Homework 6 Solutions

Due: Never, To Be Used as Preparation for Exam 2

Testing Nonstationarity

For $n \in \{10, 100, 500\}$ simulate 100 time series according to each of the following models, all of which assume $w_t \overset{i.i.d.}{\sim} \mathcal{N}(0, 1)$:

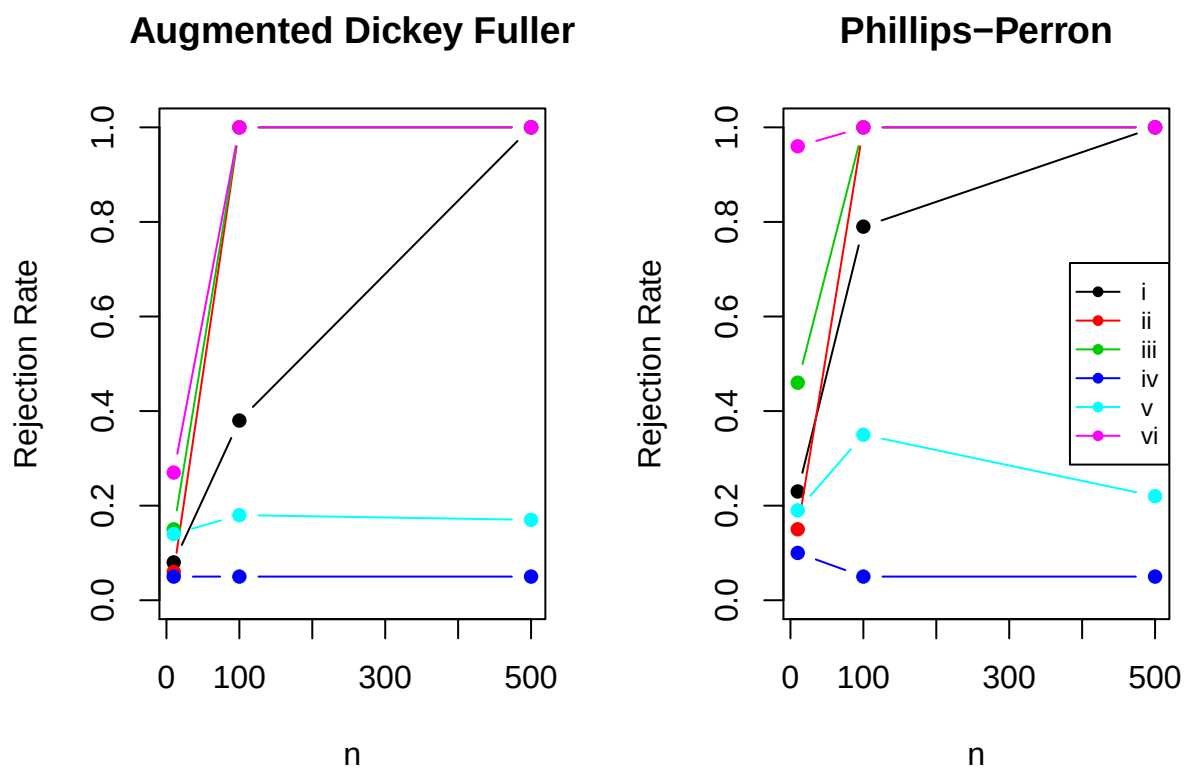
(i) $x_t = 0.5x_{t-1} + 0.25x_{t-2} + 0.125x_{t-3} + w_t$;
(ii) $x_t = -10(t/n - 1/2)^2 + w_t$;
(iii) $x_t = 10(t/n - 1/2)^2 w_t$;
(iv) $x_t = x_{t-1} + w_t$;
(v) $x_t = 0.5x_{t-1} + 0.25x_{t-2} + 0.25x_{t-3} + w_t$;
(vi) $x_t = -x_{t-1} + w_t$.

In total, you will simulate $3 \times 6 \times 100$ time series. For each time series, use the `ndiffs` function from the `forecast` library to perform an Augmented Dickey-Fuller an a Phillips-Perron test of the null hypothesis that the undifferenced time series is non-stationary rejects the null. Record whether or not the null is rejected.

