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 10, 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**, which can be used to conduct web queries to obtain comprehensive data 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 Phenotype Microarray 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 these information directly, this tutorial introduces strategies for visualization of the measured PM results by mapping on pathway maps and introduces analysis methods for modelling and identification of informative and less informative substrates.

2. Preparation

For just downloading information from KEGG (see Section 3), 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 5), 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. 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.

4. Accessing substrate information

The **opm** package contains a number of functions suitable for accessing precomputed information on the substrates within certain wells and entire plates. 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 results 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 wildcards, 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 wildcard 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) IDs 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`.

5. 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/`).

5.1. 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. 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 to a single substrate. This is usually only possible for the "Dunnett" and "Pairs" type of contrasts if applied to the wells. 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 among the first 15 wells against well A05 as control group.

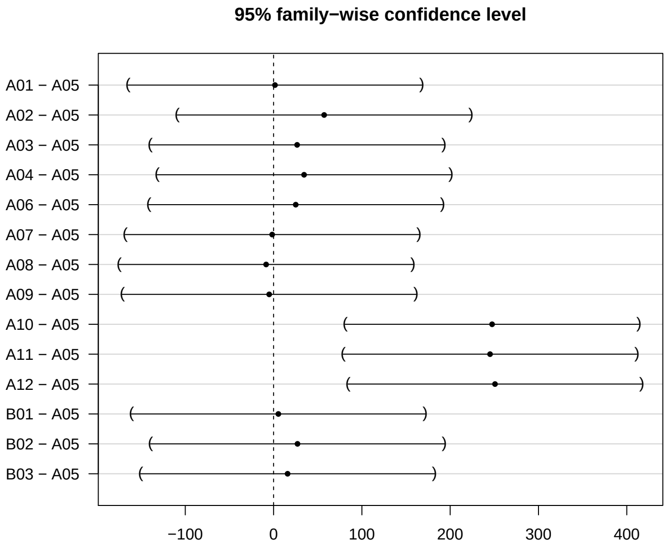


Figure 1: Point estimates and 95% confidence intervals in a manually defined comparison of group means for a specifically selected set of wells (A01 to B03) from the `vaas_4` exemplar object. In this procedure each well is compared against A05. The picture was obtained by running `opm_mcp()` and then the plotting function for the resulting `opm_glht` object. See the main tutorial for details.

```
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 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`.

A comprehensive overview of the possible results is shown for object `x` 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 6.

5.2. Visualisation in pathway maps using pathview

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, parsing of this data file, integrating user-defined data into the pathway representation, and rendering of final graph) are automatically conducted by `pathview`.

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 5.1) 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(vaas_et_al, list(Strain = "DSM18039"))
R> e.coli.type <- subset(vaas_et_al, 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 = "red"),
       low = list(gene = "white", cpd = "yellow"),
       pathway.id = "00052", species = "ko", out.suffix = "non-native",
       key.pos = "bottomright", 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, pathway.id = pathway.id,
       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, ...)
   }
```

First we show the results for the two strain on separate 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 that from the annotation objects `anno.k12` or `anno.typ`, respectively, comprehending 96 substrates, only 13 are represented in the shown pathway map in Figure 2. As described in section 5.1, for some substrates, like "pH5" or "positive Control" no KEGG ID is available, which results in NA in the vector. Accordingly, those substrates cannot be marked within pathway maps. Further not all substrates from one plate are represented in one single pathway, but only substrate groups are represented in pathway maps. By using the option `how = "value"`, `annotated` results a numeric matrix with substrate names as row names and pathway identifiers as column names. Ones and zeros indicate, if a certain substrate (rows) is element of a given pathway (columns), or not. NAs indicate, that the substrate has no KEGG ID, as for example well A01 with substrate "positive Control".

However, by summing up the columns and sorting the resulting vector, the user gets a ranking of the columns (pathways) representing how many substrates.

```
R> anno.k12.mat <- annotated(e.coli.k12, how = "value")
R> col.sums <- apply(anno.k12.mat, 2, sum, na.rm = TRUE)
R> col.sums.sort <- sort(col.sums, decreasing = TRUE)
R> # show the first ten entries
R> col.sums.sort[1:10]
```

```
          Value      exact_mass        map01100        map01110        map02010
       19707.75        17697.97           43.00           24.00           22.00
  Carbohydrates        map01120        map02060 Monosaccharides        map00052
          22.00           16.00           14.00           14.00           13.00
```

