FFORMPP: Feature-based forecast model performance prediction

Thiyanga S. Talagala
Department of Econometrics and Business Statistics, Monash University, VIC 3800, Australia, and
ARC Centre of Excellence for Mathematics and Statistical Frontiers
Email: thiyanga.talagala@monash.edu
Corresponding author

Feng Li
School of Statistics and Mathematics, Central University of Finance and Economics, Beijing 100081, China
Email: feng.li@cufe.edu.cn

Yanfei Kang
School of Economics and Management, Beihang University, Beijing 100191, China
Email: yanfeikang@buaa.edu.cn

23 October 2019

JEL classification: C10,C14,C22

FFORMPP: Feature-based forecast model performance prediction

Abstract

This paper introduces a novel meta-learning algorithm for time series forecasting. The efficient Bayesian multivariate surface regression approach is used to model forecast error as a function of features calculated from the time series. The minimum predicted forecast error is then used to identify an individual model or combination of models to produce forecasts. In general, the performance of any meta-learner strongly depends on the reference dataset used to train the model. We further examine the feasibility of using GRATIS (a feature-based time series simulation approach) in generating a realistic time series collection to obtain a diverse collection of time series for our reference set. The proposed framework is tested using the M4 competition data and is compared against several benchmarks and other commonly used forecasting approaches. The new approach obtains performance comparable to the second and the third rankings of the M4 competition.

Keywords: Time series, meta-learning, mixture autoregressive models, surface regression, M4 Competition

1 Introduction

Forecasting is an important aspect of every business operation. The selection of a suitable forecast model or a combination of models to use in forecasting is at the heart of the forecasting process (Tashman & Leach 1991). This selection process is challenging in the context of large-scale time series forecasting for several reasons: i) there is no universal method that performs best for all kinds of forecasting problems; ii) a trial-and-error process of model selection would increase the time and computational cost significantly; iii) it is not possible to derive a typical algebraic expression for choosing the best-performing model(s) out of a portfolio of algorithms; and iv) even with expert knowledge, the correct solution is not guaranteed. A meta-learning approach serves as a promising alternative to solve this problem.

The idea of using meta-learning to select the best forecasting model for a given time series has been explored by several researchers in the context of time series forecasting (Collopy &

Armstrong 1992; Shah 1997; Adya et al. 2001; Wang, Smith-Miles & Hyndman 2009; Petropoulos et al. 2014). This approach is also known as an algorithm selection problem and can be expressed firmly using Rice's framework for algorithm selection (Rice 1976). Further evidence in favour of this idea is also given in Talagala, Hyndman & Athanasopoulos (2018). In all these cases, a vector of features computed from time series is used as input to train a meta-learner. The output and the objective function to train the meta-learner is approached differently. For example, Shah (1997) uses the best forecast model as the output label and applies a discriminant analysis to train a meta-learner to predict the forecast model that is expected to perform best on a given time series. The objective function is to minimise classification error. Further, Prudêncio & Ludermir (2004) use neural network approaches to define weights for the best linear combination of methods to improve forecast accuracy; thus, the objective function is to minimise forecast error.

Although many researchers have highlighted the usefulness of the meta-learning approach to guide the way the forecasts are computed, few studies have concluded that this approach is superior to simple benchmarks and commonly used forecasting approaches. For example, Meade (2000) concludes that the summary statistics are useful in selecting a good forecasting method, but are not necessarily the best. Two possible reasons for the infeasibility of the selection process are the use of inadequate features in the meta-learning process, and having training time series data that are not as diverse as required to predict different forecast model performance (Kang, Hyndman & Li 2019).

This paper is the third in a series of papers addressing the aforementioned issues in developing a meta-learning framework for forecast model selection based on features computed from the time series. Our first attempt to develop a framework for forecast model selection is described in Talagala, Hyndman & Athanasopoulos (2018). The first framework is called FFORMS: Feature-based FORecast Model-Selection. We use the random forest algorithm to predict the forecast model that is expected to perform best on a given time series. In our second paper, Montero-Manso et al. (2019), rather than mapping time series to a single forecast model we use a gradient boosting algorithm to obtain the weights for forecast combinations. We call our second framework FFORMA: Feature-based FORecast Model Averaging. FFORMA placed second in the M4 competition (Makridakis, Spiliotis & Assimakopoulos 2018). Having revisited the literature, we found that to the best of our knowledge none of these studies have considered the correlation structure of algorithm performance in their model training process. The current paper extends this idea. The third algorithm uses the efficient Bayesian multivariate surface regression approach to estimate forecast error for each method, and then uses the minimum predicted error to select a forecasting model or choose individual models for forecast combinations.

The main contributions of this paper are as follows:

1. We propose a novel meta-learning framework for time series forecasting. We refer to this general framework as FFORMPP: Feature-based FORecast Model Performance Prediction. It consists of two phases: the offline phase and the online phase. Most of the expensive computations for processing data and training a meta-learner are performed in the offline phase. We use a collection of time series to train a meta-learner. The data processing part requires computation of a set of features and forecast errors from a pool of forecast models for the time series in our collection. Subsequently, the efficient Bayesian multivariate surface regression approach proposed by Li & Villani (2013) is used to model forecast algorithm performances (measured by a forecast error measure, which in our case is MASE: Mean Absolute Scaled Error) as a function of features calculated from the time series. This produces a meta-learner to be used in the online phase. The online phase requires only the calculation of a simple vector of features for any newly given time series and uses the pre-trained classifier to estimate forecast error for each model in the pool, which is computationally efficient for real-time implementations. This allows ranking of the forecast models with respect to their forecast errors and evaluation of their relative forecast performance without calculating forecasts from all available individual models in the pool.

2. The diversity of features in the collection of time series used to train a meta-learner plays a critical role in training a meta-learning model. Often, time series with the required amount of feature diversity and quality might not be available (Kang, Hyndman & Smith-Miles 2017; Kang, Hyndman & Li 2019). To this end, we explore the use of GRATIS (GeneRAting TIme Series with diverse and controllable characteristics) proposed by Kang, Hyndman & Li (2019) to obtain a diverse collection of time series.

3. We visualise time series in the instance space defined by the features according to the forecast models used to compute combination forecasts. This helps us to explore the distribution of locations of the time series in the instance space and their relationship with features and forecast model selection.

The remainder of the paper is organised as follows: Section 2 introduces the methodology, including the methodology for simulating time series to augment the reference set to train a model and efficient Bayesian multivariate regression approach. Section 3 discusses the results in application to the M4 competition data. Section 4 concludes the paper.

2 Methodology

Figure 1 shows the proposed framework. For each series, a set of features computed from the training period of each series comprising an input vector and the MASE values for each method gives the output vector to train a model. The description of the features calculated in each frequency category is shown in Table 1. We analyse yearly, quarterly, monthly, weekly, daily and hourly series separately. Table 2 shows the forecast models we consider within each frequency category.

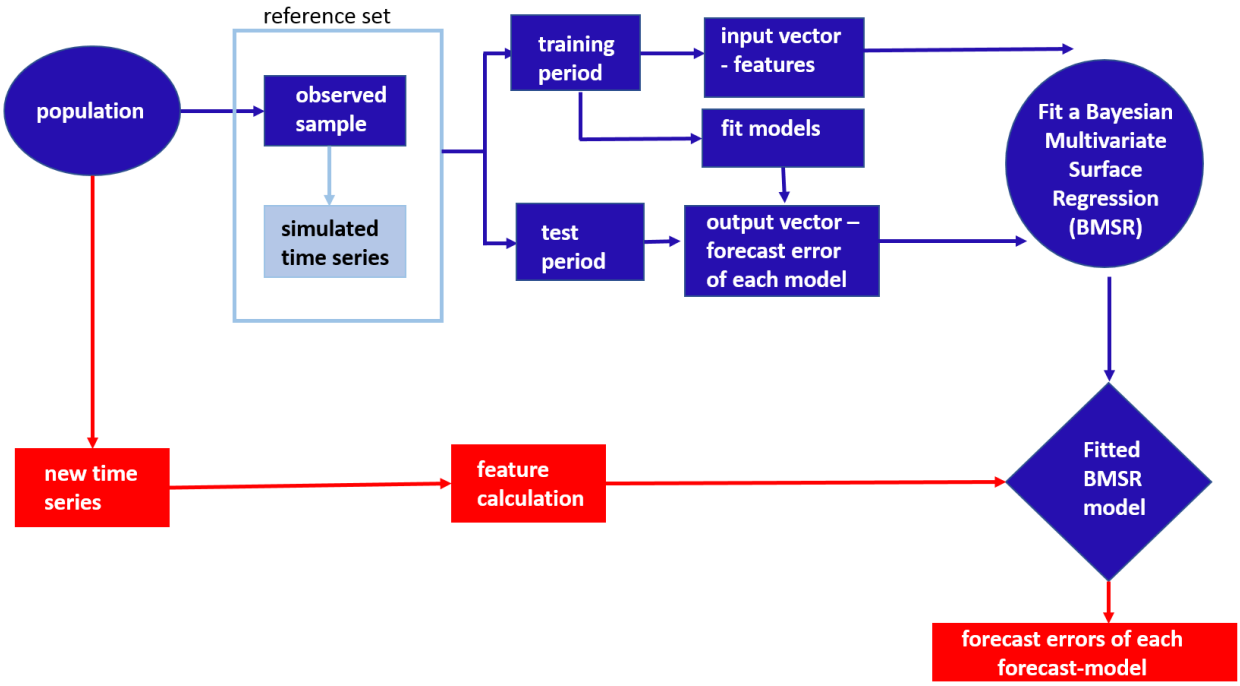

Figure 1: *FFORMPP (Feature-based FORecast Model Performance Prediction) framework. The offline phase is shown in blue and the online phase in red.*

