My Final College Paper

A Thesis
Presented to
Department of Statistical Science
Duke University

Huijia Yu

May 2018

Approved for the
Bachelor of Science in Statistical Science

_____
Merlise Clyde

_____
Committeemember O. Name

_____
Committeemember T. Name

_____
Mine Cetinkaya-Rundel, DUS

Acknowledgements

I want to thank a few people.

Preface

This is an example of a thesis setup to use the reed thesis document class (for LaTeX) and the R bookdown package, in general.

Table of Contents

List of Tables

List of Figures

Abstract

The preface pretty much says it all.

Second paragraph of abstract starts here.

Dedication

You can have a dedication here if you wish.

Chapter 1

thesisdowndss::thesis_word: default

Chapter 2

Abstract

Chapter 3

Introduction

Chapter 4

Literature Review

Chapter 5

Simulation Study

5.1 Normal Example

5.1.1 Data Generation

5.1.2 Conditional Likelihood

5.1.3 Posterior Distribution

5.1.4 Results

Estimators

Credible and Confidence Intervals

Coverage

Hypothesis Rejection(??)

Chapter 6

Hierarchical Model Simulation

Chapter 7

Application

```
knitr::opts_chunk$set(echo = TRUE)
knitr::opts_chunk$set(cache = TRUE)

library(rmeta)
```

```
Loading required package: grid
```

```
library(lme4)
```

```
Loading required package: Matrix
```

```
library("R2jags")
```

```
Loading required package: rjags

Loading required package: coda

Linked to JAGS 4.2.0

Loaded modules: basemod,bugs


Attaching package: 'R2jags'

The following object is masked from 'package:coda':

    traceplot
```

```
library(xtable)
```

