MONASH University

MONASH BUSINESS SCHOOL

ISSN 1440-771X

Department of Econometrics and Business Statistics

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

Reconciliation of structured time series forecasts with graphs

UNPUBLISHED DRAFT

Mitchell O'Hara-Wild, Rob Hyndman,
George Athanasopolous

January 2024

Working Paper

Reconciliation of structured time series forecasts with graphs

Mitchell O'Hara-Wild
Department of Econometrics & Business Statistics,
Monash University,
Clayton VIC 3800, Australia.
Email: mitch.ohara-wild@monash.edu
Corresponding author

Rob Hyndman
Department of Econometrics & Business Statistics,
Monash University,
Clayton VIC 3800, Australia.
Email: rob.hyndman@monash.edu

George Athanasopolous
Department of Econometrics & Business Statistics,
Monash University,
Clayton VIC 3800, Australia.
Email: george.athanasopoulos@monash.edu

31 January 2024

Reconciliation of structured time series forecasts with graphs

Abstract

Forecast reconciliation methods play a crucial role in ensuring coherence within large collections of structurally related time series data. Hierarchical constraints have a single aggregation pathway from the bottom time series to the top (most aggregated) series, while grouped constraints can have multiple aggregation pathways in the coherency structure. Both hierarchical and grouped constraints share the same top and bottom time series, regardless of the intermediate aggregation pathway. These coherency constraints are often represented with matrices and visualised with graphs, where each series is represented by a node and edges connect series to their disaggregated child series (a polytree). Grouped constraints are often visualised with multiple disjoint polytrees, where each polytree shows a different disaggregation pathway to the bottom time series. In this paper, we propose a novel approach using directed acyclical graphs (DAGs) to both visualise the constraints, and to facilitate forecast reconciliation computation. Relaxing the requirement of having the same top and bottom time series results in more flexible reconciliation structures than those possible in hierarchical and grouped constraints. Graph structures can represent partial reconciliation via disjoint graphs, remove redundant aggregation with unbalanced trees, and allow sparse aggregation constraints from different levels of disaggregated series. Utilising a graph structure to describe the coherency of a time series also enables improved interfaces for analysing specific areas of a hierarchy.

Keywords: forecast, reconciliation, probabilistic, coherent, graph

1 Introduction

Accurate forecasts of large collections of time series are critically important to decision makers for the efficient operation of an organisation. These collections of time series are often intrinsically structured for aggregation. Forecasting the most aggregated series in the structure is useful for organisational strategy and planning, while the disaggregated forecasts are important for managing local operations. Each time series possesses various attributes that identify their relation to other series in the collection. These attributes typically relate to what is being measured, such as product categories or store locations for the sales of a product over time. Forecasts of each series from independent models will typically not align with the aggregation structure of the data, and this inconsistency presents an inherit forecast error. Correcting for this structural error presents an opportunity to leverage additional information from other series and produce coherent forecasts.

The process of adjusting forecasts to satisfy these aggregation constraints was first introduced by Hyndman et al. (2011). This technique of forecast reconciliation has since been extended to include temporal aggregation (Athanasopoulos et al. 2017), cross-temporal aggregation (Kourentzes & Athanasopoulos 2019), and improved minimum trace based reconciliation weights (Wickramasuriya, Athanasopoulos & Hyndman 2019). Girolimetto & Di Fonzo (2023b) generalise these aggregation constraints beyond interactions of hierarchical, grouped and temporal to include any linear relationship between series. This generalisation allows forecast reconciliation to be applied on collections of time series which don't follow the typical 'upper' and 'bottom' classification of series. One such application of the generalised constraints is forecasting Australian GDP, wherein GDP can be calculated using multiple approaches (income, expenditure and production) that aren't simply aggregation (Athanasopoulos et al. 2020). This results in a constraint structure that shares a common 'upper' series, but lacks a common set of 'bottom' series since each approach to calculating GDP is distinct.

A complication of not having 'upper' and 'bottom' series classifications is that the usual "aggregation matrix" (Hyndman & Di Fonzo 2022) of the structural representation for constraints cannot be directly produced. Instead, Girolimetto & Di Fonzo (2023b) use the zero-constrained representation to describe the constraints between 'constrained' and 'unconstrained' time series, which can in turn be used to produce a "structural-like" representation of the constraints. The conversion from zero-constrained to "structural-like" representations can be difficult to solve manually, especially without including redundant constraints. Girolimetto & Di Fonzo (2023b) apply standard linear algebra techniques to remove redundant constraints, and then obtain the "linear combination" matrix (the general analogue of "aggregation matrix") for structural representation.

In extension to this work, I propose an alternative graph-based representation for coherency constraints on a collection of time series. Using directed acyclical graphs to rather than structural or zero-constrained coefficient matrices presents several key advantages. Representing constraints with graphs simplifies their construction and enables direct visualisation of the relationship between series via graph visualisation. Using graphs to describe the structure of large collections of related time series also enables improved manipulation tools to remove irrelevant or otherwise unwanted sections of data without disrupting the coherency constraints. Using graphs which constrain the parent nodes to be linear combinations of child nodes is equivalent general linear constraint matrices, however further extensions could allow for non-linear coherency constraints.

In this paper I provide an overview of cross-sectional coherency constraints including hierarchical, grouped and general linear constraints in section 2. In section 3 I introduce an alternative approach for representing these constraints using directed acyclical graphs, which offers several advantages over linear constraint matrices. An application of using graph constraints for cross-validation is shown in section 4, where I show how graph constraints can be used to replicate the results of general linear constraints. A second application is shown in 5, wherein the forecasts of Australian GDP calculated by both income and expenditure approaches is reconciled using constraints represented in a graph. The conclusion in section 6 provides a summary of graph constrained forecast reconciliation, how it relates to general linear constraints, and how forecast reconciliation can be further extended by leveraging existing knowledge from graph theory.

2 Coherency constraints

Forecast reconciliation is the process of adjusting forecasts to satisfy coherency constraints. These constraints are typically additive, in which a set of "bottom" series are aggregated to produce "upper" series of interest. There are many ways in which a set of bottom series can combine to produce upper series, however most often large collections of time series are disaggregated by their attributes in a hierarchical or grouped fashion (Hyndman, Lee & Wang 2016). Two approaches are commonly used to represent these structural constraints.

