RIFA: A Differential Gene Connectivity Algorithm

Todd D. Allen

This article describes the implementation of RIFA, a computational biology algorithm designed to identify the genes most directly responsible for creating differences in phenotype between two biological states, for example, tumorous liver tissue versus cirrhotic liver tissue.

Introduction and Background

With the invention of microarray technology, scientists finally had a means to measure global changes in gene expression between two biological states [1]. This has led to thousands of scientific publications describing long lists of differentially expressed genes in each scientist’s favorite experimental system. What has gradually become apparent to biologists is that having a list of differentially expressed genes, while an important first step in understanding the differences between two phenotypes (where phenotype represents the physical manifestation of one or more traits), is often not enough to identify the genes most directly responsible for driving changes in phenotype. While it is true that genes that are differentially expressed between two biological states may be important in explaining those differences, it is also possible that genes whose expression is not changed can also be pivotal in driving phenotypic differences.

For those unfamiliar with biology, a rough analogy may prove useful. Consider a manufacturing setting where there is a supervisor (a “boss” gene) and employees (“slave” genes) under the supervisor’s responsibility charged with manufacturing widgets (a particular phenotype, such as healthy liver tissue). A change in phenotype, such as transitioning from healthy liver tissue (manufacturing blue widgets) to cancerous liver tissue (manufacturing red widgets) can be accomplished by: (1) changing the rate that employees work (such as might happen if a supervisor shouts at the employees; this is analogous to differential expression); and/or (2) changing the instructions the supervisor is giving to employees (keeping the volume of instructions constant, but changing the information contained in the instructions; this is analogous to a mutation in the “boss” gene); and/or (3) a combination of scenarios (1) and (2). In scenario (1), there is a transition in phenotype because the employees (“slave” genes) begin working faster or slower than they have previously, which produces too many or too few gene products at the wrong
time, creating a rippling effect throughout all of the manufacturing, which ends up in a
different product (the red widget phenotype) being made. In this situation, the super-
visors instructions to the employees remain constant (manufacture blue widgets) but are
spoken with more (“shouting”) or less (“whispering”) volume. Scenario (1) reflects the
kind of information that can be measured in microarray studies, whose sole purpose is to
identify genes whose expression changes between two biological states. In scenario (2),
the rate at which employees work remains constant, but they still manufacture a different-
colored widget (phenotype), because the instructions they are receiving from their
supervisor have changed. Differences in phenotype due to scenario (2) are rarely
discovered by producing long lists of differentially expressed genes, because the primary
driving force creating a change in phenotype is a change in instruction from the supervisor
(such as a mutation in the “boss” gene) to the employees (“slave” genes), not a difference
in the manufacturing rate of employees.

For these reasons, computational biologists have begun to develop algorithms that are bet-
ter at highlighting those genes primarily responsible for driving changes in phenotype, re-
gardless of whether those genes are differentially expressed or not. This is the purpose of
the regulatory impact factor analysis (RIFA) algorithm presented here; that is, to highlight
those genes most directly responsible for driving changes in phenotype. RIFA provides a
computationally tractable way to detect differences in connectivity between genes in two
biological states. Figure 1 illustrates the basic premise of connectivity and differences in
connectivity between two biological states.

Figure 1. Two gene networks comprised of the same five genes (a through e) in two different
biological states (phenotypes). Each vertex represents a gene, and each edge represents a
connection between genes. In standard differential expression studies, each gene’s expression
level is compared to itself between the two biological states but ignores potential relationships
between different genes. When even a casual observer compares the two networks above, it is
immediately noticeable that the shape of each network is different, a difference driven by a
change in connectedness between genes within each biological state.
Regulatory impact factor analysis (RIFA) is based on seminal work by Hudson, Reverter, and Dalrymple [2], which introduced three higher-order metrics all computed from basic information obtained in differential gene expression studies. The purpose of these metrics is to use the information present in gene expression studies to quantify the connectedness between differentially expressed genes (“slave” genes, using the analogy above) and a group of gene expression regulator genes, known as transcription factors (“boss” genes, using the analogy above). The three metrics are:

Phenotype Impact Factor (PIF) = \( \frac{1}{2} (E_{i,A} + E_{i,B}) dE_i \)  

Regulatory Impact Factor 4 (RIF4) = \( \frac{1}{n_{dE}} \sum_{j=1}^{n_{dE}} (PIF_j dC_{i,j}^2) \)  

Regulatory Impact Factor 5 (RIF5) = \( \frac{1}{n_{dE}} \sum_{j=1}^{n_{dE}} \left( (E_{j,A} r_A(i,j)) - (E_{j,B} r_B(i,j)) \right)^2 \)  

Equation (1) (PIF) computes the average expression of the \( i^{th} \) gene between two biological states (A and B) and multiplies that result by the differential expression of the \( i^{th} \) gene between states A and B. In doing so, the magnitude of the differential expression of a gene is weighted by the overall expression level of the gene. PIF is then used to compute equation (2) (RIF4), which multiplies the PIF value for each differentially expressed gene by the differential co-expression (calculated using the Spearman correlation coefficient) between each differentially expressed gene (the “slave” genes in our analogy above) and each transcriptional regulator (the “boss” genes in our analogy above) between states A and B. By summing these calculations over each differentially expressed gene, a prioritized list of the most important regulators driving changes in phenotype between states A and B can be obtained. Equation (2) is designed to provide an answer to the question, which regulator is consistently highly differentially co-expressed with the most abundant differentially expressed genes? Equation (3) (RIF5) is an alternative metric to equation (2) (RIF4), which also seeks to produce a prioritized list of the most important regulators driving phenotypic change. By multiplying the expression of each differentially expressed gene by the correlation between itself and each transcription regulator twice, once in state A and once in state B, the difference in state values can be computed and then summed over each differentially expressed gene to yield an alternative prioritized list of the most important regulators. Equation (3) is designed to answer the question, which regulator has the most altered ability to predict the abundance of differentially expressed genes? Further details of these equations are presented in [3-4], but the basic idea behind the use of these metrics in RIFA is straightforward. When gene expression data (from a well-thought-out experiment) is presented to RIFA, the algorithm can use the “echoes of sound off structures” (differential gene expression data) to triangulate the location of the “rifle shot creating the sound” (identify the master gene(s) driving the changes in phenotype).
The Regulatory Impact Factor Analysis (RIFA) Algorithm

