Pathway Analysis of Metabolic Activities

Gabriele Sales

2018-05-21

Introduction

In this tutorial we are going to analyze the dataset of metabolic activities published in Terunuma et al (2017). In this paper the Authors characterized the transcriptomic and metabolomic profile of several types of human breast tumors to uncovered metabolite signatures using an untargeted discovery approach. They found that the oncometabolite 2-hydroxyglutarate (2HG) accumulates at high levels in a subset of tumors and discovered an association between increased 2HG levels and MYC pathway activation in breast cancer. As an example of mixed metabolites and genes topological analyses here we are going to use both the dataset of metabolite abundances and gene expression as reported in the supplementary files selecting only tumour samples and comparing ER+ and ER- breast cancers. We will use pathway information from the KEGG and SMPDB databases to identify a set of features whose activity changes significantly between the two sample classes. This result will hopefully hint at some specific biological activities that are pathologically altered in tumoral samples.

Data Preparation

We start by loading the experimental data into R. The measurements are stored in two simple text files, one for the genes and one for the metabolites. Thanks to the utility functions provided by R, we can combine the download and the actual reading of a table in a single operation. We also specify the details of the file format: the first row of input represents the header of the table; the tabulation character `\t` separates values; we don't want R to change strings into factors.

```
gene_url <-
  url("https://romualdi.bio.unipd.it/wp-uploads/2018/04/Terunuma_gene_expr.txt")
gexpr <- read.table(gene_url, header = TRUE, sep = "\t", row.names = 1,
                    stringsAsFactors = FALSE)
```

```
metab_url <-
  url("https://romualdi.bio.unipd.it/wp-uploads/2018/04/Terunuma_metabolite_expr.txt")
mexpr <- read.table(metab_url, header = TRUE, sep = "\t", row.names = NULL,
                    stringsAsFactors = FALSE)
```

The loaded table for the metabolites contains 436 rows and 112 columns, but not all of them actually correspond to measurements. We ask R to list the names of the columns:

```
colnames(mexpr)
```

```
##   [1] "METABOLON_ID"  "KEGG_ID"       "CAS"           "HMDB_ID"
##   [5] "POS.NORMAL"    "POS.TUMOR"     "POS.NORMAL.1"  "POS.TUMOR.1"
##   [9] "NEG.TUMOR"     "NEG.NORMAL"    "NEG.TUMOR.1"   "POS.NORMAL.2"
##  [13] "POS.TUMOR.2"   "POS.TUMOR.3"   "POS.NORMAL.3"  "NEG.NORMAL.1"
##  [17] "NEG.TUMOR.2"   "NEG.NORMAL.2"  "NEG.TUMOR.3"   "NEG.NORMAL.3"
##  [21] "NEG.TUMOR.4"   "POS.NORMAL.4"  "POS.TUMOR.4"   "POS.TUMOR.5"
##  [25] "POS.NORMAL.5"  "POS.TUMOR.6"   "NEG.NORMAL.4"  "NEG.TUMOR.5"
##  [29] "NEG.NORMAL.5"  "NEG.NORMAL.6"  "NEG.TUMOR.6"   "NEG.NORMAL.7"
##  [33] "NEG.TUMOR.7"   "NEG.TUMOR.8"   "POS.NORMAL.6"  "POS.TUMOR.7"
##  [37] "POS.NORMAL.7"  "POS.NORMAL.8"  "POS.TUMOR.8"   "POS.TUMOR.9"
##  [41] "POS.TUMOR.10"  "POS.NORMAL.9"  "POS.NORMAL.10" "POS.TUMOR.11"
```

```
##  [45] "NEG.TUMOR.9"   "NEG.TUMOR.10"  "NEG.NORMAL.8"  "NEG.TUMOR.11"
##  [49] "NEG.NORMAL.9"  "NEG.NORMAL.10" "NEG.TUMOR.12"  "POS.TUMOR.12"
##  [53] "NEG.NORMAL.11" "NEG.TUMOR.13"  "POS.TUMOR.13"  "POS.NORMAL.11"
##  [57] "POS.NORMAL.12" "POS.TUMOR.14"  "NEG.NORMAL.12" "NEG.TUMOR.14"
##  [61] "POS.NORMAL.13" "POS.TUMOR.15"  "POS.NORMAL.14" "POS.TUMOR.16"
##  [65] "POS.TUMOR.17"  "NEG.TUMOR.15"  "NEG.TUMOR.16"  "NEG.NORMAL.13"
##  [69] "POS.TUMOR.18"  "NEG.TUMOR.17"  "NEG.NORMAL.14" "NEG.NORMAL.15"
##  [73] "NEG.TUMOR.18"  "NEG.TUMOR.19"  "POS.TUMOR.19"  "POS.NORMAL.15"
##  [77] "POS.TUMOR.20"  "POS.NORMAL.16" "NEG.NORMAL.16" "NEG.TUMOR.20"
##  [81] "POS.NORMAL.17" "POS.TUMOR.21"  "POS.NORMAL.18" "POS.TUMOR.22"
##  [85] "NEG.NORMAL.17" "NEG.TUMOR.21"  "NEG.NORMAL.18" "NEG.TUMOR.22"
##  [89] "NEG.NORMAL.19" "NEG.TUMOR.23"  "POS.TUMOR.23"  "NEG.NORMAL.20"
##  [93] "NEG.TUMOR.24"  "POS.NORMAL.19" "POS.TUMOR.24"  "NEG.TUMOR.25"
##  [97] "NEG.NORMAL.21" "NEG.TUMOR.26"  "NEG.NORMAL.22" "NEG.TUMOR.27"
## [101] "NEG.NORMAL.23" "NEG.TUMOR.28"  "POS.NORMAL.20" "POS.TUMOR.25"
## [105] "POS.NORMAL.21" "POS.TUMOR.26"  "POS.NORMAL.22" "POS.TUMOR.27"
## [109] "POS.TUMOR.28"  "POS.TUMOR.29"  "POS.NORMAL.23" "POS.TUMOR.30"
```

