MathIOmica: Dynamic Transcriptome

Loading the MathIOmica Package

The functions defined in the MathIOmica context provide support for conducting analyses of omics data (See also the MathIOmica Overview).

This loads the package:

```
In[8]:= << MathIOmica`
```

Importing OmicsObject Transcriptome Data

We first import the transcriptomics data example (for details on how to import such data please refer to Dataimporter, DataimporterDirect, DataimporterDirectLabeled and OmicsObjectCreator documentation).

```
In[9]:= rnaExample = Get[FileNameJoin[\{ConstantMathIOmicaExamplesDirectory, "rnaExample"\}]]
```

```
Out[9]= <|7 -> <|\{FAM138A, RNA\} -> \{\{0\}, \{OK\}\}, \{OR4F5, RNA\} -> \{\{0\}, \{OK\}\},
\{LOC729737, RNA\} -> \{\{2.73998\}, \{OK\}\}, \{PML\}|
\{LOC100507412, RNA\} -> \{\{0\}, \{OK\}\}, \{RNA4555, RNA\} -> \{\{0\}, \{OK\}\}, \{DUX4L, RNA\} -> \{\{0\}, \{OK\}\}|>,
8 -> <|\{\}>, \{11\}, 20 -> <\{\}>, 21 -> <|\{\}|>|
```

```
large output  show less  show more  show all  set size limit...
```
There are multiple samples given by the outer associations. We can use Query to get any data. For example we can get the outer keys:

```
In[10]:= Query[Keys]@rnaExample
Out[10]= {7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21}
```

We form an association between samples to actual days of the study:

```
In[11]:= sampleToDays = 
```

We can now do a KeyMap to rename the outer keys:

```
In[12]:= rnaLongitudinal = KeyMap[sampleToDays, rnaExample]
```

```
large output show less show more show all set size limit...
```

```
Out[12]=
   <|186 -> <|{FAM138A, RNA} -> {{0.}, {OK}}, {OR4F5, RNA} -> {{0.}, {OK}}, {LOC729737, RNA} -> {{2.73998}, {OK}}|>, 255 -> <|{FAM138A, RNA} -> {{0.}, {OK}}, {LOC100507412, RNA} -> {{0.}, {OK}}, {OR4F5, RNA} -> {{0.}, {OK}}, {DUX4L, RNA} -> {{0.}, {OK}}|>, 
   380 -> <|{FAM138A, RNA} -> {{0.}, {OK}}, {DUX4L, RNA} -> {{0.}, {OK}}|>, 400 -> <|{FAM138A, RNA} -> {{0.}, {OK}}|>
```

```
large output show less show more show all set size limit...
```

Processing OmicsObject Transcriptome Data

We normalize the transcriptome data using the QuantileNormalization function.

```
In[13]:= rnaQuantileNormalized = QuantileNormalization[rnaLongitudinal, ListIndex -> 1, ComponentIndex -> 1]
```

```
large output show less show more show all set size limit...
```

```
Out[13]=
   <|186 -> <|{FAM138A, RNA} -> {{0.}, {OK}}, {OR4F5, RNA} -> {{0.}, {OK}}, {LOC729737, RNA} -> {{2.2946}, {OK}}|>, 255 -> <|{FAM138A, RNA} -> {{0.}, {OK}}, {OR4F5, RNA} -> {{0.}, {OK}}, {LOC729737, RNA} -> {{0.}, {OK}}, {DUX4L, RNA} -> {{0.}, {OK}}|>, 
   380 -> <|{FAM138A, RNA} -> {{0.}, {OK}}, {OR4F5, RNA} -> {{0.}, {OK}}, {DUX4L, RNA} -> {{0.}, {OK}}|>, 400 -> <|{FAM138A, RNA} -> {{0.}, {OK}}|>
```

```
large output show less show more show all set size limit...
```

