Package 'hts'

May 30, 2021

Title Hierarchical and Grouped Time Series

Version 6.0.2

Description Provides methods for analysing and forecasting hierarchical and grouped time series. The available forecast methods include bottom-up, top-down, optimal combination reconciliation (Hyndman et al. 2011) <doi:10.1016/j.csda.2011.03.006>, and trace minimization reconciliation (Wickramasuriya et al. 2018) <doi:10.1080/01621459.2018.1448825>.

Depends R (>= 3.2.0), forecast (>= 8.12)

Imports SparseM, Matrix, parallel, utils, methods, graphics, grDevices, stats

Suggests testthat, knitr, rmarkdown, covr

LinkingTo Rcpp (>= 0.11.0), RcppEigen

LazyLoad yes

LazyData yes

ByteCompile TRUE

URL https://pkg.earo.me/hts/

BugReports https://github.com/earowang/hts/issues

License GPL (>= 2)

VignetteBuilder knitr

RoxygenNote 7.1.1

Encoding UTF-8

NeedsCompilation yes

Author Rob Hyndman [aut] (Package creator),
Alan Lee [aut] (Fast computation using recursive methods),
Earo Wang [aut, cre],
Shanika Wickramasuriya [aut] (Reconciliation via trace minimization)

Maintainer Earo Wang <earo.wang@gmail.com>

Repository CRAN

Date/Publication 2021-05-30 04:30:13 UTC

R topics documented:

`hts-package` *Hierarchical and grouped time series*

Description

This package presents functions to create, plot and forecast hierarchical and grouped time series. In forecasting hierarchical and grouped time series, the base methods implemented include ETS, ARIMA and the naive (random walk) models. Forecasts for grouped time series are calibrated using bottom-up and optimal combination methods. Forecasts for hierarchical time series are distributed in the hierarchy using bottom-up, top-down, middle-out and optimal combination methods. Three top-down methods are available: the two Gross-Sohl methods and the forecast-proportion approach of Hyndman, Ahmed, and Athanasopoulos (2011).

Author(s)

Rob J Hyndman, Alan Lee, Earo Wang and Shanika L Wickramasuriya with contributions from Roman A Ahmed and Han Lin Shang to earlier versions of the package

References

G. Athanasopoulos, R. A. Ahmed and R. J. Hyndman (2009) Hierarchical forecasts for Australian domestic tourism, *International Journal of Forecasting*, **25**, 146-166.

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. https://robjhyndman.com/publications/hierarchical/

Hyndman, R. J., Lee, A., & Wang, E. (2016). Fast computation of reconciled forecasts for hierarchical and grouped time series. *Computational Statistics and Data Analysis*, **97**, 16-23. https://robjhyndman.com/papers/hgts7.pdf

Wickramasuriya, S. L., Athanasopoulos, G., & Hyndman, R. J. (2018). Forecasting hierarchical and grouped time series through trace minimization. *Journal of the American Statistical Association*, to appear https://robjhyndman.com/papers/mint.pdf

---

`accuracy.gts` *In-sample or out-of-sample accuracy measures for forecast grouped and hierarchical model*

Description

Returns a range of summary measures of the forecast accuracy. The function measures out-of-sample forecast accuracy based on (holdout data - forecasts) and in-sample accuracy at the bottom level when setting `keep.fitted = TRUE` in the `forecast.gts`. All measures are defined and discussed in Hyndman and Koehler (2006).

Usage

```
## S3 method for class 'gts'
accuracy(object, test, levels, ..., f = NULL)
```

Arguments

`object`	An object of class `gts`, containing the forecasted hierarchical or grouped time series. In-sample accuracy at the bottom level returns when `test` is missing.
`test`	An object of class `gts`, containing the holdout hierarchical time series
`levels`	Return the specified level(s), when carrying out out-of-sample
`...`	Extra arguments to be ignored
`f`	Deprecated. Please use `object` instead.

Details

MASE calculation is scaled using MAE of in-sample naive forecasts for non-seasonal time series, and in-sample seasonal naive forecasts for seasonal time series.

Value

Matrix giving forecast accuracy measures.

