Introduction to Stochastic Demography Project

Forecasting the Total Fertility Rate in some European countries

Oscar Granlund
37920

1. Introduction

In this project we try to find an autoregressive integrated moving average (ARIMA) model for forecasting the Total Fertility Rates (TFR) in 8 different European countries. The aim is to find one (hopefully relatively simple) model that as accurately as possible forecasts the TFR in all the countries.

Accurately forecasting the Total Fertility Rate can be important for a variety of reasons, for example in some countries where birth rates are decreasing there are problems where there number of workers per retiree is decreasing and thus the workers have to contribute more; in order to accurately say how much more each worker has to pay, the forecasts of the future number of workers need to be accurate. Since pension schemes are often regulated by governments in some way, decisions on changes are often made into political questions and thus the decision to use a particular model for forecasting birth rates are often politically charged and thus simpler models are favoured.

One could also argue that there are underlying factors that determine the TFR but such models will not be considered. This choice was made since accurate data for latent factors can be difficult to find while many countries publish relatively accurate data for the TFR and these datasets can easily be found aggregated at [1].

2. Autoregressive Integrated Moving Average models

There are many approaches for analysing and forecasting time series but the two most popular approaches are exponential smoothing and autoregressive integrated moving average (ARIMA) models [2]. Exponential smoothing models are based on the trends and seasonality of the data ARIMA models try to describe the autocorrelations of the

data. For our purposes we will consider only the ARIMA family of models since in some cases choosing an ARIMA model can be interpreted as determining what random process has generated the time series.

2.1. Autoregressive models

An autoregressive (AR) model is where the forecast for the value of the next point is a linear combination of the values of the past points or in other words, the time series is a linear regression of itself or auto(self)-regressive(regression). Mathematically we write the forecasted value y_t in the following way:

$$y_t = \phi_0 + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + \varepsilon_t$$

where p is the order of autoregression and ε_t is an error term (some random variable).

The optimal model (in terms of minimizing the sum of the squared residuals) can be found by defining the following vectors and matrices

$$\mathbf{y} = \begin{bmatrix} y_N \\ y_{N-1} \\ \vdots \\ y_{p+2} \\ y_{p+1} \end{bmatrix}, \quad \mathbf{X} = \begin{bmatrix} 1 & y_{N-1} & y_{N-2} & \cdots & y_{N-p+1} & y_{N-p} \\ 1 & y_{N-2} & y_{N-3} & \cdots & y_{N-p} & y_{N-p-1} \\ \vdots & \vdots & & & & \vdots \\ 1 & y_{p+1} & y_p & \cdots & y_3 & y_2 \\ 1 & y_p & y_{p-1} & \cdots & y_2 & y_1 \end{bmatrix}$$

and solving the equation $\mathbf{y} = \mathbf{X}\boldsymbol{\phi}$ for the least-squares solution (with respect to $\boldsymbol{\phi}$) $\boldsymbol{\phi} = (\mathbf{X}^\intercal \mathbf{X})^{-1} \mathbf{X}^\intercal \mathbf{y}$. Note the absence of the first p y_t:terms in the vector $\mathbf{y}$ and the absence of the y_N term in the matrix $\mathbf{X}$, clearly we might not be using all the information known to us using this solution. For example all N datapoints can be used to determine the coefficient ϕ_0 (the intercept or mean level).

We might also need to impose restrictions on the coefficients ϕ_i if we assume additional things about the underlying data, for example if we assume the random process is *stationary* we need to impose the constraint $-1 < \phi_1 < 1$ for an autoregressive model of order 1 (AR(1) model). Because of this the naive ordinary least squares solution is usually not used, instead a system of equations called the Yule-Walker equations are solved. The Yule-Walker equations are based upon the autocorrelations of the data.

The projection for the next datapoint $\hat{y}_{N+1}$ are given by

$$\hat{y}_{N+1} = \phi_0 + \phi_1 y_N + \phi_2 y_{N-1} + \cdots + \phi_{p-1} y_{N-p+2} + \phi_p y_{N-p+1}$$

for two steps ahead we use the projected value $\hat{y}_{N+1}$ instead the observed value y_{N+1} (of course, at time N we don't have an observed value at for time $N+1$). This way of projecting forwards is a commonly used technique in time series forecasting.

2.2. Moving Average models

The moving average (MA) model is a bit more complex than the AR models. Here the idea is that instead of regressing on past values we regress on past forecast errors.

Mathematically we write the MA model of order q (the MA(q) model) in the following way:

$$y_t = \theta_0 + \theta_1\varepsilon_{t-1} + \theta_2\varepsilon_{t-2} + \cdots + \theta_q\varepsilon_{t-q} + \varepsilon_t$$

where the terms ε_t, $t = 1, \ldots, N$ are the errors. Note here that the errors are not really "observed" in the same sense that the values y_t are observed. Instead we can get observed errors by having some forecast $\hat{y}_t$ and setting $\varepsilon_t = \hat{y}_t - y_t$. This means that the errors depend on the forecast but the forecast also depends on the errors so a regular OLS method will not work that well.

An interesting observation is that any *stationary* AR(p) model can be written as a MA(∞) model. For example an AR(1) model with $\phi_0 = 0$ (centred) can be written in the following way:

$$\begin{aligned} y_t &= \phi_1 y_{t-1} + \varepsilon_t \\ &= \phi_1 \left(\phi_1 y_{t-2} + \varepsilon_{t-1}\right) + \varepsilon_t \\ &= \phi_1^2 y_{t-2} + \phi_1\varepsilon_{t-1} + \varepsilon_t \\ &= \phi_1^2 \left(\phi_1 y_{t-3} + \varepsilon_{t-2}\right) + \phi_1\varepsilon_{t-1} + \varepsilon_t \\ &= \phi_1^3 y_{t-3} + \phi_1^2\varepsilon_{t-2} + \phi_1\varepsilon_{t-1} + \varepsilon_t \end{aligned}$$

and so on, ending up with

$$y_t = \sum_{i=0}^{\infty} \varepsilon_{t-i}\phi_1^i$$

which converges if $|\phi_1| < 1$ which was our constraint for stationary AR(1) processes.

The converse result is true for some MA processes; in other words some MA(q) can be written as AR(∞) processes. Such processes are called invertible and they give us a way to write the current error ε_t as a linear function of current and past observations y_i, $i = 1, \ldots, t$. For a centred MA(q) process we get

$$\varepsilon_t = \sum_{i=0}^{\infty} \left(-\theta_1\right)^i y_{t-i}$$

where again we need the constraint $|\theta_1| < 1$. This is the condition for a MA(q) process being invertible.

Of course, we will never have infinitely many datapoints but the results for the infinite cases are still usable in the finite case.

