Journal of Statistical Software

July 2008, Volume 27, Issue 3. *doi: 10.18637/jss.v000.i00*

Automatic Time Series Forecasting: the forecast Package for R

Rob J Hyndman
Monash University

Yeasmin Khandakar
Monash University

Abstract

This vignette to the R package **forecast** is an updated version of Hyndman and Khandakar (2008), published in the *Journal of Statistical Software*. Automatic forecasts of large numbers of univariate time series are often needed in business and other contexts. We describe two automatic forecasting algorithms that have been implemented in the **forecast** package for R. The first is based on innovations state space models that underly exponential smoothing methods. The second is a step-wise algorithm for forecasting with ARIMA models. The algorithms are applicable to both seasonal and non-seasonal data, and are compared and illustrated using four real time series. We also briefly describe some of the other functionality available in the **forecast** package.}

Keywords: ARIMA models, automatic forecasting, exponential smoothing, prediction intervals, state space models, time series, R.

1. Introduction

Automatic forecasts of large numbers of univariate time series are often needed in business. It is common to have over one thousand product lines that need forecasting at least monthly. Even when a smaller number of forecasts are required, there may be nobody suitably trained in the use of time series models to produce them. In these circumstances, an automatic forecasting algorithm is an essential tool. Automatic forecasting algorithms must determine an appropriate time series model, estimate the parameters and compute the forecasts. They must be robust to unusual time series patterns, and applicable to large numbers of series without user intervention. The most popular automatic forecasting algorithms are based on either exponential smoothing or ARIMA models.

In this article, we discuss the implementation of two automatic univariate forecasting methods in the **forecast** package for R. We also briefly describe some univariate forecasting methods

that are part of the **forecast** package.

The **forecast** package for the R system for statistical computing (R Development Core Team 2008) is available from the Comprehensive R Archive Network at `https://CRAN.R-project.org/package=forecast`. Version 8.11 of the package was used for this paper. The **forecast** package contains functions for univariate forecasting and a few examples of real time series data. For more extensive testing of forecasting methods, the **fma** package contains the 90 data sets from Makridakis, Wheelwright, and Hyndman (1998), the **expsmooth** package contains 24 data sets from Hyndman, Koehler, Ord, and Snyder (2008b), and the **Mcomp** package contains the 1001 time series from the M-competition (Makridakis, Anderson, Carbone, Fildes, Hibon, Lewandowski, Newton, Parzen, and Winkler 1982) and the 3003 time series from the M3-competition (Makridakis and Hibon 2000).

The **forecast** package implements automatic forecasting using exponential smoothing, ARIMA models, the Theta method (Assimakopoulos and Nikolopoulos 2000), cubic splines (Hyndman, King, Pitrun, and Billah 2005a), as well as other common forecasting methods. In this article, we primarily discuss the exponential smoothing approach (in Section 2) and the ARIMA modelling approach (in Section 3) to automatic forecasting. In Section 4, we describe the implementation of these methods in the **forecast** package, along with other features of the package.

2. Exponential smoothing

Although exponential smoothing methods have been around since the 1950s, a modelling framework incorporating procedures for model selection was not developed until relatively recently. Ord, Koehler, and Snyder (1997), Hyndman, Koehler, Snyder, and Grose (2002) and Hyndman, Koehler, Ord, and Snyder (2005b) have shown that all exponential smoothing methods (including non-linear methods) are optimal forecasts from innovations state space models.

Exponential smoothing methods were originally classified by Pegels' (1969) taxonomy. This was later extended by Gardner (1985), modified by Hyndman *et al.* (2002), and extended again by Taylor (2003), giving a total of fifteen methods seen in the following table.

Trend Component		Seasonal Component		
		N (None)	A (Additive)	M (Multiplicative)
N	(None)	N,N	N,A	N,M
A	(Additive)	A,N	A,A	A,M
A_d	(Additive damped)	A_d,N	A_d,A	A_d,M
M	(Multiplicative)	M,N	M,A	M,M
M_d	(Multiplicative damped)	M_d,N	M_d,A	M_d,M

Table 1: The fifteen exponential smoothing methods.

Some of these methods are better known under other names. For example, cell (N,N) describes

the simple exponential smoothing (or SES) method, cell (A,N) describes Holt's linear method, and cell (A_d,N) describes the damped trend method. The additive Holt-Winters' method is given by cell (A,A) and the multiplicative Holt-Winters' method is given by cell (A,M). The other cells correspond to less commonly used but analogous methods.

2.1. Point forecasts for all methods

We denote the observed time series by $y_1, y_2, \dots, y_n$. A forecast of y_{t+h} based on all of the data up to time t is denoted by $\hat{y}_{t+h|t}$. To illustrate the method, we give the point forecasts and updating equations for method (A,A), the Holt-Winters' additive method:

$$\begin{aligned}
\text{Level:} \quad & \ell_t = \alpha(y_t - s_{t-m}) + (1-\alpha)(\ell_{t-1} + b_{t-1}) && \text{(1a)}\\
\text{Growth:} \quad & b_t = \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)b_{t-1} && \text{(1b)}\\
\text{Seasonal:} \quad & s_t = \gamma(y_t - \ell_{t-1} - b_{t-1}) + (1-\gamma)s_{t-m} && \text{(1c)}\\
\text{Forecast:} \quad & \hat{y}_{t+h|t} = \ell_t + b_t h + s_{t-m+h_m^+}. && \text{(1d)}
\end{aligned}$$

where m is the length of seasonality (e.g., the number of months or quarters in a year), ℓ_t represents the level of the series, b_t denotes the growth, s_t is the seasonal component, $\hat{y}_{t+h|t}$ is the forecast for h periods ahead, and $h_m^+ = \big[(h-1) \bmod m\big] + 1$. To use method (1), we need values for the initial states ℓ_0, b_0 and $s_{1-m}, \dots, s_0$, and for the smoothing parameters α, β^* and γ. All of these will be estimated from the observed data.

Equation (1c) is slightly different from the usual Holt-Winters equations such as those in Makridakis *et al.* (1998) or Bowerman, O'Connell, and Koehler (2005). These authors replace (1c) with

$$s_t = \gamma^*(y_t - \ell_t) + (1-\gamma^*)s_{t-m}.$$

If ℓ_t is substituted using (1a), we obtain

$$s_t = \gamma^*(1-\alpha)(y_t - \ell_{t-1} - b_{t-1}) + \{1 - \gamma^*(1-\alpha)\}s_{t-m}.$$

Thus, we obtain identical forecasts using this approach by replacing γ in (1c) with $\gamma^*(1-\alpha)$. The modification given in (1c) was proposed by Ord *et al.* (1997) to make the state space formulation simpler. It is equivalent to Archibald's (1990) variation of the Holt-Winters' method.

Table 2 gives recursive formulae for computing point forecasts h periods ahead for all of the exponential smoothing methods. Some interesting special cases can be obtained by setting the smoothing parameters to extreme values. For example, if $\alpha = 0$, the level is constant over time; if $\beta^* = 0$, the slope is constant over time; and if $\gamma = 0$, the seasonal pattern is constant over time. At the other extreme, naïve forecasts (i.e., $\hat{y}_{t+h|t} = y_t$ for all h) are obtained using the (N,N) method with $\alpha = 1$. Finally, the additive and multiplicative trend methods are special cases of their damped counterparts obtained by letting $\phi = 1$.

