DIFFpop Data Application

Jeremy Ferlic, Jiantao Shi, Thomas O. McDonald, and Franziska Michor

Application: Binary Labeling of the Mouse Hematopoietic System

Background

To begin our example, let us consider a subtree of the hematopoietic system shown in Figure 1.

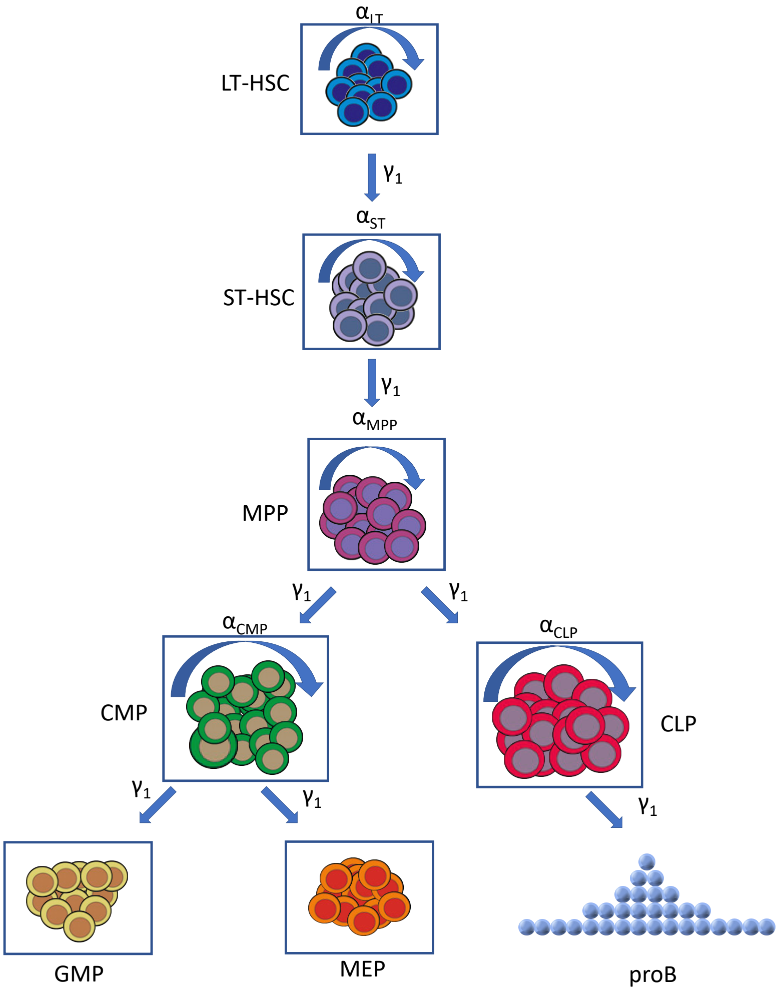


Figure 1: Hematopoietic system modeled using FixedPops. Abbreviations: long-term hematopoietic stem cell (LT-HSC), short-term hematopoietic stem cell (ST-HSC), multi-potent progenitor (MPP), common myeloid progenitor (CMP), common lymphoid progenitor (CLP), granulocyte-macrophage progenitor (GMP), megakaryocyte-erythroid progenitor (MEP). Events between populations consist of mitosis (α) and mitosis-independent differentiation (γ_1).

This system has been studied in mice by researchers who have developed a mouse model that introduces a fluorescent tag into certain cell population (Busch, 2015). Once activated, this tag, integrated into the genome of the cell, will continue to be present in the progeny of the initially labeled cell. The uptake and loss of the label can then be observed as it descends through the differentiation hierarchy. Assuming that the system has reached a steady state with no major population fluxes, Busch et al. employed an ordinary differential equation model to estimate the sizes of the compartments and rates of transitions between compartments. To validate our tool using the experimental results obtained in (Busch, 2015), we used the previously derived parameter estimates determined to provide the best fit to the data and predicted, using our tool, the cell numbers at time points not used for parameter estimation.