As you can see the first four columns actually contain metabolite identifiers. We store their indices in a variable for future reference.

```
idcols <- 1:4
```

Missing Values

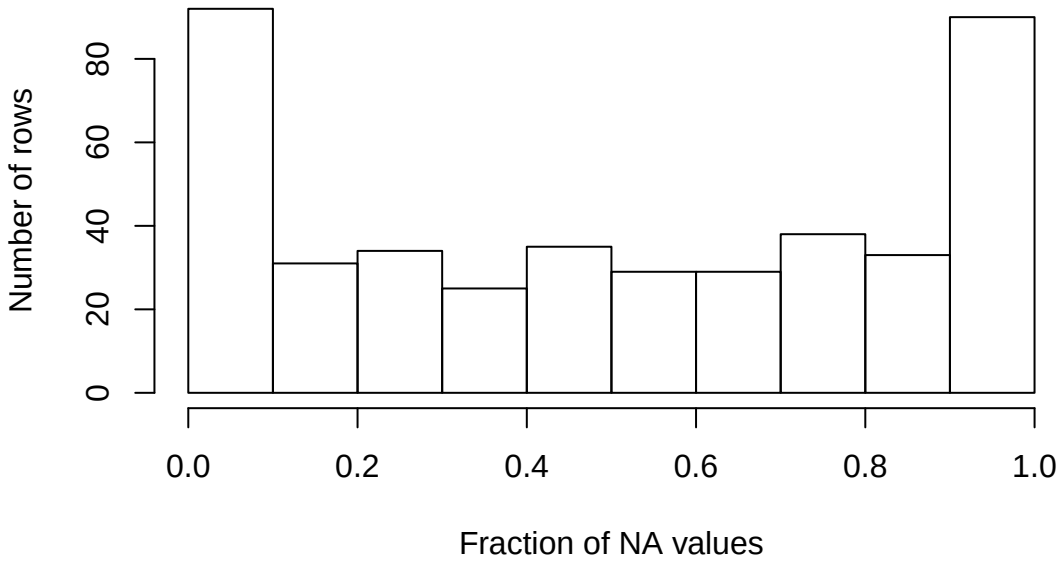
We display the first few rows of the table to get a feeling for its content. For the moment, we concentrate on the metabolite measurements.

```
head(mexpr[,5:10])
```

```
##   POS.NORMAL POS.TUMOR POS.NORMAL.1 POS.TUMOR.1 NEG.TUMOR NEG.NORMAL
## 1   24562.00  93036.29    108741.08    26753.39  340237.2   32141.90
## 2         NA        NA     33376.83    33413.49  242799.0   53352.70
## 3         NA 151243.74           NA    45499.88  592226.4         NA
## 4         NA        NA     37739.89          NA        NA         NA
## 5   24939.09 109819.69     71440.01   185000.21  193026.5   94471.26
## 6         NA 128398.33           NA    26460.05  359925.3         NA
```

It looks like there is a significant number of missing values. Let's plot a distribution of the fraction of NAs for each row.

```
nas_by_row <-
  rowSums(is.na(data.matrix(mexpr[,-idcols]))) /
  (ncol(mexpr) - length(idcols))
hist(nas_by_row, xlab = "Fraction of NA values", ylab = "Number of rows",
     main = NULL)
```



We cannot make good use of rows with too many missing values. We decide to drop them from our dataset. Specifically, we are going to remove anything with more than 50% of NAs.

```
dexpr <- mexpr[nas_by_row < 0.5,]
```

This operation leaves us with 216 of the original 216 rows.

We are not yet satisfied, though. The downstream analysis won't be able to cope with *any* NA. What would happen if we were to apply a more stringent procedure, removing any NA?

```
sum(rowSums(is.na(dexpr[,-idcols])) == 0)
```

```
## [1] 51
```

The above command is a little bit dense, so let's go through it in small steps. First we select all columns, except the identifiers. Then we check each value, testing if it's an NA. We tally how many NAs we have in each row and we consider only those in which the count is zero (meaning, no NA at all). The outer sum tells us how many rows would survive our filter.

The final verdict is quite grim. Only 51 out of 216 measurements would be used. We could do much better using a strategy that has become quite common in cases like this: *imputation*.

Instead of implementing it ourselves, we are going to use the excellent BioConductor package impute.

```
library(impute)
iexpr <- cbind(dexpr[,idcols],
               impute.knn(data.matrix(dexpr[,-idcols]))$data,
               stringsAsFactors = FALSE)
head(iexpr[, 1:6])
```

```
##    METABOLON_ID KEGG_ID         CAS   HMDB_ID POS.NORMAL  POS.TUMOR
## 1         35186                                 24562.00   93036.29
## 2         34214  C03819                         25996.12  114752.62
```

```
## 5         27665  C02918  1005-24-9; HMDB00699   24939.09  109819.69
## 8         33960  C03916 19420-56-5; HMDB02815   62594.21  393447.09
## 11        33955  C04102 17364-16-8;             49247.53 3392199.22
## 12        21188  C01885   123-94-4;             24076.66  132101.01
```

We can explicitly check there are not NAs left:

```
sum(is.na(iexpr[,-idcols]))
```

```
## [1] 0
```

