Subset Selection for Forecast Reconciliation

Xiaoqian Wang

2023-01-17

1 Hierarchical time series

Create a hierarchical structure as complete as possible. Only nested structure is considered here. Then the summing matrix $\boldsymbol{S}$ is given and $\hat{\boldsymbol{y}}_h$ can be obtained easily by specifying forecasting models.

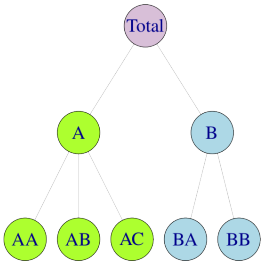

Figure 1: A 2-level hierarchical tree structure

For example,

$$\boldsymbol{S} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \text{ and } \hat{\boldsymbol{y}}_h = (10, 6, 5, 1, 4, 0, 2, 5)'.$$

2 Linear forecast reconciliation

$$\tilde{\boldsymbol{y}}_h = \boldsymbol{S}\boldsymbol{G}_h\hat{\boldsymbol{y}}_h,$$

where $\boldsymbol{G}_h$ maps the base forecasts $\hat{\boldsymbol{y}}_h$ into the bottom level, it combines all base forecasts to form bottom-level forecasts.

- **Bottom-up approach:** $\boldsymbol{G}_{bu} = \begin{bmatrix} \boldsymbol{O}_{n_b \times n_a} & \boldsymbol{I}_{n_b} \end{bmatrix}$
- **Top-down approach:** $\boldsymbol{G}_{td} = \begin{bmatrix} \boldsymbol{p} & \boldsymbol{O}_{n_b \times (n-1)} \end{bmatrix}$, where $\boldsymbol{p}$ is an n_b-dimensional vector including

the set of disaggregation proportions.

- **Middle-out approach:** $\boldsymbol{G}_{mo} = \begin{bmatrix} \boldsymbol{O}_{n_b \times n_t} & \boldsymbol{P}_{n_b \times n_l} & \boldsymbol{O}_{n_b \times n_d} \end{bmatrix}$, where $n_t + n_l + n_d = n$, $n_d \geq n_b$, $n_t \geq 1$, $\sum_{i=1}^{n_b} \sum_{j=1}^{n_l} p_{ij} = n_l$.
- **Optimization approaches:** $\boldsymbol{G}_h = \left(\boldsymbol{S}'\boldsymbol{W}_h^{-1}\boldsymbol{S}\right)^{-1}\boldsymbol{S}'\boldsymbol{W}_h^{-1}$ is obtained by solving the optimization problem

$$\min_{\boldsymbol{G}_h} \left(\hat{\boldsymbol{y}}_h - \boldsymbol{S}\boldsymbol{G}_h\hat{\boldsymbol{y}}_h\right)' \boldsymbol{W}_h^{-1} \left(\hat{\boldsymbol{y}}_h - \boldsymbol{S}\boldsymbol{G}_h\hat{\boldsymbol{y}}_h\right), \text{ s.t. } \boldsymbol{G}_h\boldsymbol{S} = \boldsymbol{I}_{n_b}$$

3 Subset selection

a. About $\boldsymbol{G}_h$. Our goal is to **zero out some columns** so that the corresponding base forecasts in $\hat{\boldsymbol{y}}_h$ are not used to form the reconciled bottom-level forecasts and, furthermore, all reconciled forecasts.

b. About $\boldsymbol{S}$. It sums up the reconciled bottom-level forecasts to get the full set of reconciled forecasts. For the purpose of automatic selection and comparison, we don't need to zero out the corresponding rows.

We have to add additional constraints on $\boldsymbol{G}_h$. This leads to a main question:

How to mathematically describe the constraint?

3.1 Ideas based on MinT

3.1.1 Solution 1

$$\tilde{\boldsymbol{y}}_h = \boldsymbol{S}\boldsymbol{G}_h\boldsymbol{B}_h\hat{\boldsymbol{y}}_h,$$

where $\boldsymbol{B}_h$ is an $n \times n$ diagonal matrix with elements of the main diagonal being either zero or one.