(a) Plot the rejection rates as a function of n for each model.

```
ns <- c(10, 100, 500)
sim <- 100
tests <- array(dim = c(2, 6, length(ns), sim))
for (k in 1:6) {
for (i in 1:length(ns)) {
  n <- ns[i]
  for (j in 1:sim) {
    if (k == 1) {
      x <- arima.sim(n, model = list(order = c(3, 0, 0), ar = c(0.5, 0.25, 0.125)),
                     sd = 1)
    } else if (k == 2) {
      x <- -10*(((1:n) - n/2)/n)^2 + rnorm(n)
    } else if (k == 3) {
      x <- rnorm(n)*(10*(((1:n) - n/2)/n)^2)
    } else if (k == 4) {
      x <- numeric(n)
      for (l in 1:n) {
        if (l == 1) {
          x[l] <- rnorm(1)
        } else {
          x[l] <- x[l-1] + rnorm(1)
        }
      }
    } else if (k == 5) {
      x <- numeric(n)
      for (l in 1:n) {
        if (l == 1) {
          x[l] <- rnorm(1)
        } else if (l == 2) {

          x[l] <- 0.5*x[l-1] + rnorm(1)
```

```
        } else if (l == 3) {

          x[l] <- 0.25*x[l-2] + 0.5*x[l-1] + rnorm(1)

        } else {
          x[l] <- 0.25*x[l-3] + 0.25*x[l-2] + 0.5*x[l-1] + rnorm(1)
        }
      }
    } else if (k == 6) {
      x <- numeric(n)
      for (l in 1:n) {
        if (l == 1) {
          x[l] <- rnorm(1)
        } else {
          x[l] <- -x[l-1] + rnorm(1)
        }
      }
    }
    tests[1, k, i, j] <- ndiffs(x, alpha = 0.05, test = "adf", type = "level")
    tests[2, k, i, j] <- ndiffs(x, alpha = 0.05, test = "pp", type = "level")
  }
}
}
rr <- apply(tests == 0, c(1, 2, 3), mean)
par(mfrow = c(1, 2))
plot(ns, rr[1, 1, ], type = "n", ylim = c(0, 1), xlab = "n", ylab = "Rejection Rate",
     main = "Augmented Dickey Fuller")
for (i in 1:(dim(rr)[2])) {
  lines(ns, rr[1, i, ], type = "b", col = i, pch = 16)
}
plot(ns, rr[1, 1, ], type = "n", ylim = c(0, 1), xlab = "n", ylab = "Rejection Rate",
     main = "Phillips-Perron")
for (i in 1:(dim(rr)[2])) {
  lines(ns, rr[2, i, ], type = "b", col = i, pch = 16)
}
legend("right", col = 1:6, lty = rep(1, 6), pch = rep(16, 6),
       legend = c("i","ii","iii","iv","v","vi"),
       cex = 0.75)
```



(b) The probability that a test rejects the null when the alternative is true is the **power** of a test. Based on your figure in (a), describe how the power of the tests depend on n.

Under models i-iii. and vi. the power is increasing in n, under model iv. the power is decreasing in n, and under model v. the power is neither increasing nor decreasing in n.

(c) Based on your plot in (a), explain in one sentence how well the tests can detect a non-stationary $\mathbf{AR}(3)$ process if ϕ_1 is not greater than 1 and n is large.

Based on the performance of both tests under model v., both tests can detect a non-stationary $\mathbf{AR}(3)$ process if ϕ_1 is not greater than 1 and n is large reasonably well.

(d) Based on your plot in (a), explain in one sentence how well the tests can detect variance nonstationarity when n is large.

Based on the performance of both tests under model iii., neither test can detect variance nonstationarity well, even when when n is large.

(e) Based on your plot in (a), explain in one sentence how well the tests can detect a non-stationary random walk when n is large.

Based on the performance of both tests under model iv., both tests can detect can detect a non-stationary random walk very well when n is large.

(f) Based on your plot in (a), explain in one sentence how well the tests can detect a nonlinear mean function when n is large.

Based on the performance of both tests under model ii., neither test can detect a nonlinear mean function well, even when when n is large.

(g) Based on your plot in (a), explain in one sentence how the tests perform when x_t is a non-stationary $\mathbf{AR}(1)$ time series with an autoregressive polynomial root that is not equal to 1 and n is large.

Based on the performance of both tests under model vi., neither test can detect a non-stationary $\mathbf{AR}(1)$ time series with an autoregressive polynomial root that is not equal to 1 well, even when when n is large.

