Package ‘FoReco’

July 4, 2022

Type Package

Title Point Forecast Reconciliation

Version 0.2.5

Description Classical (bottom-up and top-down), optimal and heuristic combination forecast reconciliation procedures for cross-sectional, temporal, and cross-temporal linearly constrained time series (Di Fonzo and Girolimetto, 2021) <doi:10.1016/j.ijforecast.2021.08.004>.

License GPL-3

URL https://github.com/daniGiro/FoReco, https://danigiro.github.io/FoReco/

BugReports https://github.com/daniGiro/FoReco/issues

Depends R (>= 2.10), Matrix, osqp, stats

Imports cli, corpcor, methods, mathjaxr

Suggests knitr, rmarkdown

RdMacros mathjaxr

VignetteBuilder knitr

Encoding UTF-8

LazyData true

NeedsCompilation no

RoxygenNote 7.2.0

R topics documented:

`FoReco-package` *FoReco: point forecast reconciliation*

Description

An R package offering classical (bottom-up and top-down), and modern (optimal and heuristic combination) forecast reconciliation procedures for cross-sectional, temporal, and cross-temporal linearly constrained time series.

Details

The `FoReco` package is designed for point forecast reconciliation, a post-forecasting process aimed to improve the accuracy of the base forecasts for a system of linearly constrained (e.g. hierarchical/grouped) time series. The main functions are:

`htsrec()`: cross-sectional (contemporaneous) forecast reconciliation.

`thfrec()`: forecast reconciliation for a single time series through temporal hierarchies.

`lccrec()`: level conditional forecast reconciliation for genuine hierarchical/grouped time series.

`tdrec()`: top-down (cross-sectional, temporal, cross-temporal) forecast reconciliation for genuine hierarchical/grouped time series.

`ctbu()`: bottom-up cross-temporal forecast reconciliation.

`tcsrec()`: heuristic first-temporal-then-cross-sectional cross-temporal forecast reconciliation.

`cstrec()`: heuristic first-cross-sectional-then-temporal cross-temporal forecast reconciliation.

`iterec()`: heuristic iterative cross-temporal forecast reconciliation.

`octrec()`: optimal combination cross-temporal forecast reconciliation.

Author(s)

Tommaso Di Fonzo and Daniele Girolimetto, Department of Statistical Sciences, University of Padua (Italy).

References

Di Fonzo, T., and Girolimetto, D. (2021), Cross-temporal forecast reconciliation: Optimal combination method and heuristic alternatives, *International Journal of Forecasting*, in press doi:10.1016/j.ijforecast.2021.08.004.

Di Fonzo, T., Girolimetto, D. (2022), Forecast combination based forecast reconciliation: insights and extensions, *International Journal of Forecasting*, in press.

Cmatrix *Cross-sectional (contemporaneous) aggregation matrix*

Description

This function allows the user to easily build the $(n_a \times n_b)$ cross-sectional (contemporaneous) matrix mapping the n_b bottom level series into the n_a higher level ones. (*Experimental version*)

Usage

```
Cmatrix(formula, data, sep = "_", sparse = TRUE, top_label = "Total")
```

Arguments

formula	Specification of the hierarchical structure: grouped hierarchies are specified using ~ g1 * g2 and nested hierarchies are specified using ~ parent / child. Mixtures of the two formulations are also possible, like ~ g1 * (grandparent / parent / child).
data	A dataset in which each column contains the values of the variables in the formula and each row identifies a bottom level time series.
sep	Character to separate the names of the aggregated series (*default* is "_").
sparse	Option to return sparse matrix (*default* is TRUE).
top_label	Label of the top level variable (*default* is "Total").

Value

A (na x nb) matrix.

See Also

Other utilities: FoReco2ts(), commat(), ctf_tools(), hts_tools(), lcmat(), oct_bounds(), score_index(), shrink_estim(), thf_tools()

Examples

```
## Balanced hierarchy
#         T
#    |--------|
#    A        B
#  |---|   |--|--|
# AA   AB  BA BB BC
# Names of the bottom level variables
data_bts <- data.frame(X1 = c("A", "A", "B", "B", "B"),
                       X2 = c("A", "B", "A", "B", "C"),
                       stringsAsFactors = FALSE)
# Cross-sectional aggregation matrix
C <- Cmatrix(~ X1 / X2, data_bts, sep = "")

## Unbalanced hierarchy (1)
#             T
#    |--------|------|
#    A        B      C
```

```
#  |---|   |--|--|
# AA   AB  BA BB BC
# Names of the bottom level variables
data_bts <- data.frame(X1 = c("A", "A", "B", "B", "B", "C"),
                       X2 = c("A", "B", "A", "B", "C", NA),
                       stringsAsFactors = FALSE)
# Cross-sectional aggregation matrix
C <- Cmatrix(~ X1 / X2, data_bts, sep = "")

## Unbalanced hierarchy (2)
#                 T
#       |---------|---------|
#       A         B         C
#     |---|     |---|     |---|
#    AA   AB   BA   BB   CA   CB
#  |----|         |----|
# AAA  AAB       BBA  BBB
# Names of the bottom level variables
data_bts <- data.frame(X1 = c("A", "A", "A", "B", "B", "B", "C", "C"),
                       X2 = c("A", "A", "B", "A", "B", "B", "A", "B"),
                       X3 = c("A", "B", NA, NA, "A", "B", NA, NA),
                       stringsAsFactors = FALSE)
# Cross-sectional aggregation matrix
C <- Cmatrix(~ X1 / X2 / X3, data_bts, sep = "")

## Grouped hierarchy
#         C               S
#    |--------|      |--------|
#    A        B      M        F
#  |---|    |---|
# AA   AB  BA   BB
# Names of the bottom level variables
data_bts <- data.frame(X1 = c("A", "A", "B", "B", "A", "A", "B", "B"),
                       X2 = c("A", "B", "A", "B", "A", "B", "A", "B"),
                       Y1 = c("M", "M", "M", "M", "F", "F", "F", "F"),
                       stringsAsFactors = FALSE)
# Cross-sectional aggregation matrix
C <- Cmatrix(~ Y1 * (X1 / X2), data_bts, sep = "")
```

`commat` *Commutation matrix*

Description

This function returns the $(rc \times rc)$ commutation matrix $\mathbf{P}$ such that

$$\mathbf{P}\text{vec}(\mathbf{Y}) = \text{vec}(\mathbf{Y}'),$$

where $\mathbf{Y}$ is a $(r \times c)$ matrix.

Usage

```
commat(r, c)
```

Arguments

`r`	Number of rows of $\mathbf{Y}$.
`c`	Number of columns of $\mathbf{Y}$.

Value

A sparse $(rc \times rc)$ matrix, $\mathbf{P}$.

References

Magnus, J.R., Neudecker, H. (2019), Matrix Differential Calculus with Applications in Statistics and Econometrics, third edition, New York, Wiley, pp. 54-55.

See Also

Other utilities: `Cmatrix()`, `FoReco2ts()`, `ctf_tools()`, `hts_tools()`, `lcmat()`, `oct_bounds()`, `score_index()`, `shrink_estim()`, `thf_tools()`

Examples

```
Y <- matrix(rnorm(30), 5, 6)
P <- commat(5, 6)
P %*% as.vector(Y) == as.vector(t(Y)) # check
```

`cstrec` *Heuristic first-cross-sectional-then-temporal cross-temporal forecast reconciliation*

Description

Cross-temporal forecast reconciliation according to the heuristic procedure by Kourentzes and Athanasopoulos (2019), where the order of application of the two reconciliation steps (temporal-first-then-cross-sectional, as in the function `tcsrec()`), is inverted. The function `cstrec()` performs cross-sectional reconciliation (`htsrec()`) first, then temporal reconciliation (`thfrec()`), and finally applies the average of the projection matrices obtained in the second step to the one dimensional reconciled values obtained in the first step.

Usage

```
cstrec(basef, hts_comb, thf_comb, res, ...)
```

Arguments

`basef`	$(n \times h(k^* + m))$ matrix of base forecasts to be reconciled, $\widehat{\mathbf{Y}}$; n is the total number of variables, m is the highest time frequency, k^* is the sum of (a subset of) $(p-1)$ factors of m, excluding m, and h is the forecast horizon for the lowest frequency time series. Each row identifies a time series, and the forecasts are ordered as [lowest_freq' ... highest_freq']'.
`hts_comb, thf_comb`	Type of covariance matrix (respectively $(n \times n)$ and $((k^* + m) \times (k^* + m))$) to be used in the cross-sectional and temporal reconciliation, see more in `comb` param of `htsrec()` and `thfrec()`.

`res`	$(n \times N(k^* + m))$ matrix containing the residuals at all the temporal frequencies ordered as [lowest_freq' ... highest_freq']' (columns) for each variable (row), needed to estimate the covariance matrix when `hts_comb` = {`"wls"`, `"shr"`, `"sam"`} and/or `hts_comb` = {`"wlsv"`, `"wlsh"`, `"acov"`, `"strar1"`, `"sar1"`, `"har1"`, `"shr"`, `"sam"`}. The rows must be in the same order as `basef`.
`...`	any other options useful for `htsrec()` and `thfrec()`, e.g. `m`, `C` (or `Ut` and `nb`), `nn` (for non-negative reconciliation only at the first step), `mse`, `corpcor`, `type`, `sol`, `settings`, `W`, `Omega`,...

Details

Warning, the two-step heuristic reconciliation allows non negativity constraints only in the first step. This means that it is not guaranteed the non-negativity of the final reconciled values.

Value

The function returns a list with two elements:

`recf`	$(n \times h(k^* + m))$ reconciled forecasts matrix, $\widetilde{\mathbf{Y}}$.
`M`	Matrix which transforms the uni-dimensional reconciled forecasts of step 1 (projection approach) .

References

Di Fonzo, T., and Girolimetto, D. (2021), Cross-temporal forecast reconciliation: Optimal combination method and heuristic alternatives, *International Journal of Forecasting*, in press.

Kourentzes, N., Athanasopoulos, G. (2019), Cross-temporal coherent forecasts for Australian tourism, *Annals of Tourism Research*, 75, 393-409.

Schäfer, J.L., Opgen-Rhein, R., Zuber, V., Ahdesmaki, M., Duarte Silva, A.P., Strimmer, K. (2017), *Package 'corpcor'*, R package version 1.6.9 (April 1, 2017), https://CRAN.R-project.org/package=corpcor.

Schäfer, J.L., Strimmer, K. (2005), A Shrinkage Approach to Large-Scale Covariance Matrix Estimation and Implications for Functional Genomics, *Statistical Applications in Genetics and Molecular Biology*, 4, 1.

Stellato, B., Banjac, G., Goulart, P., Bemporad, A., Boyd, S. (2020). OSQP: An Operator Splitting Solver for Quadratic Programs, *Mathematical Programming Computation*, 12, 4, 637-672.

Stellato, B., Banjac, G., Goulart, P., Boyd, S., Anderson, E. (2019), OSQP: Quadratic Programming Solver using the 'OSQP' Library, R package version 0.6.0.3 (October 10, 2019), https://CRAN.R-project.org/package=osqp.

See Also

Other reconciliation procedures: `ctbu()`, `htsrec()`, `iterec()`, `lccrec()`, `octrec()`, `tcsrec()`, `tdrec()`, `thfrec()`

Examples

```
data(FoReco_data)
obj <- cstrec(FoReco_data$base, m = 12, C = FoReco_data$C,
              hts_comb = "shr", thf_comb = "acov", res = FoReco_data$res)
```

ctbu — *Bottom-up cross-temporal forecast reconciliation*

Description

Cross temporal reconciled forecasts for all series at any temporal aggregation level are computed by appropriate summation of the high-frequency bottom base forecasts $\widehat{\mathbf{b}}_i, i = 1, ..., n_b$, according to a bottom-up procedure like what is currently done in both the cross-sectional and temporal frameworks.

Usage

```
ctbu(Bmat, m, C)
```

Arguments

Bmat	$(n_b \times hm)$ matrix of high-frequency bottom time series base forecasts ($\widehat{\mathbf{B}}^{[1]}$). h is the forecast horizon for the lowest frequency (most temporally aggregated) time series.
m	Highest available sampling frequency per seasonal cycle (max. order of temporal aggregation, m), or a subset of the p factors of m.
C	$(n_a \times n_b)$ cross-sectional (contemporaneous) matrix mapping the bottom level series into the higher level ones.

Details

Denoting by $\ddot{\mathbf{Y}}$ the $(n \times h(k^* + m))$ matrix containing the bottom-up cross temporal reconciled forecasts, it is:

$$\ddot{\mathbf{Y}} = \begin{bmatrix} \mathbf{C}\widehat{\mathbf{B}}^{[1]}\mathbf{K}_1' & \mathbf{C}\widehat{\mathbf{B}}^{[1]} \\ \widehat{\mathbf{B}}^{[1]}\mathbf{K}_1' & \widehat{\mathbf{B}}^{[1]} \end{bmatrix},$$

where $\mathbf{C}$ is the cross-sectional (contemporaneous) aggregation matrix, $\mathbf{K}_1$ is the temporal aggregation matrix with $h = 1$, and $\widehat{\mathbf{B}}^{[1]}$ is the matrix containing the high-frequency bottom time series base forecasts. This expression is equivalent to $\text{vec}(\ddot{\mathbf{Y}}') = \widetilde{\mathbf{S}}\text{vec}(\widehat{\mathbf{Y}}')$ for $h = 1$, where $\widetilde{\mathbf{S}}$ is the cross-temporal summing matrix for $\text{vec}(\widehat{\mathbf{Y}}')$, and $\widehat{\mathbf{Y}}$ is the $(n \times h(k^* + m))$ matrix containing all the base forecasts at any temporal aggregation order.

Value

The function returns a $(n \times h(k^* + m))$ matrix of bottom-up cross-temporally reconciled forecasts, $\ddot{\mathbf{Y}}$.

References

Di Fonzo, T., and Girolimetto, D. (2021), Cross-temporal forecast reconciliation: Optimal combination method and heuristic alternatives, *International Journal of Forecasting*, in press.

See Also

Other reconciliation procedures: cstrec(), htsrec(), iterec(), lccrec(), octrec(), tcsrec(), tdrec(), thfrec()

Examples

```
data(FoReco_data)
# monthly base forecasts
id <- which(simplify2array(strsplit(colnames(FoReco_data$base),
                                    split = "_"))[1, ] == "k1")
hfbts <- FoReco_data$base[-c(1:3), id]
obj <- ctbu(Bmat = hfbts, m = 12, C = FoReco_data$C)
rownames(obj) <- rownames(FoReco_data$base)
```

---

ctf_tools   *Cross-temporal reconciliation tools*

Description

Some useful tools for the cross-temporal forecast reconciliation of a linearly constrained (hierarchical/grouped) multiple time series.

Usage

```
ctf_tools(C, m, h = 1, Ut, nb, sparse = TRUE)
```

Arguments

C	$(n_a \times n_b)$ cross-sectional (contemporaneous) matrix mapping the bottom level series into the higher level ones.
m	Highest available sampling frequency per seasonal cycle (max. order of temporal aggregation, m), or a subset of the p factors of m.
h	Forecast horizon for the lowest frequency (most temporally aggregated) time series (*default* is 1).
Ut	Zero constraints cross-sectional (contemporaneous) kernel matrix $(\mathbf{U}'\mathbf{y} = \mathbf{0})$ spanning the null space valid for the reconciled forecasts. It can be used instead of parameter C, but nb $(n = n_a + n_b)$ is needed if $\mathbf{U}' \neq [\mathbf{I} - \mathbf{C}]$. If the hierarchy admits a structural representation, $\mathbf{U}'$ has dimension $(n_a \times n)$.
nb	Number of bottom time series; if C is present, nb and Ut are not used.
sparse	Option to return sparse object (*default* is TRUE).

