

See Also

reconc_BUIS()

Examples

```
library(bayesRecon)

# Create a minimal hierarchy with 2 bottom and 1 upper variable
rec_mat <- get_reconc_matrices(agg_levels=c(1,2), h=2)
S <- rec_mat$S

#Set the parameters of the Poisson base forecast distributions
lambda1 <- 2
lambda2 <- 4
lambdaY <- 9
lambdas <- c(lambdaY,lambda1,lambda2)

base_forecasts = list()
for (i in 1:nrow(S)) {
 base_forecasts[[i]] = lambdas[i]
}
```

```
#Sample from the reconciled forecast distribution using MCMC
mcmc = reconc_MCMC(S,base_forecasts=lambdas,distr="poisson",
                  num_samples=30000, seed=42)
samples_mcmc <- mcmc$reconciled_samples

#Compare the reconciled means with those obtained via BUIS
buis = reconc_BUIS(S, base_forecasts, in_type="params",
                   distr="poisson", num_samples=100000, seed=42)
samples_buis <- buis$reconciled_samples

print(rowMeans(samples_mcmc))
print(rowMeans(samples_buis))
```

schaferStrimmer_cov *Schäfer Strimmer covariance shrinkage*

Description

Computes the Schäfer Strimmer shrinkage estimator for a covariance matrix from a matrix of samples.

Usage

```
schaferStrimmer_cov(x)
```

Arguments

x	matrix of samples with dimensions nxp (n samples, p dimensions).

Details

This function computes the shrinkage to a diagonal covariance with unequal variances. Note that here we use the estimators $S = XX^T/n$ and $T = diag(S)$ and we internally use the correlation matrix in place of the covariance to compute the optimal shrinkage factor.

Value

A list containing the shrinkage estimator and the optimal lambda. The list has the following named elements:

- `shrink_cov`: the shrinked covariance matrix (p x p);
- `lambda_star`: the optimal lambda for the shrinkage;

References

Schäfer, Juliane, and Korbinian Strimmer. (2005). *A Shrinkage Approach to Large-Scale Covariance Matrix Estimation and Implications for Functional Genomics.* Statistical Applications in Genetics and Molecular Biology 4: Article32. doi:10.2202/15446115.1175.

Examples

```
# Generate some multivariate normal samples
# Parameters
nSamples <- 200
pTrue <- 2

# True moments
trueSigma <- matrix(c(3,2,2,2), nrow=2)
chol_trueSigma <- chol(trueSigma)
trueMean <- c(0,0)

# Generate samples
set.seed(42)
x <- replicate(nSamples, trueMean) +  t(chol_trueSigma)%*%matrix(rnorm(pTrue*nSamples),
                                                             nrow=pTrue,ncol=nSamples)
x <- t(x)
res_shrinkage <- schaferStrimmer_cov(x)
res_shrinkage$lambda_star # should be 0.01287923
```

`temporal_aggregation` *Temporal aggregation of a time series*

Description

Creates a list of aggregated time series from a time series of class ts.

Usage

```
temporal_aggregation(y, agg_levels = NULL)
```

Arguments

`y`	univariate time series of class ts.
`agg_levels`	user-selected list of aggregation levels.

Details

If `agg_levels=NULL` then `agg_levels` is automatically generated by taking all the factors of the time series frequency.

Value

A list of ts objects each containing the aggregates time series in the order defined by `agg_levels`.

See Also

`get_reconc_matrices()`

Examples

```
# Create a monthly count time series with 100 observations
y <- ts(data=stats::rpois(100,lambda = 2),frequency = 12)

# Create the aggregate time series according to agg_levels
y_agg <- temporal_aggregation(y,agg_levels = c(2,3,4,6,12))

# Show annual aggregate time series
print(y_agg$`f=1`)
```

Index