RIFA is template driven, meaning the algorithm expects several pieces of user-defined information to be provided in a notebook cell that is used as a template for entering information. As RIFA was designed to process output from AffyDGED [5], it will be assumed the reader is familiar with AffyDGED as well. The features of RIFA are illustrated using data from a microarray study comparing gene expression profiles of tumorous liver tissue to cirrhotic liver tissue [6]. All microarray data used in this study and presented here is publicly available at NCBI’s Gene Expression Omnibus portal (www.ncbi.nlm.nih.gov/geo), using the access number GSE17548.

```mathematica
Needs["JLink"]
ReinstallJava[JVMArguments -> "-Xmx512m"];

timecoursedata = 
  Flatten[
    Import[
      "C:\\Users\\Wookie\\Desktop\\Mathematica
      Projects\\Mathematica Journal
      Projects\\Differential Wiring & RIF
      development\\Rif validation - Yildiz -
      liver\\liver all expression data for rifa.xls"],
    1];

conditiononecol = {2, 7};
conditiontwoocol = {8, 13};

rawtranscriptionreg =
  Flatten[
    Import[
      "C:\\Users\\Wookie\\Desktop\\Mathematica
      Projects\\Mathematica Journal
      Projects\\Differential Wiring & RIF
      development\\Rif validation - Yildiz -
      liver\\hgplus2 trfactors.xls"]];

rawdegenes =
  Flatten[
    Import[
      "C:\\Users\\Wookie\\Desktop\\Mathematica
      Projects\\Mathematica Journal
      Projects\\Differential Wiring & RIF
      development\\Rif validation - Yildiz -
      liver\\liver de genes for rifa.xls"]];
```
affyginlocation = Import[
"C:\Users\Wookie\Desktop\Mathematica
Projects\Mathematica Journal
Projects\Data\AffyChip Description
Files\HG-U133_Plus_2\LibFiles\HG-U133_Plus_2.gin"];

tsavelocationroot = "C:\Users\Wookie\Desktop\";

studynamer = livercirrhcancer;

The template cell begins with a command to purposefully reinstall Java, for the express purpose of expanding the memory available to import large datasets into Mathematica.

The remainder of the template cell defines several variables requiring user input.

1. timecoursedata: This variable points to the directory containing the microarray gene expression data, in spreadsheet format, to be processed by RIFA. While this variable uses the term "timecourse" as part of its name, it is not necessary for the microarray data to be part of a time course experiment. The spreadsheet format of the data is non-negotiable and requires strict organization. To aid in instruction, a screen shot of the timecoursedata that will be described throughout this paper is included here (Figure 2).

![Figure 2. Formatting of gene expression data for variable <timecoursedata>.

Column A contains unique transcript identification information from the microarray chip used in the study. Columns B through X contain gene expression measurement from samples (or time points) under the same experimental condition of the study. The columns after X contain gene expression measurements from samples (or time points) under the same control condition of the study. For example, in the liver study referenced above, tumor samples from multiple patients were randomly placed into six groups and compared to six groups of cirrhotic liver tissue by AffyDGED. Column B contains the gene expression measurements (transcript abundance, not differential expression) for the first group of tumor samples processed with AffyDGED, column C contains the gene...
expression measurements from the second group of tumor samples, and so on. In this example, the last column containing tumor (experimental condition) data is column G. Column H is the first column containing gene expression measurements from the first group of cirrhotic (control condition) tissue, column I from the second group of cirrhotic tissue, and so on. Notice how columns B and H contain output from AffyDGED’s processing of the first groups of tumor and cirrhotic tissues, respectively.

2. conditiononecol: This contains a short list defining the first and last column positions within timecoursedata to contain experimental condition data.

3. conditiontwocol: This contains a short list defining the first and last column positions within timecoursedata to contain control condition data.

4. rawtranscriptionreg: This variable points to the location of a spreadsheet file containing a list (organized into a single column) of known or suspected transcription factor genes present on the microarray chip being used. The file used here was created by parsing the biological process column of the annotation file for Affymetrix’s Human Genome U133 Plus 2.0 chip (available at www.affymetrix.com) for genes linked to the transcription process. The probeset identifiers referring to this group of genes were used to build a list of transcription factor genes.

5. rawdegenes: This link points to the spreadsheet file containing lists of differentially expressed genes (referenced by their probeset IDs, organized into columns) created by processing the experimental and control groups referenced in timecoursedata (above) with AffyDGED. In the liver example here, there are six columns of differentially expressed genes created by using AffyDGED to compare the six groups of tumorous livers with the six groups of cirrhotic livers.

6. affyginlocation: This variable holds the directory location for finding the Affymetrix .gin (gene information) file that provides the necessary information to annotate output from RIFA.

7. savelocationroot: This variable holds the location where the user would like the final results of the analysis to be saved.

8. studyname: This variable allows the user to name the output files generated by RIFA with study-specific information.

The first tasks completed by RIFA include the loading, parsing, and organization of raw data to facilitate downstream computation.

\[
\text{starttime} = \text{AbsoluteTime}[];
\]
\[
\text{allmicroarraytranscripts} = \text{timecoursedata}[[\text{All, 1}]];
\]
\[
\text{conditiononetimepts} = \text{timecoursedata}[[\text{All, conditiononecol}[1]]];, \text{conditiononecol}[2]]];
\]
\[
\text{conditiontwotimepts} = \text{timecoursedata}[[\text{All, conditiontwocol}[1]]];, \text{conditiontwocol}[2]]];
\]
empties = Flatten[Map[Position[rawtranscriptionreg, #] &, Complement[rawtranscriptionreg, allmicroarraytranscripts]], {1, 2}];

transcriptionreg = Delete[rawtranscriptionreg, empties];

tregpositions = Flatten[Map[Position[allmicroarraytranscripts, #1] &, transcriptionreg]]; 

tregconditiononetimepts = conditiononetimepts[[tregpositions]]; 
tregconditiontwotimepts = conditiontwotimepts[[tregpositions]]; 

delist = Drop[Union[rawdegenes], 1]; 

delistpositions = Flatten[Map[Position[allmicroarraytranscripts, #1] &, delist]]; 

delistconditiononetimepts = conditiononetimepts[[delistpositions]]; 
delistconditiontwotimepts = conditiontwotimepts[[delistpositions]]; 

condition1liststocorrelate = Flatten[Outer[List, tregconditiononetimepts, delistconditiononetimepts, 1], {1, 2}]; 

condition2liststocorrelate = Flatten[Outer[List, tregconditiontwotimepts, delistconditiontwotimepts, 1], {1, 2}]; 

Upon completion of this first section of code, the transcription factor genes (the “boss” genes from the analogy above) are grouped with the differentially expressed genes (the “slave” genes from above) to facilitate calculation of each pairings’ Spearman rank correlation coefficient.

