

Department of Econometrics and Business Statistics

<http://monash.edu/business/ebs/research/publications>

Optimal non-negative forecast reconciliation

Shanika L Wickramasuriya

Berwin A Turlach

Rob J Hyndman

March 2019

Working Paper ??/??

Optimal non-negative forecast reconciliation

Shanika L Wickramasuriya

Department of Statistics,
University of Auckland,
Auckland, New Zealand.
Email: s.wickramasuriya@auckland.ac.nz

Berwin A Turlach

Centre for Applied Statistics,
The University of Western Australia,
Crawley, Australia.
Email: berwin.turlach@uwa.edu.au

Rob J Hyndman

Department of Econometrics and Business Statistics,
Monash University,
Melbourne, Australia.
Email: rob.hyndman@monash.edu

21 March 2019

Abstract

A problem with MinT and its variants proposed by Wickramasuriya, Athanasopoulos & Hyndman (2019) is that the coherent forecasts may result in negative values, even when all of the base forecasts are non-negative. This has become a serious issue in applications that are inherently non-negative such as with sales data. While overcoming this difficulty, we considered the analytical solution of MinT as a least squares minimization problem. The non-negativity constraints were then imposed on the minimization problem to ensure that the reconciled forecasts are strictly non-negative.

Considering the dimension and sparsity of the matrices involved, and the alternative representation of MinT, this constrained quadratic programming problem was solved using three algorithms. They are the block principal pivoting algorithm, projected conjugate gradient algorithm, and scaled gradient projection algorithm. It was observed that when obtaining the coherent forecasts using the OLS approach, the scaled gradient projection algorithm simplifies to the standard gradient projection algorithm. Hence, use of this algorithm is problematic for very large collections of time series.

A Monte Carlo simulation was performed to evaluate the computational performances of these algorithms. The results demonstrated that the block principal pivoting algorithm clearly outperforms the rest, and projected conjugate gradient is the second best. The superior performance of the block principal pivoting algorithm can be partially attributed to the alternative representation of the weight matrix in the MinT approach.

An empirical investigation was carried out to assess the impact of imposing the non-negativity constraints on forecast reconciliation. It was observed that slight gains have occurred at the most disaggregated level. At the aggregated level slight losses were also observed. Although the gains or losses are negligible, the procedure plays an important role in decision and policy implementation processes.

Keywords: Aggregation, Australian tourism, Coherent forecasts, Contemporaneous error correlation, Forecast combinations, Least squares, Non-negative, Spatial correlations.

1 Introduction

Forecast reconciliation is the problem of ensuring that disaggregated forecasts add up to the corresponding forecasts of the aggregated time series. This is a common problem in manufacturing, for example, where time series of sales are disaggregated in several ways — by region, product-type, and so on. There are often tens of thousands of forecasts at the most disaggregated level, and these are required to sum to give forecasts at higher levels of aggregation — a property known as “coherence”.

A simple solution would be to forecast the most disaggregated series and sum the results. However, this tends to give poor aggregate forecasts due to the low signal-to-noise ratio in the disaggregated time series. Instead, a better solution is to forecast all the series at all levels of aggregation, and then reconcile them so they are coherent; that is, so that the forecasts of the disaggregated series add up to the forecasts of the aggregated series. Least squares reconciliation was proposed by Hyndman et al. (2011), whereby the reconciled forecasts are as close as possible (in the L_2 sense) to the original (“base”) forecasts subject to the aggregation constraint. This result was extended by Hyndman, Lee & Wang (2016) to a larger class of reconciliation problems, and by Wickramasuriya, Athanasopoulos & Hyndman (2019) who showed that the resulting reconciled forecasts are minimum variance unbiased estimators.

Most of the applications that we have found are inherently non-negative in nature, where the time series take only non-negative values such as revenue, demand, and counts of people. In such circumstances, it is important to ensure that the reconciled forecasts are non-negative, as forecasters and practitioners need to be able to make meaningful managerial decisions. Unfortunately, the MinT approach proposed by Wickramasuriya, Athanasopoulos & Hyndman (2019) and its variants fail to guarantee this property even when the base forecasts are non-negative.

This can be overcome by explicitly imposing the non-negativity constraints on the reconciliation procedure. One simple technique is to overwrite any negatives in the reconciled forecasts with zeros. However, the resulting approximate solution has ill-defined mathematical properties. This type of overwriting approximation method is commonly used in alternating least squares estimations. The problem is that it does not necessarily lower the objective function in each iteration, and therefore the convergence of the solution to the least squares minimum is not guaranteed. There are available algorithms that solve these problems in a mathematically rigorous way; we discuss how these can be applied to the forecast reconciliation problem.

2 The optimization problem

2.1 Notation and MinT reconciliation

We follow the notation of Wickramasuriya, Athanasopoulos & Hyndman (2019) and let $\mathbf{y}_t \in \mathbb{R}^m$ denote a vector of observations at time t comprising all series of interest including both disaggregated and aggregated time series. We also define $\mathbf{b}_t \in \mathbb{R}^n$ to be the vector of the most disaggregated series at time t . These two vectors are connected via $\mathbf{y}_t = \mathbf{S}\mathbf{b}_t$ where \mathbf{S} is the “summing” matrix of order $m \times n$ showing how the various aggregated time series in \mathbf{y}_t are constructed from the disaggregated series in \mathbf{b}_t .

Let $\hat{\mathbf{y}}_T(h)$ is a vector of initial (“base”) h -step-ahead forecasts, made at time T , stacked in the same order as \mathbf{y}_t . These will not generally be coherent. The least squares reconciled forecasts are given by

$$\tilde{\mathbf{y}}_T(h) = \mathbf{S}\tilde{\mathbf{b}}_T(h),$$

where

$$\tilde{\mathbf{b}}_T(h) = \left[(\mathbf{S}'\mathbf{\Lambda}_h^{-1}\mathbf{S})^{-1}\mathbf{S}'\mathbf{\Lambda}_h^{-1} \right] \hat{\mathbf{y}}_T(h), \quad (1)$$

and $\mathbf{\Lambda}_h$ is a weighting matrix. Wickramasuriya, Athanasopoulos & Hyndman (2019) showed that setting $\mathbf{\Lambda}_h = \text{var}[\mathbf{y}_{t+h} - \hat{\mathbf{y}}_t(h) \mid \mathcal{I}_t]$ to be the covariance matrix of the h -step-ahead base forecast errors minimizes the trace of $\text{var}[\mathbf{y}_{t+h} - \tilde{\mathbf{y}}_t(h) \mid \mathcal{I}_t]$ amongst all possible unbiased reconciliations. They derived an alternative expression for $\tilde{\mathbf{b}}_T(h)$:

$$\tilde{\mathbf{b}}_T(h) = \left[\mathbf{J} - \mathbf{J}\mathbf{\Lambda}_h\mathbf{U}(\mathbf{U}'\mathbf{\Lambda}_h\mathbf{U})^{-1}\mathbf{U}' \right] \hat{\mathbf{y}}_T(h), \quad (2)$$

where $\mathbf{J} = [\mathbf{0}_{n \times (m-n)} \mid \mathbf{I}_n]$, $\mathbf{U}' = [\mathbf{I}_{m-n} \mid -\mathbf{C}_{(m-n) \times n}]$, and $\mathbf{S} = \begin{bmatrix} \mathbf{C}_{(m-n) \times n} \\ \mathbf{I}_n \end{bmatrix}$. The use of Eq. (2) is computationally less demanding especially for high-dimensional hierarchical time series. It needs only one matrix inversion of order $(m-n) \times (m-n)$, whereas Eq. (1) needs two matrix inversions of orders $m \times m$ and $n \times n$. Typically in many applications $m-n < n < m$.

2.2 A quadratic programming solution

To ensure that all entries in $\tilde{\mathbf{y}}_T(h)$ are non-negative, it is sufficient to guarantee that all entries in $\tilde{\mathbf{b}}_T(h)$ are non-negative. Even though the solution of $\tilde{\mathbf{b}}_T(h)$ is derived based on a minimization of the variances of the reconciled forecast errors across the entire structure, it is also apparent from Eq. (1) that $\tilde{\mathbf{b}}_T(h)$ is the generalized least squares solution to the following regression problem:

$$\min_{\tilde{\mathbf{b}}} \frac{1}{2} [\hat{\mathbf{y}}_T(h) - \mathbf{S}\tilde{\mathbf{b}}]' \mathbf{\Lambda}_h^{-1} [\hat{\mathbf{y}}_T(h) - \mathbf{S}\tilde{\mathbf{b}}] = \min_{\tilde{\mathbf{b}}} \frac{1}{2} \tilde{\mathbf{b}}' \mathbf{S}' \mathbf{\Lambda}_h^{-1} \mathbf{S} \tilde{\mathbf{b}} - \tilde{\mathbf{b}}' \mathbf{S}' \mathbf{\Lambda}_h^{-1} \hat{\mathbf{y}}_T(h) + \frac{1}{2} \hat{\mathbf{y}}_T'(h) \mathbf{\Lambda}_h^{-1} \hat{\mathbf{y}}_T(h).$$

This suggests that the non-negativity issue can be handled by solving the following quadratic programming problem:

$$\begin{aligned} \min_{\tilde{\mathbf{b}}} q(\tilde{\mathbf{b}}) &:= \min_{\tilde{\mathbf{b}}} \frac{1}{2} \tilde{\mathbf{b}}' \mathbf{S}' \mathbf{\Lambda}_h^{-1} \mathbf{S} \tilde{\mathbf{b}} - \tilde{\mathbf{b}}' \mathbf{S}' \mathbf{\Lambda}_h^{-1} \hat{\mathbf{y}}_T(h) \\ \text{s.t. } &\tilde{\mathbf{b}} \geq \mathbf{0}, \end{aligned} \quad (3)$$

where the inequalities in Eq. (3) have to hold component-wise. The final non-negative reconciled forecasts are then

$$\check{\mathbf{y}}_T(h) = \mathbf{S}\check{\mathbf{b}}_T(h),$$

where $\check{\mathbf{b}}_T(h)$ is the solution to the quadratic programming problem in Eq. (3). This estimation problem is also referred to as “non-negative least squares” (NNLS).

