University of Edinburgh
School of Mathematics
Incomplete Data Analysis

Supporting Materials for Lecture 4

Systolic blood pressure example

We start with the example on slide 2. I first load the data by doing

```
load("datasbp.Rdata")
```

(i) We are first asked to compute the overall mean of the systolic blood presure variable (and its associated standard error) using a complete case analysis. In R we just need to do

```
ind=which(datasbp$R==1)
mccoverall=mean(datasbp$SBP,na.rm=TRUE)
seccoverall=sd(datasbp$SBP,na.rm=TRUE)/sqrt(length(ind))
mccoverall; seccoverall

## [1] 146.525
## [1] 5.529613
```

We now, as asked in the question, compute the mean of this variable separately for men and women.

```
#cc analysis women
indf=which(datasbp$R==1 & datasbp$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(datasbp$R==1 & datasbp$Sex=="Male")
mccm=mean(datasbp$SBP[indm])
seccm=sd(datasbp$SBP[indm])/sqrt(length(indm))
mccm; seccm

## [1] 156.5545
## [1] 5.269076
```

(ii) We do the same as in the first part of (i) but using mean imputation.

```
n=nrow(datasbp)
sbpmi=numeric(n)
for(i in 1:n){
sbpmi[i]=ifelse(datasbp$R[i]==0,mean(datasbp$SBP,na.rm=TRUE),datasbp$SBP[i])
}

mmi=mean(sbpmi)
semi=sd(sbpmi)/sqrt(n)
mmi; semi


## [1] 146.525
## [1] 4.394491
```

Note that in the above code I created a new variable, `sbpmi`, of length $n = 20$, such that if the corresponding ith value ($i = 1, \ldots, n$) is missing (i.e., if $R_i = 0$), then such value will be imputed with the mean of the SBP variable, otherwise, the correspoding value is the observed value. We then compute the mean and standard error of the corresponding complete variable `sbpmi`.

(iii) We repeat the same but now the missing values are filled-in using regression imputation. For the 16 complete cases, we fit a regression model with `SBP` as our response variable and `age` and `gender` as our explanatory variables/covariates. We can easily fit such model in `R` using the `lm` function.

```
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
```

Using the notation of slide 5, we have that $\widehat{\beta}_0 = 40.8784$, $\widehat{\beta}_1 = 1.6518$, and $\widehat{\beta}_2 = 29.6318$. We can calculate 'by hand' the imputed values under this approach, by using the equation in slide 6. If we do that, we obtain that the imputed value for subject 6 is 144.84, for subject 13 is 135.03, for subject 15 is 133.38, and for subject 17 is 120.17. All values were rounded to two decimal places. We can do this

more automatically by using the function `predict` of `R` 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 whole dataset, 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
```

We can now apply a procedure similar to the one in (ii).

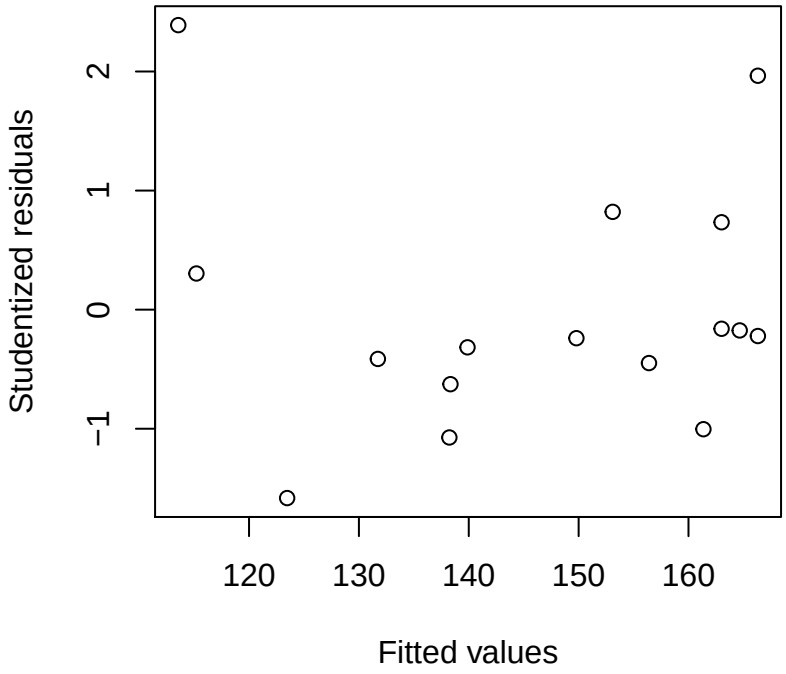
```
sbpri=numeric(n)
for(i in 1:n){
sbpri[i]=ifelse(datasbp$R[i]==0,predri[i],datasbp$SBP[i])
}

