Problem Background

Fitting Time Series Models

In this lab we are going to fit time series models to data sets consisting of daily returns on various instruments.

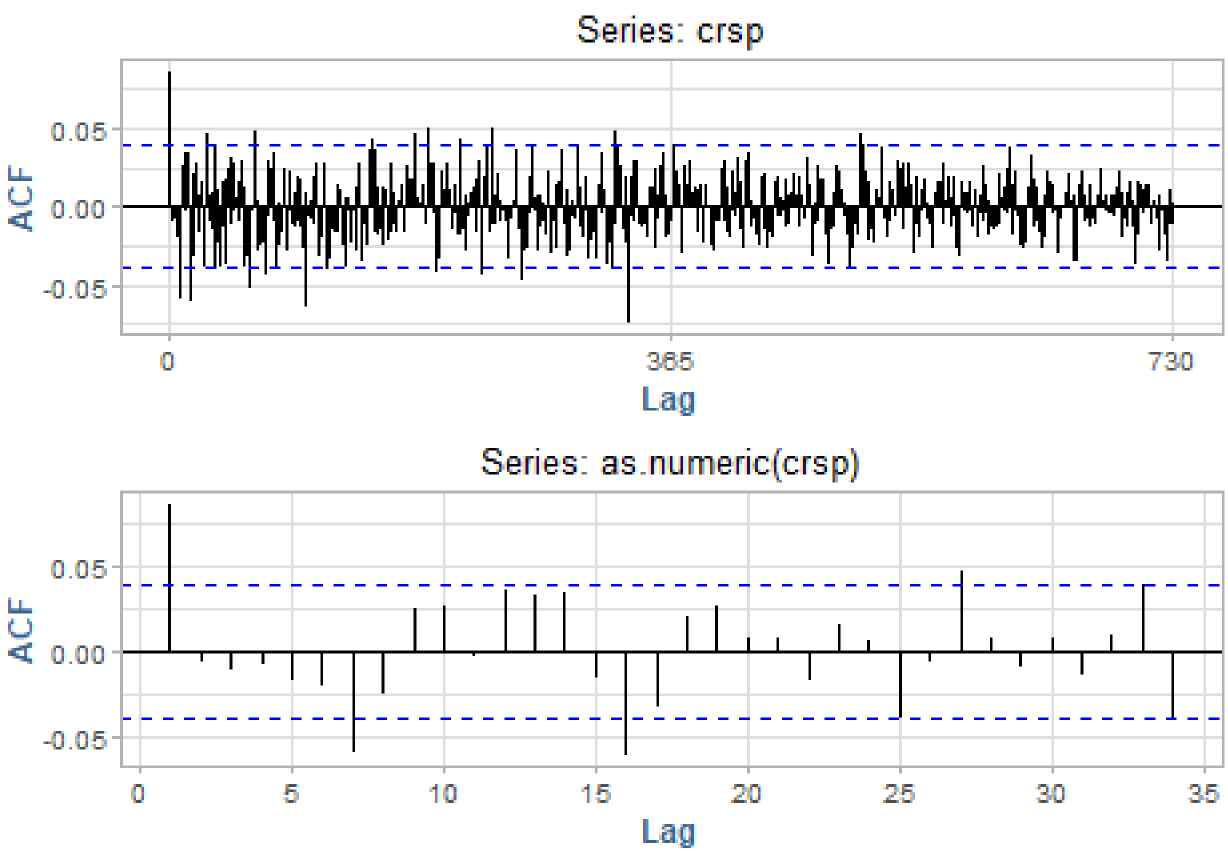
First, we will look a set of CRSP daily returns.

```
data("CRSPday")

crsp <- CRSPday[, 7]
```

Problem 1

Explain what "lag" means in the two ACF plots. Why does lag differ between the plots?

```
p1 <- ggAcf(crsp)
p2 <- ggAcf(as.numeric(crsp))

grid.arrange(p1, p2, nrow =2)
```



```r
head(crsp) # peek the ts object
```

```
Time Series:
Start = c(1969, 1)
End = c(1969, 6)
Frequency = 365
[1] -0.007619  0.013016  0.002815  0.003064  0.001633 -0.001991
```

Lag is a function of the frequency of the time series object *crsp*, which set the unit of time inverval represented by each data point.

From the quick summary of the data, we see that the frequency is 365 (days/year), so the first plot represents and interval of 1/365, or 0.00274. When we cast the data to a pure numeric representation (*as.vector*), this truncates the frequency property from the time series object, and the default reverts to 7/365, or 0.01918. The charts have the same data, just displayed on different time scales.

