```
---
title: "Phenotype_tutorial"
author: "R. Burke Squires, Ryan Brinkman, Adrin Jalali, Nima Aghaeepour"
date: "8/15/2017"
output: html_document
---

```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)
```


flowType/RchyOptimyx and flowDensity
Automated analysis for discovery and diagnosis in Big Flow Cytometry Data

- flowDensity: Pipeline for Diagnosis
  - Finds what you want to find, how you want find it
  - Based on density estimation techniques
  - Seconds per FCS file
  - Identical to the manual practice of 2D gating

- flowType/RchyOptimyx: Pipeline for Discovery
  - You split FCS files into groups
  - Pipeline finds best cell populations that correlate with that split
  - One graph summary of very large datasets
  - Can be used as input to large multi-group studies

R code from vignette source 'RchyOptimyx.Rnw'

Grab the HIV data that is included as example data

Ganesan, A. et al. Immunologic and Virologic Events in Early HIV Infection Predict Subsequent Rate of Progression. Journal of Infectious Diseases 2

United States Military HIV Natural History Study
▶ PBMCs of 466 HIV+ personnel and beneficiaries from Army, Navy, Marines, and Air Force
▶ 13 surface markers and KI-67 (cell proliferation)
▶ Clinical Data: Survival times including 135 events

An event is defined as progression to AIDS or initiation of HAART.


```{r}
library(flowType)
data(HIVData)
data(HIVMetaData)
HIVMetaData <- HIVMetaData[which(HIVMetaData[,'Tube']==2),];
```



```{r}
Labels=(HIVMetaData[,2]=='+')+1;
```



```{r}
library(flowCore)
library(RchyOptimyx)
```



```{r}

##Markers for which cell proportions will be measured.
PropMarkers <- 5:10
```



```{r}

##Markers for which MFIs will be measured.
MFIMarkers <- PropMarkers
```



```{r}

##Marker Names
MarkerNames <- c('Time', 'FSC-A','FSC-H','SSC-A',
                 'IgG','CD38','CD19','CD3',
                 'CD27','CD20', 'NA', 'NA')
```


Apply flowType

```{r}
ResList <- fsApply(HIVData, 'flowType', PropMarkers,
                   MFIMarkers, 'kmeans', MarkerNames);
```


Extract phenotype names
```

```{r}
phenotype.names=unlist(lapply(ResList[[1]]@PhenoCodes,function(x){return(decodePhenotype(x,MarkerNames[PropMarkers],ResList[[1]]@PartitionsPerMarke
names(ResList[[1]]@PhenoCodes)=phenotype.names
```



```{r}
all.proportions <- matrix(0,length(ResList[[1]]@CellFreqs),length(HIVMetaData[,1]))
for (i in 1:length(ResList))
  all.proportions[,i] = ResList[[i]]@CellFreqs / ResList[[i]]@CellFreqs[1]
```



```{r}
Pvals <- vector();
EffectSize <- vector();
for (i in 1:dim(all.proportions)[1]){

  ##If all of the cell proportions are 1 (i.e., the phenotype
  ##with no gates) the p-value is 1.
  if (length(which(all.proportions[i,]!=1))==0){
    Pvals[i]=1;
    EffectSize[i]=0;
    next;
  }
  temp=t.test(all.proportions[i, Labels==1],
              all.proportions[i, Labels==2])
  Pvals[i] <- temp$p.value
  EffectSize[i] <- abs(temp$statistic)
}

Pvals[is.nan(Pvals)]=1
names(Pvals)=phenotype.names

##Bonferroni's correction
## TODO: no good results shows up and there is no adjusted p-value <0.6
selected <- which(p.adjust(Pvals)<0.65);

print(names(selected))
```



```{r}
res<-RchyOptimyx(ResList[[1]]@PhenoCodes, -log10(Pvals),
                 ResList[[1]]@PhenoCodes[selected[38]], factorial(6),FALSE)
plot(res, phenotypeScores=-log10(Pvals), phenotypeCodes=ResList[[1]]@PhenoCodes, marker.names=MarkerNames[PropMarkers], ylab='-log10(Pvalue)')
```



```{r}
res<-RchyOptimyx(pheno.codes=ResList[[1]]@PhenoCodes, phenotypeScores=-log10(Pvals),
                 startPhenotype=ResList[[1]]@PhenoCodes[selected[38]], 2, FALSE)
plot(res, phenotypeScores=-log10(Pvals), phenotypeCodes=ResList[[1]]@PhenoCodes, marker.names=MarkerNames[PropMarkers], ylab='-log10(Pvalue)')
```



```{r}
res<-RchyOptimyx(pheno.codes=ResList[[1]]@PhenoCodes, phenotypeScores=-log10(Pvals),
                 startPhenotype=ResList[[1]]@PhenoCodes[selected[1]], 1, FALSE,trim.level=4)
for (i in 2:length(selected)){
  temp<-RchyOptimyx(pheno.codes=ResList[[1]]@PhenoCodes, phenotypeScores=-log10(Pvals),
                    startPhenotype=ResList[[1]]@PhenoCodes[selected[i]], 1, FALSE,trim.level=4)
  res=merge(res,temp)
}
plot(res, phenotypeScores=-log10(Pvals), phenotypeCodes=ResList[[1]]@PhenoCodes, marker.names=MarkerNames[PropMarkers], ylab='-log10(Pvalue)')
```