

Session 4

Multi-test, multi-population models

Matt Denwood

2021-06-29

Why stop at two tests?

In *traditional* diagnostic test evaluation, one test is assumed to be a gold standard from which all other tests are evaluated

So it makes no difference if you assess one test at a time or do multiple tests at the same time

Using a latent class model each new test adds new information - so we should analyse all available test results in the same model

Simulating data

Simulating data using an arbitrary number of independent tests is quite straightforward:

```
# Parameter values to simulate:
```

```
N <- 200
```

```
sensitivity <- c(0.8, 0.9, 0.95)
```

```
specificity <- c(0.95, 0.99, 0.95)
```

```
Populations <- 2
```

```
prevalence <- c(0.25, 0.5)
```

```
data <- tibble(Population = sample(seq_len(Populations), N,  
  ↪ replace=TRUE)) %>%
```

```
  mutate(Status = rbinom(N, 1, prevalence[Population])) %>%
```

```
  mutate(Test1 = rbinom(N, 1, sensitivity[1]*Status +  
  ↪ (1-specificity[1])*(1-Status))) %>%
```

```
  mutate(Test2 = rbinom(N, 1, sensitivity[2]*Status +  
  ↪ (1-specificity[2])*(1-Status))) %>%
```

```
  mutate(Test3 = rbinom(N, 1, sensitivity[3]*Status +  
  ↪ (1-specificity[3])*(1-Status))) %>%
```

```
  select(-Status)
```

Model specification

Like for two tests, except it is now a $2 \times 2 \times 2$ table

Model specification

Like for two tests, except it is now a 2x2x2 table

```
Tally[1:8,p] ~ dmulti(prob[1:8,p], TotalTests[p])
```

```
# Probability of observing Test1- Test2- Test3-
```

```
prob[1,p] <- prev[p] * ((1-se[1])*(1-se[2])*(1-se[3])) +  
  (1-prev[p]) * (sp[1]*sp[2]*sp[3])
```

```
# Probability of observing Test1+ Test2- Test3-
```

```
prob[2,p] <- prev[p] * (se[1]*(1-se[2])*(1-se[3])) +  
  (1-prev[p]) * ((1-sp[1])*sp[2]*sp[3])
```

```
## snip ##
```

```
# Probability of observing Test1+ Test2+ Test3+
```

```
prob[3,p] <- prev[p] * (se[1]*se[2]*se[3]) +  
  (1-prev[p]) * ((1-sp[1])*(1-sp[2])*(1-sp[3]))
```

Model specification

Like for two tests, except it is now a 2x2x2 table

```
Tally[1:8,p] ~ dmulti(prob[1:8,p], TotalTests[p])
```

```
# Probability of observing Test1- Test2- Test3-  
prob[1,p] <- prev[p] * ((1-se[1])*(1-se[2])*(1-se[3])) +  
                (1-prev[p]) * (sp[1]*sp[2]*sp[3])
```

```
# Probability of observing Test1+ Test2- Test3-  
prob[2,p] <- prev[p] * (se[1]*(1-se[2])*(1-se[3])) +  
                (1-prev[p]) * ((1-sp[1])*sp[2]*sp[3])
```

```
## snip ##
```

```
# Probability of observing Test1+ Test2+ Test3+  
prob[3,p] <- prev[p] * (se[1]*se[2]*se[3]) +  
                (1-prev[p]) * ((1-sp[1])*(1-sp[2])*(1-sp[3]))
```

- We need to take **extreme** care with these equations, and the multinomial tabulation!!!

Are the tests conditionally independent?

- What do we mean by “conditionally independent?”
 - Independent of each other conditional on the latent state

Are the tests conditionally independent?

- What do we mean by “conditionally independent?”
 - Independent of each other conditional on the latent state
- Example: we have one blood test, one milk test, and one faecal test
 - But the blood and milk test are basically the same test, just on different samples
 - Therefore the blood and milk tests are more likely to give the same result

Are the tests conditionally independent?

- What do we mean by “conditionally independent?”
 - Independent of each other conditional on the latent state
- Example: we have one blood test, one milk test, and one faecal test
 - But the blood and milk test are basically the same test, just on different samples
 - Therefore the blood and milk tests are more likely to give the same result
- Example: we test people for COVID using an antigen test on a nasal swab, a PCR test on a throat swab, and the same antigen test on the same throat swab
 - The virus may be present in the throat, or nose, or neither, or both

Are the tests conditionally independent?