There are a few important features of this minimization problem that are worth discussing. Consider the following definitions:

Definition 2.1. A vector $\tilde{\mathbf{b}}$ is said to be feasible if it satisfies all of the constraints in the quadratic programming problem in Eq. (3). The feasible region is the set of all feasible vectors $\tilde{\mathbf{b}}$, and the quadratic programming problem is said to be feasible if the feasible region is non-empty.

Definition 2.2. If $\mathbf{S}' \mathbf{\Lambda}_h^{-1} \mathbf{S}$ is a positive definite matrix, i.e., $\mathbf{x}' (\mathbf{S}' \mathbf{\Lambda}_h^{-1} \mathbf{S}) \mathbf{x} > 0$ for all $\mathbf{x} \neq \mathbf{0}$, then the objective function of the minimization problem in Eq. (3) is a strictly convex function.

It is easy to show that $\mathbf{S}' \mathbf{\Lambda}_h^{-1} \mathbf{S}$ is a positive definite matrix by using the fact that the matrix \mathbf{S} is of full column rank and assuming that $\mathbf{\Lambda}_h$ is positive definite. These simple non-negativity constraints will also ensure that the feasible region is non-empty. Therefore, the quadratic

programming problem in Eq. (3) has a unique global solution; i.e., there are no local minima apart from the global minimum (Turlach & Wright, 2015).

Unlike Wickramasuriya, Athanasopoulos & Hyndman (2019), we will not impose a constraint of unbiasedness on the reconciled forecasts obtained as the solution of the minimization problem in Eq. (3).

The well-known quadratic programming problem in Eq. (3) is easy to solve for small scale hierarchical or grouped structures using the quadprog package for R, which is designed to handle dense matrices (Turlach & Weingessel, 2013). However, we require matrices to be stored in a sparse format due to the large size of the structures that typically arise in forecast reconciliation.

2.3 First-order optimality conditions

We can derive first-order necessary conditions for $\check{\mathbf{b}}_T(h)$ to minimize the non-negative least squares problem.

Consider the Lagrangian function for the minimization problem in Eq. (3):

$$\mathcal{L}(\check{\mathbf{b}}, \boldsymbol{\lambda}) = q(\check{\mathbf{b}}) - \boldsymbol{\lambda}'\check{\mathbf{b}},$$

where $q(\check{\mathbf{b}}) = \frac{1}{2}\check{\mathbf{b}}' \mathbf{S}' \boldsymbol{\Lambda}_h^{-1} \mathbf{S} \check{\mathbf{b}} - \check{\mathbf{b}}' \mathbf{S}' \boldsymbol{\Lambda}_h^{-1} \hat{\mathbf{y}}_T(h)$ and $\boldsymbol{\lambda}$ is a Lagrange multiplier vector. The Karush-Kuhn-Tucker (KKT) optimality conditions that need to be satisfied by $\check{\mathbf{b}}_T(h)$ are

$$\nabla_{\check{\mathbf{b}}} \mathcal{L}[\check{\mathbf{b}}_T(h), \boldsymbol{\lambda}^*] = \mathbf{0}, \quad (4a)$$

$$\check{b}_{T,i}(h) = 0, \quad \forall i \in \mathcal{A}[\check{\mathbf{b}}_T(h)], \quad (4b)$$

$$\check{b}_{T,i}(h) > 0, \quad \forall i \notin \mathcal{A}[\check{\mathbf{b}}_T(h)], \quad (4c)$$

$$\lambda_i^* \geq 0, \quad \forall i \in \mathcal{A}[\check{\mathbf{b}}_T(h)], \quad (4d)$$

$$\lambda_i^* = 0, \quad \forall i \notin \mathcal{A}[\check{\mathbf{b}}_T(h)], \quad (4e)$$

$$\lambda_i^* \check{b}_{T,i}(h) = 0, \quad \forall i \in \{1, 2, \dots, n\}, \quad (4f)$$

where $\check{b}_{T,i}(h)$ is the i th component of $\check{\mathbf{b}}_T(h)$ and $\mathcal{A}[\check{\mathbf{b}}_T(h)]$ is referred to as the active set, and is defined as

$$\mathcal{A}[\check{\mathbf{b}}_T(h)] = \left\{ i \in \{1, 2, \dots, n\} \mid \check{b}_{T,i}(h) = 0 \right\}.$$

The first optimality condition in Eq. (4a) leads to λ^* being computed as

$$\lambda^* = \nabla q[\check{\mathbf{b}}_T(h)] = \mathbf{S}'\mathbf{\Lambda}_h^{-1}\mathbf{S}\check{\mathbf{b}}_T(h) - \mathbf{S}'\mathbf{\Lambda}_h^{-1}\hat{\mathbf{y}}_T(h).$$

The conditions given in Eq. (4f) are referred as complementarity conditions. They indicate that at the optimal solution, either the i th constraint is active or $\lambda_i^* = 0$, or both. Specifically, it implies that the Lagrange multipliers that are associated with inactive inequality constraints are zero.

3 Algorithms

Since the pioneering work of Lawson & Hanson (1974), a variety of methods for solving the non-negative least squares problem have been proposed. In general, these can be divided into two main categories: active set(-like) and projection-based methods. The following sections briefly explain a few of the algorithms that belong to each of these classes and that are suitable for solving large-scale problems. A detailed review of methods for non-negative least squares is given by Chen & Plemmons (2009).

3.1 Active set(-like) methods

The first widely used active set method for solving non-negative least squares was that proposed by Lawson & Hanson (1974). The basic idea of this method is to transform the inequality constrained least squares problem into a sequence of equality constrained problems.

The feasible region that corresponds to a given set of consistent inequality constraints is given by a volume in multidimensional space. Any point in this region (a feasible solution) consists of one of two types of variables, depending on whether it lies on the boundary or within the feasible region. The former set of variables is referred to as the “active set”. These are variables that can only lower the objective function by moving away from the feasible region, and must be assumed to lie on the boundary in order to ensure the feasibility of the solution. The latter set of variables is referred to as the “passive set”. The active set method relies on the idea that, if the true active set is known in advance, the solution to the least squares problem is given by the unconstrained least squares solution, obtained by using the variables in the passive set and setting the regression coefficients for the active set variables to zero. This can be interpreted as an attempt to forecast hierarchical or grouped time series by simply refining the structure through the removal of the bottom-level series that are responsible for any negative or zero

reconciled forecasts. In practice, though, it is difficult to define such sets in advance. Therefore, we have to search for the optimal set iteratively.

Even though the standard active set method guarantees to terminate within a finite number of iterations, it has pessimistic upper bound on the numbers of iterations needed to reach the optimal solution. Suppose that we have n series at the bottom level, resulting in a total of 2^n possible active sets. Since the solution is unique, one of the 2^n combinations corresponds to the optimal active set. Therefore, finding the optimal set gives rise to a problem known as combinatorial difficulty. The main shortcoming of the standard active set method is that the algorithm is initialized by assuming an empty passive set, and only one variable is added from active set to passive set. Although QR updating and down-dating techniques are used to speed up the computations, more iterations (in other words, more time) might be required to find the optimal active set when handling large scale non-negative least squares problems. Therefore, several alternatives have been proposed for overcoming these difficulties.

One possibility is to initialize the algorithm with a non-empty passive set. A simple way to introduce such a set is to solve the original unconstrained least squares problem and then overwrite any negative values with zeros. This type of initialization is used widely in the field of multivariate curve resolution (Andrew & Hancewicz, 1998; Gemperline & Cash, 2003). Wang & Wang (2012) pointed out that such a guess would be reasonable in cases where there are few active constraints or when non-negativity is enforced in order to remove any observational or measurement errors that may be present. They proposed an alternative initialization strategy by using the solution of gradient projection methods after a few iterations. They showed theoretically that the sequence of sets generated by the projection-based algorithm converges to the optimal passive set of the active set method. Furthermore, they compared their choice of initialization with that of the overwrite solution of unconstrained least squares, and found the former to be better.

The second approach to enhancing the speed of the active set method involves including more than one variable from active set. This should be handled carefully, as it could lead to endless loops in the algorithm. Thus, these types of methods, which are referred to as “block principal pivoting methods”, include a procedure for selecting a group of variables to exchange, and a backup rule in order to ensure the finite termination of the algorithm. Of the different variations of the active set methods that we have discussed thus far, we are particularly interested in this

type of active-set-like method, as they are quite efficient for handling large scale strictly convex quadratic programming problems with non-negativity constraints.

3.1.1 Block principal pivoting method

This section briefly reviews the theory behind the block principal pivoting method, which was developed for non-negative least squares problems by Júdice & Pires (1994). The procedure begins with the monotone linear complementarity problem (LCP) induced by the KKT optimal conditions given in Eq. (4), which needs to be satisfied by the optimal solution.

