Calibrate a Model of U.S. Health Insurance Markets

The objective of this documen is to construct and calibrate a model of U.S. insurance markets.

To do: get standard errors around calibration targets.

Prepare Data

```
## -- Attaching packages ---------------------------------------------------------------------------
## v ggplot2 3.2.1     v purrr   0.3.3
## v tibble  2.1.3     v dplyr   0.8.3
## v tidyr   0.8.3     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0
## -- Conflicts ------------------------------------------------------------------------------------
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
##
## Attaching package: 'rlang'
## The following objects are masked from 'package:purrr':
##
##     %@%, as_function, flatten, flatten_chr, flatten_dbl,
##     flatten_int, flatten_lgl, flatten_raw, invoke, list_along,
##     modify, prepend, splice
##
## Attaching package: 'kableExtra'
## The following object is masked from 'package:dplyr':
##
##     group_rows
##
## Attaching package: 'magrittr'
## The following object is masked from 'package:rlang':
##
##     set_names
## The following object is masked from 'package:purrr':
##
##     set_names
## The following object is masked from 'package:tidyr':
##
##     extract
## Loading required package: foreach
##
## Attaching package: 'foreach'
## The following objects are masked from 'package:purrr':
##
##     accumulate, when
```

```
## Loading required package: iterators

## Loading required package: parallel

## Loading required package: survival

##
## Attaching package: 'eha'

## The following objects are masked from 'package:flexsurv':
##
##     dgompertz, dllogis, hgompertz, Hgompertz, hllogis, Hllogis,
##     hlnorm, Hlnorm, hweibull, Hweibull, pgompertz, pllogis,
##     qgompertz, qllogis, rgompertz, rllogis

## Registered S3 method overwritten by 'pryr':
##   method      from
##   print.bytes Rcpp

##
## Attaching package: 'heemod'

## The following object is masked from 'package:rlang':
##
##     modify

## The following object is masked from 'package:purrr':
##
##     modify

## here() starts at /Users/gravesj/Dropbox/Projects/modeling-health-insurance

## Loading required package: rngWELL

## This is randtoolbox. For an overview, type 'help("randtoolbox")'.

## Loading required package: nlme

##
## Attaching package: 'nlme'

## The following object is masked from 'package:directlabels':
##
##     gapply

## The following object is masked from 'package:dplyr':
##
##     collapse

## This is mgcv 1.8-28. For overview type 'help("mgcv-package")'.

## [conflicted] Will prefer dplyr::filter over any other package

## [conflicted] Will prefer rlang::set_names over any other package

## [conflicted] Will prefer dplyr::lag over any other package

## [conflicted] Removing existing preference

## [conflicted] Will prefer rlang::set_names over any other package

## [conflicted] Removing existing preference

## [conflicted] Will prefer dplyr::filter over any other package

## [conflicted] Will prefer dplyr::count over any other package
```

```r
create_sipp_data = FALSE
if (create_sipp_data) source(here("R/read-and-tidy-SIPP-data.R"))

df_sipp_full <- read_rds(here("input/sipp/01_sipp-tidy_v1-0.rds"))

# Get survey weight (use value from first month)
df_w <-
  df_sipp_full %>%
  mutate(idnumber = id) %>%
  group_by(idnumber) %>%
  mutate(year = ifelse(swave ==1 , 2014, 2015)) %>%
  mutate(month = ifelse(swave ==1 , monthcode, monthcode+12)) %>%
  filter(month==1) %>%
  select(idnumber,weight = wpfinwgt) %>%
  ungroup()

df_sipp <-
  df_sipp_full %>%
  filter(age < 65 & age > 18) %>%
  mutate(year = ifelse(swave ==1 , 2014, 2015)) %>%
  mutate(month = ifelse(swave ==1 , monthcode, monthcode+12)) %>%
  mutate(idnumber = id) %>%
  mutate(insurance_type = factor(hicov, labels = insurance_sipp_lut)) %>%
  select(idnumber,month,insurance_type,expansion_state, sex, race, state)  %>%
  # !!!! Note expansion state is time-varying if someone moves.
  group_by(idnumber) %>%
  mutate(expansion_state = max(expansion_state)) %>%
  ungroup()  %>%
  left_join(df_w,"idnumber")

ex_ante <-
  df_sipp %>%
  filter(month==1) %>%
  group_by(insurance_type) %>%
  summarise(n = sum(weight,na.rm=TRUE)) %>%
  ungroup() %>%
  mutate(pct = n/sum(n))

ex_ante %>%
  write_rds(here("output/ex-ante-overall-population/ex-ante-distribution.rds"))


ex_post <-
  df_sipp %>%
  filter(month==13) %>%
  group_by(insurance_type) %>%
  summarise(n = sum(weight,na.rm=TRUE)) %>%
  ungroup() %>%
  mutate(pct = n/sum(n))

ex_post %>%
  write_rds(here("output/ex-ante-overall-population/ex-post-distribution.rds"))
```

```r
# CPS Coverage Targets (2016-2018); see calibration excel file in  data/CPS_calibration-targets/cps-cal
# https://www.census.gov/data/tables/time-series/demo/income-poverty/cps-hi/hi.html


lst_targets <- list(ESI = c(0.630, 0.630,0.627),
                    NG = c(0.09,0.088,0.085),
                    PUB = c(0.175,0.169,0.166),
                    UNIN = c(0.105,0.112,0.122))
# based on ACS tables for 19-64 year olds in ACS data as constructed at http://statehealthcompare.shada
lst_targets <- list(ESI = c(0.64,   0.635,  0.629,  0.624),
                    NG = c(0.082,   0.086,  0.092,  0.092),
                    PUB = c(0.155,  0.158,  0.158,  0.154),
                    UNIN = c(0.124, 0.122,  0.12,   0.131))



