Working with substrate information in opm

Lea A.I. Vaas
Leibniz Institute DSMZ

Johannes Sikorski
Leibniz Institute DSMZ

Markus Göker
Leibniz Institute DSMZ

Abstract

This is the substrate-information tutorial of **opm** in the version of October 12, 2013. The precomputed information on the known Phenotype Microarray (PM) substrates is explained, as well as the methods available to query this information. IDs for a variety of databases are stored within **opm** and can be used to conduct web queries to obtain comprehensive information on the substrates of interest. We show how these data can be used to visualize results from PM experiments, including the outcome from advanced multiple-comparison statistics, within biochemical pathway maps. Moreover, methods are described to automatically detect the substrate features that potentially explain a given experimental outcome. This includes determining the relevant pathways to be used in the visualizations.

Keywords: Respiration Kinetics, pathways, CAS, MeSH, ChEBI, Metacyc, KEGG, **pathview**.

1. Introduction

A detailed description of the OmniLog® Phenotype MicroArray (PM) system, its measuring procedure and data characteristics are found in the vignette "**opm**: An R Package for Analysing OmniLog® Phenotype MicroArray Data" (called "main tutorial" in the following). The description of the methods below presupposes that the user is familiar with the usage of **opm** and has studied the main tutorial as well as the entries of the **opm** manual relevant to her or his research. Especially the concepts behind, and the methods available for, the different classes of **opm** objects should be known before starting with this tutorial.

In addition to visual inspection or statistical comparative analyses of PM data, as described in the main tutorial, users might be interested in specific information on the substrates used in PM assays. The **opm** package contains a large variety of additional data on PM substrates. Beside methods for assessing this information directly, this tutorial introduces strategies for visualizing the measured PM results by mapping them on pathway maps. Furthermore, analysis methods are described for modelling the effect of substrate features on the PM results and thus, e.g., for the identification of those pathways that are particularly suitable for visualizing the PM results in pathway graphs.

2. Preparation

For just downloading information from KEGG (see Section 3.1), install the Bioconductor package **KEGGREST**. It needs not be loaded into your R session. For also drawing PM information into KEGG pathway maps (see Section 4), install the Bioconductor package **pathview** and load

it into your session. Note that it is important to load **pathview** before **opm**, which is needed throughout this tutorial, since otherwise some methods are not visible and the package does not work properly. In this vignette this is enforced by optionally detaching **opm** and loading it (again) as follows:

```
R> suppressPackageStartupMessages(library("pathview"))
R> if ("package:opm" %in% search())
     detach("package:opm")
R> library("opm")
R> data(vaas_et_al, package = "opmdata")
```

3. Accessing plate and substrate information

The **opm** package contains a number of functions suitable for accessing precomputed information on entire plates and on the substrates within certain wells.

3.1. Available plate information

Currently substrate layouts of various plates are available within **opm**. An overview of the plate types available in the respective version of **opm** is obtained by entering

```
R> plate_type(full = TRUE)
```

The resulting vector of names does not only include OmniLog® plates; see the manual and the main tutorial for further details. Using other values for `full`, or additional arguments, distinct spelling variants of the plate names can be obtained.

3.2. Available substrate information

In the manual and help pages these functions are explained within the family "naming-functions" with according cross-references.

One usually would start a search by determining the exact spelling of an internally used name with `find_substrate()`:

```
R> substrates <- find_substrate(c("Glutamine", "Glutamic acid"))
R> substrates
```

The result is a list (of the S3 class "substrate_match") containing character vectors with the results for each query name as values. Surprisingly, nothing was found for "Glutamic acid" but several values for "Glutamine". The default `search` argument is "exact", which is exact (case-sensitive) matching of *substrings* of the names. One might want to use "glob" searching mode:

```
R> substrates <- find_substrate(c("L-Glutamine", "L-Glutamic acid"), "glob")
R> substrates
```

But with so-called wild-cards, i.e. "*" for zero to many and "?" for a single arbitrary character the search is more flexible:

```
R> substrates <- find_substrate(c("*L-Glutamine", "*L-Glutamic acid"), "glob")
R> substrates
```

This fetches all terms that end in either query character string, and does so case-insensitively. Advanced users can apply the much more powerful "regex" and "approx" search modes; see the manual for details, entry `?find_substrate`.

Note that **opm** appends a concentration (or just repetition) indicator as a number after a hash sign ("#") to the substrate names wherever necessary. Thus a wild-card "*" at the end of a name might often by the most useful search pattern.

Once the internally used names (which are not guaranteed to be stable between distinct **opm** releases) have been found, information on the substrates can be queried such as their occurrences and positions on plates:

```
R> positions <- find_positions(substrates)
R> positions
```

This yields a nested list containing two-column matrices with plate names in the first and well coordinates in the second column. References to external data resources for each substrate name can be obtained using `substrate_info()`:

```
R> subst.info <- substrate_info(substrates)
R> subst.info
```

By default this yields CAS numbers (http://www.cas.org/content/chemical-substances/faqs), but MeSH names (useful for conducting PubMed queries; see http://www.ncbi.nlm.nih.gov/mesh/) (Coletti and Bleich 2001), ChEBI IDs (Hastings, de Matos, Dekker, Ennis, Harsha, Kale, Muthukrishnan, Owen, Turner, Williams, and Steinbeck 2013), KEGG compound IDs, KEGG drug IDs (Kanehisa, Goto, Furumichi, Tanabe, and Hirakawa 2010) and MetaCyc IDs (Caspi, Altman, Dreher, Fulcher, Subhraveti, Keseler, Kothari, Krummenacker, Latendresse, Mueller, Ong, Paley, Pujar, Shearer, Travers, Weerasinghe, Zhang, and Karp 2012) have also been collected for the majority of the substrates. Using the "browse" argument, full URLs can be created and optionally also directly opened in the default web browser. Using the "download" argument, if KEGG drug or compound IDs have been selected, these can be downloaded from the KEGG server if the **KEGGREST** is available and converted into customized objects. It is possible to nicely display all available information at once:

```
R> subst.info <- substrate_info(substrates, "all")
R> subst.info
```

Another use of `substrate_info()` is to convert substrate names to lower case but protecting name components such as abbreviations or chemical symbols. See the manual for further details, help page `?substrate_info`.

4. Visualisation of PM information within pathway maps

In conjunction with other R packages, it is possible to visualize PM results directly in pre-existing pathway maps as, for example, those from the KEGG database. These maps are essentially manually drawn biochemical pathway maps representing the currently available knowledge on substrates, enzymes and genes and their connections within pathways. Depending on the availability of genome and gene-annotation information within KEGG, organism-specific, individual maps can be obtained (Kanehisa *et al.* 2010).