7.1 Genome Study

- description from previous papers

```
load("tp53.Rdata")
getdata<- function(snp.name, iter=5000, drop= NULL){

 #more iterations, coda traceplots, maybe look at the prior still- cauchy prior? ideal
 #


if(!is.null(drop)) {
  use = !(tp53epi.wsi$site %in% drop)
  } else {
    use = rep(TRUE, nrow(tp53epi.wsi))
  }

  p53.snp = tp53geno.wsi[use, snp.name]
  tp53epi.wsi = tp53epi.wsi[use,]

  missing.geno = is.na(p53.snp)
  site.names = levels(factor(tp53epi.wsi[!missing.geno,"site"]))

  p53.data = list(CaseCon=tp53epi.wsi$casecon[!missing.geno], site=as.numeric(factor(tp5

  p53.df = data.frame(p53.data)

  J = length(p53.data$CaseCon)
  n.sites = length(unique(p53.data$site))
  p53.data$J = J
  p53.data$n.sites = n.sites

  #this is a new indicator
  p53.data$discovery.sites <- which(levels(factor(tp53epi.wsi$site))%in% drop)
  p53.data$missing.geno<-missing.geno

  return(p53.data)

}

snp.name="rs12951053n"
```

```
discovery.sitenames= c("POL", "MAY", "NCO")
#regular data
p53.data<- getdata(snp.name)
#validation data
p53.dataval<- getdata(snp.name, drop=discovery.sitenames)
```

```
Modes <- function(x, min.size) {
  ### Initial Checks
  if(missing(x)) stop("The x argument is required.")
  x <- as.vector(as.numeric(as.character(x)))
  x <- x[is.finite(x)]
  ### Amodal
  if(sd(x)==0)
    return(list(modes=NA, mode.dens=NA, size=1))
  ### Differentiate kernel density by x
  length(density(x)$y)
  dens.y.diff <- density(x)$y[-1] - density(x)$y[-length(density(x)$y)]
  incr <- dens.y.diff
  incr[which(dens.y.diff > 0)] <- 1
  incr[which(dens.y.diff <= 0)] <- 0
  ### Kernel density by increasing/decreasing density regions
  begin <- 1; count <- 1
  for (i in 2:length(incr)) {
    if(incr[i] != incr[i-1]) {
      count <- count + 1
      begin <- c(begin, i)}
  }
  begin <- c(begin, length(incr))
  size <- modes <- mode.dens <- rep(0, count/2)
  init <- 1
  dens <- density(x); sumdens <- sum(dens$y)
  if(incr[1] == 0) {
    size[1] <- sum(dens$y[1:begin[2]]) / sumdens
    init <- 2}
  j <- init
  for (i in init:length(size)) {
    size[i] <- sum(dens$y[begin[j]:begin[j+2]]) / sumdens
    kde <- dens
    kde$x <- kde$x[begin[j]:begin[j+2]]
    kde$y <- kde$y[begin[j]:begin[j+2]]
    modes[i] <- kde$x[kde$y == max(kde$y)]
    mode.dens[i] <- kde$y[kde$y == max(kde$y)]
    j <- j + 2
  }
```

```
  ### Order everything by density
  size <- size[order(mode.dens, decreasing=TRUE)]
  modes <- modes[order(mode.dens, decreasing=TRUE)]
  mode.dens <- mode.dens[order(mode.dens, decreasing=TRUE)]
  ### Remove modes with size < 10%
  if(any(size < min.size)) {
    modes <- modes[-which(size < min.size)]
    mode.dens <- mode.dens[-which(size < min.size)]
    size <- size[-which(size < min.size)]
  }
  if(sum(size) > 1) size <- size / sum(size)
  #Output
  return(list(modes=modes, mode.dens=mode.dens, size=size))
}

is.multimodal <- function(x, min.size=0.01)
{
  if(length(Modes(x, min.size)[[1]]) > 1) return(TRUE)
  else return(FALSE)
}



HPDM <- function(obj, e = 0, prob=0.95, min.size=.01, plot=TRUE){
  vals <- apply(obj, 2, sort)
  if(!is.matrix(vals)) stop("obj must have nsamp > 1.")
  nsamp <- nrow(vals)
  npar <- ncol(vals)
  gap <- max(1, min(nsamp - 1, round(nsamp * prob)))
  init <- 1:(nsamp - gap)
  inds <- apply(vals[init + gap, , drop=FALSE] -
                  vals[init, , drop=FALSE], 2, which.min)
  ansmm <- cbind(vals[cbind(inds, 1:npar)],
                 vals[cbind(inds + gap, 1:npar)])
  dimnames(ansmm) <- list(colnames(obj), c("Lower", "Upper"))

