Segunda Avaliação

Séries Temporais

Paulo Ricardo Seganfredo Campana

10 de abril de 2024

Considere a série temporal chicken, preço de aves inteiras, do pacote astsa. Este série é mensal com início em agosto de 2001 até julho de 2016, totalizando 180 observações.

1. Apresente a analise descritiva da série. Comente os resultados.

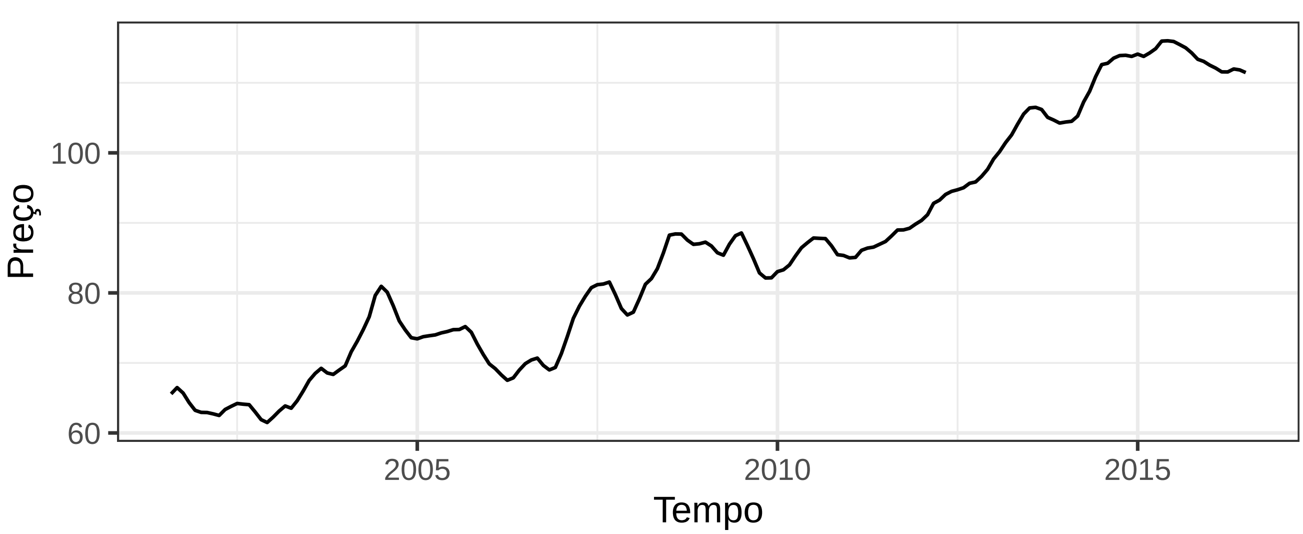
Primeiramente, a série tem frequência anual:

```r
library(forecast)
library(ggplot2)

chicken <- astsa::chicken
frequency(chicken)
## [1] 12
```

Pelo gráfico da série podemos ver que o preço mensal em centavos de dólar por libra de galinha tem tendência crescente e pouca ou nenhuma sazonalidade. O preço médio varia de 61 centavos de dólar em dezembro de 2002 até 116 centavos em junho de 2015.

```r
autoplot(chicken) +
    labs(x = "Tempo", y = "Preço")
```