# Set Up the Multi-State Data
ls_ms <-
  df_sipp %>%
  mutate(insurance_type = paste0(insurance_type)) %>%
  mutate(constant = 1) %>%
  prepare_multistate_data(idvar = idnumber,
                          timevar = month,
                          statevar = insurance_type)

categories <- names(ls_ms$trans_mat)

fit_transition <- map(categories,
                      ~ (
                        fit_multistate_model(
                          df = ls_ms$df_ms %>% pluck(.x),
                          tmat = ls_ms$trans_mat %>% pluck(.x),
                          fit_type = "km",
                          ff = Surv(Tstart, Tstop, status) ~ 1,
                          idvar = idnumber,
                          prediction_vals = data.frame(constant = 1)
                        )
                      )) %>%
  set_names(categories)
```

```
## Warning: partial match of 'std' to 'std.err'

## Warning: partial match of 'std' to 'std.err'

## Warning: partial match of 'std' to 'std.err'

## Warning: partial match of 'std' to 'std.err'

## Warning: partial match of 'std' to 'std.err'

## Warning: partial match of 'std' to 'std.err'

## Warning: partial match of 'std' to 'std.err'
```

```
## Warning: partial match of 'std' to 'std.err'

## Warning: partial match of 'std' to 'std.err'

## Warning: partial match of 'std' to 'std.err'

## Warning: partial match of 'std' to 'std.err'

## Warning: partial match of 'std' to 'std.err'

## Warning: partial match of 'std' to 'std.err'

## Warning: partial match of 'std' to 'std.err'

## Warning: partial match of 'std' to 'std.err'

## Warning: partial match of 'std' to 'std.err'

## Warning: partial match of 'std' to 'std.err'

## Warning: partial match of 'std' to 'std.err'

## Warning: partial match of 'std' to 'std.err'

## Warning: partial match of 'std' to 'std.err'

## Warning: partial match of 'std' to 'std.err'

## Warning: partial match of 'std' to 'std.err'

## Warning: partial match of 'std' to 'std.err'
```

```
fit_transition_gom <- map(categories,
                          ~ (
                            fit_multistate_model(
                              df = ls_ms$df_ms %>% pluck(.x),
                              tmat = ls_ms$trans_mat %>% pluck(.x),
                              fit_type = "gompertz",
                              ff = Surv(Tstart, Tstop, status) ~ constant,
                              idvar = idnumber,
                              prediction_vals = data.frame(constant = 1)
                            )
                          )) %>%
  set_names(categories)
```

```
## Warning in seq.default(along = temp): partial argument match of 'along' to
## 'along.with'

## Warning: partial match of 'coef' to 'coefficients'

## Warning: partial match of 'coef' to 'coefficients'

## Warning in seq.default(along = temp): partial argument match of 'along' to
## 'along.with'
```

```
## Warning: partial match of 'coef' to 'coefficients'

## Warning: partial match of 'coef' to 'coefficients'

## Warning in seq.default(along = temp): partial argument match of 'along' to
## 'along.with'

## Warning: partial match of 'coef' to 'coefficients'

## Warning: partial match of 'coef' to 'coefficients'

## Warning in seq.default(along = temp): partial argument match of 'along' to
## 'along.with'

## Warning: partial match of 'coef' to 'coefficients'

## Warning: partial match of 'coef' to 'coefficients'

## Warning in seq.default(along = temp): partial argument match of 'along' to
## 'along.with'

## Warning: partial match of 'coef' to 'coefficients'

## Warning: partial match of 'coef' to 'coefficients'

## Warning in seq.default(along = temp): partial argument match of 'along' to
## 'along.with'

## Warning: partial match of 'coef' to 'coefficients'

## Warning: partial match of 'coef' to 'coefficients'

## Warning in seq.default(along = temp): partial argument match of 'along' to
## 'along.with'

## Warning: partial match of 'coef' to 'coefficients'

## Warning: partial match of 'coef' to 'coefficients'

## Warning in seq.default(along = temp): partial argument match of 'along' to
## 'along.with'

## Warning: partial match of 'coef' to 'coefficients'

## Warning: partial match of 'coef' to 'coefficients'

## Warning in seq.default(along = temp): partial argument match of 'along' to
## 'along.with'

## Warning: partial match of 'coef' to 'coefficients'

## Warning: partial match of 'coef' to 'coefficients'

## Warning in seq.default(along = temp): partial argument match of 'along' to
## 'along.with'

## Warning: partial match of 'coef' to 'coefficients'

## Warning: partial match of 'coef' to 'coefficients'

## Warning in seq.default(along = temp): partial argument match of 'along' to
## 'along.with'
```

```
## Warning: partial match of 'coef' to 'coefficients'

## Warning: partial match of 'coef' to 'coefficients'

## Warning in seq.default(along = temp): partial argument match of 'along' to
## 'along.with'

## Warning: partial match of 'coef' to 'coefficients'

## Warning: partial match of 'coef' to 'coefficients'
```

```
trans_probs <- map_df(categories,
                      ~(
                        get_cumHaz(dist = fit_transition[[1]]$fit_type ,
                                   ls_fit = fit_transition %>% pluck(.x),
                                   tt = 1:24,
                                   lut = fit_transition %>%
