University of Edinburgh, School of Mathematics
Incomplete Data Analysis, 2020/2021
Solutions to the practice examples

Vanda Inácio

1. After mean imputation, the estimated mean of Y_2 is 35 (mean imputation does not change the mean of the observed data).

2. (a) We were first asked to compute the overall mean of the systolic blood pressure variable (and its associated standard error) using a complete case analysis. We obtain that the overall estimated mean is 146.53 and that the associated standard error is 5.53.

```
load("datasbp.Rdata")
ind <- which(is.na(datasbp$SBP) == FALSE) #indices of subjects with SBP observed
mccoverall <- mean(datasbp$SBP, na.rm = TRUE)
seccoverall <- sd(datasbp$SBP, na.rm = TRUE)/sqrt(length(ind))
mccoverall; seccoverall
```

```
## [1] 146.525
```

```
## [1] 5.529613
```

Further, and as asked in the question, the mean SBP for men and for women are, respectively, 156.55 (5.27) and 124.46 (6.10). The values in brackets are the standard errors.

```
#cc analysis women
indf <- which(is.na(datasbp$SBP) == FALSE & datasbp$Sex == "Female")
mccf <- mean(datasbp$SBP[indf])
seccf <- sd(datasbp$SBP[indf])/sqrt(length(indf))
mccf; seccf
```

```
## [1] 124.46
```

```
## [1] 6.09759
```

```
#cc analysis men
indm <- which(is.na(datasbp$SBP) == FALSE & datasbp$Sex == "Male")
mccm <- mean(datasbp$SBP[indm])
seccm <- sd(datasbp$SBP[indm])/sqrt(length(indm))
mccm; seccm
```

```
## [1] 156.5545
```

```
## [1] 5.269076
```

(b) Using mean imputation, the estimated SBP mean is 146.53 (we already knew this from the complete case analysis), while the associated standard error is 4.39. Of course, the standard error of the mean based on mean imputation is smaller than that of the complete case analysis. Why? Have a look at both the sample variance and standard error formulae. For instance, for the sample variance, the

numerators under a complete case analysis and a mean imputation are the same but the denominator is bigger for mean imputation.

```
sbpmi <- ifelse(is.na(datasbp$SBP) == TRUE, mean(datasbp$SBP, na.rm = TRUE), datasbp$SBP)

n <- length(sbpmi)
mmi <- mean(sbpmi)
semi <- sd(sbpmi)/sqrt(n)
mmi; semi
```

```
## [1] 146.525
```

```
## [1] 4.394491
```

Note that in the above code the new variable, `sbpmi`, of length $n = 20$, is such that if the corresponding ith value ($i = 1, \ldots, n$) is missing (i.e., if `is.na(datasbp$SBP) == TRUE`), then such value will be imputed with the mean of the SBP variable (observed data on this variable), otherwise, the corresponding value is the observed value. We then compute the mean and standard error of the corresponding *completed* variable `sbpmi`.

(c) We will now use regression imputation, conditioning on sex and age, to impute the missing SBP values and consequently compute its mean and associated standard error. The regression model used is as follows

$$\text{SBP} = \beta_0 + \beta_1 \text{Age} + \beta_2 \text{Sex} + \varepsilon, \qquad \varepsilon \sim N(0, \sigma^2).$$

Here, gender is a dummy (binary) variable taking the value 0 if the subject is female (reference category) and 1 if the subject is male. This is in accordance with what `R` does. To check whether the variable is coded as a factor we do the following.

```
is.factor(datasbp$Sex)
```

```
## [1] TRUE
```

```
levels(datasbp$Sex)
```

```
## [1] "Female" "Male"
```

Factors in `R` are ordered alphabetically, and so female is chosen as the reference category.