At what values of lag are there significant autocorrelations in the CRSP returns?

Let's grab the data from the plot for analysis.

```r
vals <- as.data.table(p2$data)[, .(Acf = Freq, Lag = lag)]

sig.vals <- vals[vals$Acf > 0.05 | vals$Acf < -0.05]

pretty_kable(sig.vals, "Significant Autocorrelations", dig = 2)
```

Table 1: Significant Autocorrelations

Acf	Lag
0.09	1
-0.06	7
-0.06	16

We can see the lags with the most significant values are at: 1, 7 and 16.

For which of these values do you think the statistical significance might be due to chance?

We can run a Ljung-Box test on these lags to further test for significance which test successive lags for stronger confidence.

```r
Box.test(crsp, lag = 1, type = "Ljung-Box")
```

```

    Box-Ljung test

data:  crsp
X-squared = 18.41, df = 1, p-value = 1.781e-05
```

At lag 1, we strongly reject the null hypothesis and conclude serial correlation.

```
Box.test(crsp, lag = 7, type = "Ljung-Box")
```

```

	Box-Ljung test

data:  crsp
X-squared = 29.509, df = 7, p-value = 0.0001168
```

At lag 7, we still reject the null and conclude there is serial correlation, but with less confidence than at 1 lag.

```
Box.test(crsp, lag = 16, type = "Ljung-Box")
```

```

	Box-Ljung test

data:  crsp
X-squared = 53.068, df = 16, p-value = 7.355e-06
```

At lag 16, we accept the null hypothesis and conclude this is i.i.d, and the correlation is from randomness.

Problem 2

Next, we will fit AR(1) and AR(2) models to the CRSP returns:

```
(fit1 <- arima(crsp, order = c(1, 0, 0)))
```

```
Call:
arima(x = crsp, order = c(1, 0, 0))

Coefficients:
         ar1  intercept
      0.0853      7e-04
s.e.  0.0198      2e-04

sigma^2 estimated as 5.973e-05:  log likelihood = 8706.18,  aic = -17406.37
```

```
(fit2 <- arima(crsp, order = c(2, 0, 0)))
```

```
Call:
arima(x = crsp, order = c(2, 0, 0))

Coefficients:
         ar1      ar2  intercept
      0.0865  -0.0141      7e-04
s.e.  0.0199   0.0199      2e-04

sigma^2 estimated as 5.972e-05:  log likelihood = 8706.43,  aic = -17404.87
```

In comparing these two models we would take the one with lower Akaike information criterion (AIC), or Bayesian information criterion (BIC).

```
pretty_kable(data.table(Model = c("AR(1)", "AR(2)"),
                        AIC = c(AIC(fit1), AIC(fit2)),
                        BIC = c(BIC(fit1), BIC(fit2))), "Model Fit Comparison")
```

Table 2: Model Fit Comparison

Model	AIC	BIC
AR(1)	-17406.37	-17388.86
AR(2)	-17404.87	-17381.53

Here, we would take AR(1) over AR(2), irrespective of the preferred metric.

Find a 95% confidence interval for ϕ for the AR(1) model:

```
alpha <- 0.05

ci <- fit1$model$phi + 0.019 * qnorm(1 - (alpha/2)) * c(-1, 1)

pretty_kable(data.table(Lower = ci[1], Upper = ci[2]), "95\\% Confidence Interval", dig = 5)
```

