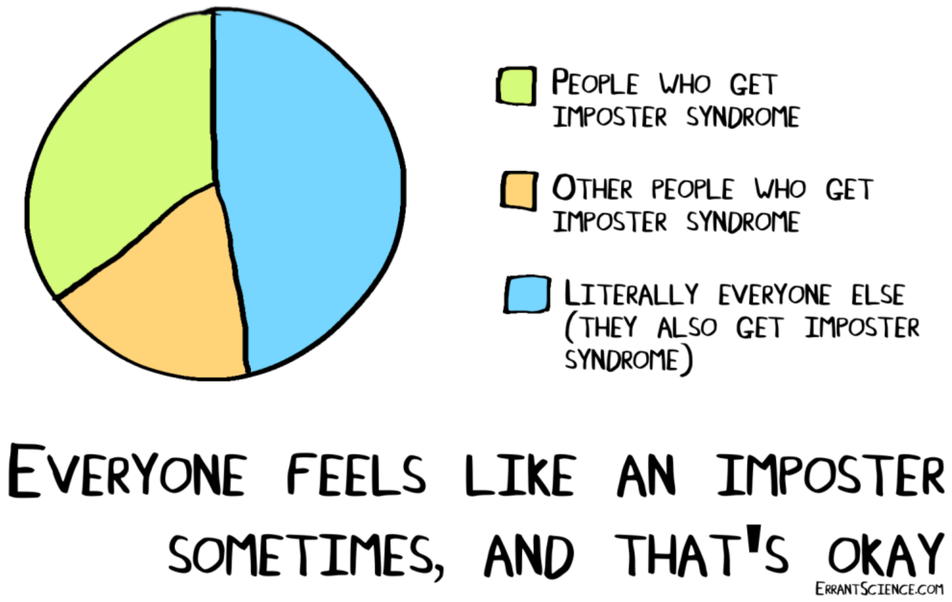
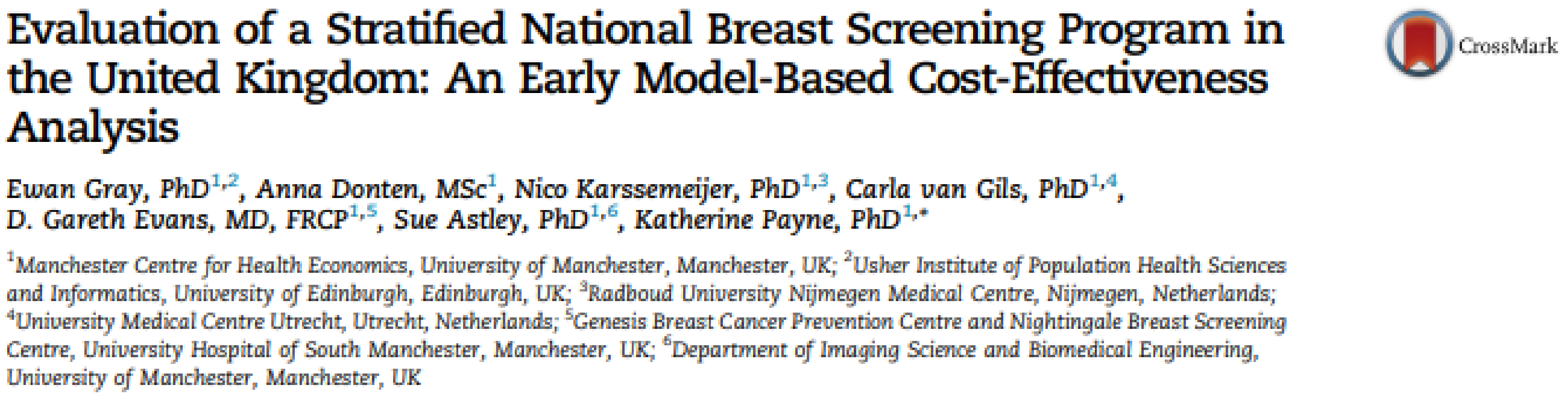
# Improving the speed of a discrete event simulation model in breast cancer screening…

Stuart Wright, PhD
stuart.j.wright@manchester.ac.uk

## ….the experience of an intermediate level R user!

- Caveats
  - Learnings from an ongoing process
  - Huge amount of work went into original code
  - Very happy to hear additional suggestions
  - I apologise in advance for my novice markdown copy and pasting!