RIFA proceeds by calculating the Spearman rank correlation coefficients, which requires that each vector of gene expression measurements be tested for the presence of duplicate entries, which requires special handling to calculate Spearman rho. This is the purpose of the tieCheck module below. Based on the results of tieCheck, the code calls the spearmanControl module to optimize calculation of Spearman rho, taking advantage of function listability and the use of compilable expressions, where appropriate, to maximize speed.
tieCheck[origdata_] := Module[{tiecheck1, tiecheck2, tiepos, tiecheckresult, 
datawithnomultiples, tiedata, notiepos},

tiecheck1 = Length[origdata[[1, 1]]];
tiecheck2 = Map[Length[Union[#]] & , origdata, {2}];

tiepos = Position[tiecheck2, {x_, y_} /; (x < tiecheck1) ∨ (y < tiecheck1), {1}];
notiepos = Position[tiecheck2, {x_, y_} /; (x == tiecheck1 & y == tiecheck1), {1}];

If[Length[Flatten[tiepos]] == 0,
(tiecheckresult = False),
(datawithnomultiples = Delete[origdata, tiepos];
tiedata = Extract[origdata, tiepos];
tiecheckresult = {datawithnomultiples, tiedata, 
tiepos, notiepos})]

spearmanControl[tieresult_, condliststocorrelate_] := Module[{sprho, tierho, tierhotopos, notierhotopos, 
joinrho, rhosorted, finalrho},

If[tieresult === False,
srho = fastSpearNoTie[condliststocorrelate[[All, 1]],
condliststocorrelate[[All, 2]]],

sprho = fastSpearNoTie[tieresult[[1]][[All, 1]],
tieresult[[1]][[All, 2]]];
tierho = Table[SpearmanRho[Apply[Sequence, tieresult[[2, i]]]],
{i, 1, Length[tieresult[[2]]]]];

tierhotopos = Partition[
Flatten[MapThread[List, {tierho, tieresult[[3]]}]],
2];
notierhotopos = Partition[
Flatten[MapThread[List, {sprho, tieresult[[4]]}]], 2];
joinrho = Join[tierhotopos, notierhotopos];
rhosorted = Sort[joinrho, #1[[2]] < #2[[2]] &];
finalrho = rhosorted[[All, 1]]]

fastSpearNoTie[vector1_, vector2_] := Module[{vector1sort, vector2sort, rankvec1, rankvec2},

The Mathematica Journal 17 © 2015 Wolfram Media, Inc.
vector1sort = fastSort[vector1];
vector2sort = fastSort[vector2];

rankvec1 = fastRank[vector1sort, vector1];
rankvec2 = fastRank[vector2sort, vector2];

listableSpear[rankvec1, rankvec2]

fastSort = Compile[{{vector, _Real, 2}},
    Module[{vectorsort},
    vectorsort = Map[Sort, vector],
    CompilationTarget -> "WVM", Parallelization -> True];

fastRank = Compile[{{vectorsort, _Real, 2}, {vector, _Real, 2}},
    Module[{rankvec},
    rankvec = Map[Flatten,
        Table[Position[vectorsort[[i]], vector[[i, j]]],
        {i, 1, Length[vectorsort]},
        {j, 1, Length[vectorsort[[1]]]}], {1}],
    CompilationTarget -> "WVM", Parallelization -> True];

listableSpear = Compile[{{rankvec1, _Integer, 2},
    rankvec2, _Integer, 2}},
    Module[{bottom, final},
    top = 6 (* (Total[(rankvec1 - rankvec2)^2, {2}])];
    bottom = Length[rankvec1[[1]]] *
        (Length[rankvec1[[1]]]^2 - 1);
    final = 1 - (top / bottom) // N],
    {{top, _Integer, 1}}, CompilationTarget -> "WVM",
    Parallelization -> True, RuntimeAttributes -> {Listable]};
**Caution:** Due to the sheer volume of computation that needs to be completed using the data described in this paper, the next segment of code will likely take 20–40 minutes to complete (depending on the speed of your computer) and consume roughly 28 Gb of RAM. Computations on machines with less RAM will finish but will require significant use of the hard drive, slowing computation considerably.

\[
\text{tieresultscond1} = \text{tieCheck[condition1liststocorrelate]};
\text{tieresultscond2} = \text{tieCheck[condition2liststocorrelate]};
\text{condition1corr} = \text{spearmanControl[tieresultscond1, condition1liststocorrelate]};
\text{Clear[tieresultscond1, condition1liststocorrelate]};
\text{condition2corr} = \text{spearmanControl[tieresultscond2, condition2liststocorrelate]};
\text{Clear[tieresultscond2, condition2liststocorrelate]};
\]

By pairing (and taking a sampling of) the correlations of the “boss” and “slave” genes between the tumor and cirrhotic livers, a satisfying (but not biologically surprising) pattern reveals itself (Figure 3). It is clear that there are many more strongly positive correlations between the “boss” and “slave” genes in the cirrhotic liver than the tumorous liver. This supports what biologists have known for a long time; that is, gene regulation in tumorous tissue is significantly uncoordinated.

\[
\text{temp1} = \text{MapThread[List, \{condition1corr, condition2corr\}]};
\text{temp2} = \text{RandomSample[temp1, 100000]};
\]

\[
\text{Histogram}[\text{temp2[[All, 1]], temp2[[All, 2]]},
\text{ChartLegends} \rightarrow \text{Placed}[\{"Tumorous liver", "Cirrhotic liver"},
\text{Center}], \text{PerformanceGoal} \rightarrow \text{"Speed"}, \text{Frame} \rightarrow \text{True},
\text{FrameLabel} \rightarrow \{\{"SpearmanRho Count", "\"},
\{"SpearmanRho Bins", "\"\}]
\]

\[\text{Figure 3. A histogram of the Spearman rank correlation coefficients between regulator and differentially expressed genes in tumorous and cirrhotic liver biopsies.}\]
Following the correlation calculations, RIFA calculates several important metrics, including PIF, RIF4, and RIF5 of equations (1), (2), and (3) described above.