The mapping itself works by using information produced by **opm** for a colour coding of the nodes (here, representing the substrates) within those maps, as can be done similarly with several other types of "OMICS" data such as transcriptomics or proteomics data. For details, see the description on the KEGG website (`http://www.genome.jp/kegg/`).

4.1. Providing suitable input data

The work flow starts with either an `OPMX` object containing the aggregated values or the result from an opm_mcp analysis. The first step in both cases is to convert the data into a suitable format, which is a named vector created by the function `annotated`.

```
R> x <- annotated(vaas_1)
R> head(x)
```

```
    <NA>    C00721    C00208    C01083    C00185    C08240
123.4558 248.1809 284.0994 269.7548 180.7536 287.7959
```

The resulting vector contains the numeric values (selected parameter estimates or `opm_mcp` results, as explained below) as well as an annotation of the according substrates. For substrates such as "Positive Control" or "pH 5" no KEGG ID is available, which results in `NA` values in the vector. Accordingly, those substrates cannot be marked within pathway maps (see section 4.2.1). The `what` argument, passed as eponymous argument to `substrate_info`, selects the kind of information to be used for the annotation.

Although `annotated` works directly on `OPMX` objects containing aggregated data for single plates or bundles of plates, please note, that the output allows for only one value per substrate. Thus, when applying `annotated` to a set of plates, make sure that only one experimental group is comprised, since the resulting values are averaged per well over all plates in the input object. Using the `output` argument, one can select the parameter of interest, for example area under the curve instead of maximum height:

```
R> x <- annotated(vaas_1, output = param_names()[4])
R> head(x)
```

```
     <NA>     C00721     C00208     C01083     C00185     C08240
 8918.137 18391.590 21960.080 18531.180 11831.150 19254.160
```

Visualization of the results of an opm_mcp analysis is also possible, which offers more (statistically interesting) opportunities for making sense of the PM data in the context of pathways. However, this method only makes sense if each coefficient estimated by `opm_mcp` can be linked

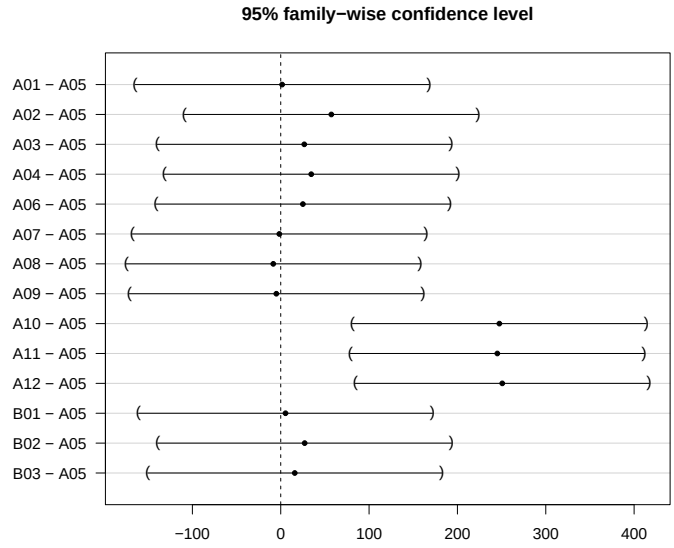


Figure 1: Point estimates and 95% confidence intervals in a manually defined comparison of group means for a specifically selected set of wells from the `vaas_4` exemplar object. In this procedure each selected well is compared against A05. The picture was obtained by running `opm_mcp()` and then the plotting function for the resulting "opm_glht" object. See the main tutorial for details.

to a single substrate. This is usually only possible for the "Dunnett" and "Pairs" type of contrasts if applied to the wells (see Section 4.2.3). See the main tutorial and the manual for details on this restriction.

The results from an `opm_mcp` procedure are treated with `annotated` as shown before with an `OPMX` object, but additional options are available. In the following example, first an "opm_glht" object is generated from the `vaas_4` exemplar object by performing a Dunnett-type multiple comparison of the selected 13 wells against well A05 as control group.

```
R> x <- opm_mcp(vaas_4[, , 1:15],
   output = "mcp", model = ~ Well,
     linfct = c(Dunnett.A05 = 1), full = FALSE)
```

The resulting 95% confidence intervals for the difference of means are plotted in Figure 1.

Using the above generated "opm_glht" object, the options modifying the output of `annotated` will be illustrated. Apparently only three comparisons exhibit a statistically significant difference, namely the comparisons A10 - A05, A11 - A05 and A12 - A05, all showing that the reactions in A05 are weaker than those in A10, A11 and A12, respectively.

Using the `output` argument, users are able to obtain various statistically relevant categorical results instead of the simple numerical output of the respective point estimator. The options `upwards` and `downwards` result in a classification into three categories (`FALSE`, `NA`, or `TRUE`). These indicate whether or not the cut-off (zero per default) is included in the confidence

interval (`NA`) and thus a decision possible. If not, the category indicates the direction of the shift relative to the cut-off. The options `different`, `smaller`, `larger` and `equal` work similarly, but use only the two categories `TRUE` and `FALSE`. Short-cuts are available for all options, enabling the user to set the cut-off together with the kind of output. See the manual for details.

A comprehensive overview of the possible results for object `x` is shown in the following data frame:

```
   Comparison    numeric upwards downwards different equal smaller larger
1   A01 - A05   1.577317      NA        NA     FALSE  TRUE   FALSE  FALSE
2   A02 - A05  57.274661      NA        NA     FALSE  TRUE   FALSE  FALSE
3   A03 - A05  26.661023      NA        NA     FALSE  TRUE   FALSE  FALSE
4   A04 - A05  34.537328      NA        NA     FALSE  TRUE   FALSE  FALSE
5   A06 - A05  25.078779      NA        NA     FALSE  TRUE   FALSE  FALSE
6   A07 - A05  -1.606236      NA        NA     FALSE  TRUE   FALSE  FALSE
7   A08 - A05  -8.458879      NA        NA     FALSE  TRUE   FALSE  FALSE
8   A09 - A05  -4.975284      NA        NA     FALSE  TRUE   FALSE  FALSE
9   A10 - A05 247.470724    TRUE     FALSE      TRUE FALSE   FALSE   TRUE
10  A11 - A05 245.163382    TRUE     FALSE      TRUE FALSE   FALSE   TRUE
11  A12 - A05 250.763650    TRUE     FALSE      TRUE FALSE   FALSE   TRUE
12  B01 - A05   5.417463      NA        NA     FALSE  TRUE   FALSE  FALSE
13  B02 - A05  27.079437      NA        NA     FALSE  TRUE   FALSE  FALSE
14  B03 - A05  15.870312      NA        NA     FALSE  TRUE   FALSE  FALSE
```

All these results are obtained with the setting `how = "ids"`; for the usage of `how = "value"` see Section 5.

