# Deterministic Bayesian inference methods for the Naomi model

PhD Student Presentations & Networking Event

Adam Howes

Imperial College London

April 2023

# The Naomi model

- Naomi is a complicated spatio-temporal evidence synthesis model
- Used by countries to produce HIV estimates in a yearly process supported by UNAIDS
- Fast inference is important to allow for interactive review and development of estimates
- Inference for Naomi is currently conducted using Template Model Builder (`TMB`) (Kristensen et al. 2016)


Figure 1: A supermodel

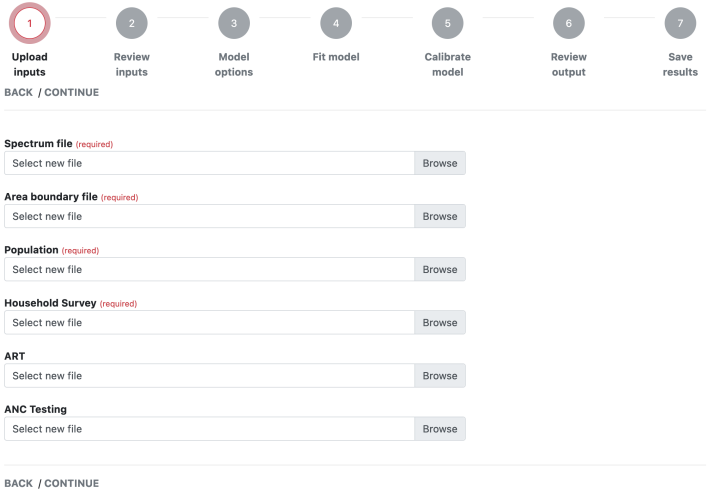


Figure 2: Example of the user interface from https://naomi.unaids.org/

# Why do we use TMB

1. It runs quickly
2. It is flexible enough to be compatible with the model
3. We don't have better viable options

# What problem we are trying to solve

- Ideally we want exact Bayesian inference: compute the posterior distribution of the parameters of the model given the data
- Computationally this amounts to solving a difficult integral
- We can't solve this, but we can give approximate answers

> Goal: approximate this integral for Naomi better than `TMB` does, without taking too long. In doing so, more accurately reflect uncertainty over hyperparameters

# Two deterministic methods

- We use two deterministic[1] methods to approximate our integral

1. The Laplace approximation
2. Quadrature

[1] In contrast to the most famous approximate Bayesian inference method, Markov chain Monte Carlo, which is fundamentally stochastic.

# The Laplace approximation

- If you pretend the posterior distribution is a Gaussian, computation is easy

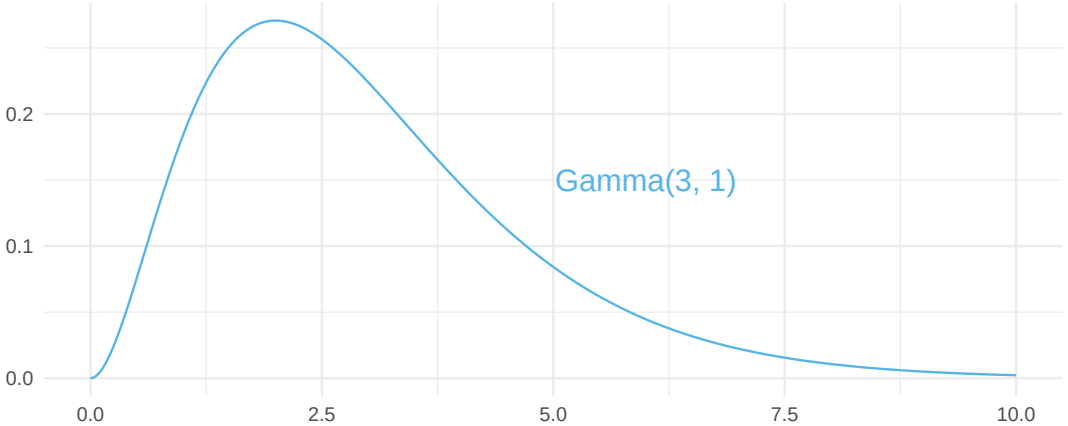


Figure 3: A Gamma prior with $a = 3$ and $b = 1$.

