Solution to Series 8

1. a) The likelihood reads

$$L(\beta; (x_1, m_1, N_1), \ldots, (x_n, m_n, N_n)) = \prod_{i=1}^{n} \binom{m_i}{N_i} \pi(x_i)^{N_i} (1 - \pi(x_i))^{m_i - N_i} \ ;$$

taking the logarithm gives

$$\ell(\beta; (x_1, m_1, N_1), \ldots, (x_1, m_1, N_1)) = \sum_{i=1}^{n} \left[\log \binom{m_i}{N_i} + N_i \log(\pi(x_i)) + (m_i - N_i) \log\left(1 - \pi(x_i)\right) \right] \ . \quad (1)$$

From the logistic transform,

$$g(x_i) = \log\left(\frac{\pi(x_i)}{1 - \pi(x_i)}\right) = \log(\pi(x_i)) - \log(1 - \pi(x_i)) \ ,$$

where we leave out the parameter β of g for better readability, we find $\log(\pi(x_i)) = g(\beta; x_i) + \log(1 - \pi(x_i))$. Solving for $\pi(x_i)$ yields

$$\pi(x_i) = \frac{e^{g(x_i)}}{1 + e^{g(x_i)}} \quad \text{and} \quad 1 - \pi(x_i) = \frac{1}{1 + e^{g(x_i)}} \ ,$$

and hence $\log(1 - \pi(x_i)) = -\log\left(1 + e^{g(x_i)}\right)$. Plugging those identities for $\log(\pi(x_i))$ and $\log(1 - \pi(x_i))$ into Equation (1) yields the claimed result.

b) We write a simple help function that calculates $g(\beta; x)$:

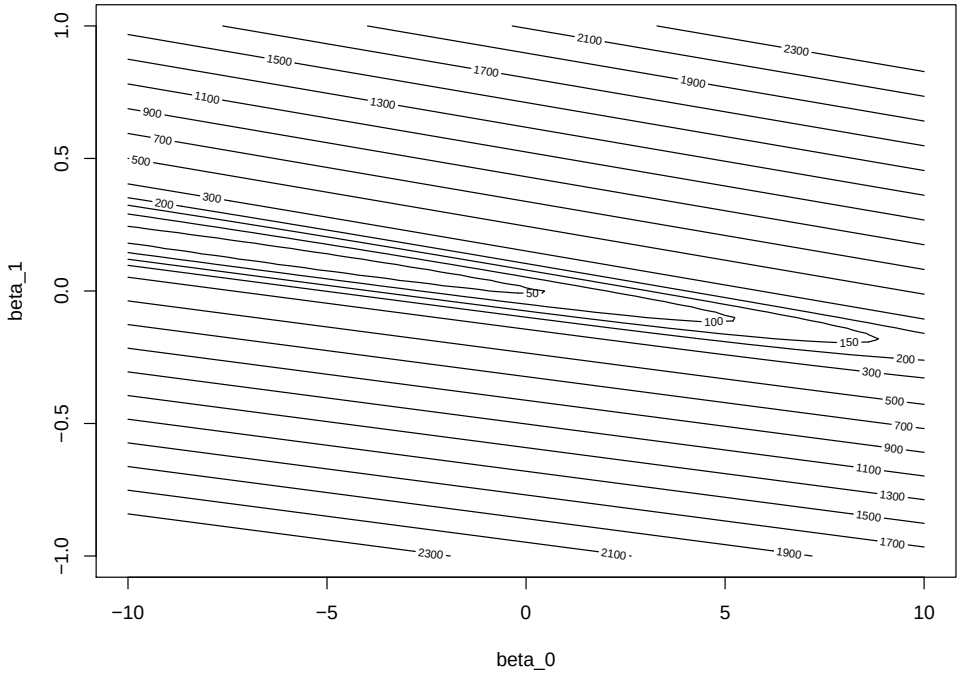
```
> g <- function(beta, x) beta[1] + beta[2]*x
```