                                     pluck(.x) %>% pluck("lut"))  %>%
                          mssample(Haz=.,trans=ls_ms$trans_mat %>%
                                     pluck(.x),tvec=unique(.$time),clock="reset", M=1000) %>%
                          magrittr::set_names(c("time",.x,fit_transition %>% pluck(.x) %>%
                                                  pluck("lut") %>% pull(transition_type))) %>%
                          mutate(baseline = .x)
                      )) %>%
  arrange(time,baseline) %>%
  select_at(c("time","baseline",categories)) %>%
  group_by(time) %>%
  nest()

trans_probs %>%
  write_rds(here("output/ex-ante-overall-population/transition-probabilities-kaplan-meier.rds"))

trans_probs_gom <- map_df(categories,
                      ~(
                        get_cumHaz(dist = fit_transition_gom[[1]]$fit_type ,
                                   ls_fit = fit_transition_gom %>% pluck(.x),
                                   tt = 1:24,
                                   lut = fit_transition_gom %>%
                                     pluck(.x) %>% pluck("lut"))  %>%
                          mssample(Haz=.,trans=ls_ms$trans_mat %>%
                                     pluck(.x),tvec=unique(.$time),clock="reset", M=1000) %>%
                          magrittr::set_names(c("time",.x,fit_transition_gom %>% pluck(.x) %>%
                                                  pluck("lut") %>% pull(transition_type))) %>%
                          mutate(baseline = .x)
                      )) %>%
  arrange(time,baseline) %>%
  select_at(c("time","baseline",categories)) %>%
  group_by(time) %>%
  nest()

trans_probs_gom %>%
  write_rds(here("output/ex-ante-overall-population/transition-probabilities-gompertz.rds"))
```

```
p <- ex_ante$pct %>% as.matrix() %>% t()
R <- trans_probs %>%
  filter(time == 13) %>%
```

```
  pull(data) %>% pluck(1) %>%
  select(-baseline) %>%
  as.matrix()

    p_ESI_NG <- R[1,2]
    p_ESI_PUB <- R[1,3]
    p_ESI_UNIN <- R[1,4]

    p_NG_ESI <- R[2,1]
    p_NG_PUB <- R[2,3]
    p_NG_UNIN <- R[2,4]

    p_PUB_ESI <- R[3,1]
    p_PUB_NG <- R[3,2]
    p_PUB_UNIN <- R[3,4]

    p_UNIN_ESI <- R[4,1]
    p_UNIN_NG <- R[4,2]
    p_UNIN_PUB <- R[4,3]

    ev <- c(p_ESI_NG = p_ESI_NG, p_ESI_PUB = p_ESI_PUB, p_ESI_UNIN = p_ESI_UNIN,
            p_NG_ESI = p_NG_ESI, p_NG_PUB = p_NG_PUB, p_NG_UNIN = p_NG_UNIN,
            p_PUB_ESI = p_PUB_ESI, p_PUB_NG = p_PUB_NG, p_PUB_UNIN = p_PUB_UNIN,
            p_UNIN_ESI = p_UNIN_ESI, p_UNIN_NG = p_UNIN_NG, p_UNIN_PUB = p_UNIN_PUB)


run_insurance_markov <- function(v_params) {
  with(as.list(v_params), {
    n_t <- 4
    v_n  <- c("ESI", "NG", "PUB", "UNIN") # the 4 states of the model: ESI, non-group, public, uninsure
    n_s <- length(v_n)              # number of insurance categories

    ####### INITIALIZATION ###########################################
    # create the cohort trace
    m_M <- matrix(NA, nrow = n_t + 1 ,
                  ncol = n_s,
                  dimnames = list(0:n_t, v_n))     # create Markov trace (n_t + 1 because R doesn't und

    m_M[1, ] <- p                       # initialize Markov trace

    # create transition probability matrix for NO treatment
    m_P <- matrix(0,
                  nrow = n_s,
                  ncol = n_s,
                  dimnames = list(v_n, v_n))
    # fill in the transition probability array
    ### From ESI
    m_P["ESI", "ESI"]  <- 1 - (p_ESI_NG + p_ESI_PUB + p_ESI_UNIN)
    m_P["ESI", "NG"] <- p_ESI_NG
    m_P["ESI", "PUB"] <- p_ESI_PUB
    m_P["ESI", "UNIN"] <- p_ESI_UNIN

    ### From Non-Group
```

```
    m_P["NG", "NG"]  <- 1 - (p_NG_ESI + p_NG_PUB + p_NG_UNIN)
    m_P["NG", "ESI"] <- p_NG_ESI
    m_P["NG", "PUB"] <- p_NG_PUB
    m_P["NG", "UNIN"] <- p_NG_UNIN

    ### From Non-Group
    m_P["PUB", "NG"]  <- p_PUB_NG
    m_P["PUB", "ESI"] <- p_PUB_ESI
    m_P["PUB", "PUB"] <- 1 - (p_PUB_ESI + p_PUB_NG + p_PUB_UNIN)
    m_P["PUB", "UNIN"] <- p_PUB_UNIN


    ### From UNINSURED
    m_P["UNIN", "NG"]  <- p_UNIN_NG
    m_P["UNIN", "ESI"] <- p_UNIN_ESI
    m_P["UNIN", "PUB"] <- p_UNIN_PUB
    m_P["UNIN", "UNIN"] <- 1 - (p_UNIN_ESI + p_UNIN_NG + p_UNIN_PUB)


    # check rows add up to 1
    if (!isTRUE(all.equal(as.numeric(rowSums(m_P)), as.numeric(rep(1, n_s))))) {
      stop("This is not a valid transition Matrix")
    }

        ############## PROCESS ##############################################

    for (t in 1:n_t){                                 # throughout the number of cycles
      m_M[t + 1, ] <- m_M[t, ] %*% m_P           # estimate the Markov trace for cycle the next cycle (
    }

    ####### EPIDEMIOLOGICAL OUTPUT  ############################################
    #### Overall Survival (OS) ####
    #v_os <- 1 - m_M[, "UNIN"]                # calculate the overall survival (OS) probability for no
    v_ESI <-  m_M[,"ESI"]
    v_NG <- m_M[,"NG"]
    v_PUB <- m_M[,"PUB"]
    v_UNIN <- m_M[,c("UNIN")]


    ####### RETURN OUTPUT  #############################################
    #out <- list(coverage = m_M[2,])
    out <- list(ESI = m_M[-1,1],
                NG = m_M[-1,2],
                PUB = m_M[-1,3],
                UNIN = m_M[-1,4])

    return(out)
  }
  )
}

# Check that it works
v_params_test <- c(p_ESI_NG = 0.2, p_ESI_UNIN = 0.2)
```

```r
run_insurance_markov(ev) # It works!