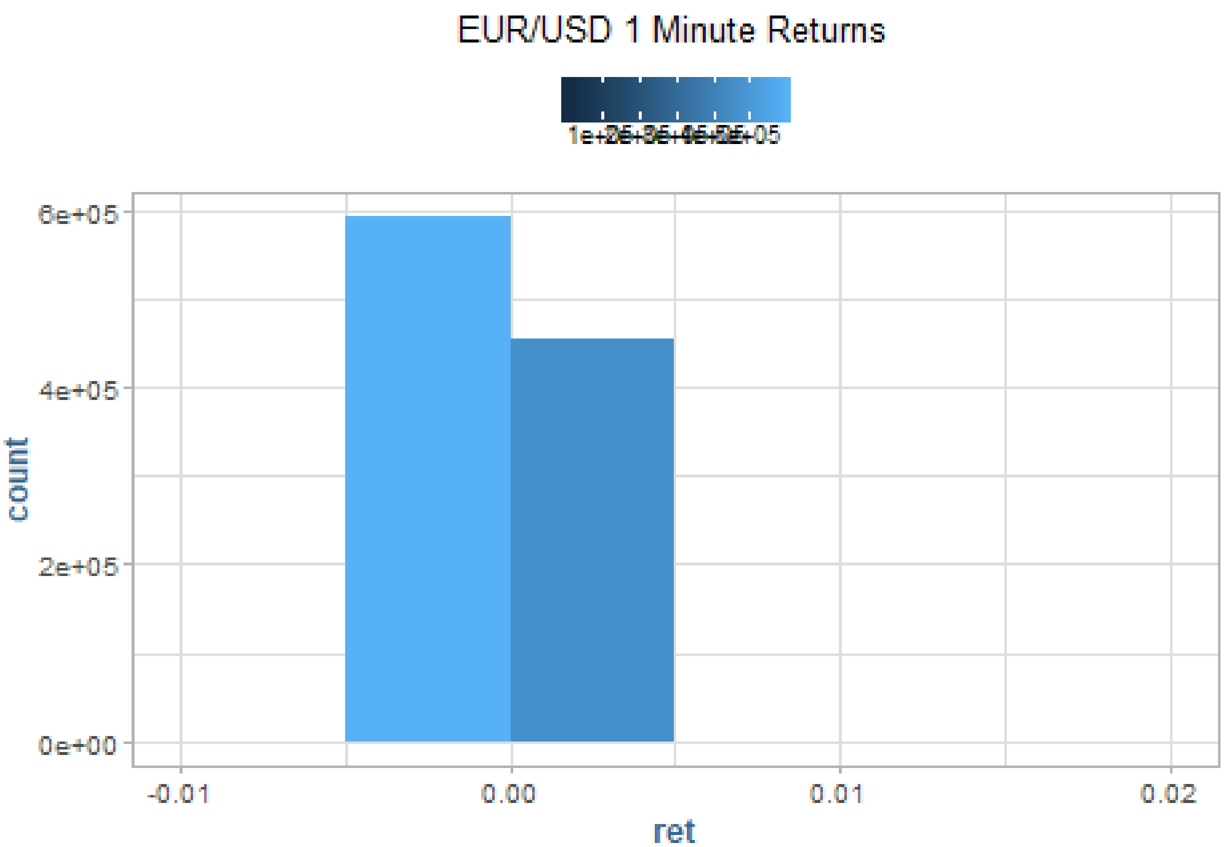
Table 3: 95% Confidence Interval

Lower	Upper
0.04806	0.12254

Problem 3

Next, will look at EURUSD currency rate data on a one minute interval.

```
eurusd <- read.csv(paste0(data.dir, "EURUSD mid.csv"),
                   header = F)

prices <- eurusd[, 6]
n <- length(prices)
returns <- diff(prices)/prices[1:(n-1)]

dat <- data.table(ret = returns)

ggplot(dat, aes(ret)) +
   geom_histogram(aes(fill = ..count..), breaks = pretty(dat$ret)) +
   labs(title = "EUR/USD 1 Minute Returns")
```




```
pretty_kable(data.table( Mean = mean(returns), SD = sd(returns)), "EUR/USD Summary", dig = 5)
```

Table 4: EUR/USD Summary

Mean	SD
0	0.00021

Problem 4

Now we will find the 'best' AR(p) model, **m0**, for the return series using the Bayesian information criterion.

For the training data, we will use the first 1M bars.

```
train.size <- 1000000
test.size <- 1000

data.train <- returns[1:train.size]
data.test <- returns[train.size+1:test.size]

stopifnot(length(data.train) == train.size & length(data.test) == test.size)

summary(m0.train <- auto.arima(data.train, ic = "bic"))
```

```
Series: data.train
ARIMA(0,0,0) with zero mean

sigma^2 estimated as 4.482e-08:  log likelihood=7041363
AIC=-14082723   AICc=-14082723   BIC=-14082711

Training set error measures:
                        ME         RMSE          MAE MPE MAPE      MASE
Training set -7.600912e-09 0.0002117083 0.0001290001 100  100 0.6852113
                    ACF1
Training set 0.001829255
```

The best AR(p) model for the EUR/USD rates is an **AR(0)** model.