The optimization problem can be written as

$$\begin{aligned} \min_{\boldsymbol{G}_h} \left(\hat{\boldsymbol{y}}_h - \boldsymbol{S}\boldsymbol{G}_h\boldsymbol{B}_h\hat{\boldsymbol{y}}_h\right)' \boldsymbol{W}_h^{-1} \left(\hat{\boldsymbol{y}}_h - \boldsymbol{S}\boldsymbol{G}_h\boldsymbol{B}_h\hat{\boldsymbol{y}}_h\right) + \lambda||\boldsymbol{B}_h\mathbf{1}||_1, \\ \text{s.t. } \boldsymbol{G}_h\boldsymbol{B}_h\boldsymbol{S} = \boldsymbol{I}_{n_b} \\ b_{ii}(1 - b_{ii}) = 0 \text{ for } i = 1, 2, \ldots, n \\ b_{ij} = 0 \text{ for } i \neq j \end{aligned}$$

The difficulty is that we wish to find the value of $\boldsymbol{G}_h$ and $\boldsymbol{B}_h$ simultaneously.

3.1.2 Solution 2

$$\tilde{\boldsymbol{y}}_h = \boldsymbol{S}\check{\boldsymbol{G}}_h\check{\boldsymbol{B}}_h\hat{\boldsymbol{y}}_h,$$

where $\check{\boldsymbol{G}}_h$ is given using ols, wls, structural scaling, or shrinkage estimator of $\boldsymbol{W}_h$. $\check{\boldsymbol{B}}_h$ is an $n \times n$ diagonal matrix with elements of the main diagonal being either greater than or equal to zero. So $\check{\boldsymbol{B}}_h$ can adaptively changes given $\check{\boldsymbol{G}}_h$ and we can only estimate $\check{\boldsymbol{B}}_h$. But the reconciled forecasts are likely to no longer preserve the unbiasedness.

The optimization problem can be written as

$$\begin{aligned} \min_{\boldsymbol{G}_h} \left(\hat{\boldsymbol{y}}_h - \boldsymbol{S}\check{\boldsymbol{G}}_h\check{\boldsymbol{B}}_h\hat{\boldsymbol{y}}_h\right)' \boldsymbol{W}_h^{-1} \left(\hat{\boldsymbol{y}}_h - \boldsymbol{S}\check{\boldsymbol{G}}_h\check{\boldsymbol{B}}_h\hat{\boldsymbol{y}}_h\right) + \lambda||\check{\boldsymbol{B}}_h\mathbf{1}||_1, \\ \text{s.t. } b_{ii} \geq 0 \text{ for } i = 1, \ldots, n \\ b_{ij} = 0 \text{ for } i \neq j \end{aligned}$$

R implementation

```
library(CVXR)


y_hat <- c(10, 6, 5, 1, 4, 0, 2, 5)
S <- rbind(matrix(c(rep(1, 5), c(rep(1, 3), rep(0, 2)), c(rep(0, 3), rep(1, 2))),
                  nrow = 3, byrow = TRUE),
           diag(1, nrow = 5))

W_inv <- solve(diag(c(5,3,2,1,1,1,1,1))) # structural scaling approximation
G <- solve(t(S) %*% W_inv %*% S) %*% t(S) %*% W_inv

b <- Variable(8)
lambda <- 0.3

e <- y_hat - S %*% G  %*% diag(b) %*% y_hat
obj <- quad_form(e, W_inv) + lambda * norm1(b)
prob <- Problem(Minimize(obj), constraints = list(b >= 0))
res <- solve(prob)
res$getValue(b)
```

```
##               [,1]
## [1,]  2.076187e+00
## [2,]  6.941964e-05
## [3,]  9.090869e-01
## [4,]  6.774912e-01
## [5,]  9.474978e-01
```

```
## [6,]  4.014566e-12
## [7,] -4.811095e-05
## [8,]  5.999817e-01
```

```
(B_check <- diag(round(as.vector(res$getValue(b)), digits = 3)))
```

```
##       [,1] [,2]  [,3]  [,4]  [,5] [,6] [,7] [,8]
## [1,] 2.076    0 0.000 0.000 0.000    0    0  0.0
## [2,] 0.000    0 0.000 0.000 0.000    0    0  0.0
## [3,] 0.000    0 0.909 0.000 0.000    0    0  0.0
## [4,] 0.000    0 0.000 0.677 0.000    0    0  0.0
## [5,] 0.000    0 0.000 0.000 0.947    0    0  0.0
## [6,] 0.000    0 0.000 0.000 0.000    0    0  0.0
## [7,] 0.000    0 0.000 0.000 0.000    0    0  0.0
## [8,] 0.000    0 0.000 0.000 0.000    0    0  0.6
```

```
(y_tilde <- as.vector(S %*% G %*% B_check %*% y_hat))
```

```
## [1] 10.923333  5.183500  5.739833  0.916500  4.027500  0.239500  1.369917
## [8]  4.369917
```