- What do we mean by “conditionally independent?”
 - Independent of each other conditional on the latent state
- Example: we have one blood test, one milk test, and one faecal test
 - But the blood and milk test are basically the same test, just on different samples
 - Therefore the blood and milk tests are more likely to give the same result
- Example: we test people for COVID using an antigen test on a nasal swab, a PCR test on a throat swab, and the same antigen test on the same throat swab
 - The virus may be present in the throat, or nose, or neither, or both

Dealing with correlation

It helps to consider the data simulation as a (simplified) biological process (where my parameters are not representative of real life!):

```
# The probability of infection with COVID in two populations:
prevalence <- c(0.01,0.05)
# The probability of shedding COVID in the nose conditional on
↪ infection:
nose_shedding <- 0.8
# The probability of shedding COVID in the throat conditional on
↪ infection:
throat_shedding <- 0.8
# The probability of detecting virus with the antigen test:
antigen_detection <- 0.75
# The probability of detecting virus with the PCR test:
pcr_detection <- 0.999
# The probability of random cross-reaction with the antigen test:
antigen_crossreact <- 0.05
# The probability of random cross-reaction with the PCR test:
pcr_crossreact <- 0.01
```

Dealing with correlation

It helps to consider the data simulation as a (simplified) biological process (where my parameters are not representative of real life!):

```
# The probability of infection with COVID in two populations:
prevalence <- c(0.01,0.05)
# The probability of shedding COVID in the nose conditional on
↪ infection:
nose_shedding <- 0.8
# The probability of shedding COVID in the throat conditional on
↪ infection:
throat_shedding <- 0.8
# The probability of detecting virus with the antigen test:
antigen_detection <- 0.75
# The probability of detecting virus with the PCR test:
pcr_detection <- 0.999
# The probability of random cross-reaction with the antigen test:
antigen_crossreact <- 0.05
# The probability of random cross-reaction with the PCR test:
pcr_crossreact <- 0.01
```

Note: cross-reactions are assumed to be independent!

```

# Other parameter values to simulate:
N <- 20000
Populations <- length(prevalence)

covid_data <- tibble(Population = sample(seq_len(Populations), N,
↳ replace=TRUE)) %>%
  ## True infection status:
  mutate(Status = rbinom(N, 1, prevalence[Population])) %>%
  ## Nose shedding status:
  mutate(Nose = Status * rbinom(N, 1, nose_shedding)) %>%
  ## Throat shedding status:
  mutate(Throat = Status * rbinom(N, 1, throat_shedding)) %>%
  ## The nose swab antigen test may be false or true positive as
  ↳ follows:
  mutate(NoseAG = case_when(
    Nose == 1 ~ rbinom(N, 1, antigen_detection),
    Nose == 0 ~ rbinom(N, 1, antigen_crossreact)
  )) %>%
  ## The throat swab antigen test may be false or true positive as
  ↳ follows:
  mutate(ThroatAG = case_when(
    Throat == 1 ~ rbinom(N, 1, antigen_detection),
    Throat == 0 ~ rbinom(N, 1, antigen_crossreact)
  )) %>%
  ## The PCR test may be false or true positive as follows:
  mutate(ThroatPCR = case_when(
    Throat == 1 ~ rbinom(N, 1, pcr_detection),

```

```

# The overall sensitivity of the tests can be calculated as follows:
covid_sensitivity <- c(
  # Nose antigen:
  nose_shedding*antigen_detection +
↪ (1-nose_shedding)*antigen_crossreact,
  # Throat antigen:
  throat_shedding*antigen_detection +
↪ (1-throat_shedding)*antigen_crossreact,
  # Throat PCR:
  throat_shedding*pcr_detection + (1-throat_shedding)*pcr_crossreact
)
covid_sensitivity
## [1] 0.6100 0.6100 0.8012

```

The overall specificity of the tests is more straightforward:

```
covid_specificity <- c(  
  # Nose antigen:  
  1 - antigen_crossreact,  
  # Throat antigen:  
  1 - antigen_crossreact,  
  # Throat PCR:  
  1 - pcr_crossreact  
)  
covid_specificity  
## [1] 0.95 0.95 0.99
```

The overall specificity of the tests is more straightforward:

```
covid_specificity <- c(  
  # Nose antigen:  
  1 - antigen_crossreact,  
  # Throat antigen:  
  1 - antigen_crossreact,  
  # Throat PCR:  
  1 - pcr_crossreact  
)  
covid_specificity  
## [1] 0.95 0.95 0.99
```

However: this assumes that cross-reactions are independent!

