Session 7

Incorporating imperfect sensitivity and specificity into more complex models

Matt Denwood

2021-07-01

Recap

Models for diagnostic test evaluation require:

- At least 2 tests
- At least 2 populations, but preferably 3 or more
- Quite a lot of data

Recap

Models for diagnostic test evaluation require:

- At least 2 tests
- At least 2 populations, but preferably 3 or more
- Quite a lot of data

Fitting the models is technically quite straightforward

The real difficulty lies in the interpretation

- What exactly is the latent class?

Incorporating imperfect sensitivity and specificity into more complex models

Logistic regression in JAGS

```
model{

  for(i in 1:N){
    Observation[i] ~ dbern(prob[i])
    logit(prob[i]) <- intercept + beta1[Category[i]] +
    ↪  beta2*Covariate[i]
  }

  intercept ~ dnorm(0, 0.01)
  beta1[1] <- 0
  for(c in 2:NC){
    beta1[c] ~ dnorm(0, 0.01)
  }
  beta2 ~ dnorm(0, 0.01)

  #data# N, Observation, NC, Category, Covariate
  #monitor# intercept, beta1, beta2
  #inits# intercept, beta1, beta2
}
```

```
model{

  for(i in 1:N){
    Observation[i] ~ dbern(obs_prob[i])
    obs_prob[i] <- prob[i]*se + (1-prob[i])*(1-sp)
    logit(prob[i]) <- intercept + beta1[Category[i]] +
    ↪  beta2*Covariate[i]
  }

  se ~ dbeta(1,1)T(1-sp, )
  sp ~ dbeta(1,1)

  intercept ~ dnorm(0, 0.01)
  beta1[1] <- 0
  for(c in 2:NC){
    beta1[c] ~ dnorm(0, 0.01)
  }
  beta2 ~ dnorm(0, 0.01)

  #data# N, Observation, NC, Category, Covariate
  #monitor# intercept, beta1, beta2, se, sp
  #inits# intercept, beta1, beta2, se, sp
}
```

```
model{

  for(i in 1:N){
    Observation[i] ~ dbern(obs_prob[i])
    obs_prob[i] <- prob[i]*se + (1-prob[i])*(1-sp)
    logit(prob[i]) <- intercept + beta1[Category[i]] +
    ↪  beta2*Covariate[i]
  }

  se ~ dbeta(148.43, 16.49)T(1-sp, )
  sp ~ dbeta(240.03, 12.63)

  intercept ~ dnorm(0, 0.01)
  beta1[1] <- 0
  for(c in 2:NC){
    beta1[c] ~ dnorm(0, 0.01)
  }
  beta2 ~ dnorm(0, 0.01)

  #data# N, Observation, NC, Category, Covariate
  #monitor# intercept, beta1, beta2, se, sp
  #inits# intercept, beta1, beta2, se, sp
}
```

```
model{

  for(i in 1:N){
    Observation[i] ~ dbern(obs_prob[i])
    obs_prob[i] <- prob[i]*se + (1-prob[i])*(1-sp)
    logit(prob[i]) <- intercept + beta1[Category[i]] +
    ↪  beta2*Covariate[i]
  }

  se <- 0.9
  sp <- 0.95

  intercept ~ dnorm(0, 0.01)
  beta1[1] <- 0
  for(c in 2:NC){
    beta1[c] ~ dnorm(0, 0.01)
  }
  beta2 ~ dnorm(0, 0.01)

  #data# N, Observation, NC, Category, Covariate
  #monitor# intercept, beta1, beta2
  #inits# intercept, beta1, beta2
}
```

```
model{

  for(i in 1:N){
    Observation[i] ~ dbern(obs_prob[i])
    obs_prob[i] <- prob[i]*se + (1-prob[i])*(1-sp)
    logit(prob[i]) <- intercept + beta1[Category[i]] +
    ↪  beta2*Covariate[i]
  }

  #data# se, sp

  intercept ~ dnorm(0, 0.01)
  beta1[1] <- 0
  for(c in 2:NC){
    beta1[c] ~ dnorm(0, 0.01)
  }
  beta2 ~ dnorm(0, 0.01)

  #data# N, Observation, NC, Category, Covariate
  #monitor# intercept, beta1, beta2
  #inits# intercept, beta1, beta2
}
```

```
model{

  for(i in 1:N){
    Observation[i] ~ dbern(obs_prob[i])
    obs_prob[i] <- prob[i]*se[Test[i]] + (1-prob[i])*(1-sp[Test[i]])
    logit(prob[i]) <- intercept + beta1[Category[i]] +
    ↪  beta2*Covariate[i]
  }

  #data# se, sp

  intercept ~ dnorm(0, 0.01)
  beta1[1] <- 0
  for(c in 2:NC){
    beta1[c] ~ dnorm(0, 0.01)
  }
  beta2 ~ dnorm(0, 0.01)

  #data# N, Observation, NC, Category, Covariate, Test
  #monitor# intercept, beta1, beta2
  #inits# intercept, beta1, beta2
}
```