2.2. Innovations state space models

For each exponential smoothing method in Table 2, Hyndman *et al.* (2008b) describe two possible innovations state space models, one corresponding to a model with additive errors and the other to a model with multiplicative errors. If the same parameter values are used,

Trend	Seasonal		
	N	A	M
N	$\ell_t = \alpha y_t + (1-\alpha)\ell_{t-1}$	$\ell_t = \alpha(y_t - s_{t-m}) + (1-\alpha)\ell_{t-1}$	$\ell_t = \alpha(y_t/s_{t-m}) + (1-\alpha)\ell_{t-1}$
		$s_t = \gamma(y_t - \ell_{t-1}) + (1-\gamma)s_{t-m}$	$s_t = \gamma(y_t/\ell_{t-1}) + (1-\gamma)s_{t-m}$
	$\hat{y}_{t+h\|t} = \ell_t$	$\hat{y}_{t+h\|t} = \ell_t + s_{t-m+h_m^+}$	$\hat{y}_{t+h\|t} = \ell_t s_{t-m+h_m^+}$
A	$\ell_t = \alpha y_t + (1-\alpha)(\ell_{t-1} + b_{t-1})$	$\ell_t = \alpha(y_t - s_{t-m}) + (1-\alpha)(\ell_{t-1} + b_{t-1})$	$\ell_t = \alpha(y_t/s_{t-m}) + (1-\alpha)(\ell_{t-1} + b_{t-1})$
	$b_t = \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)b_{t-1}$	$b_t = \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)b_{t-1}$	$b_t = \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)b_{t-1}$
		$s_t = \gamma(y_t - \ell_{t-1} - b_{t-1}) + (1-\gamma)s_{t-m}$	$s_t = \gamma(y_t/(\ell_{t-1} - b_{t-1})) + (1-\gamma)s_{t-m}$
	$\hat{y}_{t+h\|t} = \ell_t + hb_t$	$\hat{y}_{t+h\|t} = \ell_t + hb_t + s_{t-m+h_m^+}$	$\hat{y}_{t+h\|t} = (\ell_t + hb_t)s_{t-m+h_m^+}$
$\mathrm{A_d}$	$\ell_t = \alpha y_t + (1-\alpha)(\ell_{t-1} + \phi b_{t-1})$	$\ell_t = \alpha(y_t - s_{t-m}) + (1-\alpha)(\ell_{t-1} + \phi b_{t-1})$	$\ell_t = \alpha(y_t/s_{t-m}) + (1-\alpha)(\ell_{t-1} + \phi b_{t-1})$
	$b_t = \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)\phi b_{t-1}$	$b_t = \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)\phi b_{t-1}$	$b_t = \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)\phi b_{t-1}$
		$s_t = \gamma(y_t - \ell_{t-1} - \phi b_{t-1}) + (1-\gamma)s_{t-m}$	$s_t = \gamma(y_t/(\ell_{t-1} - \phi b_{t-1})) + (1-\gamma)s_{t-m}$
	$\hat{y}_{t+h\|t} = \ell_t + \phi_h b_t$	$\hat{y}_{t+h\|t} = \ell_t + \phi_h b_t + s_{t-m+h_m^+}$	$\hat{y}_{t+h\|t} = (\ell_t + \phi_h b_t)s_{t-m+h_m^+}$
M	$\ell_t = \alpha y_t + (1-\alpha)\ell_{t-1}b_{t-1}$	$\ell_t = \alpha(y_t - s_{t-m}) + (1-\alpha)\ell_{t-1}b_{t-1}$	$\ell_t = \alpha(y_t/s_{t-m}) + (1-\alpha)\ell_{t-1}b_{t-1}$
	$b_t = \beta^*(\ell_t/\ell_{t-1}) + (1-\beta^*)b_{t-1}$	$b_t = \beta^*(\ell_t/\ell_{t-1}) + (1-\beta^*)b_{t-1}$	$b_t = \beta^*(\ell_t/\ell_{t-1}) + (1-\beta^*)b_{t-1}$
		$s_t = \gamma(y_t - \ell_{t-1}b_{t-1}) + (1-\gamma)s_{t-m}$	$s_t = \gamma(y_t/(\ell_{t-1}b_{t-1})) + (1-\gamma)s_{t-m}$
	$\hat{y}_{t+h\|t} = \ell_t b_t^h$	$\hat{y}_{t+h\|t} = \ell_t b_t^h + s_{t-m+h_m^+}$	$\hat{y}_{t+h\|t} = \ell_t b_t^h s_{t-m+h_m^+}$
$\mathrm{M_d}$	$\ell_t = \alpha y_t + (1-\alpha)\ell_{t-1}b_{t-1}^{\phi}$	$\ell_t = \alpha(y_t - s_{t-m}) + (1-\alpha)\ell_{t-1}b_{t-1}^{\phi}$	$\ell_t = \alpha(y_t/s_{t-m}) + (1-\alpha)\ell_{t-1}b_{t-1}^{\phi}$
	$b_t = \beta^*(\ell_t/\ell_{t-1}) + (1-\beta^*)b_{t-1}^{\phi}$	$b_t = \beta^*(\ell_t/\ell_{t-1}) + (1-\beta^*)b_{t-1}^{\phi}$	$b_t = \beta^*(\ell_t/\ell_{t-1}) + (1-\beta^*)b_{t-1}^{\phi}$
		$s_t = \gamma(y_t - \ell_{t-1}b_{t-1}^{\phi}) + (1-\gamma)s_{t-m}$	$s_t = \gamma(y_t/(\ell_{t-1}b_{t-1}^{\phi})) + (1-\gamma)s_{t-m}$
	$\hat{y}_{t+h\|t} = \ell_t b_t^{\phi_h}$	$\hat{y}_{t+h\|t} = \ell_t b_t^{\phi_h} + s_{t-m+h_m^+}$	$\hat{y}_{t+h\|t} = \ell_t b_t^{\phi_h} s_{t-m+h_m^+}$

Table 2: Formulae for recursive calculations and point forecasts. In each case, ℓ_t denotes the series level at time t, b_t denotes the slope at time t, s_t denotes the seasonal component of the series at time t, and m denotes the number of seasons in a year; α, β^*, γ and ϕ are constants, $\phi_h = \phi + \phi^2 + \cdots + \phi^h$ and $h_m^+ = \big[(h-1) \bmod m\big] + 1$.

these two models give equivalent point forecasts, although different prediction intervals. Thus there are 30 potential models described in this classification.

Historically, the nature of the error component has often been ignored, because the distinction between additive and multiplicative errors makes no difference to point forecasts.

We are careful to distinguish exponential smoothing *methods* from the underlying state space *models*. An exponential smoothing method is an algorithm for producing point forecasts only. The underlying stochastic state space model gives the same point forecasts, but also provides a framework for computing prediction intervals and other properties.

To distinguish the models with additive and multiplicative errors, we add an extra letter to the front of the method notation. The triplet (E,T,S) refers to the three components: error, trend and seasonality. So the model ETS(A,A,N) has additive errors, additive trend and no seasonality—in other words, this is Holt's linear method with additive errors. Similarly, ETS(M,$\mathrm{M_d}$,M) refers to a model with multiplicative errors, a damped multiplicative trend and multiplicative seasonality. The notation ETS($\cdot$,$\cdot$,$\cdot$) helps in remembering the order in which the components are specified.

Once a model is specified, we can study the probability distribution of future values of the series and find, for example, the conditional mean of a future observation given knowledge of the past. We denote this as $\mu_{t+h|t} = \mathsf{E}(y_{t+h} \mid \boldsymbol{x}_t)$, where $\boldsymbol{x}_t$ contains the unobserved components such as ℓ_t, b_t and s_t. For $h = 1$ we use $\mu_t \equiv \mu_{t+1|t}$ as a shorthand notation. For many models, these conditional means will be identical to the point forecasts given in Table 2, so that $\mu_{t+h|t} = \hat{y}_{t+h|t}$. However, for other models (those with multiplicative trend or multiplicative seasonality), the conditional mean and the point forecast will differ slightly for $h \geq 2$.

We illustrate these ideas using the damped trend method of Gardner and McKenzie (1985).

