

Statistical re-analysis plan: Predicting body fat proportion using anthropometric measures

Mariana Nold

```
library(mfp2)
library(rstanarm)
library(projpred)
library(ggplot2)
library(loo)
library(bayesplot)
library(bayesrules)
library(FBMS)
```

```
women <- read.delim("C:/Users/zo95yup/Documents/GitHub/Bayes_for_STRATOS/Task Bodyfat/Datafiles/BodyFat")
names(women)
```

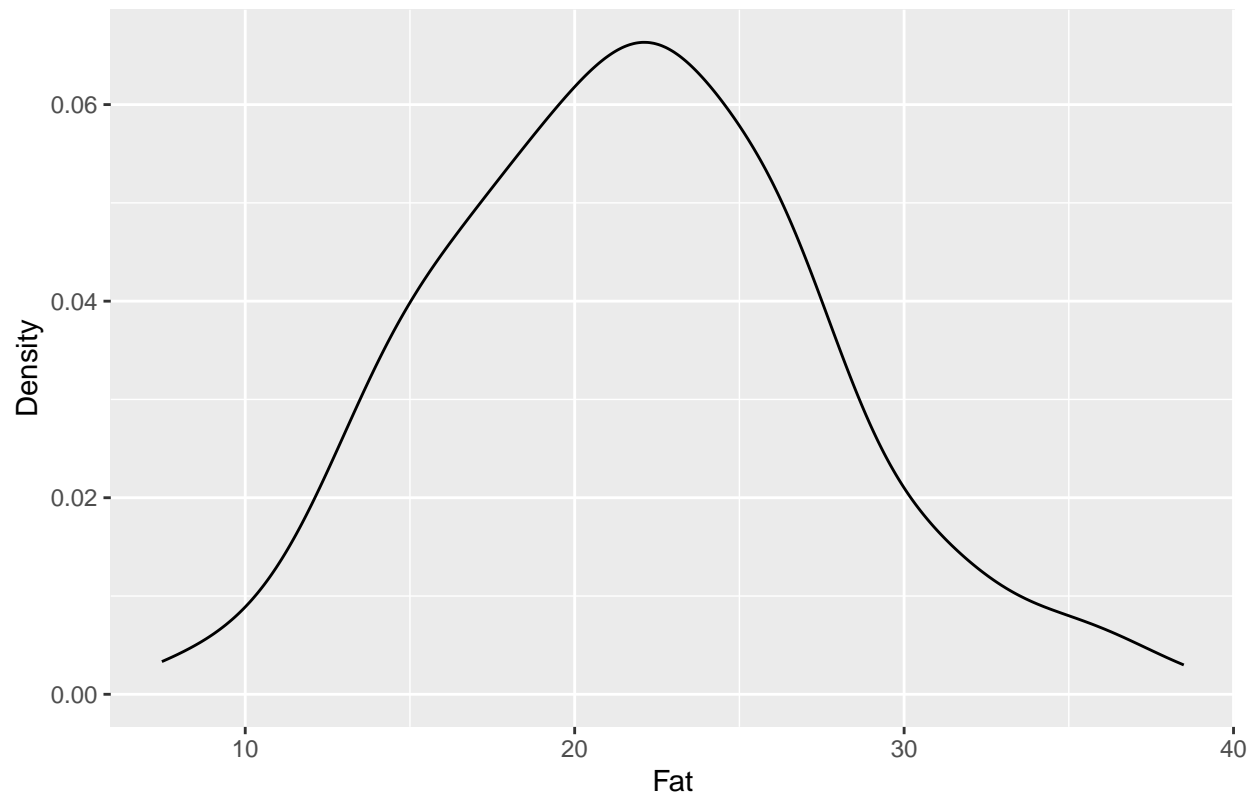
Load R packages

```
## [1] "Obs"      "Fat"      "Weight"   "Height"   "BMI"      "Age"      "Neck"
## [8] "Chest"    "Calf"     "Biceps"   "Hips"     "Waist"    "Forearm"  "PThigh"
## [15] "MThigh"   "DThigh"   "Wrist"    "Knee"     "Elbow"    "Ankle"
```

```
PPath2 <- "C:/Users/zo95yup/Nextcloud/Statistik_SoSe24/Forschung/STRATOS/P6/Code_A/Results"
```

```
fat_density <- density(women$Fat, na.rm = TRUE)
ggplot(women, aes(x = Fat)) + geom_density() +
  labs(title = "Kernel-density plot for outcome fat", x = "Fat", y = "Density")
```

Kernel-density plot for outcome fat



Prepare data

```
summary(women$Fat)
```

```
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##      7.47  17.61   21.64   21.76   25.73   38.49
```

```
summary(women$Age)
```

```
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##      1.00  19.00   20.00   20.39   21.00   25.00
```

```
women<- women[women$Age != 1, ]
women[,3:20] <- scale(women[,3:20])
head(women)
```

```
##      Obs   Fat      Weight      Height      BMI      Age      Neck      Chest
## 1      1  23.07 -1.29458090 -0.9359529 -0.9574982  0.9702795 -1.7434463 -1.3798033
## 2      2  29.50  0.08151288 -1.5820137  1.0776639  0.9702795 -0.1839318  0.3526427
## 3      3  26.99  0.35673163 -0.8498114  0.9208569  0.3269420 -0.4438509 -0.2042149
## 4      4  20.25 -0.85423089  0.2269566 -1.0444077 -0.3163955 -0.9636891 -1.0859062
## 5      5  19.95 -0.08361838  1.0883711 -0.6631153 -0.3163955  0.6608052 -0.1268736
## 6      6  26.02  0.96221290 -0.9359529  1.6864004  2.2569544  1.6355018  1.8530647
##           Calf      Biceps      Hips      Waist      Forearm      PThigh      MThigh
## 1 -0.9068981 -0.66317114 -1.0960084 -1.0337620 -1.5610578 -1.1012185 -0.5462338
## 2 -0.6839428  0.06791512 -0.4770169 -0.2266337 -0.1809922  0.2515874 -0.3492182
## 3 -0.8790287  0.47407415 -0.4770169  0.2390172 -1.0592158 -0.4248155 -0.1029487
## 4 -0.4052487 -1.15056197 -1.0471407 -0.9251101 -1.2474066 -1.1012185 -0.8417572
## 5 -0.1265545 -0.98809836 -0.6073309 -0.6922846 -1.5610578 -0.4248155 -0.3492182
## 6  0.5144420  1.28639220  1.3148007  1.1392756  0.9481524  1.0407242  0.7589945
```

```
##           DThigh      Wrist      Knee      Elbow      Ankle
## 1 -0.68363710 -0.6889801 -1.1572958 -1.1260785 -1.2200455
## 2 -0.12699377  0.1614979 -0.4658936 -0.4966577  0.5177452
## 3 -0.40531544 -1.3268386 -0.2930431  0.3215893 -0.9178210
## 4 -0.68363710 -1.2205288 -0.2930431 -0.7484260 -0.9933771
## 5  0.01216707 -0.9015996 -1.1572958  0.3845314 -0.2378160
## 6  0.98629291  1.8624538  1.4354624  1.0139522  0.8955257
```

```
modell1_f <- formula(Fat ~ fp(Waist, df = 2) + fp(Height, df = 2) + fp(Weight, df = 2))
modell1 <- mfp2(modell1_f, data = women, verbose=TRUE)
```

Derive reference model

```
##
## i Initial degrees of freedom:
##   Waist Height Weight
## df      2      2      2
##
## i Visiting order: Weight, Height, Waist
##
## -----
## i Running MFP Cycle 1
## -----
##
## Variable: Weight (keep = FALSE)
##           Powers    DF    Deviance    Versus    Deviance diff.
## FP1           0.5      6     992.1      .          .
## null          NA      4     1013.6     FP1        21.6
## linear        1       5     989.2     FP1        -2.9
##
##           P-value
## FP1          .
## null         0.0000
## linear       1.0000
## Selected: linear
##
## Variable: Height (keep = FALSE)
##           Powers    DF    Deviance    Versus    Deviance diff.
## FP1           2      6     988.5      .          .
## null          NA      4     1010.0     FP1        21.5
## linear        1      5     989.2     FP1         0.6
##
##           P-value
## FP1          .
## null         0.0000
## linear       0.4315
## Selected: linear
##
## Variable: Waist (keep = FALSE)
##           Powers    DF    Deviance    Versus    Deviance diff.
## FP1           2      6     988.8      .          .
## null          NA      4     999.5     FP1        10.7
## linear        1      5     989.2     FP1         0.4
##
##           P-value
## FP1          .
## null         0.0046
```

```
## linear          0.5445
## Selected: linear
## -----
## i Running MFP Cycle 2
## -----
##
## Variable: Weight (keep = FALSE)
##      Powers    DF   Deviance    Versus    Deviance diff.
## FP1          0.5     6     992.1      .         .
## null         NA      4     1013.6     FP1       21.6
## linear        1      5     989.2     FP1       -2.9
##      P-value
## FP1          .
## null         0.0000
## linear        1.0000
## Selected: linear
##
## Variable: Height (keep = FALSE)
##      Powers    DF   Deviance    Versus    Deviance diff.
## FP1          2      6     988.5      .         .
## null         NA      4     1010.0     FP1       21.5
## linear        1      5     989.2     FP1        0.6
##      P-value
## FP1          .
## null         0.0000
## linear        0.4315
## Selected: linear
##
## Variable: Waist (keep = FALSE)
##      Powers    DF   Deviance    Versus    Deviance diff.
## FP1          2      6     988.8      .         .
## null         NA      4     999.5     FP1       10.7
## linear        1      5     989.2     FP1        0.4
##      P-value
## FP1          .
## null         0.0046
## linear        0.5445
## Selected: linear
##
## i Fractional polynomial fitting algorithm converged after 2 cycles.
# Summarize the model
summary(model1)
```

```
##
## Call:
## glm(formula = y ~ ., family = family, data = data, weights = weights,
##      offset = offset, x = TRUE, y = TRUE)
##
## Coefficients:
##      Estimate Std. Error t value Pr(>|t|)
## (Intercept)  21.7281    0.2698  80.527 < 2e-16 ***
## Weight.1      3.1648    0.6253   5.061 1.03e-06 ***
## Height.1     -1.5464    0.3328  -4.646 6.54e-06 ***
```

```

## Waist.1      1.8501      0.5724      3.232 0.00146 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 13.3232)
##
##      Null deviance: 6158.0  on 182  degrees of freedom
## Residual deviance: 2384.9  on 179  degrees of freedom
## AIC: 999.17
##
## Number of Fisher Scoring iterations: 2
print(model1)

## Shifting, Scaling and Centering of covariates
##      shift scale center
## Weight 2.230325      1  TRUE
## Height 2.271145      1  TRUE
## Waist  1.794325      1  TRUE
##
## Final Multivariable Fractional Polynomial for y
##      df_initial select alpha selected df_final power1
## Weight      2    0.05  0.05      TRUE      1      1
## Height      2    0.05  0.05      TRUE      1      1
## Waist       2    0.05  0.05      TRUE      1      1
##
## MFP algorithm convergence: TRUE
##
## Call:  glm(formula = y ~ ., family = family, data = data, weights = weights,
##      offset = offset, x = TRUE, y = TRUE)
##
## Coefficients:
## (Intercept)      Weight.1      Height.1      Waist.1
##      21.728      3.165      -1.546      1.850
##
## Degrees of Freedom: 182 Total (i.e. Null);  179 Residual
## Null Deviance:      6158
## Residual Deviance: 2385  AIC: 999.2
# Estimate the model
model2 <- mfp2(Fat ~ fp(Waist, df = 2) + fp(Height, df = 2) + fp(Weight, df = 2)
              + fp(BMI, df = 1), data = women, verbose=TRUE)

##
## i Initial degrees of freedom:
##      Waist Height Weight BMI
## df      2      2      2    1
##
## i Visiting order: Waist, BMI, Height, Weight
##
## -----
## i Running MFP Cycle 1
## -----
##
## Variable: Waist (keep = FALSE)

```

```

##          Powers    DF    Deviance    Versus    Deviance diff.
## FP1          2      7      986.6      .          .
## null         NA      5      997.8      FP1        11.2
## linear        1      6      987.5      FP1         0.9
##          P-value
## FP1          .
## null         0.0036
## linear        0.3309
## Selected: linear
##
## Variable: BMI (keep = FALSE)
##          Powers    DF    Deviance    Versus    Deviance diff.
## null         NA      5      989.2      .          .
## linear        1      6      987.5      null        1.6
##          P-value
## null         .
## linear        0.2005
## Selected: null
##
## Variable: Height (keep = FALSE)
##          Powers    DF    Deviance    Versus    Deviance diff.
## FP1          2      6      988.5      .          .
## null         NA      4      1010.0     FP1        21.5
## linear        1      5      989.2      FP1         0.6
##          P-value
## FP1          .
## null         0.0000
## linear        0.4315
## Selected: linear
##
## Variable: Weight (keep = FALSE)
##          Powers    DF    Deviance    Versus    Deviance diff.
## FP1          0.5      6      992.1      .          .
## null         NA      4      1013.6     FP1        21.6
## linear        1      5      989.2      FP1        -2.9
##          P-value
## FP1          .
## null         0.0000
## linear        1.0000
## Selected: linear
##
## -----
## i Running MFP Cycle 2
## -----
##
## Variable: Waist (keep = FALSE)
##          Powers    DF    Deviance    Versus    Deviance diff.
## FP1          2      6      988.8      .          .
## null         NA      4      999.5      FP1        10.7
## linear        1      5      989.2      FP1         0.4
##          P-value
## FP1          .
## null         0.0046
## linear        0.5445