2.1 Reference set

We use the time series of M1 and M3 competitions as the observed sample for yearly, quarterly and monthly time series. In addition to the observed time series, we simulate 10, 000 time series based on MAR (Mixture AutoRegressive) models introduced by Kang, Hyndman & Li (2019). The observed time series and the simulated time series form the reference set to fit the model. For weekly, daily and hourly frequencies, only the simulated time series are used to create the reference set because the previous M-competitions do not contain the time series corresponding to weekly, daily and hourly frequencies. Once we have formed the reference set, we split each time series in the reference set into training period and test period. Features are calculated based on the training period of each series and the forecast error measure is calculated over the test period of each time series. Table 3 summarises the number of time series in the reference set and

Table 1: *Time series features*

	Feature	Description	Y	Q/M	W	D/H
1	T	length of time series	✓	✓	✓	✓
2	trend	strength of trend	✓	✓	✓	✓
3	seasonality_q	strength of quarterly seasonality	-	✓	-	-
4	seasonality_m	strength of monthly seasonality	-	✓	-	-
5	seasonality_w	strength of weekly seasonality	-	-	✓	✓
6	seasonality_d	strength of daily seasonality	-	-	-	✓
7	seasonality_y	strength of yearly seasonality	-	-	-	✓
8	linearity	linearity	✓	✓	✓	✓
9	curvature	curvature	✓	✓	✓	✓
10	spikiness	spikiness	✓	✓	✓	✓
11	e_acf1	first ACF value of remainder series	✓	✓	✓	✓
12	stability	stability	✓	✓	✓	✓
13	lumpiness	lumpiness	✓	✓	✓	✓
14	entropy	spectral entropy	✓	✓	✓	✓
15	hurst	Hurst exponent	✓	✓	✓	✓
16	nonlinearity	nonlinearity	✓	✓	✓	✓
17	alpha	ETS(A,A,N) $\hat{\alpha}$	✓	✓	✓	-
18	beta	ETS(A,A,N) $\hat{\beta}$	✓	✓	✓	-
19	hwalpha	ETS(A,A,A) $\hat{\alpha}$	-	✓	-	-
20	hwbeta	ETS(A,A,A) $\hat{\beta}$	-	✓	-	-
21	hwgamma	ETS(A,A,A) $\hat{\gamma}$	-	✓	-	-
22	ur_pp	test statistic based on Phillips-Perron test	✓	-	-	-
23	ur_kpss	test statistic based on KPSS test	✓	-	-	-
24	y_acf1	first ACF value of the original series	✓	✓	✓	✓
25	diff1y_acf1	first ACF value of the differenced series	✓	✓	✓	✓
26	diff2y_acf1	first ACF value of the twice-differenced series	✓	✓	✓	✓
27	y_acf5	sum of squares of first 5 ACF values of original series	✓	✓	✓	✓
28	diff1y_acf5	sum of squares of first 5 ACF values of differenced series	✓	✓	✓	✓
29	diff2y_acf5	sum of squares of first 5 ACF values of twice-differenced series	✓	✓	✓	✓
30	sediff_acf1	ACF value at the first lag of seasonally-differenced series	-	✓	✓	✓
31	sediff_seacf1	ACF value at the first seasonal lag of seasonally-differenced series	-	✓	✓	✓
32	sediff_acf5	sum of squares of first 5 autocorrelation coefficients of seasonally-differenced series	-	✓	✓	✓
33	seas_pacf	partial autocorrelation coefficient at first seasonal lag	-	✓	✓	✓
34	lmres_acf1	first ACF value of residual series of linear trend model	✓	-	-	-
35	y_pacf5	sum of squares of first 5 PACF values of original series	✓	✓	✓	✓
36	diff1y_pacf5	sum of squares of first 5 PACF values of differenced series	✓	✓	✓	✓
37	diff2y_pacf5	sum of squares of first 5 PACF values of twice-differenced series	✓	✓	✓	✓

Table 2: *Class labels*

class label	Description	Y	Q/M	W	D/H
WN	white noise process	✓	✓	✓	✓
auto.arima	ARIMA processes	✓	✓	✓	-
ets	exponential smoothing	✓	✓	-	-
rwd	random walk with drift	✓	✓	✓	✓
rw	random walk	✓	✓	✓	✓
theta	standard theta method	✓	✓	✓	✓
stlar	STL-AR method	-	✓	✓	✓
snaive	seasonal naive method	-	✓	✓	✓
tbats	TBATS forecasting	-	✓	✓	✓
nn	neural network time series forecasts	✓	✓	✓	✓
mstlets	multiple seasonal time series	-	-	✓	✓
mstlarima	multiple seasonal time series	-	-	-	✓

the new time series in each frequency category. Note that for yearly, quarterly and monthly time series, the reference set used to train a meta-learner is much smaller than the test set evaluating the meta-learner. This is often the case when applying the meta-learner in practice during the online phase.

Table 3: *Composition of the time series in the reference set and collection of new time series*

Frequency	Reference set			New series
	M1	M3	Simulated	M4
Yearly	181	645	10000	23000
Quarterly	203	756	10000	24000
Monthly	617	1428	10000	48000
Weekly	-	-	10000	359
Daily	-	-	10000	4227
Hourly	-	-	10000	414

2.2 Augmenting the observed sample with simulated time series from mixture autoregressive models

The reference set is augmented with simulated time series to obtain a more heterogeneous collection of time series for training a model. This helps to reduce overfitting to a relatively homogeneous set of data and increases generalisability of the model when applied to new time series with different conditions. Further, this is useful when there is no observed set of time series available to train a model in the offline phase. In this study, the simulated time series are generated based on the algorithm proposed by Kang, Hyndman & Li (2019), hereafter referred to as GRATIS[1]. Although there is no standard process for simulating time series, the most common approach involves simulation based on some data-generating processes (DGPs) such as exponential smoothing and ARIMA models. Instead of relying on a set of DGPs to generate time series, the GRATIS algorithm simulates time series based on a diverse set of time series features using MAR models. The algorithm can also allow a user to set controllable features for simulating time series. Table 4 lists the choice of values for parameters used to simulate time series in each frequency category. We used frequencies 1, 4, 12 and 52 to generate yearly, quarterly, monthly and weekly series respectively. Daily and hourly time series with a long history often show multiple seasonal patterns. Hence, for daily series, frequencies were set to 7 (time-of-week pattern) and 365.25 (annual seasonality), while for hourly series frequencies were set to 24 (time-of-day pattern) and 168 (time-of-week pattern). None of the time series in our hourly test dataset are longer than 8760. Hence, time-of-year pattern ($365 \times 24 = 8760$) was not considered. Except for hourly series, length of the time series is randomly chosen from

[1]The R package `tsgeneration` accompanies this work and is publicly available on https://github.com/ykang/tsgeneration.

uniform distribution. The minimum and maximum values of the distributions are selected based on lower (Q1 - 1.5 $\times$ IQR) and upper (Q3 + 1.5 $\times$ IQR) edges of the box-and-whisker plots. We use the M4 competition data to compute the associated statistics for length. The corresponding distributions are shown in Figure 2. The reason for this choice is that we use length of the time series as a feature in our meta-learning framework. Hence, lengths of the series in our reference set should cover all or the majority of time series we need to forecast. In practice, it might be difficult to obtain a reference set covering the whole length of the new time series for two main reasons: i) information regarding the range of length is not available at the offline stage (although, a rough idea about the distribution of the majority can be obtained); and ii) the range of length is very wide owing to 'outlying' observations (for example, Figure 2, quarterly and monthly series). In such circumstances, a reference set covering most of the lengths of future time series is a reasonable approach. Further, for each series we randomly select a number of mixing components from $\{1, 2, 3, 4, 5\}$. Other parameters are set the same as those in Kang, Hyndman & Li (2019). Having each parameter set as in Table 4, we generate 10000 time series from each frequency category. This is to reduce the training time of the efficient Bayesian multivariate surface regression model while maintaining the diversity of the reference set.

Table 4: *GRATIS: Choice of values for parameters for simulating time series using MAR models*

Value	Parameter	Description
frequency	seasonal period	yearly 1 quarterly 4 monthly 12 weekly 52 daily (7, 365) hourly (24, 168)
length	length of time series	yearly U(19, 75) quarterly U(24, 202) monthly U(60, 660) weekly U(93, 2610) daily U(107, 9933) hourly: 40.8% with 748 length and 59.2% with 1008 length
k	number of mixing components	For each series randomly chosen from $\{1, 2, 3, 4, 5\}$
α_k	weights of mixture components	$\alpha_k = \beta_k / \sum_{i=1}^{K} \beta_i$, where $\beta_i \sim U(0, 1)$
θ_{ki}	coefficients of the AR part	N(0, 0.5)
Θ_{kj}	coefficients of the seasonal AR parts	N(0, 0.5)
d_k	number of differences in each component	Bernoulli(0.9)
D_k	number of seasonal differences in each component	Bernoulli(0.4)