The monotone linear complementarity problem is

$$\ddot{\mathbf{g}} = \mathbf{S}'[\mathbf{\Lambda}_h^{-1}\mathbf{S}\ddot{\mathbf{b}} - \mathbf{\Lambda}_h^{-1}\hat{\mathbf{y}}_T(h)], \quad (5)$$

$$\ddot{\mathbf{g}} \geq \mathbf{0}, \quad (6)$$

$$\ddot{\mathbf{b}} \geq \mathbf{0}, \quad (7)$$

$$\ddot{b}_i \ddot{g}_i = 0, \quad i = 1, 2, \dots, n, \quad (8)$$

where \ddot{b}_i and \ddot{g}_i are the i th components of the vectors $\ddot{\mathbf{b}}$ and $\ddot{\mathbf{g}}$ respectively. The matrix $\mathbf{S}'\mathbf{\Lambda}_h^{-1}\mathbf{S}$ is positive definite, as \mathbf{S} is of full column rank and $\mathbf{\Lambda}_h$ is assumed to be positive definite. Thus, this strictly monotone LCP has a unique solution.

A point $(\ddot{\mathbf{b}}, \ddot{\mathbf{g}}) \in \mathbb{R}^{2n}$ is defined as a complementary solution if it satisfies Eq. (5) and (8).

Let the index set $\{1, 2, \dots, n\}$ be partitioned into two mutually exclusive subsets F and G , such that $F \cup G = \{1, 2, \dots, n\}$ and $F \cap G = \emptyset$. Furthermore, consider the partitions of $\ddot{\mathbf{b}}, \ddot{\mathbf{g}}$ and \mathbf{S} according to the index sets F and G using the following notation:

$$\begin{aligned} \ddot{\mathbf{b}}_F &= [\ddot{b}_i]_{i \in F}, & \ddot{\mathbf{b}}_G &= [\ddot{b}_i]_{i \in G}, \\ \ddot{\mathbf{g}}_F &= [\ddot{g}_i]_{i \in F}, & \ddot{\mathbf{g}}_G &= [\ddot{g}_i]_{i \in G}, \\ \mathbf{S}_F &= [\mathbf{S}_i]_{i \in F}, & \mathbf{S}_G &= [\mathbf{S}_i]_{i \in G}, \end{aligned}$$

where \mathbf{S}_i is the i th column of \mathbf{S} .

Based on these partitions, we can re-express Eq. (5) as

$$\begin{bmatrix} \ddot{\mathbf{g}}_F \\ \ddot{\mathbf{g}}_G \end{bmatrix} = \begin{bmatrix} \mathbf{S}'_F \mathbf{\Lambda}_h^{-1} \mathbf{S}_F & \mathbf{S}'_F \mathbf{\Lambda}_h^{-1} \mathbf{S}_G \\ \mathbf{S}'_G \mathbf{\Lambda}_h^{-1} \mathbf{S}_F & \mathbf{S}'_G \mathbf{\Lambda}_h^{-1} \mathbf{S}_G \end{bmatrix} \begin{bmatrix} \ddot{\mathbf{b}}_F \\ \ddot{\mathbf{b}}_G \end{bmatrix} - \begin{bmatrix} \mathbf{S}'_F \mathbf{\Lambda}_h^{-1} \hat{\mathbf{y}}_T(h) \\ \mathbf{S}'_G \mathbf{\Lambda}_h^{-1} \hat{\mathbf{y}}_T(h) \end{bmatrix}.$$

The algorithm starts by assigning $\ddot{\mathbf{b}}_G = \mathbf{0}$ and $\ddot{\mathbf{g}}_F = \mathbf{0}$. This particular choice will ensure that Eq. (8) is always satisfied for any values of $\ddot{\mathbf{b}}_F$ and $\ddot{\mathbf{g}}_G$.

The computation of the remaining unknown quantities $\ddot{\mathbf{b}}_F$ and $\ddot{\mathbf{g}}_G$ can be carried out by solving the following unconstrained least squares problem:

$$\bar{\mathbf{b}}_F = \min_{\bar{\mathbf{b}}_F} \frac{1}{2} \|\mathbf{S}_F \bar{\mathbf{b}}_F - \hat{\mathbf{y}}_T(h)\|_{\Lambda_h^{-1}}^2, \quad (9)$$

and then setting

$$\bar{\mathbf{g}}_G = \mathbf{S}_G' [\Lambda_h^{-1} \mathbf{S}_F \bar{\mathbf{b}}_F - \Lambda_h^{-1} \hat{\mathbf{y}}_T(h)], \quad (10)$$

where $\|\cdot\|_{\Lambda_h^{-1}} = [\mathbf{S}_F \bar{\mathbf{b}}_F - \hat{\mathbf{y}}_T(h)]' \Lambda_h^{-1} [\mathbf{S}_F \bar{\mathbf{b}}_F - \hat{\mathbf{y}}_T(h)]$. The solution pair $(\bar{\mathbf{b}}_F, \bar{\mathbf{g}}_G)$ is referred to as a complementary basic solution. If a complementary basic solution satisfies $\bar{\mathbf{b}}_F \geq \mathbf{0}$ and $\bar{\mathbf{g}}_G \geq \mathbf{0}$, then it is said to be feasible. In other words, it leads to the optimal solution of the constrained quadratic programming problem that we are interested in. Otherwise, the complementary basic solution is infeasible; i.e., there exists at least one $i \in F$ with $\bar{b}_i < 0$ or one $i \in G$ with $\bar{g}_i < 0$.

In the presence of an infeasible solution, we need to update sets F and G by exchanging the variables for which Eqs. (6) or (7) do not hold. Define the following two sets, which correspond to the infeasibilities in sets F and G :

$$I_1 = \{i \in F : \bar{b}_i < 0\} \quad \text{and} \quad I_2 = \{i \in G : \bar{g}_i < 0\}, \quad (11)$$

which gives the total set of infeasibilities, $I = I_1 \cup I_2$. For a given $\bar{I}_1 \subseteq I_1$ and $\bar{I}_2 \subseteq I_2$, F and G can be updated according to the following rules:

$$F = (F - \bar{I}_1) \cup \bar{I}_2 \quad \text{and} \quad G = (G - \bar{I}_2) \cup \bar{I}_1. \quad (12)$$

The first block principal pivoting algorithm for solving a strictly monotonic LCP is due to the work of Kostreva (1978). Unfortunately, unlike the active set method, in which the variable to be exchanged is chosen carefully with the aim of reducing the objective function, the use of blocks of variables for exchange can lead to a cycle, meaning that it is not guaranteed to provide the optimal solution. Although this occurs rarely, it is problematic. Júdice & Pires (1989) evaluated the performance of this algorithm on several test problems with a symmetric positive definite matrix. Their experiments concluded that this algorithm is quite efficient for handling

large-scale strictly monotonic LCPs. Furthermore, they noted that the occurrence of cycles is extremely rare. In a later paper, Júdice & Pires (1994) proposed an extension of this algorithm to include finite termination, by incorporating Murty's single principal pivoting algorithm. Even though the algorithm proposed by Murty (1974) has finite termination, convergence can be slow for applications with large numbers of variables, as it changes only one variable per iteration. However, this algorithm is still beneficial for obtaining a complementary basic solution with a smaller number of infeasibilities than before. The steps of the hybrid algorithm are given in Algorithm 3.1. $\text{card}(X)$ denotes the cardinality of set X .

Algorithm 3.1 Block principal pivoting algorithm

INITIALIZATION: Let $F = \emptyset$, $G = \{1, 2, \dots, n\}$, $\check{\mathbf{b}} = \mathbf{0}$, $\check{\mathbf{g}} = -\mathbf{S}'\mathbf{\Lambda}_h^{-1}\hat{\mathbf{y}}_T(h)$, $p = \bar{p} \leq 10$, and $\text{ninf} = n + 1$, and let α be a permutation of the set $\{1, 2, \dots, n\}$.

- 1: IF ($\check{\mathbf{b}}_F \geq \mathbf{0}$ & $\check{\mathbf{g}}_G \geq \mathbf{0}$) THEN
 - 2: Terminate the algorithm and $\check{\mathbf{b}} = (\check{\mathbf{b}}_F, \mathbf{0})$ is the unique global solution.
 - 3: ELSE
 - 4: Define I_1 and I_2 as given in Eq. (11).
 - 5: IF ($\text{card}(I_1 \cup I_2) < \text{ninf}$) THEN
 - 6: Set $\text{ninf} = \text{card}(I_1 \cup I_2)$, $p = \bar{p}$, $\bar{I}_1 = I_1$ and $\bar{I}_2 = I_2$.
 - 7: ELSEIF ($\text{card}(I_1 \cup I_2) \geq \text{ninf}$ and $p \geq 1$) THEN
 - 8: Set $p = p - 1$, $\bar{I}_1 = I_1$ and $\bar{I}_2 = I_2$.
 - 9: ELSEIF ($\text{card}(I_1 \cup I_2) \geq \text{ninf}$ and $p = 0$) THEN
 - 10: Set $\bar{I}_1 = \{r\}$ and $\bar{I}_2 = \emptyset$, if $r \in I_1$,
 - 11: $\bar{I}_1 = \emptyset$ and $\bar{I}_2 = \{r\}$, if $r \in I_2$,
 - 12: where r is the last element of the set $I_1 \cup I_2$ as for the order defined by α .
 - 13: ENDIF
 - 14: Update F and G as given by Eq. (12).
 - 15: Compute $\bar{\mathbf{b}}_F$ and $\bar{\mathbf{g}}_G$ using Eqs. (9) and (10) respectively, and assign $\check{\mathbf{b}}_F = \bar{\mathbf{b}}_F$ and $\check{\mathbf{g}}_G = \bar{\mathbf{g}}_G$.
 - 16: Return to line 1.
 - 17: ENDIF
-

In particular, if $\mathbf{\Lambda}_h$ is a diagonal matrix with positive elements and $\hat{\mathbf{y}}_T(h) > \mathbf{0}$, Algorithm 3.1 can start by defining the initial conditions as

$$F = \{1, 2, \dots, n\}, G = \emptyset, \check{\mathbf{b}} = \tilde{\mathbf{b}}, \check{\mathbf{g}} = \mathbf{0},$$

where $\tilde{\mathbf{b}}$ is the original unconstrained least squares solution with positive diagonal elements for $\mathbf{\Lambda}_h$.