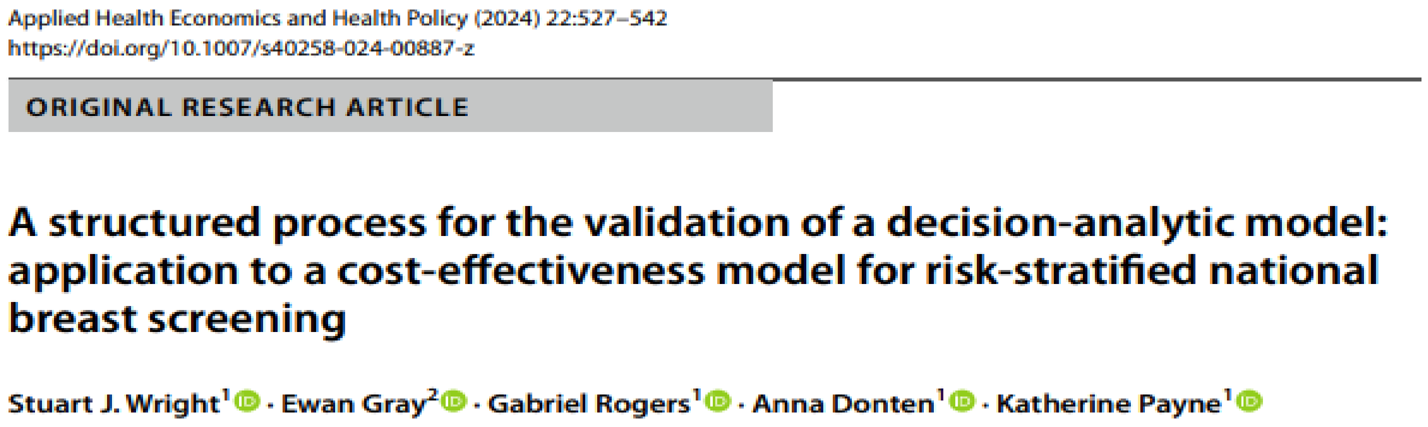

PEOPLE WHO GET IMPOSTER SYNDROME
OTHER PEOPLE WHO GET IMPOSTER SYNDROME
LITERALLY EVERYONE ELSE (THEY ALSO GET IMPOSTER SYNDROME)
EVERYONE FEELS LIKE AN IMPOSTER SOMETIMES, AND THAT'S OKAY
ERRANTSCIENCE.COM



Evaluation of a Stratified National Breast Screening Program in the United Kingdom: An Early Model-Based Cost-Effectiveness Analysis

Ewan Gray, PhD[1,2], Anna Donten, MSc[1], Nico Karssemeijer, PhD[1,3], Carla van Gils, PhD[1,4], D. Gareth Evans, MD, FRCP[1,5], Sue Astley, PhD[1,6], Katherine Payne, PhD[1,*]

[1]Manchester Centre for Health Economics, University of Manchester, Manchester, UK; [2]Usher Institute of Population Health Sciences and Informatics, University of Edinburgh, Edinburgh, UK; [3]Radboud University Nijmegen Medical Centre, Nijmegen, Netherlands; [4]University Medical Centre Utrecht, Utrecht, Netherlands; [5]Genesis Breast Cancer Prevention Centre and Nightingale Breast Screening Centre, University Hospital of South Manchester, Manchester, UK; [6]Department of Imaging Science and Biomedical Engineering, University of Manchester, Manchester, UK



