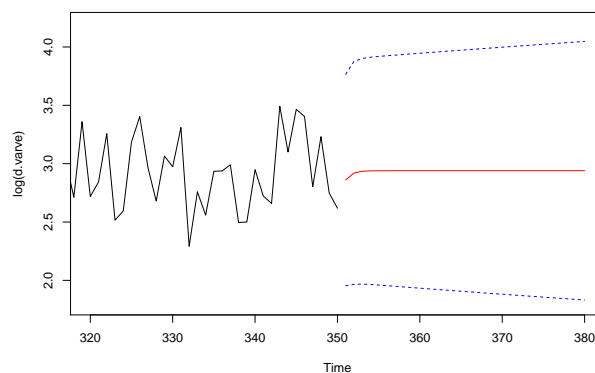


Solution to Series 8

1. a) The predicted value stabilizes after just a few points in time, and the prediction band grows linearly. Both these properties were to be expected from the theory.

R code:

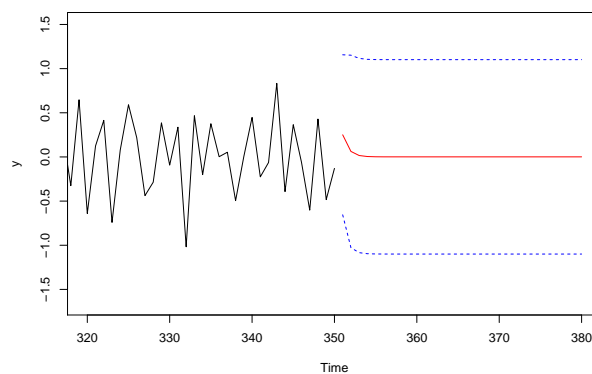
```
> ## Fitting the model
> d.varve <- ts(d.varve)
> r.varve <- arima(log(d.varve), order=c(1,1,1))
> r.pred <- predict(r.varve, n.ahead=30) ## Prediction
> plot(log(d.varve), xlim=c(320,380), ylim=c(1.8, 4.2)) ## Plotting
> lines(r.pred$pred, col=2)
> lines(r.pred$pred + 1.96 * r.pred$se, lty=2, col=4)
> lines(r.pred$pred - 1.96 * r.pred$se, lty=2, col=4)
```



- b) Here the predicted change is also quite quick to stabilize at the mean of first-order differences. The prediction variance is constant after just a few time points, and is only marginally smaller than the variance of the process.

R code:

```
> y <- diff(log(d.varve))
> ## Fitting the model
> r.varve.2 <- arima(y, order=c(1,0,1))
> r.pred.2 <- predict(r.varve.2, n.ahead=30) ## Prediction
> plot(y, xlim=c(320,380)) ## Plotting
> lines(r.pred.2$pred, col=2)
> lines(r.pred.2$pred + 1.96 * r.pred.2$se, lty=2, col=4)
> lines(r.pred.2$pred - 1.96 * r.pred.2$se, lty=2, col=4)
```

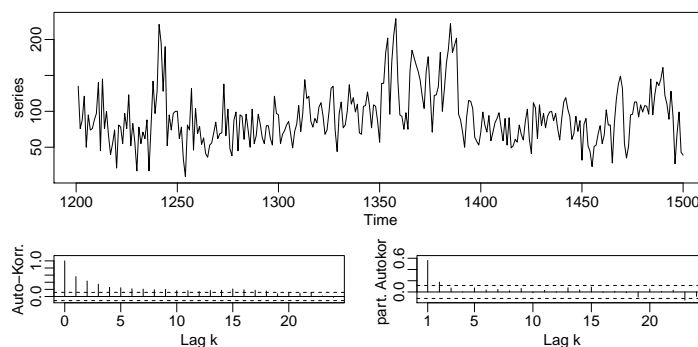


c) For Part a) the predicted values need only undergo an `exp` transformation. This means if we calculate `exp(r.pred$pred)`, we get the predicted values on the original scale. In Part b), however, the formation of differences needs reversing (by taking the cumulative sum). If we apply `predict()` to an `arima` object, R does this for us. If we use the differences, we have to apply the cumulative sum to the predicted values first, then do the `exp` transformation and then add the last known entry of the timeseries: `exp(cumsum(r.pred.2$pred))+d.varve[350]`

2. a) Here only some suggestions instead of solutions are offered. Other models might be just as well or even better.