```

```
## $ESI
##        1        2        3        4
## 0.630585 0.635661 0.639669 0.642937
##
## $NG
##         1         2         3         4
## 0.0721668 0.0820740 0.0893385 0.0946873
##
## $PUB
##        1        2        3        4
## 0.121785 0.128562 0.132045 0.133477
##
## $UNIN
##        1        2        3        4
## 0.175463 0.153703 0.138947 0.128899
```

```r
# calibration functionality
library(lhs)
library(IMIS)
```

```
## Loading required package: mvtnorm
```

```r
library(matrixStats) # package used for sumamry statistics

# visualization
library(plotrix)
library(psych)
```

```r
######################################################################
######  Specify calibration parameters  ######
######################################################################
# Specify seed (for reproducible sequence of random numbers)
set.seed(23)

prior_type = "uniform"

# number of random samples
n_resamp <- 5000

# names and number of input parameters to be calibrated
v_param_names <- c("p_ESI_NG","p_ESI_PUB","p_ESI_UNIN","p_NG_ESI","p_NG_PUB","p_NG_UNIN",
                   "p_PUB_ESI","p_PUB_NG","p_PUB_UNIN","p_UNIN_ESI", "p_UNIN_NG","p_UNIN_PUB")
n_param <- length(v_param_names)

# range on input search space
ev <- c(p_ESI_NG = R[1,2], p_ESI_PUB = R[1,3], p_ESI_UNIN =R[1,4],
        p_NG_ESI = R[2,1], p_NG_PUB = R[2,3], p_NG_UNIN = R[2,4],
        p_PUB_ESI = R[3,1], p_PUB_NG = R[3,2], p_PUB_UNIN = R[3,4],
        p_UNIN_ESI = R[4,1], p_UNIN_NG =R[4,2], p_UNIN_PUB = R[4,3]) # lower bound
lb <- c(p_ESI_NG = 0.001, p_ESI_PUB = 0.001, p_ESI_UNIN = 0.03,
        p_NG_ESI = 0.05, p_NG_PUB = 0.001, p_NG_UNIN = 0.01,
        p_PUB_ESI = 0.01, p_PUB_NG = 0.01, p_PUB_UNIN = 0.02,
        p_UNIN_ESI = 0.05, p_UNIN_NG = 0.01, p_UNIN_PUB = 0.05) # lower bound
```

```r
ub <- c(p_ESI_NG = 0.15, p_ESI_PUB = 0.15, p_ESI_UNIN = 0.15,
        p_NG_ESI = 0.30, p_NG_PUB = 0.20, p_NG_UNIN = 0.10,
        p_PUB_ESI = 0.15, p_PUB_NG = 0.1, p_PUB_UNIN = 0.3,
        p_UNIN_ESI = 0.3, p_UNIN_NG = .3, p_UNIN_PUB = 0.3) # upper bound

# number of calibration targets
v_target_names <- c("ESI","NG","PUB","UNIN")
n_target     <- length(v_target_names)

### Calibration functions

#  Write function to sample from prior
sample_prior <- function(n_samp){
  m_lhs_unit   <- randomLHS(n = n_samp, k = n_param)
  m_param_samp <- matrix(nrow = n_samp, ncol = n_param)
  colnames(m_param_samp) <- v_param_names
  for (i in 1:n_param){

  if (prior_type=="beta") {
   m_param_samp[,i] <- qbeta(m_lhs_unit[,i],shape1 = ev[i]*100,shape2=100-ev[i]*100)
  } else {
    m_param_samp[, i] <- qunif(m_lhs_unit[,i],
                               min = lb[i],
                               max = ub[i])
    }
  }
  return(m_param_samp)
}


###  PRIOR  ###
# Write functions to evaluate log-prior and prior

# function that calculates the log-prior
calc_log_prior <- function(v_params){
  if(is.null(dim(v_params))) { # If vector, change to matrix
    v_params <- t(v_params)
  }
  n_samp <- nrow(v_params)
  colnames(v_params) <- v_param_names
  lprior <- rep(0, n_samp)
  for (i in 1:n_param){
    if (prior_type=="beta") {
      lprior <- lprior + dbeta(v_params[,i], shape1 = ev[i]*100, shape2 = 100-ev[i]*100,log = T)
    } else {
    lprior <- lprior + dunif(v_params[, i],
                             min = lb[i],
                             max = ub[i],
                             log = T)
    }

  }
  return(lprior)
```

```
}
calc_log_prior(v_params = ev)
```

```
## p_ESI_NG
##     -Inf
```

```
calc_log_prior(v_params = sample_prior(10))
```

```
##  [1] 20.9941 20.9941 20.9941 20.9941 20.9941 20.9941 20.9941 20.9941
##  [9] 20.9941 20.9941
```

```
# function that calculates the (non-log) prior
calc_prior <- function(v_params) {
  exp(calc_log_prior(v_params))
}
calc_prior(v_params = ev)
```

```
## p_ESI_NG
##        0
```

```
calc_prior(v_params = sample_prior(10))
```

```
##  [1] 1311004043 1311004043 1311004043 1311004043 1311004043 1311004043
##  [7] 1311004043 1311004043 1311004043 1311004043
```

```
###  LIKELIHOOD  ###
# Write functions to evaluate log-likelihood and likelihood

# function to calculate the log-likelihood
calc_log_lik <- function(v_params){
  # par_vector: a vector (or matrix) of model parameters
  if(is.null(dim(v_params))) { # If vector, change to matrix
    v_params <- t(v_params)
  }
  n_samp <- nrow(v_params)
  v_llik <- matrix(0, nrow = n_samp, ncol = n_target)
  llik_overall <- numeric(n_samp)
  for(j in 1:n_samp) { # j=1
    jj <- tryCatch( {
      ###   Run model for parametr set "v_params" ###
      model_res <- run_insurance_markov(v_params[j, ])

      ###  Calculate log-likelihood of model outputs to targets  ###
      # TARGET 1: Survival ("Surv")
      # log likelihood
      v_llik[j, 1] <- sum(dnorm(x = lst_targets$ESI,
                                mean = model_res$ESI,
                                sd = 1*sqrt(lst_targets$ESI * (1-lst_targets$ESI))/1e2,
                                log = T))

      v_llik[j, 2] <- sum(dnorm(x = lst_targets$NG,
                                mean = model_res$NG,
                                sd = 1*sqrt(lst_targets$NG * (1-lst_targets$NG))/1e2,
                                log = T))
      v_llik[j, 3] <- sum(dnorm(x = lst_targets$PUB,
                                mean = model_res$PUB,
```

```
                                sd = 1*sqrt(lst_targets$PUB * (1-lst_targets$PUB))/1e2,
                                log = T))
      v_llik[j, 4] <- sum(dnorm(x = lst_targets$UNIN,
                                mean = model_res$UNIN,
                                sd = 1*sqrt(lst_targets$UNIN * (1-lst_targets$UNIN))/1e2,
                                log = T))

      # OVERALL
      llik_overall[j] <- sum(v_llik[j, ])
    }, error = function(e) NA)
