IDA Assignment 3

Johnny Lee, s1687781

8th April 2022

Q1.

Consider the `nhanes` dataset in mice. For more information please type help(`nhanes`) in the R console.

a)

(2 marks) What percentage of the cases is incomplete?

Answer :

```
cat("The percentage of incomplete cases is",
    (nrow(nhanes)-nrow(cc(nhanes)))*100/nrow(nhanes))
```

```
## The percentage of incomplete cases is 48
```

b)

(4 marks) Impute the data with `mice` using the defaults with `seed=1`, in step 2 predict `bmi` from `age`, `hyp`, and `chl` by the normal linear regression model, and then pool the results. What are the proportions of variance due to the missing data for each parameter? Which parameters appear to be most affected by the nonresponse?

Answer :

```
pool1 <- pool(with(mice(nhanes, printFlag = F, seed = 1),
                   lm(bmi ~ age + hyp + chl)))
kable(pool1$pooled[,c(1,3,7,8,10)], caption = "Imputation with seed=1") %>%
  kable_styling(latex_options = "hold_position")
```

Table 1: Imputation with seed=1

term	estimate	dfcom	df	lambda
(Intercept)	19.6178925	21	16.936189	0.0893899
age	-3.5528716	21	3.528053	0.6864064
hyp	2.1970175	21	9.035495	0.3504345
chl	0.0537808	21	10.228828	0.3040806

From the lecture notes, we know that to estimate the variance of $\hat{\theta}^{\text{MI}}$, we need to compute the between-imputation variance, $\mathbf{B}$ and within-imputation variance, $\bar{\mathbf{U}}$.

$$\text{B} = \frac{1}{M-1}\sum_{i=1}^{M}(\hat{\theta}^{(i)} - \hat{\theta}^{\text{MI}})^2 \bar{\text{U}} = \frac{1}{M}\sum_{i=1}^{m}\hat{\text{U}}^{(i)}$$

With the above statistics, we can calculate the total variance, $V^{\text{MI}} = \bar{U} + (1 + \frac{1}{M})B$. Since `mice()` provides a different ratio known as λ where $\lambda = \frac{B+\frac{B}{N}}{V^{\text{MI}}}$. In this setup, `bmi` is the θ.

From Table 1, we computed the pooled estimates, degree of freedom and lambda values. This `lambda` represents the proportions of variance due to the missing data for each parameter. We conclude that `age` is mostly affected by the nonresponse.

c)

(4 marks) Repeat the analysis for `seed` $\in \{2, 3, 4, 5, 6\}$. Do the conclusions remain the same?

Answer :

```
#Compute pooled estimates for each seed
pool2 <- pool(with(mice(nhanes, printFlag = F, seed = 2), lm(bmi ~ age + hyp + chl)))
pool3 <- pool(with(mice(nhanes, printFlag = F, seed = 3), lm(bmi ~ age + hyp + chl)))
pool4 <- pool(with(mice(nhanes, printFlag = F, seed = 4), lm(bmi ~ age + hyp + chl)))
pool5 <- pool(with(mice(nhanes, printFlag = F, seed = 5), lm(bmi ~ age + hyp + chl)))
pool6 <- pool(with(mice(nhanes, printFlag = F, seed = 6), lm(bmi ~ age + hyp + chl)))
#tabulating the values we obtained from mice()
parameters <- c("(Intercept)", "age", "hyp", "chl")
df <- data.frame(parameters, pool2$pooled[,10], pool3$pooled[,10],
                 pool4$pooled[,10], pool5$pooled[,10], pool6$pooled[,10])
colnames(df) <- c("parameters", "seed=2", "seed=3", "seed=4", "seed=5", "seed=6")
kable(df, caption = "Imputation with seed=2,3,4,5,6") %>%
  kable_styling(latex_options = "hold_position")
```

Table 2: Imputation with seed=2,3,4,5,6

parameters	seed=2	seed=3	seed=4	seed=5	seed=6
(Intercept)	0.4144454	0.2772900	0.1315114	0.4855733	0.4168136
age	0.4033924	0.5895051	0.2189333	0.4511896	0.6549523
hyp	0.1430995	0.4101152	0.1961083	0.5942866	0.2960364
chl	0.2959966	0.5621346	0.3305334	0.2346065	0.5196295

Here we repeated the same step as Q1 b) with different seed. From Table 2, we can see that the most affected variables will change with different number of settings. For instance, in `seed=5`, `hyp` has the highest value of lambda whereas `seed=2,3 and 6` have `age` as the most affected variable. This is unstable throughout different seed because there percentage of imputation we found in Q1 a) is considerably high with 48%.

d)

(4 marks) Repeat the analysis with $M = 100$ with the same seeds. Would you prefer these analyses over those with $M = 5$? Explain why.

Answer :

```
#compute the pooled estimates from each seed with M=100
pool1 <- pool(with(mice(nhanes, m = 100, printFlag = F, seed = 1),
                   lm(bmi ~ age + hyp + chl)))
pool2 <- pool(with(mice(nhanes, m = 100, printFlag = F, seed = 2),
                   lm(bmi ~ age + hyp + chl)))
pool3 <- pool(with(mice(nhanes, m = 100, printFlag = F, seed = 3),
                   lm(bmi ~ age + hyp + chl)))
pool4 <- pool(with(mice(nhanes, m = 100, printFlag = F, seed = 4),
                   lm(bmi ~ age + hyp + chl)))
pool5 <- pool(with(mice(nhanes, m = 100, printFlag = F, seed = 5),
                   lm(bmi ~ age + hyp + chl)))
pool6 <- pool(with(mice(nhanes, m = 100, printFlag = F, seed = 6),
                   lm(bmi ~ age + hyp + chl)))
#tabulating the values from mice()
parameters <- c("(Intercept)", "age", "hyp", "chl")
df <- data.frame(parameters, pool1$pooled[,10], pool2$pooled[,10],
                 pool3$pooled[,10], pool4$pooled[,10],
                 pool5$pooled[,10], pool6$pooled[,10])
colnames(df) <- c("parameters", "seed=1", "seed=2", "seed=3", "seed=4",
                  "seed=5", "seed=6")
kable(df, caption="Imputation with seed=1,2,3,4,5,6 and M=100") %>%
  kable_styling(latex_options = "hold_position")
```

Table 3: Imputation with seed=1,2,3,4,5,6 and M=100

parameters	seed=1	seed=2	seed=3	seed=4	seed=5	seed=6
(Intercept)	0.2290445	0.1882474	0.2199607	0.2144722	0.2294356	0.2472607
age	0.4324680	0.4031077	0.3093072	0.3943223	0.3322570	0.4430300
hyp	0.2915346	0.2825108	0.2425105	0.2565132	0.2893046	0.2860700
chl	0.3217837	0.2939693	0.3281911	0.2835232	0.2461956	0.3113085

In Table 3, we computed the lambda and we can notice that `lambda` is more stable when the value of M is higher. Thus, we would prefer $M = 100$ than $M = 5$. This is also evident in the equation where, if M is high, we can reduce the values for $\bar{U}$ and B and the total variance, V^{MI}. $M = 100$ is also a reasonable choice for the size of this dataset as the number of observation is 25. Hence, the time efficiency will not accounted for.

Q2.

(15 marks) Each of the 100 datasets contained in the file dataex2.Rdata was generated in the following way

$$y_i|x_i \overset{\text{ind.}}{\sim} \text{N}(\beta_0 + \beta_1 x_i, 1), \quad x_i \overset{\text{ind.}}{\sim} \text{Unif}(-1, 1), \quad \beta_0 = 1, \quad \beta_1 = 3$$

for $i = 1, \ldots, 100$. Additionally, some of the responses were set to be missing using a MAR mechanism. The goal of this exercise is to study the effect that acknowledging/not acknowledging parameter uncertainty when performing step 1 of multiple imputation might have on the coverage of the corresponding confidence intervals. Further suppose that the analysis of interest in step 2 is to fit the regression model that was used to generate the data, i.e., a normal linear regression model where the response is y and the covariate is x. With the aid of the `mice` package, calculate the empirical coverage probability of the 95% confidence intervals for β_1 under the following two approaches: stochastic regression imputation and the corresponding bootstrap based version. Comment. For both approaches, please consider $m = 20$ and `seed=1`. **NOTE 1:** In order to calculate the empirical coverage probability, you only need to compute the proportion of the time (over the 100 intervals) that the interval contains the true value of the parameter. **NOTE 2:** The data are stored in an array structure such that, for instance, `dataex2[, , 1]`, corresponds to the first dataset (which has 100 rows and 2 columns, with the first column containing the values of x and the second the values of y).

