Solution (100 points)

You may discuss this assignment with whomever you wish, but please prepare and submit work in groups of ***ONE to THREE*** students, no more and no fewer. ***Each group*** will submit a copy of their group's completed assignment, via BbLearn, including the ***names and student ID numbers of all group members*** who participated on the assignment. If you discover a mistake, you may submit another version before the deadline. The last submitted (on-time) version will be graded, with all team members receiving the same score, which will be recorded in BbLearn.

GROUP MEMBERS WHO DO NOT CONTRIBUTE SUBSTANTIALLY TO AN ASSIGNMENT MAY BE REQUIRED TO WORK IN THEIR OWN GROUP OF ONE FOR THE REMAINDER OF THE SEMESTER.

While you are permitted to discuss the assignment with other groups, please prepare your own group's code/output and written answers. GROUPS WHOSE CODE AND SOLUTIONS APPEAR SUBSTANTIALLY SIMILAR MAY BE SUBJECT TO A 10% PENALTY.

Please prepare solutions in a ***neat, organized and concise fashion***! I prefer typeset presentations (e.g., cut and paste code/output into MS Word with added exposition when appropriate; knitr via EMACS and ESS; knitr or R Markdown via RStudio, the latter being the method preferred by students in recent years). At the very least, you need to ensure code and output are presented with a fixed-width font. Neatly handwritten presentations may also be appropriate for some problems. Sloppily prepared or disorganized solutions will not receive full credit.

To complete the items below, I expect you to find and use material in our lecture notes, including code/output, possibly after some modification. Some questions may be answered with code and output alone, but some exposition may be required beyond code and output for other questions. It's up to you to communicate concisely!

7.1 Introduction

In §C.6, we considered a common improper prior distribution model for the prostate data with standardized inputs; this led to a known form of posterior distribution. See also §C.4 & C.5 for more on this improper prior model.

In §C.9 and C.10, we considered a partially improper and partially proper (vaguely informative) prior distribution model for the same prostate data; this led to known full conditional posterior distributions, which we exploited using our own 3-stage Gibbs sampling algorithm programmed in R (§C.9). See also §C.7 and C.8 for a similar but fully proper independence prior and related 2-stage Gibbs sampling algorithm. The form of the entire posterior was not of known form in this case. We also analyzed this case using Hamiltonian Monte Carlo (HMC) sampling as implemented in Stan using the rstan package in R (§C.10).

We summarized these analyses in §C.11.

In this final assignment, we repeat essentially these analyses and summary using the body fat data (use file `zfat.RDS` in BbLearn), instead of the prostate data. I standardized the inputs for the body fat data so that we may elicit priors as we did for the prostate data (§C.9) and so that we may compare all three analyses in a nice summary as illustrated in §C.11.

7.1.1 Data & LS Fit

I read the data and create an initial LS fit object for use later.

```
> zfat<- readRDS(file="zfat.RDS")
> zfat.lm<- lm(brozek ~ ., data=zfat)
```

7.2 Common Improper Prior Analysis

(1) Following §C.6.2, produce a posterior summary of the regression model effects (the betas!) and a corresponding plot of the marginal posterior t distributions for these effects as in that section. Add line color (use the `col` option in the `plot` and `lines` functions) in the same fashion as the `lty` option to help distinguish effect marginal distributions. You may have to perform further edits to the plot code of §C.6.2 to get reasonable results here. For example, horizontal and vertical plot limits must be changed as well as effect names in the plot legend! Incidentally, you will want to keep some objects created here for comparison to Gibbs sampling and HMC sampling of subsequent analyses. Show your summary and plot, including the code used to produce these. Be concise. Do not include more than requested here.

ANS: **Posterior Summary**. I simply cut and paste the summary code from §C.6.2, changed object names accordingly and omitted the frequentist summary (though it would be a nice check on the posterior summary given the correspondance of the Bayes results and frequentist results for this particular improper prior). We will use these results for comparison to subsequent analyses, below.

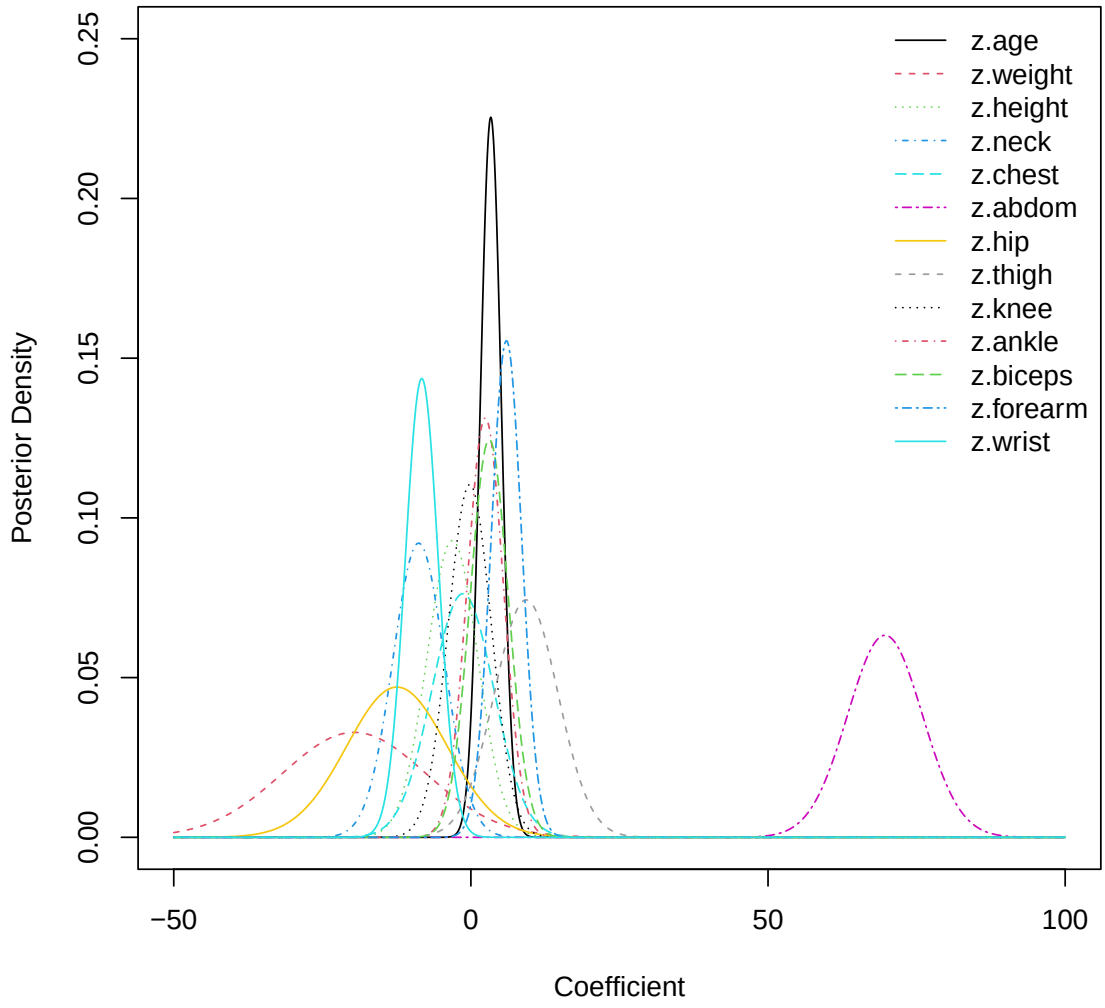
```
> ## Bayes posterior t means (df>1) and LS/MLE:
> tpostmeans<- coef(zfat.lm)
> ## Bayes posterior t df (n-p) and freq error df:
> tpostdf<- zfat.lm$df
> ## Bayes posterior t squared scales:
> tpostscales2<- diag(vcov(zfat.lm))
> ## Bayes posterior t variances (df>2):
> tpostvars<- tpostdf / (tpostdf -2) * tpostscales2
> ## Bayes posterior t standard deviations:
> tpostsds<- sqrt(tpostvars)
> ## Freq ses are Bayes t scales, not quite same as Bayes t sds:
> tfreqses<- sqrt(tpostscales2)
> ## Intervals:
> t975<- qt(p=1-0.05/2, df=tpostdf)
> tpost25lb<- tpostmeans - t975*sqrt(tpostscales2)
> tpost975ub<- tpostmeans + t975*sqrt(tpostscales2)
> ## Bayes/freq summary (See Figs. 5.10 and 5.11 in Wakefield's BFRM)
> round(cbind("pmean"=tpostmeans, "psd"=tpostsds,
+             "freqse"=tfreqses, "25lb"=tpost25lb,
+             "975ub"=tpost975ub),4)

               pmean     psd  freqse     25lb   975ub
(Intercept)   6.8223  4.0208  4.0039  -1.0653 14.7099
```

```
z.age          3.3504  1.7754  1.7679  -0.1324  6.8331
z.weight     -19.6478 12.1812 12.1299 -43.5434  4.2478
z.height      -3.1170  4.3090  4.2909 -11.5699  5.3360
z.neck        -8.7946  4.3465  4.3282 -17.3211 -0.2681
z.chest       -1.3430  5.2478  5.2257 -11.6375  8.9514
z.abdom       69.6833  6.3287  6.3020  57.2683 82.0982
z.hip        -12.4408  8.5101  8.4743 -29.1351  4.2534
z.thigh        9.2990  5.3848  5.3621  -1.2642 19.8622
z.knee        -0.1880  3.6240  3.6087  -7.2971  6.9211
z.ankle        2.4203  3.0490  3.0361  -3.5608  8.4014
z.biceps       3.0865  3.2155  3.2020  -3.2213  9.3943
z.forearm      5.9838  2.5747  2.5639   0.9330 11.0346
z.wrist       -8.2686  2.7866  2.7749 -13.7351 -2.8021


> ## Again, Bayes CI's are same (numerically) save as freq CIs:
> ## confint(zfat.lm) ## NOT REQUESTED, BUT A GOOD CHECK
```