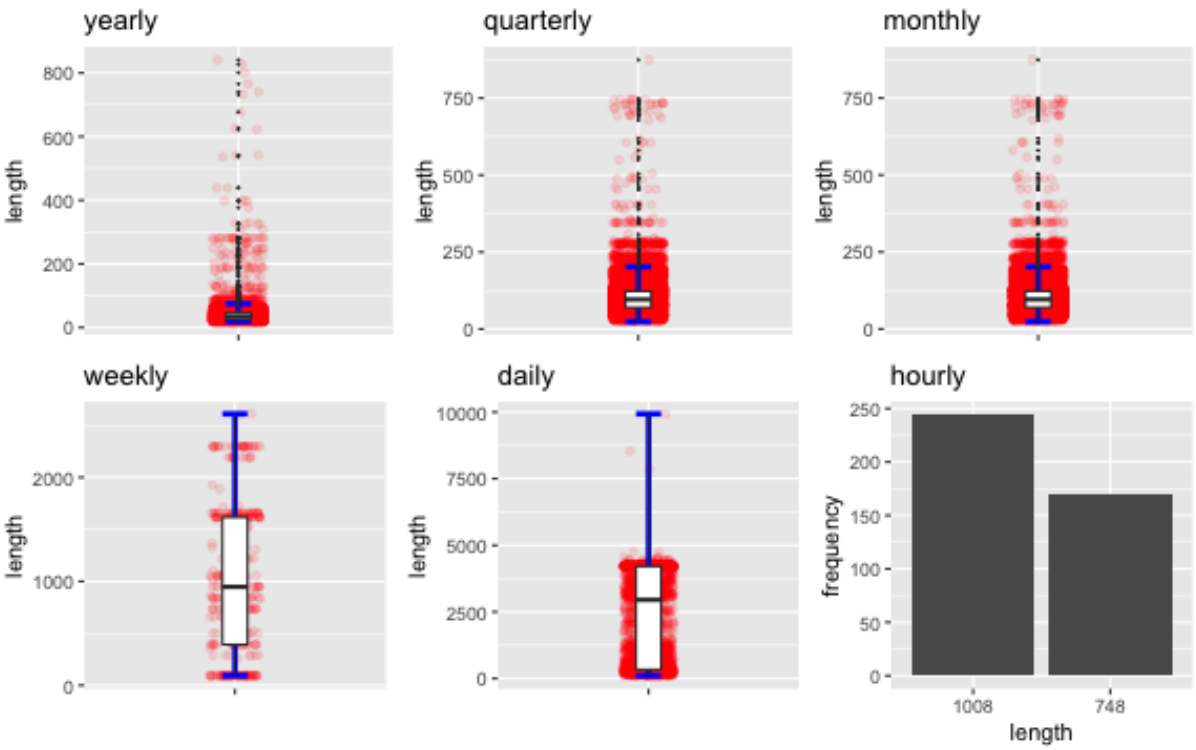

Figure 2: *Distribution of length of time series of the M4 competition across different frequency categories. The data points are shown in red. Horizontal blue solid line shows the lower and the upper whisker borders of each boxplot. Some outliers could be observed in yearly, quarterly and monthly categories. There are only two distinct levels of length for hourly series.*

2.3 Efficient Bayesian multivariate surface regression approach

The most commonly used additive spline regression assumes additivity in the regressor; that is, $E(y|x_1,...,x_q) = \sum_{j=1}^{q} f_j(x_j)$, where $f_j(x_j)$ is a spline regressor of the j^{th} regressor. Even though the assumption of additivity simplifies the model, it is quite a restrictive assumption. This problem has motivated research on surface models with interactions between regressors (Li & Villani 2013). Li & Villani (2013) proposed a general Bayesian approach for fitting surface models for a continuous multivariate response by combining additive splines and interactive splines. The proposed modelling is called efficient Bayesian multivariate surface regression.

The number of knots has an important influence in the resulting fit of spline regression: without enough knots the regression is underfitted and with too many knots it is overfitted. Choosing the locations of knots is also a challenge. This becomes even harder for surface regression than it is for additive models because any feasible set of q-dimensional knots is necessarily sparse in R^q when the number of regressors, q, is moderate or large. This causes the curse of dimensionality. The most common approach used in the literature is to use a fixed set of knot locations, and most of these algorithms place the knots at the centroids of the clusters computed based on regressor observations. Li & Villani (2013) pointed out that this is impractical when estimating a surface

with several regressors. Hence, the authors proposed a computationally efficient Markov chain Monte Carlo (MCMC) algorithm for the Gaussian multivariate surface regression to update the locations of the knots jointly. Instead of a fixed set of locations, the authors introduce 'moving knots'.

The proposed Gaussian multivariate regression model can be written as follows:

$$\boldsymbol{Y} = \boldsymbol{X_0 B_0} + \boldsymbol{X}_a(\boldsymbol{\xi}_a)\boldsymbol{B}_a + \boldsymbol{X}_s(\boldsymbol{\xi}_s)\boldsymbol{B}_s + \boldsymbol{E}, \tag{1}$$

where $\mathbf{Y}$ is a matrix of n number of observations and p number of response variables. The rows of $\mathbf{E}$ are error vectors assumed to be independent and identically distributed (iid) as $N_p(0, \Sigma)$. The proposed efficient Bayesian multivariate surface regression model contains three components:

1. **Linear component:** The linear component contains the original covariates including the constant term. This enters the model in linear form. The matrix $\mathbf{X_0}$ is a $n \times q_0$ vector in which the first column contains ones for the intercept. The corresponding regression coefficients are in $\mathbf{B_0}$.

2. **Additive component:** The second component of the model contains additive spline basis functions of the covariates in $\mathbf{X_0}$. This is represented by $\boldsymbol{X}_a(\boldsymbol{\xi}_a)$, where ξ_a represents the knots. It is important to note that the knots in the additive part of the model are scalars and this model allows an unequal number of knots for different covariates in the model. The matrix $\mathbf{B_a}$ contains the regression coefficients corresponding to the additive component. The additive component of the model captures the non-linear relationship between features and response $\mathbf{Y}$.

3. **Surface component:** The surface component of the model contains the radial basis function for capturing the remaining part of the surface and interactions. This is denoted by $\boldsymbol{X}_s(\boldsymbol{\xi}_s)$. Note that the ξ_s is a q_0-dimensional vector. The matrix $\mathbf{B_s}$ contains the regression coefficients corresponding to the surface component and ξ_s represents the surface knots.

For notational convenience Equation (1) can be written as

$$\mathbf{Y} = \mathbf{XB} + \mathbf{E}, \tag{2}$$

where $\mathbf{X} = [\mathbf{X_0}, \mathbf{X_a}, \mathbf{X_s}]$ is the $n \times q$ design matrix ($q = q_0 + q_a + q_s$) and $\mathbf{B} = [\mathbf{B_0'}, \mathbf{B_a'}, \mathbf{B_s'}]$. For a given set of fixed knot locations, the model in Equation (1) is linear in regression parameters. The

over-parameterised problem is addressed by using shrinkage priors to shrink small regression coefficients towards zero. The shrinkage parameters and knot locations of ξ_a and ξ_s are treated as unknown parameters to be estimated. Li & Villani (2013) proposed a computationally efficient MCMC algorithm to estimate shrinkage parameters and update the locations of the knots of additive and surface components jointly. This approach allows an unequal number of knots in the different covariates. In addition, separate shrinkage parameters for the linear, additive and surface parts of the model are allowed. Further, this approach permitted separate shrinkage parameters for the p responses within each of the three model parts. In contrast to other spline-bases models, this approach allows the knot locations to move freely in the regressor space, and thus fewer knots are usually required. The estimation and computation details can be found in Li & Villani (2013).

3 Application to the M4 competition data

3.1 Dissimilarity between different datasets

Principal component analysis preserves the dissimilarity between widely separated data points rather than the similarity between nearby data points. This feature is useful for improved confidence in simulated series' representativeness of real time series. For example, if the simulated time series results in isolated clusters or highly dense clusters far apart from the real-world time series, it indicates a poor representation of the real data.

We use principal component analysis to visualise dissimilarity between the datasets: observed time series (M1 and M3), simulated series and new time series (M4) in the two-dimensional instance space. The data were normalized using the z-scale transformation before applying PCA. This also helps us to gain an idea about the global structure of the location of the different collections. The results for yearly, quarterly and monthly series are shown in Figure 3. We compute principal components using the time series in the reference set (observed series and simulated series) and project new series (M4) into the two-dimensional space spanned by the first two eigen vectors. The first two principal components explain 51.3%, 48.7% and 48.3% of the total variation in the yearly, quarterly and monthly data. We see that the distribution of the simulated time series (represented by the dark orange dots) clearly nests and fills in the instance space. The simulated time series fills the instance space by further expanding the density range of observed time series rather than resulting in isolated clusters. This guarantees that the simulated data generated based on the GRATIS approach are actually representative of real data. Further, we can see that the projection of M4 series falls within the space created

by the series in the reference set (M1, M3 and simulated). This is very important because our FFORMS framework is trained based on the series in the reference set; hence, the model is valid over the space of the reference set. A few M4 time series in the monthly frequency category fall outside the convex hull of the reference set in the first two principal components owing very high length. Note that for yearly, quarterly and monthly series the size of the M4 time series collection is much greater than the size of the corresponding collection of simulated series. For yearly and quarterly data, the number of time series in the M4 collection is about twice as large as the simulated series, and for monthly data the M4 competition collection is four times larger than the simulated series. This shows the efficiency of the GRATIS simulation approach in increasing the diversity of feature space without having many time series similar in size to the new time series collection from which we wish to produce forecasts.