```

```
## Selected: linear
##
## Variable: BMI (keep = FALSE)
##           Powers    DF   Deviance    Versus      Deviance diff.
## null           NA      5     989.2      .            .
## linear          1      6     987.5     null         1.6
##           P-value
## null           .
## linear          0.2005
## Selected: null
##
## Variable: Height (keep = FALSE)
##           Powers    DF   Deviance    Versus      Deviance diff.
## FP1            2      6     988.5      .            .
## null           NA      4    1010.0     FP1          21.5
## linear          1      5     989.2     FP1           0.6
##           P-value
## FP1            .
## null           0.0000
## linear          0.4315
## Selected: linear
##
## Variable: Weight (keep = FALSE)
##           Powers    DF   Deviance    Versus      Deviance diff.
## FP1            0.5     6     992.1      .            .
## null           NA      4    1013.6     FP1          21.6
## linear          1      5     989.2     FP1          -2.9
##           P-value
## FP1            .
## null           0.0000
## linear          1.0000
## Selected: linear
##
## i Fractional polynomial fitting algorithm converged after 2 cycles.
# Summarize the model
summary(model2)
```

```
##
## Call:
## glm(formula = y ~ ., family = family, data = data, weights = weights,
##      offset = offset, x = TRUE, y = TRUE)
##
## Coefficients:
##           Estimate Std. Error t value Pr(>|t|)
## (Intercept)  21.7281    0.2698  80.527 < 2e-16 ***
## Waist.1       1.8501    0.5724   3.232  0.00146 **
## Height.1     -1.5464    0.3328  -4.646  6.54e-06 ***
## Weight.1      3.1648    0.6253   5.061  1.03e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 13.3232)
##
## Null deviance: 6158.0 on 182 degrees of freedom
```

```
## Residual deviance: 2384.9 on 179 degrees of freedom
## AIC: 999.17
##
## Number of Fisher Scoring iterations: 2
```

```
print(model2)
```

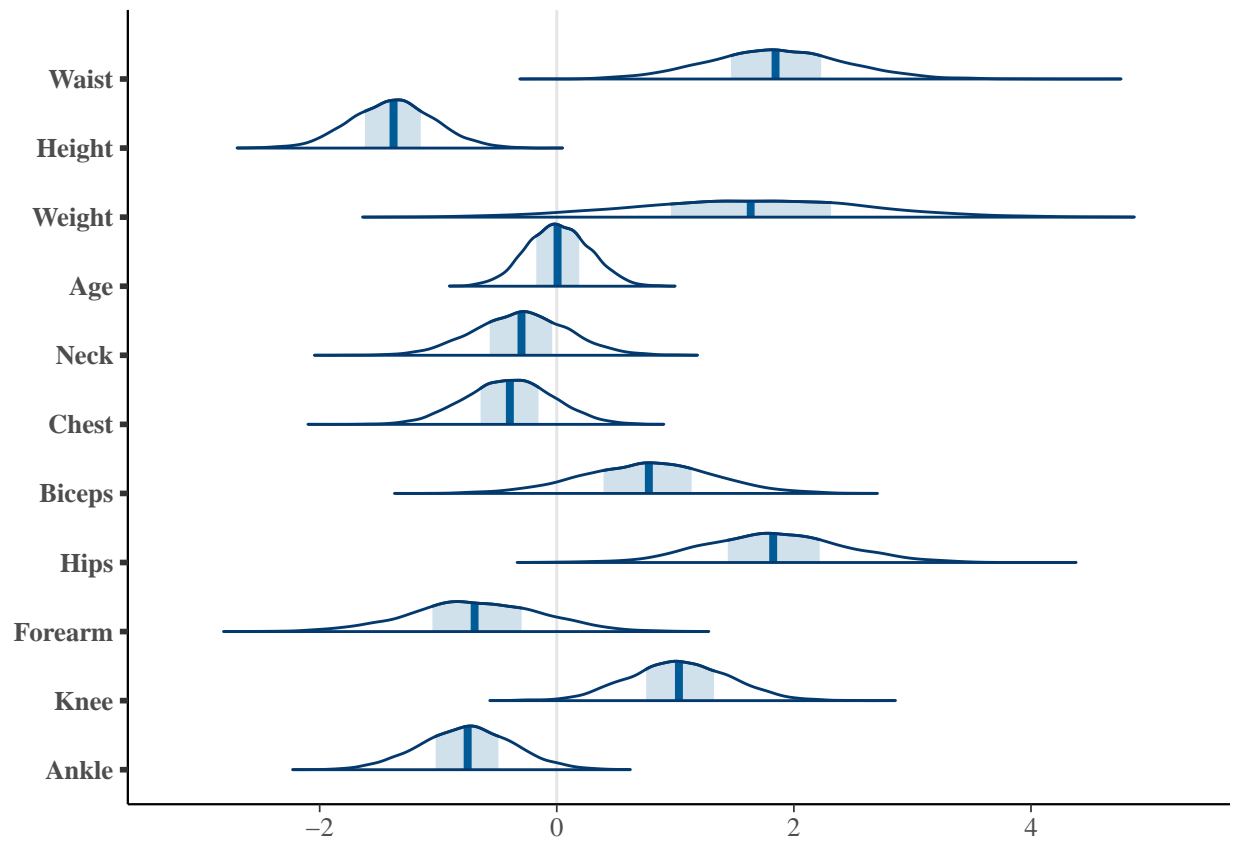
```
## Shifting, Scaling and Centering of covariates
##      shift scale center
## Waist  1.794325      1  TRUE
## BMI     2.143166      1  TRUE
## Height  2.271145      1  TRUE
## Weight  2.230325      1  TRUE
##
## Final Multivariable Fractional Polynomial for y
##      df_initial select alpha selected df_final power1
## Waist           2  0.05  0.05      TRUE          1      1
## BMI              1  0.05  0.05     FALSE          0     NA
## Height           2  0.05  0.05      TRUE          1      1
## Weight           2  0.05  0.05      TRUE          1      1
##
## MFP algorithm convergence: TRUE
##
## Call:  glm(formula = y ~ ., family = family, data = data, weights = weights,
##      offset = offset, x = TRUE, y = TRUE)
##
## Coefficients:
## (Intercept)      Waist.1      Height.1      Weight.1
##      21.728         1.850        -1.546         3.165
##
## Degrees of Freedom: 182 Total (i.e. Null);  179 Residual
## Null Deviance:      6158
## Residual Deviance: 2385  AIC: 999.2
```

```
#library(rstanarm)
```

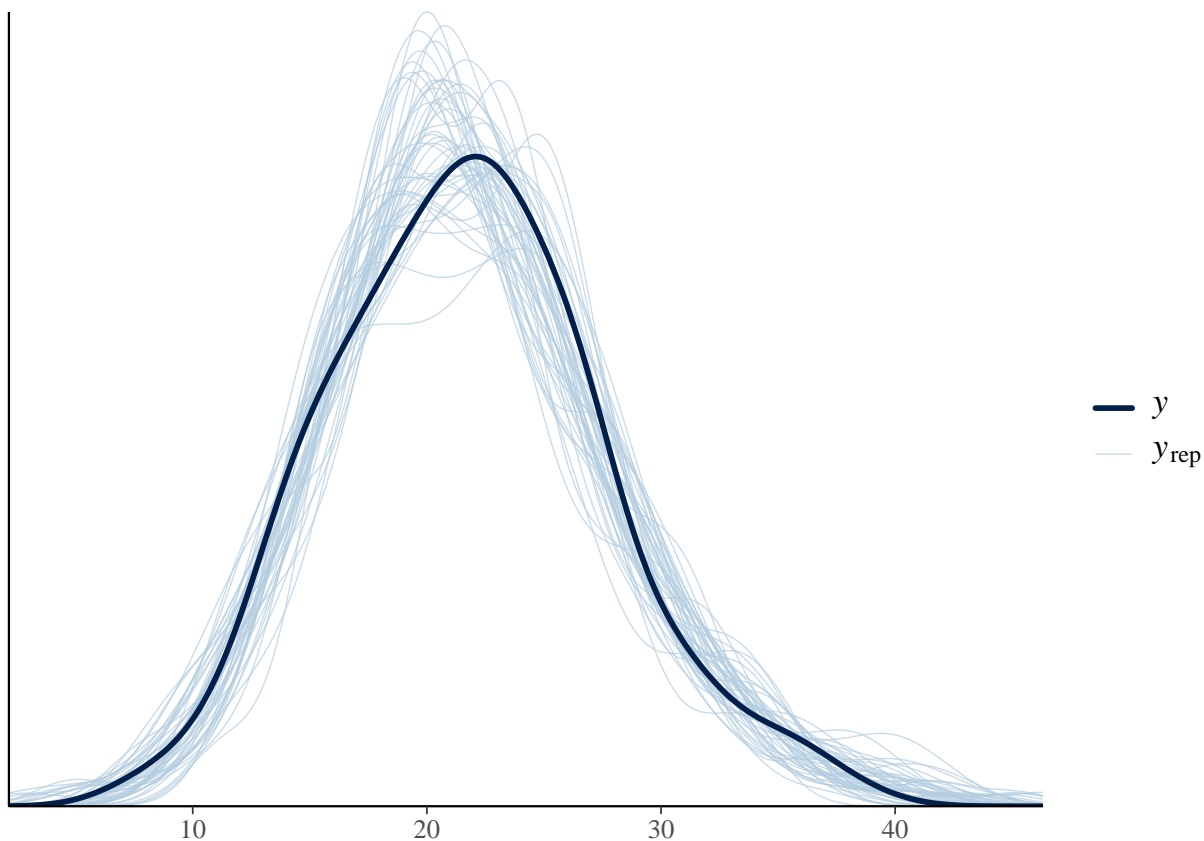
```
# Use the full model after step 3 in Georgs task, thus model 1 and all predictors but not BMI
# Only use predictors that are available in the man dataset
# Fat = siri, Age = age, Height = height, Weight = weight, Neck = neck, Chest = chest, Waist = abdomen,
rmf <- as.formula(Fat ~ Waist + Height + Weight + Age + Neck + Chest +
                  Biceps + Hips + Forearm +
                  Knee + Ankle)

# Fit a Bayesian Gaussian regression model using stan_glm
ref_fitf <- stan_glm(rmf, data = women, family = gaussian())
# Check reference model, MCMC check
summary(ref_fitf)
#shinytan::launch_shinytan(ref_fitf)

mcmc_areas(as.matrix(ref_fitf)[,2:12])
```

```
pp_check(ref_fitf)
```