ANS: **Marginal Posterior t Distributions Plot**. You must change the horizontal and vertical plot limits in the code to get a resonable looking plot. The requested additional colored lines help to distinguish effects' marginals.

```
> par(mfrow=c(1,1),mar=c(5, 4, 4, 2) +0.1)
> ## Grid of beta values to plot marg. post. t densities
> bgrid<- seq(-50,100,.05)  ## MADE REASONABLE CHANGE HERE
> ## Omit intercept marginal (param=1) as in Fig. 5.10:
> invisible(sapply(2:14, FUN=function(param,bgrid,pmean,pscale,pdf){
+     ## Using location-scale properties of t distribution to plot:
+      tmargdens<- dt((bgrid - pmean[param])/pscale[param], df=pdf)/
+          pscale[param]
+     if(param==2) ## ylim AND col CHANGE HERE
+         plot(bgrid,tmargdens,type="l",xlab="Coefficient",
+              ylab="Posterior Density",
+              ylim=c(0,0.25), lty=param-1, col=param-1)
+     else
+         lines(bgrid,tmargdens,lty=param-1, col=param-1)
+ },
+ bgrid=bgrid,
+ pmean=tpostmeans,
+ pscale=sqrt(tpostscales2),
+ pdf=tpostdf
+ ))
```

```
> ##abline(v=0,lty=4) ## not sure why 4 (not necessary here)
> legend("topright",legend=names(zfat)[2:14],
+        bty="n",lty=1:13, col=1:13)
```



7.3 Combination Improper/Proper Independence Prior Analysis: Gibbs

7.3.1 Eliciting a Prior

Following §C.9, we first elicit priors for the effects, excluding β_0 and σ^2, which are given the same improper priors of that section. Because the output/response is not on the log scale, as in the prostate example, we will consider it unlikely that the standardized covariates will change the mean body fat percentage by more than 50 percent (not 10 or log(10)) as the covariates change over their standardized range of [0,1]. This means that we believe the (absolute value of the) effects are unlikely to be more than 50. Assuming, a priori, normality and mean effects of 0 (a priori null effects), as in §C.9, this translates to a prior variance of

$$V = (50/1.96)^2$$

for the effects. (Given the results of the previous analysis, above, you might question this prior! Why? No need to answer this.)

7.3.2 Gibbs Sampling Algorithm

With above prior distribution, I repeat the 3-stage Gibbs sampling algorithm of §C.9.3, where the full conditional distribution parameters are as discussed in §C.9. (I correct an error in starting value notation, too.)

For the initially chosen values of $\sigma^2 = \sigma^{2(t=0)}$ and $\beta_0 = \beta_0^{(t=0)}$

1. sample

$$\begin{aligned}\boldsymbol{\beta}^{*(t+1)} \,|\, \beta_0^{(t)}, \sigma^{2(t)}, \mathbf{y} &\sim [\boldsymbol{\beta}^* \,|\, \beta_0^{(t)}, \sigma^{2(t)}, \mathbf{y}] \\ &= \mathrm{N}\Big((\sigma^{-2(t)}(\mathbf{X}^{*t}\mathbf{X}^*) + V^{-1}\boldsymbol{I})^{-1}(\sigma^{-2(t)}(\mathbf{X}^{*t}\mathbf{X}^*)\widehat{\boldsymbol{\beta}}^* + V^{-1}\boldsymbol{m}_0^*), \\ &\quad (\sigma^{-2(t)}(\mathbf{X}^{*t}\mathbf{X}^*) + V^{-1}\boldsymbol{I})^{-1}\Big),\end{aligned}$$

2. sample

$$\begin{aligned}\beta_0^{(t+1)} \,|\, \boldsymbol{\beta}^{*(t+1)}, \sigma^{2(t)}, \mathbf{y} &\sim [\beta_0 \,|\, \boldsymbol{\beta}^{*(t+1)}, \sigma^{2(t)}, \mathbf{y}] \\ &= \mathrm{N}\left(\bar{y}^{*(t+1)}, \frac{\sigma^{2(t)}}{n}\right)\end{aligned}$$

3. sample

$$\begin{aligned}\sigma^{2(t+1)} \,|\, \boldsymbol{\beta}^{(t+1)}, \mathbf{y} &\sim [\sigma^2 \,|\, \boldsymbol{\beta}^{(t+1)}] \\ &= \text{inv-}\chi^2\left(n, \frac{SSE + (\boldsymbol{\beta}^{(t+1)} - \widehat{\boldsymbol{\beta}})^t(\mathbf{X}^t\mathbf{X})(\boldsymbol{\beta}^{(t+1)} - \widehat{\boldsymbol{\beta}})}{n}\right)\end{aligned}$$

4. repeat 1,2 & 3 "to convergence"

All quantities are as defined in §C.9.2 of our notes.

We will run the algorithm in three chains, each chain consisting of $M = 10000$ iterations, not including starting values. I get you started with some code adapted from §C.9.4.

```
> ## Modified code from Section C.9.3 of our notes:
> y<- zfat.lm$model[,1]
> X<- model.matrix(zfat.lm)
> Xstar<- X[,-1]
> (n<- dim(X)[1])

[1] 252

> (p<- dim(X)[2])

[1] 14

> k<- p - 1
> XtX<- crossprod(X)
> XstartXstar<- XtX[-1,-1]
> ##bhat<- solve(XtX)%*%t(X)%*%y
> bhat<- solve(XtX, crossprod(X,y))
> ##prebstarhat<- solve(XstartXstar)%*%t(Xstar)
> prebstarhat<- solve(XstartXstar,t(Xstar))
> m0star<- rep(0,k) ## bstar prior mean
>
> ## Prior variance for b1-bk (i.e., for bstar not for b0)
> V<- (50/1.96)^2
> ## Prior precision for bstar
> Vinv<- V^{-1}
> B= Vinv * diag(k)
>
> ## FC df for sigma2
> nuhat<- n
> ## Intermediate computation for sigma2 FC
> SSE <- sum((y - X%*%bhat)^2)
>
> nChain<- 3
> M<- 10000
>
> sigma2 <- matrix(NA,nChain,M+1)
> beta<- array(NA,c(nChain,M+1,p))
```

7.3.3 Starting Values

Determining starting values ("the first link") in a Gibbs sampling chain (as in Markov chain Monte Carlo (MCMC)) is somewhat of an art. (We may say "initial values" or "starting values.") The instructions here for determining starting values are not beyond the realm of what is done in practice to obtain starting values. Generally speaking, we want to create "dispersed" starting values so that, hopefully, we see all of these "far-flung" starting value links converge to a common range of values, with all chains mixing among each other, which provides a necessary (not sufficient) condition for convergence to the posterior distribution. (Creating starting values that are somehow too far out in the tails of the posterior distribution may cause the algorithm to fail due to very small likelihood values or prior density/mass values creating numerical problems. Not here.)

(2) For σ^2, use the three starting values of $\widehat{\sigma}^2/10$, $\widehat{\sigma}^2$, and $10\widehat{\sigma}^2$, where $\widehat{\sigma}^2$ is the MSE from the `zfat.lm` object created in code above—the LS fit to the fat data with covariates mapped to [0,1]. Show the three starting values and the code used to produce them.

ANS:

```
> ## 3 starting values as requested:
> sigma2[1,1]<- summary(zfat.lm)$sigma^2 / 10 ## chain 1
> sigma2[2,1]<- summary(zfat.lm)$sigma^2      ## chain 2
> sigma2[3,1]<- summary(zfat.lm)$sigma^2 * 10 ## chain 3
> sigma2[,1]

[1]   1.5904  15.9039 159.0393
```

(3) You will also need three starting values for the intercept, β_0. For these three values, generate

$$\beta_0 \sim N(\widehat{\beta}_0, \sigma^{2(0)}/n),$$

one for each of the three starting values for σ^2, just mentioned, generically denoted as $\sigma^{2(0)}$. (Note that $N(\widehat{\beta}_0, \sigma^{2(0)}/n)$ is akin to the full conditional for β_0, but not exactly the same. Again, determining starting values is much an art.) Please use `set.seed(8675309 + 86011)` immediately before generating the first of these three $\beta_0^{(0)}$ starting values, with the second and third starting values generated before any other random number generation. Show the three starting values and the code used to produce them.

