Using analysisPipeline

Fraser J. Sim

August 14, 2013

1 Overview

This package is a combination of several 'wrapper' functions that enable complex analysis and visualization of gene expression without excessive re-coding of repetitive functions.

The package was originally designed based of data from Affymetrix GeneChips and Illumina BeadArrays.

This package extends a series of concepts introduced in *affycoretools*, *ROC*, *PGSEA*, and others.

2 Introduction

The *analysisPipeline* package was written to facilitate the easy, rapid and complete analysis of microarray data. Analysis has been performed with a variety of datasets. The only requirement being that data is loaded into an ExpressionSet object and that an annotation package is available and installed.

3 Initial data loading

For the vignette examples, the primary data was generated and published in **?**. This is available for download from NCBI geo at `http://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE26535`.

The experiment is a simple comparison of human glial progenitors sorted on the basis of A2B5 antigen expression (labelled **A2B5**) against unsorted human white matter dissociated cells (labelled **uns**).

Gene expression analysis was perfomed using Affymertrix U95av2 chips. Affymetrix microarray data is usally loaded usually using *affy* or an equivalent.

The global variable chipAnnotation is used by numerous functions in the *analyisPipeline* package. This variable should be set immediately after loading array data.

```
> library(affy)
> eset.complete <- justRMA()
> # setup chipAnnotation global variable
> chipAnnotation <- annotation(eset.complete)
> # export pData for editing outside R
> write.csv(pData(eset.complete), file = 'pData.csv')
```

Following initial loading and pre-processing of data, we generate and then edit a simple text file "pData.csv" file to further annotate each array, and specify sample groupings and other metadata. We use the phenotype column to specify default groupings used throughout the *analysisPipeline* package. This file is edited in an external program such as Microsoft Excel and returned to R.

```
> pData(eset.complete) <- read.csv('pData.csv')
> sampleNames(eset.complete) <- pData(eset.complete)$sampleNames
```

In our typical Affymetrix pipeline, we use *farms* to remove the non-informative probe sets from the Expression Set object. *farms* is a unsupervised filtering technique that utilizes the expression value of all probes in a probe set across all experimental samples to determine whether a probe set contains usefull data **?**. In the code below, first the distribution of informative and non-informative probes are plotted and then the ExpressionSet object is filtered.

```
> library(farms)
> affyBatch <- ReadAffy()
> eset.farms <- qFarms(affyBatch)
> INIs = INIcalls(eset.farms)
> All_probes = featureNames(eset.farms)
> I_probes = getI_ProbeSets(INIs)
> plot(INIs)
> title("Informative/Noninformative ProbeSet distribution")
> legend("topleft",paste("uninformative\n",
+  sprintf("%.0f%%",(1-length(I_probes)/length(All_probes))*100)),
+  ,bty = "n")
> legend("topright",paste("informative\n",
+  sprintf("%.0f%%",length(I_probes)/length(All_probes)*100)),
+  ,bty = "n")
> eset <- eset.complete[I_probes,]
```

To define groups within the experiment, we use the ‘phenotype’ column. In this example, there are two groups **A2B5** and **uns**.

```
> pData(eset)
```

```
                         X sample patient phenotype sampleNames
1     Patient1_A2B5WM.CEL      1       A      A2B5      A2B5_A
2 Patient1_UnsortedWM.CEL      2       A       uns       uns_A
3     Patient2_A2B5WM.CEL      3       B      A2B5      A2B5_B
4 Patient2_UnsortedWM.CEL      4       B       uns       uns_B
5     Patient3_A2B5WM.CEL      5       C      A2B5      A2B5_C
6 Patient3_UnsortedWM.CEL      6       C       uns       uns_C
```

In this example, we now load the pre-normalized and filtered data using ’data(eset)’.

```
> data(eset)
> chipAnnotation <- annotation(eset)
```

Data exploration provides a useful way of assessing the overall data structure and to determine whether their are any outlying samples. This can be achieved in *analysisPipeline* using two seperate approaches, `plot3dPCA` and `customTree`.

- `plot3dPCA` displays a 3d scatter plot of the principle components of the data (Figure **??**). Each sample is represented by a single point and, by default, uses the `eset$phenotype` factor to determine groups and coloring. The 3d plot is also diplayed in both *rgl* window which can be rotated to faciliate visualization. (Note: if `plot3dPCA` is called a second time, while the Rpackagergl window is open, the R-plot is appropriately rotated so that a PDF or other file may be exported.

- `customTree` displays the hierarchical clustering of samples with samples colored by groupings present in `eset$phenotype` (Figure **??**). Euclidean sample-sample distances are calculated between samples in the `ExpressionSet` object.