The expected values for SBP are then

$$E(\text{SBP}) = \begin{cases} \beta_0 + \beta_1 \text{Age} & \text{if the subject is female,} \\ (\beta_0 + \beta_2) + \beta_1 \text{Age} & \text{if the subject is male.} \end{cases}$$

We fit the regression model to the complete cases, obtain estimated regression coefficients $\widehat{\beta}_0$, $\widehat{\beta}_1$, and $\widehat{\beta}_2$, and then we impute the SBP values based on the regression equation. The estimated regression coefficients are $\widehat{\beta}_0 = 40.8784$, $\widehat{\beta}_1 = 1.6518$, and $\widehat{\beta}_2 = 29.6318$. The imputed values (rounded to two decimal places) are then

Subject number	Sex	Age	Imputed SBP
6	Male	45	144.84
13	female	57	135.03
15	female	56	133.38
17	Female	48	120.17

We can easily fit such model in `R`.

```
fitsbp <- lm(SBP ~ Age + Sex, data = datasbp)
summary(fitsbp)
```

```
##
## Call:
## lm(formula = SBP ~ Age + Sex, data = datasbp)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -18.869  -5.987  -2.856   4.915  27.042
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  40.8784    29.5433   1.384  0.18976
## Age           1.6518     0.5718   2.889  0.01268 *
## SexMale      29.6318     7.2447   4.090  0.00128 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 13.34 on 13 degrees of freedom
##   (4 observations deleted due to missingness)
## Multiple R-squared:  0.6848, Adjusted R-squared:  0.6363
## F-statistic: 14.12 on 2 and 13 DF,  p-value: 0.0005504
```

```
#Two different ways of extracting the regression coefficients.
coef(fitsbp)
```

```
## (Intercept)         Age     SexMale
##   40.878357    1.651811   29.631845
```

```
fitsbp$coefficients
```

```
## (Intercept)         Age     SexMale
##   40.878357    1.651811   29.631845
```

To automatically compute the predicted values, we use the function `predict`. We just need to pass to this function our fitted regression model, `fitsbp`, and the new values to predict. In this case, for the sake of simplicity, we have passed the entire data set, but we are, obviously, only interested in predicting the `NA` values and so all the other values in such variable should be ignored (as they correspond to predictions for values that we actually observe).

```
predri <- predict(fitsbp, newdata = datasbp)
predri[6]; predri[13]; predri[15]; predri[17]
```

```
##        6
## 144.8417
```

```
##       13
## 135.0316
```

```
##       15
## 133.3798
```

```
##       17
## 120.1653
```

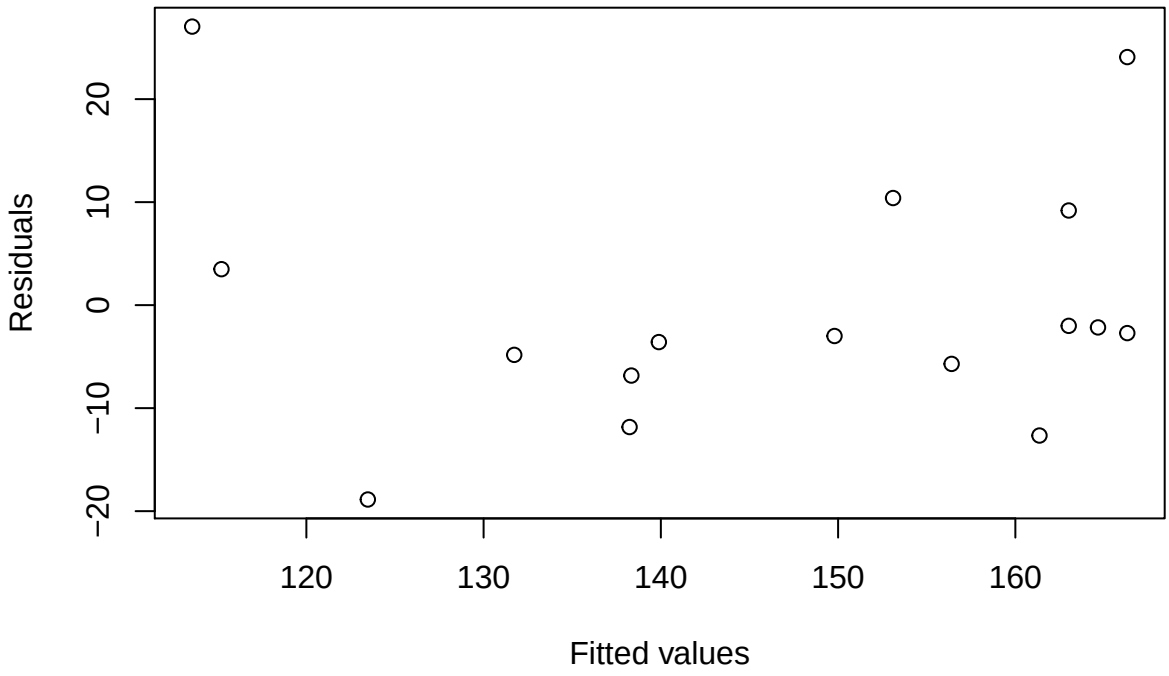
```
sbpri <- ifelse(is.na(datasbp$SBP) == TRUE, predri, datasbp$SBP)

mri <- mean(sbpri)
seri <- sd(sbpri)/sqrt(n)
mri; seri
```

```
## [1] 143.8909
```

```
## [1] 4.645933
```