2.3. Differencing a time series and stationarity

We have seen that a concept called stationarity seems to be important for time series forecasting so perhaps we should give a definition:

A time series y_t is stationary if the properties of the time series do not change with time. This will be the case if for all s, the distribution of $(y_t, \ldots, y_{t+s})$ does not depend on t.

Since the assumption of stationarity is so important there are techniques for transforming non-stationary time series into stationary time series. One such technique is *differencing* a time series, where we instead of considering y_t consider the time series $y'_t = y_t - y_{t-1}$.

If $y'_t = y_t - y_{t-1} = \varepsilon_t$ is white noise, we have a model called the random walk model where we can write $y_t = y_{t-1} + \varepsilon_t$. This is one of the simplest non-stationary models but still one of the most widely used ones.

If differencing once is not enough we can take the differences of the differenced time series and create the time series $y''_t = y'_t - y'_{t-1} = y_t - 2y_{t-1} + y_{t-2}$. Higher orders if differencing might also be necessary.

To describe differenced time series we often use the *backshift operator* B. The backshift operator B is the operator such that $By_t = y_{t-1}$. Now applying the backshift operator twice, using standard notation, we see that $B(By_t) = B^2 y_t = y_{t-2}$ and we can write the first difference $y'_t = y_t - y_{t-1}$ as the operator equation $(1-B)y_t = y_t - y_{t-1}$ and similarly the second difference as $y''_t = (1-B)^2 y_t = (1 - 2B + B^2)y_t = y_t - 2y_{t-1} + y_{t-2}$. Thus the dth-order difference is given by the operator $(1-B)^d$.

2.4. The ARIMA model

The autoregressive integrated moving average model of order (p, d, q) (ARIMA(p, d, q)) is written as the following equations:

$$\begin{aligned} y'_t &= c + \phi_1 y'_{t-1} + \cdots + \phi_p y'_{t-p} + \theta_1 \varepsilon_{t-1} + \cdots + \theta_q \varepsilon_{t-q} + \varepsilon_t \\ (1-B)^d y_t &= c + \phi_1 B(1-B)^d y_t + \cdots + \phi_p B^p (1-B)^d y_t + \theta_1 B\varepsilon_t + \cdots + \theta_q B^q \varepsilon_t + \varepsilon_t \end{aligned}$$

where y'_t is the time series differenced d times which is equal to $(1-B)^d y_t$.

The second equation can be rewritten by gathering all the y_t terms on one side, giving:

$$\left(1 - \phi_1 B - \cdots - \phi_p B^p\right)(1-B)^d y_t = c + \left(1 + \theta_1 B + \cdots + \theta_q B^q\right)\varepsilon_t.$$

These equations can be interpreted as an autoregressive model plus a moving average model on a differenced time series. The term integrated comes from the "de-differencing" of the estimates that we have to perform, or in other words, the summing up of the forecasted differences or if our timesteps approach zero, the integration we perform.

2.5. Parameter selection

Traditionally when we choose between models that are in some way nested, measures such as the Akaike information criterion (AIC), the corrected Akaike information criterion (AICc) or the Bayesian information criterion (BIC) are useful. In this case this is however not an option since these measures of fit and complexity require that the models are fit to the same datasets which is technically not the case here since different orders of differencing d change the data to be fit. For a fixed order of differencing d we

can still use AIC, AICc or BIC to choose the best model so we first need to determine the order of differencing.

In order to determine the order of differencing we can use tests for stationarity (really they are called unit root tests) for many orders d until we find the lowest order d such that the series $(1 - B)^d y$ is stationary. There are a couple of different test for stationarity, among the most popular are the Kwiatkowski–Phillips–Schmidt–Shin (KPSS), augmented Dickey–Fuller (ADF) and Phillips–Perron (PP) tests.

Once the order of differencing d has been determined, we try to determine the rest of the parameters p and q, this can be done by fitting some initial models, selecting the best one (using AIC, AICc or BIC) and then varying p and q by ± 1 and including/excluding the drift constant c. This process of selecting the best model and searching it's "neighbouring" models for a better one until no new better model can be found.

Hyndman and Khandakar [3] suggest using the following approach for automatically fitting an ARIMA(p, d, q) model to a dataset:

1. Determine the order of differencing $0 \leq d \leq 2$ using repeated KPSS tests.

2. Fit 4 initial models:
   - ARIMA(0, d, 0),
   - ARIMA(2, d, 2),
   - ARIMA(1, d, 0) and
   - ARIMA(0, d, 1),

   without constants if $d = 2$. If $d \leq 1$ fit the previous models but also the first one again but without a constant.

3. Select the best model of the previously fitted ones using the AICc (smallest AICc is best).

4. Fit the new models ARIMA($p \pm 1$, d, $q \pm 1$) where p and q are the values that gave the previous best model. Also fit the same models including/excluding the constants.

5. Repeat from step 2 until no better model can be found (e.g. when the same model is chosen as the best one twice in a row).

Another approach to selecting the correct parameters (including the order of differencing) would be to have a hold-out set of the k last observations (a validation set), fitting the models to only the first $N - k$ observations (a training set). Then the forecast on the validation set and corresponding residuals can be used to chose the model ARIMA(p, d, q) with lowest root mean squared residuals. The search can be done in the same stepwise fashion as previously suggested using the RMS residuals on the validation set instead of the AICc criteria to select the best model. In this way the order of differencing d can also be selected. A drawback is that if models are not constant throughout

Country	Finland	Sweden	Estonia	England and Wales	France	Switzer-land	Italy	Spain
First year	1939	1891	1959	1938	1946	1932	1954	1922
Last year	2015	2016	2014	2014	2015	2014	2014	2014
n	77	126	56	77	70	83	61	93

Table 1: Table showing the countries used and the extent of their datasets.

time then the suggested model might not be valid for future forecasts. Another drawback is that we have less data to fit the ARIMA models to.

A third way of selecting the parameters is to look at autocorrelation and partial autocorrelation plots of the time series and using these specifying a model according to some rules. While similar approaches can be useful confirming that a model is sound it is not very well suited for automatic model selection so we will ignore it for now.

3. Forecasting the Total Fertility Rate in eight European countries

For our purposes (finding one ARIMA(p, d, q) model that fits many time series as well as possible) we can't really fit the models individually using the procedure suggested above. Instead we can try something similar by trying to modify the procedure to work for multiple time series.

From [1] we downloaded data for the Total Fertility Rate in 8 countries, tabulated in table **??**. The validation set was set to be the years from 2010 onwards and the training set was set to start no earlier than 1900 (impacting only Sweden). The countries chosen are such that they have at least 50 data points for training, also trying to sample uniformly geographically as well as "economically".

