Code

Vizualization of a Cluster-Filtered Network (CFN) and a Co-Cluster Correlation Network (CCCN)

Introduction and Purpose

Proteins covalently modify one another to govern intracellular mechanisms of signal transduction to affect membrane traffic, cytoskeletal rearrangements, chromatin accessibility, RNA processing, translation, and transcription. To elucidate patterns in post-translational protein modifications (PTMs) that may point to characteristic disease signatures and etiology, we coupled immunoprecipitation using modification-specific antibodies with Tandem Mass Tag (TMT) mass spectrometry to compare 45 lung cancer cell lines to normal lung tissue, and cell lines treated with anti-cancer drugs. These data reveal more than 4000 proteins modified by phosphorylation, methylation, and acetylation. We hypothesized that clustering of protein modifications under different conditions reveals patterns specific to the system, and that PTM clusters that contain proteins known to interact with one another are likely to represent functional signaling pathways.

We define a cluster-filtered network (CFN) to be a protein-protein interaction (PPI) network filtered to include edges only between proteins whose modifications co-clustered. We define a co-cluster correlation network (CCCN) to represent statistical relationships among PTMs such that edges are Spearman corrleation between co-clustered modifications, filtered by a threshold. Details for the methods involved in clustering and construction of these networks may be found in the publication: Palacios-Moreno, J., Foltz, L., Guo, A., Stokes, M. P., Kuehn, E. D., George, L., Comb, M., and Grimes, M. L. Neuroblastoma Tyrosine Kinase Signaling Networks Involve FYN and LYN in Endosomes and Lipid Rafts. PLoS Comp Biol 11, 2015. e1004130–e1004133.

The goal of this vignette is to demonstrate how CFN and CCCN data structures may be visualized in Cytoscape using functions that take advantage of the bioconductor package, RCy3, and its CyREST interface with Cytoscape. In addition, we also hope to demonsrate how networks may be vizualized in browser-accessible format using CX.

Note: This is an R Markdown (http://rmarkdown.rstudio.com) Notebook. When you execute code within the notebook, the results appear beneath the code.

Try executing this chunk by clicking the *Run* button within the chunk or by placing your cursor inside it and pressing *Cmd+Shift+Enter*.

We assume the user knows how to install packages in R using CRAN (https://cran.r-project.org) and bioconductor (https://www.bioconductor.org/install/#install-bioconductor-packages).

Import the data

We developed what we call a wrangling approach to deal with hole-y mass spectrometry data that has a large number of missing values: rigorous clustering using penalized, pairwise-complete statistical relationships combined with a machine learning algorithm (t-SNE). Many of these steps are computationally demanding and beyond the scope of this vignette, so we will simply import the data that represents a small network derived from the larger CFN and CCCN for this example. We will also load packages and a few functions to plot the data and manipulate the visual properties of the graph in Cytoscape.

Hide

```
# load required packages
library(plyr)
library(gplots)
library(igraph)
library(RCy3)
# import pre-computed data frames and functions
load(file="CyChallenge.RData")
load(file="exampleRCyFunctions.RData")
```

A list of data frames follows.

example.cfn - a cluster-filtered network of genes and PPI edges ptm.cccn - PTMs of these genes that co-cluster and have a Spearman correlation greather than the threshold 0.543 gene.atts.criz, gene.atts.ires - gene nodes with descriptive attributes and quantiative attributes based on drug treatment:control ratios for crizotinib and iressa ptm.atts.criz, ptm.atts.ires - PTMs with descriptive attributes and quantiative attributes based on drug treatment:control ratios for crizotinib and iressa

A list of functions follows.

mergeEdges - due to a limitation of graphNEL objects, only one edge between each nodepair is allowed, so this function merges edges of different types between nodes. harmonize_cfs3 - this function merges two node attribute files for genes and PTMs and creates a parent-child relationship which we would like to use for metanodes edgeDprops, nodeDprops.new - these functions set node and edge appearances according to a defined key ratioprops - defines node size and color based on a quantitative node attribute that for these data is the treatment:control ratio of PTMs detected after drug treatments setEdgeWidths.log - uses edge weight to drive edge line width cloneNodePosition - to duplicate node positions from one window onto another window

Cytoscape users need to start Cytoscape and then execute the following.