    if(is.na(jj)) { llik_overall <- -Inf }
  } # End loop over sampled parameter sets
  # return LLIK
  return(llik_overall)
}
calc_log_lik(v_params = ev)
```

```
## [1] -247.022
```

```
calc_log_lik(v_params = sample_prior(10))
```

```
##  [1] -11325.809  -1960.220 -11849.213  -4044.219  -3913.066  -5847.066
##  [7]  -6539.538  -6241.004   -943.418 -18618.880
```

```
# function to calculate the (non-log) likelihood
calc_likelihood <- function(v_params){
  exp(calc_log_lik(v_params))
}
calc_likelihood(v_params = ev)
```

```
## [1] 5.24275e-108
```

```
calc_likelihood(v_params = sample_prior(1000))
```

```
##    [1]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##    [6]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##   [11]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##   [16]  0.00000e+00 8.90269e-250  0.00000e+00  0.00000e+00  0.00000e+00
##   [21]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##   [26]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##   [31]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##   [36]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##   [41]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##   [46]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##   [51]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##   [56]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##   [61]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##   [66]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##   [71]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##   [76]  0.00000e+00 3.19937e-138  0.00000e+00  0.00000e+00  0.00000e+00
##   [81]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##   [86]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##   [91]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##   [96]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [101]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [106]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
```

```
##  [111]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [116]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [121]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [126]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [131]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [136]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [141]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [146]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [151]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [156]  0.00000e+00  0.00000e+00  0.00000e+00 1.79154e-255  0.00000e+00
##  [161]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [166]  1.08705e-16  0.00000e+00  0.00000e+00  0.00000e+00 6.57328e-119
##  [171]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [176]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [181]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00 8.66453e-253
##  [186]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [191]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [196]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [201]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [206]  0.00000e+00  0.00000e+00  0.00000e+00 3.07743e-187  0.00000e+00
##  [211]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [216]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [221]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [226]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [231]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [236]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [241]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [246]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [251]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [256]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [261]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [266]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [271]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [276]  0.00000e+00  0.00000e+00 2.59908e-224  0.00000e+00  0.00000e+00
##  [281] 2.32058e-112  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [286]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [291]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [296]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [301]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [306]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [311]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [316]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [321]  0.00000e+00 1.36189e-215  0.00000e+00  0.00000e+00  0.00000e+00
##  [326]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [331]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  2.68881e-57
##  [336]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [341]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [346]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [351]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [356]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [361]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [366]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [371]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [376]  0.00000e+00  0.00000e+00  0.00000e+00  4.33885e-25  0.00000e+00
```

```
