W271-2 – Spring 2016 – **HW 8**

Juanjo Carin, Kevin Davis, Ashley Levato, Minghu Song

April 6, 2016

Contents

Build an univariate linear time series model (i.e AR, MA, and ARMA models) using the series in `hw08_series.csv`.

- **Use all the techniques that have been taught so far to build the model, including date examination, data visualization, etc.**
- **All the steps to support your final model need to be shown clearly.**
- **Show that the assumptions underlying the model are valid.**
- **Which model seems most reasonable in terms of satisfying the model's underling assumption?**
- **Evaluate the model performance (both in- and out-of-sample)**
- **Pick your "best" models and conduct a 12-step ahead forecast. Discuss your results. Discuss the choice of your metrics to measure "best".**

---

1 Exploratory Data Analysis

1.1 Loading the Data

First we load the series:

```
hw08 <- read.csv('hw08_series.csv', header = TRUE)
str(hw08)
```

```
## 'data.frame':    372 obs. of  2 variables:
##  $ X: int  1 2 3 4 5 6 7 8 9 10 ...
##  $ x: num  40.6 41.1 40.5 40.1 40.4 41.2 39.3 41.6 42.3 43.2 ...
```

```
all(hw08$X == 1:dim(hw08)[1]) # check if 1st column is just an incremental index
```

```
## [1] TRUE
```

```
hw08 <- hw08[, -1]
```

The file has two columns but the first one is just an incremental index so we discard it. The second column (that is stored in a numeric vector called `hw08`) contains 372 observations. 372 is a multiple of 12 ($372/12 = 31$) so **we'll assume that the series contains montly observations from 31 years (*for labelling purposes only, sometimes we'll also assume that the period goes from 1980 to 2010*)**.

1.2 Exploratory Data Analysis

Let's explore the main descriptive statistics of the series, as well as its histogram and time-series plot:

```
# Exploratory Data Analysis -----------------------------------------------
# See the definition of the function in ## @knitr Libraries-Functions-Constants
desc_stat(hw08, 'Time series', 'Descriptive statistics of the time series.')
```

Table 1: Descriptive statistics of the time series.

	Time series
Mean	84.83
St. Dev	31.95
1st Quartile	57.38
Median	76.45
3rd Quartile	111.53
Min	36.00
Max	152.60

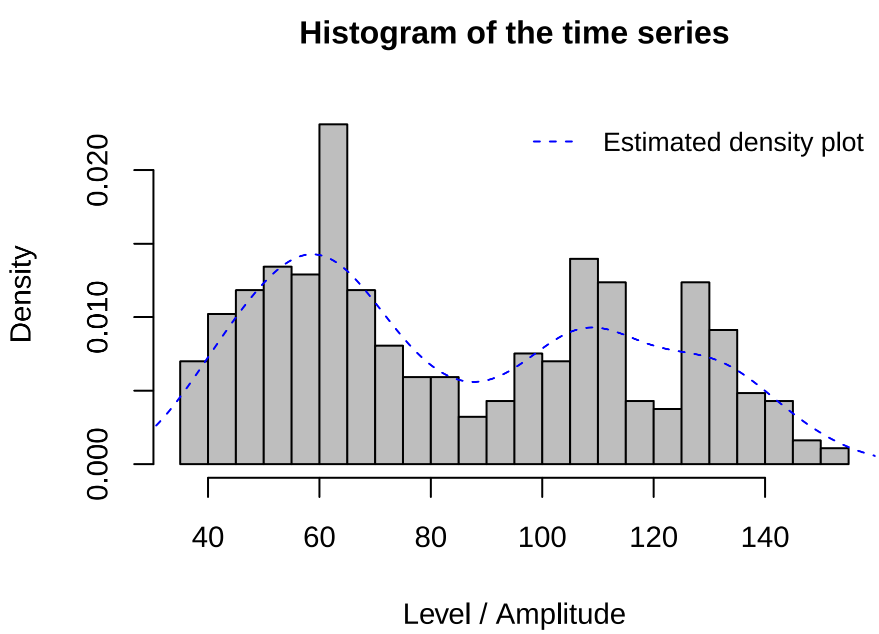

Figure 1: Histogram of the data.

The histogram shows that the distribution of the data is multimodal, and hence far from normal. But as usual, it tells us nothing about the dynamics of the time series: only what values were more or less frequent, but not when they happened.

To label the time-series plot, we will assume (as mentioned) that the data were collected on a monthly basis and will use 1980 as an **arbitrary** starting point.

```
hw08.ts <- ts(hw08, start = c(1980,1), frequency = 12)
```

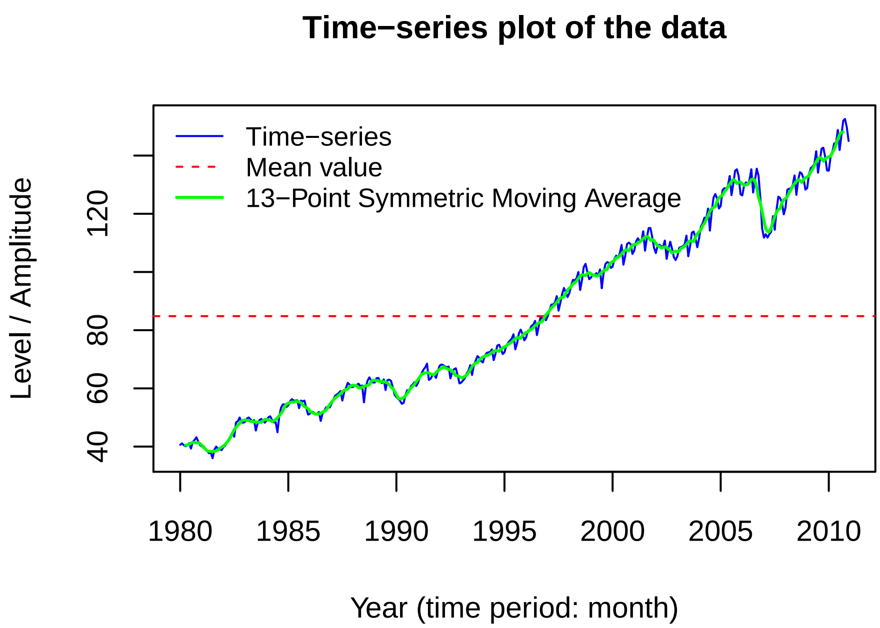

Figure 2: Time-series plot (assuming monthly data, from 1980 until 2010).

Our assumption that the data corresponds to a monthly time series seems reasonable after noticing that there seems to be some seasonality every 12 time periods (see Figure 3 below, which shows only the last observations: the level increases over the first 6 months—especially in February and June—, goes down in July, up from August to October, and down again the last 2 months of the year).

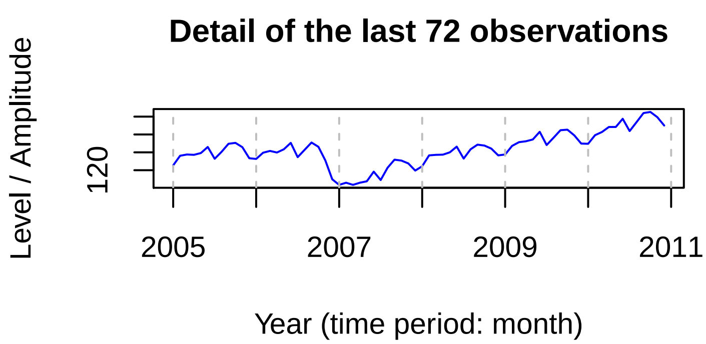

Figure 3: Time-series plot of (the last 72 observations—6 years?—of) the data.

Apart from showing that the time series is **not (mean) stationary** (the mean depends on time, with an increasing trend, and the time series is very **persistent**), Figure 2 in the previous page shows that the time is also **not variance stationary:** the variance is not constant but changes with time (increasing in the last years, especially the last 7); see Table 2 and Figure 4 in the next page.

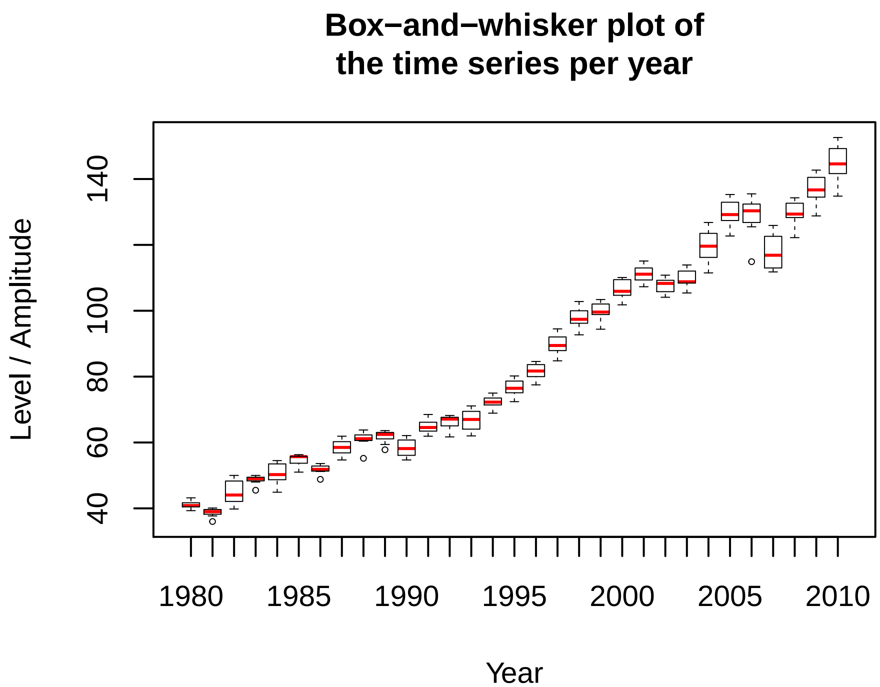

Figure 4: Boxplot of the series, per year (every 12 observations).

Table 2: Variance of the time-series amplitude per year (for the first 30 out of 31).

Year	Mean	Variance	Year	Mean	Variance	Year	Mean	Variance
1980	41.05	1.12	1990	58.29	6.61	2000	106.34	7.84
1981	38.79	1.36	1991	64.82	3.66	2001	111.15	7.23
1982	44.91	13.06	1992	66.25	4.21	2002	107.78	5.03
1983	48.70	1.37	1993	66.70	9.79	2003	109.58	7.79
1984	50.57	8.63	1994	72.24	3.31	2004	119.73	22.65
1985	54.84	2.59	1995	76.51	5.49	2005	129.77	13.84
1986	51.87	1.73	1996	81.67	5.59	2006	129.29	30.61
1987	58.49	4.94	1997	89.83	8.18	2007	117.78	29.39
1988	61.06	4.42	1998	97.77	9.03	2008	129.81	12.05
1989	61.86	3.17	1999	100.00	6.37	2009	137.07	16.68

Both results indicate that the data does not seem to be a realization of a stationary process, so **an ARMA model may not be a good fit for our data** (maybe it is—as it happened with the USNZ series we analyzed in class—, but it will certainly not be good for forecasting). At the very least, we should transform the data to stabilize the variance, take first differences of the data until they're stationary, and so forth.

To continue the Exploratory Data Analysis, let's decompose the time series to check the growing (though not exactly linear) trend and seasonality:

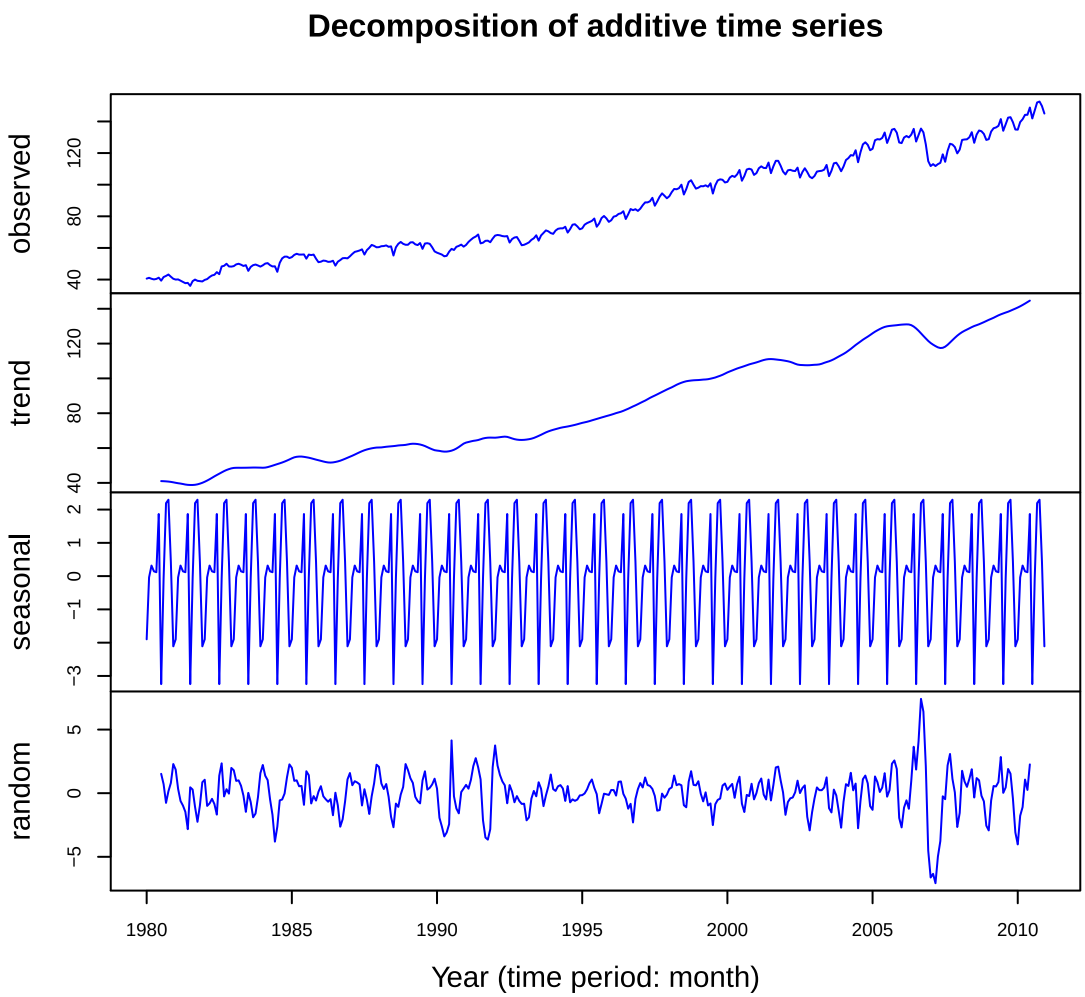

Figure 5: (Additive) decomposition of the time series.

1.3 ACF and PACF of the Time Series

The correlogram (where 2 years—or 24 1-month time displacements—are plotted) also shows how persistent the series is, looking very much like that of a random walk with drift. That is also an indication that an MA model may not be a good fit for this time series. The PACF drops off very sharply after the 1st lag, though the PACF of the 12th lag is also significant (probably due to the seasonal component).

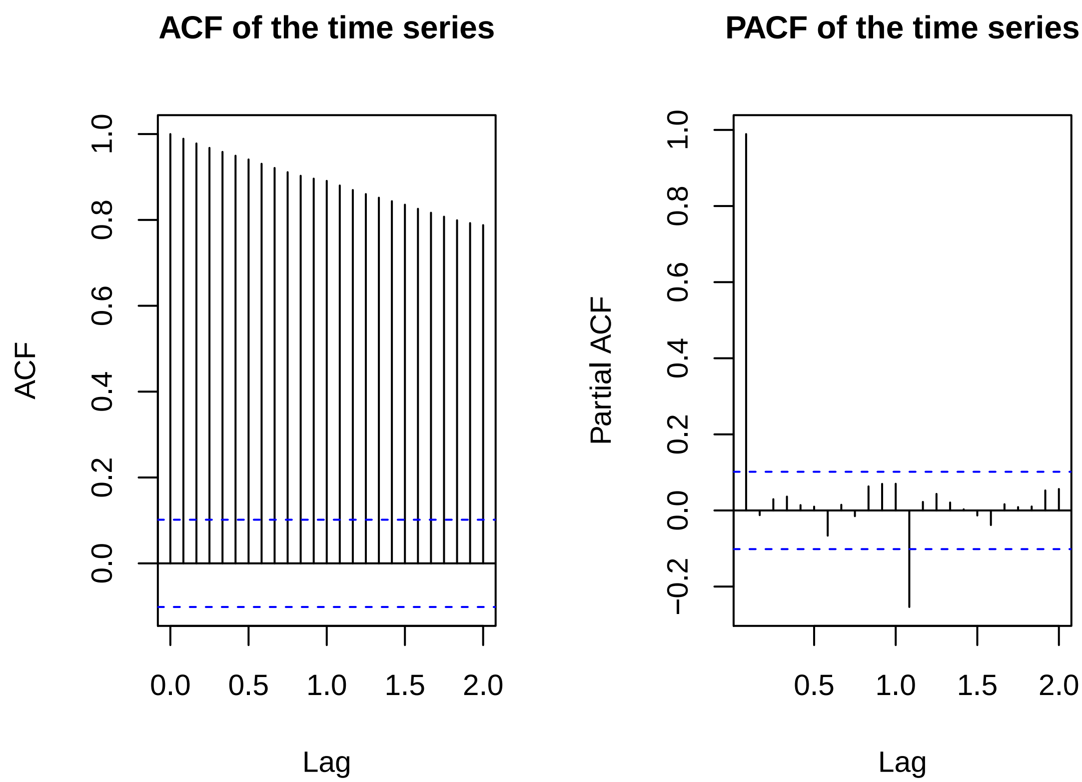

Figure 6: Autocorrelation and partial autocorrelation graphs

2 Univariate Linear Time Series models

2.1 Candidate Models

We are going to build several models of each family (ARMA, AR, and MA—the last two ones are just an special case of the former), with up to 12 coefficients each (i.e., the most complex AR and MA models will be AR(12) and MA(12), respectively; as for the ARMA models, they could be ARMA(11,1), ARMA(10,2), . . . , ARMA(6,6)). We will select those models (within each family) with the lowest value of **AIC** and **BIC**—which puts a greater penalty on the number of coefficients—, and test their behavior in terms of their residuasl and (in- and out-of-sample) fit.

There are $13^2 = 169$ possible combinations of 2 orders from 0 to 12 each, but since we're excluding the cases where $p = 0$ and $q = 0$ at the same time (that is, an AR(0) or MA(0) or ARMA(0,0) model, which is simply a white noise), plus all the combinations for which $p + q > 12$, the number is limited to 90. And, depending on the time series, some of them will not be valid: those that yield a non-stationary process (the function `arima()` will yield an error in those cases). In this particular case, only 37 of those 90 potential models can actually be estimated.

```
max_coef <- 12
orders <- data.frame(permutations(n = max_coef + 1, r = 2, v = 0:max_coef,
                                  set = FALSE, repeats.allowed = TRUE))
dim(orders)[1] # Number of models up to max_coef
```

```
## [1] 169
```

```
colnames(orders) <- c("p", "q")
orders <- orders %>% filter(p + q <= max_coef & p + q > 0)
dim(orders)[1] # Number of models considered
```

```
## [1] 90
```

```
orders %>% sample_n(10) # A 10-sample of the possible orders
```

```
##     p  q
## 11  0 11
## 56  5  1
## 54  4  8
## 55  5  0
## 75  7  5
## 87 10  2
## 1   0  1
## 20  1  7
## 85 10  0
## 42  3  6
```

```
aic_list <- orders %>% rowwise() %>%
  mutate(aic = try_default(AIC(Arima(hw08, order = c(p, 0, q))), default = NA,
                          quiet = TRUE))
aic_list <- aic_list %>% filter(!is.na(aic))
dim(aic_list)[1] # Number of models estimated
```

```
## [1] 38
```

```
# Add the BIC (and the corresponding family)
aic_list <- aic_list %>%
  mutate(bic = BIC(Arima(hw08, order = c(p, 0, q))),
         family = ifelse(q == 0, "AR", ifelse(p == 0, "MA", "ARMA")))
```

As shown below, the range of the AIC (and BIC) values is quite reduced for AR models, slightly wider for ARMA models (which have lower AIC and BIC values in most cases), and quite huge for MA models, which have the highest (and hence worst) values of all models: for the number of coefficients that we have considered, even the best MA model is worse (with respect to these criteria, at least) than the worst of the AR or ARMA models.

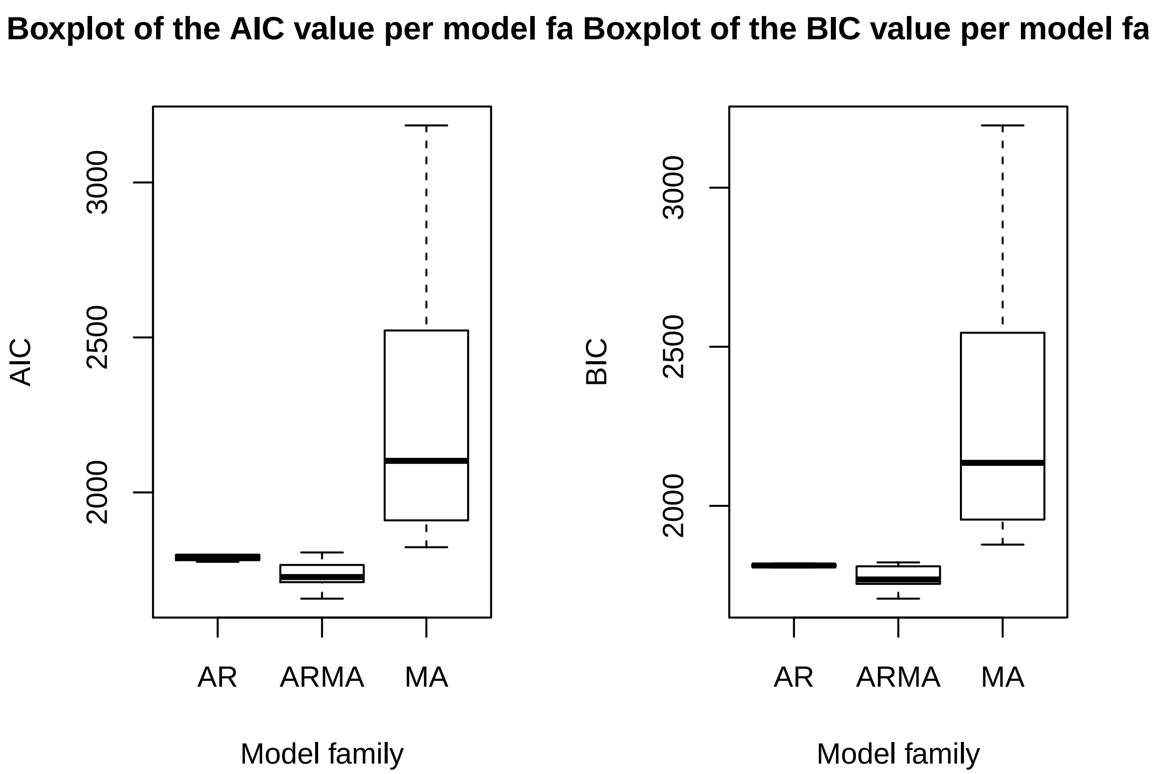

Figure 7: Box-plots of the AIC and BIC values per model family

The following plots show the AIC and BIC values for the AR and MA models, depending on the number of coefficients (p and q, respectively). As we can see:

- Only 4 of the possible 12 AR models could be estimated (the others were not stationary).
- The AIC values of the AR(1) and AR(2) models are quite similar. Those of the AR(3) and AR(9) models are lower. And though the AIC value of the AR(9) is the lowest, its BIC value is larger than all the others (because of the extra parameters).
- The AIC and BIC values of the MA models are all higher than those of the AR models. They decrease with the number of coefficients, though they are pretty much the same for 10, 11, or 12 coefficients.