Model specification

```
prob[1,p] <- prev[p] * ((1-se[1])*(1-se[2])*(1-se[3]) +covse12
↪ +covse13 +covse23) +
      (1-prev[p]) * (sp[1]*sp[2]*sp[3] +covsp12 +covsp13
↪ +covsp23)

prob[2,p] <- prev[p] * (se[1]*(1-se[2])*(1-se[3]) -covse12 -covse13
↪ +covse23) +
      (1-prev[p]) * ((1-sp[1])*sp[2]*sp[3] -covsp12 -covsp13
↪ +covsp23)

## snip ##

# Covariance in sensitivity between tests 1 and 2:
covse12 ~ dunif( (se[1]-1)*(1-se[2]) , min(se[1],se[2]) -
↪ se[1]*se[2] )
# Covariance in specificity between tests 1 and 2:
covsp12 ~ dunif( (sp[1]-1)*(1-sp[2]) , min(sp[1],sp[2]) -
↪ sp[1]*sp[2] )

## snip ##
```

Model specification

```
prob[1,p] <- prev[p] * ((1-se[1])*(1-se[2])*(1-se[3]) +covse12
↪ +covse13 +covse23) +
      (1-prev[p]) * (sp[1]*sp[2]*sp[3] +covsp12 +covsp13
↪ +covsp23)

prob[2,p] <- prev[p] * (se[1]*(1-se[2])*(1-se[3]) -covse12 -covse13
↪ +covse23) +
      (1-prev[p]) * ((1-sp[1])*sp[2]*sp[3] -covsp12 -covsp13
↪ +covsp23)

## snip ##

# Covariance in sensitivity between tests 1 and 2:
covse12 ~ dunif( (se[1]-1)*(1-se[2]) , min(se[1],se[2]) -
↪ se[1]*se[2] )
# Covariance in specificity between tests 1 and 2:
covsp12 ~ dunif( (sp[1]-1)*(1-sp[2]) , min(sp[1],sp[2]) -
↪ sp[1]*sp[2] )

## snip ##
```

It is quite easy to get the terms slightly wrong!

The model code and data format for an arbitrary number of populations (and tests) can be determined automatically using the `template_huiwalter` function from the `runjas` package:

```
template_huiwalter(covid_data %>% select(Population, NoseAG, ThroatAG,  
  ↳ ThroatPCR), outfile = 'covidmodel.txt')  
## The model and data have been written to covidmodel.txt in the current  
↳ working directory  
## You should check and alter priors before running the model
```

This generates self-contained model/data/initial values etc:

```
model{

  ## Observation layer:

  # Complete observations (N=20000):
  for(p in 1:Populations){
    Tally_RRR[1:8,p] ~ dmulti(prob_RRR[1:8,p], N_RRR[p])

    prob_RRR[1:8,p] <- se_prob[1:8,p] + sp_prob[1:8,p]
  }

  ## Observation probabilities:

  for(p in 1:Populations){

    # Probability of observing NoseAG- ThroatAG- ThroatPCR- from a
    ↪ true positive::
    se_prob[1,p] <- prev[p] * ((1-se[1])*(1-se[2])*(1-se[3])
    ↪ +covse12 +covse13 +covse23)
    # Probability of observing NoseAG- ThroatAG- ThroatPCR- from a
    ↪ true negative::
    sp_prob[1,p] <- (1-prev[p]) * (sp[1]*sp[2]*sp[3] +covsp12
    ↪ +covsp13 +covsp23)
```

And can be run directly from R:

```
results <- run.jags('covidmodel.txt')  
## Loading required namespace: rjags  
results
```

	Lower95	Median	Upper95	SSeff	psrf
se[1]	0.581	0.632	0.682	9221	1.001
se[2]	0.715	0.767	0.816	7625	1.000
se[3]	0.945	0.982	1.000	5704	1.000
sp[1]	0.944	0.947	0.950	11678	1.000
sp[2]	0.948	0.951	0.954	9657	1.000
sp[3]	0.987	0.989	0.991	7867	1.000
prev[1]	0.005	0.007	0.009	9740	1.000
prev[2]	0.037	0.042	0.046	8014	1.000
covse12	0.000	0.000	0.000	NA	NA
covsp12	0.000	0.000	0.000	NA	NA
covse13	0.000	0.000	0.000	NA	NA
covsp13	0.000	0.000	0.000	NA	NA
covse23	0.000	0.000	0.000	NA	NA
covsp23	0.000	0.000	0.000	NA	NA

- Modifying priors must still be done directly in the model file
 - Same for adding .RNG.seed and the deviance monitor
- The model needs to be re-generated if the data changes
 - But remember that your modified priors will be reset
- There must be a single column for the population (as a factor), and all of the other columns (either factor, logical or numeric) are interpreted as being test results

- Modifying priors must still be done directly in the model file
 - Same for adding .RNG.seed and the deviance monitor
- The model needs to be re-generated if the data changes
 - But remember that your modified priors will be reset
- There must be a single column for the population (as a factor), and all of the other columns (either factor, logical or numeric) are interpreted as being test results
- Covariance terms are all deactivated by default