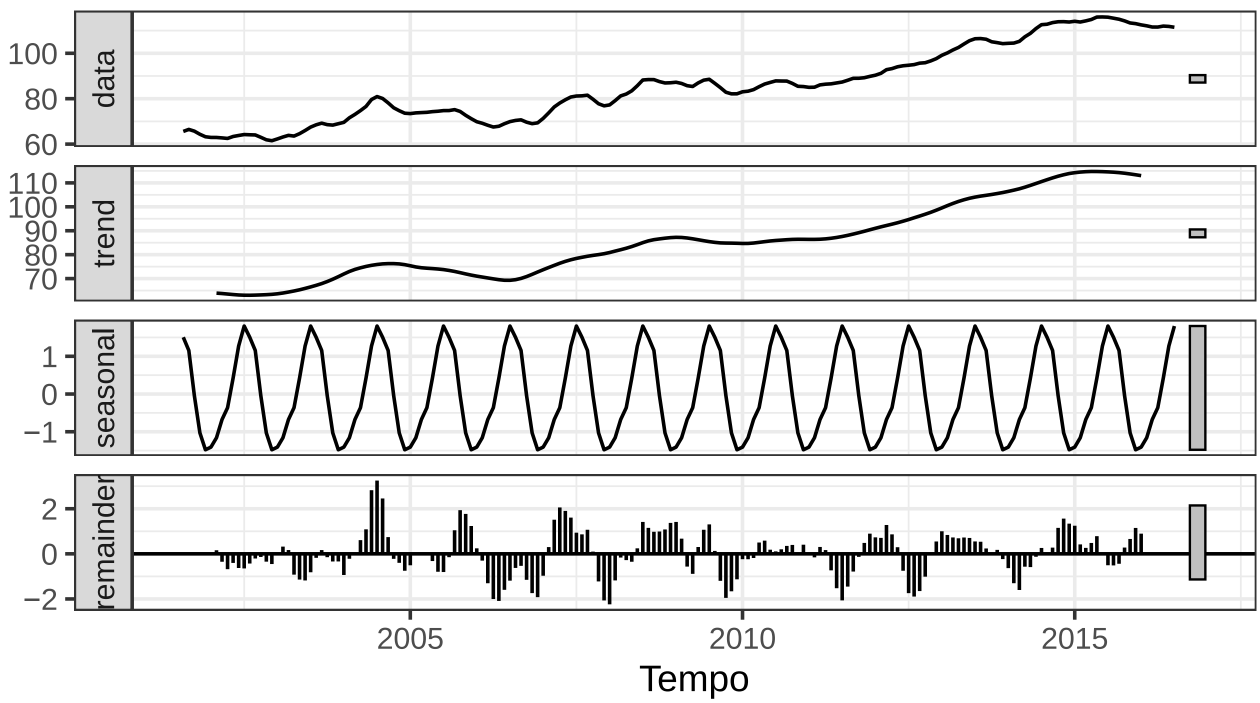


2. Explore e descreva todas a informações/características relevantes sobre a série em questão.

(a) Apresente e analise os gráficos da série e os teste de hipótese que sejam relevantes para descrever a série. Vale ressaltar aqui que todos testes de hipóteses que você considerar, é preciso descrever as hipóteses nula e alternativa, nível de significância adotado e a conclusão do teste.

Isso é sugerido também pelo gráfico da decomposição clássica da série, separando os componentes de tendência, sazonalidade e aleatoriedade, a sazonalidade detectada pela decomposição é fraca: há uma discrepância de cerca de 3 centavos entre os pontos mínimos e máximos da sazonalidade, comparado com uma média global de cerca de 80 centavos.

```
decomp <- decompose(chicken, type = "additive")
autoplot(decomp) +
    labs(x = "Tempo", title = NULL)
```



Com um nível de significância de 5%, pelo p-valor, o teste KPSS rejeita a hipótese de estacionariedade e os testes de Dickey-Fuller e Phillips–Perron não rejeitam a hipótese de não estacionariedade, assim concluímos que a série tem tendência de fato.

```
tseries::kpss.test(chicken)$p.value
## [1] 0.01
tseries::adf.test(chicken)$p.value
## [1] 0.4109
tseries::pp.test(chicken)$p.value
## [1] 0.4398
```

Surpreendentemente, todos os testes de sazonalidade do pacote `seastests` rejeitam a hipótese de não haver sazonalidade na série, assim as pequenas mudanças de mês em mês são significantes.

```
seastests::qs(chicken)$Pval
## [1] 6.268e-07
```

```r
seastests::fried(chicken)$Pval
## [1] 3.765e-12
seastests::kw(chicken)$Pval
## [1] 1.966e-10
seastests::seasdum(chicken)$Pval
## [1] 4.53e-13
seastests::welch(chicken)$Pval
## [1] 6.887e-11
```

3. Utilize a função auto.arima do pacote forecast para selecionar os ordens dos modelos ARIMA/SARIMA.

(a) Quais os modelos ARIMA/SARIMA que minimizam os critérios AIC e BIC?
(b) Os coeficientes dos parâmetros dos modelos estimados são significativos? Comente, argumente e descreva passo a passo com detalhes.

