

02_loess-series-tiempo

Ita Santiago

```
knitr::opts_chunk$set(echo = TRUE, error=TRUE)

# cargamos paquetes
library(tidyverse)

## -- Attaching packages ----

## v ggplot2 3.3.2     v purrr    0.3.4
## v tibble   3.0.3     v dplyr    1.0.1
## v tidyverse 1.1.1     v stringr  1.4.0
## v readr    1.3.1     vforcats  0.5.0

## -- Conflicts --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()   masks stats::lag()

library(dplyr)
```

Series de tiempo

Consideraremos la ventas semanales de un producto a lo largo de 5 años, transformaremos la variable de ventas utilizando el logaritmo.

1. Describe que observas en la gráfica.

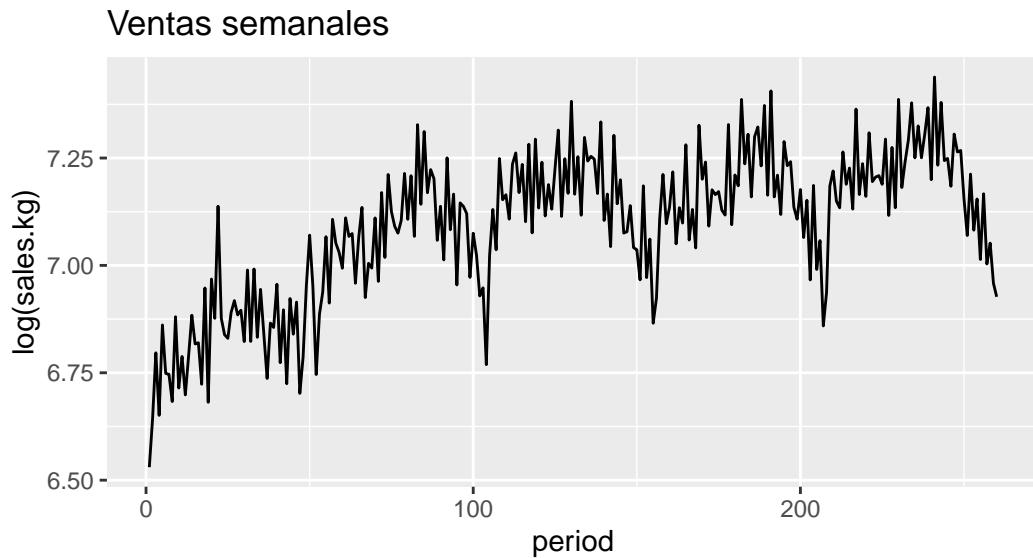
```
ventas <- read_csv("ventas_semanal.csv")

## Parsed with column specification:
## cols(
##   period = col_double(),
##   sales.kg = col_double()
## )

head(ventas)

## # A tibble: 6 x 2
##   period sales.kg
##   <dbl>    <dbl>
## 1 1       686.
## 2 2       768.
## 3 3       895.
## 4 4       774.
## 5 5       955.
## 6 6       853.
```

```
ggplot(ventas, aes(x = period, y = log(sales.kg))) +
  geom_line(size = 0.5) +
  ggtitle("Ventas semanales")
```



Se puede observar una tendencia positiva a lo largo del tiempo, esto es que, conforme pasa el tiempo las ventas incrementan. También se puede observar que cada cierto tiempo (cada 50 periodos aproximadamente) hay decremetos, estos patrones podriamos pensar son estacionales.

