1 Modified R Functions from GAMLSS

Read in the NHANES data

```
require(epicalc, warn.conflicts = FALSE)

## Loading required package:  epicalc
## Loading required package:  foreign
## Loading required package:  survival
## Loading required package:  splines
## Loading required package:  MASS
## Loading required package:  nnet

require(gamlss, warn.conflicts = FALSE)

## Loading required package:  gamlss
## Loading required package:  gamlss.data
## Loading required package:  gamlss.dist
## Loading required package:  nlme
##  **********  GAMLSS Version 4.3-0  **********
## For more on GAMLSS look at http://www.gamlss.org/
## Type gamlssNews() to see new features/changes/bug fixes.

require(lattice, warn.conflicts = FALSE)

## Loading required package:  lattice

require(car, warn.conflicts = FALSE)

## Loading required package:  car

require(xtable, warn.conflicts = FALSE)

## Loading required package:  xtable

require(foreign, warn.conflicts = FALSE)
```

2 Modified Functions from GAMLSS Package

```
mycent <- function(obj, xvar = NULL, cent = c(1, 3, 5, 15, 25, 50, 75, 85, 95,
    97, 99), legend = TRUE, ylab = "y", xlab = "x", main = NULL, main.gsub = "@",
    xleg = min(xvar), yleg = max(obj$y), xlim = range(xvar), ylim = range(obj$y),
    save = FALSE, plot = TRUE, points = TRUE, pch = "+", col = "blue", col.centiles = 1:l
        2, lty.centiles = 1, lwd.centiles = 1, ...) {
    if (!is.gamlss(obj))
        stop(paste("This is not an gamlss object", "\n", ""))
    if (is.null(xvar))
        stop(paste("The xvar argument is not specified", "\n", ""))
    fname <- obj$family[1]
    qfun <- paste("q", fname, sep = "")
    Title <- paste("Centile curves using", fname, sep = " ")
```

```
    main <- if (is.null(main))
        paste("Centile curves using", fname, sep = " ") else gsub(main.gsub, Title, main)
    oxvar <- xvar[order(xvar)]
    oyvar <- obj$y[order(xvar)]
    if (is.matrix(obj$y)) {
        oyvar <- obj$y[, 1][order(xvar)]
        ylim <- range(obj$y[, 1])
        yleg = max(obj$y[, 1])
    }
    if (plot) {
        lty.centiles <- rep(lty.centiles, length(cent))
        lwd.centiles <- rep(lwd.centiles, length(cent))
        col.centiles <- rep(col.centiles, length(cent))
        if (points == TRUE) {
            plot(oxvar, oyvar, type = "p", col = col, pch = pch, xlab = xlab,
                ylab = ylab, xlim = xlim, ylim, ...)
        } else {
            plot(oxvar, oyvar, type = "n", col = col, pch = pch, xlab = xlab,
                ylab = ylab, xlim = xlim, ylim, ...)
        }
        title(main)
    }
    col <- 3
    lpar <- length(obj$parameters)
    ii <- 0
    per <- rep(0, length(cent))
    for (var in cent) {
        if (lpar == 1) {
            newcall <- call(qfun, var/100, mu = fitted(obj, "mu")[order(xvar)])
        } else if (lpar == 2) {
            newcall <- call(qfun, var/100, mu = fitted(obj, "mu")[order(xvar)],
                sigma = fitted(obj, "sigma")[order(xvar)])
        } else if (lpar == 3) {
            newcall <- call(qfun, var/100, mu = fitted(obj, "mu")[order(xvar)],
                sigma = fitted(obj, "sigma")[order(xvar)], nu = fitted(obj,
                  "nu")[order(xvar)])
        } else {
            newcall <- call(qfun, var/100, mu = fitted(obj, "mu")[order(xvar)],
                sigma = fitted(obj, "sigma")[order(xvar)], nu = fitted(obj,
                  "nu")[order(xvar)], tau = fitted(obj, "tau")[order(xvar)])
        }
        ii <- ii + 1
        ll <- eval(newcall)
        if (plot) {
            lines(oxvar, ll, col = col.centiles[ii], lty = lty.centiles[ii],
                lwd = lwd.centiles[ii], ...)
        }
        per[ii] <- (1 - sum(oyvar > ll)/length(oyvar)) * 100
        if (!save)
            cat("% of cases below ", var, "centile is ", per[ii], "\n")
```

```
    }
    if (plot) {
        if (legend == TRUE)
            legend(list(x = xleg, y = yleg), legend = cent, col = col.centiles,
                lty = lty.centiles, lwd = lwd.centiles, ncol = 1, ...)
    }
    if (save) {
        # return(cbind(cent, per))
        TAU = tau
        L = nu
        Median = mu
        S = sigma
        return(cbind(L, Median, S, cent, per))
    }
}
```

```
mycent.pred <- function(obj, type = c("centiles", "z-scores", "standard-centiles"),
    xname = NULL, xvalues = NULL, power = NULL, yval = NULL, cent = c(1, 3,
        5, 15, 25, 50, 75, 85, 95, 97, 99), dev = c(-4, -3, -2, -1, 0, 1, 2,
        3, 4), plot = FALSE, legend = TRUE, ...) {
    calc.cent <- function(xvar, cent) {