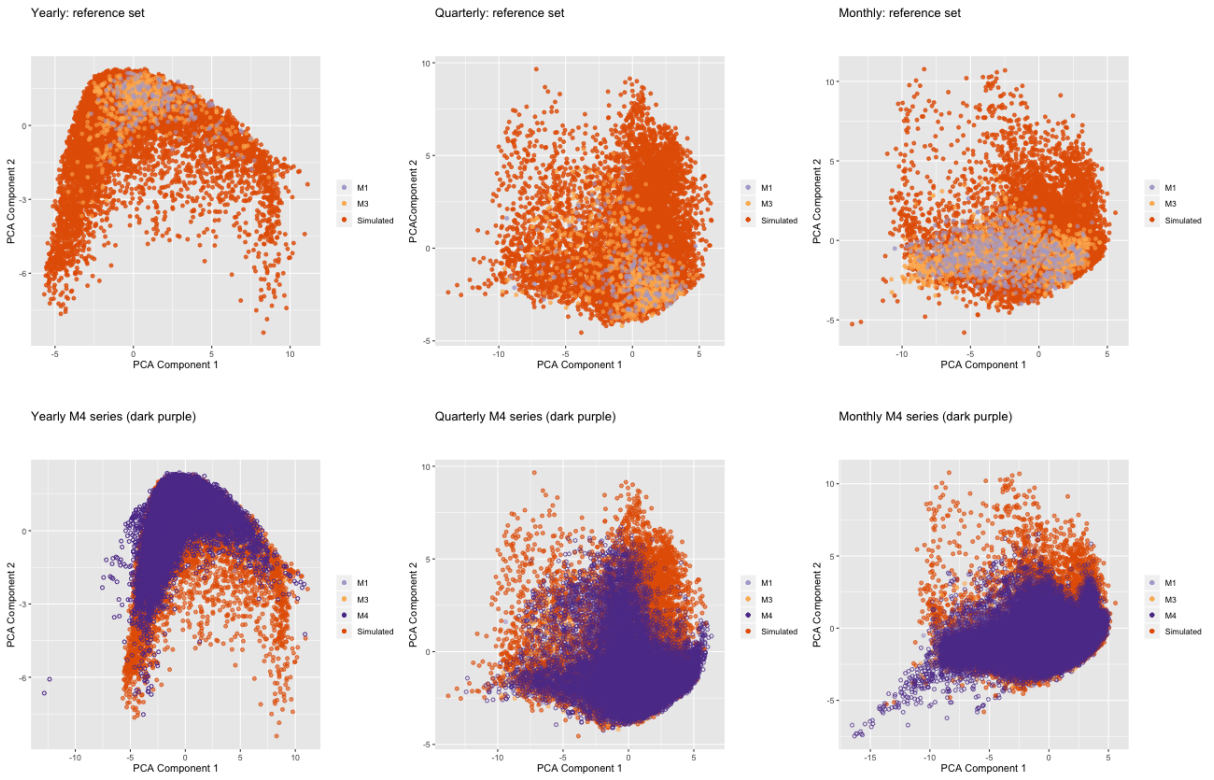

Figure 3: *Distribution of yearly, quarterly and monthly time series on the PCA space. On each graph, dark orange represents the simulated series and purple denotes new time series (M4 data). In the first row observed time series in the reference are highlighted in light orange and light purple. PCA space is computed based on the time series in the reference set and then the M4 competition (new series) are projected into the two-dimensional PCA space. Except for a few series, the majority of new time series we need to forecast fall within the space of the reference set. Note that for yearly, quarterly and monthly series the size of the test set is much larger than the size of the corresponding reference set.*

Figure 4 shows the principal component projections of weekly, daily and hourly time series. The first two principal components explain 48.7%, 43.4% and 45.4% of the total variance of

weekly, daily and hourly series. In all three frequency categories, M4-competition data fall within the feature space of simulated series. For weekly, daily and hourly frequency categories, simulated time series based on the GRATIS approach successfully fill in the space of the new time series. This shows the efficiency of the GRATIS approach in generating a realistic set of representative time series. This also shows that the GRATIS approach is a good choice to augment the reference set when only small amounts of training data are available. However, daily series of M4 competition series are clustered into a single location of the feature space. This is due to the quality issues of the M4 competition data. Daily series in the M4 competition are very similar to each other owing to data leakages. Examples of data leakage include, use of different segments of time series to create a new time series, and addition of a constant value to create new time series (Ingel et al. 2019). This reveals that the GRATIS simulated-based approach can also be used to evaluate the quality issues in the data.

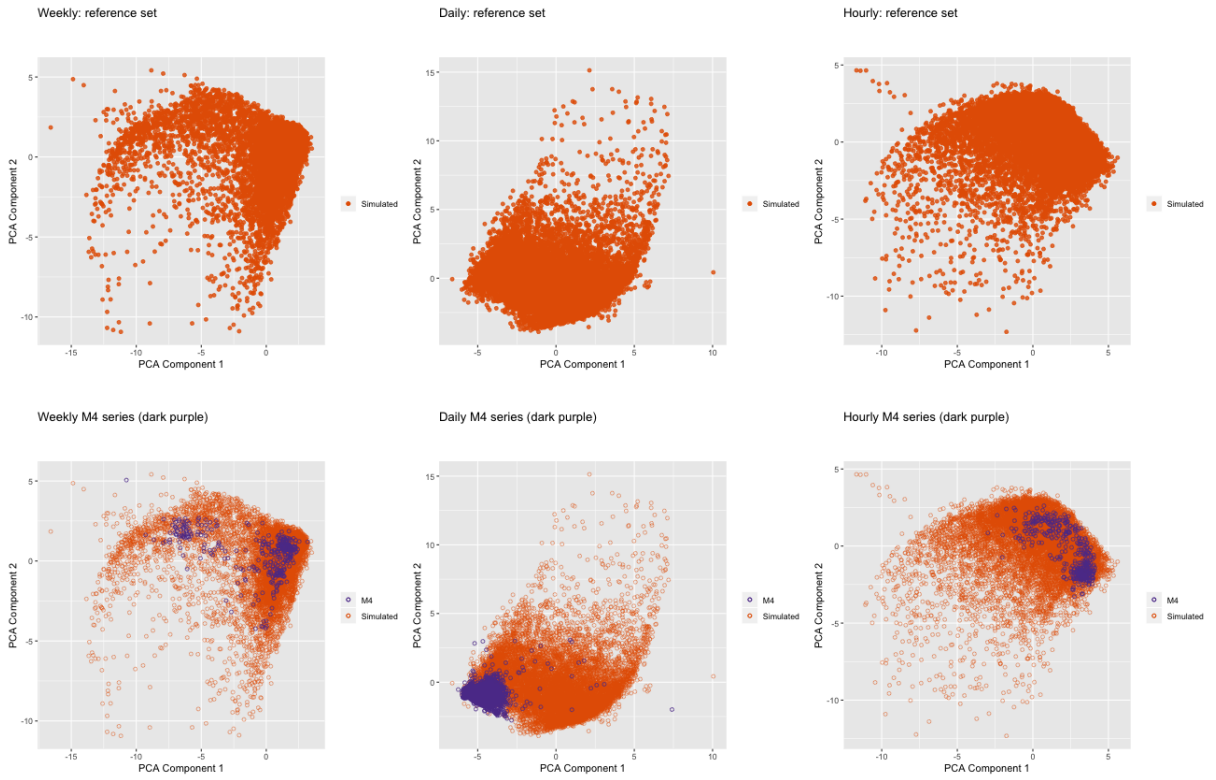

Figure 4: *Distribution of weekly, daily and hourly time series on the PCA space. On each graph, dark orange represents the simulated series and purple denotes new time series (M4 data). PCA space is computed based on the time series in the reference set and then the M4 competition (new series) are projected into the two-dimensional PCA space. In each frequency category, new series fall within the PCA space of reference set.*

3.2 Coverage analysis

To obtain a more accurate estimate of the exact coverage of simulated data, we adapt the idea used by Kang, Hyndman & Li (2019). For this purpose, the t-Stochastic Neighbor Embedding (t-SNE) approach is adapted. The t-SNE is a non-linear dimension reduction technique that seeks to find a low-dimensional sub-manifold that preserves the local structure of the dataset. In other words, the t-SNE approach keeps the neighbouring points in the original data space close in the embedding space. To quantify the miscoverage, first a grid of 900 squares was superimposed on the region of space covered by each scatter plot shown in Figure 5. Then the miscoverage of dataset A over dataset B is computed as follows (Kang, Hyndman & Li 2019):

$$miscoverage_{A/B} = N^{-2} \sum_{i=1}^{N^2} (1 - I_{i,A}) \times I_{i,B},$$

where $I_{i,A} = 1$ if points in dataset A fall within i^{th} square and $I_{i,A} = 0$ otherwise. An analogous definition is applied to $I_{i,B}$ computed based on the dataset B. We computed miscoverage of the simulated dataset over the M4 dataset and vice versa. The results are shown in Table 5. Table 5 shows that our attempt to increase the diversity of the reference set using the GRATIS approach is successful.