Rather than selecting a subset of variables from I_1 and I_2 , we speed up the computations by using the full sets of variables as \bar{I}_1 and \bar{I}_2 respectively. This is generally referred to as the “full exchange rule”. The variable p in Algorithm 3.1 acts as a buffer for determining the number of full exchange rules that may be tried. If the number of infeasible variables increases due to the exchange rule, then p decreases by one. If the number of infeasible variables is still larger after $p - 1$ steps, then Murty’s method is used until it reaches a complementary solution with a smaller number of infeasible variables than a certain required value that is stored in $ninf$. This will occur in a finite number of iterations, due to the finite termination property of the Murty’s method. The algorithm then switches to the full exchange rule, and iterates until the number of infeasible variables reaches zero.

This algorithm works well for numerically non-degenerate problems; i.e., each variable in F and G has a value that is clearly different from zero, given by the unconstrained solver in the feasible solution. When such is not the case, a variable with a value that is close to zero may be passed between F and G , due to the slight negative error caused by the unconstrained solver. In handling this issue, Cantarella & Piatek (2004) suggested that we assign

$$\begin{aligned}\ddot{b}_{F,i} &= 0, & \text{if } i \in H_1, \\ \ddot{g}_{G,i} &= 0, & \text{if } i \in H_2,\end{aligned}$$

where $H_1 = \{i \in F : \ddot{b}_i < \epsilon\}$ and $H_2 = \{i \in G : \ddot{g}_i < \epsilon\}$; $\ddot{b}_{F,i}$ and $\ddot{g}_{G,i}$ are the i th components of the \ddot{b}_F and \ddot{g}_G vectors respectively, and ϵ is set to 10^{-12} .

The choice of p plays an important rule. It should be fairly small in order to prevent unnecessary computations in which the full exchange rule is not effective. However, it should not be too small, otherwise Murty’s method may be activated several times, thus reducing the efficiency of the algorithm. In general, $p = 3$ is a good choice (Júdice & Pires, 1994).

The performance of the block pivoting algorithm depends heavily on the number of times the full exchange rule fails and it has to switch to Murty’s method. Based on an extensive study, Kim & Park (2011) noted that Murty’s method was not activated during the algorithm, which suggests that the full exchange rule will work effectively in practice.

3.2 Projection-based methods

This class includes a set of methods that can incorporate multiple active constraints at each iteration, using the gradient information. The gradient projection methods that belong to this

class is commonly used in practice, and is extremely useful for obtaining a set of non-negative reconciled forecasts for hierarchical and grouped time series because, in this context, (i) the constraints defined are simple, and (ii) the matrices involved are large and/or sparse (Nocedal & Wright, 2006). These facts facilitate the efficient performance of the computations, because iterations of the algorithm mainly involve projections onto the feasible set and the determination of a (steepest) descent direction.

The following sections review the details of two algorithms that are of interest in this paper. The first algorithm combines the gradient projection with a conjugate gradient method in order to solve the non-negative least squares problem (Nocedal & Wright, 2006). The second algorithm uses the scaled gradient projection method, which differs from standard gradient projection methods by the presence of a scaling matrix that multiplies the gradient. The numerical experiments have shown that this method is also effective at handling large scale problems for a proper scaling matrix (Bonettini, Zanella & Zanni, 2009).

There are also various other methods that fall into this class. For a detailed discussion, the interested reader is referred to the work of Chen & Plemmons (2009) and the references provided therein.

3.2.1 Gradient projection + conjugate gradient approach

Each iteration of the algorithm is designed to follow two main steps. In the first step, the current feasible solution $\tilde{\mathbf{b}}$ is updated by searching along the steepest direction; in other words, the direction $-\tilde{\mathbf{g}}$ from $\tilde{\mathbf{b}}$. If the lower bound of the inequality constraints (i.e., 0) is encountered before a minimizer is found along the line, the search direction is “bent” to ensure that it remains feasible. The search is continued along the resulting piecewise-linear path in order to locate the first local minimizer of the objective function, q . This point is referred to as the Cauchy point. Based on the Cauchy point, it is possible to define a set of constraints that are active at this point. Hence, in the second step, a subproblem is solved by fixing the constraints of the active set to zero. The key steps of this method are given in Algorithm 3.2. Refer Nocedal & Wright (2006) for a detailed implementation of each step involved.

3.2.2 Scaled gradient projection

This section briefly reviews the details of the diagonally scaled gradient projection algorithm proposed by Bonettini, Zanella & Zanni (2009). The algorithm propagates by determining a descent direction at each iteration, based on the current feasible solution, step length and scaling

Algorithm 3.2 Gradient projection based on the Cauchy pointINITIALIZATION: Choose a feasible initial solution $\check{\mathbf{b}}^0$.

```

1: FOR k in 0, 1, 2, ... DO
2:   IF  $\check{\mathbf{b}}^k$  satisfies the KKT conditions THEN
3:     Terminate the algorithm.  $\check{\mathbf{b}} = \check{\mathbf{b}}^k$  is the unique global solution.
4:   ELSE
5:     Find the Cauchy point  $\mathbf{b}^c$  using  $\check{\mathbf{b}}^k$ .
6:     Use projected conjugate gradient algorithm with a diagonal preconditioner to find an
       approximate feasible solution  $\check{\mathbf{b}}^+$  that satisfies  $q(\check{\mathbf{b}}^+) \leq q(\mathbf{b}^c)$ .
7:      $\check{\mathbf{b}}^{k+1} = \check{\mathbf{b}}^+$ .
8:   ENDIF
9: ENDFOR

```

matrix. The solution vector is then adjusted along this direction using a non-monotone line search that does not guarantee a decrease in the objective function value at each iteration. This is in order to increase the likelihood of locating a global optimum in practice (Birgin, Martínez & Raydan, 2003). The main steps are given in Algorithm 3.3.

In selecting the step-length parameter for the scaled gradient projection algorithm, Bonettini, Zanella & Zanni (2009) extended the original ideas of Barzilai & Borwein (1988) that are commonly used for improving the convergence rate of standard gradient methods. The generalized Barzilai & Borwein (BB) rules are

$$\alpha_k^{BB1} = \frac{(\mathbf{u}^{k-1})' \mathbf{D}^{-1} \mathbf{D}^{-1} \mathbf{u}^{k-1}}{(\mathbf{u}^{k-1})' \mathbf{D}^{-1} \mathbf{v}^{k-1}},$$

$$\alpha_k^{BB2} = \frac{(\mathbf{u}^{k-1})' \mathbf{D} \mathbf{v}^{k-1}}{(\mathbf{v}^{k-1})' \mathbf{D} \mathbf{D} \mathbf{v}^{k-1}},$$

where $\mathbf{u}^{k-1} = \check{\mathbf{b}}^k - \check{\mathbf{b}}^{k-1}$ and $\mathbf{v}^{k-1} = \ddot{\mathbf{g}}(\check{\mathbf{b}}^k) - \ddot{\mathbf{g}}(\check{\mathbf{b}}^{k-1})$. Specifically, these equations reduce to the standard BB rules when $\mathbf{D} = \mathbf{I}_n$.

It is interesting to note that when $[\check{\mathbf{b}}^k - \alpha_k \mathbf{D} \ddot{\mathbf{g}}(\check{\mathbf{b}}^k)]_+$ is used to generate a descent direction from $\check{\mathbf{b}}^k$, the diagonal scaling matrix with $\Lambda_h \propto \mathbf{I}_n$ and the generalized BB rules indicate that the whole process reduces to the use of a standard gradient projection method with standard BB rules. Thus, the scaled gradient projection algorithm might not be computationally advantageous for obtaining non-negative reconciled forecasts from the OLS approach proposed by Hyndman et al. (2011).

Algorithm 3.3 Diagonally scaled gradient projection algorithm

INITIALIZATION: Choose a feasible initial solution $\ddot{\mathbf{b}}^0$. Set the parameters $\eta, \theta \in (0, 1)$, $0 < \alpha_{\min} < \alpha_{\max}$, and a positive integer M . Use a diagonal scaling matrix $\mathbf{D} = \text{diag}(d_1, d_2, \dots, d_n)$ with elements $d_i = 1 / (\mathbf{S}' \mathbf{\Lambda}_h^{-1} \mathbf{S})_{ii}$ for $i = 1, 2, \dots, n$.

- 1: FOR k in $0, 1, 2, \dots$ DO
- 2: Choose the step-length parameter $\alpha_k \in [\alpha_{\min}, \alpha_{\max}]$ using the alternation strategy proposed by Bonettini, Zanella & Zanni (2009).
- 3: Projection: $\mathbf{z}^k = [\ddot{\mathbf{b}}^k - \alpha_k \mathbf{D}_k \ddot{\mathbf{g}}(\mathbf{b}^k)]_+$, where $(x)_+ = \max(0, x)$.
- 4: IF $\mathbf{z}^k = \ddot{\mathbf{b}}^k$ THEN
- 5: Terminate the algorithm. $\check{\mathbf{b}} = \mathbf{b}^k$ is the unique global minimum.
- 6: ELSE
- 7: Descent direction: $\mathbf{d}^k = \mathbf{z}^k - \ddot{\mathbf{b}}^k$.
- 8: Set $\lambda_k = 1$ and $q_{\max} = \max_{j \in \{0, 1, \dots, \min(k, M-1)\}} q(\ddot{\mathbf{b}}^{k-j})$
- 9: Backtracking loop:
 - IF $q(\ddot{\mathbf{b}}^k + \lambda_k \mathbf{d}^k) \leq q_{\max} + \eta \lambda_k [\ddot{\mathbf{g}}(\ddot{\mathbf{b}}^k)]' \mathbf{d}^k$ THEN
 - Go to line 10.
 - ELSE
 - Set $\lambda_k = \theta \lambda_k$ and go to line 9.
 - ENDIF
- 10: Set $\ddot{\mathbf{b}}^{k+1} = \ddot{\mathbf{b}}^k + \lambda_k \mathbf{d}^k$.
- 11: ENDIF
- 12: ENDFOR

Selection of tuning parameters

This section highlights the roles of the tuning parameters and along with their recommended values in practice.

