Tampereen yliopisto

Jaakko Eskonen

Moni-imputoinnin toteuttaminen R-ohjelmistossa

Informaatioteknologian ja viestinnän tiedekunta
Kandidaatintutkielma
Huhtikuu 2020

Tiivistelmä

Jaakko Eskonen: Moni-imputoinnin toteuttaminen R-ohjelmistossa
Kandidaatintutkielma
Tampereen yliopisto
Matematiikan ja tilastotieteen tutkinto-ohjelma
Huhtikuu 2020

Tutkimuksessa tarkastellaan R-ohjelmiston moni-imputointiin tarkoitettujen funktioiden toimintaa ja tehdään simulaatiotesti, jossa niiden suoriutumista vertaillaan. Moni-imputointi on keino luoda aineistosta parempia tilastollisia malleja tilanteessa, jossa aineisto sisältää puuttuvaa dataa. Moni-imputoinnin imputointivaiheeseen on olemassa erilaisia algoritmeja. Tässä tutkimuksessa tarkastellaan Micen, Hmisc:n ja Amelian sisältämiä moni-imputoinnin imputointivaiheen algoritmeja.

Työn alussa tarkastellaan, miten moni-imputointi tapahtuu ja kuinka Micen, Hmisc:n ja Amelian sisältämät moni-imputoinnin imputointivaiheen algoritmit toimivat teoreettisesti. Työn simulaatiotestiosassa vertaillaan käytettyjen pakkausten moni-imputointifunktioiden suoriutumista. Työssä kuvataan, kuinka moni-imputointi kullakin käytetyllä pakkauksella suoritetaan.

Simulaatiotestissä käytetään miehistä kerättyjä terveystietoja. Tähän aineistoon luodaan ensin satunnaisesti puuttuvaa dataa yhteen muuttujaan, jonka jälkeen moni-imputointifunktioilla luodaan regressiomallit valituista muuttujista. Testi suoritetaan kolmella tapaa. Testissä luodaan puuttuvaa dataa 10%:iin, 30%:iin ja 50%:iin valittuun muuttujaan. Vertailemalla kokonaisen aineiston pohjalta tehtyä regressiomallia ja moni-imputointifunktion avulla, vajaasta aineistosta saatuja regressiomalleja, pyritään selvittämään, kuinka hyvin käytetyt moni-imputointifunktiot käytännössä toimivat.

Tutkimuksella ei pystytty todistamaan, että joku käytetyistä moni-imputointifunktioista olisi muita parempi. Tämä johtuu vakioidusta datasta, johon testi luotiin. Amelia-pakkaus suoriutui kuitenkin parhaiten simulaatiotestissä, vaikkakaan aineiston muuttujat eivät yleisesti olleet normaalisti jakautuneita ja Amelia-pakkauksen moni-imputointifunktio olettaa datan multinormaalijakautuneisuutta. Näin voidaan päätellä, että Amelia voi olla hyvä vaihtoehto moni-imputoinnille, vaikka data ei olisi täysin multinormaalijakautunut. Amelia on pakkauksena luotu vain moni-imputointiin, toisin kuin Mice, jolla oli vahvuutena useat vaihtoehtoiset imputointitavat. Simulaatiotestissä Micen ja Hmisc:n tulokset olivat todella lähellä toisiaan.

Avainsanat: R, moni-imputointi, puuttuva tieto, regressioanalyysi

Tämän julkaisun alkuperäisyys on tarkastettu Turnitin OriginalityCheck –ohjelmalla.

Sisältö

1 Johdanto **4**

2 Moni-imputointi **5**

3 Käytetyt ohjelmistokirjastot **6**
3.1 Mice ja pmm 6
3.2 Ohjelmakirjastot Hmisc, pmm ja bootstrap 7
3.3 Amelia ja EM-algoritmi 7

4 Simulaatiotesti **9**

5 Päätelmät **15**

Lähteet **16**

1 Johdanto

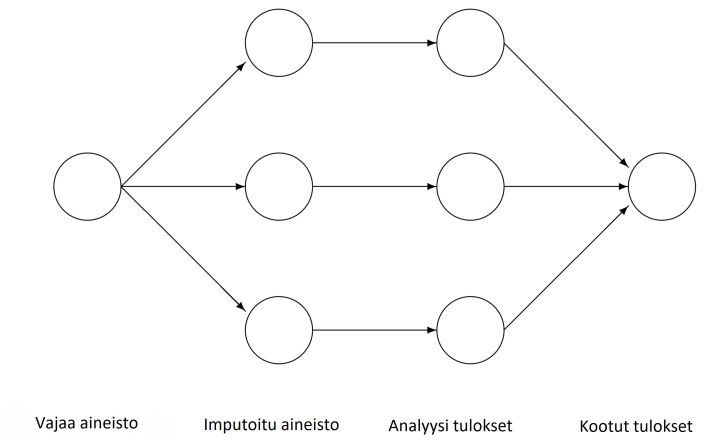
Tämän tutkimuksen tarkoituksena on vertailla R-ohjelmiston yleisimpiä moni-imputointiin käytettyjä pakkauksia ja niiden moni-imputointifunktioiden eroja. Tutkimuksen alkuosassa vertaillaan, käytettyjen pakkausten moni-imputointifunkioiden toiminnan eroja. Tämän jälkeen tehdään simulaatiotesti, jolla pyritään testaamaan tarkasteltavien pakkausten sisältämien moni-imputointifunktioiden suoriutumista.

