Hierarchical Modeling of Multiple Exposures

Alpine Exposome Summer School 2021

Patrick T. Bradshaw, PhD

June 2021

Consider that we have data on 10 environmental chemical exposures (contained in the matrix `x`) that we would like to investigate for their association with breast cancer risk (in the variable `y`). We also have a covariate `age` (which we will denote with w) that we would like to additionally adjust for.

Load packages and read data

```
library(MASS)
library(R2jags)
library(coda)
library(knitr)

load("HLMData.Rdata")
```

We want to estimate the log-odds ratios (OR) for the exposure-breast cancer risk. The primary model of interest (our Stage 1 model) is:

$$\text{logit}(\Pr[Y = 1|\mathbf{x}, w, \alpha, \beta, \gamma]) = \alpha + \mathbf{x}\beta + w\gamma \quad (1)$$

where α is the intercept term, β is the vector of log-ORs for the 10 chemical exposures, and γ is the log-OR per year increase in age (which has been scaled to zero mean and unit variance).

Single exposure model

We start by estimating single exposure models (10 separate models, one per exposure), and accumulate the results:[1]

```
Nx <- 10 # We have 10 exposures
beta.serial <- se.serial <- rep(NA, Nx) # Initialize a vector for the results

# Run separate model for each exposure,
# and collect beta coefficient (2nd one in each model)
for (i in 1:Nx){
```

[1] We would ideally do these in a fully Bayesian setting (with JAGS) to compare to the Bayesian hierarchical model, but this is a bit more practical.

```
     fit.serial <- glm(y~x[,i] + age, family=binomial)
     beta.serial[i] <- coef(fit.serial)[2]
     se.serial[i] <- sqrt(vcov(fit.serial)[2,2])
}
```

Naïve multiple exposure models

We know that the effects of these chemicals may confound each other, so a model that incorporates all of the exposures simultaneously should improve our inference. We then instead estimate a model with all 10 exposures included at the same time:

```
fit.naive <- glm(y~x + age, family=binomial)
# only keep beta coefficients (omit intercept, age)
beta.naive <- coef(fit.naive)[2:(Nx+1)]
se.naive <- sqrt(diag(vcov(fit.naive))[2:(Nx+1)])
```

Bayesian Hierarchical Model

We would like to estimate a model that included all of these exposures simultaneously, but we are concerned that some of these chemical exposures may be highly correlated, which can yield instability in our regression model:

```
round(cor(x),2)
```

```
##       [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
##  [1,] 1.00 0.94 0.79 0.54 0.42 0.31 0.30 0.22 0.10  0.02
##  [2,] 0.94 1.00 0.93 0.75 0.60 0.42 0.43 0.37 0.27  0.22
##  [3,] 0.79 0.93 1.00 0.88 0.73 0.57 0.61 0.53 0.45  0.45
##  [4,] 0.54 0.75 0.88 1.00 0.94 0.80 0.82 0.80 0.72  0.74
##  [5,] 0.42 0.60 0.73 0.94 1.00 0.90 0.89 0.85 0.81  0.82
##  [6,] 0.31 0.42 0.57 0.80 0.90 1.00 0.94 0.89 0.83  0.81
##  [7,] 0.30 0.43 0.61 0.82 0.89 0.94 1.00 0.97 0.96  0.92
##  [8,] 0.22 0.37 0.53 0.80 0.85 0.89 0.97 1.00 0.94  0.91
##  [9,] 0.10 0.27 0.45 0.72 0.81 0.83 0.96 0.94 1.00  0.95
## [10,] 0.02 0.22 0.45 0.74 0.82 0.81 0.92 0.91 0.95  1.00
```

