When we analyze the regression trend models we should consider overfitting and underfitting situations; **underfitting** indicates high bias and low variance while **overfitting** indicates low bias and high variance.

**Bias** is the difference between the expected value of fitted values and observed values:

The **variance** of fitted values is the expected value of squared deviation from the mean of fitted values:

The **adjusted coefficient of determination** is used in the different degrees of polynomial trend regression models comparing. In the below formula **p** denotes the number of explanatory terms and **n** denotes the number of observations.

SSEis the residual sum of squares:

SSTis the total sum of squares:

When we examine the above formulas, we can notice the similarity between SSE and bias. We can easily say that if bias decreases, SSE will decrease and will increase. So we will use instead of bias to balance with variance and find the optimal degree of the polynomial regression.

First of all, we will create all the polynomial regression models which we’re going to compare.

a <- 1:15

models <- list()

for (i in seq\_along(a)){

models[[i]] <- assign(paste0('model\_', i),lm(new\_cases ~ poly(index, i), data=tur))

}

The variances of fitted values of all the degrees of polynomial regression models:

variance <- c()

for (i in seq\_along(a)) {

variance[i] <- mean((models[[i]][["fitted.values"]]-mean(models[[i]][["fitted.values"]]))^2)

}

To create an adjusted R-squared object we first create a summary object of the trend regression models; because the adj.r.squared feature is calculated in summary function.

models\_summ <- list()

adj\_R\_squared <- c()

for (i in seq\_along(a)) {

models\_summ[[i]] <- summary(models[[i]])

adj\_R\_squared[i] <- (models\_summ[[i]][["adj.r.squared"]])

}

Before analyzing numeric results of variance and we will show all the trend regression lines in separate plots for comparing models.

# Facetting models by degree for finding the best fit

library(ggplot2)

dat <- do.call(rbind, lapply(1:15, function(d) {

x <- tur$index

preds <- predict(lm(new\_cases ~ poly(x, d), data=tur), newdata=data.frame(x=x))

data.frame(cbind(y=preds, x=x, degree=d))

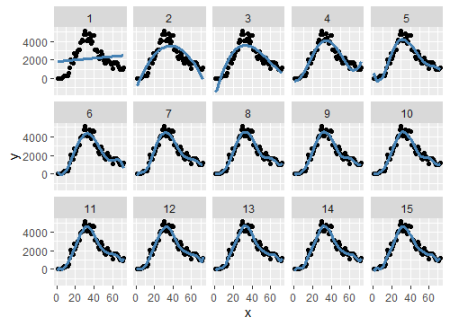
}))

ggplot(dat, aes(x,y)) +

geom\_point(data=tur, aes(index, new\_cases)) +

geom\_line(color="steelblue", lwd=1.1)+

facet\_wrap(~ degree,nrow = 3)



When we examine the above plots, we should pay attention to the curved of the tails; because it indicates overfitting which shows extreme sensitivity to the observed data points. In light of this approach, the second and third-degree of models appear to be more convenient to the data.

Let’s examine -variance tradeoff on the plot we created below.

library(gridExtra)

plot\_variance <- ggplot(df\_tradeoff,aes(degree,variance))+

geom\_line(size=2,colour="orange")+

scale\_x\_continuous(breaks = 1:15)+

theme\_bw(base\_line\_size = 2)

plot\_adj.R.squared <- ggplot(df\_tradeoff,aes(degree,adj\_R\_squared))+

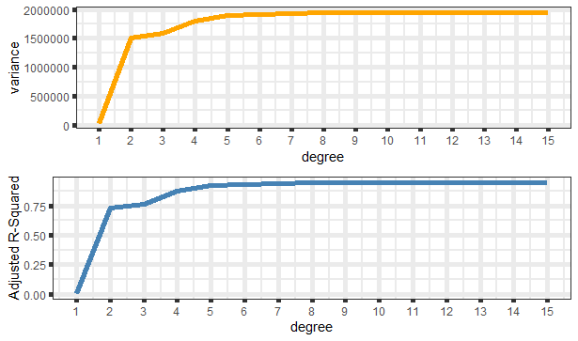
geom\_line(size=2,colour="steelblue")+

labs(y="Adjusted R-Squared")+

scale\_x\_continuous(breaks = 1:15)+

theme\_bw(base\_line\_size = 2)

grid.arrange(plot\_variance,plot\_adj.R.squared,ncol=1)



As we can see above, adjusted R-squared and variance have very similar trend lines. The more adjusted R-squared means the more complexity and low bias, but we have to take into account the variance; otherwise, we fall into the overfitting trap. So we need to look for low bias (or high ) and low variance as much as possible for optimal selection.

When we examine the variance plot, we can see that there is not much difference between second and third-degree; but after the third degree, there seems to be a more steep increase; it’s some kind of breaking. This might lead to overfitting. Thus the most reasonable option seems to be the 3rd degree. This approach is not a scientific fact but could be used for an optimal solution.