Additive error model: ETS(A,A_d,N)

Let $\mu_t = \hat{y}_t = \ell_{t-1} + b_{t-1}$ denote the one-step forecast of y_t assuming that we know the values of all parameters. Also, let $\varepsilon_t = y_t - \mu_t$ denote the one-step forecast error at time t. From the equations in Table 2, we find that

$$y_t = \ell_{t-1} + \phi b_{t-1} + \varepsilon_t \tag{2}$$

$$\ell_t = \ell_{t-1} + \phi b_{t-1} + \alpha\varepsilon_t \tag{3}$$

$$b_t = \phi b_{t-1} + \beta^*(\ell_t - \ell_{t-1} - \phi b_{t-1}) = \phi b_{t-1} + \alpha\beta^*\varepsilon_t. \tag{4}$$

We simplify the last expression by setting $\beta = \alpha\beta^*$. The three equations above constitute a state space model underlying the damped Holt's method. Note that it is an *innovations* state space model (Anderson and Moore 1979; Aoki 1987) because the same error term appears in each equation. We an write it in standard state space notation by defining the state vector as $\boldsymbol{x}_t = (\ell_t, b_t)'$ and expressing (2)–(4) as

$$y_t = [1\phi]\,\boldsymbol{x}_{t-1} + \varepsilon_t \tag{5a}$$

$$\boldsymbol{x}_t = \begin{bmatrix} 1 & \phi \\ 0 & \phi \end{bmatrix} \boldsymbol{x}_{t-1} + \begin{bmatrix} \alpha \\ \beta \end{bmatrix} \varepsilon_t. \tag{5b}$$

The model is fully specified once we state the distribution of the error term ε_t. Usually we assume that these are independent and identically distributed, following a normal distribution with mean 0 and variance σ^2, which we write as $\varepsilon_t \sim \mathrm{NID}(0, \sigma^2)$.

Multiplicative error model: ETS(M,A_d,N)

A model with multiplicative error can be derived similarly, by first setting $\varepsilon_t = (y_t - \mu_t)/\mu_t$, so that ε_t is the relative error. Then, following a similar approach to that for additive errors, we find

$$\begin{aligned} y_t &= (\ell_{t-1} + \phi b_{t-1})(1 + \varepsilon_t) \\ \ell_t &= (\ell_{t-1} + \phi b_{t-1})(1 + \alpha\varepsilon_t) \\ b_t &= \phi b_{t-1} + \beta(\ell_{t-1} + \phi b_{t-1})\varepsilon_t, \end{aligned}$$

or

$$\begin{aligned} y_t &= [1\phi]\, \boldsymbol{x}_{t-1}(1 + \varepsilon_t) \\ \boldsymbol{x}_t &= \begin{bmatrix} 1 & \phi \\ 0 & \phi \end{bmatrix} \boldsymbol{x}_{t-1} + [1\phi]\, \boldsymbol{x}_{t-1} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} \varepsilon_t. \end{aligned}$$

Again we assume that $\varepsilon_t \sim \text{NID}(0, \sigma^2)$.

Of course, this is a nonlinear state space model, which is usually considered difficult to handle in estimating and forecasting. However, that is one of the many advantages of the innovations form of state space models — we can still compute forecasts, the likelihood and prediction intervals for this nonlinear model with no more effort than is required for the additive error model.

2.3. State space models for all exponential smoothing methods

There are similar state space models for all 30 exponential smoothing variations. The general model involves a state vector $\boldsymbol{x}_t = (\ell_t, b_t, s_t, s_{t-1}, \ldots, s_{t-m+1})'$ and state space equations of the form

$$y_t = w(\boldsymbol{x}_{t-1}) + r(\boldsymbol{x}_{t-1})\varepsilon_t \tag{6a}$$

$$\boldsymbol{x}_t = f(\boldsymbol{x}_{t-1}) + g(\boldsymbol{x}_{t-1})\varepsilon_t \tag{6b}$$

where $\{\varepsilon_t\}$ is a Gaussian white noise process with mean zero and variance σ^2, and $\mu_t = w(\boldsymbol{x}_{t-1})$. The model with additive errors has $r(\boldsymbol{x}_{t-1}) = 1$, so that $y_t = \mu_t + \varepsilon_t$. The model with multiplicative errors has $r(\boldsymbol{x}_{t-1}) = \mu_t$, so that $y_t = \mu_t(1 + \varepsilon_t)$. Thus, $\varepsilon_t = (y_t - \mu_t)/\mu_t$ is the relative error for the multiplicative model. The models are not unique. Clearly, any value of $r(\boldsymbol{x}_{t-1})$ will lead to identical point forecasts for y_t.

All of the methods in Table 2 can be written in the form (6a) and (6b). The specific form for each model is given in Hyndman *et al.* (2008b).

Some of the combinations of trend, seasonality and error can occasionally lead to numerical difficulties; specifically, any model equation that requires division by a state component could involve division by zero. This is a problem for models with additive errors and either multiplicative trend or multiplicative seasonality, as well as for the model with multiplicative errors, multiplicative trend and additive seasonality. These models should therefore be used with caution.

The multiplicative error models are useful when the data are strictly positive, but are not numerically stable when the data contain zeros or negative values. So when the time series is not strictly positive, only the six fully additive models may be applied.

The point forecasts given in Table 2 are easily obtained from these models by iterating equations (6a) and (6b) for $t = n+1, n+2, \ldots, n+h$, setting $\varepsilon_{n+j} = 0$ for $j = 1, \ldots, h$. In most cases (notable exceptions being models with multiplicative seasonality or multiplicative trend for $h \geq 2$), the point forecasts can be shown to be equal to $\mu_{t+h|t} = \mathsf{E}(y_{t+h} \mid \boldsymbol{x}_t)$, the conditional expectation of the corresponding state space model.

The models also provide a means of obtaining prediction intervals. In the case of the linear models, where the forecast distributions are normal, we can derive the conditional variance $v_{t+h|t} = \mathsf{VAR}(y_{t+h} \mid \boldsymbol{x}_t)$ and obtain prediction intervals accordingly. This approach also works for many of the nonlinear models. Detailed derivations of the results for many models are given in Hyndman *et al.* (2005b).

A more direct approach that works for all of the models is to simply simulate many future sample paths conditional on the last estimate of the state vector, $\boldsymbol{x}_t$. Then prediction intervals can be obtained from the percentiles of the simulated sample paths. Point forecasts can also be obtained in this way by taking the average of the simulated values at each future time period. An advantage of this approach is that we generate an estimate of the complete predictive distribution, which is especially useful in applications such as inventory planning, where expected costs depend on the whole distribution.

2.4. Estimation

In order to use these models for forecasting, we need to know the values of $\boldsymbol{x}_0$ and the parameters α, β, γ and ϕ. It is easy to compute the likelihood of the innovations state space model (6), and so obtain maximum likelihood estimates. Ord *et al.* (1997) show that

$$L^*(\boldsymbol{\theta}, \boldsymbol{x}_0) = n \log\left(\sum_{t=1}^{n} \varepsilon_t^2\right) + 2\sum_{t=1}^{n} \log|r(\boldsymbol{x}_{t-1})| \tag{7}$$

is equal to twice the negative logarithm of the likelihood function (with constant terms eliminated), conditional on the parameters $\boldsymbol{\theta} = (\alpha, \beta, \gamma, \phi)'$ and the initial states $\boldsymbol{x}_0 = (\ell_0, b_0, s_0, s_{-1}, \ldots, s_{-m+1})'$, where n is the number of observations. This is easily computed by simply using the recursive equations in Table 2. Unlike state space models with multiple sources of error, we do not need to use the Kalman filter to compute the likelihood.

The parameters $\boldsymbol{\theta}$ and the initial states $\boldsymbol{x}_0$ can be estimated by minimizing L^*. Most implementations of exponential smoothing use an ad hoc heuristic scheme to estimate $\boldsymbol{x}_0$. However, with modern computers, there is no reason why we cannot estimate $\boldsymbol{x}_0$ along with $\boldsymbol{\theta}$, and the resulting forecasts are often substantially better when we do.

We constrain the initial states $\boldsymbol{x}_0$ so that the seasonal indices add to zero for additive seasonality, and add to m for multiplicative seasonality. There have been several suggestions for restricting the parameter space for α, β and γ. The traditional approach is to ensure that the various equations can be interpreted as weighted averages, thus requiring α, $\beta^* = \beta/\alpha$, $\gamma^* = \gamma/(1-\alpha)$ and ϕ to all lie within $(0, 1)$. This suggests

$$0 < \alpha < 1, \qquad 0 < \beta < \alpha, \qquad 0 < \gamma < 1 - \alpha, \qquad \text{and} \qquad 0 < \phi < 1.$$

However, Hyndman, Akram, and Archibald (2008a) show that these restrictions are usually stricter than necessary (although in a few cases they are not restrictive enough).