On hyvin yleistä, että tutkittavista aineistoista puuttuu dataa. Datan puuttuminen on tärkeä huomioida tutkimusprosessissa. Data voi puuttua aineistosta kolmella tapaa: MCAR, MAR JA MNAR. MCAR(missing comletely at random) tarkoittaa, että todennäköisyys datan puuttumiselle muuttujassa, ei liity millään tapaa mihinkään arvoihin joita on havaittu tai joita ei ole pystytty havaitsemaan. MAR(missing at random) tarkoittaa, että puuttuva data muuttujassa liittyy mitattuun tai mitattuihin arvoihin muissa muuttujissa, mutta ei puuttuvaan arvoon itseensä, eli arvoihin joita olisi voitu havaita. MNAR(missing not at random) tarkoittaa, että datan puuttuminen muuttujassa liittyy muuttujan arvoon itseensä. Moni-imputoinnilla, voidaan parantaa aineiston analyyseja MAR tilanteessa.(Enders 2010, s. 187–189) Tämän tutkimuksen tarkoituksena on erityisesti esitellä ja vertailla R:n tarjoamia moni-imputointiin tarkoitettuja ohjelmistoja.

Ohjelmistot, joita tässä tutkielmassa käytetään ovat MICE, Hmisc ja Amelia. Valinta näiden R:n tarjoamien moni-imputointiohjelmistojen välillä ei ole helppoa ja tutkielmassa vertaillaan näiden ohjelmistojen hyviä ja huonoja puolia.

2 Moni-imputointi

Moni-imputoinnissa on kolme vaihetta: imputointi-, analyysi- ja kokoamisvaihe (Enders 2010, s. 187). Imputointivaiheen algoritmit vaihtelevat eri ohjelmistoissa. Kunkin käytetyn pakkauksen imputointivaihe esitellään pakkauksen kuvauksessa, kappaleessa "Käytetyt pakkaukset". Ensimmäisessä vaiheessa (imputointivaiheessa) luodaan useita vaihtoehtoisia aineistoja, täyttämällä puuttuva data tietyillä algoritmilla lasketuilla arvoilla. Näin saadaan useita vaihtoehtoja aineistosta, jotka eivät sisällä puuttuvaa dataa. Toisessa vaiheessa eli analyysivaiheessa luodaan tilastolliset mallit luotujen aineistojen pohjalta. Viimeisessä vaiheessa, kokoamisvaiheessa, luodut tilastolliset mallit yhdistetään yhdeksi malliksi. Lopullisessa mallissa parametrien estimaatit lasketaan vaihtoehtoisten mallien kyseisten parametrien arvoista.



Kuva 2.1. Moni-imputoinnin vaiheet (Van Buuren 2012, s. 17).

Moni-imputoinnilla ei yritetä luoda uudelleen aineistoa, vaan luodaan tilastollinen malli, joka parhaiten kuvaa aineistoa ja sen käyttäytymistä. Tästä syystä moni-imputointiohjelmistojen vertailussa keskitytään vertailemaan estimoitujen tilastollisten mallien parametrien arvoja.

3 Käytetyt ohjelmistokirjastot

3.1 Mice ja pmm

Mice on yksi laajimpia R:n tarjoamia puuttuvan datan käsittelyyn tarkoitettuja ohjelmistokirjastoja. Sen avulla voidaan käsittellä puuttuvaa dataa niin perinteisillä metodeilla (kuten keskiarvoimputoimalla), kuin myös moni-imputoimalla. Käytännössä Mice-pakkauksen avulla moni-imputointi tapahtuu funktioilla, `mice()`(imputointivaihe), `with()`(analyysivaihe) ja `pool()`(kokoamisvaihe). Mice tarjoaa myös tässäkin tutkimuksessa käytetyn ampute()-funktion, jonka avulla aineistoon saadaan generoitua puuttuvaa tietoa.

Oletuksena `mice()`-funktio tarjoaa numeerisille muuttujille pmm moni-imputointitavan, joten tässä tutkimuksessa käytetään nimenomaan tätä. Pmm eli predictive mean matching on yleisesti käytetty menetelmä imputointivaiheeseen ja se pystyy käsittelemään myös diskreettejä muuttujia. Pmm-moni-imputoinnin vaiheet voidaan jakaa kuuteen eri osaan.

Oletetaan, että on yksi muuttuja y, jolla on joissain tapauksissa puuttuvaa tietoa, ja joukko muuttujia X (ilman puuttuvaa tietoa), joita käytetään y:n imputointiin.

1. Tapausten avulla, joista tietoja ei puutu, estimoidaan muuttujalle y lineaarinen regressiomalli, selittäjinä muuttujat X. Tämä tuottaa kertoimienestimaattien joukon b.

2. Otetaan satunnaisotos b:n "posteriorijakaumasta"tuottaen uuden kerroin estimaattien joukon b*. Tyypillisesti tämä olisi satunnaisotos multinormaalijakaumasta, missä odotusarvo ja kovarianssimatriisi on estimoitu joukosta b.

3. Luodaan b*:llä ennustetut arvot y:lle kaikissa tapauksissa, sekä tapauksissa, joissa y:n tieto puuttuu, että niissä, joissa on tietoa.

4. Jokaiselle tapaukselle, jolla puuttuu y, määritetään joukko tapauksia, joissa on havaittu y, joiden ennustetut arvot ovat lähellä ennustettua arvoa tapaukselle, jolla puuttuu tietoja.

5. Näiden läheisten tapausten joukosta valitaan satunnaisesti yksi ja määritetään sen havaittu arvo korvaamaan puuttuva arvo.

6. Toistetaan vaiheita 2–5 jokaiselle täydennetylle aineistolle.

Pmm-algoritmin vaiheet osoitteesta (A. Paul (2015). *Imputation by Predictive Mean Matching: Promise and Peril*. URL: https://statisticalhorizons.com/predictive-mean-matching (viitattu 02. 03. 2020)).