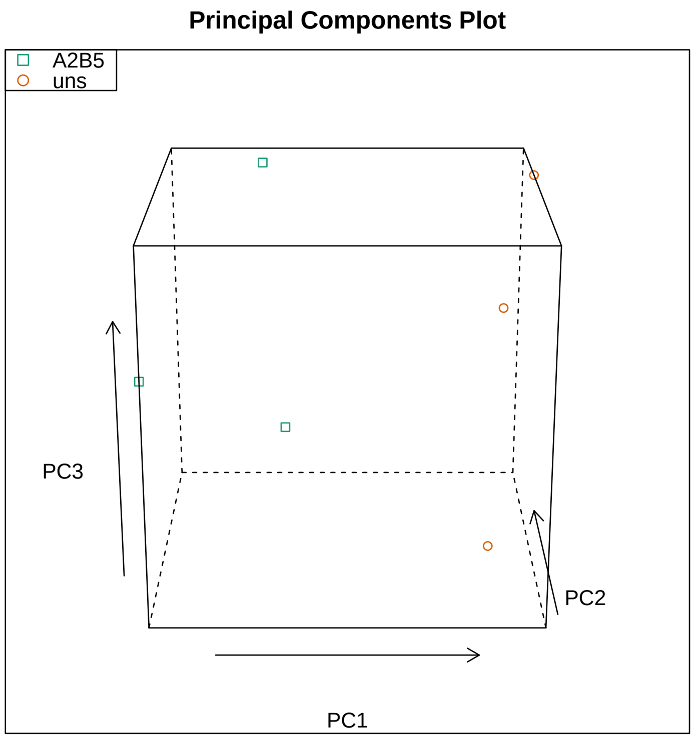

Figure 1: 3d PCA. This figure shows a 3 dimensional principle components plot of the individual RNA/array samples.

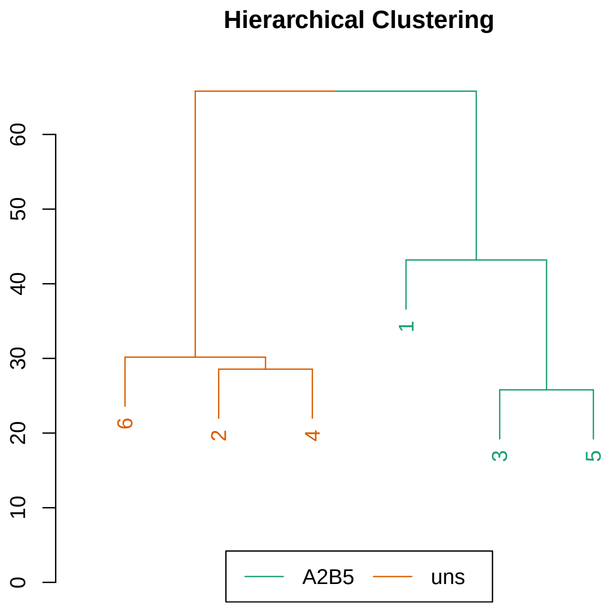

Figure 2: Hierarchical clustering. This figure shows a colored clustering dendrogram generated from sample-sample distances.

4 Computing Differential Expression

Although various packages exist for determination of differentially expressed genes, our pipeline utilizes the package *limma*. This package is flexible and can analyze various experimental designs from simple to complex multivariate analyses. A detailed explaination of *limma* is beyond the scope of this vignette, please see the "LIMMA User's guide".

In this example, a simple model is setup whereby the grouping defined by `eset$phenotype` is used. In this experiment, with two groups, only one comparison or contrast can be defined - "A2B5 vs. unsorted cells".

```
> library(limma)
> design = model.matrix(~-1 + eset$phenotype)
> colnames(design) <- levels(eset$phenotype)
> fit <- lmFit(eset, design)
> # contrast.matrix may be set up using multiple contrasts
> contrast.matrix <- makeContrasts(A2B5 - uns, levels = design)
> fit2 <- contrasts.fit(fit, contrast.matrix)
> fit3 <- eBayes(fit2)
```

One of the challenges of differential gene expression analysis is to choose appropriate statistical and fold-change cut-off values to select genes-of-interest. Using the `decideCutoffs` function (Figure **??**), the user can visualize the effects of various p-value cutoffs and the effect of applying false discovery rate (FDR)-based corrections **?**.

Having selected appropriate cut-offs, the results can be parsed into a list object known as `contrast.details`. To do this, we use the `parseContrasts` function.

```
> contrast.details <- parseContrasts(fit3, FCcutoff = 2,
+  FDRcutoff = 0.05, adjust = "fdr")
```

This function generates comma-delimited text files in the working directory and populates the contrast.details list variable for subsequent pathway analysis (see below).

We can also generate a series of heatmaps from each comparison. An example is shown in Figure **??**.

```
> generateHeatMaps(eset)
```

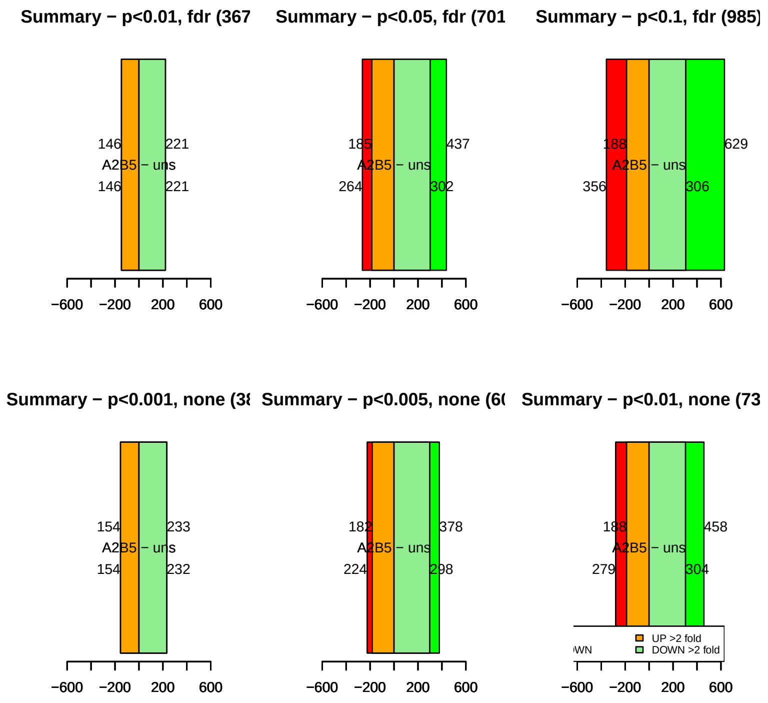

Figure 3: decideCutoffs. These bar graphs help selection of p-value and FC cutoffs

