KEGG Pathway Analysis in R

Environmental Genomics, 2014

Thomas H. Hampton

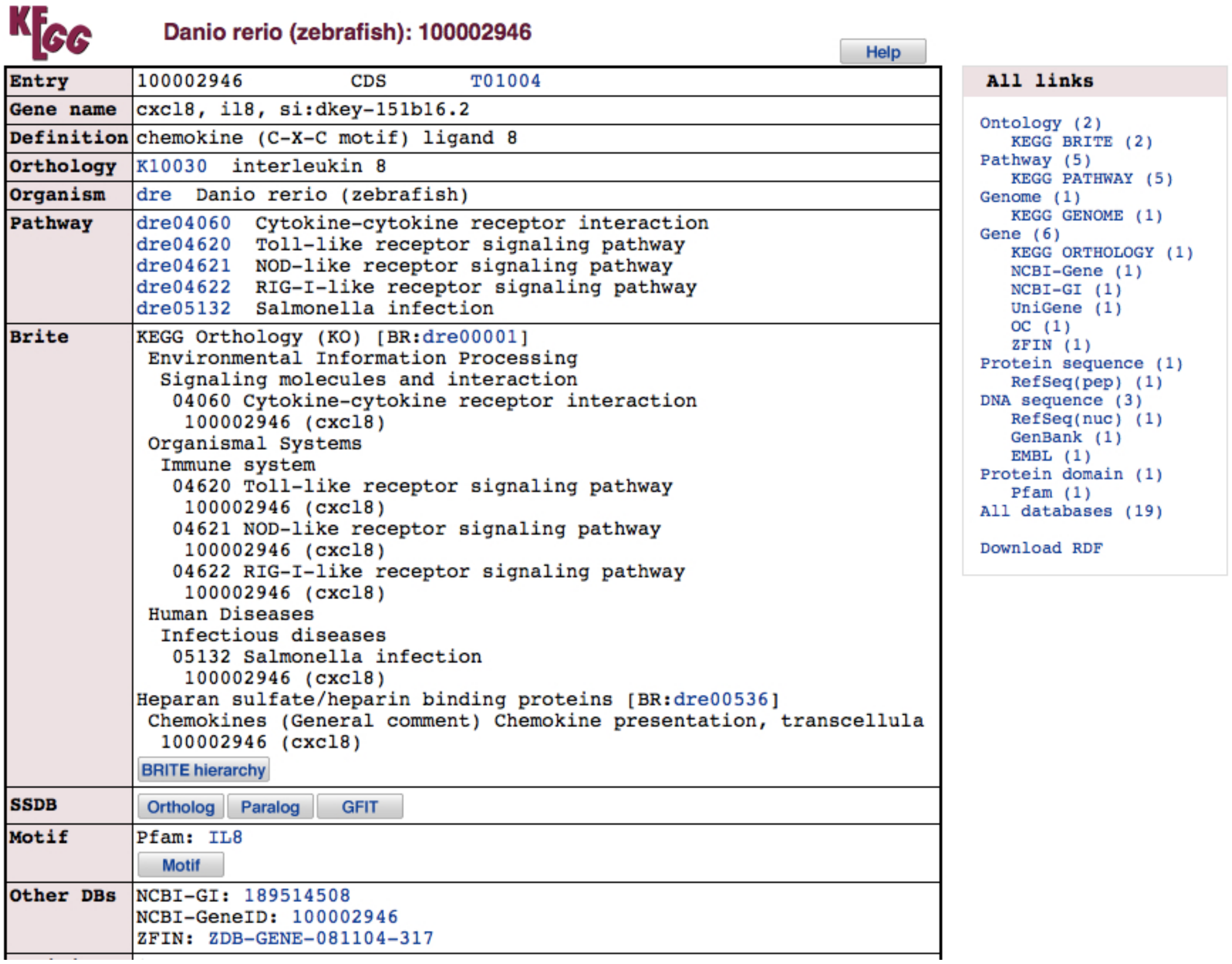
The Kyoto Encyclopedia of Genes and Genomes

Hanover Bioinformatics Symposium, 2011

Overview

- Simulate and analyze a zebrafish gene expression experiment

- Use the R interface to the KEGG pathway database to assess and visualize associations between differential gene expression and KEGG paths.



