KU Leuven Summer School
Segment 5B
Information Flow in the IFR Inference Problem

Paul Gustafson

September 16, 2022

Consider a different/hypothetical pandemic (in code)

There are K=10 jurisdictions. Unbeknownst to us, here is the truth:

```r
### infection fatality rate
### known to be constant across jurisditions, but value unknown

ifr <- 0.025

### actual infection rates, again unknown

ir <- c(0.2, 0.1, 0.6, 0.5, 0.3, 0.35, 0.2, 0.1, 0.5, 0.45)

### actual extents of preferential testing

phi <- c(8, 2, 3, 3, 4, 7, 6, 2, 3, 7)
```

What does nature reveal to us?

```
### death rate
dr.obs <- ir*ifr
dr.obs
```

```
##  [1] 0.00500 0.00250 0.01500 0.01250 0.00750 0.00875 0.00500
##  [8] 0.00250 0.01250 0.01125
```

✓

```
### test positivity rate
tp.obs <- 1-(1-ir)^phi
tp.obs
```

```
##  [1] 0.832 0.190 0.936 0.875 0.760 0.951 0.738 0.190 0.875 0.985
```

✓

```
### and ensure no cheating!
rm(ir, ifr, phi)
```

And what else do we have going for us?

We have a valid bound for all the preferential testing parameters

```r
phi.int <- c(1,10)
```

✓

So what do individual jurisdictions tell us about the IFR

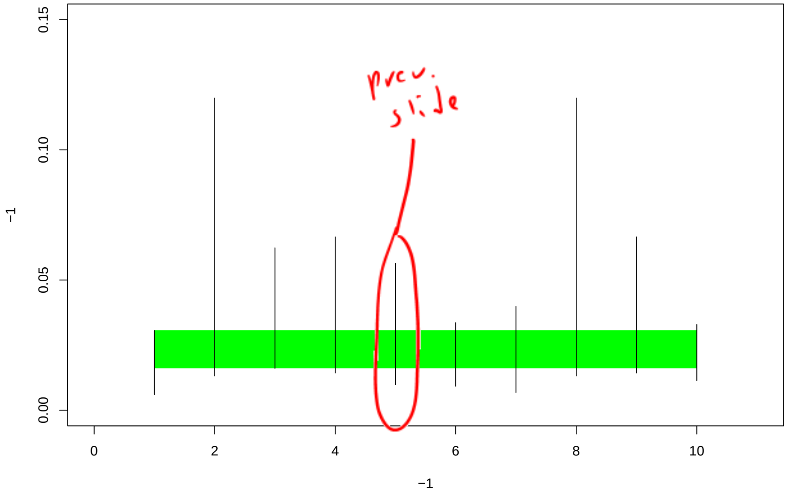
Since $TP_5 = 1 - (1-IR_5)^{\phi_5}$ and $DR_5 = IR_5 IFR_5$

```
### e.g., jurisdiction 5 tells us that IFR must lie in
dr.obs[5] / ( 1 - (1-tp.obs[5])^(1/phi.int) )
```

```
## [1] 0.00987 0.05641
```

```
### do for all jurisdictions
ifr.bnd.lo <- dr.obs / ( 1 - (1-tp.obs)^(1/phi.int[1]) )
ifr.bnd.hi <- dr.obs / ( 1 - (1-tp.obs)^(1/phi.int[2]) )
```

Visualize how well we have squeezed in on our target



regions 1 & 3 are our friends

Have learned that the IFR lies in

```
round(c(max(ifr.bnd.lo), min(ifr.bnd.hi)),3)
```

```
## [1] 0.016 0.031
```

But getting back to Bayes

after lunch

The above proof-of-concept takes place in asymptotia, no explict Bayes inference.

So now back to Bayes, but for demonstration, want/need a simpler sandbox than the full COVID example

Generative Model

target parameter typical log IFR

prior $f(\mu_a)f(\sigma_a)f(\mu_b)f(\sigma_b)f(\omega)$

presume ν_k and κ_k known

$$
\begin{aligned}
(a_1, \dots a_k | \mu_a, \sigma_a^2) &\overset{iid}{\sim} N(\mu_a, \sigma_a^2) \\
(b_1, \dots b_k | \mu_b, \sigma_b^2) &\overset{iid}{\sim} N(\mu_b, \sigma_b^2) \\
(c_1, \dots c_k) &\overset{iid}{\sim} U(0, \omega)
\end{aligned}
$$

log IFR_k

log IR_k

$$
\begin{aligned}
\log(D_k/P_k) &\sim N(\boxed{a_k} + \boxed{b_k}, \nu_k^2) \\
\log(CC_k/T_k) &\sim N(b_k + \boxed{c_k}, \kappa_k^2)
\end{aligned}
$$

simpler biased sampling parameter

$c_k = 0$ ↔ random sampling

Code this up

```
genmod.string <- "
model {

  mn.a ~ dnorm(0, 0.01)
  sd.a ~ dbeta(1,19) ### limited ifr variation
  prc.a <- pow(sd.a,-2)

  mn.b ~ dnorm(0, 0.01)
  sd.b ~ dunif(0,1) ### not very limited ir variation
  prc.b <- pow(sd.b,-2)

  for (i in 1:k) {
    a[i] ~ dnorm(mn.a, prc.a)
    b[i] ~ dnorm(mn.b, prc.b)
    c[i] ~ dunif(0,2) ### effect of preferential testing

    log.dth.rate[i] ~ dnorm(a[i] + b[i], prc.dth[i])
    log.tps.rate[i] ~ dnorm(b[i] + c[i], prc.tps[i])
  }
}"
```