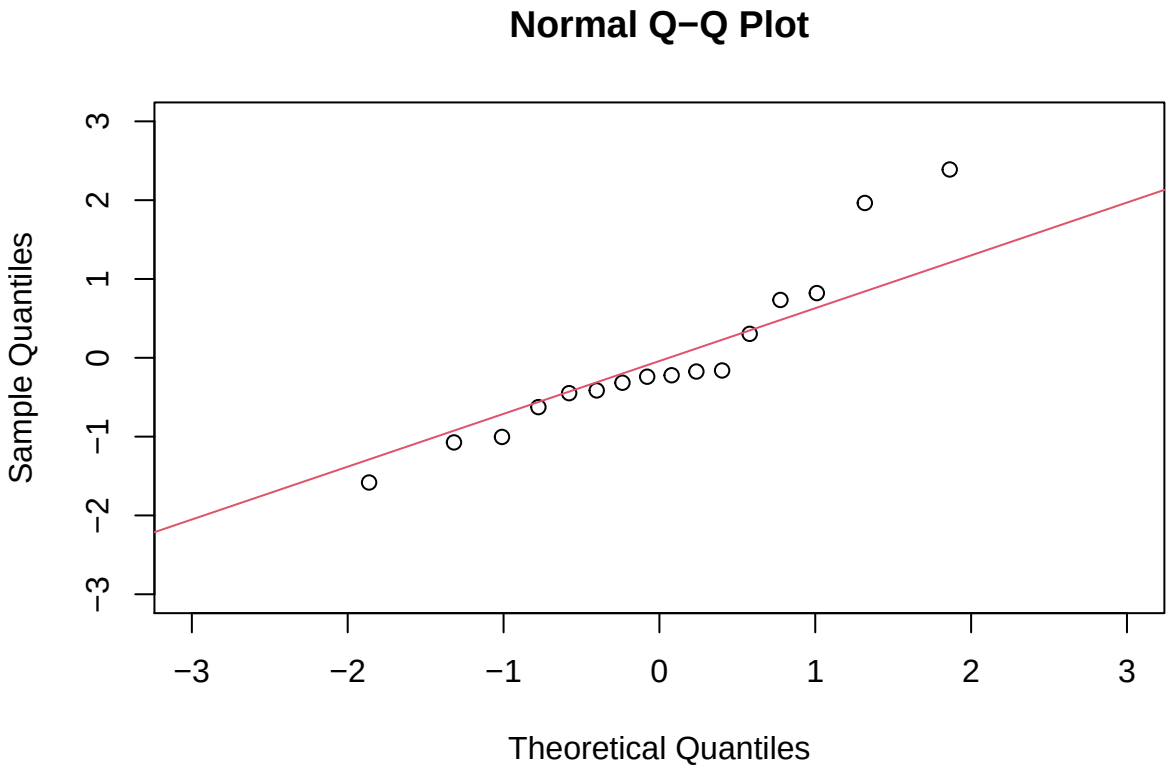
We should examine whether the linearity assumption is roughly met. We can informally check it using a plot of the fitted values against the residuals. If the assumption is met, no pattern should be visible.

```
plot(fitsbp$fitted.values, residuals(fitsbp), xlab = "Fitted values", ylab = "Residuals")
```



There is no obvious pattern, so there is no reason to suspect of a nonlinear relationship or of no constant variance. Although it is not of fundamental importance here in regression imputation, we can also check the assumption of normality of error terms using a QQ-plot.

```
qqnorm(rstandard(fitsbp), xlim = c(-3, 3), ylim = c(-3, 3))
qqline(rstandard(fitsbp), col = 2)
```