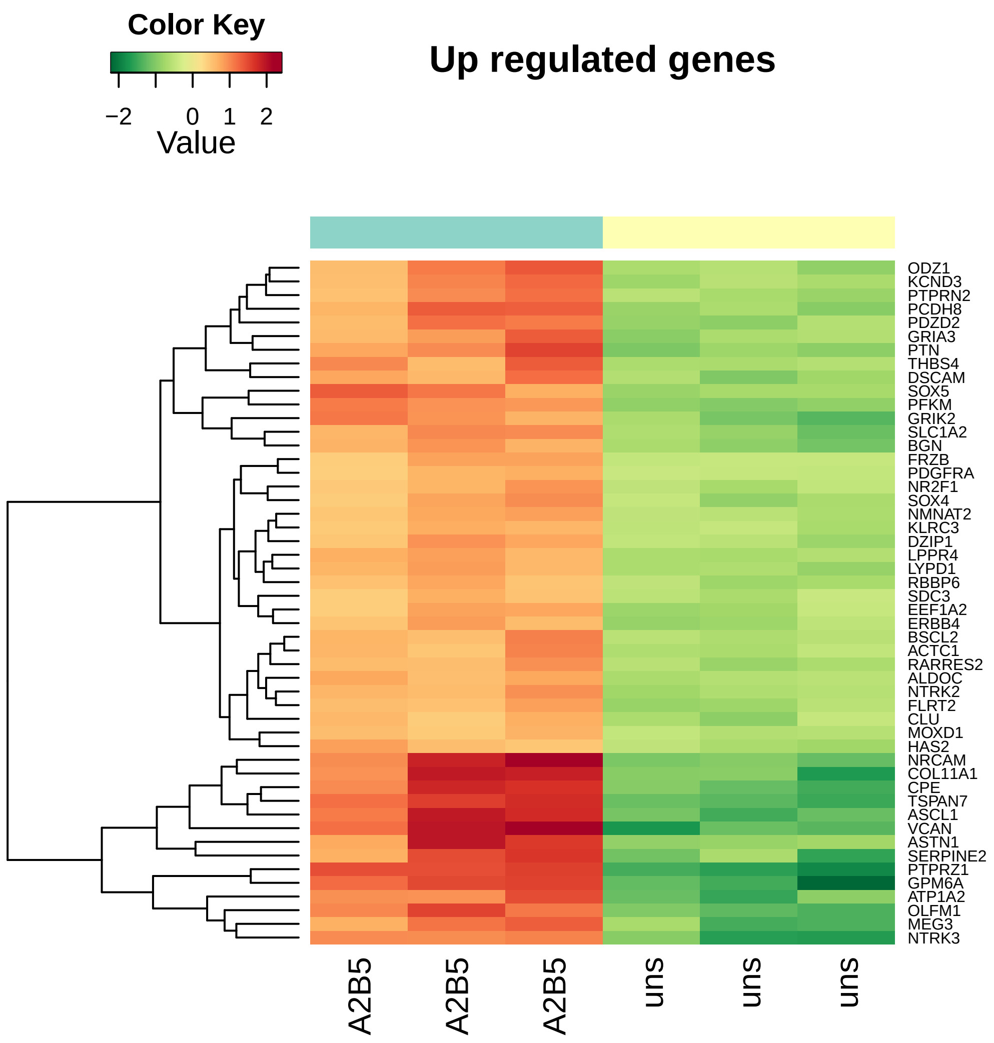

Figure 4: Heatmap of top 50 up-regulated genes

5 Pathway Analysis

In this package, pathway analysis may be performed in an automated process or manually following selection of custom probe lists.

5.1 Gene Ontology Summary

As our analyses have primarily been interested in genes involved in cell-cell signaling and cell fate regulation, the `runAutomatedGOSummary` function was developed to parse the gene lists into various function categories using Gene Ontology (GO) annotations.

```
> runAutomatedGOSummary(contrast.details)

A2B5 - uns
132 genes total.
9 ligands.
22 receptors.
15 transcription factors.
11 ECM molecules.
40 enzymes.
49 unannotated genes.
Done UP genes.
211 genes total.
26 ligands.
25 receptors.
20 transcription factors.
5 ECM molecules.
52 enzymes.
105 unannotated genes.
Done DOWN genes
```

The results are exported to comma-delimited text files. An excerpt is show in Table **??**.

5.2 Hypergeometric testing

Over-representation analysis of Gene Ontology and KEGG databases can be performed using `runAutomatedHyperG`. This performs hypergeometric testing of all gene lists using the elim alogrithm from the *topGO* package (p

< 0.01), and by hypergeometric testing using KEGG database pathways (p < 0.05).