The structural representation (Hyndman et al. 2011; Athanasopoulos, Ahmed & Hyndman 2009) imposes linear constraints between the "upper" and "bottom" level series using a $n \times n_b$ "summing" or "structural" matrix $\boldsymbol{S}$ of the form

$$\boldsymbol{S} = \begin{bmatrix} \boldsymbol{A} \\ \boldsymbol{I}_{n_b} \end{bmatrix}.$$

$\boldsymbol{S}$ is comprised of a $n_a \times n_b$ matrix $\boldsymbol{A}$ is that encodes how the n_b bottom series ($\boldsymbol{b}_t$) are linearly combined to produce the n_a upper time series, such that $\boldsymbol{y}_t = \boldsymbol{S}\boldsymbol{b}_t$ for the set of all series $\boldsymbol{y}_t$.

The zero-constrained representation (Di Fonzo & Girolimetto 2023) of the linear constraints uses the equivalent expression $\boldsymbol{C}\boldsymbol{y}_t = \boldsymbol{0}_{n_a}$, where the full rank $n_a \times n$ "constraint" matrix $\boldsymbol{C}$ is

$$\boldsymbol{C} = \begin{bmatrix} \boldsymbol{I}_{n_a} & -\boldsymbol{A} \end{bmatrix}.$$

Both structural and zero-constrained representations produce the same coherent forecasts after reconciliation, however the zero-constrained representation is more efficient (when $n_a < b_b$) since the solution involves inverting a smaller matrix.

2.1 Hierarchical constraints

Hierarchical constraints disaggregate a collection of time series by its attributes in sequence, such that there is a single path from the most aggregated series to the most disaggregated series. An example of a simple hierarchical structure is shown in figure 1, with the corresponding structural representation in equation 1 and zero-constrained coefficient matrix in equation 2.

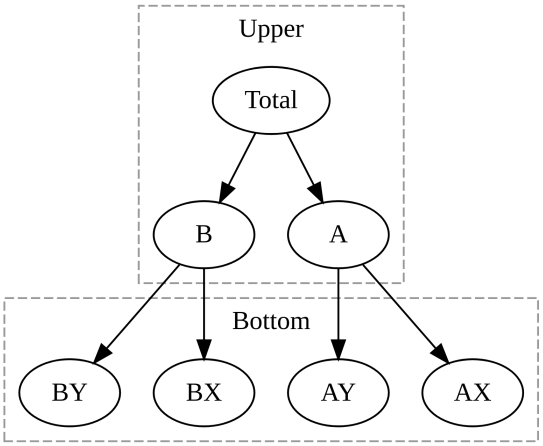


Figure 1: *A three level hierarchy of two attributes each with two values.*

$$\boldsymbol{S} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ \hdashline 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{A} \\ \boldsymbol{I}_4 \end{bmatrix}. \tag{1}$$

$$\boldsymbol{C} = \left[\begin{array}{ccc:cccc} 1 & 0 & 0 & -1 & -1 & -1 & -1 \\ 0 & 1 & 0 & -1 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & -1 & -1 \end{array}\right] = \begin{bmatrix} \boldsymbol{I}_3 & -\boldsymbol{A} \end{bmatrix}. \tag{2}$$

2.2 Grouped constraints

Grouped constraints follow a similar disaggregation by attributes as hierarchical structures, however the order in which the attributes disaggregate the top level can vary. This results in multiple pathways from the top to bottom series, and for larger collections of time series the number of constraints can grow quickly. Visually this is commonly shown using multiple disjoint graphs which present the pathways from top to bottom nodes. Figure 2 shows an example of the usual visualisation of grouped constraints, which is improved upon in 5. Equation 3 shows the example's structural representation and the equivalent zero-constrained coefficient matrix is in equation 4.

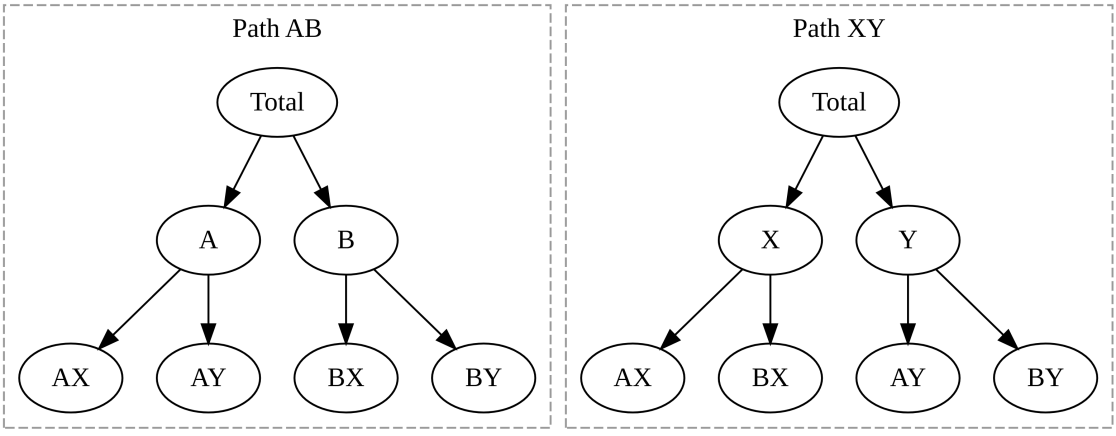


Figure 2: *A three level grouped structure of two attributes each with two values.*

$$\boldsymbol{S} = \left[\begin{array}{cccc} 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ \hdashline 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{array}\right] = \begin{bmatrix} \boldsymbol{A} \\ \boldsymbol{I}_4 \end{bmatrix}. \tag{3}$$

$$\boldsymbol{C} = \left[\begin{array}{ccccc:cccc} 1 & 0 & 0 & 0 & 0 & -1 & -1 & -1 & -1 \\ 0 & 1 & 0 & 0 & 0 & -1 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & -1 & -1 \\ 0 & 0 & 0 & 1 & 0 & -1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & -1 & 0 & -1 \end{array}\right] = \begin{bmatrix} \boldsymbol{I}_5 & -\boldsymbol{A} \end{bmatrix}. \tag{4}$$

A special case of grouped constraints is found in temporal reconciliation (Athanasopoulos et al. 2017), where aggregates of varying temporal granularity are obtained from a frequently observed bottom time series. The combination of temporal and cross-sectional (hierarchical and grouped constraints) reconciliation is known as cross-temporal reconciliation (Kourentzes & Athanasopoulos 2019; Di Fonzo & Girolimetto 2023), which also results in a set of grouped constraints.