Using DIFFpop in R

The following script was used to run simulations of the hematopoietic system as modeled by Busch et al. in their 2015 Nature paper. To mimic the experimental procedure, we initially labeled only LT-HSC cells. Because the parameter estimates were calculated under the assumption that steady state hematopoiesis had been reached, we modeled the system using FixedPops, which maintain constant population sizes across the system.

```r
library(foreach)
library(doParallel)

cores = detectCores()
cl = makeCluster(cores[1]-1)
registerDoParallel(cl)

ntrials = 1000

foreach(i_=1:ntrials) %dopar%{
  print(i_)
  library(diffpop)

  # Simulation size and label parameter
  nLT = 5000
  LT_lbl = 0.01

  # Blank DiffTree object
  tree = DiffTree()

  # Add all populations to tree using population sizes estimated from Busch et al.
  FixedPop(tree, "LT", nLT, LT_lbl)
  FixedPop(tree, "ST", as.integer(2.9*nLT), 0.0)
  FixedPop(tree, "MPP", as.integer(9*nLT), 0.0)
  FixedPop(tree, "CMP", as.integer(39*nLT), 0.0)
  FixedPop(tree, "CLP", as.integer(13*nLT), 0.0)
  FixedPop(tree, "GMP", as.integer(0.24*39*nLT), 0.0)
  FixedPop(tree, "MEP", as.integer(0.39*39*nLT), 0.0)
  FixedPop(tree, "proB", as.integer(108*13*nLT), 0.0)

  # Add self-renewal/mitosis events
  addEdge(tree, "LT", "LT", "alpha", 0.009)
  addEdge(tree, "ST", "ST", "alpha", 0.042)
  addEdge(tree, "MPP", "MPP", "alpha", 4)
  addEdge(tree, "CLP", "CLP", "alpha", 3.00)
  addEdge(tree, "CMP", "CMP", "alpha", 4)

  # Add differentiation events
  # Note: Busch et al. assume mitosis-independent differentiation
  addEdge(tree, "LT", "ST", "gamma1", 0.009)
  addEdge(tree, "ST", "MPP", "gamma1", 0.045)
  addEdge(tree, "MPP", "CLP", "gamma1", 0.022)
  addEdge(tree, "MPP", "CMP", "gamma1", 3.992)
  addEdge(tree, "CLP", "proB", "gamma1", 2.000)
  addEdge(tree, "CMP", "GMP", "gamma1", 2)
  addEdge(tree, "CMP", "MEP", "gamma1", 3)

  # Set LT population as root of tree
  setRoot(tree, "LT")

  # Simulate tree for 800 time units (days)
  # Note: we use FixedPops here because parameters were
  #       estimated for steady state hematopoiesis
  simulateTree(tree = tree,
               fixed = TRUE,
               time = 800,
               indir = paste("input/", i_, sep = ""),
               outdir = "output/",
               census = -1)

}

stopCluster(cl)
```

The above script was run on a cluster with multiple simulation trajectories running in parallel.

Comparing Simulation Output to Experimental Data

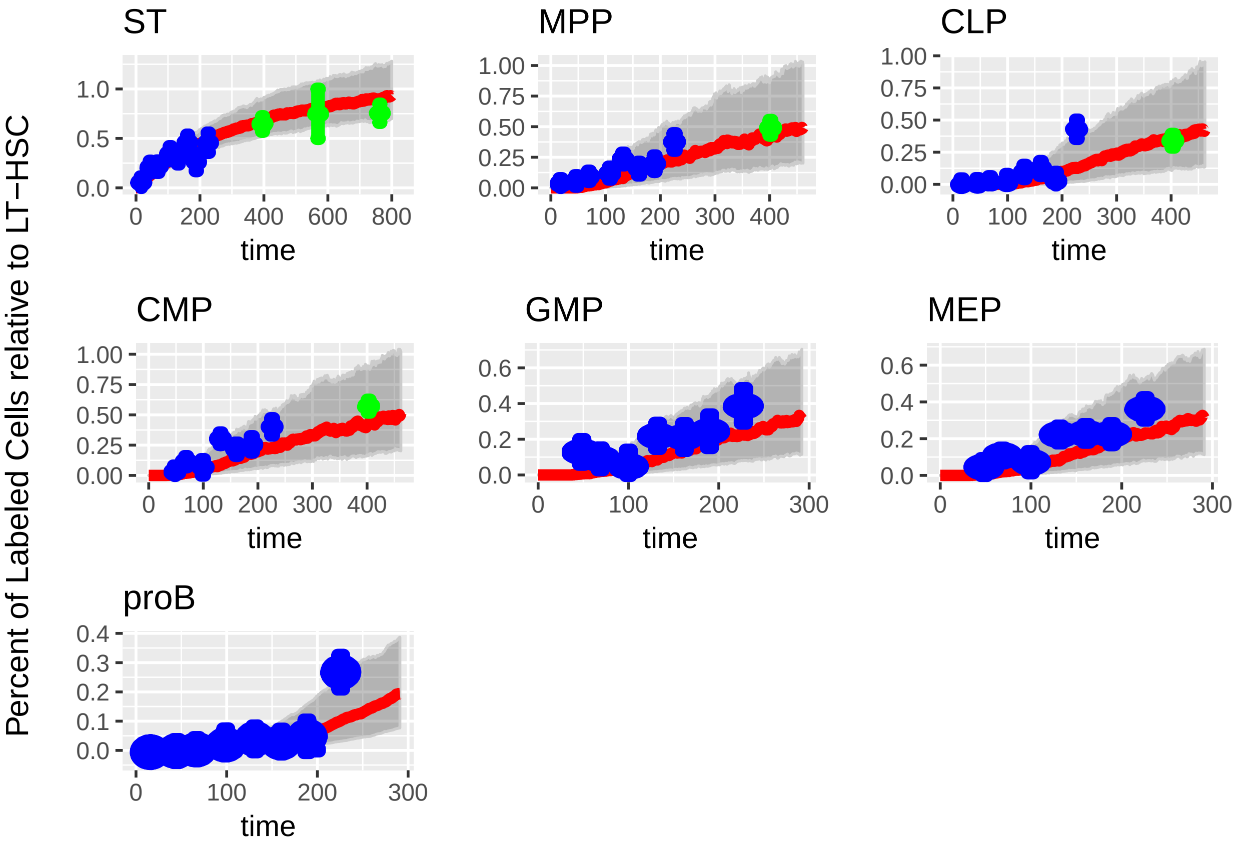
The raw experimental data from the mouse model provides the percentage of cells in each population that express the label. These raw percentages are then normalized by the label percentage in the LT-HSC population. This same information can be calculated from the information in the simulation label output files (*prefix*_label.csv). We downloaded all of these label output files from the cluster and stored them in a local directory. We then ran the following script to load the data into R and transform the raw label percentages to percentages relative to the LT-HSC population.

```
library(reshape2)
library(ggplot2)
library(grid)
library(gridExtra)


# Set working directory to that which contains our label output files
inDir = "C:/Users/Jeremy/Desktop/diffpop_review/label_newpop2/"
setwd(inDir)

# Generate a list of all filenames
lblfiles = list.files(inDir, pattern="^out.*_label.csv$", full.names=F)
```

```r
# Read in all of the label files into one dataframe
myMergedData <-
  do.call(rbind,
          lapply(lblfiles, read.csv))

# Label each row with a corresponding file id
nfiles = length(lblfiles)
myMergedData$id = rep(1:nfiles, each = 801)

# Remove any data point where the LT label has died out
#        (necessary to divide by it in next step)
myMergedData = myMergedData[myMergedData$LT != 0.0,]
myMergedData[,3:9] = myMergedData[,3:9] / myMergedData[,2]
```