        o <- order(xvar)
        mat <- xvar[o]
        cent <- cent
        for (var in cent) {
            if (lpar == 1) {
                newcall <- call(qfun, var/100, mu = mu[o])
            } else if (lpar == 2) {
                newcall <- call(qfun, var/100, mu = mu[o], sigma = sigma[o])
            } else if (lpar == 3) {
                newcall <- call(qfun, var/100, mu = mu[o], sigma = sigma[o],
                  nu = nu[o])
            } else {
                newcall <- call(qfun, var/100, mu = mu[o], sigma = sigma[o],
                  nu = nu[o], tau = tau[o])
            }
            ll <- eval(newcall)
            mat <- cbind(mat, ll)
        }
        mat <- as.data.frame(mat)
        nnn <- paste("C", as.character(cent), sep = "")
        names(mat) <- c(xname, nnn)
        return(mat)
    }
    plot.mat <- function(mat, cent, legend, ...) {
        lcent <- dim(mat)[2]
        xleg <- min(mat[, 1])
        yleg <- max(mat[, 2:lcent])
        plot(mat[, 1], mat[, 2], type = "n", ...)
        for (i in 2:lcent) lines(mat[, 1], mat[, i], col = i)
```

```
    if (legend)
        legend(list(x = xleg, y = yleg), legend = cent, col = c(2, 3, 4,
            5, 6, 7, 8, 9, 10, 11, 12), lty = 1, ncol = 1, bg = "white")
    invisible()
}
if (!is.gamlss(obj))
    stop(paste("This is not an gamlss object", "\n", ""))
if (is.null(xvalues))
    stop(paste("The xvalues  argument is not specified", "\n", ""))
if (is.null(xname))
    stop(paste("The xname argument is not specified", "\n", ""))
if (!is.character(xname))
    stop(paste("The xname argument is not a character", "\n", ""))
xvar <- if (!is.null(power))
    xvar <- xvalues^power else xvalues
newx <- data.frame(xvar)
colnames(newx) <- xname
lpar <- length(obj$parameters)
if ("mu" %in% obj$parameters) {
    if (is.null(obj$mu.fix))
        mu <- predict(obj, what = "mu", newdata = newx, type = "response",
            ...) else if (obj$mu.fix == TRUE)
        mu <- rep(fitted(obj, "mu")[1], length(xvar))
}
if ("sigma" %in% obj$parameters) {
    if (is.null(obj$sigma.fix))
        sigma <- predict(obj, what = "sigma", newdata = newx, type = "response",
            ...) else if (obj$sigma.fix == TRUE)
        sigma <- rep(fitted(obj, "sigma")[1], length(xvar))
}
if ("nu" %in% obj$parameters) {
    if (is.null(obj$nu.fix))
        nu <- predict(obj, what = "nu", newdata = newx, type = "response",
            ...) else if (obj$nu.fix == TRUE)
        nu <- rep(fitted(obj, "nu")[1], length(xvar))
}
if ("tau" %in% obj$parameters) {
    if (is.null(obj$tau.fix))
        tau <- predict(obj, what = "tau", newdata = newx, type = "response",
            ...) else if (obj$tau.fix == TRUE)
        tau <- rep(fitted(obj, "tau")[1], length(xvar))
}
type <- match.arg(type)
if (type == "centiles") {
    fname <- obj$family[1]
    qfun <- paste("q", fname, sep = "")
    xvar <- xvalues
    mat <- calc.cent(xvar = xvar, cent = cent)
    if (plot)
        plot.mat(mat, cent, legend, ...)
```

```
        # return(mat)
        TAU = tau
        L = nu
        Median = mu
        S = sigma
        Age <- mat[, 1]
        return(cbind(Age, L, Median, S, mat[, -1]))
    }
    if (type == "z-scores") {
        if (is.null(yval))
            stop("the y values should be set if type=z-scores is used")
        if (length(yval) != length(xvalues))
            stop("length of xvalues and yval is not the same")
        fname <- obj$family[1]
        qfun <- paste("p", fname, sep = "")
        if (lpar == 1) {
            newcall <- call(qfun, yval, mu = mu)
        } else if (lpar == 2) {
            newcall <- call(qfun, yval, mu = mu, sigma = sigma)
        } else if (lpar == 3) {
            newcall <- call(qfun, yval, mu = mu, sigma = sigma, nu = nu)
        } else {
            newcall <- call(qfun, yval, mu = mu, sigma = sigma, nu = nu, tau = tau)
        }
        cdf <- eval(newcall)
        rqres <- qnorm(cdf)
        return(rqres)
    }
    if (type == "standard-centiles") {
        cent <- pnorm(dev) * 100
        fname <- obj$family[1]
        qfun <- paste("q", fname, sep = "")
        xvar <- xvalues
        mat <- calc.cent(xvar = xvar, cent = cent)
        nnn <- paste(as.character(dev), sep = "")
        names(mat) <- c(xname, nnn)
        if (plot)
            plot.mat(mat, dev, legend, ...)
        # return(mat)
        TAU = tau
        L = nu
        Median = mu
        S = sigma
        Age <- mat[, 1]
        return(cbind(Age, L, Median, S, mat[, -1]))
    }
}
```