2.3 General linear constraints

In both hierarchical and grouped structures there exists a single top level series and a common set of bottom series. This is suitable for most datasets where series are disaggregated by their attributes, but time series don't necessarily need to be constrained by common attributes. Girolimetto & Di Fonzo (2023b) introduce the idea of general linear constraints, which allow for any linear constraint(s) to be imposed on the structure. This generalisation allows for one or more top level series in a structure, which can have multiple disaggregation pathways toward one or more sets of bottom series. A graphical representation of general linear constraints is shown in figure 3, wherein a common total series is disaggregated by two distinct pathways toward different sets of bottom series.

Without a clear distinction between "upper" and "bottom" series, the structural representation cannot be directly produced. Using the categorisation of "constrained" and "unconstrained" (rather than "upper" and "bottom") allows the zero-constrained representation to be produced directly. Direct

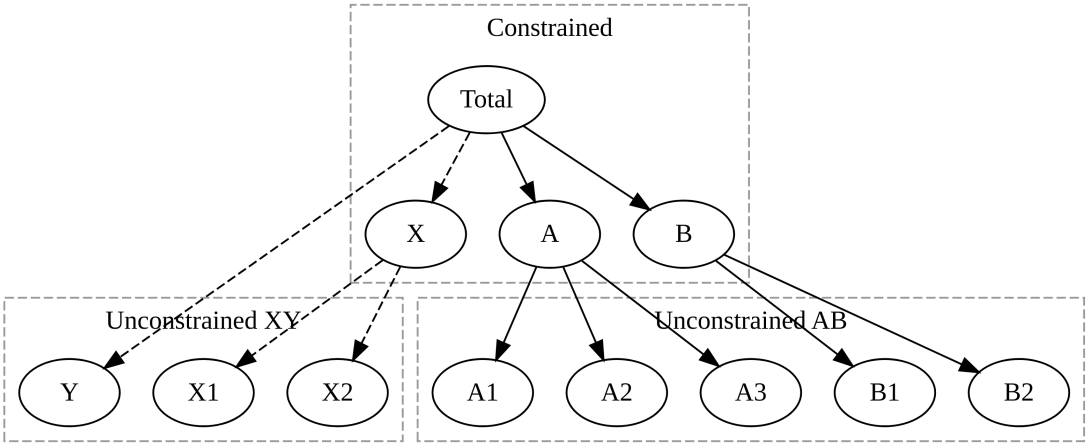


Figure 3: *A small example of a general linearly constrained structure. In this example, different sets of bottom (unconstrained) time series are used to aggregate the top level series. This example also features mixed depth disaggregation for X and Y series.*

construction of this constraint matrix isn't guaranteed to be full rank, and so we denote this general constraint coefficient matrix $\mathbf{\Gamma}$ (rather than C), where $\mathbf{\Gamma} \boldsymbol{y}_t = \mathbf{0}$. Equation 5 shows the zero-constrained representation coefficient matrix for the example in figure 3. The categorisation of "constrained" ($\mathbf{\Gamma}_c$) and "unconstrained" ($\mathbf{\Gamma}_u$) series are found on the left and right of the vertical dashed line respectively. Since there are two distinct pathways for the total series, the first two rows of the general constraint matrix $\mathbf{\Gamma}$ have the same coefficients on the "constrained" side.

$$\mathbf{\Gamma} = \left[\begin{array}{cccc:cccccccc} 1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & -1 & -1 & -1 \\ 1 & 0 & 0 & 0 & -1 & -1 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & -1 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & -1 & -1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 \end{array}\right] = \left[\begin{array}{c:c} \mathbf{\Gamma}_c & \mathbf{\Gamma}_u \end{array}\right]. \tag{5}$$

While the structural matrix $\boldsymbol{S}$ cannot be expressed from series with general constraints, a similar "structural-like" representation can be obtained (Girolimetto & Di Fonzo 2023b). If there are no redundant constraints and $\mathbf{\Gamma}_c$ is full rank, then the $\boldsymbol{A}$ matrix can be directly solved as $\boldsymbol{A} = -(\mathbf{\Gamma}_c)^{-1}\mathbf{\Gamma}_u$. Usually this approach cannot be taken (as is the case in example plotted in figure 3), since the $\mathbf{\Gamma}_c$ matrix is rarely full rank and redundant constraints are common in larger structures. In these cases it is still possible to obtain the "structural-like" representation using converting $\mathbf{\Gamma}$ into reduced row echelon form (Meyer & Stewart 2023) or decomposing it with QR decomposition and column pivoting (Lyche 2020). Both of these techniques allow the rearranged partitioning of $\mathbf{\Gamma}$ into $\boldsymbol{C} = [\boldsymbol{I}_{n_c} \quad -\boldsymbol{A}]$,

and are implemented in the FoReco R package's `lcmat()` function (Girolimetto & Di Fonzo 2023a). The equivalent "structural-like" representation matrix $\bar{\boldsymbol{S}}$ for the coherency constraints in equation 5 is

$$\bar{\boldsymbol{S}} = \left[\begin{array}{ccccccc} 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ -1 & -1 & 1 & 1 & 1 & 1 & 1 \\ \hdashline 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{array}\right] = \left[\begin{array}{c} \boldsymbol{A} \\ \boldsymbol{I}_7 \end{array}\right].$$

3 Graph reconciliation constraints

An alternative to matrix based representations of coherency constraints is to use graph data structures. Graphs are often used to visualise coherency constraints (as shown in figures 1, 2, and 3), but they can also represent hierarchical, grouped, and general linear constraints with directed acyclical graphs.

To represent coherency constraints with a graph, let the graph's set of vertices $\boldsymbol{V}$ represent the time series and the set of directed edges $\boldsymbol{E} \subseteq \left\{(x, y) | (x, y) \in \boldsymbol{V}^2, x \neq y\right\}$ represent the constraints between aggregated (y) and disaggregated (x) series. This is sufficient to represent hierarchical and grouped coherence constraints, where the upper time series are obtained from the bottom time series by simple aggregation. Figure 4 shows an equivalent graph of the structural matrix in equation 3 obtained by constructing a graph with edges between the upper series and bottom series in accordance with the aggregation matrix $\boldsymbol{A}$.