\[
\text{diffwiring} = \text{condition1corr} - \text{condition2corr};
\]

\[
\text{diffexp} = \text{Map}[\text{Mean}, \text{delistconditiononetimepts}] - \text{Map}[\text{Mean}, \text{delistconditiontwotimepts}];
\]

\[
\text{timeptscombinedmean} = \text{Map}[\text{Mean}, \text{Map}[\text{Flatten}[\#] &,
\text{MapThread}[\text{List}, \{\text{delistconditiononetimepts},
\text{delistconditiontwotimepts}\}]])];
\]

\[
\text{pif} = \text{timeptscombinedmean} \times \text{diffexp};
\]

\[
\text{pifcopyforrif} = \text{Flatten}[(\text{Table}[\text{pif}, \{\text{Length}[\text{transcriptionreg}]\})]];
\]

\[
\text{rif4} = (\text{Map}[\text{Total}, \text{Partition}[\text{pifcopyforrif} \times (\text{diffwiring}^2),
\text{Length}[\text{delist}]])] / \text{Length}[\text{diffexp}];
\]

\[
\text{rif4standard} = \text{Standardize}[\text{rif4}];
\]

\[
\text{conditiononemeans} = \text{Map}[\text{Mean}, \text{delistconditiononetimepts}];
\]

\[
\text{conditiontwomeans} = \text{Map}[\text{Mean}, \text{delistconditiontwotimepts}];
\]

\[
\text{cond1meancopyforrif5} = \text{Flatten}[(\text{Table}[\text{conditiononemeans},
\{\text{Length}[\text{transcriptionreg}]\})]);
\]

\[
\text{cond2meancopyforrif5} = \text{Flatten}[(\text{Table}[\text{conditiontwomeans},
\{\text{Length}[\text{transcriptionreg}]\})]);
\]

\[
\text{rif5} = (\text{Map}[\text{Total},
\text{Partition}[(\text{(cond1meancopyforrif5} \times \text{condition1corr})^2) -
(\text{(cond2meancopyforrif5} \times \text{condition2corr})^2),
\text{Length}[\text{delist}]])] / \text{Length}[\text{diffexp}];
\]

\[
\text{rif5standard} = \text{Standardize}[\text{rif5}];
\]

\[
\text{avgrif} = \text{Map}[\text{Mean}, \text{MapThread}[\text{List}, \{\text{rif4standard}, \text{rif5standard}\}]];\]

\[
\text{avgresult} = \text{MapThread}[\text{List}, \{\text{transcriptionreg}, \text{avgrif}\}];
\]

\[
\text{rif4result} = \text{MapThread}[\text{List},
\{\text{transcriptionreg}, \text{rif4standard}\}];
\]

\[
\text{rif5result} = \text{MapThread}[\text{List},
\{\text{transcriptionreg}, \text{rif5standard}\}];
\]
The resulting plots (Figure 4) of PIF (equation (1)), RIF4 (equation (2)), and RIF5 (equation (3)) reveal the bidirectional nature of each of the three metrics. In other words, regardless of the metric used, –8 and +8 are equally influential to the underlying biology. This makes sense when one remembers that gene expression measurements, used in the calculations of the metrics above, are represented on a log2 scale.

```
temp3 = Histogram[pif, ChartStyle -> Orange, Frame -> True, FrameLabel -> {"PIF Count", ""}, {"PIF Bins", ""}], PerformanceGoal -> "Speed"];
temp4 = Histogram[rif4standard, ChartStyle -> Green, Frame -> True, FrameLabel -> {"RIF4 Count", ""}, {"RIF4 Bins", ""}], PerformanceGoal -> "Speed"];
temp5 = Histogram[rif5standard, ChartStyle -> Red, Frame -> True, FrameLabel -> {"RIF5 Count", ""}, {"RIF5 Bins", ""}], PerformanceGoal -> "Speed"];
GraphicsColumn[{temp3, temp4, temp5}]
```

▲ Figure 4. Histograms for each of the three primary metrics used in the RIFA algorithm. Positive and negative values should be interpreted as equally important (i.e., a gene that is fourfold down in expression is equally as likely to be important as a gene that is fourfold up in expression.)
After the metric calculations are completed, four files are exported containing all the results in file formats directly usable by *Mathematica* and Microsoft Excel. One set of files is appended with the phrase “RifSortByAvg” and contains the following information in table form, sorted by the average of RIF4 and RIF5 values.

- **Column 1**: unique transcript (gene) IDs
- **Column 2**: the average of RIF4 and RIF5 values
- **Column 3**: RIF4 values
- **Column 4**: RIF5 values
- **Column 5**: genbank accession numbers
- **Column 6**: gene names
- **Column 7**: gene product information

A second set of files is appended with the phrase “sortedPIF” and contains the following information in table form, sorted by PIF values.

- **Column 1**: unique transcript (gene) IDs
- **Column 2**: PIF values
- **Column 3**: genbank accession numbers
- **Column 4**: gene names
- **Column 5**: gene product information

As described above, the RIF4 and RIF5 results are most useful for identifying the “boss” genes and the PIF results are most useful for identifying the “slave” genes. Both the “boss” and “slave” genes can be influential in creating differences in phenotypes between two states.

```mathematica
resultginpositions = Flatten[Map[Position[affyginlocation[[All, 4]], #] &, avgresult[[All, 1]]]];

ginannotationdata = affyginlocation[[resultginpositions]][[All, 8 ;; 10]];

combinationresult = MapThread[List, {avgresult[[All, 1]], avgresult[[All, 2]], rif4result[[All, 2]], rif5result[[All, 2]], ginannotationdata}];

combinationresult = Table[Flatten[combinationresult[[i]]], {i, 1, Length[combinationresult]}];

combosortbyavg = Sort[combinationresult, #1[[2]] > #2[[2]] &];
combosortbyrif4 = Sort[combinationresult, #1[[3]] > #2[[3]] &];
```
combosortbyrif5 = Sort[combinationresult, 
   #1[[4]] > #2[[4]] &];

pifid = MapThread[List, {delist, pif}];

pifgipositions = 
   Flatten[Map[Position[affyginlocation[[All, 4]], #] &, pifid[[All, 1]]]]; 

pifannotationdata = affyginlocation[[pifgipositions]][[All, 8 ;; 10]]; 

pifresults = 
   Sort[MapThread[List, {delist, pif, pifannotationdata}], 
   #1[[2]] > #2[[2]] &]; 

pifresults = Table[Flatten[pifresults[[i]]], 
   {i, 1, Length[pifresults]}]; 

date = DateString[]; 

foldername = StringJoin[ToString[studyname], " - ", date]; 

SetDirectory[savelocationroot]; 
savelocationfinal = CreateDirectory[foldername]; 
SetDirectory[savelocationfinal]; 

Put[combosortbyavg, StringJoin[ToString[studyname], 
   " - RifSortbyAvg"]]; 

Export[StringJoin[ToString[studyname], 
   " - RifSortbyAvg.csv"], combosortbyavg]; 

Put[pifresults, StringJoin[ToString[studyname], 
   " - sortedPIF"]]; 

Export[StringJoin[ToString[studyname], " - sortedPIF.csv"], 
   pifresults]; 

Print[ 
   Style["RIFA calculations complete. All data saved to: ", 
   Bold], Style[Tostring[savelocationroot], Bold]]; 

endtime = AbsoluteTime[] - starttime; 
Print["Computational Time: ", endtime, " seconds"];