Answer :

```
count <- 0 # initialize a counter
for (i in 1:nrow(dataex2)) {
  #perform SRI with m=20
  impute.sri <- mice(dataex2[, , i], m = 20, method = "norm.nob", printFlag = F, seed = 1)
  fit.sri <- with(impute.sri, lm(Y ~ X)) #step 2
  pool.sri <- pool(fit.sri) # step 3
  summary.sri <- summary(pool.sri, conf.int = TRUE)
  if (summary.sri[2, 7] <= 3 & summary.sri[2, 8] >= 3) {
  #increment the count if beta1 is contained in the confidence interval
    count <- count + 1
  }
}
cat("the empirical coverage probability for Stochastic Imputation is", count/nrow(dataex2))
```

```
## the empirical coverage probability for Stochastic Imputation is 0.88
```

```
# initialize a counter
count <- 0
for (i in 1:nrow(dataex2)) {
  #perform bootstrap imputation with m=20
  impute.bootstrap <- mice(dataex2[,,i], m = 20, method = "norm.boot",
                           printFlag = FALSE, seed = 1)
  fit.bootstrap <- with(impute.bootstrap, lm(Y ~ X)) #step 2
  pool.bootstrap <- pool(fit.bootstrap) # step 3
  summary.bootstrap <- summary(pool.bootstrap, conf.int = TRUE)
  if (summary.bootstrap[2, c(7)] <= 3 & summary.bootstrap[2, c(8)] >= 3) {
    #increment the count if beta1 is contained in the confidence interval
    count = count + 1
  }
}
cat("the empirical coverage probability for Bootstrap is", count/nrow(dataex2))
```

```
## the empirical coverage probability for Bootstrap is 0.95
```

We performed both stochastic imputation and bootstrap imputation to compute the empirical coverage

probability. As a result we obtained 0.88 and 0.95 respectively. The reason behind this is that SRI does not take the variability of the function weight into account. This can result in the missing consideration on the uncertainty of the imputed values. Thus, we will have an improper multiple imputation and the confidence intervals based on the total variance can be narrow. For the bootstrap imputation, it is the proper multiple imputation and thus the confidence interval will be wider.

Q3.

(9 marks) Show that for a linear (in the coefficients) regression model, the following two strategies coincide:

(i) Computing the predicted values (point estimates) from each fitted model in step 2 and then pooling them according to Rubin's rule for point estimates (i.e., averaging the predicted values across the imputed datasets).

(ii) Pooling the regression coefficients from each fitted model in step 2 using Rubin's rule for point estimates and then computing the predicted values afterwards

Answer :

We consider a linear regression model given a dataset as $\{y_i, x_{1i}, \ldots, x_{ni}\}$

$$y_i = \beta_0 + \beta_1 x_{1i} + \cdots + \beta_n x_{ni} + \varepsilon_i, \quad \varepsilon_i \sim N(0, \sigma^2), \quad i = 1, \ldots, n$$

Now we look into Case (i), we compute the predicted values for each fitted model from step 2. Then we obtain as below,

$$\hat{y}_i^{(m)} = \hat{\beta}_0^{(m)} + \hat{\beta}_1^{(m)} x_{1i} + \cdots + \hat{\beta}_n^{(m)} x_{ni}, \quad i = 1, \ldots, n$$

Then we pool them according to Rubin's rule for point estimates.

$$\begin{aligned}
\bar{y}_i &= \frac{1}{M} \sum_{i=1}^{M} \hat{y}^{(m)} \\
&= \frac{1}{M} \sum_{i=1}^{M} \left(\hat{\beta}_0^{(m)} + \hat{\beta}_1^{(m)} x_{1i} + \cdots + \hat{\beta}_n^{(m)} x_{ni} \right) \\
&= \frac{1}{M} \sum_{i=1}^{M} \hat{\beta}_0^{(m)} + \frac{1}{M} \sum_{i=1}^{M} \hat{\beta}_1^{(m)} x_{1i} + \cdots + \frac{1}{M} \sum_{i=1}^{M} \hat{\beta}_n^{(m)} x_{ni} \\
&= \bar{\beta}_0 + \bar{\beta}_1 x_{1i} + \cdots + \bar{\beta}_n x_{ni}, \quad i = 1, \ldots, n
\end{aligned}$$

Now, let us consider Case (ii) to validate if they coincide. We pool the regression coefficients from each fitted model in step 2 using Rubin's rule for point estimates.

$$\begin{aligned}
\bar{\beta}_0 &= \frac{1}{M} \sum_{i=1}^{M} \hat{\beta}_0^{(m)} \\
&\vdots \\
\bar{\beta}_n &= \frac{1}{M} \sum_{i=1}^{M} \hat{\beta}_n^{(m)}
\end{aligned}$$

Then we can compute the predicted values as follow

$$\bar{y}_i = \bar{\beta}_0 + \bar{\beta}_1 x_{1i} + \cdots + \bar{\beta}_n x_{ni}, \quad i = 1, \ldots, n$$

As shown above, the order of the computation of predicted values for each fitted model in step 2 and pooling according to Rubin's rule for point estimates do not matter mathematically. Therefore, we conclude here by saying that both cases coincide.

Q4.

The goal of this exercise is to study different ways of using `mice` when the analysis model of interest/substantive model involves an interaction term between incomplete variables. The model used to generate the data (available in `dataex4.Rdata`), which corresponds to our model of interest in step 2, was the following one:

$$y_i = \beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} + \beta_3 x_{1i} x_{2i} + \varepsilon_i, x_{1i} \overset{\text{iid}}{\sim} \text{N}(0,1), \quad x_{2i} \overset{\text{iid}}{\sim} \text{N}(1.5,1), \quad \varepsilon_i \overset{\text{iid}}{\sim} \text{N}(0,1)$$

for $i = 1, \ldots, 1000$, $\beta_0 = 1.5$, $\beta_1 = 1$, $\beta_2 = 2$ and $\beta_3 = 1$. Additionally, missingness was imposed on y and x_1 and so the interaction variable $x_1 x_2$ also has missing values, although the missingness in this interaction variable is induced by the missing in the covariate x_1. In the following, please use $M = 50$ and `seed=1`.

```
kable(head(dataex4), caption = "first 6 values of dataex4")  %>%
  kable_styling(latex_options = "hold_position")
```

Table 4: first 6 values of dataex4

y	x1	x2
NA	NA	1.0983213
4.609476	0.0265944	1.1290673
NA	-1.5165531	1.0748538
1.718231	-1.3626533	1.9411515
NA	1.1784892	0.8272496
1.820674	-0.9341513	1.9166941

a)

(6 marks) By only imputing the y and x_1 variables in step 1, provide the estimates of β_1, β_2, and β_3 along with 95% confidence intervals. Comment. Note that this approach where the interaction variable is left outside the imputation process and calculated afterwards in the analysis model, is known as *Impute, then transform*

Answer :

```
#perform sri with m=50
impute.sri <- mice(dataex4, m = 50, seed = 1, printFlag = FALSE)
fit.sri <- with(impute.sri, lm(y ~ x1 + x2 + x1*x2)) #step 2
pool.sri <- pool(fit.sri) #step 3
kable(summary(pool.sri, conf.int = TRUE)[, c(1,2,3,7,8)],
      caption = "Summary Statistics of Imputation of $y$ and $x_1$") %>%
  kable_styling(latex_options = "hold_position")
```

Table 5: Summary Statistics of Imputation of y and x_1

term	estimate	std.error	2.5 %	97.5 %
(Intercept)	1.5929831	0.0954133	1.404501	1.7814655
x1	1.4112333	0.0973291	1.219397	1.6030697
x2	1.9658191	0.0532322	1.860657	2.0709812
x1:x2	0.7550367	0.0570146	0.642302	0.8677715

From Table 4, we can observe the missingness of the data. Suppose we denote $\hat{\beta}_2$ as the estimated coefficient of x_2, the confidence interval contains the true values. On the other hand, for x_1 and $x_1 x_2$ the true value is not contained within the confidence interval. This is because, there is no missing values for x_2 but in x_1. The corresponding $x_1 x_2$ will also contain the missing value due to x_1.

b)

(10 marks) Now, start by calculating the interaction variable in the incomplete data and append it as a variable to your dataset. Then, use *passive imputation* to impute the interaction variable. Provide the estimates of β_1, β_2, and β_3 along with 95% confidence intervals. Comment.