The results are exported to comma-delimited text files for offline viewing (csv files).

```
> contrast.HyperG <- runAutomatedHyperG(eset, contrast.details)
```

In this analysis, 26 GO Biological Process Terms were significantly over-represented in A2B5-sorted cells (see Table ?? for a partial list).

To further examine the enrichment of genes in a given GO Term across the enture gene expression profile, ROC analysis was performed on the one of the top 10 terms - "central nervous system development". See (Figure ??).

```
> gs.EG <- unlist(mget("GO:0007417", org.Hs.egGO2ALLEGS))
> ROCresults <- plotROC(gs.EG, fit3, coef = 1,
+   main = "Enrichment of CNS development genes\n(GO:0007417)")
```

The function `runAutomatedHyperG` performs hypergeometric testing on all KEGG pathways. The results are similarly exported to csv files. An example of KEGG pathway analysis results is shown in Table ??.

The top KEGG pathway - "Neuroactive ligand-receptor interaction pathway" - was plotted on a heatmap to further investigate which genes were significantly regulated (Figure ??).

These approaches can be easily applied to much more complex data sets using the same simple functions `runAutomatedGOSummary` and `runAutomatedHyperG`.

6 Gene set enrichment analysis

In addition to over-representation analysis on differentially expressed genes, the *analysisPipeline* package also encorporates parametric gene set enrichment analysis (PGSEA), as described in ? and provided as an R package in *PGSEA*.

This function automates the process of performing parametric gene set enrichment (PGSEA) on a number of bioinformatics databases. Using `PGSEA` as the basis for gene set enrichment and *limma* to calculate significance based on the experimental design specified in `design`. (Note: See documentation of `run.GSEA` for details on individual GSEA analysis). The following GeneSetCollections are analyzed in turn to identify enriched/depleted GeneSets:

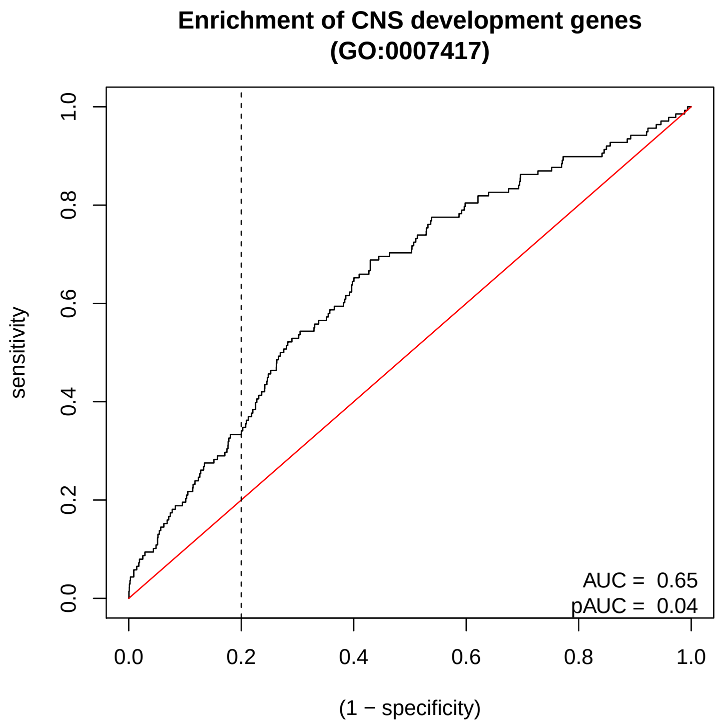

Figure 5: ROC curve of GO term enrichment.

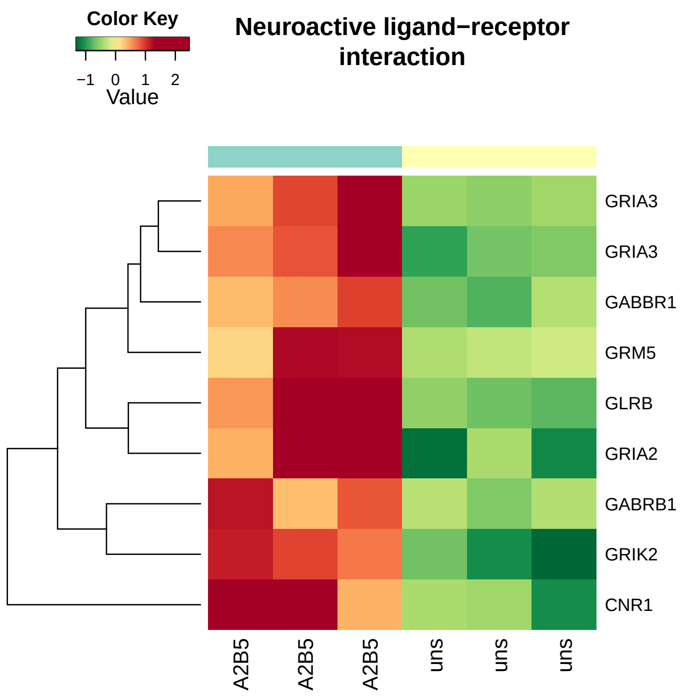

Figure 6: KEGG pathway. Only significantly up-regulated genes are shown.