  mm <- apply(obj, 2, is.multimodal, min.size)
  if(any(mm)) {
    cat("\n\nPotentially multimodal column vectors:\n",
        which(mm),"\n")
    vals <- apply(obj, 2, sort)
    if(!is.matrix(vals)) stop("obj must have nsamp > 1.")
    for (m in which(mm)) {
      X<- vals[,m]
```

```
n<- length(X)
zeroes<- which(abs(X)<=e)
if(length(zeroes)==0){
  d<-X
  epsilon= 1e10
}
else{
  d<-X[-zeroes]
  epsilon<- min(abs(X[max(zeroes)+1]),abs(X[min(zeroes)-1]))/20

  }
pi<- length(d)/n
kde <- density(d)
dens <- rbind(data.frame(approx(kde$x, kde$y, d)),
              data.frame(x = rep(0,(n-length(d))),y= rep(0,(n-length(d)))))
dens<-dens[order(dens$x),]
#mix of normals
dens$mix<- pi*dens$y+(1-pi)*dnorm(dens$x, 0, epsilon)/dnorm(0, 0, epsilon)
dens.ind <- dens$mix >= as.vector(quantile(dens$mix,
                                           probs=1-prob)) * 1

ints <- ""
count <- 1
for (i in 1:nrow(vals)) {
  if((i == 1) & (dens.ind[i] == 1)) {
    ints <- paste("(",round(vals[i,m],3),",",sep="")
    if(count > ncol(ansmm)) ansmm <- cbind(ansmm,NA)
    ansmm[m,count] <- vals[i,m]
    count <- count + 1
  }
  if(i > 1) {
    if((dens.ind[i] == 0) & (dens.ind[i-1] == 1)) {
      ints <- paste(ints,round(vals[i-1,m],3),")",sep="")
      if(count > ncol(ansmm)) ansmm <- cbind(ansmm,NA)
      ansmm[m,count] <- vals[i-1,m]
      count <- count + 1
    }
    if((dens.ind[i] == 1) & (dens.ind[i-1] == 0))  {
      ints <- paste(ints," (",round(vals[i,m],3),",",sep="")
      if(count > ncol(ansmm)) ansmm <- cbind(ansmm,NA)
      ansmm[m,count] <- vals[i,m]
      count <- count + 1
    }
  }
```

```
      }
      if((dens.ind[i] == 1) & (dens.ind[i-1] == 1)) {
        ints <- paste(ints,round(vals[i,m],3),")",sep="")
        if(count > ncol(ansmm)) ansmm <- cbind(ansmm,NA)
        ansmm[m,count] <- vals[i,m]
        count <- count + 1
      }
      cat("\nColumn", m, "multimodal intervals:", ints, "\n")
      if(plot){
        #plot(dens$x, dens$mix, type = "l")
        plotvar(X,e)
        points(ansmm, dens$mix[sapply(ansmm, function(a) which(dens$x==a)[1])], col="re
      }

      return(ansmm)
    }
  }
  else{
    return(ansmm)
  }
}

plotvar = function(x, e = 1e-04, nsteps = 500, newplot=TRUE) {
  zeroes = which(abs(x)<e)
  prob0=length(zeroes)/length(x)
  xne0= x
  if(prob0>0){
    xne0=x[-zeroes]
  }
  if(prob0==1){
    xlower = -0
    xupper = 0
    xmax = 1
  }
  m=mean(xne0)
  s= sd(xne0)
  #qmin = min(qnorm(e/2, m, s ))
  #qmax = max(qnorm(1 - e/2, m, s))
  #xlower = min(qmin, 0)
  #xupper = max(0, qmax)
  xlower=min(max(qnorm(e/2, m, s ),min(x)),0)
  xupper=max(min(max(x),qnorm(1 - e/2, m, s)),0)

  xx = seq(xlower, xupper, length.out = nsteps)
```

```
  yy = rep(0, times = length(xx))
  maxyy = 1
  if (prob0 < 1 ) {
    # kdeneg<- density(xne0[xne0<0])
    # kdepos<-density(xne0[xne0>0])
    # yyneg<- approx(kdeneg$x, kdeneg$y, xx)$y*length(xne0[xne0<0])/length(xne0)
    # yyneg[is.na(yyneg)]<-0
    #
    # yypos<- approx(kdepos$x, kdepos$y, xx)$y*length(xne0[xne0>0])/length(xne0)
    # yypos[is.na(yypos)]<-0
    # yy = yyneg+yypos
    kde<- density(xne0)
    yy= approx(kde$x, kde$y, xx)$y

    #yy = dt(x=(x-m)/s, df=)/s
    maxyy = max(yy)
  }

  ymax = max(prob0, 1 - prob0)
  if(newplot){
  plot(c(xlower, xupper), c(0, ymax), type = "n",
       xlab = "", ylab = "")
  }
  lines(c(0, 0), c(0, prob0), lty = 1, lwd = 3,col=as.numeric(newplot)+1)
  lines(xx, (1 - prob0) * yy/maxyy, lty = 1, lwd = 1,col=as.numeric(newplot)+1)
  #invisible()
}
```