Ajustando dois modelos com e sem sazonalidade, temos que aqueles que minimizaram os critérios de seleção foram um ARIMA de ordem (2,1,1) e um SARIMA de ordem (2,1,1)(0,0,1). O modelo SARIMA obteve melhores AIC e BIC em geral.

```r
arima <- auto.arima(chicken, seasonal = FALSE)
sarima <- auto.arima(chicken, seasonal = TRUE)
arima
sarima
```

```
Series: chicken
ARIMA(2,1,1) with drift

Coefficients:
        ar1     ar2     ma1  drift
      1.353  -0.599  -0.457  0.268
s.e.  0.171   0.118   0.206  0.108

sigma^2 = 0.429:  log likelihood = -176.8
AIC=363.5   AICc=363.8   BIC=379.4
```

```
Series: chicken
ARIMA(2,1,1)(0,0,1)[12] with drift

Coefficients:
        ar1     ar2     ma1   sma1  drift
      1.293  -0.537  -0.402  0.276  0.252
s.e.  0.222   0.154   0.257  0.069  0.143

sigma^2 = 0.396:  log likelihood = -169.5
AIC=351   AICc=351.5   BIC=370.1
```

Pelo teste de significância dos coeficientes do modelo ARIMA, todos eles rejeitam a hipótese do seu coeficiente ser igual a 0, assim são todos significativos.

```r
lmtest::coeftest(arima)
##
## z test of coefficients:
##
##       Estimate Std. Error z value Pr(>|z|)
## ar1      1.353      0.171    7.92  2.4e-15 ***
## ar2     -0.599      0.118   -5.07  4.0e-07 ***
## ma1     -0.457      0.206   -2.22    0.027 *
## drift    0.268      0.108    2.49    0.013 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Porém no modelo SARIMA o coeficiente relacionado a médias móveis de ordem 1 e a constante aditiva do modelo não são significantes, com p-valores 0.1177 e 0.0777 respectivamente.