2.5. Model selection

Forecast accuracy measures such as mean squared error (MSE) can be used for selecting a model for a given set of data, provided the errors are computed from data in a hold-out set and not from the same data as were used for model estimation. However, there are often too few out-of-sample errors to draw reliable conclusions. Consequently, a penalized method based on the in-sample fit is usually better.

One such approach uses a penalized likelihood such as Akaike's Information Criterion:

$$\text{AIC} = L^*(\hat{\boldsymbol{\theta}}, \hat{\boldsymbol{x}}_0) + 2q,$$

where q is the number of parameters in $\boldsymbol{\theta}$ plus the number of free states in $\boldsymbol{x}_0$, and $\hat{\boldsymbol{\theta}}$ and $\hat{\boldsymbol{x}}_0$ denote the estimates of $\boldsymbol{\theta}$ and $\boldsymbol{x}_0$. We select the model that minimizes the AIC amongst all of the models that are appropriate for the data.

The AIC also provides a method for selecting between the additive and multiplicative error models. The point forecasts from the two models are identical so that standard forecast accuracy measures such as the MSE or mean absolute percentage error (MAPE) are unable to select between the error types. The AIC is able to select between the error types because it is based on likelihood rather than one-step forecasts.

Obviously, other model selection criteria (such as the BIC) could also be used in a similar manner.

2.6. Automatic forecasting

We combine the preceding ideas to obtain a robust and widely applicable automatic forecasting algorithm. The steps involved are summarized below.

1. For each series, apply all models that are appropriate, optimizing the parameters (both smoothing parameters and the initial state variable) of the model in each case.
2. Select the best of the models according to the AIC.
3. Produce point forecasts using the best model (with optimized parameters) for as many steps ahead as required.
4. Obtain prediction intervals for the best model either using the analytical results of Hyndman, Koehler, et al. (2005), or by simulating future sample paths for $\{y_{n+1}, \dots, y_{n+h}\}$ and finding the $\alpha/2$ and $1-\alpha/2$ percentiles of the simulated data at each forecasting horizon. If simulation is used, the sample paths may be generated using the normal distribution for errors (parametric bootstrap) or using the resampled errors (ordinary bootstrap).

Hyndman *et al.* (2002) applied this automatic forecasting strategy to the M-competition data (Makridakis *et al.* 1982) and the IJF-M3 competition data (Makridakis and Hibon 2000) using a restricted set of exponential smoothing models, and demonstrated that the methodology is particularly good at short term forecasts (up to about 6 periods ahead), and especially for seasonal short-term series (beating all other methods in the competitions for these series).

3. ARIMA models

A common obstacle for many people in using Autoregressive Integrated Moving Average (ARIMA) models for forecasting is that the order selection process is usually considered

subjective and difficult to apply. But it does not have to be. There have been several attempts to automate ARIMA modelling in the last 25 years.

Hannan and Rissanen (1982) proposed a method to identify the order of an ARMA model for a stationary series. In their method the innovations can be obtained by fitting a long autoregressive model to the data, and then the likelihood of potential models is computed via a series of standard regressions. They established the asymptotic properties of the procedure under very general conditions.

Gómez (1998) extended the Hannan-Rissanen identification method to include multiplicative seasonal ARIMA model identification. Gómez and Maravall (1998) implemented this automatic identification procedure in the software **TRAMO** and **SEATS**. For a given series, the algorithm attempts to find the model with the minimum BIC.

Liu (1989) proposed a method for identification of seasonal ARIMA models using a filtering method and certain heuristic rules; this algorithm is used in the **SCA-Expert** software. Another approach is described by Mélard and Pasteels (2000) whose algorithm for univariate ARIMA models also allows intervention analysis. It is implemented in the software package "Time Series Expert" (**TSE-AX**).

Other algorithms are in use in commercial software, although they are not documented in the public domain literature. In particular, **Forecast Pro** (Goodrich 2000) is well-known for its excellent automatic ARIMA algorithm which was used in the M3-forecasting competition (Makridakis and Hibon 2000). Another proprietary algorithm is implemented in **Autobox** (Reilly 2000). Ord and Lowe (1996) provide an early review of some of the commercial software that implement automatic ARIMA forecasting.

3.1. Choosing the model order using unit root tests and the AIC

A non-seasonal ARIMA(p, d, q) process is given by

$$\phi(B)(1 - B^d)y_t = c + \theta(B)\varepsilon_t$$

where $\{\varepsilon_t\}$ is a white noise process with mean zero and variance σ^2, B is the backshift operator, and $\phi(z)$ and $\theta(z)$ are polynomials of order p and q respectively. To ensure causality and invertibility, it is assumed that $\phi(z)$ and $\theta(z)$ have no roots for $|z| < 1$ (Brockwell and Davis 1991). If $c \neq 0$, there is an implied polynomial of order d in the forecast function.

The seasonal ARIMA$(p, d, q)(P, D, Q)_m$ process is given by

$$\Phi(B^m)\phi(B)(1 - B^m)^D(1 - B)^d y_t = c + \Theta(B^m)\theta(B)\varepsilon_t$$

where $\Phi(z)$ and $\Theta(z)$ are polynomials of orders P and Q respectively, each containing no roots inside the unit circle. If $c \neq 0$, there is an implied polynomial of order $d + D$ in the forecast function.

The main task in automatic ARIMA forecasting is selecting an appropriate model order, that is the values p, q, P, Q, D, d. If d and D are known, we can select the orders p, q, P and Q via an information criterion such as the AIC:

$$\text{AIC} = -2\log(L) + 2(p + q + P + Q + k)$$

where $k = 1$ if $c \neq 0$ and 0 otherwise, and L is the maximized likelihood of the model fitted to the *differenced* data $(1-B^m)^D(1-B)^d y_t$. The likelihood of the full model for y_t is not actually defined and so the value of the AIC for different levels of differencing are not comparable.

One solution to this difficulty is the "diffuse prior" approach which is outlined in Durbin and Koopman (2001) and implemented in the `arima()` function (Ripley 2002) in R. In this approach, the initial values of the time series (before the observed values) are assumed to have mean zero and a large variance. However, choosing d and D by minimizing the AIC using this approach tends to lead to over-differencing. For forecasting purposes, we believe it is better to make as few differences as possible because over-differencing harms forecasts (Smith and Yadav 1994) and widens prediction intervals. (Although, see Hendry 1997, for a contrary view.)

Consequently, we need some other approach to choose d and D. We prefer unit-root tests. However, most unit-root tests are based on a null hypothesis that a unit root exists which biases results towards more differences rather than fewer differences. For example, variations on the Dickey-Fuller test (Dickey and Fuller 1981) all assume there is a unit root at lag 1, and the HEGY test of Hylleberg, Engle, Granger, and Yoo (1990) is based on a null hypothesis that there is a seasonal unit root. Instead, we prefer unit-root tests based on a null hypothesis of no unit-root.

For non-seasonal data, we consider ARIMA(p, d, q) models where d is selected based on successive KPSS unit-root tests (Kwiatkowski, Phillips, Schmidt, and Shin 1992). That is, we test the data for a unit root; if the test result is significant, we test the differenced data for a unit root; and so on. We stop this procedure when we obtain our first insignificant result.

For seasonal data, we consider ARIMA$(p, d, q)(P, D, Q)_m$ models where m is the seasonal frequency and $D = 0$ or $D = 1$ depending on an extended Canova-Hansen test (Canova and Hansen 1995). Canova and Hansen only provide critical values for $2 < m < 13$. In our implementation of their test, we allow any value of $m > 1$. Let C_m be the critical value for seasonal period m. We plotted C_m against m for values of m up to 365 and noted that they fit the line $C_m = 0.269m^{0.928}$ almost exactly. So for $m > 12$, we use this simple expression to obtain the critical value.