`ME`	Mean Error
`RMSE`	Root Mean Square Error
`MAE`	Mean Absolute Error
`MAPE`	Mean Absolute Percentage Error
`MPE`	Mean Percentage Error
`MASE`	Mean Absolute Scaled Error

Author(s)

Rob J Hyndman and Earo Wang

References

R. J. Hyndman and A. Koehler (2006), Another look at measures of forecast accuracy, *International Journal of Forecasting*, **22**, 679-688.

See Also

hts, plot.gts, forecast.gts, accuracy

Examples

```
data <- window(htseg2, start = 1992, end = 2002)
test <- window(htseg2, start = 2003)
fcasts <- forecast(data, h = 5, method = "bu")
accuracy(fcasts, test)
accuracy(fcasts, test, levels = 1)
```

---

aggts *Extract selected time series from a gts object*

Description

The time series from selected levels of a hierarchical/grouped time series or a forecasted hierarchical/grouped time series are returned as a multivariate time series.

Usage

```
aggts(y, levels, forecasts = TRUE)
```

Arguments

y	An object of class {gts}.
levels	Integer(s) or string(s) giving the specified level(s).
forecasts	If y contains forecasts and historical data, then forecasts indicates whether to return the forecasts or the historical data. Otherwise it is ignored.

Author(s)

Earo Wang

See Also

allts

Examples

```
aggts(htseg1, levels = c(0, 2))
aggts(infantgts, levels = "State")
```

`allts` *Extract all time series from a gts object*

Description

The time series from all levels of a hierarchical/grouped time series or a forecasted hierarchical/grouped time series are returned as a multivariate time series.

Usage

```
allts(y, forecasts = TRUE)
```

Arguments

`y`	An object of class `gts`.
`forecasts`	If `y` contains forecasts and historical data, then `forecasts` indicates whether to return the forecasts or the historical data. Otherwise it is ignored.

Author(s)

Rob J Hyndman

See Also

`aggts`

Examples

```
allts(htseg1)
```

combinef *Optimally combine forecasts from a hierarchical or grouped time series*

Description

Using the methods of Hyndman et al. (2016) and Hyndman et al. (2011), this function optimally combines the forecasts at all levels of a hierarchical time series. The `forecast.gts` calls this function when the `comb` method is selected.

Usage

```
combinef(
  fcasts,
  nodes = NULL,
  groups = NULL,
  weights = NULL,
  nonnegative = FALSE,
  algorithms = c("lu", "cg", "chol", "recursive", "slm"),
  keep = c("gts", "all", "bottom"),
  parallel = FALSE,
  num.cores = 2,
  control.nn = list()
)
```

Arguments

`fcasts`	Matrix of forecasts for all levels of the hierarchical time series. Each row represents one forecast horizon and each column represents one time series from the hierarchy.
`nodes`	If the object class is `hts`, a list contains the number of child nodes referring to `hts`.
`groups`	If the object class is `gts`, a gmatrix is required, which is the same as `groups` in the function `gts`.
`weights`	A numeric vector. The default is `NULL` which means that ordinary least squares is implemented.
`nonnegative`	Logical. Should the reconciled forecasts be non-negative?
`algorithms`	An algorithm to be used for computing reconciled forecasts. See `forecast.gts` for details.
`keep`	Return a `gts` object or the the reconciled forecasts at the bottom level.
`parallel`	Logical. Import parallel package to allow parallel processing.
`num.cores`	Numeric. Specify how many cores are going to be used.
`control.nn`	A list of control parameters to be passed on to the block principal pivoting algorithm. See 'Details'.

Details

The `control.nn` argument is a list that can supply any of the following components:

`ptype` Permutation method to be used: `"fixed"` or `"random"`. Defaults to `"fixed"`.

`par` The number of full exchange rules that may be tried. Defaults to 10.

`gtol` The tolerance of the convergence criteria. Defaults to `sqrt(.Machine$double.eps)`.

Value

Return the (non-negative) reconciled `gts` object or forecasts at the bottom level.