##  [381]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [386]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [391]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [396]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [401]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [406]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [411]  0.00000e+00  0.00000e+00  9.68875e-84  0.00000e+00  0.00000e+00
##  [416]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [421]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00 3.05679e-214
##  [426]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  1.40209e-75
##  [431]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [436]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [441]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [446]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [451]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [456]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [461]  0.00000e+00 1.32366e-264  0.00000e+00  0.00000e+00  0.00000e+00
##  [466]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [471]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [476]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [481]  0.00000e+00 4.58299e-181  0.00000e+00  0.00000e+00  0.00000e+00
##  [486]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [491]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [496]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [501]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [506]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [511]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [516]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [521]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [526]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [531]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [536]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [541]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [546]  0.00000e+00  0.00000e+00  0.00000e+00 8.57708e-319  0.00000e+00
##  [551]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [556]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [561]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [566]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [571]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [576] 4.58222e-236  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [581]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [586]  0.00000e+00  0.00000e+00  0.00000e+00 1.47946e-183  0.00000e+00
##  [591]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [596]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [601]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [606]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [611]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [616]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [621]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [626]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [631]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [636]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [641]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [646]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
```

```
##  [651]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [656]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [661]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [666]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [671]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [676]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [681]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [686]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [691]  0.00000e+00  0.00000e+00 3.78358e-298  0.00000e+00  0.00000e+00
##  [696]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [701]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [706]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [711]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [716]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [721]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [726]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [731]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [736]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [741]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [746]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [751]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [756]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [761] 8.81970e-249  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [766]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [771]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [776]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [781]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [786]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [791]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [796]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [801]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [806]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [811]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [816]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [821]  0.00000e+00 9.07204e-308  0.00000e+00  0.00000e+00  0.00000e+00
##  [826]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [831]  0.00000e+00  0.00000e+00 6.83216e-223  0.00000e+00  0.00000e+00
##  [836]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [841]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [846]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [851]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [856]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [861]  0.00000e+00  0.00000e+00  0.00000e+00 2.07273e-242  0.00000e+00
##  [866]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [871]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [876]  0.00000e+00 3.25672e-297  0.00000e+00  0.00000e+00  0.00000e+00
##  [881] 3.31673e-191  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [886]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [891]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [896]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [901]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [906]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [911]  0.00000e+00  3.05176e-42  0.00000e+00  0.00000e+00  0.00000e+00
##  [916] 8.28256e-190  0.00000e+00  0.00000e+00  0.00000e+00 1.13734e-178
```