1. bioCarta - data parsed from `www.bioCarta.com`
2. broadC2
3. broadC3
4. GO
5. KEGG

BroadC2 and C3 databases refer to the mSigDb maintained at the Broad Institute and available for download from `http://www.broadinstitute.org/gsea/msigbdb/index.jsp`. The database was originally published in **?**.

Using the global variable `design` which contains a *limma* design matrix. Limma is performed to identify enrichent GeneSets. Significantly regulated GeneSets are returned and a heatmap of relative enrichment of significant genesets is generated and saved as a PDF file.

This can be called using the wrapper function `runAutomatedGSEA`.

```
> runAutomatedGSEA(selCoef = "A2B5 - uns",
+                  refSample = "uns", blnPrompt = TRUE)
```

For brevity in this vignette, only bioCarta GSEA analysis is performed.

```
Biocarta GSEA analysis running...

20 significant pathways found.
```

Following completion of GSEA, a new directory is created which contains all of the results files generated during the analysis. For all GSEAs, a heatmap is produced using *PGSEA* to visualize significantly regulated gene sets in each database. For example, the bioCarta GSEA heatmap is shown in Figure **??**.

In addition for bioCarta and KEGG pathways, a graphical representation of enriched gene sets is generated using *Rgraphviz*. For example the most significant bioCarta pathway is shown as a graph in Figure **??**, and as a gene-expression heatmap (Figure **??**).

For Broad mSigDB-based GSEA databases, the csv results files are annotated with additional details regarding each data set.

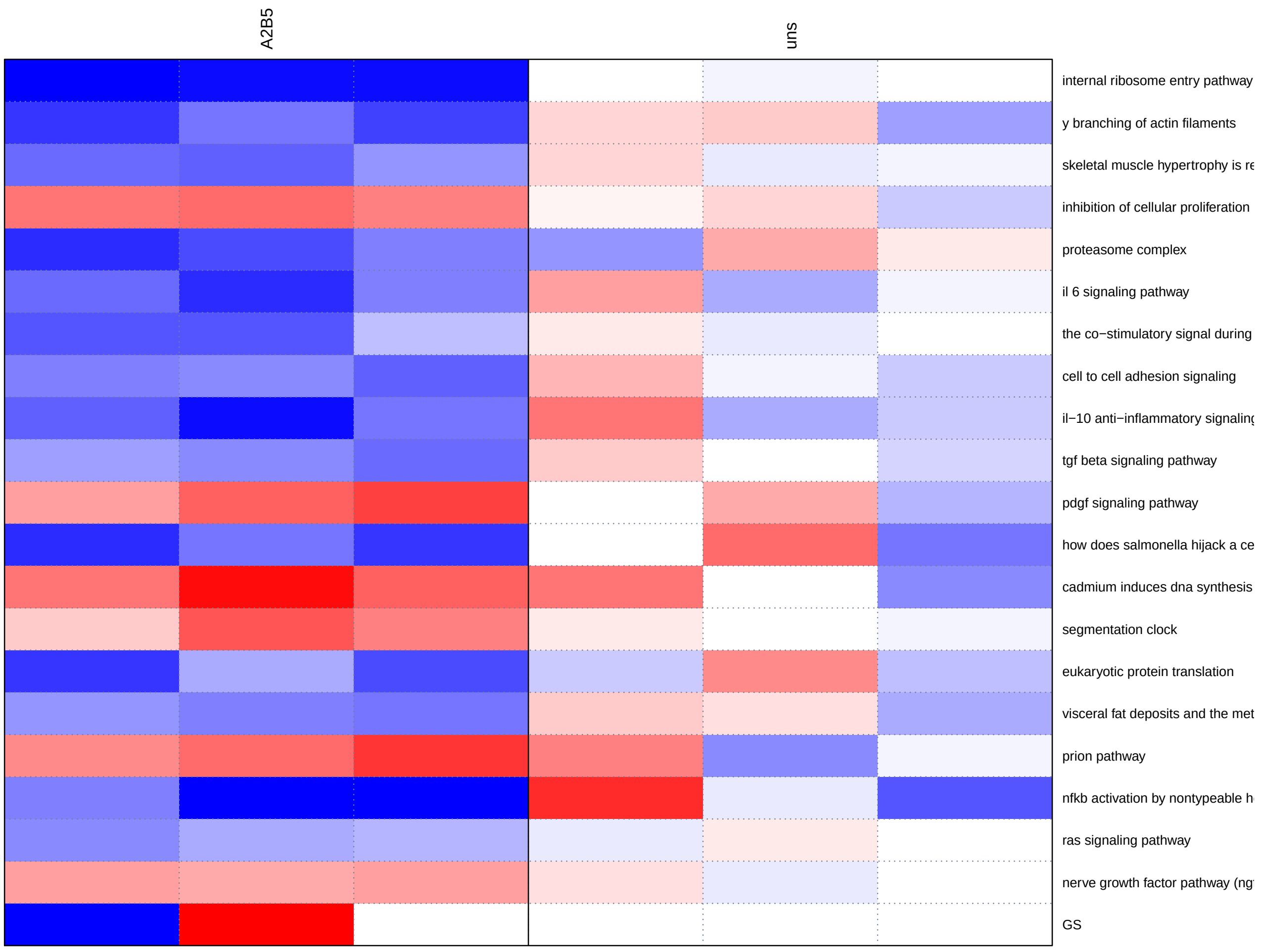

Figure 7: bioCarta GSEA. This heatmap shows the relative enrichment and depletion of bioCarta pathways in each sample (enriched in red, depleted in blue). Only pathways significantly regulated are shown in this heatmap.