Author(s)

Alan Lee, Rob J Hyndman, Earo Wang and Shanika L Wickramasuriya

References

Hyndman, R. J., Ahmed, R. A., Athanasopoulos, G., & Shang, H. L. (2011). Optimal combination forecasts for hierarchical time series. *Computational Statistics and Data Analysis*, **55**(9), 2579–2589. https://robjhyndman.com/publications/hierarchical/

Hyndman, R. J., Lee, A., & Wang, E. (2016). Fast computation of reconciled forecasts for hierarchical and grouped time series. *Computational Statistics and Data Analysis*, **97**, 16–32. https://robjhyndman.com/publications/hgts/

Wickramasuriya, S. L., Turlach, B. A., & Hyndman, R. J. (to appear). Optimal non-negative forecast reconciliation. *Statistics and Computing*. https://robjhyndman.com/publications/nnmint/

See Also

`hts`, `forecast.gts`

Examples

```
# hts example
## Not run:
h <- 12
ally <- aggts(htseg1)
allf <- matrix(NA, nrow = h, ncol = ncol(ally))
for(i in 1:ncol(ally))
allf[,i] <- forecast(auto.arima(ally[,i]), h = h)$mean
allf <- ts(allf, start = 51)
y.f <- combinef(allf, get_nodes(htseg1), weights = NULL, keep = "gts", algorithms = "lu")
plot(y.f)

## End(Not run)

## Not run:
h <- 12
ally <- abs(aggts(htseg2))
allf <- matrix(NA, nrow = h, ncol = ncol(ally))
```

```
for(i in 1:ncol(ally))
  allf[,i] <- forecast(auto.arima(ally[,i], lambda = 0, biasadj = TRUE), h = h)$mean
b.f <- combinef(allf, get_nodes(htseg2), weights = NULL, keep = "bottom",
algorithms = "lu")
b.nnf <- combinef(allf, get_nodes(htseg2), weights = NULL, keep = "bottom",
algorithms = "lu", nonnegative = TRUE)

## End(Not run)

# gts example
## Not run:
abc <- ts(5 + matrix(sort(rnorm(200)), ncol = 4, nrow = 50))
g <- rbind(c(1,1,2,2), c(1,2,1,2))
y <- gts(abc, groups = g)
h <- 12
ally <- aggts(y)
allf <- matrix(NA,nrow = h,ncol = ncol(ally))
for(i in 1:ncol(ally))
  allf[,i] <- forecast(auto.arima(ally[,i]),h = h)$mean
allf <- ts(allf, start = 51)
y.f <- combinef(allf, groups = get_groups(y), keep ="gts", algorithms = "lu")
plot(y.f)

## End(Not run)
```

`forecast.gts` *Forecast a hierarchical or grouped time series*

Description

Methods for forecasting hierarchical or grouped time series.

Usage

```
## S3 method for class 'gts'
forecast(
  object,
  h = ifelse(frequency(object$bts) > 1L, 2L * frequency(object$bts), 10L),
  method = c("comb", "bu", "mo", "tdgsa", "tdgsf", "tdfp"),
  weights = c("wls", "ols", "mint", "nseries"),
  fmethod = c("ets", "arima", "rw"),
  algorithms = c("lu", "cg", "chol", "recursive", "slm"),
  covariance = c("shr", "sam"),
  nonnegative = FALSE,
  control.nn = list(),
  keep.fitted = FALSE,
  keep.resid = FALSE,
  positive = FALSE,
  lambda = NULL,
```

```
  level,
  FUN = NULL,
  xreg = NULL,
  newxreg = NULL,
  parallel = FALSE,
  num.cores = 2,
  ...
)
```