Value

ctf list with:

Ht	Full row-rank cross-temporal zero constraints (kernel) matrix coherent with $\mathbf{y} = \text{vec}(\mathbf{Y}')$: $\mathbf{H}'\mathbf{y} = \mathbf{0}$.
Hbrevet	Complete, not full row-rank cross-temporal zero constraints (kernel) matrix coherent with $\mathbf{y} = \text{vec}(\mathbf{Y}')$: $\breve{\mathbf{H}}'\mathbf{y} = \mathbf{0}$.
Hcheckt	Full row-rank cross-temporal zero constraints (kernel) matrix coherent with $\check{\mathbf{y}}$ (structural representation): $\check{\mathbf{H}}'\check{\mathbf{y}} = \mathbf{0}$.
Ccheck	Cross-temporal aggregation matrix $\check{\mathbf{C}}$ coherent with $\check{\mathbf{y}}$ (structural representation).

Scheck	Cross-temporal summing matrix $\check{\mathbf{S}}$ coherent with $\check{\mathbf{y}}$ (structural representation).
Fmat	Cross-temporal summing matrix $\widetilde{\mathbf{F}}$ coherent with $\mathbf{y} = \text{vec}(\mathbf{Y}')$.

hts list from hts_tools .

thf list from thf_tools .

See Also

Other utilities: Cmatrix(), FoReco2ts(), commat(), hts_tools(), lcmat(), oct_bounds(), score_index(), shrink_estim(), thf_tools()

Examples

```
# One level hierarchy (na = 1, nb = 2) with quarterly data
obj <- ctf_tools(C = matrix(c(1, 1), 1), m = 4)
```

FoReco-hts *Simple examples to compare* FoReco *and* hts *packages*

Description

Two datasets of the **hts** package are used to show how to get the same results using **FoReco**. First, we consider the htseg1 dataset (a simulated three level hierarchy, with a total of 8 series, each of length 10). Then, we take the htseg2 dataset (a simulated four level hierarchy with a total of 17 series, each of length 16). htseg1 and htseg2 are objects of class hts in **hts**.

References

Hyndman, R. J., Lee, A., Wang, E., and Wickramasuriya, S. (2020). hts: Hierarchical and Grouped Time Series, *R package version 6.0.1*, https://CRAN.R-project.org/package=hts.

Examples

```
## Not run:
library(hts)
require(FoReco)

####### htseg1 #######
data <- allts(htseg1)
n <- NCOL(data)
nb <- NCOL(htseg1$bts)
na <- n-nb
C <- smatrix(htseg1)[1:na, ]

# List containing the base forecasts
# Forecast horizon: 10
base <- list()
for (i in 1:n) {
  base[[i]] <- forecast(auto.arima(data[, i]))
}

# Create the matrix of base forecasts
```

```
BASE <- NULL
for (i in 1:n) {
  BASE <- cbind(BASE, base[[i]]$mean)
}
colnames(BASE) <- colnames(data)

# Create the matrix of residuals
res <- NULL
for (i in 1:n) {
  res <- cbind(res, base[[i]]$residuals)
}
colnames(res) <- colnames(data)

## Comparisons
# ols
# two commands in hts...
Y_hts_forecast <- forecast(htseg1, method = "comb", fmethod = "arima", weights = "ols")
Y_hts_ols <- combinef(BASE, nodes = get_nodes(htseg1), keep = "all")
# ...with the same results:
sum(abs(allts(Y_hts_forecast) - Y_hts_ols) > 1e-10)

Y_FoReco_ols <- htsrec(BASE, C = C, comb = "ols")$recf
sum(abs(Y_hts_ols - Y_FoReco_ols) > 1e-10)

# struc
w <- 1 / apply(smatrix(htseg1), 1, sum)
Y_hts_struc <- combinef(BASE, nodes = get_nodes(htseg1), weights = w, keep = "all")
Y_FoReco_struc <- htsrec(BASE, C = C, comb = "struc")$recf
sum(abs(Y_hts_struc - Y_FoReco_struc) > 1e-10)

# shr
Y_hts_shr <- MinT(BASE, nodes = get_nodes(htseg1), keep = "all",
                  covariance = "shr", residual = res)
Y_FoReco_shr <- htsrec(BASE, C = C, comb = "shr", res = res)$recf
sum(abs(Y_hts_shr - Y_FoReco_shr) > 1e-10)

# sam - hts error "MinT needs covariance matrix to be positive definite."
#       The covariance matrix is ill-conditioned, hts considers it as non-invertible
Y_hts_sam <- MinT(BASE, nodes = get_nodes(htseg1), keep = "all",
                  covariance = "sam", residual = res)
Y_FoReco_sam <- htsrec(BASE, C = C, comb = "sam", res = res)$recf
# sum((Y_hts_sam-Y_FoReco_sam)>1e-10)

####### htseg2 #######
data <- allts(htseg2)
n <- NCOL(data)
nb <- NCOL(htseg2$bts)
na <- n-nb
C <- smatrix(htseg2)[1:na, ]

## Computation of the base forecasts
# using the auto.arima() function of the package forecast (loaded by hts)
# List containing the base forecasts
# Forecast horizon: 10
base <- list()
for (i in 1:n) {
  base[[i]] <- forecast(auto.arima(data[, i]))
```

```
}

# Create the matrix of base forecasts
BASE <- NULL
for (i in 1:n) {
  BASE <- cbind(BASE, base[[i]]$mean)
}
colnames(BASE) <- colnames(data)

# Create the matrix of residuals
res <- NULL
for (i in 1:n) {
  res <- cbind(res, base[[i]]$residuals)
}
colnames(res) <- colnames(data)

## Comparisons
# ols
Y_hts_ols <- combinef(BASE, nodes = get_nodes(htseg2), keep = "all")
Y_FoReco_ols <- htsrec(BASE, C = C, comb = "ols")$recf
sum(abs(Y_hts_ols - Y_FoReco_ols) > 1e-10)

# struc
w <- 1 / apply(smatrix(htseg2), 1, sum)
Y_hts_struc <- combinef(BASE, nodes = get_nodes(htseg2), weights = w, keep = "all")
Y_FoReco_struc <- htsrec(BASE, C = C, comb = "struc")$recf
sum(abs(Y_hts_struc - Y_FoReco_struc) > 1e-10)

# shr
Y_hts_shr <- MinT(BASE, nodes = get_nodes(htseg2), keep = "all", covariance = "shr", residual = res)
Y_FoReco_shr <- htsrec(BASE, C = C, comb = "shr", res = res)$recf
sum(abs(Y_hts_shr- Y_FoReco_shr) > 1e-10)

## End(Not run)
```

FoReco-thief — *Simple examples to compare* FoReco *and* thief *packages*

Description

The dataset in the thief package is used to show how to get the same results with the FoReco package. In particular, we take the weekly data of Accident and Emergency demand in the UK, AEdemand, from 1 January 2011 to 31 December 2014.

References

Hyndman, R. J., Kourentzes, N. (2018), thief: Temporal HIErarchical Forecasting, *R package version 0.3*, https://cran.r-project.org/package=thief.

Examples

```
## Not run:
library(thief)
require(FoReco)
```

```
dataset <- window(AEdemand[, 12], start = c(2011, 1), end = c(2014, 52))
data <- tsaggregates(dataset)
# Base forecasts
base <- list()
for (i in 1:5) {
  base[[i]] <- forecast(auto.arima(data[[i]]))
}
base[[6]] <- forecast(auto.arima(data[[6]]), h = 2)
# Base forecasts vector
base_vec <- NULL
for (i in 6:1) {
  base_vec <- c(base_vec, base[[i]]$mean)
}
# Residual vector
res <- NULL
for (i in 6:1) {
  res <- c(res, base[[i]]$residuals)
}

# OLS
# two commands in thief...
obj_thief <- thief(dataset, m = 52, h = 2 * 52, comb = "ols", usemodel = "arima")
obj_thief <- tsaggregates(obj_thief$mean)
y_thief <- NULL
for (i in 6:1) {
  y_thief <- c(y_thief, obj_thief[[i]])
}
obj_thief_ols <- reconcilethief(base, comb="ols")
y_thief_ols <- NULL
for (i in 6:1) {
  y_thief_ols <- c(y_thief_ols, obj_thief_ols[[i]]$mean)
}
# ...with the same results:
sum(abs(y_thief_ols - y_thief) > 1e-10)

y_FoReco_ols <- thfrec(base_vec, 52, comb = "ols")$recf
sum(abs(y_FoReco_ols - y_thief_ols) > 1e-10)

# STRUC
obj_thief_struc <- reconcilethief(base, comb="struc")
y_thief_struc <- NULL
for (i in 6:1) {
  y_thief_struc <- c(y_thief_struc, obj_thief_struc[[i]]$mean)
}
y_FoReco_struc <- thfrec(base_vec, 52, comb = "struc")$recf
sum(abs(y_FoReco_struc - y_thief_struc) > 1e-10)

# BU
obj_thief_bu <- reconcilethief(base, comb="bu")
y_thief_bu <- NULL
for (i in 6:1) {
  y_thief_bu <- c(y_thief_bu, obj_thief_bu[[i]]$mean)
}
y_FoReco_bu <- thfrec(base_vec, 52, comb = "bu")$recf
sum(abs(y_FoReco_bu - y_thief_bu) > 1e-10)

# SHR
```

```
obj_thief_shr <- reconcilethief(base, comb="shr")
y_thief_shr <- NULL
for (i in 6:1) {
  y_thief_shr <- c(y_thief_shr, obj_thief_shr[[i]]$mean)
}
y_FoReco_shr <- thfrec(base_vec, 52, comb = "shr", res = res)$recf
sum(abs(y_FoReco_shr - y_thief_shr) > 1e-10)

## End(Not run)
```

FoReco2ts *Reconciled forecasts matrix/vector to time-series class*

Description

Function to transform the matrix/vector of reconciled forecasts (`recf` from htsrec, thfrec, tdrec, octrec, lccrec, tcsrec, cstrec, iterec, ctbu) into a list of time series objects.

Usage

```
FoReco2ts(recf, ...)
```

Arguments

`recf`	$(h(k^* + m) \times 1)$ reconciled forecasts vector from thfrec, $(h \times n)$ reconciled forecasts matrix from htsrec or $(n \times h(k^* + m))$ reconciled forecasts matrix from octrec, tcsrec, cstrec, iterec, ctbu.
`...`	optional arguments to ts (i.e. starting date); frequency is required only for the cross-sectional case.

Value

A list of class `"ts"` objects

See Also

Other utilities: `Cmatrix()`, `commat()`, `ctf_tools()`, `hts_tools()`, `lcmat()`, `oct_bounds()`, `score_index()`, `shrink_estim()`, `thf_tools()`

Examples

```
data(FoReco_data)
# Cross-temporal framework
oct_recf <- octrec(FoReco_data$base, m = 12, C = FoReco_data$C,
                   comb = "bdshr", res = FoReco_data$res)$recf
obj_oct <- FoReco2ts(recf = oct_recf, start = c(15, 1))

# Cross-sectional framework
# monthly base forecasts
id <- which(simplify2array(strsplit(colnames(FoReco_data$base),
                                    split = "_"))[1, ] == "k1")
mbase <- t(FoReco_data$base[, id])
# monthly residuals
```

```
id <- which(simplify2array(strsplit(colnames(FoReco_data$res),
                                    split = "_"))[1, ] == "k1")
mres <- t(FoReco_data$res[, id])
hts_recf <- htsrec(mbase, C = FoReco_data$C, comb = "shr", res = mres)$recf
obj_hts <- FoReco2ts(recf = hts_recf, start = c(15, 1), frequency = 12)

# Temporal framework
# top ts base forecasts ([lowest_freq' ...  highest_freq']')
topbase <- FoReco_data$base[1, ]
# top ts residuals ([lowest_freq' ...  highest_freq']')
topres <- FoReco_data$res[1, ]
thf_recf <- thfrec(topbase, m = 12, comb = "acov", res = topres)$recf
obj_thf <- FoReco2ts(recf = thf_recf, start = c(15, 1))
```

FoReco_data — *Forecast reconciliation for a simulated linearly constrained, genuine hierarchical multiple time series*

Description

A two-level hierarchy with $n = 8$ monthly time series. In the cross-sectional framework, at any time it is $Tot = A + B + C$, $A = AA + AB$ and $B = BA + BB$ (the bottom time series being AA, AB, BA, BB, and C, it is $n_b = 5$). The monthly observations are aggregated to their annual ($k = 12$), semi-annual ($k = 6$), four-monthly ($k = 4$), quarterly ($k = 3$), and bi-monthly ($k = 2$) counterparts. The monthly bottom time series are simulated from five different SARIMA models (see Using the 'FoReco' package). There are 180 (15 years) monthly observations: the first 168 values (14 years) are used as training set, and the last 12 form the test set.

Usage

```
data(FoReco_data)
```

Format

An object of class "list":

base (8×28) matrix of base forecasts. Each row identifies a time series and the forecasts are ordered as [lowest_freq' ... highest_freq']'.

test (8×28) matrix of test set. Each row identifies a time series and the observed values are ordered as [lowest_freq' ... highest_freq']'.

res (8×392) matrix of in-sample residuals. Each row identifies a time series and the in-sample residuals are ordered as [lowest_freq' ... highest_freq']'.

C (3×5) cross-sectional (contemporaneous) aggregation matrix.

obs List of the observations at any levels and temporal frequencies.

Examples

```
data(FoReco_data)
# Cross-sectional reconciliation for all temporal aggregation levels
# (monthly, bi-monthly, ..., annual)
K <- c(1,2,3,4,6,12)
hts_recf <- NULL
for(i in 1:length(K)){
  # base forecasts
  id <- which(simplify2array(strsplit(colnames(FoReco_data$base),
                                      split = "_"))[1, ] == paste("k", K[i], sep=""))
  mbase <- t(FoReco_data$base[, id])
  # residuals
  id <- which(simplify2array(strsplit(colnames(FoReco_data$res),
                                      split = "_"))[1, ] == paste("k", K[i], sep=""))
  mres <- t(FoReco_data$res[, id])
  hts_recf[[i]] <- htsrec(mbase, C = FoReco_data$C, comb = "shr",
                          res = mres, keep = "recf")
}
names(hts_recf) <- paste("k", K, sep="")

# Forecast reconciliation through temporal hierarchies for all time series
# comb = "acov"
n <- NROW(FoReco_data$base)
thf_recf <- matrix(NA, n, NCOL(FoReco_data$base))
dimnames(thf_recf) <- dimnames(FoReco_data$base)
for(i in 1:n){
  # ts base forecasts ([lowest_freq' ...  highest_freq']')
  tsbase <- FoReco_data$base[i, ]
  # ts residuals ([lowest_freq' ...  highest_freq']')
  tsres <- FoReco_data$res[i, ]
  thf_recf[i,] <- thfrec(tsbase, m = 12, comb = "acov",
                         res = tsres, keep = "recf")
}

# Iterative cross-temporal reconciliation
# Each iteration: t-acov + cs-shr
ite_recf <- iterec(FoReco_data$base, note=FALSE,
                   m = 12, C = FoReco_data$C,
                   thf_comb = "acov", hts_comb = "shr",
                   res = FoReco_data$res, start_rec = "thf")$recf

# Heuristic first-cross-sectional-then-temporal cross-temporal reconciliation
# cs-shr + t-acov
cst_recf <- cstrec(FoReco_data$base, m = 12, C = FoReco_data$C,
                   thf_comb = "acov", hts_comb = "shr",
                   res = FoReco_data$res)$recf

# Heuristic first-temporal-then-cross-sectional cross-temporal reconciliation
# t-acov + cs-shr
tcs_recf <- tcsrec(FoReco_data$base, m = 12, C = FoReco_data$C,
                   thf_comb = "acov", hts_comb = "shr",
                   res = FoReco_data$res)$recf

# Optimal cross-temporal reconciliation
# comb = "bdshr"
oct_recf <- octrec(FoReco_data$base, m = 12, C = FoReco_data$C,
```

```
comb = "bdshr", res = FoReco_data$res, keep = "recf")
```

htsrec *Cross-sectional (contemporaneous) forecast reconciliation*

Description

Cross-sectional (contemporaneous) forecast reconciliation of a linearly constrained (e.g., hierarchical/grouped) multiple time series. The reconciled forecasts are calculated either through a projection approach (Byron, 1978, see also van Erven and Cugliari, 2015, and Wickramasuriya et al., 2019), or the equivalent structural approach by Hyndman et al. (2011). Moreover, the classic bottom-up approach is available.