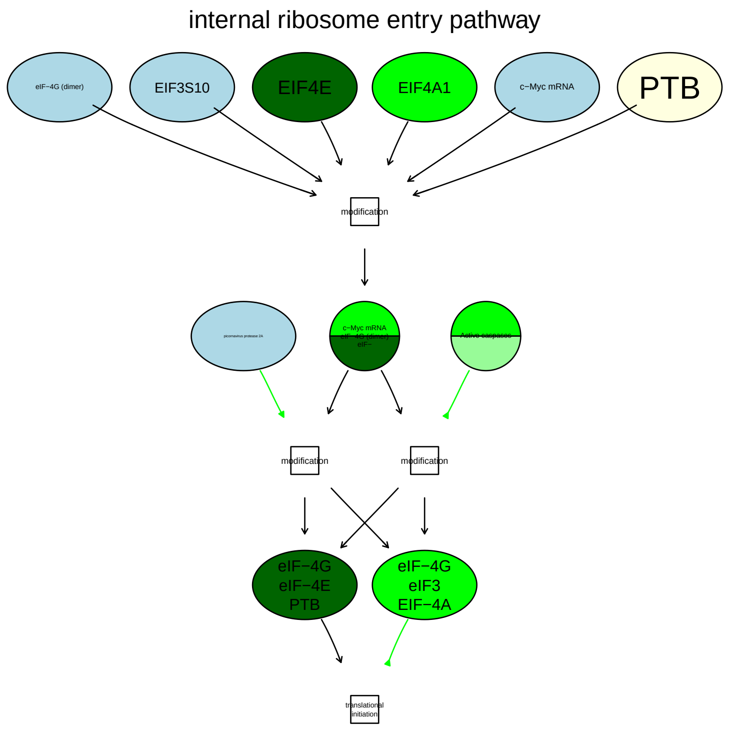

Figure 8: bioCartaPathway. Selected bioCarta pathways can be visualized using bioCarta.plot. The relative expression of individual genes in nodes is shown by the fill color; green representing down-regulation and red up-regulation in the profile of interest.

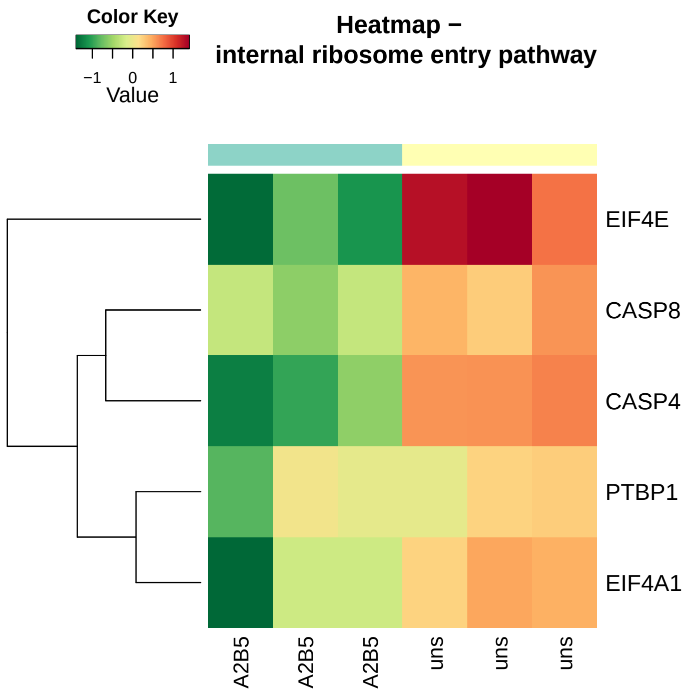

Figure 9: bioCartaHeatmap

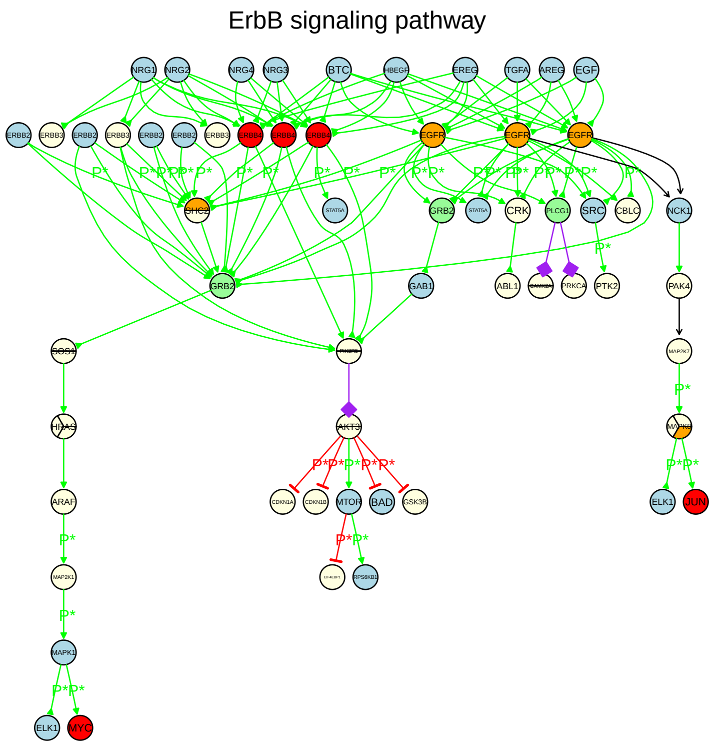

Figure 10: KEGGplot. This figure shows a graphical representation of the KEGG - Hedgehog signaling pathway.