(d) We will now impute the SBP values based on stochastic regression imputation. The steps are essentially the same as those in regression imputation, but we add a random term, normally distributed with mean zero and variance equal to the estimated variance of the residuals, which in this case is $\widehat{\sigma}^2 = 13.34^2$. To the predictions we have obtained in (c) for the missing values, we now add a random normal noise $z_i \overset{\text{iid}}{\sim} \text{N}(0, \widehat{\sigma}^2)$. The imputed values (rounded to tow decimal places) in this case are

Subject number	Sex	Age	Imputed SBP
6	Male	45	133.90
13	female	57	126.75
15	female	56	148.38
17	Female	48	119.95

Of course, since we are adding a random term, different execution of the code will lead to different imputed values. The values obtained above were obtained using `set.seed(1)`. The estimated mean is 143.67 (4.71).

```
set.seed(1)
predsri <- predict(fitsbp, newdata = datasbp) + rnorm(n, 0, sigma(fitsbp))
predsri[6]; predsri[13]; predsri[15]; predsri[17]
```

```
##        6
## 133.8977
```

```
##       13
## 126.7451
```

```
##       15
## 148.3849
```

```
##       17
## 119.9493
```

```
sbpsri <- ifelse(is.na(datasbp$SBP) == TRUE, predsri, datasbp$SBP)
```

```r
msri <- mean(sbpsri)
sesri <- sd(sbpsri)/sqrt(n)
msri; sesri
```

```
## [1] 143.6689
```

```
## [1] 4.711592
```

(e) Finally, we apply the hot deck imputation method. We are told to define the strata based on gender and age (younger or 50 years old and older than 50 years old). For instance, we see that subject number 6 is male and is 45 years old. Thus, possible candidates to give him his SBP value are subjects number 1, 2, 7, and 8. We now just need to pick one of these subjects in a random fashion. The estimated mean SBP is 142.40 (4.809).

```r
indhdm1 <- which(is.na(datasbp$SBP) == FALSE & datasbp$Sex == "Male" & datasbp$Age <= 50)
indhdm1
```

```
## [1] 1 2 7 8
```

```r
donor6 <- sample(indhdm1, 1, replace = TRUE)
donor6
```

```
## [1] 2
```

```r
indhdf1 <- which(is.na(datasbp$SBP) == FALSE & datasbp$Sex == "Female" & datasbp$Age <= 50)
indhdf1
```

```
## [1] 14 16 18
```

```r
donor17 <- sample(indhdf1, 1, replace = TRUE)
donor17
```

```
## [1] 16
```

```r
indhdf2 <- which(is.na(datasbp$SBP) == FALSE & datasbp$Sex == "Female" & datasbp$Age > 50)
indhdf2
```

```
## [1] 19 20
```

```r
donor13 <- sample(indhdf2, 1, replace = TRUE)
donor15 <- sample(indhdf2, 1, replace = TRUE)
donor13; donor15
```

```
## [1] 19
```

```
## [1] 20
```

```r
#creating a new sbp variable with imputed values from the hot deck procedure
sbphd <- c(datasbp$SBP[is.na(datasbp$SBP) == FALSE],datasbp$SBP[donor6],datasbp$SBP[donor17],
        datasbp$SBP[donor13],datasbp$SBP[donor15])
mhd <- mean(sbphd); sehd <- sd(sbphd)/sqrt(n)
mhd; sehd
```

```
## [1] 142.395
```

```
## [1] 4.808816
```