Next, we implement the negative log-likelihood calculated in task a):

```
> neg.ll <- function(beta, data){
   - sum(log(choose(data$m, data$N)) +
         data$N * g(beta,data$age) -
         data$m * log( 1 + exp(g(beta,data$age))))
 }
```

We then generate the contour plot:

```
> heart <- read.table("http://stat.ethz.ch/Teaching/Datasets/heart.dat",
                      header = TRUE)
> beta0.grid <- seq(-10, 10, length = 101)
> beta1.grid <- seq(-1, 1, length = 101)
> ## initialize grid:
> neg.ll.values <- matrix(0, nrow = length(beta0.grid), ncol = length(beta1.grid))
> ## evaluate negative log-likelihood on every combination of beta0 and beta1
> for (i in 1:length(beta0.grid))
   for (j in 1:length(beta1.grid))
     neg.ll.values[i, j] <- neg.ll(c(beta0.grid[i], beta1.grid[j]), heart)
> contour(beta0.grid, beta1.grid, neg.ll.values, xlab = "beta_0",
         ylab = "beta_1",levels=c(seq(50,200,50),seq(300,2300,200)))
```



c)
```
> fit <- glm(cbind(N, m - N) ~ age, family = binomial, data = heart)
> summary(fit)

Call:
glm(formula = cbind(N, m - N) ~ age, family = binomial, data = heart)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-1.36404  -0.54656   0.02468   0.55254   1.53533

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  -5.0993     1.1090  -4.598 4.26e-06 ***
age           0.1084     0.0238   4.555 5.24e-06 ***
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 53.466  on 42  degrees of freedom
Residual deviance: 25.153  on 41  degrees of freedom
AIC: 63.888

Number of Fisher Scoring iterations: 4
```

We obtain $\hat{\beta}_0 = -5.1$ for the intercept and $\hat{\beta}_1 = 0.11$ for the coefficient of age. The influence of the covariable age is significant (p-value < 0.001). The positive sign of $\hat{\beta}_1$ means that the logit increases with age, and since the logit is an increasing function of the probability of having symptoms, this probability increases with age as well. The absolute value of $\widehat{\beta}$ is harder to interpret due to the logit-scale.

By optimizing our own log-likelihood function, we get the same parameters:

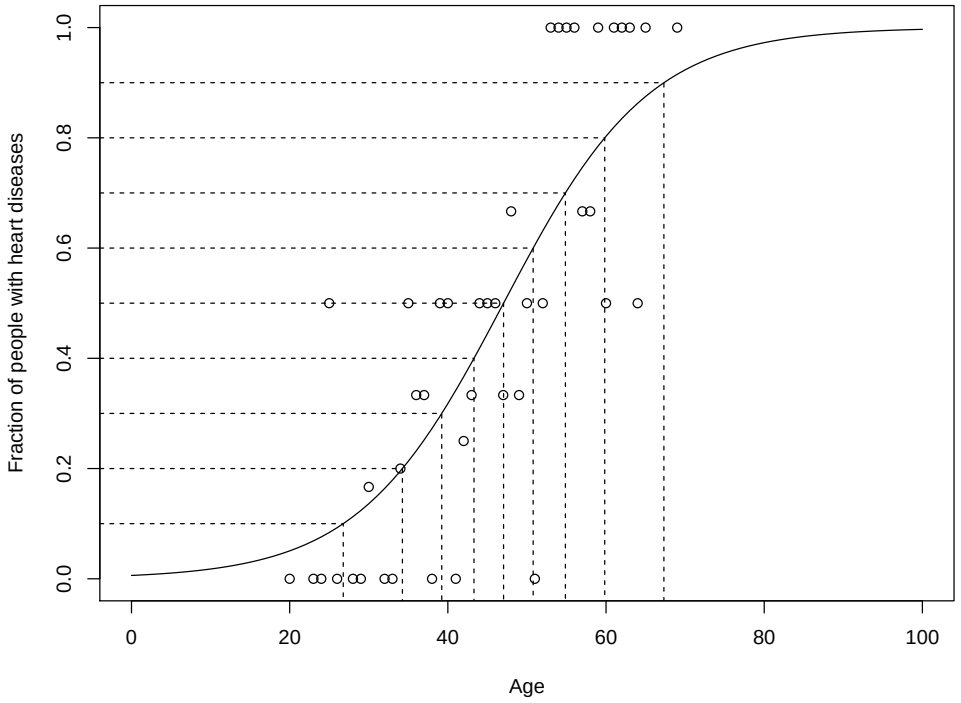
```
> optim(c(0, 0), neg.ll, data = heart)$par
[1] -5.0990932  0.1083889
```