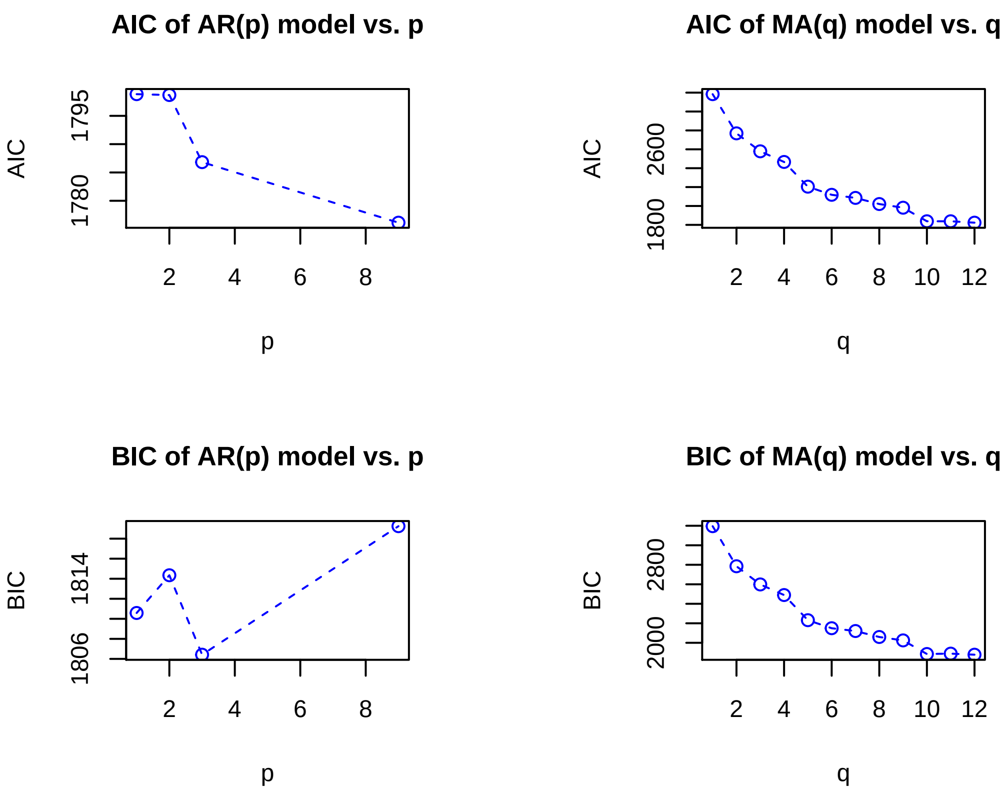

Figure 8: AIC and BIC values of the AR and MA models vs. the number of coefficients

Next we plot two similar graphs (only for the AIC; the one for the BIC is quite similar), one for all models, and only for the ARMA models (because including the MA models increases the scale so it's harder to compare the lengths of the bars). They both show that the model with the lowest AIC value is the ARMA(8,3) one. Do note that there are no values for half of the $p-q$ plane (since we limited our search to the region $p+q \leq 12$).

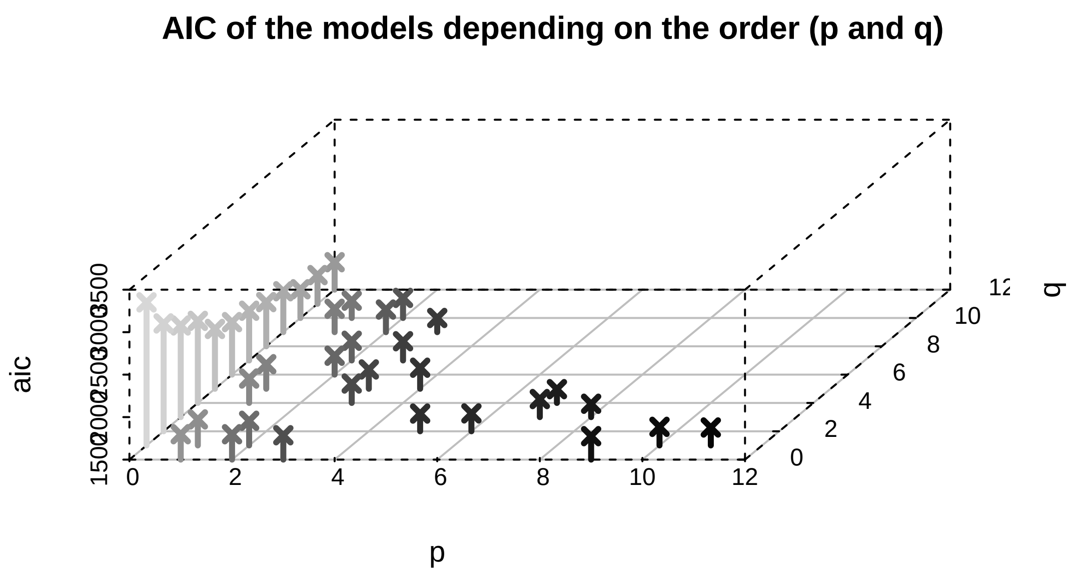

Figure 9: AIC of the models depending on the order (p and q)

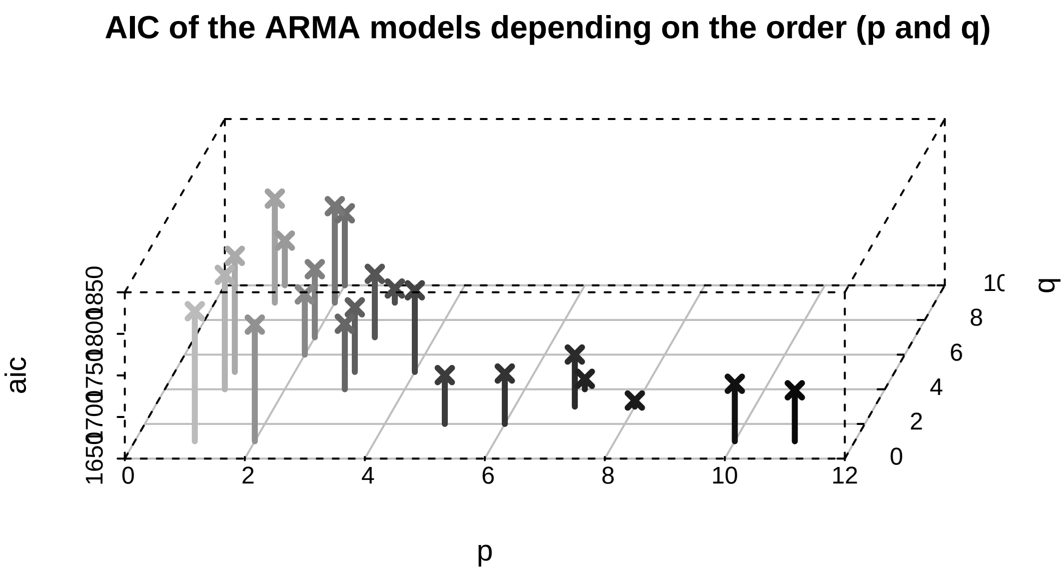

Figure 10: AIC of the ARMA models depending on the order (p and q)

A table (ordered by the AIC or BIC value, in increasing order) is more helpful to check which are the better models.

Table 3: Top 4 models per family, based on their AIC value

p	q	aic	bic	family
8	3	1657.3	1708.3	ARMA
7	4	1662.6	1713.6	ARMA
3	9	1667.0	1721.9	ARMA
1	10	1703.8	1754.7	ARMA
9	0	1776.1	1819.2	AR
3	0	1786.8	1806.4	AR
2	0	1798.7	1814.3	AR
1	0	1798.8	1810.6	AR
0	12	1823.2	1878.1	MA
0	11	1838.0	1889.0	MA
0	10	1838.2	1885.2	MA
0	9	1981.6	2024.7	MA

Table 4: Top 4 models per family, based on their BIC value

p	q	aic	bic	family
8	3	1657.3	1708.3	ARMA
7	4	1662.6	1713.6	ARMA
3	9	1667.0	1721.9	ARMA
5	2	1708.7	1744.0	ARMA
3	0	1786.8	1806.4	AR
1	0	1798.8	1810.6	AR
2	0	1798.7	1814.3	AR
9	0	1776.1	1819.2	AR
0	12	1823.2	1878.1	MA
0	10	1838.2	1885.2	MA

But the lowest AIC or BIC value may not necessarily involve the best model (with highest explanatory power). That criterion should be combined with others, such as how much the residuals of the model resemble a white noise (and then selecting the simplest model among those). The problem with this time series, due to its stationality, is that the correlogram of the residuals never look like that of a white noise: the autocorrelation at lag 12 (and multiples of 12) is always highly significant (unless we transformed the time series, which is not the purpose of this exercise). So we could (and will) inspect visually the correlograms, but most of them do not differ too much between one another, and it's hard to find the differences based on the heights in the correlogram. One approach—still far from perfect—that might be worth exploring is sum the absolute value of all the auto-correlations (and partial auto-correlations) that are significant (i.e., that exceed $2/\sqrt{n}$, the variance of the lag k autocorrelation—ρ_k—of a white noise, in absolute value).

```
sum_acf <- function(model) {
  # Get the ACFs of first 24 lags
  ACF <- acf(model$residuals, plot = FALSE, lag.max = 24)$acf
  # Exclude (assign 0) to those not significant
  significant_ACF <- ifelse(abs(ACF) < qnorm(.975) / sqrt(model$nobs), 0,
                            abs(ACF))
  # Sum absolute values (exluding lag 0)
  return(sum(significant_ACF[-1]))
}
sum_pacf <- function(model) {
  # Get the PACFs of first 24 lags
  PACF <- pacf(model$residuals, plot = FALSE, lag.max = 24)$acf
  # Exclude (assign 0) to those not significant
  significant_PACF <- ifelse(abs(PACF) < qnorm(.975) / sqrt(model$nobs), 0,
                             abs(PACF))
  # Sum absolute values
  return(sum(significant_PACF))
}
aic_list <- aic_list %>%
  mutate(ACF = sum_acf(Arima(hw08, order = c(p, 0, q))),
         PACF = sum_pacf(Arima(hw08, order = c(p, 0, q)))) %>%
  select(p, q, aic, bic, ACF, PACF, family)
```

Table 5: Top 5 models based on the sum of the absolute value of their (significant) auto-correlations

p	q	aic	bic	ACF	PACF	family
1	10	1703.8	1754.7	1.9	2.6	ARMA
9	0	1776.1	1819.2	2.1	2.2	AR
3	7	1726.3	1773.4	2.2	2.3	ARMA
4	5	1748.2	1791.3	2.3	2.7	ARMA
1	5	1789.6	1821.0	2.3	1.9	ARMA

Table 6: Top 5 models based on the sum of the absolute value of their (significant) partial auto-correlations

p	q	aic	bic	ACF	PACF	family
1	5	1789.6	1821.0	2.3	1.9	ARMA
1	4	1787.6	1815.0	2.8	2.1	ARMA
9	0	1776.1	1819.2	2.1	2.2	AR
2	6	1722.4	1761.6	2.5	2.3	ARMA
3	7	1726.3	1773.4	2.2	2.3	ARMA

We find some new models (as well as others that we already considered such as AR(9)) which do not have the lowest AIC or BIC value, but that have lowest significant autocorrelations, overall, and hence look closer to a white noise (whose ideal value, for the 2 parameters shown above, would be close to zero[1]). We add the best one with regards to each criterion (ACF: **ARMA(1,10)**; PACF: **ARMA(1,5)**) to the list of models to consider.

```
ar_models_coefs <- data.frame(aic_list %>% arrange(aic) %>%
                              filter(family == "AR") %>%
                              top_n(-2, aic))[, 1]
ma_models_coefs <- data.frame(aic_list %>% arrange(bic) %>%
                              filter(family == "MA") %>%
                              top_n(-2, bic))[, 2]
arma_models_coefs <- data.frame(aic_list %>% arrange(bic) %>%
                                filter(family == "ARMA") %>%
                                top_n(-4, bic))[, 1:2] %>%
  rbind(data.frame(aic_list %>% arrange(ACF))[1, 1:2]) %>%
  rbind(data.frame(aic_list %>% arrange(PACF))[1, 1:2])
ar_models <- lapply(ar_models_coefs, function(p)
  Arima(hw08.ts, order = c(p, 0, 0)))
ma_models <- lapply(ma_models_coefs, function(q)
  Arima(hw08.ts, order = c(0, 0, q)))
arma_models <- apply(arma_models_coefs, 1, function(arma_coef)
  Arima(hw08.ts, order = c(arma_coef[1], 0, arma_coef[2])))
```

[1] Not exactly zero because 5% of the auto-correlations, on average, will be significant.

2.2 AR Models

2.2.1 Estimation

We start with the best AR model according to the BIC value: **AR(3)**.

```
ar3 <- ar_models[[2]]
```

Table 7: Coefficients, SEs, and 95% CIs of the estimated AR(3) model

	Coefficient	SE	95% CI lower	95% CI upper
ar1	0.9061	0.0511	0.8038	1.0083
ar2	-0.0994	0.0692	-0.2379	0.0390
ar3	0.1922	0.0511	0.0900	0.2945
intercept	91.5517	45.2799	0.9918	182.1115

Note that the 2nd coefficient is not significant (its 95% confidence interval includes zero).

```
(roots_ar <- polyroot(c(1, -ar3$coef[1:(length(ar3$coef)-1)])))

## [1]  1.000870-0.000000i -0.241821+2.266871i -0.241821-2.266871i

all(Mod(roots_ar) > 1) # Stationarity condition

## [1] TRUE
```

If we consider the **AR(9)** model instead, the last 5 coefficients (5 to 9) and 2 others (2 and 3) are not significant:

```
ar9 <- ar_models[[1]]
```

Table 8: Coefficients, SEs, and 95% CIs of the estimated AR(9) model

	Coefficient	SE	95% CI lower	95% CI upper
ar1	0.8536	0.0518	0.7500	0.9571
ar2	-0.0680	0.0680	-0.2040	0.0679
ar3	0.0542	0.0679	-0.0816	0.1900
ar4	0.1888	0.0680	0.0527	0.3248
ar5	0.0694	0.0685	-0.0677	0.2064
ar6	-0.0362	0.0687	-0.1736	0.1012
ar7	0.0978	0.0695	-0.0411	0.2367
ar8	-0.0615	0.0697	-0.2009	0.0779
ar9	-0.0993	0.0526	-0.2046	0.0060
intercept	92.5837	43.8900	4.8036	180.3638

```
(roots_ar <- polyroot(c(1, -ar9$coef[1:(length(ar9$coef)-1)])))
```

```
## [1]  0.667996+1.017786i -1.162291+0.812354i -0.088607-1.271074i
## [4]  1.001209+0.000000i -0.088607+1.271074i -1.162291-0.812354i
## [7]  0.667996-1.017786i  1.232324-0.000000i -1.686545+0.000000i
```

```
all(Mod(roots_ar) > 1) # Stationarity condition
```

```
## [1] TRUE
```

2.2.2 Diagnostics using Residuals

If we examine the residuals of the **AR(3)** model we observe that, though their distribution looks like normal, they follow a tend and the variance seems to increase over time. Their ACF and PACF do not look like the ones of white noise (especially—but not only—because the ACF and PACF at lag 12).

```
summary(ar9$resid)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -11.2800  -0.8212   0.3881   0.2913   1.7670   9.0830
```

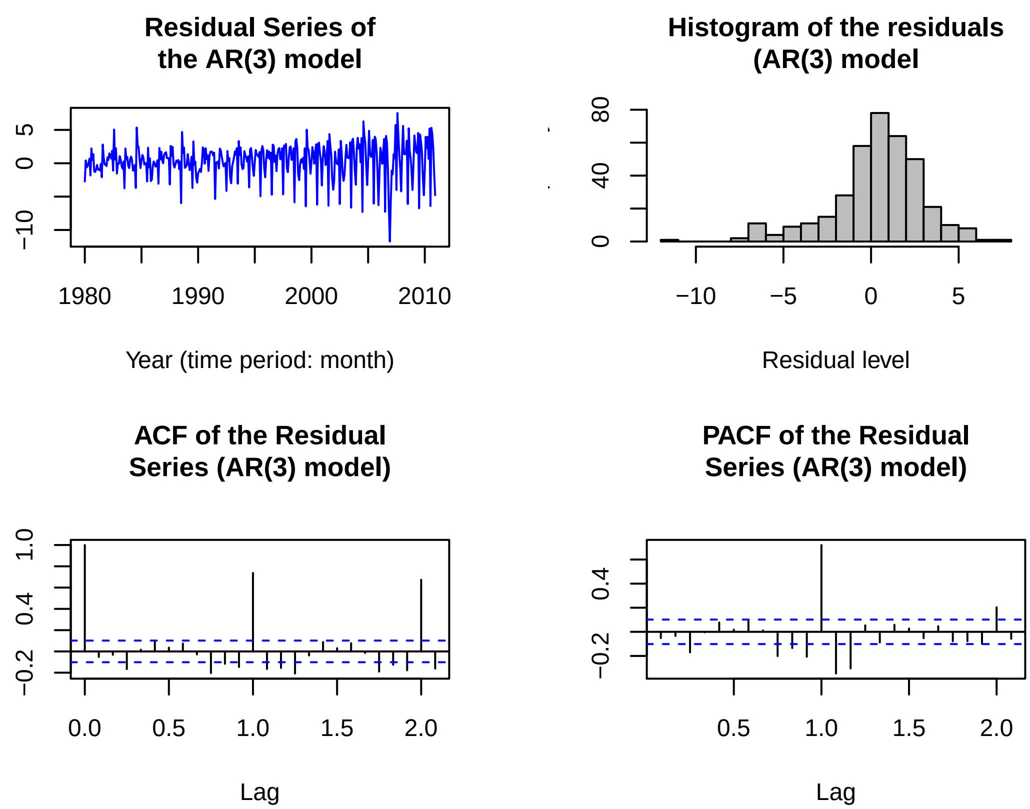

Figure 11: AR(3) model diagnostic based on the residuals

Nonetheless, we cannot reject the hypothesis of independence of the residual series:

```
Box.test(ar3$resid, type = "Ljung-Box")
```

```
##
##  Box-Ljung test
##
## data:  ar3$resid
## X-squared = 1.0155, df = 1, p-value = 0.3136
```

The residuals of the **AR(9)** model look pretty much the same as those of the AR(3) model.

```
summary(ar9$resid)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -11.2800  -0.8212   0.3881   0.2913   1.7670   9.0830
```

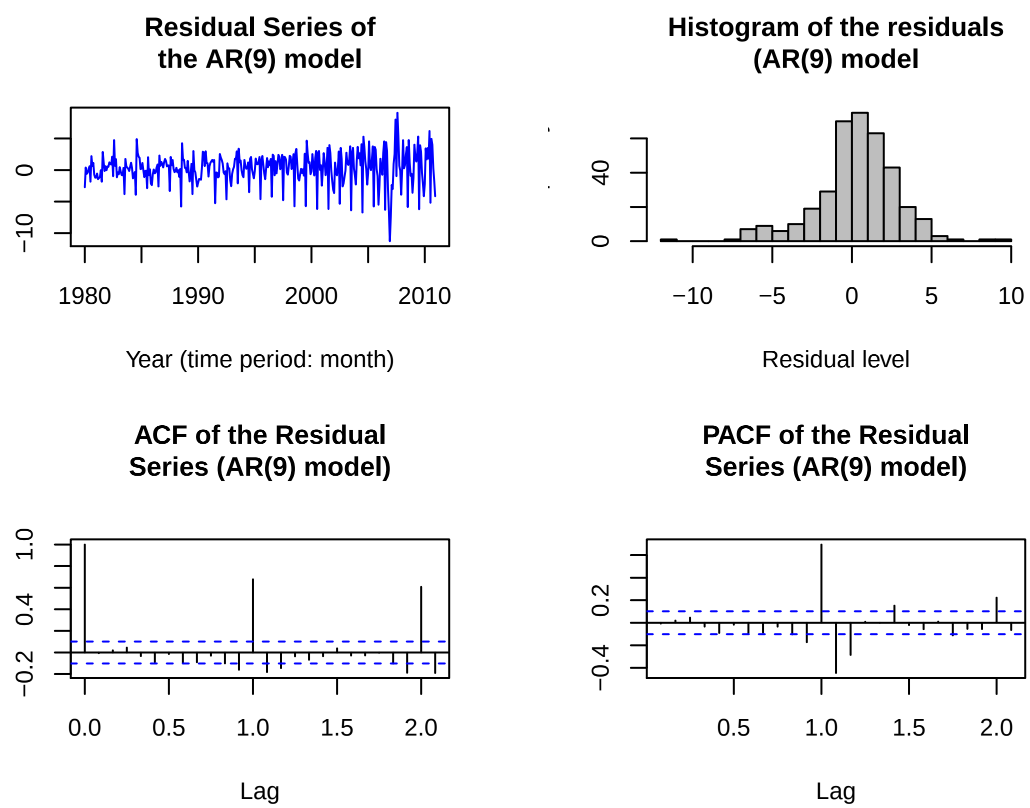

Figure 12: AR(9) model diagnostic based on the residuals

Again, we cannot reject the hypothesis of independence of the residual series (and now the p-value of the Box-Ljung test is even less significant, close to one):

```
Box.test(ar9$resid, type = "Ljung-Box")
```

```
##
##  Box-Ljung test
##
## data:  ar9$resid
## X-squared = 0.018184, df = 1, p-value = 0.8927
```

2.2.3 Model Performance Evaluation

2.2.3.1 In-sample fit

Despite the fact that the original series is not stationary and (hence) the residuals of the **AR(3)** model do not resemble a white noise, the in-sample fit looks reasonable. . . though not completely: the estimated series is lagged 1 period (compare each value in the first column of the following table with the value in the second column and the next row).