We note in passing that the null hypothesis for the Canova-Hansen test is not an ARIMA model as it includes seasonal dummy terms. It is a test for whether the seasonal pattern changes sufficiently over time to warrant a seasonal unit root, or whether a stable seasonal pattern modelled using fixed dummy variables is more appropriate. Nevertheless, we have found that the test is still useful for choosing D in a strictly ARIMA framework (i.e., without seasonal dummy variables). If a stable seasonal pattern is selected (i.e., the null hypothesis is not rejected), the seasonality is effectively handled by stationary seasonal AR and MA terms.

After D is selected, we choose d by applying successive KPSS unit-root tests to the seasonally differenced data (if $D = 1$) or the original data (if $D = 0$). Once d (and possibly D) are selected, we proceed to select the values of p, q, P and Q by minimizing the AIC. We allow $c \neq 0$ for models where $d + D < 2$.

3.2. A step-wise procedure for traversing the model space

Suppose we have seasonal data and we consider ARIMA$(p, d, q)(P, D, Q)_m$ models where p and q can take values from 0 to 3, and P and Q can take values from 0 to 1. When $c = 0$ there is a total of 288 possible models, and when $c \neq 0$ there is a total of 192 possible models, giving 480 models altogether. If the values of p, d, q, P, D and Q are allowed to range more widely, the number of possible models increases rapidly. Consequently, it is often not feasible to simply fit every potential model and choose the one with the lowest AIC. Instead, we need

a way of traversing the space of models efficiently in order to arrive at the model with the lowest AIC value.

We propose a step-wise algorithm as follows.

Step 1: We try four possible models to start with.

- ARIMA$(2, d, 2)$ if $m = 1$ and ARIMA$(2, d, 2)(1, D, 1)$ if $m > 1$.
- ARIMA$(0, d, 0)$ if $m = 1$ and ARIMA$(0, d, 0)(0, D, 0)$ if $m > 1$.
- ARIMA$(1, d, 0)$ if $m = 1$ and ARIMA$(1, d, 0)(1, D, 0)$ if $m > 1$.
- ARIMA$(0, d, 1)$ if $m = 1$ and ARIMA$(0, d, 1)(0, D, 1)$ if $m > 1$.

If $d + D \leq 1$, these models are fitted with $c \neq 0$. Otherwise, we set $c = 0$. Of these four models, we select the one with the smallest AIC value. This is called the "current" model and is denoted by ARIMA(p, d, q) if $m = 1$ or ARIMA$(p, d, q)(P, D, Q)_m$ if $m > 1$.

Step 2: We consider up to seventeen variations on the current model:

- where one of p, q, P and Q is allowed to vary by ± 1 from the current model;
- where p and q both vary by ± 1 from the current model;
- where P and Q both vary by ± 1 from the current model;
- where the constant c is included if the current model has $c = 0$ or excluded if the current model has $c \neq 0$.

Whenever a model with lower AIC is found, it becomes the new "current" model and the procedure is repeated. This process finishes when we cannot find a model close to the current model with lower AIC.

There are several constraints on the fitted models to avoid problems with convergence or near unit-roots. The constraints are outlined below.

- The values of p and q are not allowed to exceed specified upper bounds (with default values of 5 in each case).
- The values of P and Q are not allowed to exceed specified upper bounds (with default values of 2 in each case).
- We reject any model which is "close" to non-invertible or non-causal. Specifically, we compute the roots of $\phi(B)\Phi(B)$ and $\theta(B)\Theta(B)$. If either have a root that is smaller than 1.001 in absolute value, the model is rejected.
- If there are any errors arising in the non-linear optimization routine used for estimation, the model is rejected. The rationale here is that any model that is difficult to fit is probably not a good model for the data.

The algorithm is guaranteed to return a valid model because the model space is finite and at least one of the starting models will be accepted (the model with no AR or MA parameters). The selected model is used to produce forecasts.

3.3. Comparisons with exponential smoothing

There is a widespread myth that ARIMA models are more general than exponential smoothing. This is not true. The two classes of models overlap. The linear exponential smoothing models are all special cases of ARIMA models—the equivalences are discussed in Hyndman *et al.* (2008a). However, the non-linear exponential smoothing models have no equivalent ARIMA counterpart. On the other hand, there are many ARIMA models which have no exponential smoothing counterpart. Thus, the two model classes overlap and are complimentary; each has its strengths and weaknesses.

The exponential smoothing state space models are all non-stationary. Models with seasonality or non-damped trend (or both) have two unit roots; all other models—that is, non-seasonal models with either no trend or damped trend—have one unit root. It is possible to define a stationary model with similar characteristics to exponential smoothing, but this is not normally done. The philosophy of exponential smoothing is that the world is non-stationary. So if a stationary model is required, ARIMA models are better.

One advantage of the exponential smoothing models is that they can be non-linear. So time series that exhibit non-linear characteristics including heteroscedasticity may be better modelled using exponential smoothing state space models.

For seasonal data, there are many more ARIMA models than the 30 possible models in the exponential smoothing class of Section 2. It may be thought that the larger model class is advantageous. However, the results in Hyndman *et al.* (2002) show that the exponential smoothing models performed better than the ARIMA models for the seasonal M3 competition data. (For the annual M3 data, the ARIMA models performed better.) In a discussion of these results, Hyndman (2001) speculates that the larger model space of ARIMA models actually harms forecasting performance because it introduces additional uncertainty. The smaller exponential smoothing class is sufficiently rich to capture the dynamics of almost all real business and economic time series.

4. The forecast package

The algorithms and modelling frameworks for automatic univariate time series forecasting are implemented in the **forecast** package in R. We illustrate the methods using the following four real time series shown in Figure 1.

- Figure 1(a) shows 125 monthly US government bond yields (percent per annum) from January 1994 to May 2004.
- Figure 1(b) displays 55 observations of annual US net electricity generation (billion kwh) for 1949 through 2003.
- Figure 1(c) presents 113 quarterly observations of passenger motor vehicle production in the U.K. (thousands of cars) for the first quarter of 1977 through the first quarter of 2005.
- Figure 1(d) shows 240 monthly observations of the number of short term overseas visitors to Australia from May 1985 to April 2005.

4.1. Implementation of the automatic exponential smoothing algorithm

The innovations state space modelling framework described in Section 2 is implemented via the `ets()` function in the **forecast** package. (The default settings of `ets()` do not allow models with multiplicative trend, but they can be included using `allow.multiplicative.trend=TRUE`.)

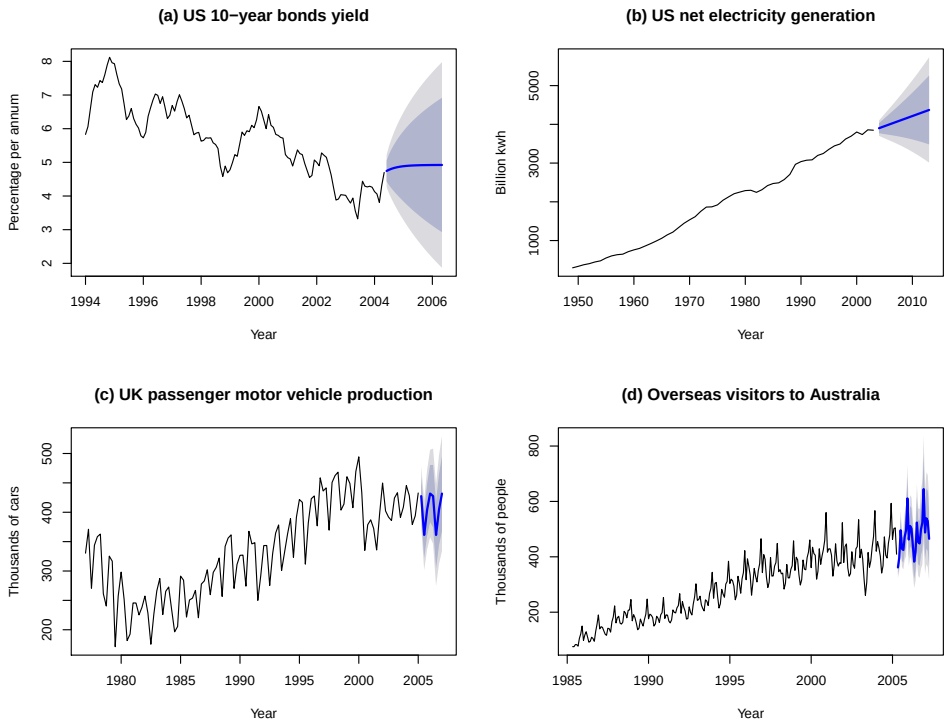

Figure 1: Four time series showing point forecasts and 80% & 95% prediction intervals obtained using exponential smoothing state space models.