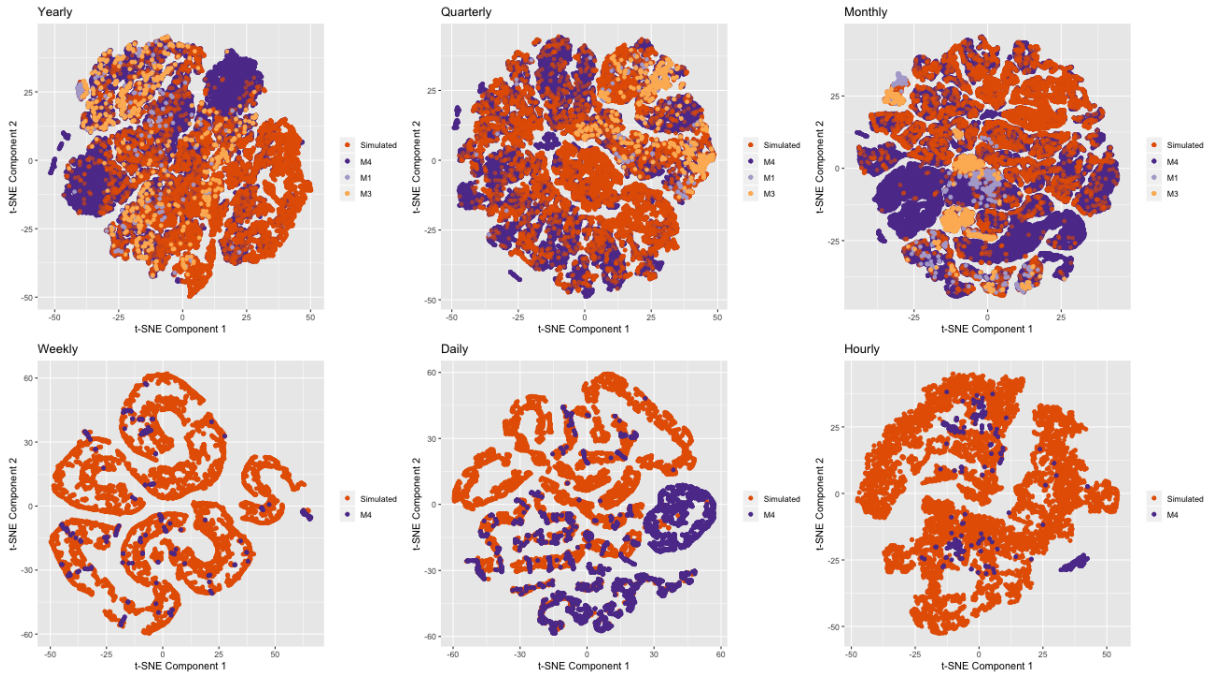

Figure 5: *Two-dimensional t-SNE instance spaces of the M1, M3, M4 and simulated series for yearly, quarterly, monthly, weekly, daily and hourly series.*

Table 5: *Coverage analysis*

	Yearly	Quarterly	Monthly	Weekly	Daily	Hourly
Miscoverage of the 'Simulated' dataset over the 'M4' dataset	0.068	0.043	0.132	0.001	0.061	0.011
Miscoverage of the 'M4' dataset over the 'Simulated' dataset	0.026	0.022	0.011	0.438	0.278	0.490

3.3 Forecasting results

We evaluate the out-of-sample performances of our proposed meta-learning framework to the benchmark based on the M4 competition series. The mean absolute scale error (MASE) (Hyndman & Koehler 2006) was used to evaluate forecast accuracy. We compute both individual and combination forecasts for the M4 competition series. Individual forecasts are computed based on the model corresponding to the minimum predicted MASE. The combination forecasts are computed by taking the median of individual forecasts corresponding to the four models with minimum predicted MASE. The results are shown in Table 6. The results suggest that the forecast combination obtained through the FFORMPP framework outperforms the corresponding benchmark and other commonly used methods of forecasting. Further, we compare our results with those of the top three places of the M4 competition. The winning method of the competition is based on the Hybrid approach, which is a combination of exponential smoothing models and neural network approaches (Makridakis, Spiliotis & Assimakopoulos 2018). The second and third approaches are based on combination forecasts computed based on nine and seven individual forecast models respectively. According to the results of Table 6, we can see that our approach achieved comparable results in a much more cost- and time-effective manner because our combination forecasts are calculated based on four individual models.

We also reported the relative frequencies with which each forecast model is selected as a component of the calculation of combination forecasts based on FFORMPP. Further, to gain an idea of the different types of model combinations used to compute forecast, we cluster time series based on the models that are used to compute FFORMPP combination. For this purpose, we first create a design matrix of 1 and 0. The columns of the design matrix correspond to each individual forecast model, and rows correspond to each time series. The cell values of the matrix are 1s and 0s, where 1 is assigned if a corresponding model is used for combination forecast, and 0 otherwise. Then, hierarchical clustering was done by using a binary distance metric and ward clustering method. A cluster analysis was performed separately for each frequency category.

For each frequency category we identified three main clusters. The results are shown in Table 7

Table 6: *MASE values calculated over the M4 competition data*

	Yearly	Quarterly	Monthly	Weekly	Daily	Hourly
FFORMPP-combination*	**3.07**	**1.13**	**0.89**	**2.46**	3.62	**0.96**
FFORMPP-individual	3.37	1.17	1.05	2.53	4.26	1.06
auto.arima	3.40	1.17	0.93	2.55	-	-
ets	3.44	1.16	0.95	-	-	-
theta	3.37	1.24	0.97	2.64	3.33	1.59
rwd	**3.07**	1.33	1.18	2.68	3.25	11.45
rw	3.97	1.48	1.21	2.78	**3.27**	11.60
nn	4.06	1.55	1.14	4.04	3.90	1.09
stlar	-	2.02	1.33	3.15	4.49	1.49
snaive	-	1.66	1.26	2.78	24.46	2.86
tbats	-	1.19	1.05	2.49	**3.27**	1.30
wn	13.42	6.50	4.11	49.91	38.07	11.68
mstlarima	-	-	-	-	3.84	1.12
mstlets	-	-	-	-	3.73	1.23
combination (median)	3.29	1.22	0.95	2.57	3.52	1.33
combination (mean)	4.09	1.58	1.16	6.96	7.94	3.93
M4 competition top 3 places (MASE)						
M4-1st	2.98	1.12	0.88	2.36	3.45	0.89
M4-2nd	3.06	1.11	0.89	2.11	3.34	0.81
M4-3rd	3.13	1.12	0.91	2.16	2.64	0.87

and the associated graphical representation is shown in Figure 6. According to the results of Table 6, for yearly data we can see that `auto.arima`, `ets`, `theta` and `rwd` give the best individual forecast. From Table 7 we can see that those four models were most frequently selected to the FFORMPP combination forecast. Similarly, from Table 6 we can see that for quarterly and monthly series `auto.arima`, `ets` and `tbats` provide the best individual forecasts, and according to Table 7, we can see that those models are selected most often (approximately greater than 75%) for quarterly and monthly series. Similarly, we can interpret the results for weekly, daily and hourly series. Cluster 3 in the yearly series is very similar to cluster 2, the only difference being that, cluster 3 uses `nn` instead of `theta` model. The series in the first cluster uses a different combination of models from that which we considered for the yearly series, apart from the two combinations used in cluster 2 and cluster 3 respectively. For quarterly series, the biggest cluster is cluster 3, in which `auto.arima`, `ets`, `rwd` and `tbats` are used to calculate combination forecasts. Similar to the results of quarterly series, for monthly and daily series we observe two clusters that are homogeneous in terms of models used to compute combination forecasts. In terms of weekly data, most of the series use `auto.arima`, `theta`, `rwd` and `tbats` for combination forecasts. For daily and hourly series, it is interesting to observe that all the series in cluster 2 and cluster 3 use at least one of the models `mstlarima` or `mstlets`, which handle multiple seasonality, as a component in calculating combination forecast.

Table 7: *Relative frequencies that each forecast model was selected as a component to the calculation of combination forecasts based on FFORMPP (All values are shown in percentages).*

Source	No. of series	auto.arima	ets	theta	rwd	rw	nn	stlar	snaive	tbats	wn	mstlarima	mstlets
Yearly	23000	**82.7**	**80.4**	**79.2**	**79.1**	33.7	34.6	-	-	-	10.2	-	-
Cluster 1	9287	57.2	51.5	84.3	48.3	83.5	49.9	-	-	-	25.3	-	-
Cluster 2	10391	100	100	100	100	0	0	-	-	-	0	-	-
Cluster 3	3322	100	100	0	100	0	100	-	-	-	0	-	-
Quarterly	24000	**93.1**	**82.8**	13.8	47.8	11.4	35.8	12.9	9.1	**87.4**	5.8	-	-
Cluster 1	7126	76.6	42.1	46.6	22.5	38.4	22.6	43.7	30.5	57.6	19.5	-	-
Cluster 2	6994	100	100	0	0	0	100	0	0	100	0	-	-
Cluster 3	9880	100	100	0	100	0	0	0	0	100	0	-	-
Monthly*	48000	**88.9**	**74.4**	30.6	45.4	16.1	31.8	19.9	7.1	**83.1**	2.8	-	-
Cluster 1	12615	74.1	40.0	72.0	20.9	37.5	25.5	46.8	16.5	60.1	6.5	-	-
Cluster 2	11084	100	100	0	100	0	0	0	0	100	0	-	-
Cluster 3	6301	100	100	0	0	0	100	0	0	100	0	-	-
Weekly	359	**94.4**	-	**74.7**	69.4	9.8	31.5	29.5	10.9	**75.5**	4.5	-	-
Cluster 1	105	80.9	-	57.1	6.7	76.2	63.8	100	37.1	31.4	15.2	-	-
Cluster 2	68	100	-	32.4	82.4	39.7	67.6	1.5	0	76.5	0	-	-
Cluster 3	186	100	-	100	100	0	0	0	0	100	0	-	-
Daily	4227	-	-	18.4	8.7	9.5	**85.9**	**72.9**	5.63	**87.7**	4.1	65.8	44.3
Cluster 1	1891	-	-	34.5	19.5	21.2	68.5	39.9	12.6	72.6	9.1	65.4	52.3
Cluster 2	791	-	-	0	0	0	100	100	0	100	0	0	100
Cluster 3	1545	-	-	0	0	0	100	100	0	100	0	100	0
Hourly	414	-	-	2.9	4.8	16.2	68.8	57.7	14.3	**91.3**	21.0	**78.9**	43.9
Cluster 1	151	-	-	40.4	29.1	25.2	80.1	50.3	19.2	76.2	21.2	46.4	11.9
Cluster 2	162	-	-	0	0	0	100	100	0	0	0	100	100
Cluster 3	101	-	-	47.5	0	0	16.8	20.8	14.9	100	0	100	100

Note: *Cluster analysis is based on 30000 series owing to limited computing resources.