3.1. Determining the order of differencing

In order to determine the order of differencing we use all three tests and take the mode of their suggestions. We also restrict the maximum number of differencing to 3 and set the α values of the tests to 0.05. In listing 1 the code for computing the optimal number of differences for each country is shown. Table 2 shows the suggested number of differences for each country and test-type.

```
numberOfDiffs = matrix(nrow = length(countries), ncol = 3)
for (i in 1:length(countries)) {
  createAutocorrelationPlot(trainingset[[i]], countries[[i]][1]) %>%
  print()
  t   <- "level"
  md  <- 3
  alp <- 0.05
  numberOfDiffs[i, 1] <- ndiffs(trainingset[[i]]$TFR, test = "kpss",
   type = t, max.d = md, alpha = alp)
```

```
  numberOfDiffs[i, 2] <- ndiffs(trainingset[[i]]$TFR, test = "adf",
    type = t, max.d = md, alpha = alp)
  numberOfDiffs[i, 3] <- ndiffs(trainingset[[i]]$TFR, test = "pp",
    type = t, max.d = md, alpha = alp)
  print(getMode(numberOfDiffs[i, ]))
}
```

Listing 1: R-code for computing all suggested differences.

Autocorrelationplots to confirm that the differenced series look reasonable were also created as a part of the R-code in listing 1. These plots can be found in Appendix A.

3.2. Finding the optimal model

Now that the correct order of differencing has been found, we start by fitting 9 ARIMA models to each of the countries validation sets by setting p, $q = 0, 1, 2$. From here the accuracies of each model in terms of AICc and RMSE (on the validation set) was recorded. The models were then scored by scaling their AICc and RMSE measures so that for each country, the model that performed the best on AICc/RMSE got 1 point and the model that performed the worst got 0 points. In other words, the following equation was used (where $s_{\mathcal{M},\mathrm{C}}$ is the AICc or RMSE of a particular model $\mathcal{M}$ in a particular country C and $s'_{\mathcal{M},\mathrm{C}}$ is the scaled version)

$$s'_{\mathcal{M},\mathrm{C}} = \frac{\max_{\mathcal{M}}(s_{\mathcal{M},\mathrm{C}}) - s_{\mathcal{M},\mathrm{C}}}{\max_{\mathcal{M}}(s_{\mathcal{M},\mathrm{C}}) - \min_{\mathcal{M}}(s_{\mathcal{M},\mathrm{C}})}.$$

The two scores for each model and country were then summed and a final score for a model $\mathcal{M}$ was found by summing all the scaled and summed scores over the countries. The result of this can be seen in table 3. The scores suggest that (1, 1, 0) is the best model, winning in both scaled AICc and scaled RMSE as well as overall. We have also tested all the alternative $p \pm 1$, $q \pm 1$ models so there is no need to extend the search since we seem to be in some optimum.

Country	KPSS	ADF	PP	Mode
Finland	1	1	1	1
Sweden	1	1	1	1
Estonia	1	2	1	1
England and Wales	1	1	1	1
France	1	1	1	1
Switzerland	1	2	1	1
Italy	1	2	1	1
Spain	1	1	1	1

Table 2: Table showing results of the differencing calculation.

Model	Sum of scaled AICc	Sum of scaled RMSE	Final score
(0, 1, 0)	0.634	5.361	5.995
(0, 1, 1)	4.065	5.957	10.022
(0, 1, 2)	5.867	4.738	10.605
(1, 1, 0)	6.227	6.956	13.184
(1, 1, 1)	5.992	3.310	9.301
(1, 1, 2)	6.129	3.501	9.631
(2, 1, 0)	6.121	3.877	9.998
(2, 1, 1)	5.622	1.924	7.546
(2, 1, 2)	5.933	3.144	9.077

Table 3: Table showing the scoring of the different models.

3.3. Forecasting using the ARIMA(1, 1, 0) model

Now that we finally have chosen a model for all of the countries in a somewhat systematical way we can use this model to forecast the TFR for the years up to 2020. We can also compute the forecast for the years 2010 and onwards to see how well the forecast lines up with the real data in order to get an (informal) idea of how good the forecast is. The forecasts are plotted in figure 1.

Looking at the figures we see that the validation set forecast was always within the confidence intervals but the forecast was almost always too high, suggesting that something might be going on that the model does not pick up on. We also see that many of the time series follow a similar pattern, with a sharp decline followed by a relatively flat region. This suggests that some other kind of model might be better suited for this kind of modelling, for example for the countries that are seen as being not as far along this process of decline followed by stabilisation (and maybe increase) we could smooth, rescale and lag (or move "forwards") the forecasts of the countries further along and use that as a forecast.

To further validate the choice of the ARIMA(1, 1, 0) model one could perform some kind of independence or stationarity test on the residuals. For example one could perform a Ljung-Box test on the residuals in order to see if they are significantly different from a process with no serial correlation, e.g. seeing if there is some unexploited structure in the autocorrelations.

4. Concluding remarks

While this kind of modelling did result in a model with a reasonably good fit to the data, this is not a given when using an ARIMA model. For example the Total Fertility Rate can never be below zero while if we had a time series consisting of the data up to the middle of the sharp decline in TFR an ARIMA model might very well have forecasted a TFR below zero after 10-25 years which is absurd. Perhaps this means that some

