# Your experiment title here

Your name

April 24, 2015

## Contents

## 1 Data loading

This is a report for the curatedOvarianData meta-clustering. Initial clustering was done using k-means and selecting k via the maximum mean cluster consensus (I found this gave stable results, but I am still exploring 1 other metric.) First, I loaded up my data just past the CoINcIDE$_getAdjMatrices()function(that'sthefunctionthattakesareallylongtimetorun/ishogging$ $values)$ :

```
> ##grab data matrix list, clust features list
> CoINcIDE_rankFeatures <- "/home/kplaney/ovarian_analysis/metaFeatures_200.RData.gzip"
> load(CoINcIDE_rankFeatures)
> metaFeatures <- metaFeatures
> CoINcIDE_clusterOutput <-  "/home/kplaney/ovarian_analysis/curatedOvarianData_kmeansConsensus_200Feature
> load(CoINcIDE_clusterOutput)
> clusterOutput <- kmeansConsensus
> load("/home/kplaney/ovarian_analysis/esets_proc_TCGAcombat.RData.gzip")
> source("/home/kplaney/gitRepos/CoINcIDE/coincide/CoINcIDE/R/CoINcIDE_geneExprProcess.R")
> source("/home/kplaney/gitRepos/CoINcIDE/coincide/CoINcIDE/R/CoINcIDE_communityDetection.R")
> source("/home/kplaney/gitRepos/CoINcIDE/coincide/CoINcIDE/R/CoINcIDE_visualization.R")
> source("/home/kplaney/gitRepos/CoINcIDE/coincide/CoINcIDE/R/CoINcIDE_metaClusterAnalysis.R")
> #now format just as a list of data matrices.
> dataMatrixList <- exprSetListToMatrixList(esets,featureDataFieldName="gene")
> names(dataMatrixList) <- names(esets)
> ##do for each 200,500,1000,2000 (load different metaFeatures RData object each time.)
> #remove datasets with too many missing top gene features
> if(length(metaFeatures$datasetListIndicesToRemove)>0){
+
+   dataMatrixList <- dataMatrixList[-metaFeatures$datasetListIndicesToRemove]
+
+ }
> origToNewIndexMap <- cbind(1:length(dataMatrixList),na.omit(match(names(dataMatrixList),names(esets))))
> clustSampleIndexList <-  kmeansConsensus$clustSampleIndexList_meanConsensusCluster
> clustFeatureIndexList <- kmeansConsensus$clustFeatureIndexList_meanConsensusCluster
> CoINcIDE_computeEdgesObject <- "/home/kplaney/ovarian_analysis/kmeansConsensus_200F_meanMatrix_distCor.R
> load(CoINcIDE_computeEdgesObject)
> output <- kmeansConsensus_200F_meanMatrix_distCor
> inputVariablesDF <- output$inputVariablesDF
> computeTrueSimilOutput <- output$computeTrueSimilOutput
> pvalueMatrix <- output$pvalueMatrix
> clustIndexMatrix <- output$clustIndexMatrix
> ###inputs for edge detection
> meanEdgePairPvalueThresh <- .05
> indEdgePvalueThresh <- .1
> minTrueSimilThresh <- .8
> maxTrueSimilThresh <- Inf
> clustSizeFractThresh <- inputVariablesDF$clustSizeFractThresh
> clustSizeThresh <- inputVariablesDF$clustSizeThresh
> fractFeatIntersectThresh <- inputVariablesDF$fractFeatIntersectThresh
>   numFeatIntersectThresh <- inputVariablesDF$numFeatIntersectThresh
> saveDir <- "/home/kplaney/ovarian_analysis/"
```

### 1.1 Input variables used to derive adjacency matrix

```
> message("Input variables used to derive the adjacency matrix:\n")
> inputVariablesDF
                 date edgeMethod numParallelCores minTrueSimilThresh maxTrueSimilThresh
1 2015-04-21 14:47:54    distCor                8               0.25                Inf
   sigMethod maxNullFractSize numSims includeRefClustInNull fractFeatIntersectThresh
1 meanMatrix              0.2     500                  TRUE                      0.8
  numFeatIntersectThresh clustSizeThresh clustSizeFractThresh
1                    150               5                 0.05
> message("There were ",nrow(clustIndexMatrix), " total input clusters from ",length(unique(clustIndexMatri
> message("The total number of input features was ",length(metaFeatures$finalFeatures))
> message("Across the entire square (nonsymmetric) p-value matrix, there are ",length(which(pvalueMatrix<=
> message("Across the entire square (nonsymmetric) p-value matrix, there are ",length(which(pvalueMatrix<=
> message("Across the entire square (symmetric) similarity matrix, there are ",length(which(computeTrueSimi
> message("Across the entire square (symmetric) similarity matrix, there are ",length(which(computeTrueSimi
```