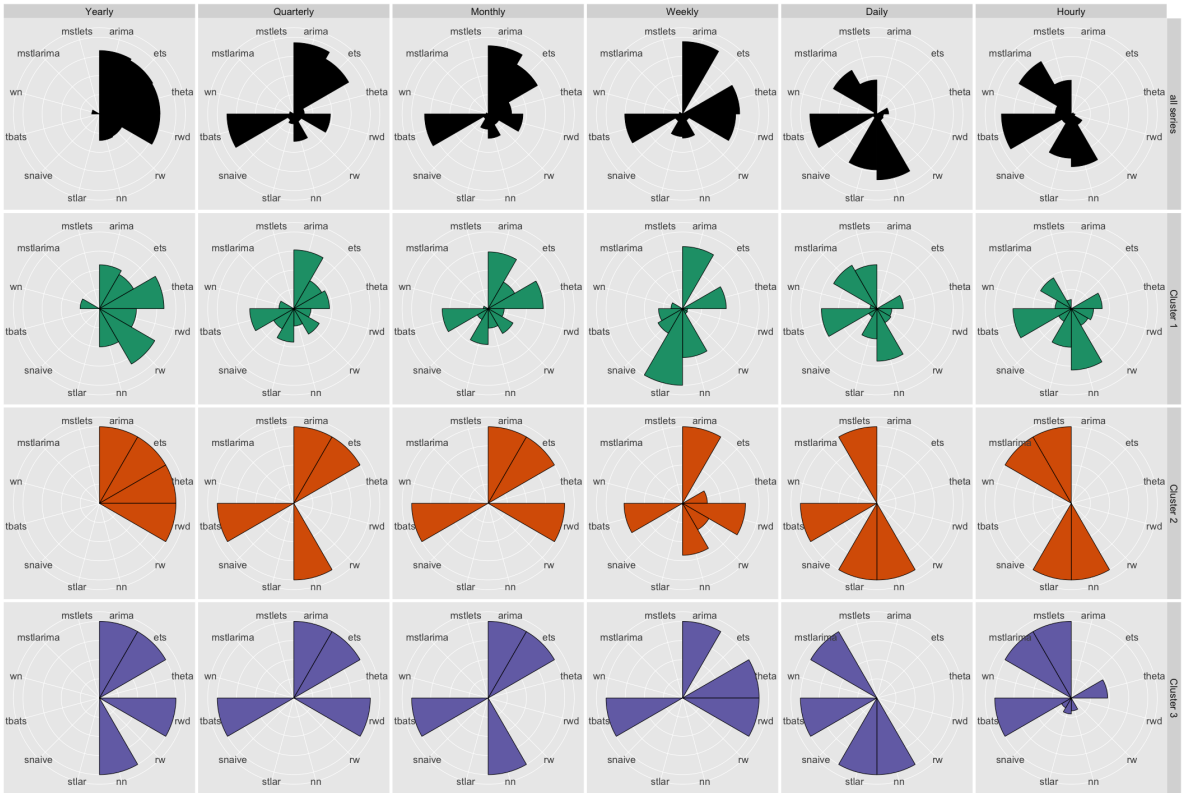

Figure 6: *Visual representation of relative frequencies that each forecast model was selected as a component to the calculation of combination forecasts based on FFORMPP. Each polar coordinate shows relative frequency that each forecast model was selected as a component to the calculation of combination forecasts.*

We now examine the locations of different clusters of M4 competition series in the instance space defined by features. Figure 7 shows the locations of the three clusters in the instance space computed based on the t-SNE approach. For yearly series we can see that most of the series in cluster 1 fall just below the diagonal, while series in clusters 2 and 3 fall within the upper triangular region. Further, clusters 2 and 3 are similar and the series corresponding to cluster 2 and 3 stay close together. For quarterly series, an interesting pattern of clusters can be observed. The time series in cluster 1 fall within the inner circle, while the series in clusters 2 and 3 fall within the second and third outer rings of the circle. This pattern of clusters is called non-spherical. For monthly series, the grouping of the clusters is not obvious. However, a close inspection of the instance space shows a spiral pattern of clusters. Further, cluster 3 is preferred by the series in the upper right corner. For weekly series, clusters are more dispersed across the instance space. For daily and hourly series, clear separation of cluster 1 from clusters 2 and 3 can be observed. Clearly, within each frequency category, similar clusters (for example, clusters 2 and 3 for yearly, quarterly and monthly series) are located close together in the instance space ensuring the similarity of the features of those time series.

The useful description provided by Figure 7 further prompts us to consider the challenge of identifying how the features of time series influence the grouping of these clusters. We now consider how different features vary across the instance space to understand how the locations of different time series reveal the relationship between the features and cluster separation. The preliminary results of the M4 competition (Makridakis, Spiliotis & Assimakopoulos 2018) show that randomness of time series is the most critical factor influencing the forecast accuracy followed by linearity. Further, in their follow-up paper Spiliotis et al. (2019) point out that highly trended and seasonal time series tend to be easier to forecast. The information about remainder series is useful for gaining an idea of the random variation not explained by the trend and seasonality of the series. Hence, we explore the instance space corresponding to the features, strength of trend (`trend`), linearity, strength of seasonality (`seasonality`) and `e_acf1` (the first autocorrelation coefficient of the remainder series after applying STL decomposition on the time series) (Cleveland et al. 1990). The results are shown in Figure 8 - Figure 13 for yearly, quarterly, monthly, weekly, daily and hourly series respectively. Combining the views of the instance space in Figure 7 and Figure 8 - Figure 13 gives us a picture of how the features contribute to the differences in clusters. According to the results of Figure 8- Figure 11, for yearly, quarterly, monthly and weekly series, highly trended series are more likely to fall within clusters 2 and 3. For yearly, quarterly and monthly series, most of the time series in the green cluster (in Figure 7) are less trended with low linearity (in Figure 8 - Figure 10). Hence, those time

series can be considered hard or challenging series to forecast. This is further confirmed from the results of Table 6 and Figure 7: cluster 1 is more heterogeneous according to the way the models are selected to compute combination forecasts (because they use different combinations of individual forecast models), in contrast to clusters 2 and 3. Further, according to the Figure 8-Figure 10, we can see that for yearly, quarterly and monthly series the instance space coloured by linearity exhibits similar structure to the corresponding cluster distribution across instance space shown in Figure 7. For daily series, the features seasonality and e_acf1 clearly separate cluster 1 from the rest. However, for other frequency categories, a clear separation of features with respect to seasonality and e_acf1 cannot be observed. Figure 13 shows that, for hourly time series, features appear to separate the instance space into left and right.

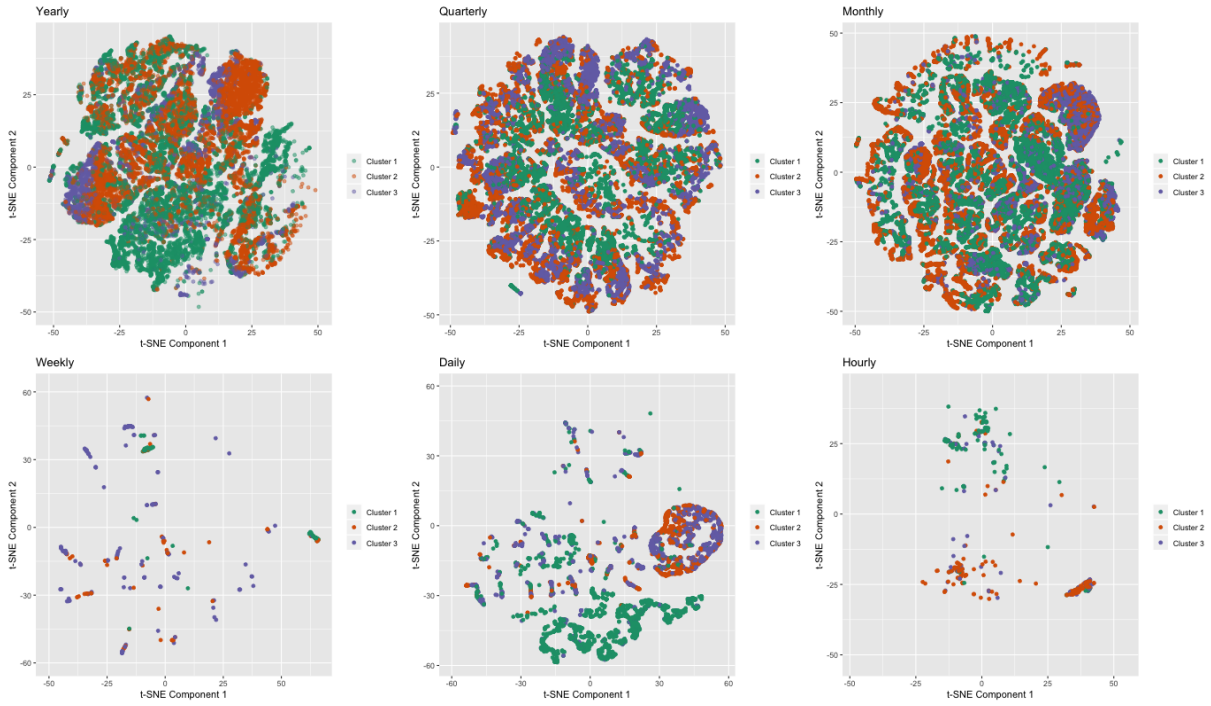

Figure 7: *Location of the three clusters in the instance space. (For interpretation of the references to colour in this figure, the reader is referred to the web version of this article.)*