7.1.1 EDA

- plot of data points?
- conditional likelihood, FDR, etc as function of Y
-

```
OR.freq = function(snp.name, tp53epi.wsi,tp53geno.wsi, psdir="ps", drop=NULL,iter=

  if (!is.null(drop)) {
    use = !(tp53epi.wsi$site %in% drop)}
  else {
    use = rep(TRUE, nrow(tp53epi.wsi))
  }
```

```
  p53.snp = tp53geno.wsi[use, snp.name]
  tp53epi.wsi = tp53epi.wsi[use,]

  missing.geno = is.na(p53.snp)
  site.names = levels(factor(tp53epi.wsi[!missing.geno,"site"]))

  p53.data = list(CaseCon=tp53epi.wsi$casecon[!missing.geno], site=as.numeric(factor(tp5

  p53.df = data.frame(p53.data)
  write.csv(p53.df, file=paste(snp.name, ".csv", sep=""))

  p53.full = glm(CaseCon ~ factor(site) + factor(site)*p53 + factor(site)*Age + BC, data
 p53.pooled = glm(CaseCon ~ factor(site) + p53 + factor(site)*Age + BC, data=p53.df, fa
   p53.null = glm(CaseCon ~ factor(site) +  factor(site)*Age + BC, data=p53.df, family=b

  p53.df$Site = factor(p53.df$site)

  p53.me = glmer(CaseCon ~ BC + p53 + Age + (1|site)  + (0 + p53 | site) + (0 + Age | si

  test=anova(p53.full, p53.pooled, p53.null, test="Chi")
  coef = summary(p53.full)$coef
  ns = length(site.names)
  OR = coef[c(ns+1, (ns+4):(ns+ns+2)),1]
  x = p53.full$x
  p = predict(p53.full, type="response")
  var = solve(t(x)%*% diag(p*(1-p)) %*%x)[c(ns+1, (ns+4):(ns+ns+2)),c(ns+1, (ns+4):(ns+

  sqrt(diag(var))

  eff = matrix(0, ns,ns)
  eff[,1] = 1
  for (i in 2:ns) eff[i,i] = 1

  OR = eff %*% OR
  OR.SE = sqrt(diag(eff %*% var %*% t(eff)))

  DS = meta.summaries(OR, OR.SE, method="random", names=site.names, logscale=F)
  p.value = pnorm(-(abs(DS$summary/DS$se.summary)))*2
  BF0 = -exp(1)*p.value*log(p.value)
  return(list(snp=snp.name, DS=DS, OR=OR, SE=OR.SE, p.value=p.value, BF.Ha = 1/BF0, test
}
```

```
validation.sitenames = c("AUS" ,"HAW" ,"MAL" ,"NEC" ,"NHS" ,"SEA" ,"STA" ,"UCI","UKO" ,"
```

```
freq<-OR.freq(snp.name, tp53epi.wsi,tp53geno.wsi, psdir="ps", drop=validation.site
freq
```

```
$snp
[1] "rs12951053n"

$DS
Random-effects meta-analysis
Call: meta.summaries(d = OR, se = OR.SE, method = "random", logscale = F,
    names = site.names)
Summary effect=0.303   95% CI (0.075, 0.531)
Estimated heterogeneity variance: 0  p= 0.485

$OR
          [,1]
[1,] 0.2689085
[2,] 0.4317372
[3,] 0.0911083

$SE
[1] 0.2253285 0.1673257 0.2320561

$p.value
[1] 0.009180839

$BF.Ha
[1] 8.542625

$test
Analysis of Deviance Table

Model 1: CaseCon ~ factor(site) + factor(site) * p53 + factor(site) *
    Age + BC
Model 2: CaseCon ~ factor(site) + p53 + factor(site) * Age + BC
Model 3: CaseCon ~ factor(site) + factor(site) * Age + BC
  Resid. Df Resid. Dev Df Deviance Pr(>Chi)
1      2558     2907.6
2      2560     2909.1 -2  -1.4777  0.47766
3      2561     2915.7 -1  -6.6070  0.01016 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

$p53.me
Generalized linear mixed model fit by maximum likelihood (Laplace
```

```
  Approximation) [glmerMod]
 Family: binomial  ( logit )
Formula: CaseCon ~ BC + p53 + Age + (1 | site) + (0 + p53 | site) + (0 +
    Age | site)
   Data: p53.df
      AIC       BIC    logLik  deviance  df.resid
 2936.888  2977.844 -1461.444  2922.888      2561
Random effects:
 Groups Name        Std.Dev.
 site   (Intercept) 0.000e+00
 site.1 p53         1.311e-05
 site.2 Age         7.216e-03
Number of obs: 2568, groups:  site, 3
Fixed Effects:
(Intercept)           BC          p53          Age
   -1.73137      0.44222      0.29692      0.01013
```

- put this into a table
- p values,fdr, BF interpretations

7.1.2 model(s)