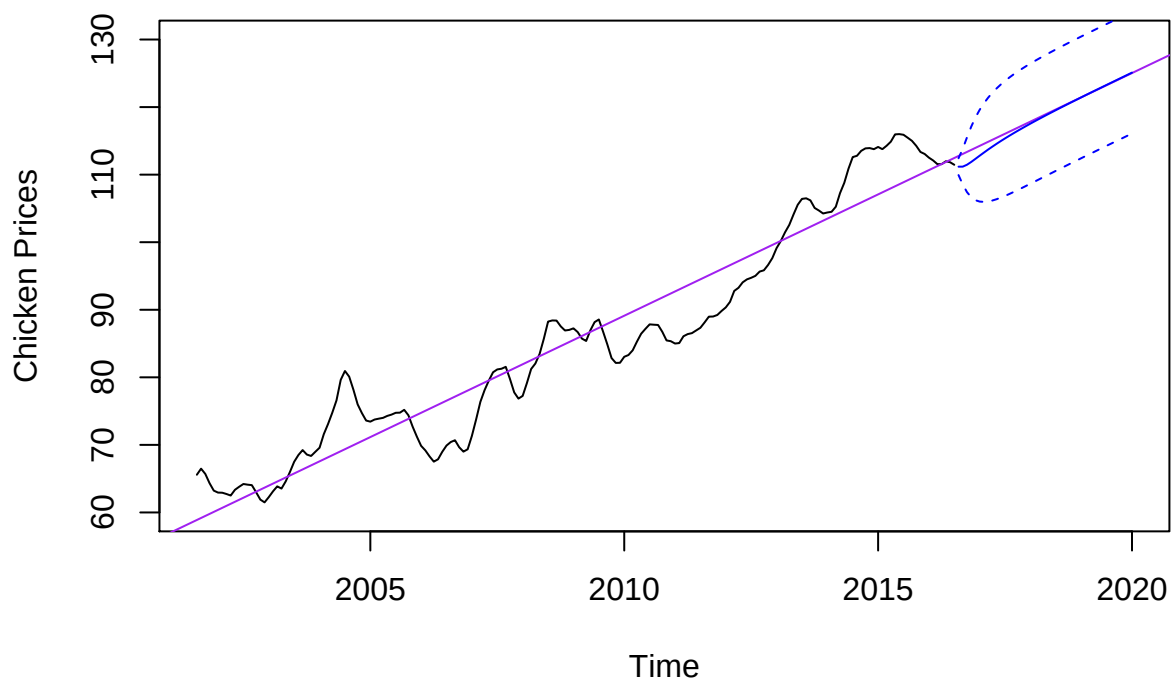
(h) Based on everything you've observed here, is it sufficient to just apply non-stationarity tests to determine whether or not an observed time series is stationary, or is it aso important to look at the data? Answer in at most one sentence.

The non-stationarity tests we have considered cannot detect all kinds of nonstationarity, so it is important to look at the data in addition to the results of nonstationarity tests.

An ARIMA Example

On Homework 4, we did forecasted future values of the `chicken` chicken data by fitting a linear time trend and applying an **ARMA**(p, q) model for the residuals. This homework will ask you to return to that. Specifically, you should have had a figure like the following:

```
library(astsa)

data(chicken)

r <- lm(chicken~time(chicken))$res
fit <- arima(r, order = c(2, 0, 3))
plot(chicken, type = "l", xlim = c(min(time(chicken)), 2020),
     ylim = c(60, 130), xlab = "Time", ylab = "Chicken Prices")
lm <- lm(chicken~time(chicken))
abline(a = lm$coef[1], b = lm$coef[2], col = "purple")
n.ahead <-12*3 + 6
pred <- predict(fit, n.ahead = n.ahead, se.fit = TRUE)
t.new <- max(time(chicken)) + 1:length(pred$pred)/12
lines(t.new, lm$coef[1] + lm$coef[2]*t.new + pred$pred, col = "blue")
lines(t.new, lm$coef[1] + lm$coef[2]*t.new + pred$pred + qnorm(0.975)*pred$se,
      col = "blue", lty = 2)
lines(t.new, lm$coef[1] + lm$coef[2]*t.new + pred$pred - qnorm(0.975)*pred$se,
      col = "blue", lty = 2)
```



(a) First - on Homework 4 we took a somewhat ad-hoc two-step approach by first regressing out the time trend and then adding it back later to obtain forecasts, ignoring any estimation error. Using the `arima` function, find the AIC minimizing values for $0 \leq p \leq 3$ and $0 \leq q \leq 3$ for the model:

$$\phi(B)(x_t - a - bt) = \theta(B) w_t,$$

where $w_t \overset{i.i.d.}{\sim} \mathcal{N}\left(0, \sigma_w^2\right)$. Do you get the same AIC minimizing values of p and q?

