Subtraction_vignette

AUTHOR
Alec Barrett

LittleBites Subtraction Vignette

load in libraries

```
library(stringr)
library(LittleBites)
library(bayestestR)
library(tibble)
library(purrr)
```

Load in the Vignette dataset, these are subsets of the data used in the CeNGEN group paper currently on Biorxiv. Full datasets are downloadable from cengen.org

```
bulk <- read.table('Data/Bulk_data_5k.tsv')

sc <- read.table('Data/singleCell_reference_5k.tsv')
```

Load ground truth expression data

```
gt_matrix <- read.table('Data/ubiquitous_non_neuronal/ubituiqtous_and_nonNeuronal_gt_gen

train_test <- readRDS('Data/ubiquitous_non_neuronal/ubituiqtous_and_nonNeuronal_gt_genes

gt_train <- gt_matrix[train_test$training_genes,]
gt_test <- gt_matrix[train_test$testing_genes,]
```

Calculate specificity scores from the single cell reference

In general, this approach tries to be quite conservative on which genes to subtract out, and how much to subtract. One way to guides this process is to give a gene-level weight based on some metric of how much subtraction should be done at each step.

Here we presume that genes expressed in only 1 cell type are free-game to subtract out, while genes expressed in increasing numbers of cell types should be subtracted with more caution. Thus we can use a tissue specificity score to generate a 0 to 1 weight for each gene, where a score close to 1 indicates effectively perfect specificity (only in intestines for instance), while a score close to 0 indicates ubiquitous expression, and scores in between indicate that the gene is expressed in some smaller proportion of cell types.

I use the Spm measure here, because it is quite sensitive to genes being expressed in even 2 tissue types, and scores fall in value quite quickly, resulting in a more conservative subtraction with less risk of over-fitting.

```
cells <- colnames(sc) |> unique() |> sort()

## calculate specificty scores for each gene using the single cell reference
specificity <- apply(sc |> log1p(), 1, LittleBites::Spm)
```

Curate a list of potential contaminant cell types that you want to remove

In this case I remove only non-neuronal contaminants as neuronal profiles are generally too similar and it can produce major problems during subtraction,

```
contaminants <- c('Excretory', 'Glia', 'Hypodermis', 'Intestine', 'Muscle_mesoderm', 'Re

neurons <- cells[!cells %in% c(contaminants)]
```

make a sample x cell_types matrix describing each cell type to be modeled in the subtraction process

the first column should be the target cell type, and the remaining columns should represent putative contaminants

This matrix will tell the algorithm 1) which cell type we're targeting to aid in modeling, and 2) which profiles are designated for removal

```
cell_types_matrix <- sapply(colnames(bulk), function(sample_){

  cell <- str_split_fixed(sample_, 'r', 2)[,1]

  return(c(cell, contaminants))


}) |> t()

cell_types_matrix
```

```
       [,1]  [,2]       [,3]  [,4]       [,5]       [,6]
AFD#38 "AFD" "Excretory" "Glia" "Hypodermis" "Intestine" "Muscle_mesoderm"
AFD#39 "AFD" "Excretory" "Glia" "Hypodermis" "Intestine" "Muscle_mesoderm"
AIM#193 "AIM" "Excretory" "Glia" "Hypodermis" "Intestine" "Muscle_mesoderm"
AIN#185 "AIN" "Excretory" "Glia" "Hypodermis" "Intestine" "Muscle_mesoderm"
ASI#154 "ASI" "Excretory" "Glia" "Hypodermis" "Intestine" "Muscle_mesoderm"
ASI#155 "ASI" "Excretory" "Glia" "Hypodermis" "Intestine" "Muscle_mesoderm"
AVK#110 "AVK" "Excretory" "Glia" "Hypodermis" "Intestine" "Muscle_mesoderm"
AVK#112 "AVK" "Excretory" "Glia" "Hypodermis" "Intestine" "Muscle_mesoderm"
AWB#52 "AWB" "Excretory" "Glia" "Hypodermis" "Intestine" "Muscle_mesoderm"
BAG#119 "BAG" "Excretory" "Glia" "Hypodermis" "Intestine" "Muscle_mesoderm"
PHA#206 "PHA" "Excretory" "Glia" "Hypodermis" "Intestine" "Muscle_mesoderm"
PVM#126 "PVM" "Excretory" "Glia" "Hypodermis" "Intestine" "Muscle_mesoderm"
PVP#3  "PVP" "Excretory" "Glia" "Hypodermis" "Intestine" "Muscle_mesoderm"
PVQ#236 "PVQ" "Excretory" "Glia" "Hypodermis" "Intestine" "Muscle_mesoderm"
RIC#135 "RIC" "Excretory" "Glia" "Hypodermis" "Intestine" "Muscle_mesoderm"
RIM#224 "RIM" "Excretory" "Glia" "Hypodermis" "Intestine" "Muscle_mesoderm"
RIS#131 "RIS" "Excretory" "Glia" "Hypodermis" "Intestine" "Muscle_mesoderm"
SMB#196 "SMB" "Excretory" "Glia" "Hypodermis" "Intestine" "Muscle_mesoderm"
VC#140  "VC"  "Excretory" "Glia" "Hypodermis" "Intestine" "Muscle_mesoderm"
VC#142  "VC"  "Excretory" "Glia" "Hypodermis" "Intestine" "Muscle_mesoderm"
  [,7]
AFD#38  "Reproductive"
AFD#39  "Reproductive"
AIM#193 "Reproductive"
AIN#185 "Reproductive"
ASI#154 "Reproductive"
ASI#155 "Reproductive"
AVK#110 "Reproductive"
AVK#112 "Reproductive"
AWB#52  "Reproductive"
BAG#119 "Reproductive"
PHA#206 "Reproductive"
PVM#126 "Reproductive"
PVP#3  "Reproductive"
PVQ#236 "Reproductive"
RIC#135 "Reproductive"
RIM#224 "Reproductive"
RIS#131 "Reproductive"
SMB#196 "Reproductive"
VC#140  "Reproductive"
VC#142  "Reproductive"
```