Activating covariance terms

Find the lines for the covariances that we want to activate (i.e. the two Throat tests):

```
# Covariance in sensitivity between ThroatAG and ThroatPCR tests:
# covse23 ~ dunif( (se[2]-1)*(1-se[3]) , min(se[2],se[3]) - se[2]*se[3]
↪ ) ## if the sensitivity of these tests may be correlated
  covse23 <- 0 ## if the sensitivity of these tests can be assumed to be
  ↪ independent
# Covariance in specificity between ThroatAG and ThroatPCR tests:
# covsp23 ~ dunif( (sp[2]-1)*(1-sp[3]) , min(sp[2],sp[3]) - sp[2]*sp[3]
↪ ) ## if the specificity of these tests may be correlated
  covsp23 <- 0 ## if the specificity of these tests can be assumed to be
  ↪ independent
```


And edit so it looks like:

```
# Covariance in sensitivity between ThroatAG and ThroatPCR tests:
covse23 ~ dunif( (se[1]-1)*(1-se[2]) , min(se[1],se[2]) - se[1]*se[2] )
↪ ## if the sensitivity of these tests may be correlated
  # covse23 <- 0 ## if the sensitivity of these tests can be assumed to
  ↪ be independent
# Covariance in specificity between ThroatAG and ThroatPCR tests:
covsp23 ~ dunif( (sp[1]-1)*(1-sp[2]) , min(sp[1],sp[2]) - sp[1]*sp[2] )
↪ ## if the specificity of these tests may be correlated
  # covsp23 <- 0 ## if the specificity of these tests can be assumed to
  ↪ be independent
```

[i.e. swap the comments around]

You will also need to uncomment out the relevant initial values for BOTH chains (on lines 117-122 and 128-133):

```
# "covse12" <- 0  
# "covse13" <- 0  
# "covse23" <- 0  
# "covsp12" <- 0  
# "covsp13" <- 0  
# "covsp23" <- 0
```

So that they look like:

```
# "covse12" <- 0  
# "covse13" <- 0  
"covse23" <- 0  
# "covsp12" <- 0  
# "covsp13" <- 0  
"covsp23" <- 0
```

```

results <- run.jags('covidmodel.txt', sample=50000)
## Compiling rjags model...
## Calling the simulation using the rjags method...
## Adapting the model for 1000 iterations...
## Burning in the model for 4000 iterations...
## Running the model for 50000 iterations...
## Simulation complete
## Calculating summary statistics...
## Note: The monitored variables 'covse12', 'covsp12',
## 'covse13' and 'covsp13' appear to be non-stochastic;
## they will not be included in the convergence
## diagnostic
## Calculating the Gelman-Rubin statistic for 14
## variables....
## Finished running the simulation
results
##
## JAGS model summary statistics from 100000 samples (chains = 2;
↳ adapt+burnin = 5000):
##
##           Lower95      Median      Upper95      Mean
## se[1]      0.59213      0.6552      0.7221      0.65615
## se[2]      0.54353      0.66687      0.78293      0.6648
## se[3]      0.74346      0.87547      0.99999      0.87137
## sp[1]       0.9452      0.94914      0.95367      0.94921
## sp[2]       0.94586      0.94929      0.95268      0.94928
## sp[3]       0.9854      0.98785      0.99004      0.98781

```

Practical considerations

- Correlation terms add complexity to the model in terms of:
 - Opportunity to make a coding mistake
 - Reduced identifiability

Practical considerations

- Correlation terms add complexity to the model in terms of:
 - Opportunity to make a coding mistake
 - Reduced identifiability
- The `template_huiwalter` function helps us with coding mistakes
- Only careful consideration of covariance terms can help us with identifiability

Practical considerations

- Correlation terms add complexity to the model in terms of:
 - Opportunity to make a coding mistake
 - Reduced identifiability
- The `template_huiwalter` function helps us with coding mistakes
- Only careful consideration of covariance terms can help us with identifiability

We will return to these themes tomorrow!

Practical session 4

Points to consider

1. How does including a third test impact the inference for the first two tests?
2. What happens if we include correlation between tests?
3. Can we include correlation if we only have 2 tests?

Summary

- Including multiple tests is technically easy but philosophically more difficult
- Complexity of adding correlation terms increases non-linearly with more tests
 - Probably best to stick to correlations with biological justification?
- Adding/removing test results may change the posterior for
 - Other test Se / Sp
 - Prevalence

Summary

- Including multiple tests is technically easy but philosophically more difficult
- Complexity of adding correlation terms increases non-linearly with more tests
 - Probably best to stick to correlations with biological justification?
- Adding/removing test results may change the posterior for
 - Other test Se / Sp
 - Prevalence

Homework: think about what exactly the latent class is in these situations:

1. An antigen plus antibody test