4.2. Visualisation in pathway maps using pathview

4.2.1. Visualisation of group means in pathway maps

Here we will use the function `pathview` from the package of the same name (Luo and Brouwer 2013). This function can download a user-defined pathway graph from KEGG, optionally integrate additional data from other sources, and render the result. For integrating experimental data from other "OMICS" approaches (such as transcriptomics and proteomics), see the corresponding chapter in the **pathview** vignette for details.

Here the `pathview` function serves for integrating and visualizing information produced by **opm** and provided by `annotated`. The user only has to specify the pathway and provide the **opm** data. All other necessary steps (download of pathway graph data as XML file from KEGG, parsing of this data file, integrating user-defined data into the pathway representation, and rendering of final output graphics) are automatically conducted by `pathview`. See the **pathview** vignette for technical details.

In the case of KEGG, a pathway map is described as "a molecular interaction/reaction network diagram represented in terms of the KEGG Orthology (KO) groups" (see `http://www.genome.jp/kegg/kegg3a.html` for further details). KEGG contains a collection of distinct types of pathway maps identified by a code containing between two and four letters as a prefix, followed by five digits.

The prefixes have the following meanings:

map - Reference pathway

ko - Reference pathway (KO)

ec - Reference pathway (EC)

rn - Reference pathway (Reaction)

org - Organism-specific pathway map (*org* is a wild-card for the organism-specific abbreviation composed of two to four letters)

Only the first reference pathway map is drawn manually; all other maps are computationally generated. The *ko*-maps contain the manually defined ortholog groups (*ko* entries) for all proteins and functional RNAs that correspond to KEGG pathway nodes, BRITE hierarchy nodes, and KEGG module nodes, whereas the (*ko* entries) are converted to gene identifiers if organism-specific pathways maps are generated. A list of the existing maps and their corresponding numbers are available on the KEGG homepage (see above).

`pathview` allows only for using KEGG orthology (the *ko* maps) or species-specific letter codes. See http://www.genome.jp/kegg/catalog/org_list.html for an up-to-date list of organisms with complete genome information in KEGG.

A vector as returned by `annotated` (see Section 4.1) serves as input for the visualization procedure based on `pathview`. For demonstration purposes, we use subsets of `vaas_et_al` containing the *Escherichia coli* strains from the first biological repetition.

```
R> e.coli.sub <- subset(vaas_et_al, list(Species = "Escherichia coli",
     Experiment = "First replicate"))
R> e.coli.k12 <- subset(e.coli.sub, list(Strain = "DSM18039"))
R> e.coli.type <- subset(e.coli.sub, list(Strain = "DSM30083T"))
```

Afterwards we create the annotated vectors containing the average maximum curve heights for the two groups separately:

```
R> anno.k12 <- annotated(e.coli.k12)
R> anno.type <- annotated(e.coli.type)
```

For a more convenient drawing of **opm** data on KEGG pathway maps, we suggest a wrapper for the `pathview` function, providing other default settings for some arguments. All graphics below are produced using this wrapper, but the user is of course free to use the original `pathview` function or write an alternative wrapper.

```
R> opm_path <- function(cpd.data, gene.data = NULL,
       high = list(gene = "green4", cpd = "blue"),
       mid = list(gene = "lightsteelblue1", cpd = "yellow"),
       low = list(gene = "white", cpd = "yellow"),
       species = "ko", out.suffix = "non-native",
       key.pos = "topright", afactor = 1000,
       limit = list(gene = 2, cpd = 400),
       bins = list(gene = 0.5, cpd = 40),
       both.dirs = list(gene = FALSE, cpd = FALSE),
```

```
    sign.pos = "topleft", cpd.lab.offset = 0,
    same.layer = FALSE,
    na.col = "white", ...) {
  pathview(cpd.data = cpd.data, gene.data = gene.data,
    high = high, mid = mid, low = low,
    species = species, out.suffix = out.suffix, key.pos = key.pos,
    afactor = afactor, limit = limit, bins = bins,
    both.dirs = both.dirs, sign.pos = sign.pos,
    cpd.lab.offset = cpd.lab.offset, same.layer = same.layer,
    na.col = na.col, ...)
}
```

The data for the two strains are shown on the correspondingly separated maps in Figure 2.

```
R> e.coli.map.k12 <- opm_path(cpd.data = anno.k12, species = "ko",
     out.suffix = "k12.ko", pathway.id = "00052")
R> e.coli.map.type <- opm_path(cpd.data = anno.type, species = "ko",
     out.suffix = "type.ko", pathway.id = "00052")
```

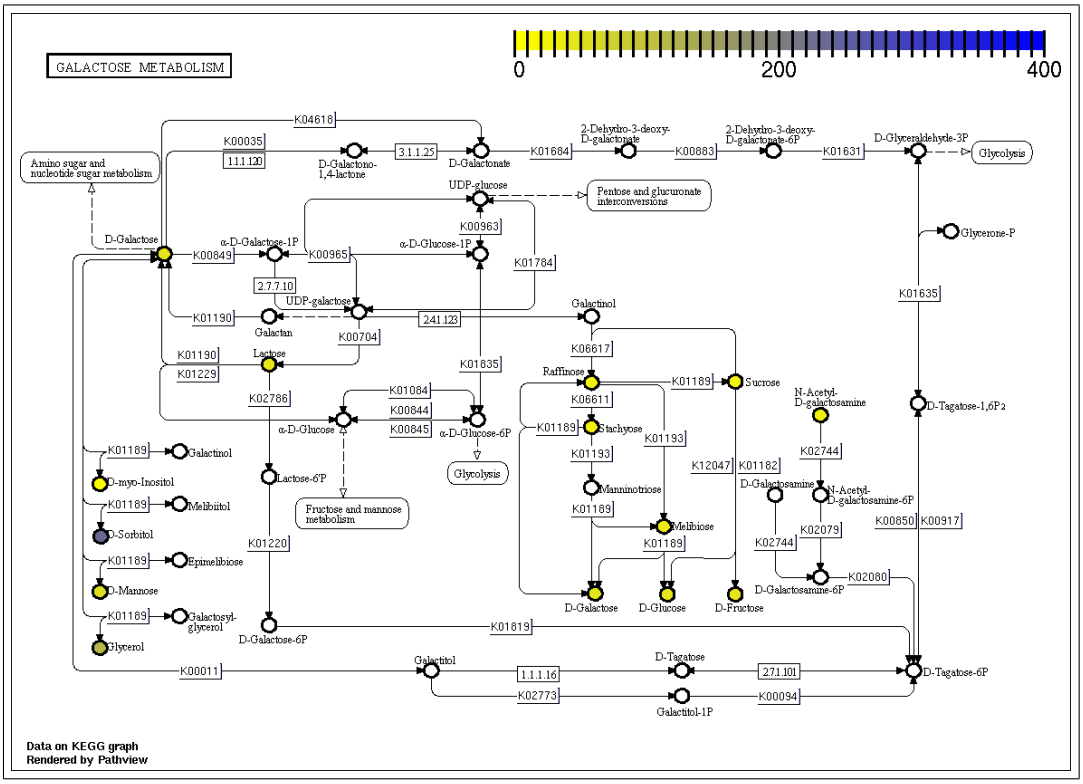
Note particularly the substrates Raffinose, Stachyose and Sucrose (in the middle of the map), which exhibit large respiratory differences, while Sorbitol (on the very left of the map) yields only small respiratory differences between the two strains.

