

User guide for the BMat model

This is a brief guide to running the BMat R scripts to produce the UN MMEIG maternal mortality estimates¹

In order to run the scripts basic knowledge of R programming language is necessary. If needed, we recommend 'free-introduction-to-r' course provided on datacamp.com for quick introduction to R. WHO/SRH is unable to provide support with basic R programming language.

Caveat

- The output from the runs may differ slightly from the published figures due to run-to-run variation.

the BMat R files are bundled within the 'bmat2019_code.Rproj' project file

Upon clicking the 'bmat2019_code.Rproj' that is included with the BMat codes, you will see the different folders and files in the project

- inputs
- Main.R
- output
- R

the input folder contains the following files :

- BMat2019_datainputs.csv which is the input data for the model
- jags_model_file.txt which is the bayesian model specification for JAGS
- meta.rds which contains program meta data such as country ISO codes, live births e.t.c

¹ Trends in maternal mortality 2000 to 2017: estimates by WHO, UNICEF, UNFPA, World Bank Group and the United Nations Population Division. Geneva: World Health Organization; 2019.

the output folder

This folder holds the different run outputs specified in main.R code

the R folder

This folder contains different R scripts:

- 2a_getjagsdata.R
- 2c_mcmc.R
- 3_getcountryresults.R
- misc_functions.R

the 'main.R' script

The simulation runs are made using the 'main.R' code which is described in details below

In the first line of the code you choose the model output runname, for example in the code below "test" is chosen as the run name

- a folder will be created on the outputs folder with the chosen runname

```
# output will be saved in folder output/runname  
runname <- "test" # choose your runname
```

In the next lines, different packages are loaded

```
# Load Libraries  
library(rjags)  
library(msm)  
library(mvtnorm)  
library(R2jags)  
library(foreign)  
library(tidyverse)  
library(readxl)  
library(R.utils)  
library(truncnorm)
```

In lines 27-28 of the code ,different r scripts in the 'R' folder are loaded and run

```
# Load Libraries  
# source scripts in R subfolder with functions  
Rfiles <- list.files(file.path(paste0(getwd(), "/R/")), ".R")  
Rfiles <- Rfiles[grepl(".R", Rfiles)]  
sapply(paste0(paste0(getwd(), "/R/"), Rfiles), source)
```

In line 34 the directory in output is created

```
# create output folder and store relevant info in it
output.dir <- MakeDirs(runname)
```

In line 35-48, the input data is read and the JAGS data input is created using the GetJagsData function

```
dat <- read_csv("inputs/BMat2019_datainputs.csv", na = "NA",
               # use guess_max to avoid mmr_obs and SE column turning into
               # (alternatively, specify all column types)
               guess_max = 4424)
# rename for consistency with code
datall <- dat %>%
  rename(final_pm = pm_obs, final_env = env,
         rhovr = crvs_completeness) %>%
  filter(modelinclude)
saveRDS(datall, paste0(output.dir, "/datall.rds"))
meta <- readRDS("inputs/meta.rds")
saveRDS(meta, paste0(output.dir, "/meta.rds"))
file.copy("inputs/jags_model_file.txt", paste0(output.dir, "/model.txt"),
         overwrite = TRUE)
GetJagsData(runname = runname)
```

In lines 52 to 54 you choose which kind of run to make using the RunMCMC function

- test
- quick
- long (for actual results)

```
# run model
RunMCMC(runname, runsettings = "test") # just to get test results
#RunMCMC(runname, runsettings = "quick") # 1 hour run
#RunMCMC(runname, runsettings = "long") # for actual results
```

The RunMCMC function in '2c_mcmc.R' code in the R folder has the following parameter settings

```
RunMCMC <- function(
  runname = runname,
  # mcmc SETTINGS
  runsettings,
  ## choose from ("test", "quick", "long")
  ## test is only for checking if the model runs (just a few iterations);
  # quick is for getting approximate results in an hour or so;
  # long is for getting final results
  run.on.server = TRUE # using doMC and foreach libraries
)
```

Before making the final run you can choose whether to run the code on server or not (essentially which way to run the code in parallel) - you can toggle this option using run.on.server option in RunMCMC

if you choose `run.on.server = FALSE` then it runs the `jags.parallel` functions as shown in code snippet below

```
mod <- jags.parallel(jagsdata,jags.params, model.file = paste0(output.dir,
"model.txt"),
                    inits = inits(meta =meta),
                    n.chains=n.chains,
                    n.burnin = n.burnin, n.iter= n.burnin+n.iter.perstep*N.STEPS, n.thin =
n.thin)
)
```

If you choose `run.on.server = TRUE` then it invokes the `foreach` parallel process, see code below

```
library(foreach)
library(doMC)
registerDoMC()

foreach(chainNum=ChainNums) %dopar% {
  set.seed(chainNum)
  temp <- rnorm(chainNum)
  mod <- jags(data = jagsdata,
              inits = inits(meta = meta),
              parameters.to.save = jags.params,
              n.chains = 1,
              n.iter = n.iter.perstep+n.burnin,
              n.burnin = n.burnin, n.thin = n.thin,
              model.file= paste0(output.dir, "model.txt"),
              jags.seed = 123*chainNum,
              working.directory= getwd())
}
```

You can choose the number of cores for the parallel using `registerDoMC()`, e.g to use 4 cores you would set as `registerDoMC(4)`

```
# set parallel process to use 4 cores of the machine
registerDoMC(4)
```

- the settings holds the type of run test, quick or long

in line 58-59 of the `main.R` code you choose the percentiles for the summaries

```
percentiles <- c(0.1, 0.5, 0.9)
percnames <- paste0("perc_", 100*percentiles, "%")
```

After the codes completes to run you get summaries for each country using the `GetCountryResults(runname)` function, see below

```
GetCountryResults(runname)
```

After running the `GetCountryResults(runname)` function you get different R output objects are created in the `output/runname` folder

- CI.rds
- matdeaths.cts.rds

you can load the CI.rds (see below) to get the summaries and 80% bayesian credible intervals

```
CIIs <- readRDS(paste0(output.dir, "CIIs.rds"))
names(CIIs)
CIIs$mmr.cqt
CIIs$mmr.cqt
```

The following output summaries are available within the CI.rds object

- mmr.cqt (maternal mortality ratio)
- pm.cqt (proportion of deaths among women of reproductive age due to maternal causes)
- pmmean.ct (proportion of deaths among women of reproductive age due to maternal causes mean)
- mmrmean.ct (maternal mortality ratio mean)
- mmraids.ct (aids maternal mortality ratio mean)
- mmrate.cqt (maternal mortality rate)
- ltr.cqt (life time risk of maternal death)
- aidsdeaths.ct (aids deaths)
- matdeaths.cqt (maternal deaths)
- arrscts.kx (annual rate of reduction)

Finally you can produce specific country estimates and export to csv using the function WriteCountryCsvs included in the “misc_functions.R”