Pmm-imputoinnissa käytetään Bayes-tilastotieteelle ominaista tapaa kuvata parametrejä satunnaismuuttujina, joilla on jakauma. Tästä jakaumasta otetaan, pmm-imputoinnin vaiheessa kaksi, satunnaisotos.

3.2 Ohjelmakirjastot Hmisc, pmm ja bootstrap

Hmisc on todella monipuolinen R-ohjelmiston kirjasto. Se sisältää funktioita niin puuttuvan datan käsittelyyn, kuin myös muihin data-analyysin osa-alueisiin. Hmisc sisältää moni-imputointiin funktion `aregImpute()`. `aregImpute()` on funktio, joka käyttää myös hyväkseen pmm-algoritmia, mutta sen koko imputointivaihe on hieman erilainen. Siinä otetaan aluksi bootstrap-otokset koko aineistosta ja luodaan niiden pohjalta jokaiselle imputointikierrokselle omat joustavat additiiviset mallit. Luotujen mallien pohjalta käytetään pmm-algoritmia puuttuvien arvojen imputoimiseksi.

Oletetaan nyt sama lähtötilanne, kuin pmm-agoritmissa. Tällöin `aregImpute()` funktion vaiheet ovat:

1. Otetaan bootstrap-otos koko aineistosta.
2. Luodaan otoksen pohjalta yleistetty additiivinen malli b*.
3. Luodaan b*:llä ennustetut arvot y:lle kaikissa tapauksissa, sekä tapauksissa, joissa y:n tieto puuttuu, että niissä, joissa on tietoa.
4. Jokaiselle tapaukselle, jolla puuttuu y, määritetään joukko tapauksia, joissa on havaittu y, joiden ennustetut arvot ovat lähellä ennustettua arvoa tapaukselle, jolla puuttuu tietoja.
5. Näiden läheisten tapausten joukosta valitaan satunnaisesti yksi ja määritetään sen havaittu arvo korvaamaan puuttuva arvo.
6. Toistetaan vaiheita 1–5 jokaiselle täydennetylle aineistolle.

`aregImpute()` käyttää bootstrap-otoksia aineistosta tuottaakseen satunnaisotoksia prediktiivisestä jakaumasta. Tämä on suurimpia eroja Mice:n pmm-algoritmiin, jossa pyritään samaan, ottamalla satunnaisotos vastaluodun mallin jakaumasta.

3.3 Amelia ja EM-algoritmi

Amelia on pakkauksena nimenomaan suunnattu moni-imputointiin. Sen imputointivaiheen oletus algoritmina on EM-algoritmi. Amelia siis olettaa, että data on multinormaalijakautunut. Tällöin data on jakautunut parametrein

$$\Theta = (\mu, \Sigma). \tag{3.1}$$

Jos merkitään, että D on koko aineisto ja D^{mis} sen puuttuva osa niin tällöin Θ posteorijakauma on muotoa

$$\int p(D|\Theta) dD^{mis} \tag{3.2}$$

(Honaker, Gary ja Blackwell 2011, s. 4) Tästä posteriorijakaumasta (3.2) Amelia ottaa satunnaisotoksia, aluksi ottamalla bootstrap-otoksia koko aineistosta ja sen jälkeen etsimällä posteriorijakauman moodia EM-algoritmilla. Kun Θ posteriorijakaumasta on saatu otos, tapahtuu imputointi näiden parametrien ja havaittujen arvojen perusteella.

EM-algoritmin (Expectation-Maximization eli odotusarvon maksimointi) toiminta perustuu nimensä mukaisesti suurimman uskottavuuden estimaattien löytämiseen tilastollisen mallin parametreille. Aluksi EM-algoritmissa asetetaan Θ arvoille jotkut alkuarvot Θ^0, jonka jälkeen siirrytään toistamaan kahta eri vaihetta. Nämä kaksi askelta ovat, niin kutsutut E-askel ja M-askel. E-askeleessa asetetaan puuttuvalle datalle arvot sen hetkisten Θ^t parametrien avulla ja lasketaan aineiston logaritminen uskottavuus. Tämän jälkeen siirrytään M-askeleeseen, jossa etsitään sellainen Θ^{t+1}, että uskottavuus maksimoituu. Näitä kahta askelta toistetaan, kunnes parametrissa Θ^t ei tapahdu enää muutosta.(McLachlan ja Krishnan 2007, s. 18–20)

4 Simulaatiotesti

Mise-, Hmisc- ja Ameliapakkausten testaamiseksi luotiin simulaatiotesti, jonka tarkoituksena oli tarkastella näiden moni-imputointitapojen toimivuutta käytännössä. Simulaatiotestin pohja on lainattu kirjasta Van Buuren 2012, s. 59. Testi suoritettiin luomalla MAR tyyppistä puuttuvaa dataa kokonaiseen aineistoon, jonka jälkeen koottiin yhden selittävän muuttujan regressiomallit kullakin jo esitetyllä moni-imputointi tavalla. Malli oli muotoa

$$Height = \beta_0 + \beta_1 * Weight \tag{4.1}$$

Mallin Height-muuttuja kuvaa miehistä mitattua pituutta ja Weight-muuttuja miehistä mitattua painoa. β_0 on regressiomallin vakio ja β_1 on muuttujan Weight regressiokerroin.

Luoduista malleista kerättiin tarkasteltavan Weight-muuttujan regressiokerroin ja sen 95%:n luottamusväli. Luottamus väli lasketaan kaavalla

$$\hat{\beta}_1 \pm t_{0.025,n-2} \frac{s}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}} \tag{4.2}$$

Luottamusväli saadaan siis t-jakaumaa käyttäen. Kaavassa s on jäännösvarianssin neliöjuuri ja n havaintoyksiköiden määrä.(Rencher ja Schaalje 2008, s. 128–133)

Testiä suoritettiin useita kertoja, jonka jälkeen kunkin moni-imputointitavan toimivuutta mitattiin viiden eri muuttujan avulla. Nämä muuttujat esitellään myöhemmin.