Using the default settings, `pathview` yields a raster image in PNG format, which is stored in the current working directory and shown in Figure 2. For demonstration purposes the pathway number "00052", which encodes for the Galactose metabolism pathway map, is chosen. Genes (boxes) are annotated with KEGG ontology numbers (set by choosing `species = "ko"`), where available or, alternatively, with EC numbers. Note that the `species` arguments offers the possibility to use species-specific genome information available in the KEGG directory; see above for the letter codes. The substrates (circles) in the maps get standard compound names, which are automatically retrieved from the CHEMBL database using the compound IDs.

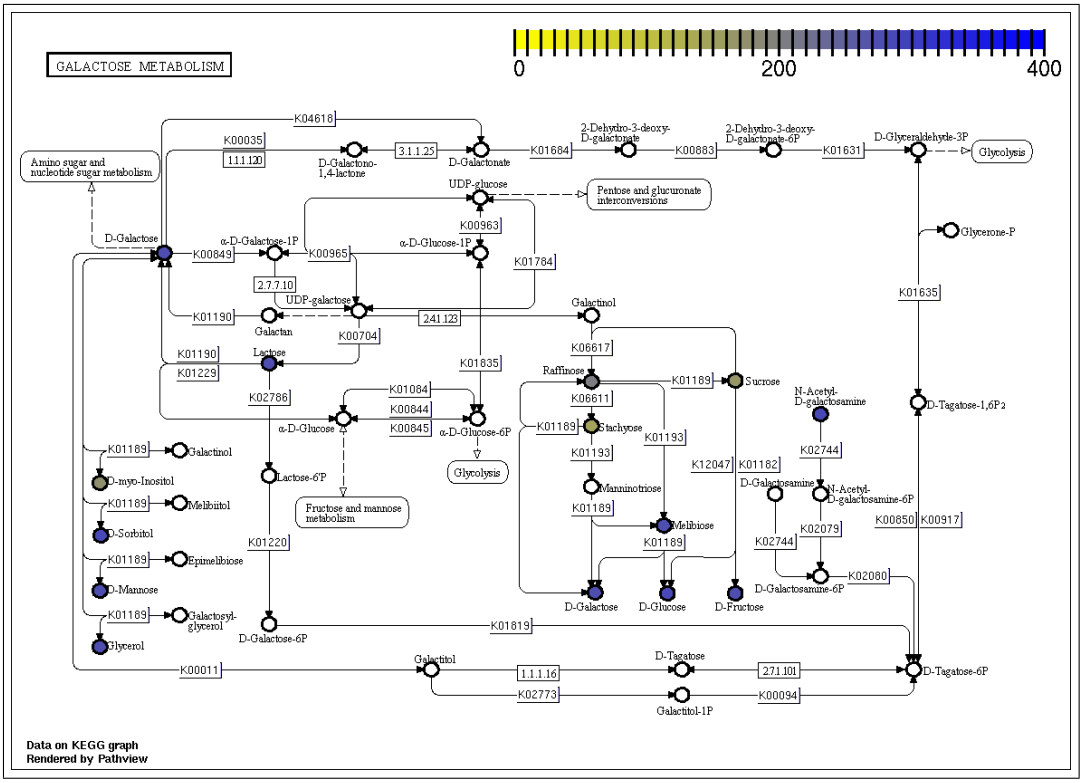
The data for the two strains can be shown analogously using the categorical output of `annotated`, see Figure 3. This works because the underlying OPMX object contains discretized data. Whereas `annotated` would by default return a logical vector in that case, the `lmap` argument can be used to create a numeric vector on the fly. See the manual for its usage, entry `?annotated`.

```
R> anno.k12.bin <- annotated(e.coli.k12, output = param_names("disc.name"),
     lmap = 1:3)
R> anno.type.bin <- annotated(e.coli.type, output = param_names("disc.name"),
     lmap = 1:3)
R> e.coli.map.k12.bin <- opm_path(cpd.data = anno.k12.bin, species = "ko",
     out.suffix = "k12.ko.bin", pathway.id = "00052",
     limit = list(gene = 2, cpd = c(1, 3)), bins = list(gene = 0.5, cpd = 3))
R> e.coli.map.type.bin <- opm_path(cpd.data = anno.type.bin, species = "ko",
     out.suffix = "type.ko.bin", pathway.id = "00052",
     limit = list(gene = 2, cpd = c(1, 3)), bins = list(gene = 0.5, cpd = 3))
```