Arguments

object	Hierarchical or grouped time series object of class {gts}
h	Forecast horizon
method	Method for distributing forecasts within the hierarchy. See details
weights	Weights used for "optimal combination" method: weights="ols" uses an unweighted combination (as described in Hyndman et al 2011); weights="wls" uses weights based on forecast variances (as described in Hyndman et al 2016); weights="mint" uses a full covariance estimate to determine the weights (as described in Wickramasuriya et al 2019); weights="nseries" uses weights based on the number of series aggregated at each node.
fmethod	Forecasting method to use for each series.
algorithms	An algorithm to be used for computing the combination forecasts (when method=="comb"). The combination forecasts are based on an ill-conditioned regression model. "lu" indicates LU decomposition is used; "cg" indicates a conjugate gradient method; "chol" corresponds to a Cholesky decomposition; "recursive" indicates the recursive hierarchical algorithm of Hyndman et al (2016); "slm" uses sparse linear regression. Note that algorithms = "recursive" and algorithms = "slm" cannot be used if weights="mint".
covariance	Type of the covariance matrix to be used with weights="mint": either a shrinkage estimator ("shr") with shrinkage towards the diagonal; or a sample covariance matrix ("sam").
nonnegative	Logical. Should the reconciled forecasts be non-negative?
control.nn	A list of control parameters to be passed on to the block principal pivoting algorithm. See 'Details'.
keep.fitted	If TRUE, keep fitted values at the bottom level.
keep.resid	If TRUE, keep residuals at the bottom level.
positive	If TRUE, forecasts are forced to be strictly positive (by setting lambda=0).
lambda	Box-Cox transformation parameter.
level	Level used for "middle-out" method (only used when method = "mo").
FUN	A user-defined function that returns an object which can be passed to the forecast function. It is applied to all series in order to generate base forecasts. When FUN is not NULL, fmethod, positive and lambda are all ignored. Suitable values for FUN are tbats and stlf for example.

xreg	When fmethod = "arima", a vector or matrix of external regressors used for modelling, which must have the same number of rows as the original univariate time series
newxreg	When fmethod = "arima", a vector or matrix of external regressors used for forecasting, which must have the same number of rows as the h forecast horizon
parallel	If TRUE, import parallel package to allow parallel processing.
num.cores	If parallel = TRUE, specify how many cores are going to be used.
...	Other arguments passed to ets, auto.arima or FUN.

Details

Base methods implemented include ETS, ARIMA and the naive (random walk) models. Forecasts are distributed in the hierarchy using bottom-up, top-down, middle-out and optimal combination methods.

Three top-down methods are available: the two Gross-Sohl methods and the forecast-proportion approach of Hyndman, Ahmed, and Athanasopoulos (2011). The "middle-out" method "mo" uses bottom-up ("bu") for levels higher than level and top-down forecast proportions ("tdfp") for levels lower than level.

For non-hierarchical grouped data, only bottom-up and combination methods are possible, as any method involving top-down disaggregation requires a hierarchical ordering of groups.

When xreg and newxreg are passed, the same covariates are applied to every series in the hierarchy.

The control.nn argument is a list that can supply any of the following components:

ptype Permutation method to be used: "fixed" or "random". Defaults to "fixed".

par The number of full exchange rules that may be tried. Defaults to 10.

gtol The tolerance of the convergence criteria. Defaults to sqrt(.Machine$double.eps).

Value

A forecasted hierarchical/grouped time series of class gts.

Note

In-sample fitted values and resiuals are not returned if method = "comb" and nonnegative = TRUE.

Author(s)

Earo Wang, Rob J Hyndman and Shanika L Wickramasuriya

References

Athanasopoulos, G., Ahmed, R. A., & Hyndman, R. J. (2009). Hierarchical forecasts for Australian domestic tourism, *International Journal of Forecasting*, **25**, 146-166.

Hyndman, R. J., Ahmed, R. A., Athanasopoulos, G., & Shang, H. L. (2011). Optimal combination forecasts for hierarchical time series. *Computational Statistics and Data Analysis*, **55**(9), 2579–2589. https://robjhyndman.com/publications/hierarchical/

Hyndman, R. J., Lee, A., & Wang, E. (2016). Fast computation of reconciled forecasts for hierarchical and grouped time series. *Computational Statistics and Data Analysis*, **97**, 16–32. https://robjhyndman.com/publications/hgts/

Wickramasuriya, S. L., Athanasopoulos, G., & Hyndman, R. J. (2019). Optimal forecast reconciliation for hierarchical and grouped time series through trace minimization. *Journal of the American Statistical Association*, **114**(526), 804–819. https://robjhyndman.com/publications/mint/

Wickramasuriya, S. L., Turlach, B. A., & Hyndman, R. J. (to appear). Optimal non-negative forecast reconciliation. *Statistics and Computing*. https://robjhyndman.com/publications/nnmint/

Gross, C., & Sohl, J. (1990). Dissagregation methods to expedite product line forecasting, *Journal of Forecasting*, **9**, 233–254.