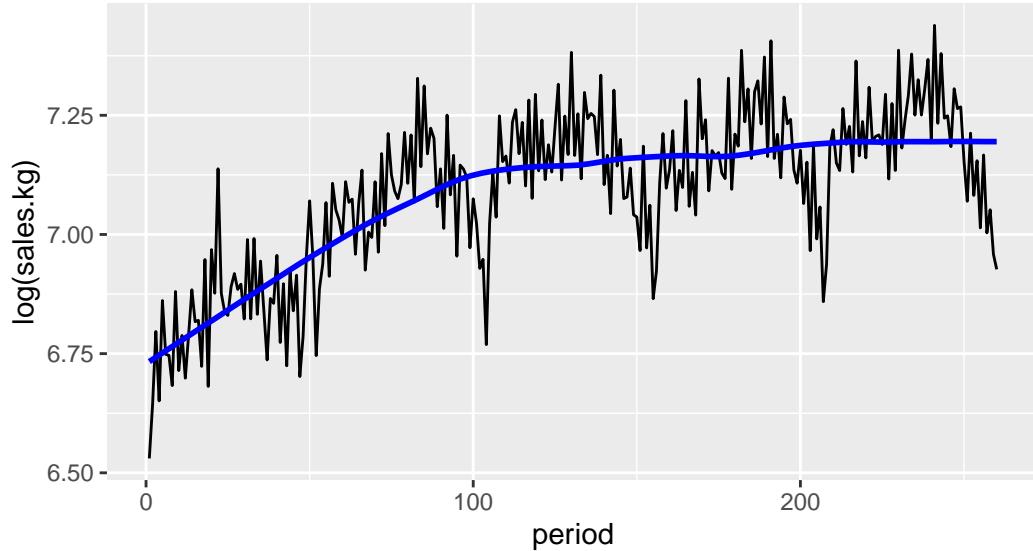
Por lo tanto nuestra serie de tiempo tiene tendencia y estacionalidad

Intentaremos usar suavizamiento para capturar los distintos tipos de variación que observamos en la serie.

2. Utiliza un suavizador *loess* para capturar la tendencia de la serie.

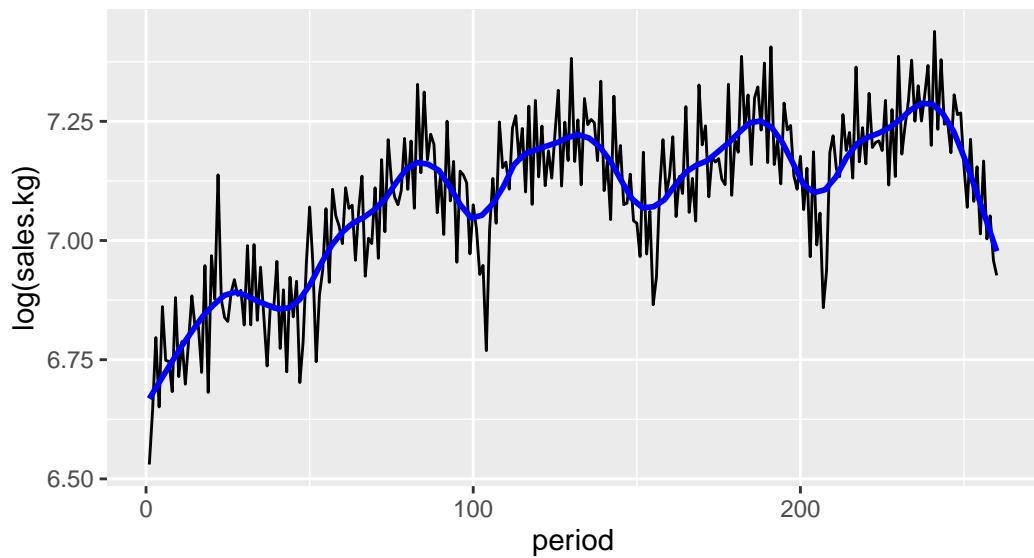
```
ggplot(ventas, aes(x = period, y = log(sales.kg))) +
  geom_line(size = 0.5) +
  geom_smooth(method = "loess", span = 0.5, method.args = list(degree = 1),
             se = FALSE, size = 1, color = "blue")
```

`geom_smooth()` using formula 'y ~ x'