Usage

```
htsrec(basef, comb, C, res, Ut, nb, mse = TRUE, corpcor = FALSE,
       type = "M", sol = "direct", keep = "list",  v = NULL, nn = FALSE,
       nn_type = "osqp", settings = list(), bounds = NULL, W = NULL)
```

Arguments

basef — $(h \times n)$ matrix of base forecasts to be reconciled; h is the forecast horizon and n is the total number of time series.

comb — Type of the reconciliation. Except for Bottom-up, each option corresponds to a specific $(n \times n)$ covariance matrix:
- **bu** (Bottom-up);
- **ols** (Identity);
- **struc** (Structural variances);
- **wls** (Series variances) - uses res;
- **shr** (Shrunk covariance matrix - MinT-shr) - uses res;
- **sam** (Sample covariance matrix - MinT-sam) - uses res;
- **w** use your personal matrix W in param W.

C — $(n_a \times n_b)$ cross-sectional (contemporaneous) matrix mapping the bottom level series into the higher level ones.

res — $(N \times n)$ in-sample residuals matrix needed when comb = {"wls", "shr", "sam"}. The columns must be in the same order as basef.

Ut — Zero constraints cross-sectional (contemporaneous) kernel matrix $(\mathbf{U}'\mathbf{y} = \mathbf{0})$ spanning the null space valid for the reconciled forecasts. It can be used instead of parameter C, but nb $(n = n_a + n_b)$ is needed if $\mathbf{U}' \neq [\mathbf{I} - \mathbf{C}]$. If the hierarchy admits a structural representation, $\mathbf{U}'$ has dimension $(n_a \times n)$.

nb — Number of bottom time series; if C is present, nb and Ut are not used.

mse — Logical value: TRUE (*default*) calculates the covariance matrix of the in-sample residuals (when necessary) according to the original **hts** and **thief** formulation: no mean correction, T as denominator.

corpcor — Logical value: TRUE if **corpcor** (Schäfer et al., 2017) must be used to shrink the sample covariance matrix according to Schäfer and Strimmer (2005), otherwise the function uses the same implementation as package **hts**.

`type`	Approach used to compute the reconciled forecasts: `"M"` for the projection approach with matrix M (*default*), or `"S"` for the structural approach with summing matrix S.
`sol`	Solution technique for the reconciliation problem: either `"direct"` (*default*) for the closed-form matrix solution, or `"osqp"` for the numerical solution (solving a linearly constrained quadratic program using `solve_osqp`).
`keep`	Return a list object of the reconciled forecasts at all levels (if keep = "list") or only the reconciled forecasts matrix (if keep = "recf").
`v`	vector index of the fixed base forecast ($\min(v) > 0$ and $\max(v) < n$).
`nn`	Logical value: `TRUE` if non-negative reconciled forecasts are wished.
`nn_type`	"osqp" (default), "KAnn" (only `type == "M"`) or "sntz".
`settings`	Settings for **osqp** (object `osqpSettings`). The default options are: `verbose = FALSE`, `eps_abs = 1e-5`, `eps_rel = 1e-5`, `polish_refine_iter = 100` and `polish = TRUE`. For details, see the **osqp** documentation (Stellato et al., 2019).
`bounds`	$(n \times 2)$ matrix containing the cross-sectional bounds: the first column is the lower bound, and the second column is the upper bound.
`W`	This option permits to directly enter the covariance matrix: 1. `W` must be a p.d. $(n \times n)$ matrix or a list of h matrix (one for each forecast horizon); 2. if `comb` is different from `"w"`, `W` is not used.

Details

Let $\mathbf{y}$ be a $(n \times 1)$ vector of target point forecasts which are wished to satisfy the system of linearly independent constraints

$$\mathbf{U}'\mathbf{y} = \mathbf{0}_{(r\times 1)},$$

where $\mathbf{U}'$ is a $(r \times n)$ matrix, with $\text{rank}(\mathbf{U}') = r \leq n$, and $\mathbf{0}_{(r\times 1)}$ is a $(r \times 1)$ null vector. Let $\widehat{\mathbf{y}}$ be a $(n \times 1)$ vector of unbiased point forecasts, not fulfilling the linear constraints (i.e., $\mathbf{U}'\widehat{\mathbf{y}} \neq \mathbf{0}$).

We consider a regression-based reconciliation method assuming that $\widehat{\mathbf{y}}$ is related to $\mathbf{y}$ by

$$\widehat{\mathbf{y}} = \mathbf{y} + \varepsilon,$$

where ε is a $(n \times 1)$ vector of zero mean disturbances, with known p.d. covariance matrix $\mathbf{W}$. The reconciled forecasts $\widetilde{\mathbf{y}}$ are found by minimizing the generalized least squares (GLS) objective function $(\widehat{\mathbf{y}} - \mathbf{y})' \mathbf{W}^{-1} (\widehat{\mathbf{y}} - \mathbf{y})$ constrained by $\mathbf{U}'\mathbf{y} = \mathbf{0}_{(r\times 1)}$:

$$\widetilde{\mathbf{y}} = \text{argmin}_{\mathbf{y}} \, (\mathbf{y} - \widehat{\mathbf{y}})' \mathbf{W}^{-1} (\mathbf{y} - \widehat{\mathbf{y}}), \quad \text{s.t. } \mathbf{U}'\mathbf{y} = \mathbf{0}.$$

The solution is given by

$$\widetilde{\mathbf{y}} = \widehat{\mathbf{y}} - \mathbf{W}\mathbf{U}\left(\mathbf{U}'\mathbf{W}\mathbf{U}\right)^{-1}\mathbf{U}'\widehat{\mathbf{y}} = \mathbf{M}\widehat{\mathbf{y}},$$

where $\mathbf{M} = \mathbf{I}_n - \mathbf{W}\mathbf{U}\left(\mathbf{U}'\mathbf{W}\mathbf{U}\right)^{-1}\mathbf{U}'$ is a $(n \times n)$ projection matrix. This solution is used by `htsrec` when `type = "M"`.

Denoting with $\mathbf{d}_{\widehat{\mathbf{y}}} = \mathbf{0} - \mathbf{U}'\widehat{\mathbf{y}}$ the $(r \times 1)$ vector containing the *coherency* errors of the base forecasts, we can re-state the solution as

$$\widetilde{\mathbf{y}} = \widehat{\mathbf{y}} + \mathbf{W}\mathbf{U}\left(\mathbf{U}'\mathbf{W}\mathbf{U}\right)^{-1}\mathbf{d}_{\widehat{y}},$$

which makes it clear that the reconciliation formula simply adjusts the vector $\widehat{\mathbf{y}}$ with a linear combination – according to the smoothing matrix $\mathbf{L} = \mathbf{WU}\left(\mathbf{U}'\mathbf{WU}\right)^{-1}$ – of the coherency errors of the base forecasts.

In addition, if the error term ε is gaussian, the reconciliation error $\widetilde{\varepsilon} = \widetilde{\mathbf{y}} - \mathbf{y}$ is a zero-mean gaussian vector with covariance matrix

$$E\left(\widetilde{\mathbf{y}} - \mathbf{y}\right)\left(\widetilde{\mathbf{y}} - \mathbf{y}\right)' = \mathbf{W} - \mathbf{WU}\left(\mathbf{U}'\mathbf{WU}\right)^{-1}\mathbf{U}' = \mathbf{MW}.$$

Hyndman et al. (2011, see also Wickramasuriya et al., 2019) propose an equivalent, alternative formulation as for the reconciled estimates, obtained by GLS estimation of the model

$$\widehat{\mathbf{y}} = \mathbf{S}\beta + \varepsilon,$$

where $\mathbf{S}$ is the *structural summation matrix* describing the aggregation relationships operating on $\mathbf{y}$, and β is a subset of $\mathbf{y}$ containing the target forecasts of the bottom level series, such that $\mathbf{y} = \mathbf{S}\beta$. Since the hypotheses on ε remain unchanged,

$$\widetilde{\beta} = \left(\mathbf{S}'\mathbf{W}^{-1}\mathbf{S}\right)^{-1}\mathbf{S}'\mathbf{W}^{-1}\widehat{\mathbf{y}}$$

is the best linear unbiased estimate of β, and the whole reconciled forecasts vector is given by

$$\widetilde{\mathbf{y}} = \mathbf{S}\widetilde{\beta} = \mathbf{SG}\widehat{\mathbf{y}},$$

where $\mathbf{G} = \left(\mathbf{S}'\mathbf{W}^{-1}\mathbf{S}\right)^{-1}\mathbf{S}'\mathbf{W}^{-1}$, and $\mathbf{M} = \mathbf{SG}$. This solution is used by `htsrec` when `type = "S"`.

Bounds on the reconciled forecasts

The user may impose bounds on the reconciled forecasts. The parameter `bounds` permits to consider lower ($\mathbf{a}$) and upper ($\mathbf{b}$) bounds like $\mathbf{a} \leq \widetilde{\mathbf{y}} \leq \mathbf{b}$ such that:

$$\begin{array}{c} a_1 \leq \widetilde{y}_1 \leq b_1 \\ \dots \\ a_n \leq \widetilde{y}_n \leq b_n \end{array} \Rightarrow \text{bounds} = [\mathbf{a}\ \mathbf{b}] = \begin{bmatrix} a_1 & b_1 \\ \vdots & \vdots \\ a_n & b_n \end{bmatrix},$$

where $a_i \in [-\infty, +\infty]$ and $b_i \in [-\infty, +\infty]$. If y_i is unbounded, the i-th row of `bounds` would be equal to `c(-Inf, +Inf)`. Notice that if the `bounds` parameter is used, `sol = "osqp"` must be used. This is not true in the case of non-negativity constraints:

- `sol = "direct"`: first the base forecasts are reconciled without non-negativity constraints, then, if negative reconciled values are present, the `"osqp"` solver is used;
- `sol = "osqp"`: the base forecasts are reconciled using the `"osqp"` solver.

In this case it is not necessary to build a matrix containing the bounds, and it is sufficient to set `nn = "TRUE"`.

Non-negative reconciled forecasts may be obtained by setting `nn_type` alternatively as:

- `nn_type = "KAnn"` (Kourentzes and Athanasopoulos, 2021)
- `nn_type = "sntz"` ("set-negative-to-zero")
- `nn_type = "osqp"` (Stellato et al., 2020)

Value

If the parameter keep is equal to "recf", then the function returns only the $(h \times n)$ reconciled forecasts matrix, otherwise (keep="all") it returns a list that mainly depends on what type of representation (type) and solution technique (sol) have been used:

recf
: $(h \times n)$ reconciled forecasts matrix, $\widetilde{\mathbf{Y}}$.

W
: Covariance matrix used for forecast reconciliation, $\mathbf{W}$.

nn_check
: Number of negative values (if zero, there are no values below zero).

rec_check
: Logical value: has the hierarchy been respected?

varf (type="direct")
: $(n \times 1)$ reconciled forecasts variance vector for $h = 1$, diag$(\mathbf{MW})$.

M (type="direct")
: Projection matrix, $\mathbf{M}$ (projection approach).

G (type="S" and type="direct")
: Projection matrix, $\mathbf{G}$ (structural approach, $\mathbf{M} = \mathbf{SG}$).

S (type="S" and type="direct")
: Cross-sectional summing matrix, $\mathbf{S}$.

info (type="osqp")
: matrix with information in columns for each forecast horizon h (rows): run time (run_time), number of iteration (iter), norm of primal residual (pri_res), status of osqp's solution (status) and polish's status (status_polish). It will also be returned with nn = TRUE if a solver (see nn_type) will be used.

Only if comb = "bu", the function returns recf, S and M.

References

Byron, R.P. (1978), The estimation of large social accounts matrices, *Journal of the Royal Statistical Society A*, 141, 3, 359-367.

Di Fonzo, T., and Girolimetto, D. (2021), Cross-temporal forecast reconciliation: Optimal combination method and heuristic alternatives, *International Journal of Forecasting*, in press.

Di Fonzo, T., Marini, M. (2011), Simultaneous and two-step reconciliation of systems of time series: methodological and practical issues, *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 60, 2, 143-164

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

Kourentzes, N., Athanasopoulos, G. (2021), Elucidate structure in intermittent demand series, *European Journal of Operational Research*, 288, 1, pp. 141–152.

Schäfer, J.L., Opgen-Rhein, R., Zuber, V., Ahdesmaki, M., Duarte Silva, A.P., Strimmer, K. (2017), *Package 'corpcor'*, R package version 1.6.9 (April 1, 2017), https://CRAN.R-project.org/package=corpcor.

Schäfer, J.L., Strimmer, K. (2005), A Shrinkage Approach to Large-Scale Covariance Matrix Estimation and Implications for Functional Genomics, *Statistical Applications in Genetics and Molecular Biology*, 4, 1.

Stellato, B., Banjac, G., Goulart, P., Bemporad, A., Boyd, S. (2020). OSQP: An Operator Splitting Solver for Quadratic Programs, *Mathematical Programming Computation*, 12, 4, 637-672.

Stellato, B., Banjac, G., Goulart, P., Boyd, S., Anderson, E. (2019), OSQP: Quadratic Programming Solver using the 'OSQP' Library, R package version 0.6.0.3 (October 10, 2019), https://CRAN.R-project.org/package=osqp.

van Erven, T., Cugliari, J. (2015), Game-theoretically Optimal Reconciliation of Contemporaneous Hierarchical Time Series Forecasts, in Antoniadis, A., Poggi, J.M., Brossat, X. (eds.), *Modeling and Stochastic Learning for Forecasting in High Dimensions*, Berlin, Springer, 297-317.

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.