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

```
## [1] 143.8909
## [1] 4.645933
```

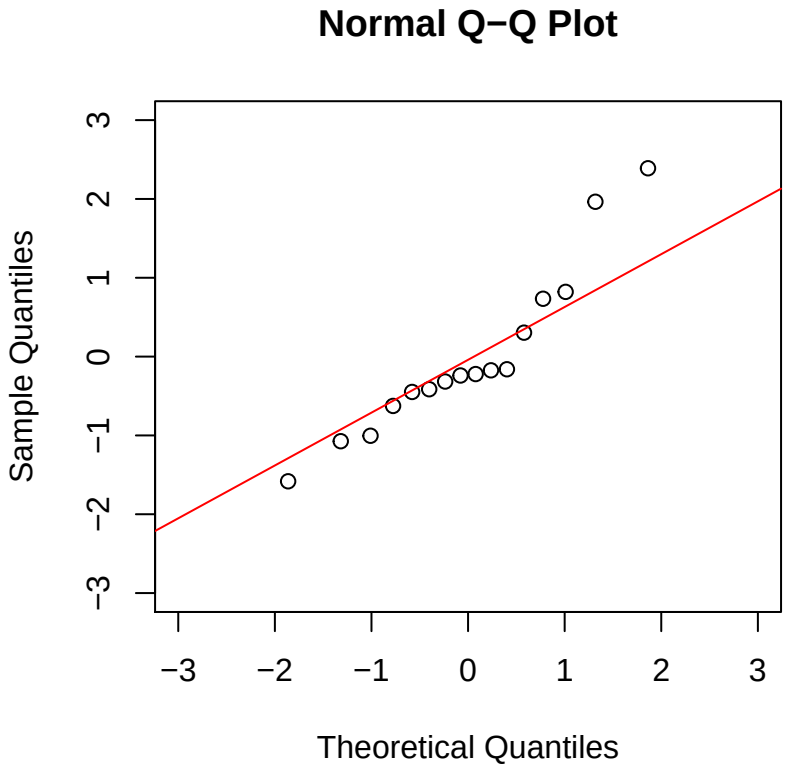
We should examine whether the linearity assumption is roughly met. We can informally check it using a plot of the fitted values against the residuals. If the assumption is met, no pattern should be visible.

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



There is no obvious pattern, so there is no reason to suspect of a nonlinear relationship. 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(stdres,xlim=c(-3,3),ylim=c(-3,3))
qqline(stdres,col=2)
```



(iv) In this part of the question, the values will be imputed using stochastic regression imputation. From the output above, we know that $\widehat{\sigma}^2 = 13.34^2$. To the predictions we have obtained in (iii) for the missing values, we now add a random normal noise $z_i \overset{\text{iid}}{\sim} \text{N}(0, \widehat{\sigma}^2)$.

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


##        6
## 133.8977
##       13
## 126.7451
##       15
## 148.3849
##       17
## 119.9493
```

we now apply a similar procedure to the ones in (iii) and (iv).

```
sbpsri=numeric(n)
for(i in 1:n){
sbpsri[i]=ifelse(datasbp$R[i]==0,predsri[i],datasbp$SBP[i])
}
```

```
msri=mean(sbpsri); sesri=sd(sbpsri)/sqrt(n)
msri; sesri


## [1] 143.6689
## [1] 4.711592
```

(v) 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). From the four subjects with missing values, one is male and younger than 50 years old (45!) and the other three are female, two older than 50 and one younger. We start by looking for the possible donors (of values) and then randomly sample a SBP value from the set of possible donors.

```
indhdm1=which(datasbp$R==1 & datasbp$Sex=="Male" & datasbp$Age<=50)
indhdm1

## [1] 1 2 7 8

donor6=sample(indhdm1,1,replace=TRUE)
donor6

## [1] 8

indhdf1=which(datasbp$R==1 & datasbp$Sex=="Female" & datasbp$Age<=50)
indhdf1

## [1] 14 16 18

donor17=sample(indhdf1,1,replace=TRUE)
donor17

## [1] 16

indhdf2=which(datasbp$R==1 & datasbp$Sex=="Female" & datasbp$Age>50)
indhdf2

## [1] 19 20

donor13=sample(indhdf2,1,replace=TRUE)
donor15=sample(indhdf2,1,replace=TRUE)
donor13; donor15

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

sbphd=c(datasbp$SBP[datasbp$R==1],datasbp$SBP[donor6],datasbp$SBP[donor17],
        datasbp$SBP[donor13],datasbp$SBP[donor15])
mhd=mean(sbphd); sehd=sd(sbphd)/sqrt(n)
mhd; sehd

## [1] 143.185
## [1] 4.773795
```