Käytetty aineisto oli peräisin ikääntyneiden miesten eturauhassyöpälääketutkimuksesta. Käsiteltyjä muuttujia oli 17 ja havaintoyksiköitä oli 297. Muuttujina oli muun muassa pituus, paino, sekä testosteronin ja kreatiinin määrä veressä. Kun aineistoa tarkastellaan Shapiro-Wilk -testin avulla (kuva (4.1)), nähdään, että vain käytännössä kaksi muuttujaa on normaalisti jakautunut(p-arvo > 0.05).

Amelian moni-imputointifunktio olettaa, että data olisi multinormaalijakautunut, joten tulos ei vaikuta optimaaliselta tälle moni-imputointi tavalle.

Tähän aineistoon luotiin puuttuvaa dataa paino-muuttujaan Micen tarjoaman `ampute()` funktion avulla (kuva (4.2)). Puuttuva data luotiin MAR-tyyppisesti.

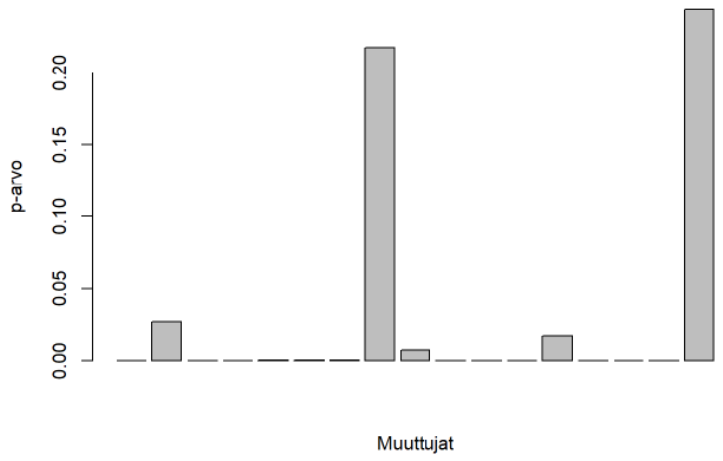
```
make.missing <- function(p = 0.5){
  #luodaan puuttuvaa dataa yhteen muuttujaan
  puuttuu <- ampute(kokoD,prop = p, mech = "MAR")
  patt <- puuttuu$patterns
  #poistetaan dataa nimenomaan Weight muuttujasta
  patti <- patt[,"Weight"]
  puuttuu2 <- ampute(kokoD, prop = p,patterns = patti, mech = "MAR")


  vajaaD <- puuttuu2$amp

  vajaaD
}
```

Kuva 4.2. Puuttuvan datan luonti aineistoon

```
l <- lapply(kokoD, shapiro.test)
lista = c()
for(item in l){lista = c(lista,item[2]$p.value)}
barplot(lista, xlab="Muuttujat", ylab = "p-arvo")
```



Kuva 4.1. Shapiro-Wilk -testin p-arvot aineistolle

Micen, Amelian ja Hmiscin moni-imputointifunktioille luotiin omat funktiot, jotka palauttavat tutkittavat muuttujat. Kuvassa (4.3) näkyy Micen moni-imputoinnilla tehty funktio. Tässä imputointivaihe tapahtui `Mice()` funktiolla. Analyysivaihe suoritettiin `with()` funktiolla ja kokoamisvaihe `pool()` funktiolla.

```
test.impute.mice <- function(data, m = 5, method = NULL) {

  imputoituData <- mice(data, m)

  impModelt <- with(imputoituData,lm(Height ~ Weight))
  impModel = pool(impModelt)
  tab <- summary(impModel, "all", conf.int = TRUE)
  as.numeric(tab["Weight", c("estimate", "2.5 %", "97.5 %")])
}
```

Kuva 4.3. Micen moni-imputointi

Näiden edellä luotujen funktioiden avulla saatiin kokoon funktio, joka luo aluksi käytettyyn dataan puuttuvaa tietoa ja sen jälkeen listaa saadut tulokset jokaisella imputointitavalla (4.4).

Tällä luodulla funktiolla voidaan säädellä, kuinka paljon puuttuvaa dataa luodaan Weight muuttujaan ja montako kertaa testiä toistetaan. Funktio palauttaa tulokset sisältävän taulukon.

Testin palauttamasta taulukosta laskettiin jokaiselle moni-imputointitavalle viisi sen suoriutumista kuvaavaa arvoa. Ensimmäinen arvo oli keskimääräinen ero täydestä aineistosta luodun regressiomallin ja estimoidun mallin, Weight-muuttujan regressiokertoimen välillä (RB). Toisena arvona laskettiin arvon RB ero prosentteina (PB). Kolmantena arvona laskettiin prosenttiosuus, jolla moni-imputointifunktiolla esti-

```
simulate <- function(runs = 10, missingp = 0.5 ,m = 5) {
  res <- array(NA, dim = c(3, runs, 3))
  dimnames(res) <- list(c("Mice", "Hmisc", "Amelia"),
                        as.character(1:runs),
                        c("estimate", "2.5 %","97.5 %"))
  for(run in 1:runs) {
    data <- make.missing(p = missingp)
    res[1, run, ] <- test.impute.mice(data, m = m)
    res[2, run, ] <- test.impute.hmisc(data, m = m)
    res[3, run, ] <- test.impute.amelia(data, m = m)
  }
  res
}
```

Kuva 4.4. koottu simulaatiotesti

moidun regressiokertoimen arvon 95%:n luottamusvälille koko aineistosta laskettu regressiokerroin osui (CR). Neljäntenä arvona oli tämän moni-imputointifunktiolla saadun regressiokertoimen 95%:n luottamusvälin keskimääräinen pituus (AW). Viides laskettu arvo oli keskipoikkeama moni-imputointifunktiolla saadun kertoimen ja koko aineistosta lasketun kertoimen välillä (RMSE).