```
lmtest::coeftest(sarima)
##
## z test of coefficients:
##
##        Estimate Std. Error z value Pr(>|z|)
## ar1     1.2933     0.2220    5.82  5.7e-09 ***
## ar2    -0.5375     0.1542   -3.48  0.00049 ***
## ma1    -0.4019     0.2569   -1.56  0.11766
## sma1    0.2756     0.0692    3.98  6.9e-05 ***
## drift   0.2518     0.1428    1.76  0.07771 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Deste modo, o modelo SARIMA deve ser refeito para que não haja coeficientes não significantes. A remoção destes parâmetros não impacta muito os critérios de seleção.

```
sarima <- Arima(
    chicken,
    order = c(2, 1, 0),
    seasonal = c(0, 0, 1),
    include.drift = FALSE,
)
sarima
## Series: chicken
## ARIMA(2,1,0)(0,0,1)[12]
##
## Coefficients:
##          ar1     ar2   sma1
##        0.930  -0.259  0.287
## s.e.  0.073   0.073  0.067
##
## sigma^2 = 0.402:  log likelihood = -171.8
## AIC=351.6   AICc=351.9   BIC=364.4
lmtest::coeftest(sarima)
##
## z test of coefficients:
##
##       Estimate Std. Error z value Pr(>|z|)
## ar1    0.9302     0.0732   12.70  < 2e-16 ***
## ar2   -0.2589     0.0731   -3.54    4e-04 ***
## sma1   0.2871     0.0672    4.27  1.9e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

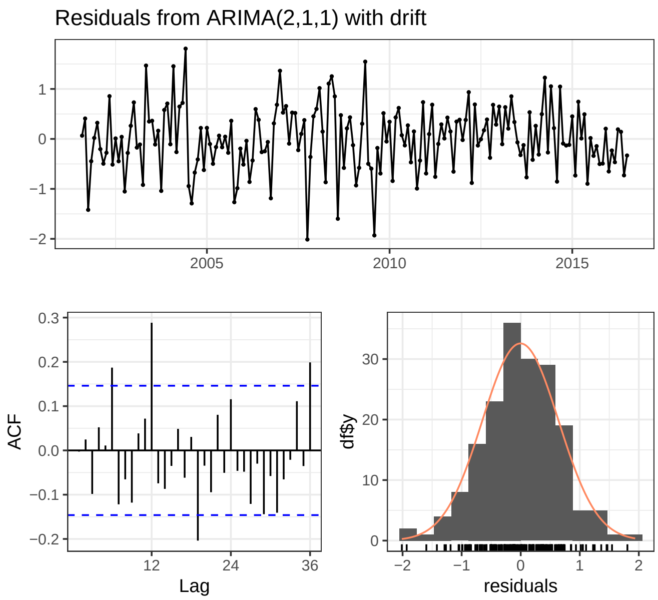
4. Cheque a qualidade do ajuste do(s) modelo(s) selecionado(s) no item anterior e explique cada os resultados obtidos para avaliar a qualidade do ajuste.

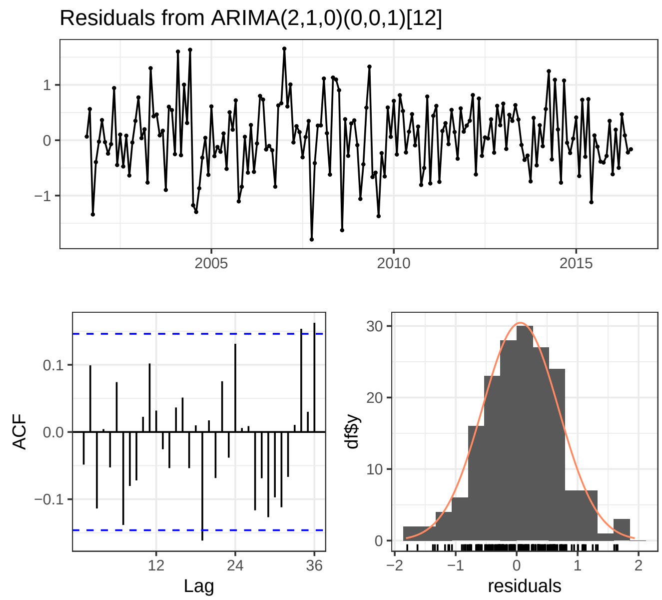
(a) Qual a conclusão do teste de Ljung e Box (apresente as hipótese)? Comente.

(b) Apresente os gráficos do diagnóstico. Interprete e comente.

Gráficamente, os resíduos parecem com ruído branco por serem independentes e normalmente distribuidos, as autocorrelações entre os resíduos só são significantes para alguns lags altos.

```
checkresiduals(arima, test = FALSE)
checkresiduals(sarima, test = FALSE)
```





Verificando a independência dos resíduos do modelo pelos testes de Box-Pierce e Box-Ljung, não rejeitamos a hipótese de que os resíduos são independentes em ambos os modelos.

```
Box.test(arima$residuals, type = "Box-Pierce")$p.value
## [1] 0.9715
Box.test(arima$residuals, type = "Ljung-Box")$p.value
## [1] 0.9713
Box.test(sarima$residuals, type = "Box-Pierce")$p.value
## [1] 0.5154
Box.test(sarima$residuals, type = "Ljung-Box")$p.value
## [1] 0.5119
```

Também não rejeitamos a hipótese de normalidade dos resíduos, ambos os modelos são válidos.