d) From the model equation, we infer that

$$x_i = \frac{\log\left(\frac{\pi_i}{1-\pi_i}\right) - \beta_0}{\beta_1}.$$

Setting $\pi_i = 0.1, 0.2, \ldots, 0.9$, and plugging in the estimated values for β_0 and β_1, we obtain the age at which we expect $10\%, 20\%, \ldots, 90\%$ of people to have symptoms of heart disease.

```
> new.age <- 0:100
> heart.pred <- predict(fit, new = data.frame(age = new.age), type = "response")
> plot(heart$age, heart$N/heart$m, xlim = c(0, 100), ylim = c(0,1),
      xlab = "Age", ylab = "Fraction of people with heart diseases")
> lines(new.age, heart.pred)
> perc         <- (1:9)/10
> x.age        <- (log(perc/(1-perc)) - coef(fit)[1])/coef(fit)[2]
> names(x.age) <- perc
> for(n in 1:9)
     lines(c(-4, x.age[n], x.age[n]), c(perc[n], perc[n], -0.04), lty = 2)
```



symptoms	10%	20%	30%	40%	50%	60%	70%	80%	90%
age	27	34	39	43	47	51	55	60	67

Between ages 39 and 55, the predicted probability of having symptoms increases linearly ($+10\%$ every 4 years). Out of this range, the probability increases less fast.

2. **a)** The scatterplot matrix of the data is already available in the manuscript. There is a clear outlier in the variable wdsp (leverage point!) which should be removed. To avoid heteroscedastic errors we take the log of the response upo3.

We load the data. Transform the response and remove the original response.

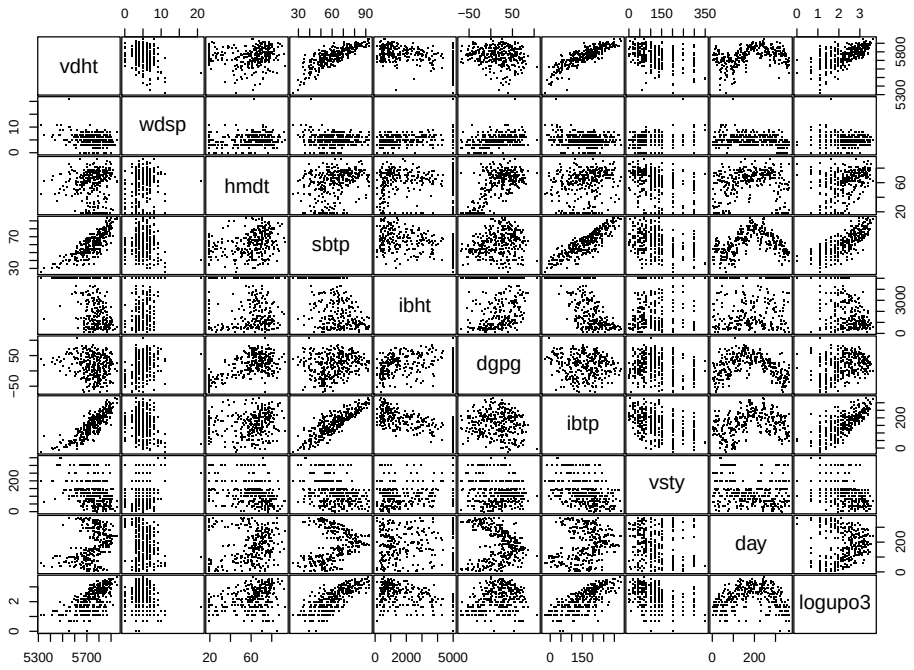
```
> data(ozone, package = "gss")
> ozone$logupo3 <- log(ozone$upo3)
> d.ozone <- subset(ozone, select=-upo3)
```