We first use LowValueTag to tag values of 0 as Missing:

```
In[14]:= rnaZeroTagged = LowValueTag[rnaQuantileNormalized, 0]
```

```
large output show less show more show all set size limit...
```

```
Out[14]=
   <|186 -> <|{FAM138A, RNA} -> {{Missing[]}, {OK}}, {OR4F5, RNA} -> {{Missing[]}, {OK}}, {LOC729737, RNA} -> {{2.2946}, {OK}}|>, 255 -> <|{FAM138A, RNA} -> {{Missing[]}, {OK}}, {OR4F5, RNA} -> {{Missing[]}, {OK}}, {LOC729737, RNA} -> {{2.2946}, {OK}}|>, 
   380 -> <|{FAM138A, RNA} -> {{Missing[]}, {OK}}, {DUX4L, RNA} -> {{Missing[]}, {OK}}|>, 400 -> <|{FAM138A, RNA} -> {{Missing[]}, {OK}}|>
```

```
large output show less show more show all set size limit...
```
We next use `LowValueTag` again to set all FPKM values $< 1$ to unity:

```wolfram
In[15]:= rnaNoiseAdjusted = LowValueTag[rnaZeroTagged, 1, ValueReplacement -> 1]
```

```
Out[15]=

```

We filter out data using `FilterMissing` where the reference healthy point "255" is missing and retain data with at least 3/4 points available:

```wolfram
In[16]:= rnaFiltered = FilterMissing[rnaNoiseAdjusted, 3/4, Reference -> "255"]
```
We extract the times for the filtered RNA data using TimeExtractor:

\[ \text{In[17]:= TimeExtractor[rnaFiltered]} \]

For each gene we now extract a time series (list of values) corresponding to these times using CreateTimeSeries:

\[ \text{In[18]:=} \quad \text{timeSeriesRNA} = \text{CreateTimeSeries[\text{rnaFiltered}]} \]

\[ \{ \{ \text{FAM138A}, \text{RNA} \to \{\text{Missing}, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1\} \}, \{ \text{OR4F5}, \text{RNA} \to \{\text{Missing}, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1\} \}, \{ \text{LOC729737}, \text{RNA} \to \{2.2946, 1, 4.67694, 4.48131, 4.95597, 1, 1.25726, 2.14767, 1.93219, 1, 2.58217, 2.31381, 4.10284, 3.80929, 1.45471\} \}, \{ \text{DDX11L1}, \text{RNA} \to \{5.91665, 4.32803, 3.19599, 3.64164, 2.7327, 2.13461, 2.17168, 3.2342, 1.98576, 3.0267, 4.34004, 7.27601, 2.01132, 0.27701, 7.54415\} \}, \{ \text{RNA5-SS5, RNA} \to \{\text{Missing}, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1\} \}, \{ \text{LOC100597412, RNA} \to \{\text{Missing}, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1\} \}, \{ \text{RNA4555, RNA} \to \{\text{Missing}, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1\} \}, \{ \text{DXA4L1, RNA} \to \{\text{Missing}, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1\}\} \]  

\[ \text{Out[18]}= \]

We use SeriesApplier to implement a logarithm transformation:

\[ \text{In[19]:=} \quad \text{timeSeriesRNALog} = \text{SeriesApplier[Log, timeSeriesRNA]} \]

\[ \{ \{ \text{FAM138A}, \text{RNA} \to \{\text{Missing}, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\} \}, \{ \text{OR4F5}, \text{RNA} \to \{\text{Missing}, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\} \}, \{ \text{LOC729737}, \text{RNA} \to \{0.830556, 0, 1.54264, 1.49992, 1.60641, 0, 0.228935, 0.764385, 0.658653, 0, 0.94863, 0.838548, 1.41168, 1.33744, 0.374807\} \}, \{ \text{DDX11L1}, \text{RNA} \to \{1.77777, 1.46344, 1.1619, 1.29243, 1.09529, 0.758282, 0.775501, 1.17381, 0.639619, 1.10747, 1.46788, 1.98376, 0.698792, 2.22754, 2.02077\} \}, \{ \text{RNA5-SS5, RNA} \to \{\text{Missing}, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\} \}, \{ \text{LOC100597412, RNA} \to \{\text{Missing}, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\} \}, \{ \text{RNA4555, RNA} \to \{\text{Missing}, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\} \}, \{ \text{DXA4L1, RNA} \to \{\text{Missing}, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\}\} \]  

\[ \text{Out[19]}= \]