```
shapiro.test(arima$residuals)$p.value
## [1] 0.6741
nortest::lillie.test(arima$residuals)$p.value
## [1] 0.6683
shapiro.test(sarima$residuals)$p.value
## [1] 0.8816
nortest::lillie.test(sarima$residuals)$p.value
## [1] 0.9531
```

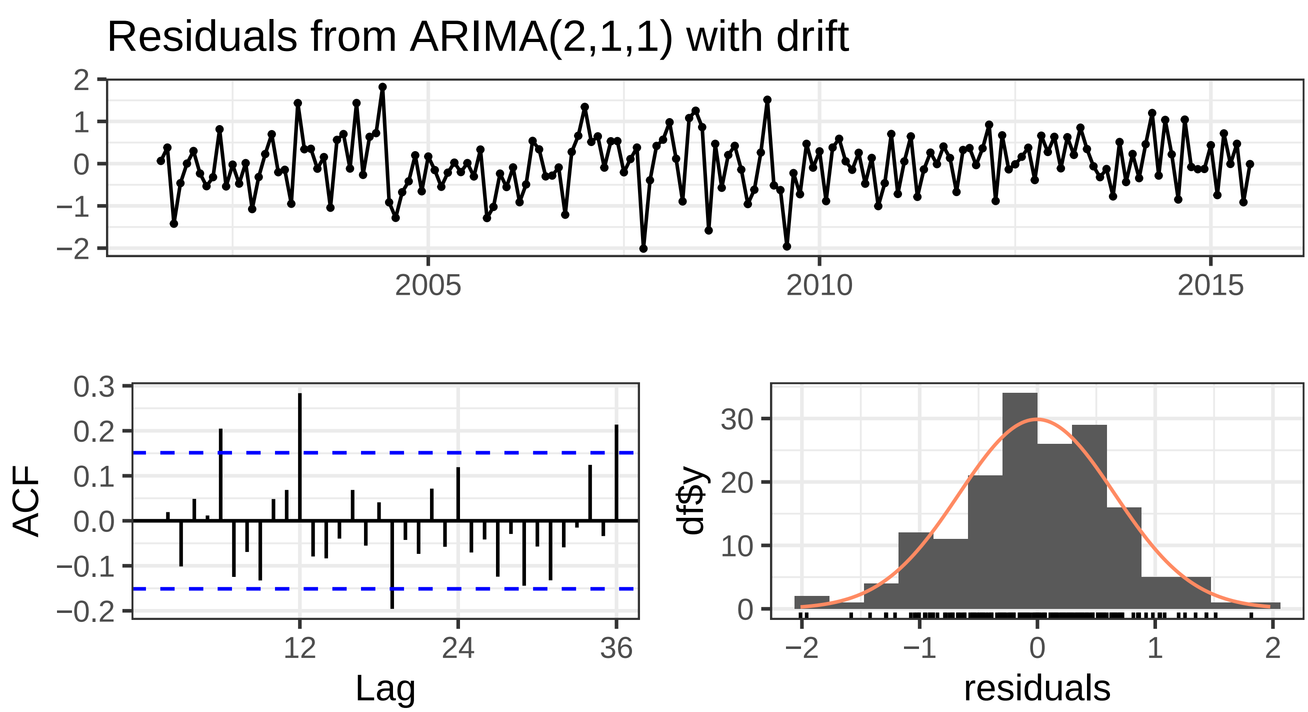
5. Exclua as 12 ultimas observações da série temporal e faça previsão considerando a série truncada (`chicken.truc`).