Some are indeed highly correlated (note a number of ρs >0.7) We can use a Bayesian Hierarchical model to possibly reign in some of this instability. We believe that several of these chemicals may be related through possible metabolic pathways, and we can use this to group chemicals that might have similar effects. The first 4 exposures (x1-x4) operate in pathway 1 (but not 2), the second 4 (x5-x8) operate in pathway 2 (but not 1), and the last 2 (x9-x10) operate in both. We can encode a predictor matrix `Z` to indicate this:

```
# 1st 4 exposures in class 1, 2nd 4 in class 2, last 2 in both
Z <- cbind(c(rep(1,4),rep(0,4), rep(1,2)),
           c(rep(0,4),rep(1,4), rep(1,2)))
colnames(Z) <- c("Path 1","Path 2")
rownames(Z) <- paste("x",1:Nx, sep="")
Z
```

```
##     Path 1 Path 2
## x1       1      0
## x2       1      0
## x3       1      0
## x4       1      0
## x5       0      1
## x6       0      1
## x7       0      1
## x8       0      1
## x9       1      1
## x10      1      1
```

The expression of the hierarchical model follows from equation (1), but also incorporates a model (prior) for the coefficients as a function of their pathway effect:

$$\begin{aligned} \beta_j &= \mathbf{z}_j \pi + \delta_j \\ \delta_j &\sim N(0, \tau) \end{aligned} \tag{2}$$

where $\pi = (\pi_1, \pi_2)$ is the vector of pathway effects, and the $\delta = (\delta_1, \delta_2, \ldots, \delta_{10})$ are the random (residual) effects of each chemical, independent of their pathway effects.

In a Bayesian framework, we can specify non-informative (vague) priors for α,γ, and π. We will specify the prior precision τ to reflect a 95% *prior* probability belief that the OR lies in a 4-fold range of its point estimate ($OR_{\text{upper}}/OR_{\text{lower}}$=4).

Fitting the 2-stage hierarchical model in a fully Bayseian framework using JAGS:

```
# Define the function for the prior:
hierarchical.model <- function(){
     # First stage model:
     for (i in 1:N) {
          logit(p.y[i]) <- a + inprod(x[i,1:Nx], b[1:Nx]) + age[i]*g;
          y[i] ~ dbin(p.y[i], 1);
     }

     # Second stage model:
     for (j in (1:Nx)) {
          b[j] <- inprod(z[j,1:Nz], pi[1:Nz]) + delta[j];
          delta[j] ~ dnorm(0, tau.b);
          OR[j] <- exp(b[j]); # Calculate ORs for reporting
     }

     # Priors on fixed parameters
     # First stage model:
     a ~ dnorm(0,0.001); # 1st stage intercept
     g ~ dnorm(0,0.001); # 1st stage age effect

     # Second-stage model
     for (k in 1:Nz) { pi[k] ~ dnorm(0, 0.001);}
}
```