This version of the graph lacks potentially useful information about disaggregation pathway from top to bottom nodes, and so it is more informative (and efficient) to instead use a graph with edges between vertices along a disaggregation pathway. To distinguish between multiple sets of disaggregations for a time series, edges must also have an attribute i to identify its disaggregation group from the set of all paths from top to bottom $\boldsymbol{P}$, making the edges now $\boldsymbol{E} \subseteq \left\{(x, y, i) | (x, y) \in \boldsymbol{V}^2, x \neq y, i \in \boldsymbol{P}\right\}$. This identifying attribute enables any relationship between series to be represented in the graph, including

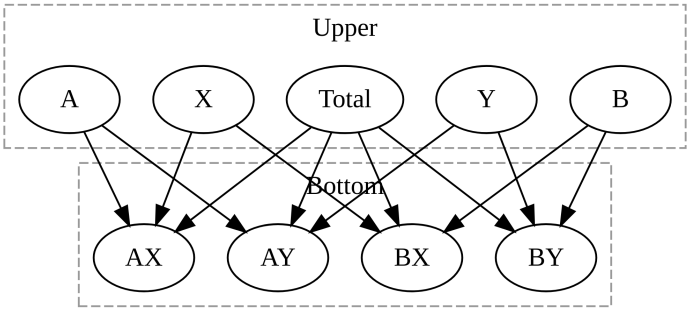


Figure 4: *The same grouped structure from equation 3 shown in a single graph.*

structures with multiple top level series, different sets of bottom series and completely disjoint structures. Figure 5 shows the graph representation of the coherence constraints with identified disaggregation groups to enable multi-layer structures.

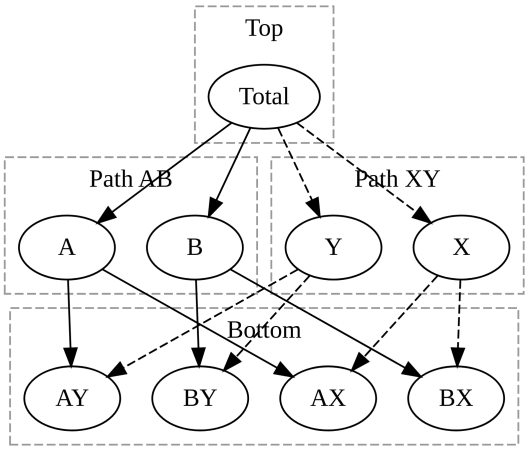


Figure 5: *The equivalent three level grouped structure from equation 3 shown in a single graph, the line type identifies edges by the disaggregation pathway.*

The graph's edges $\boldsymbol{E}$ can also be weighted to represent the linear combination of vertices for the aggregation such that $\boldsymbol{E} \subseteq \left\{(x, y, i, w) | (x, y) \in \boldsymbol{V}^2, x \neq y, i \in \boldsymbol{P}, w \in \mathbb{R}\right\}$. This results in an equivalent representation of general linear constraints, however the graph structure makes it trivial to identify constrained and unconstrained series based on the existence of child vertices. The graph's edges can be directly used to create the zero-constrained coefficient matrix, since there is a one-to-one mapping between the rows of the zero-constrained coefficient matrix $\mathbf{\Gamma}$, and each set of edges that share a common disaggregation group i and aggregated series y. For each row and set of edges, the column corresponding to the aggregated series y is 1, the column(s) for the disaggregated series $\boldsymbol{x}$ will be $-\boldsymbol{w}$ ($\boldsymbol{w} = 1$ for unweighted graphs), and all other columns are 0. Using the methods described in

section 2 a "structural-like" representation can be obtained, and optimally reconciled forecasts can be obtained using all representations (Wickramasuriya, Athanasopoulos & Hyndman 2019).

Graph coherency also enables the representation of non-linear coherency constraints. For each set of edges sharing a common group i and aggregated series y, a functional constraint f_i is used to describe the relationship between y and it's child vertices $\boldsymbol{x}$ where $y = f_i(x_1, x_2, \ldots)$. Non-linear aggregation structures have applications in forecasting business revenue (the summed product of price and sales) and in mortality modelling (Shang & Hyndman 2017; Li & Chen 2022).

Representing coherency constraints using graphs rather than matrices offers several practical advantages for working with large collections of related time series. In addition to visualising the coherency structure, having an underlying graph structure to the time series allows specific related sections of the graph to be identified using semantics from graph-theory. This vastly improves the manipulation tools available for filtering and merging time series, for example subgraphs can be extracted or removed from the collection while maintaining the coherency constraints for the resulting series. Partial reconciliation is also possible by creating a graph with multiple top-level nodes, or completely disjoint graphs can be used to perform cross-validation on reconciled forecasts.

3.1 Software implementation

Coherent datasets are typically stored with only the most disaggregated (bottom) time series, since the coherency constraints can be used to compute aggregate series from them. Constructing the aggregation structure can be difficult, as either strict column naming, column ordering or separately managed metadata is needed. The hts R package (Hyndman et al. 2021) uses a matrix of time series where either hierarchical or grouped structures are described by column ordering, column naming, or column metadata. The FoReco package (Girolimetto & Di Fonzo 2023a) directly accepts the aggregation matrix A, and provides the `Cmatrix` function to help construct this matrix by specifying most common hierarchical and grouped relationships using column metadata and a symbolic formula. Much of these limitations arise from the underlying `ts` objects (Becker, Chambers & Wilks 2018) used in these packages, which is based on vectors and matrices with limited support for additional metadata.

The coherency constraints implemented in the R package fabletools (O'Hara-Wild, Hyndman & Wang 2023) builds upon the rectangular time series data frame tsibble, which identifies multiple time series in the same dataset using combinations of key variables (Wang, Cook & Hyndman 2020). Using a rectangular data structure allows for interoperability with the tidyverse (Wickham et al. 2019), and the necessary metadata for cross-sectional aggregation structures can be embedded within these identifying key variables. Unlike the hts and FoReco R packages, this metadata is directly attached to

the time series via additional columns. This representation of time series and their interdependence is closely matches what is typically available from data sources, further reducing the effort and complexity of constructing these data structures.

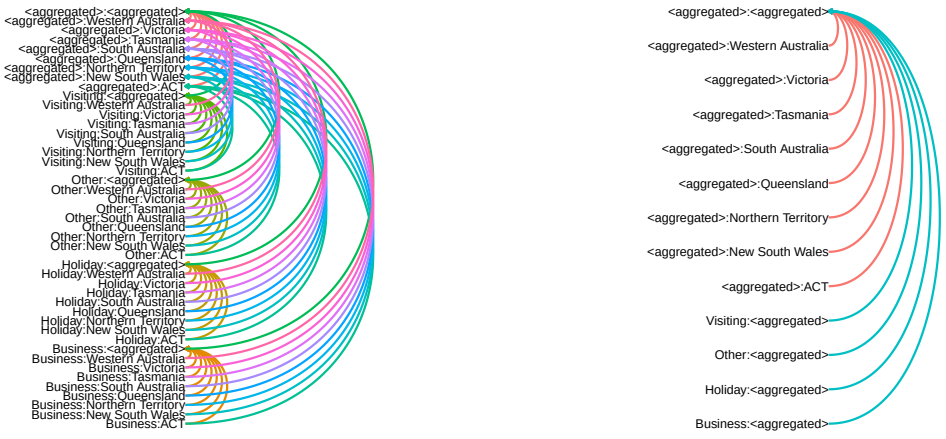
To produce cross-sectional aggregations, the `aggregate_key()` function from fabletools utilises the symbolic representation of factorial models by Wilkinson & Rogers (1973) to succinctly specify all common hierarchical and grouped aggregations from the key variables. Aggregations of the measured variable(s) are produced using user-specified functions, which can be later stored in the graph as linear and non-linear constraints. The resulting tsibble dataset with aggregates stores the coherency constraints using a graph structure embedded in the key variable(s), retaining the rectangular nature of the data.