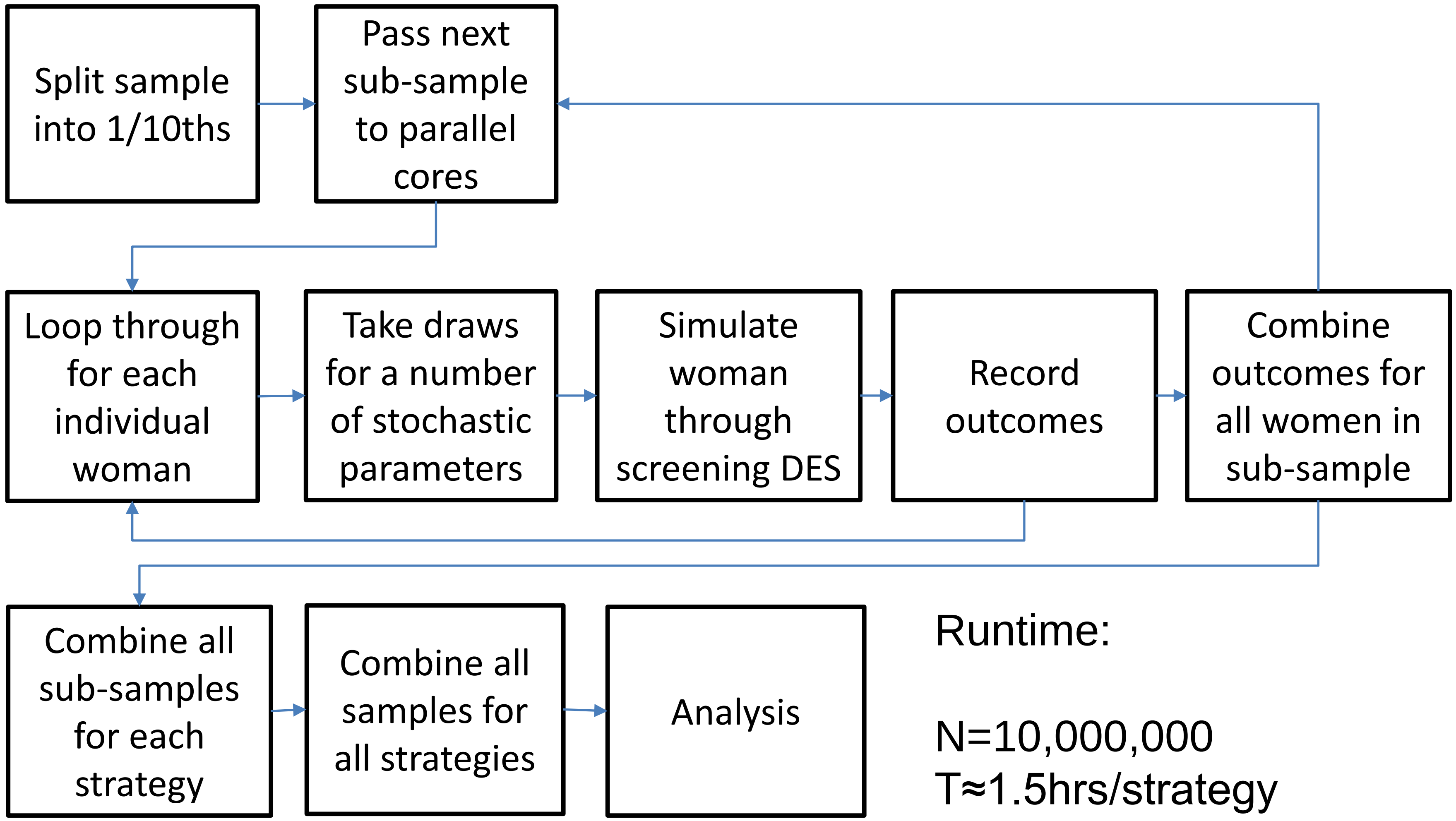
Applied Health Economics and Health Policy (2024) 22:527–542
https://doi.org/10.1007/s40258-024-00887-z

ORIGINAL RESEARCH ARTICLE

A structured process for the validation of a decision-analytic model: application to a cost-effectiveness model for risk-stratified national breast screening

Stuart J. Wright[1] · Ewan Gray[2] · Gabriel Rogers[1] · Anna Donten[1] · Katherine Payne[1]

# Rough Outline of Original Structure


Split sample into 1/10ths
Pass next sub-sample to parallel cores
Loop through for each individual woman
Take draws for a number of stochastic parameters
Simulate woman through screening DES
Record outcomes
Combine outcomes for all women in sub-sample
Combine all sub-samples for each strategy
Combine all samples for all strategies
Analysis

Runtime:

N=10,000,000
T≈1.5hrs/strategy

## Key Approaches to Speeding Up a Model

- Improve model code -> increase speed per run
- Reduce unnecessary variance -> reduce total N
- Improve computational power -> increase speed per run/improve parallelisation

## Improve Model Code

- Everything in a loop is done i times

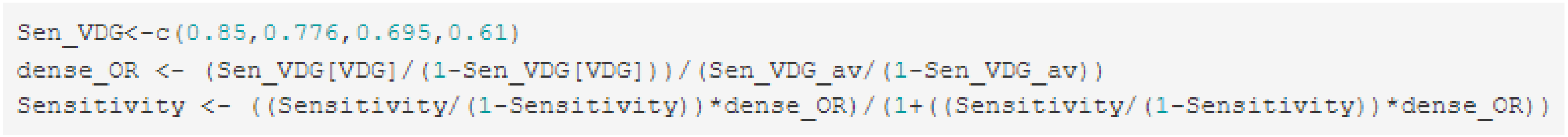
- Check for redundant code and where possible take these things out of a loop:
  - Defining new parameters
  - If statements (particularly if complex)
  - Objects that increase in size

## Example: Function to see if cancer detected

-Function is called at every screening event

-That’s around 4-5 times per individual

Don’t need to define this vector 40mil times!