As we have a look at the data we see that there is an outlier in the variable wdsp. We take out the outlier and save the edited data file.

```
> pairs(d.ozone, pch = ".",gap = 0.1)
> (out <- which.max(d.ozone["wdsp"]))
```

```
[1] 92
```

```
> d.ozone.e <- d.ozone[-out,]
```



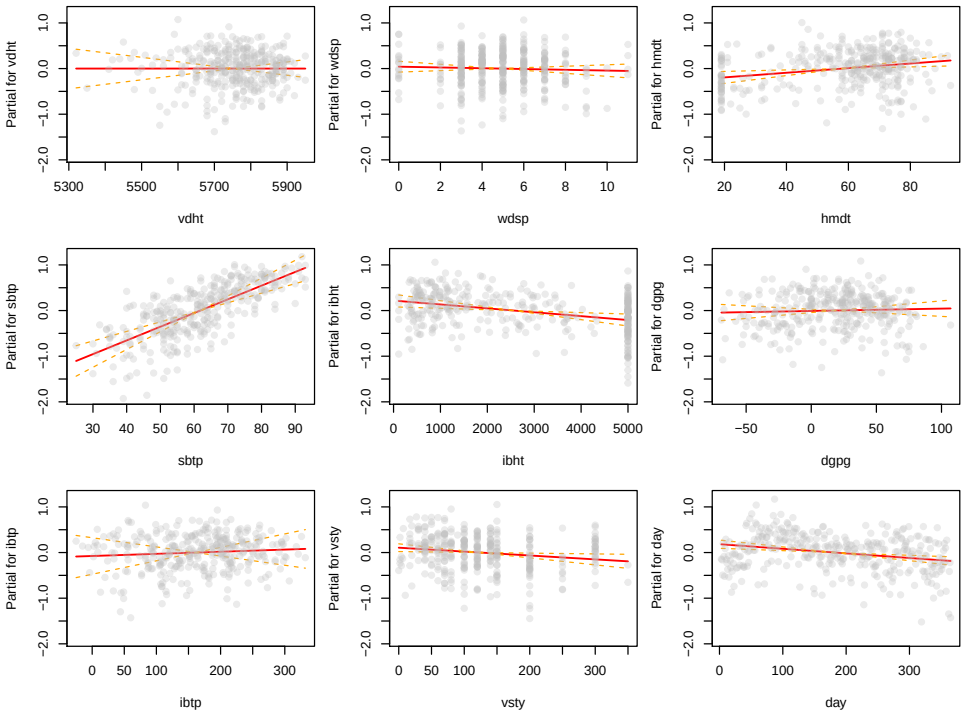
b) We fit the linear regression model with maximal interaction degree 1 and compare it to an additive model.

```
> ## package for formula
> require(sfsmisc)
> ##Linear models
> ##fit 1 (polynomial of degree 1)
> form1 <- as.formula("logupo3~.")
> fit1 <- lm(form1, data = d.ozone.e)
> ##fits 2 to 5 (polynomial of degree d={2,..,5})
>
> form2 <- wrapFormula(form1, data = d.ozone.e, wrapString="poly(*,degree=2)")
> fit2 <- lm(form2, data = d.ozone.e)
> form3 <- wrapFormula(form1, data = d.ozone.e, wrapString="poly(*,degree=3)")
> fit3 <- lm(form3, data = d.ozone.e)
> form4 <- wrapFormula(form1, data = d.ozone.e, wrapString="poly(*,degree=4)")
> fit4 <- lm(form4, data = d.ozone.e)
> form5 <- wrapFormula(form1, data = d.ozone.e, wrapString="poly(*,degree=5)")