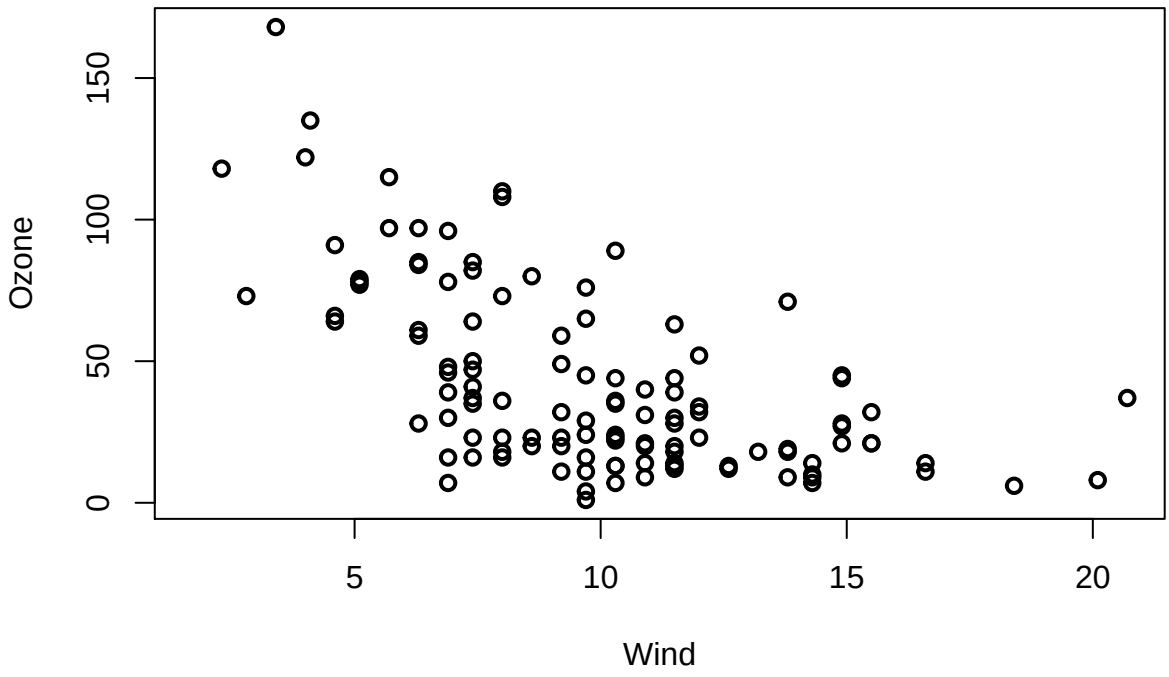
Cautionary note

The imputed values from regression imputation and stochastic regression imputation are only as good as the model used to impute them and so, as we have emphasised assumptions must be checked. If the model that we use to impute the values is poorly specified this can lead to invalid estimates of the target

parameters. Thus far, we have used normal linear regression models, which assume normality of the error terms, homoscedasticity (i.e., the variance of the error terms is constant or, equivalently, does not change with the covariates), independence of the error terms, and that the relationship between the response variable and the covariates is linear. All these assumptions can be checked informally using diagnostic plots (c.f. your favourite regression book. I like the following that is available online from the library: *Linear Models with R*, with the author being Julian Faraway).

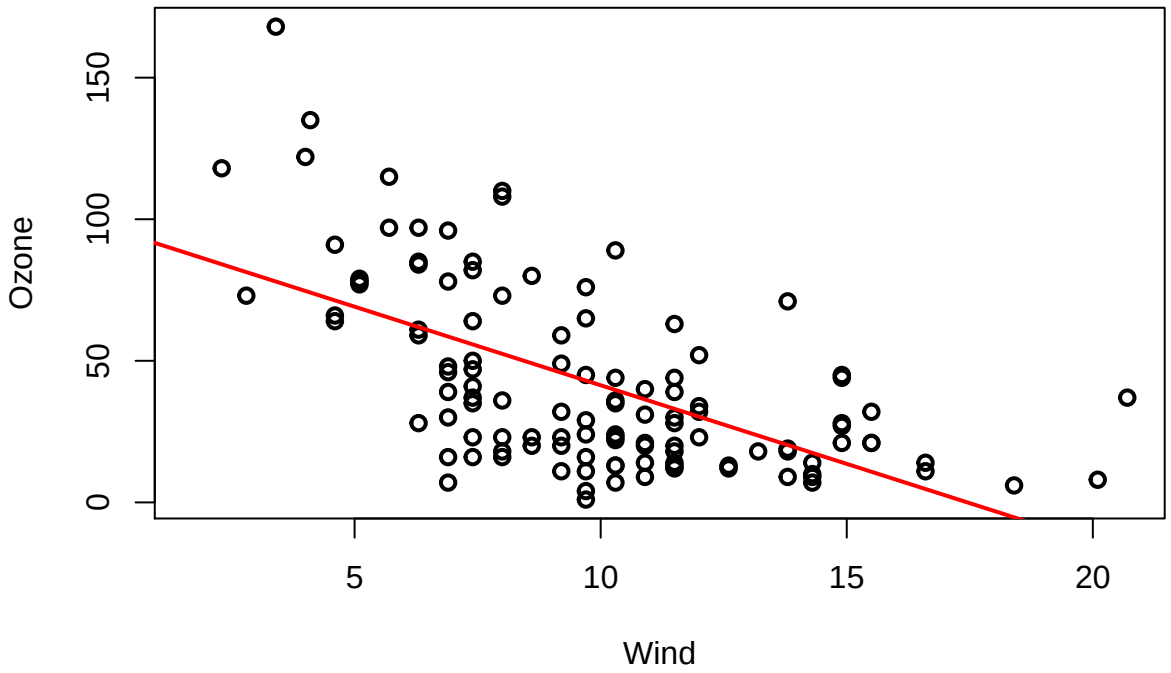
We give a specific example concerning the linearity of the relationship between the response and the covariates. Let us use the `airquality` data set again. Below we have a scatterplot of the variables wind and ozone (which, as we already know, has missing values, but this is not the point here).

```
plot(airquality$Wind, airquality$Ozone, xlab = "Wind", ylab = "Ozone", lwd = 2)
```