We compare every value in each series to the healthy "2SS" time point, which is the second element in each series. We use SeriesInternalCompare:

\[ \text{In[20]:=} \quad \text{rnaCompared} = \text{SeriesInternalCompare[timeSeriesRNALog, \text{ComparisonIndex} \to 2]} \]

\[ \{ \{ \text{FAM138A}, \text{RNA} \to \{\text{Missing}, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\} \}, \{ \text{OR4F5}, \text{RNA} \to \{\text{Missing}, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\} \}, \{ \text{LOC729737}, \text{RNA} \to \{0.830556, 0, 1.54264, 1.49992, 1.60641, 0, 0.228935, 0.764385, 0.658653, 0, 0.94863, 0.838548, 1.41168, 1.33744, 0.374807\} \}, \{ \text{DDX11L1}, \text{RNA} \to \{0.314326, 0, -0.301545, -0.171011, -0.458154, -0.705162, -0.687943, -0.289634, -0.823824, -0.35597, 0.0444068, 0.520314, -0.764652, 0.764695, 0.557328\} \}, \{ \text{RNA5-SS5, RNA} \to \{\text{Missing}, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\} \}, \{ \text{LOC100597412, RNA} \to \{\text{Missing}, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\} \}, \{ \text{RNA4555, RNA} \to \{\text{Missing}, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\} \}, \{ \text{DXA4L1, RNA} \to \{\text{Missing}, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\}\} \]  

\[ \text{Out[20]}= \]
Next, we normalize each series, using again SeriesApplier:

In[21]:= normedRNACompared = SeriesApplier[{Normalize, RNACompared}]

Out[21]= <|{FAM138A, RNA} -> {Missing[], 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
{OR4F5, RNA} -> {Missing[], 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
{LOC729737, RNA} -> {0.218293, 0., 0.40545, 0.39422, 0.420632, 0., 0.0681705,
0.209092, 0.173112, 0., 0.249326, 0.220394, 0.371029, 0.351517, 0.0958097,
0.025826, 0.156411, 0., -0.159051, -0.0850959, -0.22798, -0.358093, -0.343234,
-0.1414124, -0.46994, -0.177133, 0.00220971, 0.2589111, -0.3804055, 0.380218, 0.277333},
{RNA5-SS5, RNA} -> {Missing[], 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
{LOC100597412, RNA} -> {Missing[], 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
{RNA4555, RNA} -> {Missing[], 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
{DXX4L, RNA} -> {Missing[], 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}|

Large output: show less show more show all set size limit...

Finally, we use ConstantSeriesClean to remove constant series, as we are interested in changing time patterns:

In[22]:= rnaFinalTimeSeries = ConstantSeriesClean[normedRNACompared]

Removed series and returning filtered list. If you would like a list of removed keys run the command ConstantSeriesClean[data, ReturnDropped -> True].

Out[22]= <|{LOC729737, RNA} -> {0.218293, 0., 0.40545, 0.39422, 0.420632, 0.,
0.0681705, 0.209092, 0.173112, 0., 0.249326, 0.220394, 0.371029, 0.351517, 0.0958097,
0.025826, 0.156411, 0., -0.159051, -0.0850959, -0.22798, -0.358093, -0.343234,
-0.1414124, -0.46994, -0.177133, 0.00220971, 0.2589111, -0.3804055, 0.380218, 0.277333},
{RNA5-SS5, RNA} -> {Missing[], 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
{LOC100597412, RNA} -> {Missing[], 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
{RNA4555, RNA} -> {Missing[], 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
{DXX4L, RNA} -> {Missing[], 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}|

Large output: show less show more show all set size limit...

---

**Resampling Transcriptome Data**

In addition to the above, we want to create a resampled distribution for the transcriptome dataset prior to classification and clustering. We repeat the steps in the processing section above using a resampled set of measurements.

We create a resampling of 100000 sets using BootstrapGeneral:

In[23]:= rnaBootstrap = BootstrapGeneral[rnaLongitudinal, 100000]

Out[23]= <|186 -> {1 -> {{0}, (OK)}, 2 -> {{0}, (OK)}, 3 -> {{10.0429}, (OK)}},
4 -> {{12.8612}, (OK)}, 5 -> {{2.37963}, (OK)}}, 6 -> {{0.0850959}, 99997 -> {{43.8242}, (OK)}},
99998 -> {{0.468797}, (OK)}, 99999 -> {{4.68299}, (OK)}, 100000 -> {{3.29531}, (OK)}},
255 -> {{0}, (OK)}, 400 -> {{0}, (OK)}|>
As with the regular data we: 1. normalize, 2. tag zero values, 3. tag values of FPKM <1, 4. filter missing data, 5. create a time series, 6. take a logarithm, 7. compare to "255" reference, 8. take the norm of each time series, 9. clean out constant series