See Also

Other reconciliation procedures: `cstrec()`, `ctbu()`, `iterec()`, `lccrec()`, `octrec()`, `tcsrec()`, `tdrec()`, `thfrec()`

Examples

```
data(FoReco_data)
# monthly base forecasts
id <- which(simplify2array(strsplit(colnames(FoReco_data$base),split = "_"))[1, ] == "k1")
mbase <- t(FoReco_data$base[, id])
# monthly residuals
id <- which(simplify2array(strsplit(colnames(FoReco_data$res),split = "_"))[1, ] == "k1")
mres <- t(FoReco_data$res[, id])
obj <- htsrec(mbase, C = FoReco_data$C, comb = "shr", res = mres)

# FoReco is able to work also with covariance matrix that are not equal
# across all the forecast horizon. For example, we can consider the
# normalized squared differences (see Di Fonzo and Marini, 2011) where
# Wh = diag(|yh|):
Wh <- lapply(split(mbase, row(mbase)), function(x) diag(abs(x)))

# Now we can introduce the list of the covariance matrix in htsrec throught
# the parameter "W" and setting comb = "w".
objh <- htsrec(mbase, C = FoReco_data$C, W = Wh, comb = "w")
```

---

hts_tools *Cross-sectional reconciliation tools*

Description

Some useful tools for the cross-sectional forecast reconciliation of a linearly constrained (e.g., hierarchical/grouped) multiple time series.

Usage

```
hts_tools(C, h = 1, Ut, nb, sparse = TRUE)
```

Arguments

`C`	$(n_a \times n_b)$ cross-sectional (contemporaneous) matrix mapping the bottom level series into the higher level ones.
`h`	Forecast horizon (*default* is 1).

Ut	Zero constraints cross-sectional (contemporaneous) kernel matrix $(\mathbf{U}'\mathbf{y} = \mathbf{0})$ spanning the null space valid for the reconciled forecasts. It can be used instead of parameter C, but nb is needed if $\mathbf{U}' \neq [\mathbf{I} \;\; -\mathbf{C}]$. If the hierarchy admits a structural representation, $\mathbf{U}'$ has dimension $(n_a \times n)$.
nb	Number of bottom time series; if C is present, nb and Ut are not used.
sparse	Option to return sparse matrices (*default* is TRUE).

Value

A list of five elements:

C	$(n \times n_b)$ cross-sectional (contemporaneous) aggregation matrix.
S	$(n \times n_b)$ cross-sectional (contemporaneous) summing matrix, $\mathbf{S} = \begin{bmatrix} \mathbf{C} \\ \mathbf{I}_{n_b} \end{bmatrix}$.
Ut	$(n_a \times n)$ zero constraints cross-sectional (contemporaneous) kernel matrix. If the hierarchy admits a structural representation $\mathbf{U}' = [\mathbf{I} \;\; -\mathbf{C}]$
n	Number of variables $n_a + n_b$.
na	Number of upper level variables.
nb	Number of bottom level variables.

See Also

Other utilities: Cmatrix(), FoReco2ts(), commat(), ctf_tools(), lcmat(), oct_bounds(), score_index(), shrink_estim(), thf_tools()

Examples

```
# One level hierarchy (na = 1, nb = 2)
obj <- hts_tools(C = matrix(c(1, 1), 1), sparse = FALSE)
```

---

iterec — *Iterative heuristic cross-temporal forecast reconciliation*

Description

Iterative procedure which produces cross-temporally reconciled forecasts by alternating forecast reconciliation along one single dimension (either cross-sectional or temporal) at each iteration step. **Each iteration** consists in the first two steps of the heuristic procedure by Kourentzes and Athanasopoulos (2019), so the forecasts are reconciled by alternating cross-sectional (contemporaneous) reconciliation, and reconciliation through temporal hierarchies in a cyclic fashion. The choice of the dimension along which the first reconciliation step in each iteration is performed is up to the user (param start_rec), and there is no particular reason why one should perform the temporal reconciliation first, and the cross-sectional reconciliation then. The iterative procedure allows the user to get non-negative reconciled forecasts.

Usage

```
iterec(basef, thf_comb, hts_comb, res, itmax = 100, tol = 1e-5,
       start_rec = "thf", norm = "inf", note = TRUE, plot = "mti", ...)
```

Arguments

basef
: $(n \times h(k^* + m))$ matrix of base forecasts to be reconciled, $\widehat{\mathbf{Y}}$; n is the total number of variables, m is the highest time frequency, k^* is the sum of (a subset of) $(p-1)$ factors of m, excluding m, and h is the forecast horizon for the lowest frequency time series. Each row identifies a time series, and the forecasts are ordered as [lowest_freq' ... highest_freq']'.

hts_comb, thf_comb
: Type of covariance matrix (respectively $(n \times n)$ and $((k^* + m) \times (k^* + m))$) to be used in the cross-sectional and temporal reconciliation, see more in `comb` param of `htsrec()` and `thfrec()`.

res
: $(n \times N(k^* + m))$ matrix containing the residuals at all the temporal frequencies ordered [lowest_freq' ... highest_freq']' (columns) for each variable (row), needed to estimate the covariance matrix when `hts_comb = {"wls", "shr", "sam"}` and/or `hts_comb = {"wlsv", "wlsh", "acov", "strar1", "sar1", "har1", "shr", "sam"}`. The row must be in the same order as `basef`.

itmax
: Max number of iteration (`100`, *default*) (old version `maxit`).

tol
: Convergence tolerance (`1e-5`, *default*).

start_rec
: Dimension along with the first reconciliation step in each iteration is performed: it start from temporal reconciliation with `"thf"` (*default*), from cross-sectional with `"hts"` and it does both reconciliation with `"auto"`.

norm
: Norm used to calculate the temporal and the cross-sectional incoherence. There are two alternatives: `"inf"` ($\max |x_1|, |x_2|, \ldots$, *default*) or `"one"` ($\sum |x_i|$).

note
: If `note = TRUE` (*default*) the function writes some notes to the console, otherwise no note is produced (also no plot).

plot
: Some useful plots: `"mti"` (*default*) marginal trend inconsistencies, `"pat"` step by step inconsistency pattern for each iteration, `"distf"` distance forecasts iteration i and i-1, `"all"` all the plots.

...
: any other options useful for `htsrec()` and `thfrec()`, e.g. `m`, `C` (or `Ut` and `nb`), `nn` (for non negativity reconciliation only at first step), `mse`, `corpcor`, `type`, `sol`, `settings`, `W`, `Omega`,...

Details

This reconciliation procedure can be seen as an extension of the well known iterative proportional fitting procedure (Deming and Stephan, 1940, Johnston and Pattie, 1993), also known as RAS method (Miller and Blair, 2009), to adjust the internal cell values of a two-dimensional matrix until they sum to some predetermined row and column totals. In that case the adjustment follows a proportional adjustment scheme, whereas in the cross-temporal reconciliation framework each adjustment step is made according to the penalty function associated to the single-dimension reconciliation procedure adopted.

Control status of iterative reconciliation:

- `-2` Temporal/Cross-sectional reconciliation does not work.
- `-1` Convergence not achieved (maximum iteration limit reached).
- `0` Convergence achieved.
- `+1` Convergence achieved: incoherence has increased in the next iteration (at least one time).
- `+2` Convergence achieved: incoherence has increased in the next two or more iteration (at least one time).
- `+3` The forecasts are already reconciled.

Value

`iterec` returns a list with:

recf	$(n \times h(k^* + m))$ reconciled forecasts matrix, $\widetilde{\mathbf{Y}}$.
d_cs	Cross-sectional incoherence at each iteration.
d_te	Temporal incoherence at each iteration.
start_rec	Starting coherence dimension (thf or hts).
tol	Tolerance.
flag	Control code (see *details*).
time	Elapsed time.
dist	If `start_rec = "auto"`, matrix of distances of the forecasts reconciled from the base.

References

Deming, E., Stephan, F.F. (1940), On a least squares adjustment of a sampled frequency table when the expected marginal totals are known, *The Annals of Mathematical Statistics*, 11, 4, 427–444.

Di Fonzo, T., and Girolimetto, D. (2021), Cross-temporal forecast reconciliation: Optimal combination method and heuristic alternatives, *International Journal of Forecasting*, in press.

Johnston, R.J., Pattie, C.J. (1993), Entropy-Maximizing and the Iterative Proportional Fitting Procedure, *The Professional Geographer*, 45, 3, 317–322.

Kourentzes, N., Athanasopoulos, G. (2019), Cross-temporal coherent forecasts for Australian tourism, *Annals of Tourism Research*, 75, 393-409.

Miller, R.E., Blair, P.D. (2009), Input-output analysis: foundations and extensions, 2nd edition, New York, Cambridge University Press.

Schäfer, J.L., Opgen-Rhein, R., Zuber, V., Ahdesmaki, M., Duarte Silva, A.P., Strimmer, K. (2017), *Package 'corpcor'*, R package version 1.6.9 (April 1, 2017), https://CRAN.R-project.org/package=corpcor.

Schäfer, J.L., Strimmer, K. (2005), A Shrinkage Approach to Large-Scale Covariance Matrix Estimation and Implications for Functional Genomics, *Statistical Applications in Genetics and Molecular Biology*, 4, 1.

Stellato, B., Banjac, G., Goulart, P., Bemporad, A., Boyd, S. (2020). OSQP: An Operator Splitting Solver for Quadratic Programs, *Mathematical Programming Computation*, 12, 4, 637-672.

Stellato, B., Banjac, G., Goulart, P., Boyd, S., Anderson, E. (2019), OSQP: Quadratic Programming Solver using the 'OSQP' Library, R package version 0.6.0.3 (October 10, 2019), https://CRAN.R-project.org/package=osqp.

See Also

Other reconciliation procedures: `cstrec()`, `ctbu()`, `htsrec()`, `lccrec()`, `octrec()`, `tcsrec()`, `tdrec()`, `thfrec()`

Examples

```
data(FoReco_data)
obj <- iterec(FoReco_data$base, note = FALSE,
  m = 12, C = FoReco_data$C, thf_comb = "acov",
  hts_comb = "shr", res = FoReco_data$res, start_rec = "thf")
```

`lccrec` *Level conditional coherent forecast reconciliation for genuine hierarchical/grouped time series*

Description

Forecast reconciliation procedure built on and extending the original proposal by Hollyman et al. (2021). Level conditional coherent reconciled forecasts may be computed in cross-sectional, temporal, and cross-temporal frameworks. The reconciled forecasts are conditional to (i.e., constrained by) the base forecasts of a specific upper level of the hierarchy (exogenous constraints). The linear constraints linking the variables may be dealt with endogenously as well (Di Fonzo and Girolimetto, 2022). *Combined Conditional Coherent* (CCC) forecasts may be calculated as simple averages of LCC and bottom-up reconciled forecasts, with either endogenous or exogenous constraints.

Usage

```
lccrec(basef, m, C, nl, weights, bnaive = NULL, const = "exogenous",
       CCC = TRUE, nn = FALSE, nn_type = "osqp", settings = list(), ...)
```

Arguments

`basef`	matrix/vector of base forecasts to be reconciled: $(h \times n)$ matrix in the cross-sectional framework; $(h(k^* + m) \times 1)$ vector in the temporal framework; $(n \times h(k^* + m))$ matrix in the cross-temporal framework. n is the total number of variables, m is the highest time frequency, k^* is the sum of (a subset of) $(p-1)$ factors of m, excluding m, and h is the forecast horizon.
`m`	Highest available sampling frequency per seasonal cycle (max. order of temporal aggregation, m), or a subset of the p factors of m.
`C`	$(n_a \times n_b)$ cross-sectional (contemporaneous) matrix mapping the bottom level series into the higher level ones (or a list of matrices forming $\mathbf{C} = [\mathbf{C}_1' \ \mathbf{C}_2' \ ... \ \mathbf{C}_L']'$, $1, ..., L$ being the number of the cross-sectional upper levels.
`nl`	$(L \times 1)$ vector containing the number of time series in each level of the hierarchy (`nl[1] = 1`).
`weights`	covariance matrix or a vector (weights used in the reconciliation: either $(n_b \times 1)$ for exogenous or $(n \times 1)$ for endogenous constraints).
`bnaive`	matrix/vector of naive bts base forecasts (e.g., seasonal averages, as in Hollyman et al., 2021): $(h \times n_b)$ matrix in the cross-sectional framework; $(hm \times 1)$ vector in the temporal framework; $(n_b \times hm)$ matrix in the cross-temporal framework. Ignore it, if only basef are to be used as base forecasts.
`const`	**exo**genous (*default*) or **endo**genous constraints
`CCC`	Option to return Combined Conditional Coherent reconciled forecasts (*default* is TRUE).
`nn`	Logical value: `TRUE` if non-negative reconciled forecasts are wished.
`nn_type`	Non-negative method: "osqp" (*default*) or "sntz" (*set-negative-to-zero*, only if `CCC = TRUE`) with exogenous constraints (`const = "exo"`); "osqp" (*default*), "KAnn" (only `type == "M"`) or "sntz" with endogenous constraints (`const = "endo"`).
`settings`	Settings for **osqp** (object `osqpSettings`). The default options are: `verbose = FALSE`, `eps_abs = 1e-5`, `eps_rel = 1e-5`, `polish_refine_iter = 100` and `polish = TRUE`. For details, see the **osqp** documentation (Stellato et al., 2019).
`...`	Additional functional arguments passed to htsrec of FoReco.

Details

Cross-sectional hierarchies

To be as simple as possible, we fix the forecast horizon equal to 1. Let the base forecasts be a vector $\widehat{\mathbf{y}} = \left[\widehat{\mathbf{a}}' \; \widehat{\mathbf{b}}'\right]'$, where vector $\widehat{\mathbf{a}}$ consists of the sub-vectors forming each of the upper L levels of the hierarchy/grouping:

$$\widehat{\mathbf{a}} = \begin{bmatrix} \widehat{a}_1 \\ \widehat{\mathbf{a}}_2 \\ \vdots \\ \widehat{\mathbf{a}}_L \end{bmatrix},$$

where $\widehat{\mathbf{a}}_l$, $l = 1, \dots, L$, has dimension $(n_l \times 1)$ and $\sum_{l=1}^{L} n_l = n_a$. Denote $\mathbf{C}_l$ the $(n_l \times n_b)$ matrix mapping the bts into the level-l uts, then the aggregation matrix $\mathbf{C}$ may be written as

$$\mathbf{C} = \begin{bmatrix} \mathbf{C}_1 \\ \mathbf{C}_2 \\ \vdots \\ \mathbf{C}_L \end{bmatrix},$$

where the generic matrix $\mathbf{C}_L$ is $(n_L \times n_b)$, $l = 1, \dots, L$. Given a generic level l, the reconciled forecasts coherent with the base forecasts of that level are the solution to a quadratic minimization problem with linear exogenous constraints (`const = "exo"`):

$$\widetilde{\mathbf{b}}_l = \arg\min_{\mathbf{b}} \left(\mathbf{b} - \widehat{\mathbf{b}}\right)' \mathbf{W}_b^{-1} \left(\mathbf{b} - \widehat{\mathbf{b}}\right) \quad \text{s.t. } \mathbf{C}_l\mathbf{b} = \widehat{\mathbf{a}}_l, \quad l = 1, \dots, L,$$
$$\Downarrow$$
$$\widetilde{\mathbf{b}}_l = \widehat{\mathbf{b}} + \mathbf{W}_b\mathbf{C}_l' \left(\mathbf{C}_l\mathbf{W}_b\mathbf{C}_l'\right)^{-1} \left(\widehat{\mathbf{a}}_l - \mathbf{C}_l\widehat{\mathbf{b}}\right), \quad l = 1, \dots, L,$$

where $\mathbf{W}_b$ is a $(n_b \times n_b)$ p.d. matrix (in Hollyman et al., 2021, $\mathbf{W}_b$ is diagonal). If endogenous constraints (`const = "endo"`) are considered, denote $\widehat{\mathbf{y}}_l = \left[\widehat{\mathbf{a}}_l' \; \widehat{\mathbf{b}}'\right]'$ and $\mathbf{U}_l' = \left[\mathbf{I}_{n_l}' \; \mathbf{C}_l'\right]'$, then

$$\widetilde{\mathbf{y}}_l = \arg\min_{\mathbf{y}_l} \left(\mathbf{y}_l - \widehat{\mathbf{y}}_l\right)' \mathbf{W}_l^{-1} \left(\mathbf{y}_l - \widehat{\mathbf{y}}_l\right) \quad \text{s.t. } \mathbf{U}_l'\mathbf{y}_l = \mathbf{0}, \quad l = 1, \dots, L,$$
$$\Downarrow$$
$$\widetilde{\mathbf{y}}_l = \left(\mathbf{I}_{n_b+n_l} - \mathbf{W}_l\mathbf{U}_l \left(\mathbf{U}_l'\mathbf{W}_l\mathbf{U}_l\right)^{-1} \mathbf{U}_l'\right) \widehat{\mathbf{y}}_l, \quad l = 1, ..., L,$$

where $\mathbf{W}_l$ is a $(n_l + n_b \times n_l + n_b)$ p.d. matrix. Combined Conditional Coherent (CCC) forecasts are obtained by taking the simple average of the L level conditional, and of the bottom-up reconciled forecasts, respectively (Di Fonzo and Girolimetto, 2022):

$$\widetilde{\mathbf{y}}_{CCC} = \frac{1}{L+1} \sum_{l=1}^{L+1} \mathbf{S}\widetilde{\mathbf{b}}_l,$$

with

$$\widetilde{\mathbf{b}}_{L+1} = \widehat{\mathbf{b}}.$$

Non-negative reconciled forecasts may be obtained by setting `nn_type` alternatively as:

- to apply non-negative constraints to each level:
  - `nn_type = "KAnn"` (only `const = "endo"`)
  - `nn_type = "osqp"`
- to apply non-negative constraints only to the CCC forecasts:

– nn_type = "sntz" ("set-negative-to-zero")

Temporal hierarchies

The extension to the case of **temporal hierarchies** is quite simple. Using the same notation as in thfrec(), the following 'equivalences' hold between the symbols used for the level conditional cross-sectional reconciliation and the ones used in temporal reconciliation:

$$L \equiv p - 1, \quad (n_a, n_b, n) \equiv (k^*, m, k^* + m),$$

and

$$\mathbf{C} \equiv \mathbf{K},\ \mathbf{C}_1 \equiv \mathbf{K}_1 = \mathbf{1}'_m,\ \mathbf{C}_2 \equiv \mathbf{K}_2 = \mathbf{I}_{\frac{m}{k_{p-1}}},\ \ldots,\ \mathbf{C}_L \equiv \mathbf{K}_{p-1} = \mathbf{I}_{\frac{m}{k_2}} \otimes \mathbf{1}'_{k_2},\ \mathbf{S} \equiv \mathbf{R}.$$

The description of the **cross-temporal extension** is currently under progress.

Value

The function returns the level reconciled forecasts in case of an elementary hierarchy with one level. Otherwise it returns a list with

recf — Level Conditional Coherent (CCC = FALSE) forecasts matrix/vector calculated as simple averages of upper level conditional reconciled forecasts, with either endogenous or exogenous constraints. If CCC = TRUE then it is the Combined Conditional Coherent matrix/vector, as weighted averages of LCC and bottom-up reconciled forecasts.

levrecf — list of level conditional reconciled forecasts (+ BU).

info (type="osqp") — matrix with some useful indicators (columns) for each forecast horizon h (rows): run time (run_time), number of iteration, norm of primal residual (pri_res), status of osqp's solution (status) and polish's status (status_polish).

References

Di Fonzo, T., and Girolimetto, D. (2021), Cross-temporal forecast reconciliation: Optimal combination method and heuristic alternatives, *International Journal of Forecasting*, in press.

Di Fonzo, T., Girolimetto, D. (2022), Forecast combination based forecast reconciliation: insights and extensions, *International Journal of Forecasting*, in press.

Hollyman, R., Petropoulos, F., Tipping, M.E. (2021), Understanding Forecast Reconciliation, *European Journal of Operational Research* (in press).

See Also

Other reconciliation procedures: cstrec(), ctbu(), htsrec(), iterec(), octrec(), tcsrec(), tdrec(), thfrec()

Examples

```
data(FoReco_data)
### LCC and CCC CROSS-SECTIONAL FORECAST RECONCILIATION
# Cross sectional aggregation matrix
C <- rbind(FoReco_data$C, c(0,0,0,0,1))
# monthly base forecasts
id <- which(simplify2array(strsplit(colnames(FoReco_data$base), split = "_"))[1, ] == "k1")
mbase <- t(FoReco_data$base[, id])[,c("T", "A","B","C","AA","AB","BA","BB","C")]
```

```
# residuals
id <- which(simplify2array(strsplit(colnames(FoReco_data$res), split = "_"))[1, ] == "k1")
mres <- t(FoReco_data$res[, id])[,c("T", "A","B","C","AA","AB","BA","BB","C")]
# covariance matrix of all the base forecasts, computed using the in-sample residuals
Wres <- cov(mres)
# covariance matrix of the bts base forecasts, computed using the in-sample residuals
Wb <- Wres[c("AA","AB", "BA", "BB", "C"),
           c("AA","AB", "BA", "BB", "C")]
# bts seasonal averages
obs_1 <- FoReco_data$obs$k1
bts_sm <- apply(obs_1, 2, function(x) tapply(x[1:168],rep(1:12, 14), mean))
bts_sm <- bts_sm[,c("AA", "AB", "BA", "BB", "C")]

## EXOGENOUS CONSTRAINTS AND DIAGONAL COVARIANCE MATRIX (Hollyman et al., 2021)
# Forecast reconciliation for an elementary hierarchy:
# 1 top-level series + 5 bottom-level series (Level 2 base forecasts are not considered).
# The input is given by the base forecasts of the top and bottom levels series,
# along with a vector of positive weights for the bts base forecasts
exo_EHd <- lccrec(basef = mbase[, c("T","AA","AB", "BA", "BB", "C")],
                 weights = diag(Wb), bnaive = bts_sm)

# Level conditional reconciled forecasts
# recf/L1: Level 1 reconciled forecasts for the whole hierarchy
# L2: Middle-Out reconciled forecasts hinging on Level 2 conditional reconciled forecasts
# L3: Bottom-Up reconciled forecasts
exo_LCd <- lccrec(basef = mbase, C = C, nl = c(1,3), weights = diag(Wb),
                 CCC = FALSE, bnaive = bts_sm)

# Combined Conditional Coherent (CCC) reconciled forecasts
# recf: CCC reconciled forecasts matrix
# L1: Level 1 conditional reconciled forecasts for the whole hierarchy
# L2: Middle-Out reconciled forecasts hinging on Level 2 conditional reconciled forecasts
# L3: Bottom-Up reconciled forecasts
exo_CCCd <- lccrec(basef = mbase, C = C, nl = c(1,3), weights = diag(Wb))

## EXOGENOUS CONSTRAINTS AND FULL COVARIANCE MATRIX
# Simply substitute weights=diag(Wb) with weights=Wb
exo_EHf <- lccrec(basef = mbase[, c("T","AA","AB", "BA", "BB", "C")], weights = Wb, bnaive = bts_sm)
exo_LCf <- lccrec(basef = mbase, C = C, nl = c(1,3), weights = Wb, CCC = FALSE, bnaive = bts_sm)
exo_CCCf <- lccrec(basef = mbase, C = C, nl = c(1,3), weights = Wb, bnaive = bts_sm)

## ENDOGENOUS CONSTRAINTS AND DIAGONAL COVARIANCE MATRIX
# parameters of function htsrec(), like "type" and "nn_type" may be used

# Forecast reconciliation for an elementary hierarchy with endogenous constraints
W1 <- Wres[c("T","AA","AB", "BA", "BB", "C"),
           c("T","AA","AB", "BA", "BB", "C")]
endo_EHd <- lccrec(basef = mbase[, c("T","AA","AB", "BA", "BB", "C")],
                 weights = diag(W1), const = "endo", CCC = FALSE)

# Level conditional reconciled forecasts with endogenous constraints
endo_LCd <- lccrec(basef = mbase, C = C, nl = c(1,3), weights = diag(Wres),
                  const = "endo", CCC = FALSE)

# Combined Conditional Coherent (CCC) reconciled forecasts with endogenous constraints
endo_CCCd <- lccrec(basef = mbase, C = C, nl = c(1,3),
                    weights = diag(Wres), const = "endo")
```

```
## ENDOGENOUS CONSTRAINTS AND FULL COVARIANCE MATRIX
# Simply substitute weights=diag(Wres) with weights=Wres
endo_EHf <- lccrec(basef = mbase[, c("T","AA","AB", "BA", "BB", "C")],
                   weights = W1,
                   const = "endo")
endo_LCf <- lccrec(basef = mbase, C = C, nl = c(1,3),
                   weights = Wres, const = "endo", CCC = FALSE)
endo_CCCf <- lccrec(basef = mbase-40, C = C, nl = c(1,3),
                   weights = Wres, const = "endo")

### LCC and CCC TEMPORAL FORECAST RECONCILIATION
# top ts base forecasts ([lowest_freq' ...  highest_freq']')
topbase <- FoReco_data$base[1, ]
# top ts residuals ([lowest_freq' ...  highest_freq']')
topres <- FoReco_data$res[1, ]
Om_bt <- cov(matrix(topres[which(simplify2array(strsplit(names(topres),
            "_"))[1,]=="k1")], ncol = 12, byrow = TRUE))
t_exo_LCd <- lccrec(basef = topbase, m = 12, weights = diag(Om_bt), CCC = FALSE)
t_exo_CCCd <- lccrec(basef = topbase, m = 12, weights = diag(Om_bt))

### LCC and CCC CROSS-TEMPORAL FORECAST RECONCILIATION
idr <- which(simplify2array(strsplit(colnames(FoReco_data$res), "_"))[1,]=="k1")
bres <- FoReco_data$res[-c(1:3), idr]
bres <- lapply(1:5, function(x) matrix(bres[x,], nrow=14, byrow = TRUE))
bres <- do.call(cbind, bres)
ctbase <- FoReco_data$base[c("T", "A","B","C","AA","AB","BA","BB","C"),]
ct_exo_LCf <- lccrec(basef = ctbase, m = 12, C = C, nl = c(1,3),
                    weights = diag(cov(bres)), CCC = FALSE)
ct_exo_CCCf <- lccrec(basef = ctbase, m = 12, C = C, nl = c(1,3),
                     weights = diag(cov(bres)), CCC = TRUE)
```

`lcmat` *Linear Combination Matrix for a general linearly constrained multiple time series*

Description

When working with a general linearly constrained multiple (n-variate) time series ($\mathbf{x}_t$), getting a linear combination matrix $\bar{\mathbf{C}}$ is a critical step to obtain a *structural-like* representation such that, for $t = 1, ..., T$,

$$\bar{\mathbf{U}}' = [\mathbf{I} \quad -\bar{\mathbf{C}}] \quad \Rightarrow \quad \mathbf{y}_t = \mathbf{P}\mathbf{x}_t = \begin{bmatrix} \mathbf{v}_t \\ \mathbf{f}_t \end{bmatrix} = \begin{bmatrix} \bar{\mathbf{C}} \\ \mathbf{I} \end{bmatrix} \mathbf{f}_t = \bar{\mathbf{S}}\mathbf{f}_t,$$

where $\bar{\mathbf{U}}'$ is the $(n_v \times n)$ full rank zero constraints matrix, $\bar{\mathbf{S}}$ is the $(n \times n_f)$ matrix analogous of the summing matrix $\mathbf{S}$ for a genuine hierarchical/groupped times series, $\bar{\mathbf{C}}$ is the $(n_v \times n_f)$ linear combination matrix such that $\mathbf{v}_t = \bar{\mathbf{C}}\mathbf{f}_t$, $\mathbf{v}_t$ is the $(n_v \times 1)$ vector of 'basic' variables, and $\mathbf{f}_t$ is the $(n_f \times 1)$ vector of 'free' variables (Di Fonzo and Girolimetto, 2022).

Usage

```
lcmat(Gt, alg = "rref", tol = sqrt(.Machine$double.eps),
       verbose = FALSE, sparse = TRUE)
```

Arguments

`Gt`	$(r \times n)$ coefficient matrix $(\mathbf{\Gamma}')$ for a general linearly constrained multiple time series $(\mathbf{x}_t)$ such that $\mathbf{\Gamma}'\mathbf{x}_t = \mathbf{0}_{(r\times 1)}$.
`alg`	Technique used into trasform $\mathbf{\Gamma}'$ in $\bar{\mathbf{U}}' = [\mathbf{I} \quad -\bar{\mathbf{C}}]$, such that $\bar{\mathbf{U}}'\mathbf{y}_t = \mathbf{0}_{(n_v\times 1)}$. Use `"rref"` for the Row Reduced Echelon Form through Gauss-Jordan elimination (*default*), or `"qr"` for the (pivoting) QR decomposition (Strang, 2019).
`tol`	Tolerance for the `"rref"` or `"qr"` algorithm.
`verbose`	If `TRUE`, intermediate steps are printed (*default* is `FALSE`).
`sparse`	Option to return a sparse $\bar{\mathbf{C}}$ matrix (*default* is `TRUE`).

Details

Looking for an analogous of the summing matrix $\mathbf{S}$, say $\bar{\mathbf{S}} = \begin{bmatrix} \bar{\mathbf{C}} \\ \mathbf{I} \end{bmatrix}$, the `lcmat` function transforms $\mathbf{\Gamma}'$ into $\bar{\mathbf{U}}' = [\mathbf{I} \quad -\bar{\mathbf{C}}]$, such that $\bar{\mathbf{U}}'\mathbf{y}_t = \mathbf{0}_{(n_v\times 1)}$. Consider the simple example of a linearly constrained multiple time series consisting of two hierarchies, each with distinct bottom time series, with a common top-level series (X):

$$\begin{array}{l} 1)\ X = C + D, \\ 2)\ X = A + B, \\ 3)\ A = A1 + A2. \end{array}$$

The coefficient matrix $\mathbf{\Gamma}'$ of the linear system $\mathbf{\Gamma}'\mathbf{x}_t = \mathbf{0}$ ($\mathbf{x}_t = \{X, C, D, A, B, A1, A2\}$) is

$$\mathbf{\Gamma}' = \begin{bmatrix} 1 & -1 & -1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & -1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & -1 & -1 \end{bmatrix}.$$

The lcmat function returns

$$\bar{\mathbf{C}} = \begin{bmatrix} 0 & 1 & 1 & 1 \\ -1 & 1 & 1 & 1 \\ 0 & 0 & -1 & -1 \end{bmatrix}.$$

Then

$$\bar{\mathbf{U}}' = \left[\begin{array}{ccc|cccc} 1 & 0 & 0 & 0 & -1 & -1 & -1 \\ 0 & 1 & 0 & 1 & -1 & -1 & -1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 \end{array}\right], \quad \text{with} \quad \bar{\mathbf{U}}'\mathbf{y}_t = \bar{\mathbf{U}}' \begin{bmatrix} \mathbf{v}_t \\ \mathbf{f}_t \end{bmatrix} = \mathbf{0},$$

where $\mathbf{v}_t = \{X, C, A\}$, and $\mathbf{f}_t = \{D, B, A1, A2\}$.

