Session 4

Multi-test, multi-population models

Matt Denwood

2021-06-29

Why stop at two tests?

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

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

Why stop at two tests?

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

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: simple example

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 2x2x2 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?

- Example: we have one blood, one milk, and one faecal test
    - But the blood and milk test are basically the same test
    - Therefore they are more likely to give the same result

Are the tests conditionally independent?

- Example: we have one blood, one milk, and one faecal test
  - But the blood and milk test are basically the same test
  - Therefore they 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, nose, neither, or both
  - But we use the same antigen test twice
    - Might it cross-react with the same non-target virus?

Are the tests conditionally independent?

- Example: we have one blood, one milk, and one faecal test
    - But the blood and milk test are basically the same test
    - Therefore they 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, nose, neither, or both
    - But we use the same antigen test twice
        - Might it cross-react with the same non-target virus?
- In both situations we have pairwise correlation between some of the tests

Dealing with correlation: Covid example

TODO: add DAG showing blocking path

TODO: refer back to this from session 5

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

```r
# 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:
```

Dealing with correlation: Covid example

TODO: add DAG showing blocking path

TODO: refer back to this from session 5

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:
```

Simulating latent states:

```r
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))
```

Simulating test results:

```r
covid_data <- covid_data %>%
  ## The nose swab antigen test may be false or true positive:
  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:
  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:
  mutate(ThroatPCR = case_when(
    Throat == 1 ~ rbinom(N, 1, pcr_detection),
    Throat == 0 ~ rbinom(N, 1, pcr_crossreact)
  ))
```

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

TODO: show formulation as proportion of possible correlation

```
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] )
```

Model specification

TODO: show formulation as proportion of possible correlation

```
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] )
```

It is quite easy to get the terms slightly wrong!

Template Hui-Walter

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 runjags package:

```
template_huiwalter(
  covid_data %>% select(Population, NoseAG, ThroatAG, ThroatPCR),
  outfile = 'covidmodel.txt')
```

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)

        # Probability of observing NoseAG+ ThroatAG- ThroatPCR- from a
        ↪  true positive::
```