There is some 'curvature' in the data. Below we show the scatterplot of the data with the fitted regression line superimposed.

```
fit <- lm(Ozone ~ Wind, data = airquality)

plot(airquality$Wind, airquality$Ozone, xlab = "Wind", ylab = "Ozone", lwd = 2)
abline(fit, lwd = 2, col = "red")
```

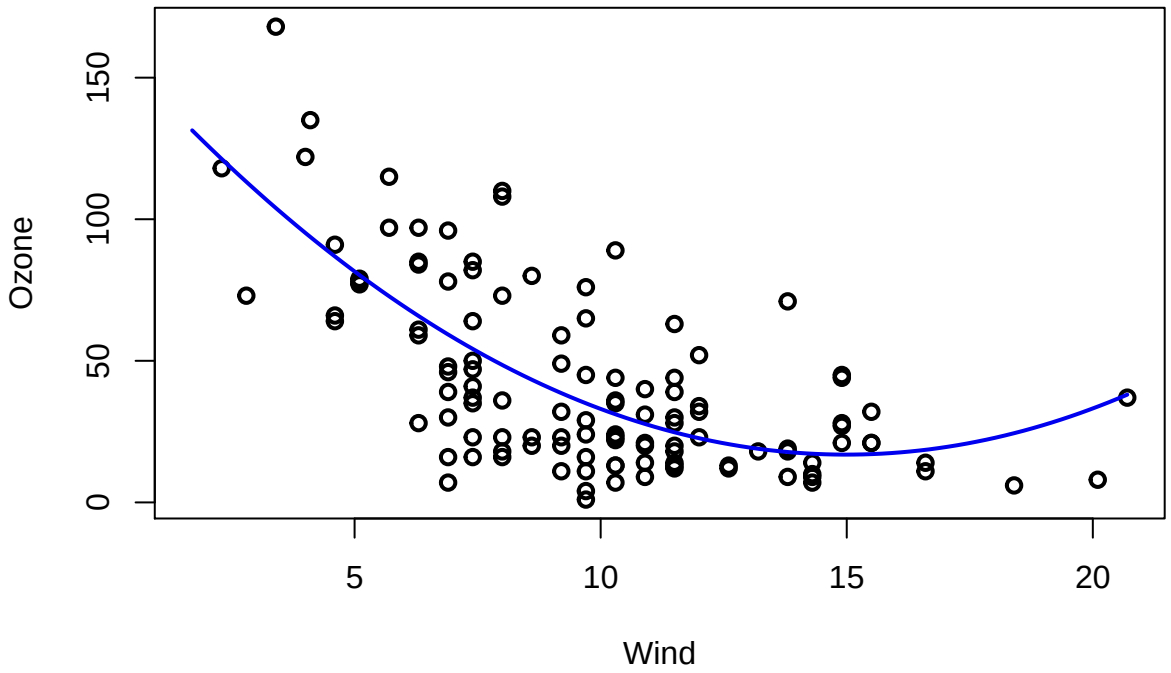


The fit is not terrible, but let us see if we can improve it. The nonlinear curvature might suggest the inclusion of a quadratic term. We have then fitted the model

$$\text{Ozone} = \beta_0 + \beta_1 \text{Wind} + \beta_2 \text{Wind}^2 + \epsilon, \qquad \epsilon \sim \text{N}(0, \sigma^2).$$