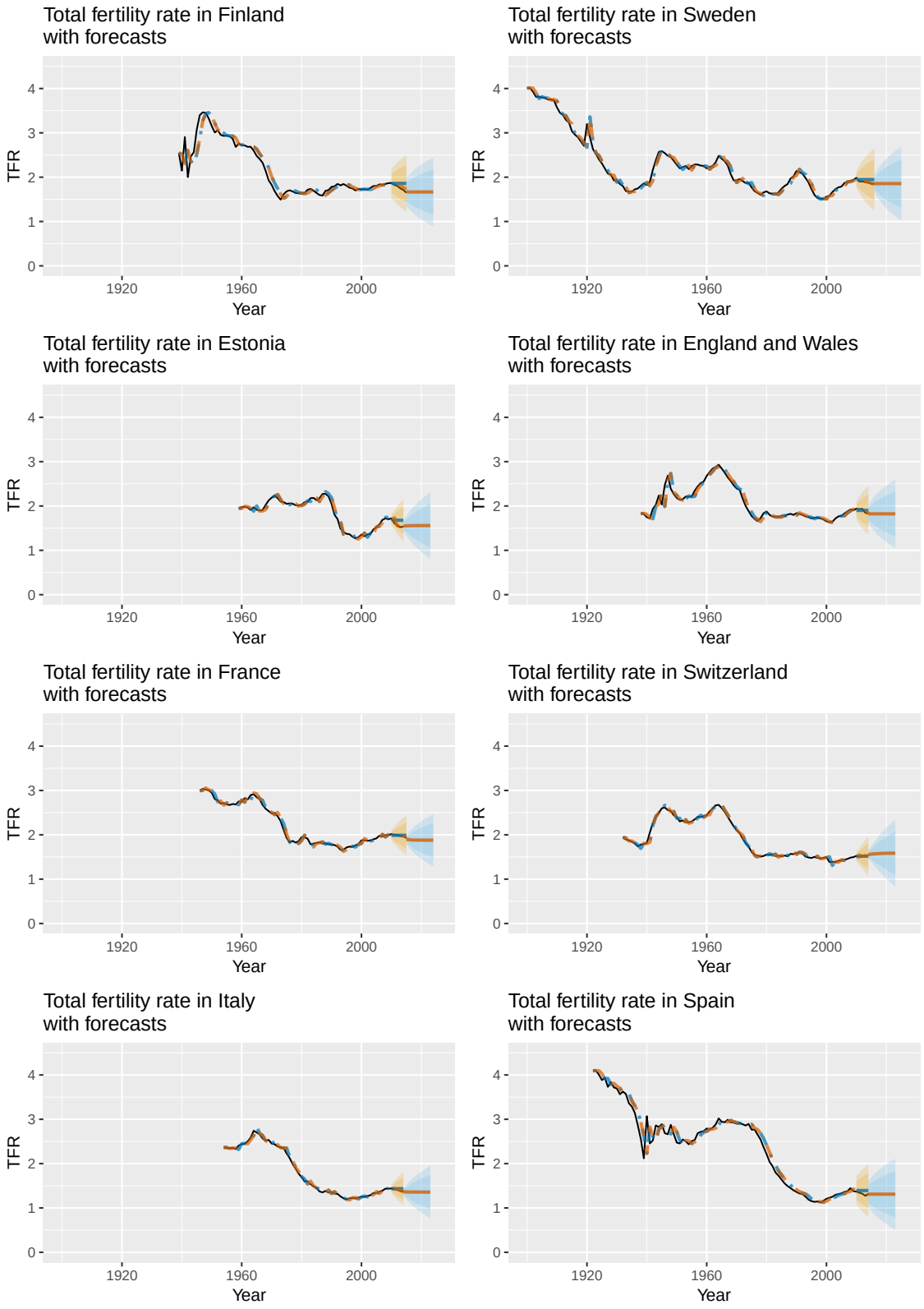


Figure 1: Plots of the forecasts of the Total Fertility Rate in the 8 countries for the years in the validation sets in blue (orange confidence intervals) as well as a final forecast for 10 years forwards after the end of the validation set in orange (blue confidence intervals).

transformation of the data should have been performed (for example a log or a logit transform).

Following the idea of using a log or a logit transform perhaps there is some generalization of the ARIMA models similar to the generalized linear model or perhaps using a logistic, poisson or an overdispersed poisson regression model using information about the country as latent factors would yield a better result. The drawback here would be that the models are more complex, perhaps not as statistically well developed and there are more decisions required during modelling.

One could also criticize the basis of the decisions made during our model-selection step. For example there is probably some modification or interpretation similar to the information criteria that would be more suited for judging what stochastic process is likely to have created a number of samples of a time series. Or perhaps the assumption that the TFR in different countries follow the same stochastic process is not correct.

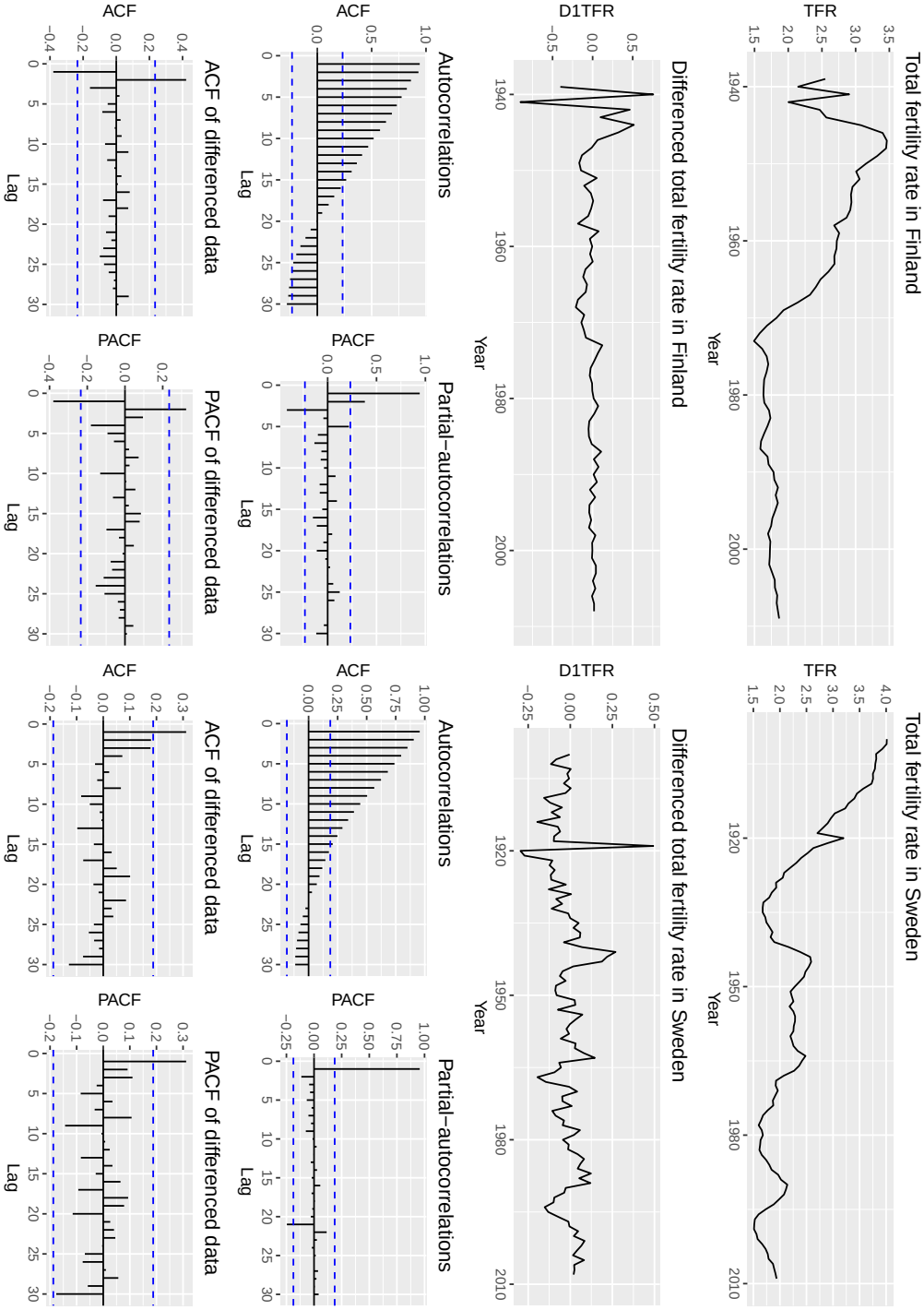
There is also a problem with using the unweighed RMSE as a model selection criteria since the validation sets were not all of the same size and the RMSE as an estimate of the models accuracy is therefore not equally accurate in all cases. The RMSE could also have been used as the only criterion and then the model search could have been done in a stepwise from start to finish, also searching over the differences but this might require a larger validation set since the RMSE becomes more important as a criteria and therefore also needs to be more accurate. This becomes a problem if the kind of stochastic process is not constant.