Danio rerio (zebrafish): 100002946

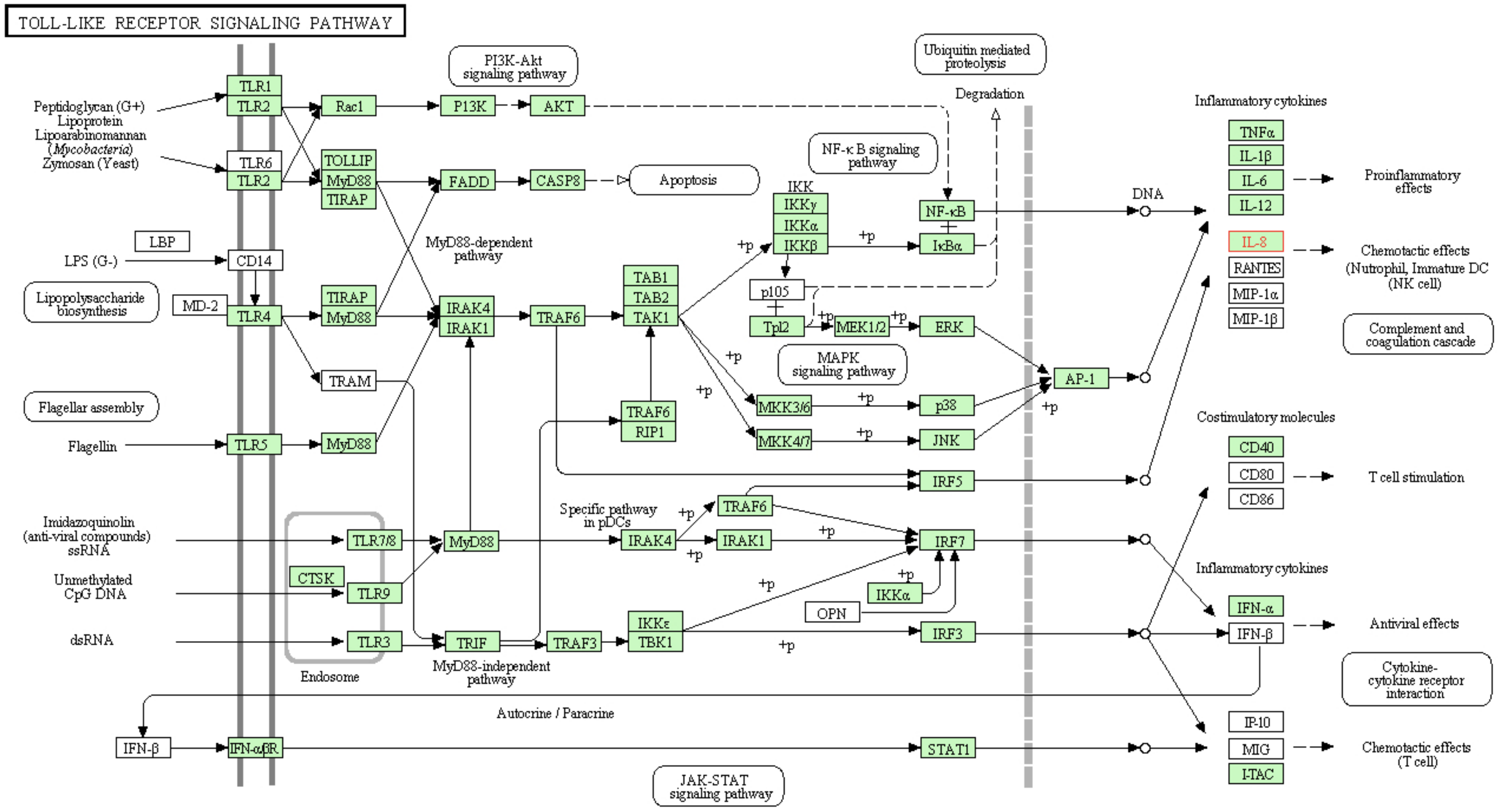
Help

Entry	100002946 CDS T01004
Gene name	cxcl8, il8, si:dkey-151b16.2
Definition	chemokine (C-X-C motif) ligand 8
Orthology	K10030 interleukin 8
Organism	dre Danio rerio (zebrafish)
Pathway	dre04060 Cytokine-cytokine receptor interaction dre04620 Toll-like receptor signaling pathway dre04621 NOD-like receptor signaling pathway dre04622 RIG-I-like receptor signaling pathway dre05132 Salmonella infection
Brite	KEGG Orthology (KO) [BR:dre00001] Environmental Information Processing Signaling molecules and interaction 04060 Cytokine-cytokine receptor interaction 100002946 (cxcl8) Organismal Systems Immune system 04620 Toll-like receptor signaling pathway 100002946 (cxcl8) 04621 NOD-like receptor signaling pathway 100002946 (cxcl8) 04622 RIG-I-like receptor signaling pathway 100002946 (cxcl8) Human Diseases Infectious diseases 05132 Salmonella infection 100002946 (cxcl8) Heparan sulfate/heparin binding proteins [BR:dre00536] Chemokines (General comment) Chemokine presentation, transcellula 100002946 (cxcl8) BRITE hierarchy
SSDB	Ortholog Paralog GFIT
Motif	Pfam: IL8 Motif
Other DBs	NCBI-GI: 189514508 NCBI-GeneID: 100002946 ZFIN: ZDB-GENE-081104-317

All links

Ontology (2)
  KEGG BRITE (2)
Pathway (5)
  KEGG PATHWAY (5)
Genome (1)
  KEGG GENOME (1)
Gene (6)
  KEGG ORTHOLOGY (1)
  NCBI-Gene (1)
  NCBI-GI (1)
  UniGene (1)
  OC (1)
  ZFIN (1)
Protein sequence (1)
  RefSeq(pep) (1)
DNA sequence (3)
  RefSeq(nuc) (1)
  GenBank (1)
  EMBL (1)
Protein domain (1)
  Pfam (1)
All databases (19)

Download RDF


TOLL-LIKE RECEPTOR SIGNALING PATHWAY
Peptidoglycan (G+)
Lipoprotein
Lipoarabinomannan
(Mycobacteria)
Zymosan (Yeast)
TLR1
TLR2
TLR6
TLR2
LBP
LPS (G-)
CD14
MD-2
TLR4
Lipopolysaccharide biosynthesis
Flagellar assembly
Flagellin
TLR5
Imidazoquinolin
(anti-viral compounds)
ssRNA
Unmethylated CpG DNA
dsRNA
TLR7/8
CTSK
TLR9
TLR3
Endosome
Rac1
PI3K-Akt signaling pathway
PI3K
AKT
TOLLIP
MyD88
TIRAP
FADD
CASP8
Apoptosis
MyD88-dependent pathway
TIRAP
MyD88
IRAK4
IRAK1
TRAF6
TAB1
TAB2
TAK1
TRAM
MyD88
TRAF6
RIP1
MyD88
IRAK4
IRAK1
TRAF6
Specific pathway in pDCs
TRIF
TRAF3
IKKε
TBK1
MyD88-independent pathway
IKK
IKKγ
IKKα
IKKβ
p105
Tpl2
MEK1/2
ERK
NF-κB signaling pathway
NF-κB
IκBα
Ubiquitin mediated proteolysis
Degradation
MAPK signaling pathway
MKK3/6
MKK4/7
p38
JNK
AP-1
IRF5
IRF7
IKKα
OPN
IRF3
+p
DNA
Inflammatory cytokines
TNFα
IL-1β
IL-6
IL-12
IL-8
RANTES
MIP-1α
MIP-1β
Proinflammatory effects
Chemotactic effects (Nutrophil, Immature DC (NK cell)
Complement and coagulation cascade
Costimulatory molecules
CD40
CD80
CD86
T cell stimulation
Inflammatory cytokines
IFN-α
IFN-β
Antiviral effects
Cytokine-cytokine receptor interaction
IP-10
MIG
I-TAC
Chemotactic effects (T cell)
Autocrine / Paracrine
IFN-β
IFN-α/βR
STAT1
JAK-STAT signaling pathway

Basic Enrichment Idea

- We say that a gene set is enriched in genes related to some path when the gene set contains more genes from that path than you would predict by chance.
- For example: if 0.1% of all genes are differentially expressed, and 0.2% of genes (twice as many) from path X are differentially expressed in our experiment, we might suspect enrichment.
- Fisher's exact test can estimate the probability that a random draw from an idealized gene universe would produce the 2-fold (or more) enrichment we observed.
- Since there are 162 paths, you'd expect 5% (about 8) to achieve significance by chance...