First test dataset

$$IFR_k = e^{\mu_a + \sigma_a Z}$$

```
k <- 30

set.seed(13)
```

μ_a σ_a

```
a.true <- rnorm(k, log(0.04), log(1.005))
b.true <- rnorm(k,  log(0.35), log(1.15))
c.true <- runif(k, 0,2)

prc.dth <- rep(100000*(0.02/0.98), k) ### ???
prc.tps <- rep(1000*(0.5/0.5), k)     ### ???

log.dth.rate <- rnorm(k, a.true+b.true, 1/sqrt(prc.dth))
log.tps.rate <- rnorm(k, b.true+c.true, 1/sqrt(prc.tps))
```

Binomial proportion

$$Var(\hat{p}) = \frac{\hat{p}(1-\hat{p})}{n} \Rightarrow Var \log(\hat{p}) \approx \frac{1-p}{pn}$$

Take a look

```r
head(cbind(log.dth.rate, sqrt(1/prc.dth),
           log.tps.rate, sqrt(1/prc.tps)))
```

```
##       log.dth.rate        log.tps.rate
## [1,]        -4.29 0.0221      -0.8597 0.0316
## [2,]        -4.11 0.0221      -0.6240 0.0316
## [3,]        -4.32 0.0221       0.1447 0.0316
## [4,]        -4.33 0.0221       0.0265 0.0316
## [5,]        -4.29 0.0221      -0.1526 0.0316
## [6,]        -4.35 0.0221      -1.1286 0.0316
```

Turn the crank

```r
### generative model, data go in
mod <- jags.model(
  textConnection(genmod.string),
  data=list(log.dth.rate=log.dth.rate, prc.dth=prc.dth,
            log.tps.rate=log.tps.rate, prc.tps=prc.tps, k=k),
  n.chains=5)

update(mod, 2500) # burn-in

###  MC output comes out
opt.JAGS <- coda.samples(mod, n.iter=500000, thin=100,
  variable.names=c("mn.a","a[1]","b[1]","c[1]"))
```

slow

Get an answer

```
MCMCsummary(opt.JAGS)
```

```
##         mean     sd  2.5%    50%  97.5% Rhat n.eff
## a[1] -3.17 0.0566 -3.28 -3.173 -3.056    1  5869
## b[1] -1.12 0.0597 -1.24 -1.120 -1.004    1  6424
## c[1]  0.26 0.0674  0.13  0.259  0.394    1  7616
## mn.a -3.17 0.0446 -3.26 -3.169 -3.080    1  3769
```

truth
log(.04)
= -3.22

inferring log IFR
to ± .09

→ inferring IFR to within
10%

Second test dataset

only change - less actual heterogeneity in how preferential the sampling is

```
k <- 30

set.seed(13)
a.true <- rnorm(k, log(0.04), log(1.005))
b.true <- rnorm(k,  log(0.35), log(1.05))
c.true <- runif(k, 0.4,1.6)

prc.dth <- rep(100000*(0.02/0.98), k) ### ???
prc.tps <- rep(1000*(0.5/0.5), k)     ### ???

log.dth.rate <- rnorm(k, a.true+b.true, 1/sqrt(prc.dth))
log.tps.rate <- rnorm(k, b.true+c.true, 1/sqrt(prc.tps))
```

Turn the crank

```
### generative model, data go in
mod <- jags.model(
  textConnection(genmod.string),
  data=list(log.dth.rate=log.dth.rate, prc.dth=prc.dth,
            log.tps.rate=log.tps.rate, prc.tps=prc.tps, k=k),
  n.chains=5)

update(mod, 2500) # burn-in

###  MC output comes out
opt.JAGS <- coda.samples(mod, n.iter=500000, thin=100,
  variable.names=c("mn.a","a[1]","b[1]","c[1]"))
```

Get an answer

broken

```
MCMCsummary(opt.JAGS)
```

```
##          mean    sd  2.5%    50%  97.5% Rhat n.eff
## a[1] -3.138 0.231 -3.57 -3.110 -2.789 1.02    90
## b[1] -1.150 0.231 -1.50 -1.178 -0.719 1.02    90
## c[1]  0.608 0.233  0.17  0.636  0.969 1.02    89
## mn.a -3.132 0.230 -3.56 -3.103 -2.786 1.02    93
```

but to the extent trustworthy. factor of five increase (on log-scale)

large posterior uncertainty about IFR

Our pesky folk theorem rears its inconvenient head again!

As we approach weaker information settings off-the-shelf MCMC can let us down

Thoughts?