Unlike common abstract data types for graphs which utilise lists or matrices to describe edges between vertices of a graph, the key variables of the aggregated data forms a rectangular graph using the interaction of their values. These rectangular graph abstract data type is implemented in the graphvec R package which provides `agg_vec()` and `graph_vec()` vector classes (O'Hara-Wild 2024). Simple graph structures produced with `aggregate_key()` utilise `agg_vec()` vectors, where the direction of the edge between parent and child vertices is defined using a special `<aggregated>` value. Arbitrary graphs can be constructed directly using `graph_vec()`, which stores an adjacency list to describe edges between the levels of a factor. Multiple graph vectors across columns of a data frame can be used jointly to describe a more complicated graph based on interacting the values within each graph vector. A core advantage of this graph data structure is that it can be directly used with other rectangular data manipulation tools including the tidyverse (Wickham et al. 2019). The coupling of edges (constraints) and vertices (data) within the same data frame enables processing of data to also affect the relationships in the graph. Since the values of metadata about each vertex is used in describing the graph's edges, this makes it easy to navigate, filter, and otherwise manipulate the graph with existing rectangular data tools.

4 Application: Australian tourism

To demonstrate the functionality of graph coherency constraints I will apply the techniques to the quarterly number of visitor nights in Australia. Tourism in Australia is measured by the total nights spent away from home (visitor nights), which is disaggregated by the purpose of travel and geographical destination (Athanasopoulos, Ahmed & Hyndman 2009). In this dataset, there are four purposes of travel (business, holiday, visiting friends and family, and other) to the eight states and territories of Australia that are further split into regions. From these 304 bottom series, we

compute aggregated series using two different structures. A grouped structure containing all possible aggregates is compared to a graph structure which removes the bottom level and disaggregates total visitor nights by either state/territory or purpose of travel (but not both). These grouped constraints are shown in figure 6a, and figure 6b shows second graph constraint structure. The symbolic representation for these aggregations are `Purpose * (State / Region)` and `Purpose + (State / Region)` respectively.



(a) *Grouped structure including the interaction of* `State` *and* `Purpose`

(b) *Graph structure that only disaggregate by* `State` *or* `Purpose`*, but not both.*

Figure 6: *Graph visualisation of the coherency constraints, regions within states are not shown for brevity. Coloured edges identify the disaggregation group.*

To evaluate the forecasting performance of these structures, time series cross-validation (Hyndman & Athanasopoulos 2021) is performed using a growing temporal window which starts at 13 years of data and increments by 1 year to produce 7 folds that evaluate quarterly forecast accuracy over a 1 year horizon. The cross-validation forecasts are reconciled simultaneously by representing the coherency constraints of each fold using disjoint graphs. Optimally reconciled forecasts using in-sample 1-step forecast error variance as weights from independent automatically selected exponential smoothing models (Hyndman & Khandakar 2008) are evaluated using mean absolute scaled error (MASE, Hyndman & Koehler 2006)

$$\text{MASE} = \frac{1}{4}\sum_{h=1}^{h=4}\left|\frac{y_{T+h} - \hat{y}_{T+h|T}}{\frac{1}{T-4}\sum_{t=5}^{T}|y_t - y_{t-4}|}\right|,$$

and a skill score of continuous ranked probability skill (CRPS, Gneiting & Katzfuss 2014)

$$\text{CRPS} = \frac{1}{4}\sum_{h=1}^{h=4}\int_{\mathbb{R}}(F_{T+h|T}(z) - \mathbf{1}\{y_{T+h} \leq z\})^2 dz$$

where $F_{T+h|T}$ is the h-step forecast distribution's CDF, y is the time series and $\hat{y}$ is the forecast mean.

The resulting forecast performance is summarised in table 1, which shows that partial reconciliation via the graph structure produces the most accurate forecasts.

Table 1: *Overall forecasting accuracy averaged over the common series in the structure. The graph structure produces the most accurate forecasts on Australian visitor nights on average.*

Model	Average MASE	Average skill score CRPS
Base forecasts	0.931	0.365
Graph constraints	**0.927**	**0.368**
Grouped constraints	0.938	0.334

The forecasting performance is more varied on individual series, where the MASE and CRPS for each series is shown in figures 7 and 8 respectively.

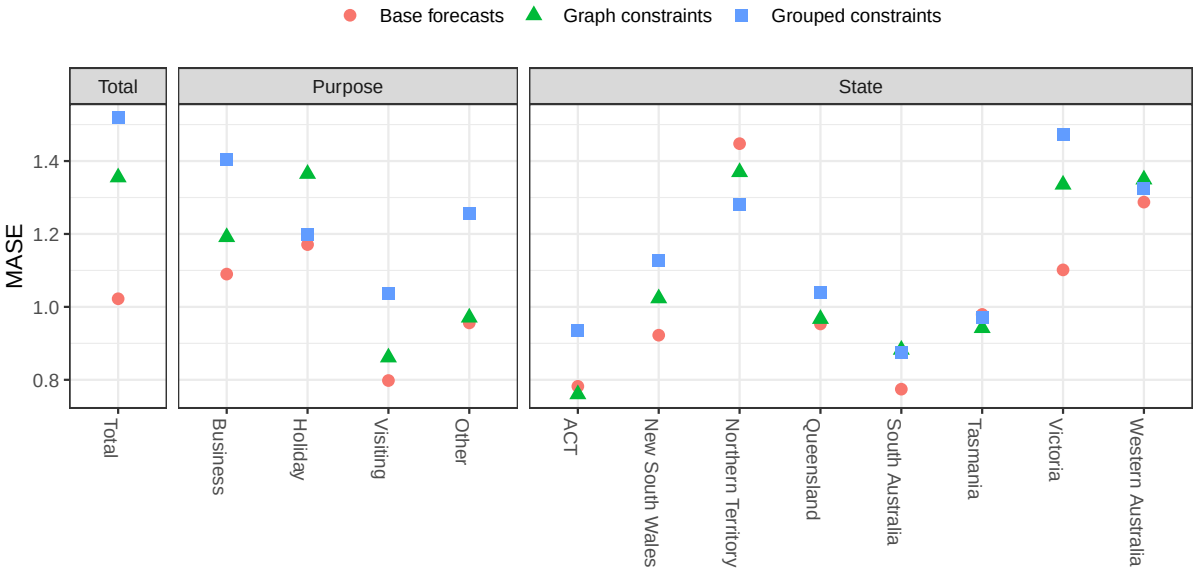


Figure 7: *The cross-validated MASE for each series and set of coherency constraints.*