7 Other functions

7.1 KEGG pathway visualization

To assist in the visualization of KEGG pathways, the *analysisPipeline* package contains the `KEGG.plot` function which generates a graph representing a selected KEGG pathway. If the *ratios* argument is specified then nodes representing genes or groups of genes are colored according to gene expression levels.

An example is shown in Figure **??**.

```
> ratios <- fit3$coefficients[,1]
> results <- KEGG.plot("04012",ratios,chipAnnotation)
```

7.2 Homology mapping functions

To facilitate cross-species analysis, there are several functions available to convert between NCBI Entrez Gene identifiers.

Function	From	To
`convertMouseToHumanGeneID.R`	Mouse	Human
`convertHumanToMouseGeneID.R`	Human	Mouse
`convertRatToHumanGeneID.R`	Rat	Human
`convertHumanToRatGeneID.R`	Human	Rat
`convertBovineToHumanGeneID.R`	Bovine	Human
`convertHumanToBovineGeneID.R`	Human	Bovine

These functions by default use Ensembl and Bioconductor homology packages. To enable *annotationTools* package functionality **?**, see examples in the help page for `setupFindHomologs`.

In the follow example, we ask whether mouse oligodendrocyte progenitor expressed genes are enriched in the human A2B5$^+$ cell vs. Unsorted cell profiles. First we load the list of probe sets identified in **?** as oligodendrocyte progenitor expressed.

```
> data(mouseOPC)
> mouseOPC[1:5]

[1] "1421917_at"   "1418310_a_at" "1440884_s_at" "1450251_a_at"
[5] "1423341_at"
```

`mouseOPC` contains a character vector of significant probe sets for the Affymetrix mouse 430 2.0 microarray. We next annotate these probe sets to mouse NCBI Entrez GeneIDs, remove unannotated genes and then use `convertMouseToHumanGeneID` to easily convert these to their human homologs.

```
> library(mouse4302.db)
> mouseOPC.MmEG <- unlist(mget(mouseOPC, mouse4302ENTREZID,
+                              ifnotfound = NA))
> mouseOPC.MmEG <- mouseOPC.MmEG[!is.na(mouseOPC.MmEG)]
> # convert to human annotation
> mouseOPC.HsEG <- convertMouseToHumanGeneID(mouseOPC.MmEG)

Converting mouse geneIds to human....done. 94 % annotated.
```

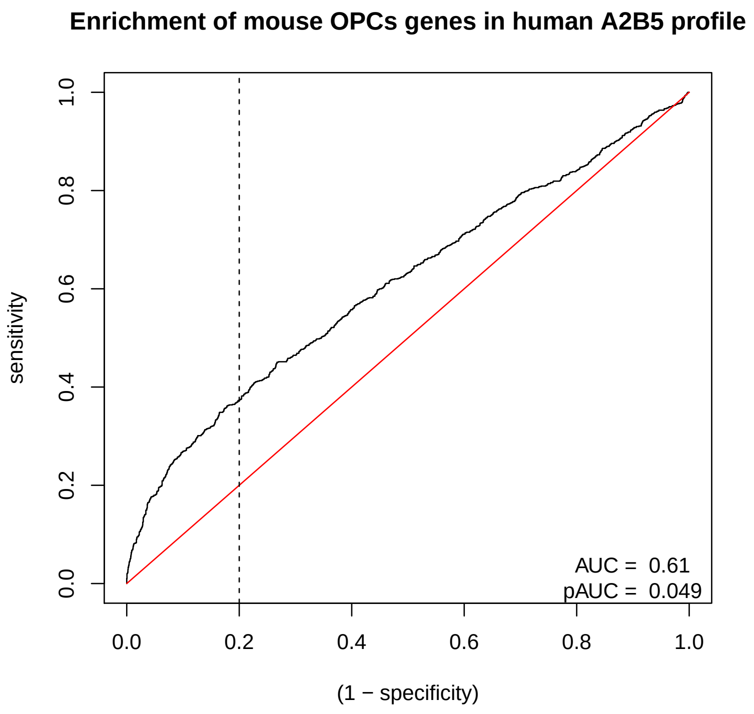

Figure 11: mouse OPCs gene enrichment in human profile

Using `plotROC`, the relative enrichment of these mouse oligodendrocyte progenitor genes can be visualized (Figure **??**).

```
> ROCresults <- plotROC(mouseOPC.HsEG, fit3, coef = 1,
+  main = "Enrichment of mouse OPCs genes in human A2B5 profile")
```

8 Conclusion

The *analysisPipeline* package provides a unified workflow for the analysis of gene expression data from data exploration through to complex pathway analysis and visualization in an accesible form. The package was designed to be flexible for complex experiments and easy to learn to enable complex analysis by relative beginners to R/Bioconductor.