The models chosen via the algorithm for the four data sets were:

- ETS(A,A_d,N) for monthly US 10-year bonds yield
  ($\alpha = 0.9999$, $\beta = 0.09545$, $\phi = 0.8026$, $\ell_0 = 5.3252$, $b_0 = 0.5934$);
- ETS(M,A,N) for annual US net electricity generation
  ($\alpha = 0.9999$, $\beta = 0.2191$, $\ell_0 = 254.9338$, $b_0 = 38.3125$);
- ETS(A,N,A) for quarterly UK motor vehicle production
  ($\alpha = 0.6199$, $\gamma = 1e{-}04$, $\ell_0 = 314.2568$, $s_{-3} = 25.5223$, $s_{-2} = 21.1956$, $s_{-1} = -44.9601$, $s_0 = -1.7579$);
- ETS(M,A,M) for monthly Australian overseas visitors
  ($\alpha = 0.6146$, $\beta = 0.00019$, $\gamma = 0.1920$, $\ell_0 = 92.9631$, $b_0 = 2.2221$, $s_{-11} = 0.8413$, $s_{-10} = 0.8755$, $s_{-9} = 1.0046$, $s_{-8} = 0.9317$, $s_{-7} = 0.8219$, $s_{-6} = 1.0012$, $s_{-5} = 1.1130$, $s_{-4} = 1.3768$, $s_{-3} = 0.9625$, $s_{-2} = 1.0669$, $s_{-1} = 1.0666$, $s_0 = 0.9378$).

Although there is a lot of computation involved, it can be handled remarkably quickly on modern computers. Each of the forecasts shown in Figure 1 took no more than a few seconds on a standard PC. The US electricity generation series took the longest as there are no analytical prediction intervals available for the ETS(M,M_d,N) model. Consequently, the prediction intervals for this series were computed using simulation of 5000 future sample paths.

To apply the algorithm to the US net electricity generation time series `usnetelec`, we use the following command.

```
R> etsfit <- ets(usnetelec)
```

The object `etsfit` is of class "`ets`" and contains all of the necessary information about the

fitted model including model parameters, the value of the state vector $\boldsymbol{x}_t$ for all t, residuals and so on. Printing the `etsfit` object shows the main items of interest.

```
R> etsfit

ETS(M,A,N)

Call:
 ets(y = usnetelec)

  Smoothing parameters:
    alpha = 0.9999
    beta  = 0.2191

  Initial states:
    l = 254.9338
    b = 38.3125

  sigma:  0.0259

     AIC     AICc      BIC
634.0437 635.2682 644.0803
```

Some goodness-of-fit measures (defined in Hyndman and Koehler 2006) are obtained using `accuracy()`.

```
R> accuracy(etsfit)

                    ME     RMSE      MAE        MPE     MAPE      MASE
Training set 1.162583 52.00363 36.77721 0.2629582 1.942062 0.5211014
                     ACF1
Training set 0.006113498
```

There are also `coef()`, `plot()`, `summary()`, `residuals()`, `fitted()` and `simulate()` methods for objects of class "`ets`". The `plot()` function shows time plots of the original time series along with the extracted components (level, growth and seasonal).

The `forecast()` function computes the required forecasts which are then plotted as in Figure 1(b).

Printing the `fcast` object gives a table showing the prediction intervals.

```
R> fcast

     Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
2004       3900.329 3770.801 4029.857 3702.233 4098.425
2005       3952.650 3747.279 4158.022 3638.562 4266.738
2006       4004.972 3725.589 4284.355 3577.692 4432.251
```

```
2007       4057.293 3701.885 4412.701 3513.743 4600.842
2008       4109.614 3674.968 4544.259 3444.881 4774.347
2009       4161.935 3644.367 4679.503 3370.383 4953.487
2010       4214.256 3609.881 4818.632 3289.944 5138.569
2011       4266.577 3571.428 4961.726 3203.439 5329.716
2012       4318.898 3528.985 5108.812 3110.830 5526.967
2013       4371.220 3482.552 5259.888 3012.119 5730.320
```

The `ets()` function also provides the useful feature of applying a fitted model to a new data set. For example, we could withhold 10 observations from the `usnetelec` data set when fitting, then compute the one-step forecast errors for the out-of-sample data.

```
R> fit <- ets(usnetelec[1:45])
R> test <- ets(usnetelec[46:55], model = fit)
R> accuracy(test)
```

We can also look at the measures of forecast accuracy where the forecasts are based on only the fitting data.

```
R> accuracy(forecast(fit,10), usnetelec[46:55])
```

4.2. The HoltWinters() function

There is another implementation of exponential smoothing in R via the `HoltWinters()` function (Meyer 2002) in the **stats** package. It implements only the (N,N), (A,N), (A,A) and (A,M) methods. The initial states $\boldsymbol{x}_0$ are fixed using a heuristic algorithm. Because of the way the initial states are estimated, a full three years of seasonal data are required to implement the seasonal forecasts using `HoltWinters()`. (See Hyndman and Kostenko (2007) for the minimal sample size required.) The smoothing parameters are optimized by minimizing the average squared prediction errors, which is equivalent to minimizing (7) in the case of additive errors.

There is a `predict()` method for the resulting object which can produce point forecasts and prediction intervals. Although it is nowhere documented, it appears that the prediction intervals produced by `predict()` for an object of class `HoltWinters` are based on an equivalent ARIMA model in the case of the (N,N), (A,N) and (A,A) methods, assuming additive errors. These prediction intervals are equivalent to the prediction intervals that arise from the (A,N,N), (A,A,N) and (A,A,A) state space models. For the (A,M) method, the prediction interval provided by `predict()` appears to be based on Chatfield and Yar (1991) which is an approximation to the true prediction interval arising from the (A,A,M) model. Prediction intervals with multiplicative errors are not possible using the `HoltWinters()` function.

4.3. Implementation of the automatic ARIMA algorithm

The algorithm of Section 3 is applied to the same four time series. Unlike the exponential smoothing algorithm, the ARIMA class of models assumes homoscedasticity, which is not always appropriate. Consequently, transformations are sometimes necessary. For these four

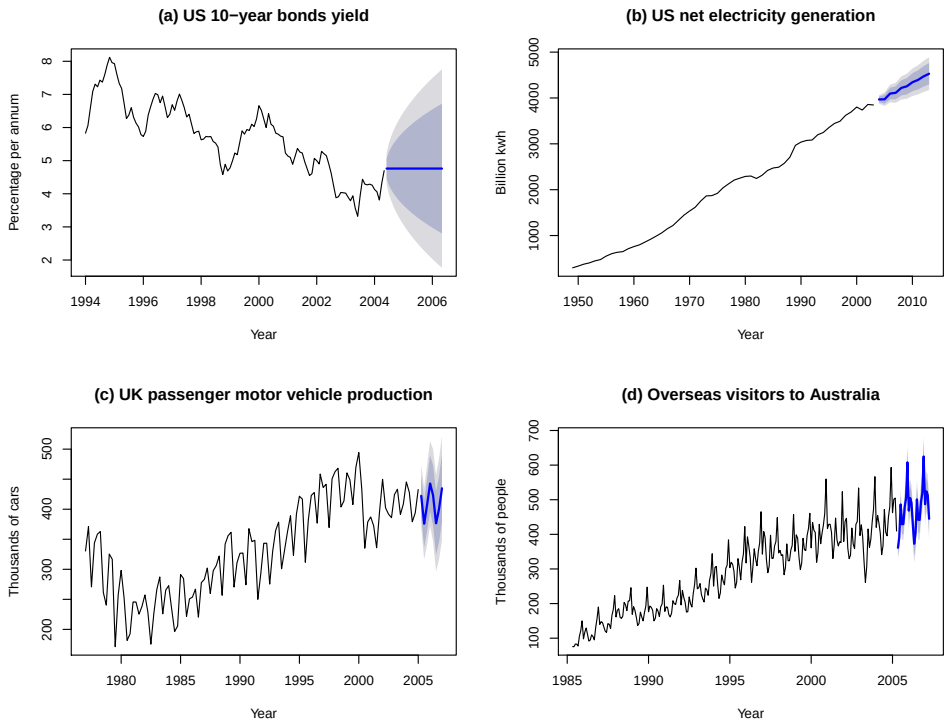

Figure 2: Four time series showing point forecasts and 80% & 95% prediction intervals obtained using ARIMA models.

time series, we model the raw data for series (a)–(c), but the logged data for series (d). The prediction intervals are back-transformed with the point forecasts to preserve the probability coverage.