```r
n <- length(chicken)
chicken.trunc <- ts(chicken[1:(n - 12)], freq = 12, start = c(2001, 8))
```

(a) Considere a série (`chicken.truc`) e apresente os valores previstos 12 passos a frente considerando os modelos ARIMA/SARIMA. Importante: Note que é preciso buscar novos ajustes/checar qualidade do ajuste para chicken.truc, e depois fazer as previsões.

A ordem estimada do modelo ARIMA, a significância de seus coeficientes e as hipóteses de independência e normalidade dos resíduos continuam as mesmas.

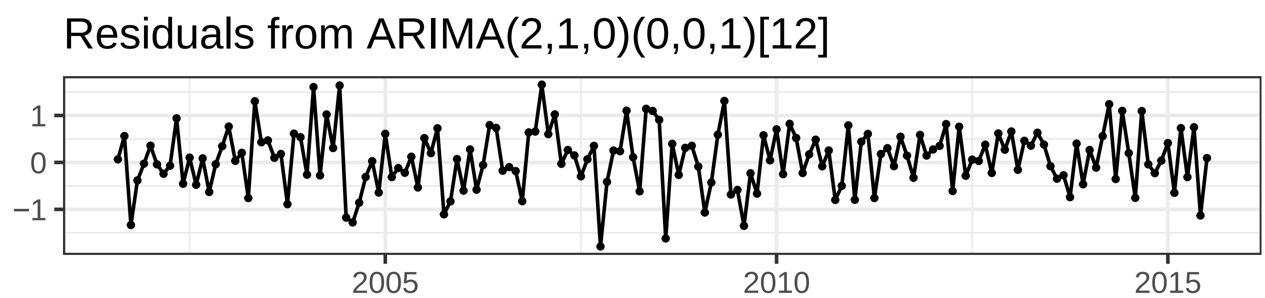
```r
arima.trunc <- auto.arima(chicken.trunc, seasonal = FALSE)
lmtest::coeftest(arima.trunc)
##
## z test of coefficients:
##
##        Estimate Std. Error z value Pr(>|z|)
## ar1      1.379      0.162    8.53  < 2e-16 ***
## ar2     -0.622      0.110   -5.66  1.5e-08 ***
## ma1     -0.493      0.198   -2.49   0.0128 *
## drift    0.313      0.107    2.93   0.0034 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
checkresiduals(arima.trunc, test = FALSE)
```

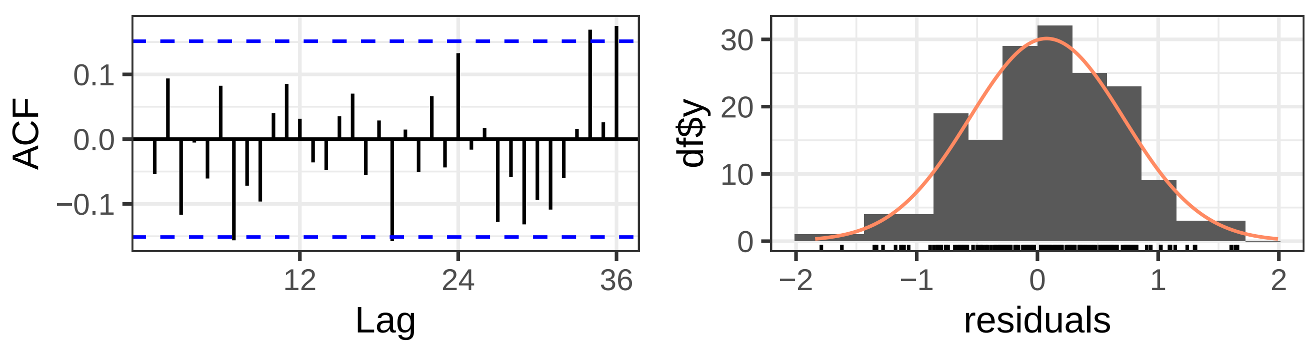


O mesmo acontece para o modelo SARIMA, as ordens estimadas são as mesmas e possui coeficientes não significantes que foram removidos da mesma forma.

```r
sarima.trunc <- Arima(
    chicken.trunc,
    order = c(2, 1, 0),
```

```
    seasonal = c(0, 0, 1),
    include.drift = FALSE,
)
lmtest::coeftest(sarima.trunc)
##
## z test of coefficients:
##
##      Estimate Std. Error z value Pr(>|z|)
## ar1    0.9339     0.0755   12.36  < 2e-16 ***
## ar2   -0.2675     0.0754   -3.55  0.00038 ***
## sma1   0.2960     0.0709    4.17    3e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
checkresiduals(sarima.trunc, test = FALSE)
```





Os valores previstos dos modelos ARIMA e SARIMA para o ano retirado da série truncada são:

```
arima.predict <- forecast(arima.trunc, h = 12)$mean
sarima.predict <- forecast(sarima.trunc, h = 12)$mean
arima.predict
##        Jan   Feb   Mar   Apr   May   Jun   Jul   Aug   Sep   Oct   Nov   Dec
## 2015                                           115.8 115.8 116.0 116.3 116.7
## 2016 117.1 117.5 117.9 118.2 118.5 118.8 119.1
sarima.predict
##        Jan   Feb   Mar   Apr   May   Jun   Jul   Aug   Sep   Oct   Nov   Dec
## 2015                                           115.6 115.6 115.7 115.8 115.8
## 2016 115.9 115.8 115.9 116.0 116.2 116.0 115.9
```

(b) Considere a série (`chicken.truc`) e apresente os valores previstos 12 passos a frente considerando o algoritmo de alisamento exponencial.

```r
holt <- hw(chicken.trunc)
holt.predict <- forecast(holt, h = 12)$mean
holt.predict
##        Jan   Feb   Mar   Apr   May   Jun   Jul   Aug   Sep   Oct   Nov   Dec
## 2015                                           115.7 115.5 114.3 113.3 113.0
## 2016 113.4 114.0 114.6 114.9 115.9 116.9 117.6
```

(c) De acordo com as características da série, utilize os métodos simples de previsão considerando a série `chicken.truc` e apresente os valores previstos 12 passos a frente.

O método de média global não é valido pois a série possui tendência e o Naïve não é adequato pois a série truncada acaba no auge da sazonalidade, assim as previsões serão mais altas que o normal.

```r
snaive <- snaive(chicken.trunc, h = 12)
snaive.predict <- forecast(snaive, h = 12)$mean
snaive.predict
##        Jan   Feb   Mar   Apr   May   Jun   Jul   Aug   Sep   Oct   Nov   Dec
## 2015                                           112.8 113.5 113.9 113.9 113.8
## 2016 114.1 113.8 114.3 114.9 116.0 116.0 115.9

sma <- smooth::sma(chicken.trunc, order = 6, h = 12)
sma.predict <- forecast(sma, h = 12)$mean
sma.predict
##        Jan   Feb   Mar   Apr   May   Jun   Jul   Aug   Sep   Oct   Nov   Dec
## 2015                                           115.1 115.4 115.5 115.6 115.6
## 2016 115.5 115.5 115.5 115.5 115.6 115.5 115.5
```

(d) Calcule e compare os erro quadrático médio e o erro médio percentual (ver função `accuracy`) das previsões obtidas via modelos ARIMA, alisamento exponencial e métodos simples de previsão (utilizados no item anterior). O que você pode concluir?

```r
verdadeiro <- ts(chicken[(n - 11):n], freq = 12, start = c(2015, 8))
```

Para previsão de 12 passos a frente, os modelos ARIMA e SARIMA tiveram pior raiz do erro quadrático médio e média do erro absoluto percentual, o método de alisamento exponencial de Holt-Winters e Naïve sazonal obtiveram os menores erros.

O método de Naïve sazonal obteve a menor raiz do erro quadrático médio.

```r
yardstick::rmse_vec(verdadeiro, arima.predict)
## [1] 5.076
yardstick::rmse_vec(verdadeiro, sarima.predict)
## [1] 3.34
yardstick::rmse_vec(verdadeiro, holt.predict)
## [1] 2.935
yardstick::rmse_vec(verdadeiro, snaive.predict)
## [1] 2.691
yardstick::rmse_vec(verdadeiro, sma.predict)
## [1] 3.007
```

O método de Holt-Winters obteve a menor média do erro absoluto percentual.

```r
yardstick::mape_vec(verdadeiro, arima.predict)
## [1] 3.978
yardstick::mape_vec(verdadeiro, sarima.predict)
## [1] 2.671
yardstick::mape_vec(verdadeiro, holt.predict)
## [1] 1.869
yardstick::mape_vec(verdadeiro, snaive.predict)
## [1] 2.052
yardstick::mape_vec(verdadeiro, sma.predict)
## [1] 2.41
```