Install Packages and Load Libraries

```r
source("http://bioconductor.org/biocLite.R")

biocLite("KEGGREST")

biocLite("org.Dr.eg.db")

biocLite("pathview")

library(KEGGREST)

library(org.Dr.eg.db)

library(pathview)

library(gplots)
```

KEGGREST

```
ls("package:KEGGREST")
```

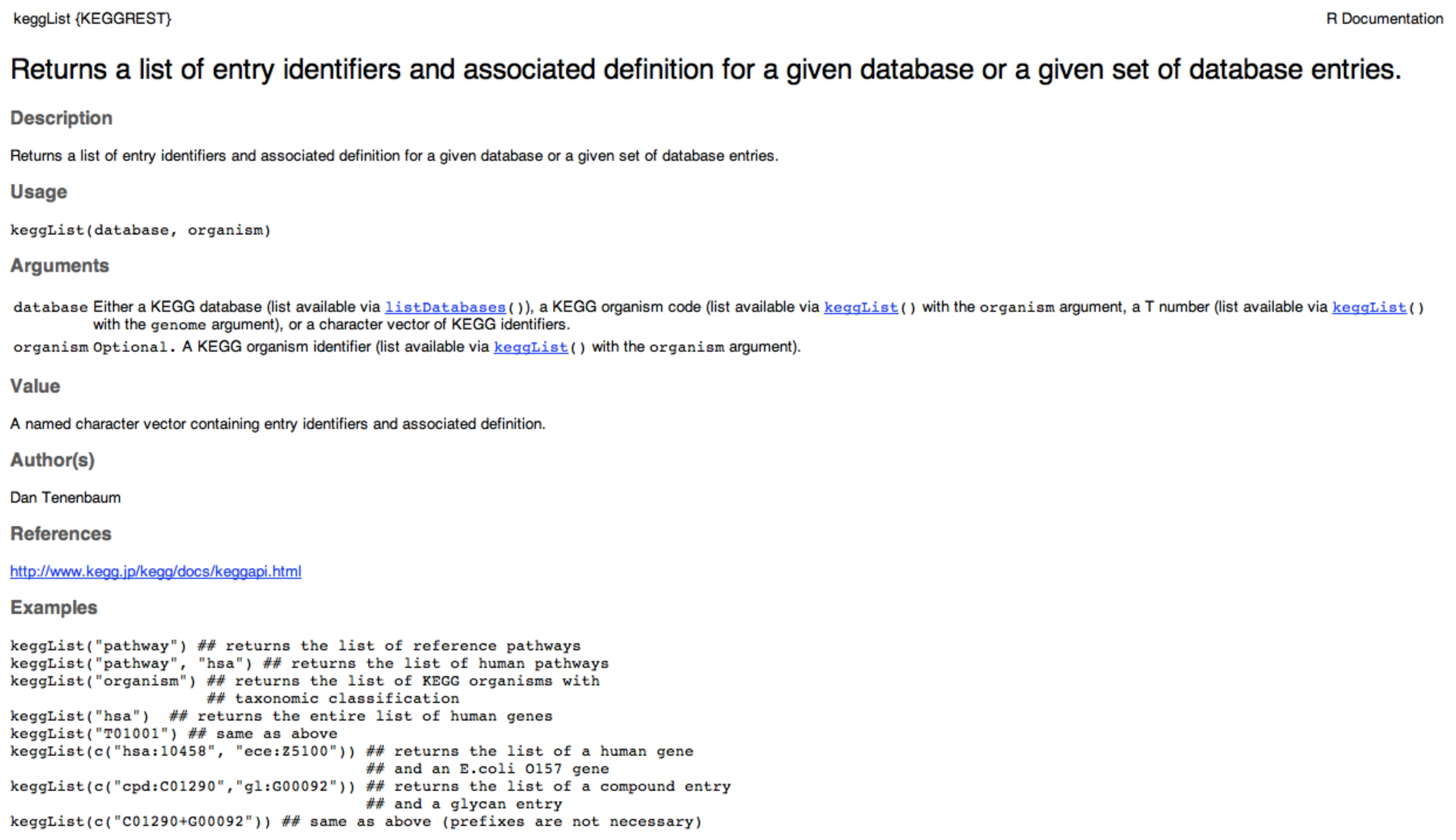
```
## [1] "color.pathway.by.objects" "keggConv"
## [3] "keggFind"                 "keggGet"
## [5] "keggInfo"                 "keggLink"
## [7] "keggList"                 "listDatabases"
## [9] "mark.pathway.by.objects"
```

```
listDatabases()
```

```
##  [1] "pathway"  "brite"    "module"   "disease"  "drug"     "environ"
##  [7] "ko"       "genome"   "compound" "glycan"   "reaction" "rpair"
## [13] "rclass"   "enzyme"   "organism"
```

Information on keggList

```
?keggList
```



keggList {KEGGREST} R Documentation

Returns a list of entry identifiers and associated definition for a given database or a given set of database entries.

Description

Returns a list of entry identifiers and associated definition for a given database or a given set of database entries.

Usage

```
keggList(database, organism)
```

Arguments

`database`	Either a KEGG database (list available via `listDatabases()`), a KEGG organism code (list available via `keggList()` with the `organism` argument, a T number (list available via `keggList()` with the `genome` argument), or a character vector of KEGG identifiers.
`organism`	`Optional`. A KEGG organism identifier (list available via `keggList()` with the `organism` argument).