Time	Original series	Estimated series	Residuals
Jul 2010	141.9	148.3	-6.4
Aug 2010	146.9	141.6	5.3
Sep 2010	152.0	147.7	4.3
Oct 2010	152.6	150.5	2.1
Nov 2010	149.7	151.5	-1.8
Dec 2010	145.0	149.8	-4.8

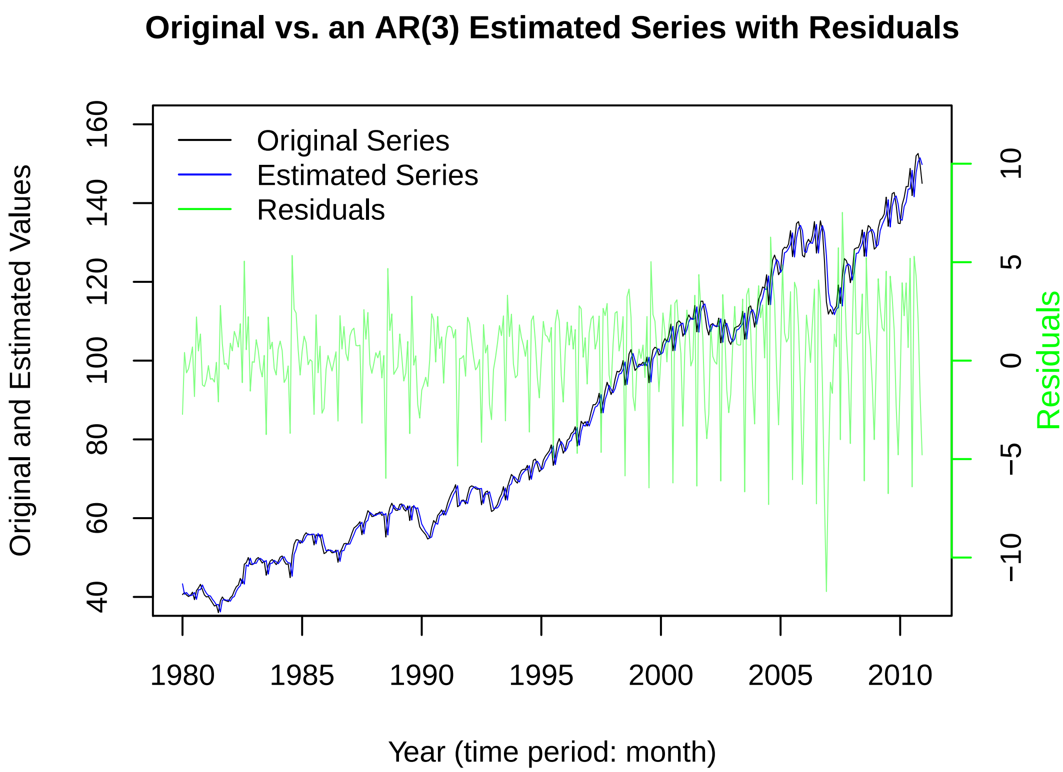

Figure 13: AR(3) model performance evaluation (in-sample)

The same happens with the **AR(9)** model:

Time	Original series	Estimated series	Residuals
Jul 2010	141.9	147.1	-5.2
Aug 2010	146.9	141.9	5.0
Sep 2010	152.0	148.0	4.0
Oct 2010	152.6	152.3	0.3
Nov 2010	149.7	151.4	-1.7
Dec 2010	145.0	149.1	-4.1

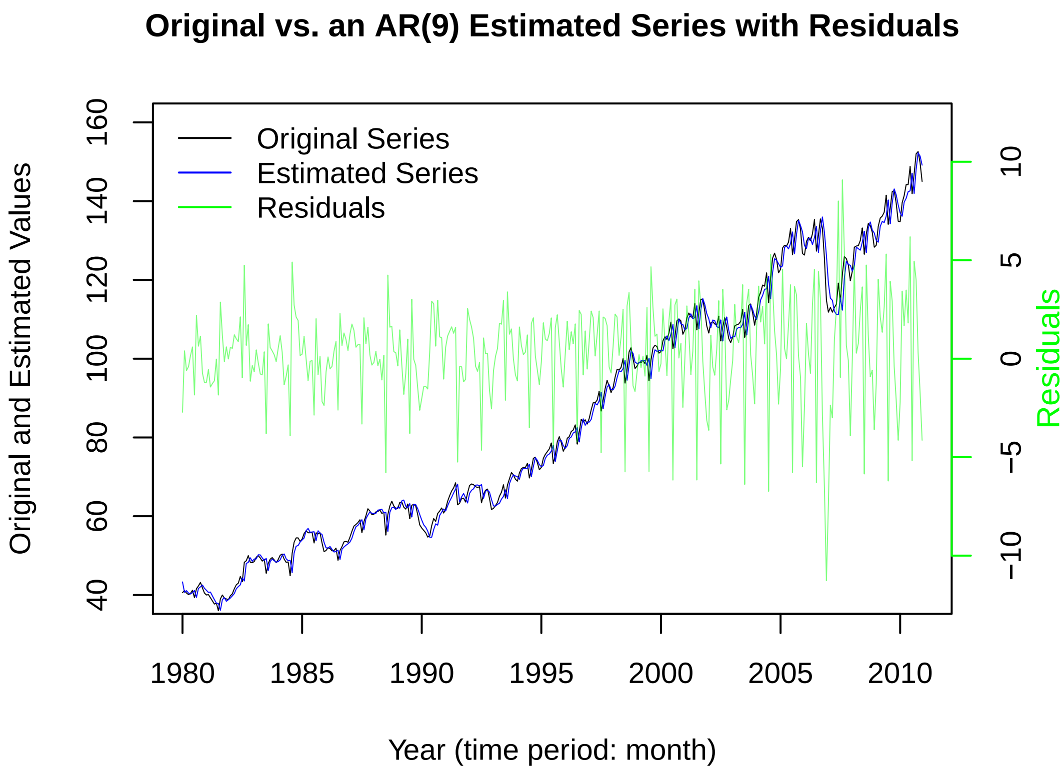

Figure 14: AR(9) model performance evaluation (in-sample)

2.2.3.2 Out-of-sample fit

The time series has 372 observations (which we believe correspond to 31 years of monthly observations). To evaluate the out-of-sample fit we will build the model without the last 10% observations or so: that would exclude 37.2 observations, but we'll limit that number to 36 (supposedly 3 years: 2008 to 2010 under our arbitrary assumption that the time series starts in 1980).

> If we repeated the process in 2.1 with this shortened version of the time series we might get a different AR model, but the purpose of this step is to evaluate our selected model, not a potentially different one.

```
hw08.ts_train <- window(hw08.ts, start = 1980, end=c(2007,12))
hw08.ts_test <- window(hw08.ts, start = 2008)
(ar3.oos.fit <- Arima(hw08.ts_train, order = c(ar_models_coefs[2], 0, 0)))
```

```
## Series: hw08.ts_train
## ARIMA(3,0,0) with non-zero mean
##
## Coefficients:
##          ar1      ar2     ar3  intercept
##       0.9139  -0.0980  0.1823    82.4741
## s.e.  0.0538   0.0732  0.0538    33.3170
##
## sigma^2 estimated as 6.159:  log likelihood=-784.92
## AIC=1579.83   AICc=1580.01   BIC=1598.92
```

Time	Original series	Estimated series	Residuals
Jan 1980	40.6	43.4	-2.8
Feb 1980	41.1	40.7	0.4
Mar 1980	40.5	41.1	-0.6
Apr 1980	40.1	40.5	-0.4
May 1980	40.4	40.3	0.1
Jun 1980	41.2	40.5	0.7

```
ar3.oos.fit.fcast <- forecast.Arima(ar3.oos.fit, h = 36)
(acc_ar3 <- accuracy(ar3.oos.fit.fcast, hw08.ts_test))
```

```
##                     ME      RMSE       MAE        MPE      MAPE     MASE
## Training set  0.288027  2.481795  1.834386  0.3020873  2.335485 0.362138
## Test set     17.459603 19.197181 17.459603 12.4368690 12.436869 3.446814
##                     ACF1 Theil's U
## Training set -0.04395443        NA
## Test set      0.82318235  4.991299
```

Though the **AR(3)** model fitted quite well, the out-of-sample forecast is not very good on the long-term, but it is on the short-term: almost all the values of the time series for the 1st year out-of-sample (2008 in the Figure below) are within the 95% confidence interval (so the difference with the forecast is not statistically significant), but most of them for the remaining 2 years left out of the sample are outside that region. Also compare the RMSE, MAE, and other parameters, for both the training and test sets, which were shown in the previous page.

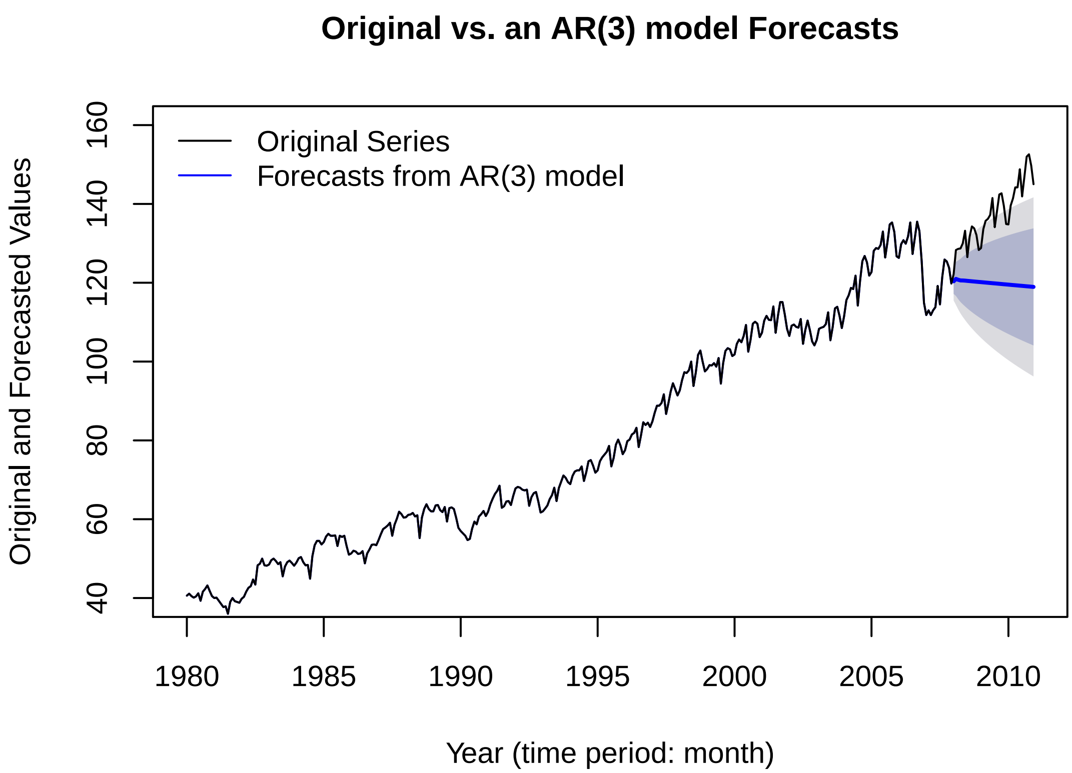

Figure 15: AR(3) model performance evaluation (out of-sample)

The **AR(9)** model is a better (out-of-sample) fit: the mean of the forecasts is still decreasing (while the trend of the original series is positive during the last 3 years) but the slope is lower, and as a result most of the original values (with the exception of the last ones, that would correspond to 2010) are within the 95% confidence interval.

```
(ar9.oos.fit <- Arima(hw08.ts_train, order = c(ar_models_coefs[1], 0, 0)))
```

```
## Series: hw08.ts_train
## ARIMA(9,0,0) with non-zero mean
##
## Coefficients:
##          ar1      ar2     ar3     ar4     ar5      ar6     ar7      ar8
##       0.8735  -0.0714  0.0486  0.1764  0.0637  -0.0595  0.1089  -0.0656
## s.e.  0.0546   0.0724  0.0722  0.0724  0.0730   0.0733  0.0742   0.0745
##           ar9  intercept
##       -0.0766    80.8778
## s.e.   0.0560    32.4706
```

```
##
## sigma^2 estimated as 5.864:  log likelihood=-776.83
## AIC=1575.66   AICc=1576.47   BIC=1617.65
```

```
ar9.oos.fit.fcast <- forecast.Arima(ar9.oos.fit, h = 36)
(acc_ar9 <- accuracy(ar9.oos.fit.fcast, hw08.ts_test))
```

```
##                      ME     RMSE       MAE        MPE      MAPE      MASE
## Training set  0.2594558  2.42164  1.761408  0.2654737  2.250817 0.3477311
## Test set     14.9442122 16.97655 14.944212 10.5974519 10.597452 2.9502344
##                      ACF1 Theil's U
## Training set -0.005828742        NA
## Test set      0.833604573  4.391924
```

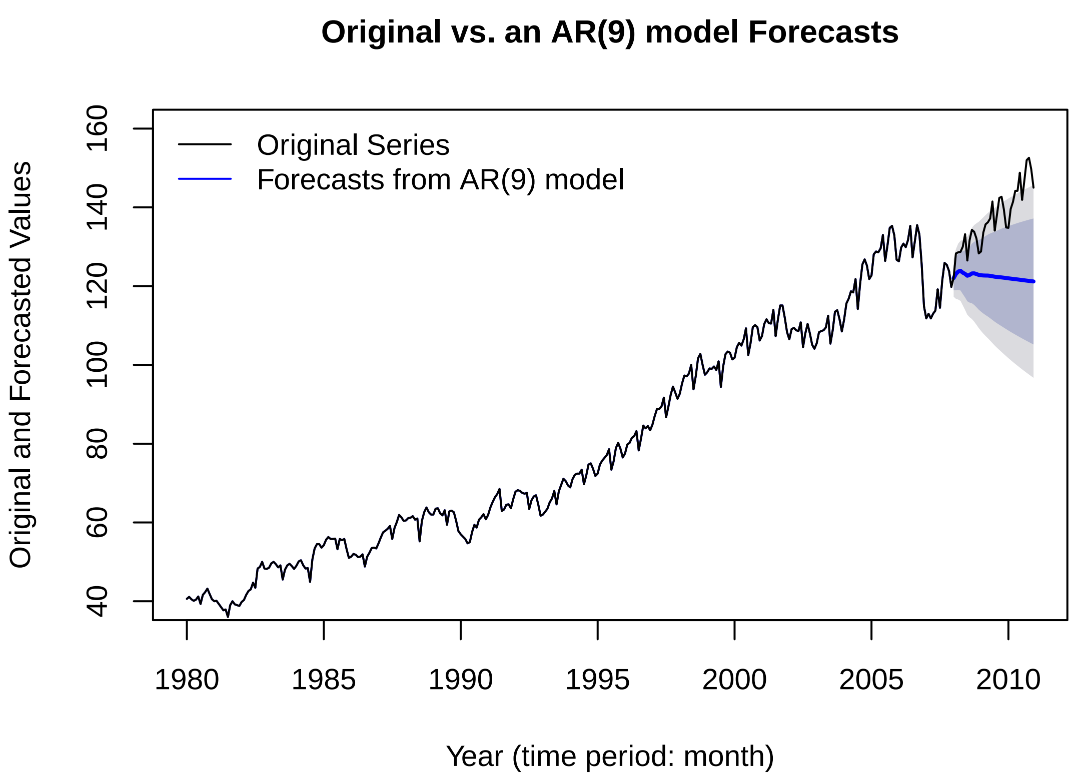

Figure 16: AR(9) model performance evaluation (out of-sample)

2.3 MA Models

2.3.1 Estimation

We start with the best MA model based on both criteria (AIC and BIC): **MA(12)**. As shown below, all its coefficients are significant(ly different from zero).

```
ma12 <- ma_models[[1]]
```

Table 12: Coefficients, SEs, and 95% CIs of the estimated MA(12) model

	Coefficient	SE	95% CI lower	95% CI upper
ma1	1.6018	0.0610	1.4797	1.7239
ma2	2.2564	0.0998	2.0569	2.4560
ma3	2.7434	0.1331	2.4772	3.0095
ma4	3.1213	0.1618	2.7976	3.4449
ma5	3.1775	0.1853	2.8069	3.5482
ma6	3.0715	0.1920	2.6876	3.4555
ma7	2.8412	0.1857	2.4697	3.2127
ma8	2.3741	0.1716	2.0308	2.7174
ma9	1.7471	0.1482	1.4508	2.0434
ma10	0.9415	0.1154	0.7107	1.1722
ma11	0.2418	0.0819	0.0780	0.4057
ma12	0.2666	0.0564	0.1538	0.3794
intercept	84.9583	3.4143	78.1296	91.7870

```
(roots_ma <- polyroot(c(1, ma12$coef[1:(length(ma12$coef)-1)])))
```

```
##  [1]  0.7285230+0.7062992i -0.7243000+0.6894863i -0.7243000-0.6894863i
##  [4]  0.7285230-0.7062992i  0.2500503+0.9802538i -1.0018400+0.2610601i
##  [7]  0.2500503-0.9802538i -0.3113504-0.9888311i -0.3113504+0.9888311i
## [10] -1.0018400-0.2610601i  0.6053508-1.6503610i  0.6053508+1.6503610i
```

```
all(Mod(roots_ma) > 1) # Invertibility condition
```

```
## [1] TRUE
```

The **MA(10)** had only slighlty higher AIC and BIC values. Its coefficients are also significant (and, as expected, not very different from the first 10 coefficients of the MA(12) model).

```
ma10 <- ma_models[[2]]
```

Table 13: Coefficients, SEs, and 95% CIs of the estimated MA(10) model

	Coefficient	SE	95% CI lower	95% CI upper
ma1	1.7752	0.0366	1.7020	1.8484
ma2	2.4564	0.0761	2.3042	2.6086
ma3	2.8855	0.1076	2.6702	3.1008
ma4	3.2105	0.1172	2.9761	3.4450
ma5	3.2417	0.1174	3.0068	3.4766
ma6	3.1000	0.1139	2.8722	3.3278
ma7	2.7966	0.1117	2.5733	3.0200
ma8	2.2592	0.1013	2.0566	2.4619
ma9	1.6077	0.0673	1.4730	1.7423
ma10	0.8203	0.0333	0.7537	0.8869
intercept	84.9553	3.4698	78.0158	91.8949

```
(roots_ma <- polyroot(c(1, -ma10$coef[1:(length(ma10$coef)-1)])))
```

```
##  [1]  0.3273636-0.0000000i -1.0168099+0.6504934i -0.4617923-1.0899515i
##  [4]  0.7434476-0.7765943i  0.7434476+0.7765943i -1.2182180+0.0000000i
##  [7]  0.2006390-1.1203328i  0.2006390+1.1203328i -0.4617923+1.0899515i
## [10] -1.0168099-0.6504934i
```

```
all(Mod(roots_ma) > 1) # Stationarity condition
```

```
## [1] FALSE
```

2.3.2 Diagnostics using Residuals

The residuals of the **MA(12)** model do not look exactly like those of a white noise: the time plot shows a growing trend, the histogram is right-skewed, and many of the auto-correlations (apart from $k = 0$) and partial auto-correlations are significantly different from zero.

```
summary(ma12$resid)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -6.62200 -1.72800 -0.36740  0.03438  1.58600  7.63200
```

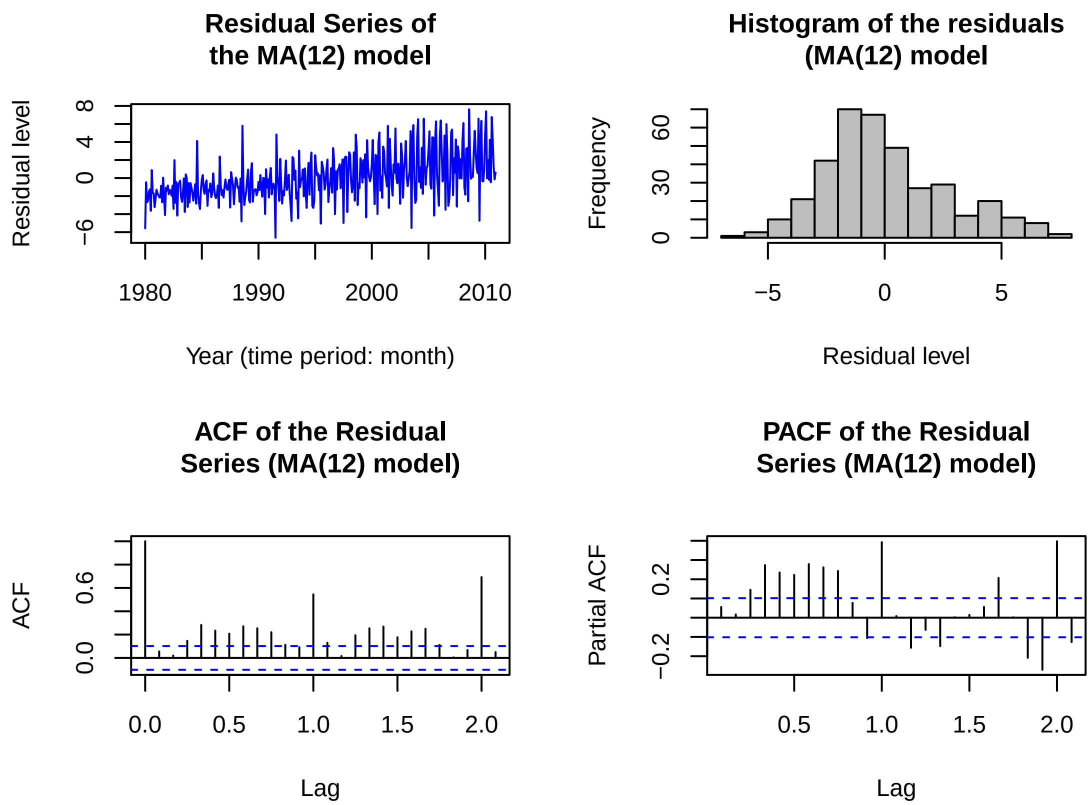

Figure 17: MA(12) model diagnostic based on the residuals

The result of a Llung-Box test is that we cannot reject the hypothesis of independence of the residual series:

```
Box.test(ma12$resid, type = "Ljung-Box")
```

```
##
##  Box-Ljung test
##
## data:  ma12$resid
## X-squared = 1.1728, df = 1, p-value = 0.2788
```

Something very similar happens with the **MA(10)**) model.

```
summary(ma10$resid)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -6.47600 -1.71100 -0.47050  0.03308  1.38500  9.91800
```

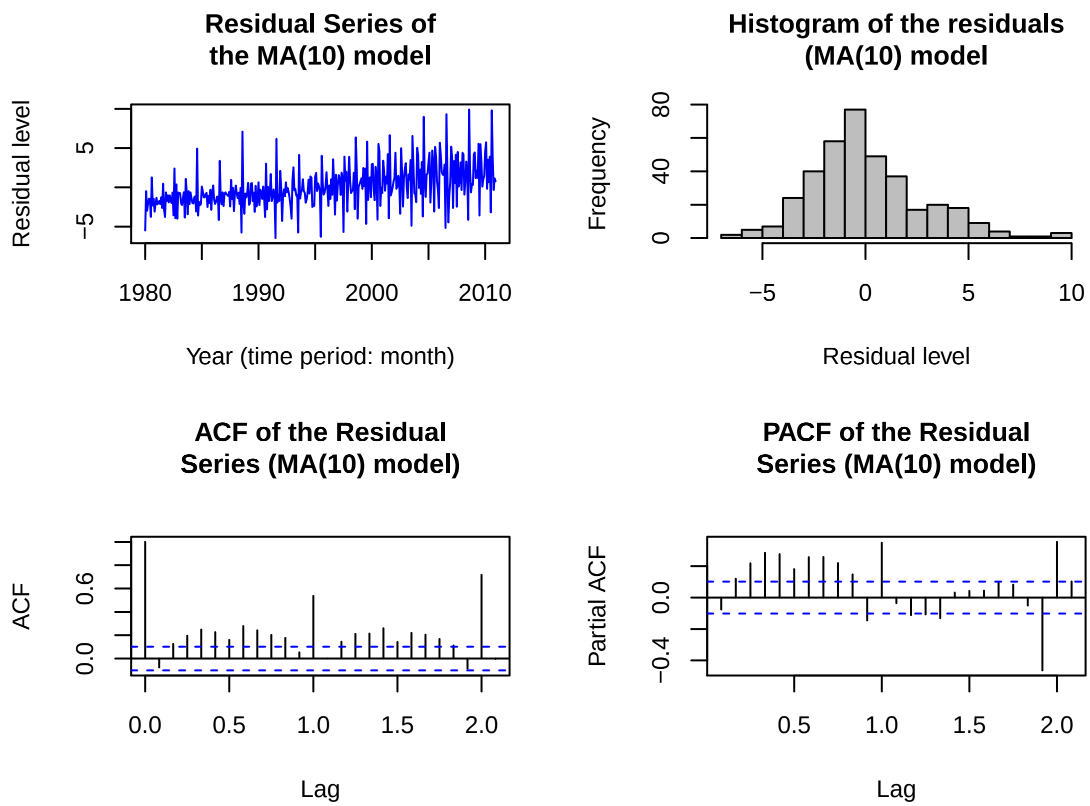

Figure 18: MA(10) model diagnostic based on the residuals

```
Box.test(ma10$resid, type = "Ljung-Box")
```

```
##
##  Box-Ljung test
##
## data:  ma10$resid
## X-squared = 2.1716, df = 1, p-value = 0.1406
```

2.3.3 Model Performance Evaluation

2.3.3.1 In-sample fit

Surprisingly, the **MA(12)** model fits the data almost as well as the AR models we've analyzed (it captures the trend and seasonality, but it's approximately lagged 1 time period), but since its residuals took higher values (and were more volatile), the differences between the fitted and original values are greater.