Answer :

```
#store each columns
x1 <- dataex4$x1; x2 <- dataex4$x2; dataex4$x1x2 <- x1*x2
#perform the null imputaton with baseline
impute.null <- mice(dataex4, maxit = 0)
method <- impute.null$method
#specify formula to calculate x1x2
method["x1x2"] <- "~I(x1*x2)"
pred <- impute.null$predictorMatrix
pred[c("x1", "x2"), "x1x2"] <- 0
#change visiting scheme
visit.seq <- impute.null$visitSequence
visit.seq
```

```
## [1] "y"    "x1"   "x2"   "x1x2"
```

```
#performing passive imputation with m=50
impute.passive <- mice(dataex4, method = method, predictorMatrix = pred,
                visitSequence = visit.seq, m = 50, seed = 1, printFlag = FALSE)
pool.passive <- pool(with(impute.passive, lm(y ~ x1 + x2 + x1*x2))) #step 2, 3
kable(summary(pool.passive, conf.int = TRUE)[,c(1,2,3,7,8)],
      caption = "Summary Statistics of Imputation of $y$ and $x_1$") %>%
  kable_styling(latex_options = "hold_position")
```

Table 6: Summary Statistics of Imputation of y and x_1

term	estimate	std.error	2.5 %	97.5 %
(Intercept)	1.5534782	0.0884221	1.3788626	1.7280939
x1	1.1926170	0.0958435	1.0034980	1.3817360
x2	1.9964402	0.0493658	1.8989468	2.0939336
x1:x2	0.8740573	0.0567852	0.7615712	0.9865434

We performed *passive imputation* and presented the `estimates`, `std.error` and the confidence intervals. As Q4 a), only x_2 contains the true value within the confidence interval whereas x_1 and x_1x_2 do not contain. However, we need to note that the estimates are closer to the confidence interval compared to the previous part.

c)

(10 marks) Now that you have already appended the interaction variable to the dataset, impute it as it was *just another variable* (or like any other variable) in the dataset and use this variable for the interaction term in step 2. Provide the estimates of β_1, β_2 and β_3 along with 95% confidence intervals. Comment.

Answer :

```
#performing just anotehr variable imputation with m=50
impute.jav <- mice(dataex4, m = 50, seed = 1, printFlag = FALSE)
fit.jav <- with(impute.jav, lm(y ~ x1 + x2 + x1x2)) #step 2
pool.jav <- pool(fit.jav) #step 3
kable( summary(pool.jav, conf.int = TRUE)[, c(1,2,3,7,8)],
      caption = "Summary Statistics of Imputation of $y$ and $x_1$") %>%
  kable_styling(latex_options = "hold_position")
```

Table 7: Summary Statistics of Imputation of y and x_1

term	estimate	std.error	2.5 %	97.5 %
(Intercept)	1.499714	0.0782144	1.3452011	1.654227
x1	1.003930	0.0822837	0.8414967	1.166363
x2	2.026180	0.0437161	1.9398113	2.112548
x1x2	1.017793	0.0442807	0.9303479	1.105238

We performed *just another variable* imputation and presented the results in Table 7. This time, we observe that all confidence intervals of the coefficients includes the true values $\beta_0 = 1.5$, $\beta_1 = 1$, $\beta_2 = 2$ and $\beta_3 = 1$.

d)

(6 marks) What is the obvious conceptual drawback of the *just another variable* approach for imputing interactions?

Answer :

The conceptual drawback of the *just another variable* approach for imputing interaction is the imputation on x_1x_2 is no using x_1 and x_2 from the observed dataset. Thus, the product of x_1 and x_2 is not equal to x_1x_2. As a result, this will result in the biasness as the unbiased estimator for the parameters from the regression will no longer hold.

Q5

(30 marks) The file `NHANES2.Rdata` contains a subset of data from the *National Health and Nutrition Examination Survey* (NHANES), whose goal is to assess the health and nutritional status of adults and children in the United States. The variables in the dataset are the following:

- `wgt`: weight in kg,
- `gender`: male vs female,
- `bili`: bilirubin concentration in mg/dL,
- `age`: in years,
- `chol`: total serum cholestrol in mg/dL,
- `HDL`: High-density lipoprotein cholestrol in mg/dL,
- `hgt`: height in metres,
- `educ`: educational status; 5 ordered categories,
- `race`: 5 unordered categories,
- `SBP`: systolic blood pressure in mmHg,
- `hypten`: hyptertensive status; binary,
- `WC`: waist circumference in cm.

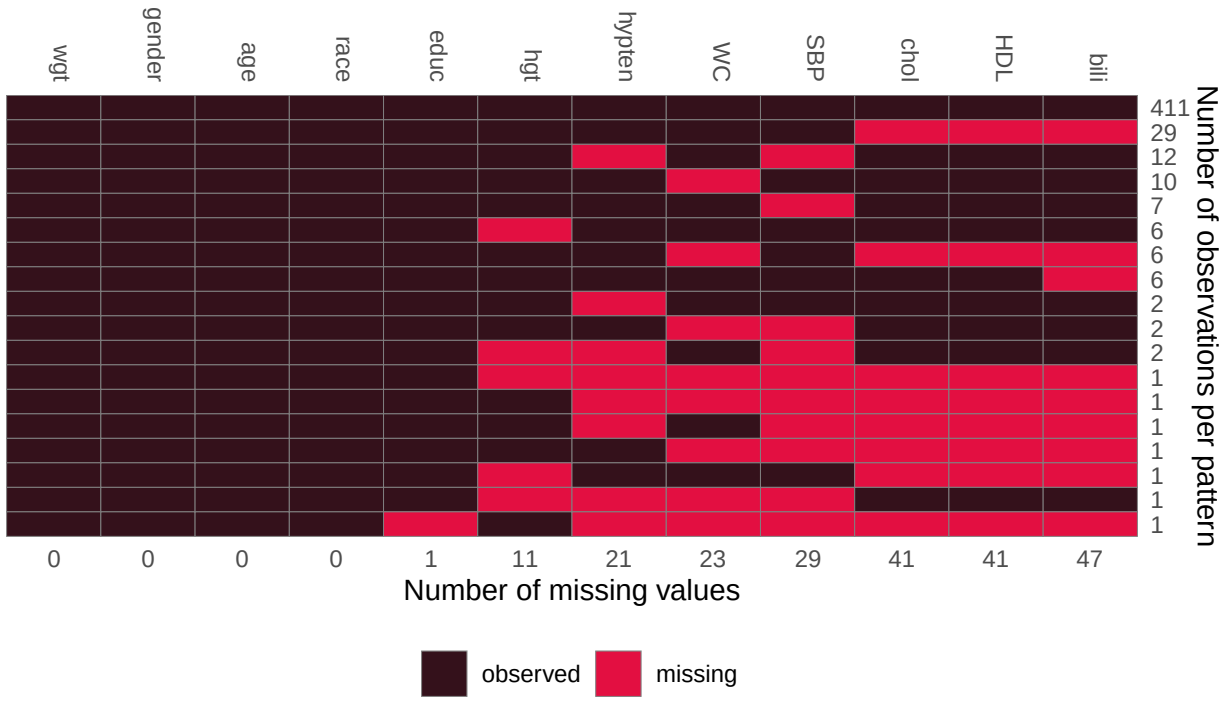
The analysis of interest is the following:

$$\text{wgt} = \beta_0 + \beta_1 \text{gender} + \beta_2 \text{age} + \beta_3 \text{hgt} + \beta_4 \text{WC} + \varepsilon, \quad \varepsilon \sim N(0, \sigma^2).$$

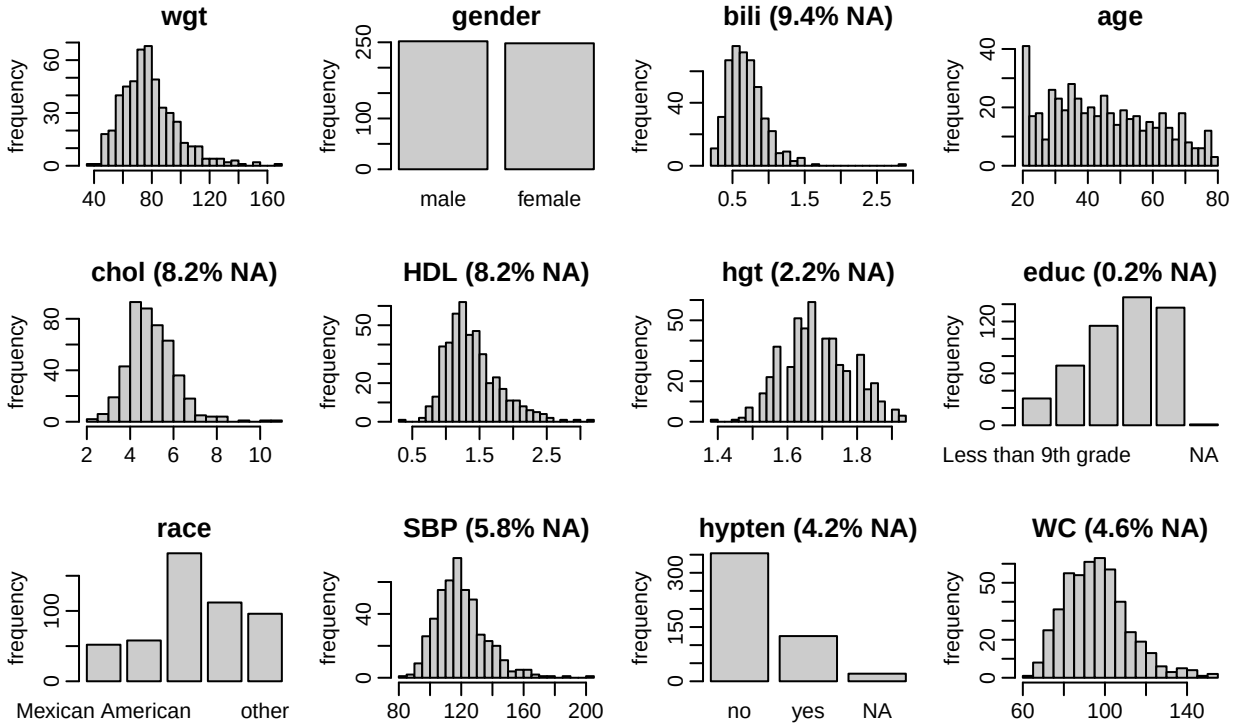
Using multiple imputation and conducting all necessary checks, report your findings.

