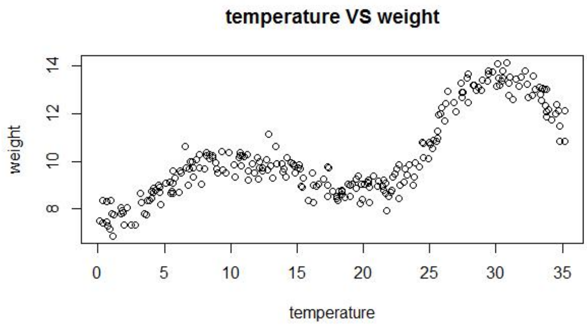
1.

data<-read.csv("nectar.csv")

plot(data$temp,data$weight,xlab = "temperature",ylab = "weight",main = "temperature VS weight")



When the temperature of the region when the nectar production is between 0-10 and 20-30, the average amount of nectar produced per month per plant on average increases with temperature.

When the temperature is between 10-20 and more than 30, the weight decreases with increasing temperature.

2.

library(splines)

#(a)

data.cs.bs=bs(data$temp,degree = 3,knots = c(7.5,15,22.5,30),intercept = T)

#(b)

data.cs.fit=lm(data$weight~-1+data.cs.bs, data=data)

summary(data.cs.fit)

#(c)

plot(x=data$temp,y=data$weight,xlab = "temperature",ylab = "weight")

lines(x=sort(data$temp),y=fitted(data.cs.fit)[order(data$temp)],col="red",lwd=3.5)

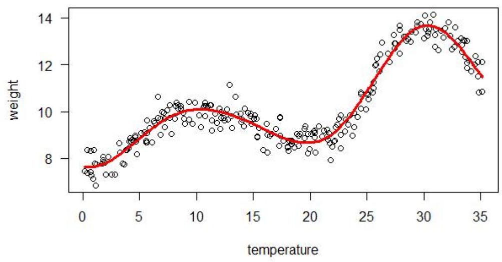
#(d)

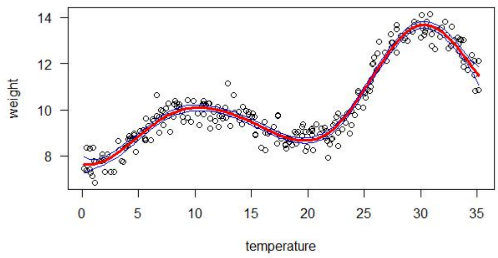
pre1<-predict(data.cs.fit,newdata = data,interval = "confidence")

#(e)

lines(x=sort(data$temp),y=pre1[,2][order(data$temp)],col="blue")

lines(x=sort(data$temp),y=pre1[,3][order(data$temp)],col="blue")






```
Call:
lm(formula = data$weight ~ -1 + data.cs.bs, data = data)

Residuals:
    Min      1Q  Median      3Q     Max 
-1.0500 -0.3074  0.0292  0.2899  1.2552 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
data.cs.bs1   7.6309     0.1732   44.06   <2e-16 ***
data.cs.bs2   7.4778     0.2073   36.06   <2e-16 ***
data.cs.bs3  10.7397     0.1948   55.12   <2e-16 ***
data.cs.bs4   9.6827     0.1594   60.73   <2e-16 ***
data.cs.bs5   7.3280     0.1606   45.62   <2e-16 ***
data.cs.bs6  15.5524     0.1988   78.25   <2e-16 ***
data.cs.bs7  12.6270     0.2055   61.46   <2e-16 ***
data.cs.bs8  11.4639     0.2008   57.08   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.4309 on 242 degrees of freedom
Multiple R-squared:  0.9983,    Adjusted R-squared:  0.9983 
F-statistic: 1.788e+04 on 8 and 242 DF,  p-value: < 2.2e-16
```

3.