Time	Original series	Estimated series	Residuals
Jan 1980	40.6	46.2	-5.6
Feb 1980	41.1	41.6	-0.5
Mar 1980	40.5	43.2	-2.7
Apr 1980	40.1	42.6	-2.5
May 1980	40.4	41.9	-1.5
Jun 1980	41.2	42.5	-1.3

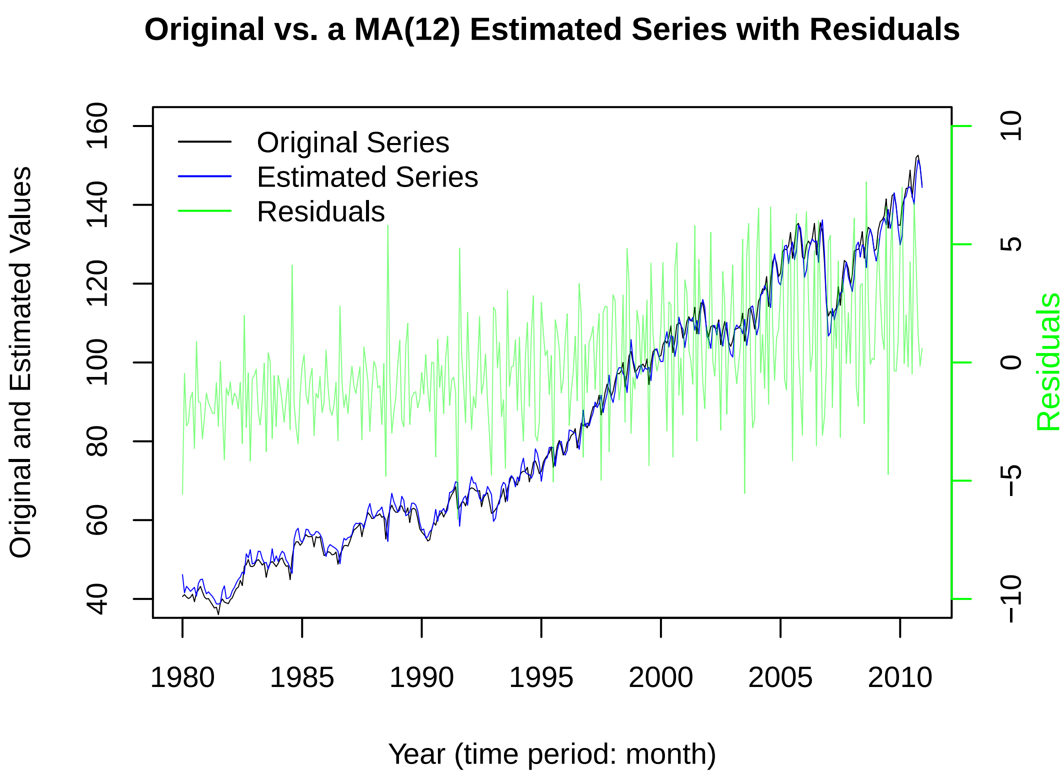

Figure 19: MA(12) model performance evaluation (in-sample)

Again, there is not much difference with the **MA(10)** model.

Time	Original series	Estimated series	Residuals
Jan 1980	40.6	46.1	-5.5
Feb 1980	41.1	41.6	-0.5
Mar 1980	40.5	43.5	-3.0
Apr 1980	40.1	42.2	-2.1
May 1980	40.4	41.9	-1.5
Jun 1980	41.2	42.7	-1.5

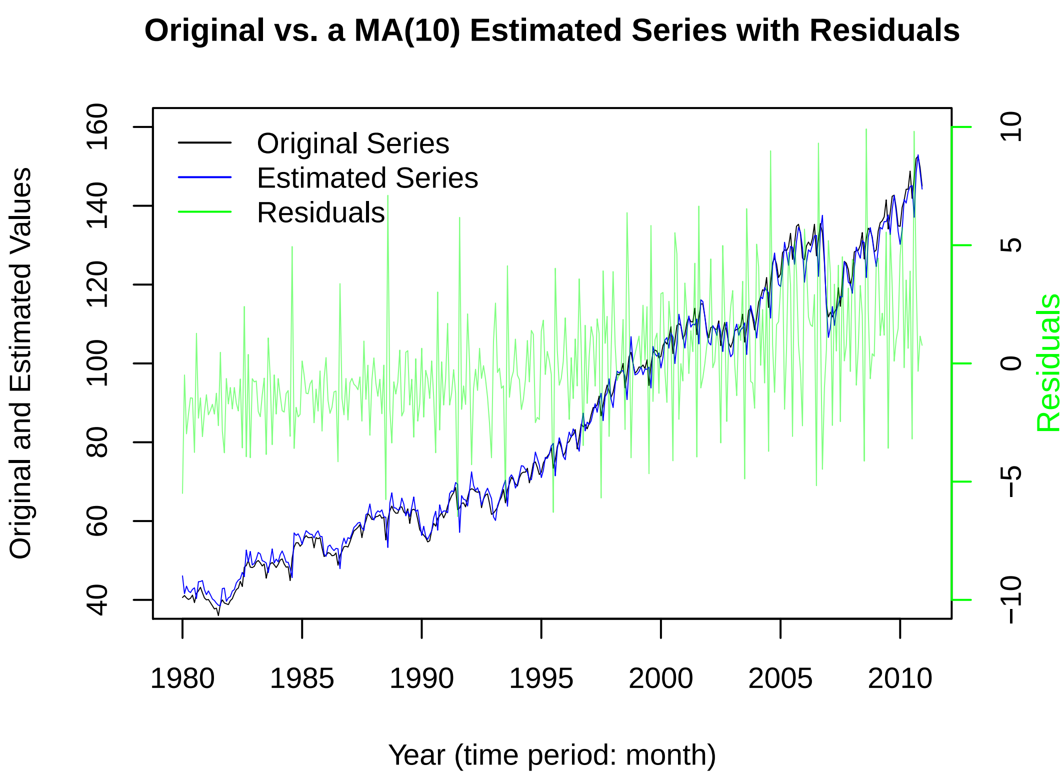

Figure 20: MA(10) model performance evaluation (in-sample)

2.3.3.2 Out-of-sample fit

The out-of-sample fit of the **MA(12)** is horrible (see the Figure in the next page). This is because the original time series is not mean-stationary but has an increasing trend, while MA models are stationary (and hence the value of their realizations may be above or below the mean, but does not deviate much from it in the long-term), so their forecasts always tend (after a few observations) to the mean of the original time series (84.8 in this case), while the time series keeps increasing. As a result, and though the 95% confidence interval comprises a very wide region, it does not include any of the original values.

```
(ma12.oos.fit <- Arima(hw08.ts_train, order = c(0, 0, ma_models_coefs[1])))
```

```
## Series: hw08.ts_train
## ARIMA(0,0,12) with non-zero mean
##
## Coefficients:
##          ma1     ma2     ma3     ma4     ma5     ma6     ma7     ma8
##       1.5601  2.1758  2.6189  2.9784  2.9965  2.8647  2.6605  2.2204
## s.e.  0.0625  0.1074  0.1456  0.1745  0.1965  0.2027  0.1944  0.1778
##          ma9    ma10    ma11    ma12  intercept
##       1.6022  0.8326  0.2048  0.2685    79.2168
## s.e.  0.1489  0.1139  0.0909  0.0609     3.1802
##
## sigma^2 estimated as 6.086:  log likelihood=-790.54
## AIC=1609.09   AICc=1610.4   BIC=1662.53
```

Time	Original series	Estimated series	Residuals
Jan 1980	40.6	45.7	-5.1
Feb 1980	41.1	41.5	-0.4
Mar 1980	40.5	42.9	-2.4
Apr 1980	40.1	42.4	-2.3
May 1980	40.4	41.8	-1.4
Jun 1980	41.2	42.4	-1.2

```
ma12.oos.fit.fcast <- forecast.Arima(ma12.oos.fit, h = 36)
(acc_ma12 <- accuracy(ma12.oos.fit.fcast, hw08.ts_test))
```

```
##                        ME     RMSE       MAE      MPE      MAPE       MASE
## Training set  0.03470022  2.46705  1.943478 -0.55356  2.612738  0.3836747
## Test set     53.09921645 55.50212 53.099216 38.22458 38.224576 10.4826628
##                    ACF1 Theil's U
## Training set 0.05601392        NA
## Test set     0.82932838  14.68666
```

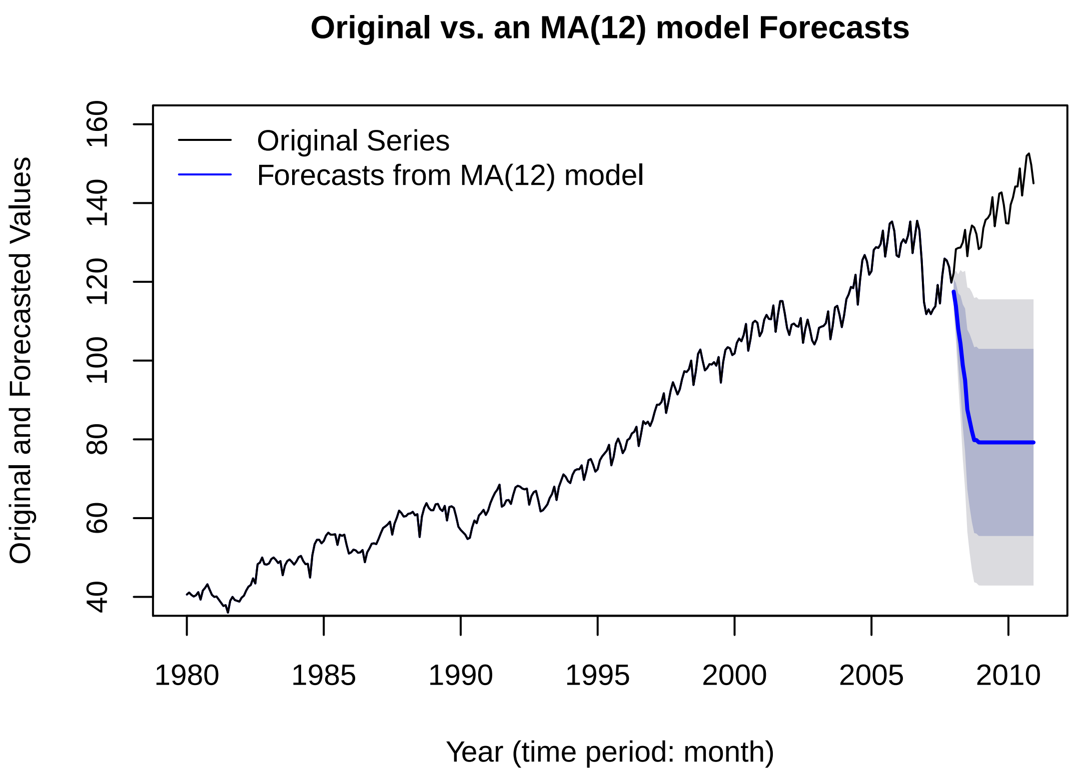

Figure 21: MA(12) model performance evaluation (out of-sample)

The **MA(10)**) model does not do a much better job.

```
(ma10.oos.fit <- Arima(hw08.ts_train, order = c(0, 0, ma_models_coefs[2])))
```

```
## Series: hw08.ts_train
## ARIMA(0,0,10) with non-zero mean
##
## Coefficients:
##          ma1     ma2     ma3     ma4     ma5     ma6     ma7     ma8
##       1.7470  2.3906  2.7867  3.1214  3.1742  3.0178  2.6761  2.1470
## s.e.  0.0468  0.0881  0.1058  0.1320  0.1391  0.1381  0.1356  0.1039
##          ma9    ma10  intercept
##       1.5559  0.8028    79.2904
## s.e.  0.0748  0.0517     3.3419
##
## sigma^2 estimated as 6.476:  log likelihood=-799.88
## AIC=1623.77   AICc=1624.74   BIC=1669.57
```

```
ma10.oos.fit.fcast <- forecast.Arima(ma10.oos.fit, h = 36)
(acc_ma10 <- accuracy(ma10.oos.fit.fcast, hw08.ts_test))
```

```
##                      ME     RMSE      MAE        MPE     MAPE
## Training set 0.03210533 2.544719 1.926767 -0.5424929 2.618881
```

```
## Test set     52.63498261 55.211518 52.634983 37.8700136 37.870014
##                    MASE        ACF1 Theil's U
## Training set  0.3803756 -0.09181661        NA
## Test set     10.3910153  0.84375222  14.59938
```

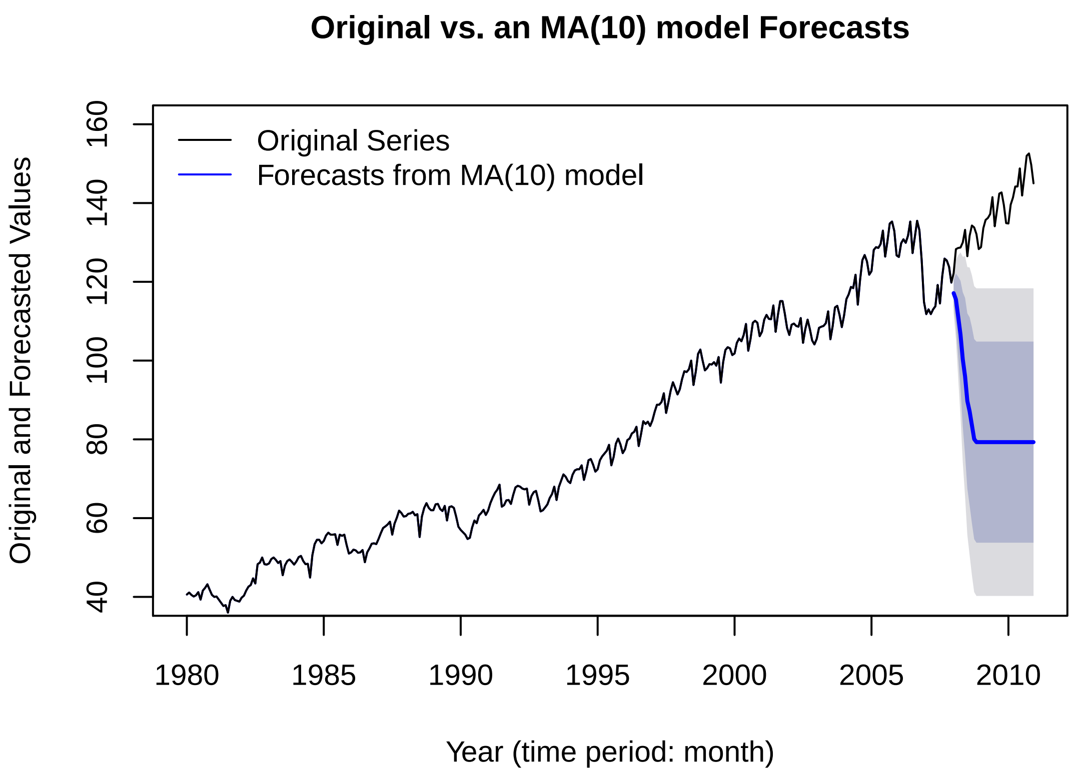

Figure 22: MA(10) model performance evaluation (out of-sample)

2.4 ARMA Models

2.4.1 Estimation

Finally, we analyze the four ARMA models that we mentioned in 2.1: the three models with the lowest values of both the AIC and the BIC, and the ARMA(5,2) model, which is the 4th one with the lowest BIC value (but fewer coefficients).

All the AR coefficients of the **ARMA(8,3)** model, except the 4th, are not statistically significant.

```
arma83 <- arma_models[[1]]
```

Table 17: Coefficients, SEs, and 95% CIs of the estimated ARMA(8,3) model

	Coefficient	SE	95% CI lower	95% CI upper
ar1	-0.0889	0.0507	-0.1904	0.0126
ar2	-0.0629	0.0501	-0.1631	0.0374
ar3	-0.0352	0.0503	-0.1358	0.0655
ar4	0.6742	0.0497	0.5748	0.7735
ar5	0.0881	0.0507	-0.0134	0.1895
ar6	0.0667	0.0499	-0.0331	0.1664
ar7	0.0367	0.0503	-0.0639	0.1373
ar8	0.3213	0.0500	0.2213	0.4212
ma1	1.0557	0.0192	1.0173	1.0941
ma2	1.0657	0.0171	1.0316	1.0998
ma3	0.9892	0.0156	0.9580	1.0204
intercept	86.1417	NaN	NaN	NaN

```
(roots_ar <- polyroot(c(1, -arma83$coef[1:arma_models_coefs[1, 1]])))

## [1]  0.8829221+1.001065i -1.0003007+0.000000i -0.0004076-1.001681i
## [4]  1.0000101-0.000000i -0.0004076+1.001681i -0.9395001+0.926276i
## [7] -0.9395001-0.926276i  0.8829221-1.001065i

all(Mod(roots_ar) > 1) # Stationarity condition

## [1] TRUE

(roots_ma <- polyroot(c(1, arma83$coef[(arma_models_coefs[1, 1] + 1):
                                        (arma_models_coefs[1, 1] +
                                           arma_models_coefs[1, 2])])))

## [1] -0.0332301+0.9994752i -1.0108738+0.0000000i -0.0332301-0.9994752i

all(Mod(roots_ma) > 1) # Invertibility condition

## [1] TRUE
```

Two of the AR coefficients of the **ARMA(7,4)** model, as well as the mean, are not statistically significant.

```
arma74 <- arma_models[[2]]
```

Table 18: Coefficients, SEs, and 95% CIs of the estimated ARMA(7,4) model

	Coefficient	SE	95% CI lower	95% CI upper
ar1	-0.8930	NaN	NaN	NaN
ar2	-0.1169	0.0701	-0.2572	0.0233
ar3	0.1292	0.0503	0.0285	0.2298
ar4	0.9923	0.0082	0.9758	1.0087
ar5	0.8861	NaN	NaN	NaN
ar6	0.1244	0.0704	-0.0164	0.2652
ar7	-0.1219	0.0490	-0.2200	-0.0239
ma1	1.9494	NaN	NaN	NaN
ma2	2.0239	0.0136	1.9967	2.0511
ma3	1.9388	NaN	NaN	NaN
ma4	0.8714	NaN	NaN	NaN
intercept	85.0538	NaN	NaN	NaN

```r
(roots_ar <- polyroot(c(1, -arma74$coef[1:arma_models_coefs[2, 1]])))
```

```
## [1]  0.000430+1.003870i -1.000378-0.000000i  0.000430-1.003870i
## [4]  1.000001-0.000000i -1.308022+0.725508i -1.308022-0.725508i
## [7]  3.635815-0.000000i
```

```r
all(Mod(roots_ar) > 1) # Stationarity condition
```

```
## [1] TRUE
```

```r
(roots_ma <- polyroot(c(1, arma74$coef[(arma_models_coefs[2, 1] + 1):
                                        (arma_models_coefs[2, 1] +
                                           arma_models_coefs[2, 2])])))
```

```
## [1] -0.0408253+0.9991898i -1.1005315+0.0000000i -0.0408253-0.9991898i
## [4] -1.0426797-0.0000000i
```

```r
all(Mod(roots_ma) > 1) # Invertibility condition
```

```
## [1] TRUE
```

In the **ARMA(3,9)** model, it's some of their MA coefficients which are not significant.

```
arma39 <- arma_models[[3]]
```