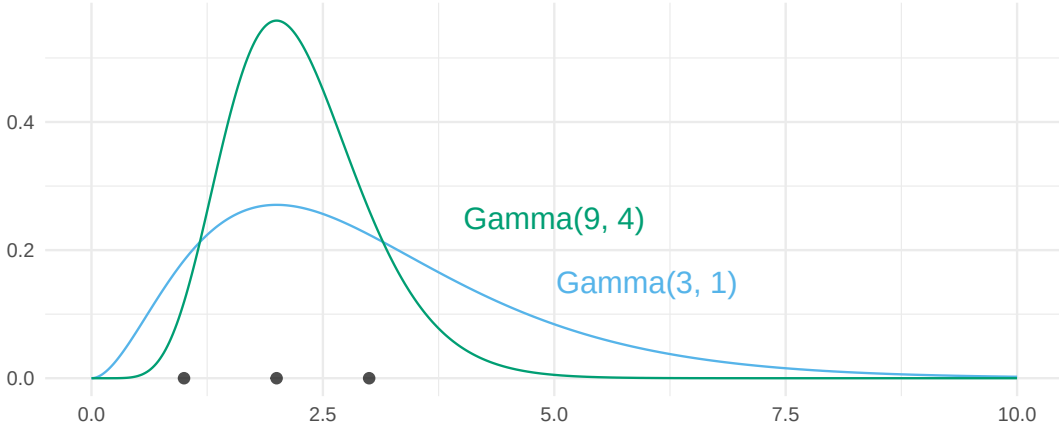


Figure 4: Draw 3 points from Poisson(3), then compute the posterior.

```r
fn <- function(x) dgamma(x, a + sum(y), b + length(y), log = TRUE)

# Here we are using numerical derivatives
ff <- list(
  fn = fn,
  gr = function(x) numDeriv::grad(fn, x),
  he = function(x) numDeriv::hessian(fn, x)
)

opt_bfgs <- aghq::optimize_theta(
  ff, 1, control = aghq::default_control(method = "BFGS")
)
```

# Laplace approximation

```
laplace <- posterior +
  stat_function(
    data = data.frame(x = c(0, 10)),
    aes(x),
    fun = dnorm,
    n = 500,
    args = list(mean = opt_bfgs$mode, sd = sqrt(1 / opt_bfgs$hessian)),
    col = cbpalette[3]
  )
```

# Laplace approximation

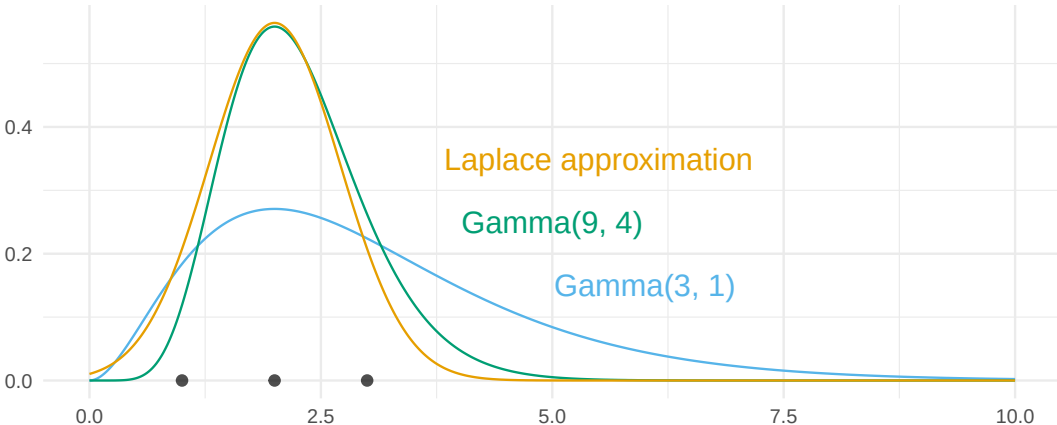


Figure 5: The Laplace approximation in this case is good near the mode but not in the tails.