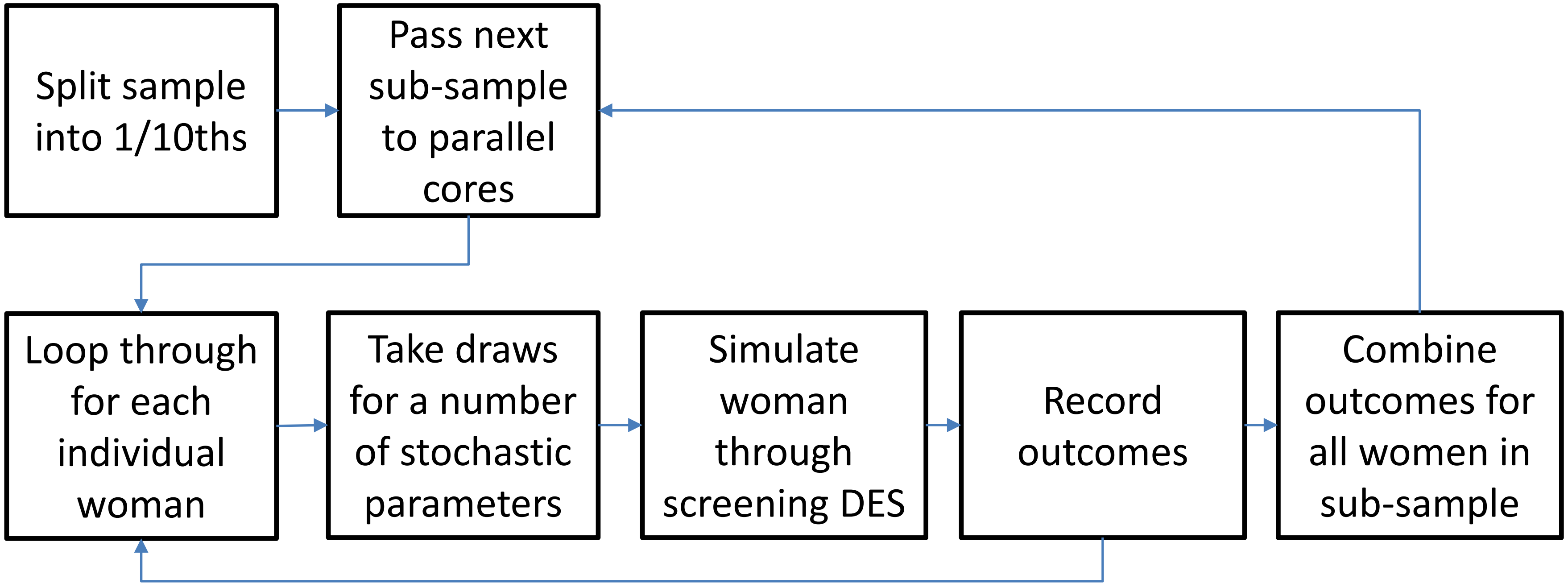
```
Sen_VDG<-c(0.85,0.776,0.695,0.61)
dense_OR <-  (Sen_VDG[VDG]/(1-Sen_VDG[VDG]))/(Sen_VDG_av/(1-Sen_VDG_av))
Sensitivity <-  ((Sensitivity/(1-Sensitivity))*dense_OR)/(1+((Sensitivity/(1-Sensitivity))*dense_OR))
```

## The Big One!

Pre-create a sample (data.frame) of individuals with pre-drawn stochastic parameters

Iterate over each row of the data.frame

# The Problem



Split sample into 1/10ths
Pass next sub-sample to parallel cores
Loop through for each individual woman
Take draws for a number of stochastic parameters
Simulate woman through screening DES
Record outcomes
Combine outcomes for all women in sub-sample

A: We have to do multiple ***individual draws from parameter distributions*** for each woman

B: Each sample of women for each strategy is ***different*** – they vary in ways that won't affect the results

## A Useful Rule for Stochastic Models (inc PSA)

It is much quicker to take X random draws from a distribution than to do individually draw from the distribution X times

```r
library(tictoc)
probdraw<-numeric(length(1000000))
tic()
for (i in 1:1000000){
probdraw[i]<-rbeta(1,50,50)
}
toc()
```

```
## 1.65 sec elapsed
```