Testi suoritettiin kolmella eri tavalla. Puuttuvan datan prosenttiosuus Weight-muuttujassa oli 50%, 30% ja 10%. Jokaisella kerralla testi tehtiin tuhat kertaa ja niiden pohjalta laskettiin edellä esitellyt arvot.

```r
set.seed(1919)
res <- simulate(1000, missingp = 0.5, m = 5)
```

```r
apply(res, c(1, 3), mean, na.rm = TRUE)
```

```
##        estimate 2.5 % 97.5 %
## Mice      0.230 0.146  0.314
## Hmisc     0.223 0.145  0.301
## Amelia    0.238 0.136  0.339
```

```r
#Katsotaan mitkä ovat alkuperäisen datan tulokset.
kokoModel <- lm(Height ~ Weight, data = kokoD)
summary(kokoModel)
```

```
##
## Call:
## lm(formula = Height ~ Weight, data = kokoD)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -17.141  -3.626   0.344   4.056  20.421
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 152.6021     2.0609   74.05   <2e-16 ***
## Weight        0.2575     0.0262    9.84   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.76 on 295 degrees of freedom
## Multiple R-squared:  0.247,  Adjusted R-squared:  0.245
## F-statistic: 96.8 on 1 and 295 DF,  p-value: <2e-16
```

```r
#Luodaan taulukko tuloksista.
true <- 0.25749
RB <- rowMeans(res[,, "estimate"]) - true
PB <- 100 * abs((rowMeans(res[,, "estimate"]) - true)/ true)
CR <- rowMeans(res[,, "2.5 %"] < true & true < res[,, "97.5 %"])
AW <- rowMeans(res[,, "97.5 %"] - res[,, "2.5 %"])
RMSE <- sqrt(rowMeans((res[,, "estimate"] - true)^2))
data.frame(RB, PB, CR, AW, RMSE)
```

```
##             RB    PB    CR    AW   RMSE
## Mice   -0.0275 10.67 0.851 0.168 0.0523
## Hmisc  -0.0346 13.43 0.823 0.156 0.0533
## Amelia -0.0199  7.74 0.955 0.204 0.0455
```

Kuva 4.5. Koko datasta laskettu lineaarinen regressiomalli ja 50%:n puuttuvan datan testi

Kuvassa alimman taulukon pmm on Mice:n moni-imputointifunktion tulos. Kaikkien pakkausten antamat tulokset lineaarisen regressiomallin Weight muuttujan kertoimeksi, olivat yleisesti alakanttiin ja Amelian antama tulos oli keskimäärin lähimpänä. Amelian moni-imputointifunktiolla kokoaineistosta laskettu kerroin osui myös useimpaan, sen 95%:n luottamusvälille, 95.5% kerroista. Micen ja Hmisc:in tulos tässä on selvästi heikompaa ja CR oli Micellä 85.1% ja Hmisc:illä 82.3%. Hmisc:in ja Micen tulos verrattuna Ameliaan oli siis yli 10 prosenttiyksikköä heikompaa CR arvoa tarkasteltaessa. Tätä selittää osittain Amelian keskimääräisen luottamusvälin suuruus (AW), joka on huomattavasti isompi, kuin Hmisc:illä ja Amelialla.

```r
res <- simulate(1000, missingp = 0.3, m = 5)
```

```r
apply(res, c(1, 3), mean, na.rm = TRUE)
```

```
##        estimate 2.5 % 97.5 %
## Mice      0.237 0.163  0.311
## Hmisc     0.235 0.166  0.303
## Amelia    0.249 0.177  0.321
```

```r
#Luodaan taulukko tuloksista.
true <- 0.25749
RB <- rowMeans(res[,, "estimate"]) - true
PB <- 100 * abs((rowMeans(res[,, "estimate"]) - true)/ true)
CR <- rowMeans(res[,, "2.5 %"] < true & true < res[,, "97.5 %"])
AW <- rowMeans(res[,, "97.5 %"] - res[,, "2.5 %"])
RMSE <- sqrt(rowMeans((res[,, "estimate"] - true)^2))
data.frame(RB, PB, CR, AW, RMSE)
```

```
##              RB   PB    CR    AW   RMSE
## Mice   -0.02074 8.06 0.968 0.148 0.0378
## Hmisc  -0.02263 8.79 0.955 0.137 0.0372
## Amelia -0.00862 3.35 0.991 0.143 0.0302
```