- α_0 : We consider a method similar to that of Figueiredo, Nowak & Wright (2007), implemented in standard gradient projection methods, within the context of the scaled gradient projection methods. It considers the initial value α_0 as the exact minimizer of the objective function along the direction of $\ddot{\mathbf{b}}^0 - \alpha \mathbf{D} \ddot{\mathbf{g}}(\ddot{\mathbf{b}}^0)$, if no constraints are to be satisfied. This involves defining

$$\mathbf{p}^0 = (p_i^0) = \begin{cases} [\ddot{\mathbf{g}}(\ddot{\mathbf{b}}^0)]_i, & \text{if } \ddot{b}_i^0 > 0 \text{ or } [\ddot{\mathbf{g}}(\ddot{\mathbf{b}}^0)]_i < 0, \\ 0, & \text{otherwise.} \end{cases}$$

The initial guess is estimated as

$$\alpha_0 = \min_{\alpha} q[\ddot{\mathbf{b}}^0 - \alpha \mathbf{D} \ddot{\mathbf{g}}(\ddot{\mathbf{b}}^0)],$$

and can be computed analytically using

$$\alpha_0 = \frac{(\mathbf{p}^0)' \mathbf{D} \mathbf{p}^0}{(\mathbf{S} \mathbf{D} \mathbf{p}^0)' \mathbf{\Lambda}_h^{-1} (\mathbf{S} \mathbf{D} \mathbf{p}^0)}.$$

- η and θ control the amount by which the objective function should be decreased and the number of backtracking reductions to be performed, respectively. The values of $\eta = 10^{-4}$ and $\theta = 0.4$ have been used in order to get a sufficiently large step size with fewer reductions (Bonettini, Zanella & Zanni, 2009).
- α_{\min} and α_{\max} are the lower and upper bounds of the step-length parameter α_k , to avoid using unnecessary extreme values. Even though a large range is defined for the BB-type rules in practice, Bertero et al. (2013) found the interval $(10^{-5}, 10^5)$ to be suitable for the generalized BB rules.
- τ_1 is the initial switching condition that activates the step-length alternation strategy, and a value of 0.5 is suitable in many applications (Bonettini, Zanella & Zanni, 2009; Bertero et al., 2013).
- We set $M_{\kappa} = 3$, as was used by Bonettini, Zanella & Zanni (2009) and Bertero et al. (2013).
- M determines the monotone ($M = 1$) or non-monotone ($M > 1$) line search to be performed in the backtracking loop. This value should not be too large, as the decrease in the objective function is difficult to control, and is set to 10 here, as was done by Bonettini, Zanella & Zanni (2009).

4 Monte Carlo experiments

This section aims to show the practical usefulness of the aforementioned algorithms for obtaining a set of non-negative reconciled forecasts using a few special cases of MinT. For the sake of simplicity, the OLS and WLS based on structural weights (WLS_s) approaches are considered (Wickramasuriya, Athanasopoulos & Hyndman, 2019). The computational efficiency of these algorithms is evaluated over a series of hierarchies, ranging in size from small to large. We study the behaviours of two possible choices of the initial solution: (i) base forecasts at the bottom level;

and (ii) the unconstrained OLS or WLS_s forecasts. The latter choice can sometimes be a better alternative than the former, as it is aggregate consistent, and computationally less demanding. For the projection-based approaches, the unconstrained OLS or WLS_s forecasts are projected on to the non-negative orthant, as these algorithms need a feasible initial solution. However, the block principal pivoting algorithm can use the unconstrained solution in its original condition. The acronyms used to distinguish the algorithms and their variations of interest are listed in Table 1.

Table 1: *Acronyms for non-negative least squares algorithms.*

Algorithm	Notation
Scaled gradient projection	SGP
Gradient projection + conjugate gradient	PCG
Block principal pivoting	BPV

Sections 4.1 and 4.2 demonstrate for OLS and WLS_s respectively, the simulation design, the numbers of negative reconciled forecasts observed at the bottom level and the performances of the algorithms. All experiments are performed in R on a Linux machine equipped with a 3.20GHz Intel Quad-core processor and 8GB memory. A parallel computing environment with two workers is established by using the `doParallel` (Revolution Analytics & Weston, 2015a) and `foreach` (Revolution Analytics & Weston, 2015b) packages in R to parallelize the procedure of computing non-negative reconciled forecasts for different forecast horizons.

4.1 Non-negativity in the OLS approach

Initially, we consider a hierarchy with $K = 1$ level, having three series at the bottom level. The base forecast for the most aggregated series in the hierarchy is generated from a uniform distribution on the interval $(e^K, 1.2e^K)$, where K is the number of levels in the hierarchy. This is then disaggregated to the bottom level based on a set of proportions that sum to one. Specifically, a set of values is chosen from a gamma distribution, with the shape and scale parameters set to two, and the values are normalized to ensure that they sum to one. Noise is then added to the series at the aggregated levels to make them aggregate-inconsistent. If any of the base forecasts become negative after the noise is added, they are set to zero in order to ensure that all base forecasts in the hierarchy are strictly non-negative. The whole procedure is repeated until there is at least one negative reconciled forecast at the bottom level, and six of these sets with negative reconciled forecasts are used to denote a forecast horizon of length 6. The number of levels in the hierarchy is increased gradually to construct much larger hierarchies, by adding three

nodes below each of the bottom-level nodes in the preceding hierarchy. Table 2 presents the structure of each hierarchy constructed and the number of negative reconciled forecasts at the bottom level for each forecast horizon. Each hierarchy constructed contains approximately 10% of negative reconciled forecasts at the bottom level. These are then revised using the algorithms discussed in Section 3, in order to obtain a set of non-negative reconciled forecasts.

Table 2: *The structure of each hierarchy generated and the number of negative reconciled forecasts that result from the OLS approach.*

K	m	n	Forecast horizon (h)					
			1	2	3	4	5	6
1	4	3	1	1	1	1	1	1
2	13	9	1	1	1	2	1	2
3	40	27	1	1	3	2	1	1
4	121	81	1	4	7	2	1	5
5	364	243	9	12	11	15	4	18
6	1093	729	50	45	36	39	37	48
7	3280	2187	137	147	138	137	143	134
8	9841	6561	532	476	572	490	524	520
9	29524	19683	1604	1920	1589	1551	1778	1662
10	88573	59049	6087	6106	5888	5199	5243	6193
11	265720	177147	21696	19439	20935	19318	19121	21674
12	797161	531441	64668	63866	71347	68648	63031	69600

K : number of levels.

m : total number of series.

n : number of series at the bottom level.

Tables 3 and 4 present the numbers of iterations and average computational times in seconds (s) that are required to reach the KKT optimality conditions given in Eq. (4) when the base and (projected) unreconciled OLS forecasts, respectively, are used as the initial solution. Cases where an algorithm reaches the maximum number of iterations considered (10^4) are marked with an asterisk, and the average time corresponding to that number of iterations is given. The first row in each hierarchy of Table 3 gives the computational time required to produce the unconstrained OLS reconciled forecasts, which is always the best time, because the weights matrix computes only ones for all forecast horizons. The bold entries identify the non-negative algorithms with the best computational performances.

The main conclusion that can be drawn from these sets of results is that the BPV algorithm always has the best computational performance. This is due in part to the alternative analytical representation proposed for MinT (Wickramasuriya, Athanasopoulos & Hyndman, 2019). In addition, it should be noted that the computational performance of BPV algorithm depends

Table 3: Computational efficiency of the non-negative forecast reconciliation from the OLS approach using base forecasts as the initial solution.

K	m	n		Forecast horizon (h)						Time (s)
				1	2	3	4	5	6	
1	4	3	OLS							0.01
			SGP	3	3	3	3	3	3	0.03
			PCG	1	1	1	1	1	1	0.03
2	13	9	OLS							0.01
			SGP	495	435	431	659	35	517	2.34
			PCG	1	2	1	2	2	2	0.11
3	40	27	OLS							0.01
			SGP	268	299	476	272	333	416	1.69
			PCG	2	2	3	3	2	1	0.15
4	121	81	OLS							0.01
			SGP	384	387	10^{4*}	7700	510	518	21.85
			PCG	4	6	9	4	3	7	0.30
5	364	243	OLS							0.01
			SGP	10^{4*}	4381	976	6114	2948	2524	30.44
			PCG	9	11	12	15	7	16	0.53
6	1093	729	OLS							0.02
			SGP	10^{4*}	10^{4*}	10^{4*}	10^{4*}	10^{4*}	10^{4*}	74.22
			PCG	49	41	44	43	41	44	1.32
7	3280	2187	OLS							0.03
			SGP	10^{4*}	10^{4*}	10^{4*}	10^{4*}	10^{4*}	10^{4*}	77.53
			PCG	133	139	168	113	136	137	4.48
8	9841	6561	OLS							0.09
			PCG	461	445	440	384	423	374	26.25
9	29524	19683	OLS							0.19
			PCG	1296	1663	1206	1359	1625	1388	227.40
10	88573	59049	OLS							0.55
			PCG	5792	4046	3684	3551	4385	4100	3274.80

Notes: OLS defines the unconstrained OLS approach.

