Session 7

Incorporating imperfect sensitivity and specificity into more complex models

Matt Denwood

2021-07-01

Recap

NOTE: THIS MATERIAL IS NOT YET FINALISED, PLEASE CHECK BACK SOON!

TODO: redo so it is covariates

- covariates on Se/Sp
- covariates on prevalence (random effects)
- categorical predictors on prevalence (population/group becomes predictor) - example where we modify the code to add a fixed effect across e.g. 3 populations
- categorical predictors on se/sp: group by population and allow to vary - example with age group and covid paper
- continuous covariates and/or random effects on prevalence

Recap

NOTE: THIS MATERIAL IS NOT YET FINALISED, PLEASE CHECK BACK SOON!

TODO: redo so it is covariates

- covariates on Se/Sp
- covariates on prevalence (random effects)
- categorical predictors on prevalence (population/group becomes predictor) - example where we modify the code to add a fixed effect across e.g. 3 populations
- categorical predictors on se/sp: group by population and allow to vary - example with age group and covid paper
- continuous covariates and/or random effects on prevalence

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
```

```
results
##
## JAGS model summary statistics from 20000 samples (chains = 2;
↪  adapt+burnin = 5000):
##
##                      Lower95   Median Upper95     Mean
## regression_precision 0.87833   1.9898  3.4844   2.0727
## intercept             4.5724   5.0333  5.4993   5.0341
## group_effect[1]            0        0       0        0
## group_effect[2]       -1.003 -0.37486 0.30503 -0.37316
## deviance              40.188   42.728  48.648   43.401
## resid.sum.sq          8.7294   9.4237  12.186   9.8179
##
##                           SD     Mode     MCerr MC%ofSD
## regression_precision  0.6922   1.8534 0.0053601     0.8
## intercept            0.23443   5.0226 0.0016577     0.7
## group_effect[1]            0        0        --      --
## group_effect[2]      0.33152 -0.38747 0.0023663     0.7
## deviance              2.6157   41.704  0.021374     0.8
## resid.sum.sq          1.2379   9.0347  0.009917     0.8
##
##                      SSeff      AC.10    psrf
## regression_precision 16677 -0.0071322       1
## intercept            20000 -0.0016243       1
## group_effect[1]         --         --      --
## group_effect[2]      19627 -0.0037905  1.0001
## deviance             14977 -0.0097212 0.99999
```

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!
  - https://www.survey-xact.dk/LinkCollector?key=RKMUENCXS11N
  - I will send a reminder email later today with (the same) survey link

Before you go...

- Feedback on the course would be extremely welcome!
  - https://www.survey-xact.dk/LinkCollector?key=RKMUENCXS11N
  - I will send a reminder email later today with (the same) 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?