Answer :

```
nhanes2 <- NHANES2
#compute number of oberservation per pattern
md_pattern(nhanes2, pattern = FALSE, color = c('#34111b', '#e30f41'))
```




```
par(mar = c(3,3,2,1), mgp = c(2,0.6,0))
#visaulise the distribution of each feature
plot_all(nhanes2, breaks = 30, ncol = 4)
```



```
#compute dry set up
impute.null <- mice(nhanes2, maxit = 0)
#change default imputation method
meth <- impute.null$method
meth["hgt"] <- "norm"
meth
```

```
##      wgt   gender     bili      age     chol      HDL      hgt     educ
##       ""       ""    "pmm"       ""    "pmm"    "pmm"   "norm"   "polr"
##     race      SBP   hypten       WC
##       ""    "pmm" "logreg"    "pmm"
```

```
post <- impute.null$post
#specifying the range of hgt
post["hgt"] <- "imp[[j]][,i] <- squeeze(imp[[j]][,i], c(1.397, 1.9304))"
```

While inspecting the missing data pattern, we found 411 observations with observed values on all 12 variables. Also, 10 observations for which on `WC` is missing, 6 observations for which on `hgt` is missing Interestingly, we can observe that there are 29 observations for which on `chol`, `HDL` and `bili` are missing.

We can visualise the variables' distributions using `plot_all()`. In the plot above, depicting the distribution of the observed data for the different variables, we could appreciate that `hgt` following a normal distribution is possibly not a completely unreasonable idea. Let us then change the default from predictive mean matching method (`pmm`) to `norm` for the variable `hgt`.

However, we need to be careful, because we do not want to risk imputing a negative value for the height. With the below syntax all imputed values of `hgt` that are outside the interval `(min(nhanes2$hgt), max(nhanes2$hgt))` will be set to those limiting values.

Since our model is limited to this, $\text{wgt} = \beta_0 + \beta_1 \text{gender} + \beta_2 \text{age} + \beta_3 \text{hgt} + \beta_4 \text{WC} + \varepsilon, \quad \varepsilon \sim N(0, \sigma^2)$.. We will end our data preprocessing and proceed to the next them in tuning the hyperparameters for imputation

```
#compute the pooled estimates with different seed and M values
seed1.5 <- pool(with(mice(nhanes2, methods = meth, post = post,
                          m = 5, seed = 1, printFlag = FALSE),
                     lm(wgt ~ gender + age + hgt + WC)))
seed1.10 <- pool(with(mice(nhanes2, methods = meth, post = post,
                          m = 10, seed = 1, printFlag = FALSE),
                     lm(wgt ~ gender + age + hgt + WC)))
seed1.20 <- pool(with(mice(nhanes2, methods = meth, post = post,
                          m = 20, seed = 1, printFlag = FALSE),
                     lm(wgt ~ gender + age + hgt + WC)))
seed1.25 <- pool(with(mice(nhanes2, methods = meth, post = post,
                          m = 25, seed = 1, printFlag = FALSE),
                     lm(wgt ~ gender + age + hgt + WC)))

seed2.5 <- pool(with(mice(nhanes2, methods = meth, post = post, maxit = 20,
                          m = 5, seed = 2, printFlag = FALSE),
                     lm(wgt ~ gender + age + hgt + WC)))
seed2.10 <- pool(with(mice(nhanes2, methods = meth, post = post, maxit = 20,
                          m = 10, seed = 2, printFlag = FALSE),
                     lm(wgt ~ gender + age + hgt + WC)))
seed2.20 <- pool(with(mice(nhanes2, methods = meth, post = post, maxit = 20,
                          m = 20, seed = 2, printFlag = FALSE),
                     lm(wgt ~ gender + age + hgt + WC)))
seed2.25 <- pool(with(mice(nhanes2, methods = meth, post = post, maxit = 20,
                          m = 25, seed = 2, printFlag = FALSE),
```

```
                    lm(wgt ~ gender + age + hgt + WC)))
```

```
kable(data.frame(summary(seed1.5, conf.int = TRUE)[, c(1, 2, 3, 7, 8)],
                 lambda = seed1.5$pooled[,10]), caption = "Seed=1 and m=5") %>%
  kable_styling(latex_options = "hold_position")
```

Table 8: Seed=1 and m=5

term	estimate	std.error	X2.5..	X97.5..	lambda
(Intercept)	-101.4588664	7.5938549	-116.3816321	-86.5361006	0.0195179
genderfemale	-1.4195844	0.8425729	-3.0758469	0.2366781	0.0355672
age	-0.1589451	0.0213354	-0.2008787	-0.1170115	0.0293297
hgt	53.0413585	4.3496144	44.4934595	61.5892575	0.0227495
WC	1.0225600	0.0223080	0.9787286	1.0663914	0.0049737

```
kable(data.frame(summary(seed1.10, conf.int = TRUE)[, c(1, 2, 3, 7, 8)],
                 lambda = seed1.10$pooled[,10]), caption = "Seed=1 and m=10") %>%
  kable_styling(latex_options = "hold_position")
```

Table 9: Seed=1 and m=10

term	estimate	std.error	X2.5..	X97.5..	lambda
(Intercept)	-101.6700795	7.8551715	-117.1260787	-86.2140802	0.0937749
genderfemale	-1.3085502	0.8493953	-2.9787291	0.3616288	0.0667687
age	-0.1576161	0.0215356	-0.1999575	-0.1152746	0.0616516
hgt	52.9908125	4.4813711	44.1741136	61.8075113	0.0899366
WC	1.0244107	0.0222726	0.9806462	1.0681752	0.0168680

```
kable(data.frame(summary(seed1.20, conf.int = TRUE)[, c(1, 2, 3, 7, 8)],
                 lambda = seed1.20$pooled[,10]), caption = "Seed=1 and m=20") %>%
  kable_styling(latex_options = "hold_position")
```

Table 10: Seed=1 and m=20

term	estimate	std.error	X2.5..	X97.5..	lambda
(Intercept)	-101.2897304	7.8011807	-116.6273935	-85.9520673	0.0822178
genderfemale	-1.3626948	0.8438271	-3.0211578	0.2957682	0.0527619
age	-0.1602819	0.0216544	-0.2028488	-0.1177150	0.0690126
hgt	52.7103557	4.4457568	43.9703249	61.4503865	0.0766459
WC	1.0265327	0.0222736	0.9827681	1.0702973	0.0122374

```
kable(data.frame(summary(seed1.25, conf.int = TRUE)[, c(1, 2, 3, 7, 8)],
                 lambda = seed1.25$pooled[,10]), caption = "Seed=1 and m=25") %>%
  kable_styling(latex_options = "hold_position")
```

```
kable(data.frame(summary(seed2.5, conf.int = TRUE)[, c(1, 2, 3, 7, 8)],
                 lambda = seed2.5$pooled[,10]), caption = "Seed=2 and m=5") %>%
  kable_styling(latex_options = "hold_position")
```

```
kable(data.frame(summary(seed2.10, conf.int = TRUE)[, c(1, 2, 3, 7, 8)],
                 lambda = seed2.10$pooled[,10]), caption = "Seed=2 and m=10") %>%
  kable_styling(latex_options = "hold_position")
```

Table 11: Seed=1 and m=25

term	estimate	std.error	X2.5..	X97.5..	lambda
(Intercept)	-101.0547825	7.7120024	-116.2127029	-85.8968621	0.0624659
genderfemale	-1.3203389	0.8335918	-2.9583505	0.3176727	0.0279959
age	-0.1593655	0.0214497	-0.2015208	-0.1172102	0.0512832
hgt	52.5781406	4.4232119	43.8838452	61.2724359	0.0683664
WC	1.0259679	0.0224005	0.9819518	1.0699840	0.0233236