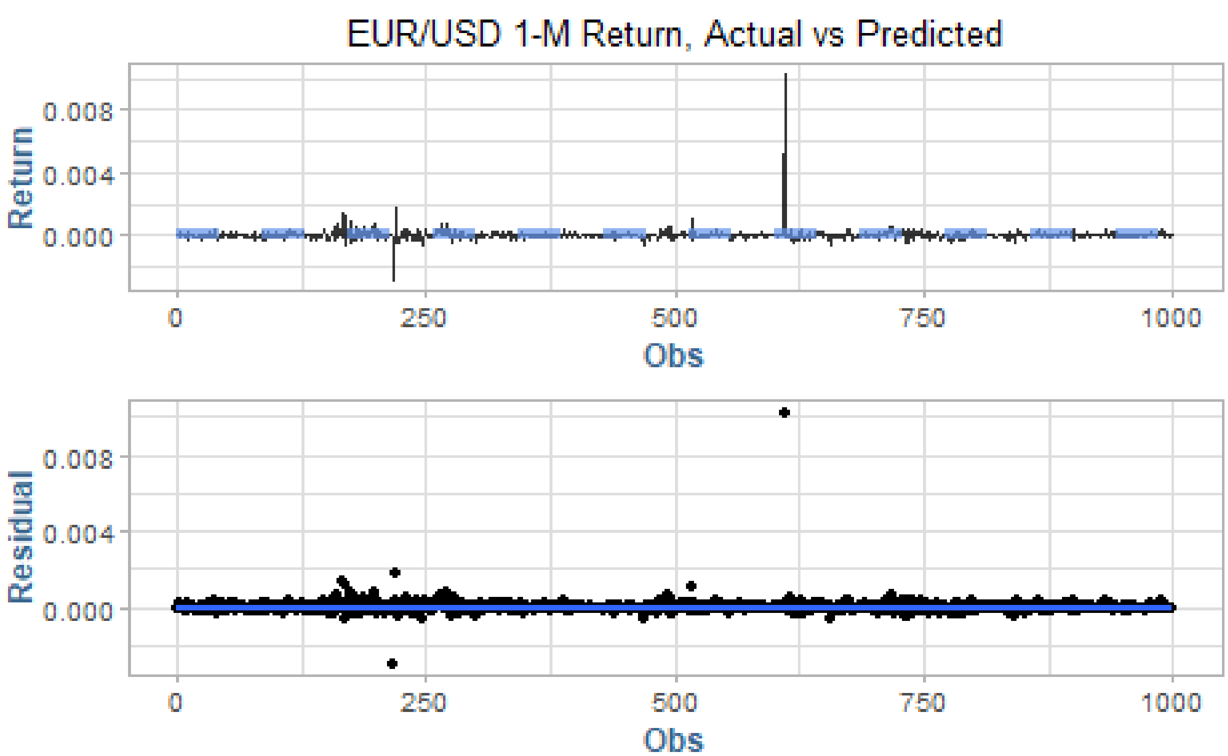
```
summary(m0.test <- Arima(data.test, model = m0.train))
```

```
Series: data.test
ARIMA(0,0,0) with zero mean

sigma^2 estimated as 4.482e-08:  log likelihood=6437.07
AIC=-12872.15   AICc=-12872.14   BIC=-12867.24

Training set error measures:
```

```
                       ME         RMSE          MAE MPE MAPE      MASE
Training set 1.864936e-05 0.0003874158 0.0001381423 100  100 0.6602008
                   ACF1
Training set -0.0209842
```

```
m0.forecast <- forecast(m0.test)
m0.results <- data.table(Actual = data.test, Pred = m0.forecast$fitted, Residual = m0.forecast$

f1 <- ggplot(m0.results, aes(x = Obs)) +
  geom_line(aes(y = Actual), lwd = .5, col = "black", alpha = .8) +
  geom_line(aes(y = Pred), lwd = 1.5, col = "cornflowerblue", alpha = .7, linetype = 2) +
  labs(title = "EUR/USD 1-M Return, Actual vs Predicted", y = "Return")

f2 <- ggplot(m0.results, aes(x = Obs, y = Residual)) +
   geom_point() +
   geom_smooth(method = "lm")

grid.arrange(f1, f2, nrow = 2)
```

```
Don't know how to automatically pick scale for object of type ts. Defaulting to continuous.
```



```
m0.accuracy <- m0.results[, .(Correct = Actual == Pred)][,
                           .(Correct = sum(Correct), Total = .N,
                             Pct = (sum(Correct) / .N) * 100)]

pretty_kable(m0.accuracy, "m0 Prediction Accuracy", dig = 5)
```

Table 5: m0 Prediction Accuracy

Correct	Total	Pct
104	1000	10.4