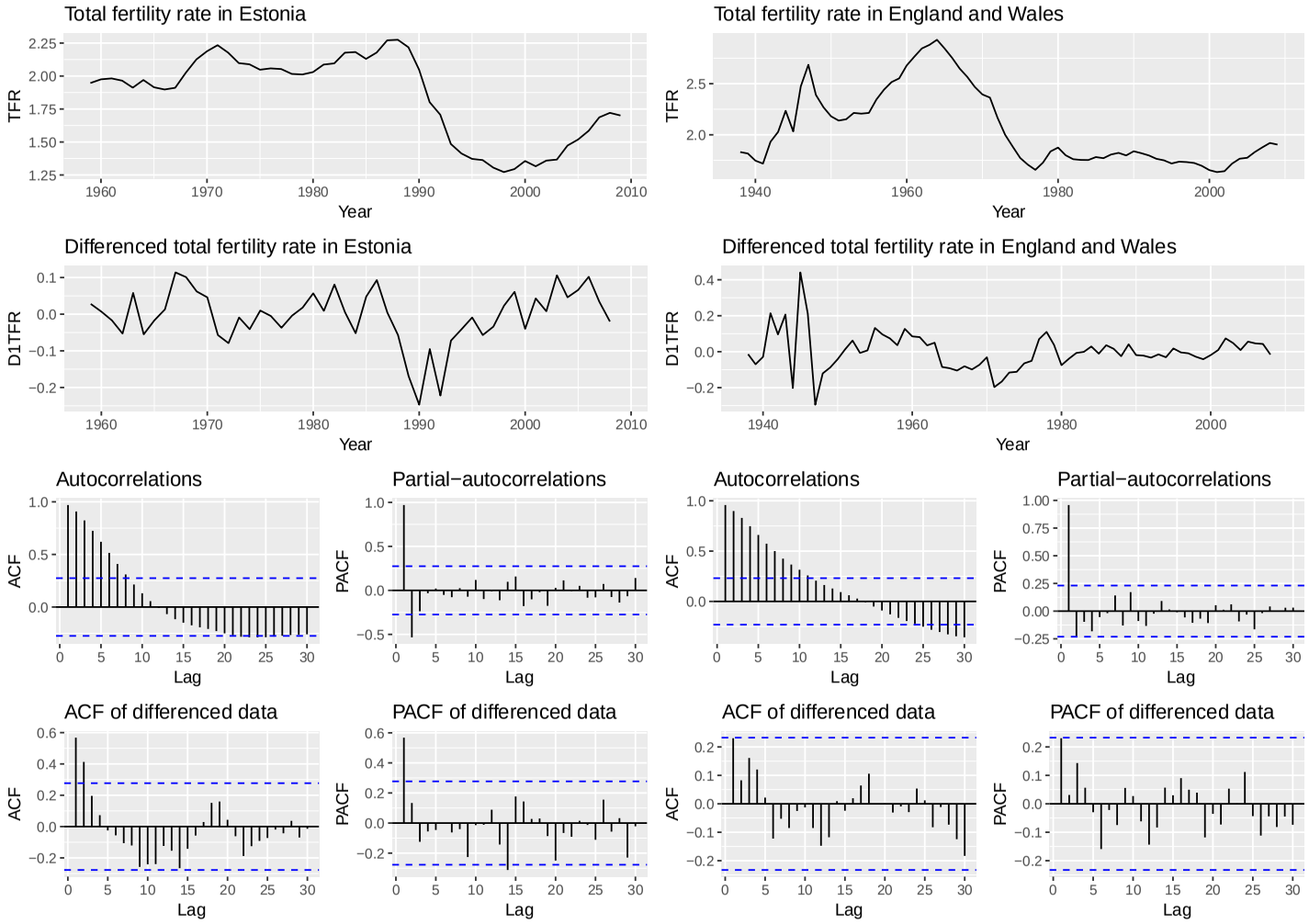
One could also try to find out if the coefficients produced using our method are the same for all countries, at least for an ARIMA(p, d, 0) model (an AR(p model in disguise) there should be relatively simple tests for finding out if the coefficients are significantly different. Such tests would probably be similar to such kinds of tests for regression models.