Kuva 4.6. 30%:n puuttuvan datan testi

Kun puuttuvan datan osuutta lasketaan testissä 30%:n, paranevat myös tulokset moni-imputointi malleilla. Amelian 95%:n luottamusvälin pituus lyheni huomattavasti edelliseen testiin verrattuna, noin 0.20:sta noin 0.14:sta. Huomattavaa on myös, että Micen ja Hmisc:in CR arvo nousi yli kymmenen prosenttiyksikköä.

```r
res <- simulate(1000, missingp = 0.1, m = 5)
```

```r
apply(res, c(1, 3), mean, na.rm = TRUE)
```

```
##        estimate 2.5 % 97.5 %
## Mice      0.249 0.190  0.309
## Hmisc     0.248 0.190  0.307
## Amelia    0.255 0.199  0.310
```

```r
#Luodaan taulukko tuloksista.
true <- 0.25749
RB <- rowMeans(res[,, "estimate"]) - true
PB <- 100 * abs((rowMeans(res[,, "estimate"]) - true)/ true)
CR <- rowMeans(res[,, "2.5 %"] < true & true < res[,, "97.5 %"])
AW <- rowMeans(res[,, "97.5 %"] - res[,, "2.5 %"])
RMSE <- sqrt(rowMeans((res[,, "estimate"] - true)^2))
data.frame(RB, PB, CR, AW, RMSE)
```

```
##              RB   PB CR    AW   RMSE
## Mice   -0.00815 3.17  1 0.119 0.0178
## Hmisc  -0.00899 3.49  1 0.117 0.0177
## Amelia -0.00295 1.15  1 0.110 0.0146
```

Kuva 4.7. 10%:n puuttuvan datan testi

Laskettaessa puuttuvan datan osuus 10%:iin, ei muutokset ole niin dramaattisia enään kuin 50%:n ja 30%:n välillä. Tällöin kaikkien pakkausten CR arvo on tasan 100%, ja keskimääräinen ero koko aineistosta laskettuun regressiokertoimeen todella pieni.

5 Päätelmät

Simulaatiotestin perusteella ei ole mahdollista tehdä johtopäätöstä, että joku esitetyistä moni-imputointitavoista olisi muita parempi. Simulaatiotestissä tuli esille Amelia-pakkauksen vahva suoriutuminen, siitä huolimatta, että muuttujat aineistossa eivät yleisesti olleet normaalisti jakautuneita ja Amelia-pakkauksen moni-imputointifunktio olettaa teoreettisesti datan multinormaalijakautuneisuutta. Näin ollen Amelian EM-algoritmiin perustuvaa moni-imputointia ei välttämättä kannata hylätä, vaikka data ei olisikaan täysin multinormaalijakautunutta.

Käytettävyydeltään Mice-pakkaus oli selvästi sujuvin ja se tarjoaa erittäin kattavasti myös vaihtoehtoisia imputointitapoja. Micellä ja Hmisc:illa saadut tulokset olivat hyvin samantyyppisiä ja sille yksi selittävä tekijä voi olla niiden todella samantyyppiset moni-imputointialgoritmit.

Koko moni-imputointitestin heikkoutena on kuitenkin vakioitu data, johon puuttuva data luodaan. Tällöin ei voida verrata miten käsitellyt moni-imputointifunktiot toimivat erilaisten jakaumien tapauksissa.

Lähteet

Enders, C. (2010). *Applied Missing Data Analysis*. 1. painos. Guilford Publications.

Honaker, J., K. Gary ja M. Blackwell (2011). ”Amelia II: A Program for Missing Data”. With comments. *JournalofStatisticalSoftware* 45.

McLachlan, G. J. ja T. Krishnan (2007). *The EM Algorithm and Extensions*. 2. painos. John Wiley & Sons.

Paul, A. (2015). *Imputation by Predictive Mean Matching: Promise and Peril*. URL: https://statisticalhorizons.com/predictive-mean-matching (viitattu 02. 03. 2020).

Rencher, A. C. ja G. B. Schaalje (2008). *Linear models in statistics*. 2. painos. John Wiley & Sons.

Van Buuren, S. (2012). *Flexible Imputation of Missing Data*. 1. painos. Chapman ja Hall/CRC.

Liitteet

Ladataan käytettävät pakkaukset ja aineisto, joka muunnetaan muotoon, jota voidaan testeissä käyttää.

```
library(mice)
library(dplyr)
library(varhandle)
library(eeptools)
library(Hmisc)
library(Zelig)
library(Amelia)

options(digits = 3)

ekat <- read.table("C:\\Users\\jaska\\OneDrive - TUNI.fi\\kandi\\data
\\ekatTiedot.txt", header = TRUE)
tokat <- read.table("C:\\Users\\jaska\\OneDrive - TUNI.fi\\kandi\\data
\\tokatTiedot.txt",header = TRUE)
ekatKaks <- merge(ekat, tokat, by = "Patient_no")
kolmas <- read.table("C:\\Users\\jaska\\OneDrive - TUNI.fi\\kandi\\data
\\kolmannetTiedot.txt",header = TRUE)
kokoD <- merge(ekatKaks, kolmas, by = "Patient_no")

kokoD <- na.omit(kokoD)

kokoD <- unfactor(kokoD)
kokoD$Prostate <- as.numeric(gsub(",", ".", gsub("\\.", "", kokoD$Prostate)))
kokoD$Testosterone <- as.numeric(gsub(",", ".", gsub("\\.", "", kokoD$Testosterone)))
kokoD$Haematocrit <- as.numeric(gsub(",", ".", gsub("\\.", "", kokoD$Haematocrit)))
kokoD$White_cells <- as.numeric(gsub(",", ".", gsub("\\.", "", kokoD$White_cells)))
kokoD$Erythrocytes <- as.numeric(gsub(",", ".", gsub("\\.", "", kokoD$Erythrocytes)))
kokoD$Corpuscular_volume <- as.numeric(gsub(",", ".", gsub("\\.", ""
, kokoD$Corpuscular_volume)))
kokoD$Corpuscular_haemaglobin <- as.numeric(gsub(",", ".", gsub("\\.", ""
, kokoD$Corpuscular_haemaglobin)))
kokoD$Concentration <- as.numeric(gsub(",", ".", gsub("\\.", "", kokoD$Concentration)))
kokoD$pH <- as.numeric(gsub(",", ".", gsub("\\.", "", kokoD$pH)))
kokoD$Height <- as.numeric(gsub(",", ".", gsub("\\.", "", kokoD$Height)))
kokoD$Weight <- as.numeric(gsub(",", ".", gsub("\\.", "", kokoD$Weight)))
kokoD$Width <- as.numeric(gsub(",", ".", gsub("\\.", "", kokoD$Width)))
kokoD$Length <- as.numeric(gsub(",", ".", gsub("\\.", "", kokoD$Length)))

#Lasketaan päivämäärästä ja syntymäajasta henkilön ikä.
kokoD$Date <- as.Date(kokoD$Date, "%d.%m.%Y")
kokoD$Birthdate <- as.Date(kokoD$Birthdate, "%d.%m.%Y")
kokoD <- mutate(kokoD, age = age_calc(kokoD$Birthdate,
enddate= kokoD$Date, units = "years"))
kokoD$age <- as.integer(kokoD$age)

kokoD <- kokoD[,!(names(kokoD) %in% c("Birthdate" , "Date"))]
#Poistetaan muuttujat, joita ei haluta käyttää.
kokoD = subset(kokoD, select = -c(Patient_no,Visit.x,Visit.y,Visit) )
```

```r
#Yhdellä rivillä puuttuu dataa, joten poistetaan se asioiden helpottamiseksi.
kokoD <- na.omit(kokoD)
```