4.2.2. Finding substrates within pathways

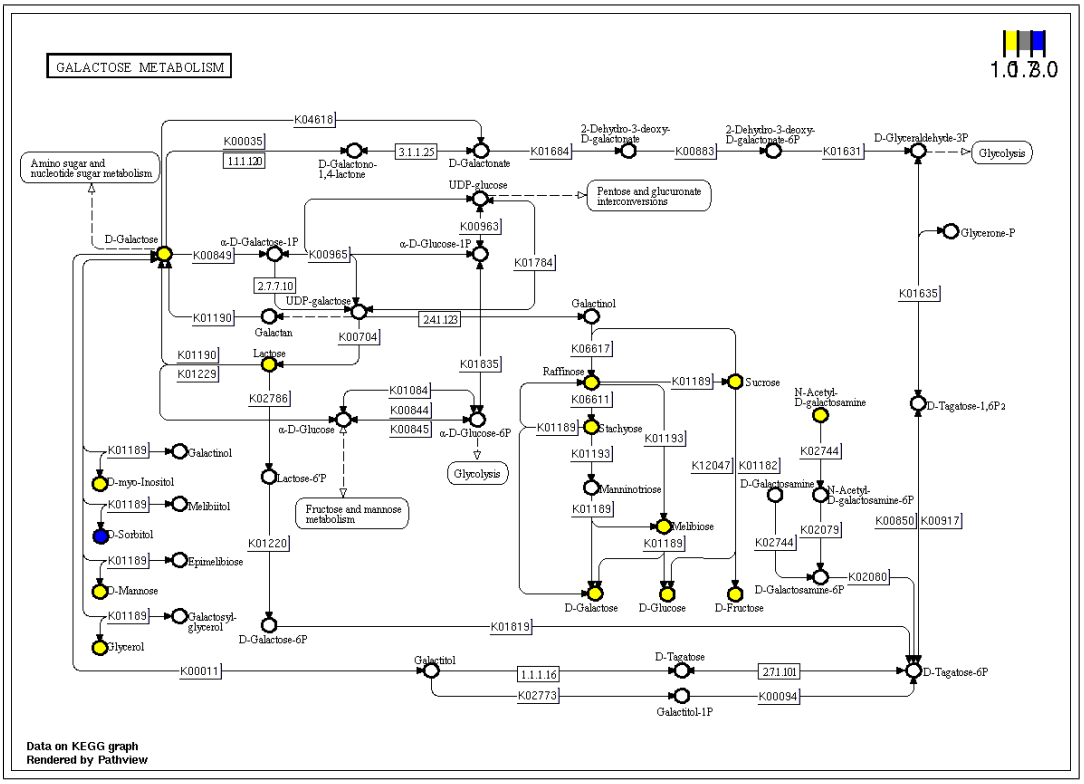


(a) Respiratory data (mean maximum height) of *Escherichia coli* strain K12 mapped on the KEGG Galactose pathway.

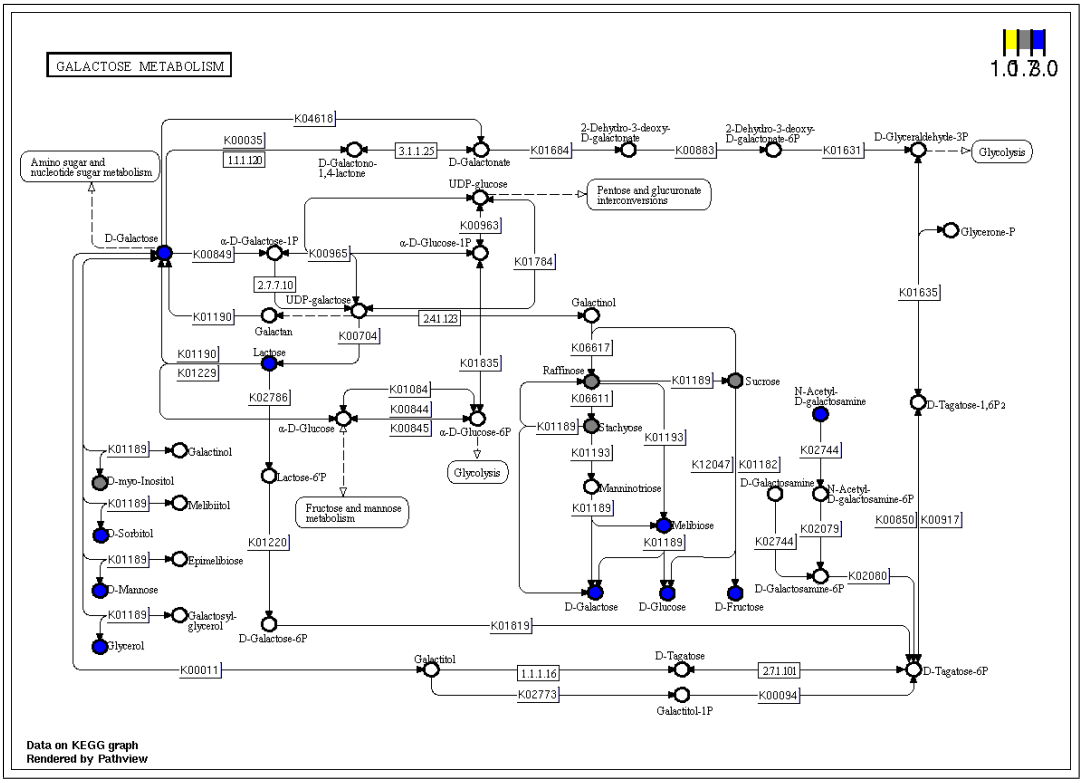


(b) Respiratory data (mean maximum height) of *Escherichia coli* type strain DSM 30083^{T} mapped on the KEGG Galactose pathway.

Figure 2: Galactose metabolism pathway maps for the two *E. coli* strains K12 and DSM 30083^{T} showing the mean maximum height on 13 substrates. The aggregated measurement data are represented by according colours and mapped on the corresponding substrates (circles) in the graph.



(a) Respiratory data (mean maximum height, discretized) of *Escherichia coli* strain K12 mapped on the KEGG Galactose pathway.



(b) Respiratory data (mean maximum height, discretized) of *Escherichia coli* type strain DSM 30083^{T} mapped on the KEGG Galactose pathway.

Figure 3: Galactose metabolism pathway maps for the two *E. coli* strains K12 and DSM 30083^{T} showing the mean maximum height on 13 substrates. The aggregated and discretized measurement data are represented by according colours and mapped on the corresponding substrates (circles) in the graph.

Note that from the annotation objects `anno.k12` or `anno.type`, respectively, comprising 96 substrates, only 13 are represented in the shown pathway map in Figure 2. This is no wonder because the PM plates are not arranged according to their affiliation to KEGG pathways. It often makes sense to restrict the considered substrates beforehand if the pathway of interest is already known. This particularly saves running time in the calls to `opm_mcp` and the `annotated` method for "opm_glht" objects.

When using the option `how = "value"`, `annotated` yields a numeric matrix with substrate names as row names and pathway identifiers as column names. Whereas the main use of such a matrix is described in Section 5, it can also be used simply to show the distribution of substrates over pathways. Ones and zeros indicate whether or not a certain substrate (row) is contained in a certain pathway (column). `NA`s indicate that a substrate has no KEGG ID, as for example well A01 containing the (pseudo-)substrate "Negative Control".