Table 12: Seed=2 and m=5

term	estimate	std.error	X2.5..	X97.5..	lambda
(Intercept)	-100.9966596	7.7300251	-116.2136672	-85.7796520	0.0752096
genderfemale	-1.3775163	0.8404887	-3.0302117	0.2751791	0.0471154
age	-0.1580193	0.0213131	-0.1999240	-0.1161147	0.0436862
hgt	52.5424942	4.4106181	43.8613258	61.2236627	0.0718380
WC	1.0259230	0.0222557	0.9821911	1.0696550	0.0142010

Table 13: Seed=2 and m=10

term	estimate	std.error	X2.5..	X97.5..	lambda
(Intercept)	-101.5186858	7.5571364	-116.3699779	-86.6673938	0.0311981
genderfemale	-1.3215513	0.8256067	-2.9438439	0.3007412	0.0185686
age	-0.1592563	0.0213802	-0.2012842	-0.1172285	0.0510297
hgt	52.8833997	4.3363259	44.3609058	61.4058936	0.0384335
WC	1.0255032	0.0222773	0.9817285	1.0692778	0.0197263

```
kable(data.frame(summary(seed2.20, conf.int = TRUE)[, c(1, 2, 3, 7, 8)],
                 lambda = seed2.20$pooled[,10]), caption = "Seed=2 and m=20") %>%
  kable_styling(latex_options = "hold_position")
```

Table 14: Seed=2 and m=20

term	estimate	std.error	X2.5..	X97.5..	lambda
(Intercept)	-101.6993777	7.6823878	-116.7986779	-86.600078	0.0547441
genderfemale	-1.3663151	0.8370945	-3.0113631	0.278733	0.0398821
age	-0.1576975	0.0214614	-0.1998791	-0.115516	0.0557808
hgt	53.0055071	4.4023585	44.3525473	61.658467	0.0590317
WC	1.0246288	0.0223533	0.9807051	1.068553	0.0239163

```
kable(data.frame(summary(seed2.25, conf.int = TRUE)[, c(1, 2, 3, 7, 8)],
                 lambda = seed2.25$pooled[,10]),
      caption = "Seed=2 and m=25") %>%
  kable_styling(latex_options = "hold_position")
```

Looking at Tables above, we have the (pooled) estimates, standard errors, the confident intervals and lambda which get more stable as M increases and we can be more confident in any one specific run. Since we run the multiple imputation with a sufficiently large M, the results will with high probability only differ by a small amount. Thus we will choose `seed=1` with `M=25` and proceed to the multiple imputation.

Table 15: Seed=2 and m=25

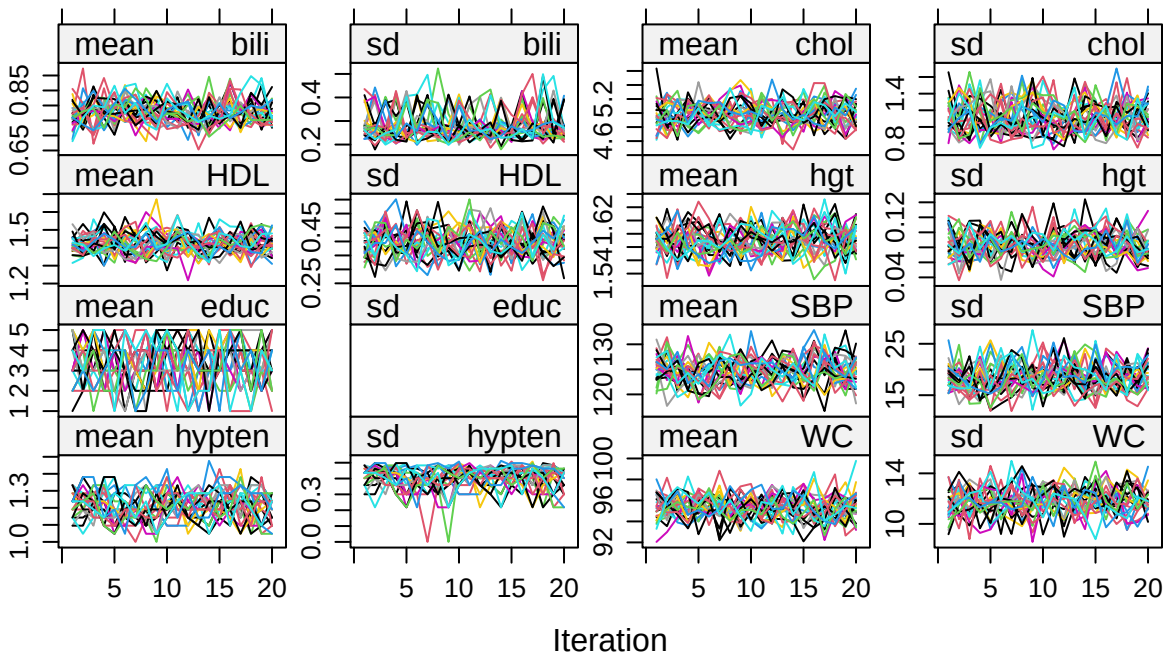
term	estimate	std.error	X2.5..	X97.5..	lambda
(Intercept)	-101.4521949	7.6290783	-116.4452338	-86.4591561	0.0472501
genderfemale	-1.3125011	0.8317349	-2.9468852	0.3218831	0.0303398
age	-0.1585538	0.0213320	-0.2004759	-0.1166316	0.0456505
hgt	52.8201428	4.3724292	44.2269381	61.4133475	0.0516546
WC	1.0255459	0.0223605	0.9816083	1.0694836	0.0243072

```
#imputation with the chosen seed and M
imp <- mice(nhanes2, method = meth, post = post, maxit = 20,
            m = 25, seed = 1, printFlag = FALSE)
imp$loggedEvents
```

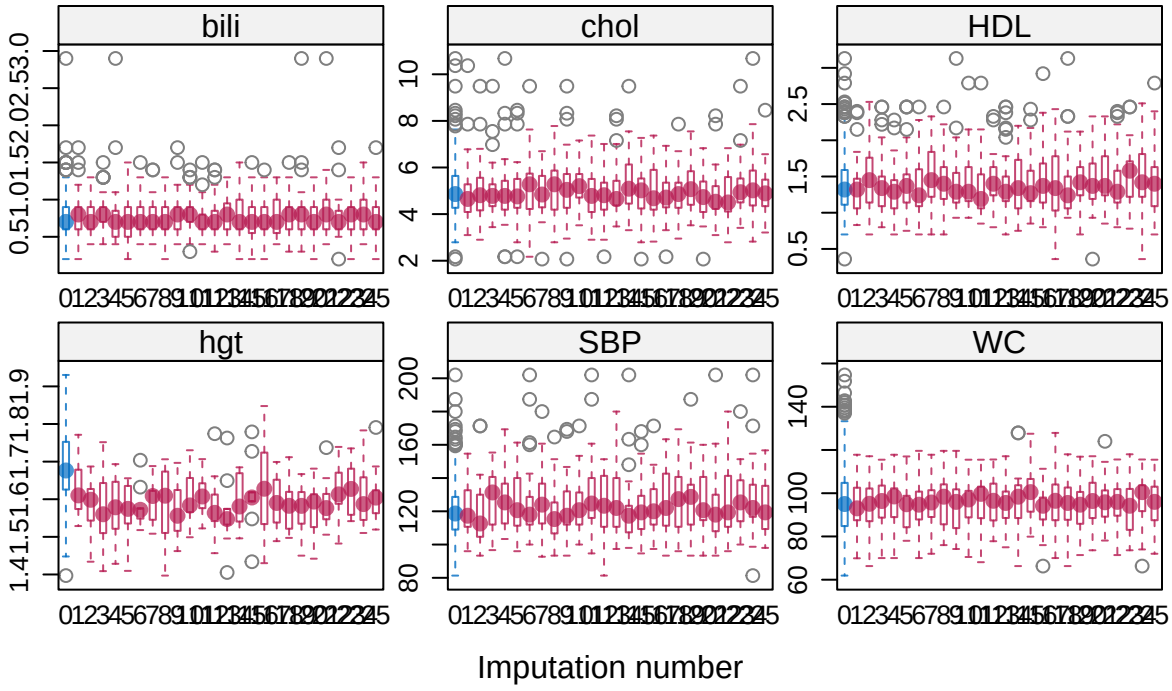
```
## NULL
```

We can also confirm through `loggedEvents` that no problems occurred during the imputation.