Value

A named character vector containing entry identifiers and associated definition.

Author(s)

Dan Tenenbaum

References

http://www.kegg.jp/kegg/docs/keggapi.html

Examples

```
keggList("pathway") ## returns the list of reference pathways
keggList("pathway", "hsa") ## returns the list of human pathways
keggList("organism") ## returns the list of KEGG organisms with
                     ## taxonomic classification
keggList("hsa")  ## returns the entire list of human genes
keggList("T01001") ## same as above
keggList(c("hsa:10458", "ece:Z5100")) ## returns the list of a human gene
                                      ## and an E.coli O157 gene
keggList(c("cpd:C01290","gl:G00092")) ## returns the list of a compound entry
                                      ## and a glycan entry
keggList(c("C01290+G00092")) ## same as above (prefixes are not necessary)
```

Identify the KEGG Gene Universe (~30,000 Genes)

1. Get all the zebrafish genes known to KEGG:

```
ZebraGenesList <- keggList("dre") # dre = danio rerio
```

2. Look at the first entry:

```
ZebraGenesList[1]
```

```
# Gene ID of first entry:
names(ZebraGenesList)[1]
```

```
## [1] "dre:573207"
```

Extract Unique Gene Identifiers

3. Remove "dre:" from all gene IDs:

```
Gene.IDs <- unlist(
        lapply(names(ZebraGenesList), function(x) {
            gsub("dre:", "", x)
    }))

Gene.IDs
```

Identify KEGG Paths

4. Get all the zebrafish KEGG paths:

```
Daniopaths <- keggList("pathway", "dre")
as.character(Daniopaths)
```

```
##  [1] "Glycolysis / Gluconeogenesis - Danio rerio (zebrafish)"
##  [2] "Citrate cycle (TCA cycle) - Danio rerio (zebrafish)"
##  [3] "Pentose phosphate pathway - Danio rerio (zebrafish)"
##  [4] "Pentose and glucuronate interconversions - Danio rerio (zebrafish)"
##  [5] "Fructose and mannose metabolism - Danio rerio (zebrafish)"
##  [6] "Galactose metabolism - Danio rerio (zebrafish)"
##  [7] "Ascorbate and aldarate metabolism - Danio rerio (zebrafish)"
##  [8] "Fatty acid biosynthesis - Danio rerio (zebrafish)"
##  [9] "Fatty acid elongation - Danio rerio (zebrafish)"
## [10] "Fatty acid degradation - Danio rerio (zebrafish)"
## [11] "Synthesis and degradation of ketone bodies - Danio rerio (zebrafish)"
## [12] "Steroid biosynthesis - Danio rerio (zebrafish)"
## [13] "Primary bile acid biosynthesis - Danio rerio (zebrafish)"
## [14] "Ubiquinone and other terpenoid-quinone biosynthesis - Danio rerio (zebrafish)"
## [15] "Steroid hormone biosynthesis - Danio rerio (zebrafish)"
## [16] "Oxidative phosphorylation - Danio rerio (zebrafish)"
```

Extract Unique Path Identifiers

5. Remove "path:dre" from all entries:

```
# Look at first path entry:
Daniopaths[1]
```

```
##                                            path:dre00010
## "Glycolysis / Gluconeogenesis - Danio rerio (zebrafish)"
```

```
names(Daniopaths)[1]
```

```
## [1] "path:dre00010"
```

```
Paths.IDs  <- unlist(lapply(names(Daniopaths), function(x) {
                gsub("path:dre", "", x)
}))
```