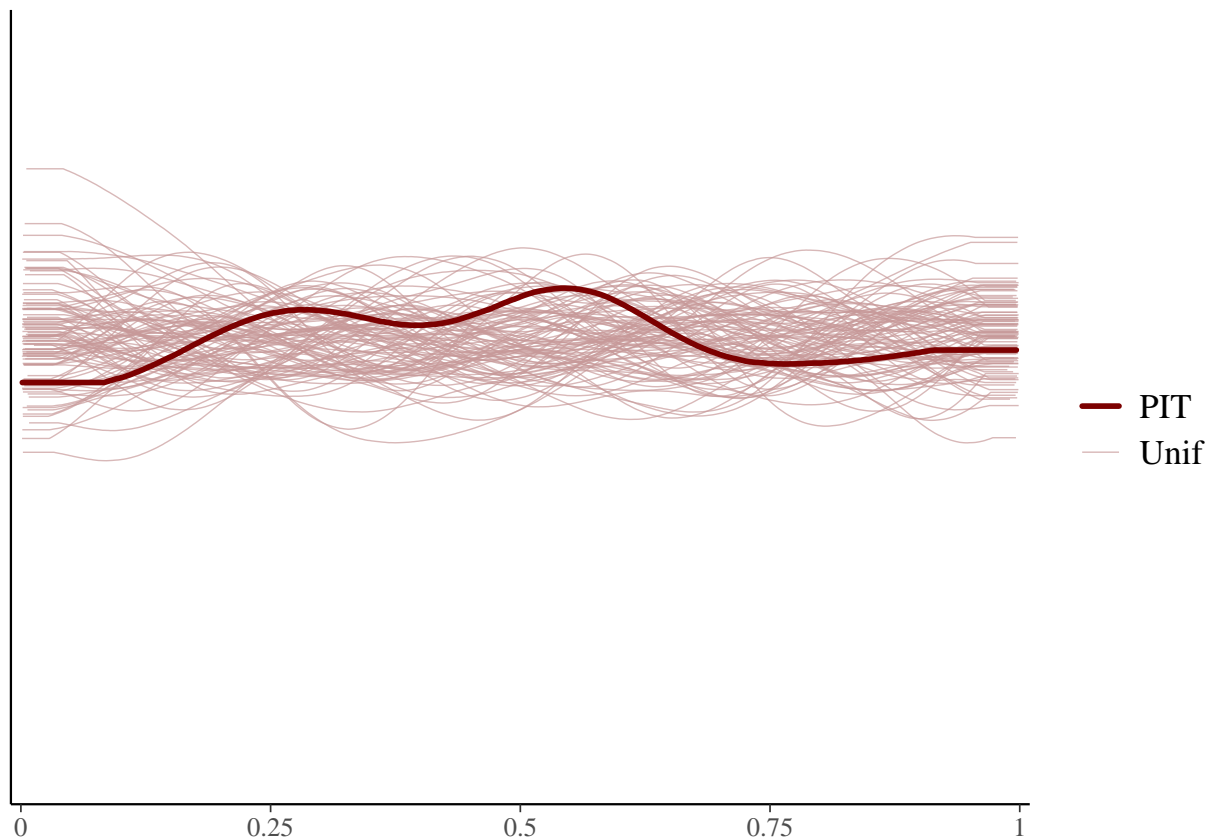


```

y_rep_rmf <- posterior_predict(ref_fitf)
loo_rmf <- loo(ref_fitf , save_psis = TRUE, cores = 4)
psis1_rmf <- loo_rmf$psis_object
lw_rmf <- weights(psis1_rmf)
color_scheme_set("red")
#pdf(file.path(PPath2, "/LOOPITrmf.pdf"))
ppc_loo_pit_overlay(y= women$Fat,yrep=y_rep_rmf,lw=lw_rmf)

```

NOTE: The kernel density estimate assumes continuous observations and is not optimal for discrete observations



```
#dev.off()

# Summarize the posterior distributions simple model
summary_rmf_cv <- prediction_summary_cv(model = ref_fitf , data = women, k = 3)

# Print the summary simple model
print(summary_rmf_cv)

## $folds
##   fold      mae mae_scaled within_50 within_95
## 1    1 2.639294  0.7793865 0.4098361 0.8032787
## 2    2 2.227154  0.5941096 0.5737705 0.9672131
## 3    3 2.264260  0.6081508 0.6065574 0.9344262
##
## $cv
##      mae mae_scaled within_50 within_95
## 1 2.376903  0.660549 0.5300546 0.9016393

pred <- c("Waist" , "Height" , "Weight" , "Age" , "Neck" , "Chest" ,
          "Biceps" , "Hips" , "Forearm" ,
          "Knee" , "Ankle")

# Extract the posterior samples for the estimator
posterior_samples <- as.matrix(ref_fitf, pars = pred)
```

```

# Define the range
lower_bound <- -0.05
upper_bound <- 0.05

# Apply the condition to each element of the matrix
prob_between <- apply(posterior_samples, 2, function(par) mean(par >= lower_bound & par <= upper_bound))

# Print the result
print(prob_between)

```

Compute the probability for each estimator that it is close to zero

```

##   Waist Height Weight      Age   Neck   Chest Biceps   Hips Forearm   Knee
## 0.00000 0.00025 0.00900 0.14825 0.06875 0.06275 0.02500 0.00100 0.03475 0.00350
##   Ankle
## 0.01650

print(prob_between[prob_between > 0.25])

```

```
## named numeric(0)
```

```

set.seed(2341)
vs_cvf <- cv_varsel(ref_fitf)

```

Use projpred to downsize model even more

```

## Warning in warn_pareto(n07 = sum(pareto_k > 0.7), n05 = sum(0.7 >= pareto_k & :
## In the calculation of the reference model's PSIS-L00 CV weights, 1 (out of 183)
## Pareto k-values are in the interval (0.5, 0.7]. Moment matching (see the loo
## package), mixture importance sampling (see the loo package), and `reloo`-ing
## (see the brms package) are not supported by projpred. If these techniques (run
## outside of projpred, i.e., for the reference model only; note that `reloo`-ing
## may be computationally costly) result in a markedly different reference model
## ELPD estimate than ordinary PSIS-L00 CV does, we recommend to use K-fold CV
## within projpred.

```

```

save(vs_cvf, file = file.path(PPath2, "vs_cvf"))
#load(file = file.path(PPath2, "vs_cv"))
nset <- suggest_size(vs_cvf)
nset

rank_vs_cvf <- ranking(vs_cvf)
rank_vs_cvf
solterms_final_vs_cvf <- head(rank_vs_cvf$fulldata, suggest_size(vs_cvf))

solterms_final_vs_cvf

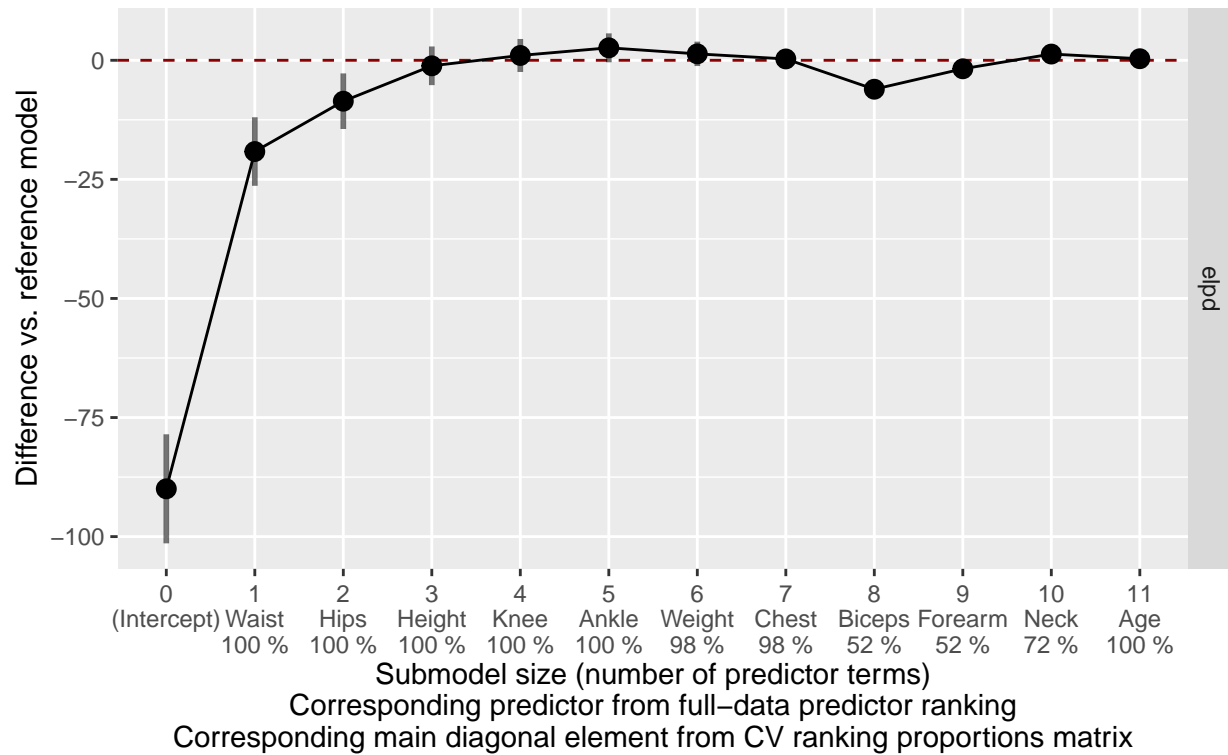
# Print the summary of the variable selection
summary(vs_cvf)

# Plot the variable selection
#pdf(file.path(PPath2, "cv_varsel_plotf.pdf"))
plot(vs_cvf, deltas = TRUE)

```

Predictive performance

Vertical bars indicate 68.3% normal-approximation intervals



```
#dev.off()
#
```

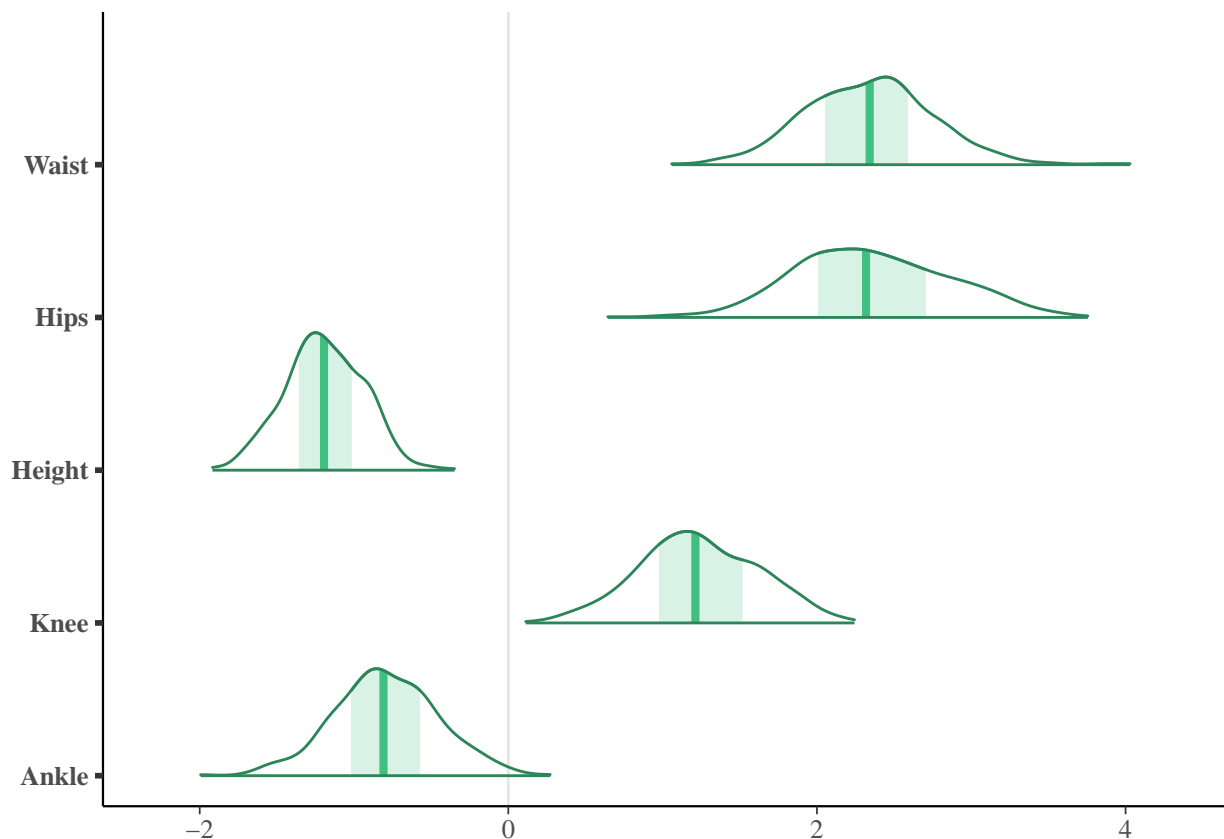
```
solterms_final_fivef <- head(rank_vs_cvf$fulldata, 5)
solterms_final_fivef
```

```
## [1] "Waist" "Hips" "Height" "Knee" "Ankle"
```

```
# https://cran.r-project.org/web/packages/projpred/vignettes/projpred.html#post-selection-inference
proj_ref_fitf <- project(vs_cvf, predictor_terms = solterms_final_fivef, ns = 4000)
```

Projected posterior for the selected model

```
color_scheme_set(scheme = "green")
mcmc_areas(as.matrix(proj_ref_fitf), pars = solterms_final_fivef)#
```



Show results

Derive second reference model based on FBMS: Using BMA, marginal likelihoods inherently account for model complexity, and PIPs derived from them reflect this balance.