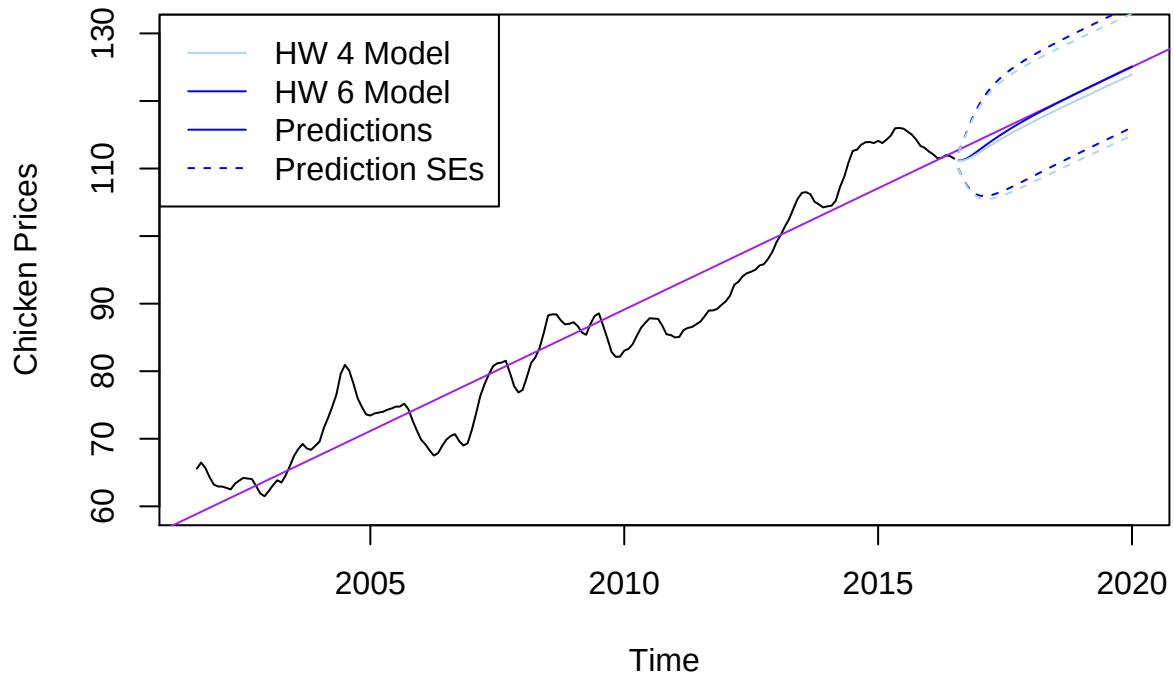
```
n <- length(r)
ps <- 0:3
qs <- 0:3
aics <- matrix(nrow = length(ps), ncol = length(qs))
for (p in ps) {
  for (q in qs) {
    if (!(p == q & p == 0)) {
      fit <- arima(chicken, order = c(p, 0, q), xreg = time(chicken))
      aics[which(p == ps), which(q == qs)] <- log(fit$sigma2) +
        (n + 2*(p + q + 1 + 1))/n
    }
  }
}
which.min <- which(aics == min(aics, na.rm = TRUE), arr.ind = TRUE)
```

On Homework 4, we selected an **ARMA**$(2, 2)$ model. The less ad-hoc approach gives the same answer, we select an **ARMA**$(2, 2)$ model with a linear time trend.

(b) Add predicted values and prediction standard errors from the model in (a) to your plot. In one sentence, comment on why they do or do not match the predicted values and prediction standard errors we obtained in Homework 4.

```
fit <- arima(r, order = c(2, 0, 2))
plot(chicken, type = "l", xlim = c(min(time(chicken)), 2020),
     ylim = c(60, 130), xlab = "Time", ylab = "Chicken Prices")
lm <- lm(chicken~time(chicken))
abline(a = lm$coef[1], b = lm$coef[2], col = "purple")
n.ahead <-12*3 + 6
pred <- predict(fit, n.ahead = n.ahead, se.fit = TRUE)
t.new <- max(time(chicken)) + 1:length(pred$pred)/12
lines(t.new, lm$coef[1] + lm$coef[2]*t.new + pred$pred, col = "blue")
lines(t.new, lm$coef[1] + lm$coef[2]*t.new + pred$pred + qnorm(0.975)*pred$se,
      col = "blue", lty = 2)
lines(t.new, lm$coef[1] + lm$coef[2]*t.new + pred$pred - qnorm(0.975)*pred$se,
      col = "blue", lty = 2)


fit <- arima(chicken, order = c(2, 0, 2), method = "ML",  xreg = (time(chicken)))
pred <- predict(fit, n.ahead = n.ahead, se = TRUE,
                newxreg = (max(time(chicken)) + 1:n.ahead/12))
lines(t.new, pred$pred, col = "lightblue")
lines(t.new,pred$pred + qnorm(0.975)*pred$se,
      col = "lightblue", lty = 2)
lines(t.new, pred$pred - qnorm(0.975)*pred$se,
      col = "lightblue", lty = 2)
legend("topleft", col = c("lightblue", "blue", "blue", "blue"),
       legend = c("HW 4 Model", "HW 6 Model",
```

```
                "Predictions", "Prediction SEs"),
       lty = c(1, 1, 1, 2))
```



The predicted means and standard errors from both models match almost exactly, which makes sense because both are just different ways of fitting the same model.

(c) Instead of modeling x_t as stationary about a linear trend, we might use an **ARIMA**(p, d, q) model. Use the `ndiffs` function from the `forecast` package to determine how many times you should difference the data. Include a linear trend and use a level-0.05 Augmented Dickey Fuller test. What does the test indicate you should choose for d?

```
ndiffs(chicken, alpha = 0.05, test = "adf", type = "trend")
```