Table 19: Coefficients, SEs, and 95% CIs of the estimated ARMA(3,9) model

	Coefficient	SE	95% CI lower	95% CI upper
ar1	-0.6093	0.0429	-0.6952	-0.5234
ar2	0.7836	0.0247	0.7341	0.8330
ar3	0.8244	0.0472	0.7301	0.9187
ma1	1.6918	0.0573	1.5772	1.8064
ma2	0.8174	0.0978	0.6217	1.0131
ma3	-0.0955	0.0996	-0.2947	0.1036
ma4	-0.0256	0.0980	-0.2215	0.1703
ma5	-0.4382	0.1047	-0.6477	-0.2287
ma6	-0.7110	0.1053	-0.9217	-0.5003
ma7	0.2189	0.1042	0.0105	0.4273
ma8	0.8918	0.0867	0.7184	1.0652
ma9	0.3367	0.0538	0.2292	0.4443
intercept	90.8452	80.0698	-69.2944	250.9848

```
(roots_ar <- polyroot(c(1, -arma39$coef[1:arma_models_coefs[3, 1]])))
```

```
## [1]  1.0003869+0.000000i -0.9754081+0.510959i -0.9754081-0.510959i
```

```
all(Mod(roots_ar) > 1) # Stationarity condition
```

```
## [1] TRUE
```

```
(roots_ma <- polyroot(c(1, arma39$coef[(arma_models_coefs[3, 1] + 1):
                                       (arma_models_coefs[3, 1] +
                                          arma_models_coefs[3, 2])])))
```

```
## [1]  0.2489030+0.9929879i -0.9638957+0.3498137i -0.7257859-0.6990640i
## [4]  1.0929169-0.4058682i  1.0929169+0.4058682i -0.7257859+0.6990640i
## [7]  0.2489030-0.9929879i -0.9638957-0.3498137i -1.9527387+0.0000000i
```

```
all(Mod(roots_ma) > 1) # Invertibility condition
```

```
## [1] TRUE
```

All the coefficients of the **ARMA(5,2)** are significant.

```
arma52 <- arma_models[[4]]
```

Table 20: Coefficients, SEs, and 95% CIs of the estimated ARMA(5,2) model

	Coefficient	SE	95% CI lower	95% CI upper
ar1	-0.6054	0.0484	-0.7023	-0.5086
ar2	0.4349	0.0573	0.3204	0.5494
ar3	0.6374	0.0520	0.5334	0.7414
ar4	0.1633	0.0573	0.0487	0.2779
ar5	0.3669	0.0486	0.2698	0.4641
ma1	1.6729	0.0151	1.6426	1.7031
ma2	1.0000	0.0162	0.9675	1.0325
intercept	85.9216	49.3736	-12.8256	184.6689

```
(roots_ar <- polyroot(c(1, -arma52$coef[1:arma_models_coefs[4, 1]])))
```

```
## [1]  1.0006271+0.0000000i -0.8705524+0.5073589i -0.8705524-0.5073589i
## [4]  0.1476883-1.6312214i  0.1476883+1.6312214i
```

```
all(Mod(roots_ar) > 1) # Stationarity condition
```

```
## [1] TRUE
```

```
(roots_ma <- polyroot(c(1, arma52$coef[(arma_models_coefs[4, 1] + 1):
                                       (arma_models_coefs[4, 1] +
                                          arma_models_coefs[4, 2])])))
```

```
## [1] -0.8364658+0.5480416i -0.8364658-0.5480416i
```

```
all(Mod(roots_ma) > 1) # Invertibility condition
```

```
## [1] TRUE
```

In the **ARMA(1,10)** model, many of the MA coefficients are not significant; the AR(1) coefficient is close to one.

```
arma110 <- arma_models[[5]]
```

Table 21: Coefficients, SEs, and 95% CIs of the estimated ARMA(1,10) model

	Coefficient	SE	95% CI lower	95% CI upper
ar1	0.9980	0.0016	0.9948	1.0011
ma1	-0.0524	0.0532	-0.1587	0.0540
ma2	-0.0036	0.0715	-0.1466	0.1394
ma3	-0.0036	0.0764	-0.1564	0.1491
ma4	0.2385	0.0606	0.1173	0.3597
ma5	-0.4682	0.0476	-0.5634	-0.3730
ma6	0.0848	0.0414	0.0020	0.1677
ma7	0.4776	0.0653	0.3471	0.6081
ma8	-0.2125	0.0764	-0.3652	-0.0598
ma9	-0.1845	0.0629	-0.3104	-0.0587
ma10	-0.0797	0.0777	-0.2351	0.0758
intercept	105.0086	28.0139	48.9808	161.0364

```
(roots_ar <- polyroot(c(1, -arma110$coef[1:arma_models_coefs[4, 1]])))

## [1]  0.7492441+1.118268i -1.1941092+1.063525i  0.9050346+0.000000i
## [4]  0.7492441-1.118268i -1.1941092-1.063525i

all(Mod(roots_ar) > 1) # Stationarity condition

## [1] FALSE

(roots_ma <- polyroot(c(1, arma110$coef[(arma_models_coefs[4, 1] + 1):
                                        (arma_models_coefs[4, 1] +
                                           arma_models_coefs[4, 2])])))

## [1] 2.759198+2.042877i 2.759198-2.042877i

all(Mod(roots_ma) > 1) # Invertibility condition

## [1] TRUE
```

Finally, R is not able to find the SE for the AR(1) coefficient (and the mean) in the **ARMA(1,5)** model.

```
arma15 <- arma_models[[6]]
```

Table 22: Coefficients, SEs, and 95% CIs of the estimated ARMA(1,5) model

	Coefficient	SE	95% CI lower	95% CI upper
ar1	0.9973	NaN	NaN	NaN
ma1	-0.1137	0.0567	-0.2272	-0.0003
ma2	-0.1482	0.0526	-0.2533	-0.0431
ma3	-0.1154	0.0567	-0.2288	-0.0020
ma4	0.0839	0.0736	-0.0632	0.2310
ma5	0.0154	0.0584	-0.1014	0.1321
intercept	134.9602	NaN	NaN	NaN

```
(roots_ar <- polyroot(c(1, -arma15$coef[1:arma_models_coefs[4, 1]])))

## [1]  1.127321+0.738505i -1.372086+1.278710i -1.372086-1.278710i
## [4]  1.127321-0.738505i  1.865452+0.000000i

all(Mod(roots_ar) > 1) # Stationarity condition

## [1] TRUE

(roots_ma <- polyroot(c(1, arma110$coef[(arma_models_coefs[4, 1] + 1):
                                         (arma_models_coefs[4, 1] +
                                            arma_models_coefs[4, 2])])))

## [1] 2.759198+2.042877i 2.759198-2.042877i

all(Mod(roots_ma) > 1) # Invertibility condition

## [1] TRUE
```

2.4.2 Diagnostics using Residuals

The residuals of the **ARMA(8,3)** model still do not look like white noise: though the mean seems constant and close to zero, its variance grows over time, and some auto-correlations and partial auto-correlations are significantly different from zero, especially at lag=12.

```
summary(arma83$resid)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -8.5680 -0.8973  0.4072  0.4213  1.4660  8.3250
```

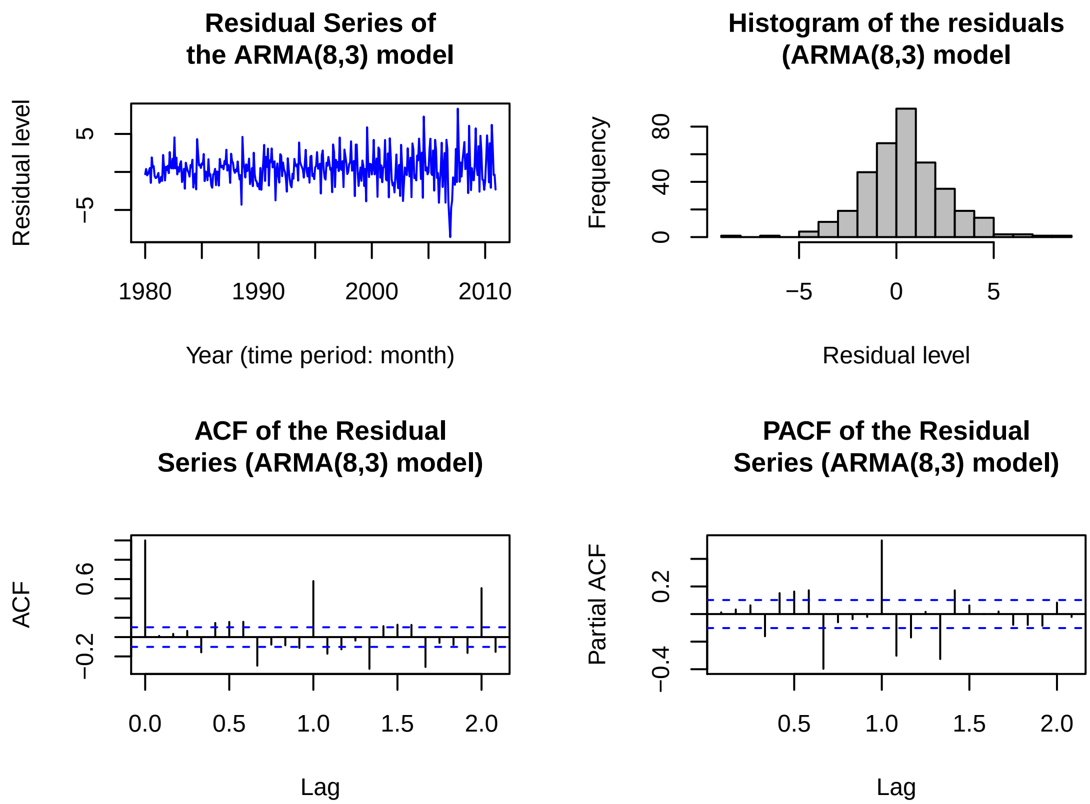

Figure 23: ARMA(8,3) model diagnostic based on the residuals

The result of a Llung-Box test is that we can reject the hypothesis of independence of the residual series:

```
Box.test(arma83$resid, type = "Ljung-Box")
```

```
##
##  Box-Ljung test
##
## data:  arma83$resid
## X-squared = 0.0577, df = 1, p-value = 0.8102
```

Something similar happens with the **ARMA(7,4)**, **ARMA(3,9)**, and **ARMA(5,2)** models. In these 3 cases, the ACFs and PACFs are even bigger for some lags.

```
summary(arma74$resid)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -9.2580 -0.9233  0.1971  0.2773  1.3510  7.7930
```

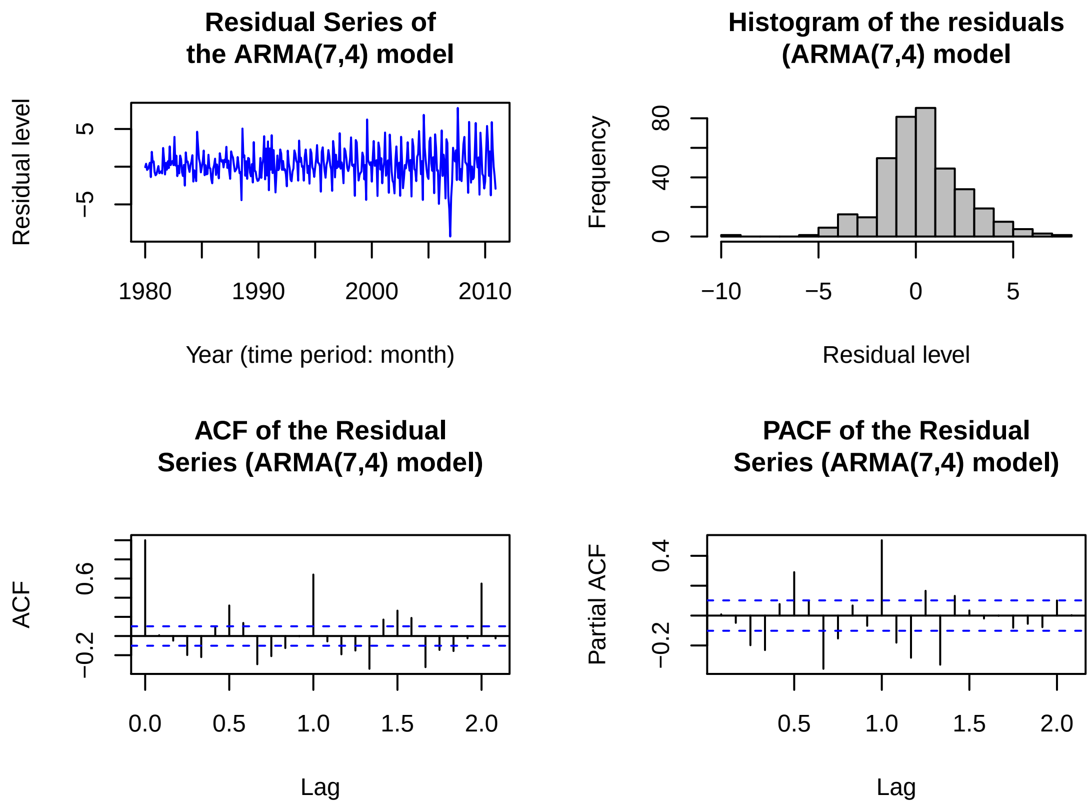

Figure 24: ARMA(7,4) model diagnostic based on the residuals

```
Box.test(arma74$resid, type = "Ljung-Box")
```

```
##
##  Box-Ljung test
##
## data:  arma74$resid
## X-squared = 0.024997, df = 1, p-value = 0.8744
```

```r
summary(arma39$resid)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -10.2300  -1.1020   0.2481   0.2606   1.5540   6.7180
```

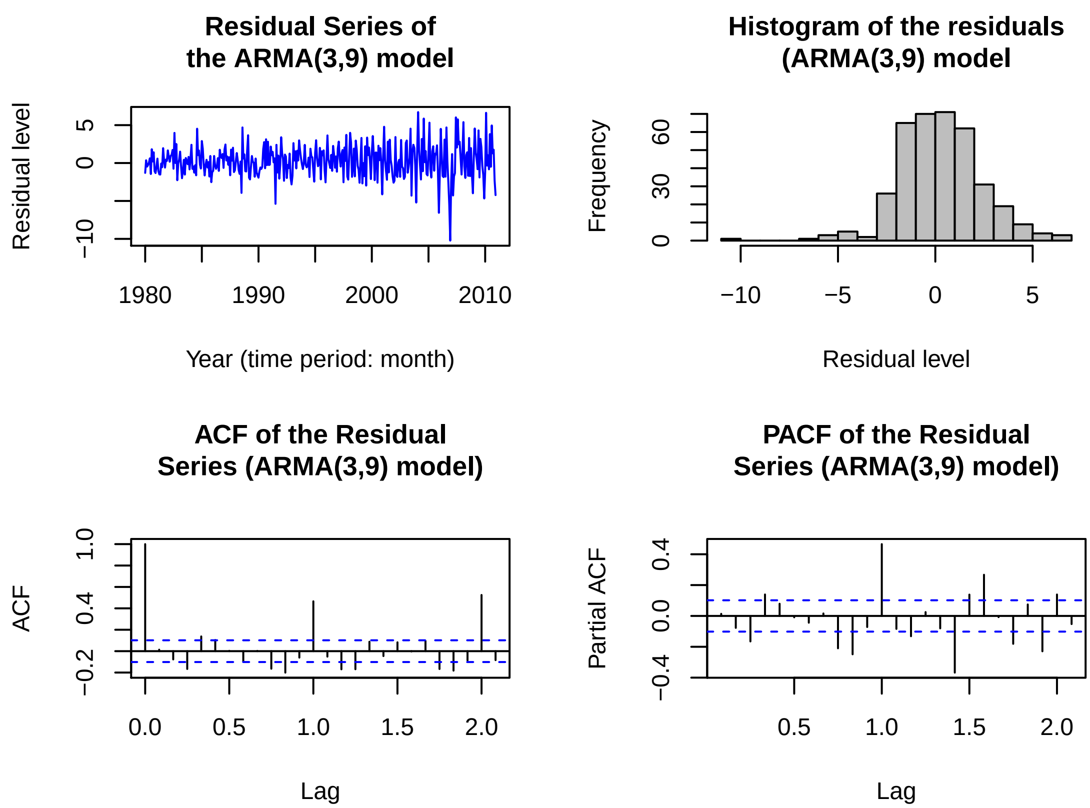

Figure 25: ARMA(3,9) model diagnostic based on the residuals

```r
Box.test(arma39$resid, type = "Ljung-Box")
```

```
##
##  Box-Ljung test
##
## data:  arma39$resid
## X-squared = 0.071046, df = 1, p-value = 0.7898
```

```r
summary(arma52$resid)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -9.0980 -0.8343  0.3685  0.3603  1.7200  7.1100
```

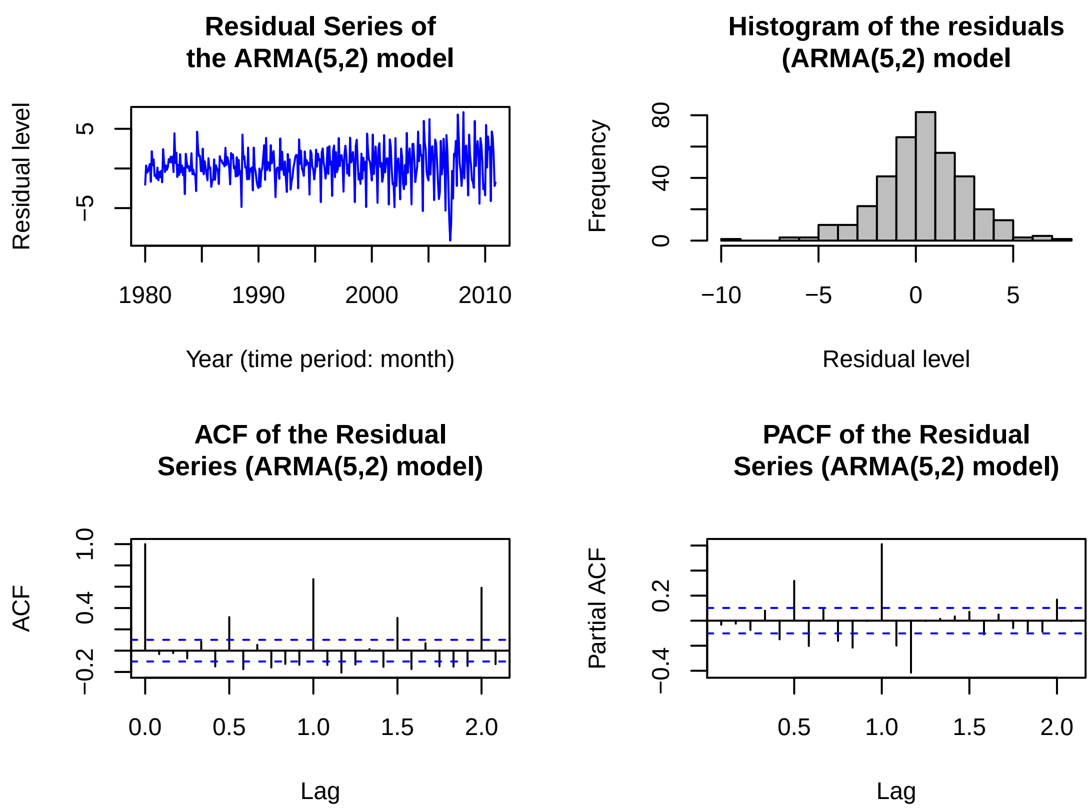

Figure 26: ARMA(5,2) model diagnostic based on the residuals

```r
Box.test(arma52$resid, type = "Ljung-Box")
```

```
##
##  Box-Ljung test
##
## data:  arma52$resid
## X-squared = 0.39674, df = 1, p-value = 0.5288
```

The ACFs of the **ARMA(1,10)** model are, as expected, lower than those of the previous models. But the residuals are still volatile (their variance increases over time), and due to the ACF at lags 12 and 24, it does not look exactly like white noise, either.

```
summary(arma110$resid)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -10.9000  -1.0350   0.3203   0.2799   1.5300   7.9250
```

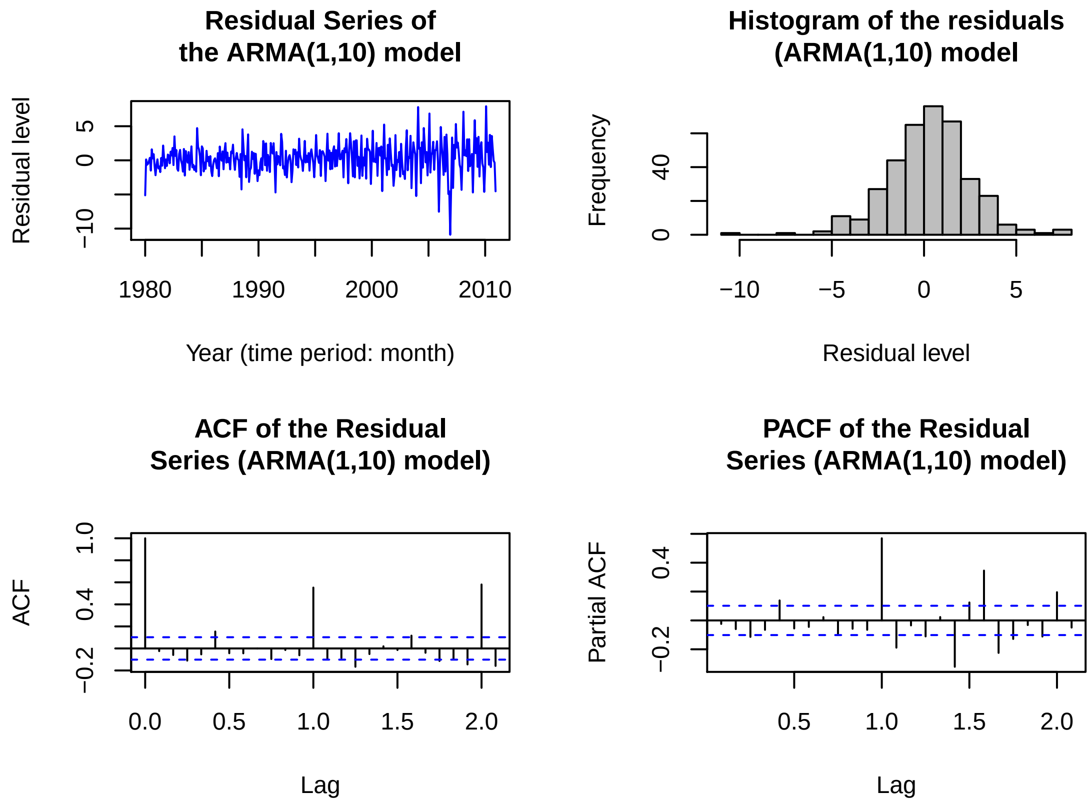

Figure 27: ARMA(1,10) model diagnostic based on the residuals

```
Box.test(arma110$resid, type = "Ljung-Box")
```

```
##
##  Box-Ljung test
##
## data:  arma110$resid
## X-squared = 0.22205, df = 1, p-value = 0.6375
```