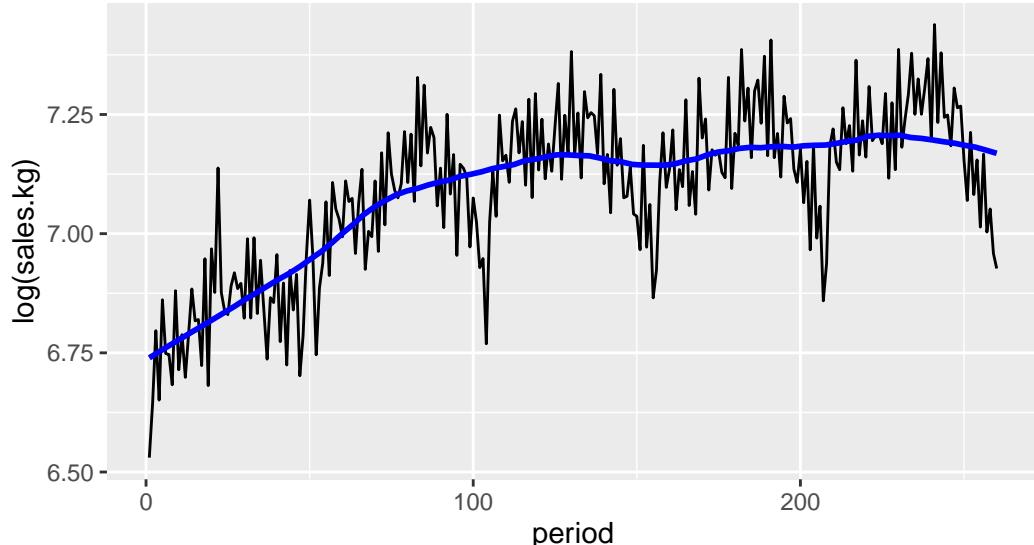
Se puede apreciar la tendencia de la serie, sin embargo ya no se observa la estacionalidad.

```
ggplot(ventas, aes(x = period, y = log(sales.kg))) +  
  geom_line(size = 0.5) +  
  geom_smooth(method = "loess", span = 0.1, method.args = list(degree = 1),  
             se = FALSE, size = 1, color = "blue")  
  
## `geom_smooth()` using formula 'y ~ x'
```



En esta otra podemos observar que la función loess ajusta también la parte estacional.

```
ggplot(ventas, aes(x = period, y = log(sales.kg))) +  
  geom_line(size = 0.5) +  
  geom_smooth(method = "loess", span = 0.3, method.args = list(degree = 1),  
             se = FALSE, size = 1, color = "blue")  
  
## `geom_smooth()` using formula 'y ~ x'
```



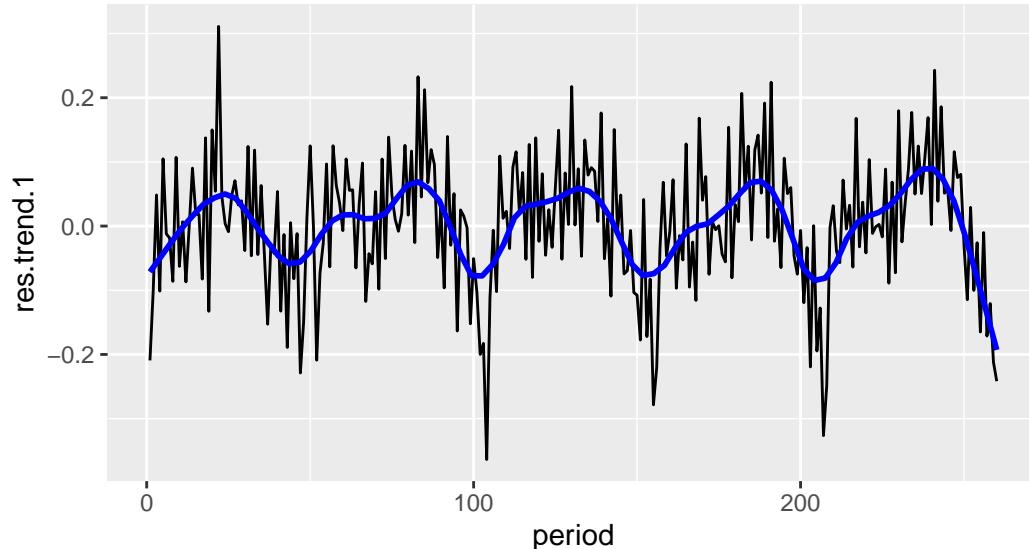
```
fit_trend <- loess(log(sales.kg) ~ period, ventas, span = 0.3, degree = 1)
```

Por ultimo en esta se puede observar perfectamente la tendencia sin perder un poco la estacionalidad de

nuestra serie de tiempo.