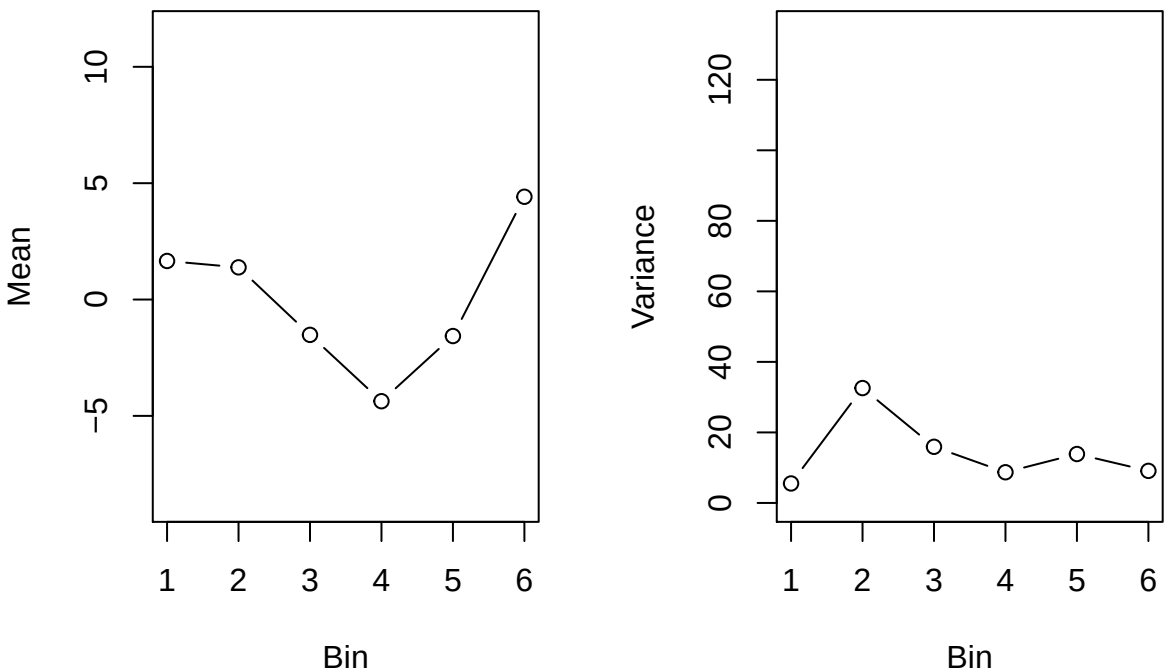
```
## [1] 0
```

The test indicates that we should choose $d = 0$.

(d) Use the `binned.mv.plot` function to make a plot of the residuals from the linear model fit, $x_t - a - \hat{b}t$, using bins of size 30. Do you think that the deviations from a the fitted time trend look stationary?

```
binned.mv.plot <- function(x, bin.size = 30) {

  bins <- floor(0:(length(x) - 1)/bin.size)
  small.bins <- which(table(bins) == min(table(bins)) & !table(bins) == max(table(bins)))
  bins[bins == small.bins - 1] <- NA

  bin.means <- aggregate(as.numeric(x), list("bins" = bins), mean)$x
  bin.vars <- aggregate(as.numeric(x), list("bins" = bins), var)$x

  par(mfrow = c(1, 2))
  plot(bin.means, type = "b", xlab = "Bin", ylab = "Mean", ylim = range(x))
  plot(bin.vars, type = "b", xlab = "Bin", ylab = "Variance", ylim = c(0, max(x^2)))
}
```

```r
binned.mv.plot(r, bin.size = 30)
```



The deviations from the fitted time trend do not quite look stationary, there does appear to be some residual time trend in the mean.

(e) Using the `arima` function, find the AIC minimizing values for $0 \leq p \leq 3$ and $0 \leq q \leq 3$ for the model:

$$\phi(B)(\nabla x_t - c) = \theta(B) w_t,$$

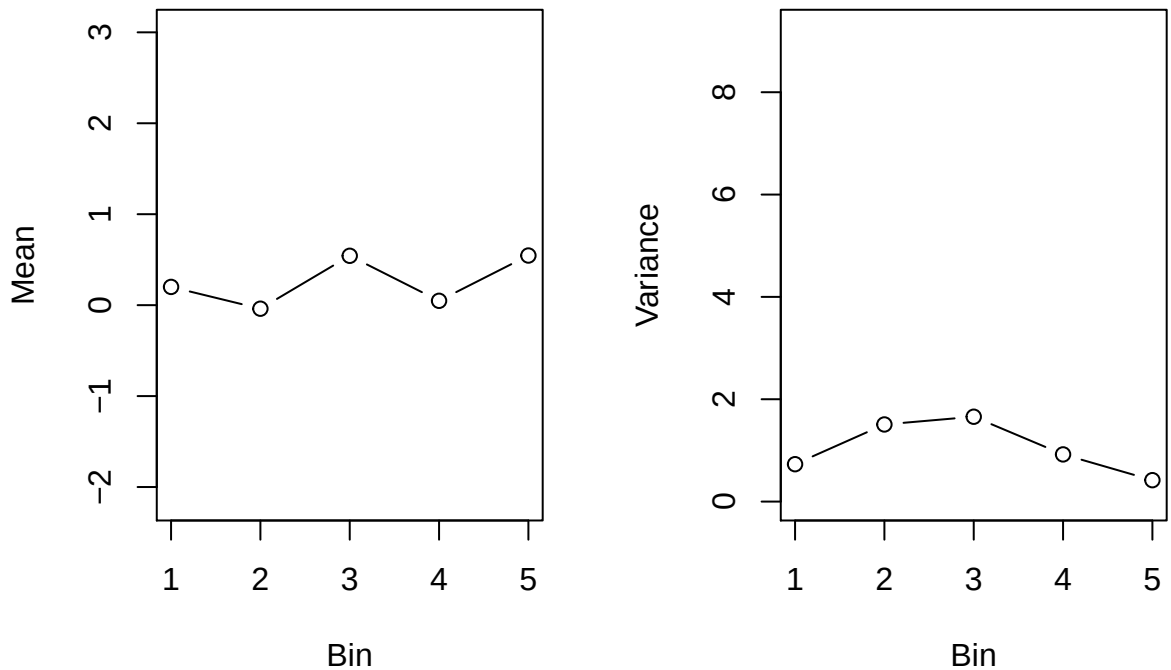
where $w_t \overset{i.i.d.}{\sim} \mathcal{N}\left(0, \sigma_w^2\right)$. To force the inclusion of the intercept c, you will need to use the `xreg` argument. Add predicted values and prediction standard errors from the AIC minimizing model to the plot from (b). Because you used the `xreg` argument to get the ARIMA fit with an intercept, you will need to use the `newxreg` argument of the `predict` function applied to the ARIMA object.

```r
fit <- arima(r, order = c(2, 0, 3))
plot(chicken, type = "l", xlim = c(min(time(chicken)), 2020),
     ylim = c(60, 130), xlab = "Time", ylab = "Chicken Prices")
lm <- lm(chicken~time(chicken))
abline(a = lm$coef[1], b = lm$coef[2], col = "purple")
n.ahead <-12*3 + 6
pred <- predict(fit, n.ahead = n.ahead, se.fit = TRUE)
t.new <- max(time(chicken)) + 1:length(pred$pred)/12
lines(t.new, lm$coef[1] + lm$coef[2]*t.new + pred$pred, col = "blue")
lines(t.new, lm$coef[1] + lm$coef[2]*t.new + pred$pred + qnorm(0.975)*pred$se,
      col = "blue", lty = 2)
lines(t.new, lm$coef[1] + lm$coef[2]*t.new + pred$pred - qnorm(0.975)*pred$se,
      col = "blue", lty = 2)


fit <- arima(chicken, order = c(2, 0, 2), method = "ML",  xreg = (time(chicken)))
pred <- predict(fit, n.ahead = n.ahead, se = TRUE, newxreg = (max(time(chicken)) + 1:n.ahead/12))