> fit5 <- lm(form5, data = d.ozone.e)
> ## GAM
> require(mgcv)
> gamForm <- wrapFormula(form1, data = d.ozone.e)
> g1 <- gam(gamForm, data = d.ozone.e)
> summary(g1)
Family: gaussian
Link function: identity

Formula:
logupo3 ~ s(vdht) + s(wdsp) + s(hmdt) + s(sbtp) + s(ibht) + s(dgpg) +
    s(ibtp) + s(vsty) + s(day)

Parametric coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   2.2148     0.0172   128.8   <2e-16 ***
---
Signif. codes:
0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

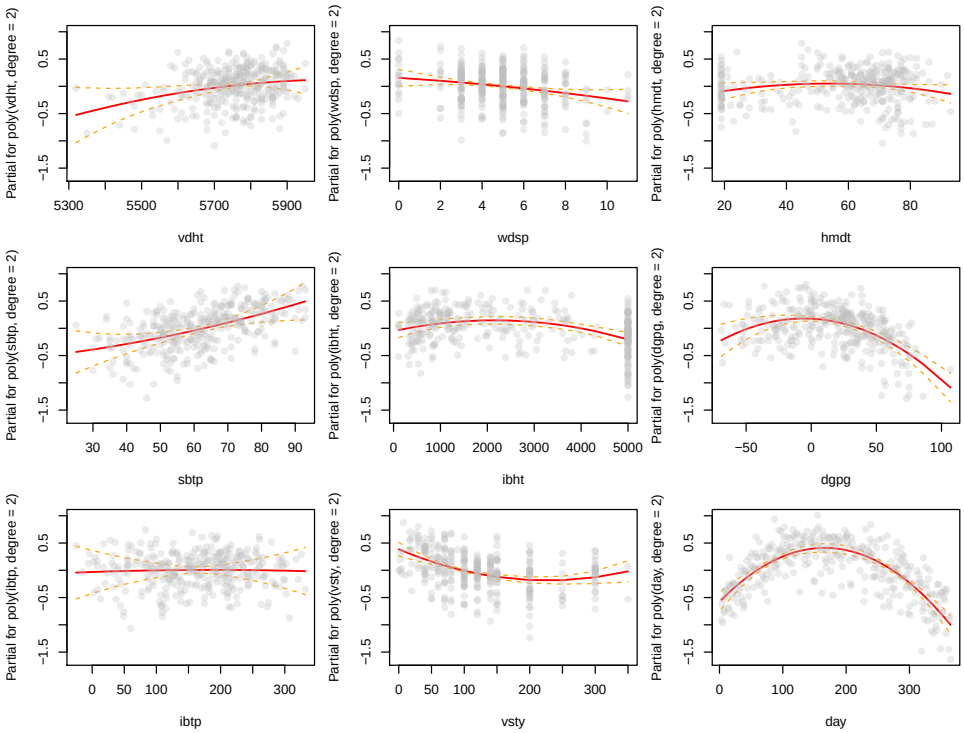
```
Approximate significance of smooth terms:
          edf Ref.df      F  p-value
s(vdht) 1.000  1.000 10.878  0.00109 **
s(wdsp) 1.046  1.090  8.290  0.00358 **
s(hmdt) 2.372  2.983  2.454  0.06364 .
s(sbtp) 3.855  4.803  4.163  0.00141 **
s(ibht) 2.785  3.407  5.181  0.00111 **
s(dgpg) 3.267  4.153 14.364 5.01e-11 ***
s(ibtp) 1.000  1.000  0.441  0.50731
s(vsty) 5.513  6.671  6.047 2.33e-06 ***
s(day)  4.616  5.741 25.029  < 2e-16 ***
---
Signif. codes:
0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