#Marginal likelihoods inherently account for model complexity by integrating over all possible parameters

```
transforms <- c("p2","p3")# transforms planed in my re-sap
```

```
probs <- gen.probs.gmjmcmc(transforms)
probs$gen <- c(0,1,0,1) # Only modifications!
params <- gen.params.gmjmcmc(women[,c("Waist","Height","Weight","Age","Neck","Chest","Biceps","Hips","Forearm","Ankle","Knee","BMI")])
params$feat$pop.max <- 20 #set the population size to be 20
params$feat$D <- 1 # Set depth of features to 1
#@ Aliaksandr(1): please decide about this options
params$loglik$r <- exp(-2) # to correspond to AIC complexity that Georg uses in mfp
params$loglik$var <- "unknown"
modelFBMS <- as.formula(Fat ~ Waist + Height + Weight + Age + Neck + Chest +
                        Biceps + Hips + Forearm
                        + Knee + Ankle + BMI)
```

```
set.seed(1234)
```

```
result <- fbms(modelFBMS, data = women, method = "gmjmcmc.parallel",
               family = "gaussian",
               transforms = transforms, probs = probs, params = params, P = 50,
               runs = 40, cores = 10)
```

```
summary(result, labels = c("Waist", "Height", "Weight", "Age", "Neck", "Chest", "Biceps", "Hips", "Forearm", "Kn
```

The final model, which includes five predictors, is identical

```
##              Importance | Feature
##              | Forearm
##              | Biceps
##              | Chest
##              #| Weight
##              ##| BMI
##              ###| Ankle
##              #####| Knee
##  #####| Height
##  #####| Waist
##  #####| Hips
##
## Best   population: 16  thread: 2  log marginal posterior: -499.3607
## Report population: 16  thread: 2  log marginal posterior: -499.3607
##
##   feats.strings marg.probs
## 1      Hips 0.99540412
## 2      Waist 0.97552176
## 3      Height 0.93993647
## 4      Knee 0.59048472
## 5      Ankle 0.12427203
## 6      BMI 0.07414405
## 7      Weight 0.05028350
## 8      Chest 0.01719150
## 9      Biceps 0.01489256
## 10     Forearm 0.01041275
##
formula_male <- as.formula(Fat ~ Waist + Hips + Height + Knee + Ankle)
```

Fit model to male data set and compare the posterior predictive distributions

```
library(mfp)
```

See Task Bodyfat.qmd

```
## Warning: Paket 'mfp' wurde unter R Version 4.4.2 erstellt
## Lade nötiges Paket: survival
##
## Attache Paket: 'mfp'
## Das folgende Objekt ist maskiert 'package:mfp2':
##
##   fp
##
data(bodyfat)

bodyfat_corr <- bodyfat

bodyfat_corr[42, "height"] <- 69.5      # one-digit correction of height
```

```
bodyfat_corr[48,"density"] <- 1.0865 # one-digit correction of density
bodyfat_corr[76,"density"] <- 1.0566 # one-digit correction of density
bodyfat_corr[96,"density"] <- 1.0591 # one-digit correction of density

# recompute siri formula based on corrected density

bodyfat_corr[, "siri"] <- pmax(round(495 / bodyfat_corr$density - 450, 1), 0)
bodyfat_corr[, "brozek"] <- pmax(round(457 / bodyfat_corr$density - 414.2, 1), 0)
```

Aligning the two data sets

```
bodyfat_corr$height <- bodyfat_corr$height * 2.54/100
bodyfat_corr$weight <- bodyfat_corr$weight * 0.454592
```

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ----- tidyverse 2.0.0 --
## v dplyr      1.1.4      v readr      2.1.5
## v forcats    1.0.0      v stringr    1.5.1
## v lubridate  1.9.3      v tibble     3.2.1
## v purrr      1.0.2      v tidyr      1.3.1
```

```
## -- Conflicts ----- tidyverse_conflicts() --
```

```
## x dplyr::filter() masks stats::filter()
```

```
## x dplyr::lag() masks stats::lag()
```

```
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
men <- bodyfat_corr %>%
```

```
  rename(Fat = siri, Age = age, Height = height, Weight = weight,
         Neck = neck, Chest = chest, Waist = abdomen, Hips = hip,
         Knee = knee, Ankle = ankle, Forearm = forearm,
         Biceps = biceps, Thigh = thigh)
```

```
names(men)
```

```
## [1] "case" "brozek" "Fat" "density" "Age" "Weight" "Height"
## [8] "Neck" "Chest" "Waist" "Hips" "Thigh" "Knee" "Ankle"
## [15] "Biceps" "Forearm" "wrist"
```

```
# Add BMI
```

```
men$BMI <- men$Weight / (men$Height^2)
```

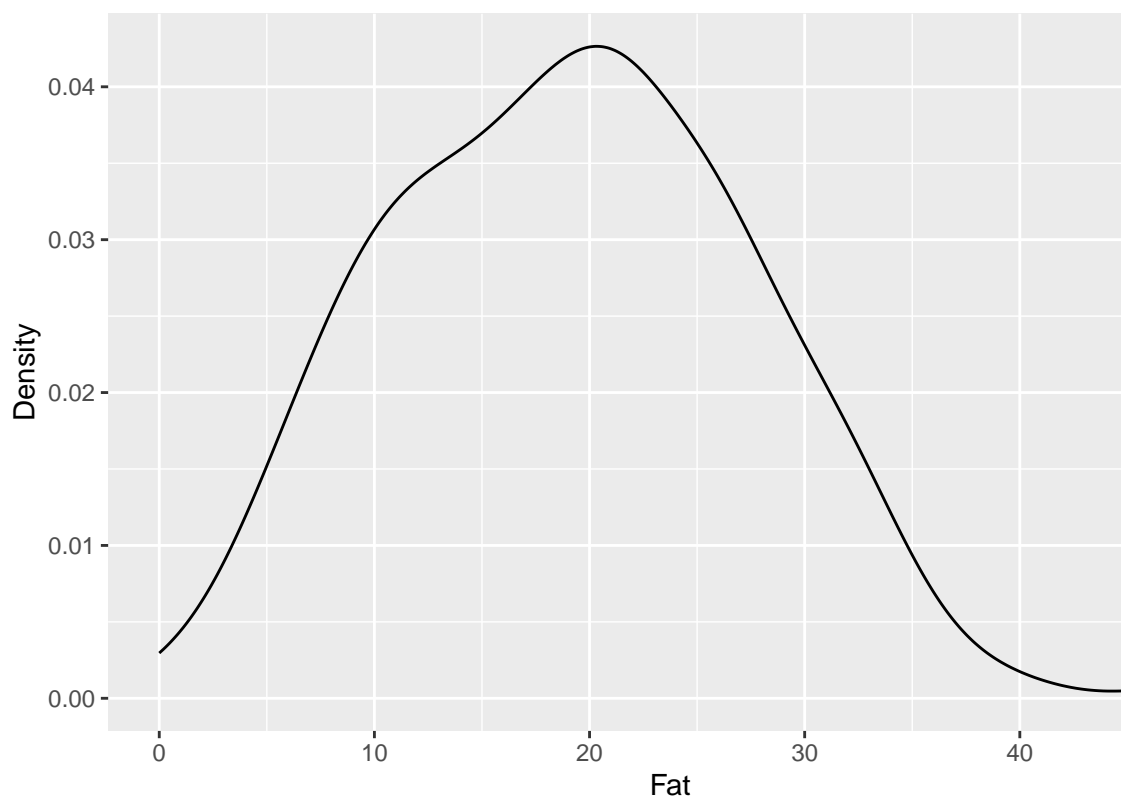
```
men[,4:18] <- scale(men[,4:18])
```

```
fat_density_male <- density(men$Fat, na.rm = TRUE)
```

```
ggplot(men, aes(x = Fat)) + geom_density() +
```

```
  labs(title = "Kernel-density plot for outcome fat for males", x = "Fat", y = "Density")
```


Kernel-density plot for outcome fat for males

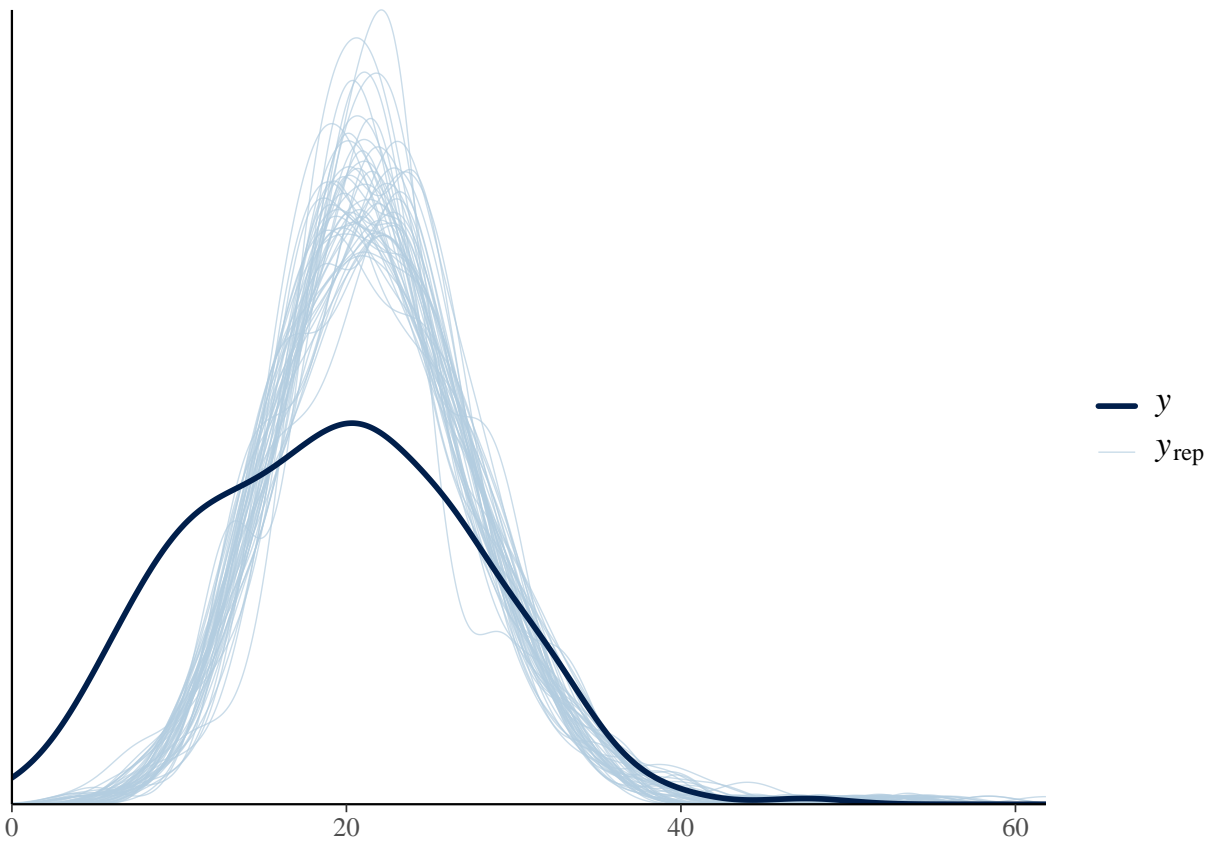


Prepare data for males

```
summary(men$Fat)
```

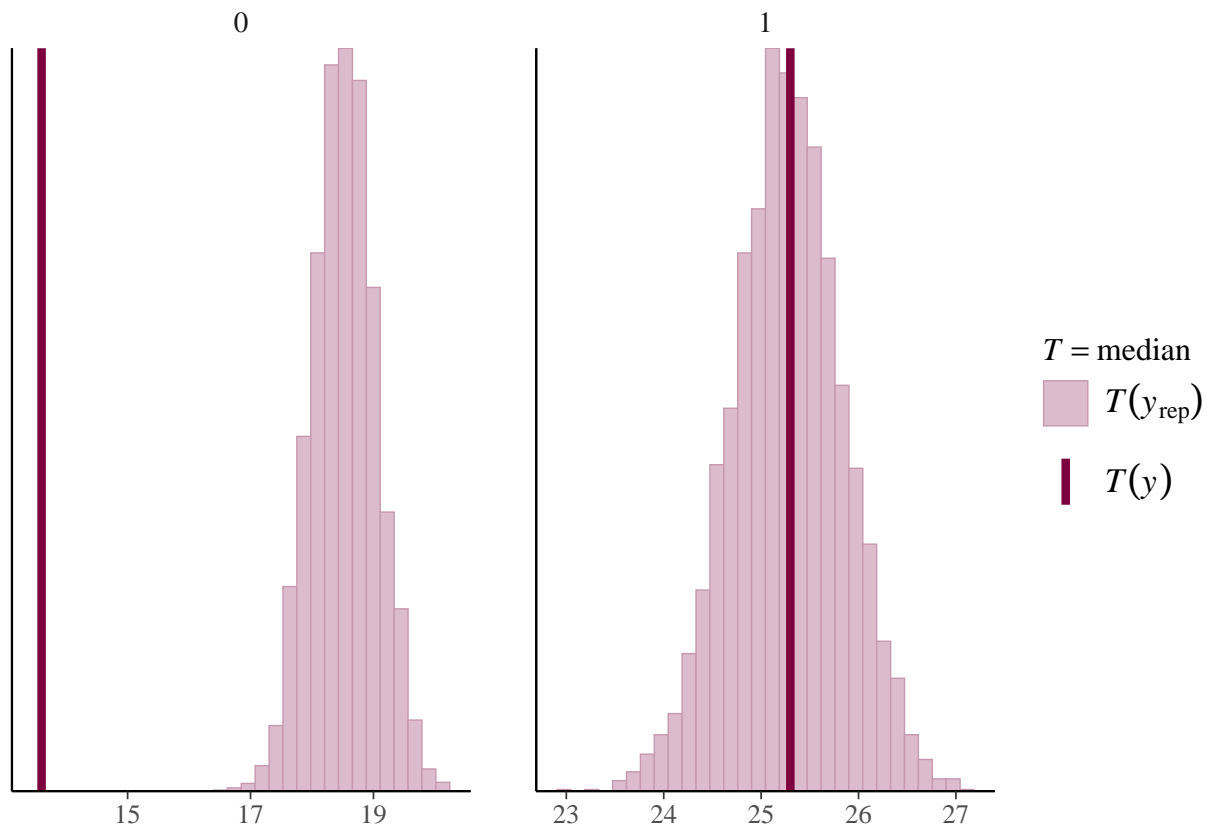
```
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##      0.00  12.40   19.20   19.16  25.30   47.50
```