Missing Identifiers

We now concentrate on the metabolite identifiers. We start again from the first few rows.

```
head(iexpr[,idcols])
```

```
##    METABOLON_ID KEGG_ID         CAS   HMDB_ID
## 1         35186
## 2         34214  C03819
## 5         27665  C02918  1005-24-9; HMDB00699
## 8         33960  C03916 19420-56-5; HMDB02815
## 11        33955  C04102 17364-16-8;
## 12        21188  C01885   123-94-4;
```

The CAS identifiers have a trailing ; we don't need and there are a number of empty strings, which really represent missing values. We fix both these issues.

```
iexpr$CAS <- sub("\\s*;.*$", "", iexpr$CAS)
iexpr[,idcols][iexpr[,idcols] == ""] <- NA
head(iexpr[,idcols])
```

```
##    METABOLON_ID KEGG_ID        CAS   HMDB_ID
## 1         35186    <NA>       <NA>      <NA>
## 2         34214  C03819       <NA>      <NA>
## 5         27665  C02918  1005-24-9 HMDB00699
## 8         33960  C03916 19420-56-5 HMDB02815
## 11        33955  C04102 17364-16-8      <NA>
## 12        21188  C01885   123-94-4      <NA>
```

We get a measure of how many identifiers are present or missing:

```
summary(is.na(iexpr[,idcols]))
```

```
##  METABOLON_ID     KEGG_ID           CAS           HMDB_ID
##  Mode :logical   Mode :logical   Mode :logical   Mode :logical
##  FALSE:216       FALSE:118       FALSE:133       FALSE:120
##                  TRUE :98        TRUE :83        TRUE :96
```

We have a Metabolon ID for each metabolite in the matrix, while in the case of CAS identifiers we find only 133 usable rows.

Unfortunately, at this point `graphite` does not support the Metabolon IDs. We do rely on CAS even if that means loosing a significant fraction of the rows.

```
valid_cas <- !is.na(iexpr$CAS)
cas_col <- which(names(iexpr) == "CAS")
cexpr <- iexpr[valid_cas, c(cas_col, seq.int(5, ncol(iexpr)))]
head(cexpr[,1:6])
```

```
##           CAS POS.NORMAL POS.TUMOR POS.NORMAL.1 POS.TUMOR.1  NEG.TUMOR
## 5   1005-24-9   24939.09  109819.7     71440.01   185000.21   193026.5
## 8  19420-56-5   62594.21  393447.1     80080.94   106041.34  3678588.6
## 11 17364-16-8   49247.53 3392199.2    233592.07  1006804.03 14434438.6
## 12   123-94-4   24076.66  132101.0     74279.39    85883.97   190430.8
## 13 19420-57-6  142204.89 1229674.4    173834.16   325500.75  9719220.2
## 14 69747-55-3   14310.14  428114.4     38336.17    89277.85   653406.9
```

There is no reason at this point to keep the identifiers as a column *inside* the dataset. We move such information to the row names and transform the `data.frame` into a numeric `matrix`.

```
mexpr <- data.matrix(cexpr[,-1])
rownames(mexpr) <- paste("CAS", cexpr$CAS, sep = ":")
colnames(mexpr) <- colnames(gexpr)
head(mexpr[,1:6])
```

```
##                POS.NORMAL POS.TUMOR POS.NORMAL.1 POS.TUMOR.1  NEG.TUMOR
## CAS:1005-24-9    24939.09  109819.7     71440.01   185000.21   193026.5
## CAS:19420-56-5   62594.21  393447.1     80080.94   106041.34  3678588.6
## CAS:17364-16-8   49247.53 3392199.2    233592.07  1006804.03 14434438.6
## CAS:123-94-4     24076.66  132101.0     74279.39    85883.97   190430.8
## CAS:19420-57-6  142204.89 1229674.4    173834.16   325500.75  9719220.2
## CAS:69747-55-3   14310.14  428114.4     38336.17    89277.85   653406.9
##                NEG.NORMAL
## CAS:1005-24-9    94471.26
## CAS:19420-56-5   69666.65
## CAS:17364-16-8  107447.35
## CAS:123-94-4     32521.62
## CAS:19420-57-6  178529.64
## CAS:69747-55-3   14600.61
```

Finally, we convert the metabolite levels by taking the logarithm and we merge the two matrices (one for the genes and one for the metabolites) together.

```
fexpr <- rbind(data.matrix(gexpr), log(mexpr))
```

Pathway Analysis

To make sense of the changes in metabolic activity recorded in the data we have just loaded, we are going to use pathway information from KEGG (via the graphite package) and a statistical analysis capable of exploiting the topology of such pathways (here we'll rely on clipper). `clipper` performs a test on means and covariances between two conditions and it is able to deal with variables of heterogeneous nature.

We import the required packages:

```
library(graphite)
library(clipper)
```

`clipper` will need three pieces of information:

1. a set of pathways
2. the activity matrix (`mexpr` in our case)
3. a vector representing the two sample groups we are going to compare

KEGG Pathways

Getting pathways is quite easy thanks to `graphite`.

```
kpaths <- pathways("hsapiens", "kegg")
```

The above command retrieves all KEGG pathways for *Homo sapiens* (301 in total). We take a peek at the first entry:

```
kpaths[[1]]
```

```
## "Glycolysis / Gluconeogenesis" pathway
## Native ID       = hsa:00010
## Database        = KEGG
## Species         = hsapiens
## Number of nodes = 94
## Number of edges = 1191
## Retrieved on    = 27-04-2018
## URL             = http://www.kegg.jp/kegg-bin/show_pathway?org_name=hsa&mapno=00010
```