Simulation study

Here, I show how to reproduce the table in page 16. I started by cleaning the workspace, fixing the seed and loading the package `MASS` (which will be needed to simulate from the bivariate normal distribution).

```
rm(list=ls())
set.seed(1)
require(MASS)
```

Next, I've set the parameters for the population values and simulated `nsim=1000` datasets, each with a sample size of `n=200`. For the sake of simplicity, I will work afterwards with the two variables separately.

```
mu1=0; mu2=0; sigma12=1; sigma22=1; rho=0.5
cov12=rho*sqrt(sigma12)*sqrt(sigma22)
Sigma=matrix(c(sigma12,cov12,cov12,sigma22),nrow=2,ncol=2,byrow=TRUE)

nsim=1000; n=200
y=array(0,c(n,2,nsim))
for(i in 1:nsim){
y[,,i]=mvrnorm(n=n,mu=c(mu1,mu2),Sigma=Sigma)
}
y1=y2=matrix(0,nrow=n,ncol=nsim)
for(i in 1:nsim){
y1[,i]=y[,1,i]
y2[,i]=y[,2,i]
}
```

The next step is to generate the missing data indicator. In this case, we will impose MCAR missing data by assuming $\Pr(R_2 = 1 \mid Y_1, Y_2) = 0.5$. Thus, we simply need to do the following in `R`

```
r=matrix(0,nrow=n,ncol=nsim); prob=0.5
for(i in 1:nsim){
r[,i]=rbinom(n,1,prob)
}
```

I will create now the vector `y2mcar`, that will replace by a `NA` each value for which we have a missing indicator (generated above) of zero. The vector `y1mcar` will only be used for the complete case analysis (where estimates on both variables are only conducted for the cases with fully observed variables).

```
y2mcar=matrix(0,nrow=n,ncol=nsim)
y1mcar=matrix(0,nrow=n,ncol=nsim)
for(i in 1:nsim){
for(j in 1:n){
y2mcar[j,i]=ifelse(r[j,i]==1,y2[j,i],NA)
y1mcar[j,i]=ifelse(r[j,i]==1,y1[j,i],NA)
}
}
```

We can now estimate each parameter using a complete case analysis.

```
#complete case analysis
estimatescc=matrix(0,nrow=5,ncol=nsim)
for(i in 1:nsim){
estimatescc[1,i]=mean(y1mcar[,i],na.rm=TRUE)
estimatescc[2,i]=mean(y2mcar[,i],na.rm=TRUE)
estimatescc[3,i]=var(y1mcar[,i],na.rm=TRUE)
estimatescc[4,i]=var(y2mcar[,i],na.rm=TRUE)
estimatescc[5,i]=cor(y1mcar[,i],y2mcar[,i],use="complete.obs")
}
```

At the end of this step, we have as many estimates of the parameters as simulated datasets (1000 in this case). We can now compute the mean across all estimates (we could additionally also compute the standard deviation of each set of estimates).

```
#complete case analysis
mu1mcarcc=mean(estimatescc[1,])
mu2mcarcc=mean(estimatescc[2,])
var1mcarcc=mean(estimatescc[3,])
var2mcarcc=mean(estimatescc[4,])
cormcarcc=mean(estimatescc[5,])
mu1mcarcc; mu2mcarcc; var1mcarcc; var2mcarcc; cormcarcc

## [1] -0.005565697
## [1] 0.0006887913
## [1] 0.9991581
## [1] 1.001576
## [1] 0.4961997
```