lines(t.new, pred$pred, col = "lightblue")
lines(t.new,pred$pred + qnorm(0.975)*pred$se,
```

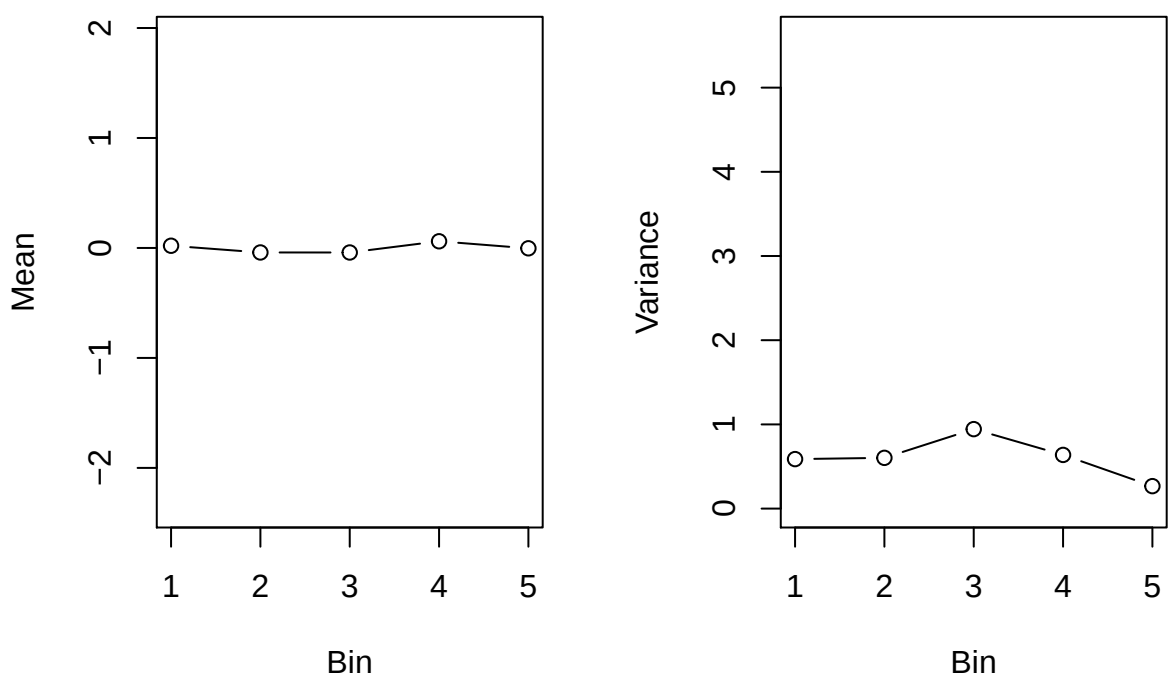
```r
      col = "lightblue", lty = 2)
lines(t.new, pred$pred - qnorm(0.975)*pred$se,
      col = "lightblue", lty = 2)

d <- 1
ps <- 0:3
qs <- 0:3
aics <- array(dim = c(nrow = length(ps), ncol = length(qs)))
for (p in ps) {
  for (q in qs) {
    if (!(p == q & p == 0)) {
      fit <- arima(chicken, order = c(p, d, q), xreg = (time(chicken))^d)
      aics[which(p == ps), which(q == qs)] <- log(fit$sigma2) +
        (n-d + 2*(p + q + 1))/(n-d)
    }
  }
}
which.min <- which(aics == min(aics, na.rm = TRUE), arr.ind = TRUE)
fit <- arima(chicken, order = c(ps[which.min[1]], 1, qs[which.min[2]]),
             method = "ML",  xreg = (time(chicken))^d)
pred <- predict(fit, n.ahead = n.ahead, se = TRUE,
                newxreg = (max(time(chicken)) + 1:n.ahead/12)^d)
lines(t.new, pred$pred, col = "red")
lines(t.new,pred$pred + qnorm(0.975)*pred$se,
      col = "red", lty = 2)
lines(t.new, pred$pred - qnorm(0.975)*pred$se,
      col = "red", lty = 2)
legend("topleft", col = c("lightblue", "blue", "red", "blue", "blue"),
       legend = c("HW 4 Model", "HW 6 a Model", "HW 6 f Model",
                  "Predictions", "Prediction SEs"),
       lty = c(1, 1, 1, 1, 2))
```