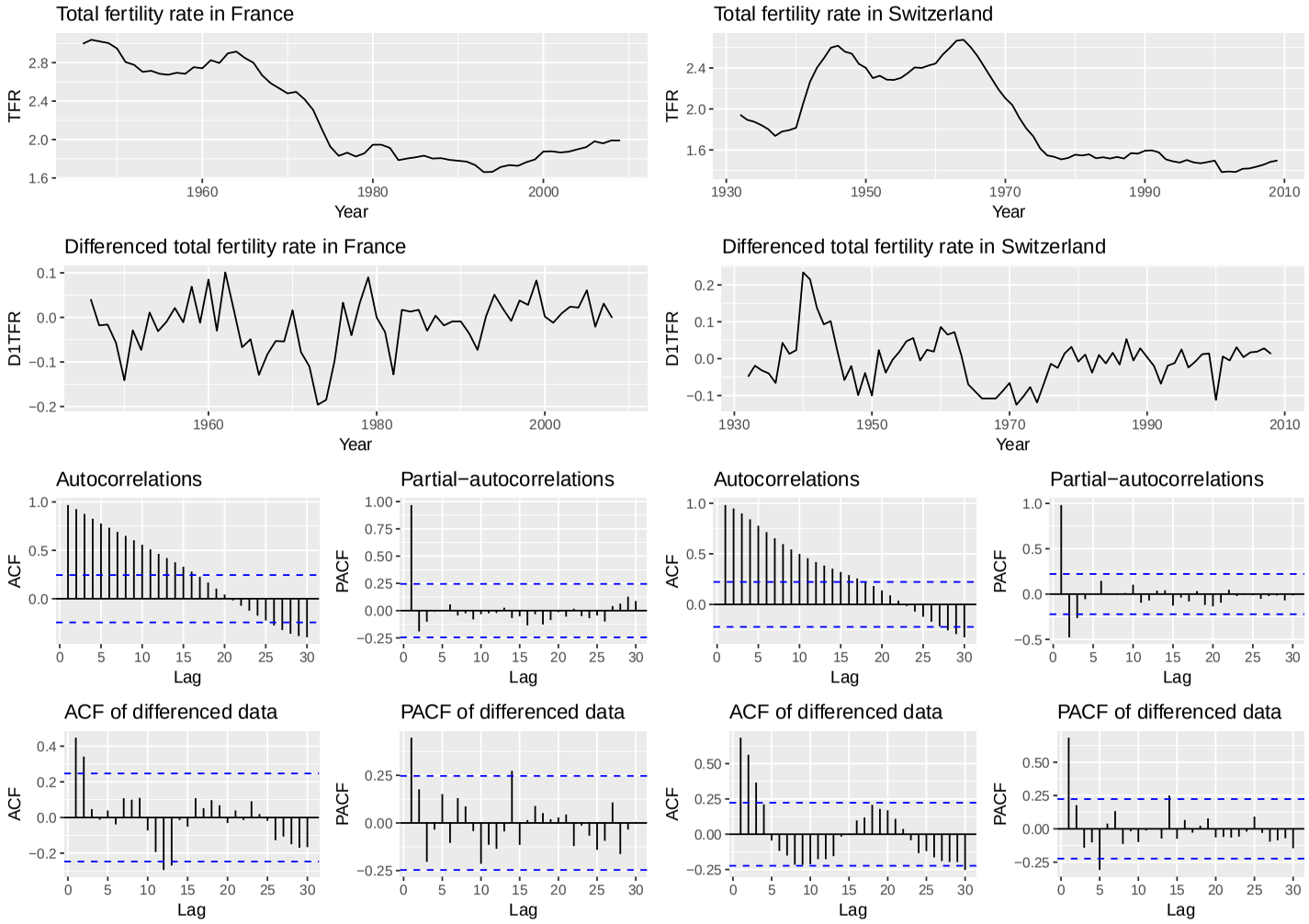
In other words, this is one relatively simple way to select one model for many samples of time series that in this case selected a relatively simple model. However, there are still many questions left unanswered and even with our method there are still some arbitrarily made decisions regarding what model selection criteria to use.

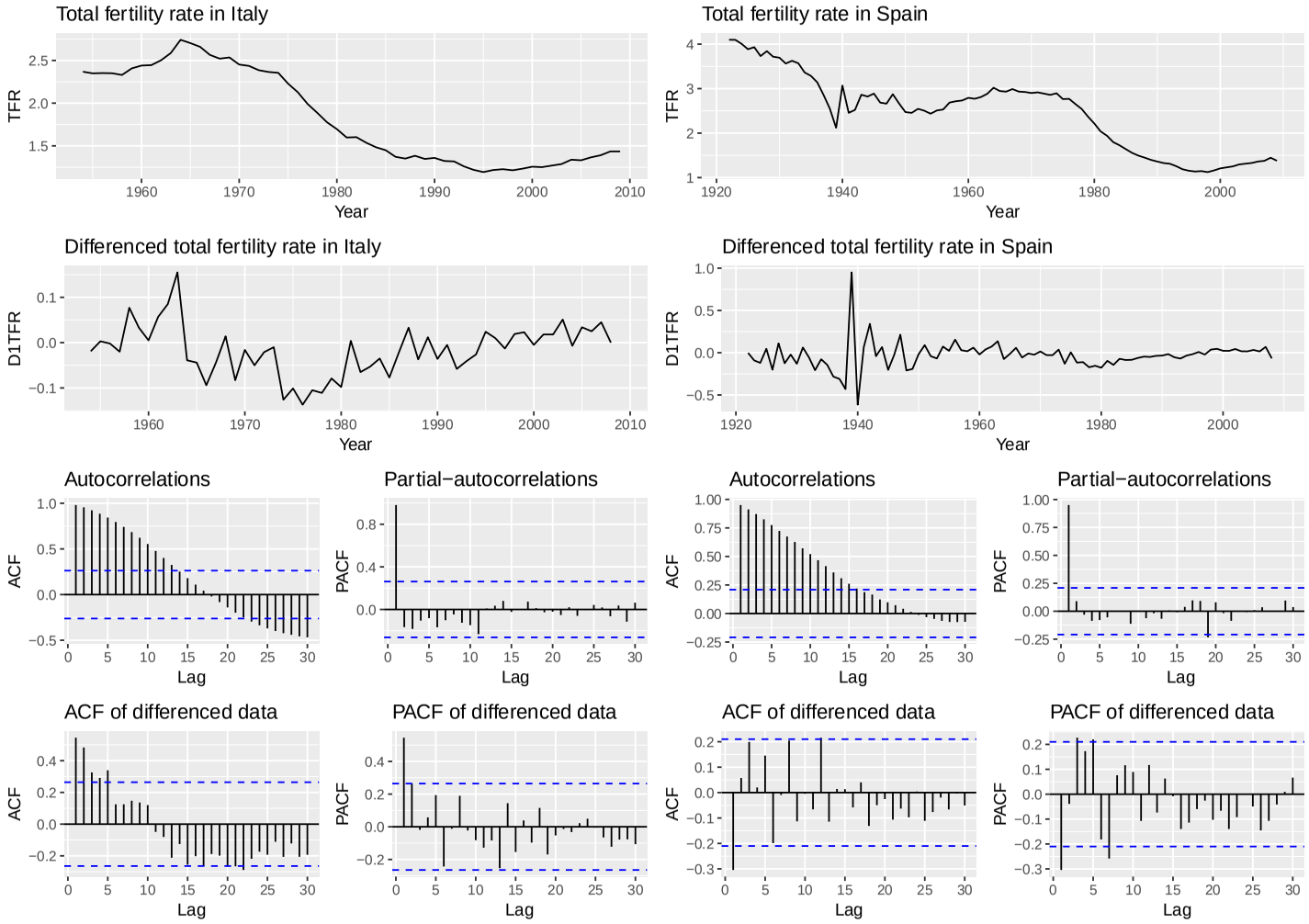
Appendices

A. Autocorrelation plots




Total fertility rate in Estonia
TFR
Year
Total fertility rate in England and Wales
TFR
Year
Differenced total fertility rate in Estonia
D1TFR
Year
Differenced total fertility rate in England and Wales
D1TFR
Year
Autocorrelations
ACF
Lag
Partial−autocorrelations
PACF
Lag
Autocorrelations
ACF
Lag
Partial−autocorrelations
PACF
Lag
ACF of differenced data
ACF
Lag
PACF of differenced data
PACF
Lag
ACF of differenced data
ACF
Lag
PACF of differenced data
PACF
Lag


