

Session 2

Basic Hui-Walter models

Matt Denwood

2021-06-28

Session 2: Basic Hui-Walter models

Hui-Walter Model

- A particular model formulation that was originally designed for evaluating diagnostic tests in the absence of a gold standard
- Not necessarily (or originally) Bayesian but often implemented using Bayesian MCMC
- But evaluating an imperfect test against another imperfect test is a bit like pulling a rabbit out of a hat
 - If we don't know the true disease status, how can we estimate sensitivity or specificity for either test?

Model Specification

```
model{
  Tally ~ dmulti(prob, N)

  # Test1- Test2-
  prob[1] <- (prev * ((1-se[1])*(1-se[2]))) + ((1-prev) *
    ↪ ((sp[1])*(sp[2])))

  # Test1+ Test2-
  prob[2] <- (prev * ((se[1])*(1-se[2]))) + ((1-prev) *
    ↪ ((1-sp[1])*(sp[2])))

  # Test1- Test2+
  prob[3] <- (prev * ((1-se[1])*(se[2]))) + ((1-prev) *
    ↪ ((sp[1])*(1-sp[2])))
```

```

# Test1+ Test2+
  prob[4] <- (prev * ((se[1])*(se[2]))) + ((1-prev) *
    ↪ ((1-sp[1])*(1-sp[2])))

prev ~ dbeta(1, 1)
se[1] ~ dbeta(1, 1)
sp[1] ~ dbeta(1, 1)
se[2] ~ dbeta(1, 1)
sp[2] ~ dbeta(1, 1)

#data# Tally, N
#monitor# prev, prob, se, sp, deviance
#inits# prev, se, sp
}

```

```
twoXtwo <- matrix(c(48, 12, 4, 36), ncol=2, nrow=2)
twoXtwo
##      [,1] [,2]
## [1,]   48   4
## [2,]   12  36
```

```
library('runjags')
```

```
Tally <- as.numeric(twoXtwo)
N <- sum(Tally)
```

```
prev <- list(chain1=0.05, chain2=0.95)
se <- list(chain1=c(0.01,0.99), chain2=c(0.99,0.01))
sp <- list(chain1=c(0.01,0.99), chain2=c(0.99,0.01))

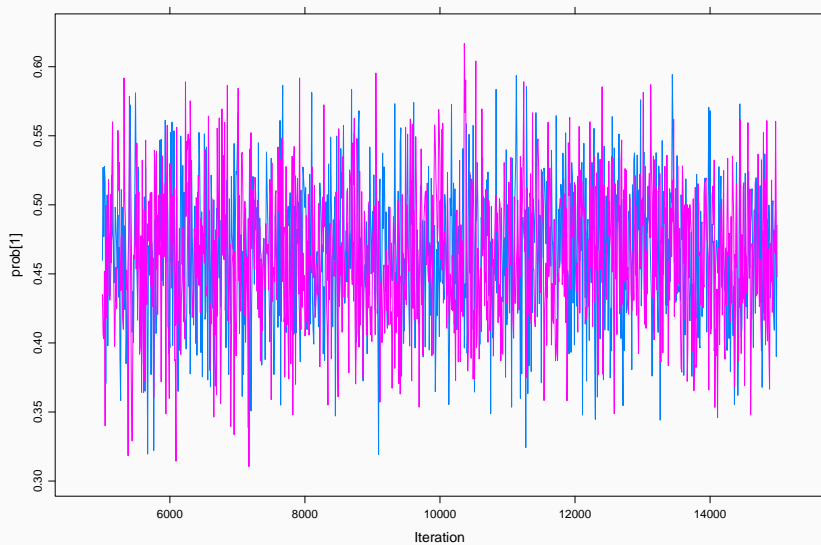
results <- run.jags('basic_hw.bug', n.chains=2)
```