Hide

```
    cy <-  CytoscapeConnection ()
```

To get started, let's manipulate node and edge data frames to get them in a form for graphing networks in Cytoscape. The PPI edges were derived from Pathway Commons; BioPlex; String; PhosphoSite; and GeneMANIA, with a focus on physical and pathway interactions (not text mining and co-expression). The example network has multiple edges between node pairs, which are derived from different data sets, but the graphNEL object does not allow more than one edge between an ordered node pair (you can make two edges if the order of the pair is reversed). Therefore, we wrote a function to merge edges. (We also create an alternative weight attirbibute, which comes in handy for dealing with negative edge weights and gene-peptide edges.)

Also, the two node attribute files for genes and PTMs (detected as peptides by mass spectrometry) will be merged for plotting genes and PTMs together on the same graph. We create a gene-peptide edge to link genes and PTMs.

Hide

```
    # merge edges due to graphNel limitations
    merged.cfn <- mergeEdges(example.cfn)
    # make Alt.Weight
    merged.cfn$Alt.Weight <- merged.cfn$Weight
    # combine node attribute cytoscape file
    node.atts.criz <- harmonize_cfs3(pepcf=ptm.atts.criz, genecf=gene.atts.criz)
    node.atts.ires <- harmonize_cfs3(pepcf=ptm.atts.ires, genecf=gene.atts.ires)
    # make gene-ptm edges
    net.gpe <- data.frame(Gene.1=ptm.atts.criz$Gene.Name, Gene.2=ptm.atts.criz$Peptide.Name, ed
geType="peptide", Weight=100, Alt.Weight=1)
    # combine edge files
    merged.cfn <- merged.cfn[, c("Gene.1", "Gene.2", "Weight", "edgeType", "Alt.Weight")]
    combined.edges <- rbind(net.gpe, ptm.cccn, merged.cfn)
```

Now let's plot the CFN in Cytoscape.

Hide

```
    # Graph in Cytoscape using RCy3 and CyREST
    # CFN
    cfn.g1 <- cyPlot(gene.atts.criz, merged.cfn)
    cfn.w1 <- CytoscapeWindow(paste("Example CFN", 1+length(getWindowList(cy))), cfn.g1)
    displayGraph(cfn.w1)
    layoutNetwork(cfn.w1, "genemania-force-directed")
    # too crowded
```

Show Results

On Cytoscape, the nodes appear too crowded, so we tweak the layout parameters to spread the graph out so the nodes are all visible.

Hide

```
gm.layout.values <- getLayoutPropertyValue(cy, "genemania-force-directed", getLayoutPropertyName
s(cy, "genemania-force-directed"))
    setLayoutProperties(cy, "genemania-force-directed", list(numIterations=100, defaultSpringCo
efficient=0.1, defaultSpringLength=50, minNodeMass=0.001, maxNodeMass=2000, midpointEdges=250, c
urveSteepness=7.0e-03, isDeterministic=1, singlePartition=0, ignoreHiddenElements=1))
    layoutNetwork(cfn.w1, layout.name= "genemania-force-directed")
    # better
```

Show Results

I usually manually adjust node positions for clarity in Cytoscape, which is easy to do with the Cytoscape GUI. Try it!

Now let's make the nodes distinctive shapes and the edges colored depending on their attributes. (A node and edge key may be plotted at the end of this vinette.)