Testifunktioiden luonti

```r
#Funktio, jolla luodaan MAR-tyyppistä puuttuvaa dataa.
make.missing <- function(p = 0.5){
  #Luodaan puuttuvaa dataa yhteen muuttujaan.
  puuttuu <- ampute(kokoD,prop = p, mech = "MAR")
  patt <- puuttuu$patterns
  #Poistetaan dataa nimenomaan Weight-muuttujasta.
  patti <- patt[,"Weight"]
  puuttuu2 <- ampute(kokoD, prop = p,patterns = patti, mech = "MAR")


  vajaaD <- puuttuu2$amp

  vajaaD
}

#Micen moni-imputointi
test.impute.mice <- function(data, m = 5, method = NULL) {

  imputoituData <- mice(data, m)

  impModelt <- with(imputoituData,lm(Height ~ Weight))
  impModel = pool(impModelt)
  tab <- summary(impModel, "all", conf.int = TRUE)
  as.numeric(tab["Weight", c("estimate", "2.5 %", "97.5 %")])
}
#hmisc:in moni-imputointi
test.impute.hmisc <- function(data, m = 5){
  dformula = (Weight ~ Prostate + Testosterone +
              Alkaline + Creatinine + Haematocrit + Haemoglobin + White_cells +
              Erythrocytes + Corpuscular_volume + Corpuscular_haemaglobin +
              Concentration + pH + Height + Width + Length + age)
  f <- aregImpute(formula = dformula, n.impute=m, data=data, nk = 0)

  lmalli <- fit.mult.impute(Height ~ Weight,lm, f, data=data)
  tab <- lmalli$coefficients
  a <- as.numeric(tab["Weight"])
  tab2 <- confint(lmalli)
  b <- tab2["Weight","2.5 %"]
  c <- tab2["Weight","97.5 %"]
  c(a,b,c)
}
#Amelian moni-imputointi
test.impute.amelia <- function(data, m= 5){
  tulos <- amelia( data , m = m)
  lm.amelia.out <- lapply(tulos$imputations, function(i) lm(Height ~ Weight , data = i))
  impModel = pool(lm.amelia.out)
  tab <- summary(impModel, "all", conf.int = TRUE)
```

```
  as.numeric(tab["Weight", c("estimate", "2.5 %", "97.5 %")])
}
```