```
# LOOPIT and bayesrules summary for male data based on female model
# Prediction for male data based on female model
fit_female <- stan_glm(formula_male, data = women, family = gaussian())
#proj_ref_fitf <- project(vs_cvf, predictor_terms = solterms_final_fivef, ns = 4000)
# https://www.rdocumentation.org/packages/projpred/versions/2.0.2/topics/proj-pred
#y_rep_male <- proj_linpred(vs_cvf, solution_terms= solterms_final_fivef, newdata = men)#
y_rep_male <- posterior_predict(fit_female, newdata = men)
color_scheme_set("blue")
bayesplot::pp_check(object = men$Fat, yrep = y_rep_male[1:50,], fun = ppc_dens_overlay)
```



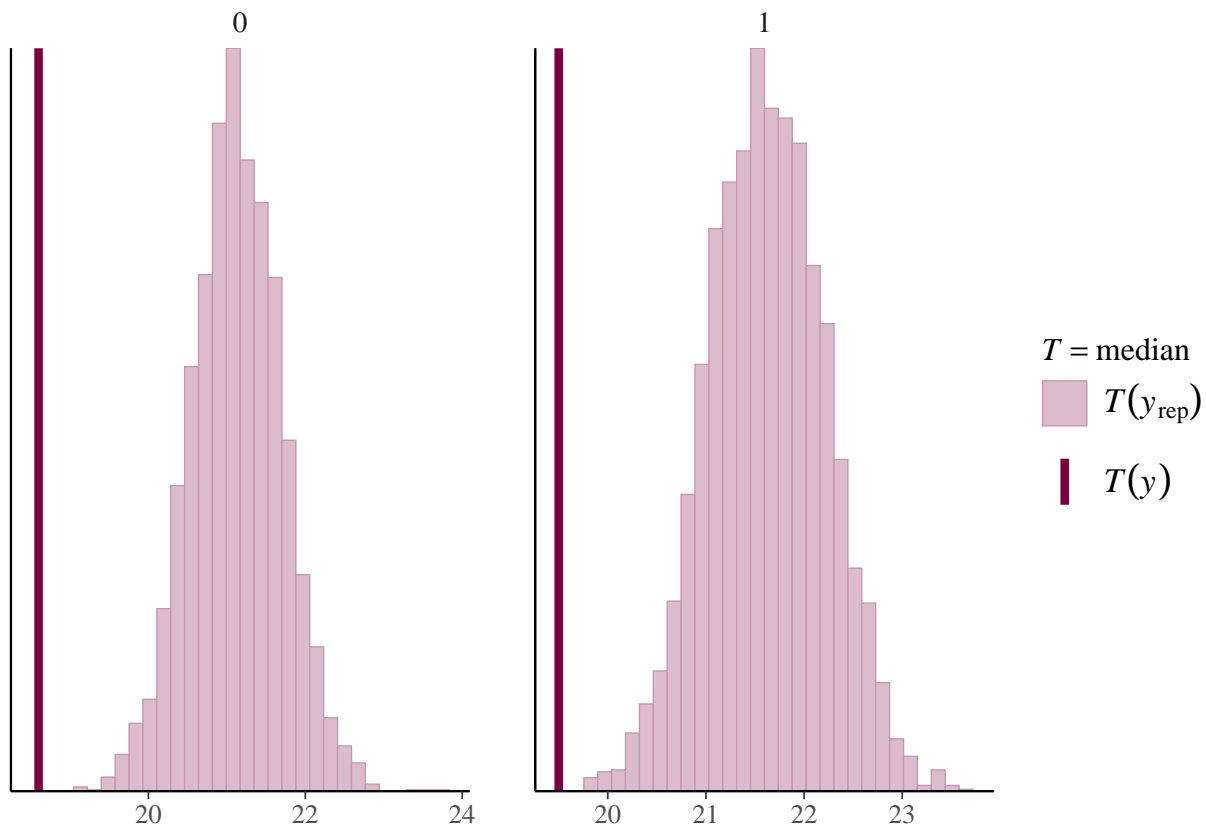
```
binary_waist_male <- ifelse(men$Waist <= 0, 0, 1)
binary_height_male <- ifelse(men$Height <= 0, 0, 1)
binary_hips_male <- ifelse(men$Hips <= 0, 0, 1)
binary_knee_male <- ifelse(men$Knee <= 0, 0, 1)
binary_ankle_male <- ifelse(men$Ankle <= 0, 0, 1)
color_scheme_set("pink")
bayesplot::pp_check(object = men$Fat, yrep = y_rep_male,
                    fun = "stat_grouped", group = binary_waist_male, stat = "median")

## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



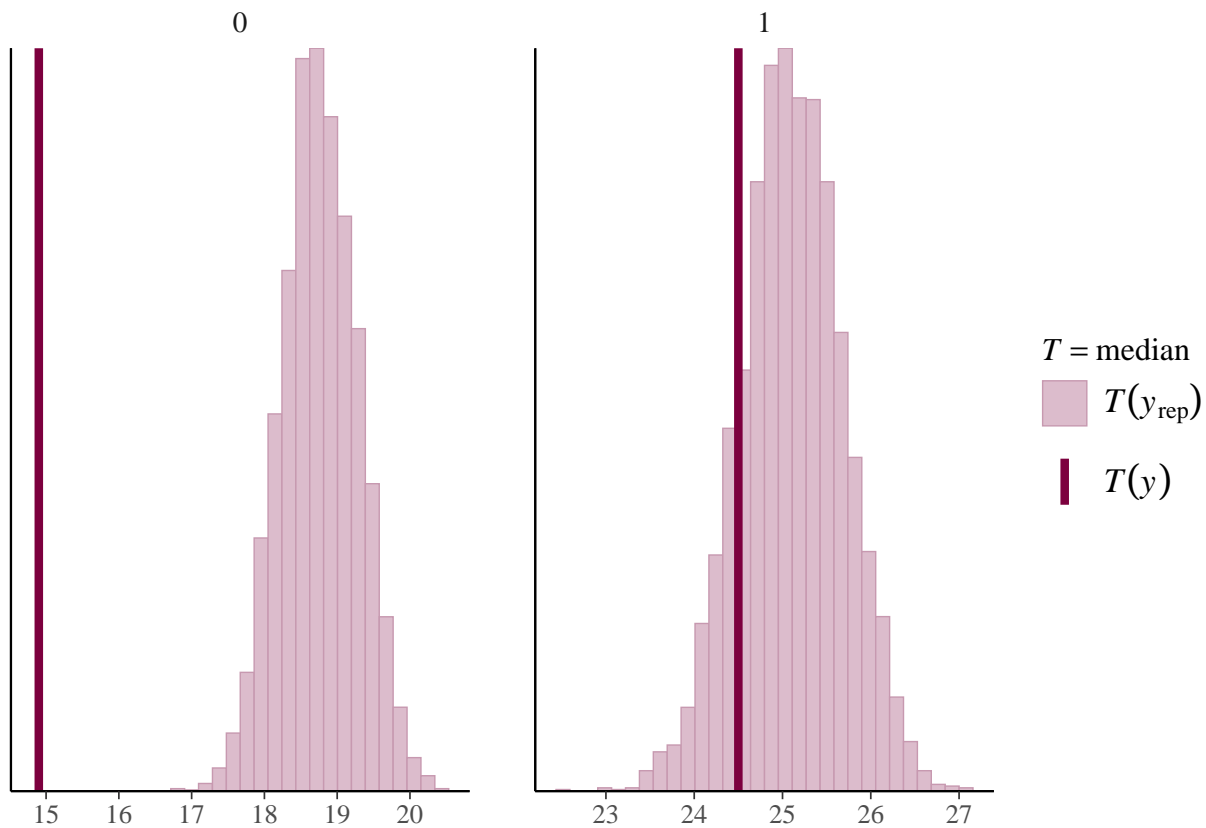
```
bayesplot::pp_check(object = men$Fat, yrep = y_rep_male,
  fun = "stat_grouped", group = binary_height_male, stat = "median")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



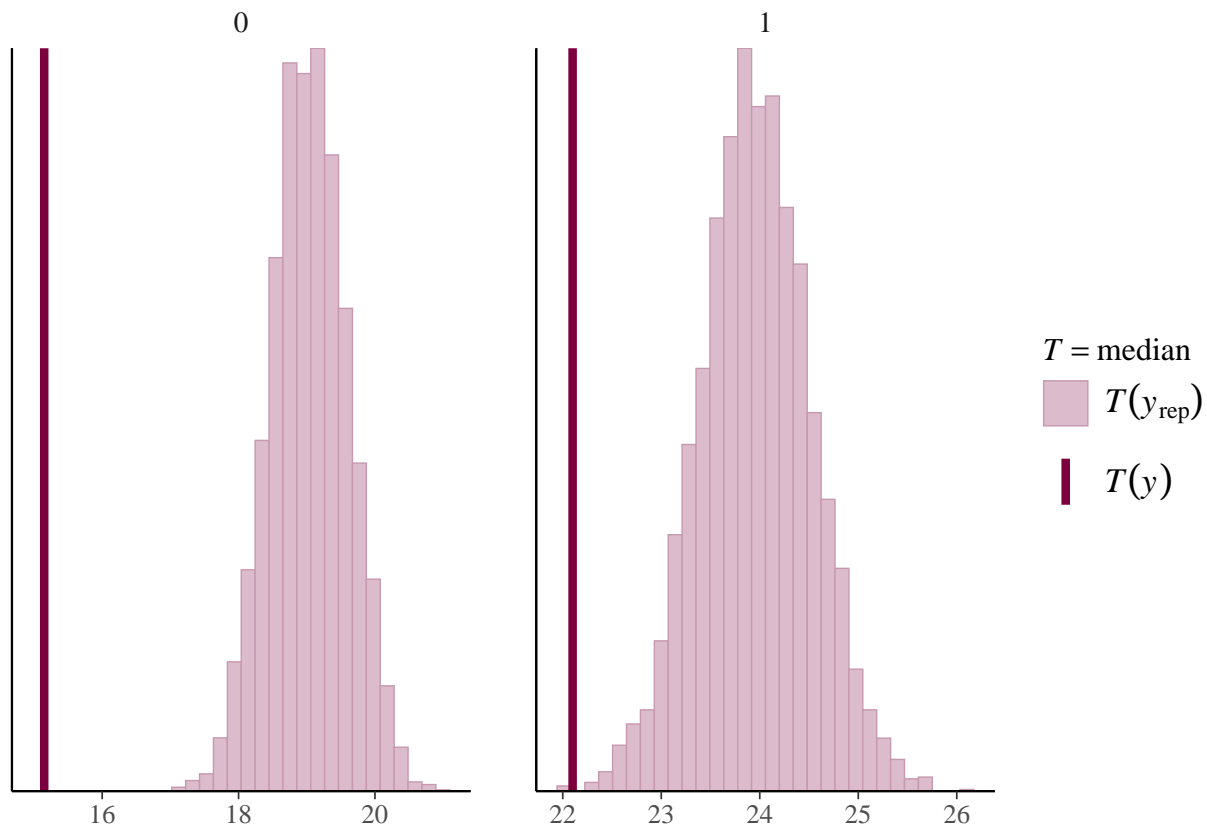
```
bayesplot::pp_check(object = men$Fat, yrep = y_rep_male,
  fun = "stat_grouped", group = binary_hips_male, stat = "median")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