We summarized the 1,000 trajectories by calculating the median trajectory, as well as the 25th and 75th percentiles to act as confidence bounds. We then looped over all of the populations to plot. We also added in the data points at the best resolution possible that come from the Busch et al. paper (Figure 3 and Extended Data Figure 6). In the following plots, the blue data points were used by Busch et al. to fit the model parameters, whereas the green data points were used for validation.

```r
# Melt the data set in order to plot using ggplot2
myMergedData = melt(myMergedData, id.vars = c("time", "id"))

# Rename some columns
names(myMergedData) = c("time", "id", "pop", "value")

# Experimental results directory
exp_dir = "C:/Users/Jeremy/Desktop/diffpop_review/"

# Plotting function
plot_dat = function(pop, df){
  exp_pts = read.csv(paste(exp_dir, tolower(pop), "_fit.csv", sep = ""), header = F)
  names(exp_pts) = c("time", "value")
  exp_pts$id = 1
  exp_pts$color = "blue"

  if(file.exists(paste(exp_dir, tolower(pop), "_pred.csv", sep = ""))){
    pred_pts = read.csv(paste(exp_dir, tolower(pop), "_pred.csv", sep = ""), header = F)
    names(pred_pts) = c("time", "value")
    pred_pts$id = 1
    pred_pts$color = "green"

    exp_pts = rbind(exp_pts, pred_pts)
  }

  p <- ggplot(df[df$pop == pop,], aes(x = time, y = value, group = id))
  p = p +  stat_summary(aes(group = 1), geom = "ribbon",
                        fun.ymin = function(x) quantile(x, 0.25),
                        fun.ymax = function(x) quantile(x, 0.75),
                        col = "grey80", alpha = 0.3) +
    stat_summary(aes(group = 1), geom = "line", fun.y = median, col = "red", size = 2) +
    geom_point(data = exp_pts, col = exp_pts$color, alpha = 1.0) +
    xlim(0, max(exp_pts$time + 50)) +
    ggtitle(pop) +
    ylab("")

  return(p)
}

# Apply plotting function to each population
plot.list = lapply(c("ST", "MPP", "CLP", "CMP", "GMP", "MEP", "proB"),
                   plot_dat, df = myMergedData)

# Arrange the plots in a nice grid
grid.arrange(grobs = plot.list,
             left = textGrob("Percent of Labeled Cells relative to LT-HSC",
                             rot = 90, vjust = 1))
```



References

Busch, Katrin, et al. “Fundamental properties of unperturbed haematopoiesis from stem cells in vivo.” Nature 518.7540 (2015): 542.