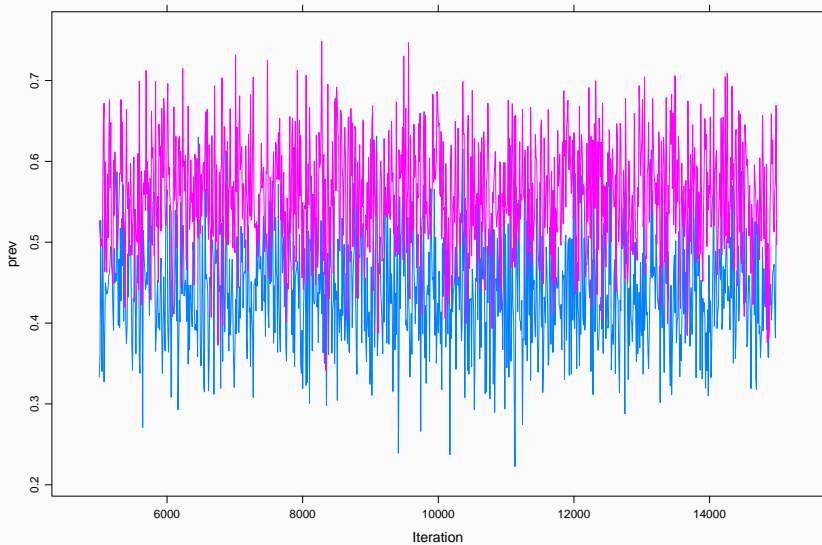
[Remember to check convergence and effective sample size!]

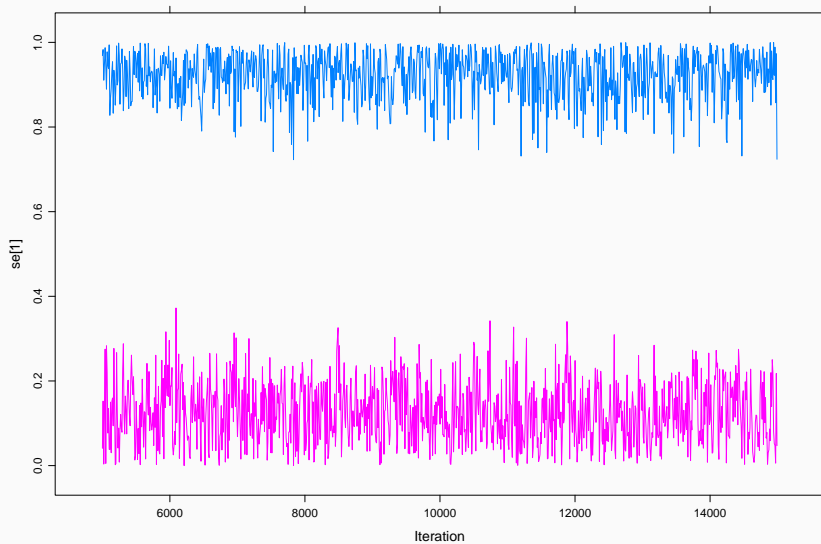
results

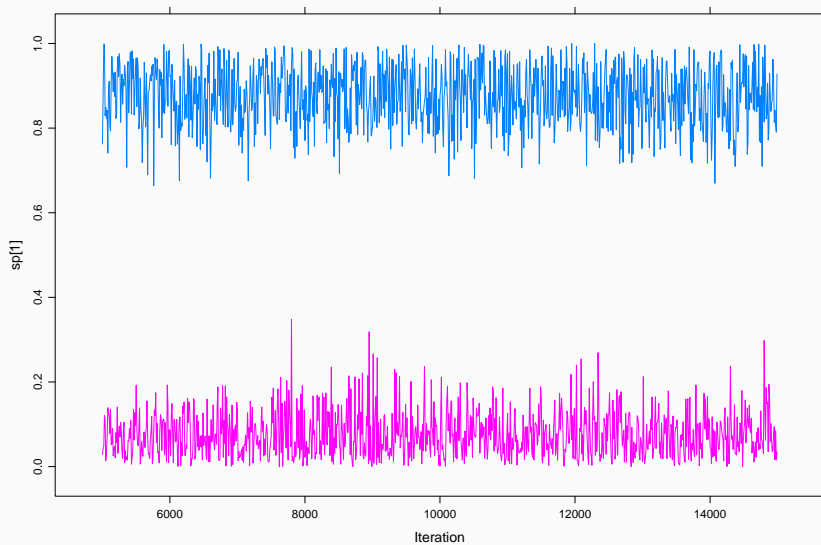
	Lower95	Median	Upper95	SSeff	psrf
prev	0.332	0.498	0.672	4095	2.316
prob[1]	0.367	0.463	0.558	13452	1.000
prob[2]	0.073	0.133	0.200	13575	1.000
prob[3]	0.017	0.055	0.103	9722	1.000
prob[4]	0.255	0.344	0.435	13917	1.000
se[1]	0.029	0.528	1.000	4464	15.062
se[2]	0.000	0.370	0.964	4913	13.833
sp[1]	0.000	0.504	0.971	4645	15.324
sp[2]	0.036	0.594	1.000	4279	13.710
deviance	12.301	15.112	21.423	8572	1.000