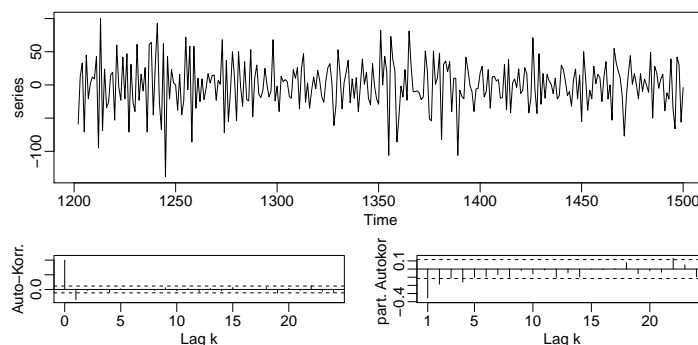
Pine data:

```
> f.acf(d.foe)
```



The acf-plot is decaying very slowly. There might be a trend.

```
> f.acf(diff(d.foe, 1))
```



The differenced data looks better, could either be MA(1) or ARMA(1,1). If you check the residuals, ARMA(1,1) yields better results and MA(1) still shows some dependence. So we continue with an ARIMA(1,1,1)-model. There is no season since we have yearly data.

```
> fit <- arima(d.foe, order=c(1,1,1))
```

```
> fit
```

Call:

```
arima(x = d.foe, order = c(1, 1, 1))
```

Coefficients:

```
      ar1      ma1
      0.406 -0.907
s.e.  0.075  0.041
```

```
sigma^2 estimated as 1005: log likelihood = -1458, aic = 2922
```

```
> t.pr <- predict(fit, n.ahead=50)
```

```
> t.u <- t.pr$pred+1.96*t.pr$se
```

```
> t.l <- t.pr$pred-1.96*t.pr$se
```

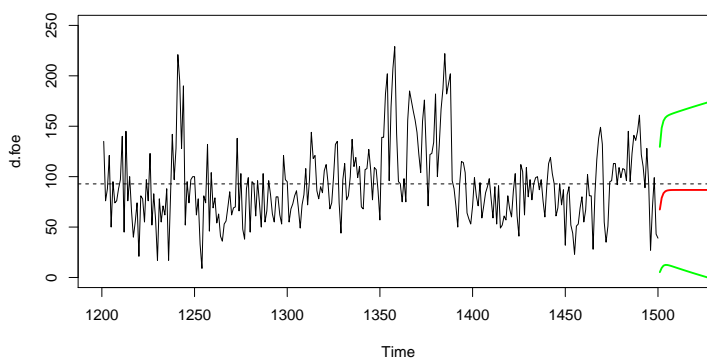
```
> plot(d.foe, xlim=c(1200, 1520), ylim=c(0, 250))
```

```
> abline(h=mean(d.foe), lty=2)
```

```
> lines(t.pr$pred, col="red", lwd=2)
```

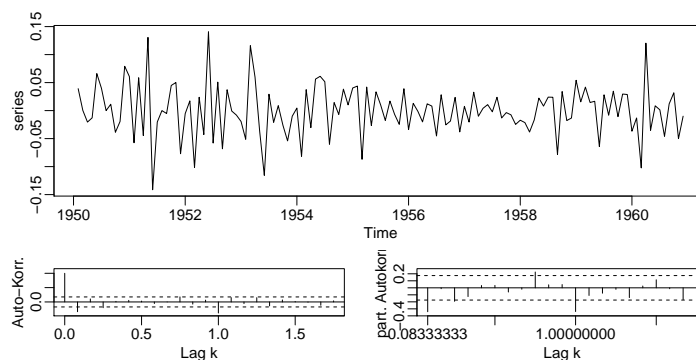
```
> lines(t.u, col="green", lwd=2)
```

```
> lines(t.l, col="green", lwd=2)
```



Airline data: Use the log-transform, we have seasonal data and a trend.

```
> d.air1 <- log(AirPassengers)
> d.air2 <- diff(d.air1, lag=12)
> d.air3 <- diff(d.air2, 1)
> f.acf(d.air3)
```



A $SARIMA(0,1,1)(0,1,1)^{12}$ could be an appropriate model.

```
> s1.air <- arima(log(AirPassengers), order=c(0,1,1), seasonal=c(0,1,1))
> s1.air
```

Call:

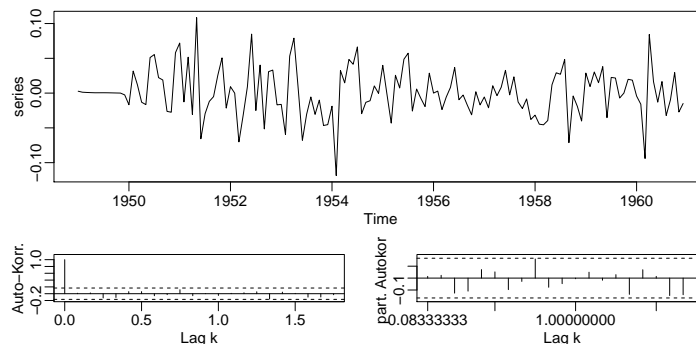
```
arima(x = log(AirPassengers), order = c(0, 1, 1), seasonal = c(0, 1, 1))
```