```
##  [921]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [926]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [931]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [936]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [941]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [946]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [951]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [956]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [961]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [966]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [971]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [976]  0.00000e+00 2.45908e-206  0.00000e+00  0.00000e+00  0.00000e+00
##  [981]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [986]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [991]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
##  [996]  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00
```

```
### POSTERIOR ###
# Write functions to evaluate log-posterior and posterior

# function that calculates the log-posterior
calc_log_post <- function(v_params) {
  lpost <- calc_log_prior(v_params) + calc_log_lik(v_params)
  return(lpost)
}
calc_log_post(v_params = ev)
```

```
## p_ESI_NG
##     -Inf
```

```
calc_log_post(v_params = sample_prior(10))
```

```
##  [1]  -2229.62  -7388.25  -8206.49  -5565.80  -1441.97  -8880.22 -11573.37
##  [8] -10426.18  -8763.88  -6569.15
```

```
# function that calculates the (non-log) posterior
calc_post <- function(v_params) {
  exp(calc_log_post(v_params))
}
calc_post(v_params = ev)
```

```
## p_ESI_NG
##        0
```

```
calc_post(v_params = sample_prior(10))
```

```
##  [1] 0 0 0 0 0 0 0 0 0 0
```

```
########################################################################
###### Calibrate! ######
########################################################################
# record start time of calibration
t_init <- Sys.time()

### Bayesian calibration using IMIS ###
# define three functions needed by IMIS: prior(x), likelihood(x), sample.prior(n)
```

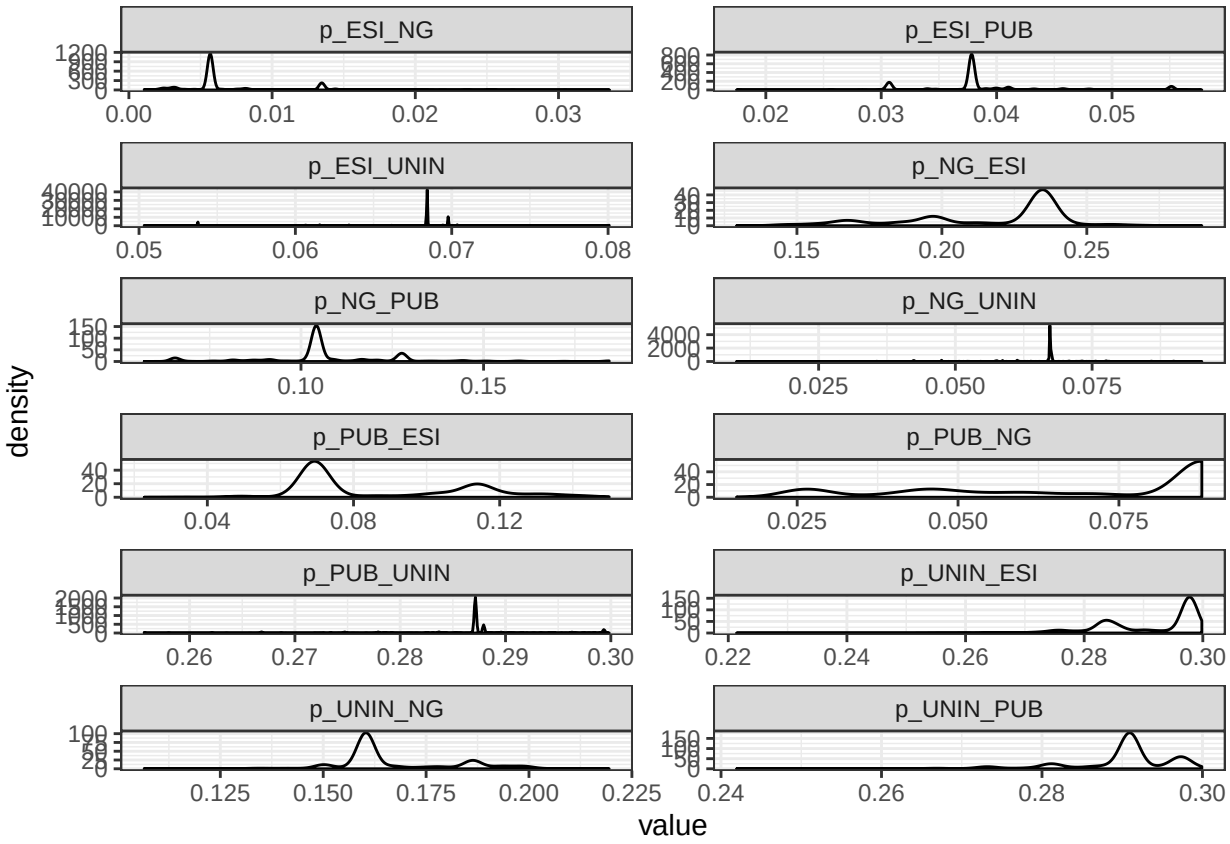
```r
# based on ACS tables for 19-64 year olds in ACS data as constructed at http://statehealthcompare.shada
```

```r
prior <- calc_prior
likelihood <- calc_likelihood
sample.prior <- sample_prior

# run IMIS
fit_imis <- IMIS(B = 5000, # the incremental sample size at each iteration of IMIS
                 B.re = n_resamp, # the desired posterior sample size
                 number_k = 10, # the maximum number of iterations in IMIS
                 D = 0)
```

```
## [1] "50000 likelihoods are evaluated in 0.1 minutes"
## [1] "Stage   MargLike   UniquePoint   MaxWeight   ESS"
## [1]  1.000 25.130  3.065  0.667  1.832
## Warning in sort.int(x, na.last = na.last, decreasing = decreasing, ...):
## partial argument match of 'index' to 'index.return'
## [1]  2.000 38.742  6.083  0.969  1.064
## Warning in sort.int(x, na.last = na.last, decreasing = decreasing, ...):
## partial argument match of 'index' to 'index.return'
## [1]  3.000 42.340 18.673  0.332  4.188
## Warning in sort.int(x, na.last = na.last, decreasing = decreasing, ...):
## partial argument match of 'index' to 'index.return'
## [1]  4.000 42.602 82.184  0.764  1.657
## Warning in sort.int(x, na.last = na.last, decreasing = decreasing, ...):
## partial argument match of 'index' to 'index.return'
## [1]   5.000  41.185 342.494   0.245  13.435
## Warning in sort.int(x, na.last = na.last, decreasing = decreasing, ...):
## partial argument match of 'index' to 'index.return'
## [1]   6.000  43.498 309.921   0.373   6.010
## Warning in sort.int(x, na.last = na.last, decreasing = decreasing, ...):
## partial argument match of 'index' to 'index.return'
## [1]   7.000  43.968 521.123   0.168  16.782
## Warning in sort.int(x, na.last = na.last, decreasing = decreasing, ...):
## partial argument match of 'index' to 'index.return'
## [1]   8.000  44.727 668.145   0.138  23.658
## Warning in sort.int(x, na.last = na.last, decreasing = decreasing, ...):
## partial argument match of 'index' to 'index.return'
## [1]   9.000  45.580 703.879   0.207  12.442
## Warning in sort.int(x, na.last = na.last, decreasing = decreasing, ...):
## partial argument match of 'index' to 'index.return'
## [1]  10.000  46.934 473.426   0.525   3.421
```

```
## Warning in sort.int(x, na.last = na.last, decreasing = decreasing, ...):
## partial argument match of 'index' to 'index.return'
```

```r
# obtain draws from posterior
m_calib_res <- fit_imis$resample



# Calculate log-likelihood (overall fit) and posterior probability of each sample
m_calib_res <- cbind(m_calib_res,
                     "Overall_fit" = calc_log_lik(m_calib_res[,v_param_names]),
                     "Posterior_prob" = calc_post(m_calib_res[,v_param_names]))

# normalize posterior probability
m_calib_res[,"Posterior_prob"] <- m_calib_res[,"Posterior_prob"]/sum(m_calib_res[,"Posterior_prob"])

# Calculate computation time
comp_time <- Sys.time() - t_init

######################################################################
######  Exploring posterior distribution  ######
######################################################################

m_calib_res %>%
  tbl_df() %>%
  gather(parameter,value) %>%
  filter(grepl("^p_",parameter)) %>%
  ggplot(aes(x = value)) + geom_density() +
  facet_wrap(~parameter, scales = "free",nrow = 7) +
  theme_bw()
```

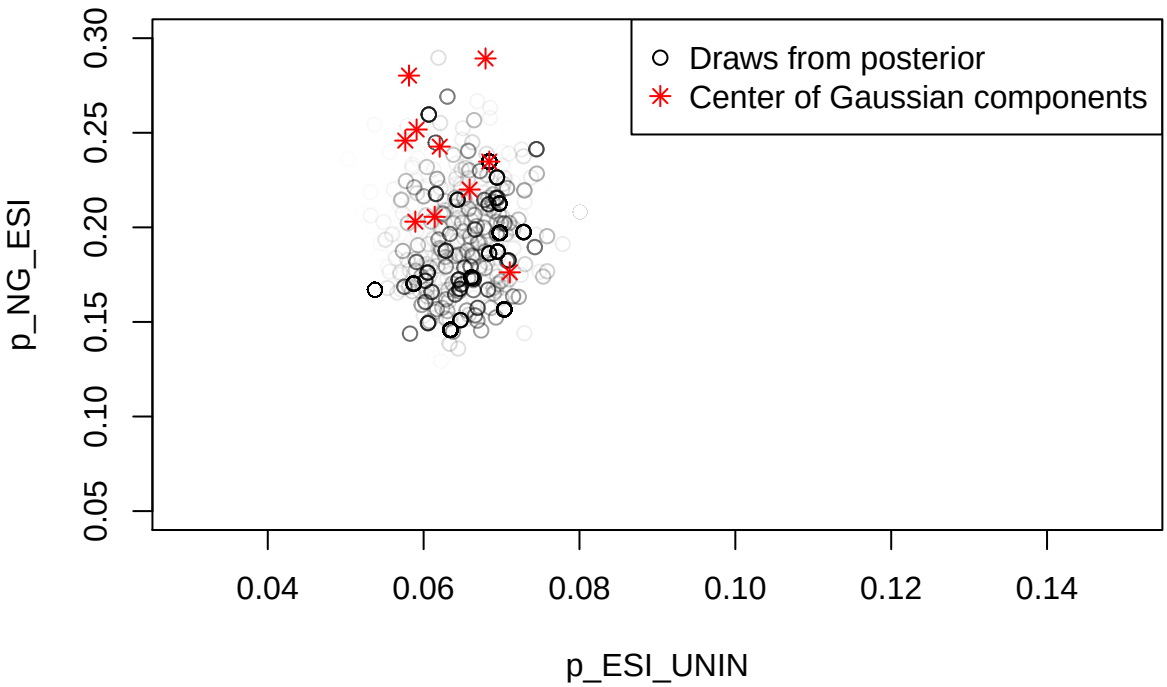



```
fit_imis$center %>%
  tbl_df() %>%
  gather(parameter,value)
```

```