This summary doesn't tell us which identifiers are used for the nodes in the pathway. We can get that from the list of edges.

```
head(edges(kpaths[[1]], "mixed"))
```

```
##   src_type    src dest_type   dest direction    type
## 1 ENTREZID 130589  KEGGCOMP C00221  directed Process
## 2 ENTREZID 130589  KEGGCOMP C00267  directed Process
## 3 ENTREZID 160287  KEGGCOMP C00022  directed Process
## 4 ENTREZID   1737  KEGGCOMP C15973  directed Process
## 5 ENTREZID   2023  KEGGCOMP C00631  directed Process
## 6 ENTREZID   2026  KEGGCOMP C00631  directed Process
```

As we might expect, KEGG pathways are using KEGG compounds IDs. Since we're relying on CAS in our data, we should convert them. `convertIdentifiers` to the rescue!

```
rpaths <- convertIdentifiers(convertIdentifiers(kpaths, "ENSEMBL"), "CAS")
```

The edges of the first pathway have been converted into:

```
head(edges(rpaths[[1]], "mixed"))
```

```
##   src_type     src dest_type            dest  direction    type
## 1      CAS 57-60-3   ENSEMBL ENSG00000166800   directed Process
## 2      CAS 57-60-3   ENSEMBL ENSG00000166800 undirected Process
## 3      CAS 57-60-3   ENSEMBL ENSG00000134333   directed Process
## 4      CAS 57-60-3   ENSEMBL ENSG00000134333 undirected Process
## 5      CAS 57-60-3   ENSEMBL ENSG00000111716   directed Process
## 6      CAS 57-60-3   ENSEMBL ENSG00000111716 undirected Process
```

To make the rest of the analysis more robust, we are going to filter pathways requiring that 1) metabolites represent at least 30% of the nodes and 2) at least 10 edges connect genes or metabolites for which we have experimental measurements.

```
filter_pathways <- function(pathways, expr, min_edge_num) {
  node_names <- rownames(expr)

  pred <- function(p) {
    ns <- nodes(p, "mixed")
```

```
    if (length(ns) == 0) {
      return(FALSE)
    }

    frac <- length(grep("^CAS:", ns)) / length(ns)
    if (frac < 0.3) {
      return(FALSE)
    }

    es <- edges(p, "mixed")
    mask <- (paste(es$src_type, es$src, sep = ":") %in% node_names) &
            (paste(es$dest_type, es$dest, sep = ":") %in% node_names)
    sum(mask) >= min_edge_num
  }

  Filter(pred, pathways)
}

fpaths <- filter_pathways(rpaths, fexpr, 10)
```

Now about sample classes. We are going to split our dataset into two: `POS TUMOR` samples in the first group and `NEG TUMOUR` samples in the other. `clipper` wants from us a vector with as many entries as there are samples. An entry should be `1` if the corresponding samples belongs to the first class, or `2` in the other case.

```
pos_indices <- grep("^POS", colnames(fexpr))
neg_indices <- grep("^NEG", colnames(fexpr))

classes <- numeric(length = ncol(fexpr))
classes[neg_indices] <- 1
classes[pos_indices] <- 2
classes
```

```
##   [1] 2 2 2 2 1 1 1 2 2 2 2 1 1 1 1 1 1 2 2 2 2 2 1 1 1 1 1 1 1 2 2 2 2 2
##  [36] 2 2 2 2 2 1 1 1 1 1 1 1 2 1 1 2 2 2 2 1 1 2 2 2 2 2 1 1 1 2 1 1 1 1 1
##  [71] 2 2 2 2 1 1 2 2 2 2 1 1 1 1 1 1 2 1 1 2 2 1 1 1 1 1 1 1 2 2 2 2 2 2 2
## [106] 2 2 2
```

We use the helper function `runClipper` to start our analysis. Note that we explicitly require metabolites from the pathways.

```
out <- runClipper(fpaths, fexpr, classes, "mean", "mixed",
                  maxNodes = 150, seed = 42)
```

We check that there are no errors and the we extract the list of pathways which appears to be significantly altered between the two conditions according to `clipper`.

```
names(out$results)
```

```
##  [1] "Citrate cycle (TCA cycle)"
##  [2] "Fatty acid biosynthesis"
##  [3] "Arginine biosynthesis"
##  [4] "Alanine, aspartate and glutamate metabolism"
##  [5] "Cysteine and methionine metabolism"
##  [6] "Taurine and hypotaurine metabolism"
##  [7] "D-Glutamine and D-glutamate metabolism"
##  [8] "Amino sugar and nucleotide sugar metabolism"
##  [9] "Glyoxylate and dicarboxylate metabolism"
```