The Mathematica Journal 17 © 2015 Wolfram Media, Inc.
The final output of RIFA is a network graph that associates the most strongly correlated, highest impact PIF scores with the highest impact RIF scores. In this graph, the top 10 most positive and negative average RIF entries, the top 10 most positive and negative RIF4 entries, and the top 10 most positive and negative RIF5 entries are linked to the phenotype of interest with red edges. In other words, the red edges highlight the “boss” genes most responsible for driving changes in phenotype. The nodes of the graph use tooltips to identify the gene represented by the node. The phenotype of interest node is abbreviated “POI.” Blue edges are used to highlight the “slave” genes most responsible and most correlated to the “boss” genes for driving changes in phenotype. In this way, the graph highlights the “slave” genes responding to the “boss” genes’ orders to change phenotype. Here, the highest 0.5% of positive and the lowest 0.5% of negative PIF scores are connected to transcription regulators (represented by RIF nodes) if they share a Spearman rho value of ±0.9 with the transcription regulator.

```
RIFA calculations complete. All data saved to:
C:\Users\Wookie\Desktop\ 

Computational Time: 1541.1711362 seconds
```

```
graphRIFA :=
Module[{graphregdata, pifsize, pifforgrph,
       delistposofpifids, topregpos, cond1corrpart,
       cond2corrpart, pifbyregcond1, pifbyregcond2,
       pifcorrstringency1, pifcorrstringency2,
       pifgrphidsonly, regtopifposition1, regtopifposition2,
       cond1regtopif, cond2regtopif, regpifgrphdata,
       finalgrphdata, grph1},

graphregdata =
Union[Join[combosortbyavg[[1 ;; 10]],
        combosortbyavg[[-10 ;; -1]],
        combosortbyrif4[[1 ;; 10]],
        combosortbyrif4[[-10 ;; -1]],
        combosortbyrif5[[1 ;; 10]],
        combosortbyrif5[[-10 ;; -1]]][[All, 1 ;; 2]]];

grph1 = Thread[graphregdata[[All, 1]] -> "POI"];

pifsize = Ceiling[(Length[pif] * 0.01) / 2];

pifforgrph = Join[pifresults[[1 ;; pifsize]],
       pifresults[[-pifsize ;; -1]]];

delistposofpifids =
Flatten[Map[Position[delist, #] &,
       pifforgrph[[All, 1]]]];
topregpos = Flatten[Map[Position[transcriptionreg, #] &, graphregdata[[All, 1]]]]; 
cond1corrpart = Partition[condition1corr, Length[delist]][[topregpos]]; 
cond2corrpart = Partition[condition2corr, Length[delist]][[topregpos]]; 
pifbyregcond1 = Table[cond1corrpart[[i]][[delistposofpifids]], {i, 1, Length[cond1corrpart]}]; 
pifbyregcond2 = Table[cond2corrpart[[i]][[delistposofpifids]], {i, 1, Length[cond2corrpart]}]; 
pifcorrstringency1 = Table[Flatten[Position[pifbyregcond1[[i]], x_ /; (x <= -0.90 || x >= 0.90)]], {i, 1, Length[pifbyregcond1]}]; 
pifcorrstringency2 = Table[Flatten[Position[pifbyregcond2[[i]], x_ /; (x <= -0.90 || x >= 0.90)]], {i, 1, Length[pifbyregcond2]}]; 
pifforgrphidsonly = pifforgrph[[All, 1]]; 
regtopifposition1 = Table[pifforgrphidsonly[[pifcorrstringency1[[i]]]], {i, 1, Length[pifcorrstringency1]}]; 
regtopifposition2 = Table[pifforgrphidsonly[[pifcorrstringency2[[i]]]], {i, 1, Length[pifcorrstringency2]}]; 
cond1regtopif = Table[Thread[graphregdata[[All, 1]][[i]] -> regtopifposition1[[i]]], {i, 1, Length[graphregdata[[All, 1]]]}]; 
cond2regtopif = Table[Thread[graphregdata[[All, 1]][[i]] -> regtopifposition2[[i]]], {i, 1, Length[graphregdata[[All, 1]]]}];
regpifgrphdata = Flatten[Delete[Union[cond1regtopif, cond2regtopif], 1]];
finalgrphdata = Union[Flatten[Join[regpifgrphdata, grph1]]]

rifagrpresults = graphRIFA;

Print[
Style[
"Transcription factor and correlated PIF gene network:",
Bold]];

Graph[MapThread[Tooltip,
{Join[rifagrpresults[[All, 1]], rifagrpresults[[All, 2]],
   \"POI\"]], Join[rifagrpresults[[All, 1]],
   rifagrpresults[[All, 2]], \"POI\"]}], rifagrpresults,
DirectedEdges \to False,
GraphHighlight \to Select[rifagrpresults, 
#[[2]] \to \"POI\" \&],
GraphLayout \to \"RadialDrawing\"
]
Gaining Confidence in RIFA

Three primary lines of evidence show that RIFA is performing to design specifications. Evidence line 1: RIFA was created to provide a Mathematica implementation of the regulatory impact factor algorithm originally described by Hudson et al. to highlight those genes most directly responsible for driving changes in phenotype. During development, RIFA was vetted with the original data used by these authors, to highlight the genes most responsible for driving phenotypic differences between Wagyu and Piedmontese cattle. The most prominent (and well-characterized) phenotypic difference between these breeds of cattle is the increased musculature of Piedmontese animals, which is known to be due to a mutation in the breed’s myostatin (GDF8) gene [7]. Using the author’s own data, RIFA correctly identifies GDF8 at the bottom (most negative value of −3.02) of its RIF5 output.

combosortbyrif5[[-3, -1]] (* do not execute, output from another dataset *)

([FOXO1, -4.06869, -5.24315, -2.89423],
 [GATA3, -3.97711, -5.05679, -2.89744],
 [GDF8, -2.77124, -2.51563, -3.02685])

Evidence line 2: The Piedmontese/Wagyu data represents the only dataset that is completely and publicly available to validate RIFA. For this reason, evidence line 1 represents the strongest line of evidence that RIFA is functioning properly, as RIFA is able to duplicate the results of Hudson et al. Even so, other gene expression datasets have been analyzed and discussed in the literature that allow for comparison to RIFA output. Please keep in mind that RIFA’s results cannot be identical to these other examples, as the full list of normalized gene expression data and the full list of transcription regulators, both necessary input to RIFA, are not publicly available. Reverter and colleagues discuss their analysis of porcine gene expression data from [8] and attempt to explain why their results do not prioritize SRY, a gene that is arguably one of the most important sex-determining genes known to science [9]. Reanalysis of this same data using normalized gene expression from AffyDGED and an alternative list of gene regulators shows that RIFA does highlight SRY as the fourth most negatively prioritized gene. While being a satisfying result, it also serves to highlight the fact that all algorithms (RIFA included) are sensitive to the quality of the input data provided to them.

Evidence line 3: Keeping in mind the discussion above, a similar reanalysis of data referenced in [10–11] shows that RIFA highlights CDK8 [12] as the 33rd most negatively prioritized gene. Hudson’s analysis of this data highlights CDK8 as the fourth most positive regulator of colon cancer. Why does RIFA rank CDK8 at position 33, while Hudson’s analysis ranks it at 4? The most reasonable explanation is that RIFA processed a transcription regulator list that included 6,685 regulators, while Hudson’s work employed a smaller regulator list of 1,292 entries. Fourth out of 1292 (4 divided by 1292) is 0.0031, while 33rd out of 6685 is 0.0049. On a percentage basis of the transcription regulator list size, RIFA’s output is nearly identical to that obtained by the original authors who developed the algorithm.
Interpreting RIFA Results

From the evidence presented, we know that RIFA is performing as expected and can begin to ask if results from other studies make biological sense. If RIFA is working correctly, it should highlight genes that have been linked to cirrhosis and/or cancer in the scientific literature. Keep in mind that not every gene likely to be linked to cirrhosis or cancer has been discovered or characterized yet—which is the value in using a program like RIFA, that is, to find new connections between genes and phenotype. A small sample of output will be reprinted for easier referencing here. Additionally, the output will be reformatted to fit within printable margins by forcing the data in each entry to occupy two or more rows of the table below.