R-sq.(adj) =  0.826   Deviance explained =   84%
GCV = 0.10579  Scale est. = 0.097287  n = 329
```

c) We now plot the fits of the linear models.

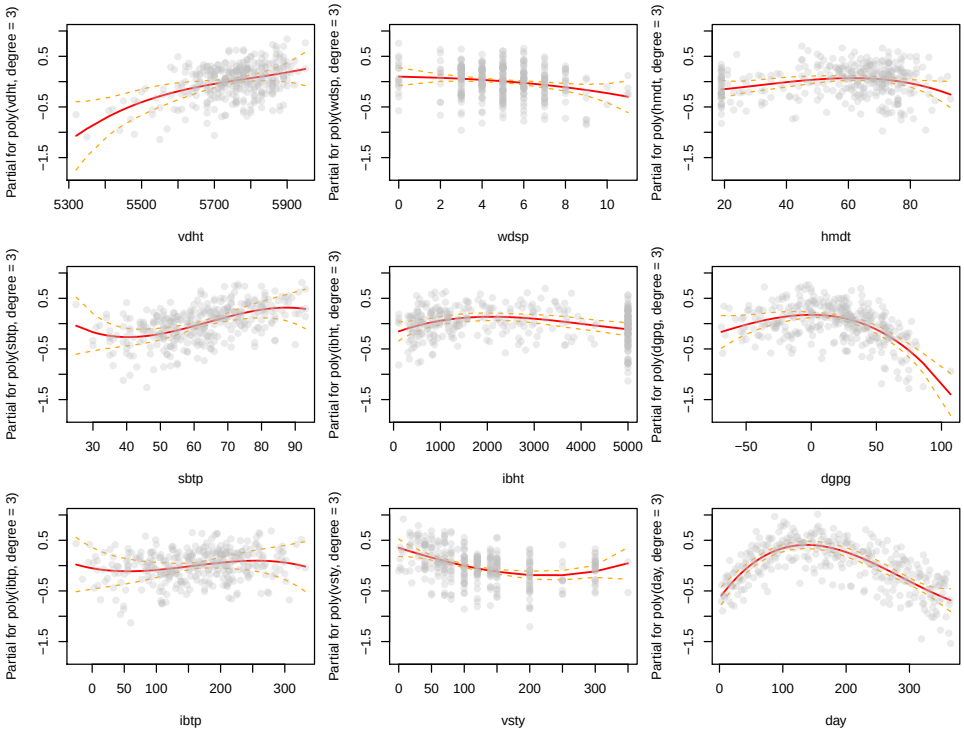
```
> par(mfrow=c(3,3))
> termplot(fit1, partial.resid=TRUE, rug=FALSE, se=TRUE, col.res='#C0C0C050', pch=19)
```



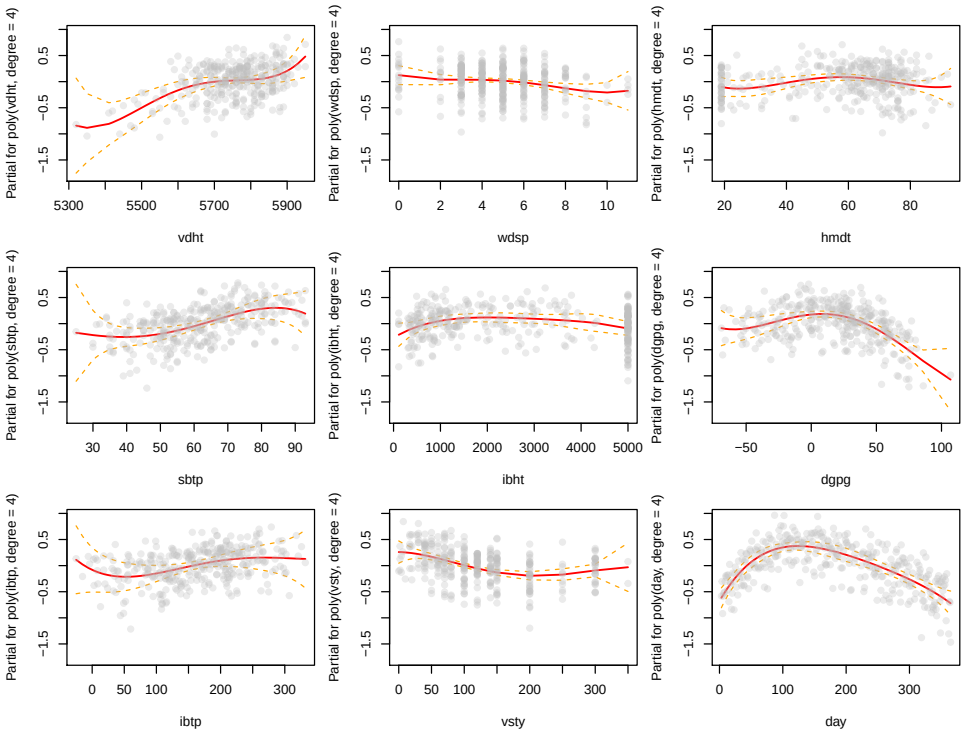