Other types of GL(M)M

You can use template.jags as inspiration:

```
template.jags(weight ~ group, family="gaussian", data=data,
↪  file="linear_model.txt")
## Your model template was created at "linear_model.txt" - it is highly
↪  advisable to examine the model syntax to be sure it is as intended
## You can then run the model using run.jags("linear_model.txt")
results <- run.jags("linear_model.txt")
## Loading required namespace: rjags
## module glm loaded
## module dic loaded
## Compiling rjags model...
## Calling the simulation using the rjags method...
## Note: the model did not require adaptation
## Burning in the model for 4000 iterations...
## Running the model for 10000 iterations...
## Simulation complete
## Calculating summary statistics...
## Note: The monitored variable 'group_effect[1]'
## appears to be non-stochastic; it will not be included
## in the convergence diagnostic
## Calculating the Gelman-Rubin statistic for 6
## variables....
## Finished running the simulation
```

```
results
##
## JAGS model summary statistics from 20000 samples (chains = 2;
↪  adapt+burnin = 5000):
##
##                       Lower95   Median Upper95     Mean
## regression_precision    0.877   1.9921  3.4796   2.0653
## intercept              4.5658   5.0327  5.4977   5.0329
## group_effect[1]             0        0       0        0
## group_effect[2]       -1.0436 -0.36951 0.26252 -0.37097
## deviance               40.184   42.735  48.629   43.426
## resid.sum.sq           8.7293   9.4237  12.237   9.8261
##
##                           SD Mode     MCerr MC%ofSD SSeff
## regression_precision 0.68937   -- 0.0054399     0.8 16059
## intercept             0.2362   --  0.001655     0.7 20369
## group_effect[1]            0    0        --      --    --
## group_effect[2]      0.33136   -- 0.0023661     0.7 19613
## deviance               2.658   --  0.021438     0.8 15372
## resid.sum.sq          1.2374   -- 0.0098384     0.8 15817
##
##                          AC.10   psrf
## regression_precision  -0.010936      1
## intercept             0.0087506      1
## group_effect[1]              --     --
## group_effect[2]       0.0052194      1
## deviance             -0.0010354      1
```

Supported features:

- Gaussian, binomial, Poisson, negative binomial, ZIB, ZIP, ZINB
- Random intercepts

We can also add (currently manually):

- Random slopes
- Spline terms
- Interval censoring

What about other models?

MCMC is highly flexible!

What about other models?

MCMC is highly flexible!

We can have:

- Hidden Markov models
- State Space models
- Other types of latent class model

What about other models?

MCMC is highly flexible!

We can have:

- Hidden Markov models
- State Space models
- Other types of latent class model

But does your data match your ambitions?

- All models can be specified
- Relatively few are identifiable

Before you go...

- Feedback on the course would be extremely welcome!
    - I will send an email later today with a survey link

Before you go...

- Feedback on the course would be extremely welcome!
  - I will send an email later today with a survey link
- Remember to keep an eye on the COST action website:
  - http://harmony-net.eu
  - Physical training schools are being run in September and accepting sign-ups now!

Practical session 7

Points to consider

1. When is there a benefit to adding imperfect test characteristics?
2. When is there no real benefit?