Fitting the stable isotope labelling model to data

Ada Yan

2022-06-09

Introduction

This document outlines how to fit the stable isotope labelling model to data.

Code to fit model to data

Model fitting is conducted using a Bayesian framework, using the NUTS sampler implemented in Stan. The `rstan` package provides an interface between R and Stan.

Functions used to fit the model to data are in a private github repository. Clone this repository:

`git clone https://github.com/ada-w-yan/kirdynamics`

There are two directory paths that need to be changed in the repository. The first is in `R/read_data.R`: change line 5 to the directory in which the input files are stored.

List of input files:

- Labelling_Study_KIR_Expression_Data.csv
- Labelling_Study_Participant_KIR_Data.csv
- Labelling_Study_Participant_Lymphocyte_Data.csv
- Labelling_Study_Participant_Mono_and_Granulocyte_Data.csv
- Labelling_Study_Participant_Saliva_Data.csv

The second is in `R/general_functions.R`: change `"~/git_repos/kirlabelling/"` on line 6 to the directory of the git repository.

Then run the following code:

```
ids <- get_ids(threeDL2_neg = FALSE) # get character vector of participant ids
# for the participant ids wich we're fitting, get all combinations of
# cell populations (i.e. CD8+ TCM and CD8+ TEMRA) and licensing statuses,
# excluding NK cells
cells_lic.status <- lapply(ids, get_cells_lic.status, CD8_only = TRUE)

fit_filename <- paste0(dir_name, "fit.rds")
pred_filename <- paste0(dir_name, "pred.rds")


# name of stan model which we're fitting
model <- "cell_pop_null_model"
```

```
# fit the model: note covariates = "null" means we fit the hierarchical model assuming that
# parameters for each individual and cell population are drawn from the same
```

```
# distribution regardless of disease status, licensing status etc.
# The default value of adapt_delta is 0.8; increasing it improves convergence at the expense of speed
fit <- fit_linear_stan(ids, "Monocytes", cells_lic.status, model, covariates = "null", adapt_delta = 0.9
# save posterior distribution
saveRDS(fit, fit_filename)

# calculate fraction of label over time for saliva, monocytes and lymphocytes
# for each draw of the posterior distribution
saliva_pred_model <- "saliva_model_beta_0_two_phase_pred"
gm_pred_model <- "gm_model_pred"
lymphocyte_pred_model <- "lymphocyte_model_pred"
pred <- pred_null_from_fit(fit, saliva_pred_model, gm_pred_model, lymphocyte_pred_model)
# save fraction of label
saveRDS(pred, pred_filename)
```

This code should create two files in the `"hierarchical_model_fit_files` folder: * `fit.rds` which contains the data and the fitted parameters * `pred.rds` which contains the trajectories of the fraction of label for 100 samples in the posterior distribution

Code to calculate summary statistics

Now we retrieve p and d^* for each lymphocyte population, calculate the 2.5, 50 and 97.5th percentiles and standard deviation, and write to .rds and .csv files.

```
# n is the index of the participant id (i.e. ranges from 1 to the number of participants)
# par_name can be "p", "dstar" or "delay"
get_par <- function(par_name) {
  ids <- get_ids(threeDL2_neg = FALSE)
  par_median_rds_filename <- gsub("/fit", paste0("/", par_name), fit_filename)
  par_median_csv_filename <- gsub("rds", "csv", par_median_rds_filename)
  fit <- fit_filename %>% readRDS

  inner_wrapper <- function(n) {
    # get indices of cell populations for that participant id
    pop_idx <- which(fit$data$C_to_N == n)
    # get disease, functional iKIR count, cell population and licensing status info
    cells_lic.status <- get_id_data(ids[n]) %>%
      filter_cell_data(CD8_only = TRUE)
    # get samples from posterior distribution
    par_name_indexed <- vcapply(pop_idx, index_par_name, par_name = par_name)
    pars <- extract_fit(fit$fit, N_samples = 0, par_name_indexed, drop = FALSE)

    # calculate median and 95% ci
    pars_ci <- apply(pars, 2, quantile, probs = c(0.025, 0.5, 0.975))
    pars_sd <- apply(pars, 2, sd)
    # collate into tibble
    pars <- tibble(lower = pars_ci[1,],
                   median = pars_ci[2,],
                   upper = pars_ci[3,],
                   sd = pars_sd,
                   id = ids[n],
                   cells = cells_lic.status$cells,
                   lic.status = cells_lic.status$lic.status,
```

```
                  disease = cells_lic.status$disease,
                  functional_iKIR_count = cells_lic.status$functional_iKIR_count,
                  iKIR_count = cells_lic.status$iKIR_count)
    pars
  }
  pars <- lapply(seq_along(ids), inner_wrapper) %>%
    bind_rows
  saveRDS(pars, file = par_median_rds_filename)
  write.csv(pars, file = par_median_csv_filename, row.names = FALSE)
  pars
}
```

```
# calculate summary statistics for p
get_par("p")
get_par("dstar")
```