\[
\begin{align*}
\text{In[24]} &= (*1*) \text{rnaBootstrapQuantileNormed} = \text{QuantileNormalization}\left[\text{rnaBootstrap}, \text{ListIndex} \rightarrow 1, \text{ComponentIndex} \rightarrow 1\right]; \\
\text{In[24]} &= (*2*) \text{rnaBootstrapZeroTagged} = \text{LowValueTag}\left[\text{rnaBootstrapQuantileNormed}, 0\right]; \\
\text{In[24]} &= (*3*) \text{rnaBootstrapNoiseAdjusted} = \text{LowValueTag}\left[\text{rnaBootstrapZeroTagged}, 1, \text{ValueReplacement} \rightarrow 1\right]; \\
\text{In[24]} &= (*4*) \text{rnaBootstrapFiltered} = \text{FilterMissing}\left[\text{rnaBootstrapNoiseAdjusted}, 3/4, \text{Reference} \rightarrow "255"\right]; \\
\text{In[24]} &= (*5*) \text{timeSeriesBootstrapRNA} = \text{CreateTimeSeries}\left[\text{rnaBootstrapFiltered}\right]; \\
\text{In[24]} &= (*6*) \text{timeSeriesBootstrapRNALog} = \text{SeriesApplier}\left[\text{Log}, \text{timeSeriesBootstrapRNA}\right]; \\
\text{In[24]} &= (*7*) \text{rnaBootstrapCompared} = \text{SeriesInternalCompare}\left[\text{timeSeriesBootstrapRNALog}, \text{ComparisonIndex} \rightarrow 2\right]; \\
\text{In[24]} &= (*8*) \text{normedBootstrapRNACompared} = \text{SeriesApplier}\left[\text{Normalize}, \text{rnaBootstrapCompared}\right]; \\
\text{In[24]} &= (*9*) \text{rnaBootstrapFinalTimeSeries} = \text{ConstantSeriesClean}\left[\text{normedBootstrapRNACompared}\right];
\end{align*}
\]
Classification, Clustering and Visualization of Transcriptome Time Series

In this section we will classify the transcriptome time series based on patterns in the series. For the classification we will use `TimeSeriesClassification`.

Removed series and returning filtered list. If you would like a list of removed keys run the command `ConstantSeriesClean[data, ReturnDropped -> True]`.

Classification, Clustering and Visualization of Transcriptome Time Series

In this section we will classify the transcriptome time series based on patterns in the series. For the classification we will use `TimeSeriesClassification`.

Classification, Clustering and Visualization of Transcriptome Time Series

In this section we will classify the transcriptome time series based on patterns in the series. For the classification we will use `TimeSeriesClassification`.

Classification, Clustering and Visualization of Transcriptome Time Series

In this section we will classify the transcriptome time series based on patterns in the series. For the classification we will use `TimeSeriesClassification`.

Classification, Clustering and Visualization of Transcriptome Time Series

In this section we will classify the transcriptome time series based on patterns in the series. For the classification we will use `TimeSeriesClassification`.

Classification, Clustering and Visualization of Transcriptome Time Series

In this section we will classify the transcriptome time series based on patterns in the series. For the classification we will use `TimeSeriesClassification`.
Before we classify our transcriptome data, we estimate for the "LombScargle" Method a 0.95 quantile cutoff from the bootstrap transcriptome data using `QuantileEstimator`:

```mathematica
In[33]:= q95RNA = QuantileEstimator[rnaBootstrapFinalTimeSeries, timesRNA]
Out[33]= 0.85987
```

Next, we estimate the "Spikes" 0.95 quantile cutoff from the bootstrap transcriptome data:

```mathematica
In[34]:= q95RNASpikes = QuantileEstimator[rnaBootstrapFinalTimeSeries, timesRNA, Method -> "Spikes"]
Out[34]= {{14 -> {0.886757, -0.348387}, 15 -> {0.861302, -0.337344}}
```

Now we can classify the transcriptome time series data based on these cutoffs using `TimeSeriesClassification`:

```mathematica
In[35]:= rnaClassification = TimeSeriesClassification[rnaFinalTimeSeries, timesRNA, LombScargleCutoff -> q95RNA, SpikeCutoffs -> q95RNASpikes]
```

```
{SpikeMax -> {\[1\], \[7\]},
f7 -> {\{MIR6723, RNA\} -> \{0.214503, 0.000338299, 0.0390479, 0.0653206, 0.291434, 0.336712, 0.865961\},
\{1\}}, \{68\}, \{DNASE1L1, RNA\} -> \[1\]} }
```