# Computation of the Laplace approximation

- This computation was simple, and involved

1. Optimising a function
2. Taking the mode and the Hessian at the mode

# The marginal Laplace approximation

- If we don't want to pretend the whole posterior distribution is Gaussian, another option is to pretend some of its marginals are
- This is how `TMB` works: it's up to the user to choose which parameters should be Gaussian using the `random` option

# Which parameters should we treat as Gaussian?

- In spatio-temporal statistics we have data indexed by space and time
- We use random effects also indexed by space and time to model this data
- Spatio-temporal fields can be big
- Willing to make assumptions about how things vary over spacetime

$\implies$ treat the latent field parameters as Gaussian! That's the majority of the integral done.

# What about the hyperparameters?

- `TMB` uses optimisation to find the hyperparameters which maximise the marginal Laplace approximation
- This is the "outer" optimisation loop, where the "inner" is for computation of the Gaussian distribution

> Inference for the latent field is based on a single value of the hyperparameters (the mode) – so called empirical Bayes $\implies$ no uncertainty in the hyperparameters taken into account! How can you sleep at night.

# Quadrature

- This brings us to our other method for solving integrals deterministically
- Say we have a function, then quadrature has two ingredients

1. Nodes: points to evaluate the function at
2. Weights: importance of function evaluation at that point

- You evaluate the function at the nodes, then do a weighted sum to calculate your integral

# Trapezoid rule example

- Let's compute $\int_0^\pi \sin(x)\mathrm{d}x = 2$ using quadrature

```
trapezoid_rule <- function(x, spacing) {
  # Assumes nodes are evenly spaced
  w <- rep(spacing, length(x)) # Weights given by space between nodes
  w[1] <- w[1] / 2 # Apart from the first which is halved
  w[length(x)] <- w[length(x)] / 2 # And the last, also halved
  sum(w * x) # Compute the weighted sum
}
```

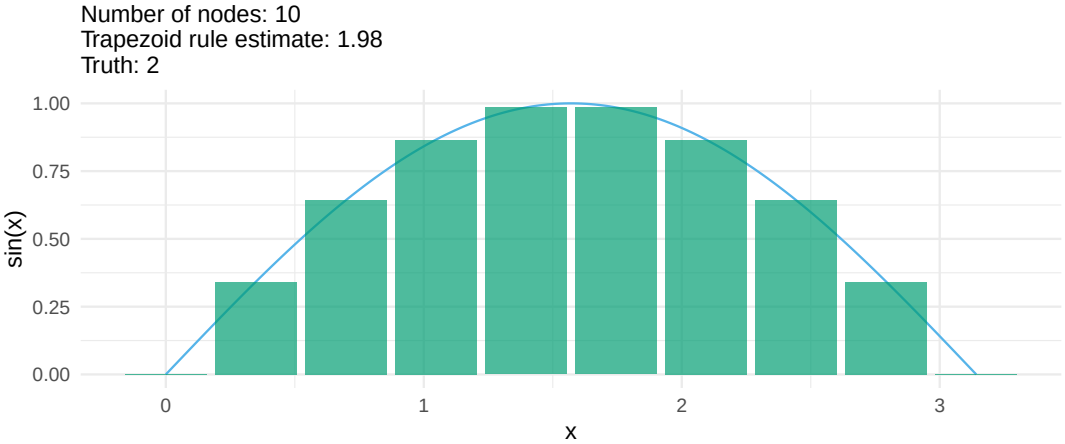


Figure 6: With 10 nodes it's 0.02 off.