See Also

hts, gts, plot.gts, accuracy.gts

Examples

```
forecast(htseg1, h = 10, method = "bu", fmethod = "arima")

## Not run:
  forecast(
    htseg2, h = 10, method = "comb", algorithms = "lu",
    FUN = function(x) tbats(x, use.parallel = FALSE)
  )

## End(Not run)
```

---

get_groups *Get nodes/groups from an hts/gts object*

Description

Get nodes/groups from an hts/gts object

Usage

```
get_groups(y)

get_nodes(y)
```

Arguments

y	An hts or gts object series.

gts *Create a grouped time series*

Description

Method for creating grouped time series.

Usage

```
gts(y, groups, gnames = rownames(groups), characters)

is.gts(xts)

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

## S3 method for class 'gts'
summary(object, ...)
```

Arguments

y	A matrix or multivariate time series contains the bottom level series.
groups	Group matrix indicates the group structure, with one column for each series when completely disaggregated, and one row for each grouping of the time series. It allows either a numerical matrix or a matrix consisting of strings that can be used for labelling. If the argument `characters` is used, then `groups` will be automatically generated within the function.
gnames	Specify the group names.
characters	A vector of integers, or a list containing vectors of integers, indicating the segments in which bottom level names can be read in order to construct the corresponding grouping matrix and its labels. A `list` class is used when a grouped time series includes one or more hierarchies. For example, a grouped time series may involve a geographical grouping and a product grouping, with each of them associated with a 2-level hierarchy. In this situation, a bottom level name such as "VICMelbAB" would indicate the state "VIC" (3 characters) followed by the city "Melb" (4 characters), then the product category "A" (1 character) followed by the sub-product category "B" (1 character). In this example, the specification of `characters` is `list(c(3,4),c(1,1))`, where the first element `c(3,4)` corresponds to the geographical hierarchy and the second element corresponds to the product hierarchy. In the special case where there is a non-hierarchical grouped time series, a vector of integers is also possible. For example, a grouped time series may involve state, age and sex grouping variables. In this situation, a bottom level name such as "VIC1F" would indicate the state "VIC", age group "1" and sex "F". Because none of these is hierarchical, we could specify `characters = list(3,1,1)`, or as a simple numeric vector: `characters = c(3,1,1)`. This

implies its non-hierarchical structure and its characters segments. Again, all bottom level names must be of the same length. Currently, the use of `characters` only supports 2-way cross-products for grouping variables. Specifying `groups` is more general (but more complicated), as any combination of grouping variables can be used.

`xts`	`gts` object.
`x`	`gts` object.
`...`	Extra arguments passed to `print` and `summary`.
`object`	`gts` object.

Value

`bts`	Multivariate time series contains the bottom level series
`groups`	Information about the groups of a grouped time series
`labels`	Information about the labels that are used for plotting.

Author(s)

Earo Wang and Rob J Hyndman

References

Hyndman, R. J., Ahmed, R. A., Athanasopoulos, G., & Shang, H. L. (2011). Optimal combination forecasts for hierarchical time series. *Computational Statistics and Data Analysis*, **55**(9), 2579–2589. https://robjhyndman.com/publications/hierarchical/