#(a)

h1=data$temp

h2=data$temp^2

h3=data$temp^3

h4=(data$temp-7.5)

h4[h4<0]=0

h4=h4^3

h5=(data$temp-15)

h5[h5<0]=0

h5=h5^3

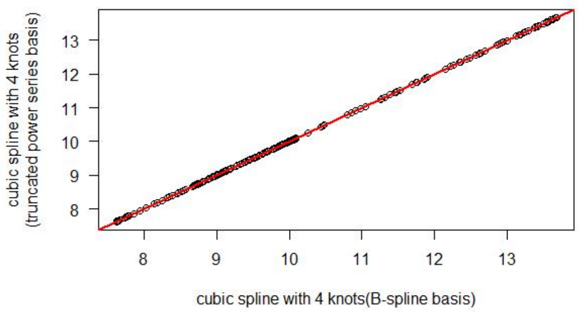
h6=(data$temp-22.5)

h6[h6<0]=0

h6=h6^3

```
h7=(data$temp-30)

h7[h7<0]=0

h7=h7^3

#(b)

data.tpsb.fit=lm(data$weight~h1+h2+h3+h4+h5+h6+h7,data = data)

summary(data.tpsb.fit)

#(c)

par(mar=c(5,5,2,2)+0.2,las=1)

plot(x=fitted(data.cs.fit),y=fitted(data.tpsb.fit),

     xlab = "cubic spline with 4 knots(B-spline basis)",

     ylab = "cubic spline with 4 knots\n(truncated power series basis)")

abline(0,1,col="red",lwd=2)
```

```
Call:
lm(formula = data$weight ~ h1 + h2 + h3 + h4 + h5 + h6 + h7,
    data = data)

Residuals:
    Min      1Q  Median      3Q     Max 
-1.0500 -0.3074  0.0292  0.2899  1.2552 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept)  7.6435138  0.1924887  39.709  < 2e-16 ***
h1          -0.0946109  0.1445307  -0.655 0.513342    
h2           0.1016888  0.0283155   3.591 0.000398 ***
h3          -0.0069195  0.0015683  -4.412 1.54e-05 ***
h4           0.0088098  0.0020662   4.264 2.88e-05 ***
h5           0.0031725  0.0009944   3.190 0.001610 ** 
h6          -0.0163280  0.0011120 -14.684  < 2e-16 ***
h7           0.0212147  0.0046106   4.601 6.78e-06 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.4309 on 242 degrees of freedom
Multiple R-squared:  0.9437,	Adjusted R-squared:  0.9421 
F-statistic: 579.5 on 7 and 242 DF,  p-value: < 2.2e-16
```



The t-values of the coefficients estimates of the truncated power series basis are generally smaller than those of B-spline basis in Question 2. This indicates the uncertainty relative to the magnitnude of the coefficients is larger for truncated power series basis, than for the B-spline basis. Therefore, in general, there is greater uncertainty associated with the coefficient estimates of the truncated power series basis, than those of B-spline basis.

Teacher’ solution:

(c) How do the uncertainty associated with the coefficient estimates in part (b) compare to those obtained in Question 2? Explain the differences you have observed. [3 marks]

In general, the t-values of the coefficients of the truncated power series basis (in part (b)) are smaller than those of the B-spline basis created in Question 2. This means that the uncertainty associated with the coefficient estimates of the truncated power series basis is larger relative to the effect size. This is due to the fact that the truncated power series basis tends to be highly correlated.

4.

(a)

If we have K knots in a a cubic spline, then number of degrees of freedom is K+4

A nature cubicspline with K knots has K degrees of freedom.

If we want to fit a natural cubic spline that has the same degrees of freedom as the cubic spline

model in Question 2, 4+4-2=6 internal knots is needed.

28-7=21

21/7=3

7,10,13,16,19,22,25,28

```
#(b)

data.ns=ns(x=data$temp,knots = c(10,13,16,19,22,25),

                  Boundary.knots = c(7,28),intercept = T)

#(c)

data.ns.fit=lm(data$weight~ -1+data.ns, data=data)

summary(data.ns.fit)
```

#(d)

```
plot(x=data$temp,y=data$weight,xlab = "temperature",ylab = "weight")

lines(x=sort(data$temp),y=fitted(data.ns.fit)[order(data$temp)],col="red",lwd=3.5)
```

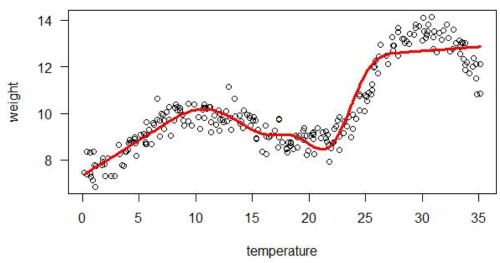
#(e)

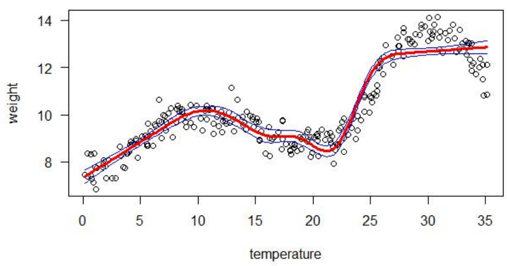
```
pre2<-predict(data.ns.fit,newdata = data,interval = "confidence")
```