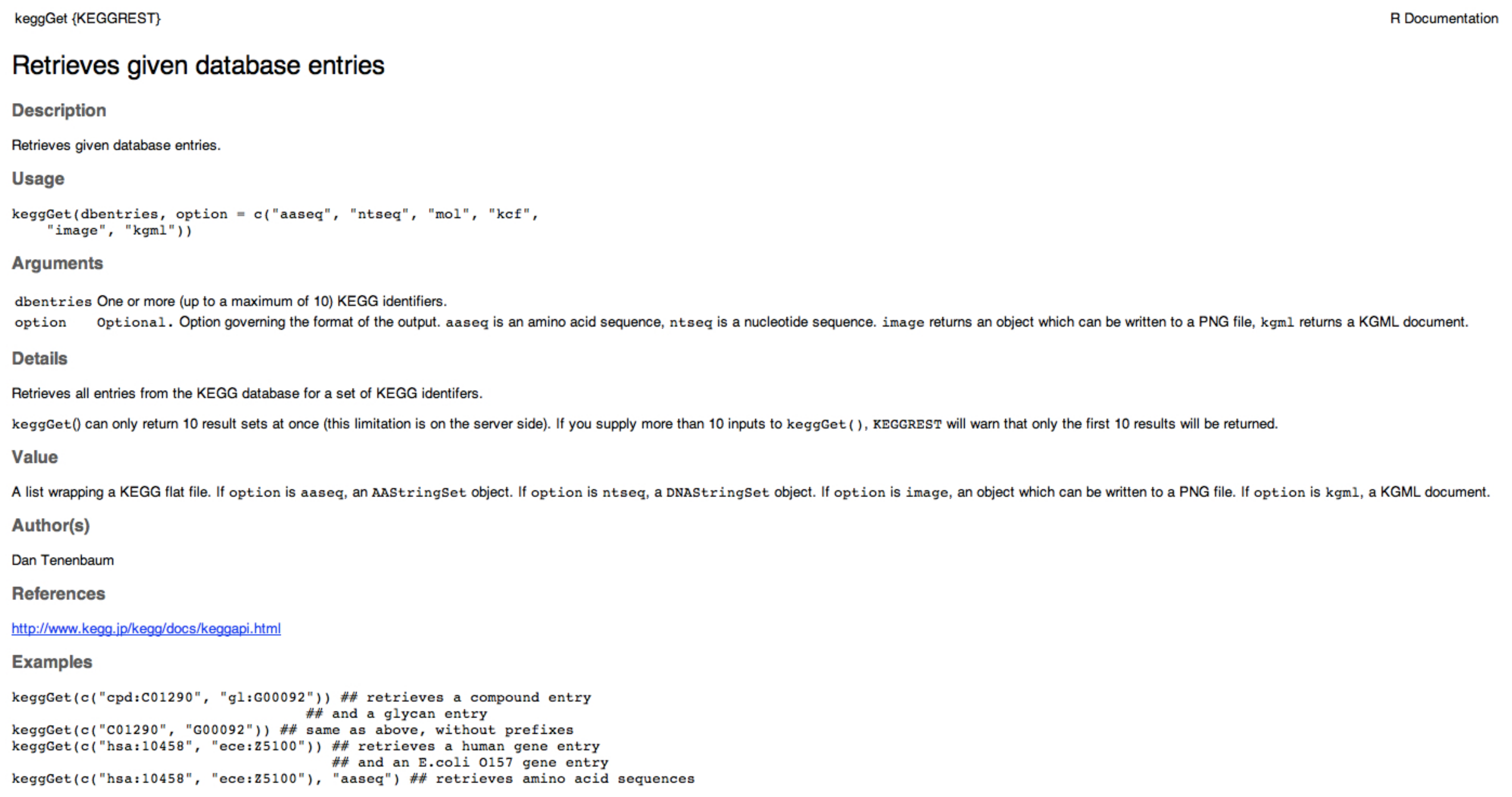
Paths IDs

```
Paths.IDs
```

```
##   [1] "00010" "00020" "00030" "00040" "00051" "00052" "00053" "00061"
##   [9] "00062" "00071" "00072" "00100" "00120" "00130" "00140" "00190"
##  [17] "00230" "00232" "00240" "00250" "00260" "00270" "00280" "00290"
##  [25] "00300" "00310" "00330" "00340" "00350" "00360" "00380" "00400"
##  [33] "00410" "00430" "00450" "00460" "00471" "00472" "00480" "00500"
##  [41] "00510" "00511" "00512" "00514" "00520" "00524" "00531" "00532"
##  [49] "00533" "00534" "00561" "00562" "00563" "00564" "00565" "00590"
##  [57] "00591" "00592" "00600" "00601" "00603" "00604" "00620" "00630"
##  [65] "00640" "00650" "00670" "00730" "00740" "00750" "00760" "00770"
##  [73] "00780" "00785" "00790" "00830" "00860" "00900" "00910" "00920"
##  [81] "00970" "00980" "00982" "00983" "01040" "01100" "01200" "01210"
##  [89] "01212" "01220" "01230" "02010" "03008" "03010" "03013" "03015"
##  [97] "03018" "03020" "03022" "03030" "03040" "03050" "03060" "03320"
## [105] "03410" "03420" "03430" "03440" "03450" "03460" "04010" "04012"
## [113] "04020" "04060" "04068" "04070" "04080" "04110" "04114" "04115"
## [121] "04120" "04122" "04130" "04140" "04141" "04142" "04144" "04145"
## [129] "04146" "04150" "04210" "04260" "04261" "04270" "04310" "04320"
## [137] "04330" "04340" "04350" "04370" "04510" "04512" "04514" "04520"
## [145] "04530" "04540" "04614" "04620" "04621" "04622" "04623" "04630"
```

Information on keggGet

```
?keggGet
```



keggGet {KEGGREST} R Documentation

Retrieves given database entries

Description

Retrieves given database entries.

Usage

```
keggGet(dbentries, option = c("aaseq", "ntseq", "mol", "kcf",
    "image", "kgml"))
```

Arguments

`dbentries`	One or more (up to a maximum of 10) KEGG identifiers.
`option`	`Optional.` Option governing the format of the output. `aaseq` is an amino acid sequence, `ntseq` is a nucleotide sequence. `image` returns an object which can be written to a PNG file, `kgml` returns a KGML document.

Details

Retrieves all entries from the KEGG database for a set of KEGG identifers.

`keggGet`() can only return 10 result sets at once (this limitation is on the server side). If you supply more than 10 inputs to `keggGet()`, `KEGGREST` will warn that only the first 10 results will be returned.

Value

A list wrapping a KEGG flat file. If `option` is `aaseq`, an `AAStringSet` object. If `option` is `ntseq`, a `DNAStringSet` object. If `option` is `image`, an object which can be written to a PNG file. If `option` is `kgml`, a KGML document.

Author(s)

Dan Tenenbaum

References

http://www.kegg.jp/kegg/docs/keggapi.html

Examples

```
keggGet(c("cpd:C01290", "gl:G00092")) ## retrieves a compound entry
                                  ## and a glycan entry
keggGet(c("C01290", "G00092")) ## same as above, without prefixes
keggGet(c("hsa:10458", "ece:Z5100")) ## retrieves a human gene entry
                                     ## and an E.coli O157 gene entry
keggGet(c("hsa:10458", "ece:Z5100"), "aaseq") ## retrieves amino acid sequences
```

Content of a Path

```r
str(keggGet("path:dre04916"))
```

```
## List of 1
##  $ :List of 10
##   ..$ ENTRY      : Named chr "dre04916"
##   .. ..- attr(*, "names")= chr "Pathway"
##   ..$ NAME       : chr "Melanogenesis - Danio rerio (zebrafish)"
##   ..$ DESCRIPTION: chr "Cutaneous melanin pigment plays a critical role in camouflage, mimi
##   ..$ CLASS      : chr "Organismal Systems; Endocrine system"
##   ..$ PATHWAY_MAP: Named chr "Melanogenesis"
##   .. ..- attr(*, "names")= chr "dre04916"
##   ..$ ORGANISM   : Named chr "NA  Danio rerio (zebrafish) [GN:dre]"
##   .. ..- attr(*, "names")= chr "Danio rerio (zebrafish) [GN:dre]"
##   ..$ GENE       : chr [1:274] "353221" "pomca; proopiomelanocortin a [KO:K05228]" "798574'
##   ..$ COMPOUND   : Named chr [1:6] "Calcium cation" "L-Tyrosine" "Diacylglycerol" "3',5'-Cy
##   .. ..- attr(*, "names")= chr [1:6] "C00076" "C00082" "C00165" "C00575" ...
##   ..$ KO_PATHWAY : chr "ko04916"
##   ..$ REFERENCE  :List of 13
##   .. ..$ :List of 4
##   .. .. ..$ REFERENCE: chr "PMID:15383650"
##   .. .. ..$ AUTHORS  : chr "Slominski A, Tobin DJ, Shibahara S, Wortsman J."
```