Total fertility rate in France
TFR
Year
Total fertility rate in Switzerland
TFR
Year
Differenced total fertility rate in France
D1TFR
Year
Differenced total fertility rate in Switzerland
D1TFR
Year
Autocorrelations
ACF
Lag
Partial−autocorrelations
PACF
Lag
Autocorrelations
ACF
Lag
Partial−autocorrelations
PACF
Lag
ACF of differenced data
ACF
Lag
PACF of differenced data
PACF
Lag
ACF of differenced data
ACF
Lag
PACF of differenced data
PACF
Lag


Total fertility rate in Italy
TFR
Year
Total fertility rate in Spain
TFR
Year
Differenced total fertility rate in Italy
D1TFR
Year
Differenced total fertility rate in Spain
D1TFR
Year
Autocorrelations
ACF
Lag
Partial−autocorrelations
PACF
Lag
Autocorrelations
ACF
Lag
Partial−autocorrelations
PACF
Lag
ACF of differenced data
ACF
Lag
PACF of differenced data
PACF
Lag
ACF of differenced data
ACF
Lag
PACF of differenced data
PACF
Lag

B. R-code

B.1. Analysis.R

```
library(forecast)
library(ggplot2)
library(ggpubr)
#library(gridExtra)
#library(grid)
#library(gtable)
#library(egg)
source("utils.R")

# Please note that the development version of forecast might be needed
# for ndiffs to work, as per
   https://github.com/robjhyndman/forecast/issues/695.
# Install by running
# devtools::install_github("robjhyndman/forecast")
# and make sure the devtools are installed before,
# install.packages("devtools")

countries <- list(
list("Finland",           "FINtfrRR.txt"    ),
list("Sweden",            "SWEtfrRR.txt"    ),
list("Estonia",           "ESTtfrRR.txt"    ),
list("England and Wales", "GBRTENWtfrRR.txt"),
list("France",            "FRATNPtfrRR.txt" ),
list("Switzerland",       "CHEtfrRR.txt"    ),
list("Italy",             "ITAtfrRR.txt"    ),
list("Spain",             "ESPtfrRR.txt"    ))

data <- list()
for (i in 1:length(countries)) {
  data[[i]] <- read.table(countries[[i]][[2]][1], skip=2, header=TRUE)
}

trainingStart   <- 1900
validationStart <- 2010
trainingset   <- list()
validationset <- list()
for (i in 1:length(countries)) {
  trainingset[[i]]  <- subset(data[[i]], (data[[i]]$Year <
    validationStart & data[[i]]&Year  >= trainingStart))
  trainingset[[i]]  <- na.omit(trainingset[[i]])
  validationset[[i]]  <- subset(data[[i]],  data[[i]]$Year >=
    validationStart)
}

numberOfDiffs = matrix(nrow = length(countries), ncol = 3)
for (i in 1:length(countries)) {
  createAutocorrelationPlot(trainingset[[i]], countries[[i]][1]) %>%
  #ggexport(filename=paste(c("AC", countries[[i]][1], ".pdf"),
```

```
    collapse=""), width=5.83, height=8.27)
  print()
  t   <- "level"
  md  <- 3
  alp <- 0.05
  numberOfDiffs[i, 1] <- ndiffs(trainingset[[i]]$TFR, test = "kpss",
    type = t, max.d = md, alpha = alp)
  numberOfDiffs[i, 2] <- ndiffs(trainingset[[i]]$TFR, test = "adf",
    type = t, max.d = md, alpha = alp)
  numberOfDiffs[i, 3] <- ndiffs(trainingset[[i]]$TFR, test = "pp",
    type = t, max.d = md, alpha = alp)
  print(getMode(numberOfDiffs[i, ]))
}

models <- list()
for (i in 1:length(countries)) {
  models[[i]]      <- list()
  models[[i]][[1]] <- Arima(trainingset[[i]]$TFR, order = c(0, 1, 0),
    method = "ML")
  models[[i]][[2]] <- Arima(trainingset[[i]]$TFR, order = c(0, 1, 1),
    method = "ML")
  models[[i]][[3]] <- Arima(trainingset[[i]]$TFR, order = c(0, 1, 2),
    method = "ML")
  models[[i]][[4]] <- Arima(trainingset[[i]]$TFR, order = c(1, 1, 0),
    method = "ML")
  models[[i]][[5]] <- Arima(trainingset[[i]]$TFR, order = c(1, 1, 1),
    method = "ML")
  models[[i]][[6]] <- Arima(trainingset[[i]]$TFR, order = c(1, 1, 2),
    method = "ML")
  models[[i]][[7]] <- Arima(trainingset[[i]]$TFR, order = c(2, 1, 0),
    method = "ML")
  models[[i]][[8]] <- Arima(trainingset[[i]]$TFR, order = c(2, 1, 1),
    method = "ML")
  models[[i]][[9]] <- Arima(trainingset[[i]]$TFR, order = c(2, 1, 2),
    method = "ML")
}

# Writing accuracies to datafile for tabulation in excel/latex
writeAccuracies("accuracies2.csv", validationset, models, countries)

# Based upon scoring models according to their realtive AICc and
    RMSE_Test
fm <- c(1, 1, 0)
finalmodels <- list()
p <- list()
for (i in 1:length(countries)) {
  curts <- trainingset[[i]]
  curvs <- validationset[[i]]
  cur   <- rbind(curts, curvs)
  finalmodels[[i]] <- Arima(cur$TFR, order = fm, method = "ML")
  p[[i]] <- plotForecasts(cur, curts, curvs, models[[i]][[4]],
    finalmodels[[i]], countries[[i]][[1]])
```

```
}

pf <- ggarrange(p[[1]], p[[2]], p[[3]], p[[4]], p[[5]], p[[6]],
   p[[7]], p[[8]], ncol = 2, nrow = 4) %>%
#ggexport(filename = "arima110Forecasts.pdf", width = 8.27, height =
   11.69)
print()
```

Listing 2: The main code used for the analysis, requires the `utils.R` file with help functions for IO to be present. Some parts of the code left commented, mostly ones used to write data to disk.