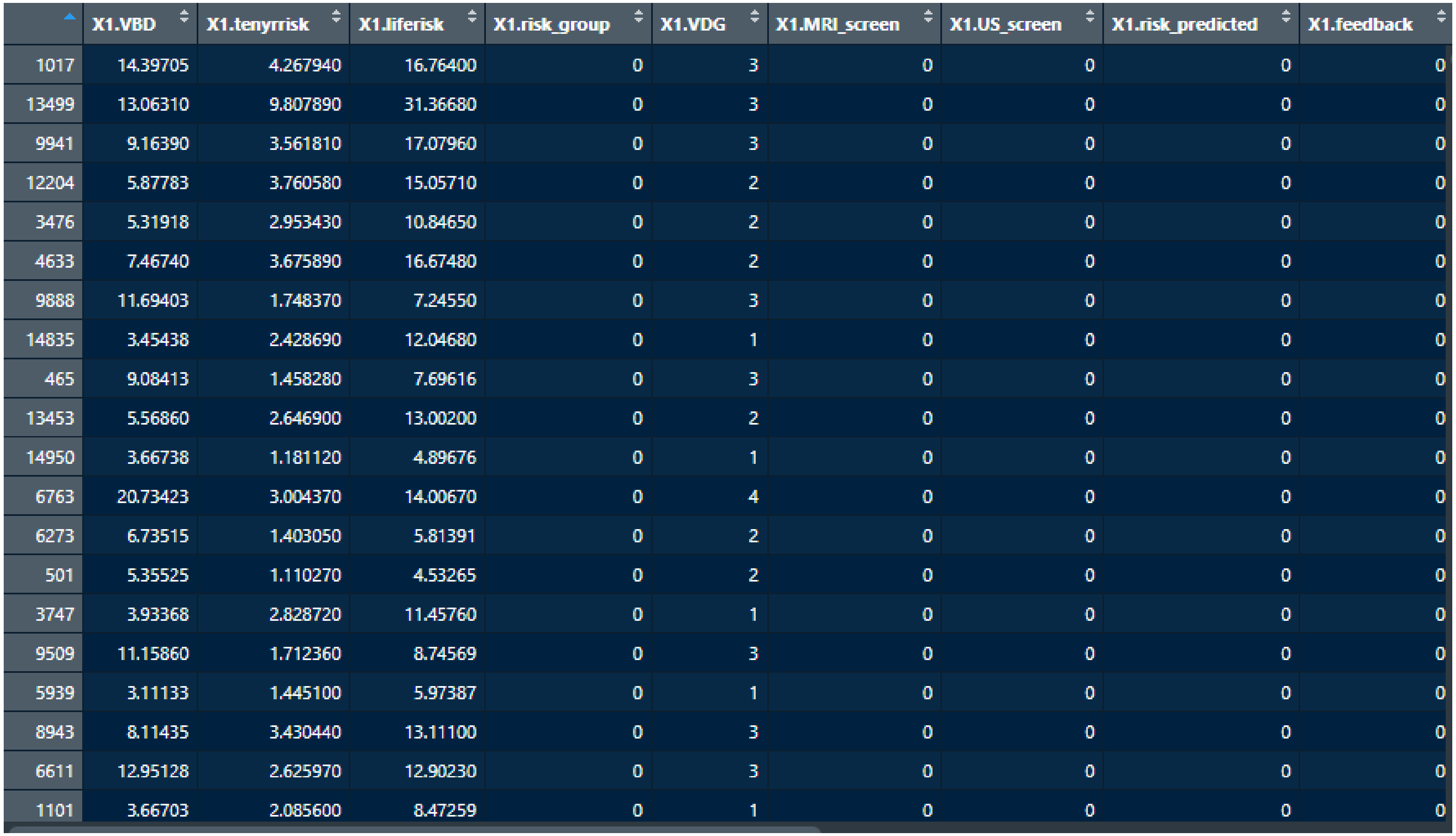
Vector of probabilities is different every time code is run

```r
probdraw<-numeric(length(1000000))
fairprobdraw<-numeric(length(1000000))
tic()
probdraw<-rbeta(1000000,50,50)
for (i in 1:1000000){
  fairprobdraw[i]<-probdraw[i]
}
toc()
```

```
## 0.33 sec elapsed
```