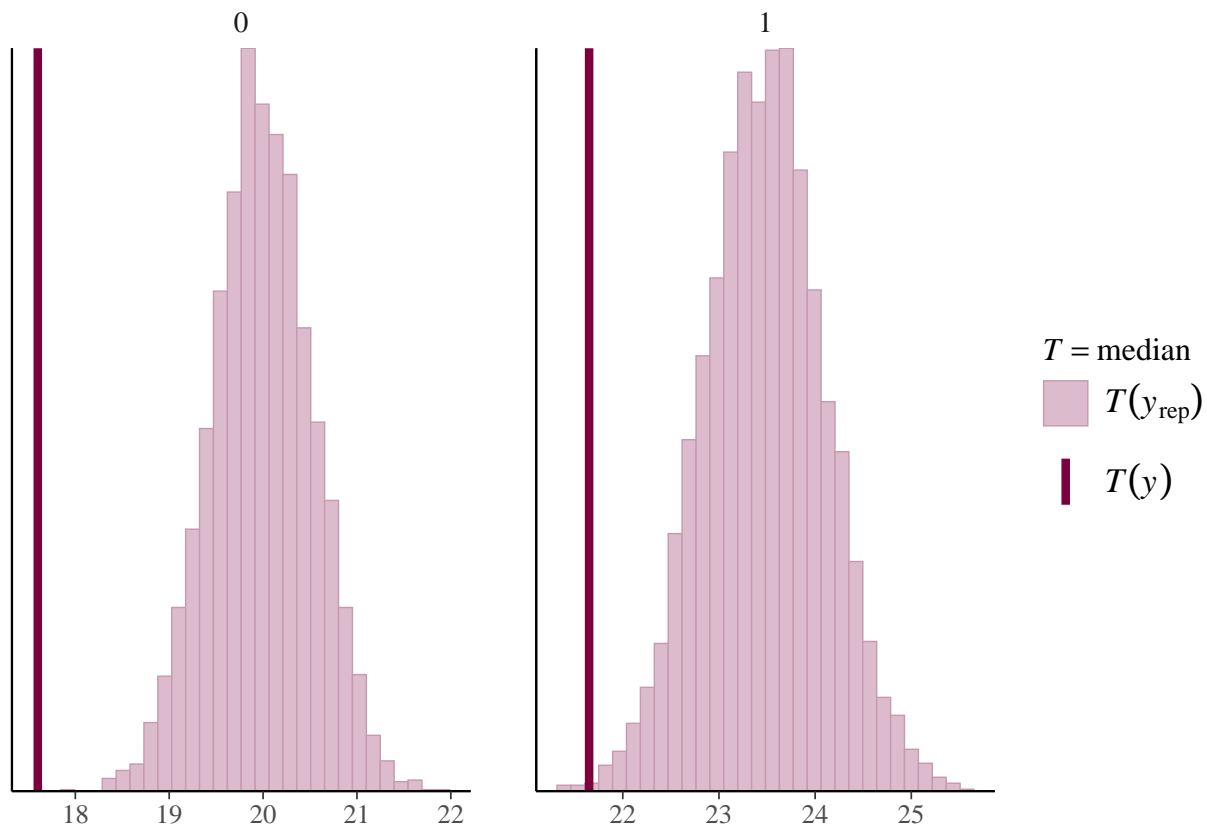
```
bayesplot::pp_check(object = men$Fat, yrep = y_rep_male, fun = "stat_grouped",
  group = binary_knee_male, stat = "median")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



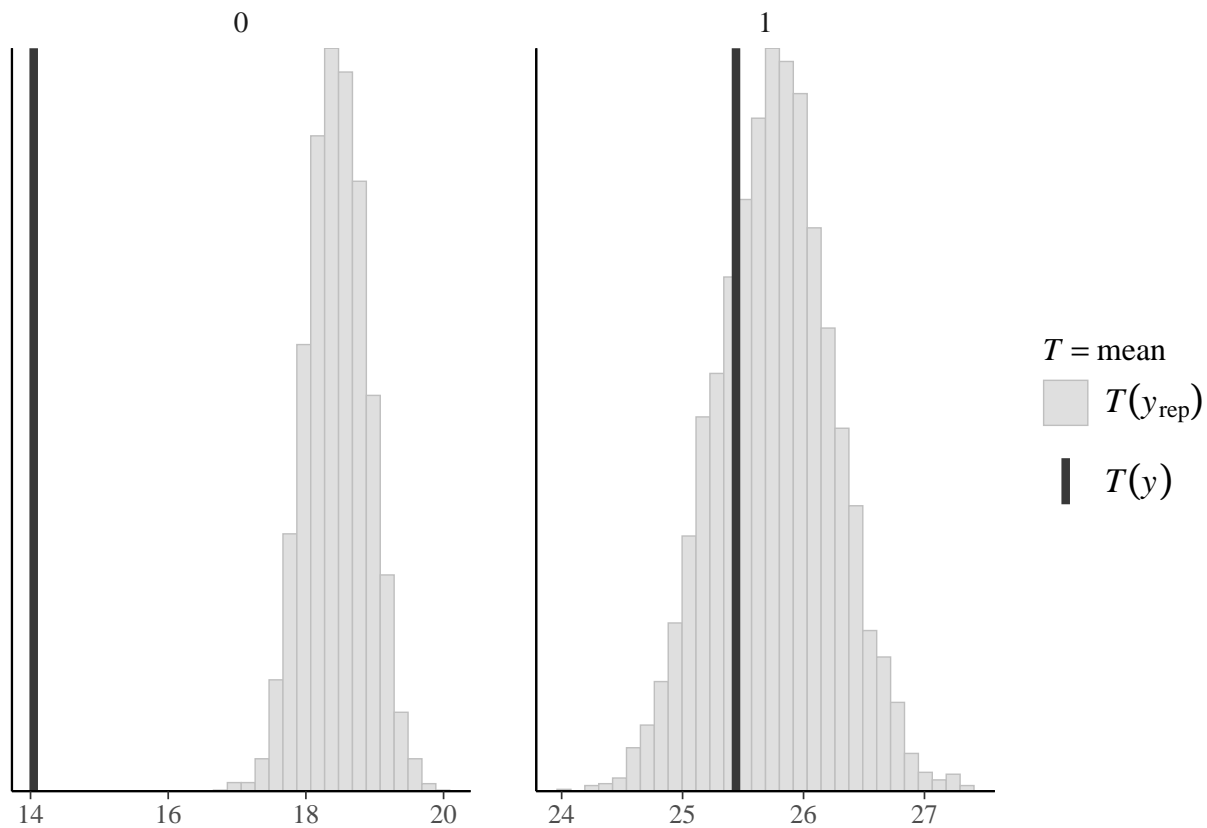
```
bayesplot::pp_check(object = men$Fat, yrep = y_rep_male, fun = "stat_grouped",
  group = binary_ankle_male, stat = "median")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



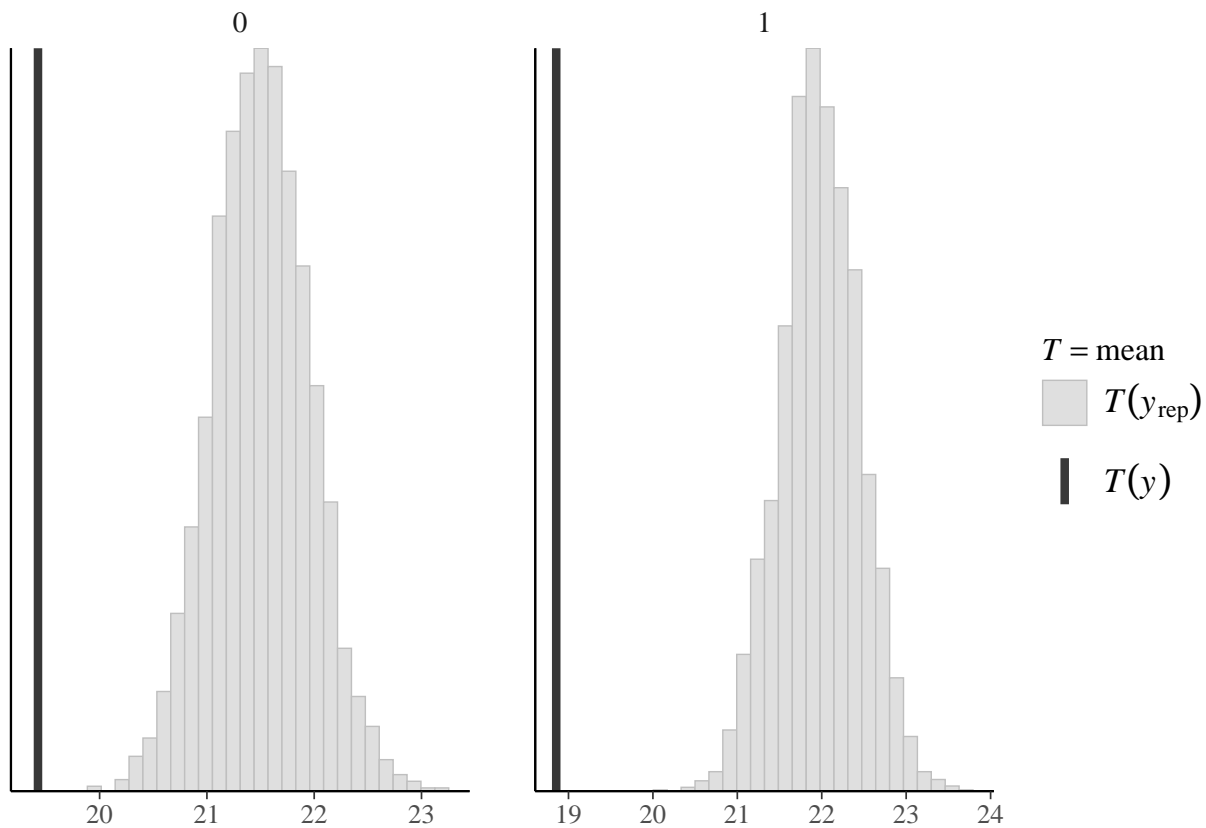
```
color_scheme_set("gray")
bayesplot::pp_check(object = men$Fat, yrep = y_rep_male,
  fun = "stat_grouped", group = binary_waist_male, stat = "mean")
```

`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.



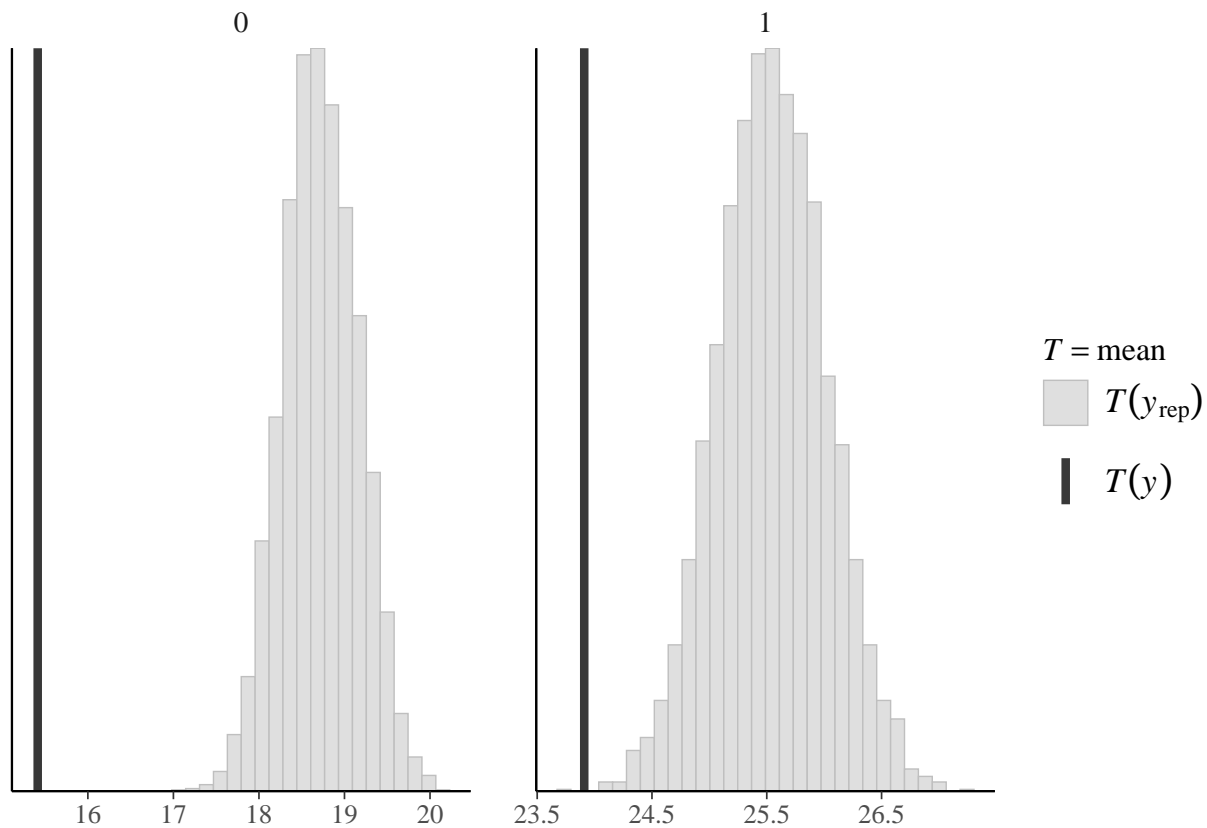
```
bayesplot::pp_check(object = men$Fat, yrep = y_rep_male,
  fun = "stat_grouped", group = binary_height_male, stat = "mean")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

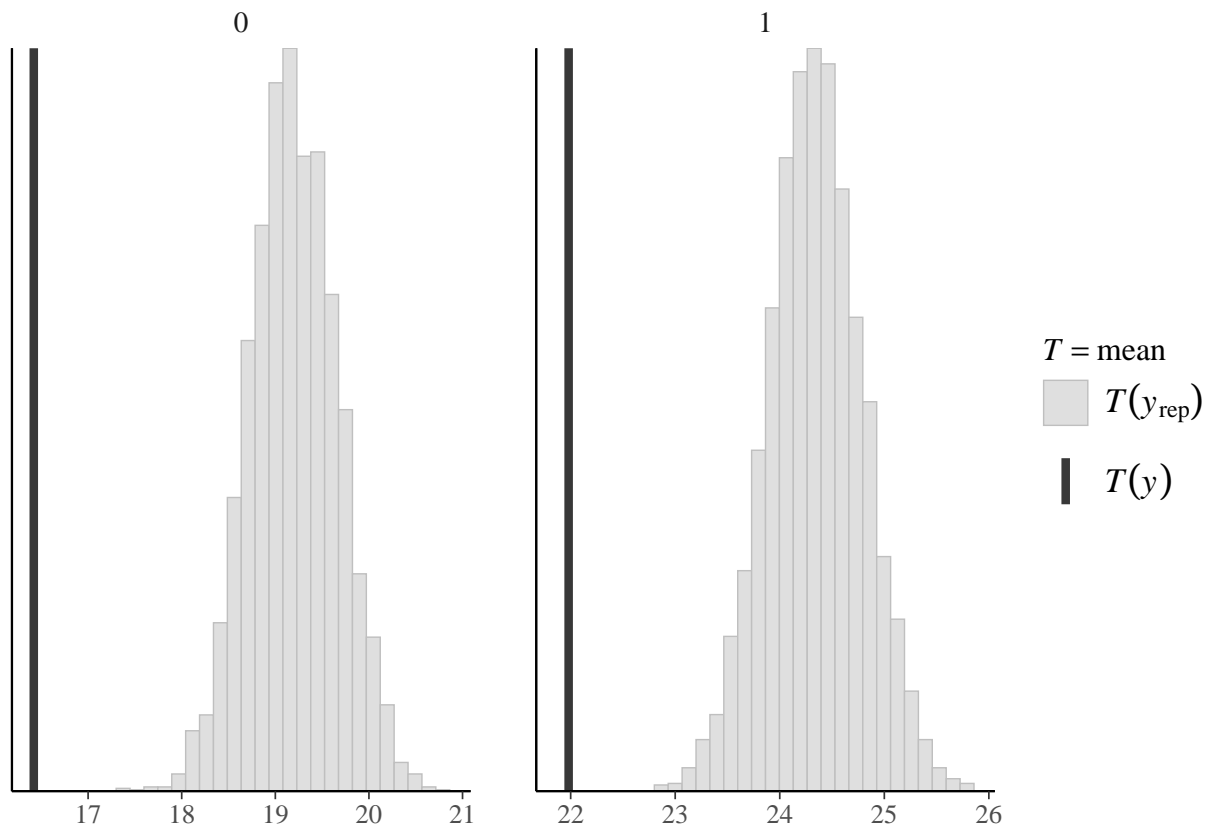
```
bayesplot::pp_check(object = men$Fat, yrep = y_rep_male,
  fun = "stat_grouped", group = binary_hips_male, stat = "mean")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