To obtain the possible frequencies we simply run `LombScargle` over the desired times for one of the time series and set the `FrequenciesOnly` option to True:

```mathematica
In[36]:= LombScargle[rnaFinalTimeSeries[[1]], timesRNA, FrequenciesOnly -> True]
Out[36]= {f1 -> 0.00500668, f2 -> 0.0104306, f3 -> 0.0158545, f4 -> 0.0212784, f5 -> 0.0267023, f6 -> 0.0321262, f7 -> 0.0375501}
```

We now cluster our RNA data using `TimeSeriesClusters`:

```mathematica
In[37]:= rnaClusters = TimeSeriesClusters[rnaClassification, PrintDendrograms -> True]
```

```
{SpikeMax -> {\[1\], \[7\]},
f7 -> {\{MIR6723, RNA\} -> \{0.214503, 0.000338299, 0.0390479, 0.0653206, 0.291434, 0.336712, 0.865961\},
\{1\}}, \{68\}, \{DNASE1L1, RNA\} -> \[1\]} }
```

For each class we can generate a dendrogram/heatmap plot using TimeSeriesDendrogramsHeatmaps, with groupings represented on the left, and highlighted to represent the grouping level. The G, S, columns represent the groupings and subgroupings generated by the clustering. The legend shows the corresponding groupings and subgrouping, and the number of elements in each group subgroup.

In[40]:= TimeSeriesDendrogramsHeatmaps[rnaClusters]

Out[40]=

Out[39]=

large output  show less  show more  show all  set size limit...
SpikeMin

Dendrogram  HeatMap  GroupAssociations Index

Group=\text{Subgroup}\_\text{p}-\text{Size}

G  S

Time (arbitrary units)

f1

Dendrogram  HeatMap  GroupAssociations Index

Group=\text{Subgroup}\_\text{p}-\text{Size}

G  S

Time (arbitrary units)

f2

Dendrogram  HeatMap  GroupAssociations Index

Group=\text{Subgroup}\_\text{p}-\text{Size}

G  S

Time (arbitrary units)
Out[40]=

G S
Time (arbitrary units)

f3

Dendrogram  HeatMap  GroupAssociations Index
Group=Subgroup=#Size

Out[40]=

G S
Time (arbitrary units)

f4

Dendrogram  HeatMap  GroupAssociations Index
Group=Subgroup=#Size

Out[40]=

G S
Time (arbitrary units)

f5

Dendrogram  HeatMap  GroupAssociations Index
Group=Subgroup=#Size

Out[40]=

G S
Time (arbitrary units)
Annotation and Enrichment

We can carry out Gene Ontology analysis using GOAnalysis for all the classes and groups/subgroups. We only report terms for which there are at least 3 members (2 sets of GO terms, one each for proteomics and transcriptomics). Please note that this may be a time consuming computation.

\[\text{In[41]}:\quad \text{goAnalysisRNA} = \text{GOAnalysis[rnaClusters, OntologyLengthFilter \rightarrow 3, ReportFilter \rightarrow 3];}\]

The output of GOAnalysis has enrichments for each class and group.

\[\text{In[42]}:\quad \text{Query[Keys]@goAnalysisRNA}\]

\[\text{Out[42]}:\quad \{\text{SpikeMax, SpikeMin, f1, f2, f3, f4, f5, f6, f7}\}\]
We can view results for any of the groups (and also check out the behavior using the heatmaps generated in the previous section)
Outbase 2018 Wolfram: 0033574 0000729

centriole

cytosol

RAD50

CEP135

ZNF573

SNTB2

PPP1R12B

RB1CC1

LPIN1

BDP1

PSMD11

SCLKT, RNA

(KIAA1377, RNA)

(RFX3, RNA)

(1DPCP, RNA)

DNA binding, molecular function

(ZNF268B, RNA)

(ZFP14, RNA)

(KIAA0200, RNA)

(EPM2AIP1, RNA)

(SRRM1, RNA)

(CHOD, RNA)

(BDPI, RNA)

(PIAS2, RNA)

(RAD50, RNA)

(CHD9, RNA)

(ZNF292, RNA)

(1LCOR, RNA)

(HMBOX1, RNA)

(SPI150, RNA)

(ZNF605, RNA)

(BRCA1, RNA)

(ZNF673, RNA)

(ZNF629, RNA)

(SYCP2, RNA)

(ZBED3, RNA)

(RFX3, RNA)

(ZNF169, RNA)

(RNA-ASEK-C17orf49, RNA)

(ZNF563, RNA)

(ZNF573, RNA)

