Basic example of history matching with emulation workflow

Packages used in this vignette.

```
library(hmsandwich)
```

SIR model

We'll use the simple susceptible-infected-removed model used in the `{hmer}` package examples.

Our package used tidy models for inspiration .

Input data

```
ranges <- list(
  b = c(1e-5, 1e-4),       # birth rate
  mu = c(1e-5, 1e-4),      # rate of death from other causes
  beta1 = c(0.2, 0.3),     # infection rate at time t=0
  beta2 = c(0.1, 0.2),     # infection rates at time t=100
  beta3 = c(0.3, 0.5),     # infection rates at time t=270
  epsilon = c(0.07, 0.21), # rate of becoming infectious after infection
  alpha = c(0.01, 0.025),  # rate of death from the disease
  gamma = c(0.05, 0.08),   # recovery rate
  omega = c(0.002, 0.004)  # rate at which immunity is lost following recovery
)

targets <- list(
  I25 = list(val = 115.88, sigma = 5.79),
  I40 = list(val = 137.84, sigma = 6.89),
  I100 = list(val = 26.34, sigma = 1.317),
  I200 = list(val = 0.68, sigma = 0.034),
  I300 = list(val = 29.55, sigma = 1.48),
  I350 = list(val = 68.89, sigma = 3.44),
  R25 = list(val = 125.12, sigma = 6.26),
  R40 = list(val = 256.80, sigma = 12.84),
  R100 = list(val = 538.99, sigma = 26.95),
  R200 = list(val = 444.23, sigma = 22.21),
  R300 = list(val = 371.08, sigma = 15.85),
  R350 = list(val = 549.42, sigma = 27.47)
)
```

SIR model functions

```
# provides the solution of the differential equations for a given
# set of parameters. assumes an initial population of
# 900 susceptible individuals, 100 exposed individuals, and no infectious
# or recovered individuals.
ode_results <- function(parms, end_time = 365*2) {
  forcer = matrix(c(0,   parms['beta1'],
                    100, parms['beta2'],
                    180, parms['beta2'],
                    270, parms['beta3']),
                  ncol = 2,
                  byrow = TRUE)
  force_func <- approxfun(
    x = forcer[, 1],
    y = forcer[, 2],
    method = "linear",
    rule = 2)

  des <- function(time, state, parms) {
    with(as.list(c(state, parms)), {
      # N <- S + E + I + R
      dS <- b*(S+E+I+R) - force_func(time)*I*S / (S+E+I+R) + omega*R - mu*S
      dE <- force_func(time)*I*S / (S+E+I+R) - epsilon*E - mu*E
      dI <- epsilon*E - alpha*I - gamma*I - mu*I
      dR <- gamma*I - omega*R - mu*R
      return(list(c(dS, dE, dI, dR)))
    })
  }
  yini <- c(S = 900, E = 100, I = 0, R = 0)
  times <- seq(0, end_time, by = 1)

  deSolve::ode(yini, times, des, parms)
}

#' Wrapper for `ode_results` to subset which outputs and times should be returned
#'
#' For example, to obtain the number of infected and susceptible individuals
#' at t=25 and t=50
#' times = c(25,50)
#' outputs = c('I','S')
#' @returns
#'
sir_model <- function(params, times, outputs) {
  t_max <- max(times)
  all_res <- ode_results(params, t_max)
  actual_res <- all_res[all_res[,'time'] %in% times, c('time', outputs)]
  shaped <- reshape2::melt(actual_res[, outputs])

  names_shaped <- paste0(shaped$Var2, actual_res[,'time'], sep = "")

  return(setNames(shaped$value, names_shaped))
}
```

Analysis

Create analysis object.

```
sir_hmer <-
  hmer_analysis() |>
  add_input_ranges(ranges) |>
  add_targets(targets) |>
  add_model(sir_model)
```