The computational time is averaged over 50 replications for $K = 1$ to $K = 9$, but only 10 for $K = 10$, as the computational time is considerable.

Only PCG is performed up to $K = 10$, due to the high computational time.

strongly on how often the full exchange rule fails and Murty's method or the back-up rule has to be activated. Interestingly, the back-up rule was inactive for all experiments carried out in this section. This is also observed in the tests performed by Kim & Park (2011). However, there is no theoretical justification for this, nor do we have conditions under which we know that the back-up rule is always inactive. Hence, the performance of the algorithm will be affected slightly if it is activated in a certain application.

Table 4: Computational efficiency of the non-negative forecast reconciliation from the OLS approach using (projected) unconstrained OLS forecasts as the initial solution.

K	m	n		Forecast horizon (h)						Time (s)
				1	2	3	4	5	6	
1	4	3	SGP	1	1	1	1	1	1	0.03
			PCG	1	1	1	1	1	1	0.03
			BPV	1	1	1	1	1	1	0.06
2	13	9	SGP	451	27	430	692	11	488	1.75
			PCG	1	1	1	1	1	1	0.07
			BPV	1	1	1	1	1	1	0.07
3	40	27	SGP	395	323	548	274	339	383	1.91
			PCG	1	1	2	1	1	1	0.12
			BPV	1	1	2	1	1	1	0.09
4	121	81	SGP	339	348	10 ^{4*}	2833	535	448	22.09
			PCG	1	1	2	2	1	1	0.20
			BPV	1	2	3	2	1	2	0.12
5	364	243	SGP	10 ^{4*}	10 ^{4*}	1715	10 ^{4*}	529	7930	64.04
			PCG	1	1	1	4	1	3	0.39
			BPV	1	1	1	2	1	2	0.12
6	1093	729	SGP	10 ^{4*}	10 ^{4*}	10 ^{4*}	10 ^{4*}	10 ^{4*}	10 ^{4*}	75.79
			PCG	38	9	13	11	15	18	1.04
			BPV	2	2	2	2	3	2	0.19
7	3280	2187	SGP	10 ^{4*}	10 ^{4*}	10 ^{4*}	10 ^{4*}	10 ^{4*}	10 ^{4*}	85.74
			PCG	49	49	71	54	64	43	2.64
			BPV	2	2	3	3	3	3	0.31
8	9841	6561	PCG	255	201	287	218	196	213	17.58
			BPV	3	3	3	2	3	2	1.13
9	29524	19683	PCG	699	781	704	630	807	679	182.34
			BPV	3	3	3	3	3	3	2.80
10	88573	59049	PCG	2622	2501	2166	2448	2081	2583	2302.11
			BPV	4	4	4	4	4	3	9.33
11	265720	177147	BPV	4	4	4	4	4	4	28.99
12	797161	531441	BPV	4	4	4	4	4	5	98.57

Notes: The computational time is averaged over 50 replications for $K = 1$ to $K = 9$, but only 10 for $K > 9$, as the computational time is considerable.

Only PCG is performed up to $K = 10$, due to the high computational time.

The second best timing is achieved by the PCG algorithm. Unfortunately, it is inefficient for larger hierarchies, as locating the Cauchy point can be time consuming. Of the two initial solutions considered, the (projected) unconstrained OLS is a good choice, as expected. The scaled gradient projection algorithm performed the worst. This is not surprising, as it reduces to using the standard gradient projection algorithm for the OLS approach.

4.2 Non-negativity in the WLS_s approach

The simulation design of Section 4.1 is biased against the scaled gradient projection algorithms. We therefore reassess the performances of the competing algorithms by changing the structure of the hierarchy and using WLS with structural weights as the forecast reconciliation approach.

As with the previous simulation setup, we consider a hierarchy with $K = 1$ level and three series at the bottom level. The base forecast for the most aggregated series in the hierarchy is generated from a uniform distribution on the interval $(1.5e^K, 2e^K)$, where K is the number of levels in the hierarchy. Then the same set of steps are followed as in Section 4.1 to generate six sets of reconciled forecasts with negatives at the bottom level. The number of levels in the hierarchy is increased gradually to construct much larger hierarchies, by adding a mixture of three and four nodes to the bottom-level nodes in the preceding hierarchy. Table 5 presents the structure of each hierarchy constructed and the number of negative reconciled forecasts at the bottom level for each forecast horizon. Each hierarchy contains approximately 10% of negative reconciled forecasts at the bottom level.

Table 5: *The structure of each hierarchy generated and the numbers of negative reconciled forecasts that result from the WLS_s approach.*

K	m	n	Forecast horizon (h)					
			1	2	3	4	5	6
1	4	3	1	1	1	1	1	1
2	14	10	1	1	1	2	1	1
3	49	35	1	1	1	1	2	1
4	171	122	1	1	3	5	2	4
5	598	427	13	10	10	9	6	7
6	2092	1494	62	38	60	56	44	43
7	7321	5229	211	229	255	270	222	203
8	25622	18301	1128	1166	1026	1268	1285	1186
9	89675	64053	4738	5619	6244	4812	5779	5637
10	249808	160133	17744	17790	16023	17261	19462	17258
11	650141	400333	36271	47845	44040	40790	47879	38753
12	1650974	1000833	105462	84998	76817	102130	95530	80090

Note: For $K > 9$, either two or three nodes are added to each of the bottom-level nodes in the preceding hierarchy.

Tables 6 and 7 present the numbers of iterations and the average computational times in seconds (s) that are required to reach the KKT optimality conditions when the base and projected (unconstrained) WLS_s forecasts, respectively, are used as the initial solution. Cases where an algorithm reaches the maximum number of iterations (10^4) are marked with an asterisk, and the

Table 6: Computational efficiency of the non-negative forecast reconciliation from the WLS_s approach using base forecasts as the initial solution.

K	m	n		Forecast horizon (h)						Time (s)
				1	2	3	4	5	6	
1	4	3	WLS_s							0.01
			ABB	3	3	3	3	3	3	0.04
			PCG	1	1	1	1	1	1	0.04
2	14	10	WLS_s							0.01
			ABB	342	265	23	481	22	497	1.52
			PCG	1	2	1	1	1	2	0.10
3	49	35	WLS_s							0.01
			ABB	276	461	391	392	737	423	1.86
			PCG	2	2	2	1	1	2	0.17
4	171	122	WLS_s							0.01
			ABB	456	496	389	480	364	498	2.13
			PCG	2	2	2	2	2	2	0.29
5	598	427	WLS_s							0.01
			ABB	375	357	379	352	359	270	1.94
			PCG	2	3	2	3	2	3	0.33
6	2092	1494	WLS_s							0.02
			ABB	1239	354	419	7803	354	382	29.11
			PCG	5	6	6	6	4	6	0.51
7	7231	5229	WLS_s							0.06
			ABB	1020	851	10^4^*	10^4^*	614	369	103.16
			PCG	9	8	8	9	7	7	2.09
8	25622	18301	WLS_s							0.19
			PCG	13	14	13	13	13	12	19.08
9	89675	64053	WLS_s							0.48
			PCG	16	16	16	17	17	21	244.68
10	249808	160133	WLS_s							1.43
			PCG	19	20	19	17	20	20	1660.12

Notes: WLS_s defines the unconstrained WLS_s approach.

The computational time is averaged over 50 replications for $K = 1$ to $K = 9$, but only 10 for $K = 10$, as the computational time is considerable.

Only PCG is performed up to $K = 10$, due to the high computational time.

average computational time corresponding to that number of iterations is given. The first row in each hierarchy of Table 6 gives the computational times required to produce the unconstrained WLS_s reconciled forecasts. As with OLS, this always shows the best timing. The bold entries identify the non-negative algorithms with the best computational performances.

The overall pattern that is observed in these two tables is quite similar to that in Section 4.1. However, for moderate-sized hierarchies, the scaled gradient projection algorithms based on BB

Table 7: Computational efficiency of the non-negative forecast reconciliation from the WLS_s approach using (projected) unconstrained WLS_s forecasts as the initial solution.

K	m	n		Forecast horizon (h)						Time (s)
				1	2	3	4	5	6	
1	4	3	ABB	1	1	1	1	1	1	0.03
			PCG	1	1	1	1	1	1	0.03
			BPV	1	1	1	1	1	1	0.06
2	14	10	ABB	287	17	22	512	357	35	0.88
			PCG	1	1	1	1	1	2	0.09
			BPV	1	1	1	1	1	2	0.08
3	49	35	ABB	290	347	297	374	621	388	1.80
			PCG	1	1	1	1	1	1	0.13
			BPV	1	1	1	1	1	1	0.07
4	171	122	ABB	515	312	423	431	368	12	2.02
			PCG	1	1	1	1	1	1	0.17
			BPV	1	1	1	1	1	1	0.09
5	598	427	ABB	455	380	363	334	419	368	2.27
			PCG	1	1	1	1	1	1	0.22
			BPV	1	1	1	1	2	1	0.12
6	2092	1494	ABB	1214	282	424	5947	390	340	22.32
			PCG	3	2	2	2	2	1	0.36
			BPV	2	2	1	2	2	1	0.17
7	7321	5229	ABB	879	663	10 ^{4*}	10 ^{4*}	766	374	101.72
			PCG	3	4	6	4	3	3	0.68
			BPV	2	2	2	2	2	2	0.57
8	25622	18301	PCG	6	8	8	7	6	5	3.18
			BPV	2	2	2	2	3	2	1.76
9	89675	64053	PCG	12	11	14	11	10	14	35.08
			BPV	3	3	3	3	3	3	6.45
10	249808	160133	PCG	16	16	18	17	17	16	250.22
			BPV	3	3	3	3	3	3	21.00
11	650141	400333	PCG	18	19	21	22	18	21	1597.10
			BPV	3	3	3	3	3	3	56.84
12	1650974	1000833	BPV	3	3	3	3	3	3	3247.09

Notes: The computational time is averaged over 50 replications for $K = 1$ to $K = 9$, but only 10 for $K > 9$, as the computational time is considerable.