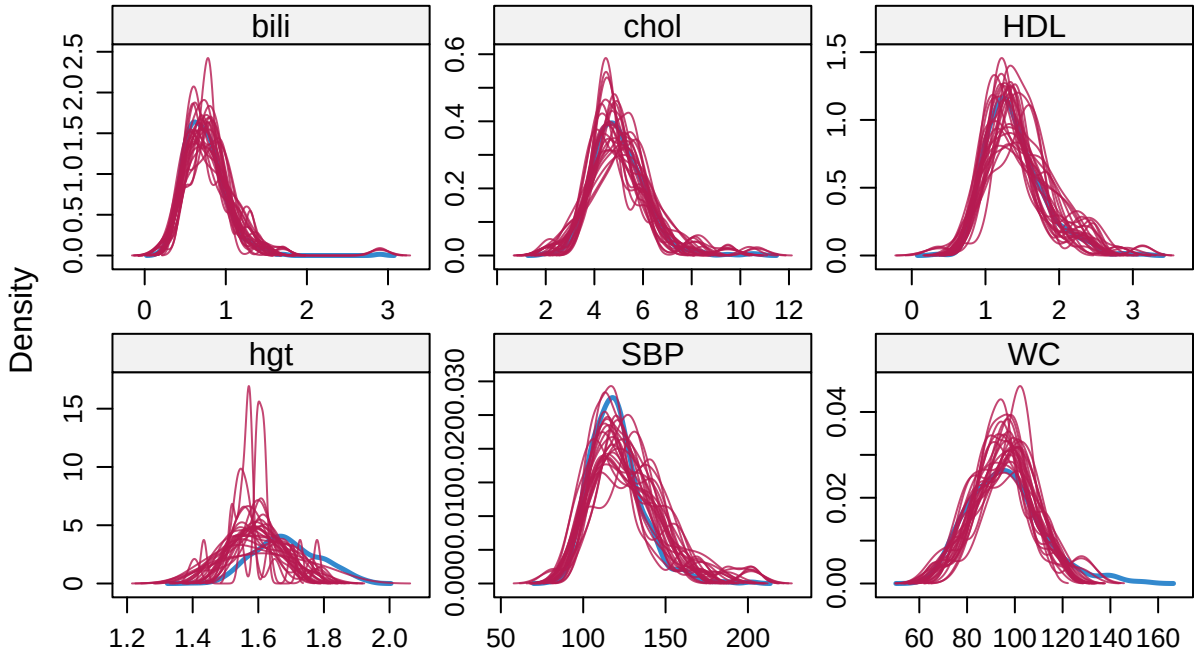
```
plot(imp, layout=c(4,4))
```



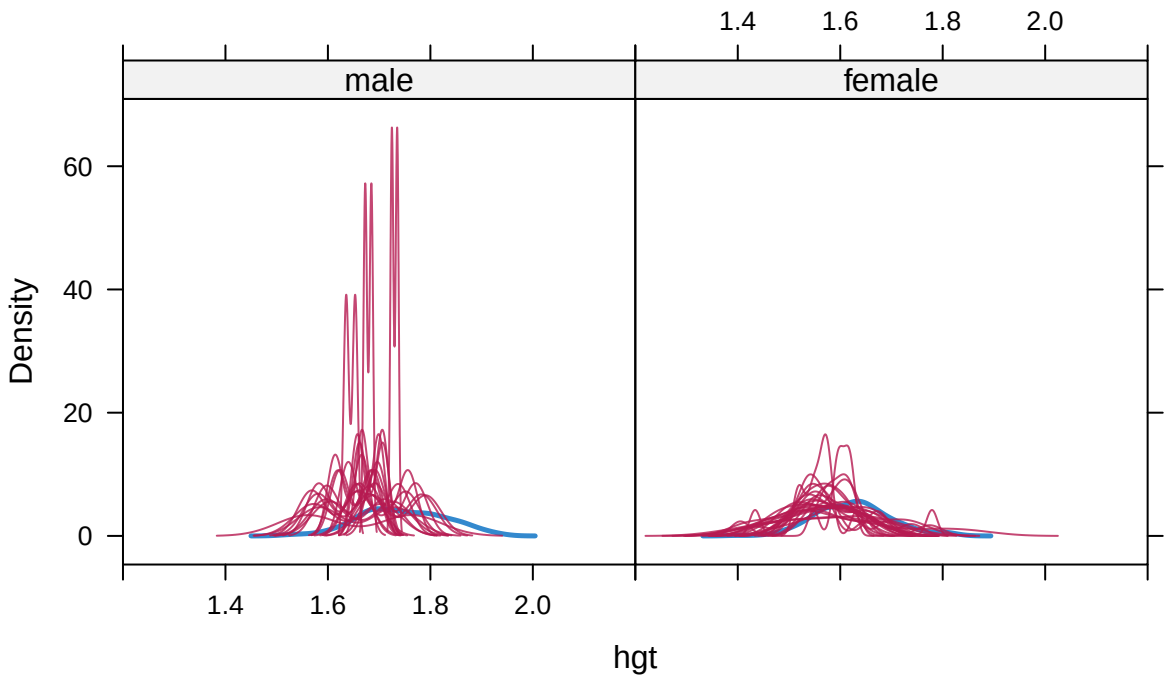

```
bwplot(imp)[c(2,4,5,6,7,8)]
```



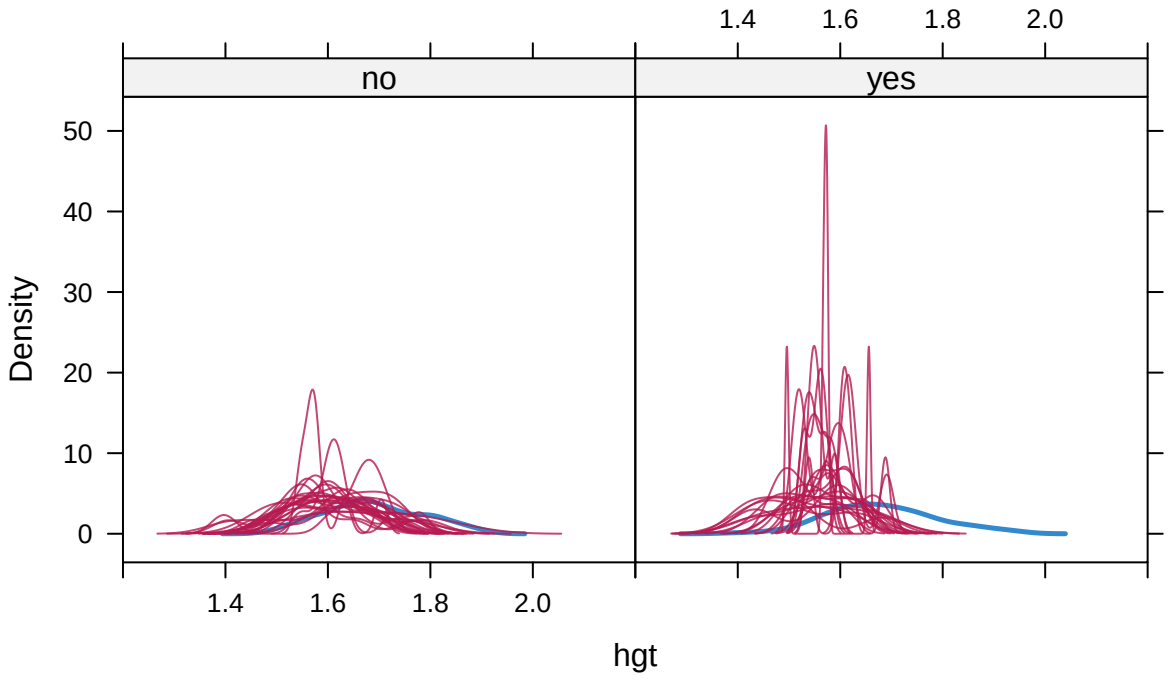

```
densityplot(imp)
```



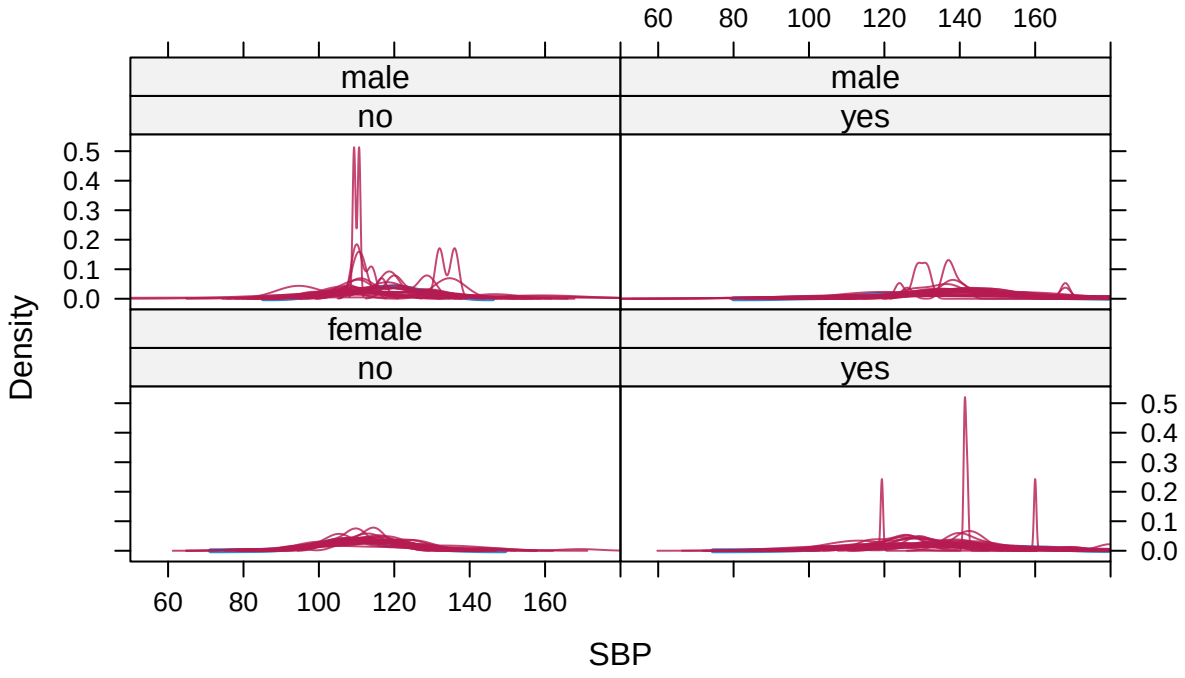

```
densityplot(imp,~hgt|gender, xlim = c(1.2, 2.2))
```