The reduced structural form made possible by graph reconciliation produces marginally improved forecasting performance to other methods while requiring many less forecasts. This is particularly useful for larger collections of time series, where there are many attributes which when used together create very sparse and difficult to forecast time series. By removing these highly disaggregated series from the reconciliation structure we are able to include series disaggregated by other attributes of interest simultaneously.

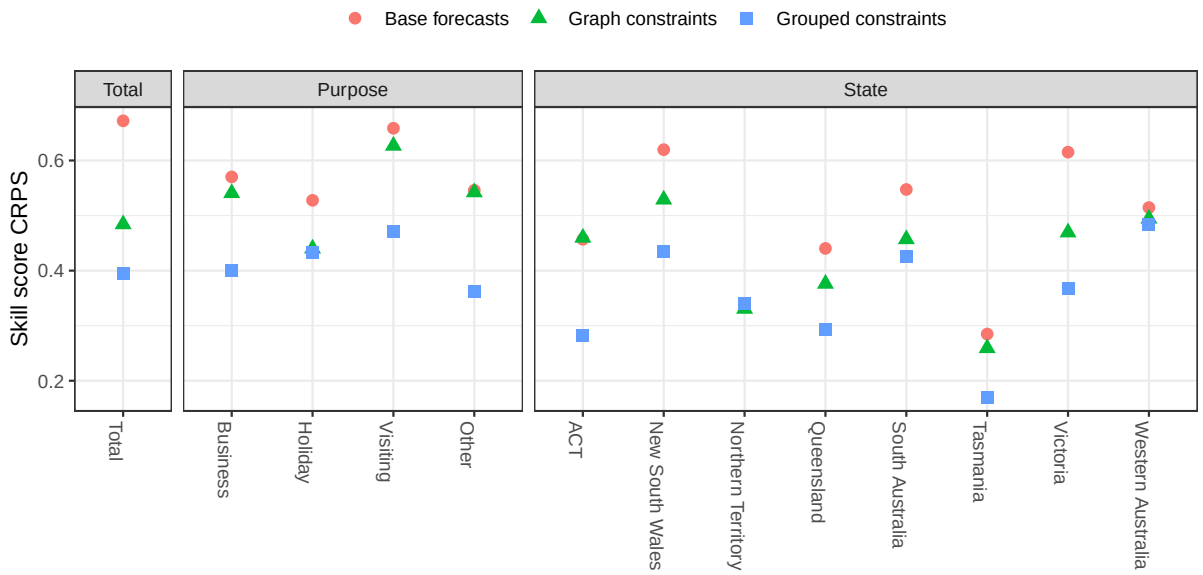


Figure 8: *The cross-validated skill score CRPS for each series and set of coherency constraints.*

5 Application: Australian GDP

The Australian Gross Domestic Product (GDP) is estimated using three approaches based on various measures with time series known as the Australian National Accounts. Measures within each approach are structurally related to each other, and ultimately aggregate to an estimate of GDP. Athanasopoulos et al. (2020) first produced coherent forecasts for GDP by separately utilising the structural information from income and expenditure approaches. Girolimetto & Di Fonzo (2023b) then reconciled forecasts of GDP using both income and expenditure methods simultaneously with general linear coherency constraints. In this section I show how graph representations of aggregation structure can also be used to describe the coherency constraints for Australian GDP.

Unlike the tourism data shown in section 4, this collection of time series is not structured with shared disaggregating variables. Rather each time series relates to each other much more generally, necessitating an arbitrary graph to be constructed for these constraints. The resulting graph that represents these constraints is shown in figure 9.

For each time series in the collection we estimate univariate ARIMA models with automatically selected hyperparameters (Hyndman & Khandakar 2008), and produce forecasts over the next four quarters. These forecasts are then reconciled in accordance with the coherency constraints imposed by the graph using weighted least squares (Hyndman et al. 2011). The accuracy of the forecasts is assessed using time series cross-validation, with forecasts originating from Q1 2000 with each fold incrementing by 1 year for a total of 23 folds.

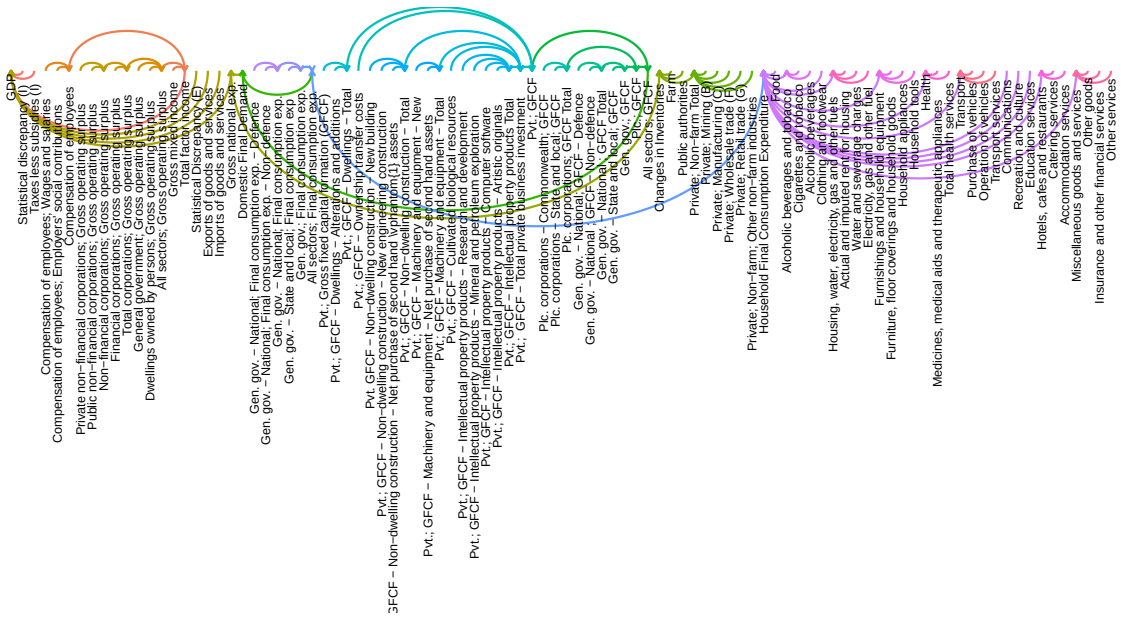


Figure 9: *The graph constraints imposed by calculating GDP using income and expenditure approaches.*

The overall accuracy of the base and reconciled forecasts are summarised in table 2, which shows that both forecasts perform similarly.