```mathematica
optempl1 = Join[combosortbyavg[[1 ;; 10]],
    combosortbyavg[[-10 ;; -1]], combosortbyrif4[[1 ;; 10]],
    combosortbyrif4[[-10 ;; -1]], combosortbyrif5[[1 ;; 10]],
    combosortbyrif5[[-10 ;; -1]]];

optempl1[[6, 7]] = 
    "Homo sapiens mRNA;\ncDNA DKFZp667F2113\n(from clone DKFZp667F2113)";

optempl1[[8, 7]] = 
    "Homo sapiens cDNA FLJ21380 fis,\ncClone MAMMA1002556";

optempl1[[10, 7]] = 
    "peroxisome proliferative activated receptor\ngamma, coactivator 1";

optempl1[[13, 7]] = 
    "TATA box binding protein (TBP)-associated factor,\nRNA polymerase I, A, 48kD";

optempl1[[15, 7]] = 
    "Homo sapiens cDNA: FLJ22281 fis,\ncClone HRC03849,\nhighly similar to S69002 human mRNA\nfor AML1-EVI-1";

optempl1[[19, 7]] = 
    "Homo sapiens cDNA FLJ11655 fis,\ncClone HEMBA1004554";

optempl1[[20, 7]] = 
    "ESTs, weakly similar to S47072 finger protein\nHZF10,\nKrueppel-related [H.sapiens]";

optempl1[[33, 7]] = 
    "signal transducer and activator of\ntranscription\n3 (acute-phase response factor)";

optempl1[[36, 7]] = 
    "Homo sapiens cDNA FLJ11655 fis,\ncClone HEMBA1004554";

optempl1[[40, 7]] = 
    "ESTs, weakly similar to S47072 finger protein\nHZF10,\nKrueppel-related [H.sapiens]";

optempl1[[41, 7]] = 
    "Homo sapiens mRNA;\ncDNA DKFZp434P228 (from clone DKFZp434P228)";

optempl1[[42, 7]] = 
    "nuclear receptor subfamily 4,\ngroup A, member 3";