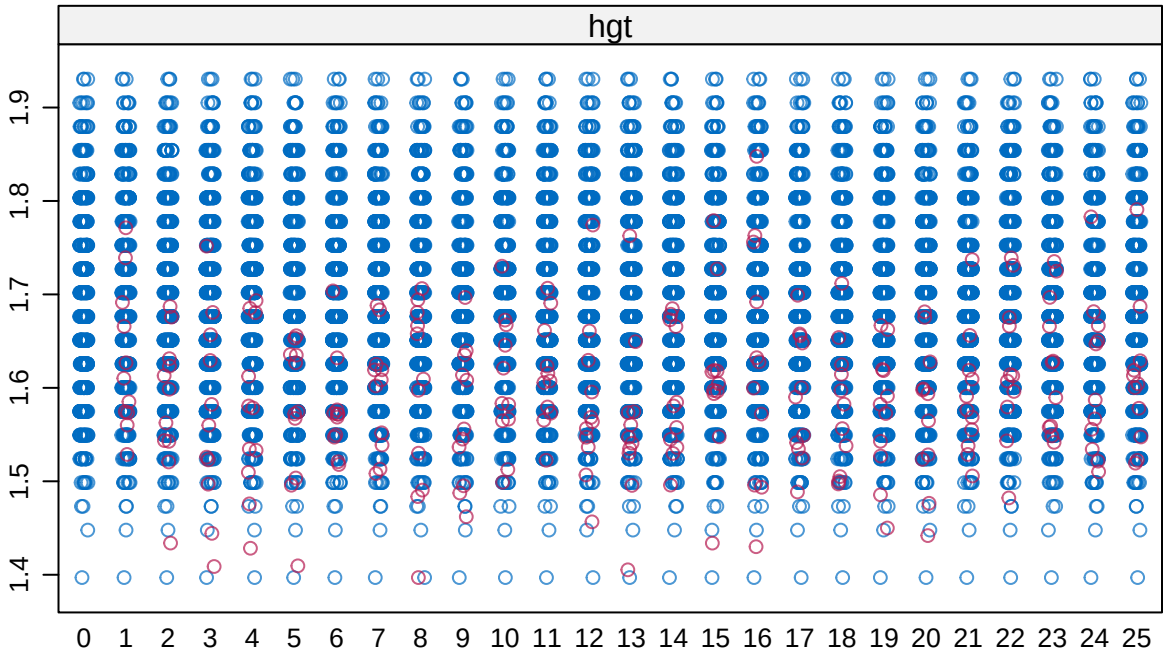

```
densityplot(imp,~hgt|hypten, xlim = c(1.2, 2.2))
```



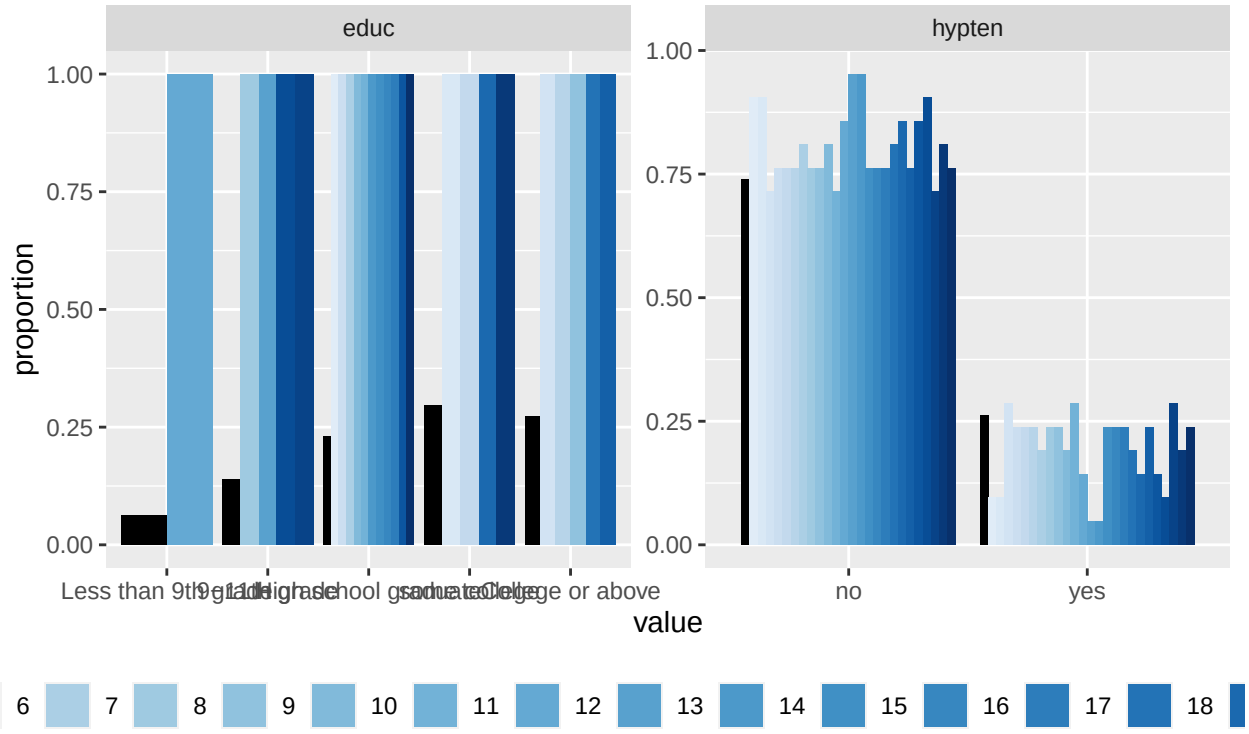

```
densityplot(imp,~SBP|hypten + gender, xlim=c(50, 180))
```




```
stripplot(imp)[6]
```




```
propplot(imp)
```



Now let us process the plots one by one to check if the `mice()` converged. Except for `educ`, rest of the variables are continuous are we proceed to the first plot. The reason for a missing plot for `educ` is because there is only one missing value `educ` so no standard deviation. The traceplot indicates that the iterative algorithm appears to have converged for all variables that were imputed. The next plots are boxplots and density plots for the observed and imputed ones where we label them as blue and red respectively.

Note that we are using `M=25` and because the density of the observed data is possibly plotted first, we can barely see it. However, we can generally confirm that most imputed datasets follow a similar distribution. The most outstanding plots are the ones from `SBP` and `hgt` as there is a shift towards lower values.

Specifically, we investigated with `hgt` conditional on the `gender` and `hgt` conditional on `hypten`. We can see that to a certain extent, both plots explain the differences between the observed and imputed values for `hgt`. This is because `male` and `positive` have much narrower distribution compared to wider distribution for `female` and `negative`. For `SBP` we could observe similarity. Using `stripplot()` we can double confirm that the imputation seems reasonable.

We observe an abnormal imputation pattern for the `educ` variable however this is not a cause for concern since we are only imputing a single missing value out of 500 cases. Meanwhile, for the `hypten` variable, we observe a reasonable amount between imputation variance but not enough such that the general distribution of this variable is lost. In summary, all imputations for variables with missing data have been performed successfully.

We can compare the distribution of the imputed values against the distribution of the observed categorical variables' values using `propplot()`. This compares the proportion of values in each category. This shows a large discrepancy between the observed and imputed data distributions for the `educ`. However educ only contains 1 missing variable and we can conclude that there is insufficient evidence to say that the imputation `educ` has problem. For the `hypten`, there is a reasonable proportion between the imputation variances.