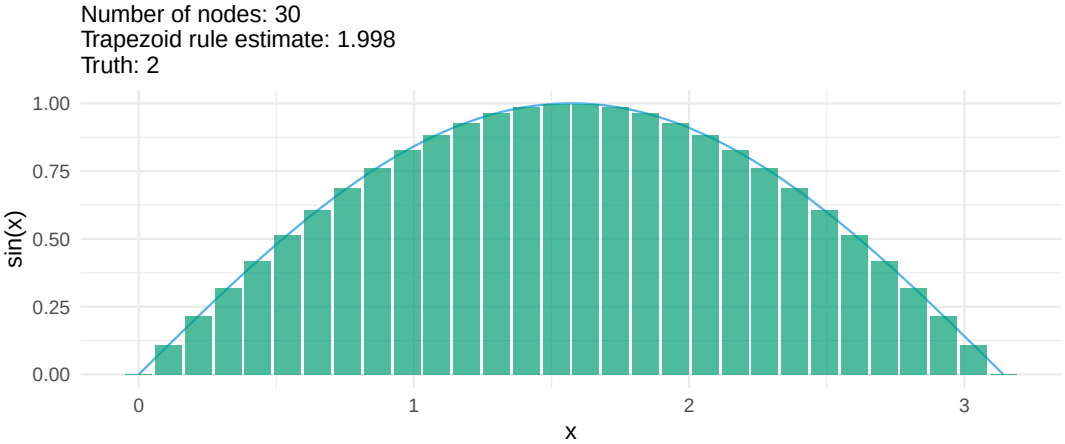


Figure 7: With 30 nodes it's 0.002 off.

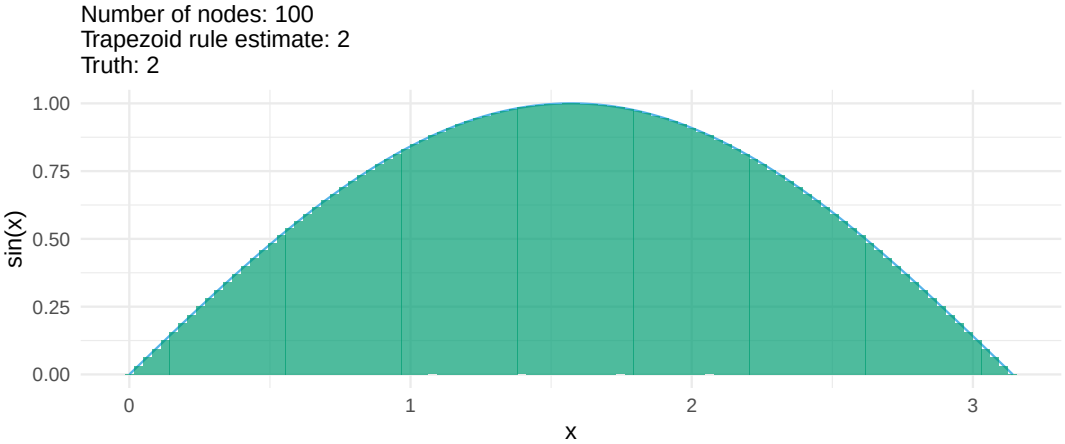


Figure 8: With 100 nodes it's pretty much correct.

# Adaptive Gauss-Hermite quadrature

- Gauss-Hermite quadrature is a method for picking nodes and weights based on the theory of polynomial interpolation
- It works especially well for statistical problems where the integrand looks like something multiplied by a Gaussian distribution
- The adaptive part means the nodes and weights are changed depending on the integrand – this makes sense, especially when the integrand is also a function of the data
- Implemented by the `aghq` package (Stringer 2021)

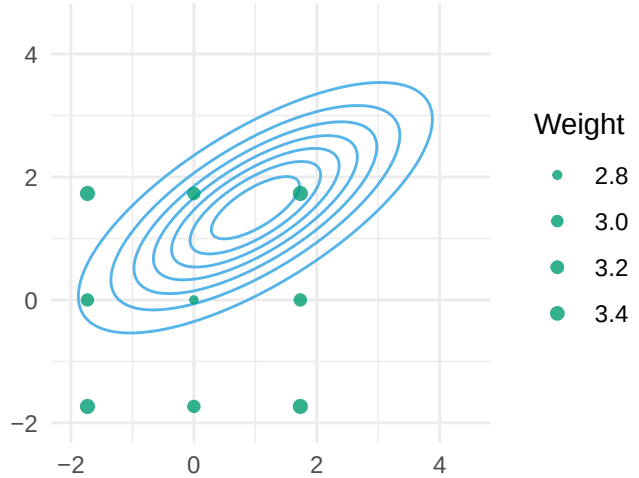


Figure 9: Unadapted points in two dimensions with $k = 3$.