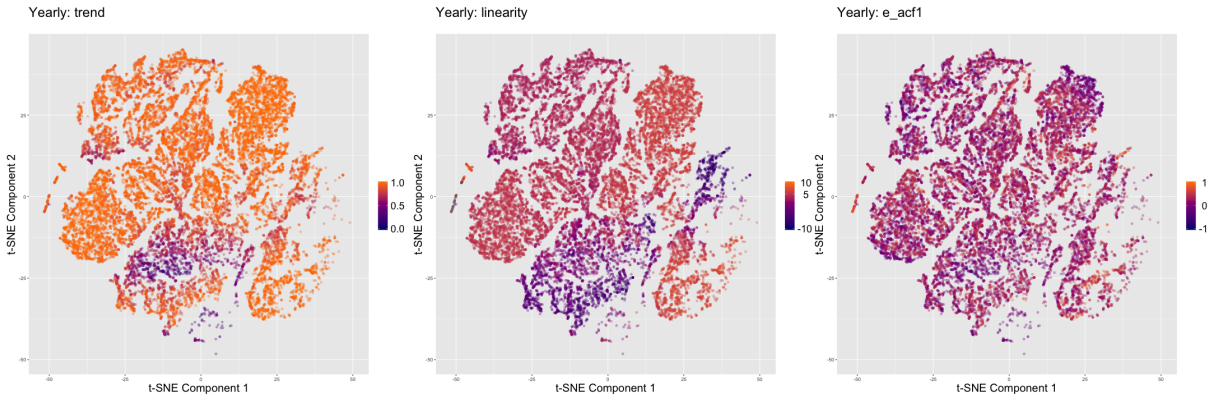

Figure 8: *Distribution of features trend, linearity and e_acf1 across instance space of M4 competition yearly series. Yearly series in cluster 1 (in Figure 7) are mostly less trended and yearly series in cluster 2 are linear and trended. (For interpretation of the references to colour in this figure, the reader is referred to the web version of this article.)*

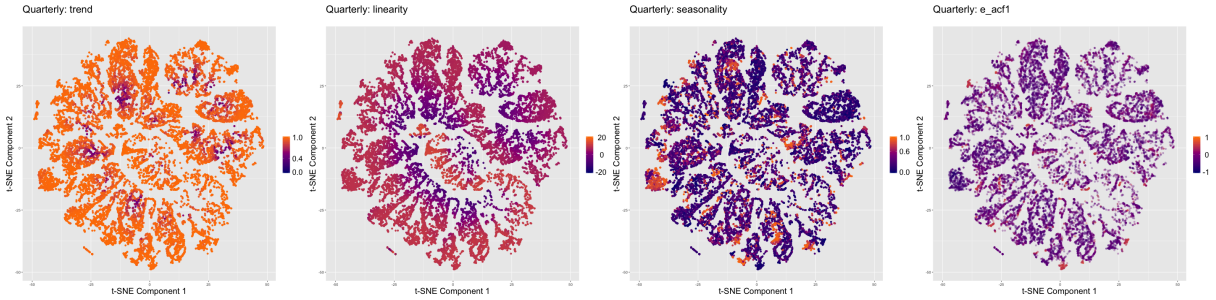

Figure 9: *Distribution of features trend, linearity, seasonality and e_acf1 across instance space of M4 competition quarterly series. Quarterly series in cluster 1 (Figure 7) have low values for linearity. (For interpretation of the references to colour in this figure, the reader is referred to the web version of this article.)*

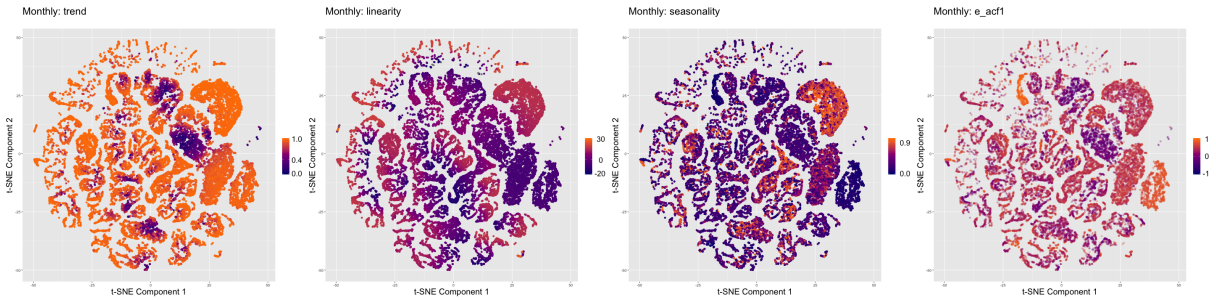

Figure 10: *Distribution of features trend, linearity, seasonality and e_acf1 across instance space of M4 competition monthly series. The locations of monthly time series in cluster 1 (Figure 7) take low values for trend and linearity. (For interpretation of the references to colour in this figure, the reader is referred to the web version of this article.)*

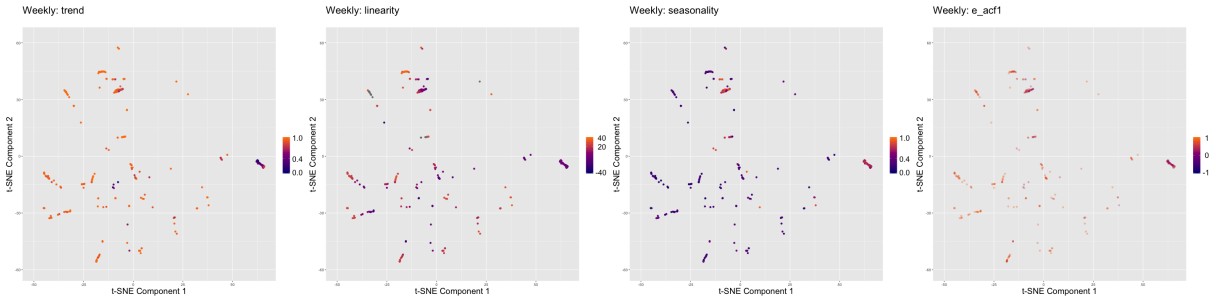

Figure 11: *Distribution of features trend, linearity, seasonality and e_acf1 across instance space of M4 competition weekly series. Most of the highly seasonal time series belong to cluster 1 in Figure 7. (For interpretation of the references to colour in this figure, the reader is referred to the web version of this article.)*

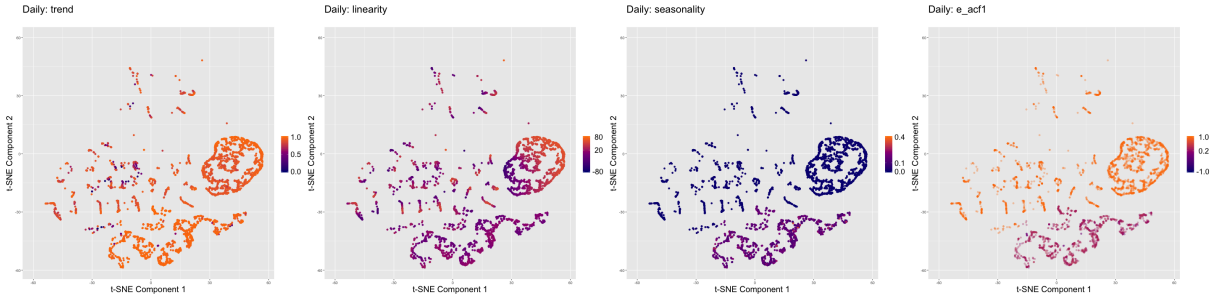

Figure 12: *Distribution of features trend, linearity, seasonality and e_acf1 across instance space of M4 competition daily series. The series in the bottom half of the instance space have high values for strength of seasonality. (For interpretation of the references to colour in this figure, the reader is referred to the web version of this article.)*

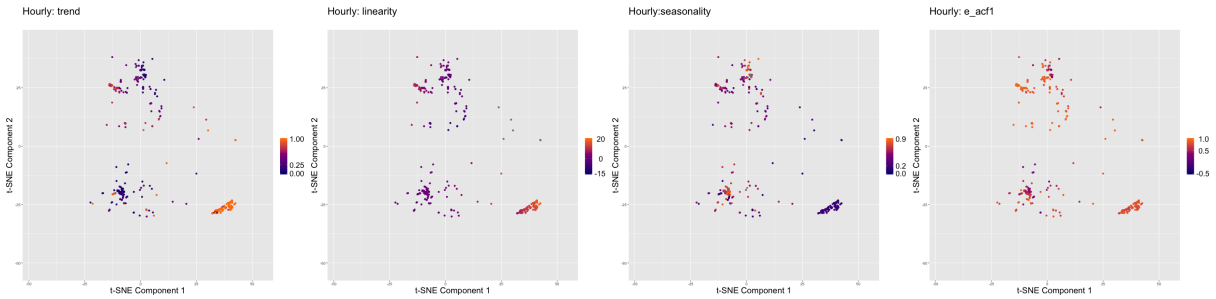

Figure 13: *Distribution of features trend, linearity seasonality and e_acf1 across instance space of M4 competition hourly series. The instances in the lower right corner of the instance space are highly trended and less seasonal. (For interpretation of the references to colour in this figure, the reader is referred to the web version of this article.)*