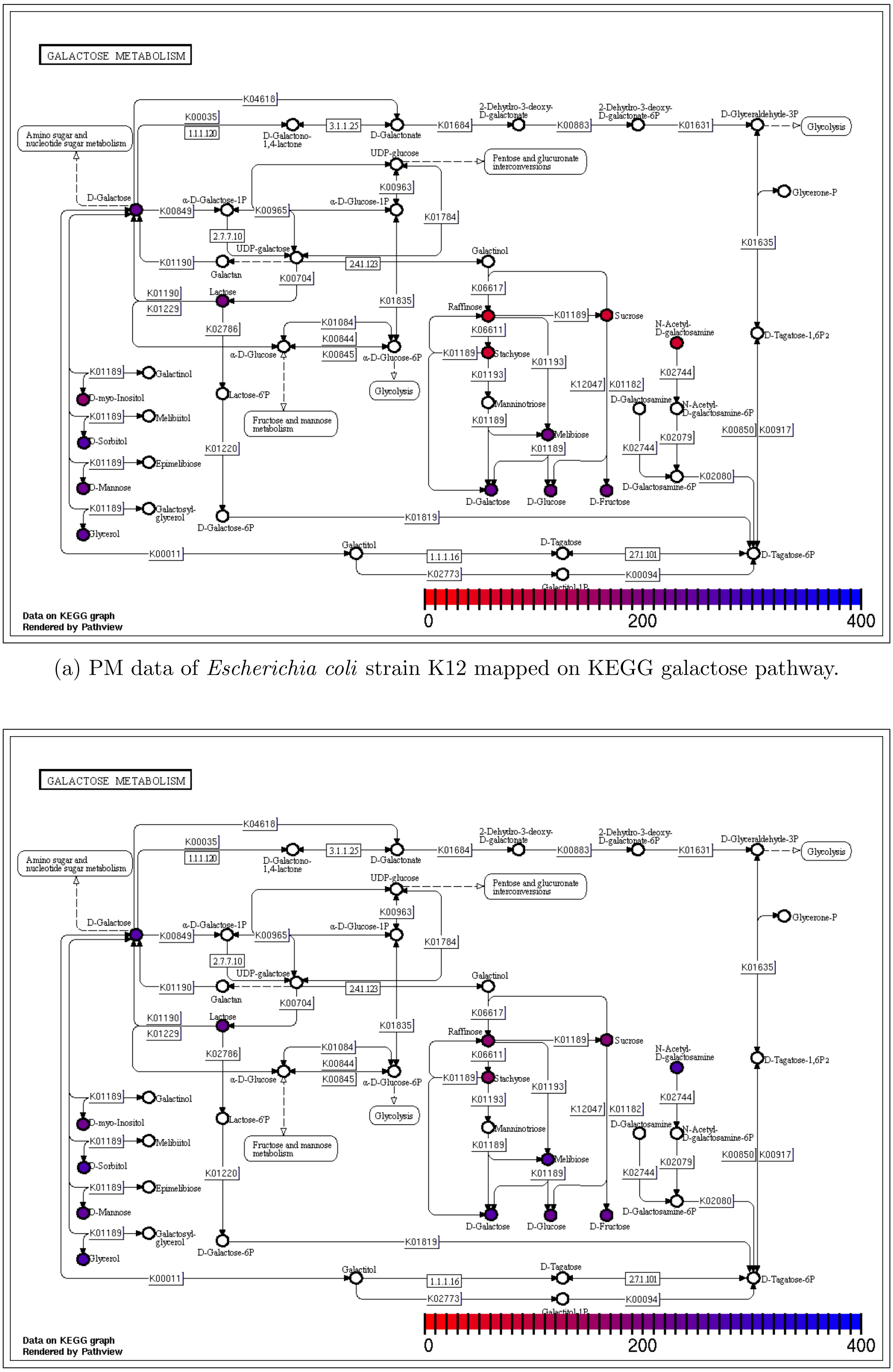
Further, the user is easily able to identify the substrates, represented in a choosen pathway. In the here shown example that are the substrates represented in pathway number 00052, which is galactose metabolism.

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

```
 [1] "Sucrose"                   "Stachyose"                 "D-Raffinose"
 [4] "a-D-Lactose"               "D-Melibiose"               "N-Acetyl-D-Galactosamine"
 [7] "D-Glucose"                 "D-Mannose"                 "D-Fructose"
[10] "D-Galactose"               "D-Sorbitol"                "myo-Inositol"
[13] "Glycerol"
```

The user is also able to extract the positions of these substrate on the plate

```
R> e.subs.pos <- find_positions(e.subs, type = "Gen III")
R> e.subs.pos
```



(a) PM data of *Escherichia coli* strain K12 mapped on KEGG galactose pathway.