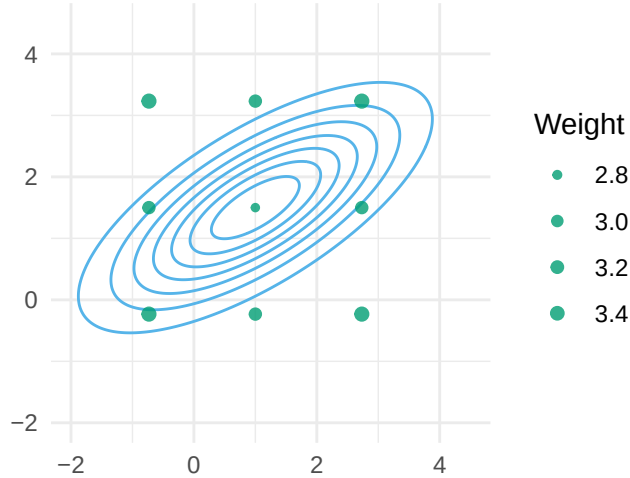


Figure 10: Add the mean $z + \hat{\theta}$.

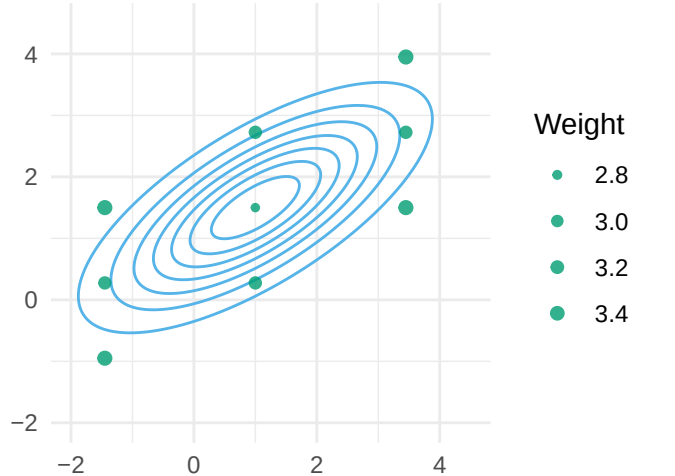


Figure 11: First option: rotate by the lower Cholesky $Lz + \hat{\theta}$.

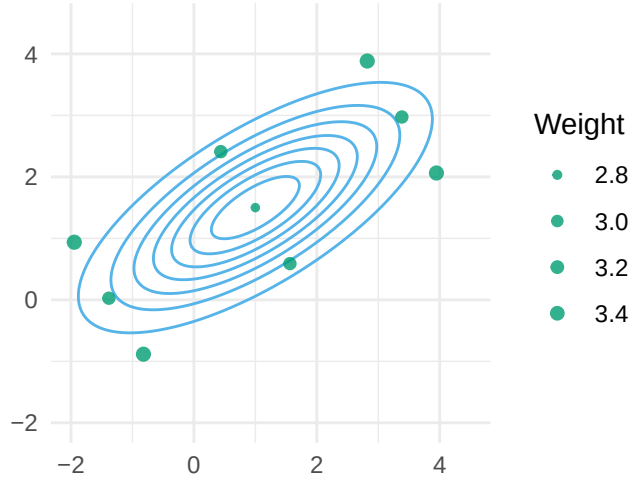


Figure 12: Second option: rotate using the eigendecomposition $E\Lambda^{1/2}z + \hat{\theta}$.

