STAT 436 / 536 - Lecture 10: Key

October 5, 2018

Regression with Seasonality

- We have seen the presence of seasonality in several of the datasets we have considered. This section focuses on regression techniques for seasonality.

Indicator variables for seasonality

- One approach for seasonality is to fit indicator indicator models for the seasonal components.

- Assume we are looking at monthly data (e.g. airline passengers), then we will need to include 12 indicator variables to account for each month.

- Write out a linear model for seasonality with no trend.

$$x_t = s_t + z_t$$

where $s_t = \beta_i$ *when* t *falls in the* i^{th} *month and* z_t *is the residual error. Otherwise this model can be written as:*

$$x_t = \begin{cases} \beta_1 + z_t \\ \beta_2 + z_t \\ \vdots \\ \beta_{12} + z_t \end{cases}$$

- This model can be fit in R (for the airline passengers) using the following commands

```
data("AirPassengers")
AirPassengers.df <- data.frame(season = as.factor(as.numeric(cycle(AirPassengers))),
                               count = as.numeric(AirPassengers))
lm(count ~ season - 1, data = AirPassengers.df)
```

```
##
## Call:
## lm(formula = count ~ season - 1, data = AirPassengers.df)
##
## Coefficients:
##  season1   season2   season3   season4   season5   season6   season7
##    241.8     235.0     270.2     267.1     271.8     311.7     351.3
##  season8   season9  season10  season11  season12
##    351.1     302.4     266.6     232.8     261.8
```

- Now consider the same framework with a trend too

- *We can formulate this in an additive framework as*

$$x_t = m_t + s_t + z_t$$

where $m_t = \alpha t$ *and* $s_t = \beta_i$ *when* t *falls in the* i^{th} *month and* z_t *is the residual error. Otherwise this model can be written as:*

$$x_t = \begin{cases} \alpha t + \beta_1 + z_t \\ \alpha t + \beta_2 + z_t \\ \vdots \\ \alpha t + \beta_{12} + z_t \end{cases}$$

- This model can be fit in R (for the airline passengers) using the following commands

```
data("AirPassengers")
AirPassengers.df <- data.frame(season = as.factor(as.numeric(cycle(AirPassengers))),
                               count = as.numeric(AirPassengers), time = 1:length(AirPassengers))
lm(count ~ time + season - 1, data = AirPassengers.df)
```

```
##
## Call:
## lm(formula = count ~ time + season - 1, data = AirPassengers.df)
##
## Coefficients:
##     time   season1   season2   season3   season4   season5   season6
##     2.66     63.51     54.10     86.60     80.86     82.95    120.12
##  season7   season8   season9  season10  season11  season12
##   157.13    154.22    102.89     64.40     27.99     54.33
```

- *Or We can formulate this in an multiplicative framework as*

$$x_t = m_t \times s_t \times z_t$$

where $m_t = \alpha t$ *and* $s_t = \beta_i$ *when* t *falls in the* i^{th} *month and* z_t *is the residual error. Otherwise this model can be written as:*

$$x_t = \begin{cases} \alpha t \times \beta_1 + z_t \\ \alpha t \times \beta_2 + z_t \\ \vdots \\ \alpha t \times \beta_{12} + z_t \end{cases}$$

- We will discuss fitting this model shortly, but note that it can be acheived with a log transform.

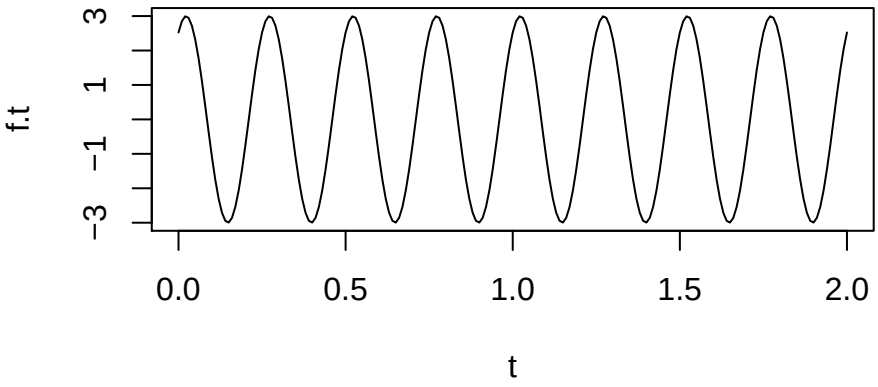
Harmonic seasonal models

- The indicator variables cause a stair-step like seasonal pattern where each season has a step function or a separate intercept. An alternative for a smooth seasonal pattern is to use sine and cosine functions.