- rationale for priors
- plot for mixture
- plots of cond likelihood of mu

```
p53.data.aug<-p53.data
p53.data.aug$zeroes<- rep(0,p53.data$n.sites)
p53.data.aug$zero<- 0

p53.data.normal<-p53.dataval
p53.data.normal$n.discovery<- length(p53.data.normal$discovery.sites)
p53.data.normal$zeroes<- rep(0,p53.data.normal$n.discovery)
p53.data.normal$MLE<- freq$OR[,1]
p53.data.normal$SE<- freq$SE
p = 0.00325
p53.data.normal$q<-qnorm(1-p/2)

bfdata <- p53.dataval
bfdata$p <- freq$p.value
```

```
p53.model = function() {
  for (j in 1:J) {
    CaseCon[j] ~ dbern(theta[j])
```

```
    logit(theta[j]) <- beta.site[site[j]] + beta.p53[site[j]]*p53[j] +
      beta.Age[site[j]]*Age[j] + beta.BC*BC[j]
  }
  for (k in 1:n.sites) {
    beta.site[k] ~ dnorm(mu.site, phi.site)
    beta.p53[k] ~ dnorm(mu.p53, phi.p53)
    beta.Age[k] ~ dnorm(mu.Age, phi.Age)
  }

  beta.BC ~ dnorm(0, 3)

  mu.site ~ dnorm(0, .1)
  phi.site <- pow(sigma.site, -2)
  sigma.site ~ dunif(0,5)

  mu.p53 ~ dnorm(0, .1)
  phi.p53 <- pow(sigma.p53, -2)
  sigma.p53 ~ dunif(0, 5)

  mu.Age ~ dnorm(0, .1)
  phi.Age <- pow(sigma.Age, -2)
  sigma.Age  ~ dunif(0, 5)

}
```

```
#mixture with association variable
p53.newmodel3 = function() {
  for (j in 1:J) {
    CaseCon[j] ~ dbern(theta[j])
    logit(theta[j]) <- beta.site[site[j]] + beta.p53[site[j]]*p53[j] +
      beta.Age[site[j]]*Age[j] + beta.BC*BC[j]    }

  for (l in 1:n.sites) {
    beta.site[l] ~ dnorm(mu.site, phi.site)
    beta.p53.1[l] ~ dnorm(mu.p53, phi.p53)
    beta.p53[l] <- beta.p53.1[l]*assoc
    beta.Age[l] ~ dnorm(mu.Age, phi.Age)
  }
  beta.BC ~ dnorm(0, .1)

  mu.site ~ dnorm(0, .1)
  phi.site ~ dgamma(1,.05)
  sigma.site <- pow(phi.site, -.5)
```

```
  #E[prec] 20
  #(based on range .5 to 2 for OR => range = 1.4 = 6 sigma  sigma = 1.4/6 ~= .2
  mu.p53.1 ~ dnorm(0,.1)
  mu.p53<-mu.p53.1*assoc
  phi.p53 ~ dgamma(1, .05)
  #    phi.p53 ~ dgamma(2, .02)
  sigma.p53 <- pow(phi.p53, -.5)

  mu.Age ~ dnorm(0, .1)
  phi.Age ~ dgamma(1, .05)
  sigma.Age  <- pow(phi.Age, -.5)

  assoc ~ dbern(pind)
  pind ~ dbeta(2,6)
}
```

```
#mixture using zeroes trick (without assoc latent var)
p53.newmodel4 = function() {
  for (j in 1:J) {
    CaseCon[j] ~ dbern(theta[j])
    logit(theta[j]) <- beta.site[site[j]] + beta.p53[site[j]]*p53[j] +
      beta.Age[site[j]]*Age[j] + beta.BC*BC[j]    }

  for (l in 1:n.sites) {
    beta.site[l] ~ dnorm(mu.site, phi.site)
    beta.Age[l] ~ dnorm(mu.Age, phi.Age)
    beta.p53.1[l] ~ dnorm(mu.p53, phi.p53)
    beta.p53[l] <- beta.p53.1[l]*(1-mu.p53.iszero)
  }
  beta.BC ~ dnorm(0, .1)

  mu.site ~ dnorm(0, .1)
  phi.site ~ dgamma(1,.05)
  sigma.site <- pow(phi.site, -.5)

  #zeroes trick
  C<-1000
  epsilon<-0.001
  tau<- pow(epsilon,-2)
  #if mu geq 0 and mu leq 0, prior is from point mass
  #mu in (-epsilon, epsilon)
  mu.p53.iszero<- step(epsilon-abs(mu.p53))
  L<- (mu.p53.iszero*(1-pind))+ #(dnorm(mu.p53,0,tau)*(1-pind))+
    (dnorm(mu.p53,0,1)*pind)
```

```
  #need pind for mixing

  phi<- -log(L)+C
  zero~dpois(phi)
  mu.p53 ~ dunif(-2,2) #not sure how big this interval should be, just picked one

  phi.p53 ~ dgamma(1, .05)
  #    phi.p53 ~ dgamma(2, .02)
  sigma.p53 <- pow(phi.p53, -.5)

  mu.Age ~ dnorm(0, .1)
  phi.Age ~ dgamma(1, .05)
  sigma.Age  <- pow(phi.Age, -.5)
  #assoc~dbern(pind)
  pind ~ dbeta(2,10)
}
```

```
#conditional likelihood prior (normal approx, using zeroes trick)