# The plan?

- Write and implement an algorithm for fast approximate Bayesian inference using the Laplace approximation and quadrature for the Naomi model
- Use `TMB` for writing the model in C++, and implementing the Laplace approximation via automatic differentitation
- Use adaptive Gauss-Hermite quadrature to integrate the hyperparameters


Figure 13: One of Canada's top engineering schools (though ungraded)

# Long prophecised

> *My main comment is that several aspects of the computational machineery that is presented by Rue and his colleagues* ***could benefit from the use of a numerical technique known as automatic differentiation (AD)*** *... By the use of AD one could obtain a system that is automatic from a user's perspective... the benefit would be a fast, flexible and easy-to-use system for doing Bayesian analysis in models with Gaussian latent variables*

– Hans J. Skaug (coauthor of `TMB`), RSS discussion of Rue, Martino, and Chopin (2009)

# One challenge

- For Malawi, Naomi has 24 hyperparameters: too many for a dense grid
- One proposed solution is to use principal components analysis (PCA) and keep only the first $s < 24$ dimensions
- We will use $k = 3$ points per dimension ($k = 2$ doesn't include the mode)

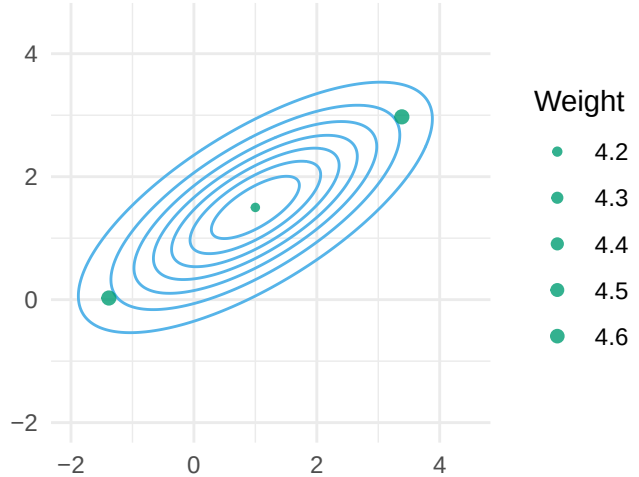


Figure 14: Illustration of what PCA-AGHQ looks like for the 2D example, keeping only the first principal component, $s = 1$.

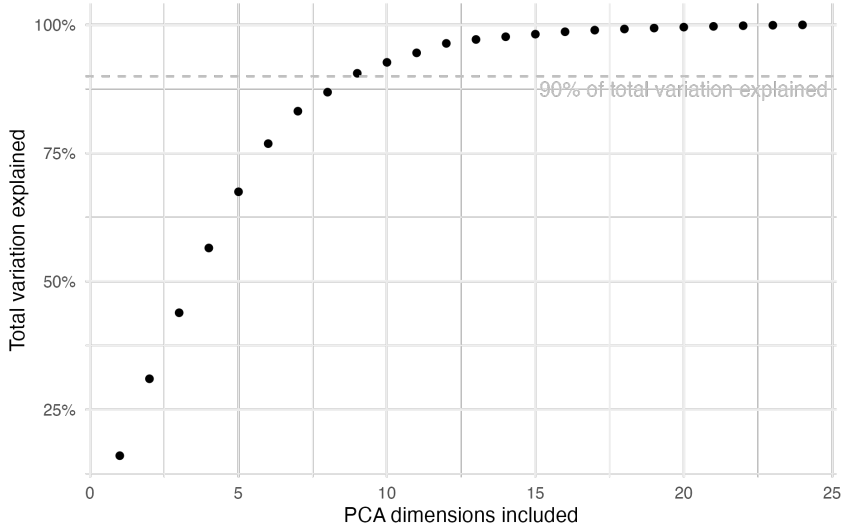


Figure 15: This Scree plot suggests 10 or so dimensions is enough. We use $s = 8$. Avoiding long computation times is still important.