ANS:

```
> ## 3 starting values as requested:
> set.seed(8675309 + 86011)
> beta[1,1,1]<- rnorm(n=1, m=bhat[1],sd=sqrt(sigma2[1,1] / n))
> beta[2,1,1]<- rnorm(n=1, m=bhat[1],sd=sqrt(sigma2[2,1] / n))
> beta[3,1,1]<- rnorm(n=1, m=bhat[1],sd=sqrt(sigma2[3,1] / n))
> beta[,1,1]

[1] 6.6996 6.6954 7.9729
```

7.3.4 Algorithm Code

(4) Continuing to follow §C.9.4, show the (remainder of the) Gibbs sampling code that you will run to produce samples from the posterior distribution. Just report the code for the moment. Are there any changes that you made to the code?

ANS: No changes to the (remaining) Gibbs sampling code from §C.9.4 is necessary, though I did change the name of the file that I saved results to.

```
> ## not evaluted in knitr
> ## betastar starting values not required (see algorithm)
>
> library(mvtnorm) ## you may need to install this
>
> for (chain in 1:nChain){ ## chain loop
+     for (i in 1:M){      ## iteration loop
+         if(i %% 1000 == 0) print(paste0("Chain = ", chain,
+                                         ". Iteration = ", i, "."))

+         ########################################
+         ## FC for bstar (beta w/o beta0)
+         #### variance
+         bstarvar<- solve(sigma2[chain, i]^{-1} * XstartXstar + B)
+         #### mean
+         bstarhat<- prebstarhat%*%(y - beta[chain,i,1])
+         bstarmean<- bstarvar %*% (sigma2[chain,i]^{-1} *
+                                   XstartXstar%*%bstarhat +
+                                   Vinv * m0star)
+         #### sample bstar (beta w/o beta0)
+         beta[chain,i+1,-1]<- as.vector(rmvnorm(n=1,mean=bstarmean,
+                                                sigma=bstarvar))
+         ########################################
```

```
+         ## FC for beta0
+         #### mean
+         ystarbar<- mean(y - Xstar%*%beta[chain,i+1,-1])
+         ## sample beta0
+         beta[chain,i+1,1]<- rnorm(n=1, mean=ystarbar,
+                                   sd=sqrt(sigma2[chain,i] / n))
+         ########################################
+         ## FC for sigma2
+         ## See BDA 3 appendix A for generating scaled inv-chi2 (see also
+         ## Wakefield's expression (5.45)...typo?):
+         sigma2hat<- (SSE + crossprod(X%*%(beta[chain,i+1,] - bhat))) / nuhat
+         sigma2[chain, i+1]<- nuhat * sigma2hat / rchisq(n=1,df=nuhat)
+     } ## end iterations
+ } ## end chains
>
> ## Good idea to save the fit to a file for later use:
> save(list=c("beta", "sigma2"), file="zfat.fit.RData")
>
> detach(package:mvtnorm)
> rm(y,X,Xstar,n,p,k,XtX,XstartXstar,bhat,prebstarhat,m0star,V,Vinv,B,nuhat,SSE)
```

7.3.5 Convergence Diagnostics

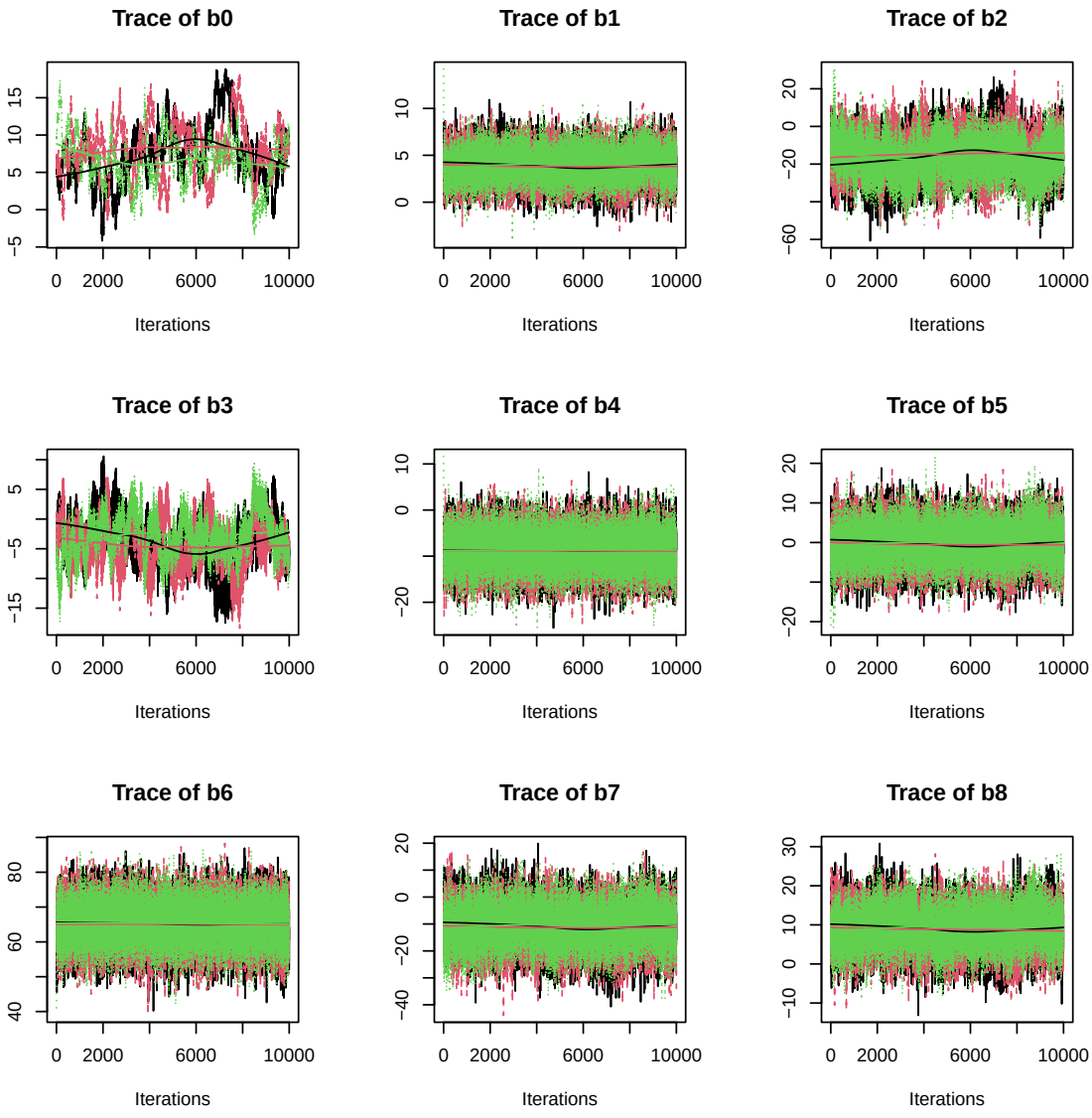
(5) Run the above code (you may need to get the `mvtnorm` package first) to obtain samples from the posterior distribution, and follow §C.9.5 to obtain history (trace) plots and the Brooks-Gelman-Rubin (BGR) potential scale reduction factor (psrf) (graphically or numerically) to assess convergence. Omit the density plots, please. Report your code and results and comment briefly on convergence. (You may need to modify slightly the code from §C.9.5 for this item.)

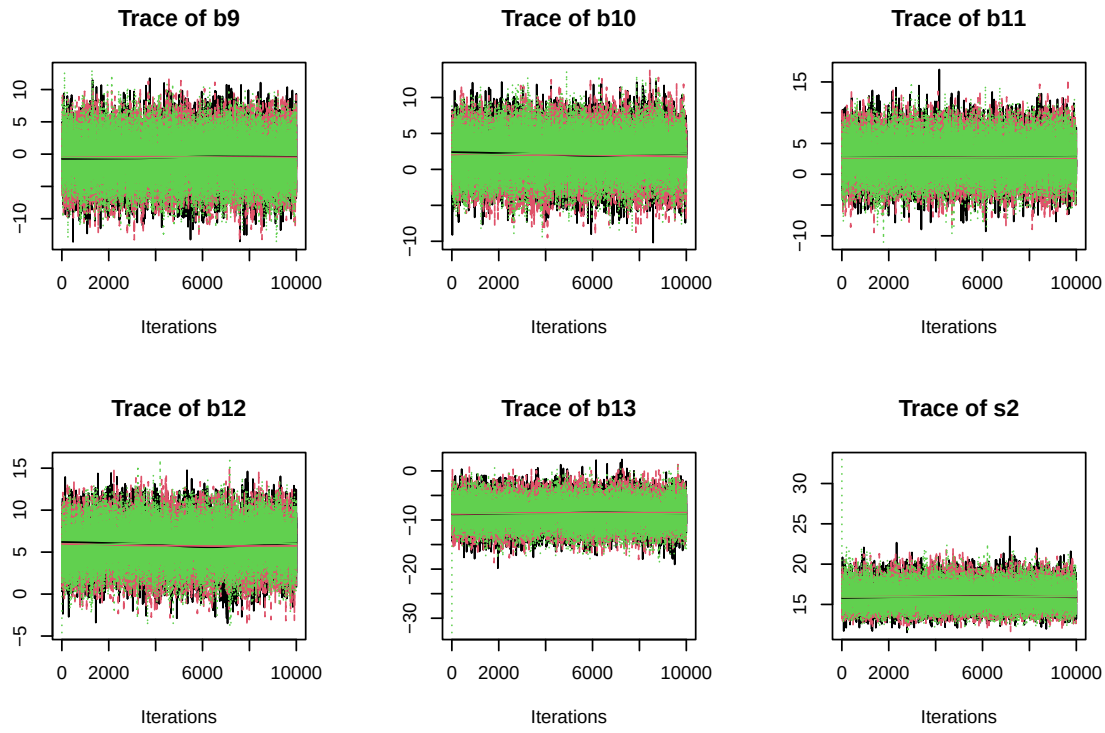
ANS: History plots indicate convergence to occur almost instantly as all chains appear to "mix" throughout their history. We cans see a few parameters whose starting values appear initially relatively far from bulk of the posterior (e.g., β_{13} and σ^2 green chains, for me) only to converge to a common range of values for all three chains after a very brief "burn-in" (or warm-up) of a few iterations.

