

Rendering pathways to convey quantitative genomic relationships

VJ Carey et al

October 27, 2023

1 Introduction

Given an R graph representing a biological pathway and a vector of numbers (e.g., estimated levels of gene expression, or quantile of gene expression value in a distribution over samples) linked to the nodes of the pathway (e.g., genes), we wish to display the graph with nodes colored to convey the relationships among the numbers.

Our primary tool for rendering graphs is *Rgraphviz*. This package uses AT&T graphviz to compute layouts, and various aspects of R graphics to create renderings.

Our primary tools for creating pathway graphs are the *graph* and *pathRender* packages.

In this vignette and associated code, we aim to simplify the use of software in these components to allow the intended renderings to be created in a flexible way.

2 An example

2.1 A pathway graph

The *graph* package contains a custom-made graph representing the pancreatic cancer initiation pathway. First we render it in isolation from data:

```
> library(graph)
> library(pathRender)
> library(Rgraphviz)
> data(pancrCaIni)
> plot(pancrCaIni, nodeAttrs=pwayRendAttrs(pancrCaIni))
```



Note that the default rendering of the pathway graph is hard to read; we use the new `pwayRenderAttrs` function to generate attributes that improve readability.

2.2 An ExpressionSet and its reduction

We will work with `ALL`.

```
> library(ALL)
> if (!exists("ALL")) data(ALL)
```

A basic problem is to reduce the information obtained using the whole-genome microarray to a set of numbers relevant to the pathway we wish to render. The `reduceES` function helps with this. Given a vector of annotation tokens (e.g., HUGO gene symbols) and a map from symbols to associated microarray probes, `reduceES` restricts the assay data to relevant probes. The map parameter can be either an `AtomicAnnDbBimap` as created in the `*.db` annotation packages, or a list with annotation tokens as element names and vectors probe identifiers as elements. Here we illustrate the use of the `Bimap`:

```
> if ("package:hgu95av2" %in% search()) detach("package:hgu95av2")
> library(hgu95av2.db)
> red1 = reduceES( ALL, nodes(pancrCaIni), revmap(hgu95av2SYMBOL), "symbol" )
> red1
```

```
ExpressionSet (storageMode: lockedEnvironment)
assayData: 30 features, 128 samples
  element names: exprs
protocolData: none
phenoData
  sampleNames: 01005 01010 ... LAL4 (128 total)
  varLabels: cod diagnosis ... date last seen (21 total)
  varMetadata: labelDescription
featureData
  featureNames: 1940_at 32159_at ... 34006_s_at (30 total)
  fvarLabels: symbol
  fvarMetadata: labelDescription
experimentData: use 'experimentData(object)'
pubMedIds: 14684422 16243790
Annotation: hgu95av2
```

```
> pData(featureData(red1))
```

| | symbol |
|------------|---------|
| 1940_at | KRAS |
| 32159_at | KRAS |
| 37901_at | PIK3R4 |
| 34254_at | RALGDS |
| 37543_at | ARHGEF6 |
| 40781_at | AKT3 |
| 1706_at | ARAF |
| 1707_g_at | ARAF |
| 1876_at | RALA |
| 1877_g_at | RALA |
| 39253_s_at | RALA |
| 2050_s_at | RAC1 |
| 40864_at | RAC1 |
| 33770_at | CHUK |
| 1861_at | BAD |
| 486_at | CASP9 |
| 487_g_at | CASP9 |
| 1130_at | MAP2K1 |
| 1844_s_at | MAP2K1 |

| | |
|------------|--------|
| 36628_at | RALBP1 |
| 177_at | PLD1 |
| 1377_at | NFKB1 |
| 1378_g_at | NFKB1 |
| 38438_at | NFKB1 |
| 1615_at | BCL2L1 |
| 34742_at | BCL2L1 |
| 976_s_at | MAPK1 |
| 2070_i_at | MAPK8 |
| 2071_s_at | MAPK8 |
| 34006_s_at | MAPK8 |

Note that the `reduceES` creates a `featureData` variable and that there are repetitions of values of this variable. We can specify that we want to collapse repetitions by specifying a function for the `collapseFun` parameter. We will use `mean`.

```
> collap1 = reduceES( ALL, nodes(pancrCaIni), revmap(hgu95av2SYMBOL), "symbol", mean )
> collap1
```

```
ExpressionSet (storageMode: lockedEnvironment)
assayData: 18 features, 128 samples
  element names: exprs
protocolData: none
phenoData
  sampleNames: 01005 01010 ... LAL4 (128 total)
  varLabels: cod diagnosis ... date last seen (21 total)
  varMetadata: labelDescription
featureData
  featureNames: AKT3 ARAF ... RALGDS (18 total)
  fvarLabels: symbol
  fvarMetadata: labelDescription
experimentData: use 'experimentData(object)'
Annotation:
```

2.3 A rendering

Now we will render information on one sample from the reduced data.

```
> library(RColorBrewer)
> plotExGraph(pancrCaIni, collap1, 1)
```