3. Ahora calcula los residuales de este ajuste y describelos mediante un suavizamiento más fino. Verifica que se ha estimado la mayor parte de la tendencia, e intenta capturar la variación estacional de los residuales.

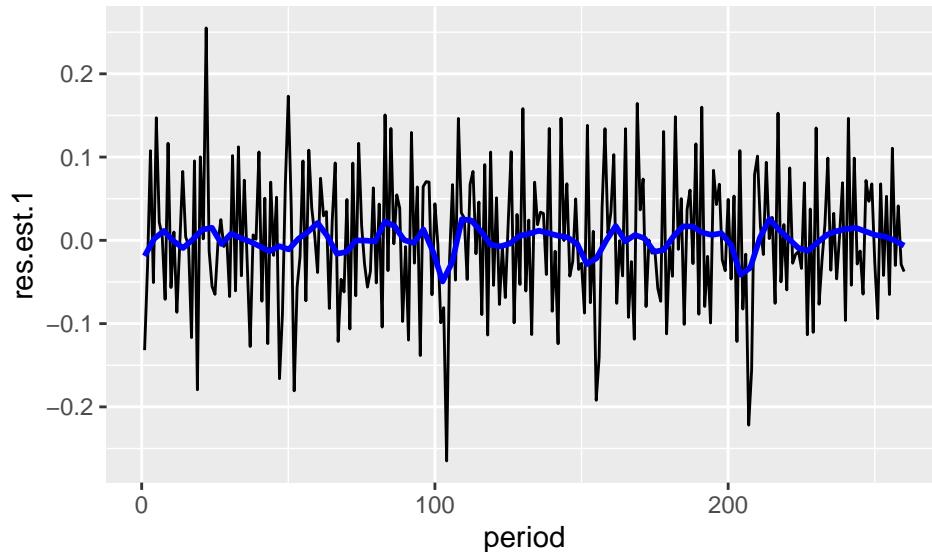
```
ventas$trend.1 <- fit_trend$fitted  
ventas$res.trend.1 <- fit_trend$residuals  
  
ggplot(ventas, aes(x = period, y = res.trend.1)) +  
  geom_line(size = 0.5) +  
  geom_smooth(method = "loess", span = 0.1, method.args = list(degree = 1),  
             se = FALSE, size = 1, color = "blue")  
  
## `geom_smooth()` using formula 'y ~ x'
```



```
fit_season <- loess(res.trend.1 ~ period, ventas, span = 0.08, degree = 1)  
ventas$est.1 <- fit_season$fitted  
ventas$res.est.1 <- fit_season$residuals
```

4. Grafica los residuales obtenidos después de ajustar el componente estacional para estudiar la componente de mayor frecuencia.

```
ggplot(ventas, aes(x = period, y = res.est.1)) +  
  geom_line(size = 0.5) +  
  geom_smooth(method = "loess", span = 0.06, method.args = list(degree = 1),  
             se = FALSE, size = 1, color = "blue")  
  
## `geom_smooth()` using formula 'y ~ x'
```