Value

A list with

`Cbar`	$(n_v \times n_f)$ linear combination matrix $\bar{\mathbf{C}}$
`pivot`	$(n \times 1)$ vector of the column permutations s.t. $\mathbf{P} = \mathbf{I}[, \text{pivot}]$

References

Di Fonzo, T., Girolimetto, D. (2022), *Point and probabilistic forecast reconciliation for general linearly constrained multiple time series* (mimeo).

Strang, G., (2019), *Linear algebra and learning from data*, Wellesley, Cambridge Press.

See Also

Other utilities: `Cmatrix()`, `FoReco2ts()`, `commat()`, `ctf_tools()`, `hts_tools()`, `oct_bounds()`, `score_index()`, `shrink_estim()`, `thf_tools()`

Examples

```
Gt <- matrix(c(1,-1,-1,0,0,0,0,
               1,0,0,-1,-1,0,0,
               0,0,0,1,0,-1,-1), nrow = 3, byrow = TRUE)
Cbar <- lcmat(Gt = Gt)$Cbar
P <- diag(1, NCOL(Gt))[,lcmat(Gt = Gt)$pivot]
```

---

octrec — *Optimal combination cross-temporal forecast reconciliation*

Description

Optimal (in least squares sense) combination cross-temporal forecast reconciliation. The reconciled forecasts are calculated either through a projection approach (Byron, 1978), or the equivalent structural approach by Hyndman et al. (2011).

Usage

```
octrec(basef, m, C, comb, res, Ut, nb, mse = TRUE, corpcor = FALSE,
       type = "M", sol = "direct", keep = "list", v = NULL, nn = FALSE,
       nn_type = "osqp", settings = list(), bounds = NULL, W = NULL,
       Omega = NULL)
```

Arguments

`basef` — $(n \times h(k^* + m))$ matrix of base forecasts to be reconciled, $\widehat{\mathbf{Y}}$; n is the total number of variables, m is the highest time frequency, k^* is the sum of (a subset of) $(p-1)$ factors of m, excluding m, and h is the forecast horizon for the lowest frequency time series. Each row identifies a time series, and the forecasts are ordered as [lowest_freq' ... highest_freq']'.

`m` — Highest available sampling frequency per seasonal cycle (max. order of temporal aggregation, m), or a subset of p factors of m.

`C` — $(n_a \times n_b)$ cross-sectional (contemporaneous) matrix mapping the bottom level series into the higher level ones.

`comb` — Type of the reconciliation. It corresponds to a specific $(n(k*+m) \times n(k^*+m))$ covariance matrix, where k^* is the sum of (a subset of) $(p-1)$ factors of m (m is not considered) and n is the number of variables:

- **ols** (Identity);
- **struc** (Cross-temporal structural variances);
- **cs_struc** (Cross-sectional structural variances and temporal independence);
- **t_struc** (Cross-sectional independence and temporal structural variances);
- **wlsh** (Hierarchy variances);
- **wlsv** (Series variances);

- **bdshr** (Shrunk cross-covariance matrix, cross-sectional framework);
- **bdsam** (Sample cross-covariance matrix, cross-sectional framework);
- **acov** (Series auto-covariance matrix);
- **Sshr** (Series shrunk cross-covariance matrix);
- **Ssam** (Series cross-covariance matrix);
- **shr** (Shrunk cross-covariance matrix);
- **sam** (Sample cross-covariance matrix);
- **w** use your personal matrix W in param `W`;
- **omega** use your personal matrix Omega in param `Omega`.

`res` $(n \times N(k^* + m))$ matrix containing the residuals at all the temporal frequencies ordered [lowest_freq' ... highest_freq']' (columns) for each variable (row), needed to estimate the covariance matrix when `comb = {"sam", "wlsv", "wlsh", "acov", "Ssam", "Sshr", "Sshr1", "shr"}`.

`Ut` Zero constraints cross-sectional (contemporaneous) kernel matrix $(\mathbf{U}'\mathbf{y} = \mathbf{0})$ spanning the null space valid for the reconciled forecasts. It can be used instead of parameter `C`, but `nb` $(n = n_a + n_b)$ is needed if $\mathbf{U}' \neq [\mathbf{I} - \mathbf{C}]$. If the hierarchy admits a structural representation, $\mathbf{U}'$ has dimension $(n_a \times n)$.

`nb` Number of bottom time series; if `C` is present, `nb` and `Ut` are not used.

`mse` Logical value: `TRUE` (*default*) calculates the covariance matrix of the in-sample residuals (when necessary) according to the original **hts** and **thief** formulation: no mean correction, T as denominator.

`corpcor` Logical value: `TRUE` if **corpcor** (Schäfer et al., 2017) must be used to shrink the sample covariance matrix according to Schäfer and Strimmer (2005), otherwise the function uses the same implementation as package **hts**.

`type` Approach used to compute the reconciled forecasts: `"M"` for the projection approach with matrix M (*default*), or `"S"` for the structural approach with summing matrix S.

`sol` Solution technique for the reconciliation problem: either `"direct"` (*default*) for the closed-form matrix solution, or `"osqp"` for the numerical solution (solving a linearly constrained quadratic program using `solve_osqp`).

`keep` Return a list object of the reconciled forecasts at all levels (if keep = "list") or only the reconciled forecasts matrix (if keep = "recf").

`v` vector index of the fixed base forecast ($\min(v) > 0$ and $\max(v) < n(k^* + m)$).

`nn` Logical value: `TRUE` if non-negative reconciled forecasts are wished.

`nn_type` "osqp" (default), "KAnn" (only `type == "M"`) or "sntz".

`settings` Settings for **osqp** (object `osqpSettings`). The default options are: `verbose = FALSE`, `eps_abs = 1e-5`, `eps_rel = 1e-5`, `polish_refine_iter = 100` and `polish = TRUE`. For details, see the **osqp** documentation (Stellato et al., 2019).

`bounds` $(n(k^* + m) \times 2)$ matrix of the bounds on the variables: the first column is the lower bound, and the second column is the upper bound.

`W, Omega` This option permits to directly enter the covariance matrix:

1. `W` must be a p.d. $(n(k^* + m) \times n(k^* + m))$ matrix or a list of h matrix (one for each forecast horizon);
2. `Omega` must be a p.d. $(n(k^* + m) \times n(k^* + m))$ matrix or a list of h matrix (one for each forecast horizon);
3. if `comb` is different from "`w`" or "`omega`", `W` or `Omega` is not used.

Details

Considering contemporaneous and temporal dimensions in the same framework requires to extend and adapt the notations used in htsrec and thfrec. To do that, we define the matrix containing the base forecasts at any considered temporal frequency as

$$\widehat{\mathbf{Y}}_{n\times h(k^*+m))} = \begin{bmatrix} \widehat{\mathbf{A}}^{[m]} & \widehat{\mathbf{A}}^{[k_{p-1}]} & \dots & \widehat{\mathbf{A}}^{[k_2]} & \widehat{\mathbf{A}}^{[1]} \\ \widehat{\mathbf{B}}^{[m]} & \widehat{\mathbf{B}}^{[k_{p-1}]} & \dots & \widehat{\mathbf{B}}^{[k_2]} & \widehat{\mathbf{B}}^{[1]} \end{bmatrix} \qquad k \in \mathcal{K},$$

where $\mathcal{K}$ is a subset of p factors of m and, $\widehat{\mathbf{B}}^{[k]}$ and $\widehat{\mathbf{A}}^{[k]}$ are the matrices containing the k-order temporal aggregates of the bts and uts, of dimension $(n_b \times hm/k)$ and $(n_a \times hm/k)$, respectively.

Let us consider the multivariate regression model

$$\widehat{\mathbf{Y}} = \mathbf{Y} + \mathbf{E},$$

where the involved matrices have each dimension $[n \times (k^* + m)]$ and contain, respectively, the base ($\widehat{\mathbf{Y}}$) and the target forecasts ($\mathbf{Y}$), and the coherency errors ($\mathbf{E}$) for the n component variables of the linearly constrained time series of interest. For each variable, $k^* + m$ base forecasts are available, pertaining to all aggregation levels of the temporal hierarchy for a complete cycle of high-frequency observation, m. Consider now two vectorized versions of model, by transforming the matrices either in original form:

$$\text{vec}\left(\widehat{\mathbf{Y}}\right) = \text{vec}\left(\mathbf{Y}\right) + \varepsilon \;\; \text{with} \;\; \varepsilon = \text{vec}\left(\mathbf{E}\right)$$

or in transposed form:

$$\text{vec}\left(\widehat{\mathbf{Y}}'\right) = \text{vec}\left(\mathbf{Y}'\right) + \eta \;\; \text{with} \;\; \eta = \text{vec}\left(\mathbf{E}'\right).$$

Denote with $\mathbf{P}$ the $[n(k^* + m) \times n(k^* + m)]$ commutation matrix such that $\mathbf{P}\text{vec}(\mathbf{Y}) = \text{vec}(\mathbf{Y}')$, $\mathbf{P}\text{vec}(\widehat{\mathbf{Y}}) = \text{vec}(\widehat{\mathbf{Y}}')$ and $\mathbf{P}\varepsilon = \eta$. Let $\mathbf{W} = \mathrm{E}[\varepsilon\varepsilon']$ be the covariance matrix of vector ε, and $\mathbf{\Omega} = \mathrm{E}[\eta\eta']$ the covariance matrix of vector η. Clearly, $\mathbf{W}$ and $\mathbf{\Omega}$ are different parameterizations of the same statistical object for which the following relationships hold:

$$\mathbf{\Omega} = \mathbf{PWP}', \qquad \mathbf{W} = \mathbf{P}'\mathbf{\Omega}\mathbf{P}.$$

In order to apply the general point forecast reconciliation according to the projection approach (`type = "M"`) to a cross-temporal forecast reconciliation problem, we may consider either two *vec*-forms , e.g. if we follow the first:

$$\tilde{\mathbf{y}} = \hat{\mathbf{y}} - \mathbf{\Omega}\mathbf{H}\left(\mathbf{H}'\mathbf{\Omega}\mathbf{H}\right)^{-1}\mathbf{H}'\hat{\mathbf{y}} = \mathbf{M}\hat{\mathbf{y}},$$

where $\widehat{\mathbf{y}} = \text{vec}(\widehat{\mathbf{Y}}')$ is the row vectorization of the base forecasts matrix $\widehat{\mathbf{Y}}$ The alternative equivalent solution (`type = "S"`) (following the structural reconciliation approach by Hyndman et al., 2011) is

$$\widetilde{\mathbf{y}} = \widetilde{\mathbf{S}}\left(\widetilde{\mathbf{S}}'\mathbf{\Omega}^{-1}\widetilde{\mathbf{S}}\right)^{-1}\widetilde{\mathbf{S}}'\mathbf{\Omega}^{-1}\widehat{\mathbf{y}} = \widetilde{\mathbf{S}}\mathbf{G}\widehat{\mathbf{y}}.$$

where $\widetilde{\mathbf{S}}$ is the cross-temporal summing matrix.

Bounds on the reconciled forecasts

When the reconciliation uses the optimization package osqp, the user may impose bounds on the reconciled forecasts. The parameter bounds permits to consider lower ($\mathbf{a}$) and upper ($\mathbf{b}$) bounds like $\mathbf{a} \leq \widetilde{\mathbf{y}} \leq \mathbf{b}$, where $\widehat{\mathbf{y}} = \text{vec}(\widehat{\mathbf{Y}}')$, such that:

$$\begin{array}{c} a_1 \leq \widetilde{y}_1 \leq b_1 \\ \dots \\ a_{n(k^*+m)} \leq \widetilde{y}_{n(k^*+m)} \leq b_{n(k^*+m)} \end{array} \Rightarrow \text{bounds} = [\mathbf{a}\;\mathbf{b}] = \begin{bmatrix} a_1 & b_1 \\ \vdots & \vdots \\ a_{n(k^*+m)} & b_{n(k^*+m)} \end{bmatrix},$$

where $a_i \in [-\infty, +\infty]$ and $b_i \in [-\infty, +\infty]$. If y_i is unbounded, the i-th row of bounds would be equal to `c(-Inf, +Inf)`. Notice that if the bounds parameter is used, `sol = "osqp"` must be used. This is not true in the case of non-negativity constraints:

- `sol = "direct"`: first the base forecasts are reconciled without non-negativity constraints, then, if negative reconciled values are present, the `"osqp"` solver is used;
- `sol = "osqp"`: the base forecasts are reconciled using the `"osqp"` solver.

In this case it is not necessary to build a matrix containing the bounds, and it is sufficient to set `nn = "TRUE"`.

Non-negative reconciled forecasts may be obtained by setting `nn_type` alternatively as:

- `nn_type = "KAnn"` (Kourentzes and Athanasopoulos, 2021)
- `nn_type = "sntz"` ("set-negative-to-zero")
- `nn_type = "osqp"` (Stellato et al., 2020)

Value

If the parameter `keep` is equal to `"recf"`, then the function returns only the $(n \times h(k^* + m))$ reconciled forecasts matrix, otherwise (`keep="all"`) it returns a list that mainly depends on what type of representation (`type`) and solution technique (`sol`) have been used:

`recf` $(n \times h(k^* + m))$ reconciled forecasts matrix, $\widetilde{\mathbf{Y}}$.

`Omega` Covariance matrix used for reconciled forecasts (vec$(\widehat{\mathbf{Y}}^{'})$ representation).

`W` Covariance matrix used for reconciled forecasts (vec$(\widehat{\mathbf{Y}})$ representation).

`nn_check` Number of negative values (if zero, there are no values below zero).

`rec_check` Logical value: `rec_check = TRUE` when the constraints have been fulfilled,

`varf (type="direct")`
$(n \times (k^* + m))$ reconciled forecasts variance matrix for $h = 1$, diag$(\mathbf{MW})$.

`M (type="direct")`
Projection matrix (projection approach).

`G (type="S" and type="direct")`
Projection matrix (structural approach, $\mathbf{M} = \mathbf{SG}$).

`S (type="S" and type="direct")`
Cross-temporal summing matrix ($\widetilde{\mathbf{F}}$vec$(\widehat{\mathbf{Y}}^{'})$ representation).

`info (type="osqp")`
matrix with some useful indicators (columns) for each forecast horizon h (rows): run time (`run_time`), number of iteration, norm of primal residual (`pri_res`), status of osqp's solution (`status`) and polish's status (`status_polish`).

References

Byron, R.P. (1978), The estimation of large social accounts matrices, *Journal of the Royal Statistical Society A*, 141, 3, 359-367.

Di Fonzo, T., and Girolimetto, D. (2021), Cross-temporal forecast reconciliation: Optimal combination method and heuristic alternatives, *International Journal of Forecasting*, in press.

Schäfer, J.L., Opgen-Rhein, R., Zuber, V., Ahdesmaki, M., Duarte Silva, A.P., Strimmer, K. (2017), *Package 'corpcor'*, R package version 1.6.9 (April 1, 2017), https://CRAN.R-project.org/package=corpcor.

Schäfer, J.L., Strimmer, K. (2005), A Shrinkage Approach to Large-Scale Covariance Matrix Estimation and Implications for Functional Genomics, *Statistical Applications in Genetics and Molecular Biology*, 4, 1.