Identify Genes That are on Paths

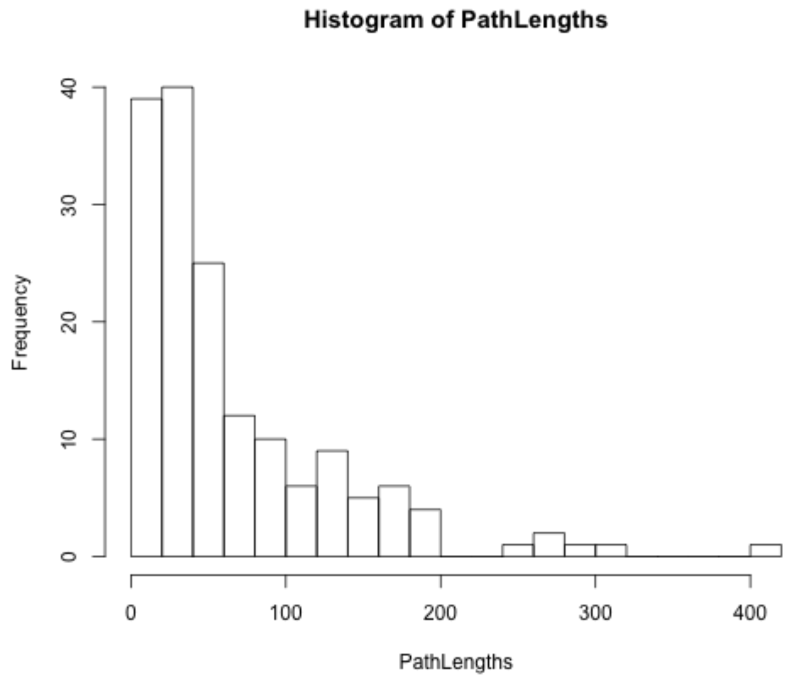
Extract genes from paths (this might take a few minutes!):

```
DaniopathsGenes <- lapply(names(Daniopaths), function (x) {
        glist = keggGet(x)[[1]]$GENE
    if(length(glist) > 0){
        glist[seq(from = 1, to = length(glist), by = 2)]
    } else {
        NULL
    }
})

names(DaniopathsGenes) <- names(Daniopaths)
DaniopathsGenes
```

```
## $`path:dre00010`
##  [1] "406339"    "406791"    "321224"    "100329357" "100536576"
##  [6] "751668"    "246095"    "246094"    "561416"    "447836"
## [11] "560944"    "570106"    "568001"    "335231"    "282672"
## [16] "445505"    "321664"    "369193"    "792692"    "406496"
## [21] "336425"    "192309"    "560753"    "317743"    "406367"
## [26] "406696"    "572733"    "323107"    "327165"    "402874"
```

How Many Genes are on Each Path?

```
PathLengths <- unlist(lapply(DaniopathsGenes, length))
hist(PathLengths, breaks = 15)
```



Some Genes are on Many Paths

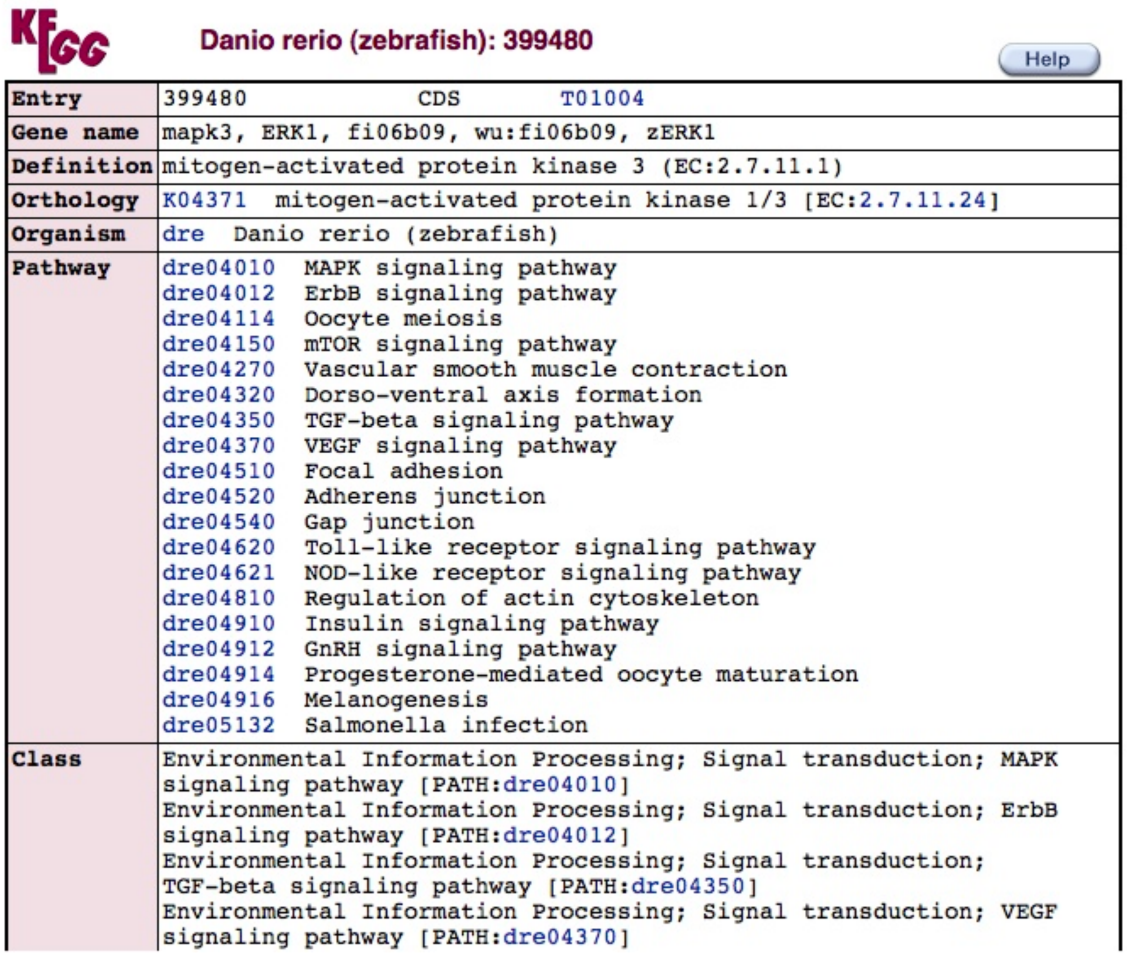
Create a superset of all path genes:

```
KeggPathGenes <- as.character(unlist(DaniopathsGenes))
sort(table(KeggPathGenes))
```