3.1.3 Solution 3

$$\tilde{\boldsymbol{y}}_h = \boldsymbol{S}\boldsymbol{G}_h\hat{\boldsymbol{y}}_h$$

According to Miyashiro & Takano (2015), we can use mixed logical programming and rewrite the problem as

$$\begin{aligned} \min_{\boldsymbol{G}_h} \left(\hat{\boldsymbol{y}}_h - \boldsymbol{S}\boldsymbol{G}_h\hat{\boldsymbol{y}}_h\right)' \boldsymbol{W}_h^{-1} \left(\hat{\boldsymbol{y}}_h - \boldsymbol{S}\boldsymbol{G}_h\hat{\boldsymbol{y}}_h\right) + \lambda \textstyle\sum_{j=1}^{n} b_j, \\ \text{s.t. } \boldsymbol{G}_h\boldsymbol{S} = \boldsymbol{I}_{n_b} \\ b_j = 0 \Rightarrow ||\boldsymbol{G}_{.j}||_1 = 0 \quad (j = 1, 2, \ldots, n) \\ b_j \in \{0, 1\} \quad (j = 1, 2, \ldots, n) \end{aligned}$$

3.2 Minimization problem

As already shown in Ben Taieb & Koo (2019),

$$\begin{aligned} &\mathrm{E}\left[\left\|\boldsymbol{y}_{T+h} - \tilde{\boldsymbol{y}}_h\right\|_2^2 \mid \boldsymbol{I}_T\right] \\ &= \left\|\boldsymbol{S}\boldsymbol{G}\left(\mathrm{E}\left[\hat{\boldsymbol{y}}_h \mid \boldsymbol{I}_T\right] - \mathrm{E}\left[\boldsymbol{y}_{T+h} \mid \boldsymbol{I}_T\right]\right) + (\boldsymbol{S} - \boldsymbol{S}\boldsymbol{G}\boldsymbol{S})\mathrm{E}\left[\boldsymbol{b}_{T+h} \mid \boldsymbol{I}_T\right]\right\|_2^2 \\ &+ \mathrm{Tr}\left(\mathrm{Var}\left[\boldsymbol{y}_{T+h} - \tilde{\boldsymbol{y}}_h \mid \boldsymbol{I}_T\right]\right). \end{aligned}$$

Under **the unbiasedness conditions (both for base and reconciled forecasts)**, minimizing the

above loss reduces to the following problem:

$$\min_{\boldsymbol{G}_h \in \mathcal{G}} \operatorname{Tr}\left(\operatorname{Var}\left[\boldsymbol{y}_{T+h} - \tilde{\boldsymbol{y}}_h \mid \boldsymbol{I}_T\right]\right) \text{ s.t. } \boldsymbol{S}\boldsymbol{G}_{\boldsymbol{h}}\boldsymbol{S} = \boldsymbol{S}.$$

The trace minimization problem can be reformulated in terms of a linear equality constrained least squares problem as follows:

$$\min_{\boldsymbol{G}_h} \left(\hat{\boldsymbol{y}}_h - \boldsymbol{S}\boldsymbol{G}_h\hat{\boldsymbol{y}}_h\right)' \boldsymbol{W}_h^{-1} \left(\hat{\boldsymbol{y}}_h - \boldsymbol{S}\boldsymbol{G}_h\hat{\boldsymbol{y}}_h\right), \text{ s.t. } \boldsymbol{G}_h\boldsymbol{S} = \boldsymbol{I}_{n_b}$$

- The MinT optimization problem must impose **two unbiasedness conditions**.
- Such an assumption does not always hold in practice, especially when we aim to zero out some columns of $\boldsymbol{G}$.
- The out-of-sample forecasting accuracy of base forecasts is an important basis for determining which columns are zeroed out.