Most psrf plots and values are all well below the rule-of-thumb values of 1.1 to 1.2, confirming the convergence seen in the history plots. The psrf for β_0 and β_3 (height), whose history plots do not appear to mix as thoroughly as other parameters, are a bit higher, with estimates near 1.1. Some people may want to run the chains for more iterations, but the history plots and psrf indicate to me sufficient convergence.

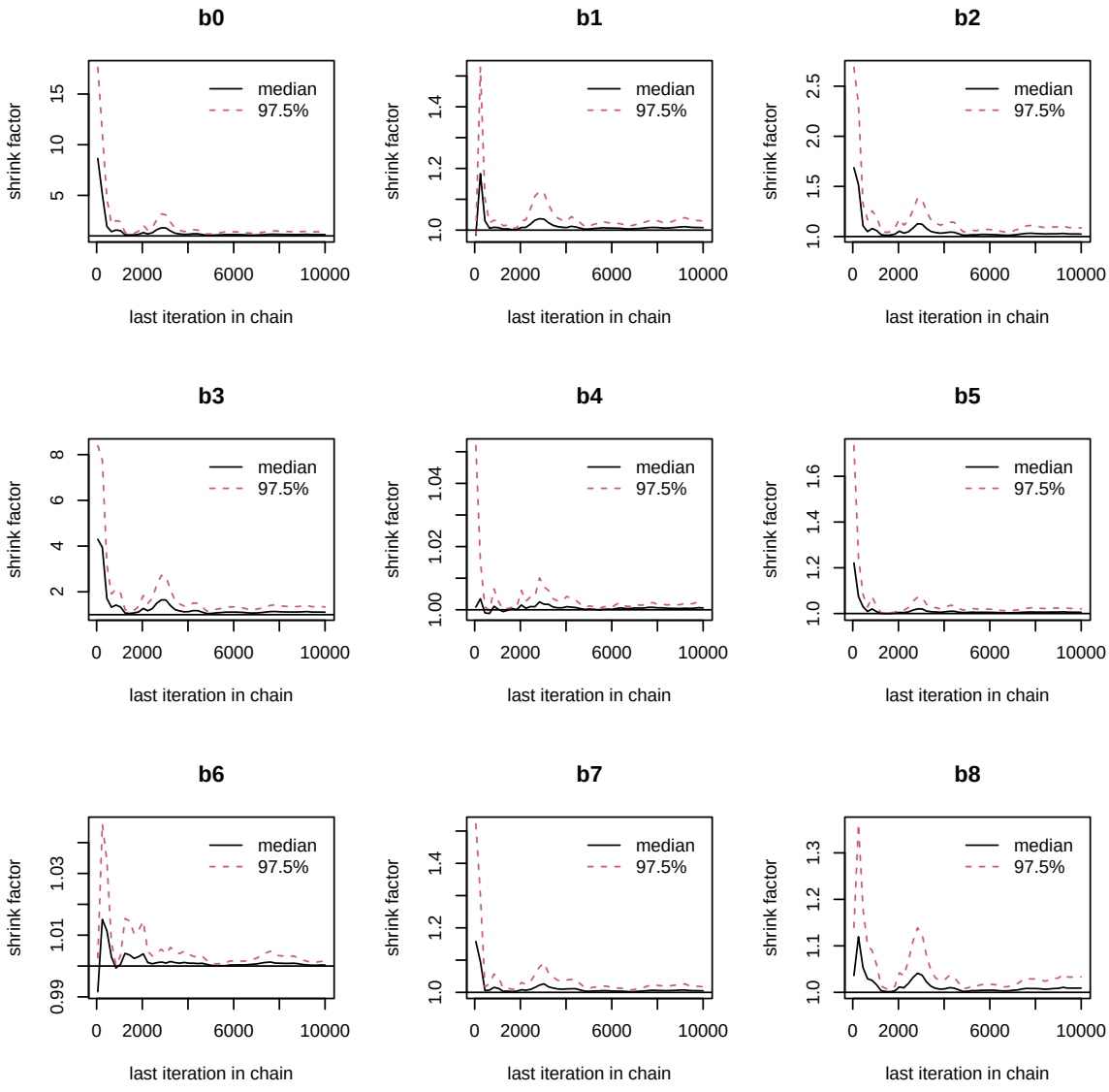
```
> load(file="zfat.fit.RData")
> ## I remove initial values or NAs (with -1 index)
> c1<- cbind(beta[1,-1,],sigma2[1,-1]) ## chain 1
> c2<- cbind(beta[2,-1,],sigma2[2,-1]) ## chain 2
> c3<- cbind(beta[3,-1,],sigma2[3,-1]) ## chain 3
> library(coda)
> zpGibbs<- mcmc.list(as.mcmc(c1),as.mcmc(c2),as.mcmc(c3))
> ##nchain(zpGibbs)
> ##nvar(zpGibbs)
> varnames(zpGibbs)<- c(paste0("b",0:13), "s2") ## not 0:8!!
```

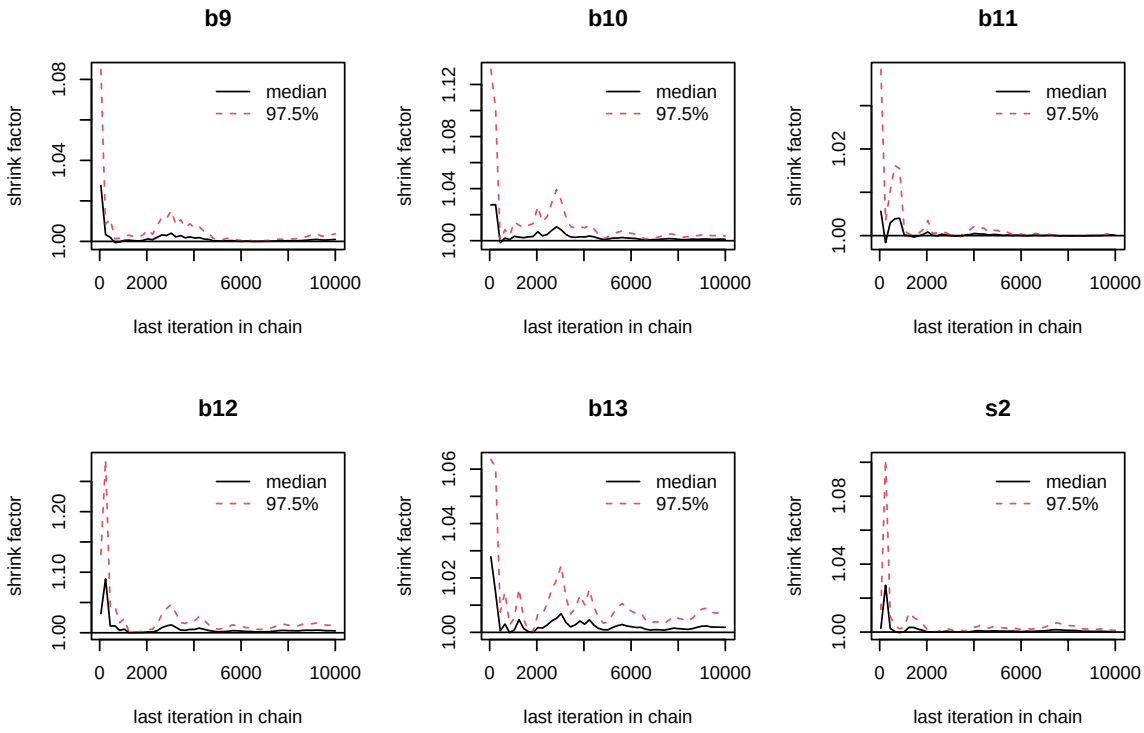
```
> plot(zpGibbs, density=FALSE)
```


Trace of b0
Trace of b1
Trace of b2
Trace of b3
Trace of b4
Trace of b5
Trace of b6
Trace of b7
Trace of b8
Iterations
0
2000
6000
10000




```
> gelman.plot(zpGibbs, ask=FALSE)
```


b0
b1
b2
b3
b4
b5
b6
b7
b8
median
97.5%
shrink factor
last iteration in chain
0
2000
6000
10000




```
> gelman.diag(zpGibbs)

Potential scale reduction factors:

    Point est. Upper C.I.
b0        1.12       1.39
b1        1.01       1.03
b2        1.02       1.09
b3        1.11       1.34
b4        1.00       1.00
b5        1.01       1.02
```

```
b6        1.00       1.00
b7        1.00       1.02
b8        1.01       1.03
b9        1.00       1.00
b10       1.00       1.00
b11       1.00       1.00
b12       1.00       1.01
b13       1.00       1.01
s2        1.00       1.00

Multivariate psrf

1.08
```

7.3.6 Posterior Summary

(6) Following §C.9.6, report the standard summary of the posterior, omitting the first 5000 iterations from each chain. Just give the code and summary results, do not use all of the code/results shown in §C.9.6. We will compare these results to those of the other analyses herein, later. No need to comment yet.