Run initial wave.

```
sir_hmer <- run_wave(sir_hmer)
# sir_hmer@run_wave

str(sir_hmer)
#> List of 10
#>  $ wave_no        : num 1
#>  $ n_train        : num 100
#>  $ n_grps_in      : NULL
#>  $ groups_in      : NULL
#>  $ indx_out       : NULL
#>  $ full_groups_out: NULL
#>  $ ranges         :List of 9
#>   ..$ b      : num [1:2] 1e-05 1e-04
#>   ..$ mu     : num [1:2] 1e-05 1e-04
#>   ..$ beta1  : num [1:2] 0.2 0.3
#>   ..$ beta2  : num [1:2] 0.1 0.2
#>   ..$ beta3  : num [1:2] 0.3 0.5
#>   ..$ epsilon: num [1:2] 0.07 0.21
#>   ..$ alpha  : num [1:2] 0.01 0.025
#>   ..$ gamma  : num [1:2] 0.05 0.08
#>   ..$ omega  : num [1:2] 0.002 0.004
#>  $ targets        :List of 12
#>   ..$ I25 :List of 2
#>   .. ..$ val  : num 116
#>   .. ..$ sigma: num 5.79
#>   ..$ I40 :List of 2
#>   .. ..$ val  : num 138
#>   .. ..$ sigma: num 6.89
#>   ..$ I100:List of 2
#>   .. ..$ val  : num 26.3
#>   .. ..$ sigma: num 1.32
#>   ..$ I200:List of 2
#>   .. ..$ val  : num 0.68
#>   .. ..$ sigma: num 0.034
#>   ..$ I300:List of 2
#>   .. ..$ val  : num 29.6
#>   .. ..$ sigma: num 1.48
#>   ..$ I350:List of 2
#>   .. ..$ val  : num 68.9
#>   .. ..$ sigma: num 3.44
#>   ..$ R25 :List of 2
#>   .. ..$ val  : num 125
#>   .. ..$ sigma: num 6.26
#>   ..$ R40 :List of 2
#>   .. ..$ val  : num 257
#>   .. ..$ sigma: num 12.8
#>   ..$ R100:List of 2
#>   .. ..$ val  : num 539
#>   .. ..$ sigma: num 26.9
#>   ..$ R200:List of 2
#>   .. ..$ val  : num 444
#>   .. ..$ sigma: num 22.2
#>   ..$ R300:List of 2
#>   .. ..$ val  : num 371
#>   .. ..$ sigma: num 15.8
#>   ..$ R350:List of 2
#>   .. ..$ val  : num 549
#>   .. ..$ sigma: num 27.5
#>  $ model          :function (params, times, outputs)
#>   ..- attr(*, "srcref")= 'srcref' int [1:8] 42 14 51 1 14 1 42 51
#>   .. ..- attr(*, "srcfile")=Classes 'srcfilecopy', 'srcfile' <environment: 0x000001a21a0ae5e0>
#>  $ wave0          :List of 5
#>   ..$ inputs    :'data.frame':   180 obs. of  9 variables:
#>   .. ..$ b      : num [1:180] 6.95e-05 4.11e-05 8.64e-05 6.86e-05 4.64e-05 ...
#>   .. ..$ mu     : num [1:180] 6.82e-05 6.29e-05 8.49e-05 3.75e-05 4.84e-05 ...
#>   .. ..$ beta1  : num [1:180] 0.207 0.229 0.254 0.243 0.242 ...
#>   .. ..$ beta2  : num [1:180] 0.159 0.153 0.176 0.127 0.17 ...
#>   .. ..$ beta3  : num [1:180] 0.364 0.464 0.388 0.345 0.369 ...
#>   .. ..$ epsilon: num [1:180] 0.152 0.137 0.156 0.186 0.117 ...
#>   .. ..$ alpha  : num [1:180] 0.0166 0.0177 0.0206 0.015 0.0168 ...
#>   .. ..$ gamma  : num [1:180] 0.0607 0.0653 0.072 0.0748 0.0632 ...
#>   .. ..$ omega  : num [1:180] 0.00269 0.0025 0.00256 0.00282 0.00262 ...
#>   ..$ results   :'data.frame':   180 obs. of  12 variables:
#>   .. ..$ I25 : num [1:180] 147 139 154 165 136 ...
#>   .. ..$ I40 : num [1:180] 191 180 183 180 187 ...
#>   .. ..$ I100: num [1:180] 37.2 31.4 18.6 16.5 38.3 ...
#>   .. ..$ I200: num [1:180] 2.061 1.212 0.599 0.337 1.882 ...
#>   .. ..$ I300: num [1:180] 24.23 30.05 6.97 4.99 12.65 ...
#>   .. ..$ I350: num [1:180] 63.2 57.9 49.8 56.1 50.6 ...
#>   .. ..$ R25 : num [1:180] 125 127 154 176 115 ...
#>   .. ..$ R40 : num [1:180] 275 280 334 367 264 ...
#>   .. ..$ R100: num [1:180] 618 620 626 637 636 ...
#>   .. ..$ R200: num [1:180] 519 522 506 500 539 ...
#>   .. ..$ R300: num [1:180] 430 442 400 383 435 ...
#>   .. ..$ R350: num [1:180] 517 566 434 415 472 ...
#>   ..$ data      :'data.frame':   180 obs. of  21 variables:
#>   .. ..$ b      : num [1:180] 6.95e-05 4.11e-05 8.64e-05 6.86e-05 4.64e-05 ...
#>   .. ..$ mu     : num [1:180] 6.82e-05 6.29e-05 8.49e-05 3.75e-05 4.84e-05 ...
#>   .. ..$ beta1  : num [1:180] 0.207 0.229 0.254 0.243 0.242 ...
#>   .. ..$ beta2  : num [1:180] 0.159 0.153 0.176 0.127 0.17 ...
#>   .. ..$ beta3  : num [1:180] 0.364 0.464 0.388 0.345 0.369 ...
#>   .. ..$ epsilon: num [1:180] 0.152 0.137 0.156 0.186 0.117 ...
#>   .. ..$ alpha  : num [1:180] 0.0166 0.0177 0.0206 0.015 0.0168 ...
#>   .. ..$ gamma  : num [1:180] 0.0607 0.0653 0.072 0.0748 0.0632 ...
#>   .. ..$ omega  : num [1:180] 0.00269 0.0025 0.00256 0.00282 0.00262 ...
#>   .. ..$ I25    : num [1:180] 147 139 154 165 136 ...
#>   .. ..$ I40    : num [1:180] 191 180 183 180 187 ...
#>   .. ..$ I100   : num [1:180] 37.2 31.4 18.6 16.5 38.3 ...
#>   .. ..$ I200   : num [1:180] 2.061 1.212 0.599 0.337 1.882 ...
#>   .. ..$ I300   : num [1:180] 24.23 30.05 6.97 4.99 12.65 ...
#>   .. ..$ I350   : num [1:180] 63.2 57.9 49.8 56.1 50.6 ...
#>   .. ..$ R25    : num [1:180] 125 127 154 176 115 ...
#>   .. ..$ R40    : num [1:180] 275 280 334 367 264 ...
#>   .. ..$ R100   : num [1:180] 618 620 626 637 636 ...
#>   .. ..$ R200   : num [1:180] 519 522 506 500 539 ...
#>   .. ..$ R300   : num [1:180] 430 442 400 383 435 ...
#>   .. ..$ R350   : num [1:180] 517 566 434 415 472 ...
#>   ..$ training  :'data.frame':   100 obs. of  21 variables:
#>   .. ..$ b      : num [1:100] 6.95e-05 4.11e-05 8.64e-05 6.86e-05 4.64e-05 ...
#>   .. ..$ mu     : num [1:100] 6.82e-05 6.29e-05 8.49e-05 3.75e-05 4.84e-05 ...
#>   .. ..$ beta1  : num [1:100] 0.207 0.229 0.254 0.243 0.242 ...
#>   .. ..$ beta2  : num [1:100] 0.159 0.153 0.176 0.127 0.17 ...
#>   .. ..$ beta3  : num [1:100] 0.364 0.464 0.388 0.345 0.369 ...
#>   .. ..$ epsilon: num [1:100] 0.152 0.137 0.156 0.186 0.117 ...
#>   .. ..$ alpha  : num [1:100] 0.0166 0.0177 0.0206 0.015 0.0168 ...