By summing up the columns and sorting the resulting vector, the user gets a ranking of the columns (pathways) indicating how many substrates are covered.

```
R> anno.k12.mat <- annotated(e.coli.k12, how = "value")
R> col.sums <- sort(colSums(anno.k12.mat, na.rm = TRUE), decreasing = TRUE)
R> col.sums[1:10]
```

```
     exact_mass           Value        map01100        map01110        map02010
       17697.97        14820.75           43.00           24.00           22.00
  Carbohydrates        map01120        map02060 Monosaccharides        map00052
          22.00           16.00           14.00           14.00           13.00
```

In the next example we search for the substrates represented in pathway number "00052", which is Galactose metabolism. Then we extract the positions of these substrates (for the plate type of interest) with `find_positions`:

```
R> e.subs <- rownames(anno.k12.mat)[!is.na(anno.k12.mat[, "map00052"]) &
     anno.k12.mat[, "map00052"] > 0]
R> e.subs.pos <- find_positions(e.subs, type = "Gen III")
R> e.subs.pos
```

```
                Sucrose                 Stachyose               D-Raffinose
                  "A07"                     "A09"                     "B01"
          a-D-Lactose               D-Melibiose N-Acetyl-D-Galactosamine
                  "B02"                     "B03"                     "B08"
            D-Glucose                 D-Mannose                D-Fructose
                  "C01"                     "C02"                     "C03"
          D-Galactose                 D-Sorbitol              myo-Inositol
                  "C04"                     "D01"                     "D04"
             Glycerol
                  "D05"
```

4.2.3. Visualisation of differences of group means in pathway maps

Next, we show the maximum-height values from the 13 substrates represented in the Galactose pathway map (number "00052") in Figure 4 to demonstrate the sizes of their differences. Remember that the vector `e.subs.pos` contains the positions of the substrates of interest as a character string. It can thus directly be used to subset `OPMX` objects.

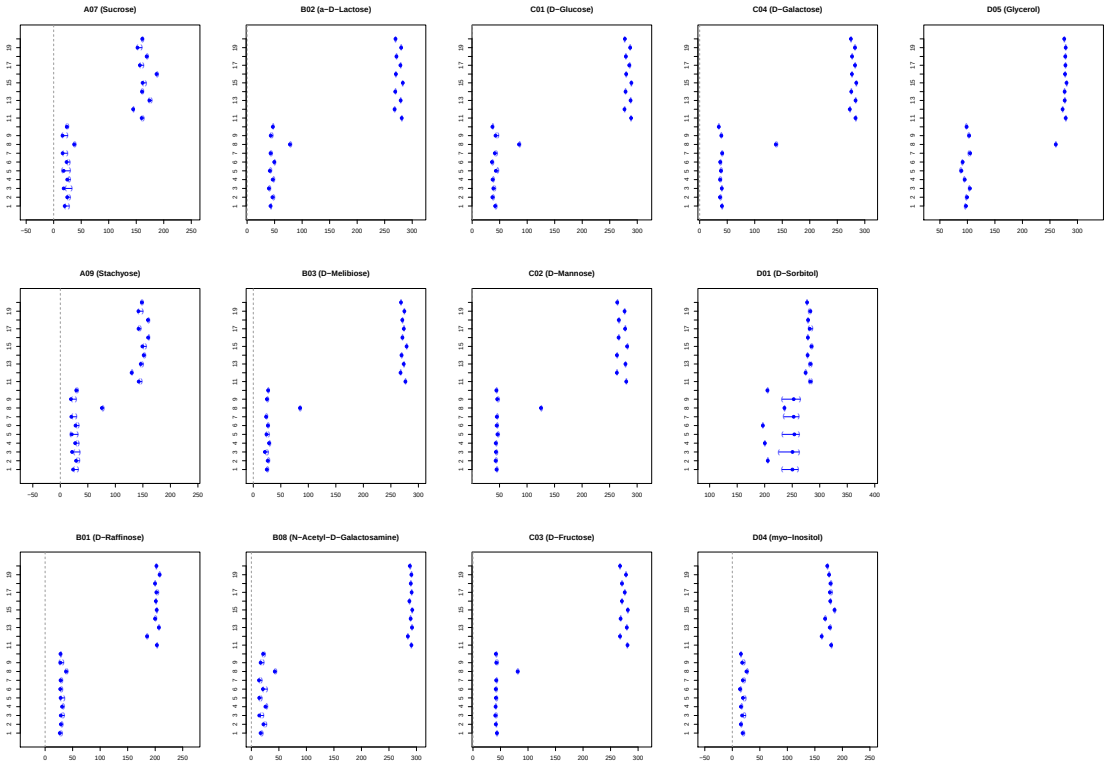


Figure 4: Point estimates and 95% confidence intervals for the single maximum-height values of the two *E. coli* strains for the subset of substrates represented in the Galactose pathway map.

```
R> ci_plot(e.coli.sub[, , e.subs.pos],
     as.labels = list("Species", "Strain"), subset = "A", x = "bottomright",
     draw.legend = T, crr = 1.33)
```

Straightforwardly, we compute a multiple comparison between only the 13 substrates included in the Galactose metabolism pathway map. Our example compares the type strain with K12; corresponding 95% CIs for differences of means for the chosen substrates are shown in Figure 5.

```
R> colicomp <- opm_mcp(e.coli.sub[, , e.subs.pos],
     output = "mcp", model = ~ J(Well + Strain), linfct = c(Pairs = 1))
```

The annotation vector for the differences of means can be obtained by simply applying `annotated` to the "opm_glht" object. The direct mapping of these differences between the two strains on the Galactose pathway is shown in Figure 6.

```
R> anno.colicomp <- annotated(colicomp)
R> e.coli.comp.map <- opm_path(cpd.data = anno.colicomp, species = "ko",
     out.suffix = "e.coli.comp.ko", pathway.id = "00052")
```

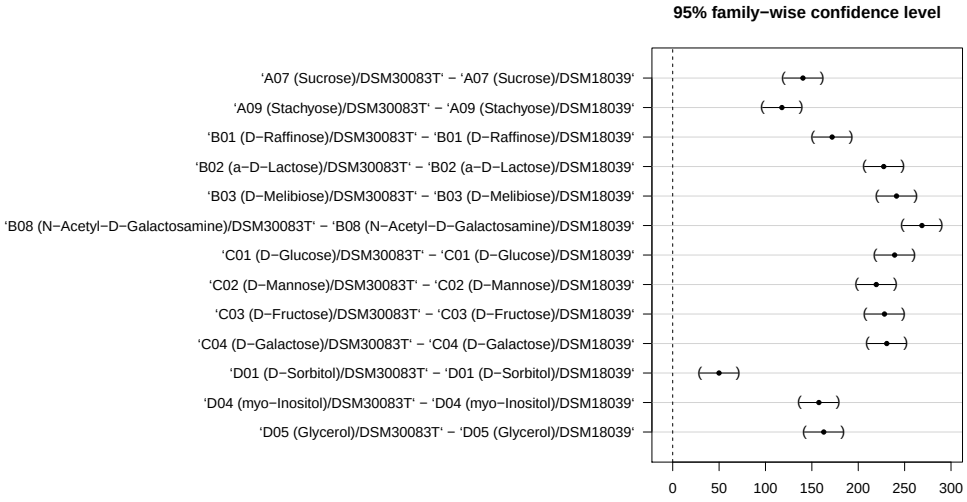


Figure 5: Point estimates and 95% confidence intervals for the differences of means between the two *E. coli* strains for the subset of substrates represented in the Galactose pathway map.