Something similar happens with the **ARMA(1,5)** model (in this case, it's the PACFs which are lower than ever).

```r
summary(arma15$resid)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -11.5300  -0.8358   0.3688   0.1846   1.6550   7.3170
```

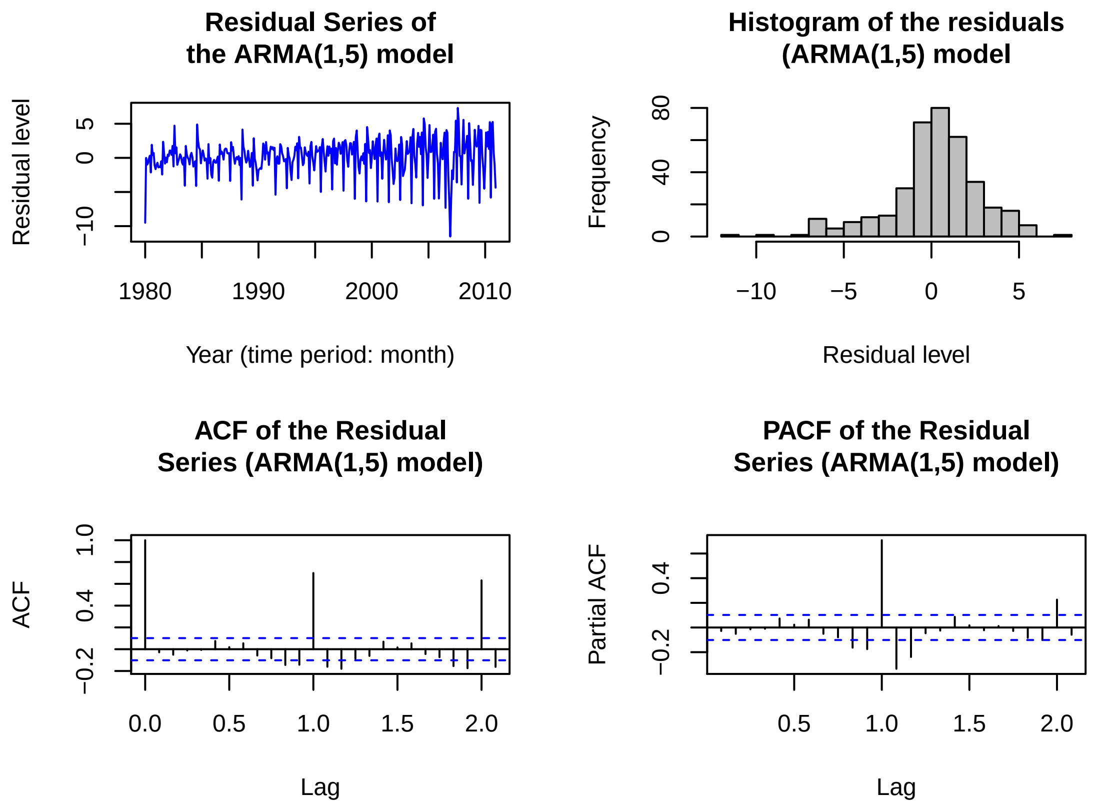

Figure 28: ARMA(1,5) model diagnostic based on the residuals

```r
Box.test(arma15$resid, type = "Ljung-Box")
```

```
##
##  Box-Ljung test
##
## data:  arma15$resid
## X-squared = 0.31171, df = 1, p-value = 0.5766
```

2.4.3 Model Performance Evaluation

2.4.3.1 In-sample fit

As it happened with the AR and MA models, the in-sample fit of the **ARMA(8,3)** (the best model in terms of AIC and BIC) looks reasonable, but again the estimated series is lagged 1 period (compare each value in the first column of the following table with the value in the second column and the next row).

Time	Original series	Estimated series	Residuals
Jul 2010	141.9	144.0	-2.1
Aug 2010	146.9	140.7	6.2
Sep 2010	152.0	148.6	3.4
Oct 2010	152.6	153.0	-0.4
Nov 2010	149.7	150.1	-0.4
Dec 2010	145.0	147.3	-2.3

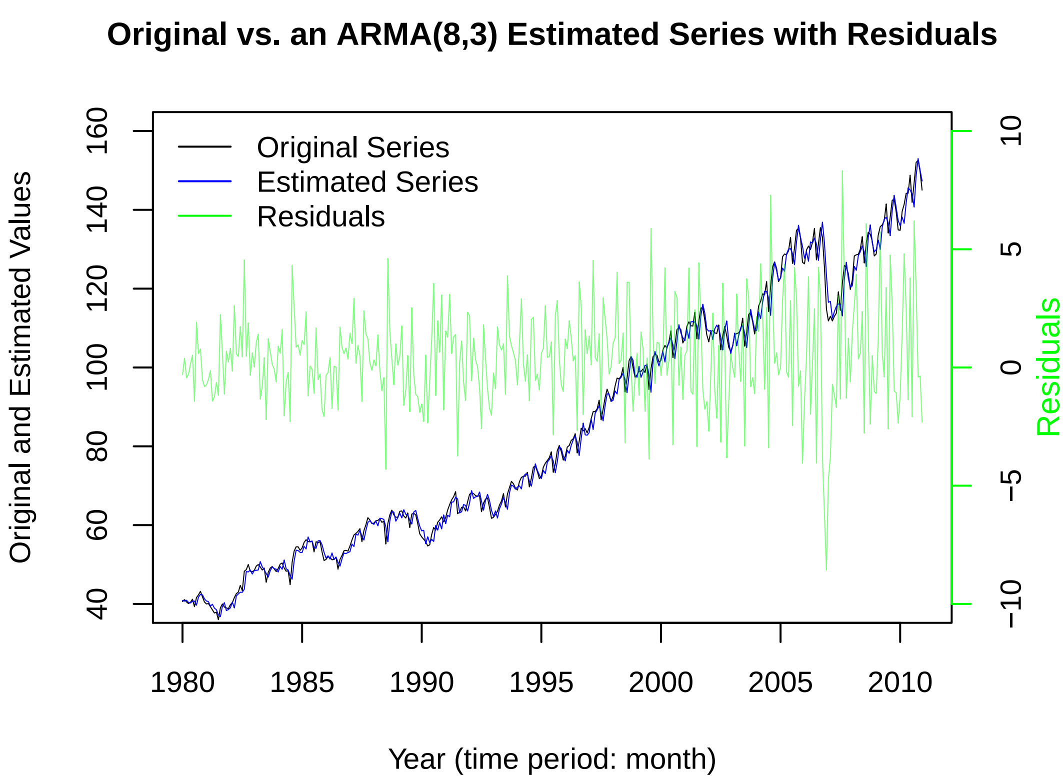

Figure 29: ARMA(8,3) model performance evaluation (in-sample)

Something similar occurs with the other 5 ARMA models under study.

ARMA(7,4):

Time	Original series	Estimated series	Residuals
Jul 2010	141.9	145.7	-3.8
Aug 2010	146.9	141.0	5.9
Sep 2010	152.0	149.6	2.4
Oct 2010	152.6	152.6	-0.0
Nov 2010	149.7	150.9	-1.2
Dec 2010	145.0	147.9	-2.9

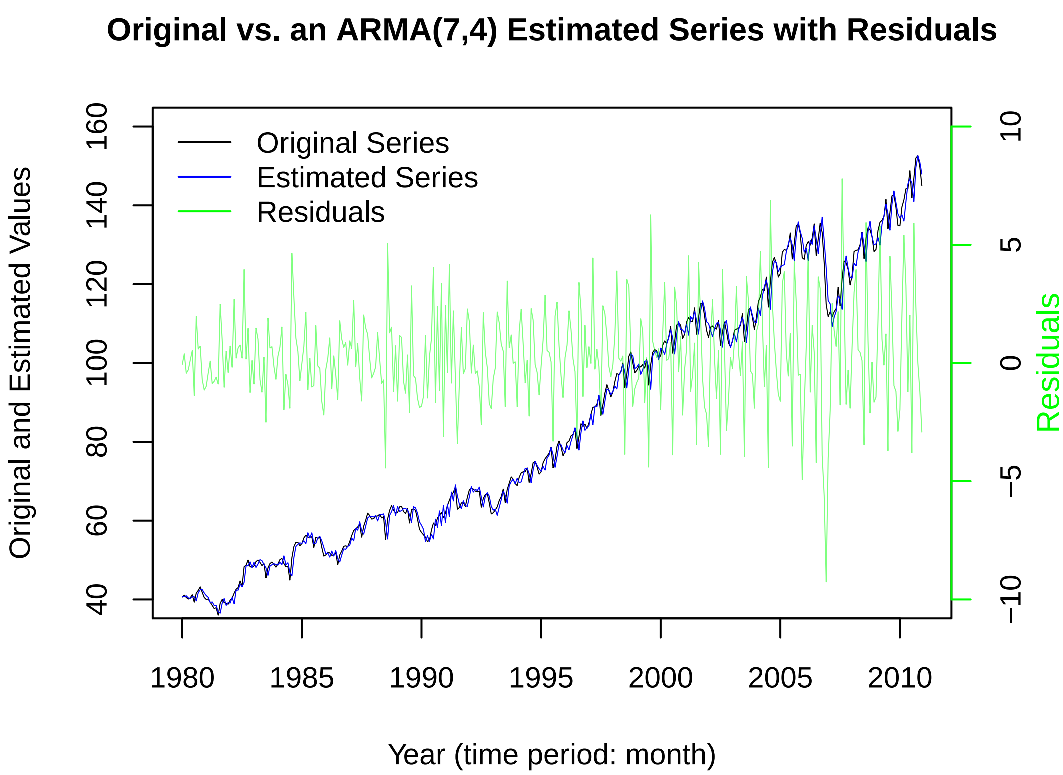

Figure 30: ARMA(7,4) model performance evaluation (in-sample)

ARMA(3,9):

Time	Original series	Estimated series	Residuals
Jul 2010	141.9	142.4	-0.5
Aug 2010	146.9	141.9	5.0
Sep 2010	152.0	150.7	1.3
Oct 2010	152.6	150.8	1.8
Nov 2010	149.7	152.0	-2.3
Dec 2010	145.0	149.2	-4.2

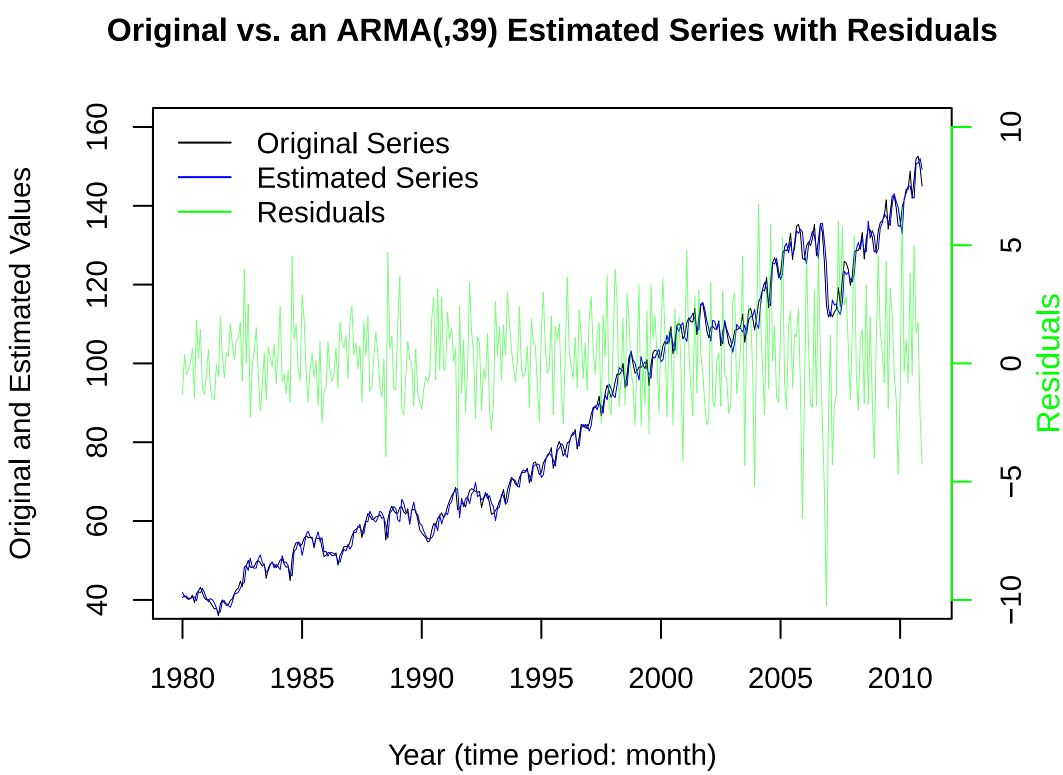

Figure 31: ARMA(3,9) model performance evaluation (in-sample)

ARMA(5,2):

Time	Original series	Estimated series	Residuals
Jul 2010	141.9	146.0	-4.1
Aug 2010	146.9	142.2	4.7
Sep 2010	152.0	148.1	3.9
Oct 2010	152.6	151.0	1.6
Nov 2010	149.7	151.9	-2.2
Dec 2010	145.0	146.8	-1.8

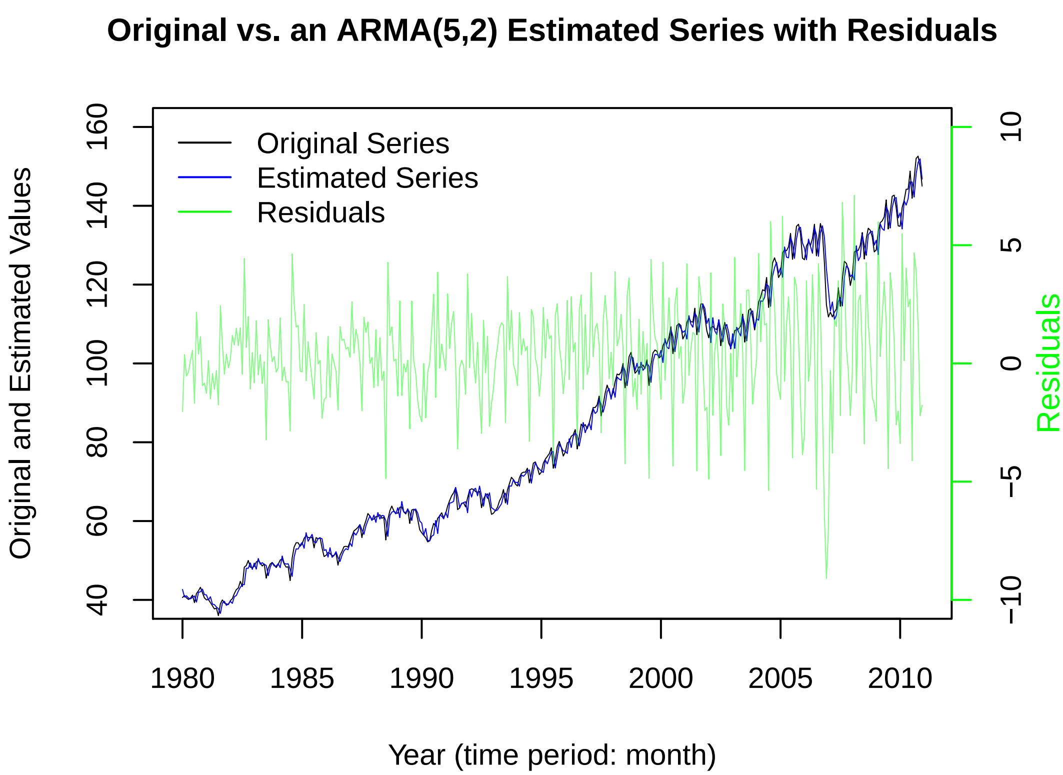

Figure 32: ARMA(5,2) model performance evaluation (in-sample)

ARMA(1,10):

Time	Original series	Estimated series	Residuals
Jul 2010	141.9	142.9	-1.0
Aug 2010	146.9	143.3	3.6
Sep 2010	152.0	150.3	1.7
Oct 2010	152.6	152.7	-0.1
Nov 2010	149.7	150.0	-0.3
Dec 2010	145.0	149.5	-4.5

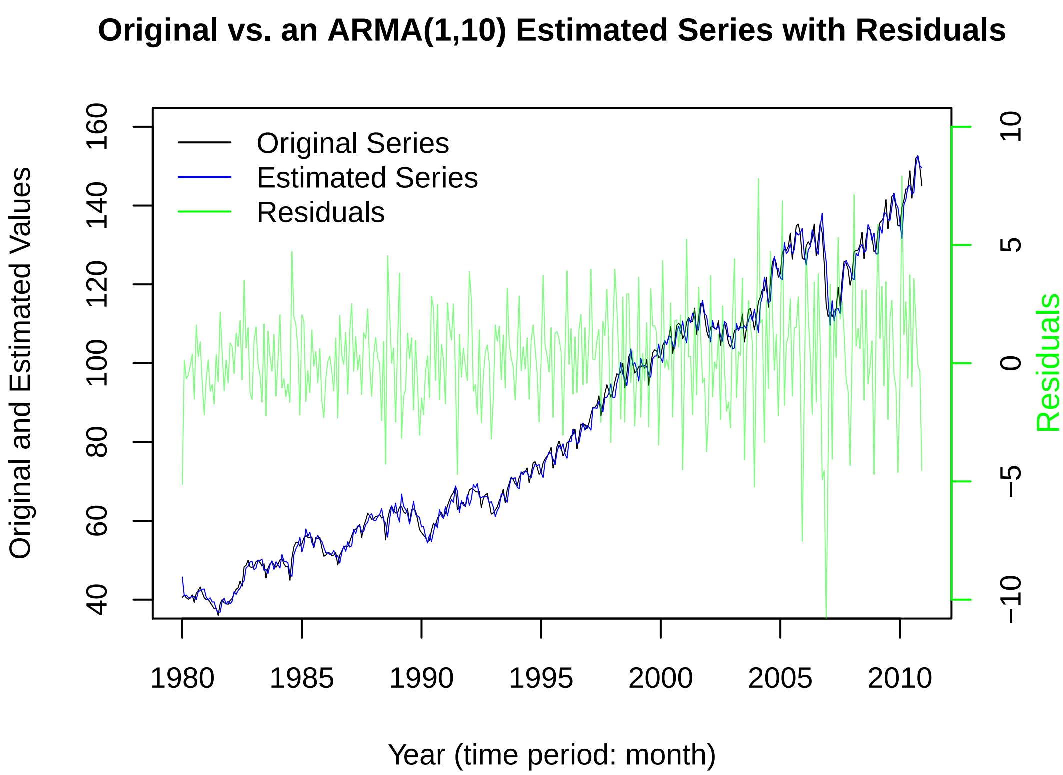

Figure 33: ARMA(1,10) model performance evaluation (in-sample)

ARMA(1,5):

Time	Original series	Estimated series	Residuals
Jul 2010	141.9	147.7	-5.8
Aug 2010	146.9	142.0	4.9
Sep 2010	152.0	146.7	5.3
Oct 2010	152.6	151.8	0.8
Nov 2010	149.7	150.7	-1.0
Dec 2010	145.0	149.4	-4.4

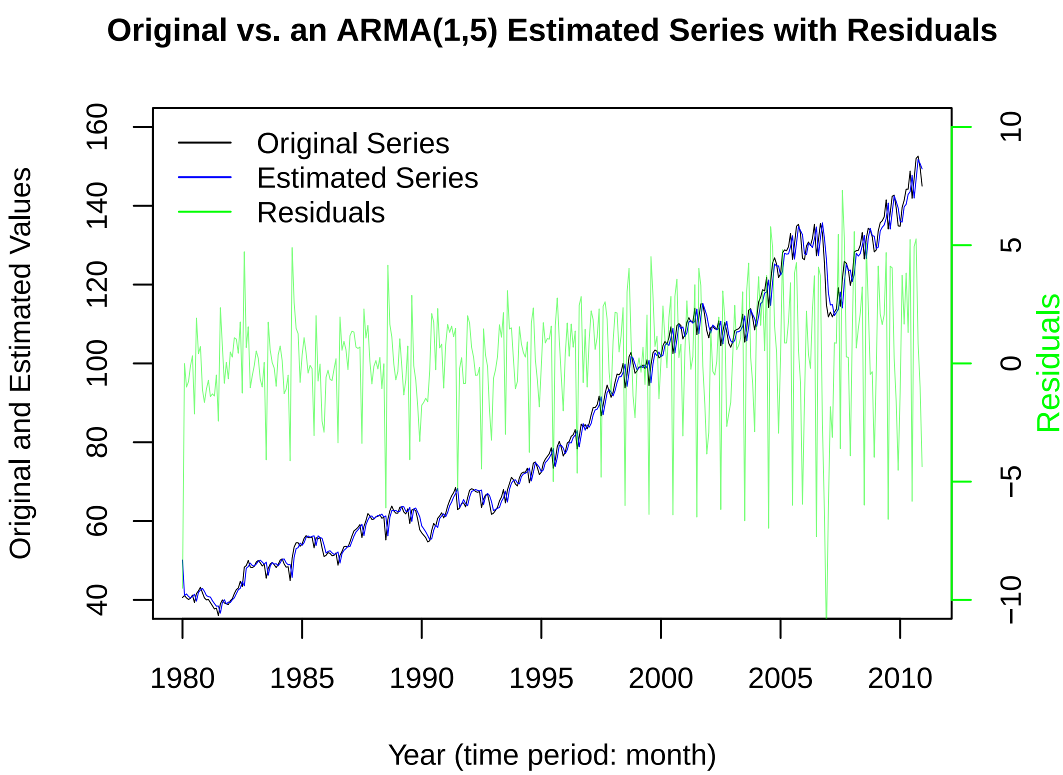

Figure 34: ARMA(1,5) model performance evaluation (in-sample)

2.4.3.2 Out-of-sample fit

Surprisingly, the out-of-sample fit of the **ARMA(8,3)** is not good: although the forecasts capture part of the seasonality (see how their mean goes up and down in the Figure in the next page), almost all the original values are outside the 95% confidence intervals.

```
(arma83.oos.fit <- Arima(hw08.ts_train, order = c(arma_models_coefs[1, 1], 0,
                                                  arma_models_coefs[1, 2])))
```

```
## Series: hw08.ts_train
## ARIMA(8,0,3) with non-zero mean
##
## Coefficients:
##           ar1      ar2      ar3     ar4     ar5     ar6     ar7     ar8
##       -0.0849  -0.0515  -0.0253  0.6891  0.0842  0.0545  0.0246  0.3057
## s.e.   0.0526   0.0523   0.0531  0.0523  0.0529  0.0527  0.0533  0.0524
##          ma1     ma2     ma3  intercept
##       1.0594  1.0667  0.9921    79.7567
## s.e.  0.0103  0.0177  0.0119    39.2770
##
## sigma^2 estimated as 4.176:  log likelihood=-726.06
## AIC=1478.12   AICc=1479.25   BIC=1527.74
```