Look at one of the most popular genes that is in 21 different paths:

```
ZebraGenesList[Gene.IDs == "399480"]
```

```
##
## "mapk3, ERK1, fi06b09, wu:fi06b09, zERK1; mitogen-activated protein kinase 3 (EC:2.7.11.1);
```



Danio rerio (zebrafish): 399480

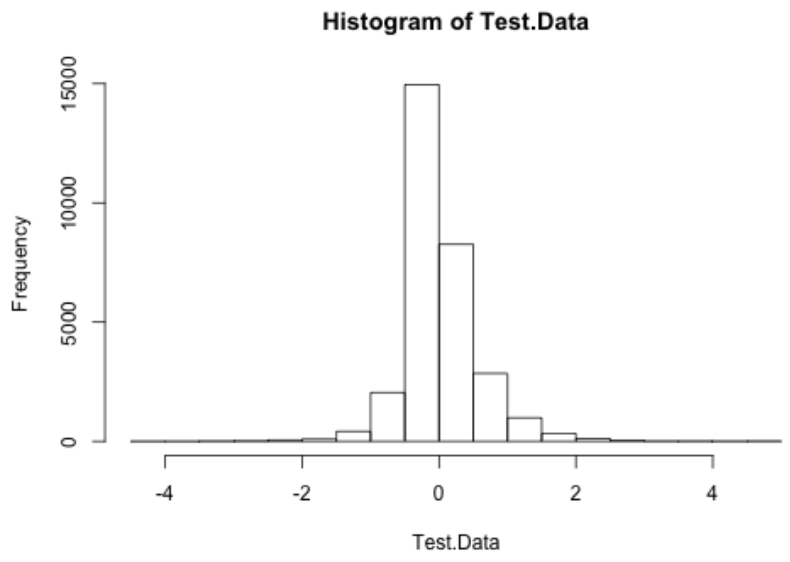
Entry	399480 CDS T01004
Gene name	mapk3, ERK1, fi06b09, wu:fi06b09, zERK1
Definition	mitogen-activated protein kinase 3 (EC:2.7.11.1)
Orthology	K04371 mitogen-activated protein kinase 1/3 [EC:2.7.11.24]
Organism	dre Danio rerio (zebrafish)
Pathway	dre04010 MAPK signaling pathway dre04012 ErbB signaling pathway dre04114 Oocyte meiosis dre04150 mTOR signaling pathway dre04270 Vascular smooth muscle contraction dre04320 Dorso-ventral axis formation dre04350 TGF-beta signaling pathway dre04370 VEGF signaling pathway dre04510 Focal adhesion dre04520 Adherens junction dre04540 Gap junction dre04620 Toll-like receptor signaling pathway dre04621 NOD-like receptor signaling pathway dre04810 Regulation of actin cytoskeleton dre04910 Insulin signaling pathway dre04912 GnRH signaling pathway dre04914 Progesterone-mediated oocyte maturation dre04916 Melanogenesis dre05132 Salmonella infection
Class	Environmental Information Processing; Signal transduction; MAPK signaling pathway [PATH:dre04010] Environmental Information Processing; Signal transduction; ErbB signaling pathway [PATH:dre04012] Environmental Information Processing; Signal transduction; TGF-beta signaling pathway [PATH:dre04350] Environmental Information Processing; Signal transduction; VEGF signaling pathway [PATH:dre04370]

Simulated Experiment

1. Generate expression data (log2 fold changes)
2. Pick 75 "regulated" genes at random
3. Pick a path at random
4. Add about half of the genes from this path to our regulated gene set, and make them upregulated
5. Use Fisher's exact test on each KEGG Path
6. Make pictures of each significant path

Generate Expression Data (log2 Fold Changes)

```
data(gse16873.d)
Test.Data <- sample(as.numeric(gse16873.d[, 1]), length(ZebraGenesList), replace=TRUE)
names(Test.Data) <- Gene.IDs
hist(Test.Data)
```



Pick a Path

Pick a path number at random, but pick a hefty one:

```r
MyPath <- sample(which(PathLengths > 20), 1)
MyPath
```

Look up the path based on this number:

```r
Daniopaths[MyPath]
```

Simulated Gene List

1. Pick 200 "regulated" genes at random, regardless of path

```
StartSet <- sample(Gene.IDs, 200, replace=FALSE)
```

2. Add about 10% of the genes from our path to our regulated gene set

```
SpikeInNumber <- as.integer(0.1* PathLengths[MyPath])

SpikedGenes <- sample(DaniopathsGenes[[MyPath]], SpikeInNumber)

RegGenes <- union(StartSet, SpikedGenes)

RegGenes
```

Add Regulation to our Path

Add upregulation for dramatic effect:

```
Test.Data[SpikedGenes]


Test.Data[SpikedGenes] <- max(Test.Data)


Test.Data[SpikedGenes]
```