#>   .. ..$ gamma  : num [1:100] 0.0607 0.0653 0.072 0.0748 0.0632 ...
#>   .. ..$ omega  : num [1:100] 0.00269 0.0025 0.00256 0.00282 0.00262 ...
#>   .. ..$ I25    : num [1:100] 147 139 154 165 136 ...
#>   .. ..$ I40    : num [1:100] 191 180 183 180 187 ...
#>   .. ..$ I100   : num [1:100] 37.2 31.4 18.6 16.5 38.3 ...
#>   .. ..$ I200   : num [1:100] 2.061 1.212 0.599 0.337 1.882 ...
#>   .. ..$ I300   : num [1:100] 24.23 30.05 6.97 4.99 12.65 ...
#>   .. ..$ I350   : num [1:100] 63.2 57.9 49.8 56.1 50.6 ...
#>   .. ..$ R25    : num [1:100] 125 127 154 176 115 ...
#>   .. ..$ R40    : num [1:100] 275 280 334 367 264 ...
#>   .. ..$ R100   : num [1:100] 618 620 626 637 636 ...
#>   .. ..$ R200   : num [1:100] 519 522 506 500 539 ...
#>   .. ..$ R300   : num [1:100] 430 442 400 383 435 ...
#>   .. ..$ R350   : num [1:100] 517 566 434 415 472 ...
#>   ..$ validation:'data.frame':   80 obs. of  21 variables:
#>   .. ..$ b      : num [1:80] 3.43e-05 7.55e-05 8.78e-05 5.54e-05 6.73e-05 ...
#>   .. ..$ mu     : num [1:80] 6.23e-05 5.58e-05 4.46e-05 9.26e-05 4.61e-05 ...
#>   .. ..$ beta1  : num [1:80] 0.269 0.274 0.242 0.264 0.27 ...
#>   .. ..$ beta2  : num [1:80] 0.178 0.114 0.179 0.193 0.171 ...
#>   .. ..$ beta3  : num [1:80] 0.395 0.433 0.353 0.312 0.44 ...
#>   .. ..$ epsilon: num [1:80] 0.1342 0.1364 0.0794 0.193 0.1951 ...
#>   .. ..$ alpha  : num [1:80] 0.0134 0.0191 0.013 0.0117 0.0153 ...
#>   .. ..$ gamma  : num [1:80] 0.0581 0.0749 0.0596 0.0674 0.0607 ...
#>   .. ..$ omega  : num [1:80] 0.00268 0.00213 0.0034 0.00314 0.00289 ...
#>   .. ..$ I25    : num [1:80] 188 142 104 222 236 ...
#>   .. ..$ I40    : num [1:80] 251 170 160 240 253 ...
#>   .. ..$ I100   : num [1:80] 31.7 17.4 74.2 19 18.4 ...
#>   .. ..$ I200   : num [1:80] 1.908 0.176 7.73 1.662 1.098 ...
#>   .. ..$ I300   : num [1:80] 26.09 1.84 33.69 21.19 61.72 ...
#>   .. ..$ I350   : num [1:80] 66.8 25.7 56.4 68.4 47.7 ...
#>   .. ..$ R25    : num [1:80] 136.7 147.9 81.2 194.8 184.7 ...
#>   .. ..$ R40    : num [1:80] 327 323 194 427 408 ...
#>   .. ..$ R100   : num [1:80] 684 629 616 699 667 ...
#>   .. ..$ R200   : num [1:80] 560 527 555 535 520 ...
#>   .. ..$ R300   : num [1:80] 461 427 467 422 456 ...
#>   .. ..$ R350   : num [1:80] 548 416 527 509 603 ...
```

Run first wave.

```
sir_hmer <- run_wave(sir_hmer)
#> Fitting regression surfaces...
#> I25
#> I40
#> I100
#> I200
#> I300
#> I350
#> R25
#> R40
#> R100
#> R200
#> R300
#> R350
#> Building correlation structures...
#> I25
#> I40
#> I100
#> I200
#> I300
#> I350
#> R25
#> R40
#> R100
#> R200
#> R300
#> R350
#> Creating emulators...
#> Performing Bayes linear adjustment...
# sir_hmer@run_wave
```