Obtain data and constants for JAGS:

```
N <- length(y) # Number of observations
Nx <- ncol(x)
Nz <- ncol(Z)

range <- log(2) - log(.5)
tau.b <- (range/4)^(-2)

# Data, parameter list and starting values
data <- list(N=N, Nx=Nx, Nz=Nz, x=x, y=y, age=age, z=Z, tau.b=tau.b)
parameters<-c("a", "b", "g","pi","OR") # Parameters to keep track of

# Sample from posterior with JAGS:
bhm <- jags(model.file=hierarchical.model, data=data,
             parameters=parameters, n.thin=5,
            jags.seed=1234, n.iter=20000)
```

```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 500
##    Unobserved stochastic nodes: 14
##    Total graph size: 8576
##
## Initializing model
```

```
print(bhm, digits=3)
```

```
## Inference for Bugs model at "/var/folders/km/6p094blx59g2pvtt6my6_4640000gn/T//RtmpXqrled/mod
##  3 chains, each with 20000 iterations (first 10000 discarded), n.thin = 5
##  n.sims = 6000 iterations saved
##          mu.vect sd.vect   2.5%    25%    50%    75%  97.5%  Rhat n.eff
## OR[1]      0.547   0.105  0.371  0.474  0.537  0.610  0.776 1.002  1800
## OR[2]      0.752   0.189  0.445  0.619  0.730  0.863  1.199 1.002  1600
## OR[3]      0.313   0.067  0.201  0.265  0.308  0.353  0.460 1.001  6000
## OR[4]      1.473   0.408  0.850  1.182  1.412  1.704  2.403 1.001  6000
## OR[5]      5.925   1.702  3.303  4.721  5.682  6.864  9.903 1.001  6000
## OR[6]      2.019   0.473  1.253  1.685  1.961  2.304  3.099 1.001  5900
## OR[7]      1.566   0.483  0.833  1.229  1.496  1.823  2.684 1.001  6000
## OR[8]      4.669   1.214  2.749  3.809  4.504  5.368  7.424 1.001  4700
## OR[9]      0.747   0.173  0.462  0.622  0.726  0.853  1.134 1.001  2700
## OR[10]     0.826   0.192  0.516  0.690  0.805  0.935  1.262 1.002  2500
## a         -0.728   0.234 -1.189 -0.891 -0.728 -0.570 -0.265 1.001  3700
## b[1]      -0.621   0.190 -0.992 -0.747 -0.621 -0.495 -0.254 1.002  1800
## b[2]      -0.316   0.249 -0.809 -0.479  -0.315  -0.148   0.182 1.002  1600
## b[3]      -1.186   0.213 -1.607 -1.328  -1.177  -1.042  -0.777 1.001  6000
## b[4]       0.351   0.269 -0.162  0.167   0.345   0.533   0.877 1.001  6000
## b[5]       1.740   0.281  1.195  1.552   1.737   1.926   2.293 1.001  6000
## b[6]       0.676   0.231  0.226  0.522   0.673   0.835   1.131 1.001  5900
## b[7]       0.403   0.301 -0.183  0.206   0.403   0.601   0.987 1.001  6000
## b[8]       1.508   0.256  1.011  1.337   1.505   1.680   2.005 1.001  5800
## b[9]      -0.318   0.229 -0.773 -0.474  -0.320  -0.159   0.126 1.001  2700
## b[10]     -0.217   0.229 -0.662 -0.371  -0.217  -0.067   0.232 1.002  2500
## g         -0.043   0.218 -0.471 -0.187  -0.043   0.104   0.382 1.001  6000
## pi[1]     -0.667   0.185 -1.029 -0.793  -0.668  -0.540  -0.312 1.001  6000
## pi[2]      0.857   0.187  0.492  0.730   0.859   0.981   1.223 1.002  1500
## deviance 104.111   9.346 91.269 98.765 103.462 108.306 120.613 1.001  3200
##
## For each parameter, n.eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor (at convergence, Rhat=1).
##
## DIC info (using the rule, pD = var(deviance)/2)
## pD = 43.7 and DIC = 147.8
## DIC is an estimate of expected predictive error (lower deviance is better).
```

```
bhm.mcmc <- as.mcmc(bhm) # convert to MCMC object for CODA processing
summ.bhm <- summary(bhm.mcmc)
# plot(bhm.mcmc) # Could check convergence

to.keep <- paste("b[",1:Nx,"]",sep="") # names of parameters to keep from summary
beta.hm <- summ.bhm$quantiles[to.keep,3] # posterior median for point estimate
sd.hm <- summ.bhm$statistics[to.keep,2] # posterior standard deviation
```

Comparison

We can compare the results (using log-ORs) from these models:

```
kable(round(cbind(beta.serial, se.serial,
            beta.naive, se.naive,
            beta.hm, sd.hm),2),
      col.names = c("beta serial", "SE serial",
                   "beta naive","se naive",
                   "beta BHM","sd BHM"))
```

	beta serial	SE serial	beta naive	se naive	beta BHM	sd BHM
x1	-0.14	0.03	0.61	1.64	-0.62	0.19
x2	-0.07	0.02	-4.07	2.93	-0.31	0.25
x3	0.01	0.02	-1.26	2.02	-1.18	0.21
x4	0.26	0.03	1.31	1.23	0.34	0.27
x5	0.52	0.05	5.36	1.50	1.74	0.28
x6	0.68	0.06	1.60	1.11	0.67	0.23
x7	0.62	0.06	-4.54	5.26	0.40	0.30

	beta serial	SE serial	beta naive	se naive	beta BHM	sd BHM
x8	0.69	0.06	5.83	1.90	1.50	0.26
x9	0.72	0.07	1.70	2.66	-0.32	0.23
x10	0.88	0.08	-1.87	0.92	-0.22	0.23