3 Reading NHANES data

```
d2 <- read.dta("hes2hes3nhanes1.dta")
attach(d2)
des(d2)

##
##  No. of observations =  23118
##   Variable      Class
## 1 muac          integer
## 2 muac_left_arm integer
## 3 dataset       character
## 4 agey          numeric
## 5 sex           factor
## 6 muac_female   integer
## 7 muac_male     integer
## 8 l_r_diff      numeric
##   Description
## 1 R arm was standard measure'
## 2 subset left arm measured
## 3
## 4 age calculated from dob/date (NHANES) or months of age (HES 2 &3)
## 5
## 6 muac, sex == female
## 7 muac, sex == male'
## 8 difference between R and L arm

str(d2)

## 'data.frame': 23118 obs. of  8 variables:
##  $ muac         : int  151 173 152 125 159 160 151 164 142 162 ...
##  $ muac_left_arm: int  NA NA NA NA NA NA NA 166 NA NA ...
##  $ dataset      : chr  "NHANES 1" "NHANES 1" "NHANES 1" "NHANES 1" ...
##  $ agey         : num  1.98 1.98 1.99 1.99 1.99 ...
##  $ sex          : Factor w/ 2 levels "female","male'": 2 2 2 1 1 1 2 1 2 1 ...
##  $ muac_female  : int  NA NA NA 125 159 160 NA 164 NA 162 ...
##  $ muac_male    : int  151 173 152 NA NA NA 151 NA 142 NA ...
##  $ l_r_diff     : num  NA NA NA NA NA NA NA -2 NA NA ...
##  - attr(*, "datalabel")= chr ""
##  - attr(*, "time.stamp")= chr "23 Mar 2011 10:11"
##  - attr(*, "formats")= chr  "%9.0g" "%9.0g" "%9s" "%9.0g" ...
##  - attr(*, "types")= int  252 252 8 254 251 252 252 254
##  - attr(*, "val.labels")= chr  "" "" "" "" ...
##  - attr(*, "var.labels")= chr  "R arm was standard measure'" "subset left arm measured
##  - attr(*, "expansion.fields")=List of 2
##   ..$ : chr  "_dta" "_lang_c" "default"
##   ..$ : chr  "_dta" "_lang_list" "default"
##  - attr(*, "version")= int 12
##  - attr(*, "label.table")=List of 1
##   ..$ sex: Named int  1 2
##   .. ..- attr(*, "names")= chr  "female" "male'"
```

```
summ(d2)

##
##
## No. of observations = 23118
##
##   Var. name     obs.  mean   median  s.d.   min.   max.
## 1 muac          23118 227.29 222     50.35  59     499
## 2 muac_left_arm 1788  229.82 226     57.41  107    485
## 3 dataset
## 4 agey          23118 12.29  12.08   5.34   1.98   26.02
## 5 sex           23118 1.491  1       0.5    1      2
## 6 muac_female   11761 226.35 224     48.22  59     499
## 7 muac_male     11357 228.26 220     52.45  105    483
## 8 l_r_diff      1788  1.62   1       8.48   -92    110
```