Time	Original series	Estimated series	Residuals
Jan 1980	40.6	42.5	-1.9
Feb 1980	41.1	40.8	0.3
Mar 1980	40.5	41.0	-0.5
Apr 1980	40.1	40.4	-0.3
May 1980	40.4	40.3	0.1
Jun 1980	41.2	40.7	0.5

```
arma83.oos.fit.fcast <- forecast.Arima(arma83.oos.fit, h = 36)
(acc_arma83 <- accuracy(arma83.oos.fit.fcast, hw08.ts_test))
```

```
##                     ME      RMSE       MAE        MPE      MAPE      MASE
## Training set  0.338869  2.043555  1.519093  0.3967078  1.967195 0.2998941
## Test set     15.779566 17.524777 15.779566 11.2294745 11.229475 3.1151470
##                    ACF1 Theil's U
## Training set 0.01857939        NA
## Test set     0.84709311  4.547937
```

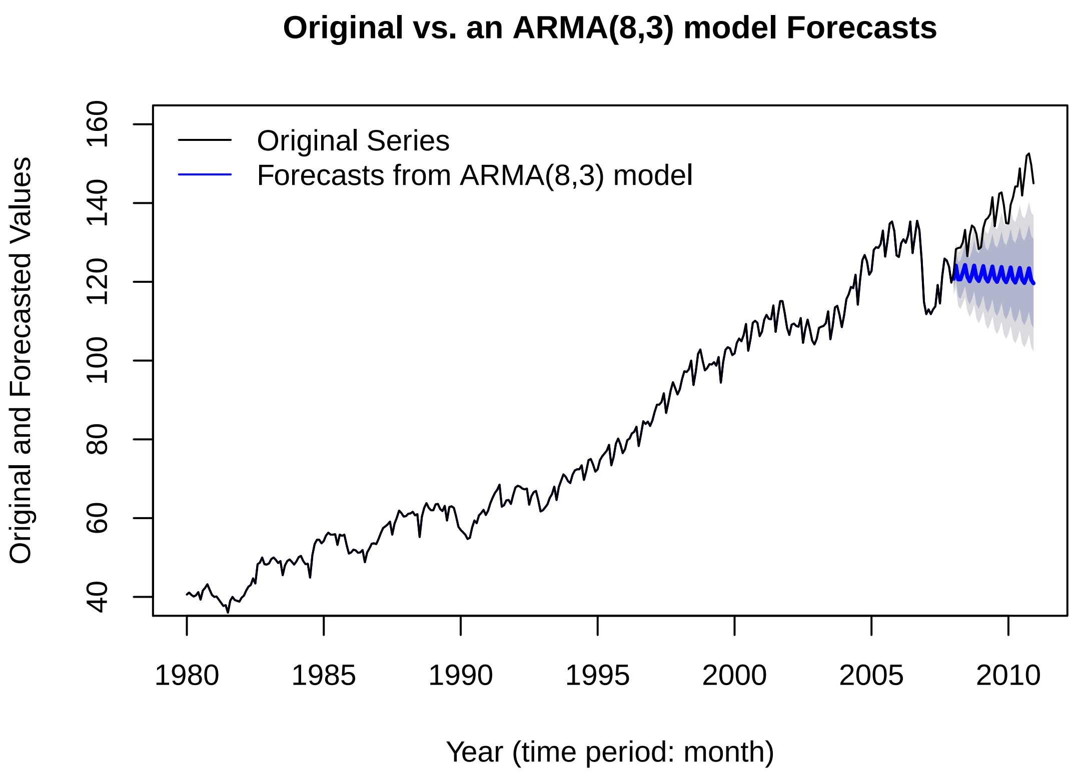

Figure 35: ARMA(8,3) model performance evaluation (out of-sample)

It is not possible to estimate an **ARMA(7,4)** model using around the first 90% of the original observations: the resulting model is not stationary. We had to decrease the training period to approximately 83.9% (i.e, 26 years instead of 28, increasing the test period from 3 to 5 years) to be able to estimate the new ARMA(7,4) model. Once we did that, the out-of-sample fit was actually pretty good, with only some original observations (those corresponding to the large drop at the end of 2006 and beginning of 2007—which were included in the training set for the other models—and just a few of the last ones) out of the 95% confidence intervals.

Since we modified the training and test sets for this particular ARMA model, the comparison between the values of the RMSE, MAE, MAPE, etc., for both sets is different than for the other models.

```
hw08.ts_train <- window(hw08.ts, start = 1980, end=c(2005,12))
hw08.ts_test <- window(hw08.ts, start = 2006)
(arma74.oos.fit <- Arima(hw08.ts_train, order = c(arma_models_coefs[2, 1], 0,
                                                  arma_models_coefs[2, 2])))
```

```
## Series: hw08.ts_train
## ARIMA(7,0,4) with non-zero mean
##
## Coefficients:
##           ar1     ar2     ar3     ar4     ar5     ar6      ar7     ma1
##       -0.2136  0.0178  0.3235  0.7671  0.0472  0.2146  -0.1566  1.1097
## s.e.      NaN     NaN     NaN     NaN     NaN     NaN      NaN     NaN
##           ma2     ma3      ma4  intercept
##        0.8625  0.3872  -0.3525    78.5060
```

```
## s.e.     NaN     NaN      NaN   374.7419
##
## sigma^2 estimated as 4.052:  log likelihood=-663.51
## AIC=1353.03   AICc=1354.25   BIC=1401.69
```

Time	Original series	Estimated series	Residuals
Jan 1980	40.6	40.7	-0.1
Feb 1980	41.1	40.7	0.4
Mar 1980	40.5	41.0	-0.5
Apr 1980	40.1	40.4	-0.3
May 1980	40.4	40.3	0.1
Jun 1980	41.2	40.6	0.6

```
arma74.oos.fit.fcast <- forecast.Arima(arma74.oos.fit, h = 60)
(acc_arma74 <- accuracy(arma74.oos.fit.fcast, hw08.ts_test))
```

```
##                     ME      RMSE      MAE       MPE     MAPE      MASE
## Training set 0.4056329  2.012966 1.552007 0.4809147 2.110533 0.3197377
## Test set     2.4092043 10.438638 8.301413 1.2295995 6.285455 1.7102210
##                     ACF1 Theil's U
## Training set -0.03666781        NA
## Test set      0.91387597  2.583199
```

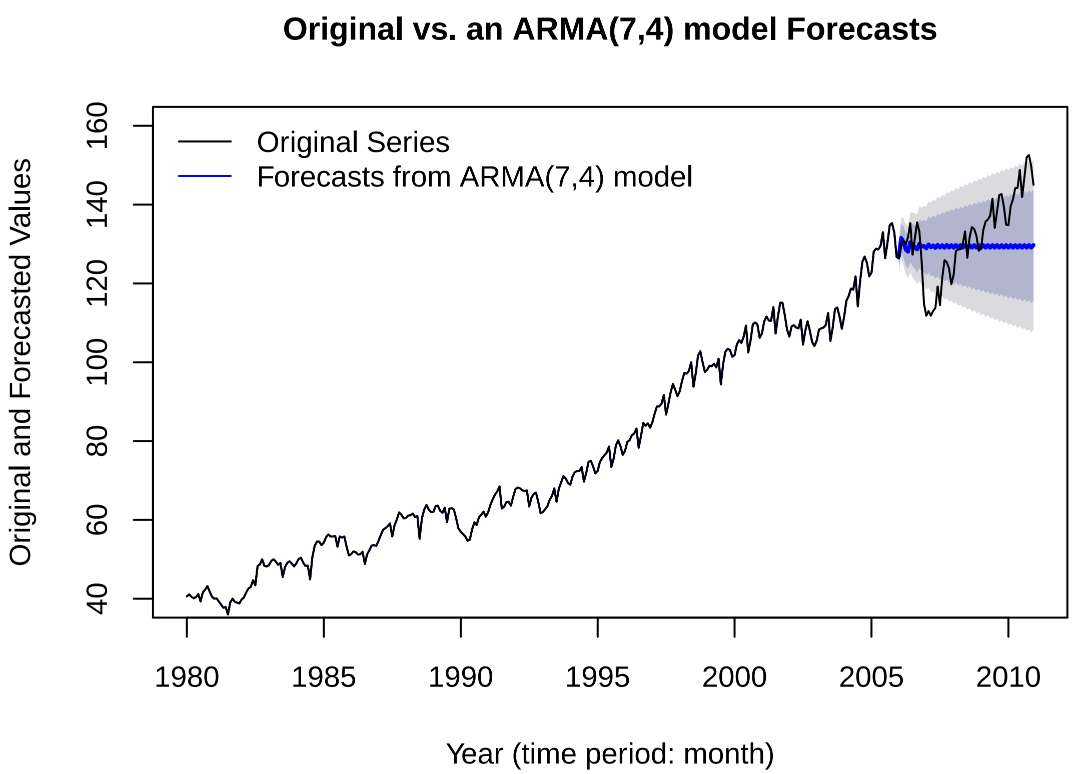

Figure 36: ARMA(7,4) model performance evaluation (out of-sample)

To estimate the out-of-sample fit of the **ARMA(3,9)** model, we use again the first 28 and last 3 years as the training & test sets. As shown in the Figure in the next page, the fit is probably the best so far.

```
hw08.ts_train <- window(hw08.ts, start = 1980, end=c(2007,12))
hw08.ts_test <- window(hw08.ts, start = 2008)
(arma39.oos.fit <- Arima(hw08.ts_train, order = c(arma_models_coefs[3, 1], 0,
                                                  arma_models_coefs[3, 2])))
```

```
## Series: hw08.ts_train
## ARIMA(3,0,9) with non-zero mean
##
## Coefficients:
##           ar1     ar2     ar3     ma1     ma2      ma3      ma4      ma5
##       -0.7138  0.7352  0.9782  1.8402  1.1000  -0.0687  -0.0853  -0.4329
## s.e.   0.0111  0.0063  0.0122  0.0505  0.1047   0.1212   0.1188   0.1319
##           ma6     ma7     ma8     ma9  intercept
##       -0.7811  0.0377  0.8837  0.4479    79.5527
## s.e.   0.1377  0.1269  0.1095  0.0483   138.1363
##
## sigma^2 estimated as 3.952:  log likelihood=-719.07
## AIC=1466.13   AICc=1467.44   BIC=1519.57
```

Time	Original series	Estimated series	Residuals
Jan 1980	40.6	41.1	-0.5
Feb 1980	41.1	40.7	0.4
Mar 1980	40.5	40.9	-0.4
Apr 1980	40.1	40.4	-0.3
May 1980	40.4	40.3	0.1
Jun 1980	41.2	40.6	0.6

```
arma39.oos.fit.fcast <- forecast.Arima(arma39.oos.fit, h = 36)
(acc_arma39 <- accuracy(arma39.oos.fit.fcast, hw08.ts_test))
```

```
##                     ME      RMSE       MAE       MPE     MAPE      MASE
## Training set  0.231925  1.987997  1.480992 0.2818627 1.902093 0.2923723
## Test set     13.215486 15.022548 13.215486 9.3743786 9.374379 2.6089554
##                   ACF1 Theil's U
## Training set 0.04500735        NA
## Test set     0.82933500  3.883417
```

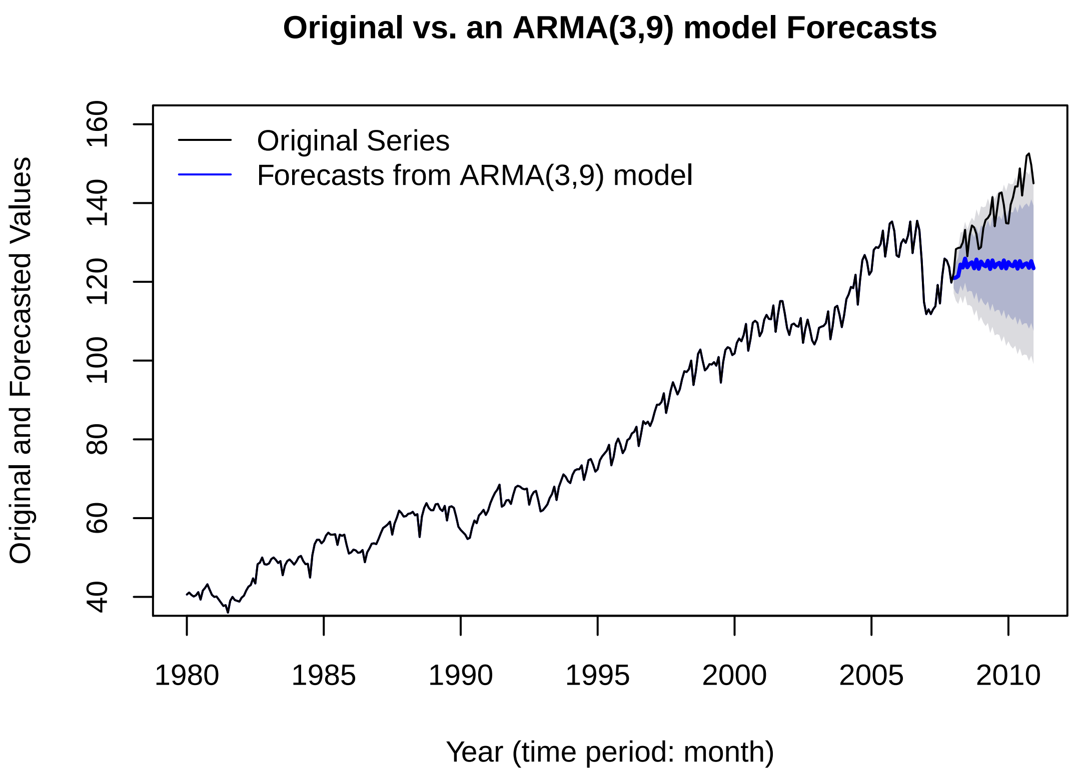

Figure 37: ARMA(3,9) model performance evaluation (out of-sample)

The out-of-sample fit of the **ARMA(5,2)** model is quite similar to that of the AR(3) model: not very good on the long-term, though the first original observations lie within the 95% confidence intervals.

```
(arma52.oos.fit <- Arima(hw08.ts_train, order = c(arma_models_coefs[4, 1], 0,
                                                  arma_models_coefs[4, 2])))
```

```
## Series: hw08.ts_train
## ARIMA(5,0,2) with non-zero mean
##
## Coefficients:
##          ar1     ar2      ar3     ar4     ar5     ma1      ma2   intercept
##       0.1962  0.7460  -0.2114  0.2538  0.0154  0.7658  -0.2292     79.9724
## s.e.  0.3548  0.0543   0.2621  0.0543  0.1112  0.3520   0.3488  80590.5467
##
## sigma^2 estimated as 5.589:  log likelihood=-767.68
## AIC=1553.35   AICc=1553.9   BIC=1587.71
```

Time	Original series	Estimated series	Residuals
Jan 1980	40.6	40.6	-0.0
Feb 1980	41.1	40.6	0.5
Mar 1980	40.5	41.0	-0.5
Apr 1980	40.1	40.4	-0.3

Time	Original series	Estimated series	Residuals
May 1980	40.4	40.2	0.2
Jun 1980	41.2	40.6	0.6

```
arma52.oos.fit.fcast <- forecast.Arima(arma52.oos.fit, h = 36)
(acc_arma52 <- accuracy(arma52.oos.fit.fcast, hw08.ts_test))
```

```
##                      ME     RMSE       MAE        MPE      MAPE     MASE
## Training set  0.3341459  2.36419  1.769173  0.3997874  2.256635 0.349264
## Test set     16.3316069 17.93680 16.331607 11.6358610 11.635861 3.224129
##                     ACF1 Theil's U
## Training set -0.02007131        NA
## Test set      0.82673778  4.662554
```

The out-of-sample fit of the **ARMA(1,10)** model (whose residuals resemble white noise, in terms of their auto-correlations, more than any other model) is worse: almost all out-of-sample observationsis lie out of the 95% confidence intervals.

```
(arma110.oos.fit <- Arima(hw08.ts_train, order = c(arma_models_coefs[5, 1], 0,
                                                   arma_models_coefs[5, 2])))
```

```
## Series: hw08.ts_train
## ARIMA(1,0,10) with non-zero mean
##
## Coefficients:
##          ar1      ma1      ma2     ma3     ma4      ma5     ma6     ma7
##       0.9979  -0.0649  -0.0132  0.0438  0.2519  -0.4505  0.0892  0.4507
## s.e.  0.0024   0.0622   0.0933  0.1094  0.0504   0.0498  0.0468  0.0682
##          ma8      ma9     ma10  intercept
##       -0.2704  -0.2014  -0.0748   103.1394
## s.e.   0.0914   0.0777   0.1263    38.2036
##
## sigma^2 estimated as 4.735:  log likelihood=-744.33
## AIC=1514.67   AICc=1515.8   BIC=1564.29
```

Time	Original series	Estimated series	Residuals
Jan 1980	40.6	45.9	-5.3
Feb 1980	41.1	41.0	0.1
Mar 1980	40.5	41.2	-0.7
Apr 1980	40.1	40.6	-0.5
May 1980	40.4	40.6	-0.2
Jun 1980	41.2	40.8	0.4

```
arma110.oos.fit.fcast <- forecast.Arima(arma110.oos.fit, h = 36)
(acc_arma110 <- accuracy(arma110.oos.fit.fcast, hw08.ts_test))
```

```
##                      ME      RMSE       MAE        MPE      MAPE      MASE
## Training set  0.2199347  2.176049  1.650985  0.1969279  2.156907 0.3259317
## Test set     18.9313081 20.478284 18.931308 13.5191668 13.519167 3.7373531
```

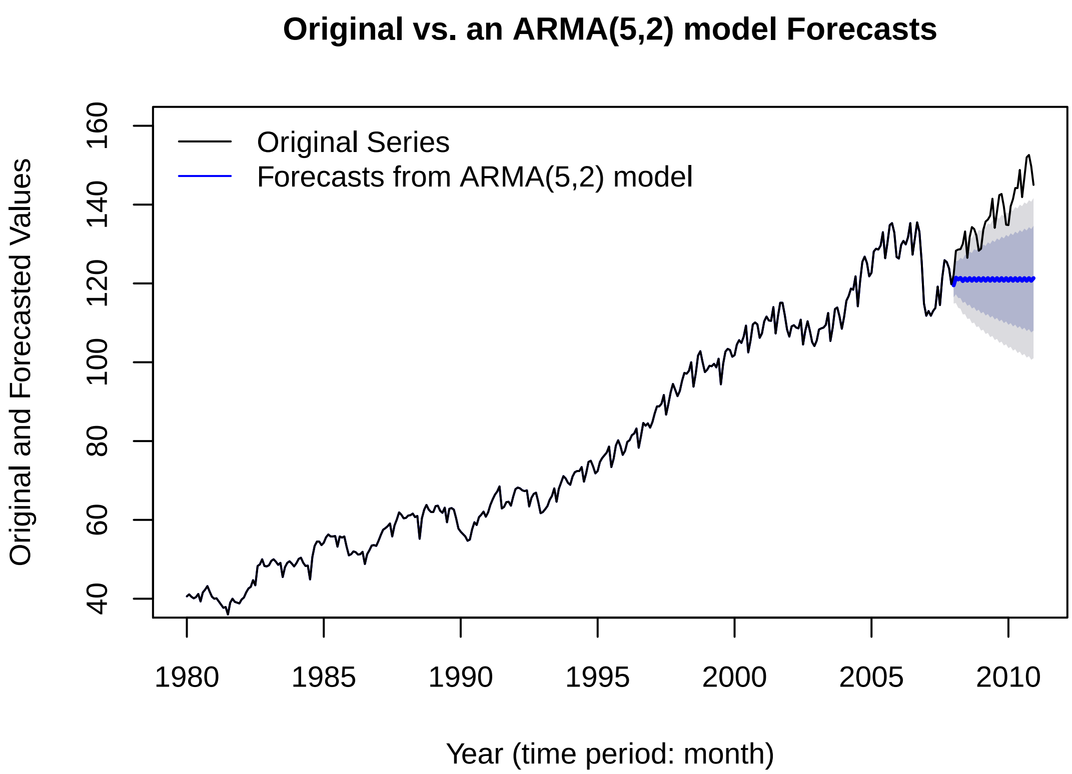

Figure 38: ARMA(5,2) model performance evaluation (out of-sample)

```
##                         ACF1 Theil's U
## Training set -0.008132337        NA
## Test set      0.796726352  5.344763
```

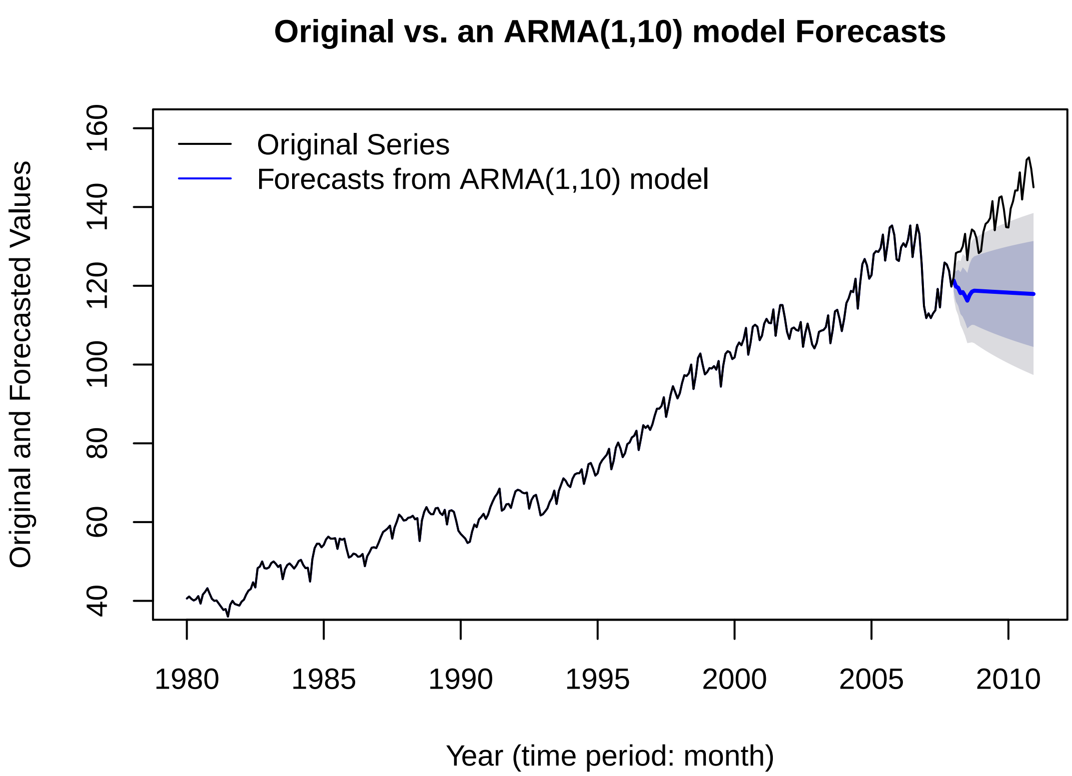

Figure 39: ARMA(1,10) model performance evaluation (out of-sample)