p53.normal = function() {
  for (j in 1:J) {
    CaseCon[j] ~ dbern(theta[j])
    logit(theta[j]) <- beta.site[site[j]] + mu.p53*p53[j] +
      beta.Age[site[j]]*Age[j] + beta.BC*BC[j]    }

  for (l in 1:n.sites) {
    beta.site[l] ~ dnorm(mu.site, phi.site)
    #beta.p53[l] ~ dnorm(mu.p53, phi.p53)
    beta.Age[l] ~ dnorm(mu.Age, phi.Age)
  }
  C<-1000

  for (k in 1:n.discovery){
    #zeroes trick for MLE~cond prob
    tau[k]<- pow(SE[k], -2)

    L[k]<- dnorm(MLE[k],mu.p53, tau[k])/(pnorm(-q*SE, mu.p53, tau[k]) +
                                          1-pnorm(q*SE, mu.p53, tau[k]))
    phi[k]<- -log(L[k])+C
    zeroes[k]~dpois(phi[k])
  }

  beta.BC ~ dnorm(0, .1)

  mu.site ~ dnorm(0, .1)
```

```
  phi.site ~ dgamma(1,.05)
  sigma.site <- pow(phi.site, -.5)

  #E[prec] 20
  #(based on range .5 to 2 for OR => range = 1.4 = 6 sigma  sigma = 1.4/6 ~= .2
  mu.p53 ~ dnorm(0,.1)
  phi.p53 ~ dgamma(1, .05)
  #    phi.p53 ~ dgamma(2, .02)
  sigma.p53 <- pow(phi.p53, -.5)

  mu.Age ~ dnorm(0, .1)
  phi.Age ~ dgamma(1, .05)
  sigma.Age  <- pow(phi.Age, -.5)
  #assoc~dbern(pind) *assoc
  pind ~ dbeta(2,6)
}
```

```
#eplogp approximation of BF for posterior prob of association
p53.bf.approx = function() {
  for (j in 1:J) {
    CaseCon[j] ~ dbern(theta[j])
    logit(theta[j]) <- beta.site[site[j]] + mu.p53*p53[j] +
      beta.Age[site[j]]*Age[j] + beta.BC*BC[j]    }

  for (l in 1:n.sites) {
    beta.site[l] ~ dnorm(mu.site, phi.site)
    beta.p53.1[l] ~ dnorm(mu.p53, phi.p53)
    #beta.p53[l] <- beta.p53.1[l]*(assoc)
    beta.Age[l] ~ dnorm(mu.Age, phi.Age)
  }
  beta.BC ~ dnorm(0, .1)
  mu.site ~ dnorm(0, .1)
  phi.site ~ dgamma(1,.05)
  sigma.site <- pow(phi.site, -.5)

  #E[prec] 20
  #(based on range .5 to 2 for OR => range = 1.4 = 6 sigma  sigma = 1.4/6 ~= .2
  mu.p53.1 ~ dnorm(0,.1)
  mu.p53<-mu.p53.1*assoc
  phi.p53 ~ dgamma(1, .05)
  #    phi.p53 ~ dgamma(2, .02)
  sigma.p53 <- pow(phi.p53, -.5)

  mu.Age ~ dnorm(0, .1)
```

```
  phi.Age ~ dgamma(1, .05)
  sigma.Age  <- pow(phi.Age, -.5)

  assoc~dbern(post.ind)
  prior.odds<- (1-pind)/pind
  BF<- -exp(1)*p*log(p)
  post.ind<-prior.odds*BF/(1+prior.odds*BF)
  pind ~ dbeta(2,6)
}
```

```
parameters = c("mu1.site", "mu1.p53",  "beta.BC", "beta.Age", "mu.site","mu.p53",

parameters4 = c("beta.BC", "beta.Age", "mu.site","mu.p53", "mu.Age", "sigma.site",

parameters.normal<- c("beta.BC", "beta.Age", "mu.site","mu.p53", "mu.Age", "sigma.

p53.sim = jags(data=p53.data, inits=NULL, parameters.to.save =parameters, model =
```

```
module glm loaded

Warning in jags.model(model.file, data = data, inits = init.values,
n.chains = n.chains, : Unused variable "discovery.sites" in data

Warning in jags.model(model.file, data = data, inits = init.values,
n.chains = n.chains, : Unused variable "missing.geno" in data

Compiling model graph
   Resolving undeclared variables
   Allocating nodes
Graph information:
   Observed stochastic nodes: 11182
   Unobserved stochastic nodes: 46
   Total graph size: 59386

Initializing model

Warning in jags.samples(model, variable.names, n.iter, thin, type = "trace", : Faile
Variable mu1.site not found

Warning in jags.samples(model, variable.names, n.iter, thin, type = "trace", : Faile
Variable mu1.p53 not found

Warning in jags.samples(model, variable.names, n.iter, thin, type = "trace", : Faile
Variable assoc not found
```

```