(ZNF514, RNA)

(HES1, RNA)

G0005829

(0.7890810^6, 0.000228605, True), (193, 3476, 47241, 34),

(cytosol, cellular_component)

(AHI1, RNA)

(AK9, RNA)

(AP153, RNA)

(ALB2, RNA)

(LP1N1, RNA)

(TUBGCP4, RNA)

(SRRM1, RNA)

(CUL5, RNA)

(CEP135, RNA)

(USP8, RNA)

(RBIC1, RNA)

(KPNA5, RNA)

(RAB5IP, RNA)

(MLLT4, RNA)

(ITSN1, RNA)

(SPRY1, RNA)

(PSMD11, RNA)

(SCLKT1, RNA)

(PLCB1, RNA)

(LGALS5, RNA)

(KIF38, RNA)

(EELA1, RNA)

(PPP1R12B, RNA)

(ASTM1, RNA)

(1SHRPA1B, RNA)

(CEP152, RNA)

(SMAD1, RNA)

(ZBED3, RNA)

(SGOL2, RNA)

(EVI5, RNA)

(INSR, RNA)

(UTS22, RNA)

(ATG10, RNA)

G0006351

(2.4515710^6, 0.000281931, True), (193, 2285, 47241, 26),

(transcription, DNA-templated biological_process)

(ZNF268B, RNA)

(ZFP14, RNA)

(LP1N1, RNA)

(CHD6, RNA)

(BDPI, RNA)

(PIAS2, RNA)

(RBIC1C1, RNA)

(CHD9, RNA)

(HELS, RNA)

(HMBOX1, RNA)

(SPI150, RNA)

(1JKF4, RNA)

(ZNF605, RNA)

(BRCA1, RNA)

(ZNF673, RNA)

(ZNF629, RNA)

(ZNF629, RNA)

(ZNF629, RNA)

(ZNF169, RNA)

(RSC1, RNA)

(ZNF563, RNA)

(ZNF573, RNA)

(ZNF514, RNA)

(HES1, RNA)

(ZNF503, RNA)

G00043234

(0.0000174221, 0.0018214, True), (193, 355, 47241, 9),

(protein complex, cellular_component)

(SNTB2, RNA)

(STRN3, RNA)

(1SPATS2L, RNA)

(BRCA1, RNA)

(RNF165, RNA)

(SMAD1, RNA)

(MREG, RNA)

(PRKCDBP, RNA)

(SASH1, RNA)

G0000729

(0.0000294668, 0.00282333, True), (193, 15, 47241, 3),

(DNA double-strand break processing, biological_process)

(RAD50, RNA)

(BRCA1, RNA)

(BARD1, RNA)

G0008022

(0.000147416, 0.0118384, True), (193, 192, 47241, 6),

(protein C-terminus binding, molecular_function)

(CEP135, RNA)

(MLLT4, RNA)

(SCLKT1, RNA)

(1SYN2BP, RNA)

(1AXN2, RNA)

(HES1, RNA)

G00080732

(0.000162862, 0.0118384, True), (193, 26, 47241, 3),

(strand displacement, biological_process)

(RAD50, RNA)

(BRCA1, RNA)

(BARD1, RNA)

G00007099

(0.000182669, 0.0123357, True), (193, 27, 47241, 3),

(centriole replication, biological_process)

(ZASS6, RNA)

(CEP135, RNA)

(CEP152, RNA)

G00051298

(0.000239975, 0.0123936, True), (193, 28, 47241, 3),

(centrosome duplication, biological_process)

(ZASS6, RNA)

(TUBGCP4, RNA)

(CEP152, RNA)

G0044822

(0.000204764, 0.0123936, True), (193, 1108, 47241, 14),

(polyA) RNA binding, molecular_function

(CELF2, RNA)

(IBA57, RNA)

(KIAA0020, RNA)

(SNTB2, RNA)

(SRRM1, RNA)

(1PPRF4B, RNA)

(1SPATS2L, RNA)

(DHDX9, RNA)

(DHDX3, RNA)

(ATTX2, RNA)

(HELZ, RNA)

(CPEB3, RNA)

(ZFC3H1, RNA)

(USP36, RNA)

G0045944

(0.000251161, 0.0126549, True), (193, 955, 47241, 13),

(positive regulation of transcription from RNA polymerase II promoter, biological_process)

(AHI1, RNA)

(LP1N1, RNA)

(PIAS2, RNA)