We will now evaluate the performance of mean imputation. We will apply a similar 'trick' to the one used in the systolic blood pressure example to construct the complete Y_2 variables.

```
#complete case analysis
y2mcarmi=matrix(0,nrow=n,ncol=nsim)
for(i in 1:nsim){
for(j in 1:n){
y2mcarmi[j,i]=ifelse(r[j,i]==0,mean(y2mcar[,i],na.rm=TRUE),y2mcar[j,i])
}
}

estimatesmi=matrix(0,nrow=5,ncol=nsim)
for(i in 1:nsim){
estimatesmi[1,i]=mean(y1[,i])
estimatesmi[2,i]=mean(y2mcarmi[,i])
estimatesmi[3,i]=var(y1[,i])
estimatesmi[4,i]=var(y2mcarmi[,i])
estimatesmi[5,i]=cor(y1[,i],y2mcarmi[,i])
}

mu1mcarmi=mean(estimatesmi[1,])
mu2mcarmi=mean(estimatesmi[2,])
var1mcarmi=mean(estimatesmi[3,])
```

```
var2mcarmi=mean(estimatesmi[4,])
cormcarmi=mean(estimatesmi[5,])
mu1mcarmi; mu2mcarmi; var1mcarmi; var2mcarmi; cormcarmi

## [1] -0.002886723
## [1] 0.0006887913
## [1] 1.000584
## [1] 0.498599
## [1] 0.3499275
```

We now turn our attention to the regression imputation case. For each simulated dataset, we will perform a linear regression of Y_2 on Y_1.

```
#complete case analysis
predri=matrix(0,nrow=n,ncol=nsim)
for(i in 1:nsim){
fit=lm(y2mcar[,i]~y1mcar[,i])
predri[,i]=fit$coefficients[1]+y1[,i]*fit$coefficients[2]
}

y2mcarri=matrix(0,nrow=n,ncol=nsim)
for(i in 1:nsim){
for(j in 1:n){
y2mcarri[j,i]=ifelse(r[j,i]==0,predri[j,i],y2mcar[j,i])
}
}

estimatesri=matrix(0,nrow=5,ncol=nsim)
for(i in 1:nsim){
estimatesri[1,i]=mean(y1[,i])
estimatesri[2,i]=mean(y2mcarri[,i])
estimatesri[3,i]=var(y1[,i])
estimatesri[4,i]=var(y2mcarri[,i])
estimatesri[5,i]=cor(y1[,i],y2mcarri[,i])
}

mu1mcarri=mean(estimatesri[1,])
mu2mcarri=mean(estimatesri[2,])
var1mcarri=mean(estimatesri[3,])
var2mcarri=mean(estimatesri[4,])
cormcarri=mean(estimatesri[5,])

mu1mcarri; mu2mcarri; var1mcarri; var2mcarri; cormcarri

## [1] -0.002886723
## [1] 0.002069861
## [1] 1.000584
## [1] 0.6269647
## [1] 0.6278931
```

Finally, we will use the stochastic regression imputation method.

```r
#complete case analysis
predsri=matrix(0,nrow=n,ncol=nsim)
for(i in 1:nsim){
fit=lm(y2mcar[,i]~y1mcar[,i])
sigmaest=summary(fit)$sigma
predsri[,i]=fit$coefficients[1]+y1[,i]*fit$coefficients[2]+rnorm(n,0,sigmaest)
}

y2mcarsri=matrix(0,nrow=n,ncol=nsim)
for(i in 1:nsim){
for(j in 1:n){
y2mcarsri[j,i]=ifelse(r[j,i]==0,predsri[j,i],y2mcar[j,i])
}
}

estimatessri=matrix(0,nrow=5,ncol=nsim)
for(i in 1:nsim){
estimatessri[1,i]=mean(y1[,i])
estimatessri[2,i]=mean(y2mcarsri[,i])
estimatessri[3,i]=var(y1[,i])
estimatessri[4,i]=var(y2mcarsri[,i])
estimatessri[5,i]=cor(y1[,i],y2mcarsri[,i])
}

mu1mcarsri=mean(estimatessri[1,])
mu2mcarsri=mean(estimatessri[2,])
var1mcarsri=mean(estimatessri[3,])
var2mcarsri=mean(estimatessri[4,])
cormcarsri=mean(estimatessri[5,])

mu1mcarsri; mu2mcarsri; var1mcarsri; var2mcarsri; cormcarsri

## [1] -0.002886723
## [1] -0.0004515947
## [1] 1.000584
## [1] 1.003591
## [1] 0.4971326
```

Tables on slides 17 and 18 are obtained exactly in the same way once you have imposed the missingness in the way you were told.