Warning in jags.samples(model, variable.names, n.iter, thin, type = "trace", : Failed to s
Variable pind not found

Warning in jags.samples(model, variable.names, n.iter, thin, type = "trace", : Failed to s
Variable phi1.site not found

Warning in jags.samples(model, variable.names, n.iter, thin, type = "trace", : Failed to s
Variable phi1.p53 not found
```

```
p53.simval = jags(data=p53.dataval, inits=NULL, parameters.to.save =parameters, model =
```

```
Warning in jags.model(model.file, data = data, inits = init.values,
n.chains = n.chains, : Unused variable "discovery.sites" in data

Warning in jags.model(model.file, data = data, inits = init.values,
n.chains = n.chains, : Unused variable "missing.geno" in data

Compiling model graph
   Resolving undeclared variables
   Allocating nodes
Graph information:
   Observed stochastic nodes: 8614
   Unobserved stochastic nodes: 37
   Total graph size: 45664

Initializing model

Warning in jags.samples(model, variable.names, n.iter, thin, type = "trace", : Failed to s
Variable mu1.site not found

Warning in jags.samples(model, variable.names, n.iter, thin, type = "trace", : Failed to s
Variable mu1.p53 not found

Warning in jags.samples(model, variable.names, n.iter, thin, type = "trace", : Failed to s
Variable assoc not found

Warning in jags.samples(model, variable.names, n.iter, thin, type = "trace", : Failed to s
Variable pind not found

Warning in jags.samples(model, variable.names, n.iter, thin, type = "trace", : Failed to s
Variable phi1.site not found

Warning in jags.samples(model, variable.names, n.iter, thin, type = "trace", : Failed to s
Variable phi1.p53 not found
```

```
p53.simnew4 = jags(data=p53.data.aug, inits=NULL, parameters.to.save =parameters4,
```

```
Warning in jags.model(model.file, data = data, inits = init.values,
n.chains = n.chains, : Unused variable "discovery.sites" in data

Warning in jags.model(model.file, data = data, inits = init.values,
n.chains = n.chains, : Unused variable "missing.geno" in data

Warning in jags.model(model.file, data = data, inits = init.values,
n.chains = n.chains, : Unused variable "zeroes" in data

Compiling model graph
   Resolving undeclared variables
   Allocating nodes
Graph information:
   Observed stochastic nodes: 11183
   Unobserved stochastic nodes: 47
   Total graph size: 59459

Initializing model

Warning in jags.samples(model, variable.names, n.iter, thin, type = "trace", : Faile
Variable assoc not found

Warning in jags.samples(model, variable.names, n.iter, thin, type = "trace", : Faile
Variable beta.p53.iszero not found
```

```
p53.simnormal = jags(data=p53.data.normal, inits=NULL, parameters.to.save =paramet
```

```
module glm loaded

Warning in jags.model(model.file, data = data, inits = init.values,
n.chains = n.chains, : Unused variable "discovery.sites" in data

Warning in jags.model(model.file, data = data, inits = init.values,
n.chains = n.chains, : Unused variable "missing.geno" in data

Compiling model graph
   Resolving undeclared variables
   Allocating nodes
Graph information:
   Observed stochastic nodes: 8614
   Unobserved stochastic nodes: 28
   Total graph size: 45642

Initializing model
```

```
Warning in jags.samples(model, variable.names, n.iter, thin, type = "trace", : Failed to s
Variable beta.p53 not found

Warning in jags.samples(model, variable.names, n.iter, thin, type = "trace", : Failed to s
Variable assoc not found