See Also

`hts`, `accuracy.gts`, `forecast.gts`, `plot.gts`

Examples

```
# Example 1 illustrating the usage of the "groups" argument
abc <- ts(5 + matrix(sort(rnorm(1600)), ncol = 16, nrow = 100))
sex <- rep(c("female", "male"), each = 8)
state <- rep(c("NSW", "VIC", "QLD", "SA", "WA", "NT", "ACT", "TAS"), 2)
gc <- rbind(sex, state)  # a matrix consists of strings.
gn <- rbind(rep(1:2, each = 8), rep(1:8, 2))  # a numerical matrix
rownames(gc) <- rownames(gn) <- c("Sex", "State")
x <- gts(abc, groups = gc)
y <- gts(abc, groups = gn)

# Example 2 with two simple hierarchies (geography and product) to show the argument "characters"
bnames1 <- c("VICMelbAA", "VICMelbAB", "VICGeelAA", "VICGeelAB",
             "VICMelbBA", "VICMelbBB", "VICGeelBA", "VICGeelBB",
             "NSWSyndAA", "NSWSyndAB", "NSWWollAA", "NSWWollAB",
             "NSWSyndBA", "NSWSyndBB", "NSWWollBA", "NSWWollBB")
bts1 <- matrix(ts(rnorm(160)), ncol = 16)
```

```
colnames(bts1) <- bnames1
x1 <- gts(bts1, characters = list(c(3, 4), c(1, 1)))

# Example 3 with a non-hierarchical grouped time series of 3 grouping variables (state, age and sex)
bnames2 <- c("VIC1F", "VIC1M", "VIC2F", "VIC2M", "VIC3F", "VIC3M",
             "NSW1F", "NSW1M", "NSW2F", "NSW2M", "NSW3F", "NSW3M")
bts2 <- matrix(ts(rnorm(120)), ncol = 12)
colnames(bts2) <- bnames2
x2 <- gts(bts2, characters = c(3, 1, 1))
```

`hts` *Create a hierarchical time series*

Description

Method for creating hierarchical time series.

Usage

```
hts(y, nodes, bnames = colnames(y), characters)

is.hts(xts)

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

## S3 method for class 'hts'
summary(object, ...)
```

Arguments

`y`	A matrix or multivariate time series contain the bottom level series.
`nodes`	A list contains the number of child nodes associated with each level, which indicates the hierarchical structure. The default is a simple hierarchy with only 2 levels (i.e. total and bottom). If the argument `characters` is used, `nodes` will be automatically generated within the function.
`bnames`	The names of the bottom time series.
`characters`	Integers indicate the segments in which the bottom level names can be read in order to construct the corresponding node structure and its labels. For instance, suppose one of the bottom series is named "VICMelb" referring to the city of Melbourne within the state of Victoria. Then `characters` would be specified as `c(3,4)` referring to states of 3 characters (e.g., "VIC") and cities of 4 characters (e.g., "Melb") All the bottom names must be of the same length, with number of characters for each segment the same for all series.
`xts`	`hts` object.
`x`	`hts` object.

...	Extra arguments passed to print and summary.
object	hts object.

Value

bts	Multivariate time series containing the bottom level series
nodes	Information about the nodes of a hierarchical time series
labels	Information about the labels that are used for plotting.

Author(s)

Earo Wang and Rob J Hyndman

References

Hyndman, R. J., Ahmed, R. A., Athanasopoulos, G., & Shang, H. L. (2011). Optimal combination forecasts for hierarchical time series. *Computational Statistics and Data Analysis*, **55**(9), 2579–2589. https://robjhyndman.com/publications/hierarchical/

See Also

gts, accuracy.gts, forecast.gts, plot.gts

Examples

```
# Example 1
# The hierarchical structure looks like 2 child nodes associated with level 1,
# which are followed by 3 and 2 sub-child nodes respectively at level 2.
nodes <- list(2, c(3, 2))
abc <- ts(5 + matrix(sort(rnorm(500)), ncol = 5, nrow = 100))
x <- hts(abc, nodes)

# Example 2
# Suppose we've got the bottom names that can be useful for constructing the node
# structure and the labels at higher levels. We need to specify how to split them
# in the argument "characters".
library(hts)
abc <- ts(5 + matrix(sort(rnorm(1000)), ncol = 10, nrow = 100))
colnames(abc) <- c("A10A", "A10B", "A10C", "A20A", "A20B",
                   "B30A", "B30B", "B30C", "B40A", "B40B")
y <- hts(abc, characters = c(1, 2, 1))
```

htseg1 *Simple examples of hierarchical time series.*

Description

These are simulated data. htseg1 has three levels with a total of 8 series each of length 10. htseg2 has four levels with a total of 17 series each of length 16.

Format

Objects of class hts.