(STRN3, RNA)

(ZNF292, RNA)

(NFIA, RNA)

(1JKF4, RNA)

(BRCA1, RNA)

(SMAD1, RNA)

(ZBED3, RNA)

(1PPARGC1B, RNA)

(RFX3, RNA)

(HES1, RNA)

G00466827

(0.000253998, 0.0126549, True), (193, 3010, 47241, 26),

(metal ion binding, molecular_function)

(GDPD1, RNA)

(DGKE, RNA)

(ZNF268B, RNA)

(1PD2, RNA)

(ZFP14, RNA)

(RAD50, RNA)

(NEK1, RNA)

(ZNF292, RNA)

(HELZ, RNA)

(1PD18, RNA)

(1JKF4, RNA)

(FG60, RNA)

(ZNF605, RNA)

(ZNF673, RNA)

(ZNF682, RNA)

(ZNF629, RNA)

(1CCRN4, RNA)

(SMAD1, RNA)

(ZBED3, RNA)

(ZNF169, RNA)

(ZNF563, RNA)

(ZNF573, RNA)

(ZNF514, RNA)

(ZNF503, RNA)

(ZFC3H1, RNA)

G0006974

(0.000373653, 0.0158252, True), (193, 224, 47241, 6),

(cellular response to DNA damage stimulus, biological_process)

(RAD50, RNA)

(TAOK3, RNA)

(TLK2, RNA)

(BRCA1, RNA)

(BARD1, RNA)

(NUGGC, RNA)

G00032574

(0.000365926, 0.0161852, True), (193, 34, 47241, 3),

(response to testosterone, biological_process)

(BGALP, RNA)

(INSR, RNA)

(UTS22, RNA)

G0000731

(0.000399031, 0.0169585, True), (193, 35, 47241, 3),

(DNA synthesis involved in DNA repair, biological_process)
We can export the reports, for example to the $UserDocumentDirectory:

In[45]:=
EnrichmentReportExport[goAnalysisRNA,
  OutputDirectory -> $UserDocumentsDirectory, AppendString -> "GOAnalysisisRNA"];

We carry out our KEGG: Kyoto Encyclopedia of Genes and Genomes pathway analysis using KEGGAnalysis for all the classes and groups/subgroups. We only report terms for which there are at least 2 members. Please note that this is a time consuming computation.

In[46]:= keggAnalysisRNA = KEGGAnalysis[rnaClusters, ReportFilter -> 2];

The output of KEGGAnalysis has enrichments for each class and group

In[47]:= Query[Keys]@keggAnalysisRNA
Out[47]= {SpikeMax, SpikeMin, f1, f2, f3, f4, f5, f6, f7}
We can export the results, for example to the UserDocumentDirectory:

```
In[49]:=
EnrichmentReportExport[keggAnalysisRNA, OutputDirectory \to UserDocumentsDirectory, AppendString \to "KEGGAnalysisRNA"]
```

We can view results for any of the groups (and also check out the behavior using the heatmaps generated in the previous section

```
In[52]:=
Query["SpikeMin", "G1S1"] @ keggAnalysisRNA
```

```
{
 path: hsa04064 \to \{ 0.25958 \times 10^{-8}, 3.50829 \times 10^{-7}, True \},
\{ 1521, 93, 7086, 44 \}, \{ NF-kappa B signaling pathway - Homo sapiens (human), \{ [PRKCB, RNA], [BCL2L1, RNA], [PRKCG, RNA], [MAP3K7, RNA], [PLCG1, RNA], [TAB2, RNA], [TRAF6, RNA], [CFLAR, RNA], [MAP3K14, RNA], [IKKB, RNA], [PARK1, RNA], [BCL2, RNA], [RIPK1, RNA], [MALT1, RNA], [ICAM1, RNA], [TRAF3, RNA], [IRAK1, RNA], [TIRAP, RNA], [CSNK2A1, RNA], [BT2K, RNA], [TAB3, RNA], [PIAS4, RNA], [CD40LG, RNA], [DDX58, RNA], [TICAM2, RNA], [CHUK, RNA], [BIRC2, RNA], [TRAF2, RNA], [ZAP70, RNA], [BCL2, RNA], [CCL4, RNA], [RELB, RNA], [TRADD, RNA], [CSNK2A2, RNA], [TAB1, RNA], [CARD11, RNA], [LCK, RNA], [PLCG2, RNA], [RELA, RNA], [TNFAIP3, RNA], [TLR4, RNA], [SYK, RNA], [LYN, RNA], [MYD88, RNA] \} \}
```