3.3 Relaxation of the unbiasedness assumptions

Ben Taieb & Koo (2019) consider the empirical risk minimization (ERM) problem:

$$\begin{aligned}\hat{G}_{\text{ERMreg}} =& \underset{\boldsymbol{G}\in\mathcal{G}}{\operatorname{argmin}} \left\{ \left\| \boldsymbol{Y} - \hat{\boldsymbol{Y}}\boldsymbol{G}'\boldsymbol{S}' \right\|_F^2 / Nn + \lambda \| \operatorname{vec}(\boldsymbol{G}) \|_1 \right\}, \\ =& \underset{\boldsymbol{G}\in\mathcal{G}}{\operatorname{argmin}} \left\{ \left\| \operatorname{vec}(\boldsymbol{Y}) - (\boldsymbol{S} \otimes \hat{\boldsymbol{Y}}) \operatorname{vec}\left(\boldsymbol{G}'\right) \right\|_2^2 + \lambda \| \operatorname{vec}(\boldsymbol{G}) \|_1 \right\}\end{aligned}$$

The problem can be reduced to **a standard LASSO problem** with $\operatorname{vec}(\boldsymbol{Y})$ as dependent variable and $(\boldsymbol{S} \otimes \hat{\boldsymbol{Y}})$ as design matrix. it is a LASSO problem with $N \times n$ observations and $m \times N$ variables.

3.4 Ideas based on the assumption relaxation

For our project...

We aim to deal with the minimization problem

$$\begin{aligned}\hat{G} =& \underset{\boldsymbol{G}\in\mathcal{G}}{\operatorname{argmin}} \left\{ \left\| \boldsymbol{Y} - \hat{\boldsymbol{Y}}\boldsymbol{G}'\boldsymbol{S}' \right\|_F^2 / Nn + \lambda \sum_{j=1}^{n} z_j \right\}, \\ =& \underset{\boldsymbol{G}\in\mathcal{G}}{\operatorname{argmin}} \left\{ \left\| \operatorname{vec}(\boldsymbol{Y}) - (\boldsymbol{S} \otimes \hat{\boldsymbol{Y}}) \operatorname{vec}\left(\boldsymbol{G}'\right) \right\|_2^2 + \lambda \sum_{j=1}^{n} z_j \right\} \\ s.t. -& Mz_j \leq \sum_{i=0}^{m-1} |g_{j+in}| \leq Mz_j \text{ (Or quadratic form)} \\ & z_j \in \{0,1\} \quad \text{for } j = 1, \ldots, n\end{aligned}$$