Vector of probabilities is the same every time code is run

BONUS!
By setting the seed, reviewers should be able to reproduce exact results



| | X1.VBD | X1.tenyrrisk | X1.liferisk | X1.risk_group | X1.VDG | X1.MRI_screen | X1.US_screen | X1.risk_predicted | X1.feedback |
|---|---|---|---|---|---|---|---|---|---|
| 1017 | 14.39705 | 4.267940 | 16.76400 | 0 | 3 | 0 | 0 | 0 | 0 |
| 13499 | 13.06310 | 9.807890 | 31.36680 | 0 | 3 | 0 | 0 | 0 | 0 |
| 9941 | 9.16390 | 3.561810 | 17.07960 | 0 | 3 | 0 | 0 | 0 | 0 |
| 12204 | 5.87783 | 3.760580 | 15.05710 | 0 | 2 | 0 | 0 | 0 | 0 |
| 3476 | 5.31918 | 2.953430 | 10.84650 | 0 | 2 | 0 | 0 | 0 | 0 |
| 4633 | 7.46740 | 3.675890 | 16.67480 | 0 | 2 | 0 | 0 | 0 | 0 |
| 9888 | 11.69403 | 1.748370 | 7.24550 | 0 | 3 | 0 | 0 | 0 | 0 |
| 14835 | 3.45438 | 2.428690 | 12.04680 | 0 | 1 | 0 | 0 | 0 | 0 |
| 465 | 9.08413 | 1.458280 | 7.69616 | 0 | 3 | 0 | 0 | 0 | 0 |
| 13453 | 5.56860 | 2.646900 | 13.00200 | 0 | 2 | 0 | 0 | 0 | 0 |
| 14950 | 3.66738 | 1.181120 | 4.89676 | 0 | 1 | 0 | 0 | 0 | 0 |
| 6763 | 20.73423 | 3.004370 | 14.00670 | 0 | 4 | 0 | 0 | 0 | 0 |
| 6273 | 6.73515 | 1.403050 | 5.81391 | 0 | 2 | 0 | 0 | 0 | 0 |
| 501 | 5.35525 | 1.110270 | 4.53265 | 0 | 2 | 0 | 0 | 0 | 0 |
| 3747 | 3.93368 | 2.828720 | 11.45760 | 0 | 1 | 0 | 0 | 0 | 0 |
| 9509 | 11.15860 | 1.712360 | 8.74569 | 0 | 3 | 0 | 0 | 0 | 0 |
| 5939 | 3.11133 | 1.445100 | 5.97387 | 0 | 1 | 0 | 0 | 0 | 0 |
| 8943 | 8.11435 | 3.430440 | 13.11100 | 0 | 3 | 0 | 0 | 0 | 0 |
| 6611 | 12.95128 | 2.625970 | 12.90230 | 0 | 3 | 0 | 0 | 0 | 0 |
| 1101 | 3.66703 | 2.085600 | 8.47259 | 0 | 1 | 0 | 0 | 0 | 0 |

The parameters above this box are all generated with if statements – now outside the loop