## # A tibble: 120 x 2
##    parameter   value
##    <chr>       <dbl>
##  1 p_ESI_NG  0.0156
##  2 p_ESI_NG  0.0224
##  3 p_ESI_NG  0.0160
##  4 p_ESI_NG  0.0206
##  5 p_ESI_NG  0.0290
##  6 p_ESI_NG  0.0158
##  7 p_ESI_NG  0.00120
##  8 p_ESI_NG  0.0115
##  9 p_ESI_NG  0.00593
## 10 p_ESI_NG  0.00567
## # ... with 110 more rows
```

```
# Plot the 1000 draws from the posterior
v_post_color <- scales::rescale(m_calib_res[,"Posterior_prob"])
plot(m_calib_res[,3:4],
     xlim = c(lb[3], ub[3]), ylim = c(lb[4], ub[4]),
     xlab = v_param_names[3], ylab = v_param_names[4],
     col = scales::alpha("black", v_post_color))
# add center of Gaussian components
points(fit_imis$center[,3:4], col = "red", pch = 8)
legend("topright", c("Draws from posterior", "Center of Gaussian components"),
```

```
       col = c("black", "red"), pch = c(1, 8))
```



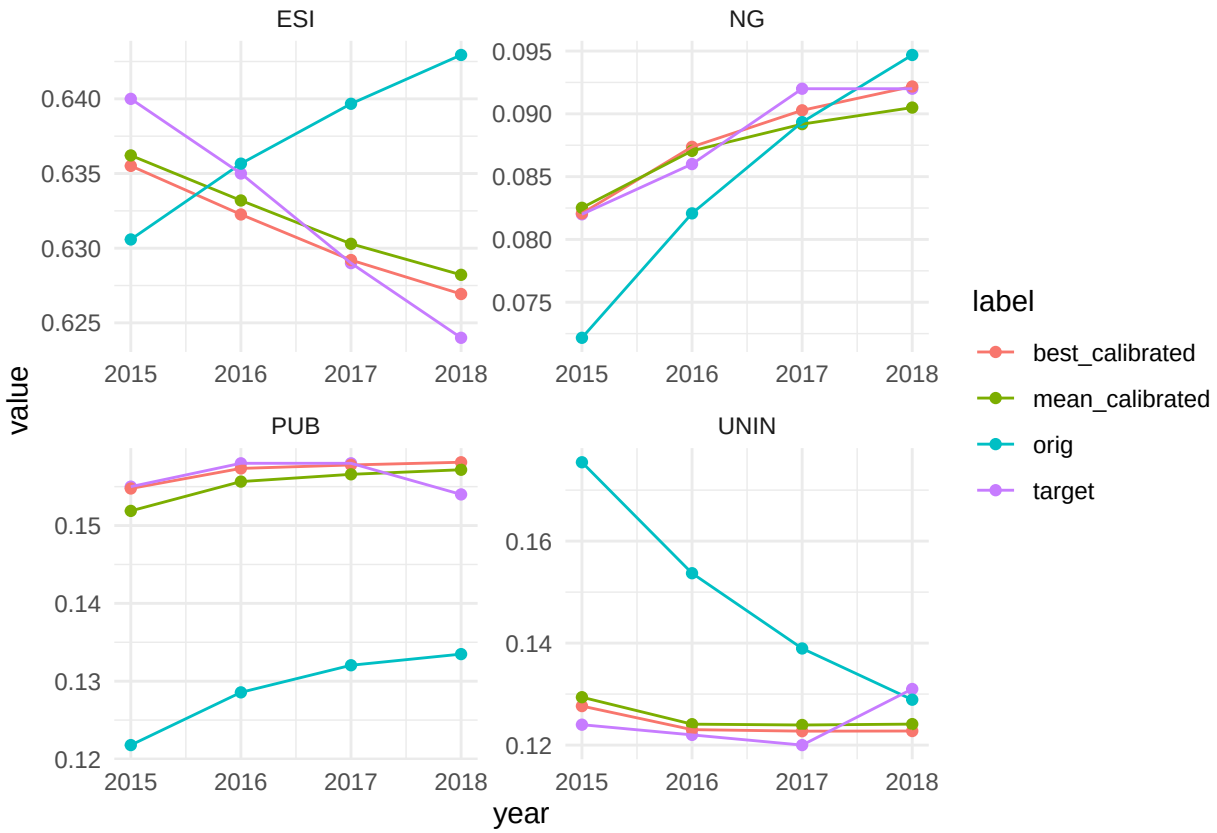

```
# Plot the 1000 draws from the posterior with marginal histograms
#pairs.panels(m_calib_res[,v_param_names])

# Compute posterior mean
v_calib_post_mean <- colMeans(m_calib_res[,v_param_names])
v_calib_post_mean
```

```
##   p_ESI_NG  p_ESI_PUB p_ESI_UNIN   p_NG_ESI   p_NG_PUB  p_NG_UNIN
##  0.0071318  0.0390686  0.0668609  0.2163980  0.1073379  0.0650929
##  p_PUB_ESI   p_PUB_NG p_PUB_UNIN p_UNIN_ESI  p_UNIN_NG p_UNIN_PUB
##  0.0881204  0.0689490  0.2868528  0.2911682  0.1670306  0.2896618
```

```
# Compute posterior median and 95% credible interval
m_calib_res_95cr <- colQuantiles(m_calib_res[,v_param_names], probs = c(0.025, 0.5, 0.975))
m_calib_res_95cr
```

```
##                  2.5%        50%     97.5%
## p_ESI_NG   0.00243843 0.00567289 0.0167908
## p_ESI_PUB  0.03069305 0.03783478 0.0551492
## p_ESI_UNIN 0.05375385 0.06842286 0.0728317
## p_NG_ESI   0.15662066 0.23475335 0.2537634
## p_NG_PUB   0.06548991 0.10419687 0.1510120
## p_NG_UNIN  0.03935383 0.06732009 0.0784554
## p_PUB_ESI  0.06246668 0.06924260 0.1386076
## p_PUB_NG   0.02643791 0.08788471 0.0878847
## p_PUB_UNIN 0.26683187 0.28713582 0.2993384
## p_UNIN_ESI 0.27257019 0.29776979 0.2977698
## p_UNIN_NG  0.14129399 0.16038882 0.1996357
## p_UNIN_PUB 0.27107836 0.29094989 0.2983666
```

```r
# Compute maximum-a-posteriori (MAP) parameter set
v_calib_map <- m_calib_res[which.max(m_calib_res[,"Posterior_prob"]),]

### Plot model-predicted output at mode vs targets ###
v_out_best <- run_insurance_markov(v_calib_map[v_param_names])
v_out_mean <- run_insurance_markov(v_calib_post_mean)
v_out_orig <- run_insurance_markov(ev)

bind_rows(
lst_targets %>% bind_cols() %>%
  mutate(year = 2015:2018) %>%
  mutate(label = "target"),
v_out_best %>% bind_cols() %>%
  mutate(year = 2015:2018) %>%
  mutate(label = "best_calibrated"),
v_out_mean %>% bind_cols() %>%
  mutate(year = 2015:2018) %>%
  mutate(label = "mean_calibrated"),
v_out_orig %>% bind_cols() %>%
  mutate(year = 2015:2018) %>%
  mutate(label = "orig")
) %>%
  gather(type,value,-year,-label) %>%
  ggplot(aes(x = year, y = value, colour = label)) +
  geom_point() +
  facet_wrap(~type, scales = "free") +
  theme_minimal() +
  geom_line()
```




```
final_probs <- v_calib_map[v_param_names]

R_c <- matrix(nrow = nrow(R), ncol = ncol(R))
R_c[1,1] <- 1 - final_probs["p_ESI_NG"] - final_probs["p_ESI_PUB"]  - final_probs["p_ESI_UNIN"]
R_c[1,2] <- final_probs["p_ESI_NG"]
R_c[1,3] <- final_probs["p_ESI_PUB"]
R_c[1,4] <- final_probs["p_ESI_UNIN"]

R_c[2,2] <- 1 - final_probs["p_NG_ESI"] - final_probs["p_NG_PUB"]  - final_probs["p_NG_UNIN"]
R_c[2,1] <- final_probs["p_NG_ESI"]
R_c[2,3] <- final_probs["p_NG_PUB"]
R_c[2,4] <- final_probs["p_NG_UNIN"]

R_c[3,3] <- 1 - final_probs["p_PUB_ESI"] - final_probs["p_PUB_NG"]  - final_probs["p_PUB_UNIN"]
R_c[3,1] <- final_probs["p_PUB_ESI"]
R_c[3,2] <- final_probs["p_PUB_NG"]
R_c[3,4] <- final_probs["p_PUB_UNIN"]

R_c[4,4] <- 1 - final_probs["p_UNIN_ESI"] - final_probs["p_UNIN_NG"]  - final_probs["p_UNIN_PUB"]
R_c[4,1] <- final_probs["p_UNIN_ESI"]
R_c[4,2] <- final_probs["p_UNIN_NG"]
R_c[4,3] <- final_probs["p_UNIN_PUB"]


run_insurance_markov(ev)  %>%
  map(~(.x %>% tbl_df())) %>%
  bind_cols() %>%
```

```
  set_names(c("ESI","NG","PUB","UNIN")) %>%
  mutate(type = "uncalibrated") %>%
  mutate(year = c("2015","2016","2017","2018"))
```

```
## # A tibble: 4 x 6
##     ESI     NG   PUB  UNIN type         year
##   <dbl>  <dbl> <dbl> <dbl> <chr>        <chr>
## 1 0.631 0.0722 0.122 0.175 uncalibrated 2015
## 2 0.636 0.0821 0.129 0.154 uncalibrated 2016
## 3 0.640 0.0893 0.132 0.139 uncalibrated 2017
## 4 0.643 0.0947 0.133 0.129 uncalibrated 2018
```

```
run_insurance_markov(final_probs) %>%
    map(~(.x %>% tbl_df())) %>%
  bind_cols() %>%
  set_names(c("ESI","NG","PUB","UNIN")) %>%
  mutate(type = "calibrated") %>%
  mutate(year = c("2015","2016","2017","2018"))
```

```
## # A tibble: 4 x 6
##     ESI     NG   PUB  UNIN type       year
##   <dbl>  <dbl> <dbl> <dbl> <chr>      <chr>
## 1 0.636 0.0821 0.155 0.128 calibrated 2015
## 2 0.632 0.0874 0.157 0.123 calibrated 2016
## 3 0.629 0.0903 0.158 0.123 calibrated 2017
## 4 0.627 0.0922 0.158 0.123 calibrated 2018
```

```
lst_targets %>%
    map(~(.x %>% tbl_df())) %>%
  bind_cols() %>%
  set_names(c("ESI","NG","PUB","UNIN")) %>%
  mutate(type = "target") %>%
  mutate(year = c("2015","2016","2017","2018"))
```

```
## # A tibble: 4 x 6
##     ESI    NG   PUB  UNIN type   year
##   <dbl> <dbl> <dbl> <dbl> <chr>  <chr>
## 1 0.64  0.082 0.155 0.124 target 2015
## 2 0.635 0.086 0.158 0.122 target 2016
## 3 0.629 0.092 0.158 0.12  target 2017
## 4 0.624 0.092 0.154 0.131 target 2018
```

```
# m_calib_res %>% tbl_df() %>%
#   select(-Overall_fit, -Posterior_prob) %>%
#   gather(parameter,value) %>%
#   filter(value<1) %>%
#   ggplot(aes(x = value, colour = parameter, group = parameter)) +
#   geom_density() +
#   theme_minimal()
```