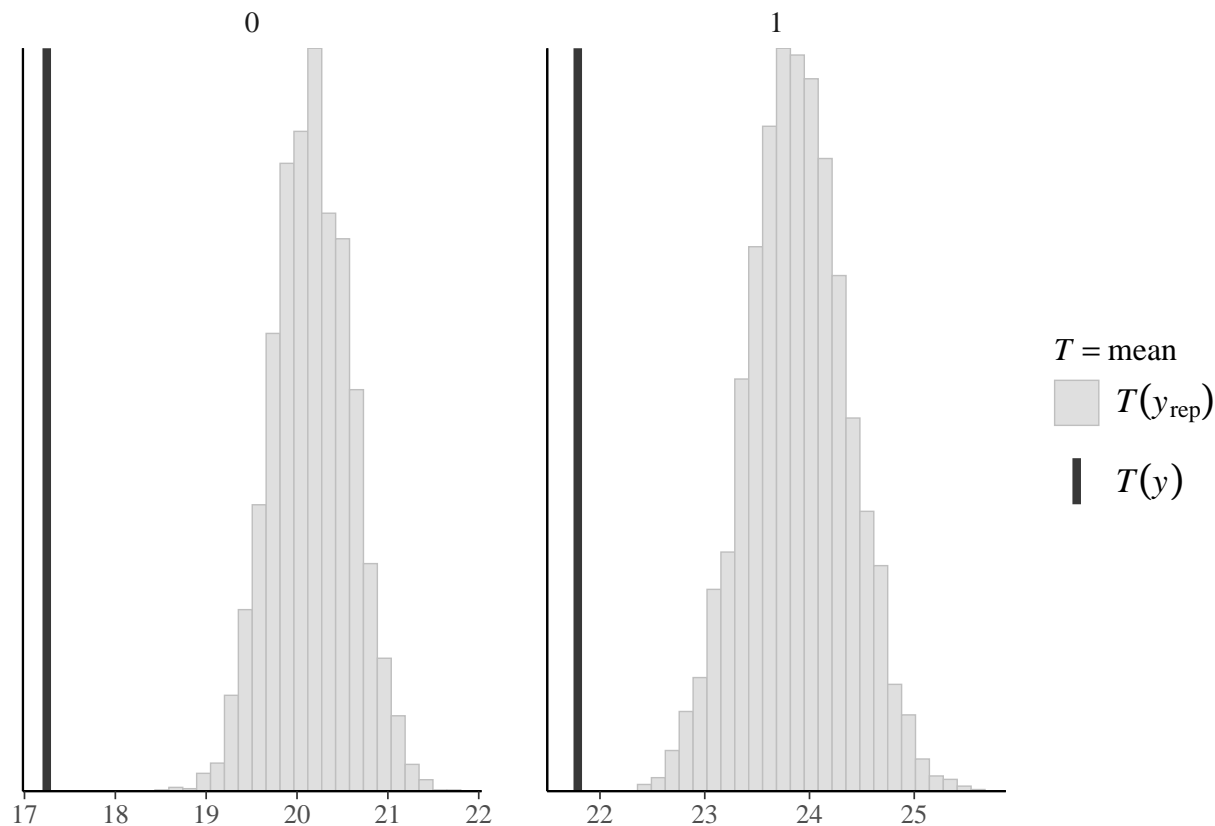
```
bayesplot::pp_check(object = men$Fat, yrep = y_rep_male, fun = "stat_grouped",
  group = binary_knee_male, stat = "mean")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



```
bayesplot::pp_check(object = men$Fat, yrep = y_rep_male, fun = "stat_grouped",
  group = binary_ankle_male, stat = "mean")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



```
# Summarize the posterior distributions
summary_male_cv <- prediction_summary_cv(model = fit_female, data = men, k = 3)

# Print the summary
print(summary_male_cv)
```

Summarize the posterior distributions

```
## $folds
##   fold      mae mae_scaled within_50 within_95
## 1     1 3.696158 0.7896502 0.4285714 0.9761905
## 2     2 2.835573 0.5887821 0.5714286 0.9642857
## 3     3 3.459173 0.7612981 0.4523810 0.9523810
##
## $cv
##      mae mae_scaled within_50 within_95
## 1 3.330301 0.7132435 0.484127 0.9642857

summary_male <- prediction_summary(model = fit_female, data = men)

# Print the summary
print(summary_male)

##      mae mae_scaled within_50 within_95
## 1 4.316417 1.225886 0.297619 0.7301587
```

```
age_group <- cut(men$Age, breaks = quantile(men$Age, probs = seq(0, 1, by = 0.25)), include.lowest = TRUE)
table(age_group)
```

Identify the groups where the prediction performs effectively

```
## age_group
## [-1.82,-0.725] (-0.725,-0.15] (-0.15,0.723] (0.723,2.87]
##          63          64          68          57
```

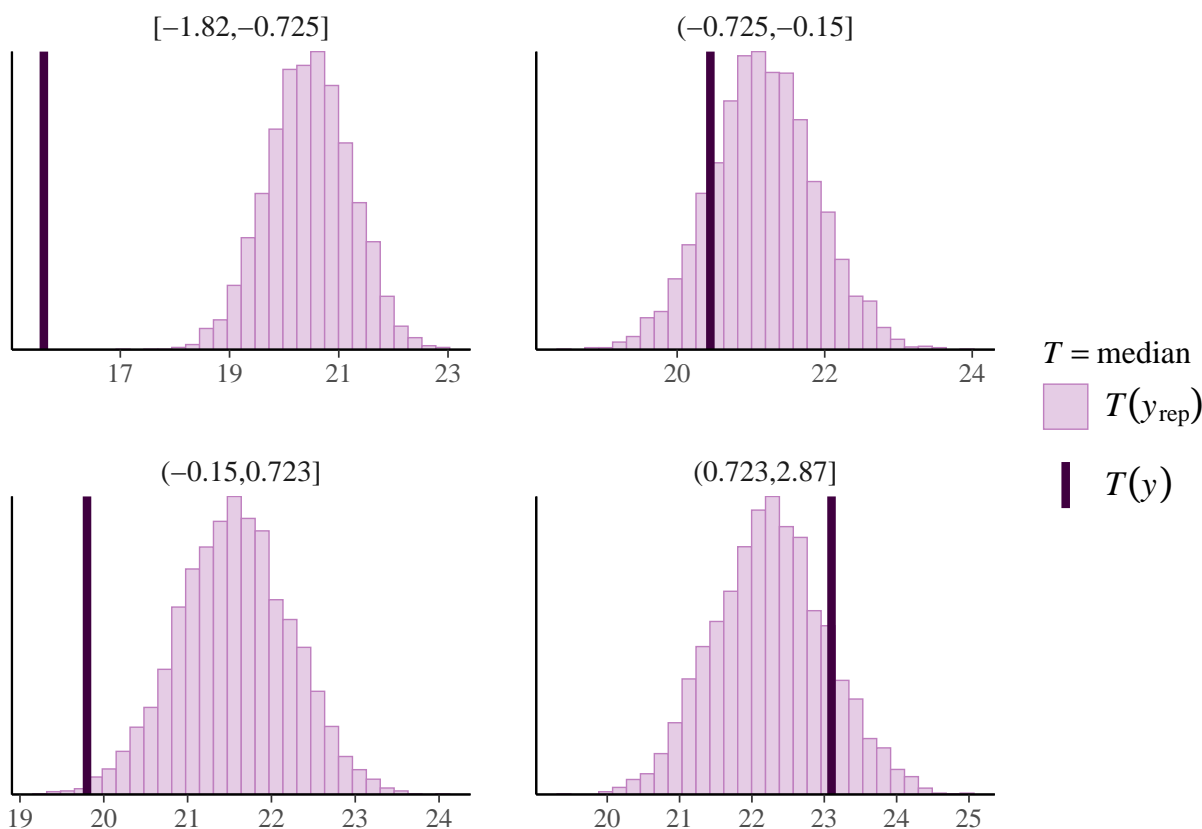
```
fat_group <- cut(men$Fat, breaks = quantile(men$Fat, probs = seq(0, 1, by = 0.25)), include.lowest = TRUE)
table(fat_group)
```

```
## fat_group
## [0,12.4] (12.4,19.2] (19.2,25.3] (25.3,47.5]
##          64          63          64          61
```

```
waist_group <- cut(men$Waist, breaks = quantile(men$Waist, probs = seq(0, 1, by = 0.25)), include.lowest = TRUE)
```