Only PCG is performed up to $K = 11$, due to the high computational time.

rules occasionally terminate with fewer iterations than the cut-off, unlike the previous simulation setup. Nevertheless, the use of this algorithm is still problematic for large hierarchies.

In summary, both simulations performed illustrate that the block principal pivoting algorithm is computationally efficient in obtaining a set of non-negative reconciled forecasts. The projected conjugate gradient algorithm is the second best. To further examine the superiority of these competing algorithms on real applications, an extensive case study is provided in Section 5.

5 Non-negative reconciled forecasts for Australian domestic tourism flows

This section evaluates the impact of imposing the strict non-negativity constraints on the forecast reconciliation approaches, using the Australian domestic tourism flows as a case study. The grouped structure consisting of 555 series is considered. A detailed description of the structure and a comprehensive analysis of the forecast performances of different reconciliation approaches are given in Wickramasuriya, Athanasopoulos & Hyndman (2019). However, the base and reconciled forecasts in these applications are allowed to take any values on the real line; in other words, they have not explicitly restricted them to be non-negative, even though the original data are. Hence, this section considers the non-negative nature of these sets of forecasts, as obtaining a set of non-negative reconciled forecasts requires the base forecasts to be non-negative too. The empirical study performed in Wickramasuriya, Athanasopoulos & Hyndman (2019) is repeated by log-transforming the original data. For the series that include zeros, a small constant — specifically, half of the minimum positive value — is added and aggregated before transforming the data. Then, ARIMA and ETS models are fitted by minimizing the AICc to obtain the base forecasts using the default settings as implemented in the `forecast` package for R Hyndman (2019) and described in Hyndman & Khandakar (2008). If the forecast densities on the transformed data are symmetric, then the back-transformed base forecasts give the median of the forecast density. Even though this is acceptable in practice, we need to obtain the mean of the forecast densities, as means can be aggregated but medians cannot.

Considering the first three terms of the Taylor series expansion around μ (i.e., the mean on the transformed scale), the mean on the back-transformed scale can be obtained as

$$e^{\mu} \left(1 + \frac{1}{2} \sigma^2 \right),$$

where σ^2 is the variance on the transformed scale.

These back-transformed means (base forecasts) are then reconciled using OLS, WLS based on variance scaling (WLS_v) and MinT. MinT uses the covariance estimator that shrinks the sample correlation matrix towards an identity matrix; refer to Wickramasuriya, Athanasopoulos & Hyndman (2019) for further details about covariance estimators. The expanding window forecast evaluation procedure results in a total of 96 1-step-ahead reconciled forecasts, 95 2-step-ahead reconciled forecasts, and so on, down to 85 12-step-ahead reconciled forecasts. These include iterations in which all reconciled forecasts are positive. Table 8 presents the summary

statistics of the negative reconciled forecasts obtained from the ARIMA and ETS base forecasts. Each cell gives (i) the number of iterations reported with negative reconciled forecasts; and the minimum and maximum of (ii) the number of negative reconciled forecasts; (iii) the largest negative value (in thousands); and (iv) the smallest negative value (in thousands) resulting from each of the forecast reconciliation approaches. For example, the first cell in the table illustrates that all 96 iterations of the 1-step-ahead OLS reconciled forecasts include negative forecasts, and the number of negatives can be as small as 5 or as large as 80. The largest negative value varies between -71.8 and -2.33 , and the smallest negative value varies between -1.52 and -0.002 . It is clear that both the number of negative reconciled forecasts and their magnitudes are considerably large with the OLS approach. The presence of such negative values can degrade the quality of the forecasts in the entire structure.

These forecasts are then revised using the algorithms discussed in Section 3. For the scaled gradient projection and projected conjugate gradient algorithms, these forecasts are projected into the non-negative orthant and used as the initial solution. When the MinT approach is coupled with the block principal pivoting algorithm, a vector of zeros is used as the initial solution. Table 9 summarizes the total computational time (in seconds) of each forecast reconciliation approach for obtaining a set of non-negative reconciled forecasts. Cases where an algorithm reaches the maximum number of iterations (10^4) are marked with an asterisk, and the computational time that corresponds to that number of iterations is given. For example, the first cell in the table illustrates that the SGP algorithm takes 971.1 seconds to revise all 96 iterations with negative reconciled forecasts.

As was observed in the simulation exercise, the BPV algorithm showed the best computational performance for the OLS approach, while PCG was the second best. The worst performance of the scaled gradient projection algorithm fits with the previous findings. The WLS_v approach based on the SGP algorithm showed the best performances, challenging the BPV algorithm.

Even though MinT resulted in fewer negative reconciled forecasts, the computational time is much larger than the best timings of the WLS_v and even OLS approaches using the BPV and PCG algorithms. This is reasonable because a full variance covariance matrix means that some of the matrix manipulations can result in dense matrices, and hence do not benefit fully from the special matrix multiplication strategies that we considered for the OLS and WLS_v approaches.

The impact of imposing the non-negativity constraints on the estimation procedure is evaluated by comparing the non-negative reconciled forecasts obtained using each reconciliation approach

with the negative reconciled forecasts. Table 10 represents the percentage differences in average RMSEs between the reconciled forecasts with non-negatives and negatives at each level in the grouped structure.

It can be seen that the non-negative reconciled forecasts from the OLS approach showed slight gains over the negative reconciled forecasts at each forecast horizon considered (with rare exceptions). These gains are most pronounced at the disaggregated levels. In addition, WLS_v and MinT also showed gains at the most disaggregated level, though the consideration of aggregated levels sometimes introduced slight losses. As these gains or losses are negligible, it can be argued that the non-negativity constraint does not significantly affect the performances of the forecast reconciliation approaches. However, we should emphasize that it is useful in real applications for making meaningful managerial decisions and policy implementations.

Table 8: Summary statistics of the negative reconciled forecasts.

	ARIMA					
	OLS		WLS _v		MinT	
$h = 1$	96		39		61	
	5	80	1	4	1	8
	-71.8	-2.33	-17.5	-0.07	-22.8	-0.05
	-1.52	-2e-3	-13.2	-0.06	-10.9	-4e-3
2	95		37		64	
	5	91	1	3	1	9
	-72.3	-2.6	-12.9	-0.09	-23.1	-0.08
	-3.98	-1e-3	-10.7	-0.09	-18.3	-0.01
3	94		37		62	
	5	90	1	2	1	8
	-67.3	-1.31	-16.3	-0.04	-14.5	-0.02
	-2.36	-3e-4	-16.3	-0.04	-14.5	-0.02
6	91		40		56	
	4	87	1	3	1	9
	-87.4	-0.62	-12.3	-0.11	-16.5	-0.03
	-1.16	-4e-4	-9.91	-0.11	-10.49	-0.03
12	85		44		59	
	5	83	1	2	1	8
	-79.5	-2.30	-19.2	-0.18	-26.4	-0.07
	-3.66	-5e-4	-19.2	-0.18	-15.6	-5e-3
	ETS					
	OLS		WLS _v		MinT	
$h = 1$	93		45		43	
	1	86	1	3	1	3
	-41.0	-0.03	-13.2	-0.02	-14.9	-0.01
	-6.16	-4e-3	-12.7	-0.02	-9.78	-0.01
2	93		43		45	
	1	63	1	3	1	3
	-38.4	-0.19	-13.2	-0.11	-16.9	-0.10
	-7.00	-1e-3	-12.0	-0.07	-8.80	-0.09
3	91		46		46	
	1	66	1	3	1	3
	-34.0	-0.12	-11.7	-0.08	-13.4	-0.02
	-4.58	-2e-3	-11.7	-0.08	-8.80	-3e-3
6	90		40		43	
	1	75	1	3	1	3
	-31.8	-0.15	-15.2	-3e-4	-13.4	-0.17
	-4.40	-1e-3	-15.2	-3e-4	-8.12	-1e-3
12	84		37		39	
	1	117	1	3	1	4
	-170	-0.15	-15.2	-0.07	-17.4	-0.10
	-3.44	-3e-3	-14.1	-0.04	-10.5	-0.05

Notes: Each cell gives (i) the number of iterations reported with negative reconciled forecasts; and the minimum and maximum of (ii) the number of negative reconciled forecasts; (iii) the largest negative value (in thousands); and (iv) the smallest negative value (in thousands).

Table 9: Computational times (in seconds) of obtaining the non-negative reconciled forecasts.

		$h = 1$	2	3	6	12	$h = 1$	2	3	6	12
		ARIMA					ETS				
OLS	ABB	971.1*	953.7*	943.8*	912.4*	858.2*	947.5*	918.4*	897.5*	885.7*	828.6*
	PCG	27.6	27.5	27.6	27.0	24.2	23.0	22.8	21.8	21.4	21.1
	BPV	3.0	2.8	2.9	2.8	2.6	2.3	2.3	bm2.2	2.1	2.0
WLS _v	ABB	2.0	1.9	1.9	1.9	1.8	2.1	2.0	2.0	1.9	1.8
	PCG	5.2	5.0	4.8	5.0	5.5	5.7	5.5	5.6	5.0	4.8
	BPV	2.2	2.1	2.1	2.0	2.0	2.2	2.1	2.2	2.0	1.9
MinT	ABB	54.4	55.2	55.2	50.5	48.7	46.8	47.1	46.8	44.7	41.4
	PCG	50.1	49.9	49.1	46.5	44.7	45.1	44.9	44.6	42.9	39.8
	BPV	38.4	37.8	37.5	36.2	33.7	38.2	37.7	37.3	36.0	33.6

Notes: The bold entries identify the non-negative algorithms with the best computational performances.