optempl1[[47, 7]] =
```
"nuclear receptor subfamily 4, \( \text{group A, member 2} \);"

\( \text{optempl1[[50, 7]]} = \)
"v-\( \text{maf musculoaponeurotic fibrosarcoma (avian)} \)
\( \text{oncogene family, protein F}};\)

\( \text{optempl1[[52, 7]]} = \)
"ESTs, highly similar to B45036 Pur beta [H.sapiens]";

\( \text{optempl1[[53, 7]]} = \)
"ESTs, weakly similar to A32891 finger protein 1, nplacentals [H.sapiens]";

\( \text{optempl1[[54, 7]]} = \)
"Homo sapiens mRNA; ncDNA DKFZp566P1124 (from clone DKFZp566P1124)";

\( \text{optempl1[[55, 7]]} = \)
"Homo sapiens cDNA FLJ11344 fis, nclone PLACE1010870, moderately similar to zinc finger protein 91";

\( \text{optemp2} = \text{optempl1[[All, 1 ;; 4]]};\)

\( \text{optemp3} = \text{optempl1[[All, 5 ;; 7]]};\)

\( \text{optemp4} = \)
\( \text{Text[Grid[Riffle[optemp2, optemp3],} \)
\( \text{Dividers} \rightarrow \{\text{False, \{True, False\}}\}]]\)
<table>
<thead>
<tr>
<th>Accession</th>
<th>Gene Symbol</th>
<th>Description</th>
<th>Log2Ratio</th>
<th>Log2FoldChange</th>
<th>P-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>gb:NM_024680.1</td>
<td>FLJ23311</td>
<td>hypothetical protein FLJ23311</td>
<td>-3.0847</td>
<td>-4.4771</td>
<td>-1.69983</td>
</tr>
<tr>
<td>gb:BE466525</td>
<td>Homo sapiens cDNA: FLJ22881 fis, clone HRC03849, highly similar to S69002 human mRNA for AML1-EVI-1</td>
<td>None</td>
<td>-3.29527</td>
<td>-4.32838</td>
<td>-2.26216</td>
</tr>
<tr>
<td>gb:NM_030937.1</td>
<td>HCLA-ISO</td>
<td>hypothetical protein hCLA-iso</td>
<td>-3.38656</td>
<td>-5.92359</td>
<td>-0.849537</td>
</tr>
<tr>
<td>gb:M31523.1</td>
<td>TCF3</td>
<td>None</td>
<td>0.97311</td>
<td>8.63574</td>
<td>0.689522</td>
</tr>
<tr>
<td>gb:NM_004089.1</td>
<td>DSIPI</td>
<td>delta sleep inducing peptide, immunoreactor</td>
<td>2.14857</td>
<td>5.78666</td>
<td>-1.48953</td>
</tr>
<tr>
<td>gb:NM_004343.2</td>
<td>CALR</td>
<td>calreticulin precursor</td>
<td>2.06738</td>
<td>5.57004</td>
<td>-1.23529</td>
</tr>
<tr>
<td>gb:AF207829.1</td>
<td>RAZ1</td>
<td>SCAN-related protein RAZ1</td>
<td>1.56291</td>
<td>4.67727</td>
<td>-1.55145</td>
</tr>
<tr>
<td>gb:AL133630.1</td>
<td>KLFZp454N0223</td>
<td>hypothetical protein</td>
<td>1.35598</td>
<td>4.57285</td>
<td>-1.86088</td>
</tr>
<tr>
<td>gb:NM_004509.1</td>
<td>IFI41</td>
<td>interferon-induced protein 41, 30kD</td>
<td>1.60648</td>
<td>4.38909</td>
<td>-1.17613</td>
</tr>
<tr>
<td>gb:AK022442.1</td>
<td>Homo sapiens cDNA FLJ11655 fis, clone HEMBA1004554</td>
<td>ESTs</td>
<td>2.12408</td>
<td>6.95010</td>
<td>-1.32836</td>
</tr>
<tr>
<td>gb:AF478446.1</td>
<td>NR1H4</td>
<td>farnesoid-X-receptor beta splice variant 2</td>
<td>1.28941</td>
<td>4.57285</td>
<td>3.469354</td>
</tr>
<tr>
<td>gb:AW780006</td>
<td>KIAA1278 protein</td>
<td>None</td>
<td>0.638527</td>
<td>4.18173</td>
<td>-2.90467</td>
</tr>
<tr>
<td>gb:AF230394.1</td>
<td>RAB18, member RAS oncogene family</td>
<td>None</td>
<td>1.14794</td>
<td>5.98942</td>
<td>3.69354</td>
</tr>
<tr>
<td>gb:AF274957.1</td>
<td>PNAS-32</td>
<td>None</td>
<td>-1.14794</td>
<td>-5.98942</td>
<td>3.69354</td>
</tr>
<tr>
<td>gb:AA634272</td>
<td>PNAS-32</td>
<td>signal transducer and activator of transcription 3 (acute-phase response factor)</td>
<td>-1.12408</td>
<td>-6.18497</td>
<td>3.936681</td>
</tr>
<tr>
<td>gb:AF230394.1</td>
<td>RAB18, member RAS oncogene family</td>
<td>None</td>
<td>1.14794</td>
<td>5.98942</td>
<td>3.69354</td>
</tr>
<tr>
<td>gb:AF274957.1</td>
<td>PNAS-32</td>
<td>None</td>
<td>-1.14794</td>
<td>-5.98942</td>
<td>3.69354</td>
</tr>
<tr>
<td>gb:AA634272</td>
<td>PNAS-32</td>
<td>signal transducer and activator of transcription 3 (acute-phase response factor)</td>
<td>-1.12408</td>
<td>-6.18497</td>
<td>3.936681</td>
</tr>
</tbody>
</table>

The Mathematica Journal 17 © 2015 Wolfram Media, Inc.
<table>
<thead>
<tr>
<th>Gene Identifier</th>
<th>Log2 Fold Change</th>
<th>P-value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1554375_a_at</td>
<td>-1.59955</td>
<td>3.98611</td>
<td>Farnesoid-X-receptor beta splice variant 2</td>
</tr>
<tr>
<td>gb:AF478446.1</td>
<td>-7.18522</td>
<td></td>
<td>Homo sapiens chromosome 19, cosmid R33590</td>
</tr>
<tr>
<td>234361_at</td>
<td>-2.01425</td>
<td>4.50073</td>
<td>Homo sapiens mRNA, cDNA DKFZp434P228 (from clone DKFZp434P228)</td>
</tr>
<tr>
<td>gb:NM_005354.2</td>
<td>-8.52923</td>
<td></td>
<td>Homo sapiens mRNA, cDNA DKFZp434P228 (from clone DKFZp434P228)</td>
</tr>
<tr>
<td>gb:NM_004509.1</td>
<td>-8.63574</td>
<td>0.689522</td>
<td>Homo sapiens mRNA, cDNA DKFZp434P228 (from clone DKFZp434P228)</td>
</tr>
<tr>
<td>gb:NM_030937.1</td>
<td>-4.66577</td>
<td>5.39053</td>
<td>Homo sapiens mRNA, cDNA DKFZp434P228 (from clone DKFZp434P228)</td>
</tr>
<tr>
<td>gb:AL117590.1</td>
<td>-3.31468</td>
<td>4.9133</td>
<td>Homo sapiens mRNA, cDNA DKFZp434P228 (from clone DKFZp434P228)</td>
</tr>
<tr>
<td>gb:NM_004343.2</td>
<td>-5.38041</td>
<td>4.86562</td>
<td>Homo sapiens mRNA, cDNA DKFZp434P228 (from clone DKFZp434P228)</td>
</tr>
<tr>
<td>gb:NM_030937.1</td>
<td>-2.41524</td>
<td>4.84516</td>
<td>Homo sapiens mRNA, cDNA DKFZp434P228 (from clone DKFZp434P228)</td>
</tr>
<tr>
<td>gb:AL518328</td>
<td>-4.32905</td>
<td>4.83752</td>
<td>Homo sapiens mRNA, cDNA DKFZp434P228 (from clone DKFZp434P228)</td>
</tr>
<tr>
<td>gb:ST7154.1</td>
<td>-2.48064</td>
<td>4.81205</td>
<td>Homo sapiens mRNA, cDNA DKFZp434P228 (from clone DKFZp434P228)</td>
</tr>
<tr>
<td>gb:NM_005681.1</td>
<td>-3.3539</td>
<td>4.5743</td>
<td>Homo sapiens mRNA, cDNA DKFZp434P228 (from clone DKFZp434P228)</td>
</tr>
<tr>
<td>gb:AL2323.1</td>
<td>-6.42241</td>
<td>4.73019</td>
<td>Homo sapiens mRNA, cDNA DKFZp434P228 (from clone DKFZp434P228)</td>
</tr>
<tr>
<td>gb:AL028632.1</td>
<td>-1.29754</td>
<td>4.64381</td>
<td>Homo sapiens mRNA, cDNA DKFZp434P228 (from clone DKFZp434P228)</td>
</tr>
<tr>
<td>gb:NC_004932.1</td>
<td>-3.3539</td>
<td>4.5743</td>
<td>Homo sapiens mRNA, cDNA DKFZp434P228 (from clone DKFZp434P228)</td>
</tr>
<tr>
<td>gb:NM_000862</td>
<td>-3.01468</td>
<td>4.9133</td>
<td>Homo sapiens mRNA, cDNA DKFZp434P228 (from clone DKFZp434P228)</td>
</tr>
<tr>
<td>gb:AL038866.1</td>
<td>-2.372165</td>
<td></td>
<td>Homo sapiens mRNA, cDNA DKFZp434P228 (from clone DKFZp434P228)</td>
</tr>
<tr>
<td>gb:AW157773</td>
<td>-0.910755</td>
<td>2.8257</td>
<td>Homo sapiens mRNA, cDNA DKFZp434P228 (from clone DKFZp434P228)</td>
</tr>
<tr>