## 2 Network Analysis

As we can see in these plots, 4 meta-clusters remained after edge filtering.

```
> finalEdgeInfo <- assignFinalEdges(computeTrueSimilOutput=computeTrueSimilOutput,pvalueMatrix=pvalueMatri
+                                 meanEdgePairPvalueThresh=meanEdgePairPvalueThresh,
+                                 minTrueSimilThresh=minTrueSimilThresh,maxTrueSimilThresh=maxTrueSimilThres
+                                 fractFeatIntersectThresh=fractFeatIntersectThresh,numFeatIntersectThresh=nu
+                                 clustSizeThresh=clustSizeThresh, clustSizeFractThresh= clustSizeFractThres
+ )
> commInfo <- findCommunities(edgeMatrix=finalEdgeInfo$filterEdgeOutput$edgeMatrix,edgeWeightMatrix=finalEd
+                                           clustIndexMatrix=output$clustIndexMatrix,fileTag="autoReport",
+                               saveDir=saveDir,minNumUniqueStudiesPerCommunity=3,clustMethodName="",
+                               commMethod=c("edgeBetween"),
+                               makePlots=TRUE,saveGraphData=FALSE,plotToScreen=TRUE)
>
>
> # advancedNetworkPlots(communityMembership=commInfo,
> #                                    brewPal = c("Set3"),
> #                                    saveDir=saveDir,saveName="network",
> #                      plotToScreen=TRUE)$network_stats
```

## 3 Gene meta-rank Analysis

I ranked genes within each meta-cluster for all samples, and then ran a Kruskal test to see which genes significantly stratified/differentiated patients across the 4 meta-clusters. I still need to implement GSEA; it turns out there's a base GSEA package in Biocondcutor so I've decided to just adapt my code and use their baseline functions.

In the heatmap: red means that gene was ranked high in terms of expression level for patients in that meta-cluster (I took the median rank across all samples in a meta-cluster to create the heatmap. Only significant genes are shown in the heatmap but at over 80 significant genes, of course the gene names are illegible...I print out the top 20 genes below.)

```
> aggregateData <- returnSampleMemberMatrix(clustSampleIndexList,dataMatrixList,communityInfo=commInfo)
> binInfo <- binarizeMetaclustStudyStatus(aggregateData$sampleClustCommKey)
> rankInfo <- computeRankMatrix(metaClustSampleNames=binInfo$metaClustSampleNames,
+                               featureNames=metaFeatures$finalFeatures,dataMatrixList,
+                               sampleClustCommKey=aggregateData$sampleClustCommKey,onlyIntersectingFeat=TI
> pvalueInfo <- computeFeaturePvalues(rankMatrix=rankInfo$rankMatrix,featureNames=rankInfo$filteredFeatures
> message("There are ",length(which(pvalueInfo$fdr.qvalue<=.05)), " genes with an FDR corrected p-value bel
> message("Top 20 significant genes:\n")
> rownames(pvalueInfo[which(pvalueInfo$fdr.qvalue<=.05)[1:20],])

 [1] "COL11A1" "MMP7"    "DEFB1"   "C7"      "MAL"     "LUM"     "SST"     "NNMT"
 [9] "VCAN"    "MFAP5"   "INHBA"   "CDKN2A"  "CXCL10"  "FOS"     "KLK10"   "CHI3L1"
[17] "TFAP2A"  "GPX3"    "RARRES1" "TAGLN"

> cat("\n")


> #not returning all communities
> metaMatrix <- commMedianRank(rankInfo$rankMatrix[which(pvalueInfo$fdr.qvalue<=.05),],rankInfo$groupings)
> message("Red in heatmap means genes were ranked higher across all samples in that meta-cluster.")
> plotMetaAnalysis(metaMatrix,saveFile=FALSE,plotToScreen=TRUE,
+                              saveDir=saveDir,fileTag="test",
+                              plotTitle="Median rank\nacross all samples/studies",
+                              key.xlab="")
>
```

## 3.1 Survival

I'm still working on determing which long-term and binary variables have the least amount of NAs across the samples in these meta-clusters, but it does look like the binary vital$_{s}tatusvariable(aliveordead)andcontinousdays_{t}o_{d}eathvariablesprovidesurv$
$yearcutoff.$