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. https://robjhyndman.com/publications/hierarchical/

Examples

```
plot(htseg1)
```

infantgts *Regional infant mortality counts across Australia from 1933 to 2003.*

Description

These are infant mortality counts. This data set is an example of gts, where the total infant mortality count in Australia can be first disaggregated by sex then by state, or vice versa.

Format

Objects of class gts.

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.

Examples

```
plot(infantgts)
```

MinT *Trace minimization for hierarchical or grouped time series*

Description

Using the method of Wickramasuriya et al. (2019), this function combines the forecasts at all levels of a hierarchical or grouped time series. The forecast.gts calls this function when the MinT method is selected.

Usage

```
MinT(
  fcasts,
  nodes = NULL,
  groups = NULL,
  residual,
  covariance = c("shr", "sam"),
  nonnegative = FALSE,
  algorithms = c("lu", "cg", "chol"),
  keep = c("gts", "all", "bottom"),
  parallel = FALSE,
  num.cores = 2,
  control.nn = list()
)
```

Arguments

fcasts	Matrix of forecasts for all levels of a hierarchical or grouped time series. Each row represents one forecast horizon and each column represents one time series of aggregated or disaggregated forecasts.
nodes	If the object class is hts, a list contains the number of child nodes referring to hts.
groups	If the object is gts, a gmatrix is required, which is the same as groups in the function gts.
residual	Matrix of insample residuals for all the aggregated and disaggregated time series. The columns must be in the same order as fcasts.
covariance	Type of the covariance matrix to be used. Shrinking towards a diagonal unequal variances ("shr") or sample covariance matrix ("sam").
nonnegative	Logical. Should the reconciled forecasts be non-negative?
algorithms	Algorithm used to compute inverse of the matrices.
keep	Return a gts object or the reconciled forecasts at the bottom level.
parallel	Logical. Import parallel package to allow parallel processing.
num.cores	Numeric. Specify how many cores are going to be used.
control.nn	A list of control parameters to be passed on to the block principal pivoting algorithm. See 'Details'.

Details

The `control.nn` argument is a list that can supply any of the following components:

`ptype` Permutation method to be used: `"fixed"` or `"random"`. Defaults to `"fixed"`.

`par` The number of full exchange rules that may be tried. Defaults to 10.

`gtol` The tolerance of the convergence criteria. Defaults to `sqrt(.Machine$double.eps)`.

Value

Return the reconciled `gts` object or forecasts at the bottom level.

Author(s)

Shanika L Wickramasuriya

References

Wickramasuriya, S. L., Athanasopoulos, G., & Hyndman, R. J. (2019). Optimal forecast reconciliation for hierarchical and grouped time series through trace minimization. *Journal of the American Statistical Association*, **114**(526), 804–819. https://robjhyndman.com/working-papers/mint/

Wickramasuriya, S. L., Turlach, B. A., & Hyndman, R. J. (to appear). Optimal non-negative forecast reconciliation. *Statistics and Computing*. https://robjhyndman.com/publications/nnmint/

Hyndman, R. J., Lee, A., & Wang, E. (2016). Fast computation of reconciled forecasts for hierarchical and grouped time series. *Computational Statistics and Data Analysis*, **97**, 16–32. https://robjhyndman.com/publications/hgts/