Type	Symbol	Description	GeneID	logFC	adj.P.Val
Ligands	PTN	pleiotrophin	5764	2.01	0.00
Ligands	THBS4	thrombospondin 4	7060	1.63	0.00
Ligands	SERPINE2	serpin peptidase inhibitor, clade E (nexin, p	5270	2.40	0.01
Ligands	PDGFRA	platelet-derived growth factor receptor, alph	5156	1.03	0.01
Ligands	SCG2	secretogranin II	7857	2.98	0.01
Ligands	CSPG5	chondroitin sulfate proteoglycan 5 (neuroglyc	10675	2.30	0.01
Ligands	TRAK1	trafficking protein, kinesin binding 1	22906	1.40	0.01
Ligands	CHGB	chromogranin B (secretogranin 1)	1114	1.29	0.02
Ligands	DOK5	docking protein 5	55816	1.39	0.04
Receptors	PTPRZ1	protein tyrosine phosphatase, receptor-type,	5803	3.13	0.00
Receptors	NTRK3	neurotrophic tyrosine kinase, receptor, type	4916	2.42	0.00
Receptors	GRIK2	glutamate receptor, ionotropic, kainate 2	2898	1.94	0.00
Receptors	NTRK2	neurotrophic tyrosine kinase, receptor, type	4915	1.39	0.00
Receptors	ERBB4	v-erb-a erythroblastic leukemia viral oncogen	2066	1.36	0.00
Receptors	PTPRN2	protein tyrosine phosphatase, receptor type,	5799	1.59	0.00
Receptors	KLRC3	killer cell lectin-like receptor subfamily C,	3823	1.16	0.00
Receptors	GRIA3	glutamate receptor, ionotrophic, AMPA 3	2892	1.67	0.01
Receptors	NR2F1	nuclear receptor subfamily 2, group F, member	7025	1.24	0.01
Receptors	PDGFRA	platelet-derived growth factor receptor, alph	5156	1.03	0.01
Receptors	IL1RAP	interleukin 1 receptor accessory protein	3556	1.00	0.01
Receptors	GABBR1	gamma-aminobutyric acid (GABA) B receptor, 1	2550	1.24	0.01
Receptors	GPR19	G protein-coupled receptor 19	2842	1.01	0.01
Receptors	GLRB	glycine receptor, beta	2743	2.04	0.01
Receptors	GRIA2	glutamate receptor, ionotropic, AMPA 2	2891	2.05	0.01
Receptors	LDLR	low density lipoprotein receptor	3949	1.69	0.02
Receptors	LPHN3	latrophilin 3	23284	1.18	0.02
Receptors	ABCC8	ATP-binding cassette, sub-family C (CFTR/MRP)	6833	1.03	0.02
Receptors	GABRB1	gamma-aminobutyric acid (GABA) A receptor, be	2560	1.22	0.02
Receptors	CNR1	cannabinoid receptor 1 (brain)	1268	2.24	0.04
Receptors	GRM5	glutamate receptor, metabotropic 5	2915	1.20	0.05
Receptors	SEMA5A	sema domain, seven thrombospondin repeats (ty	9037	1.17	0.05
Transcription Factors	ASCL1	achaete-scute complex homolog 1 (Drosophila)	429	2.85	0.00
Transcription Factors	SOX5	SRY (sex determining region Y)-box 5	6660	1.82	0.00
Transcription Factors	NR2F1	nuclear receptor subfamily 2, group F, member	7025	1.24	0.01
Transcription Factors	SOX4	SRY (sex determining region Y)-box 4	6659	1.40	0.01
Transcription Factors	SATB1	SATB homeobox 1	6304	1.43	0.01
Transcription Factors	JUND	jun D proto-oncogene	3727	1.03	0.01
Transcription Factors	JUN	jun proto-oncogene	3725	1.73	0.01
Transcription Factors	TRAK1	trafficking protein, kinesin binding 1	22906	1.40	0.01
Transcription Factors	CREB5	cAMP responsive element binding protein 5	9586	1.11	0.01

Table 1: GOSummary

GO.ID	Term	Annotated	Significant	Expected	p.value
GO:0001764	neuron migration	17	5	0.67	0.00
GO:0061102	stomach neuroendocrine cell differentiat...	2	2	0.08	0.00
GO:0033602	negative regulation of dopamine secretio...	2	2	0.08	0.00
GO:0050805	negative regulation of synaptic transmis...	7	3	0.27	0.00
GO:0045471	response to ethanol	25	5	0.98	0.00
GO:0007417	central nervous system development	138	13	5.41	0.00
GO:0030154	cell differentiation	488	42	19.12	0.00
GO:0048675	axon extension	16	4	0.63	0.00
GO:0007612	learning	16	4	0.63	0.00
GO:0051289	protein homotetramerization	8	3	0.31	0.00

Table 2: topGO results

KEGGID	Pvalue	OddsRatio	ExpCount	Count	Size	Term
4080	0.00	7.53	1.47	8	40	Neuroactive ligand-receptor interaction
4940	0.02	5.90	0.62	3	17	Type I diabetes mellitus
650	0.02	10.87	0.26	2	7	Butanoate metabolism
30	0.04	7.75	0.33	2	9	Pentose phosphate pathway
250	0.04	7.75	0.33	2	9	Alanine, aspartate and glutamate metabolism
51	0.05	6.78	0.37	2	10	Fructose and mannose metabolism

Table 3: KEGGHyperG results