Convert muac to cm and age to months. Drop if age is less than 5 and greater than 25

```
d2$muac <- d2$muac/10  # convert to mm
d2$agem <- d2$agey * 12  # convert age to months
summ(d2)

##
##
## No. of observations = 23118
##
##   Var. name     obs.  mean   median  s.d.   min.   max.
## 1 muac          23118 22.73  22.2    5.03   5.9    49.9
## 2 muac_left_arm 1788  229.82 226     57.41  107    485
## 3 dataset
## 4 agey          23118 12.29  12.08   5.34   1.98   26.02
## 5 sex           23118 1.491  1       0.5    1      2
## 6 muac_female   11761 226.35 224     48.22  59     499
## 7 muac_male     11357 228.26 220     52.45  105    483
## 8 l_r_diff      1788  1.62   1       8.48   -92    110
## 9 agem          23118 147.53 145     64.07  23.75  312.25
```

Keep data only for ages 2-25 years

```
d2 = d2[d2$agem >= 24 & d2$agem <= 300, c(5, 1, 9)]
summ(d2)

##
##
## No. of observations = 22699
##
##   Var. name obs.  mean   median  s.d.   min.   max.
## 1 sex       22699 1.494  1       0.5    1      2
## 2 muac      22699 22.62  22.1    4.98   5.9    49.9
## 3 agem      22699 144.72 143     60.92  24.05  299.99

tab1(d2$sex, graph = FALSE)
```

```
## d2$sex :
##         Frequency Percent Cum. percent
## female      11481    50.6         50.6
## male'       11218    49.4        100.0
##   Total     22699   100.0        100.0

d2$sex <- factor(d2$sex, labels = c("Female", "Male"))
tab1(d2$sex, graph = FALSE)

## d2$sex :
##         Frequency Percent Cum. percent
## Female      11481    50.6         50.6
## Male        11218    49.4        100.0
##   Total     22699   100.0        100.0
```