Notice that the coefficients from the serial (one-exposure-at-a-time) models are small, but very precise. However, they are likely confounded by failure to account for other correlated exposures. The coefficients from the naïve multi-exposure model are much more extreme, and their standard errors are MUCH larger (confidence interval width will be correspondingly imprecise). The Bayesian Hierarchical Modeling (BHM) results are generally more attenuated than the naïve multi-exposure model, with much more reasonable standard errors. Notice that coefficients from exposures in the same pathways are shrunk closer to the average effect in that pathway.

BONUS: Penalized ML

While Bayesian inference *via* MCMC is handy (and very flexible), we can fit an equivalent model using ridge (quadratic) penalized maximum likelihood, which is more computationally efficient.

We shrink the β coefficients to their expected value from the 2nd stage model: $\mathbf{z}\pi$ with strength proportional to the precision τ. To see why this is equivalent, note that the Stage 2 model for β (equation (2)) is equivalent to:

$$\beta_j \sim N(\mathbf{z}_j\pi, \tau)$$

where τ is precision (not variance) common to all j. This implies the following density function for $\beta = (\beta_1, \dots, \beta_{10})$:

$$p(\beta) = \prod_{j=1}^{10} \sqrt{\frac{\tau}{2\pi}} \exp\left(-\frac{\tau}{2}(\beta_j - \mathbf{z}_j\pi)^2\right)$$

Ignoring a distribution on π for now, and remembering that we set τ to a particular value the posterior (likelihood $\times$ prior) is then:

$$\begin{aligned} p(\beta, \alpha, \gamma | \mathbf{y}, \mathbf{x}, w, \mathbf{z}) &= L(\beta, \alpha, \gamma | \mathbf{y}, \mathbf{x}, w) \times \prod_{j=1}^{10} \sqrt{\frac{\tau}{2\pi}} \exp\left(-\frac{\tau}{2}(\beta_j - \mathbf{z}_j\pi)^2\right) \\ &\propto L(\beta, \alpha, \gamma | \mathbf{y}, \mathbf{x}, w) \times \prod_{j=1}^{10} \exp\left(-\frac{\tau}{2}(\beta_j - \mathbf{z}_j\pi)^2\right) \end{aligned}$$

where we can ignore the constant piece of the prior (and likelihood for that matter). Taking natural log of both sides, yields the log-posterior:

$$\log\left[p(\beta, \alpha, \gamma | \mathbf{y}, \mathbf{x}, w, \mathbf{z})\right] = \mathcal{L}(\beta, \alpha, \gamma | \mathbf{y}, \mathbf{x}, \mathbf{age}) \overbrace{-\frac{\tau}{2}\sum_{j=1}^{10}(\beta_j - \mathbf{z}_j\pi)^2}^{\text{Penalty function}} + C$$

where C is the stuff that does not depend on the parameters we want to estimate, therefore we can ignore.