Coefficients:

	ma1	sm1
	-0.402	-0.557
s.e.	0.090	0.073

sigma² estimated as 0.00135: log likelihood = 245, aic = -483

```
> f.acf(s1.air$residuals)
```



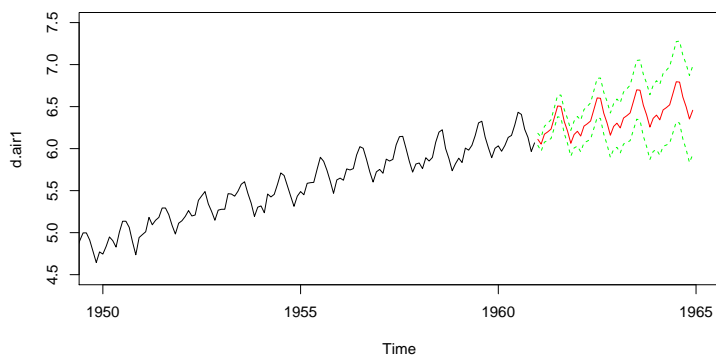
The residuals look ok.

```
> t.pr <- predict(s1.air, n.ahead=48)
> plot(d.air1, xlim=c(1950,1965), ylim=c(4.5,7.5))
> t.u <- t.pr$pred+1.96*t.pr$se
> t.l <- t.pr$pred-1.96*t.pr$se
```

```

> lines(t.pr$pred, col="red")
> lines(t.u, col="green", lty=2)
> lines(t.l, col="green", lty=2)

```

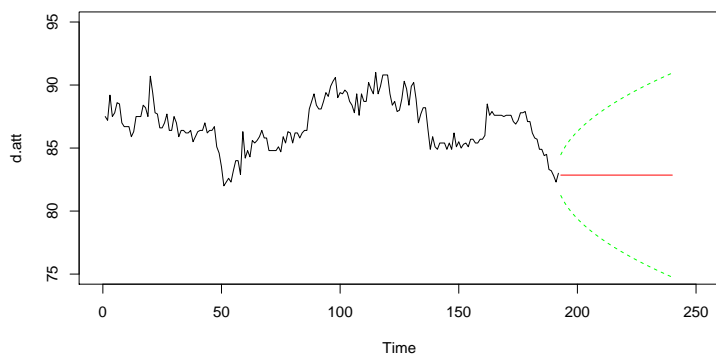


The AT&T bonds- data could be modelled using an ARIMA(0,1,1):

```

> s.att <- arima(d.att, order=c(0,1,1))
> t.pr <- predict(s.att, n.ahead=48)
> plot(d.att, xlim=c(0,250), ylim=c(75,95))
> t.u <- t.pr$pred+1.96*t.pr$se
> t.l <- t.pr$pred-1.96*t.pr$se
> lines(t.pr$pred, col="red")
> lines(t.u, col="green", lty=2)
> lines(t.l, col="green", lty=2)

```

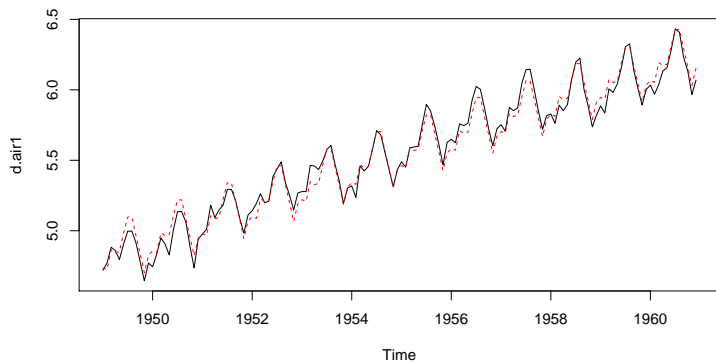


- b) A possible solution is shown for the airline data only. The trend looks linear, but a quadratic term was added, after looking at a model using only a linear trend in time.

```

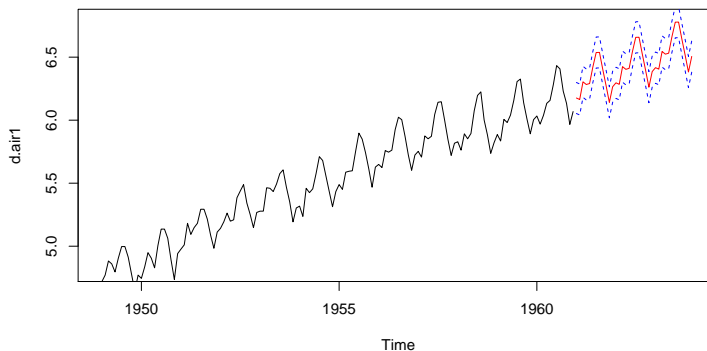
> plot(d.air1)
> time <- 1:length(d.air1)
> time <- time/12+1949
> Month <- factor(rep(month.name, length(d.air1)/12, levels=month.name))
> fit <- lm(d.air1~Month+time+I(time^2))
> fit <- lm(d.air1~Month+log(time))
> summary(fit)
> ts.fit <- ts(fitted(fit), start=1949, frequency=12)
> lines(ts.fit, col="red", lty=2)

