KU Leuven Summer School
Segment 3B
More Misclassification Models

Paul Gustafson

September 15, 2022

Start with a mystery dataset

Case-control again, but:

- Nobody has a measurement of true exposure X
- Everbody has a pair of measurements from two **different** surrogates (for X), X_1^*, X_2^*

E.g., think $X_1^* \sim$ self-report, $X_2^* \sim$ imperfect lab test

```
head(dta)
```

```
##   xstr1 xstr2 y
## 1     0     0 0
## 2     0     0 0
## 3     0     0 1
## 4     0     0 0
## 5     1     0 1
## 6     0     0 1
```

Mystery dataset, continued

```
table(dta)
```

```
## , , y = 0
##
##      xstr2
## xstr1    0    1
##     0 1839  134
##     1  404  123
##
## , , y = 1
##
##      xstr2
## xstr1    0    1
##     0 1712  142
##     1  450  196
```

Some simple analyses

```
summary(glm(y~xstr1, family=binomial))$coef
```

```
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -0.0622     0.0323   -1.92 5.44e-02
## xstr1         0.2658     0.0670    3.97 7.31e-05
```

```
summary(glm(y~xstr2, family=binomial))$coef
```

```
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -0.0368     0.0301   -1.22 0.222329
## xstr2         0.3108     0.0881    3.53 0.000419
```

Generative model

$$f(\text{params}) \prod_{i=1}^{n} f(y_i)\ f(x_i|y_i)\ f(x_1^*, x_2^*|x_i, y_i)$$

This will run (but ess/wall-time unpleasant)

```
genmod.string <- "model{

### prior distribution
gamma.0 ~ dunif(0,1)
gamma.1 ~ dunif(0,1)
sn1 ~ dunif(0.5, 1)
sp1 ~ dunif(0.5, 1)
sn2 ~ dunif(0.5, 1)
sp2 ~ dunif(0.5, 1)

trgt <- logit(gamma.1)-logit(gamma.0)

for (i in 1:n) {
  x[i] ~ dbern((1-y[i])*gamma.0+y[i]*gamma.1)
  xstr1[i] ~ dbern((1-x[i])*(1-sp1)+x[i]*sn1)
  xstr2[i] ~ dbern((1-x[i])*(1-sp2)+x[i]*sn2)
}

}"
```

Instead consider

```
genmod.string <- "model {
gamma.0 ~ dunif(0,1); gamma.1 ~ dunif(0,1)
trg <- logit(gamma.1)-logit(gamma.0)
sn1 ~ dunif(0.5,1); sp1 ~ dunif(0.5,1)
sn2 ~ dunif(0.5,1); sp2 ~ dunif(0.5,1)

### controls: dist(xstr1, xstr2 |y=0)
q.0[1] <- (1-gamma.0)*sp1*sp2 + gamma.0*(1-sn1)*(1-sn2)
q.0[2] <- (1-gamma.0)*(1-sp1)*sp2 + gamma.0*(sn1)*(1-sn2)
q.0[3] <- (1-gamma.0)*sp1*(1-sp2) + gamma.0*(1-sn1)*sn2
q.0[4] <- (1-gamma.0)*(1-sp1)*(1-sp2) + gamma.0*sn1*sn2
dat.0 ~ dmulti(q.0[], n.0)

### cases: dist(xstr1, xstr2 |y=1)
q.1[1] <- (1-gamma.1)*sp1*sp2 + gamma.1*(1-sn1)*(1-sn2)
q.1[2] <- (1-gamma.1)*(1-sp1)*sp2 + gamma.1*(sn1)*(1-sn2)
q.1[3] <- (1-gamma.1)*sp1*(1-sp2) + gamma.1*(1-sn1)*sn2
q.1[4] <- (1-gamma.1)*(1-sp1)*(1-sp2) + gamma.1*sn1*sn2
dat.1 ~ dmulti(q.1[], n.1)
}"
```

Pause: what's going on here?

Turn the crank

```
### generative model, data go in
mod <- jags.model(textConnection(genmod.string),
  data=list(dat.0=as.vector(table(dta)[,,1]),
            n.0=sum(table(dta)[,,1]),
            dat.1=as.vector(table(dta)[,,2]),
            n.1=sum(table(dta)[,,2])),
  n.chains=3)

update(mod,2000) #burn-in

### MCMC output comes out
opt.JAGS <- coda.samples(mod, n.iter=60000, thin=10,
  variable.names=c("gamma.0","gamma.1","sn1","sp1",
                   "sn2","sp2","trg"))
```

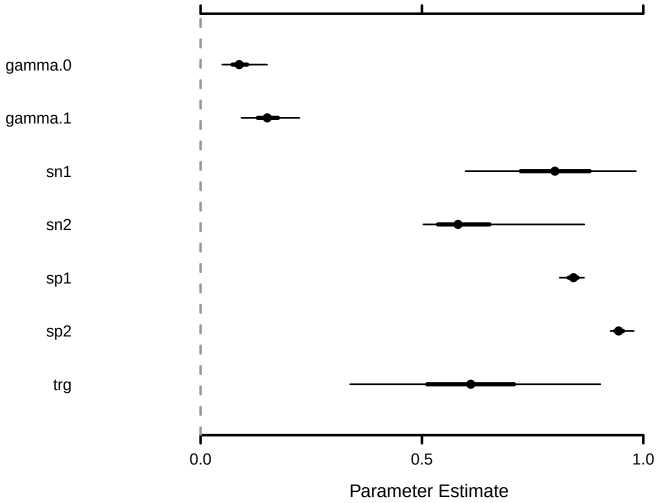
Inference

```
MCMCsummary(opt.JAGS)
```

```
##            mean     sd   2.5%    50% 97.5% Rhat n.eff
## gamma.0 0.0903 0.0256 0.0491 0.0872 0.150    1  2959
## gamma.1 0.1528 0.0337 0.0923 0.1506 0.223    1  3000
## sn1     0.7996 0.1033 0.5984 0.8003 0.983    1  3631
## sn2     0.6083 0.0955 0.5033 0.5815 0.867    1  4269
## sp1     0.8412 0.0144 0.8112 0.8423 0.866    1  4824
## sp2     0.9463 0.0136 0.9259 0.9442 0.979    1  3470
## trg     0.6121 0.1448 0.3378 0.6103 0.903    1 12140
```

Inference, continued

```
MCMCplot(opt.JAGS)
```



And a grand reveal

Pause to marvel for a moment: Asked a lot of these data, and they delivered

An aside on parameter-counting (1 of 2)

Relevant ref: Hui & Walter (1980, Biom.)

An aside on parameter-counting (2 of 2)

Relevant ref: Johnson & Hanson (2005, Stat. Sci., comment)

Pause some more: Lunch is never *completely* free

And yet another sense in which lunch isn't free