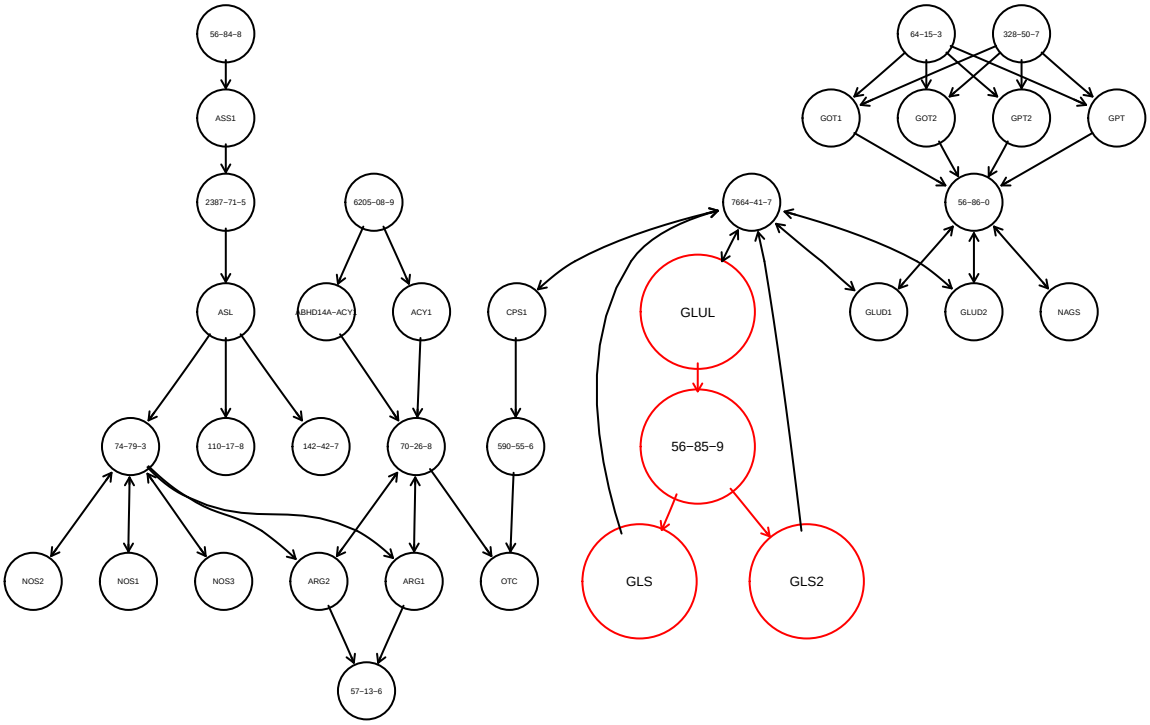
```
## [10] "Ferroptosis"
```

We use an helper function to visualize the results.

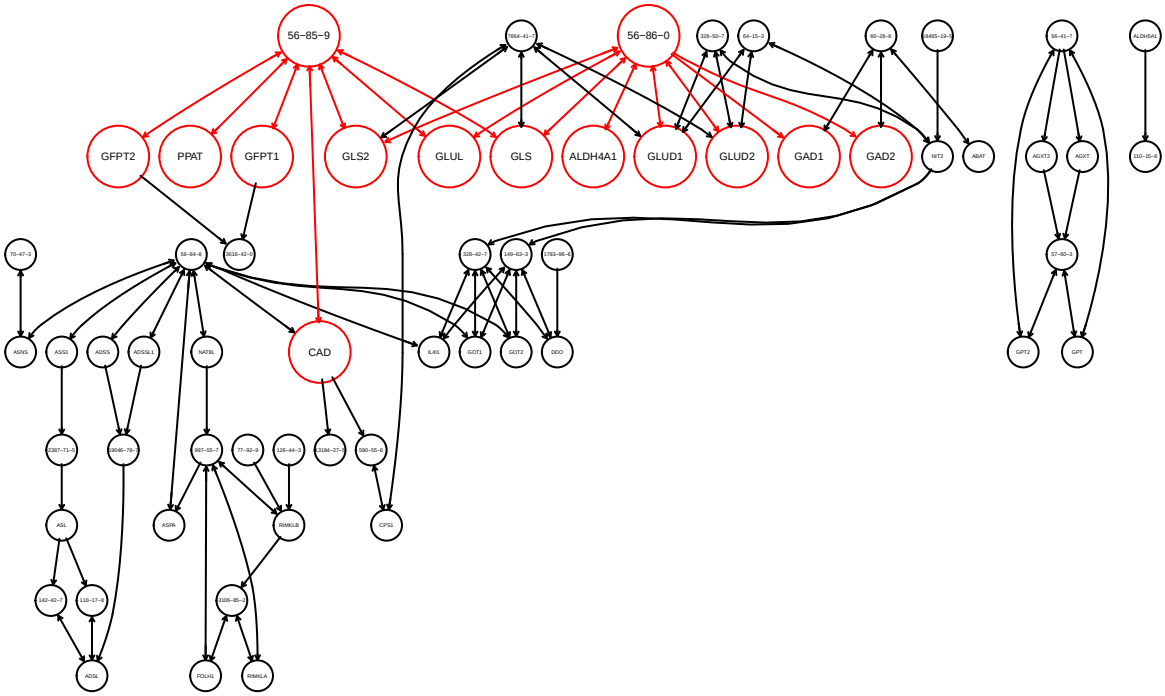
```
library(org.Hs.eg.db)

plot_altered_path <- function(result, pathways, node_scale = 2) {
  title <- names(result)
  pathway <- pathways[[title]]

  graph <- pathwayGraph(pathway, which = "mixed")

  labels <- sapply(nodes(graph), function(n) {
    n2 <- sub("^CAS:", "", n)
    if (n != n2) {
      n2
    } else {
      mapIds(org.Hs.eg.db, sub("^ENSEMBL:", "", n), "SYMBOL", "ENSEMBL",
             multiVas = "first")
    }
  })
  names(labels) <- nodes(graph)

  altered <- unlist(strsplit(result[[1]][1, "pathGenes"], "[,;]"))
  selected <- nodes(graph) %in% altered
  node_colors <- ifelse(selected, "red", "black")
  names(node_colors) <- nodes(graph)

  base <- 0.75
  heights <- ifelse(selected, base * node_scale, base)
  names(heights) <- nodes(graph)
  widths <- ifelse(selected, base * node_scale, base)
  names(widths) <- nodes(graph)

  base <- 14
  fontsizes <- ifelse(selected, base * node_scale, base)
  names(fontsizes) <- nodes(graph)

  between_altered <- function(edge_names, altered) {
    sapply(edge_names, function(edge_name) {
      nodes <- unlist(strsplit(edge_name, "~", fixed = TRUE))
      all(nodes %in% altered)
    })
  }

  edge_colors <- ifelse(between_altered(edgeNames(graph), altered),
                        "red", "black")

  plot(graph,
       attrs = list(edge = list(arrowsize = 0.5)),
       nodeAttrs = list(label = labels, color = node_colors, width = widths,
                        height = heights, fontsize = fontsizes),
       edgeAttrs = list(color = edge_colors),
       recipEdges = "combined", main = title)
}
```