3.1 Create two datasets, one for boys and the other for girls

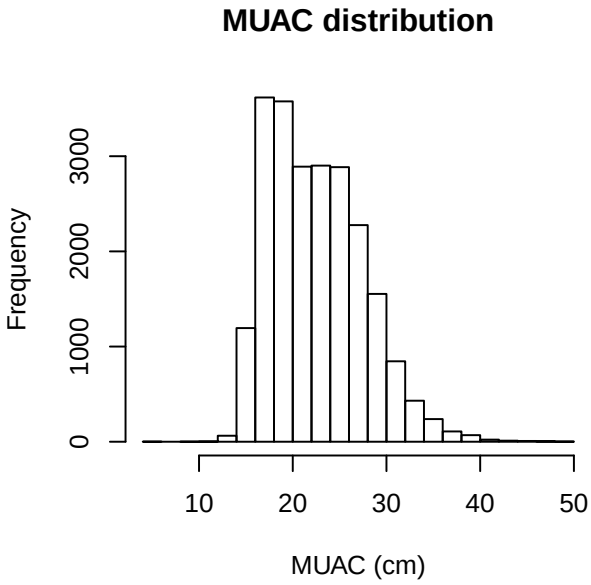
```
boysd = d2[d2$sex == "Male", ]
des(boysd)

##
##  No. of observations =  11218
##   Variable      Class           Description
## 1 sex           factor
## 2 muac          numeric
## 3 agem          numeric

summ(boysd)

##
##
## No. of observations = 11218
##
##   Var. name obs.  mean   median  s.d.   min.   max.
## 1 sex       11218 2      2       0      2      2
## 2 muac      11218 22.73  21.8    5.18   10.5   48.3
## 3 agem      11218 140.58 140.25  57.07  24.11  299.93

girlsd = d2[d2$sex == "Female", ]
des(girlsd)

##
##  No. of observations =  11481
##   Variable      Class           Description
## 1 sex           factor
## 2 muac          numeric
## 3 agem          numeric

summ(girlsd)
```

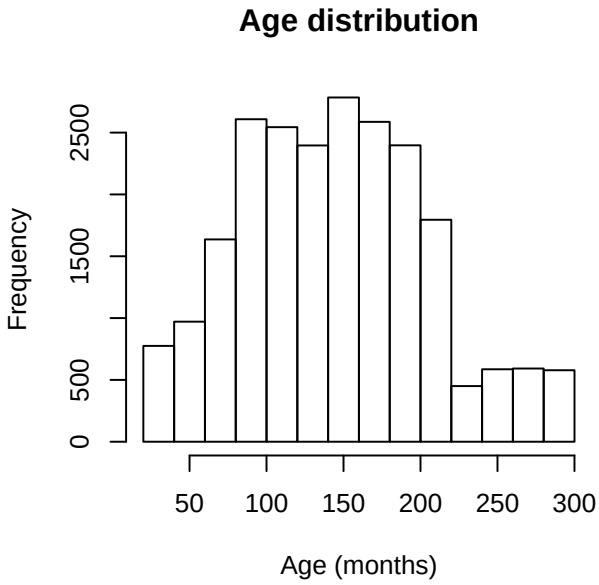
```
##
##
## No. of observations = 11481
##
##   Var. name obs.  mean   median  s.d.   min.   max.
## 1 sex       11481 1      1       0      1      1
## 2 muac      11481 22.51  22.3    4.77   5.9    49.9
## 3 agem      11481 148.76 145.28  64.22  24.05  299.99
```