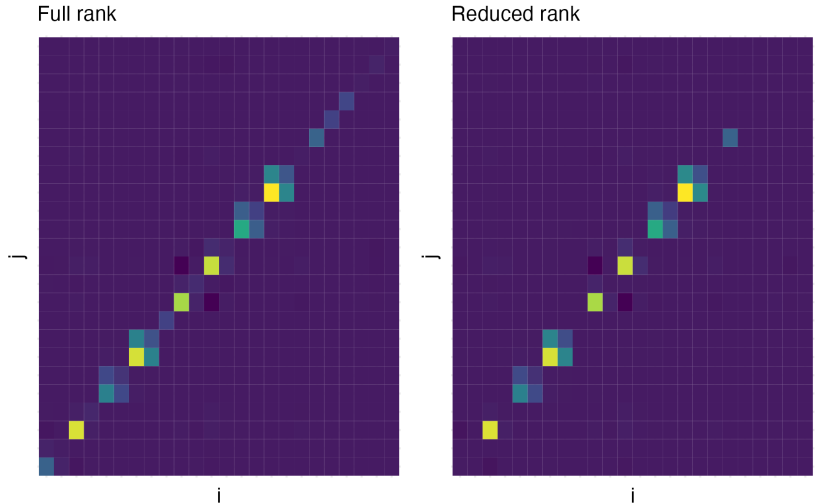


Figure 16: With 8 dimenions, the covariance matrix is accurately reproduced.

# Comparison of posterior distributions

- We run HMC for as long as it takes to get a good answer (3 days)
- Compare output of `TMB` with `aghq`: which is more similar to HMC?

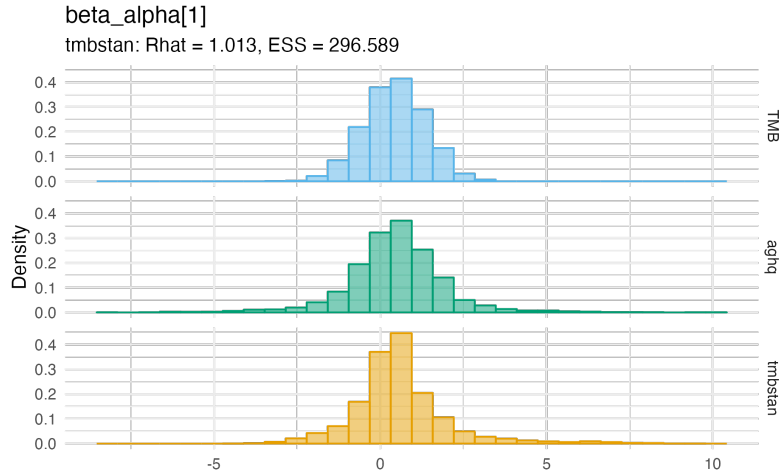


Figure 17: Histogram of sample draws from each method for one latent field parameter. Which of `TMB` and `aghq` is closer to `tmbstan`? Hard to say.

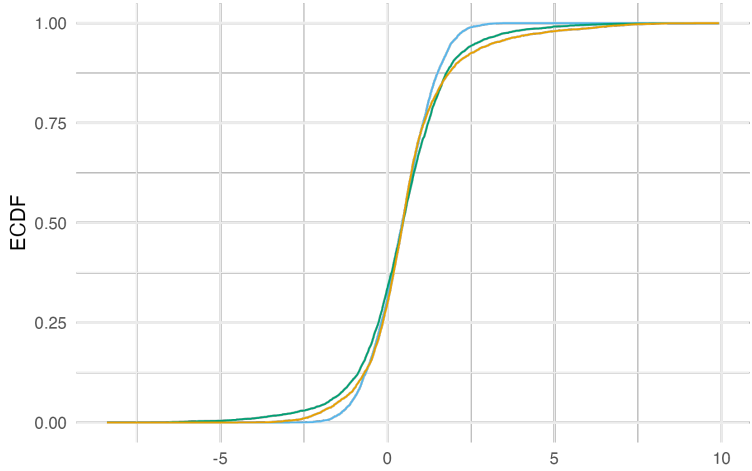


Figure 18: The empirical cumulative distribution is slightly more enlightening.