ANS:

```
> ## Obtain the last 5000 iterations of each chain (using zpGibbs mcmc.list
> ## object created previously):
> zpGibbs5to10k<- window(zpGibbs, start=5001, end=10000)
>
> ## Bayes summary:
> (bayessum<- summary(zpGibbs5to10k))



Iterations = 5001:10000
Thinning interval = 1
Number of chains = 3
Sample size per chain = 5000

1. Empirical mean and standard deviation for each variable,
   plus standard error of the mean:

       Mean    SD Naive SE Time-series SE
b0    7.781  3.76   0.0307         0.6056
b1    3.872  1.74   0.0142         0.0403
```

```
b2  -15.167 10.71   0.0874          0.7037
b3   -4.170  4.09   0.0334          0.5821
b4   -8.746  4.24   0.0346          0.0359
b5   -0.388  4.97   0.0406          0.1042
b6   65.093  6.06   0.0495          0.0500
b7  -11.186  7.69   0.0627          0.1636
b8    8.884  5.23   0.0427          0.1294
b9   -0.471  3.54   0.0289          0.0286
b10   2.014  3.01   0.0246          0.0318
b11   2.733  3.17   0.0259          0.0251
b12   5.808  2.54   0.0208          0.0419
b13  -8.599  2.78   0.0227          0.0270
s2   16.033  1.45   0.0119          0.0131

2. Quantiles for each variable:

       2.5%       25%     50%   75%  97.5%
b0    0.406   5.36536   7.645 10.02 16.343
b1    0.486   2.68849   3.868  5.04  7.291
b2  -35.842 -22.44828 -15.191 -7.94  5.855
b3  -13.019  -6.70287  -4.064 -1.47  3.815
b4  -17.085 -11.61131  -8.729 -5.90 -0.529
b5  -10.064  -3.76045  -0.417  2.94  9.393
b6   53.138  60.98258  65.113 69.24 77.044
b7  -26.194 -16.31199 -11.137 -5.99  3.844
b8   -1.408   5.37864   8.930 12.38 18.893
b9   -7.413  -2.85747  -0.470  1.94  6.469
b10  -3.892   0.00342   2.020  4.00  7.869
b11  -3.442   0.60080   2.703  4.87  8.916
b12   0.832   4.11000   5.822  7.51 10.791
b13 -14.035 -10.42449  -8.605 -6.76 -3.105
s2   13.437  15.00615  15.950 16.96 19.140
```

7.4 Combination Improper/Proper Independence Prior Analysis: Stan HMC

(7) Using code in §C.10.8, use HMC in Stan to sample $\boldsymbol{\beta}, \sigma^2 \,|\, \mathbf{y}$. Please omit (or comment out) all code for sampling $\mathbf{x}^{*t}\boldsymbol{\beta} \,|\, \mathbf{y}$ and $Y^* \,|\, \mathbf{x}^*, \mathbf{y}$; we will not do this here. Be sure to change the prior standard deviation code in the transformed data block! Show your code.

ANS:

```
> writeLines(readLines("./zfat.stan"))

functions {
  // nothing for this homework
}

data{
  int<lower=1> N;
  int<lower=1> p;
  matrix[N,p] X;
  vector[N] y;
  vector[p-1] m0;
  // vector[p] xstar; // not used for this homework
}

transformed data{
  // Cheap illustration of transformed data using prior sd of beta's:
  real rootV=50/1.96;  // changed for this homework
}

parameters{
  vector[p] beta;
  real lnsigma2;
}

transformed parameters{
  real<lower=0> sigma2 = exp(lnsigma2);
}

model{
  y ~ normal(X*beta, sqrt(sigma2));
  // Prior for beta1-betak (excluding intercept beta0):
```

```
  for(j in 2:p) beta[j]~ normal(m0[j-1], rootV);

  /*****

   Other parameters (in parameters block), without an explicit prior
   specification, here, will receive a flat prior over their
   implied support.  For us, this means [beta0] will be proportional
   to 1 over (-inf,inf), improper as desired, and that
   [lnsigma2] is proportional to 1 on (-inf, inf) (improper), which
   corresponds to [sigma2] proportional to 1/sigma2, improper as
   desired (detais omitted).

  *****/
}

generated quantities{ // not done for this homework
  // Composition sampling. (no longer MCMC at this point)
  // real xstarbeta = dot_product(xstar, beta);
  // real ystar = normal_rng(xstarbeta, sqrt(sigma2));
}
```

(8) Create the data list necessary to run your Stan code. Just show your code to create the data list that you will pass to Stan, shortly.

ANS:

```
> ## Using objects computed previously.
> X<- model.matrix(zfat.lm)
> zfat.data<- list(
+     N=dim(X)[1],
+     p=dim(X)[2],
+     X=X,
+     y=zfat.lm$model[,1],
+     m0=rep(0, dim(X)[2]-1)
+     ## xstar=xstar  ## not for this homework
+     )
> rm(X)
```

(9) We will let Stan generate starting values for us (i.e., don't follow §C.10.12). Using §C.10.13, execute your Stan model with three chains, 10000 iterations for each chain, and 5000 warmup iterations for each chain. Again, we do not obtain samples of $\mathbf{x}^{*t}\boldsymbol{\beta} \,|\, \mathbf{y}$

or $Y^* \,|\, \mathbf{x}^*, \mathbf{y}$ in this homework, so modify the code accordingly. (And, we let Stan obtain starting values for us—fingers crossed.) Add `seed = 90210` to the list of arguments in the `sampling` function. Show your code, but do not show the output yet. Don't forget to translate Stan (to C++) then compile to an executable model; see §C.10.9 and §C.10.10.

ANS:

```
> ## not evaluated in knitr
> library(rstan,quietly=TRUE)
> options(mc.cores = parallel::detectCores())
> rstan_options(auto_write = TRUE)
> zfat.stanc<- stanc(file="./zfat.stan")
```

```
> ## not evaluated in knitr
> zfat.stanmod<- stan_model(stanc_ret=zfat.stanc)
>
> zfat.fit<- sampling(zfat.stanmod,
+                     data=zfat.data,
+                     pars=c("beta","sigma2"),
+                     chains=3,
+                     iter=10000,
+                     warmup=5000,
+                     ## init=zprostate1.init, ## let Stan do this
+                     seed=90210,
+                     refresh=1000)
> ## Good idea to save the fit to a file for later use:
> save(list=c("zfat.stanc", "zfat.stanmod",
+             "zfat.fit"), file="zfatStan.fit.RData")
```

7.4.1 Convergence Diagnostics

(10) Following §C.10.14, obtain history (trace) plots and the Brooks-Gelman-Rubin (BGR) potential scale reduction factor (psrf) (graphically or numerically) to assess convergence. Omit the density plots, please. Report your code and results and comment briefly on convergence. (You may need to modify slightly the code from §C.10.14 for this item.)

ANS: History plots indicate convergence to occur almost instantly as all chains appear to mix throughout their history, and generally appear to mix much more thoroughly than those of the Gibbs analysis. Evidently, Stan chose sufficient starting values,