- A sine wave with frequency f, phase shift ϕ, and amplitude A can be written as

$$A\sin(2\pi ft + \phi)$$

```
A <- 3
f <- 4
phi <- 1
t <- seq(0,2, by=.01)
f.t <- A * sin(2 * pi * f * t + phi )
plot(t, f.t, type='l')
```



- The representation above is not linear, as ϕ is in the sine function; however, this can be reexpressed as

$$A\sin(2\pi ft + \phi) = \alpha_s \sin(2\pi ft) + \alpha_c \cos(2\pi ft),$$

where $\alpha_s = A\cos(\phi)$ and $\alpha_c = A\sin(\phi)$. This representation is linear in the parameters α_s and α_c and standard techniques (OLS) can be used to estimate these parameters.

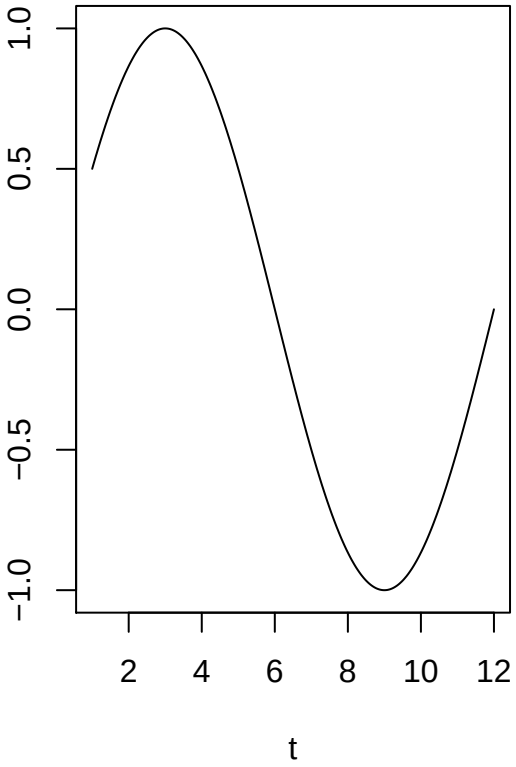
- Using this framework for harmonics with a time series $\{x_t\}$ with s seasons results in $[s/2]$ possible cycles, where [] retains the integer.

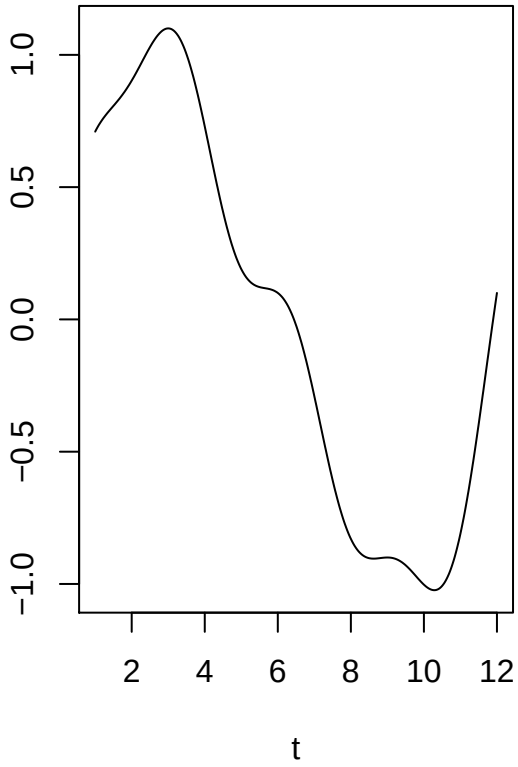
- This model can be written as

$$x_t = m_t + \sum_{i=1}^{[s/2]} (s_i \sin(2\pi it/s) + c_i \cos(2\pi it/s)) + z_t$$

where s_i and c_i are unknown parameters. This representation can *distort* the sine wave to make it more realistic.

- For instance consider the two curves below





- The first curve is defined by $x_t = \sin(2\pi t/12)$.