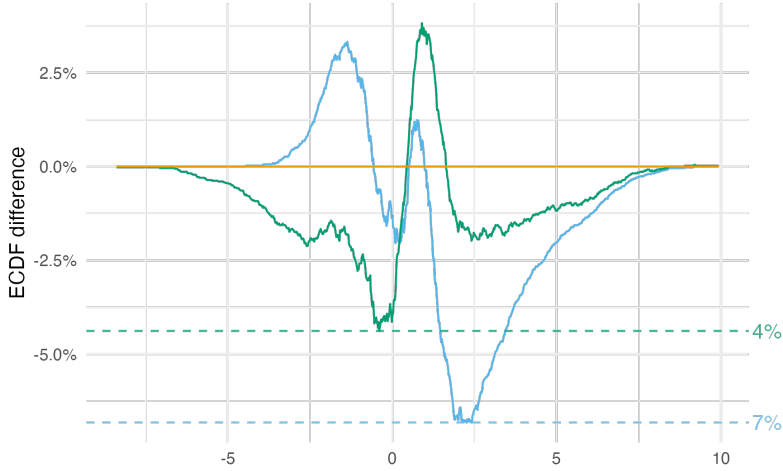


Figure 19: The Kolmogorov-Smirnov test statistic is the maximum difference between ECDFs.

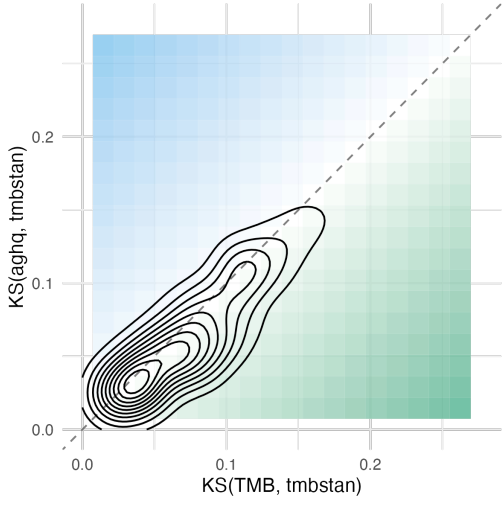


Figure 20: Generating the KS test statistic for all latent field parameters, on average `aghq` does better.

# Conclusions

- Developed an approximate Bayesian inference method more accurate than `TMB` and faster than MCMC – we're on the Pareto frontier!
- Scaled hyperparameter grid to $>$20 dimesions using PCA
- Relatively straightforward implementation using `aghq` R package, compatible with any model that has a `TMB` template

# Future work

- More comprehensive inference comparison, particularly of model outputs
  - Maximum mean discrepancy
  - Pareto-smoothed importance sampling
  - Simulation-based calibration
- Extension to Laplace marginals via Wood (2020)
  - May be more accurate, but have to think about computational costs

# Thanks for listening!

- Working on a paper based on this work called "Fast approximate Bayesian inference for small-area estimation of HIV indicators using the Naomi model" joint with Alex Stringer (Waterloo), Seth Flaxman (Oxford), Jeff Eaton (Imperial)
- Code and notebooks for this project are available at athowes.github.io/elgm-inf

Let me know if you'd be up for being an early reader!

# Risk group retrospective

April 19, 2023
Adam Howes

**Estimating risk group proportions: informal discussion**

A summary, some potentially useful takeaways, and reflections on the paper "Spatio-temporal estimates of HIV risk group proportions for adolescent girls and young women across 13 priority countries in sub-Saharan Africa"

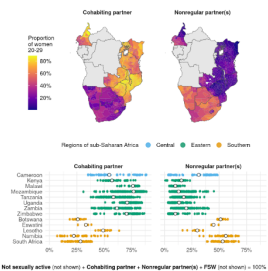


Figure 21: Available at athowes.github.io

# References I


Kristensen, Kasper, Anders Nielsen, Casper W Berg, Hans Skaug, and Bradley M Bell. 2016. "TMB: Automatic Differentiation and Laplace Approximation." *Journal of Statistical Software* 70: 1–21.

Rue, Håvard, Sara Martino, and Nicolas Chopin. 2009. "Approximate Bayesian inference for latent Gaussian models by using integrated nested Laplace approximations." *Journal of the Royal Statistical Society: Series b (Statistical Methodology)* 71 (2): 319–92.

Stringer, Alex. 2021. "Implementing Approximate Bayesian Inference using Adaptive Quadrature: the aghq Package." https://arxiv.org/abs/2101.04468.

Wood, Simon N. 2020. "Simplified Integrated Nested Laplace Approximation." *Biometrika* 107 (1): 223–30.