```
> par(mfrow=c(3,3))
> termplot(fit2, partial.resid=TRUE, rug=FALSE, se=TRUE, col.res='#C0C0C050', pch=19)
```



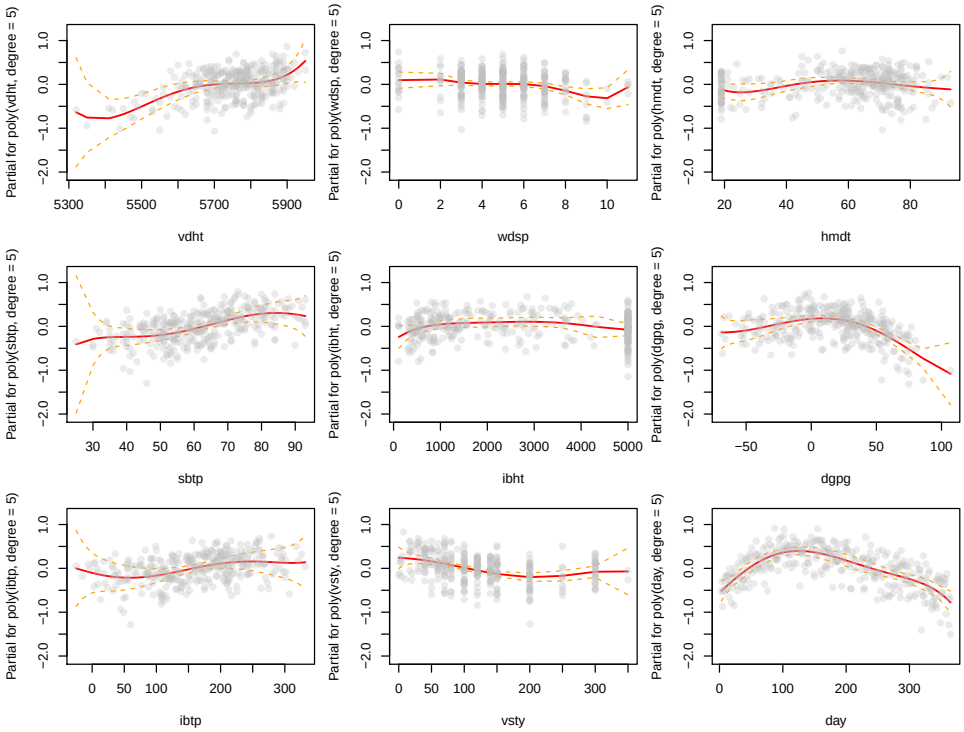

```
> par(mfrow=c(3,3))
> termplot(fit3, partial.resid=TRUE, rug=FALSE, se=TRUE, col.res='#C0C0C050', pch=19)
```