- The second curve has harmonic terms with frequencies at $\frac{1}{12}$, $\frac{2}{12}$ and $\frac{4}{12}$ and can be written as:

$$\begin{aligned} x_t &= \sin(2\pi(t)/12) \\ &+ 0.2\sin(2\pi(2t)/12) \\ &+ 0.1\sin(2\pi(4t)/12) \\ &+ 0.1\cos(2\pi(4t)/12) \end{aligned}$$

- Revisiting the equation above, this would equate to:

-$s_1 = 1$

-$c_1 = 0$

-$s_2 = 0.2$

-$c_2 = 0$

-$s_3 = 0$

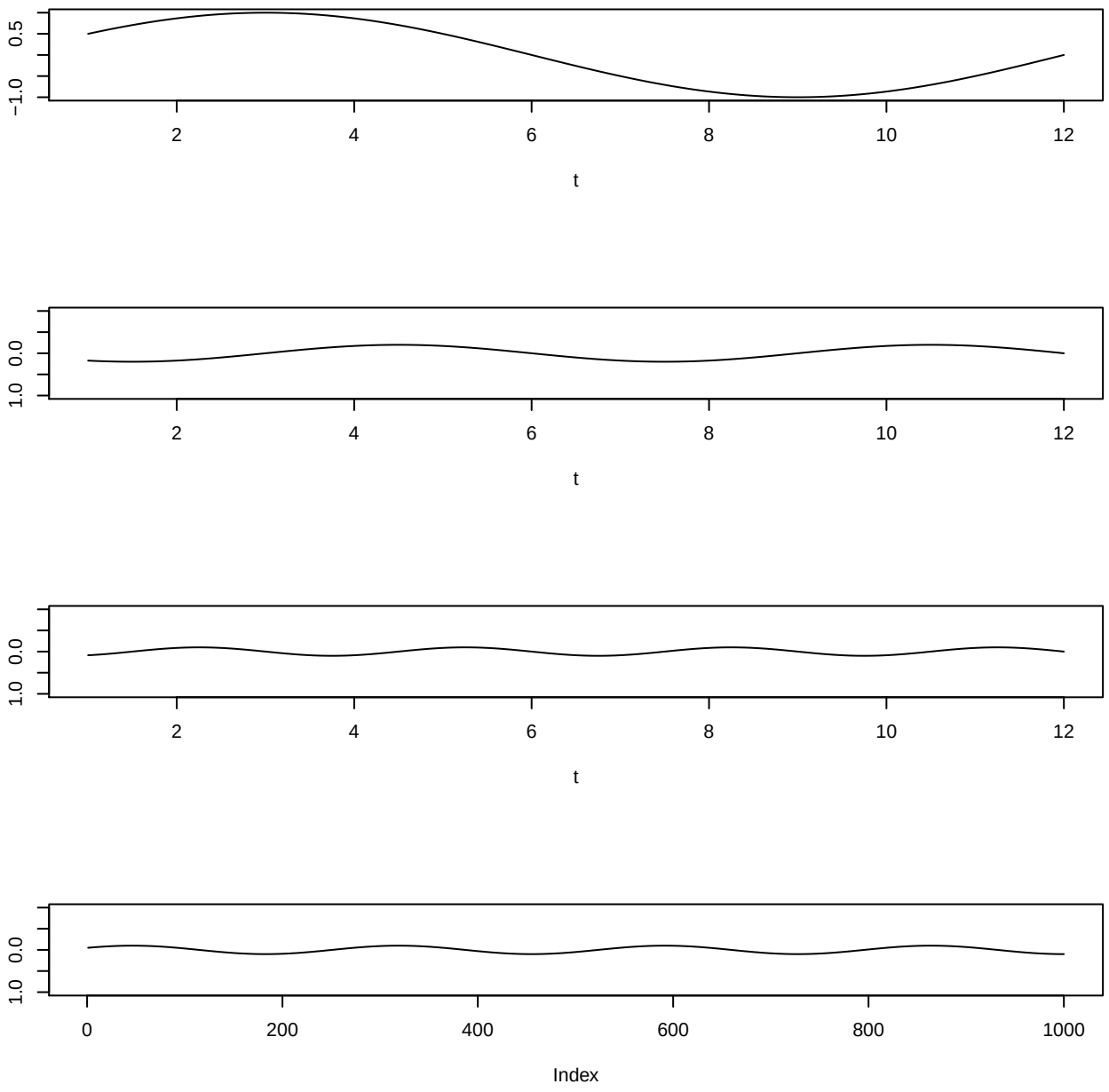
-$c_3 = 0$

-$s_4 = 0.1$

-$c_4 = 0.1$

- Create a figure that contains the four separate harmonic curves (`par(mfcol=c(4,1))` is one way to do this)

```r
par(mfcol=c(4,1))
t <- seq(1,12, length.out = 1000)
plot(t, sin(2 * pi * t / 12), type='l', ylab='')
plot(t, 0.2 * sin(2 * pi * 2 * t / 12), type = 'l', ylab='', ylim = c(1,-1))
plot(t, 0.1 * sin(2 * pi * 4 * t / 12), type = 'l', ylab='', ylim = c(1,-1))
plot(0.1 * cos(2 * pi * 4 * t / 12), type='l', ylab='', ylim = c(1,-1))
```

