Week 10: Forecasting and Time Series Regression

MATH-516 Applied Statistics

Tomas Masak

May 1st 2023

Section 1

Forecasting with SARIMA

Truth about Forecasting

Forecasting time series is like driving a car using the rear mirror only.

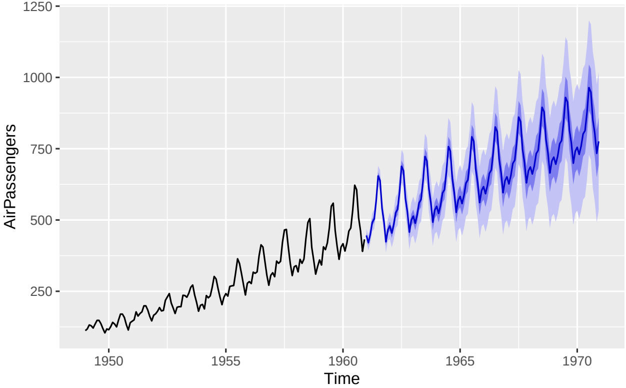
- though arguably more useful as a skill (the field is huge!)

Depending on who you ask SARIMA models are either:

- fairly old and simple, but surprisingly powerful
- overcomplicated and easily outperformed by vanilla methods
  - e.g. fitting a regression model and extrapolating

Air Passengers Revisited

```
library(forecast) # needed on every line below
# fit <- auto.arima(AirPassengers)
fit <- Arima(AirPassengers, order=c(2,1,1), seasonal=c(0,1,0)) # capital A
preds <- forecast(fit, 12*10, c(0.5,0.95)) # 2-year ahead forecast
autoplot(AirPassengers) + autolayer(preds)
```



Explanation

$$\Phi_{[P]}(B^s)\phi_{[p]}(B)(1-B^s)^D(1-B)^d X_t = \Theta_{[Q]}(B^s)\theta_{[q]}(B)Z_t$$

How are the point forecasts above calculated? In a recursive way:

1. Express X_t from the SARIMA equation above.
2. Replace t by $N+h$ in the equation.
3. Replace future observations (up to $N+h-1$) by their forecasts, future errors by zeros, and past errors by the respective residuals. (these are BLUPs given the past)

Under certain assumptions:

- $Y_t = (1-B^s)^D(1-B)^d X_t$ must be
  - stationary
  - *invertible*, i.e. polynomial $\Theta_{[Q]}(x^s)\theta_{[q]}(x)$ has no roots inside the unit circle, i.e. Z_t can be expressed as a linear combination of past X's
    - typically we also require *causality*, i.e. polynomial $\Phi_{[P]}(x^s)\phi_{[p]}(x)$ has no roots inside the unit circle, i.e. X_t can be expressed as a linear combination of past Z's

the procedure above results in the best linear unbiased prediction. (under Gaussianity, it is the best prediction in the mean square sense)

ARIMA(1,1,1) Example

Say that we fitted an ARIMA(1,1,1) model

$$(1-\widehat{\phi}B)(1-B)X_t = (1+\widehat{\theta}B)Z_t$$

The forecasting procedure above consists of the following steps:

1. $X_t = X_{t-1} + \widehat{\phi}X_{t-1} - \widehat{\phi}X_{t-2} + Z_t + \widehat{\theta}Z_{t-1}$
2. $X_{N+1} = X_N + \widehat{\phi}X_N - \widehat{\phi}X_{N-1} + Z_{N+1} + \widehat{\theta}Z_N$
3. $\widehat{X}_{N+1} = X_N + \widehat{\phi}X_N - \widehat{\phi}X_{N-1} + 0 + \widehat{\theta}\widehat{Z}_N$

This was a one-step ahead forecast. To do two steps ahead, replace N by $N+1$ in step 2 and repeat (replacing X_{N+1} by $\widehat{X}_{N+1}$ that is now available).