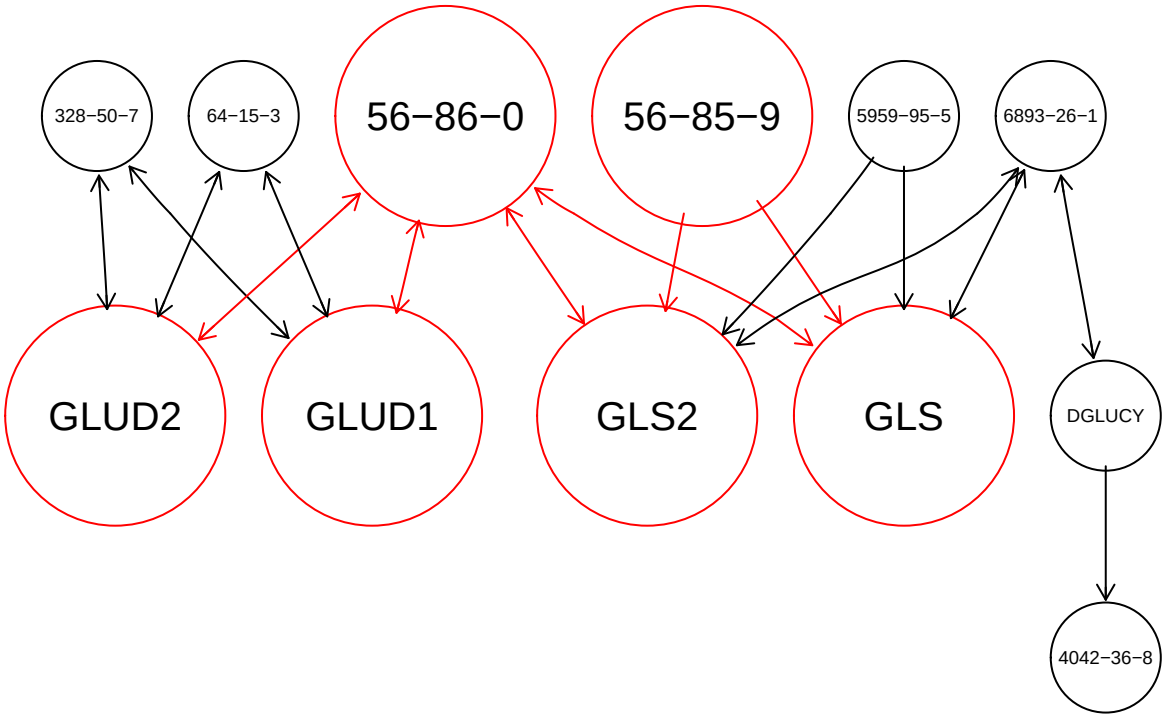
```
selection <- c(
  "Alanine, aspartate and glutamate metabolism",
  "Arginine biosynthesis",
  "D-Glutamine and D-glutamate metabolism"
)
for (idx in seq_along(out$results)) {
  res <- out$results[idx]
  if (names(res) %in% selection) {
    plot_altered_path(res, fpaths)
  }
}
```

Arginine biosynthesis



Alanine, aspartate and glutamate metabolism





graphite analyses highlight "Alanine, aspartate and glutamate metabolism" and "Arginine biosynthesis". These pathways enclose most of the genes and metabolites discussed in Terunuma et al. that follow the abundance pattern of 2HG such as N-acetyl amino acids, especially the mitochondrial N-acetyl-aspartate (NAA), the glutaminase (GLS1) protein, and the aspartoacylase (ASPA), which is generally reduced in breast tumors but most significantly within the ER-negative tumor group which suggests a possible cause for increased tumor-associated NAA. Another graphite pathway worthy of note is "D-Glutamine and D-glutamate metabolism" reported also in Terunuma et al. because it contains L-Glutamine (CAS:56-85-9) and L-Glutamic Acid (CAS:56-86-0), two of the most altered metabolites between ER+ and ER- breast tumours.