Use Fisher's Exact Test on each KEGG Path

create 2 x 2 matrix

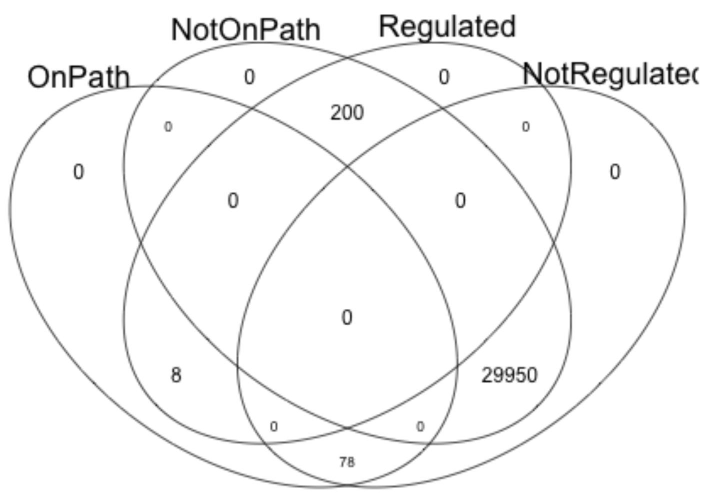
```
##             On_Path Off_Path
## Regulated         A        C
## Unregulated       B        D
```

Basic Sets

```
OnPath <- DaniopathsGenes[[MyPath]]

NotOnPath <- setdiff(Gene.IDs, DaniopathsGenes[[MyPath]])

Regulated <- RegGenes

NotRegulated <- setdiff(Gene.IDs, RegGenes)
```

Visualizing Sets with Venn Diagrams

```
venn(list("OnPath" = OnPath, "NotOnPath" = NotOnPath, "Regulated" = Regulated,
"NotRegulated" = NotRegulated))
```



Number of Regulated Genes on Path

Intersect of "On Path" and "Regulated"

Is this number higher than the number of regulated genes on this path expected by chance?

```
regulationRate <- length(RegGenes)/length(Gene.IDs)

Expected <- regulationRate*length(DaniopathsGenes[[MyPath]])

Expected
```

```
## [1] 0.5916
```

Matrix Values for 2x2 Matrix

```
vals <- c(length(intersect(OnPath,Regulated)),
        length(intersect(OnPath,NotRegulated)),
        length(intersect(NotOnPath,Regulated)),
        length(intersect(NotOnPath,NotRegulated)))
vals
```

```
## [1]     8    78   200 29950
```

```
##             On_Path Off_Path
## Regulated         A        C
## Unregulated       B        D
```

Fisher's Exact Test

```
PathMat <- matrix(vals, nrow = 2, dimnames = list(Regulated = c("Yes", "No"),
                                                  OnPath = c("Yes", "No")))
PathMat
```

```
##          OnPath
## Regulated Yes    No
##       Yes   8   200
##       No   78 29950
```

```
fisher.test(PathMat, alternative = "greater")
```

```
##
##  Fisher's Exact Test for Count Data
##
## data:  PathMat
## p-value = 1.468e-07
## alternative hypothesis: true odds ratio is greater than 1
## 95 percent confidence interval:
##  7.318   Inf
## sample estimates:
```

Function To Create Matrix Values

```r
MatVals <- function (pathno, reggenes){
        OnPath = DaniopathsGenes[[pathno]]
    NotOnPath = setdiff(Gene.IDs, DaniopathsGenes[[pathno]])
    Regulated = reggenes
    NotRegulated = setdiff(Gene.IDs, reggenes)
    vals <- c(length(intersect(OnPath,Regulated)),
              length(intersect(OnPath,NotRegulated)),
              length(intersect(NotOnPath,Regulated)),
              length(intersect(NotOnPath,NotRegulated)))
    return(vals)
}
```

Test the MatVals Function

```
MatVals(MyPath, RegGenes)
```

```
## [1]     8    78   200 29950
```

Compare output of MatVals with previously calculated vals:

```
v <- MatVals(MyPath, RegGenes)
cbind(v, vals)
```

```
##          v  vals
## [1,]     8     8
## [2,]    78    78
## [3,]   200   200
## [4,] 29950 29950
```

Run Fisher's Exact Test on Every Path

```r
Results <- lapply(1: length(Paths.IDs), function (x) {
        v <- MatVals(x, RegGenes)
        PathMat <- matrix(v, nrow = 2,
                          dimnames = list(Regulated = c("Yes", "No"),
                                          OnPath = c("Yes", "No")))

    fisher.test(PathMat, alternative = "greater")$p.value
})
```

Check Results (I)

```
Daniopaths[Results < 0.05]
```

```
##                                                      path:dre00640
##                    "Propanoate metabolism - Danio rerio (zebrafish)"
##                                                      path:dre04010
##                   "MAPK signaling pathway - Danio rerio (zebrafish)"
##                                                      path:dre04060
##   "Cytokine-cytokine receptor interaction - Danio rerio (zebrafish)"
##                                                      path:dre04145
##                               "Phagosome - Danio rerio (zebrafish)"
##                                                      path:dre04150
##                   "mTOR signaling pathway - Danio rerio (zebrafish)"
##                                                      path:dre04310
##                    "Wnt signaling pathway - Danio rerio (zebrafish)"
##                                                      path:dre04370
##                   "VEGF signaling pathway - Danio rerio (zebrafish)"
##                                                      path:dre04510
##                          "Focal adhesion - Danio rerio (zebrafish)"
##                                                      path:dre04520
##                       "Adherens junction - Danio rerio (zebrafish)"
##                                                      path:dre04620
```

Check Results (II)

Distribution of p-values:

```
hist(unlist(Results), xlab = "p-value")
```

Number of significant paths:

```
length(Daniopaths[Results < 0.05])
```

Look at top 5 paths with lowest p-values and compare to MyPath:

```
Top.paths <- order(unlist(Results))[1:5]

Daniopaths[Top.paths]

Daniopaths[MyPath]
```

Make KEGG Graphs

(this might take a while)

```r
Pv <- lapply(Paths.IDs[Results < 0.05], function (x){
                pathview(gene.data = Test.Data,
                        pathway.id = x,
                        species = "dre",
                        out.suffix = "Reg", kegg.native = T)
})
```

png files with statistically signigicant pathways are saved to the working directory.

Note

We could use these functions with real data by replacing "RegGenes" with a real gene set, and "Test.Data" with real expression data.

Thanks Again