To apply this algorithm to the US net electricity generation time series `usnetelec`, we use the following commands.

```
R> arimafit <- auto.arima(usnetelec)
R> fcast <- forecast(arimafit)
R> plot(fcast)
```

The function `auto.arima()` implements the algorithm of Section 3 and returns an object of class `Arima`. The resulting forecasts are shown in Figure 2. The fitted models are as follows:

- ARIMA(0,1,1) for monthly US 10-year bonds yield
  ($\theta_1 = 0.3220$);
- ARIMA(2,1,2) with drift for annual US net electricity generation
  ($\phi_1 = -1.3032$; $\phi_2 = -0.4332$; $\theta_1 = 1.5284$; $\theta_2 = 0.8340$; $c = 66.1585$);
- ARIMA(1,0,1)(1,1,2)$_4$ for quarterly UK motor vehicle production
  ($\phi_1 = 0.9253$; $\phi_2 = NA$; $\Phi_1 = -0.7526$; $\Phi_2 = NA$);
- ARIMA(1,0,1)(0,1,2)$_{12}$ with drift for monthly Australian overseas visitors
  ($\phi_1 = 0.8968$; $\theta_1 = -0.3187$; $\Theta_1 = -0.7110$; $\Theta_2 = 0.1461$; $c = 1.4820$).

Note that the R parameterization has $\theta(B) = (1 + \theta_1 B + \cdots + \theta_q B)$ and $\phi(B) = (1 - \phi_1 B + \cdots - \phi_q B)$, and similarly for the seasonal terms.

A summary of the forecasts is available, part of which is shown below.

```
Forecast method: ARIMA(2,1,2) with drift
Series: usnetelec

Coefficients:
          ar1      ar2     ma1     ma2    drift
      -1.3032  -0.4332  1.5284  0.8340  66.1585
s.e.   0.2122   0.2084  0.1417  0.1185   7.5595

sigma^2 estimated as 2262:  log likelihood=-283.34
AIC=578.67   AICc=580.46   BIC=590.61

Error measures:
                    ME   RMSE    MAE      MPE   MAPE    MASE     ACF1
Training set 0.046402 44.894 32.333 -0.61771 2.1012 0.45813 0.022492

Forecasts:
     Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
2004       3968.957 3908.002 4029.912 3875.734 4062.180
2005       3970.350 3873.950 4066.751 3822.919 4117.782
2006       4097.171 3971.114 4223.228 3904.383 4289.959
2007       4112.332 3969.691 4254.973 3894.182 4330.482
2008       4218.671 4053.751 4383.591 3966.448 4470.894
2009       4254.559 4076.108 4433.010 3981.641 4527.476
2010       4342.760 4147.088 4538.431 4043.505 4642.014
2011       4393.306 4185.211 4601.401 4075.052 4711.560
2012       4470.261 4248.068 4692.455 4130.446 4810.077
2013       4529.113 4295.305 4762.920 4171.535 4886.690
```

The training set error measures for the two models are very similar. Note that the information criteria are not comparable.

The **forecast** package also contains the function `Arima()` which is largely a wrapper to the `arima()` function in the **stats** package. The `Arima()` function in the **forecast** package makes it easier to include a drift term when $d+D=1$. (Setting `include.mean=TRUE` in the `arima()` function from the **stats** package will only work when $d+D=0$.) It also provides the facility for fitting an existing ARIMA model to a new data set (as was demonstrated for the `ets()` function earlier).

One-step forecasts for ARIMA models are now available via a `fitted()` function. We also provide a new function `arima.errors()` which returns the original time series after adjusting for regression variables. If there are no regression variables in the ARIMA model, then the errors will be identical to the original series. If there are regression variables in the ARIMA model, then the errors will be equal to the original series minus the effect of the regression variables, but leaving in the serial correlation that is modelled with the AR and MA terms. In contrast, `residuals()` provides true residuals, removing the AR and MA terms as well.

The generic functions `summary()`, `print()`, `fitted()` and `forecast()` apply to models ob-

tained from either the `Arima()` or `arima()` functions.

4.4. The forecast() function

The `forecast()` function is generic and has S3 methods for a wide range of time series models. It computes point forecasts and prediction intervals from the time series model. Methods exist for models fitted using `ets()`, `auto.arima()`, `Arima()`, `arima()`, `ar()`, `HoltWinters()` and `StructTS()`.

There is also a method for a `ts` object. If a time series object is passed as the first argument to `forecast()`, the function will produce forecasts based on the exponential smoothing algorithm of Section 2.

In most cases, there is an existing `predict()` function which is intended to do much the same thing. Unfortunately, the resulting objects from the `predict()` function contain different information in each case and so it is not possible to build generic functions (such as `plot()` and `summary()`) for the results. So, instead, `forecast()` acts as a wrapper to `predict()`, and packages the information obtained in a common format (the `forecast` class). We also define a default `predict()` method which is used when no existing `predict()` function exists, and calls the relevant `forecast()` function. Thus, `predict()` methods parallel `forecast()` methods, but the latter provide consistent output that is more useable.

4.5. The `forecast` class

The output from the `forecast()` function is an object of class "`forecast`" and includes at least the following information:

- the original series;
- point forecasts;
- prediction intervals of specified coverage;
- the forecasting method used and information about the fitted model;
- residuals from the fitted model;
- one-step forecasts from the fitted model for the period of the observed data.

There are `print()`, `plot()` and `summary()` methods for the "`forecast`" class. Figures 1 and 2 were produced using the `plot()` method.

The prediction intervals are, by default, computed for 80% and 95% coverage, although other values are possible if requested. Fan charts (Wallis 1999) are possible using the combination `plot(forecast(model.object, fan = TRUE))`.

4.6. Other functions

We now briefly describe some of the other features of the **forecast** package. Each of the following functions produces an object of class "`forecast`".

`croston()`

: implements the method of Croston (1972) for intermittent demand forecasting. In this method, the time series is decomposed into two separate sequences: the non-zero values and the time intervals between non-zero values. These are then independently forecast using simple exponential smoothing and the forecasts of the original series are obtained as ratios of the two sets of forecasts. No prediction intervals are provided because there is no underlying

stochastic model (Shenstone and Hyndman 2005).

`theta()`

: provides forecasts from the Theta method (Assimakopoulos and Nikolopoulos 2000). Hyndman and Billah (2003) showed that these were equivalent to a special case of simple exponential smoothing with drift.

`splinef()`

: gives cubic-spline forecasts, based on fitting a cubic spline to the historical data and extrapolating it linearly. The details of this method, and the associated prediction intervals, are discussed in Hyndman *et al.* (2005a).

`meanf()`

: returns forecasts based on the historical mean.

`rwf()`

: gives "naïve" forecasts equal to the most recent observation assuming a random walk model. This function also allows forecasting using a random walk with drift.

In addition, there are some new plotting functions for time series.

`tsdisplay()`

: provides a time plot along with an ACF and PACF.

`seasonplot()`

: produces a seasonal plot as described in Makridakis *et al.* (1998).

Bibliography

Anderson BDO, Moore JB (1979). *Optimal Filtering*. Prentice-Hall, Englewood Cliffs.

Aoki M (1987). *State Space Modeling of Time Series*. Springer-Verlag, Berlin.