Prediction Intervals

- constructed under Gaussian assumptions
  - i.e. as $\widehat{X}_{N+h} \pm q_{1-\alpha/2}\widehat{\sigma}_h$, where $q_{1-\alpha/2}$ is a Gaussian quantile and $\widehat{\sigma}_h$ is the standard error of the prediction, calculated with estimators plugged in (details omitted), hence
- only reflects uncertainty about future realizations of Z's!
- including uncertainty about parameter estimation possible via Monte Carlo simulation
  - i.e. parametric bootstrap, drawing parameter values for every simulation from their respective asymptotic distribution (was not discussed, but exists)
  - also possible to do non-parametric bootstrap instead to protect ourselves against non-Gaussian Z's (set `bootstrap=T` in `forecast()`)
- including uncertainty about model selection requires a more careful simulation study
  - a relatively easy but somewhat arbitrary option is to simulate from an unnecessarily large model and perform automatic model selection for every simulation run

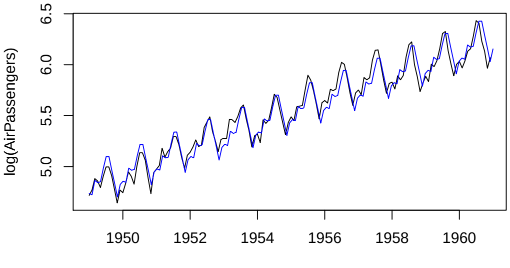
Section 2

Time Series Regression

Air Passengers Revisited

- say the trendin Air Passangers data set is linear (after log) and we want a confidence interval on the slope.

```
lmData <- data.frame(y = log(AirPassengers),
                     month = as.factor(rep(1:12,length(AirPassengers)/12)),
                     t = 1:length(AirPassengers))
lmfit <- lm(y~., data=lmData)
plot(log(AirPassengers))
points(seq(1949, 1961, length=144),fitted(lmfit), type="l", col="blue")
```



Air Passengers Revisited

```
summary(lmfit)$coefficients
```

```
##                 Estimate   Std. Error     t value      Pr(>|t|)
## (Intercept)  4.726780368 0.0188935382 250.1797341 1.966475e-177
## month2      -0.022054823 0.0242108707  -0.9109471  3.639964e-01
## month3       0.108172299 0.0242117524   4.4677600  1.691103e-05
## month4       0.076903445 0.0242132220   3.1760930  1.861594e-03
## month5       0.074530803 0.0242152792   3.0778420  2.539909e-03
## month6       0.196677004 0.0242179239   8.1211339  2.980730e-13
## month7       0.300619331 0.0242211560  12.4114362  7.210080e-24
## month8       0.291324492 0.0242249751  12.0257912  6.615459e-23
## month9       0.146689890 0.0242293811   6.0542153  1.392768e-08
## month10      0.008531649 0.0242343735   0.3520474  7.253684e-01
## month11     -0.135186061 0.0242399521  -5.5769937  1.343178e-07
## month12     -0.021321065 0.0242461164  -0.8793600  3.808163e-01
## t            0.010068805 0.0001193001  84.3989506 4.561377e-116
```

```
confint(lmfit)["t",]
```

```
##       2.5 %      97.5 %
## 0.009832801 0.010304809
```

- Problem: data are not i.i.d.
- going back to the proof of the Gauss-Markov theorem, we would realize
  - the estimates are still unbiased (does not mean much)
  - they are no longer "best" (among linear unbiased estimators)
  - the standard errors, and hence also CIs, are wrong (thinking there is more df)

Generalized Least Squares

- standard least squares (regression) model:
  - $Y = \mathbf{X}\beta + \epsilon$ with $\epsilon \sim \mathcal{N}(\mathbf{0}, \sigma^2 \mathbf{I})$
- generalized least squares model:
  - $Y = \mathbf{X}\beta + \epsilon$ with $\epsilon \sim \mathcal{N}(\mathbf{0}, \sigma^2 \Sigma)$
- if we knew the dependency structure Σ, we could change the variables:
  - $\Sigma^{-1/2} Y = \Sigma^{-1/2}\mathbf{X}\beta + \Sigma^{-1/2}\epsilon$
  - $\widetilde{Y} = \widetilde{X}\beta + \widetilde{\epsilon} \ldots$ standard least squares work here (and they correspond to Gaussian MLE)
- if we knew the dependency structure up to a fixed no. of parameters $\alpha \in \mathbb{R}^q$, we could do the Gaussian MLE jointly for β, σ^2, α
  - this is exactly the case when the residuals follow an ARMA model
  - one could also iterate between the change of variables and fitting ARMA to the residuals
    - this is called Cochrane-Orcutt method, and it provably converges to the MLE, which is usually obtained numerically (`gls()` from the `nlme` package in `R`)