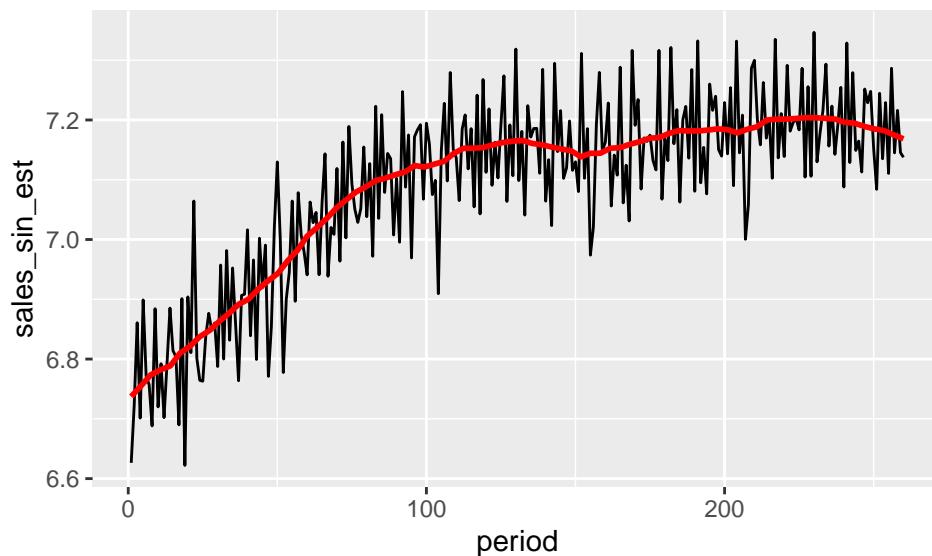
```
fit_season_high <- loess(res.est.1 ~ period, ventas, span = 0.06, degree = 1)
ventas$est.2 <- fit_season_high$fitted
ventas$res.est.2 <- fit_season_high$residuals
```

5. **Extra opcional.** Ahora que tenemos nuestra primera estimación de cada una de las componentes, podemos regresar a hacer una mejor estimación de la tendencia. La ventaja de volver es que ahora podemos suavizar más sin que en nuestra muestra compita tanto la variación estacional. Por tanto puedes suavizar un poco menos.

```
ventas$sales_sin_est <- log(ventas$sales.kg) -
  fit_season$fitted - fit_season_high$fitted

ggplot(ventas, aes(x = period, y = sales_sin_est)) +
  geom_line(size = 0.5) +
  geom_smooth(method = "loess", span = 0.08, method.args = list(degree = 1),
             se = FALSE, size = 1, color = "red")

## `geom_smooth()` using formula 'y ~ x'
```



```

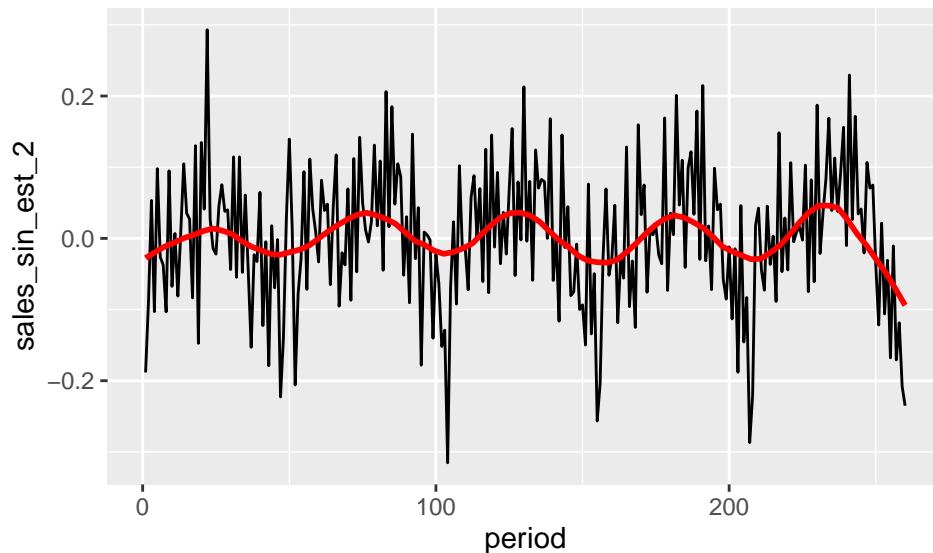
fit_trend_2 <- loess(sales_sin_est ~ period, ventas, span = 0.08, degree = 1)
ventas$trend.2 <- fit_trend_2$fitted
ventas$res.trend.2 <- log(ventas$sales.kg) - ventas$trend.2

ventas$sales_sin_est_2 <- log(ventas$sales.kg) -
  fit_trend_2$fitted - fit_season_high$fitted

ggplot(ventas, aes(x = period, y = sales_sin_est_2)) +
  geom_line(size = 0.5) +
  geom_smooth(method = "loess", span = 0.2, method.args = list(degree = 1),
  se = FALSE, size = 1, color = "red")

## `geom_smooth()` using formula 'y ~ x'