All together, we can confirm that the multiple imputation is successful and proceed to analysis of the imputed data. Since we set `M=25`, we will only look at the first case.

```r
#fitting with given parameters
fit <- with(imp, lm(wgt ~ gender + age + hgt + WC))
summary(fit$analyses[[1]])
```

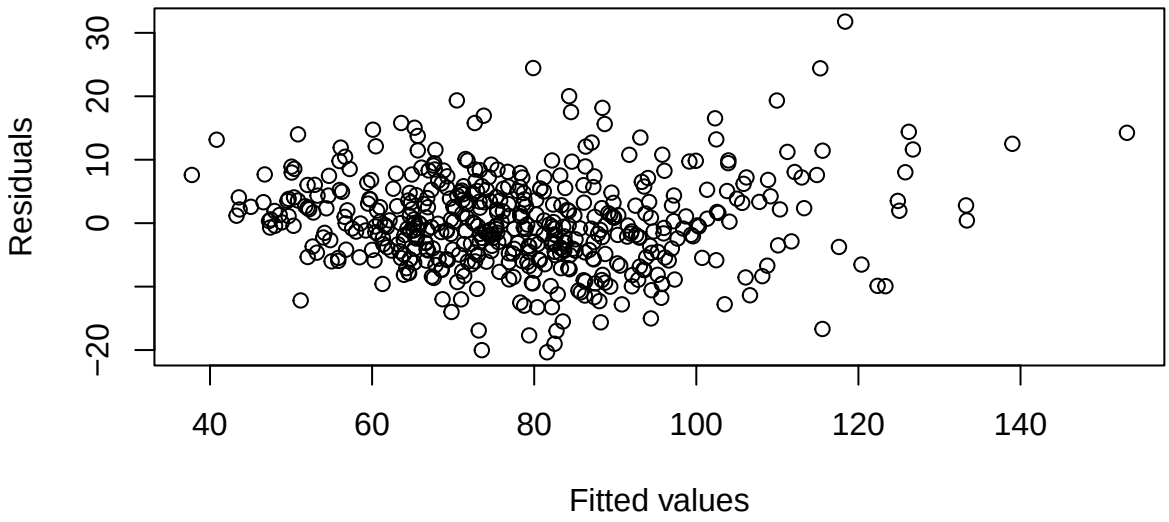
```
##
## Call:
## lm(formula = wgt ~ gender + age + hgt + WC)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -20.349  -4.568  -0.428   4.068  31.774
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -100.83781    7.51251 -13.423  < 2e-16 ***
## genderfemale   -1.36542    0.82182  -1.661   0.0973 .
## age            -0.15556    0.02085  -7.460 3.91e-13 ***
## hgt            52.42865    4.29633  12.203  < 2e-16 ***
## WC              1.02524    0.02210  46.387  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.192 on 495 degrees of freedom
## Multiple R-squared:  0.857,  Adjusted R-squared:  0.8558
## F-statistic: 741.4 on 4 and 495 DF,  p-value: < 2.2e-16
```

Looking at the summary we obtained the estimates as below,

$$\text{wgt} = -100.83781 - 1.36542 \times \text{gender} - 0.15556 \times \text{age} + 52.42865 \times \text{hgt} + 1.02524 \times \text{WC}$$

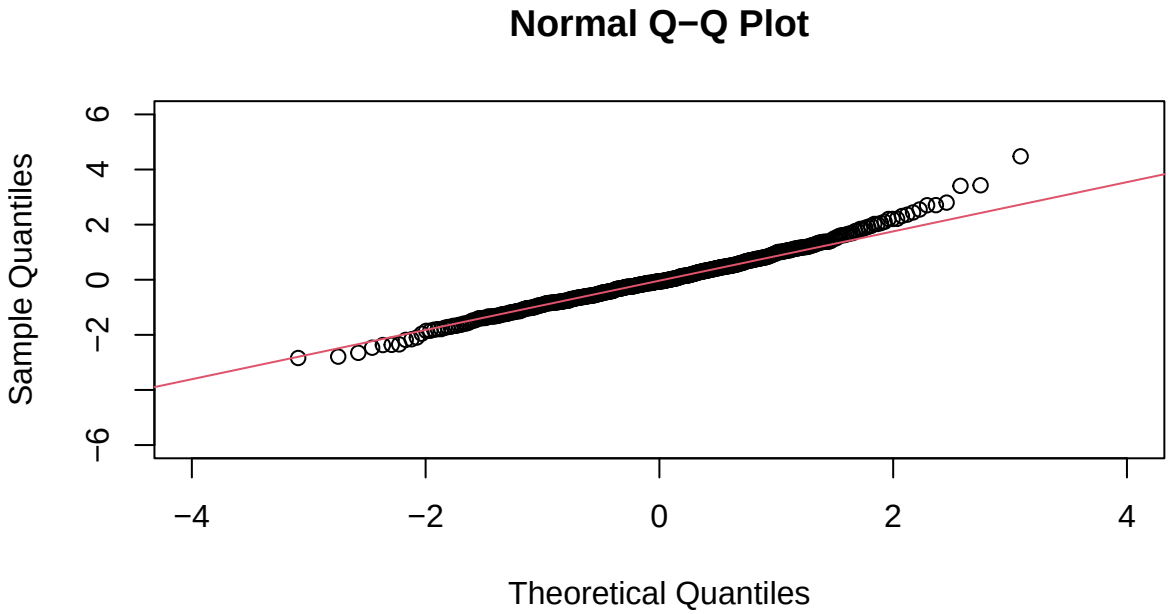
We can also check that only the coefficient for `genderfemale` has a p-value greater than 0.05. To this, we will check using Wald Test later for the importance of it.

```r
plot(fit$analyses[[1]]$fitted.values, residuals(fit$analyses[[1]]),
     xlab = "Fitted values", ylab = "Residuals")
```



In this fitted values versus residuals plot we can observe that the points are spread equally and randomly. Thus, no obvious trend and we can believe that the linear assumption holds for this model.

```
qqnorm(rstandard(fit$analyses[[1]]), xlim = c(-4, 4), ylim = c(-6, 6))
qqline(rstandard(fit$analyses[[1]]), col = 2)
```



By observing the normal Q-Q plot, we do not see the deviance from the red dotted base line. Therefore, we can conclude that the data after imputation follows normal distribution and the normality assumption holds. Lastly, we will proceed to pooling the estimates with our imputed dataset. From Table 16, the summary statistics is provided for the pooled estimates.

```
pooled_ests <- pool(fit)
kable(summary(pooled_ests, conf.int = TRUE)[c(1,2,6,7,8)],
      caption = "Summary Statistics") %>%
  kable_styling(latex_options = "hold_position")
```

Table 16: Summary Statistics

term	estimate	p.value	2.5 %	97.5 %
(Intercept)	-101.1180859	0.0000000	-116.0326895	-86.2034823
genderfemale	-1.3477467	0.1063107	-2.9844502	0.2889568
age	-0.1582103	0.0000000	-0.2004609	-0.1159597
hgt	52.6066777	0.0000000	44.0729374	61.1404180
WC	1.0256687	0.0000000	0.9818758	1.0694617

```
kable(pool.r.squared(pooled_ests, adjusted = TRUE), caption = "") %>%
  kable_styling(latex_options = "hold_position")
```

Table 17:

	est	lo 95	hi 95	fmi
adj R^2	0.8563534	0.8308076	0.8783204	0.0261765

Looking at the Table 17, the adjusted R^2 values indicates good fit.

Now we conduct Wald Test to check the importance of the features.

```
#performing wald test
fit.no.WC <- with(imp, lm(wgt ~ gender + age + hgt ))
fit.no.hgt<- with(imp, lm(wgt ~ gender + age + WC))
fit.no.age <- with(imp, lm(wgt ~ gender + hgt + WC))
fit.no.gender <- with(imp, lm(wgt ~ age + hgt + WC))
wald.stats <- rbind(D1(fit, fit.no.WC)$result, D1(fit, fit.no.hgt)$result,
            D1(fit, fit.no.age)$result, D1(fit, fit.no.gender)$result)
excluded.features <- c("WC", "hgt", "age", "gender")
df <- data.frame(excluded.features, wald.stats)
kable(df, caption = "Wald Test on each features") %>%
  kable_styling(latex_options = "hold_position")
```

Table 18: Wald Test on each features

excluded.features	F.value	df1	df2	P..F.	RIV
WC	2117.811020	1	488.6612	0.0000000	0.0174882
hgt	146.751120	1	472.7845	0.0000000	0.0394115
age	54.164973	1	447.6375	0.0000000	0.0621865
gender	2.618292	1	479.2870	0.1062959	0.0319663

We conducted the Wald test by excluding each feature one by one and used `D1()` to compute its value. From the Table, we can see that `gender` is the only feature having p-value greater than 0.05. Although it is above 0.05, we can still consider its important but limiting it into 0.1 significance level. Thus, `WC`, `hgt` and `age` have more influence than `gender` here.