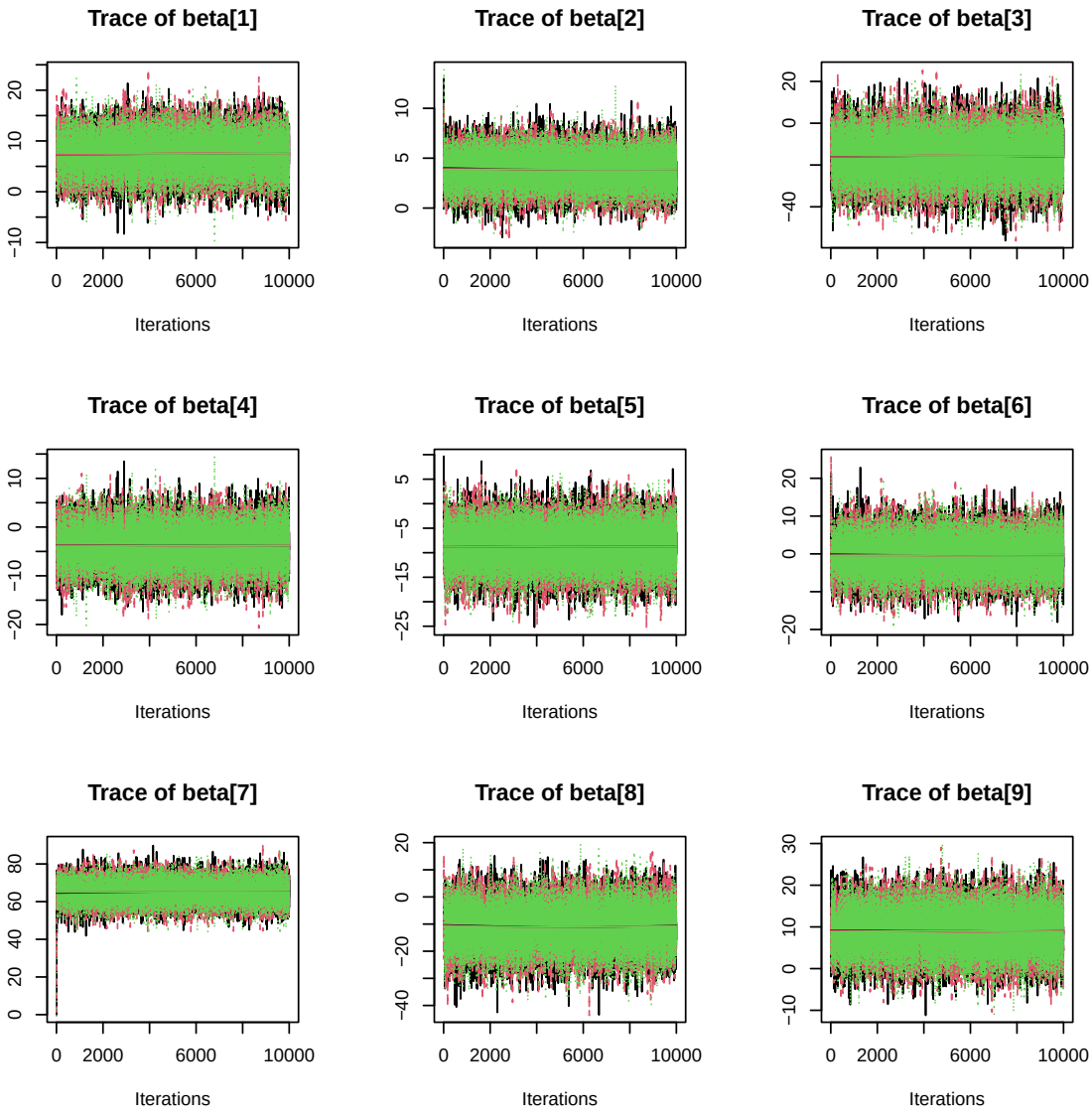
with relatively large starting values for σ^2 (for me, at least) quickly returning to mix among common values for all three chains within a few iterations. Because Stan adapts sampling during its "warm-up" period, we do not want to use the samples for iterations 1 to 5000, despite appearing to converge in much fewer iterations.

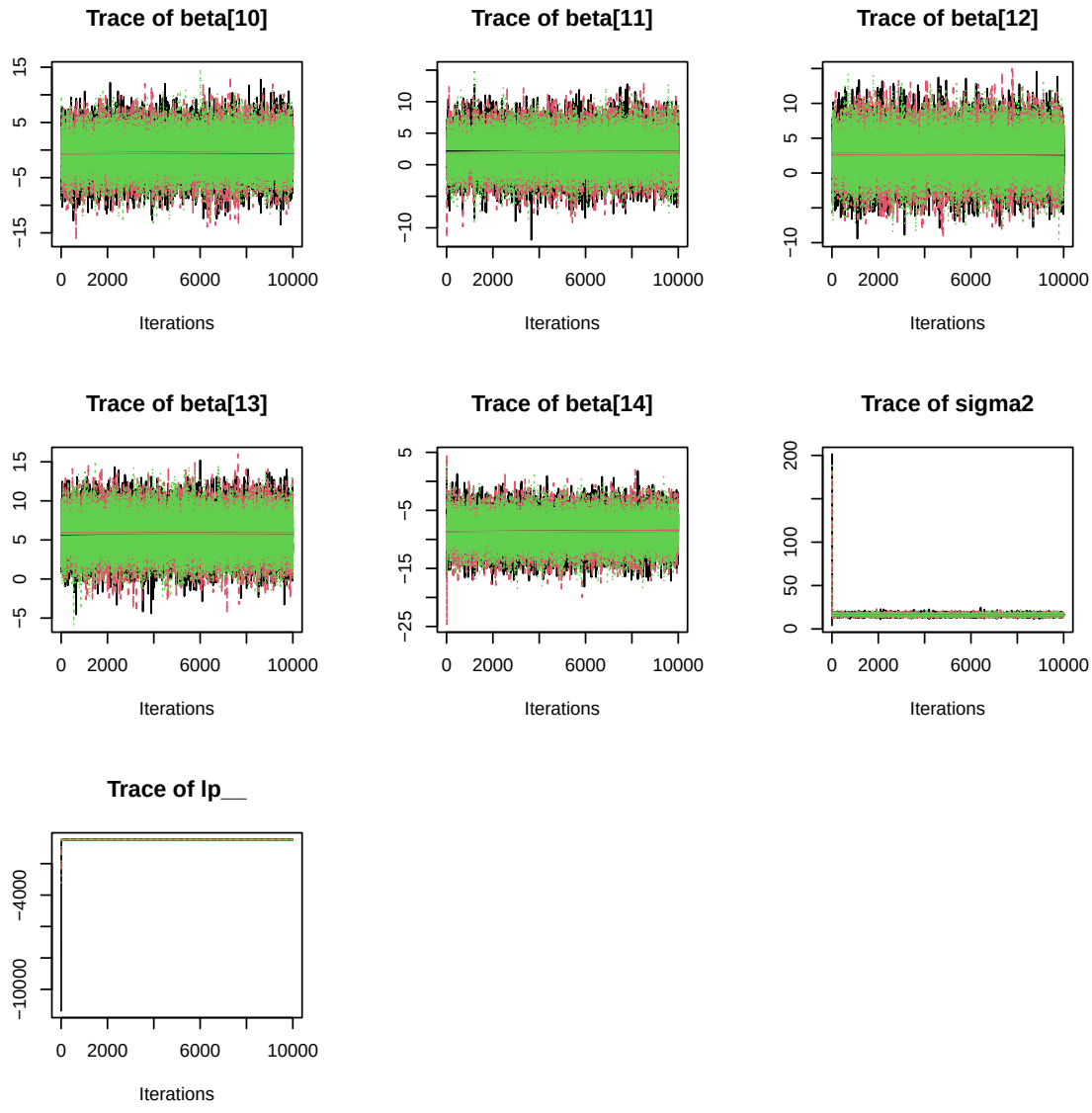
Unlike our Gibbs convergence analysis, all psrf plots and values are all well below the rule-of-thumb values of 1.1 to 1.2, confirming the convergence seen in the history plots. HMC often performs well in this regard compared to other algorithms.

```
> load(file="./zfatStan.fit.RData")
> ## Transform Stan model fit (stanfit class) to a coda mcmc.list object
> ## for use in coda.  rstan::As.mcmc.list() omits warmup!  I want it.
> ## So, I use rstan::extract() instead.  (Note capital A)
> ## zfat.mcmc<- rstan::As.mcmc.list(zfat.fit)
> class(zfat.fit)

[1] "stanfit"
attr(,"package")
[1] "rstan"

> zfat.array<- rstan::extract(zfat.fit, permuted=FALSE,
+                                     inc_warmup=TRUE)
> dim(zfat.array)

[1] 10000     3    16

> nchains<- dim(zfat.array)[2]
> zfat.mcmc.list<- vector("list", nchains)
> for(chain in 1:nchains){
+     zfat.mcmc.list[[chain]]<- coda::as.mcmc(zfat.array[,chain,])
+     }
> zfat.mcmc<- coda::as.mcmc.list(zfat.mcmc.list)
> class(zfat.mcmc)

[1] "mcmc.list"

> coda::nchain(zfat.mcmc)

[1] 3

> coda::niter(zfat.mcmc)
```