We do the same for the non-lymphocyte parameters, f, r_2 and b_w. Note that r_2 is called z in the code.

```
# retrieve non-lymphocyte parameters for each participant, calculate summary statistics and
# write to .rds and .csv files
# n is the index of the participant id (i.e. ranges from 1 to the number of participants)
# par_name can be "frac", "z" or "b_w"
get_par_id <- function(par_name) {
  ids <- get_ids(threeDL2_neg = FALSE)
  fit <- fit_filename %>% readRDS

  inner_wrapper <- function(n) {
    # get disease, functional iKIR count info
    cells_lic.status <- get_id_data(ids[n]) %>%
      slice(1)
    # get more detailed KIR info
    kir_data <- read_kir_data(ids[n]) %>%
      t %>%
      as_tibble %>%
      select(id, kir.2DL1, kir.2DL2, kir.2DL3, kir.3DL1, kir.3DL2,
             lic.2DL1, lic.2DL2, lic.2DL3, lic.3DL1, lic.3DL2,
             inhibitory.score, inhibitory.score.3DL2)
    # get samples from posterior distribution
    par_name_indexed <- index_par_name(n, par_name = par_name)
    pars <- extract_fit(fit$fit, N_samples = 0, par_name_indexed, drop = TRUE)
    # calculate median and 95% ci
    pars_sd <- sd(pars)
    pars <- quantile(pars, probs = c(0.025, 0.5, 0.975))
    # collate into tibble
    pars <- tibble(par_name = par_name,
                   lower = pars[1],
                   median = pars[2],
                   upper = pars[3],
                   sd = pars_sd,
                   id = ids[n],
                   disease = cells_lic.status$disease,
                   functional_iKIR_count = cells_lic.status$functional_iKIR_count,
                   iKIR_count = cells_lic.status$iKIR_count) %>%
      full_join(kir_data, by = "id")
    pars
```

```
  }
  lapply(seq_along(ids), inner_wrapper) %>%
    bind_rows
}

# retrive frac, b_w, z for all participants
all_no_lymphocyte_pars <- lapply(c("frac", "b_w", "z"), get_par_id) %>%
  bind_rows
```

We also calculate summary statistics for the population-level δ.

```
# retrieve population-level delta
fit <- fit_filename %>% readRDS
delta <- extract_fit(fit$fit, N_samples = 0, "delta", drop = TRUE) %>%
  quantile(probs = c(0.025, 0.5, 0.975))
delta <- tibble(par_name = "delta", lower = delta[1], median = delta[2], upper = delta[3])
all_id_pars <- bind_rows(all_no_lymphocyte_pars, delta)
par_median_rds_filename <- gsub("/fit", paste0("/", "id_pars"), fit_filename)
par_median_csv_filename <- gsub("rds", "csv", par_median_rds_filename)
saveRDS(all_id_pars, file = par_median_rds_filename)
write.csv(all_id_pars, file = par_median_csv_filename, row.names = FALSE)
```