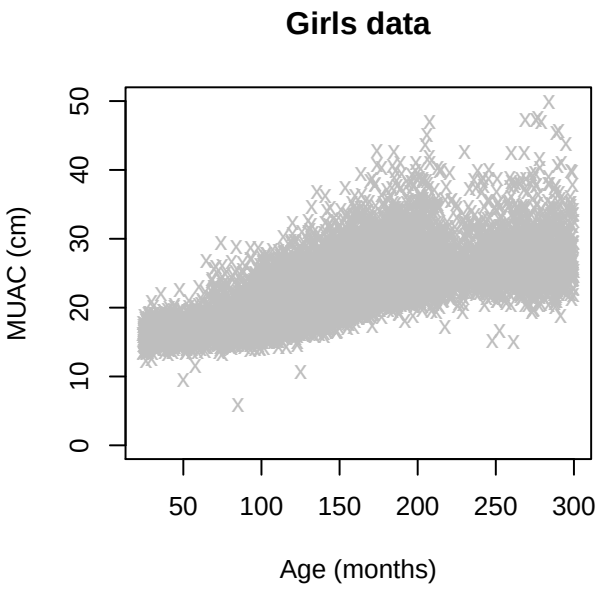
4 Fitting models, calculating Z scores

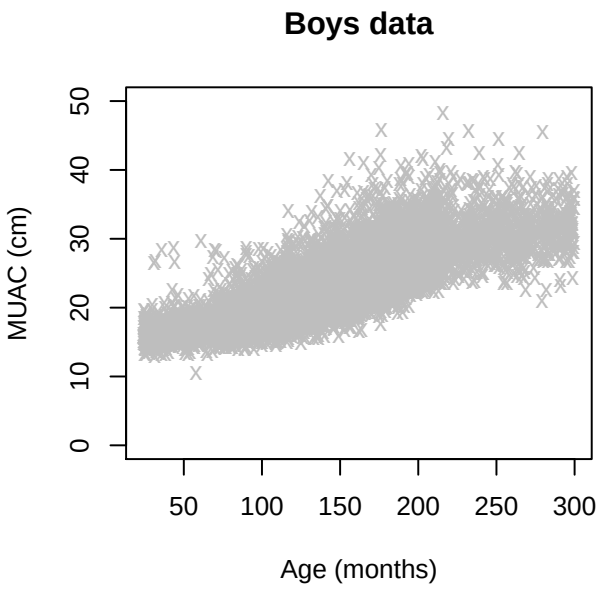
4.1 Data Explanatory Analysis

```
X11(height = 8, width = 8)
par(mfrow = c(2, 2))
hist(d2$muac, main = "MUAC distribution", xlab = "MUAC (cm)")
hist(d2$agem, main = "Age distribution", xlab = "Age (months)")
plot(muac ~ agem, data = girlsd, ylab = "MUAC (cm)", xlab = "Age (months)",
    pch = "x", col = "grey", main = "Girls data", ylim = c(0, 50))
plot(muac ~ agem, data = boysd, ylab = "MUAC (cm)", xlab = "Age (months)", pch = "x",
    col = "grey", main = "Boys data", ylim = c(0, 50))
```



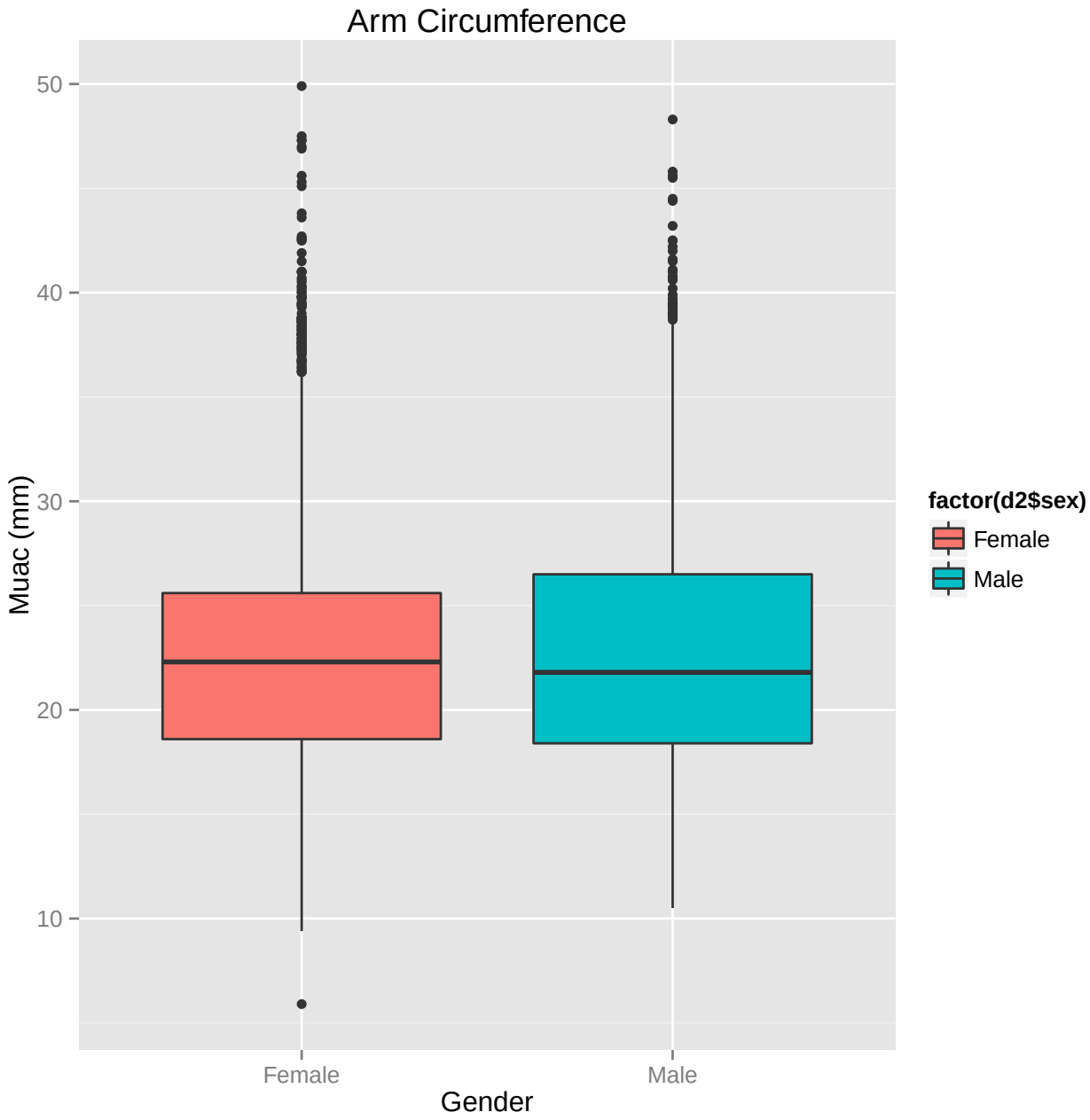







```
par(mfrow = c(1, 1))
```

```
require(ggplot2)

## Loading required package:  ggplot2

qplot(factor(d2$sex), d2$muac, data = d2, geom = c("boxplot"), main = "Arm Circumference"
    fill = factor(d2$sex), xlab = "Gender", ylab = "Muac (mm)")
```