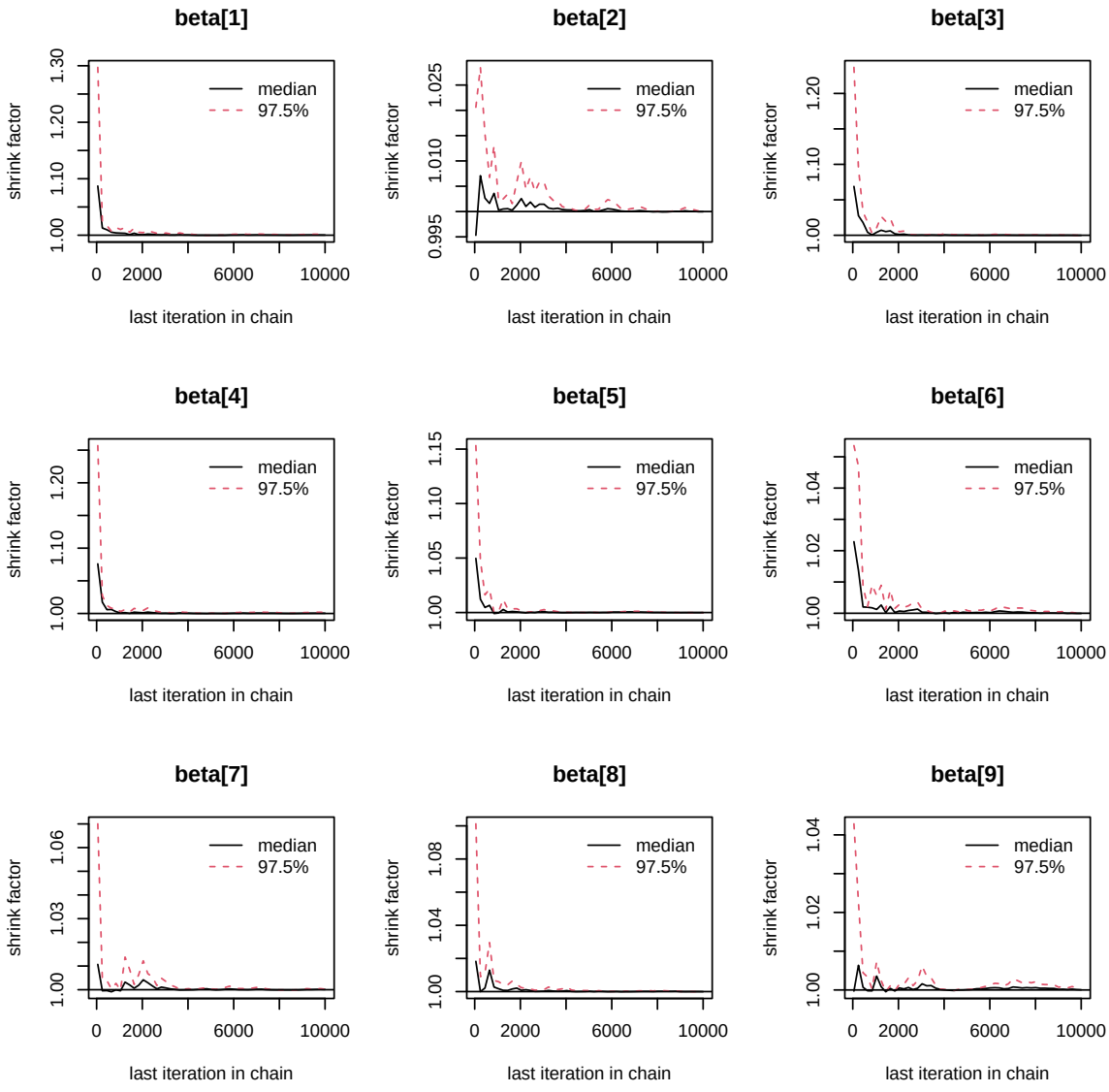
```
[1] 10000

> coda::nvar(zfat.mcmc)

[1] 16

> coda::varnames(zfat.mcmc)

 [1] "beta[1]"  "beta[2]"  "beta[3]"  "beta[4]"  "beta[5]"  "beta[6]"
 [7] "beta[7]"  "beta[8]"  "beta[9]"  "beta[10]" "beta[11]" "beta[12]"
[13] "beta[13]" "beta[14]" "sigma2"   "lp__"
```

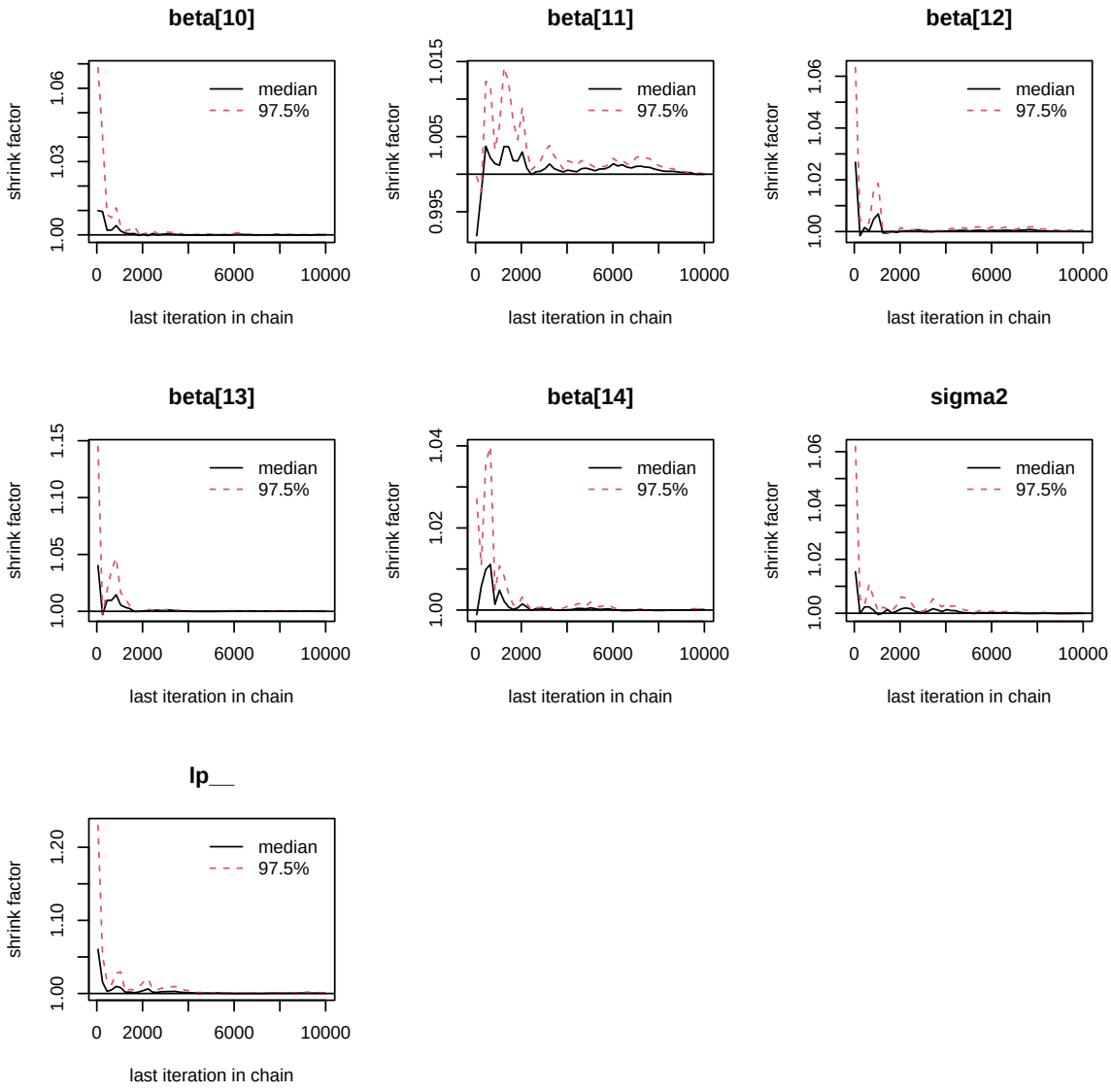
```
> coda:::plot.mcmc.list(zfat.mcmc, density=FALSE)
```


Trace of beta[1]
Trace of beta[2]
Trace of beta[3]
Trace of beta[4]
Trace of beta[5]
Trace of beta[6]
Trace of beta[7]
Trace of beta[8]
Trace of beta[9]
Iterations
0
2000
6000
10000




```
> gelman.plot(zfat.mcmc, ask=FALSE)
```


beta[1]
beta[2]
beta[3]
beta[4]
beta[5]
beta[6]
beta[7]
beta[8]
beta[9]
median
97.5%
shrink factor
last iteration in chain
0
2000
6000
10000




```
> coda::gelman.diag(zfat.mcmc)

Potential scale reduction factors:

         Point est. Upper C.I.
beta[1]           1          1
beta[2]           1          1
beta[3]           1          1
beta[4]           1          1
beta[5]           1          1
```

```
beta[6]            1          1
beta[7]            1          1
beta[8]            1          1
beta[9]            1          1
beta[10]           1          1
beta[11]           1          1
beta[12]           1          1
beta[13]           1          1
beta[14]           1          1
sigma2             1          1
lp__               1          1

Multivariate psrf

1
```

7.4.2 Posterior Summary

(11) Following §C.10.15, report the standard summary of the posterior, omitting the first 5000 iterations from each chain. Just give the code and summary results, do not use all of the code/results shown in §C.10.15. We will compare these results to those of the other analyses herein, later. No need to comment yet.