Table 10: *Impact of the non-negativity constraints on forecast performances.*

	$h = 1$	2	3	6	12	$h = 1$	2	3	6	12
ARIMA										
Australia						Australia by purpose of travel				
OLS	0.0081	-0.0138	-0.0230	-0.0094	-0.0269	-0.0757	-0.0503	0.0125	-0.0065	-0.0157
WLS _v	0.0091	0.0056	0.0100	0.0006	0.0018	0.0009	0.0027	0.0027	0.0031	0.0008
MinT	-0.0090	0.0267	0.0290	0.0427	-0.0222	0.0295	0.0143	0.0514	0.0597	0.0279
States						States by purpose of travel				
OLS	-0.0560	-0.0744	-0.0543	-0.0159	-0.0904	-0.5054	-0.8173	-0.7670	-0.8597	-0.7745
WLS _v	0.0045	0.0016	0.0004	-0.0077	0.0003	-0.0009	-0.0004	-0.0013	-0.0023	-0.0037
MinT	0.0366	0.0007	0.0123	0.0140	0.0074	0.0160	0.0120	0.0135	0.0255	-0.0164
Zones						Zones by purpose of travel				
OLS	-0.1115	-0.1038	-0.1090	-0.1141	-0.1489	-0.4320	-0.6144	-0.5723	-0.7088	-0.6646
WLS _v	0.0015	-0.0009	0.0011	0.0019	-0.0018	-0.0004	0.0005	0.0001	0.0001	-0.0071
MinT	0.0028	-0.0037	0.0031	0.0019	-0.0161	0.0004	-0.0044	0.0032	0.0072	-0.0234
Regions						Regions by purpose of travel				
OLS	-0.1474	-0.1246	-0.1164	-0.1263	-0.1480	-0.5573	-0.6581	-0.5831	-0.7258	-0.7480
WLS _v	0.0014	0.0017	0.0008	0.0048	0.0114	-0.0052	-0.0049	-0.0044	-0.0057	-0.0150
MinT	-0.0002	-0.0056	0.0002	-0.0003	-0.0012	-0.0134	-0.0189	-0.0104	-0.0097	-0.0265
ETS										
Australia						Australia by purpose of travel				
OLS	-0.0236	-0.0021	-0.0028	-0.0116	-0.0468	-0.0066	-0.0088	-0.0060	-0.0103	-0.0222
WLS _v	0.0052	0.0040	0.0044	0.0059	-0.0002	-0.0018	-0.0026	-0.0023	-0.0043	-0.0026
MinT	0.0089	0.0193	0.0207	0.0203	0.0048	-0.0012	-0.0118	0.0020	-0.0105	-0.0127
States						States by purpose of travel				
OLS	-0.0333	-0.0314	-0.0207	-0.0510	-0.2106	-0.0636	-0.0423	-0.0455	-0.0593	-0.1031
WLS _v	-0.0017	-0.0016	0.0009	0.0033	-0.0009	0.0026	0.0022	0.0050	0.0040	0.0011
MinT	0.0129	0.0090	0.0144	0.0165	0.0031	0.0101	0.0060	0.0095	0.0056	-0.0001
Zones						Zones by purpose of travel				
OLS	-0.0274	-0.0213	-0.0226	-0.0350	-0.1622	-0.0752	-0.0632	-0.0703	-0.0543	-0.0962
WLS _v	-0.0084	-0.0095	-0.0120	-0.0133	-0.0099	0.0038	0.0033	0.0028	0.0033	0.0034
MinT	-0.0008	-0.0036	-0.0057	-0.0071	-0.0093	0.0027	-0.0018	0.0004	0.0012	0.0014
Regions						Regions by purpose of travel				
OLS	-0.0262	-0.0239	-0.0297	-0.0288	-0.1026	-0.1249	-0.0940	-0.0897	-0.0912	-0.1295
WLS _v	0.0081	0.0060	0.0040	0.0076	0.0085	-0.0148	-0.0131	-0.0112	-0.0119	-0.0140
MinT	0.0121	0.0102	0.0083	0.0110	0.0135	-0.0149	-0.0148	-0.0131	-0.0138	-0.0152

Notes: Each entry shows the percentage difference in average RMSEs between reconciled forecasts with non-negatives and negatives. A negative (positive) entry shows a percentage decrease (increase) in average RMSEs relative to the negative reconciled forecasts. Bold entries identify improvements due to the non-negativity constraints.

6 Conclusions and Discussion

References

- Andrew, JJ & Hancewicz, TM (1998). Rapid analysis of Raman image data using two-way multivariate curve resolution. *Applied Spectroscopy* **52**(6), 797–807.
- Barzilai, J & Borwein, JM (1988). Two-point step size gradient methods. *IMA Journal of Numerical Analysis* **8**(1), 141–148.
- Bertero, M, Boccacci, P, Prato, M & Zanni, L (2013). Scaled gradient projection methods for astronomical imaging. *EAS Publications Series* **59**, 325–356.
- Birgin, EG, Martínez, JM & Raydan, M (2003). Inexact spectral projected gradient methods on convex sets. *IMA Journal of Numerical Analysis* **23**(4), 539–559.
- Bonettini, S, Zanella, R & Zanni, L (2009). A scaled gradient projection method for constrained image deblurring. *Inverse Problems* **25**(1).
- Cantarella, J & Piatek, M (2004). Tsnrls: A solver for large sparse least squares problems with non-negative variables. arXiv: [cs/0408029v1](https://arxiv.org/abs/cs/0408029v1).
- Chen, D & Plemmons, RJ (2009). “Nonnegativity constraints in numerical analysis”. In: *The birth of numerical analysis*. Ed. by A Bultheel & R Cools. New Jersey, NJ: World Scientific Publishing, pp.109–140.
- Figueiredo, MAT, Nowak, RD & Wright, SJ (2007). Gradient projection for sparse reconstruction: Application to compressed sensing and other inverse problems. *IEEE Journal of Selected Topics in Signal Processing* **1**(4), 586–597.
- Gemperline, PJ & Cash, E (2003). Advantages of soft versus hard constraints in self-modeling curve resolution problems. Alternating least squares with penalty functions. *Analytical Chemistry* **75**(16), 4236–4243.
- Hyndman, RJ (2019). *forecast: Forecasting functions for time series and linear models*. R package version 8.5. <https://pkg.robjhyndman/forecast>.
- Hyndman, RJ, Ahmed, RA, Athanasopoulos, G & Shang, HL (2011). Optimal combination forecasts for hierarchical time series. *Computational Statistics and Data Analysis* **55**, 2579–2589.
- Hyndman, RJ & Khandakar, Y (2008). Automatic Time Series Forecasting : The forecast Package for R. *Journal of Statistical Software* **27**(3).
- Hyndman, RJ, Lee, AJ & Wang, E (2016). Fast computation of reconciled forecasts for hierarchical and grouped time series. *Computational Statistics and Data Analysis* **97**, 16–32.

- Júdice, JJ & Pires, FM (1989). Bard-type methods for the linear complementarity problem with symmetric positive definite matrices. *IMA Journal of Management Mathematics* **2**(1), 51–68.
- Júdice, JJ & Pires, FM (1994). A block principal pivoting algorithm for large-scale strictly monotone linear complementarity problems. *Computers and Operations Research* **21**(5), 587–596.
- Kim, J & Park, H (2011). Fast nonnegative matrix factorization: An active-set-like method and comparisons. *SIAM Journal on Scientific Computing* **33**(6), 3261–3281.
- Kostreva, MM (1978). Block pivot methods for solving the complementarity problem. *Linear Algebra and its Applications* **21**, 207–215.
- Lawson, CL & Hanson, RJ (1974). *Solving least squares problems*. New Jersey, NJ: Prentice-Hall.
- Murty, KG (1974). Note on a Bard-type scheme for solving the complementarity problem. *Opsearch* **11**(2-3), 123–130.
- Nocedal, J & Wright, SJ (2006). *Numerical optimization*. Ed. by TV Mikosch, SI Resnick & SM Robinson. 2nd ed. New York, NY: Springer Science and Business Media.
- Revolution Analytics & Weston, S (2015a). *doParallel: Foreach Parallel Adaptor for the 'parallel' Package*. R package version 1.0.10. <https://CRAN.R-project.org/package=doParallel>.
- Revolution Analytics & Weston, S (2015b). *foreach: Provides Foreach Looping Construct for R*. R package version 1.4.3. <https://CRAN.R-project.org/package=foreach>.
- Turlach, BA & Weingessel, A (2013). *quadprog: Functions to solve Quadratic Programming Problems*. R package version 1.5-5. <https://CRAN.R-project.org/package=quadprog>.
- Turlach, BA & Wright, SJ (2015). Quadratic programming. *Wiley Interdisciplinary Reviews: Computational Statistics* **7**(2), 153–159.
- Wang, M & Wang, X (2012). “A jump-start of non-negative least squares solvers”. In: *High-performance scientific computing: Algorithms and applications*. Ed. by MW Berry, KA Gallivan, E Gallopoulos, A Grama, B Philippe, F Saied & Y Saad. London, England: Springer-Verlag, pp.295–310.
- Wickramasuriya, SL, Athanasopoulos, G & Hyndman, RJ (2019). Optimal forecast reconciliation of hierarchical and grouped time series through trace minimization. *Journal of the American Statistical Association*. To appear.