See Also

`hts`, `gts`, `forecast.gts`, `combinef`

Examples

```
# hts example
## Not run:
h <- 12
ally <- aggts(htseg1)
n <- nrow(ally)
p <- ncol(ally)
allf <- matrix(NA, nrow = h, ncol = p)
res <- matrix(NA, nrow = n, ncol = p)
for(i in 1:p)
{
  fit <- auto.arima(ally[, i])
  allf[, i] <- forecast(fit, h = h)$mean
  res[, i] <- na.omit(ally[, i] - fitted(fit))
}
allf <- ts(allf, start = 51)
y.f <- MinT(allf, get_nodes(htseg1), residual = res, covariance = "shr",
  keep = "gts", algorithms = "lu")
```

```
plot(y.f)
y.f_cg <- MinT(allf, get_nodes(htseg1), residual = res, covariance = "shr",
  keep = "all", algorithms = "cg")

## End(Not run)

## Not run:
h <- 12
ally <- abs(aggts(htseg2))
allf <- matrix(NA, nrow = h, ncol = ncol(ally))
res <- matrix(NA, nrow = nrow(ally), ncol = ncol(ally))
for(i in 1:ncol(ally)) {
  fit <- auto.arima(ally[, i], lambda = 0, biasadj = TRUE)
  allf[,i] <- forecast(fit, h = h)$mean
  res[,i] <- na.omit(ally[, i] - fitted(fit))
}
b.f <- MinT(allf, get_nodes(htseg2), residual = res, covariance = "shr",
  keep = "bottom", algorithms = "lu")
b.nnf <-  MinT(allf, get_nodes(htseg2), residual = res, covariance = "shr",
  keep = "bottom", algorithms = "lu", nonnegative = TRUE, parallel = TRUE)

## End(Not run)

# gts example
## Not run:
abc <- ts(5 + matrix(sort(rnorm(200)), ncol = 4, nrow = 50))
g <- rbind(c(1,1,2,2), c(1,2,1,2))
y <- gts(abc, groups = g)
h <- 12
ally <- aggts(y)
n <- nrow(ally)
p <- ncol(ally)
allf <- matrix(NA,nrow = h,ncol = ncol(ally))
res <- matrix(NA, nrow = n, ncol = p)
for(i in 1:p)
{
  fit <- auto.arima(ally[, i])
  allf[, i] <- forecast(fit, h = h)$mean
  res[, i] <- na.omit(ally[, i] - fitted(fit))
}
allf <- ts(allf, start = 51)
y.f <- MinT(allf, groups = get_groups(y), residual = res, covariance = "shr",
  keep = "gts", algorithms = "lu")
plot(y.f)

## End(Not run)
```

---

plot.gts *Plot grouped or hierarchical time series*

Description

Method for plotting grouped or hierarchical time series and their forecasts.

Usage

```
## S3 method for class 'gts'
plot(x, include, levels, labels = TRUE, col = NULL, color_lab = FALSE, ...)
```

Arguments

x	An object of class gts.
include	Number of values from historical time series to include in the plot of forecasted group/hierarchical time series.
levels	Integer(s) or string(s) giving the specified levels(s) to be plotted
labels	If TRUE, plot the labels next to each series
col	Vector of colours, passed to plot.ts and to lines
color_lab	If TRUE, colour the direct labels to match line colours. If FALSE will be as per par()$fg.
...	Other arguments passing to plot.default

Author(s)

Rob J Hyndman and Earo Wang

References

Hyndman, R. J., Ahmed, R. A., Athanasopoulos, G., & Shang, H. L. (2011). Optimal combination forecasts for hierarchical time series. *Computational Statistics and Data Analysis*, **55**(9), 2579–2589. https://robjhyndman.com/publications/hierarchical/

See Also

aggts

Examples

```
plot(htseg1, levels = c(0, 2))
plot(infantgts, include = 10, levels = "State")
plot(infantgts, include = 10, levels = "State",
    col = colours()[100:107], lty = 1:8, color_lab = TRUE)
```

`smatrix` *Summing matrix for hierarchical or grouped time series*

Description

This function returns the summing matrix for a hierarchical or grouped time series, as defined in Hyndman et al. (2011).

Usage

```
smatrix(xts)
```

Arguments

`xts`	Hierarchical or grouped time series of class `gts`.

Value

A numerical matrix.

Author(s)

Earo Wang

References

Hyndman, R. J., Ahmed, R. A., Athanasopoulos, G., & Shang, H. L. (2011). Optimal combination forecasts for hierarchical time series. *Computational Statistics and Data Analysis*, **55**(9), 2579–2589. https://robjhyndman.com/publications/hierarchical/

See Also

`hts`, `gts`, `combinef`

Examples

```
smatrix(htseg1)
```

`window.gts` *Time window of a gts object*

Description

Extracts a subset of the time series from a grouped time series object.

Usage

```
## S3 method for class 'gts'
window(x, ...)
```

Arguments

`x`	An object of class `gts`.
`...`	All other arguments are passed to `window.ts`.

Author(s)

Rob J Hyndman

Examples

```
window(htseg2, start = 2000, end = 2001)
```

Index