4 Conclusions

This paper proposes a new meta-learning framework for large-scale time series forecasting. The proposed framework can be used to compute both individual forecasts and combination forecasts. Our results show that features of time series are useful in selecting an optimal subset of models from all individual models without the need to run all possible combinations of

individual models during the online phase. Apart from the obvious utility of this approach for forecast model selection, the ranking of models provides an alternative solution to practitioners who may wish to incorporate their own judgements or expertise into the forecasting decision process. In general, the performance of any model strongly depends on the dataset (reference set) used to train the model. We investigated the feasibility of using the GRATIS approach to increase the diversity of the reference set. This approach is very useful when researchers have a small sample with which to build a reliable classifier, or no sample is available because of data privacy issues. In such circumstance, when applying GRATIS two choices have to be made. This includes: i) the choice of feature values and ii) selection of appropriate values for the parameters of the GRATIS algorithm. The values for the features can be set by randomly selecting values over the theoretical ranges of features (for example, trend [0, 1]) or based on experts' knowledge in the field of application. For parameters, instead of selecting a single value for each parameter the risk of choosing the wrong one can be reduce by selecting a set of multiple values for each parameter. Hence, it is best to generate a collection of series using multiple values for parameters in order to obtain a more diverse and representative collection of series. For each parameter, a multiple set of values can be selected from the distributions explained in Kang, Hyndman & Li (2019). We further explored the instance space defined by features to understand how certain features of the time series are influencing the forecast model selection. A further contribution of the paper is provision of empirical support for the findings of the M4 competition (Makridakis, Spiliotis & Assimakopoulos 2018) that hold that the combination forecasts, in general, outperform the best individual forecasts. An interesting future extension of this framework would be to apply this methodology to producing probabilistic forecasts. The FFORMPP framework is implemented in an R package `fformpp`, which can be downloaded from https://github.com/thiyangt/fformpp.

5 Acknowledgements

Thiyanga S. Talagala's research was supported by the Australian Research Council (ARC) the Centre of Excellence for Mathematical and Statistical Frontiers (ACEMS) and the Central University of Finance and Economics, Beijing, China. Feng Li and Yanfei Kang's research were supported by the National Natural Science Foundation of China (No. 11501587 and No. 11701022, respectively). This research was supported in part by the Monash eResearch Centre and eSolutions-Research Support Services through the use of the MonARCH High Power Computing (HPC) Cluster.

Appendix: Correlation plots

Figure 14: *Yearly: Correlation between MASE values across different forecast models for the series in the reference set*

Figure 15: *Quarterly: Correlation between MASE values across different forecast models for the series in the reference set*

Figure 16: *Monthly: Correlation between MASE values across different forecast models for the series in the reference set*

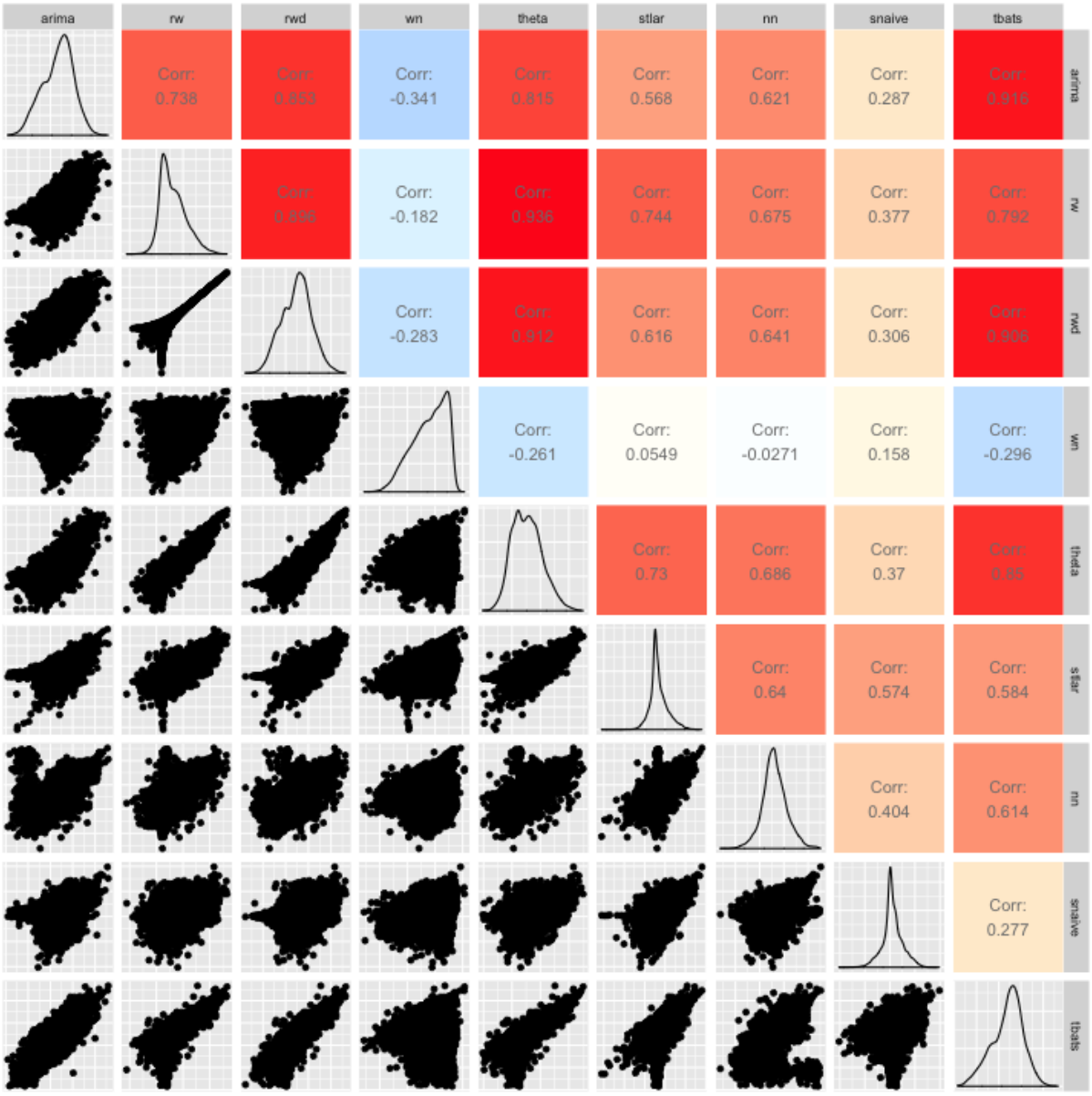

Figure 17: *Weekly: Correlation between MASE values across different forecast models for the series in the reference set*

Figure 18: *Daily: Correlation between MASE values across different forecast models for the series in the reference set*

Figure 19: *Hourly: Correlation between MASE values across different forecast models for the series in the reference set*

References

Adya, M, F Collopy, JS Armstrong & M Kennedy (2001). Automatic identification of time series features for rule-based forecasting. *International Journal of Forecasting* **17**(2), 143–157.

Cleveland, RB, WS Cleveland, JE McRae & I Terpenning (1990). STL: a seasonal-trend decomposition procedure based on loess. *Journal of Official Statistics* **6**(1), 3–73.

Collopy, F & JS Armstrong (1992). Rule-based forecasting: development and validation of an expert systems approach to combining time series extrapolations. *Management Science* **38**(10), 1394–1414.

Hyndman, RJ & AB Koehler (2006). Another look at measures of forecast accuracy. *International Journal of Forecasting* **22**(4), 679–688.

Ingel, A, N Shahroudi, M Kängsepp, A Tättar, V Komisarenko & M Kull (2019). Correlated daily time series and forecasting in the M4 competition. *International Journal of Forecasting*.

Kang, Y, RJ Hyndman & F Li (2019). GRATIS: GeneRAting TIme Series with diverse and controllable characteristics. *arXiv preprint arXiv:1903.02787*.

Kang, Y, RJ Hyndman & K Smith-Miles (2017). Visualising forecasting algorithm performance using time series instance spaces. *International Journal of Forecasting* **33**(2), 345–358.

Li, F & M Villani (2013). Efficient Bayesian multivariate surface regression. *Scandinavian Journal of Statistics* **40**(4), 706–723.

Makridakis, S, E Spiliotis & V Assimakopoulos (2018). The M4 Competition: Results, findings, conclusion and way forward. *International Journal of Forecasting* **34**(4), 802–808.

Meade, N (2000). Evidence for the selection of forecasting methods. *Journal of Forecasting* **19**(6), 515–535.

Montero-Manso, P, RJ Hyndman, G Athanasopoulos & TS Talagala (2019). FFORMA: Feature-based forecast model averaging. *International Journal of Forecasting*, [to appear].

Petropoulos, F, S Makridakis, V Assimakopoulos & K Nikolopoulos (2014). 'Horses for Courses' in demand forecasting. *European Journal of Operational Research* **237**(1), 152–163.

Prudêncio, R & T Ludermir (2004). Using machine learning techniques to combine forecasting methods. In: *Australasian Joint Conference on Artificial Intelligence*. Springer, pp.1122–1127.

Rice, JR (1976). The algorithm selection problem. *Advances in Computers* **15**, 65–118.

Shah, C (1997). Model selection in univariate time series forecasting using discriminant analysis. *International Journal of Forecasting* **13**(4), 489–500.

Spiliotis, E, A Kouloumos, V Assimakopoulos & S Makridakis (2019). Are forecasting competitions data representative of the reality? *International Journal of Forecasting*.

Talagala, TS, RJ Hyndman & G Athanasopoulos (2018). *Meta-learning how to forecast time series*. Working Paper 6/18. Department of Econometrics & Business Statistics, Monash University.

Tashman, LJ & ML Leach (1991). *Automatic forecasting software: A survey and evaluation*.

Wang, X, K Smith-Miles & RJ Hyndman (2009). Rule induction for forecasting method selection: meta-learning the characteristics of univariate time series. *Neurocomputing* **72**(10), 2581–2594.