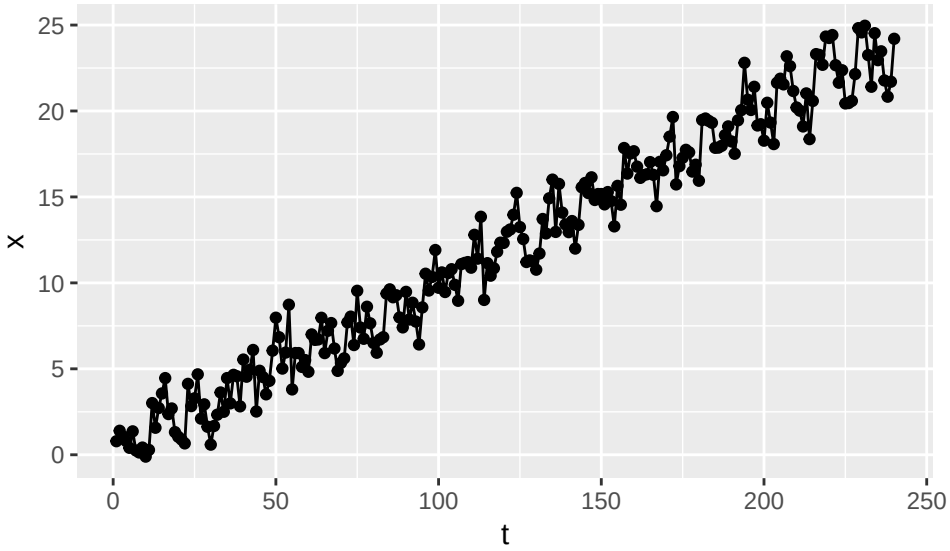


- Now we are going to build on the model from above such that

$$\begin{aligned} x_t &= .05 * t \\ &+ \sin(2\pi(t)/12) \\ &+ 0.2\sin(2\pi(2t)/12) \\ &+ 0.1\sin(2\pi(4t)/12) \\ &+ 0.1\cos(2\pi(4t)/12) \\ &+ w_t \end{aligned}$$

where $w_t \sim N(\mu = 0, \sigma^2 = 1^2)$. Simulate a time series from this model with a total fo 240 time points.

```
t <- 1:240
mean.t <- .1 * t +  sin(2 * pi * t / 12) +
        0.2 * sin(2 * pi * 2 * t / 12) +
        0.1 * sin(2 * pi * 4 * t / 12) +
        0.1 * cos(2 * pi * 4 * t / 12)
x = mean.t+rnorm(240,0,1)
library(ggplot2)
ts.df <- data.frame( x = x, t = t)
ggplot(data=ts.df, aes(y=x, x=t)) + geom_point() + geom_line()
```



- The final step is to fit a time series model. Our goal here is learn the values for s_i and c_i. To do so, we need to construct the sine and cosine components. These serve the same purpose as covariates typically denoted as X in a simple regression model.

```
t <- 1:240
SIN <- COS <- matrix(nrow = length(t), ncol=4) # restricting to 4 components
for (i in 1:4){
  COS[,i] <- cos(2 * pi * i * t / 12)
  SIN[,i] <- sin(2 * pi * i * t / 12)
}

lm.harm <- lm(x ~ t + COS + SIN)
summary(lm.harm)
```

```
##
## Call:
## lm(formula = x ~ t + COS + SIN)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.4910 -0.7051  0.0248  0.6138  3.2683
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.0818540  0.1304565   0.627  0.53099
## t            0.0996195  0.0009388 106.111  < 2e-16 ***
## COS1         0.0159431  0.0918831   0.174  0.86240
## COS2        -0.0584609  0.0918831  -0.636  0.52524
## COS3         0.1300433  0.0918831   1.415  0.15833
## COS4         0.2060345  0.0918831   2.242  0.02589 *
## SIN1         1.1348024  0.0919451  12.342  < 2e-16 ***
## SIN2         0.2939614  0.0918927   3.199  0.00157 **
## SIN3         0.0575998  0.0918831   0.627  0.53136
## SIN4         0.0041302  0.0918799   0.045  0.96418
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.006 on 230 degrees of freedom
## Multiple R-squared:  0.9801, Adjusted R-squared:  0.9794
## F-statistic:  1261 on 9 and 230 DF,  p-value: < 2.2e-16
```