- Does anybody spot a problem?

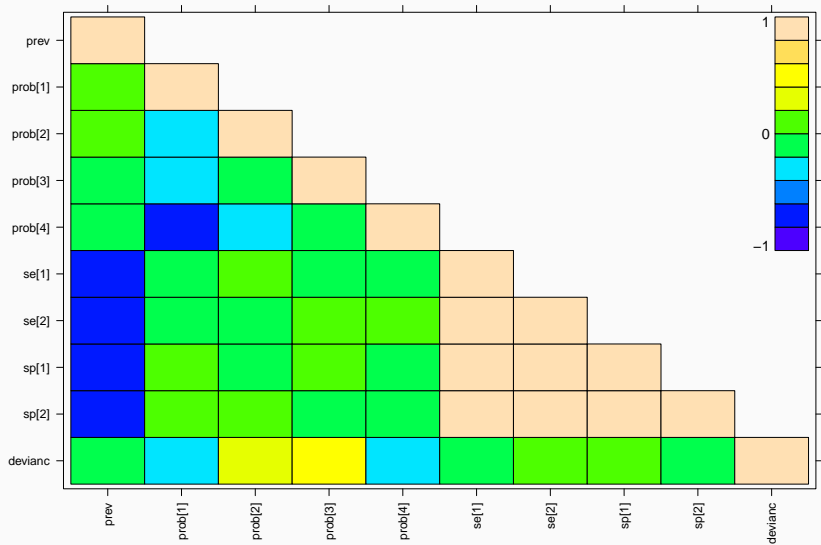








TODO: add deviance plot



Label Switching

How to interpret a test with $Se=0\%$ and $Sp=0\%$?

Label Switching

How to interpret a test with $Se=0\%$ and $Sp=0\%$?

- The test is perfect - we are just holding it upside down. . .

Label Switching

How to interpret a test with $Se=0\%$ and $Sp=0\%$?

- The test is perfect - we are just holding it upside down...

We can force $se+sp \geq 1$:

```
se[1] ~ dbeta(1, 1)
sp[1] ~ dbeta(1, 1)T(1-se[1], )
```

Or:

```
se[1] ~ dbeta(1, 1)T(1-sp[1], )
sp[1] ~ dbeta(1, 1)
```

This allows the test to be useless, but not worse than useless.

Note: the joint posterior is not necessarily what you would expect

TODO show model without data

Alternative: rejection step (ability to be added to `IACS/runiags` at

Alternatively we can have the weakly informative priors:

```
se[1] ~ dbeta(2, 1)
sp[1] ~ dbeta(2, 1)
```

To give the model some information that we expect the test characteristics to be closer to 100% than 0%.

Or we can use stronger priors for one or both tests.