(b) PM data of *Escherichia coli* type strain DSM 30083^T mapped on KEGG galactose pathway.

Figure 2: Pathway maps for the two *E. coli* strains showing TODO. Note particularly the substrates Raffinose, Stachyose and Sucrose (in the middle of the map), which exhibit TODO.

```
                 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"
```

Now it is straightforward to compute a multiple comparison between only the 13 substrates included in the Galactose pathway map, since vector `e.subs.pos` contains the positions of the substrates of interest and can serve as direct input in `opm_mcp`. Our example now compares the type strain with K12; corresponding 95% CIs for differences of means on the choosen substrates are shown in Figure 3.

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

The annotation vector for the differences of means can be obtained straightforwardly by simple application of `annotated` on the `opm_glht` object. This directly facilitates the mapping of the differences between the two strains on the galactose pathway, as it is shown in Figure 4

```
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")
```

6. Finding the pathways of interest

The obtained information about the pathway certain substrates are involved in can be used to determine manually which pathways are actually of interest or can serve as input for model-building procedures as, for example, random forest approaches, which is demonstrated below.

7. Acknowledgements

The integration of missing OmniLog® substrates into ChEBI by the ChEBI staff is gratefully acknowledged.

References

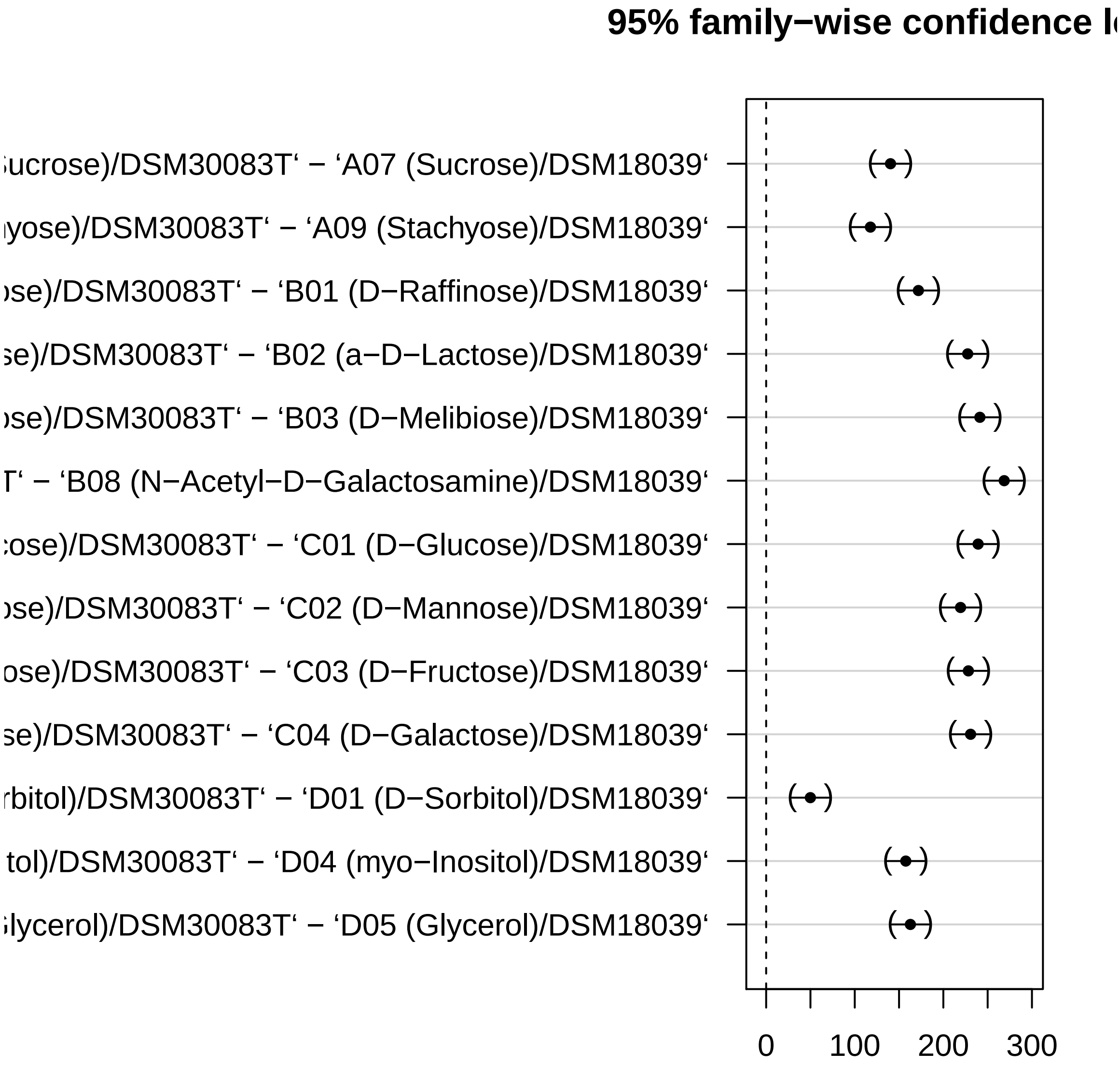


Figure 3: Point estimates and 95% confidence intervals for the differences of means between the two *E. coli* strains for the subset of substrates represented in the Galactose pathway map.

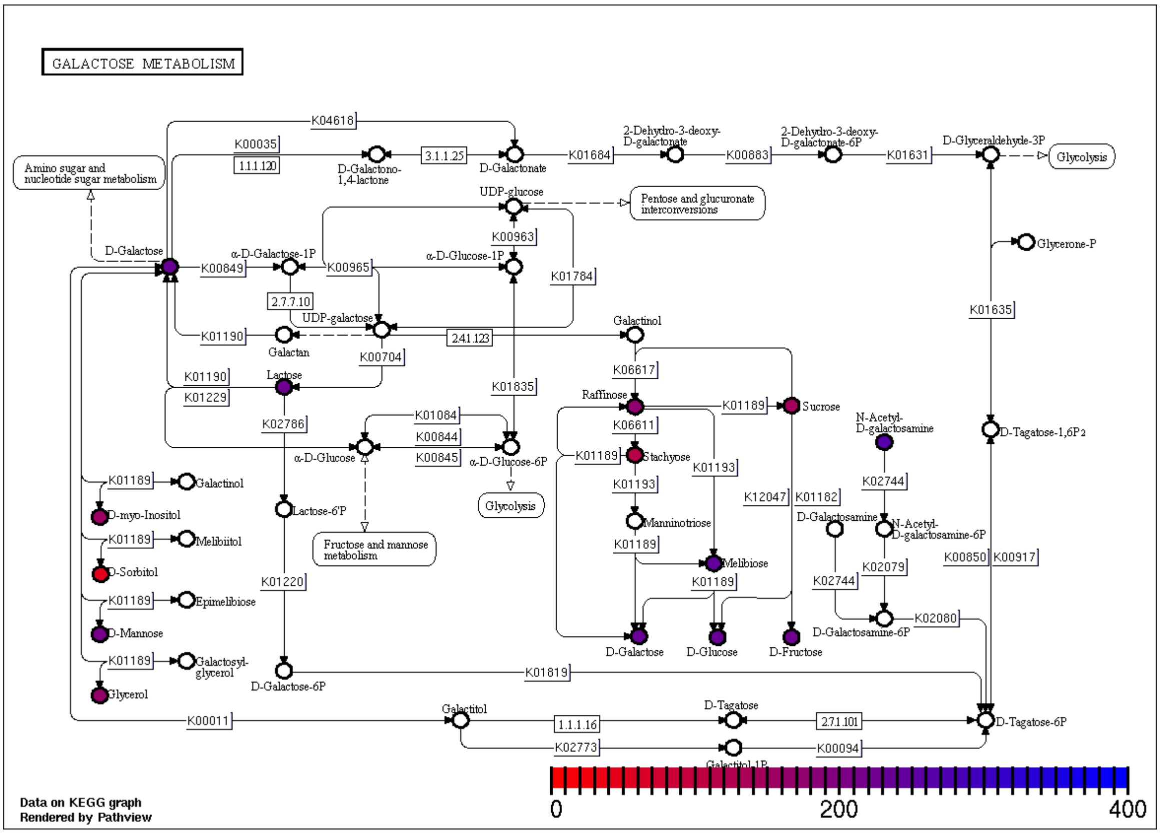


Figure 4: Mapping of differences between *E. coli* K12 and type strain DSM 30083$^{\mathrm{T}}$ Galactose pathway.

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.

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