#(f)

```
lines(x=sort(data$temp),y=pre2[,2][order(data$temp)],col="blue")

lines(x=sort(data$temp),y=pre2[,3][order(data$temp)],col="blue")
```

```
Call:
lm(formula = data$weight ~ -1 + data.ns, data = data)

Residuals:
     Min       1Q   Median       3Q      Max 
-2.02773 -0.37904  0.04421  0.43153  1.43360 

Coefficients:
         Estimate Std. Error t value Pr(>|t|)    
data.ns1   8.7159     0.1300   67.06   <2e-16 ***
data.ns2   9.9941     0.2478   40.33   <2e-16 ***
data.ns3   8.8227     0.2834   31.13   <2e-16 ***
data.ns4   9.4012     0.2644   35.55   <2e-16 ***
data.ns5   7.4992     0.2367   31.69   <2e-16 ***
data.ns6   8.5650     0.1501   57.07   <2e-16 ***
data.ns7  24.3516     0.1570  155.15   <2e-16 ***
data.ns8   4.7459     0.1201   39.51   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.6036 on 242 degrees of freedom
Multiple R-squared:  0.9967,    Adjusted R-squared:  0.9966 
F-statistic:  9097 on 8 and 242 DF,  p-value: < 2.2e-16
```





(g)

The model fit will not be as good near the boundaries. The cubic spline has waves at both boundary regions of the curve, when the true curve is almost flat in those regions.

A natural cubic spline is a cubic spline with an additional condition: the function is linear beyond the boundary knots(7 and 28).

While the constraints can reduce the variance near the boundaries, they also imply the model fit will not be as good near the boundaries.

Teacher's solution:

(g) Does the scatterplot in part (f) display any problems? Explain your answer. [3 marks]

The model does not capture the major peak at temp = 30, leading to biases in the temp range roughly between 27 and 35 degrees C. Additionally, there are wiggliness in the in the temp range between 15 and 22 degrees C, which suggests overfitting. The problem is the result of the interval between the boundary knots not covering a sufficiently wide range of temp, such that the peak is beyond boundary knots and there may be too many internal knots (and hence making the model too complex) for the interval between the boundary knots.

5.

```
#(a)

data.sm.5df=smooth.spline(x=data$temp,y=data$weight,df=5)

data.sm.5df

#(b)

data.sm.10df=smooth.spline(x=data$temp,y=data$weight,df=10)

data.sm.10df

#(c)

data.sm.20df=smooth.spline(x=data$temp,y=data$weight,df=20)

data.sm.20df

#(d)

plot(x=data$temp,y=data$weight,xlab="temperature",ylab="weight")

lines(x=sort(data$temp),y=fitted(data.sm.5df)[order(data$temp)],col="red",lwd=3.5)

plot(x=data$temp,y=data$weight,xlab="temperature",ylab="weight")

lines(x=sort(data$temp),y=fitted(data.sm.10df)[order(data$temp)],col="red",lwd=3.5)

plot(x=data$temp,y=data$weight,xlab="temperature",ylab="weight")

lines(x=sort(data$temp),y=fitted(data.sm.20df)[order(data$temp)],col="red",lwd=3.5)
```

#(e)

plot(x=data$temp,y=data$weight,xlab="temperature",ylab="weight")

lines(x=sort(data$temp),y=fitted(data.sm.5df)[order(data$temp)],col="red",lwd=2)

lines(x=sort(data$temp),y=fitted(data.sm.10df)[order(data$temp)],col="green",lwd=2
)

lines(x=sort(data$temp),y=fitted(data.sm.20df)[order(data$temp)],col="blue",lwd=2)

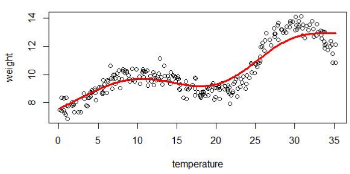
legend(0,14,lty=c(1,1,1),lwd = c(2,2,2),col = c("red","green","blue"),legend =
c("df=5","df=10","df=20"))