We can visualize any KEGG pathway using KEGGPathwayVisual, getting (1) a link to the website, (2) importing the figure (3) importing the figure with highlighted annotations, (4) importing a series of figures with intensities corresponding to each time point, (5) export a series of figures with time intensities as a movie (animation).

```mathematica
In[20] := (*1*) KEGGPathwayVisual["path:hsa04064"]

In[21] := (*2*) KEGGPathwayVisual["path:hsa04064", ResultsFormat \[Rule] "Figure"]

In[62] := (*3*) KEGGPathwayVisual["path:hsa04064", ResultsFormat \[Rule] "Figure", MemberSet \[Rule] pathwaymembers]

In[63] := (*4*) nfkBPathwayFigureList = KEGGPathwayVisual["path:hsa04064", ResultsFormat \[Rule] "Figure", MemberSet \[Rule] pathwaymembers, Intensities \[Rule] Query[Key] & pathwaymembers]rnaFinalTimeSeries
\textbf{Out[63]=}  

\textbf{In[64]:=} \texttt{ListAnimate[nfkbPathwayFigureList["Results"], ImageSize \to \text{Automatic}]}
Appendix: All Commands Up to Enrichment Analysis in One Step

As a summary, we list here all the commands up to the enrichment analysis is one step:

```plaintext
In[65]:= (*5*)KEGGPathwayVisual["path:hsa04064", ResultsFormat -> "Movie", MemberSet -> pathwaymembers, Intensities -> Query[Key[#] & /@ pathwaymembers] @ rnaFinalTimeSeries]

Out[65]= Pathway -> path:hsa04664, Results -> path_hsa04664.mov]
```
<< MathIOmica;

rnaxample = Get[FileNameJoin[ConstantMathIOmicaExamplesDirectory, "rnaxample"]];

rnaQuantileNormed = QuantileNormalization[rnaLongitudinal, ListIndex -> 1, ComponentIndex -> 1];

rnaZeroTagged = LowValueTag[rnaQuantileNormed, 0];

dginialTimeSeries = ConstantSeriesClean[normedRNACompared];

+Bootstrap+
rnaBootstrap = BootstrapGeneral[rnaLongitudinal, 100000];

+1+ rnaBootstrapQuantileNormed = QuantileNormalization[rnaBootstrap, ListIndex -> 1, ComponentIndex -> 1];

+2+ rnaBootstrapZeroTagged = LowValueTag[rnaBootstrapQuantileNormed, 0];

+3+ rnaBootstrapNoiseAdjusted = LowValueTag[rnaBootstrapZeroTagged, 1, ValueReplacement -> 1];

+4+ rnaBootstrapFiltered = FilterMissing[rnaBootstrapNoiseAdjusted, 3/4, Reference -> "255", ShowPlots -> False];

+5+ timesRNA = TimeExtractor[rnaFiltered];

timesRNA = CreateTimeSeries[rnaFiltered];

timesRNABootstrapRNA = CreateTimeSeries[rnaBootstrapFiltered];

timesRNABootstrapRNALog = SeriesApplier[Log, timesRNA];

+6+ normedRNACompared = SeriesApplier[Normalize, rnaCompared];

+7+ rnaBootstrapCompared = SeriesInternalCompare[timesRNABootstrapRNALog, ComparisonIndex -> 2];

+8+ normedBootstrapRNALog = SeriesApplier[Log, timesRNABootstrapRNALog];

+9+ rnaBootstrapFinalTimeSeries = ConstantSeriesClean[normedBootstrapRNALog];

q95RNA = QuantileEstimator[rnaBootstrapFinalTimeSeries, timesRNA, Method -> "Spikes"];

rnclassification = TimeSeriesClassification[Q95FinalTimeSeries, timesRNA, LombScargleCutoff -> q95RNA, SpikeCutoffs -> q95RNA];

rnaxlasses = TimeSeriesClusters[rnclassification, PrintDendrograms -> True];

goAnalysisRNA = GOAnalysis[rnaxlasses, OntologyLengthFilter -> 3, ReportFilter -> 3];

keggAnalysisRNA = KEGGAnalysis[rnaxlasses, ReportFilter -> 2];


TimeSeriesDendrogramsHeatmaps[rnaxlasses]

Related Tutorials

- MathIOmica Overview
- MathIOmica Tutorial
- MathIOmica Guide