```



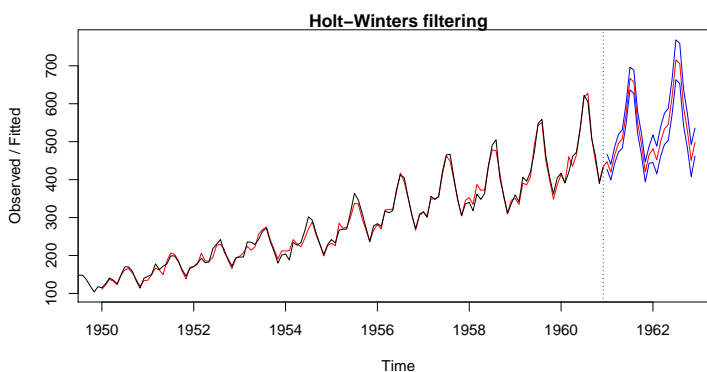
Prediction:

```
> t.pr <- predict(fit, newdata=data.frame(time=seq(1961,1964-1/12, by=1/12),
      Month=rep(month.name, 3)), interval="prediction")
> plot(d.air1, xlim=c(1949,1964), ylim=c(4.8,6.8))
> t.pr <- ts(t.pr, start=1961, frequency=12)
> lines(t.pr[,1], col="red")
> lines(t.pr[,2], col=4, lty=2)
> lines(t.pr[,3], col=4, lty=2)
```



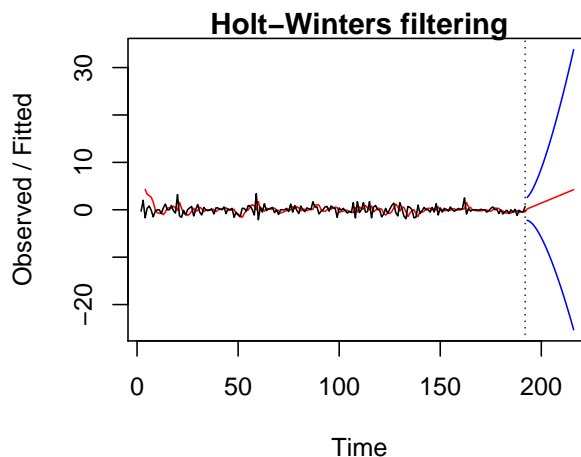
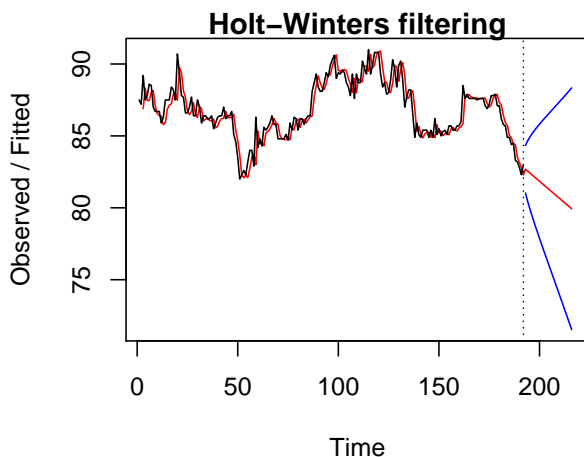
c) Airline data:

```
> fit <- HoltWinters(d.air, seasonal="multiplicative")
> t.pr <- predict(fit, 24, prediction.interval=T)
> plot(fit, t.pr)
```



AT & T bonds:

```
> fit <- HoltWinters(d.att, gamma=FALSE)
> t.pr <- predict(fit, 24, prediction.interval=T)
> fit2 <- HoltWinters(diff(d.att,1), gamma=FALSE)
> t.pr2 <- predict(fit2, 24, prediction.interval=T)
> par(mfrow=c(1,2))
> plot(fit, t.pr)
> plot(fit2,t.pr2)
```



This method does not seem to yield meaningful predictions in the case of the AT&T- data. You can try to play around with the α and β values in the `HoltWinters`-function, but it will probably not get better, since the data contains hardly any structure or information.