■= check: are the meta-clusters at least reasonably balanced? table(groupings)

load("/home/kplaney/ovarian$_{a}nalysis/esets_{p}roc_{T}CGAcombat.RData.gzip")phenoMasterDF < -createPhenoMasterTableFr$
$esets)save(phenoMasterDF, file = "/home/kplaney/ovarian_{a}nalysis/curatedOvarian_{p}henoMasterDF.RData.gzip", compre$
$"gzip")load("/home/kplaney/ovarian_{a}nalysis/curatedOvarian_{p}henoMasterDF.RData.gzip")studynumberswon'talignhere;$

origToNewIndexMap <- data.frame(origToNewIndexMap,stringsAsFactors=FALSE) colnames(origToNewIndexMap) <- c("studyNum","origStudyNum") sampleClustCommKey <- join(sampleClustCommKey,origToNewIndexMap,by="studyNum",type="full",
sampleClustCommPhenoData <- addClinicalVarToNodeAttributes(sampleClustCommKey,phenoMasterDF=phenoMasterDF)

survival analysis outcomesVarBinary="vital$_{s}tatus"outcomesVarCont = "days_{t}o_{d}eath"CutoffPointYears = 5uniquePatientID =$
$"unique_{p}atient_{I}D"groupingTerm = "community"$

only take samples with the groupingTerm you're looking at. sampleClustCommPhenoData <- sampleClustCommPhenoData[which(!is.na(sampleClustCommPhenoData[, groupingTerm])), ] remove samples with NA values. groupings <- sampleClustCommPhenoData[, groupingTerm] groupings <- groupings[which(!is.na(sampleClustCommPhenoData[,outcomesVarBinary]))]; outcomesData <- sampleClustCommPhenoData[which(!is.na(sampleClustCommPhenoData[,outcomesVarBinary])),];

keep samples with NA days to event for now? hmm...one meta-cluster is left out if use "days$_{t}o_{d}eath"...groupings <$
$-as.numeric(as.factor(groupings[which(!is.na(outcomesData[, outcomesVarCont]))]))outcomesData < -outcomesData[whi$

if binary is character string categories: make it a factor first, then numeric, otherwise coxph function will throw errors. nonCensoredTerm=1 sampleClustCommPhenoData[which(sampleClustCommPhenoData[ ,outcomesVarBinary]=="deceased"),outcome
<- 1 sampleClustCommPhenoData[which(sampleClustCommPhenoData[ ,outcomesVarBinary]=="living"),outcomesVarBinary]
<- 0 outcomesDataShort <- data.frame(as.numeric(sampleClustCommPhenoData[,outcomesVarBinary]),as.numeric(sampleClustComm
);

sometimes the names are duplicated across studies - remove this line rownames(outcomesDataShort ) <- outcomes-Data[,uniquePatientID]; colnames(outcomesDataShort) <- c(”Censoring”,”TimeToLastContactOrEvent”)

nonCensoredTerm=1 censoredTerm=0 Survival <- outcomesDataShort creating the survival objects with the time and censoring variables OverallSurvival <- Surv(Survival$TimeToLastContactOrEvent, Survival$Censoring==nonCensoredTerm); creating a survival object cutoff at a certain point CutoffPoint <- CutoffPointYears*365; CutoffSamples=Survival$TimeToLastContactC$
$CutoffPoint$!$is.na(Survival$TimeToLastContactOrEvent) SurvivalCutoff=Survival SurvivalCutoff$TimeToLastContactOrEvent[C$
$CutoffPointSurvivalCutoff$Censoring[CutoffSamples]=censoredTerm ”Surv” creates a survival object. really for binary outcomes data. OverallSurvivalCutoff=Surv(SurvivalCutoff$TimeToLastContactOrEvent, SurvivalCutoff$Censoring==nonCe

coxfit=coxph(OverallSurvival groupings, data=Survival) message(”coxfit summary for overall survival. Note that for overall survival, only 3/4 meta-clusters had data:”) summary(coxfit) plot(cox.zph(coxfit)) kmfit=survdiff(OverallSurvival groupings) message(”kaplan meier p-value for overall survival:”) 1 - pchisq(kmfit$chisq, length(kmfit$n) - 1)

message(”calculating the sign of the survival relationship”) mfit=survfit(OverallSurvival groupings) plot(mfit,main=”overall survival (for 3/4 meta-clusters with data”)

coxfit=coxph(OverallSurvivalCutoff groupings, data= SurvivalCutoff) message(”coxfit summary for survival cutoff at ”,CutoffPointYears,”years:”) summary(coxfit) plot(cox.zph(coxfit))

kmfit=survdiff(OverallSurvivalCutoff groupings) message(”kaplan meier p-value for survival cutoff:”) 1 - pchisq(kmfit$chisq, length(kmf$
- 1)

message(”a chi-square test looking at the binary recurrence status variable, as this data was recorded at least in some patients in all 4 meta-clusters:”)

chisq.test(sampleClustCommPhenoData[,”recurrence$_{s}tatus”], groupings)message(”Weseeatrendbyjusttablingtherecurrencestatu$
$50(forthesamplesthatdidn'thaveNAvalues; itlookslikealotofsampleswerestillmissingthisvariablefromcertainmeta-$
$clusters”)table(sampleClustCommPhenoData[,”recurrence_{s}tatus”], groupings)$

@