Hide

```
edgeDprops(cfn.w1)
ratioprops(cfn.w1, gene.atts.criz, "Total")
showGraphicsDetails(cfn.w1, TRUE)
nodeDprops.new(cfn.w1, gene.atts.criz)
setEdgeWidths.log(cfn.w1, factor=1.2)
```

Show Results

This graph shows the way all PTMs were affected by the ALK and MET inhibitor, crizotinib, by adjusting node visual properties. In Cytoscape, the node size indicates the magnitude of change and node fill color indicates that blue is negative (e.g., MET phosphorylation decreased the most in the presence of crizotinib). Examination of the data tables in Cytoscape reveals some aspects of gene function and data from the mass spectrometry experiments.

Next, we would like to know what happens to the PTMs of these proteins when cells were treated with the EGFR inhibitor, iressa. We employ the data frame that shows iressa data for this purpose.

Hide

```
# Plot the same network with iressa values
cfn.g <- cyPlot(gene.atts.ires, merged.cfn)
cfn.w2 <- CytoscapeWindow(paste("Example CFN", 1+length(getWindowList(cy))), cfn.g)
displayGraph(cfn.w2)
layoutNetwork(cfn.w2, layout.name= "genemania-force-directed")
#
edgeDprops(cfn.w2)
nodeDprops.new(cfn.w2, gene.atts.criz)
ratioprops(cfn.w2, gene.atts.criz, "Total")
showGraphicsDetails(cfn.w2, TRUE)
setEdgeWidths.log(cfn.w2, factor=1.2)
```

Show Results

Note that the nodes are now different sizes and colors; EGFR is now big and blue, indicating that its PTMs (phosphorylations) were most dramatically inhbited by iressa, as expected.

It is useful to compare these two networks side by side with the nodes in the same position. If you are happy with the layout and node position of the first network, you can copy it onto the second.

Hide

```
# make the node position identical
 cloneNodePosition(cfn.w1, cfn.w2)
```

Show Results

What about the PTMs themselves? We could plot the PTM CCCN by itself, or, in the next example, plot the combined PPI and CCCN. The gene-peptide edges are given a large weight so that force-directed or spring-embedded layouts pull peptides close to their genes. (Alternatively, there is an aditional addtribute in the nodes that indicates parent-child relationships. The ability to create metanodes is not yet accessbile by CyREST, however.)

Hide

```
    # Now show the PTM CCCN
    cccn.g <- cyPlot(node.atts.criz, combined.edges)
    cccn.w1 <- CytoscapeWindow(paste("Example CCCN", 1+length(getWindowList(cy))), cccn.g)
    displayGraph(cccn.w1)
    layoutNetwork(cccn.w1, layout.name= "genemania-force-directed")
    #
    nodeDprops.new(cccn.w1, node.atts.criz)
    edgeDprops(cccn.w1)
    ratioprops(cccn.w1, node.atts.criz, "Total")
    showGraphicsDetails(cccn.w1, TRUE)
    setEdgeWidths.log(cccn.w1, factor=1.2)
```

Show Results

We have defined node shape and border color to represent different classes of proteins according to the key that is plotted in the next code chunk.

Hide

```
# Set up node key graph
    molclasses <- c("unknown", "receptor tyrosine kinase",  "SH2 protein", "SH3 protein", "tyros
ine kinase",  "SRC-family kinase",   "kinase", "phosphatase", "transcription factor", "RNA bindi
ng protein")
    nodeTypes <- c(molclasses,"deacetylase","acetyltransferase","demethylase","methyltransferas
e","membrane protein", "receptor tyrosine kinase", "G protein-coupled receptor")
    nulledges <- data.frame(Gene.1="unknown", Gene.2="unknown", edgeType="unknown", Weight=0)
    nodetype.cf <- data.frame(Gene.Name=nodeTypes, nodeType=nodeTypes, size=130)
    nodetype.cf$Domains <- c("undefined", "TM", "SH2", "SH3", "tyrosine kinase", "tyrosine kinas
e", "kinase", "phosphatase", "zinc finger", "RNA binding", "undefined",
"undefined","undefined","undefined", "TM", "TM", "TM")
    node.g <- cyPlot(nodetype.cf, nulledges)
    node.w <- CytoscapeWindow("Node Key", node.g)
    displayGraph(node.w)
        #
    layoutNetwork(node.w, "grid")
     remove.autophos.cy(node.w)
     nodeDprops.new(node.w, nodetype.cf)
# Note that the scale is compressed. The ability to control the tool panel would be useful!
```