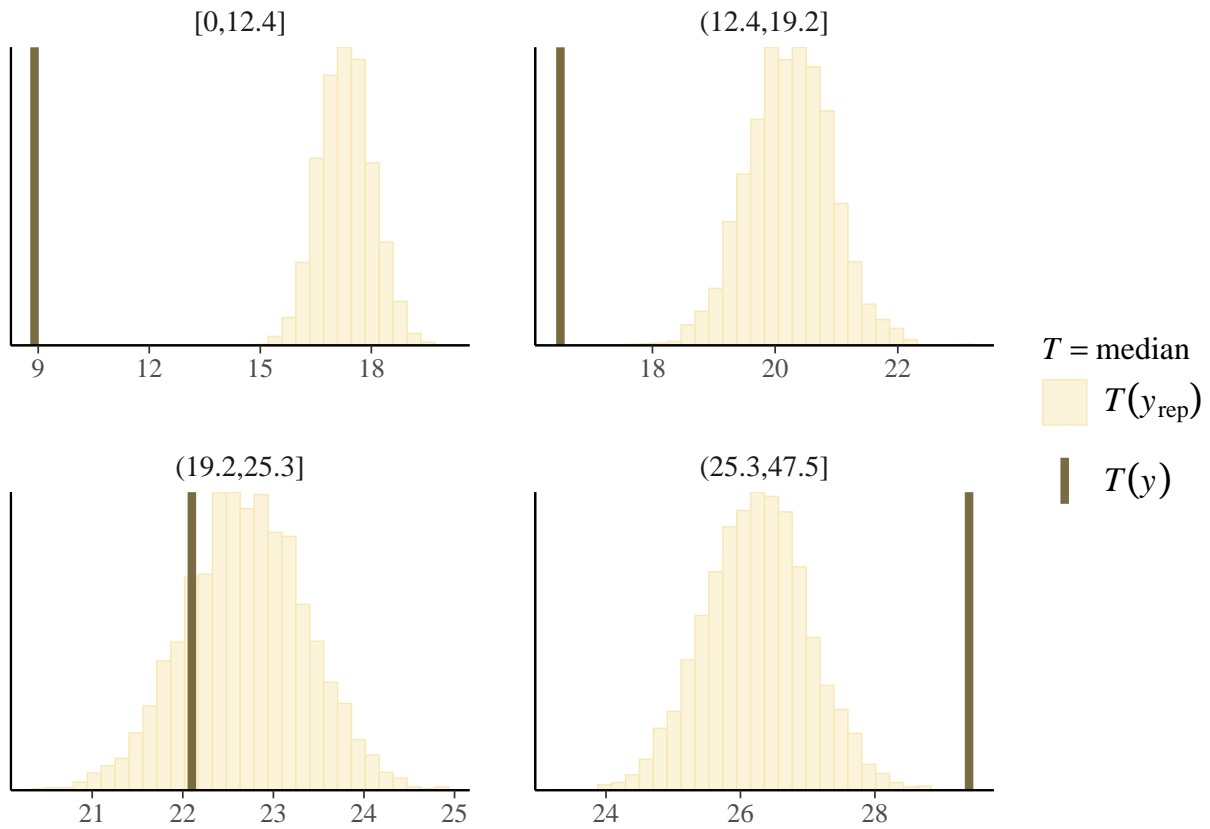
```
color_scheme_set(scheme = "purple")
bayesplot::pp_check(object = men$Fat, yrep = y_rep_male,
  fun = "stat_grouped", group = age_group, stat = "median")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



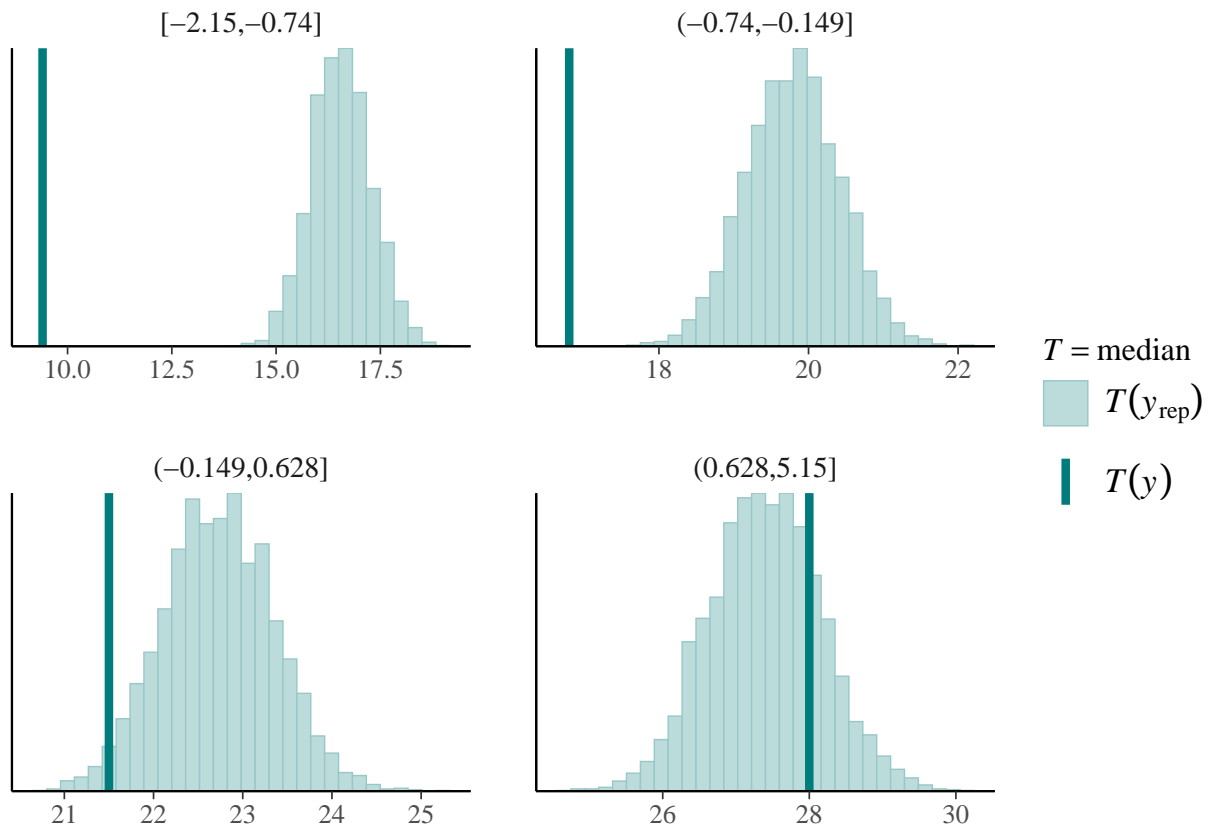
```
color_scheme_set(scheme = "yellow")
bayesplot::pp_check(object = men$Fat, yrep = y_rep_male,
  fun = "stat_grouped", group = fat_group, stat = "median")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



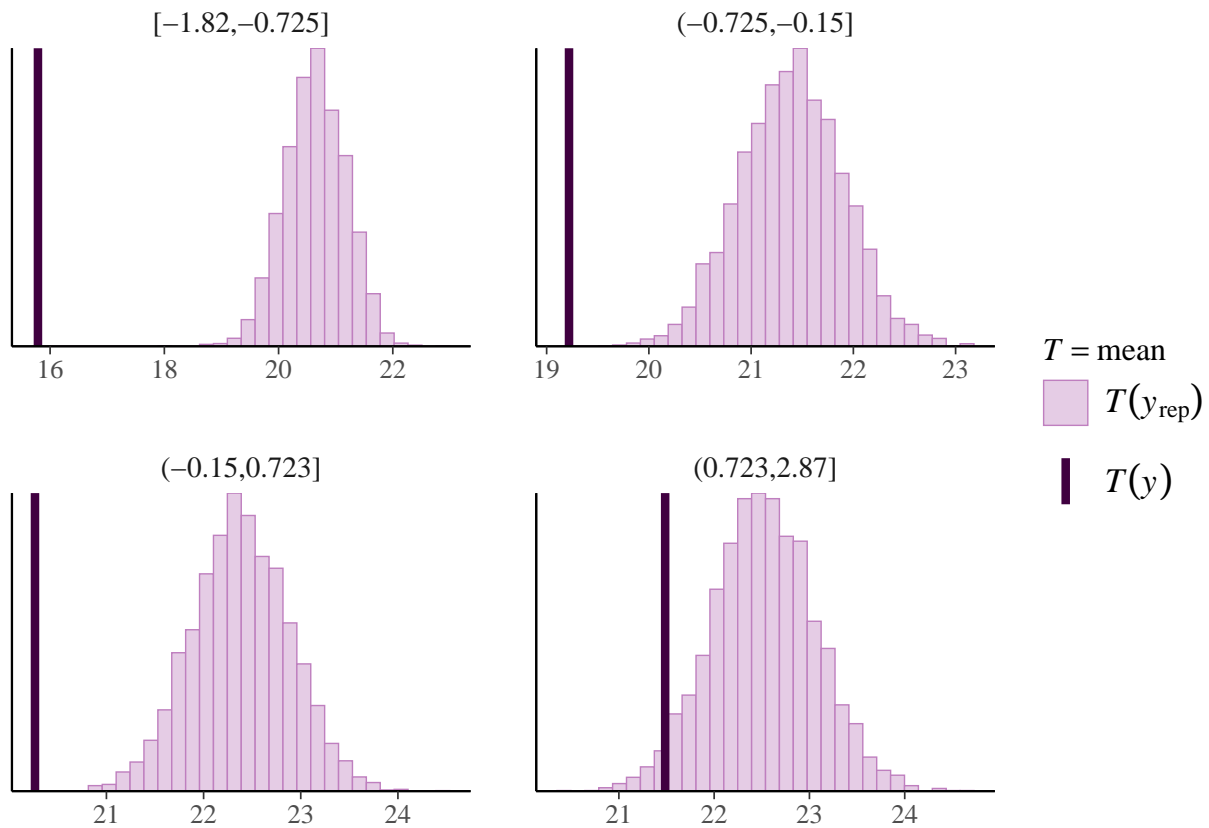
```
color_scheme_set(scheme = "teal")
bayesplot::pp_check(object = men$Fat, yrep = y_rep_male,
  fun = "stat_grouped", group = waist_group, stat = "median")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



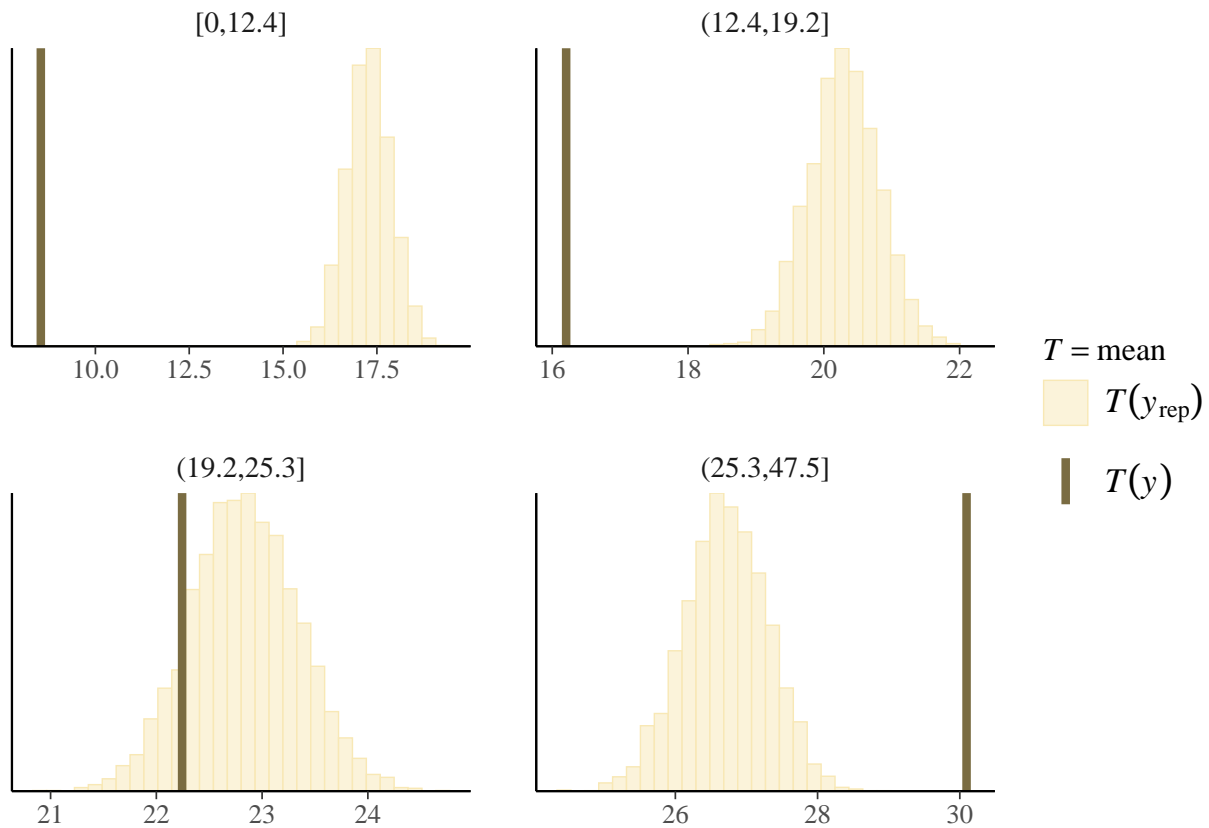
```
color_scheme_set(scheme = "purple")
bayesplot::pp_check(object = men$Fat, yrep = y_rep_male,
  fun = "stat_grouped", group = age_group, stat = "mean")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



```
color_scheme_set(scheme = "yellow")
bayesplot::pp_check(object = men$Fat, yrep = y_rep_male,
  fun = "stat_grouped", group = fat_group, stat = "mean")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
color_scheme_set(scheme = "teal")
bayesplot::pp_check(object = men$Fat, yrep = y_rep_male,
  fun = "stat_grouped", group = waist_group, stat = "mean")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