B.2. utils.R

```
createAutocorrelationPlot <- function(df, cn) {
  p1 <- ggplot(df, aes(x=Year, y=TFR)) + geom_line()
  p1 <- p1 + ggtitle(paste(c("Total fertility rate in", cn), collapse
    = " "))
  p2 <- ggAcf(df$TFR, lag.max = 30, type = c("correlation"))
  p2 <- p2 + ggtitle("Autocorrelations")
  p3 <- ggAcf(df$TFR, lag.max = 30, type = c("partial"))
  p3 <- p3 + ggtitle("Partial-autocorrelations")
  diffed <- data.frame(df[-nrow(df), ], diff(df$TFR, differences = 1))
  names(diffed)[4] <- "D1TFR"
  pd <- ggplot(data = diffed, aes(x=Year, y=D1TFR)) + geom_line()
  pd <- pd + ggtitle(paste(c("Differenced total fertility rate in",
    cn), collapse = " "))
  p4 <- ggAcf(diffed$D1TFR, lag.max = 30, type = c("correlation"))
  p4 <- p4 + ggtitle("ACF of differenced data")
  p5 <- ggAcf(diffed$D1TFR, lag.max = 30, type = c("partial"))
  p5 <- p5 + ggtitle("PACF of differenced data")
  return(ggarrange(p1, pd, ggarrange(p2, p3, ncol = 2), ggarrange(p4,
    p5, ncol = 2), nrow = 4))
}

getMode <- function(v) {
  uniqv <- unique(v)
  return(uniqv[which.max(tabulate(match(v, uniqv)))])
}

writeAccuracies <- function (f, validationset, models, countries) {
  write("Country;Order;AICc;RMSE;AIC;BIC", file = f)
  for (i in 1:length(countries)) {
    w  <- paste(countries[[i]][[1]], ";", sep = "")
    for (j in 1:length(models[[i]])) {
      if(j!=1) {w <- ";"}
      cur <- models[[i]][[j]]
      w   <- paste(w, "(",     cur$arma[1], ", 1, " ,cur$arma[2], ")",
   sep = "")
      acc <- accuracy(f = forecast(cur, h =
   length(validationset[[i]]$TFR)), x = validationset[[i]]$TFR)
      w   <- paste(w, cur$aicc, acc[4], cur$aic, cur$bic, sep = ";")
      w   <- paste(w, "", sep = "")
```

```
      write(w, file = f, append = TRUE)
    }
  }
}

plotForecasts <- function (cur, curts, curvs, prevmodel, finalmodel,
   cn) {
  df1  <- data.frame(Year = seq(cur$Year[1], tail(cur$Year, n = 1)),
   TFR = cur$TFR, ident = "Data")
  fc1  <- forecast(prevmodel, h = length(curvs$Year))
  del1 <- seq(curvs$Year[1], tail(curvs$Year, n = 1))
  df2  <- data.frame(Year = del1, f1 = fc1$mean,      ident =
   "Forecast 1")
  df3  <- data.frame(Year = del1, f1u80 = fc1$upper[,1], ident =
   "Upper 80% 1")
  df4  <- data.frame(Year = del1, f1u95 = fc1$upper[,2], ident =
   "Upper 95% 1")
  df5  <- data.frame(Year = del1, f1l80 = fc1$lower[,1], ident =
   "Lower 80% 1")
  df6  <- data.frame(Year = del1, f1l95 = fc1$lower[,2], ident =
   "Lower 95% 1")
  ahea <- 10
  fc2  <- forecast(finalmodel, h = ahea)
  del2 <- seq(tail(curvs$Year, n = 1), tail(curvs$Year, n = 1) + ahea
   - 1)
  df7  <- data.frame(Year = del2, f2 = fc2$mean,      ident =
   "Forecast 2")
  df8  <- data.frame(Year = del2, f2u80 = fc2$upper[,1], ident =
   "Upper 80% 2")
  df9  <- data.frame(Year = del2, f2u95 = fc2$upper[,2], ident =
   "Upper 95% 2")
  df10 <- data.frame(Year = del2, f2l80 = fc2$lower[,1], ident =
   "Lower 80% 2")
  df11 <- data.frame(Year = del2, f2l95 = fc2$lower[,2], ident =
   "Lower 95% 2")
  del3 <- seq(curts$Year[1], tail(curts$Year, n = 1))
  df12 <- data.frame(Year = del3, fit1 = models[[i]][[4]]$fitted,
   ident = "Fitted 1")
  df13 <- data.frame(Year = cur$Year, fit2 = finalmodels[[i]]$fitted,
   ident = "Fitted 2")
  empty  <- data.frame(Year = seq(cur$Year[1], tail(curvs$Year, n = 1)
   + ahea - 1), TFR = vector("numeric", length =
   length(seq(cur$Year[1], tail(curvs$Year, n = 1) + ahea - 1))))
  dci180 <- merge(df3, df5,  by = "Year", all = TRUE)
  dci195 <- merge(df4, df6,  by = "Year", all = TRUE)
  dci280 <- merge(df8, df10, by = "Year", all = TRUE)
  dci295 <- merge(df9, df11, by = "Year", all = TRUE)
  dci180 <- merge(empty, dci180, by = "Year")
  dci195 <- merge(empty, dci195, by = "Year")
  dci280 <- merge(empty, dci280, by = "Year")
  dci295 <- merge(empty, dci295, by = "Year")
  p <- ggplot(df1, aes(Year, TFR))
```

```
  + geom_line(data = df1, colour = "#000000")
  + geom_ribbon(data = dci180, aes(x = Year, ymin = f1l80, ymax =
    f1u80), fill = "#E69F00", alpha = 0.2)
  + geom_ribbon(data = dci195, aes(x = Year, ymin = f1l95, ymax =
    f1u95), fill = "#E69F00",   alpha = 0.2)
  + geom_line(  data = df2,    aes(x = Year, y = f1), colour =
    "#0072B2", alpha = 0.8, linetype = "solid",   size = 1.0)
  + geom_line(  data = df12,   aes(x = Year, y = fit1), colour =
    "#0072B2", alpha = 0.7, linetype = "dotdash", size = 1.0)
  + geom_ribbon(data = dci280, aes(x = Year, ymin = f2l80, ymax =
    f2u80), fill = "#56B4E9",   alpha = 0.2)
  + geom_ribbon(data = dci295, aes(x = Year, ymin = f2l95, ymax =
    f2u95), fill = "#56B4E9",   alpha = 0.2)
  + geom_line(  data = df7,    aes(x = Year, y = f2), colour =
    "#D55E00", alpha = 0.8, linetype = "solid",   size = 1.0)
  + geom_line(  data = df13,   aes(x = Year, y = fit2), colour =
    "#D55E00", alpha = 0.7, linetype = "dashed",  size = 1.0)
  + scale_x_continuous(limits = c(1900, 2025)) +
    scale_y_continuous(limits = c(0, 4.5))
  +  ggtitle(paste("Total fertility rate in ", cn, "\nwith forecasts",
    sep = ""))
  return(p)
}
```

Listing 3: A file containing some helpful functions, mostly for creating pretty plots.

References

[1] *Human Fertility Database*. Max Planck Institute for Demographic Research (Germany) and Vienna Institute of Demography (Austria). Available at www.humanfertility.org. Accessed at 19.05.2018.

[2] Hyndman, R.J., & Athanasopoulos, G. (2018). *Forecasting: principles and practice*, 2nd edition, OTexts: Melbourne, Australia. OTexts.com/fpp2. Accessed on 08.05.2019.

[3] Hyndman, R. J., & Khandakar, Y. (2008). *Automatic time series forecasting: The forecast package for R.* Journal of Statistical Software, 27(1), 1-22.