SMPDB Pathways

```
spaths <- pathways("hsapiens", "smpdb")
```

```
fpaths <- filter_pathways(
  convertIdentifiers(convertIdentifiers(spaths, "ENSEMBL"), "CAS"),
  fexpr,
  10)
```

```
out <- runClipper(fpaths, fexpr, classes, "mean", "mixed",
                  maxNodes = 150, seed = 42)
```

```
names(out$results)
```

```
##   [1] "Citrullinemia Type I"
##   [2] "Carbamoyl Phosphate Synthetase Deficiency"
##   [3] "Argininosuccinic Aciduria"
```

```
##   [4] "Glycine and Serine Metabolism"
##   [5] "Ammonia Recycling"
##   [6] "Arginine and Proline Metabolism"
##   [7] "Methionine Metabolism"
##   [8] "Sphingolipid Metabolism"
##   [9] "Lysine Degradation"
##  [10] "Amino Sugar Metabolism"
##  [11] "Pyrimidine Metabolism"
##  [12] "Purine Metabolism"
##  [13] "Citric Acid Cycle"
##  [14] "Urea Cycle"
##  [15] "Pyruvate Metabolism"
##  [16] "Tryptophan Metabolism"
##  [17] "Glutamate Metabolism"
##  [18] "2-Hydroxyglutric Aciduria (D And L Form)"
##  [19] "Adenosine Deaminase Deficiency"
##  [20] "Adenylosuccinate Lyase Deficiency"
##  [21] "AICA-Ribosiduria"
##  [22] "Beta Ureidopropionase Deficiency"
##  [23] "Cystathionine Beta-Synthase Deficiency"
##  [24] "Dihydropyrimidinase Deficiency"
##  [25] "Dihydropyrimidine Dehydrogenase Deficiency (DHPD)"
##  [26] "Glutaric Aciduria Type I"
##  [27] "Guanidinoacetate Methyltransferase Deficiency (GAMT Deficiency)"
##  [28] "Leigh Syndrome"
##  [29] "MNGIE (Mitochondrial Neurogastrointestinal Encephalopathy)"
##  [30] "Molybdenum Cofactor Deficiency"
##  [31] "Ornithine Transcarbamylase Deficiency (OTC Deficiency)"
##  [32] "Prolidase Deficiency (PD)"
##  [33] "Prolinemia Type II"
##  [34] "Purine Nucleoside Phosphorylase Deficiency"
##  [35] "Pyruvate Dehydrogenase Complex Deficiency"
##  [36] "S-Adenosylhomocysteine (SAH) Hydrolase Deficiency"
##  [37] "Sialuria or French Type Sialuria"
##  [38] "Sialuria or French Type Sialuria"
##  [39] "UMP Synthase Deficiency (Orotic Aciduria)"
##  [40] "Xanthine Dehydrogenase Deficiency (Xanthinuria)"
##  [41] "Methionine Adenosyltransferase Deficiency"
##  [42] "Glycine N-methyltransferase Deficiency"
##  [43] "Non Ketotic Hyperglycinemia"
##  [44] "Saccharopinuria/Hyperlysinemia II"
##  [45] "Salla Disease/Infantile Sialic Acid Storage Disease"
##  [46] "Dimethylglycine Dehydrogenase Deficiency"
##  [47] "4-Hydroxybutyric Aciduria/Succinic Semialdehyde Dehydrogenase Deficiency"
##  [48] "Sarcosinemia"
##  [49] "Pyruvate Decarboxylase E1 Component Deficiency (PDHE1 Deficiency)"
##  [50] "Hyperinsulinism-Hyperammonemia Syndrome"
##  [51] "Methylenetetrahydrofolate Reductase Deficiency (MTHFRD)"
##  [52] "Hypermethioninemia"
##  [53] "Metachromatic Leukodystrophy (MLD)"
##  [54] "Globoid Cell Leukodystrophy"
##  [55] "Gaucher Disease"
##  [56] "Argininemia"
##  [57] "Hyperprolinemia Type II"
```

```
##  [58] "Hyperprolinemia Type I"
##  [59] "Arginine: Glycine Amidinotransferase Deficiency (AGAT Deficiency)"
##  [60] "Ornithine Aminotransferase Deficiency (OAT Deficiency)"
##  [61] "Lesch-Nyhan Syndrome (LNS)"
##  [62] "Gout or Kelley-Seegmiller Syndrome"
##  [63] "Homocarnosinosis"
##  [64] "Tay-Sachs Disease"
##  [65] "Azathioprine Action Pathway"
##  [66] "Mercaptopurine Action Pathway"
##  [67] "Thioguanine Action Pathway"
##  [68] "Dimethylglycine Dehydrogenase Deficiency"
##  [69] "Hyperglycinemia, non-ketotic"
##  [70] "Creatine deficiency, guanidinoacetate methyltransferase deficiency"
##  [71] "Hyperornithinemia with gyrate atrophy (HOGA)"
##  [72] "Hyperornithinemia-hyperammonemia-homocitrullinuria [HHH-syndrome]"
##  [73] "L-arginine:glycine amidinotransferase deficiency"
##  [74] "Xanthinuria type I"
##  [75] "Xanthinuria type II"
##  [76] "Fabry disease"
##  [77] "Krabbe disease"
##  [78] "Hyperlysinemia I, Familial"
##  [79] "Hyperlysinemia II or Saccharopinuria"
##  [80] "G(M2)-Gangliosidosis: Variant B, Tay-sachs disease"
##  [81] "Adenine phosphoribosyltransferase deficiency (APRT)"
##  [82] "Mitochondrial DNA depletion syndrome"
##  [83] "Myoadenylate deaminase deficiency"
##  [84] "Congenital lactic acidosis"
##  [85] "Fumarase deficiency"
##  [86] "Mitochondrial complex II deficiency"
##  [87] "2-ketoglutarate dehydrogenase complex deficiency"
##  [88] "Pyruvate dehydrogenase deficiency (E3)"
##  [89] "Pyruvate dehydrogenase deficiency (E2)"
##  [90] "Primary hyperoxaluria II, PH2"
##  [91] "Pyruvate kinase deficiency"
##  [92] "Succinic semialdehyde dehydrogenase deficiency"
##  [93] "Homocystinuria-megaloblastic anemia due to defect in cobalamin metabolism, cblG complementati
##  [94] "Pyridoxine dependency with seizures"
##  [95] "Warburg Effect"
##  [96] "2-aminoadipic 2-oxoadipic aciduria"
##  [97] "3-Phosphoglycerate dehydrogenase