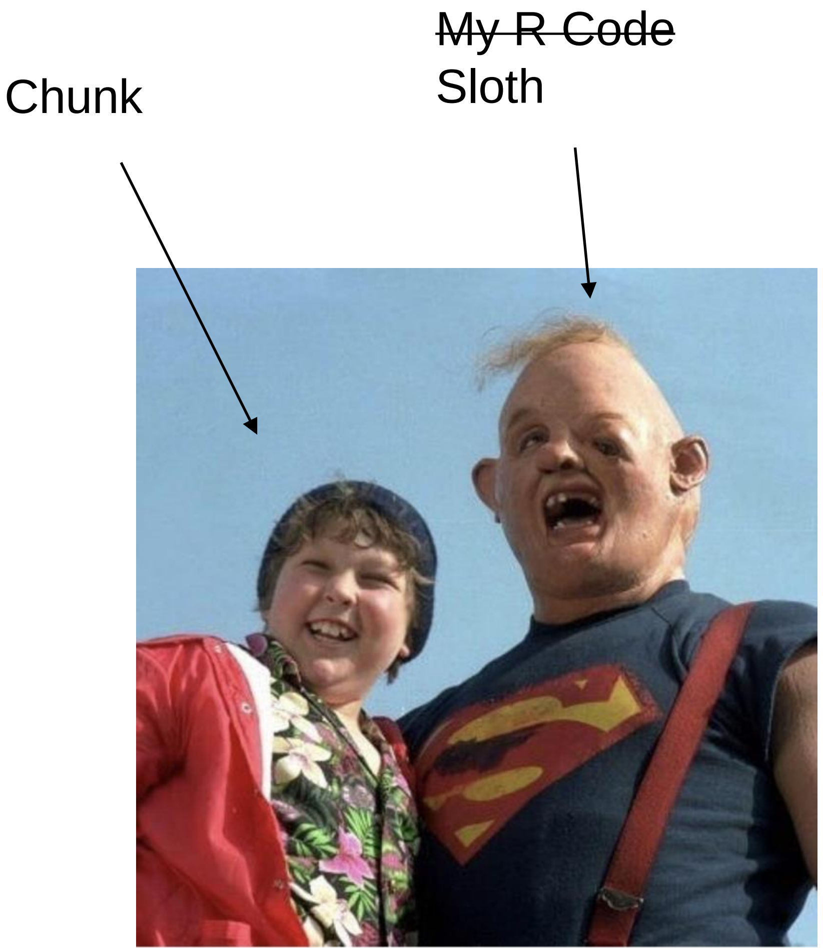
# BUT…

1. R holds data in RAM – making a huge data.frame means it will run slowly or not at all

2. Iterating over the rows of the frame rather than over a simple sequence of numbers requires a bit of a tweak

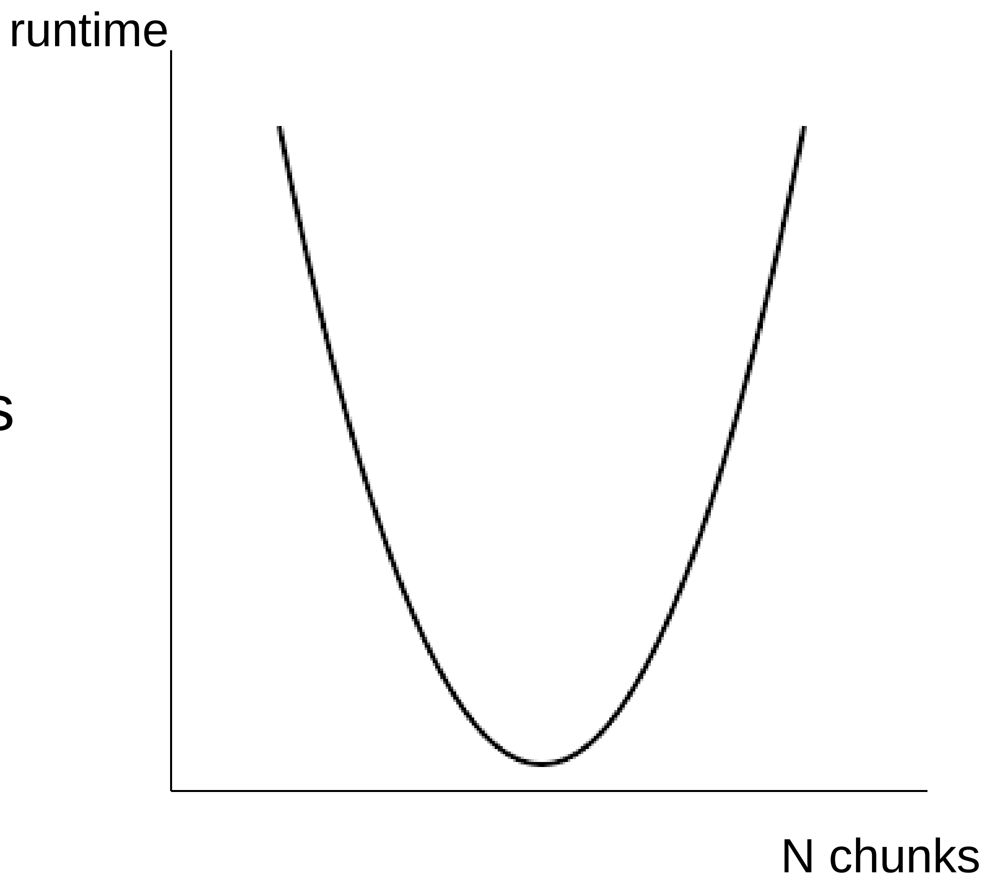
# Solution 1: Chunking

- Split the big data.frame into chunks of smaller ones
- Save the chunks to drive
- Remove the data.frames from memory
- In the outer loop load one of the chunks
- Run through model
- Save results
- Load next chunk…



# A note of caution on chunks

- More chunks means less data in memory
- BUT running each chunk requires two slow processes
  - Loading data in R
  - Initialising the parallelisation
- So need to find the optimal number of chunks which depends on N



## Solution 2: Parallelising a data.frame

```
results <- foreach(i=n/chunks,.combine = 'rbind',.packages = c('MASS','dqrng','tidyverse')) %dopar% {
```

-This is called an iterator

-Essentially R makes a sequential vector with the numbers 1:n/chunks

-PROBLEM: each core gets the same rows (1:n/chunks)