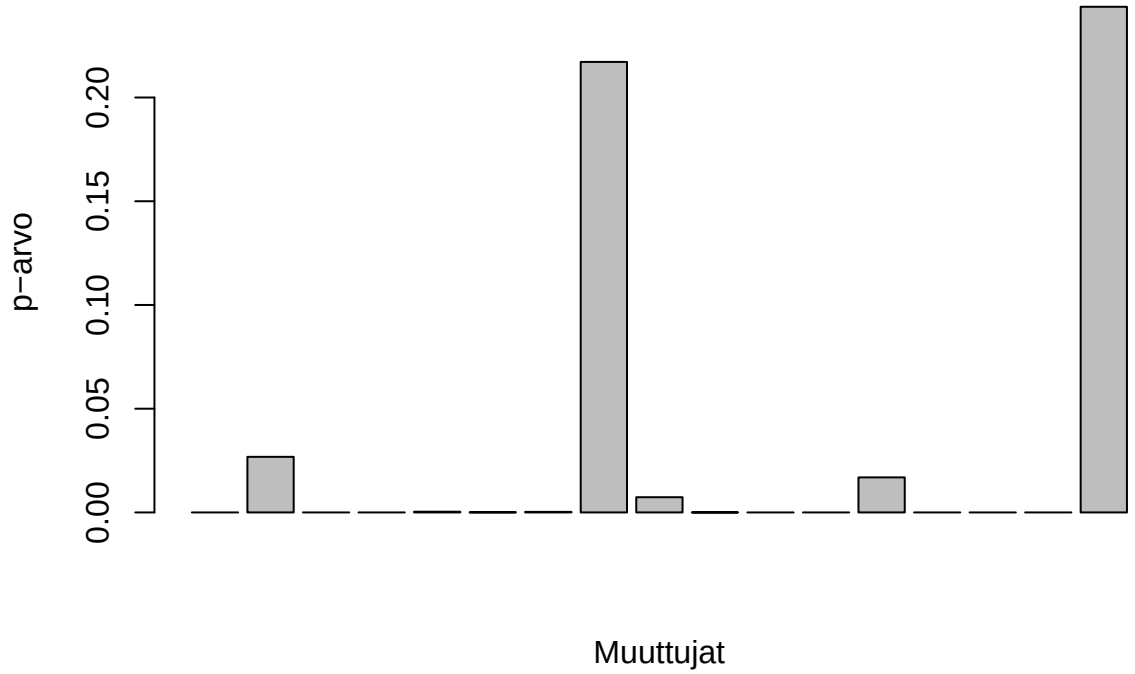
Pääfunktio

Luodaan funktio, jossa kutsutaan edellä luotuja funktioita ja joka pistää tulokset kasaan.

```
simulate <- function(runs = 10, missingp = 0.5 ,m = 5) {
  res <- array(NA, dim = c(3, runs, 3))
  dimnames(res) <- list(c("Mice", "Hmisc", "Amelia"),
                        as.character(1:runs),
                        c("estimate", "2.5 %","97.5 %"))
  for(run in 1:runs) {
    data <- make.missing(p = missingp)
    res[1, run, ] <- test.impute.mice(data, m = m)
    res[2, run, ] <- test.impute.hmisc(data, m = m)
    res[3, run, ] <- test.impute.amelia(data, m = m)
  }
  res
}
```

Tarkastellaan datan normaalisti jakautuneisuutta

```
l <- lapply(kokoD, shapiro.test)
lista = c()
for(item in l){lista = c(lista,item[2]$p.value)}
barplot(lista, xlab="Muuttujat", ylab = "p-arvo")
```



Ajetaan testit ja tulostetaan tulokset

```r
set.seed(1919)
res <- simulate(1000, missingp = 0.5, m = 5)
```

```r
apply(res, c(1, 3), mean, na.rm = TRUE)
```

```
##        estimate 2.5 % 97.5 %
## Mice      0.230 0.146  0.314
## Hmisc     0.223 0.145  0.301
## Amelia    0.238 0.136  0.339
```

```r
#Katsotaan mitkä ovat alkuperäisen datan tulokset.
kokoModel <- lm(Height ~ Weight, data = kokoD)
summary(kokoModel)
```

```
##
## Call:
## lm(formula = Height ~ Weight, data = kokoD)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -17.141  -3.626   0.344   4.056  20.421
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 152.6021     2.0609   74.05   <2e-16 ***
## Weight        0.2575     0.0262    9.84   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.76 on 295 degrees of freedom
## Multiple R-squared:  0.247,  Adjusted R-squared:  0.245
## F-statistic: 96.8 on 1 and 295 DF,  p-value: <2e-16
```

```r
#Luodaan taulukko tuloksista.
true <- 0.25749
RB <- rowMeans(res[,, "estimate"]) - true
PB <- 100 * abs((rowMeans(res[,, "estimate"]) - true)/ true)
CR <- rowMeans(res[,, "2.5 %"] < true & true < res[,, "97.5 %"])
AW <- rowMeans(res[,, "97.5 %"] - res[,, "2.5 %"])
RMSE <- sqrt(rowMeans((res[,, "estimate"] - true)^2))
data.frame(RB, PB, CR, AW, RMSE)
```

```
##             RB    PB    CR    AW   RMSE
## Mice   -0.0275 10.67 0.851 0.168 0.0523
## Hmisc  -0.0346 13.43 0.823 0.156 0.0533
## Amelia -0.0199  7.74 0.955 0.204 0.0455
```

```r
res <- simulate(1000, missingp = 0.3, m = 5)
```

```r
apply(res, c(1, 3), mean, na.rm = TRUE)
```

```
##        estimate 2.5 % 97.5 %
## Mice      0.237 0.163  0.311
## Hmisc     0.235 0.166  0.303
## Amelia    0.249 0.177  0.321
```

```r
#Luodaan taulukko tuloksista.
true <- 0.25749
RB <- rowMeans(res[,, "estimate"]) - true
PB <- 100 * abs((rowMeans(res[,, "estimate"]) - true)/ true)
CR <- rowMeans(res[,, "2.5 %"] < true & true < res[,, "97.5 %"])
AW <- rowMeans(res[,, "97.5 %"] - res[,, "2.5 %"])
RMSE <- sqrt(rowMeans((res[,, "estimate"] - true)^2))
data.frame(RB, PB, CR, AW, RMSE)
```

```
##              RB   PB    CR    AW   RMSE
## Mice   -0.02074 8.06 0.968 0.148 0.0378
## Hmisc  -0.02263 8.79 0.955 0.137 0.0372
## Amelia -0.00862 3.35 0.991 0.143 0.0302
```

```r
res <- simulate(1000, missingp = 0.1, m = 5)
```

```r
apply(res, c(1, 3), mean, na.rm = TRUE)
```

```
##        estimate 2.5 % 97.5 %
## Mice      0.249 0.190  0.309
## Hmisc     0.248 0.190  0.307
## Amelia    0.255 0.199  0.310
```

```r
#Luodaan taulukko tuloksista.
true <- 0.25749
RB <- rowMeans(res[,, "estimate"]) - true
PB <- 100 * abs((rowMeans(res[,, "estimate"]) - true)/ true)
CR <- rowMeans(res[,, "2.5 %"] < true & true < res[,, "97.5 %"])
AW <- rowMeans(res[,, "97.5 %"] - res[,, "2.5 %"])
RMSE <- sqrt(rowMeans((res[,, "estimate"] - true)^2))
data.frame(RB, PB, CR, AW, RMSE)
```

```
##              RB   PB CR    AW   RMSE
## Mice   -0.00815 3.17  1 0.119 0.0178
## Hmisc  -0.00899 3.49  1 0.117 0.0177
## Amelia -0.00295 1.15  1 0.110 0.0146
```