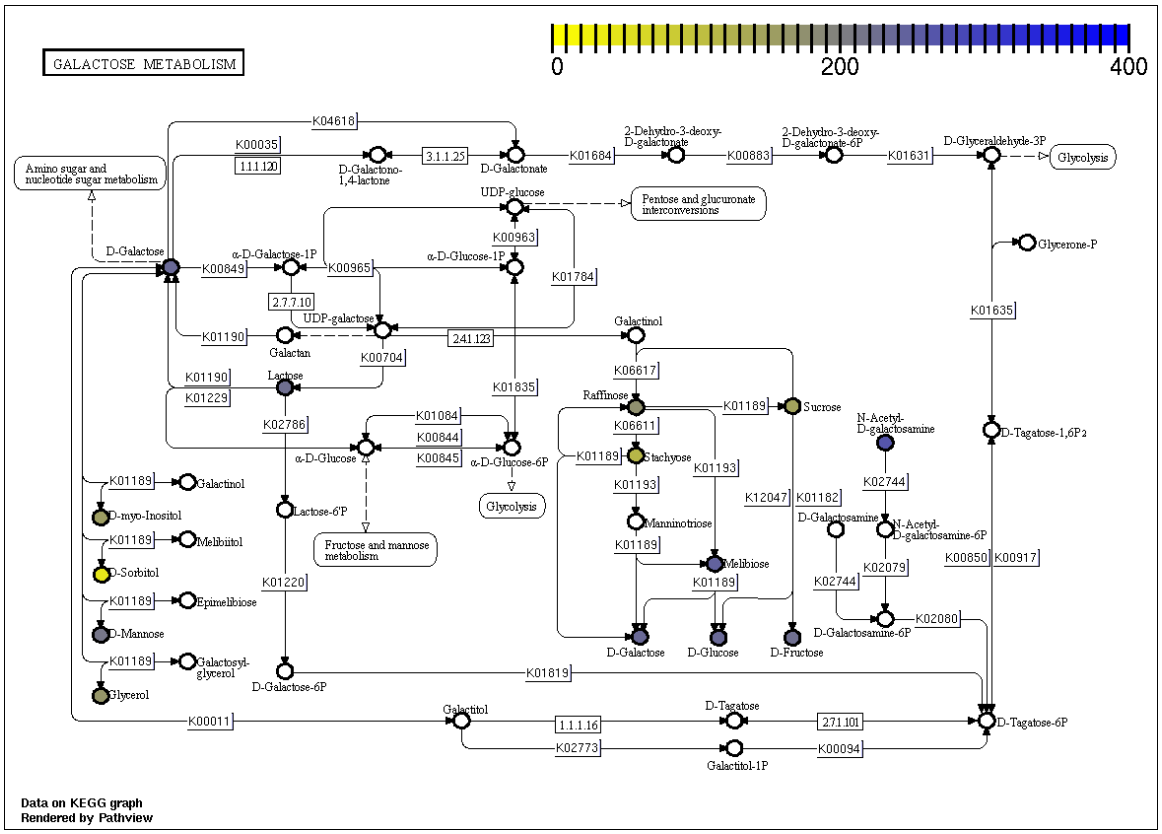


Figure 6: Respiration differences between *E. coli* K12 and DSM 30083^{T} regarding the maximum-height values mapped on the Galactose pathway.

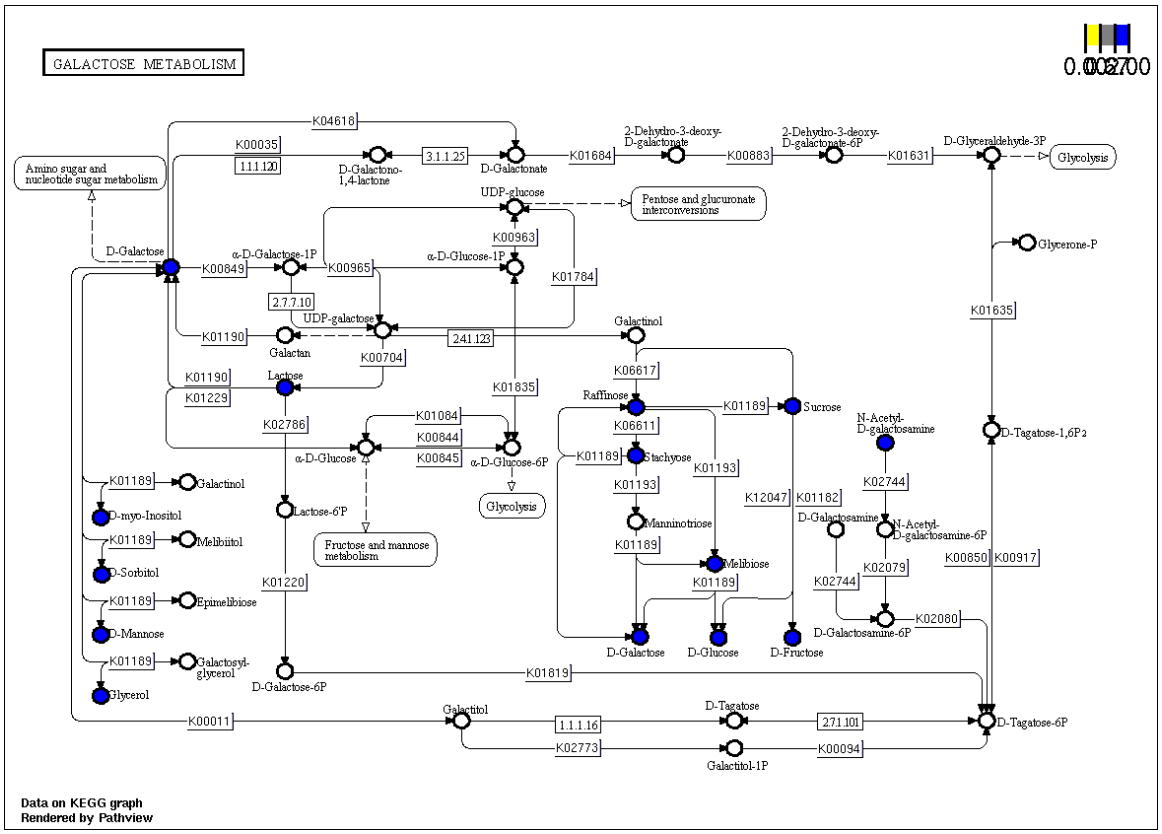


Figure 7: Significant respiration differences between *E. coli* K12 and DSM 30083^{T} regarding the maximum-height values mapped the Galactose pathway. For the direct mapping of the numeric differences see Figure 6.