Archibald BC (1990). "Parameter Space of the Holt-Winters' Model." *International Journal of Forecasting*, **6**, 199–209.

Assimakopoulos V, Nikolopoulos K (2000). "The Theta Model: A Decomposition Approach to Forecasting." *International Journal of Forecasting*, **16**, 521–530.

Bowerman BL, O'Connell RT, Koehler AB (2005). *Forecasting, Time Series and Regression: An Applied Approach*. Thomson Brooks/Cole, Belmont CA.

Brockwell PJ, Davis RA (1991). *Time Series: Theory and Methods*. 2nd edition. Springer-Verlag, New York.

Canova F, Hansen BE (1995). "Are Seasonal Patterns Constant Over Time? A Test for Seasonal Stability." *Journal of Business and Economic Statistics*, **13**, 237–252.

Chatfield C, Yar M (1991). "Prediction Intervals for Multiplicative Holt-Winters." *International Journal of Forecasting*, **7**, 31–37.

Croston JD (1972). "Forecasting and Stock Control for Intermittent Demands." *Operational Research Quarterly*, **23**(3), 289–304.

Dickey DA, Fuller WA (1981). "Likelihood Ratio Statistics for Autoregressive Time Series with a Unit Root." *Econometrica*, **49**, 1057–1071.

Durbin J, Koopman SJ (2001). *Time Series Analysis by State Space Methods*. Oxford University Press, Oxford.

Gardner Jr ES (1985). "Exponential Smoothing: The State of the Art." *Journal of Forecasting*, **4**, 1–28.

Gardner Jr ES, McKenzie E (1985). "Forecasting Trends in Time Series." *Management Science*, **31**(10), 1237–1246.

Gómez V (1998). "Automatic Model Identification in the Presence of Missing Observations and Outliers." *Working paper D-98009*, Ministerio de Economía y Hacienda, Dirección General de Análisis y Programación Presupuestaria.

Gómez V, Maravall A (1998). "Programs **TRAMO** and **SEATS**, Instructions for the Users." *Working paper 97001*, Ministerio de Economía y Hacienda, Dirección General de Análisis y Programación Presupuestaria.

Goodrich RL (2000). "The **Forecast Pro** Methodology." *International Journal of Forecasting*, **16**(4), 533–535.

Hannan EJ, Rissanen J (1982). "Recursive Estimation of Mixed Autoregressive-Moving Average Order." *Biometrika*, **69**(1), 81–94.

Hendry DF (1997). "The Econometrics of Macroeconomic Forecasting." *The Economic Journal*, **107**(444), 1330–1357.

Hylleberg S, Engle R, Granger C, Yoo B (1990). "Seasonal Integration and Cointegration." *Journal of Econometrics*, **44**, 215–238.

Hyndman RJ (2001). "It's Time To Move from 'What' To 'Why'—Comments on the M3-Competition." *International Journal of Forecasting*, **17**(4), 567–570.

Hyndman RJ, Akram M, Archibald BC (2008a). "The Admissible Parameter Space for Exponential Smoothing Models." *Annals of the Institute of Statistical Mathematics*, **60**(2), 407–426.

Hyndman RJ, Billah B (2003). "Unmasking the Theta Method." *International Journal of Forecasting*, **19**(2), 287–290.

Hyndman RJ, Khandakar Y (2008). "Automatic Time Series Forecasting: The Forecast Package for R." *Journal of Statistical Software*, **27**.

Hyndman RJ, King ML, Pitrun I, Billah B (2005a). "Local Linear Forecasts Using Cubic Smoothing Splines." *Australian & New Zealand Journal of Statistics*, **47**(1), 87–99.

Hyndman RJ, Koehler AB (2006). "Another Look at Measures of Forecast Accuracy." *International Journal of Forecasting*, **22**, 679–688.

Hyndman RJ, Koehler AB, Ord JK, Snyder RD (2005b). "Prediction Intervals for Exponential Smoothing Using Two New Classes of State Space Models." *Journal of Forecasting*, **24**, 17–37.

Hyndman RJ, Koehler AB, Ord JK, Snyder RD (2008b). *Forecasting with Exponential Smoothing: The State Space Approach*. Springer-Verlag. URL http://www.exponentialsmoothing.net/.

Hyndman RJ, Koehler AB, Snyder RD, Grose S (2002). "A State Space Framework for Automatic Forecasting Using Exponential Smoothing Methods." *International Journal of Forecasting*, **18**(3), 439–454.

Hyndman RJ, Kostenko AV (2007). "Minimum Sample Size Requirements for Seasonal Forecasting Models." *Foresight: The International Journal of Applied Forecasting*, **6**, 12–15.

Kwiatkowski D, Phillips PC, Schmidt P, Shin Y (1992). "Testing the Null Hypothesis of Stationarity Against the Alternative of a Unit Root." *Journal of Econometrics*, **54**, 159–178.

Liu LM (1989). "Identification of Seasonal Arima Models Using a Filtering Method." *Communications in Statistics: Theory & Methods*, **18**, 2279–2288.

Makridakis S, Anderson A, Carbone R, Fildes R, Hibon M, Lewandowski R, Newton J, Parzen E, Winkler R (1982). "The Accuracy of Extrapolation (Time Series) Methods: Results of a Forecasting Competition." *Journal of Forecasting*, **1**, 111–153.

Makridakis S, Hibon M (2000). "The M3-Competition: Results, Conclusions and Implications." *International Journal of Forecasting*, **16**, 451–476.

Makridakis S, Wheelwright SC, Hyndman RJ (1998). *Forecasting: Methods and Applications*. 3rd edition. John Wiley & Sons, New York. URL http://www.robhyndman.info/forecasting/.

Mélard G, Pasteels JM (2000). "Automatic ARIMA Modeling Including Intervention, Using Time Series Expert Software." *International Journal of Forecasting*, **16**, 497–508.

Meyer D (2002). "Naive Time Series Forecasting Methods." *R News*, **2**(2), 7–10. URL http://CRAN.R-project.org/doc/Rnews/.

Ord JK, Koehler AB, Snyder RD (1997). "Estimation and Prediction for a Class of Dynamic Nonlinear Statistical Models." *Journal of the American Statistical Association*, **92**, 1621–1629.

Ord K, Lowe S (1996). "Automatic Forecasting." *The American Statistician*, **50**(1), 88–94.

Pegels CC (1969). "Exponential Forecasting: Some New Variations." *Management Science*, **15**(5), 311–315.

R Development Core Team (2008). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL http://www.R-project.org/.

Reilly D (2000). "The **Autobox** System." *International Journal of Forecasting*, **16**(4), 531–533.

Ripley BD (2002). "Time Series in R 1.5.0." *R News*, **2**(2), 2–7. URL http://CRAN.R-project.org/doc/Rnews/.

Shenstone L, Hyndman RJ (2005). "Stochastic Models Underlying Croston's Method for Intermittent Demand Forecasting." *Journal of Forecasting*, **24**, 389–402.

Smith J, Yadav S (1994). "Forecasting Costs Incurred from Unit Differencing Fractionally Integrated Processes." *International Journal of Forecasting*, **10**(4), 507–514.

Taylor JW (2003). "Exponential Smoothing with a Damped Multiplicative Trend." *International Journal of Forecasting*, **19**, 715–725.

Wallis KF (1999). "Asymmetric Density Forecasts of Inflation and the Bank of England's Fan Chart." *National Institute Economic Review*, **167**(1), 106–112.

Affiliation:

Rob J Hyndman
Monash University
Department of Econometrics & Business Statistics Monash University Clayton VIC 3800, Australia
E-mail: Rob.Hyndman@monash.edu
URL: https://robjhyndman.com

Yeasmin Khandakar
Monash University
Department of Econometrics & Business Statistics Monash University Clayton VIC 3800, Australia

Journal of Statistical Software http://www.jstatsoft.org/
published by the Foundation for Open Access Statistics http://www.foastat.org/
July 2008, Volume 27, Issue 3 *Submitted:* 2007-05-29
doi:10.18637/jss.v000.i00 *Accepted:* 2008-03-22