Show Results

Note: There does not appear to be a way to adjust layout paramters for the “grid” layout, but in Cytoscape “View -> Show Tool Panel” provides a way to expand the grid.

The edge key is displayed using the following code. Edge weights can drive line widths in different ways.

Hide

```
edgeTypes <- c("pp", "controls-phosphorylation-of", "controls-expression-of", "controls-transpor
t-of",  "controls-state-change-of", "Physical interactions", "BioPlex", "in-complex-with",  'exp
eriments',  'database',   "Pathway", "Predicted", "Genetic interactions", "correlation", "negati
ve correlation", "positive correlation",  'combined_score', "merged" , "intersect", "peptide",
'homology', "Shared protein domains")

edgecolors <- col2hex(c("red", "red", "magenta", "violet", "purple",  "green", "green2", "green
3",  "aquamarine2", "cyan", "turquoise2", "cyan2", "lightseagreen", "gold",  "blue", "yellow",
"slategrey", "darkslategrey", "grey", "black", "orange", "orange2"))

myarrows <- c ('Arrow', 'Arrow', 'Arrow', 'Arrow', "Arrow", 'No Arrow', 'No Arrow', 'No Arrow',
'No Arrow', 'No Arrow', 'No Arrow', 'No Arrow', 'No Arrow', 'No Arrow', 'No Arrow', 'No Arrow',
'No Arrow', 'No Arrow', 'No Arrow', 'No Arrow', 'No Arrow', 'No Arrow')

edge_key.tbl <- data.frame(edgeTypes, colors=c("red", "red", "magenta", "violet", "purple",  "gr
een", "green2", "green3",  "aquamarine2", "cyan", "turquoise2", "cyan2", "lightseagreen",
"gold",  "blue", "yellow", "slategrey", "darkslategrey", "grey", "black", "orange", "orange2"),
myarrows, Weight=seq(-100, 110, 10))

edgefile <- data.frame(Gene.1=LETTERS[1:23], Gene.2=paste(LETTERS[1:23], 1, sep=""),
edgeType=c("unknown", edgeTypes), Weight=c(exp(seq(-20, 5, 2)), 1:6, 10:13))
edgefile[edgefile$edgeType=="negative correlation", "Weight"] <- -1.0
    negs <- edgefile[edgefile $Weight<0,'Weight']
    frnegs <- exp (negs*20)
    # plot (negs,frnegs)
    edgefile $Alt.Weight <- edgefile $Weight
    edgefile[edgefile $Weight<0,'Alt.Weight'] <- frnegs

nodefile <- data.frame(Gene.Name=c(as.character(edgefile$Gene.1),
as.character(edgefile$Gene.2)), nodeType="unknown")

edgetest.g <- cyPlot(nodefile, edgefile)
    edge.w <- CytoscapeWindow("Edge Key", edgetest.g)
    displayGraph(edge.w)
    layoutNetwork(edge.w, "hierarchical")
     edgeDprops(edge.w)
     setDefaultBackgroundColor (edge.w, "#949494") # grey 58
     setEdgeLabelRule (edge.w, 'edgeType')
        line.widths <- 4.5*abs(as.numeric(edgefile$Weight))
        setEdgeLineWidthRule(edge.w, "Weight", attribute.values=as.character(edgefile$Weight), l
ine.widths, default.width=1.2)
```

Show Results

That’s it! See if you can get the browser to do these things with CX!