```



```

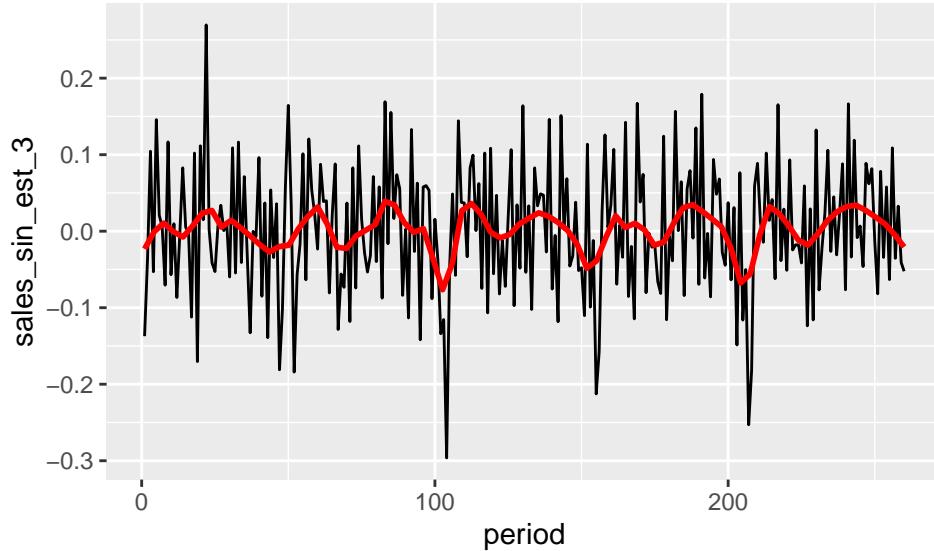
fit_season_2 <- loess(sales_sin_est_2 ~ period, ventas, span = 0.1, degree = 1)
ventas$est.2_2 <- fit_season_2$fitted
ventas$res.est.2_2 <- fit_season_2$residuals

ventas$sales_sin_est_3 <- log(ventas$sales.kg) -
  fit_trend_2$fitted - fit_season_2$fitted

ggplot(ventas, aes(x = period, y = sales_sin_est_3)) +
  geom_line(size = 0.5) +
  geom_smooth(method = "loess", span = 0.06, method.args = list(degree = 1),
  se = FALSE, size = 1, color = "red")

## `geom_smooth()` using formula 'y ~ x'