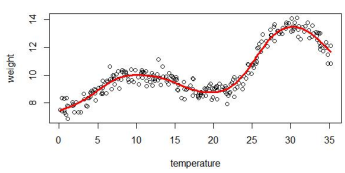
```
> data.sm.5df
Call:
smooth.spline(x = data$temp, y = data$weight, df = 5)

Smoothing Parameter  spar= 0.9860262  lambda= 0.01569168 (12 iterations)
Equivalent Degrees of Freedom (Df): 4.999287
Penalized Criterion (RSS): 111.2472
GCV: 5.80002
```

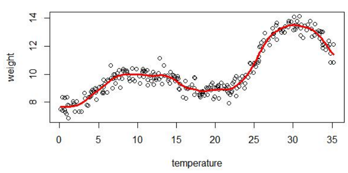
```
> data.sm.10df
Call:
smooth.spline(x = data$temp, y = data$weight, df = 10)

Smoothing Parameter  spar= 0.7893161  lambda= 0.0005943087 (12 iterations)
Equivalent Degrees of Freedom (Df): 10.00145
Penalized Criterion (RSS): 43.85253
GCV: 6.568637
```

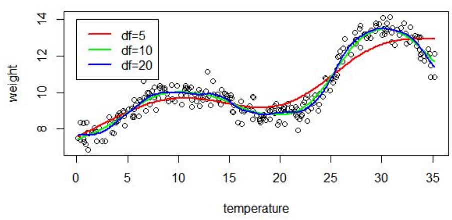
```
> data.sm.20df
Call:
smooth.spline(x = data$temp, y = data$weight, df = 20)

Smoothing Parameter  spar= 0.6079809  lambda= 2.910152e-05 (14 iterations)
Equivalent Degrees of Freedom (Df): 20.00317
Penalized Criterion (RSS): 38.79253
GCV: 7.250645
```

Df=5 Df=10 Df=20









As the effective degrees of freedom increases, the spline follows the Runge function more and more closely. In this case, setting the effective degrees of freedom to 20 leads to overfitting as it creates extra curvature that seems to be modelling to noise instead of the trend.

Teacher's solution:

(e)Examine and compare the fitted values from parts (a), (b) and (c), and discuss any concerns you have observed. [3 marks]

When df = 5, the spline underfits, and there are apparent biases. The trough around temp = 20 appears to be over-estimated, while the peak around time = 10 is underestimated and the peak around time = 30 is not captured by the model (and hence is not shown in the fitted line).

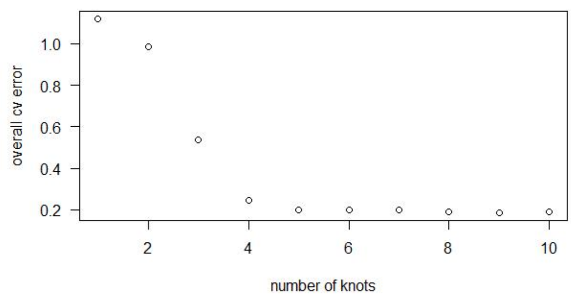
When df = 10, the spline is characterises the main trend displayed by the data points without any obvious biases.

When df = 20, the spline is clearly overfitting as the fitted function appears wiggly between temp = 7.5 and temp = 25, where the wiggliness seems to be modelling the noise rather than the trend.

6.

```
cverror<-c()

for (j in 1:10) {

  kn=seq(1,34,length.out=2+j)

  kn=kn[-1]; kn=kn[-length(kn)]

  cve<-c()

  for (i in 1:10) {

    x<-NULL

    k<-NULL

    x<-data.frame("x"=data$temp[data$group==i])

    y<-data.frame("y"=data$weight[data$group==i])

    k<-data.frame("x"=data$temp[data$group!=i],"y"=data$weight[data$group!=i])

    colnames(x) <- c('xx')

    colnames(k) <- c('xx','yy')
```

```
    data.k1.ns.fit=lm(yy~ -1+ns(x=xx,knots = kn,
                                          Boundary.knots = c(1,34),intercept = T),
data=k)

    pre.k<-predict(data.k1.ns.fit,newdata = x)
    cve[i]<-sum((y-pre.k)^2)
  }
  cverror[j]<-sum(cve)
}
CVerror=cverror/length(data$temp)
plot(1:10,CVerror,xlab="number of knots", ylab="overall cv error")
which.min(CVerror)
CVerror
```




```
> CVerror
 [1] 1.1209572 0.9843904 0.5352234 0.2451088 0.1980773 0.2004018 0.1984835
 [8] 0.1915531 0.1866142 0.1878781
```