Run Subtraction!

```
bulk_subtracted <- subtraction(bulk = bulk,
                               reference = sc,
                               cell_types_matrix = cell_types_matrix,
                               training_matrix = gt_train,
                               specificity_weights = specificity,
                               verbose = F)
```

```
[1] "done"
```

Plotting the effects

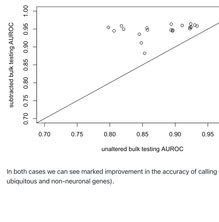
Here we'll calculate AUROC scores for each sample individually using the training and test genes to see how the subtraction performed.

First the training genes:

```
per_sample_bulk_diags <- lapply(colnames(bulk), function(sample_){

  cell_type <- str_split_fixed(sample_, 'r', 2)[,1]

  prediction <- bulk[train_test$training_genes, sample_]
  case <- gt_train[,cell_type]

  diags <- tibble(threshold = c(0,2**seq(-17,12,0.05)),
                  TPR = map_dbl(threshold, ~get_tpr(prediction, case, .x)),
                  FPR = map_dbl(threshold, ~get_fpr(prediction, case, .x)),
                  counts = "raw")

  auroc <- bayestestR::auc(diags$FPR, diags$TPR)

  return(auroc)
}) |> unlist()
names(per_sample_bulk_diags) <- colnames(bulk)



per_sample_sub_diags <- lapply(colnames(bulk_subtracted), function(sample_){

    cell_type <- str_split_fixed(sample_, 'r', 2)[,1]

    prediction <- bulk_subtracted[train_test$training_genes, sample_]
    case <- gt_train[,cell_type]

    diags <- tibble(threshold = c(0,2**seq(-17,12,0.05)),
                    TPR = map_dbl(threshold, ~get_tpr(prediction, case, .x)),
                    FPR = map_dbl(threshold, ~get_fpr(prediction, case, .x)),
                    counts = "raw")

    auroc <- bayestestR::auc(diags$FPR, diags$TPR)

    return(auroc)
  }) |> unlist()
names(per_sample_sub_diags) <- colnames(bulk_subtracted)


plot(per_sample_bulk_diags,
     per_sample_sub_diags,
     xlim = c(.7,1), ylim = c(.7,1),
     xlab = 'unaltered bulk training AUROC',
     ylab = 'subtracted bulk training AUROC')
abline(0,1)
```

next the reserved testing genes

```
per_sample_bulk_diags_test <- lapply(colnames(bulk), function(sample_){

  cell_type <- str_split_fixed(sample_, 'r', 2)[,1]

  prediction <- bulk[train_test$testing_genes, sample_]
  case <- gt_test[,cell_type]

  diags <- tibble(threshold = c(0,2**seq(-17,12,0.05)),
                  TPR = map_dbl(threshold, ~get_tpr(prediction, case, .x)),
                  FPR = map_dbl(threshold, ~get_fpr(prediction, case, .x)),
                  counts = "raw")

  auroc <- bayestestR::auc(diags$FPR, diags$TPR)

  return(auroc)
}) |> unlist()
names(per_sample_bulk_diags_test) <- colnames(bulk)



per_sample_sub_diags_test <- lapply(colnames(bulk_subtracted), function(sample_){

  cell_type <- str_split_fixed(sample_, 'r', 2)[,1]

  prediction <- bulk_subtracted[train_test$testing_genes, sample_]
  case <- gt_test[,cell_type]

  diags <- tibble(threshold = c(0,2**seq(-17,12,0.05)),
                  TPR = map_dbl(threshold, ~get_tpr(prediction, case, .x)),
                  FPR = map_dbl(threshold, ~get_fpr(prediction, case, .x)),
                  counts = "raw")

  auroc <- bayestestR::auc(diags$FPR, diags$TPR)

  return(auroc)
}) |> unlist()
names(per_sample_sub_diags_test) <- colnames(bulk_subtracted)


plot(per_sample_bulk_diags_test,
     per_sample_sub_diags_test,
     xlim = c(.7,1), ylim = c(.7,1),
     xlab = 'unaltered bulk testing AUROC',
     ylab = 'subtracted bulk testing AUROC')
abline(0,1)
```



In both cases we can see marked improvement in the accuracy of calling genes expressed (when using ubiquitous and non-neuronal genes).