Table 2: *Overall forecasting accuracy averaged over the all series in the structure. The accuracy of the forecasts are comparable.*

Model	Average MASE	Average skill score CRPS
Base forecasts	1.39	**0.803**
Coherent forecasts	**1.385**	0.797

A more detailed breakdown of forecasting performance by MASE and CRPS for more aggregated series are shown in figures 10 and 11 respectively.

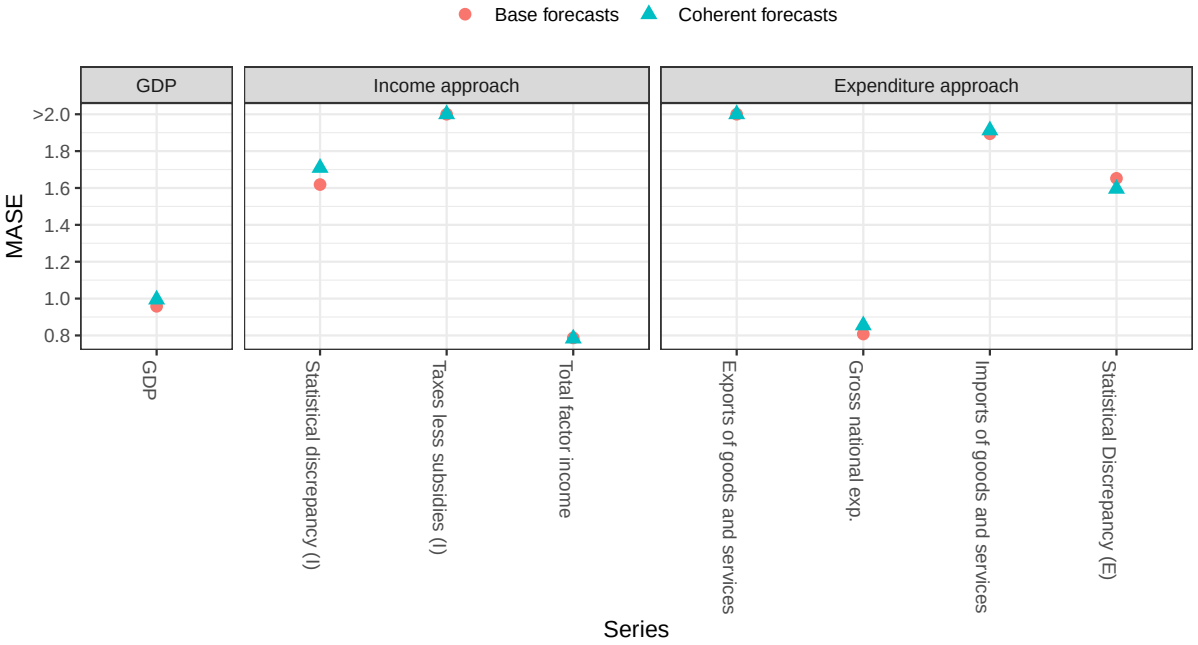


Figure 10: *The cross-validated MASE for each series and set of coherency constraints.*

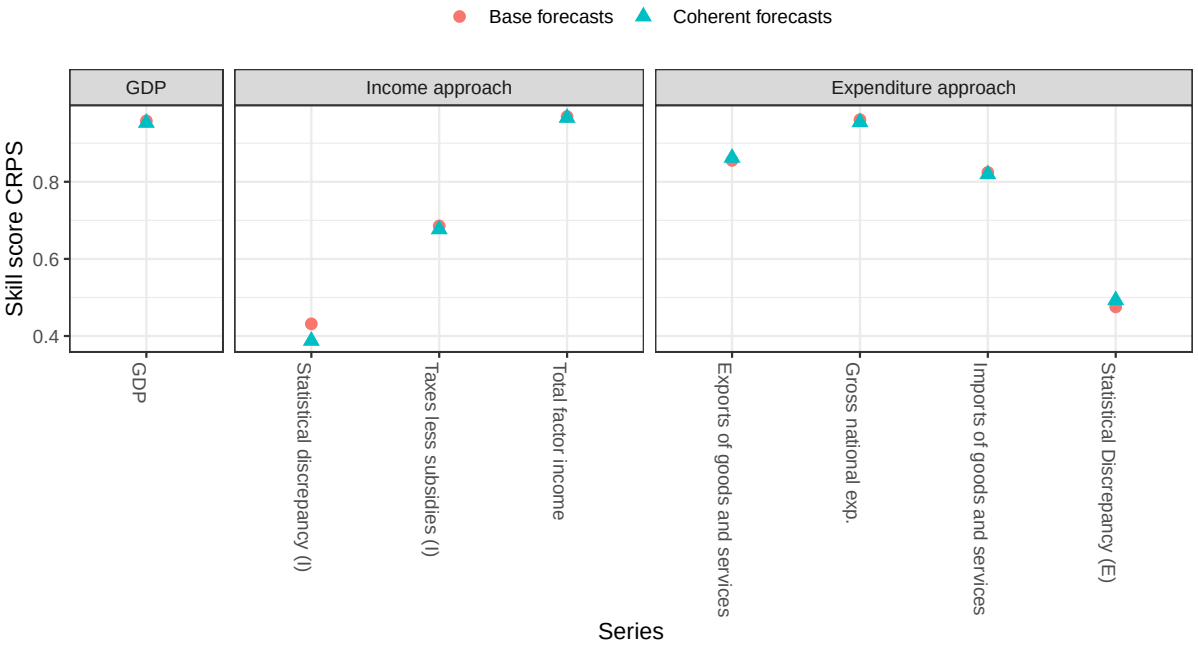


Figure 11: *The cross-validated skill score CRPS for each series and set of coherency constraints.*

6 Discussion

Forecast reconciliation continues to be an essential tool in producing reasonable and accurate forecasts for large collections of related time series. The concepts of general linear constraints and graph constraints overcome the shortcomings of hierarchical and grouped structures for describing more complicated aggregation constraints. These extensions of the literature enable partial reconciliation of incoherent forecasts, cross-validation within reconciliation, and more generally describe structural relationships between time series.