Stellato, B., Banjac, G., Goulart, P., Bemporad, A., Boyd, S. (2020). OSQP: An Operator Splitting Solver for Quadratic Programs, *Mathematical Programming Computation*, 12, 4, 637-672.

Stellato, B., Banjac, G., Goulart, P., Boyd, S., Anderson, E. (2019), OSQP: Quadratic Programming Solver using the 'OSQP' Library, R package version 0.6.0.3 (October 10, 2019), https://CRAN.R-project.org/package=osqp.

See Also

Other reconciliation procedures: `cstrec()`, `ctbu()`, `htsrec()`, `iterec()`, `lccrec()`, `tcsrec()`, `tdrec()`, `thfrec()`

Examples

```
data(FoReco_data)
obj <- octrec(FoReco_data$base, m = 12, C = FoReco_data$C,
              comb = "bdshr", res = FoReco_data$res)
```

`oct_bounds` *Optimal cross-temporal bounds*

Description

Function to export the constraints designed for the cross-sectional and/or temporal reconciled forecasts

Usage

```
oct_bounds(hts_bounds, thf_bounds, m, C, Ut)
```

Arguments

`hts_bounds`	$(n \times 2)$ matrix with cross-sectional bounds: the first column is the lower bound, and the second column is the upper bound.
`thf_bounds`	$((k^* + m) \times 2)$ matrix with temporal bounds: the first column is the lower bound, and the second column is the upper bound.
`m`	Highest available sampling frequency per seasonal cycle (max. order of temporal aggregation, m), or a subset of p factors of m.
`C`	$(n_a \times n_b)$ cross-sectional (contemporaneous) matrix mapping the bottom level series into the higher level ones.
`Ut`	Zero constraints cross-sectional (contemporaneous) kernel matrix $(\mathbf{U}'\mathbf{y} = \mathbf{0})$ spanning the null space valid for the reconciled forecasts. It can be used instead of parameter `C`, but `nb` $(n = n_a + n_b)$ is needed if $\mathbf{U}' \neq [\mathbf{I} \; -\mathbf{C}]$. If the hierarchy admits a structural representation, $\mathbf{U}'$ has dimension $(n_a \times n)$.

Value

A matrix with the cross-temporal bounds.

See Also

Other utilities: `Cmatrix()`, `FoReco2ts()`, `commat()`, `ctf_tools()`, `hts_tools()`, `lcmat()`, `score_index()`, `shrink_estim()`, `thf_tools()`

Examples

```
data(FoReco_data)
# monthly base forecasts
mbase <- t(FoReco_data$base[, which(simplify2array(strsplit(
  colnames(FoReco_data$base), split = "_"))[1, ] == "k1")])
#' # monthly residuals
mres <- t(FoReco_data$res[, which(simplify2array(strsplit(
  colnames(FoReco_data$res),split = "_"))[1, ] == "k1")])

# For example, in FoReco_data we want that BA > 78, and C > 50
cs_bound <- matrix(c(rep(-Inf, 5), 78, -Inf, 50, rep(+Inf, 8)), ncol = 2)
## Cross-sectional reconciliation
csobj <- htsrec(mbase, C = FoReco_data$C, comb = "shr", res = mres, bounds = cs_bound)

# Extension of the constraints to the cross-temporal case
ct_bound <- oct_bounds(hts_bounds = cs_bound, m = 12)
## Cross-temporal reconciliation
obj <- octrec(FoReco_data$base, m = 12, C = FoReco_data$C, comb = "bdshr",
              res = FoReco_data$res, bounds = ct_bound)
```

---

score_index — *Measuring accuracy in a rolling forecast experiment*

Description

Function to calculate the accuracy indices of the reconciled point forecasts of a cross-temporal (not only, see examples) system (more in Average relative accuracy indices). (*Experimental version*)

Usage

```
score_index(recf, base, test, m, nb, nl, type = "mse", compact = TRUE)
```

Arguments

`recf`	list of q (forecast origins) reconciled forecasts' matrices ($[n \times h(k^* + m)]$ in the cross-temporal case, $[h \times n]$ in the cross-sectional case, and vectors of length $[h(k^* \times m)]$ in the temporal framework).
`base`	list of q (forecast origins) base forecasts' matrices ($[n \times h(k^* + m)]$ in the cross-temporal case, $[h \times n]$ in the cross-sectional case, and vectors of length $[h(k^* \times m)]$ in the temporal framework).
`test`	list of q (forecast origins) test observations' matrices ($[n \times h(k^* + m)]$ in the cross-temporal case, $[h \times n]$ in the cross-sectional case, and vectors of length $[h(k^* \times m)]$ in the temporal framework).
`m`	Highest available sampling frequency per seasonal cycle (max. order of temporal aggregation, m), or a subset of p factors of m.

nb	number of bottom time series in the cross-sectional framework.
nl	$(L \times 1)$ vector containing the number of time series in each cross-sectional level of the hierarchy (nl[1] = 1).
type	type of accuracy measure ("mse" Mean Square Error, "rmse" Root Mean Square Error or "mae" Mean Absolute Error).
compact	if TRUE returns only the summary matrix.

Value

It returns a summary table called Avg_mat (if compact option is TRUE, *default*), otherwise it returns a list of six tables (more in Average relative accuracy indices).

References

Di Fonzo, T., and Girolimetto, D. (2021), Cross-temporal forecast reconciliation: Optimal combination method and heuristic alternatives, *International Journal of Forecasting*, in press.

See Also

Other utilities: Cmatrix(), FoReco2ts(), commat(), ctf_tools(), hts_tools(), lcmat(), oct_bounds(), shrink_estim(), thf_tools()

Examples

```
data(FoReco_data)

# Cross-temporal framework
oct_recf <- octrec(FoReco_data$base, m = 12, C = FoReco_data$C,
                   comb = "bdshr", res = FoReco_data$res)$recf
oct_score <- score_index(recf = oct_recf,
                         base = FoReco_data$base,
                         test = FoReco_data$test, m = 12, nb = 5)

# Cross-sectional framework
# monthly base forecasts
id <- which(simplify2array(strsplit(colnames(FoReco_data$base), split = "_"))[1, ] == "k1")
mbase <- t(FoReco_data$base[, id])
# monthly test set
mtest <- t(FoReco_data$test[, id])
# monthly residuals
id <- which(simplify2array(strsplit(colnames(FoReco_data$res), split = "_"))[1, ] == "k1")
mres <- t(FoReco_data$res[, id])
# monthly reconciled forecasts
mrecf <- htsrec(mbase, C = FoReco_data$C, comb = "shr", res = mres)$recf
# score
hts_score <- score_index(recf = mrecf, base = mbase, test = mtest, nb = 5)

# Temporal framework
# top ts base forecasts ([lowest_freq' ...  highest_freq']')
topbase <- FoReco_data$base[1, ]
# top ts residuals ([lowest_freq' ...  highest_freq']')
topres <- FoReco_data$res[1, ]
# top ts test ([lowest_freq' ...  highest_freq']')
toptest <- FoReco_data$test[1, ]
# top ts recf ([lowest_freq' ...  highest_freq']')
```

```
toprecf <- thfrec(topbase, m = 12, comb = "acov", res = topres)$recf
# score
thf_score <- score_index(recf = toprecf, base = topbase, test = toptest, m = 12)
```

`shrink_estim` *Shrinkage of the covariance matrix*

Description

Shrinkage of the covariance matrix according to Schäfer and Strimmer (2005).

Usage

```
shrink_estim(x, minT = T)
```

Arguments

`x`	residual matrix
`minT`	this param allows to calculate the covariance matrix according to the original hts formulation (`TRUE`) or according to the standard approach (`FALSE`).

Value

A list with two objects: the first (`$scov`) is the shrunk covariance matrix and the second (`$lambda`) is the shrinkage intensity coefficient.

Author(s)

This function is a modified version of the `shrink_estim()` hidden function of **hts**.

References

Schäfer, J.L., Strimmer, K. (2005), A Shrinkage Approach to Large-Scale Covariance Matrix Estimation and Implications for Functional Genomics, *Statistical Applications in Genetics and Molecular Biology*, 4, 1

Hyndman, R. J., Lee, A., Wang, E., and Wickramasuriya, S. (2020). hts: Hierarchical and Grouped Time Series, *R package version 6.0.1*, https://CRAN.R-project.org/package=hts.

See Also

Other utilities: `Cmatrix()`, `FoReco2ts()`, `commat()`, `ctf_tools()`, `hts_tools()`, `lcmat()`, `oct_bounds()`, `score_index()`, `thf_tools()`

tcsrec	*Heuristic first-temporal-then-cross-sectional cross-temporal forecast reconciliation*

Description

The cross-temporal forecast reconciliation procedure by Kourentzes and Athanasopoulos (2019) can be viewed as an ensemble forecasting procedure which exploits the simple averaging of different forecasts. First, for each time series the forecasts at any temporal aggregation order are reconciled using temporal hierarchies (`thfrec()`), then time-by-time cross-sectional reconciliation is performed (`htsrec()`). The projection matrices obtained at this step are then averaged and used to cross-sectionally reconcile the forecasts obtained at step 1, by this way fulfilling both cross-sectional and temporal constraints.

Usage

```
tcsrec(basef, thf_comb, hts_comb, res, avg = "KA", ...)
```

Arguments

`basef` $(n \times h(k^* + m))$ matrix of base forecasts to be reconciled, $\widehat{\mathbf{Y}}$; n is the total number of variables, m is the highest time frequency, k^* is the sum of (a subset of) $(p-1)$ factors of m, excluding m, and h is the forecast horizon for the lowest frequency time series. Each row identifies a time series, and the forecasts are ordered as [lowest_freq' ... highest_freq']'.

`hts_comb, thf_comb` Type of covariance matrix (respectively $(n \times n)$ and $((k^* + m) \times (k^* + m))$) to be used in the cross-sectional and temporal reconciliation, see more in `comb` param of `htsrec()` and `thfrec()`.

`res` $(n \times N(k^* + m))$ matrix containing the residuals at all the temporal frequencies ordered [lowest_freq' ... highest_freq']' (columns) for each variable (row), needed to estimate the covariance matrix when `hts_comb` = {`"wls"`, `"shr"`, `"sam"`} and/or `hts_comb` = {`"wlsv"`, `"wlsh"`, `"acov"`, `"strar1"`, `"sar1"`, `"har1"`, `"shr"`, `"sam"`}. The row must be in the same order as `basef`.

`avg` If `avg = "KA"` (*default*), the final projection matrix $\mathbf{M}$ is the one proposed by Kourentzes and Athanasopoulos (2019), otherwise it is calculated as simple average of all the involved projection matrices at step 2 of the procedure (see Di Fonzo and Girolimetto, 2020).

`...` any other options useful for `htsrec()` and `thfrec()`, e.g. `m`, `C` (or `Ut` and `nb`), `nn` (for non negativity reconciliation only at first step), `mse`, `corpcor`, `type`, `sol`, `settings`, `W`, `Omega`,...

Details

This function performs a two-step cross-temporal forecast reconciliation using the covariance matrices chosen by the user. If the combinations used by Kourentzes and Athanasopoulos (2019) are wished, `thf_comb` must be set equal to either `"struc"` or `"wlsv"`, and `hts_comb` equal to either `"shr"` or `"wls"`.

Warning, the two-step heuristic reconciliation allows non negativity constraints only in the first step. This means that non-negativity is not guaranteed in the final reconciled values.

Value

The function returns a list with two elements:

recf — $(n \times h(k^* + m))$ reconciled forecasts matrix, $\widetilde{\mathbf{Y}}$.

M — Matrix which transforms the uni-dimensional reconciled forecasts of step 1 (projection approach) .

References

Di Fonzo, T., and Girolimetto, D. (2021), Cross-temporal forecast reconciliation: Optimal combination method and heuristic alternatives, *International Journal of Forecasting*, in press.

Kourentzes, N., Athanasopoulos, G. (2019), Cross-temporal coherent forecasts for Australian tourism, *Annals of Tourism Research*, 75, 393-409.

Schäfer, J.L., Opgen-Rhein, R., Zuber, V., Ahdesmaki, M., Duarte Silva, A.P., Strimmer, K. (2017), *Package 'corpcor'*, R package version 1.6.9 (April 1, 2017), https://CRAN.R-project.org/package=corpcor.

Schäfer, J.L., Strimmer, K. (2005), A Shrinkage Approach to Large-Scale Covariance Matrix Estimation and Implications for Functional Genomics, *Statistical Applications in Genetics and Molecular Biology*, 4, 1.

Stellato, B., Banjac, G., Goulart, P., Bemporad, A., Boyd, S. (2020). OSQP: An Operator Splitting Solver for Quadratic Programs, *Mathematical Programming Computation*, 12, 4, 637-672.

Stellato, B., Banjac, G., Goulart, P., Boyd, S., Anderson, E. (2019), OSQP: Quadratic Programming Solver using the 'OSQP' Library, R package version 0.6.0.3 (October 10, 2019), https://CRAN.R-project.org/package=osqp.

See Also

Other reconciliation procedures: `cstrec()`, `ctbu()`, `htsrec()`, `iterec()`, `lccrec()`, `octrec()`, `tdrec()`, `thfrec()`

Examples

```
data(FoReco_data)
obj <- tcsrec(FoReco_data$base, m = 12, C = FoReco_data$C,
              thf_comb = "acov", hts_comb = "shr", res = FoReco_data$res)
```

---

tdrec — *Top-down forecast reconciliation for genuine hierarchical/grouped time series*

Description

Top-down forecast reconciliation for genuine hierarchical/grouped time series, where the forecast of a 'Total' (top-level series, expected to be positive) is disaggregated according to a proportional scheme given by a vector of proportions (weights). Besides the fulfillment of any aggregation constraint, the top-down reconciled forecasts should respect two main properties: - the top-level value remains unchanged; - all the bottom time series reconciled forecasts are non-negative. The top-down procedure is extended to deal with both temporal and cross-temporal cases. Since this is a post forecasting function, the weight vector must be given in input by the user, and is not calculated automatically (see Examples).

Usage

```
tdrec(topf, C, m, weights)
```

Arguments

- `topf` — $(h \times 1)$ vector of the top-level base forecast to be disaggregated; h is the forecast horizon (for the lowest temporal aggregation order in temporal and cross-temporal cases).
- `C` — $(n_a \times n_b)$ cross-sectional (contemporaneous) matrix mapping the n_b bottom level series into the n_a higher level ones.
- `m` — Highest available sampling frequency per seasonal cycle (max. order of temporal aggregation, m), or a subset of the p factors of m.
- `weights` — vector of weights to be used to disaggregate topf: $(n_b \times h)$ matrix in the cross-sectional framework; $(m \times h)$ matrix in the temporal framework; $(n_b m \times h)$ matrix in the cross-temporal framework.

Details

Fix $h = 1$, then

$$\widetilde{\mathbf{y}} = \mathbf{S}\mathbf{w}\widehat{a}_1$$

where $\widetilde{\mathbf{y}}$ is the vector of reconciled forecasts, $\mathbf{S}$ is the summing matrix (whose pattern depends on which type of reconciliation is being performed), $\mathbf{w}$ is the vector of weights, and $\widehat{a}_1$ is the top-level value to be disaggregated.

Value

The function returns an $(h \times n)$ matrix of cross-sectionally reconciled forecasts, or an $(h(k^* + m) \times 1)$ vector of top-down temporally reconciled forecasts, or an $(n \times h(k^* + m))$ matrix of top-down cross-temporally reconciled forecasts.