Air Passengers Revisited

- lets say that AR(1) is a good model for the residuals above
  - this is what we can decide doing time series analysis like last week

```
library(nlme)
R_struct <- corARMA(form=~t, p=1) # AR(1) correlation structure
glsfit <- gls(y~t+month, data=lmData, corr=R_struct, method="ML")
confint(lmfit)["t",]
```

```
##       2.5 %      97.5 %
## 0.009832801 0.010304809
```

```
confint(glsfit)["t",] # wider than above
```

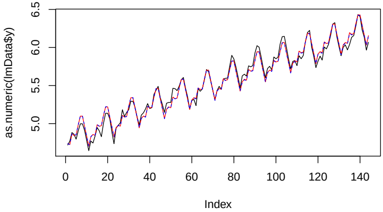
```
##       2.5 %      97.5 %
## 0.009364262 0.010624771
```

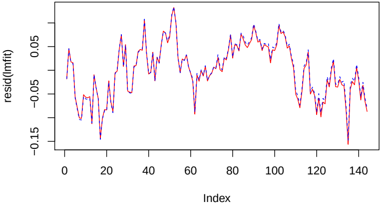
Air Passengers Revisited

- fitted values are typically very similar, and so are the residuals
  - if they are not, we should re-think our dependency structure

```
plot(as.numeric(lmData$y),type="l")
points(fitted(lmfit),type="l", col="red")
points(fitted(glsfit),type="l", col="blue", lty=2)

plot(resid(lmfit),type="l", col="red")
points(as.numeric(resid(glsfit)),type="l", col="blue", lty=2)
```





Hypothesis Testing

Maximum likelihood theory (potentially with nuissance parameters) works here:

- let $Y \sim \mathcal{N}_N(\mathbf{X}\beta, \sigma^2\Sigma)$, where $\Sigma \in \mathbb{R}^{N \times N}$ depends on a parameter vector $\alpha \in \mathbb{R}^r$ (of a fixed sized as $N \to \infty$)
- let $\theta \in \mathbb{R}^p$ is a vector containing all the parameters, i.e. β, σ^2, α, such that the hypothesis concerns the first q entries of θ, namely:
- let $\Theta_1 \times \Theta_c$, where $\Theta_1 \subset \mathbb{R}^q$ and $\Theta_c \in \mathbb{R}^{p-q}$ is the parameter space, and $\Theta_0 \subset \Theta_1$
  - $H_0 : \theta \in \Theta_0 \times \Theta_c$
  - $H_1 : \theta \in \Theta_1 \times \Theta_c$
- then under H_0 the likelihood ratio statistics approaches the χ^2-distribution:

$$LR_N = 2\big[\ell_N(\widehat{\theta}_1) - \ell_N(\widehat{\theta}_0)\big] \dot{\sim} \chi^2_q$$

  - $\widehat{\theta}_0 = \arg\max_{\theta \in \Theta_0 \times \Theta_c} \ell_N(\theta)$
  - $\widehat{\theta}_1 = \arg\max_{\theta \in \Theta_1 \times \Theta_c} \ell_N(\theta)$

Air Passengers Revisited

- `anova()` in R works
  - use `method="ML"` is used instead of `"REML"` when changing the mean structure

```
library(car)
# glsfit <- gls(y~t+month, data=lmData, corr=R_struct, method="ML")
Anova(glsfit, type=2)
```

```
## Analysis of Deviance Table (Type II tests)
##
## Response: y
##       Df  Chisq Pr(>Chisq)
## t      1 966.02  < 2.2e-16 ***
## month 11 911.92  < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# or half-manually:
subfit_1 <- gls(y~month, data=lmData, corr=R_struct, method="ML")
subfit_2 <- gls(y~t, data=lmData, corr=R_struct, method="ML")
```

Air Passengers Revisited

```
anova(subfit_1, glsfit)
```

```
##          Model df       AIC       BIC   logLik   Test L.Ratio p-value
## subfit_1     1 14 -498.1899 -456.6125 263.0950
## glsfit       2 15 -526.1789 -481.6317 278.0895 1 vs 2  29.989  <.0001
```

```
anova(subfit_2, glsfit)
```

```
##          Model df       AIC       BIC   logLik   Test  L.Ratio p-value
## subfit_2     1  4 -251.3892 -239.5100 129.6946
## glsfit       2 15 -526.1789 -481.6317 278.0895 1 vs 2 296.7897  <.0001
```

```
# or entirely manually:
1-pchisq(2*(glsfit$logLik - subfit_1$logLik),
         length(coef(glsfit)) - length(coef(subfit_1)))
```