Representing coherency constraints using directed acyclical graphs offers many opportunities for further research. Most innovative is the ability to represent non-linear constraints, which in combination with numerical optimisation and graph theory algorithms could produce coherent forecasts. The graph structure also simplifies the process of removing unwanted subgraphs of time series from the structure, which can improve the overall forecasting performance. Automated pruning of noisy time series from the graph based on time series features presents an interesting opportunity to improve forecasting performance at scale.

References

Athanasopoulos, G, RA Ahmed & RJ Hyndman (2009). Hierarchical forecasts for Australian domestic tourism. *International Journal of Forecasting* **25**(1), 146–166. https://www.sciencedirect.com/science/article/pii/S0169207008000691.

Athanasopoulos, G, P Gamakumara, A Panagiotelis, RJ Hyndman & M Affan (2020). "Hierarchical Forecasting". In: *Macroeconomic Forecasting in the Era of Big Data: Theory and Practice*. Ed. by P Fuleky. Cham: Springer International Publishing, pp. 689–719. https://doi.org/10.1007/978-3-030-31150-6_21.

Athanasopoulos, G, RJ Hyndman, N Kourentzes & F Petropoulos (2017). Forecasting with temporal hierarchies. *European Journal of Operational Research* **262**(1), 60–74. https://www.sciencedirect.com/science/article/pii/S0377221717301911.

Becker, RA, JM Chambers & AR Wilks (2018). *The New S Language: A Programming Environment for Data Analysis and Graphics*. Chapman and Hall/CRC. http://dx.doi.org/10.1201/9781351074988.

Di Fonzo, T & D Girolimetto (2023). Cross-temporal forecast reconciliation: Optimal combination method and heuristic alternatives. *International Journal of Forecasting* **39**(1), 39–57. https://www.sciencedirect.com/science/article/pii/S0169207021001266.

Girolimetto, D & T Di Fonzo (2023a). *FoReco: Point Forecast Reconciliation*. R package version 2.6. https://cran.r-project.org/package=FoReco.

Girolimetto, D & T Di Fonzo (2023b). Point and probabilistic forecast reconciliation for general linearly constrained multiple time series. *Statistical Methods & Applications*. https://doi.org/10.1007/s10260-023-00738-6.

Gneiting, T & M Katzfuss (2014). Probabilistic Forecasting. *Annual Review of Statistics and Its Application* **1**(1), 125–151.

Hyndman, R & T Di Fonzo (2022). *Notation for forecast reconciliation*. https://robjhyndman.com/hyndsight/reconciliation-notation.html.

Hyndman, R, A Lee, E Wang & S Wickramasuriya (2021). *hts: Hierarchical and Grouped Time Series*. R package version 6.0.2. https://CRAN.R-project.org/package=hts.

Hyndman, RJ & G Athanasopoulos (2021). *Forecasting: principles and practice, 3rd edition*. OTexts. https://otexts.com/fpp3/.

Hyndman, RJ, RA Ahmed, G Athanasopoulos & HL Shang (2011). Optimal Combination Forecasts for Hierarchical Time Series. *Comput. Stat. Data Anal.* **55**(9), 2579–2589. https://doi.org/10.1016/j.csda.2011.03.006.

Hyndman, RJ & Y Khandakar (2008). Automatic Time Series Forecasting: The forecast Package for R. *Journal of Statistical Software* **27**(3), 1–22. https://www.jstatsoft.org/index.php/jss/article/view/v027i03.

Hyndman, RJ & AB Koehler (2006). Another look at measures of forecast accuracy. *International Journal of Forecasting* **22**(4), 679–688. https://www.sciencedirect.com/science/article/pii/S0169207006000239.

Hyndman, RJ, AJ Lee & E Wang (2016). Fast computation of reconciled forecasts for hierarchical and grouped time series. *Computational Statistics & Data Analysis* **97**, 16–32. https://www.sciencedirect.com/science/article/pii/S016794731500290X.

Kourentzes, N & G Athanasopoulos (2019). Cross-temporal coherent forecasts for Australian tourism. *Annals of Tourism Research* **75**, 393–409. https://www.sciencedirect.com/science/article/pii/S0160738319300167.

Li, H & H Chen (2022). Hierarchical mortality forecasting with EVT tails: An application to solvency capital requirement. *International Journal of Forecasting*. https://www.sciencedirect.com/science/article/pii/S0169207022001169.

Lyche, T (2020). *Numerical linear algebra and matrix factorizations*. Vol. 22. Springer Nature.

Meyer, CD & I Stewart (2023). *Matrix analysis and applied linear algebra*. SIAM.

O'Hara-Wild, M (2024). *graphvec: Vectorised graph data structures*. R package version 0.0.0.9000. http://pkg.mitchelloharawild.com/graphvec.

O'Hara-Wild, M, R Hyndman & E Wang (2023). *fabletools: Core Tools for Packages in the 'fable' Framework*. https://fabletools.tidyverts.org/.

Shang, HL & RJ Hyndman (2017). Grouped Functional Time Series Forecasting: An Application to Age-Specific Mortality Rates. *Journal of Computational and Graphical Statistics* **26**(2), 330–343.

Wang, E, D Cook & RJ Hyndman (2020). A new tidy data structure to support exploration and modeling of temporal data. *Journal of Computational and Graphical Statistics* **29**(3), 466–478. https://doi.org/10.1080/10618600.2019.1695624.

Wickham, H, M Averick, J Bryan, W Chang, LD McGowan, R François, G Grolemund, A Hayes, L Henry, J Hester, M Kuhn, TL Pedersen, E Miller, SM Bache, K Müller, J Ooms, D Robinson, DP Seidel, V Spinu, K Takahashi, D Vaughan, C Wilke, K Woo & H Yutani (2019). Welcome to the tidyverse. *Journal of Open Source Software* **4**(43), 1686.

Wickramasuriya, SL, G Athanasopoulos & RJ Hyndman (2019). Optimal Forecast Reconciliation for Hierarchical and Grouped Time Series Through Trace Minimization. *Journal of the American Statistical Association* **114**(526), 804–819. https://doi.org/10.1080/01621459.2018.1448825.

Wilkinson, GN & CE Rogers (1973). Symbolic Description of Factorial Models for Analysis of Variance. *Journal of the Royal Statistical Society. Series C (Applied Statistics)* **22**(3), 392–399. http://www.jstor.org/stable/2346786 (visited on 10/16/2023).