In analogy to the last example, the function `annotated` can be used to produce categorical annotation vectors for the differences of means. Such vectors are very useful because they can specifically highlight the statistically significant differences, and particularly those that are significantly larger than a certain biologically relevant minimum effect size. Thus the full power of the functions from the **multcomp** package underlying `opm_mcp` (see the main tutorial and the manual for details) is available when visualizing PM data in pathway graphs.

The mapping of significant differences between the two strains on the Galactose pathway is shown in Figure 7.

```
R> cat.comp.anno <- annotated(colicomp, output = "upwards", lmap = 1:3)
R> cat.coli.comp <- opm_path(cpd.data = cat.comp.anno,
     species = "ko", out.suffix = "cat-coli-comp", pathway.id = "00052",
     limit = list(gene = 2, cpd = 2), bins = list(gene = 0.5, cpd = 3))
```

4.2.4. Visualisation of pathway maps in de-novo layout

In addition to the native KEGG visualization, `pathview` can use a *de-novo* visualisation approach. As return value, the function always generates a list containing two data frames. The data frame "plot.data.gene" contains the data for mapping the genes and, analogously,

"plot.data.cpd" stores the compound-related data. In the examples detailed above, a variety of such objects have already been generated, e.g.,

```
R> str(e.coli.map.k12)
```

```
List of 2
 $ plot.data.gene:'data.frame':        44 obs. of  9 variables:
  ..$ kegg.names: chr [1:44] "K00094" "K02773" "K01189" "K00917" ...
  ..$ labels    : chr [1:44] "K00094" "K02773" "K01189" "K00917" ...
  ..$ type      : chr [1:44] "ortholog" "ortholog" "ortholog" "ortholog" ...
  ..$ x         : num [1:44] 848 582 603 1027 977 ...
  ..$ y         : num [1:44] 755 754 458 561 561 310 153 586 548 487 ...
  ..$ width     : num [1:44] 46 46 46 46 46 46 46 46 46 46 ...
  ..$ height    : num [1:44] 17 17 17 17 17 17 17 17 17 17 ...
  ..$ kegg.names: num [1:44] NA NA NA NA NA NA NA NA NA NA ...
  ..$ mol.col   : Factor w/ 1 level "#FFFFFF": 1 1 1 1 1 1 1 1 1 1 ...
 $ plot.data.cpd :'data.frame':        43 obs. of  9 variables:
  ..$ kegg.names: chr [1:43] "C06311" "C01216" "C00137" "C00095" ...
  ..$ labels    : chr [1:43] "C06311" "C01216" "C00137" "C00095" ...
  ..$ type      : chr [1:43] "compound" "compound" "compound" "compound" ...
  ..$ x         : num [1:43] 741 682 98 800 379 169 384 518 518 519 ...
  ..$ y         : num [1:43] 754 152 520 642 152 266 330 266 205 152 ...
  ..$ width     : num [1:43] 8 8 8 8 8 8 8 8 8 8 ...
  ..$ height    : num [1:43] 8 8 8 8 8 8 8 8 8 8 ...
  ..$ mol.data  : num [1:43] NA NA 17.9 45.6 NA ...
  ..$ mol.col   : Factor w/ 8 levels "#696996","#B7B748",..: 8 8 7 4 8 4 8 8 8 8 ...
```

Next, we show how to produce graphics from these objects. TODO

5. Finding the pathways of interest

In many cases, the pathways of interest are not known in advance. The information accessible *via* `annotated` can be used in conjunction with suitable statistical procedures to automatically determine those features of the substrates that best correspond to a certain experimental outcome. Such results can be used as a starting point when searching for a causal explanation of the given **opm** results. They can also be used to simply determine the pathways to be used when drawing the **opm** results in a graph, as detailed in the previous sections. TODO

6. Acknowledgements

We are grateful to Barry Bochner (BIOLOG Inc.) for providing substrate and plate information on Phenotype MicroArray assays. The integration of missing OmniLog® substrates into ChEBI by the ChEBI staff is gratefully acknowledged. We thank Weijun Luo for providing hints regarding the use of the **pathview** package.

References

Caspi R, Altman T, Dreher K, Fulcher CA, Subhraveti P, Keseler IM, Kothari A, Krummenacker M, Latendresse M, Mueller LA, Ong Q, Paley S, Pujar A, Shearer AG, Travers M, Weerasinghe D, Zhang P, Karp PD (2012). "The MetaCyc database of metabolic pathways and enzymes and the BioCyc collection of pathway/genome databases." *Nucleic Acids Research*, **40**(D1), D742–D753. doi:10.1093/nar/gkr1014.

Coletti MH, Bleich HL (2001). "Medical Subject Headings Used to Search the Biomedical Literature." *Journal of the American Medical Informatics Association*, **8**(4), 317–323. doi:10.1136/jamia.2001.0080317.

Hastings J, de Matos P, Dekker A, Ennis M, Harsha B, Kale N, Muthukrishnan V, Owen G, Turner S, Williams M, Steinbeck C (2013). "The ChEBI reference database and ontology for biologically relevant chemistry: enhancements for 2013." *Nucleic Acids Research*, **41**(D1), D456–D463. doi:10.1093/nar/gks1146.

Kanehisa M, Goto S, Furumichi M, Tanabe M, Hirakawa M (2010). "KEGG for representation and analysis of molecular networks involving diseases and drugs." *Nucleic Acids Research*, **38**(suppl 1), D355–D360. doi:10.1093/nar/gkp896.

Luo W, Brouwer C (2013). "Pathview: an R/Biocondutor package for pathway-based data integration and visualization." *Bioinformatics*. doi:10.1093/bioinformatics/btt285. URL http://bioinformatics.oxfordjournals.org/content/29/14/1830.full.pdf+html.

Vaas LA, Sikorski J, Michael V, Göker M, Klenk H (2012). "Visualization and Curve Parameter Estimation Strategies for Efficient Exploration of Phenotype MicroArray Kinetics." *PLoS ONE*, **7**, e34846. doi:10.1371/journal.pone.0034846.

Affiliation:

Markus Göker
Leibniz Institute DSMZ – German Collection of Microorganisms and Cell Cultures
Braunschweig

Telephone: +49/531-2616-272
Fax: +49/531-2616-237
E-mail: markus.goeker@dsmz.de
URL: www.dsmz.de