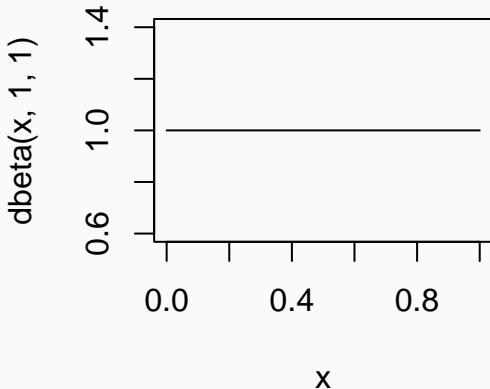
TODO: show results with truncated priors

- Be **very** careful with the order of combinations in `dmultinom!`
- Check your results carefully to ensure they make sense!
- Convergence is more problematic than usual
- These models need A LOT of data, and/or strong priors for one of the tests

A different prior

- A quick way to see the distribution of a prior:

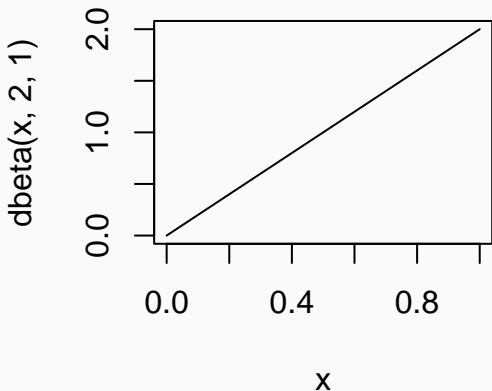
```
curve(dbeta(x, 1, 1), from=0, to=1)
```



```
qbeta(c(0.025,0.975), shape1=1, shape2=1)  
## [1] 0.025 0.975
```

- This was minimally informative, but how does that compare to a weakly informative prior for e.g. sensitivity?

```
curve(dbeta(x, 2, 1), from=0, to=1)
```

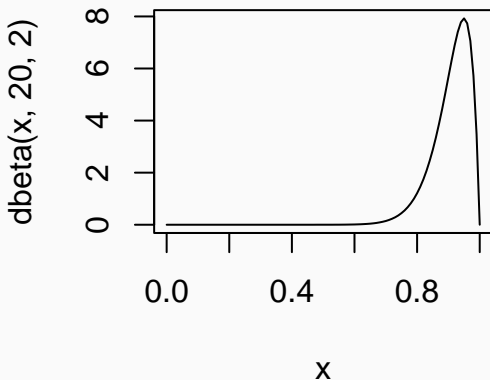


```
qbeta(c(0.025,0.975), shape1=2, shape2=1)  
## [1] 0.1581139 0.9874209
```

- Or more accurately:

- What about a more informative prior?

```
curve(dbeta(x, 20, 2), from=0, to=1)
```



```
hpd(qbeta, shape1=20, shape2=2)  
## [1] 0.7919691 0.9973994
```

Choosing a prior

- Typically we are given median and 95% confidence intervals from a paper, e.g.:

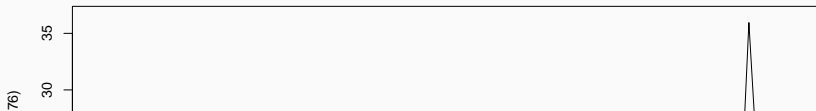
“The median (95% CI) estimates of the sensitivity and specificity of the shiny new test were 94% (92-96%) and 99% (97-100%) respectively”

- How can we generate a prior from this?

The PriorGen package

"The median (95% CI) estimates of the sensitivity and specificity of the shiny new test were 94% (92-96%) and 99% (97-100%) respectively"

```
library("PriorGen")  
## Loading required package: rootSolve  
findbeta(themedian = 0.94, percentile=0.95, percentile.value = 0.92)  
## [1] "The desired Beta distribution that satisfies the specified  
↪ conditions is: Beta( 429.95 27.76 )"  
## [1] "Here is a plot of the specified distribution."  
## [1] "Descriptive statistics for this distribution are:"  
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.  
## 0.8910 0.9322 0.9399 0.9393 0.9473 0.9749  
## [1] "Verification: The percentile value 0.92 corresponds to the 0.05  
↪ th percentile"  
  
curve(dbeta(x, shape1=429.95, shape2=27.76))
```