```

```
p53.bfsim = jags(data=bfdata, inits=NULL, parameters.to.save =parameters, model = p53.bf
```

```
Warning in jags.model(model.file, data = data, inits = init.values,
n.chains = n.chains, : Unused variable "discovery.sites" in data

Warning in jags.model(model.file, data = data, inits = init.values,
n.chains = n.chains, : Unused variable "missing.geno" in data

Compiling model graph
   Resolving undeclared variables
   Allocating nodes
Graph information:
   Observed stochastic nodes: 8614
   Unobserved stochastic nodes: 39
   Total graph size: 45659

Initializing model

Warning in jags.samples(model, variable.names, n.iter, thin, type = "trace", : Failed to s
Variable mu1.site not found

Warning in jags.samples(model, variable.names, n.iter, thin, type = "trace", : Failed to s
Variable mu1.p53 not found

Warning in jags.samples(model, variable.names, n.iter, thin, type = "trace", : Failed to s
Variable phi1.site not found

Warning in jags.samples(model, variable.names, n.iter, thin, type = "trace", : Failed to s
Variable phi1.p53 not found

Warning in jags.samples(model, variable.names, n.iter, thin, type = "trace", : Failed to s
Variable beta.p53 not found
```

```
p53.simnew3 = jags(data=p53.data, inits=NULL, parameters.to.save =parameters, model = p5
```

```
module glm loaded

Warning in jags.model(model.file, data = data, inits = init.values,
n.chains = n.chains, : Unused variable "discovery.sites" in data
```

```
Warning in jags.model(model.file, data = data, inits = init.values,
n.chains = n.chains, : Unused variable "missing.geno" in data

Compiling model graph
   Resolving undeclared variables
   Allocating nodes
Graph information:
   Observed stochastic nodes: 11182
   Unobserved stochastic nodes: 48
   Total graph size: 59407

Initializing model

Warning in jags.samples(model, variable.names, n.iter, thin, type = "trace", : Faile
Variable mu1.site not found

Warning in jags.samples(model, variable.names, n.iter, thin, type = "trace", : Faile
Variable mu1.p53 not found

Warning in jags.samples(model, variable.names, n.iter, thin, type = "trace", : Faile
Variable phi1.site not found

Warning in jags.samples(model, variable.names, n.iter, thin, type = "trace", : Faile
Variable phi1.p53 not found
```

```r
library(dplyr)
getOR<-function(sim){

OR = as.data.frame(sim$BUGSoutput$sims.matrix)%>%select( starts_with("beta.p53"),
exp(OR)
#colnames(OR) = c(site.names, "Overall")

sum.OR = t(apply(exp(OR), 2, function(x) {PI = HPDinterval(as.mcmc(x))
 return(c(median(x), PI[1], PI[2]))}
))
 return(sum.OR)
}

OR1<-getOR(p53.sim) #regular
OR3<-getOR(p53.simnew3) #w assoc
OR4<-getOR(p53.simnew4) #w point mass ind
ORn<-getOR(p53.simnormal) #MLEs for prior
ORbf<-getOR(p53.bfsim) #MLEs for prior
```

```
ORtable = data.frame(original= OR1["mu.p53",], latentvar= OR3["mu.p53",],
                 zerotrick= OR4["mu.p53",] , cond= ORn["mu.p53",], bfapprox= ORbf["mu.p5
rownames(ORtable)<- c("Median", "2.5%","97.5%")
kable(ORtable)
```

	original	latentvar	zerotrick	cond	bfapprox
Median	1.183197	1.000000	1.1849025	1.159779	1.000000
2.5%	1.042845	1.000000	0.9963451	1.011020	1.000000
97.5%	1.355430	1.246941	1.3741023	1.324258	1.154497

- discussion of OR
- at least one example of a posterior?
- plot CI comparisons
- plot something like the shrinkage
- plot CI length comparison
- some notes on disjoint CI's, log vs not logged HPD

References