<td>gb:BC005868.1</td>
<td>-2.27466</td>
<td>2.83599</td>
<td>Homo sapiens mRNA, cDNA DKFZp434P228 (from clone DKFZp434P228)</td>
</tr>
<tr>
<td>gb:AW780006</td>
<td>-2.90467</td>
<td></td>
<td>Homo sapiens mRNA, cDNA DKFZp434P228 (from clone DKFZp434P228)</td>
</tr>
<tr>
<td>gb:AL049589.1</td>
<td>-3.06202</td>
<td></td>
<td>Homo sapiens mRNA, cDNA DKFZp434P228 (from clone DKFZp434P228)</td>
</tr>
<tr>
<td>gb:AA490928</td>
<td>-3.06457</td>
<td></td>
<td>Homo sapiens mRNA, cDNA DKFZp434P228 (from clone DKFZp434P228)</td>
</tr>
<tr>
<td>gb:BC05148.1</td>
<td>-3.19123</td>
<td></td>
<td>Homo sapiens mRNA, cDNA DKFZp434P228 (from clone DKFZp434P228)</td>
</tr>
</tbody>
</table>
What information is revealed in this list? First, let us obtain a list of the unique entries present in this list (a handful of entries may be prioritized by RIF4 and RIF5 metrics simultaneously, thus showing up more than once).

\[
\text{optemp5} = \text{Union[optemp1]};
\]

Do any of the results contain entries that have been linked to cirrhotic or tumorous livers in the scientific literature? Any entry that has a gene name associated with it may have information that can be investigated further.

\[
\text{optemp6} = \text{Select[optemp5, \#[6] \neq \text{None} \& \{\text{All, 6}\}]
\]

\[
\{\text{NR1H4, KIAA0005, CALR, UVO, TOP2A, JUND, NR4A2, HLF, ZNF274, MAFF, HOXD4, TAF1A, DSIP1, NR4A3, IFI41, STAT3, TCF3, CAT, CCNT2, SOD2, TINUR, PPARGC1, FLJ23311, HCLA-ISO, RAB18, LOC51616, KIAA1278, DKFZp434N0223, RAZ1}\}
\]

\[
\text{Length[optemp6]}
\]

29

RIFA produces a list of 29 unique gene names that can be searched for in PubMed (www.ncbi.nlm.nih.gov/pubmed). Performing a literature search for these genes in association with liver disease search terms produces the results described in Table 1.

<table>
<thead>
<tr>
<th>Gene 1</th>
<th>Gene 2</th>
<th>Gene 3</th>
<th>Gene 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOD2 [33]</td>
<td>TINUR [34]</td>
<td>PPARGC1 [35]</td>
<td>HCLA-ISO [36]</td>
</tr>
<tr>
<td>RAB18 [37]</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

▲ Table 1. Results of a PubMed literature search using the names of the genes above in combination with one or more of the following search terms: “liver cancer,” “liver cirrhosis,” “cancer.” Citations listed represent a small sampling of the total hits typically discovered.
Twenty-one of the 29 RIFA output entries with a gene name associated with them yield compelling connections between each gene and the disease phenotype search terms “liver cirrhosis,” “liver cancer,” and “cancer,” suggesting that RIFA is enriching for genes driving the phenotypic changes observed between cirrhotic and tumorous liver tissue.

The remaining eight out of 29 genes do not show evidence in the scientific literature linking them to these disease phenotypes. Explanations for this abound, but it is impossible to rule out the possibility that these genes are, in fact, linked to the disease phenotypes but have not yet been characterized by the scientific community. It is simply impossible to conclude if those eight genes are or are not linked to the disease phenotypes at this time. The same conclusion must also be admitted for the other 25 RIFA output entries that have no gene name associated with them. In other words, RIFA has identified 25 potential new “boss” genes associated with the cirrhotic to tumor transition in liver tissue. These may represent valuable new avenues of research.

**RIFA Performance**

To gauge the performance of RIFA, several publicly available datasets of different sizes and complexity were analyzed. The first column of Table 2 shows the series accession number for each dataset available at NCBI’s Gene Expression Omnibus. Timings were acquired running *Mathematica* 9.0.1 under Windows 7 (64 bit) using an Intel Core i5-2500K processor overclocked to 4.48Ghz. Total system memory is 32GB. All reported timings use a fresh kernel.

Table 2 reveals that small datasets can easily be processed in under one minute, while very large datasets, involving thousands of transcription regulators, differentially expressed genes, and multiple time points can take upwards of 30 minutes. RIFA’s code base utilizes functions with the `Listable` attribute whenever possible to increase speed, which places demands on the computer’s memory infrastructure, as evidenced by the sizeable memory consumption measured with large datasets.

<table>
<thead>
<tr>
<th>Series accession number</th>
<th>Time (sec)</th>
<th>Number of transcription regulators</th>
<th>Number of differentially expressed genes</th>
<th>Number of time points per condition</th>
<th>Max memory used (bytes)</th>
</tr>
</thead>
<tbody>
<tr>
<td>GSE14739</td>
<td>19.4</td>
<td>354</td>
<td>2411</td>
<td>4</td>
<td>456 369 816</td>
</tr>
<tr>
<td>GSE7032</td>
<td>21.5</td>
<td>1595</td>
<td>1113</td>
<td>2</td>
<td>522 299 688</td>
</tr>
<tr>
<td>GSE8536</td>
<td>94.4</td>
<td>766</td>
<td>6685</td>
<td>6</td>
<td>3 561 336 832</td>
</tr>
<tr>
<td>GSE17548</td>
<td>1391</td>
<td>6685</td>
<td>8830</td>
<td>6</td>
<td>40 899 150 240</td>
</tr>
<tr>
<td>GSE4183</td>
<td>1883</td>
<td>6685</td>
<td>7777</td>
<td>8</td>
<td>47 534 803 296</td>
</tr>
</tbody>
</table>

▲ Table 2. Performance timings of RIFA using five different, publicly available datasets.
Conclusion

Changes in gene expression are at the core of what distinguishes healthy tissue from diseased tissue. Part of unraveling the mystery behind disease centers on identifying those genes most directly responsible for controlling the differences in gene expression that link those differences to disease traits. RIFA’s implementation brings to the Mathematica user community a compelling algorithm used by biomedical researchers to intelligently prioritize the thousands of genes present in an organism and tie their behavior to specific traits of interest.

References


■ About the Author

Todd Allen is an associate professor of biology at HACC, Lancaster. His interest in computational biology using Mathematica took shape during his postdoctoral research years at the University of Maryland, where he developed a custom cDNA microarray chip to study gene expression changes in the chestnut blight pathogen, Cryphonectria parasitica.

Todd D. Allen, Ph.D.
Harrisburg Area Community College (Lancaster Campus)
East 206R
1641 Old Philadelphia Pike
Lancaster, PA 17602
tdallen@hacc.edu