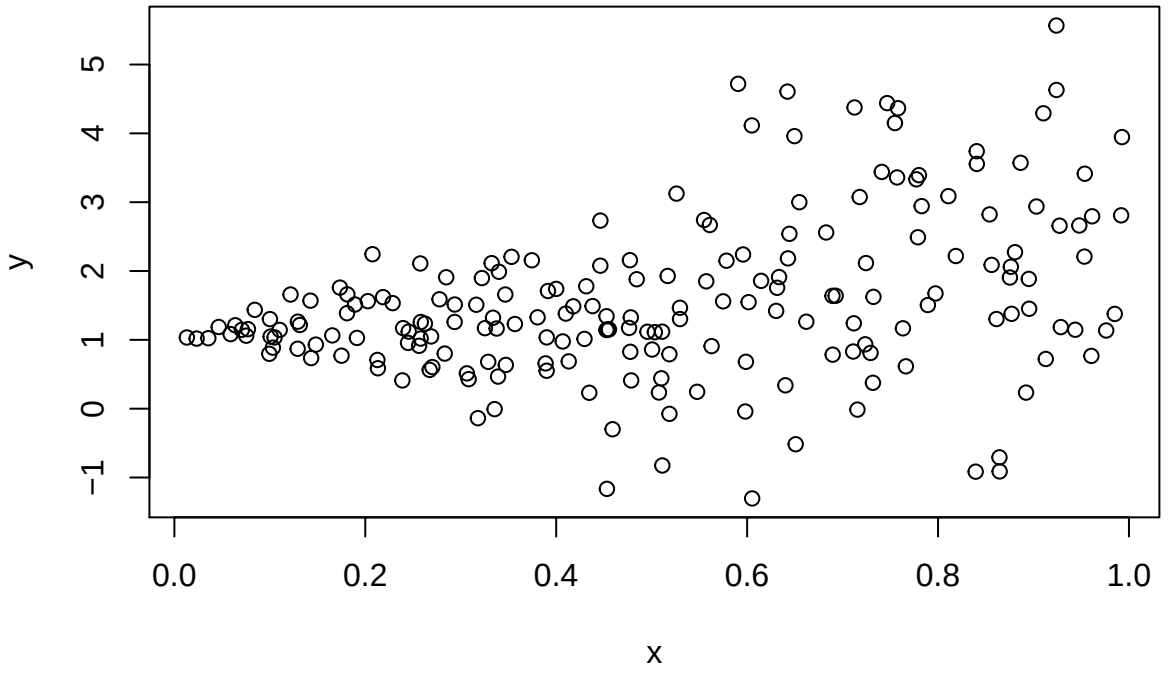
Below, we show the scatterplot of the data along with the quadratic fit, which seems better.

```
fit2 <- lm(Ozone ~ Wind + I(Wind^2), data = airquality)

gridwind <- data.frame("Wind" = seq(min(airquality$Wind), max(airquality$Wind), len = 100))
pred <- predict(fit2, newdata = gridwind)

plot(airquality$Wind, airquality$Ozone, lwd = 2, xlab = "Wind", ylab = "Ozone")
lines(seq(min(airquality$Wind), max(airquality$Wind), len = 100), pred,
      col = "blue2", lwd = 2)
```



The issue of nonlinear relationships between the response and the covariates would lead us to the topic of nonparametric regression, which out of the scope of this course. To conclude, I will simulate one data set from a normal linear model where the variance is not constant and we will check how the scatter plot looks like.

```
n <- 200
x <- runif(n)
y <- rnorm(n, 1 + x, 2*x)
plot(x, y)
```



From the scatterplot above we clearly see that the dispersion increases as x increases, a clear sign of heteroscedasticity (and, of course, we know it should be like this because we have simulated the data in that way). There are several ways of dealing with heteroscedasticity (e.g., transformations of the response and/or

covariates, modelling the variance function explicitly, weighted least squares, etc) but again this is out of the scope of this course. However, later when learning about multiple imputation, we will see software and imputation methods that can help us in these situations.