Analysing simulated data

This is useful to check that we can recover parameter values!

```
se <- c(0.9, 0.6)
sp <- c(0.95, 0.9)
N <- 1000
prevalence <- 0.25

data <- tibble(Status = rbinom(N, 1, prevalence)) %>%
  mutate(Test1 = rbinom(N, 1, se[1]*Status + (1-sp[1])*(1-Status))) %>%
  mutate(Test2 = rbinom(N, 1, se[2]*Status + (1-sp[2])*(1-Status)))

twoXtwo <- with(data, table(Test1, Test2))
Tally <- as.numeric(twoXtwo)
```

Practical Session 2

Points to consider

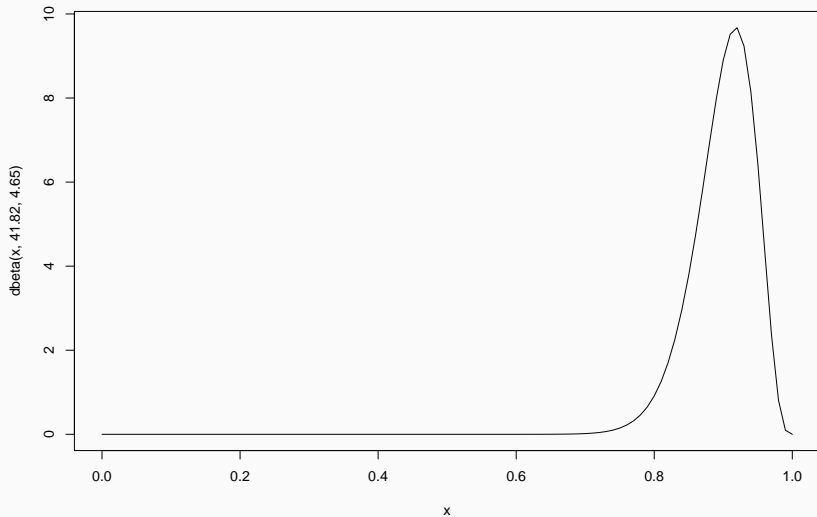
1. What is the typical autocorrelation (and therefore effective sample size) of Hui-Walter models compared to the simpler models we were running earlier? Is there any practical consequence of this?
2. How does changing the prior distributions for the se and sp of one test affect the inference for the other test parameters?

Priors

- We cannot estimate se, sp and prevalence simultaneously
 - We need strong priors for se and sp
- We can use the PriorGen package to generate Beta priors based on published results, for example:

```
PriorGen::findbeta(themean=0.9, percentile = 0.975, percentile.value =  
↳ 0.8)  
## [1] "The desired Beta distribution that satisfies the specified  
↳ conditions is: Beta( 41.82 4.65 )"  
## [1] "Here is a plot of the specified distribution."  
## [1] "Descriptive statistics for this distribution are:"  
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.  
## 0.6083 0.8744 0.9058 0.9002 0.9320 0.9922  
## [1] "Verification: The percentile value 0.8 corresponds to the 0.025  
↳ th percentile"
```

```
qbeta(c(0.025, 0.5, 0.975), 41.82, 4.65)  
## [1] 0.7999144 0.9056630 0.9677283  
curve(dbeta(x, 41.82, 4.65), from=0, to=1)
```



Exercise

- Find beta distribution priors for:
 - Sensitivity = 0.9 (95% CI: 0.85 - 0.95)
 - Specificity = 0.95 (95%CI: 0.92-0.97)
- Look at these distributions using curve and qbeta
- Modify the imperfect test model using these priors and re-estimate prevalence

Optional Exercise

- Run the same model with se and sp fixed to the mean estimate
 - How does this affect CI for prevalence?
- Run the same model with se and sp fixed to 1
 - How does this affect estimates and CI for prevalence?

Summary

- Hui-Walter models are like pulling a rabbit out of a hat, but:
 - They typically exhibit high autocorrelation
 - They may not converge immediately
 - We need a larger sample for the same effective sample size
- More informative priors for one test will
 - Improve identifiability of the model
 - Impact the inference for all other parameters in the model!