References

Athanasopoulos, G., Ahmed, R.A., Hyndman, R.J. (2009), Hierarchical forecasts for Australian domestic tourism, *International Journal of Forecasting*, 25, 1, 146–166.

See Also

Other reconciliation procedures: `cstrec()`, `ctbu()`, `htsrec()`, `iterec()`, `lccrec()`, `octrec()`, `tcsrec()`, `thfrec()`

Examples

```
data(FoReco_data)
### CROSS-SECTIONAL TOP-DOWN RECONCILIATION
# Cross sectional aggregation matrix
C <- FoReco_data$C
# monthly base forecasts
id <- which(simplify2array(strsplit(colnames(FoReco_data$base), split = "_"))[1, ] == "k1")
mbase <- t(FoReco_data$base[, id])
obs_1 <- FoReco_data$obs$k1
# average historical proportions
props <- colMeans(obs_1[1:168,-c(1:3)]/obs_1[1:168,1])
cs_td <- tdrec(topf = mbase[,1], C = C, weights = props)
```

```
### TEMPORAL TOP-DOWN RECONCILIATION
# top ts base forecasts ([lowest_freq' ...  highest_freq']')
top_obs12 <- FoReco_data$obs$k12[1:14,1]
bts_obs1 <- FoReco_data$obs$k1[1:168,1]
# average historical proportions
props <- colMeans(matrix(bts_obs1, ncol = 12, byrow = TRUE)/top_obs12)
topbase <- FoReco_data$base[1, 1]
t_td <- tdrec(topf = topbase, m = 12, weights = props)

### CROSS-TEMPORAL TOP-DOWN RECONCILIATION
top_obs <- FoReco_data$obs$k12[1:14,1]
bts_obs <- FoReco_data$obs$k1[1:168,-c(1:3)]
bts_obs <- lapply(1:5, function(x) matrix(bts_obs[,x], nrow=14, byrow = TRUE))
bts_obs <- do.call(cbind, bts_obs)
# average historical proportions
props <- colMeans(bts_obs/top_obs)
ct_td <- tdrec(topf = topbase, m = 12, C = C, weights = props)
```

---

thfrec — *Forecast reconciliation through temporal hierarchies (temporal reconciliation)*

Description

Forecast reconciliation of one time series through temporal hierarchies (Athanasopoulos et al., 2017). The reconciled forecasts are calculated either through a projection approach (Byron, 1978), or the equivalent structural approach by Hyndman et al. (2011). Moreover, the classic bottom-up approach is available.

Usage

```
thfrec(basef, m, comb, res, mse = TRUE, corpcor = FALSE,
       type = "M", sol = "direct", keep = "list", v = NULL, nn = FALSE,
        nn_type = "osqp", settings = list(), bounds = NULL, Omega = NULL)
```

Arguments

`basef` — $(h(k^* + m) \times 1)$ vector of base forecasts to be reconciled, containing the forecasts at all the needed temporal frequencies ordered as [lowest_freq' ... highest_freq']'.

`m` — Highest available sampling frequency per seasonal cycle (max. order of temporal aggregation, m), or a subset of p factors of m.

`comb` — Type of the reconciliation. Except for bottom up, all other options correspond to a different $((k^* + m) \times (k^* + m))$ covariance matrix, k^* is the sum of $(p-1)$ factors of m (excluding m):

- **bu** (Bottom-up);
- **ols** (Identity);
- **struc** (Structural variances);
- **wlsv** (Series variances);

- **wlsh** (Hierarchy variances);
- **acov** (Auto-covariance matrix);
- **strar1** (Structural Markov);
- **sar1** (Series Markov);
- **har1** (Hierarchy Markov);
- **shr** (Shrunk cross-covariance matrix);
- **sam** (Sample cross-covariance matrix);
- **omega** use your personal matrix Omega in param `Omega`.

`res` vector containing the in-sample residuals at all the temporal frequencies ordered as `basef`, i.e. [lowest_freq' ... highest_freq']', needed to estimate the covariance matrix when `comb = {"wlsv", "wlsh", "acov", "strar1", "sar1", "har1", "shr", "sam"}`.

`mse` Logical value: `TRUE` (*default*) calculates the covariance matrix of the in-sample residuals (when necessary) according to the original **hts** and **thief** formulation: no mean correction, T as denominator.

`corpcor` Logical value: `TRUE` if **corpcor** (Schäfer et al., 2017) must be used to shrink the sample covariance matrix according to Schäfer and Strimmer (2005), otherwise the function uses the same implementation as package **hts**.

`type` Approach used to compute the reconciled forecasts: `"M"` for the projection approach with matrix M (*default*), or `"S"` for the structural approach with temporal summing matrix R.

`sol` Solution technique for the reconciliation problem: either `"direct"` (*default*) for the closed-form matrix solution, or `"osqp"` for the numerical solution (solving a linearly constrained quadratic program using `solve_osqp`).

`keep` Return a list object of the reconciled forecasts at all levels (if keep = "list") or only the reconciled forecasts matrix (if keep = "recf").

`v` vector index of the fixed base forecast ($\min(v) > 0$ and $\max(v) < (k^* + m)$).

`nn` Logical value: `TRUE` if non-negative reconciled forecasts are wished.

`nn_type` "osqp" (default), "KAnn" (only `type == "M"`) or "sntz".

`settings` Settings for **osqp** (object `osqpSettings`). The default options are: `verbose = FALSE`, `eps_abs = 1e-5`, `eps_rel = 1e-5`, `polish_refine_iter = 100` and `polish = TRUE`. For details, see the **osqp** documentation (Stellato et al., 2019).

`bounds` $((k^*+m)\times 2)$ matrix with temporal bounds: the first column is the lower bound, and the second column is the upper bound.

`Omega` This option permits to directly enter the covariance matrix:

1. `Omega` must be a p.d. $((k^* + m) \times (k^* + m))$ matrix or a list of h matrix (one for each forecast horizon);
2. if `comb` is different from "`omega`", `Omega` is not used.

Details

Let m be the highest available sampling frequency per seasonal cycle, and denote $\mathcal{K} = \{k_m, k_{p-1}, \ldots, k_2, k_1\}$ the p factors of m, in descending order, where $k_p = m$, and $k_1 = 1$. Define $\mathbf{K}$ the $(k^* \times m)$ temporal aggregation matrix converting the high-frequency observations into lower-frequency (temporally aggregated) ones:

$$\mathbf{K} = \begin{bmatrix} \mathbf{1}'_m \\ \mathbf{I}_{\frac{m}{k_{p-1}}} \otimes \mathbf{1}'_{k_{p-1}} \\ \vdots \\ \mathbf{I}_{\frac{m}{k_2}} \otimes \mathbf{1}'_{k_2} \end{bmatrix}.$$

Denote $\mathbf{R} = \begin{bmatrix} \mathbf{K} \\ \mathbf{I}_m \end{bmatrix}$ the $[(k^* + m) \times m]$ *temporal summing* matrix, and $\mathbf{Z}' = [\mathbf{I}_{k^*} \;\; -\mathbf{K}]$ the zero constraints kernel matrix.

Suppose we have the $[(k^* + m) \times 1]$ vector $\widehat{\mathbf{y}}$ of unbiased base forecasts for the p temporal aggregates of a single time series Y within a complete time cycle, i.e. at the forecast horizon $h = 1$ for the lowest (most aggregated) time frequency. If the base forecasts have been independently obtained, generally they do not fulfill the temporal aggregation constraints, i.e. $\mathbf{Z}'\widehat{\mathbf{y}} \neq \mathbf{0}_{(k^* \times 1)}$. By adapting the general point forecast reconciliation according to the projection approach (`type = "M"`), the vector of temporally reconciled forecasts is given by:

$$\widetilde{\mathbf{y}} = \widehat{\mathbf{y}} - \mathbf{\Omega Z} \left(\mathbf{Z}'\mathbf{\Omega Z}\right)^{-1} \mathbf{Z}'\widehat{\mathbf{y}},$$

where $\mathbf{\Omega}$ is a $[(k^* + m) \times (k^* + m)]$ p.d. matrix, assumed known. The alternative equivalent solution (`type = "S"`) following the structural reconciliation approach by Athanasopoulos et al. (2017) is given by:

$$\widetilde{\mathbf{y}} = \mathbf{R} \left(\mathbf{R}'\mathbf{\Omega}^{-1}\mathbf{R}\right)^{-1} \mathbf{R}'\mathbf{\Omega}^{-1}\widehat{\mathbf{y}}.$$

Bounds on the reconciled forecasts

When the reconciliation makes use of the optimization package osqp, the user may impose bounds on the reconciled forecasts. The parameter `bounds` permits to consider lower ($\mathbf{a}$) and upper ($\mathbf{b}$) bounds like $\mathbf{a} \leq \widetilde{\mathbf{y}} \leq \mathbf{b}$ such that:

$$\begin{array}{c} a_1 \leq \widetilde{y}_1 \leq b_1 \\ \dots \\ a_{(k^*+m)} \leq \widetilde{y}_{(k^*+m)} \leq b_{(k^*+m)} \end{array} \Rightarrow \text{bounds} = [\mathbf{a} \; \mathbf{b}] = \begin{bmatrix} a_1 & b_1 \\ \vdots & \vdots \\ a_{(k^*+m)} & b_{(k^*+m)} \end{bmatrix},$$

where $a_i \in [-\infty, +\infty]$ and $b_i \in [-\infty, +\infty]$. If y_i is unbounded, the i-th row of `bounds` would be equal to `c(-Inf, +Inf)`. Notice that if the `bounds` parameter is used, `sol = "osqp"` must be used. This is not true in the case of non-negativity constraints:

- `sol = "direct"`: first the base forecasts are reconciled without non-negativity constraints, then, if negative reconciled values are present, the `"osqp"` solver is used;
- `sol = "osqp"`: the base forecasts are reconciled using the `"osqp"` solver.

In this case it is not necessary to build a matrix containing the bounds, and it is sufficient to set `nn = "TRUE"`.

Non-negative reconciled forecasts may be obtained by setting `nn_type` alternatively as:

- `nn_type = "KAnn"` (Kourentzes and Athanasopoulos, 2021)
- `nn_type = "sntz"` ("set-negative-to-zero")
- `nn_type = "osqp"` (Stellato et al., 2020)

Value

If the parameter `keep` is equal to `"recf"`, then the function returns only the $(h(k^* + m) \times 1)$ reconciled forecasts vector, otherwise (`keep="all"`) it returns a list that mainly depends on what type of representation (`type`) and solution technique (`sol`) have been used:

`recf`	$(h(k^* + m) \times 1)$ reconciled forecasts vector, $\widetilde{\mathbf{y}}$.
`Omega`	Covariance matrix used for forecast reconciliation, $\mathbf{\Omega}$.
`nn_check`	Number of negative values (if zero, there are no values below zero).
`rec_check`	Logical value: has the hierarchy been respected?

`varf (type="direct")`
: $((k^* + m) \times 1)$ reconciled forecasts variance vector for $h = 1$, diag($\mathbf{MW}$).

`M (type="direct")`
: Projection matrix, $\mathbf{M}$ (projection approach).

`G (type="S" and type="direct")`
: Projection matrix, $\mathbf{G}$ (structural approach, $\mathbf{M} = \mathbf{RG}$).

`S (type="S" and type="direct")`
: Temporal summing matrix, $\mathbf{R}$.

`info (type="osqp")`
: matrix with information in columns for each forecast horizon h (rows): run time (`run_time`), number of iteration (`iter`), norm of primal residual (`pri_res`), status of osqp's solution (`status`) and polish's status (`status_polish`). It will also be returned with `nn = TRUE` if a solver (see `nn_type`) will be use.

Only if `comb = "bu"`, the function returns `recf`, `R` and `M`.

References

Athanasopoulos, G., Hyndman, R.J., Kourentzes, N., Petropoulos, F. (2017), Forecasting with Temporal Hierarchies, *European Journal of Operational Research*, 262, 1, 60-74.

Byron, R.P. (1978), The estimation of large social accounts matrices, *Journal of the Royal Statistical Society A*, 141, 3, 359-367.

Di Fonzo, T., and Girolimetto, D. (2021), Cross-temporal forecast reconciliation: Optimal combination method and heuristic alternatives, *International Journal of Forecasting*, in press.

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

Nystrup, P., Lindström, E., Pinson, P., Madsen, H. (2020), Temporal hierarchies with autocorrelation for load forecasting, *European Journal of Operational Research*, 280, 1, 876-888.

Schäfer, J.L., Opgen-Rhein, R., Zuber, V., Ahdesmaki, M., Duarte Silva, A.P., Strimmer, K. (2017), Package 'corpcor', R package version 1.6.9 (April 1, 2017), https://CRAN.R-project.org/package=corpcor.

Schäfer, J.L., Strimmer, K. (2005), A Shrinkage Approach to Large-Scale Covariance Matrix Estimation and Implications for Functional Genomics, *Statistical Applications in Genetics and Molecular Biology*, 4, 1.

Stellato, B., Banjac, G., Goulart, P., Bemporad, A., Boyd, S. (2020). OSQP: An Operator Splitting Solver for Quadratic Programs, *Mathematical Programming Computation*, 12, 4, 637-672.

Stellato, B., Banjac, G., Goulart, P., Boyd, S., Anderson, E. (2019), OSQP: Quadratic Programming Solver using the 'OSQP' Library, R package version 0.6.0.3 (October 10, 2019), https://CRAN.R-project.org/package=osqp.

See Also

Other reconciliation procedures: `cstrec()`, `ctbu()`, `htsrec()`, `iterec()`, `lccrec()`, `octrec()`, `tcsrec()`, `tdrec()`

Examples

```
data(FoReco_data)
# top ts base forecasts ([lowest_freq' ...  highest_freq']')
topbase <- FoReco_data$base[1, ]
 # top ts residuals ([lowest_freq' ...  highest_freq']')
```

```
topres <- FoReco_data$res[1, ]
obj <- thfrec(topbase, m = 12, comb = "acov", res = topres)
```

---

thf_tools *Temporal reconciliation tools*

Description

Some useful tools for forecast reconciliation through temporal hierarchies.

Usage

```
thf_tools(m, h = 1, sparse = TRUE)
```

Arguments

m	Highest available sampling frequency per seasonal cycle (max. order of temporal aggregation, m), or a subset of the p factors of m.
h	Forecast horizon for the lowest frequency (most temporally aggregated) time series (*default* is 1).
sparse	Option to return sparse object (*default* is TRUE).

Value

A list of seven elements:

K	Temporal aggregation matrix.
R	Temporal summing matrix.
Zt	Zero constraints temporal kernel matrix, $\mathbf{Z}_h'\mathbf{Y}' = \mathbf{0}_{[hk^* \times n]}$.
kset	Set of factors (p) of m in descending order (from m to 1), $\mathcal{K} = \{k_p, k_{p-1}, \dots, k_2, k_1\}$, $k_p = m$, $k_1 = 1$.
m	Highest available sampling frequency per seasonal cycle (max. order of temporal aggregation).
p	Number of elements of kset, $\mathcal{K}$.
ks	Sum of $p - 1$ factors of m (out of m itself), k^*.
kt	Sum of all factors of m, $k^{tot} = k^* + m$.

See Also

Other utilities: Cmatrix(), FoReco2ts(), commat(), ctf_tools(), hts_tools(), lcmat(), oct_bounds(), score_index(), shrink_estim()

Examples

```
# quarterly data
obj <- thf_tools(m = 4, sparse = FALSE)
```

Index