(g) Use the `binned.mv.plot` function to make a plots of the mean and variance of ∇x_t and $\nabla^2 x_t$, using bins of size 30. This will yield two sets of two plots each. Do the first or second differences look more stationary?

```r
binned.mv.plot(diff(chicken, d = 1), bin.size = 30)
```




```
binned.mv.plot(diff(chicken, d = 2), bin.size = 30)
```



The second differences of the deviations from the fitted time trend look more stationary.

(h) Using the `arima` function, find the AIC minimizing values for $0 \leq p \leq 3$ and $0 \leq q \leq 3$ for the model:

$$\phi(B)\left(\nabla^2 x_t - k\right) = \theta(B)\, w_t,$$

where $w_t \overset{i.i.d.}{\sim} \mathcal{N}\left(0, \sigma_w^2\right)$. To force the inclusion of the intercept k, you will need to use the `xreg` argument. Add predicted values and prediction standard errors from the AIC minimizing model to the plot from (b). Because you used the `xreg` argument to get the ARIMA fit with an intercept, you will need to use the `newxreg` argument of the `predict` function applied to the ARIMA object.

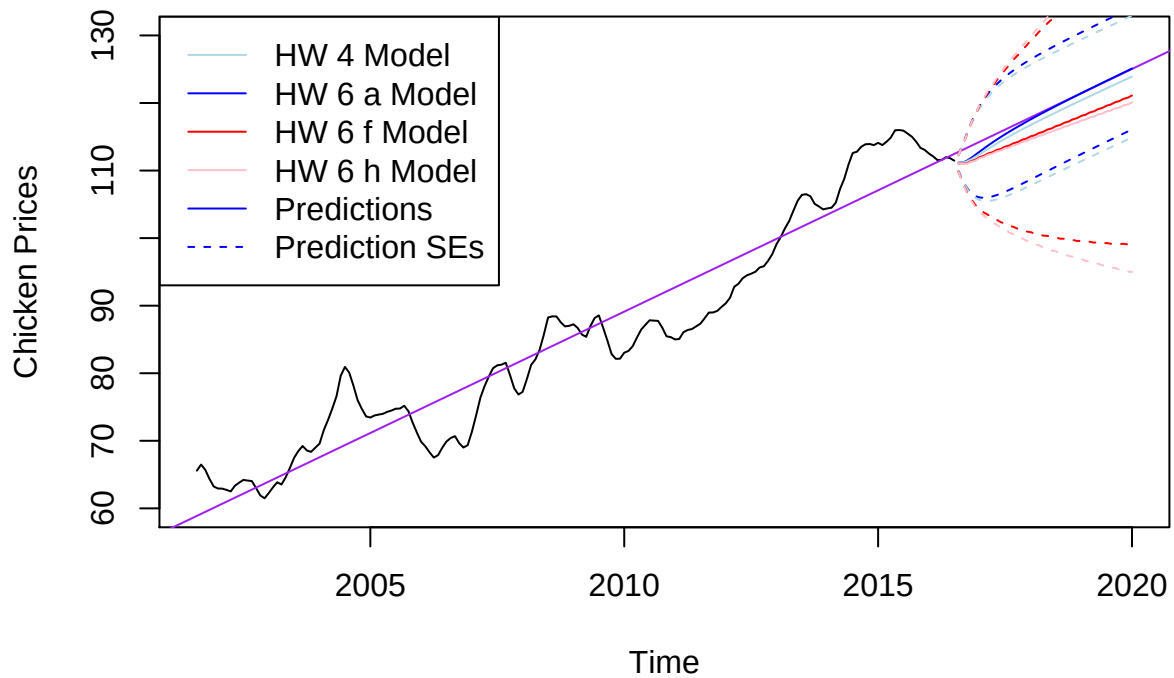
```r
fit <- arima(r, order = c(2, 0, 3))
plot(chicken, type = "l", xlim = c(min(time(chicken)), 2020),
     ylim = c(60, 130), xlab = "Time", ylab = "Chicken Prices")
lm <- lm(chicken~time(chicken))
abline(a = lm$coef[1], b = lm$coef[2], col = "purple")
n.ahead <-12*3 + 6
pred <- predict(fit, n.ahead = n.ahead, se.fit = TRUE)
t.new <- max(time(chicken)) + 1:length(pred$pred)/12
lines(t.new, lm$coef[1] + lm$coef[2]*t.new + pred$pred, col = "blue")
lines(t.new, lm$coef[1] + lm$coef[2]*t.new + pred$pred + qnorm(0.975)*pred$se,
      col = "blue", lty = 2)
lines(t.new, lm$coef[1] + lm$coef[2]*t.new + pred$pred - qnorm(0.975)*pred$se,
      col = "blue", lty = 2)

fit <- arima(chicken, order = c(2, 0, 2), method = "ML",  xreg = (time(chicken)))
pred <- predict(fit, n.ahead = n.ahead, se = TRUE,
                newxreg = (max(time(chicken)) + 1:n.ahead/12))
lines(t.new, pred$pred, col = "lightblue")
lines(t.new,pred$pred + qnorm(0.975)*pred$se,
      col = "lightblue", lty = 2)
lines(t.new, pred$pred - qnorm(0.975)*pred$se,
      col = "lightblue", lty = 2)

d <- 1
fit <- arima(chicken, order = c(3, 1, 1), method = "ML",  xreg = (time(chicken))^d)
pred <- predict(fit, n.ahead = n.ahead, se = TRUE,
                newxreg = (max(time(chicken)) + 1:n.ahead/12)^d)
lines(t.new, pred$pred, col = "red")
lines(t.new,pred$pred + qnorm(0.975)*pred$se,
      col = "red", lty = 2)
lines(t.new, pred$pred - qnorm(0.975)*pred$se,
      col = "red", lty = 2)

d <- 2
ps <- 0:3
qs <- 0:3
aics <- array(dim = c(nrow = length(ps), ncol = length(qs)))
for (p in ps) {
  for (q in qs) {
    if (!(p == q & p == 0)) {
      # Some of these models are not possible to fit without further
      # adjustment of the optimization control parameters
      # We just won't consider them
      fit <- arima(chicken, order = c(p, d, q), method = "ML", xreg = (time(chicken))^d)
      aics[which(p == ps), which(q == qs)] <- log(fit$sigma2) +
        (n - d + 2*(p + q + 1))/(n - d)
    }
  }
}
```

```
## Warning in arima(chicken, order = c(p, d, q), method = "ML", xreg =
## (time(chicken))^d): possible convergence problem: optim gave code = 1
```

```
## Warning in log(s2): NaNs produced
```

```
## Warning in log(s2): NaNs produced

## Warning in arima(chicken, order = c(p, d, q), method = "ML", xreg =
## (time(chicken))^d): possible convergence problem: optim gave code = 1
```

```
which.min <- which(aics == min(aics, na.rm = TRUE), arr.ind = TRUE)
fit <- arima(chicken, order = c(ps[which.min[1]], d, qs[which.min[2]]),
             method = "ML", xreg = (time(chicken))^d)
pred <- predict(fit, n.ahead = n.ahead, se = TRUE,
                newxreg = (max(time(chicken)) + 1:n.ahead/12)^d)
lines(t.new, pred$pred, col = "pink")
lines(t.new,pred$pred + qnorm(0.975)*pred$se,
      col = "pink", lty = 2)
lines(t.new, pred$pred - qnorm(0.975)*pred$se,
      col = "pink", lty = 2)
legend("topleft", col = c("lightblue", "blue", "red", "pink", "blue", "blue"),
       legend = c("HW 4 Model", "HW 6 a Model", "HW 6 f Model", "HW 6 h Model",
                  "Predictions", "Prediction SEs"),
       lty = c(1, 1, 1, 1, 1, 2))
```



(i) In one sentence, comment on why the predicted values and prediction standard errors change the way they do as we increase d.

The predictions become less linear, because this second difference model allows a quadratic time trend, and the standard errors get a bit larger, possibly because we have less data as d increases.

(j) Given everything you saw, explain in one sentence which predicted values and prediction standard errors you would use for forecasting for this data.

Personally, I would use the predicted values and standard errors from the second difference model because the time series looks most stationary on that scale. This is not the only possible correct answer though, it all depends on how you justify your answer!