```



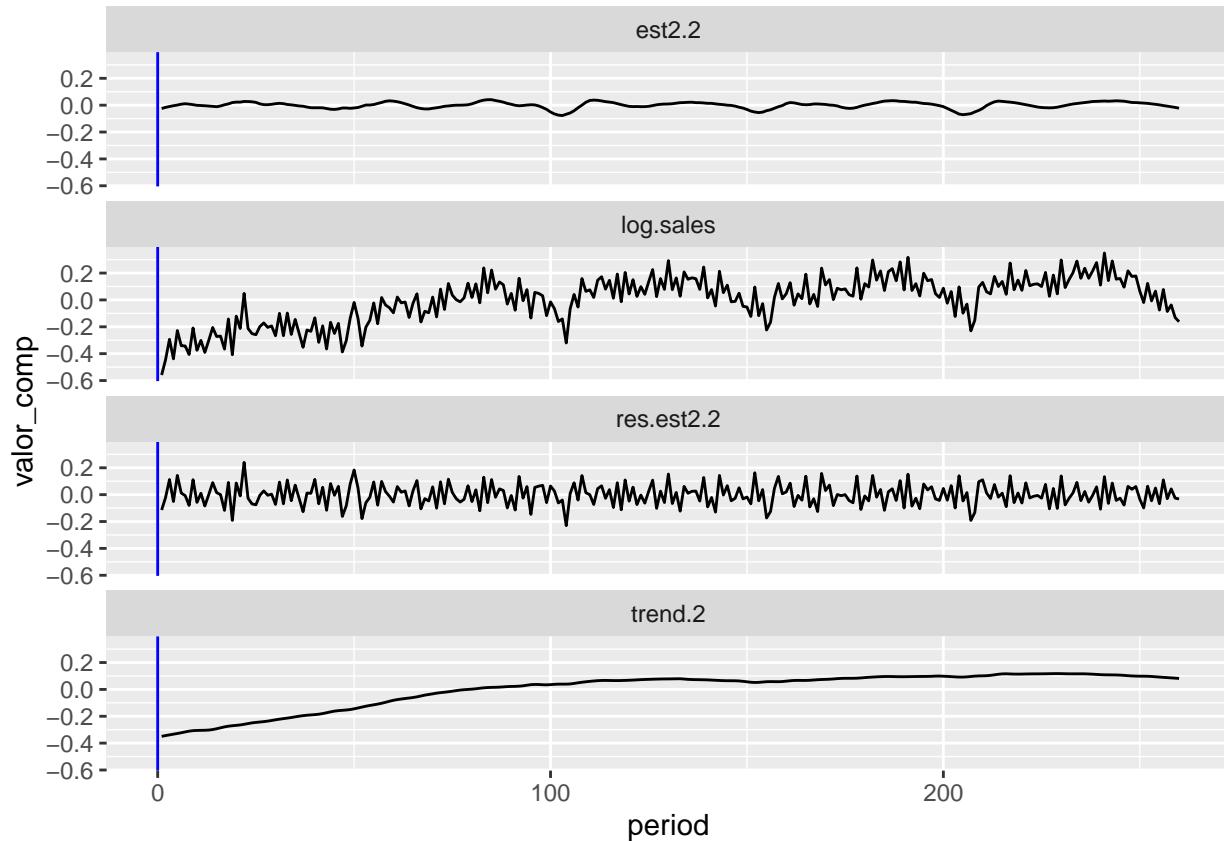
```
fit_season_3 <- loess(sales_sin_est_3 ~ period, ventas, span = 0.06, degree = 1)
ventas$est2.2 <- fit_season_3$fitted
ventas$res.est2.2 <- fit_season_3$residuals
```

6. Visualiza el ajuste, genera una gráfica de páneles, en cada uno muestra una componente de la serie de tiempo y los residuales.

```
ventas$log.sales <- log(ventas$sales.kg)
ventas_bis <- select(ventas, period, trend.2, est2.2, res.est2.2, log.sales)
ventas_bis <- gather(ventas_bis, componente, valor, -period)

## Warning: attributes are not identical across measure variables;
## they will be dropped
ventas_bis <- ventas_bis %>%
  group_by(componente) %>%
  mutate(valor_comp = valor - mean(valor))

ggplot(ventas_bis, aes(x = period, y = valor_comp)) +
  geom_vline(xintercept = 0, color = "blue") +
  geom_line(size = 0.5) +
  facet_wrap(~ componente, ncol = 1)
```



7. Genera una gráfica de cuantiles para los residuales.

```
ventas_normales <- arrange(ventas, res.est2.2)
renglones <- nrow(ventas)
ventas_normales$dist_normal <- qnorm((1:renglones - 0.5) / renglones)

ggplot(ventas_normales, aes(x = dist_normal, y = res.est2.2)) +
  geom_point(size = 1.5) +
  geom_smooth(method = "lm")

## `geom_smooth()` using formula 'y ~ x'
```