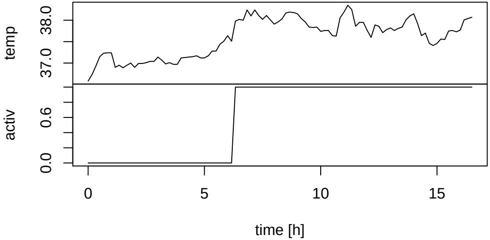
```
## [1] 4.345036e-08
```

```
1-pchisq(2*(glsfit$logLik - subfit_2$logLik),
         length(coef(glsfit)) - length(coef(subfit_2)))
```

```
## [1] 0
```

Beaver Body Temperature

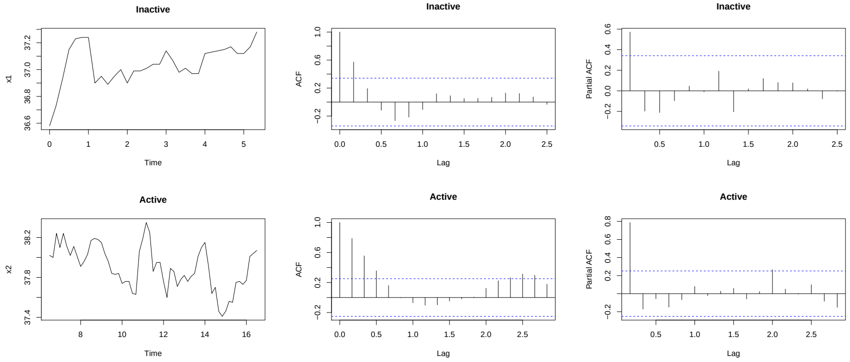
```
data(beavers)
beaver2 <- beaver2[,c(3,4)] # drop time
x <- ts(beaver2, start=0,frequency=6)
plot(x, main="", xlab="time [h]")
```



Question: What is the effect of activity on the body temperature?

Beaver Body Temperature

```r
x1 <- window(x, start=0, end=max(which(x[,2]==0))/6-1)[,1]
x2 <- window(x, start=min(which(x[,2]==1))/6)[,1]
plot(x1,main="Inactive");acf(x1, main="Inactive"); pacf(x1, main="Inactive"
plot(x2, main="Active"); acf(x2, main="Active"); pacf(x2, main="Active")
```



Lags have no meaning here, let's take AR(1) as the model

Beaver Body Temperature

```r
beaver2$time <- seq_along(beaver2$temp)
R_struct <- corARMA(form=~time, p=1) # AR(1) correlation structure
( glsfit <- gls(temp~activ, data=beaver2, corr=R_struct, method="ML") )
```

```
## Generalized least squares fit by maximum likelihood
##   Model: temp ~ activ
##   Data: beaver2
##   Log-likelihood: 66.77523
##
## Coefficients:
## (Intercept)       activ
##  37.1919453   0.6141776
##
## Correlation Structure: AR(1)
##  Formula: ~time
##  Parameter estimate(s):
##       Phi
## 0.8731771
## Degrees of freedom: 100 total; 98 residual
## Residual standard error: 0.2527856
```

Beaver Body Temperature (Alternatively)

```
arima(x[,1], c(1,0,0), xreg=x[,2])
```

```
##
## Call:
## arima(x = x[, 1], order = c(1, 0, 0), xreg = x[, 2])
##
## Coefficients:
##          ar1  intercept  x[, 2]
##       0.8733    37.1920  0.6139
## s.e.  0.0684     0.1187  0.1381
##
## sigma^2 estimated as 0.01518:  log likelihood = 66.78,  aic = -125.55
```

- standard errors are slightly different with `arima()`, since calculated empirically, while `gls()` uses asymptotic formula
- in both cases, activity increases body temperature of the beaver by about 0.6 degrees
- we could test significance of this value as above (but we see from s.e. that clearly significant)