The out-of-sample fit of the **ARMA(1,5)** model (whose residuals resemble white noise, in terms of their partial auto-correlations, more than any other model) is quite good, though not as good as that of the ARMA(3,9) model. It has the advantage, though, of being a simpler model.

```
(arma15.oos.fit <- Arima(hw08.ts_train, order = c(arma_models_coefs[6, 1], 0,
                                                  arma_models_coefs[6, 2])))
```

```
## Series: hw08.ts_train
## ARIMA(1,0,5) with non-zero mean
##
## Coefficients:
##          ar1      ma1      ma2      ma3     ma4     ma5  intercept
##       0.9985  -0.1135  -0.1667  -0.0769  0.0960  0.0935    79.4096
## s.e.  0.0019   0.0543   0.0551   0.0569  0.0645  0.0539    33.1521
##
## sigma^2 estimated as 5.976:  log likelihood=-779.9
## AIC=1575.8   AICc=1576.24   BIC=1606.34
```

Time	Original series	Estimated series	Residuals
Jan 1980	40.6	43.2	-2.6
Feb 1980	41.1	40.7	0.4
Mar 1980	40.5	41.1	-0.6
Apr 1980	40.1	40.5	-0.4
May 1980	40.4	40.3	0.1
Jun 1980	41.2	40.6	0.6

```
arma15.oos.fit.fcast <- forecast.Arima(arma15.oos.fit, h = 36)
(acc_arma15 <- accuracy(arma15.oos.fit.fcast, hw08.ts_test))
```

```
##                      ME      RMSE       MAE        MPE      MAPE      MASE
## Training set  0.2836915  2.444534  1.779499  0.2980379  2.264523 0.3513025
## Test set     16.2280695 18.141629 16.228069 11.5324975 11.532498 3.2036891
##                     ACF1 Theil's U
## Training set -0.01776885        NA
## Test set      0.82972793  4.705094
```

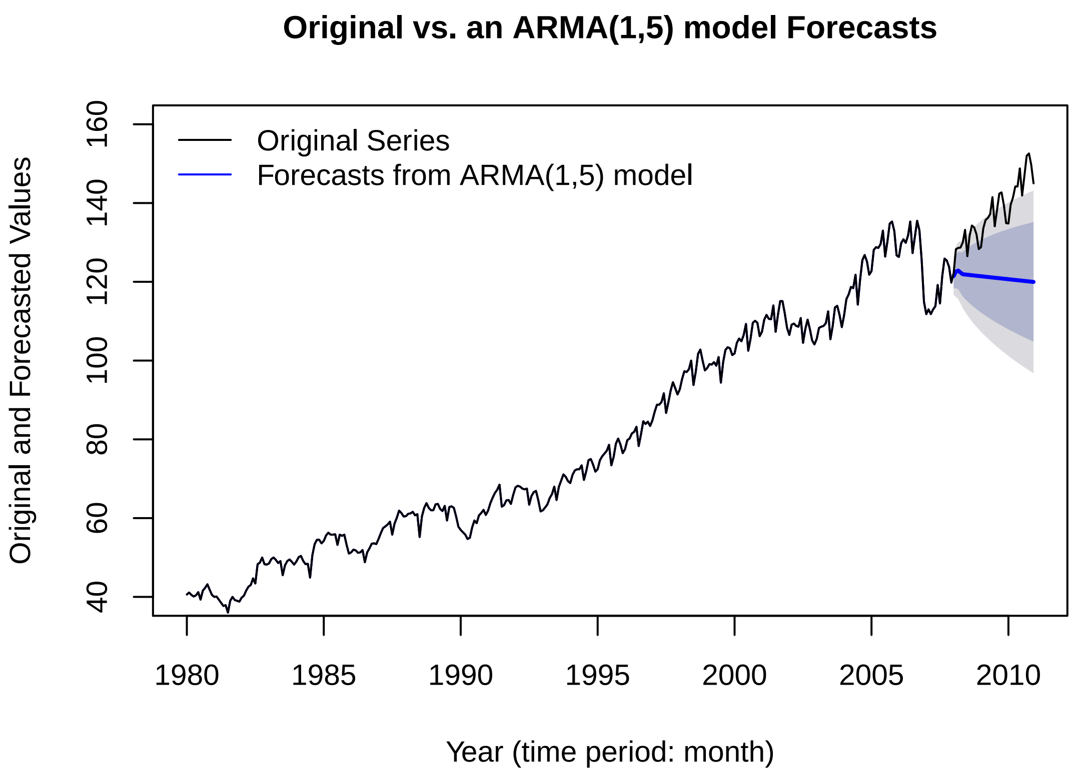

Figure 40: ARMA(1,5) model performance evaluation (out of-sample)

3 Selection of the "best" model

As a reminder, we present again the AIC and BIC values (which do not indicate how well a model will perform in terms of forecasting) of the 8 models we've analyzed:

Table 35: AIC of the models under study (in increasing order)

p	q	aic	bic	ACF	PACF	family
8	3	1657.3	1708.3	3.6	2.6	ARMA
7	4	1662.6	1713.6	4.6	2.8	ARMA
3	9	1667.0	1721.9	2.5	2.7	ARMA
1	10	1703.8	1754.7	1.9	2.6	ARMA
5	2	1708.7	1744.0	3.9	2.6	ARMA
9	0	1776.1	1819.2	2.1	2.2	AR
3	0	1786.8	1806.4	3.1	2.3	AR
1	5	1789.6	1821.0	2.3	1.9	ARMA
0	12	1823.2	1878.1	4.6	3.5	MA
0	10	1838.2	1885.2	4.8	3.6	MA

Table 36: BIC of the models under study (in increasing order)

p	q	aic	bic	ACF	PACF	family
8	3	1657.3	1708.3	3.6	2.6	ARMA
7	4	1662.6	1713.6	4.6	2.8	ARMA
3	9	1667.0	1721.9	2.5	2.7	ARMA
5	2	1708.7	1744.0	3.9	2.6	ARMA
1	10	1703.8	1754.7	1.9	2.6	ARMA
3	0	1786.8	1806.4	3.1	2.3	AR
9	0	1776.1	1819.2	2.1	2.2	AR
1	5	1789.6	1821.0	2.3	1.9	ARMA
0	12	1823.2	1878.1	4.6	3.5	MA
0	10	1838.2	1885.2	4.8	3.6	MA

As shown above, the best models (in terms of BIC or AIC) are (in general) the ARMA ones, and the worst are the MA ones. But we cannot rely exclusively in this criterion, especially for forecasting.

The table above also shows the sum of the absolute value of the significant ACFs and PACFs of each model (for the first 24 lags, excluding lag 0 in the case of the ACF). This is just a proxy for the resemblance of each model's residuals to white noise: it does not substitute the inspection of the time plot series, histogram, and correlogram, but reminds us the poor performance of the MA models, the fact that the AR(9) model is a better fit than the AR(3) one, etc.

The in-sample fit was pretty similar for the 10 models under study.

As for the out-of-sample fit, the following table (that shows the RMSE and MAE values of each model, for the training and test sets) serves as a summary of what the plots showed:

Table 37: RMSE and MAE of the models under study for the training and test sets in the out-of-sample fit

	p	q	family	RMSE	MAE
Training set	9	0	AR	2.4	1.8
Test set	9	0	AR	17.0	14.9
Training set	3	0	AR	2.5	1.8
Test set	3	0	AR	19.2	17.5
Training set	0	12	MA	2.5	1.9
Test set	0	12	MA	55.5	53.1
Training set	0	10	MA	2.5	1.9
Test set	0	10	MA	55.2	52.6
Training set	8	3	ARMA	2.0	1.5
Test set	8	3	ARMA	17.5	15.8
Training set	7	4	ARMA	2.0	1.6
Test set	7	4	ARMA	10.4	8.3
Training set	3	9	ARMA	2.0	1.5
Test set	3	9	ARMA	15.0	13.2
Training set	5	2	ARMA	2.4	1.8
Test set	5	2	ARMA	17.9	16.3
Training set	1	10	ARMA	2.2	1.7
Test set	1	10	ARMA	20.5	18.9
Training set	1	5	ARMA	2.4	1.8
Test set	1	5	ARMA	18.1	16.2

Since the other metrics are not very different from one model to the other, we use the out-of-sample fit as the main criterion for selection. Excluding the ARMA(7,4) (for which we had to use a longer out-of-sample interval), the previos Table shows that the models with the lowest RMSE and MAE values are **ARMA(3,9)** and **AR(9)**. But we mainly base our selection on the time series plots of those out-of-sample fits (Figures 15 and 16, 21 and 22, and 35 to 40), and what we commented about them.

Besides, both models have low AIC and BIC values (though not the lowest), and more importantly, their residuals are similar to those of the other models (i.e., they don't resemble a white noise mainly because of the correlations at lags 12 and 24, but most of the other correlations are not significant), and the same applies to their in-sample fit.

3.1 12-step ahead forecast

As mentioned, the "best" models are the ARMA(3,9) model, followed by the AR(9):

```
arma39.fcast <- forecast.Arima(arma39, h = 12)
arma39.fcast2 <- predict(arma39, n.ahead = 12)
pander(arma39.fcast2$pred)
```

	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
2011	144.3	144.9	145.1	148.6	149.5	149.6	146.2	147	147	146.2	147.4	146.1

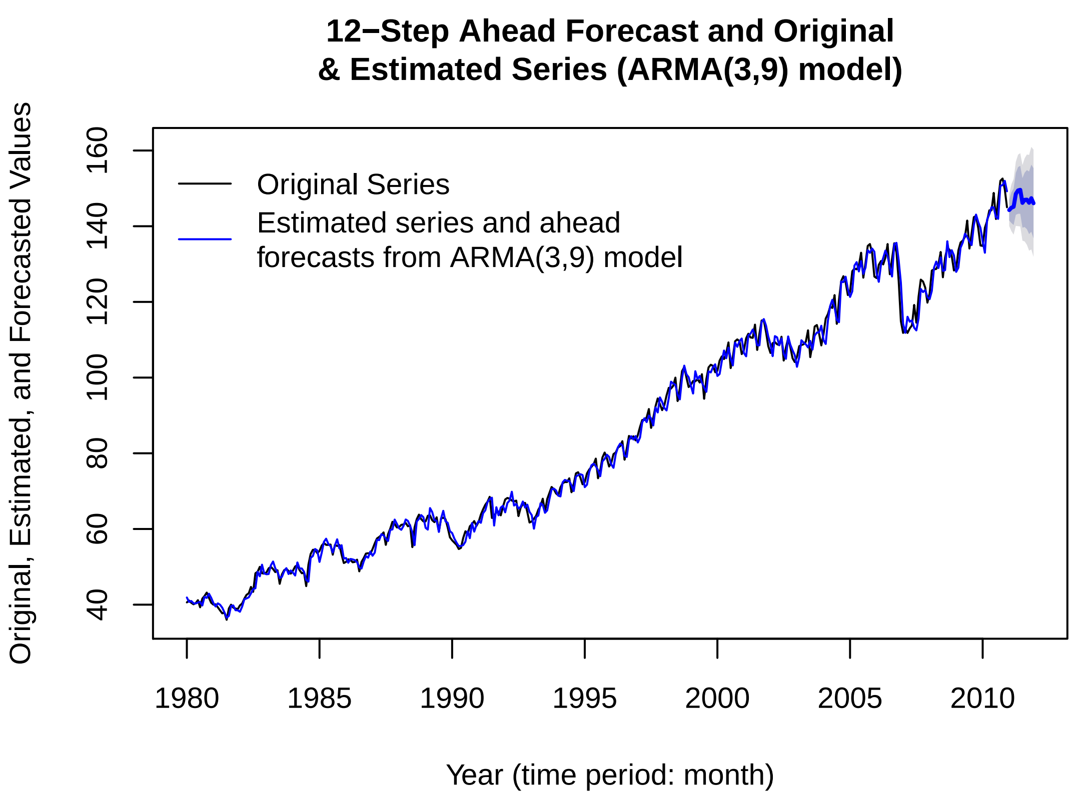

Figure 41: ARMA(3,9) model 12-step ahead forecast

Of course, the ARMA(3,9) model allows for a much variable forecast (as shown in the previous page), but its confidence region is approximately the same than that of the AR(9) model (shown below). In both cases, the forecast keeps increasing for the first observations, then decreases, then gets stable.

```
ar9.fcast <- forecast.Arima(ar9, h = 12)
ar9.fcast2 <- predict(ar9, n.ahead = 12)
pander(ar9.fcast2$pred)
```

	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
2011	147.1	148.4	148.8	149	148.6	148	147.3	147.6	148.1	148.2	147.9	147.7

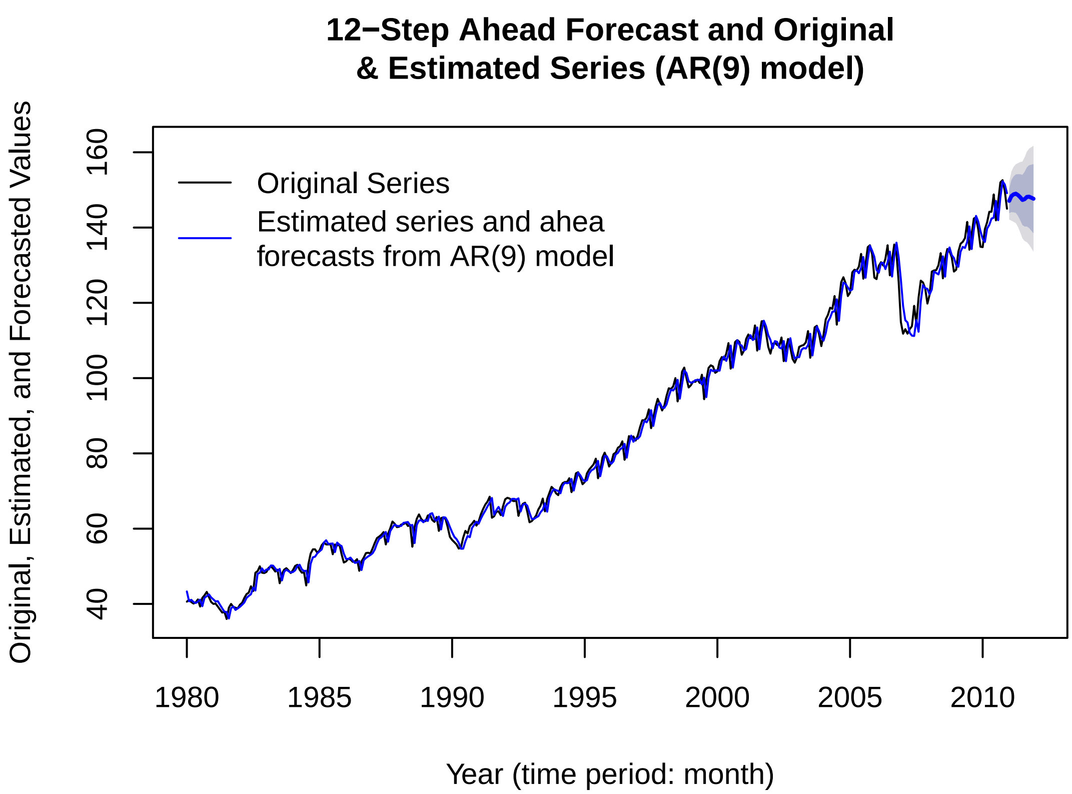

Figure 42: AR(9) model 12-step ahead forecast

We'll also forecast using the ARMA(1,5) model, which was just sligthly worse (in terms of out-of-sample fit, mainly) than the other two, but has the advantage of being simpler (6 coefficients instead of 12 and 9, respectively):

```
arma15.fcast <- forecast.Arima(arma15, h = 12)
arma15.fcast2 <- predict(arma15, n.ahead = 12)
pander(arma15.fcast2$pred)
```

	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
2011	146	146.9	147.3	146.9	146.8	146.8	146.7	146.7	146.7	146.7	146.6	146.6

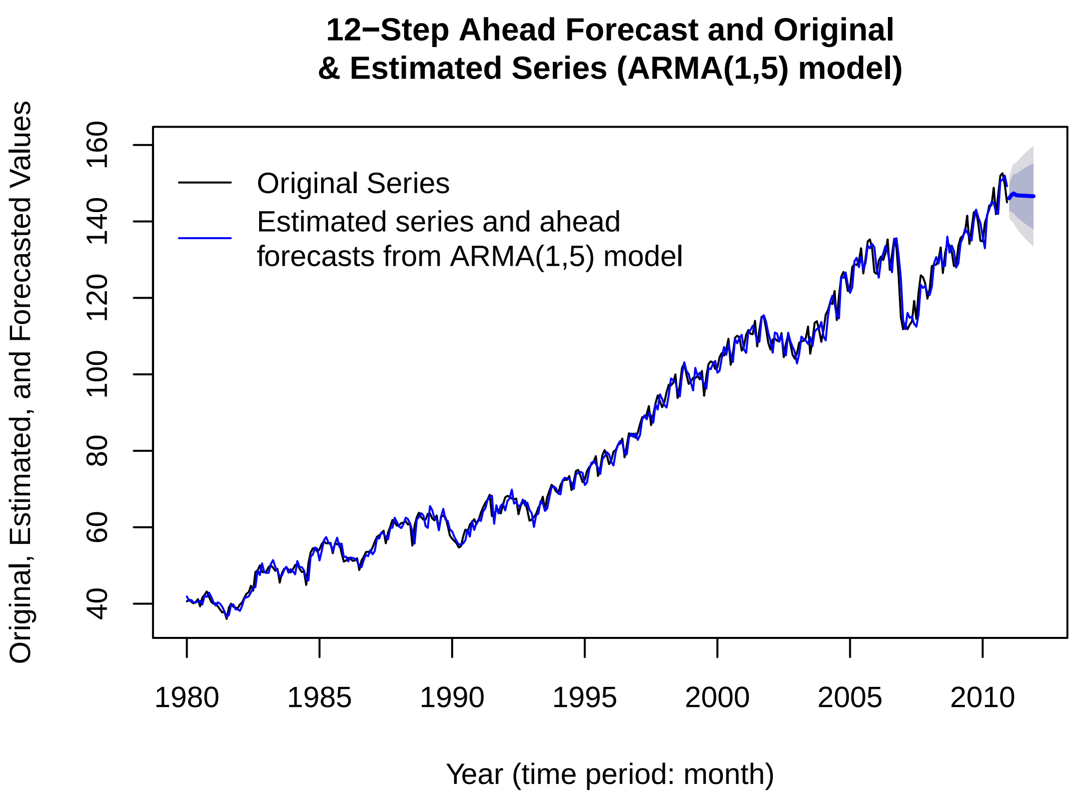

Figure 43: ARMA(1,5) model 12-step ahead forecast

The confidence region is very similar, though the yearly seasonality is not captured (the forecasts just oscillate around the final value: 146.6, close to the one in the other 2 models).

To finish (and forecast using at least one model of each kind), we also plot the forecasts for the "best" MA model, MA(12). As discussed previously, the MA forecasts tend to the mean value of the time series (about 85), which seems unlikely to happen.

```
ma12.fcast <- forecast.Arima(ma12, h = 12)
ma12.fcast2 <- predict(ma12, n.ahead = 12)
pander(ma12.fcast2$pred)
```

	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
2011	140.5	136.3	128.9	122	112.4	104.1	93.99	90.13	87.31	85.82	85.07	85.12

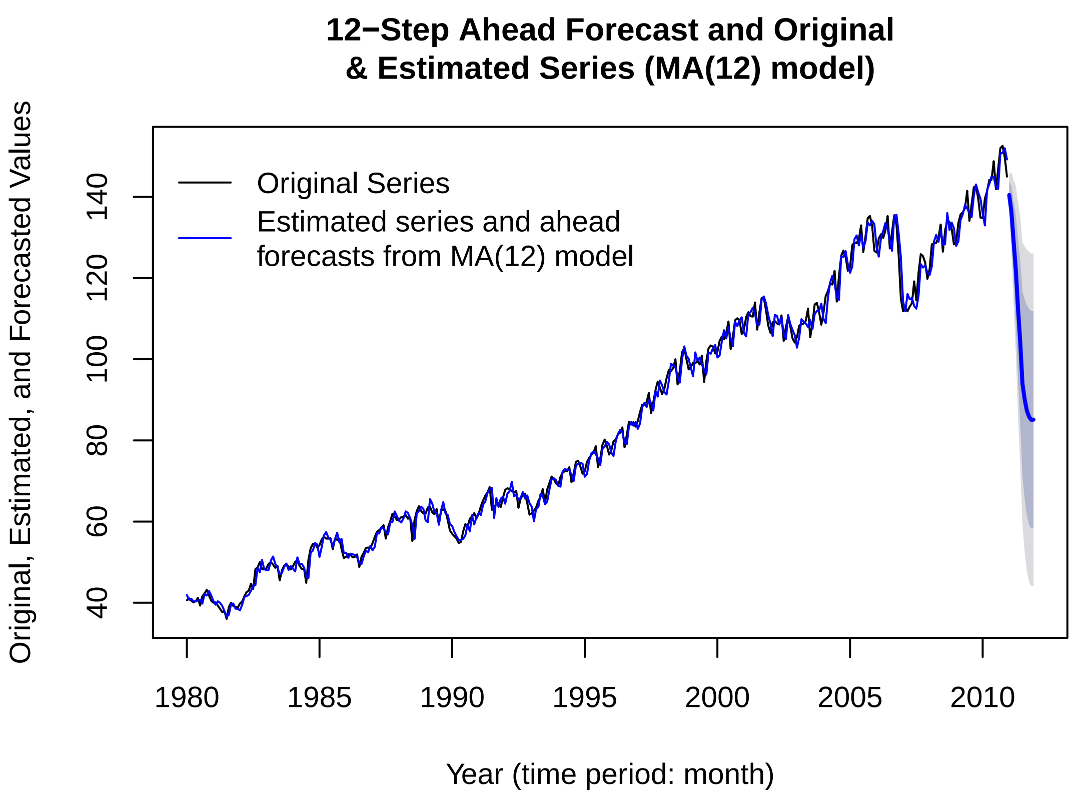

Figure 44: MA(12) model 12-step ahead forecast