ANS:

```
> ## Stan adapted for 5000 iterations, so we omit these.
> (zfat.psum<- summary(window(zfat.mcmc, start=5001)))


Iterations = 5001:10000
Thinning interval = 1
Number of chains = 3
Sample size per chain = 5000

1. Empirical mean and standard deviation for each variable,
   plus standard error of the mean:

             Mean    SD Naive SE Time-series SE
beta[1]     7.499  3.80   0.0310         0.0420
beta[2]     3.845  1.72   0.0141         0.0156
beta[3]   -15.746 10.65   0.0870         0.1043
```

```
beta[4]    -3.861  4.09   0.0334         0.0424
beta[5]    -8.744  4.27   0.0349         0.0324
beta[6]    -0.296  5.00   0.0409         0.0430
beta[7]    65.268  5.97   0.0487         0.0521
beta[8]   -10.991  7.81   0.0638         0.0696
beta[9]     8.913  5.23   0.0427         0.0450
beta[10]   -0.468  3.55   0.0290         0.0284
beta[11]    2.036  3.03   0.0247         0.0227
beta[12]    2.743  3.18   0.0259         0.0244
beta[13]    5.844  2.54   0.0207         0.0198
beta[14]   -8.591  2.74   0.0224         0.0222
sigma2     16.028  1.47   0.0120         0.0106
lp__     -479.167  2.83   0.0231         0.0365

2. Quantiles for each variable:

              2.5%        25%      50%     75%   97.5%
beta[1]    -0.0431    4.94430    7.527   10.04   14.95
beta[2]     0.4669    2.68450    3.846    5.01    7.20
beta[3]   -36.6760  -22.75603  -15.620   -8.52    4.93
beta[4]   -11.8267   -6.63108   -3.838   -1.13    4.22
beta[5]   -17.0973  -11.59162   -8.746   -5.90   -0.33
beta[6]    -9.9925   -3.67902   -0.306    3.10    9.52
beta[7]    53.3909   61.31891   65.316   69.31   76.85
beta[8]   -26.3038  -16.27782  -11.007   -5.73    4.46
beta[9]    -1.4009    5.41705    8.926   12.43   19.18
beta[10]   -7.4260   -2.85274   -0.450    1.89    6.44
beta[11]   -3.9192   -0.00887    2.041    4.08    8.00
beta[12]   -3.5075    0.61069    2.767    4.90    8.93
beta[13]    0.8826    4.15633    5.842    7.55   10.82
beta[14]  -13.9577  -10.45476   -8.602   -6.77   -3.15
sigma2     13.3840   14.99838   15.938   16.95   19.19
lp__     -485.5067 -480.83582 -478.822 -477.15 -474.65
```

7.5 Body Fat Summary

(12) Following §C.11, recreate the figure there, now for the three analyses of the body fat data done in this homework. You have to change some indicies in the code in order to create the figure correctly for the 13 effects here, not including the intercept and not including σ^2. Instead of different line types, use solid line colors black, red and green (use the `col` argument with values 1, 2 and 3). Mention two ways that you can see (if

slightly) the effect of the non-informativeness of the improper prior analysis relative to the improper/proper combination prior analyses (refer to features in the plot, of course!).

ANS: You should have changed indices as indicated in the code comments, below.

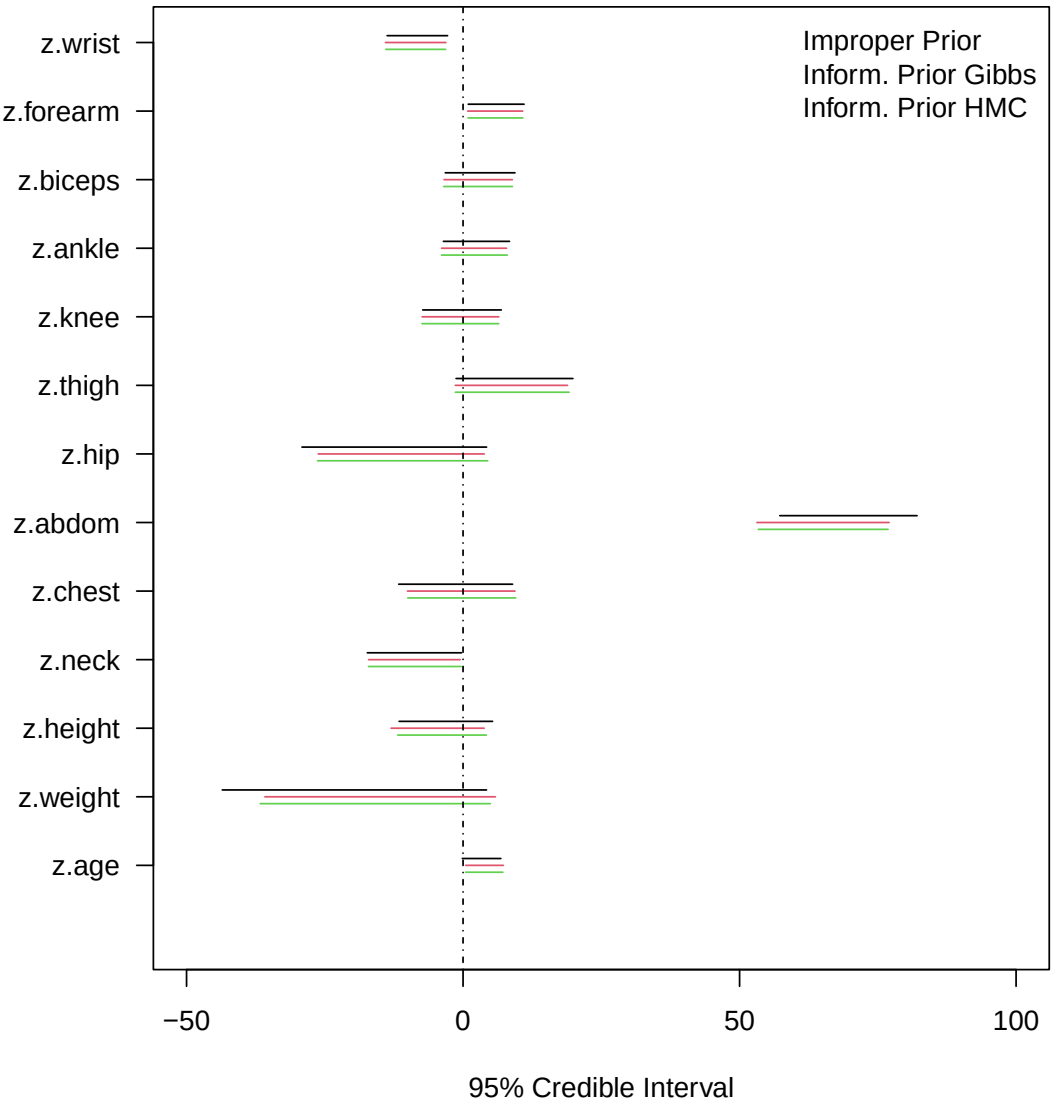
```
> ## Use the previously created objects from our improper prior analysis
> p1lo<- tpost25lb[2:14] ## not 2:8!!!
> p1hi<- tpost975ub[2:14]
>
> ## Omit the initial value and first 5000 samples, i.e., omit
> ## ``burn-in'' or ``warm-up'' iterations from Gibbs:
> p2lo<- bayessum$quantiles[-c(1,15),c("2.5%")]  ## not c(1,10)!!!
> p2hi<- bayessum$quantiles[-c(1,15),c("97.5%")]
>
> ## From Stan HMC (already omitted first 5000 iterations from each of
> ## three chains):
> names(zfat.psum)

[1] "statistics" "quantiles"  "start"      "end"        "thin"
[6] "nchain"

> p3lo<- zfat.psum$quantiles[2:14,c("2.5%")]  ## not 2:9!!!
> p3hi<- zfat.psum$quantiles[2:14,c("97.5%")]
```

Also, to get a reasonable looking plot, you should have changed the `xlim` and the `ylim` arguments of the `plot` function as well as changed the `seq` argument of the `axis` argument and the limits of the `for` loop as indicated in the code, below. (See §C.11 for values used in the original code, of course.)

We see the relative non-informativeness of the improper prior analysis by generally wider intervals (black line segments), more obvious for larger (absolute) effects, e.g., `z.abdom`. Also, when using proper priors for the effects, we see that the effects are shrunk more towards zero, the prior mean, compared to the relatively non-informative improper analysis; the red and green interval lines are closer to zero than the corresponding black interval line, more easily seen for the larger (absolute) effects.

```
> ## Similar to Wakefield's BFRM Fig 5.11:
> par(mar=c(5,7,1,1)+.1)
> plot(p1lo,p1hi,xlim=c(-50,100),type="n",xlab="95% Credible Interval",
+      ylab="",axes="F",ylim=c(0,13))
> box()
```

```
> axis(1)
> axis(2,at=seq(1,13),labels=names(zfat)[-1],las=1)
> for (i in 1:13){ ## not 1:8!
+     lines(y=c(i+.1,i+.1),x=c(p1lo[i],p1hi[i]), col=1)
+     lines(y=c(i,i),x=c(p2lo[i],p2hi[i]),col=2)
+     lines(y=c(i-.1,i-.1),x=c(p3lo[i],p3hi[i]),col=3)

+ }
> abline(v=0,lty=4)
> legend("topright",
+        legend=c("Improper Prior","Inform. Prior Gibbs", "Inform. Prior HMC"),
+        col=1:3,bty="n")
```


z.wrist
z.forearm
z.biceps
z.ankle
z.knee
z.thigh
z.hip
z.abdom
z.chest
z.neck
z.height
z.weight
z.age
Improper Prior
Inform. Prior Gibbs
Inform. Prior HMC
−50
0
50
100
95% Credible Interval