- Mixed integer programming (Do not have closed-form solution).
- The results are greatly influenced by the values of M and λ.
- If we assume all elements of $\boldsymbol{G}$ are non-negative, it can be solved by CVXR package even if this is a non-convex problem.

```
library(CVXR)


set.seed(123)
S <- rbind(matrix(c(rep(1, 5), c(rep(1, 3), rep(0, 2)), c(rep(0, 3), rep(1, 2))),
                  nrow = 3, byrow = TRUE),
           diag(1, nrow = 5))
B <- matrix(c(rnorm(10, 1, 1),
              rnorm(10, 4, 1),
              rnorm(10, 0, 1),
              rnorm(10, 2, 1),
              rnorm(10, 5, 1)),
            byrow = FALSE, nrow = 10)
Y <- B %*% t(S)
Y_hat <- matrix(c(rnorm(10, 10, 1),
                  rnorm(10, 6, 1),
                  rnorm(10, 5, 1),
                  rnorm(10, 1, 1),
                  rnorm(10, 4, 1),
                  rnorm(10, 0, 1),
                  rnorm(10, 2, 1),
                  rnorm(10, 5, 1)),
                byrow = FALSE, nrow = 10)


VecY <- as.vector(Y)
D <- kronecker(S, Y_hat)


b <- Variable(8, boolean = TRUE)
#G_trs <- Variable(8, 5)
GB_trs <- Variable(40)


lambda <- 10
```

```r
M <- 100


loss <- sum_squares(VecY - D %*% GB_trs)
penalty <- lambda * norm1(b)
constraints <- list(
  GB_trs >= 0,
  sum_entries(GB_trs[(0:4)*8 + 1]) >= - M*b[1],
  sum_entries(GB_trs[(0:4)*8 + 2]) >= - M*b[2],
  sum_entries(GB_trs[(0:4)*8 + 3]) >= - M*b[3],
  sum_entries(GB_trs[(0:4)*8 + 4]) >= - M*b[4],
  sum_entries(GB_trs[(0:4)*8 + 5]) >= - M*b[5],
  sum_entries(GB_trs[(0:4)*8 + 6]) >= - M*b[6],
  sum_entries(GB_trs[(0:4)*8 + 7]) >= - M*b[7],
  sum_entries(GB_trs[(0:4)*8 + 8]) >= - M*b[8],
  sum_entries(GB_trs[(0:4)*8 + 1]) <=  M*b[1],
  sum_entries(GB_trs[(0:4)*8 + 2]) <=  M*b[2],
  sum_entries(GB_trs[(0:4)*8 + 3]) <=  M*b[3],
  sum_entries(GB_trs[(0:4)*8 + 4]) <=  M*b[4],
  sum_entries(GB_trs[(0:4)*8 + 5]) <=  M*b[5],
  sum_entries(GB_trs[(0:4)*8 + 6]) <=  M*b[6],
  sum_entries(GB_trs[(0:4)*8 + 7]) <=  M*b[7],
  sum_entries(GB_trs[(0:4)*8 + 8]) <=  M*b[8]
)


prob <- Problem(Minimize(loss + penalty), constraints)
res <- solve(prob)
(res$getValue(b))
```

```
##              [,1]
## [1,] 5.054518e-08
## [2,] 1.237176e-05
## [3,] 1.000000e+00
## [4,] 4.694248e-08
## [5,] 4.614314e-08
## [6,] 1.000000e+00
## [7,] 1.000000e+00
```

```
## [8,] 1.000000e+00
```

```
t(matrix(round(res$getValue(GB_trs), digits = 5), 8, 5, byrow = FALSE))
```

```
##      [,1] [,2]    [,3] [,4] [,5]    [,6]    [,7]    [,8]
## [1,]    0    0 0.21959    0    0 0.26458 0.06600 0.00000
## [2,]    0    0 0.65240    0    0 0.37878 0.72390 0.00000
## [3,]    0    0 0.00000    0    0 0.41418 0.00000 0.00000
## [4,]    0    0 0.26782    0    0 0.00001 0.21685 0.14504
## [5,]    0    0 0.14379    0    0 0.10801 0.52335 0.70022
```

Benchmark

$$
\begin{aligned}
\hat{G} =& \underset{\boldsymbol{G}\in\mathcal{G}}{\operatorname{argmin}} \left\{ \left\| \boldsymbol{Y} - \hat{\boldsymbol{Y}}\boldsymbol{B}'\boldsymbol{G}'\boldsymbol{S}' \right\|_F^2 / Nn + \lambda \|B\|_1 \right\}, \\
=& \underset{\boldsymbol{G}\in\mathcal{G}}{\operatorname{argmin}} \left\{ \left\| \operatorname{vec}(\boldsymbol{Y}) - (\boldsymbol{S} \otimes \hat{\boldsymbol{Y}}) \operatorname{vec}\left(\boldsymbol{B}'\boldsymbol{G}'\right) \right\|_2^2 + \lambda \|B\|_1 \right\} \\
=& \underset{\boldsymbol{G}\in\mathcal{G}}{\operatorname{argmin}} \left\{ \left\| \operatorname{vec}(\boldsymbol{Y}) - (\boldsymbol{S} \otimes \hat{\boldsymbol{Y}})(\boldsymbol{G} \otimes \boldsymbol{I}_n) \operatorname{vec}\left(\boldsymbol{B}'\right) \right\|_2^2 + \lambda \|B\|_1 \right\} \\
=& \underset{\boldsymbol{G}\in\mathcal{G}}{\operatorname{argmin}} \left\{ \left\| \operatorname{vec}(\boldsymbol{Y}) - (\boldsymbol{S} \otimes \hat{\boldsymbol{Y}})(\boldsymbol{I}_n \otimes \boldsymbol{B}') \operatorname{vec}\left(\boldsymbol{G}'\right) \right\|_2^2 + \lambda \|B\|_1 \right\} \\
s.t. \quad & b_{ii} \in \{0, 1\} \quad \text{for } i = 1, \ldots, n \\
& b_{ij} = 0 \quad \text{for } i \neq j
\end{aligned}
$$

* Iterative process.

3.5 Temporal reconciliation

3.6 Non-negative constraints