4.2 Model Selection

Boys model selection

Note that BCT has been selected as the best other than BCCG and BCPE.

4.3 Generating Z-scores

Generating Z-scores for both of the datasets seperately starting with the female dataset.

```
# Girls model selection and fitting the model for boys
hf1 <- gamlss(muac ~ cs(agem), sigma.fo = ~cs(agem), nu.fo = ~cs(agem), tau.fo = ~cs(agem
    data = girlsd, family = BCT)

## GAMLSS-RS iteration 1: Global Deviance = 55080
## GAMLSS-RS iteration 2: Global Deviance = 55026
## GAMLSS-RS iteration 3: Global Deviance = 55025
```

```
## GAMLSS-RS iteration 4: Global Deviance = 55025
## GAMLSS-RS iteration 5: Global Deviance = 55025
## GAMLSS-RS iteration 6: Global Deviance = 55025
## GAMLSS-RS iteration 7: Global Deviance = 55025
## GAMLSS-RS iteration 8: Global Deviance = 55025
## GAMLSS-RS iteration 9: Global Deviance = 55025
## GAMLSS-RS iteration 10: Global Deviance = 55025
## GAMLSS-RS iteration 11: Global Deviance = 55025
## GAMLSS-RS iteration 12: Global Deviance = 55025
## GAMLSS-RS iteration 13: Global Deviance = 55025

hf0 <- gamlss(muac ~ cs(agem), sigma.fo = ~cs(agem), nu.fo = ~cs(agem), tau.fo = ~cs(agem
    data = girlsd, family = BCPE)

## GAMLSS-RS iteration 1: Global Deviance = 55585
## GAMLSS-RS iteration 2: Global Deviance = 55124
## GAMLSS-RS iteration 3: Global Deviance = 55111
## GAMLSS-RS iteration 4: Global Deviance = 55111
## GAMLSS-RS iteration 5: Global Deviance = 55111
## GAMLSS-RS iteration 6: Global Deviance = 55112
## GAMLSS-RS iteration 7: Global Deviance = 55112
## GAMLSS-RS iteration 8: Global Deviance = 55112
## GAMLSS-RS iteration 9: Global Deviance = 55112

AIC(hf1, hf0, k = 3)

##     df   AIC
## hf1 20 55085
## hf0 20 55172

GAIC(hf1, hf0, k = 3)

##     df   AIC
## hf1 20 55085
## hf0 20 55172
```