deficiency"
##  [98] "The oncogenic action of 2-hydroxyglutarate"
##  [99] "The Oncogenic Action of Succinate"
## [100] "The Oncogenic Action of Fumarate"
## [101] "Glutaminolysis and Cancer"
## [102] "Sarcosine Oncometabolite Pathway"
## [103] "The oncogenic action of L-2-hydroxyglutarate in  Hydroxygluaricaciduria"
## [104] "The oncogenic action of D-2-hydroxyglutarate in  Hydroxygluaricaciduria"
## [105] "Chlorphenamine H1-Antihistamine Action"
## [106] "Pheniramine H1-Antihistamine Action"
## [107] "Dexchlorpheniramine H1-Antihistamine Action"
## [108] "Brompheniramine H1-Antihistamine Action"
## [109] "Dexbrompheniramine H1-Antihistamine Action"
## [110] "Triprolidine H1-Antihistamine Action"
## [111] "Dimetindene H1-Antihistamine Action"
```

```
## [112] "Mepyramine H1-Antihistamine Action"
## [113] "Antazoline H1-Antihistamine Action"
## [114] "Chloropyramine H1-Antihistamine Action"
## [115] "Talastine H1-Antihistamine Action"
## [116] "Tripelennamine H1-Antihistamine Action"
## [117] "Histapyrrodine H1-Antihistamine Action"
## [118] "Methapyrilene H1-Antihistamine Action"
## [119] "Thonzylamine H1-Antihistamine Action"
## [120] "Diphenhydramine H1-Antihistamine Action"
## [121] "Carbinoxamine H1-Antihistamine Action"
## [122] "Doxylamine H1-Antihistamine Action"
## [123] "Orphenadrine H1-Antihistamine Action"
## [124] "Bromodiphenhydramine H1-Antihistamine Action"
## [125] "Clemastine H1-Antihistamine Action"
## [126] "Chlorphenoxamine H1-Antihistamine Action"
## [127] "Diphenylpyraline H1-Antihistamine Action"
## [128] "Phenyltoloxamine H1-Antihistamine Action"
## [129] "Cyclizine H1-Antihistamine Action"
## [130] "Chlorcyclizine H1-Antihistamine Action"
## [131] "Hydroxyzine H1-Antihistamine Action"
## [132] "Meclizine H1-Antihistamine Action"
## [133] "Buclizine H1-Antihistamine Action"
## [134] "Oxatomide H1-Antihistamine Action"
## [135] "Cetirizine H1-Antihistamine Action"
## [136] "Cinnarizine H1-Antihistamine Action"
## [137] "Levocetirizine H1-Antihistamine Action"
## [138] "Promethazine H1-Antihistamine Action"
## [139] "Alimemazine H1-Antihistamine Action"
## [140] "Cyproheptadine H1-Antihistamine Action"
## [141] "Phenbenzamine H1-Antihistamine Action"
## [142] "Fenethazine H1-Antihistamine Action"
## [143] "Hydroxyethylpromethazine H1-Antihistamine Action"
## [144] "Isothipendyl H1-Antihistamine Action"
## [145] "Mequitazine H1-Antihistamine Action"
## [146] "Methdilazine H1-Antihistamine Action"
## [147] "Oxomemazine H1-Antihistamine Action"
## [148] "Azatadine H1-Antihistamine Action"
## [149] "Ketotifen H1-Antihistamine Action"
## [150] "Doxepin H1-Antihistamine Action"
## [151] "Acrivastine H1-Antihistamine Action"
## [152] "Astemizole H1-Antihistamine Action"
## [153] "Bepotastine H1-Antihistamine Action"
## [154] "Bilastine H1-Antihistamine Action"
## [155] "Loratadine H1-Antihistamine Action"
## [156] "Desloratadine H1-Antihistamine Action"
## [157] "Ebastine H1-Antihistamine Action"
## [158] "Terfenadine H1-Antihistamine Action"
## [159] "Fexofenadine H1-Antihistamine Action"
## [160] "Levocabastine H1-Antihistamine Action"
## [161] "Mizolastine H1-Antihistamine Action"
## [162] "Rupatadine H1-Antihistamine Action"
## [163] "Olopatadine H1-Antihistamine Action"
## [164] "Azelastine H1-Antihistamine Action"
## [165] "Thiazinamium H1-Antihistamine Action"
```

```
## [166] "Quifenadine H1-Antihistamine Action"
## [167] "Betahistine H1-Antihistamine Action"
## [168] "Emedastine H1-Antihistamine Action"
## [169] "Flunarizine H1-Antihistamine Action"
## [170] "Mebhydrolin H1-Antihistamine Action"
## [171] "Phenindamine H1-Antihistamine Action"
## [172] "Epinastine H1-Antihistamine Action"
## [173] "Tolpropamine H1-Antihistamine Action"
## [174] "Embramine H1-Antihistamine Action"
## [175] "Latrepirdine H1-Antihistamine Action"
## [176] "Thenyldiamine H1-Antihistamine Action"
## [177] "Propiomazine H1-Antihistamine Action"
## [178] "Clocinizine H1-Antihistamine Action"
## [179] "Homochlorcyclizine H1-Antihistamine Action"
## [180] "Temelastine H1-Antihistamine Action"
## [181] "Alcaftadine H1-Antihistamine Action"
## [182] "Bamipine H1-Antihistamine Action"
## [183] "Deptropine H1-Antihistamine Action"
## [184] "Quetiapine H1-Antihistamine Action"
## [185] "Mirtazapine H1-Antihistamine Action"
## [186] "Pimethixene H1-Antihistamine Action"
## [187] "Pyrrobutamine H1-Antihistamine Action"
## [188] "Thenalidine H1-Antihistamine Action"
## [189] "Tritoqualine H1-Antihistamine Action"
## [190] "Histamine H1 Receptor Activation"
```

Using the SMPDB database we obtained 190 significant pathways among which we found “The oncogenic action of 2-hydroxyglutarate” that is the pathway describing the role of 2HG in tumour environment.