-Actually need each core to work on separate set of rows

Tip: supposedly a faster random number generator than in base R

# Different iterators

- Using the iterators package you can iterate over different types of objects

Chunk of main data.frame

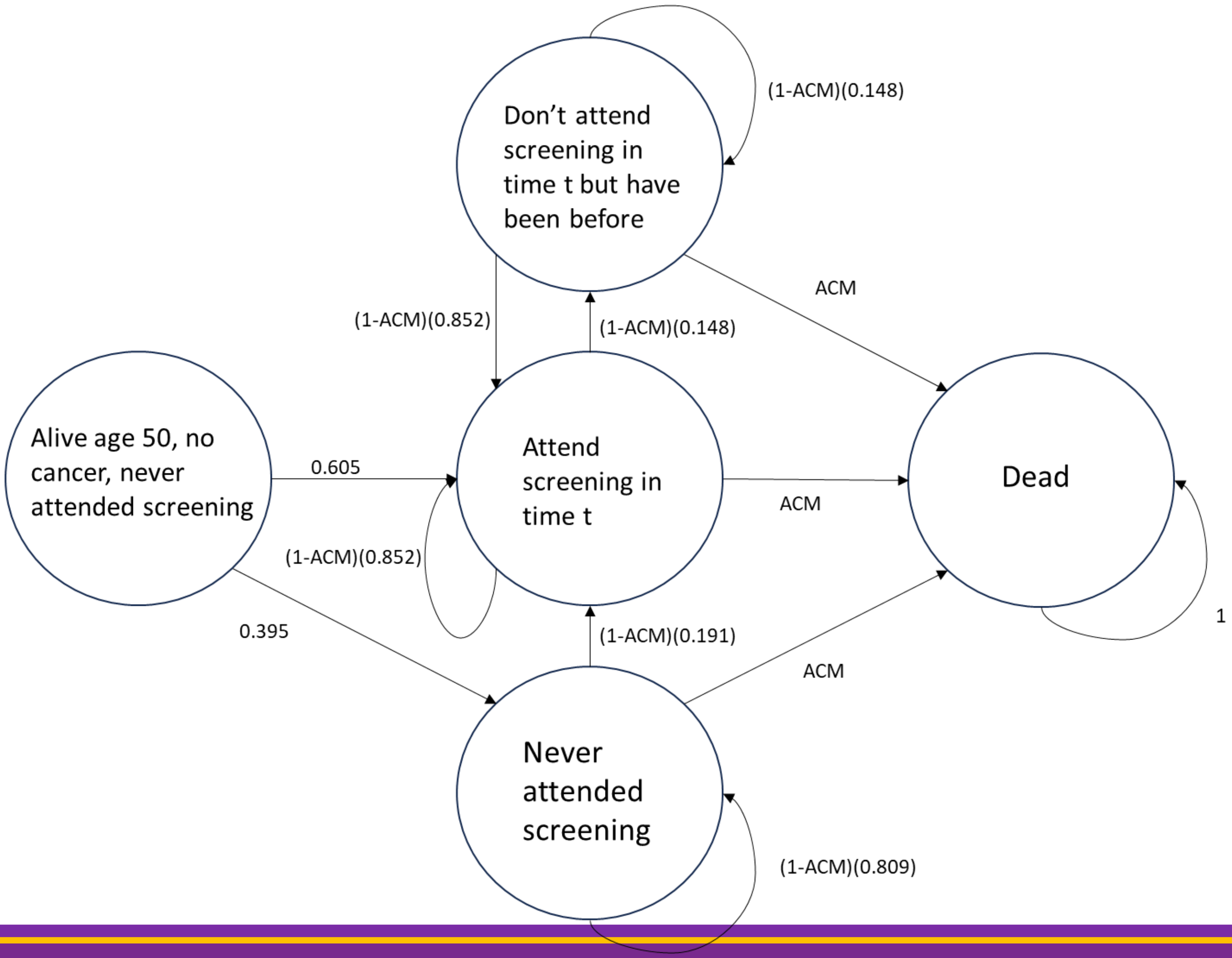
```r
itx<-iter(splitsample,by="row")
results <- foreach(i=itx,.combine = 'rbind',.packages = c('MASS','dqrng','tidyverse')) %dopar% {
```

- This one line took me about a 6 months to figure out

## Results

| | Original Model | MANC-RISK-SCREEN |
|---|---|---|
| N per strategy | 10,000,000 | 3,000,000 |
| Run time per strategy (mins) | ~90 | 10-15 depending on complexity |

## ~87% of women in the model don't get cancer



## Screening models: pushing the limits of R?

- Building a cervical cancer screening model
- Infectious disease model of HPV transmission
- Dynamic effect of vaccination over time
- Pre-cancer and cancer growth model
- Very complex screening and triage strategies
- Considering using Python for data generation, R for analysis

Thank You!