```
> par(mfrow=c(3,3))
> termplot(fit4, partial.resid=TRUE, rug=FALSE, se=TRUE, col.res='#C0C0C050', pch=19)
```



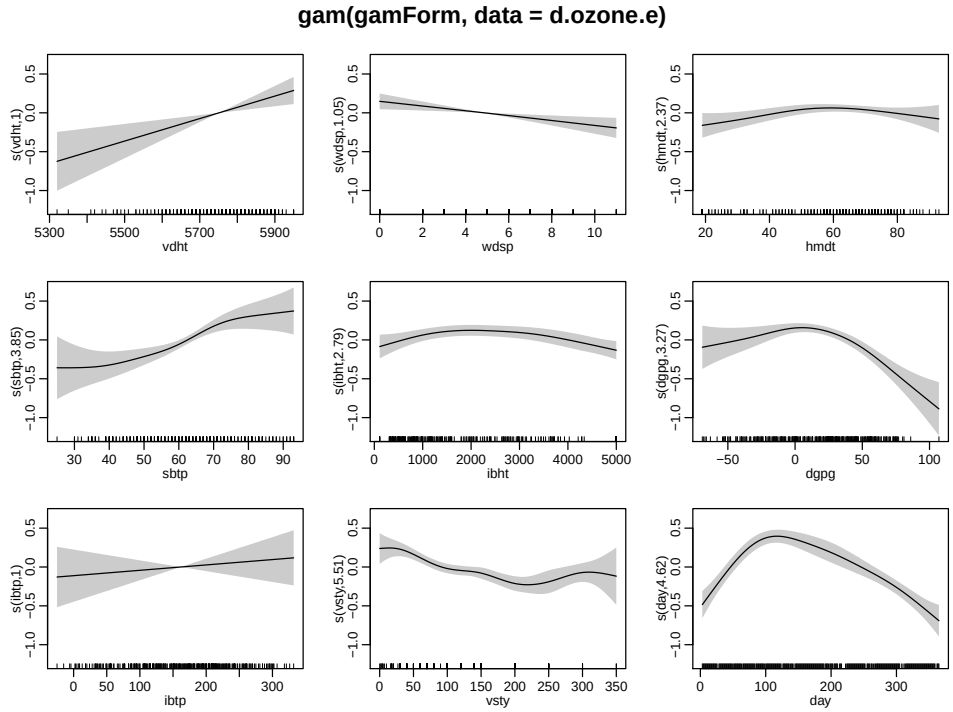

```
> par(mfrow=c(3,3))
> termplot(fit5, partial.resid=TRUE, rug=FALSE, se=TRUE, col.res='#C0C0C050', pch=19)
```



We can observe that plots of `fit3`,`fit4`,`fit5` do not differ much, except close to the ends of the data range. With larger polynomial degrees the fitted functions fit closer to the data. As polynomial of degree 2 already fits the data quite well we could argue that it would be a good choice, as the model is not too complicated. However, doing model selection by eye would not necessarily return the best model.

Now we plot the fit of generalized additive model.

```
> source("ftp://stat.ethz.ch/Teaching/maechler/CompStat/plotGAM.R")
> p.gam(g1, scheme = 1)
```




```
> #You can also use the default plot function.
> par(mfrow=c(3,3))
> plot(g1)
```

We observe that unlike the linear models, where all predictors were fitted as a polynomial of the same degree, in the additive model this does not have to be the case. Predictors `vdht`, `wdsp` and `ibtp` are fitted as lines, whereas the rest of the predictors are fitted as polynomials of a degree larger than 1. By not using the same polynomial degree for every predictor we are reducing the complexity of our model (as compared to `fit5`) while addressing the complexity of the data appropriately (as compared to `fit1`).

d) We select our preferred model by calculating the Mallows' C_p statistics for all 6 models.

```
> # Mallows Cp
>
> Cp <- function(object,sigma){
   res<-residuals(object)
   n <- length(res)
   p <- n-object$df.residual
   SSE <- sum(res^2)
   SSE/sigma^2-n+2*p
 }
> # choose sigma
> sigma<-summary(fit5)$sigma
> # Calculate and print Mallows's Cp statistics for all 5 models
> print(c(fit1 = Cp(fit1,sigma), fit2 = Cp(fit2,sigma), fit3 = Cp(fit3,sigma),
            fit4 = Cp(fit4,sigma), fit5 = Cp(fit5,sigma), g1 = Cp(g1,sigma)))
     fit1      fit2      fit3      fit4      fit5        g1
209.06499  40.80239  31.53408  33.30466  46.00000  16.82841
```

Of the linear models the model with degree 3 has the lowest Mallows' C_p statistic, which makes it our preferred model. Of all the models the generalized additive model is our preferred model for the same reason.