You may recognize this as a penalized likelihood function from our penalized MLE lecture in PB HLTH 252, but with $\mathbf{m}$ replaced by $\mathbf{z}\pi$! Many Bayesian models can be expressed as an equivalent penalized likelihood problem (and vice-versa).[2] We can then estimate β (and π) by the method of penalized maximum likelihood. The point estimates from this will correspond to the posterior *mode* (rather than median, although this difference is negligible for our purposes).

```r
library(stats4)
expit <- function(x) exp(x)/(1+exp(x))

# MLE command in R requires negative of log-likelihood:
nPLL <- function(a, b1, b2, b3, b4, b5, b6, b7, b8, b9, b10, g,
                 pi1, pi2){
  # concatenate betas together for vector multiplication
  b <- c(b1, b2, b3, b4, b5, b6, b7, b8, b9, b10)

  # concatenate pis together
  pi <- c(pi1, pi2)

  p.y <- expit(a + x %*% b + age*g) # Probability of event
  nLL <- -sum(dbinom(y, 1, p.y, log=TRUE)) # Negative of log-likelihood
  penalty <- (tau.b/2)*t(b - Z %*% pi) %*% (b - Z %*% pi); # Negative of penalty
  return(nLL + penalty) # Return negative penalized log-likelihood
  # return(nLL)
}

pmle <- mle(nPLL, start=list(a=0, b1=0, b2=0, b3=0, b4=0, b5=0, b6=0, b7=0, b8=0,
                              b9=0, b10=0, g=0, pi1=0, pi2=0), nobs=NROW(y))

summary(pmle)
```

```
## Maximum likelihood estimation
##
## Call:
## mle(minuslogl = nPLL, start = list(a = 0, b1 = 0, b2 = 0, b3 = 0,
##     b4 = 0, b5 = 0, b6 = 0, b7 = 0, b8 = 0, b9 = 0, b10 = 0,
##     g = 0, pi1 = 0, pi2 = 0), nobs = NROW(y))
##
## Coefficients:
##        Estimate Std. Error
## a   -0.69509044  0.2294112
## b1  -0.59110209  0.1881134
## b2  -0.29328578  0.2489778
## b3  -1.13506695  0.2132243
## b4   0.33950399  0.2663314
## b5   1.65409053  0.2725841
## b6   0.63445134  0.2239666
```

[2]See: Greenland, Sander, and Mohammad Ali Mansournia. "Penalization, bias reduction, and default priors in logistic and related categorical and survival regressions." *Statistics in Medicine*. 34.23 (2015): 3133-3143.

```
## b7    0.37773179  0.3006434
## b8    1.44779478  0.2497413
## b9   -0.31148353  0.2304811
## b10  -0.21628062  0.2271893
## g    -0.03885765  0.2087147
## pi1  -0.63808857  0.1812224
## pi2   0.81041106  0.1827356
##
## -2 log L: 123.4075
```

```
# The "paste" command creates a string vector
# of element names to keep.
beta.pmle <- coef(pmle)[paste("b",1:Nx,sep="")]
se.pmle <- sqrt(diag(vcov(pmle))[paste("b",1:Nx,sep="")])
```

Comparing all results:

```
kable(round(cbind(beta.serial, se.serial,
            beta.naive, se.naive,
            beta.hm, sd.hm,
            beta.pmle, se.pmle),2),
      col.names = c("beta serial", "SE serial",
                    "beta naive","se naive",
                    "beta BHM","sd BHM",
                    "beta PMLE","se PMLE"))
```

	beta serial	SE serial	beta naive	se naive	beta BHM	sd BHM	beta PMLE	se PMLE
x1	-0.14	0.03	0.61	1.64	-0.62	0.19	-0.59	0.19
x2	-0.07	0.02	-4.07	2.93	-0.31	0.25	-0.29	0.25
x3	0.01	0.02	-1.26	2.02	-1.18	0.21	-1.14	0.21
x4	0.26	0.03	1.31	1.23	0.34	0.27	0.34	0.27
x5	0.52	0.05	5.36	1.50	1.74	0.28	1.65	0.27
x6	0.68	0.06	1.60	1.11	0.67	0.23	0.63	0.22
x7	0.62	0.06	-4.54	5.26	0.40	0.30	0.38	0.30
x8	0.69	0.06	5.83	1.90	1.50	0.26	1.45	0.25
x9	0.72	0.07	1.70	2.66	-0.32	0.23	-0.31	0.23
x10	0.88	0.08	-1.87	0.92	-0.22	0.23	-0.22	0.23

The coefficient estimates and standard errors from the penalized ML procedure (PMLE) are very close to those from the Bayesian Hierarchical model (BHM). Slight differences are due to the use of MCMC in the Bayesian setting, and perhaps some slight skewness in the posterior distributions. Nevertheless, they are both essentially equivalent for problems such as this.