MetaboDynamics: a worked example

Katja Danielzik

2024-08-20

This package was built to facilitate the analysis of longitudinal metabolomics data. Most tools only allow the comparison between two time points or experimental conditions and are using frequentist statistical methods.

Here we want to show a complete workflow to analyze concentration tables. As example we have a data set of an irradiated cancer cell line that was observed over four time points after irradiation with different doses. For each time point and radiation dose it contains three replicates of "cpc-values". Cpc refers here to the metabolite concentration per cell.

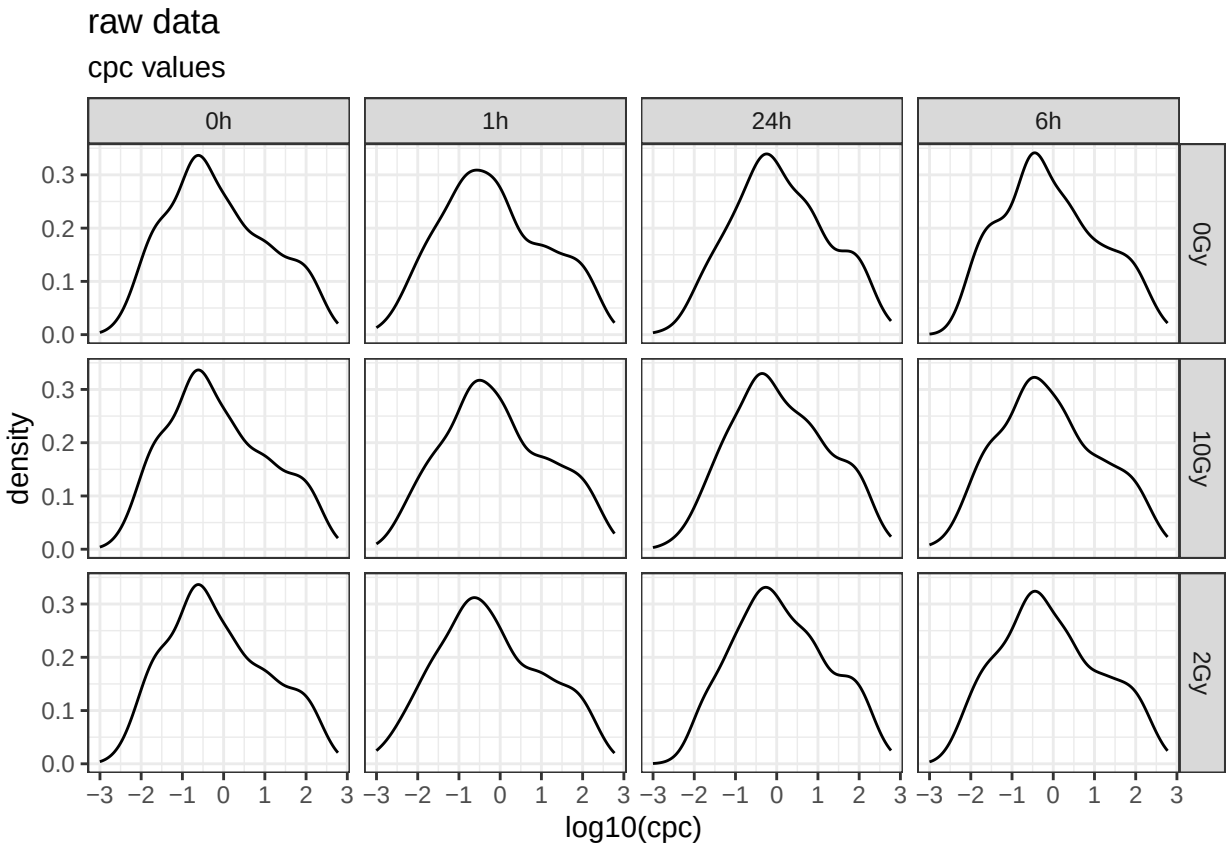
setup: load required packages

```
library(MetaboDynamics)
library(ggplot2)
library(dplyr)
library(tidyr)
```

load data and plot data overview

```
data("intra")
```

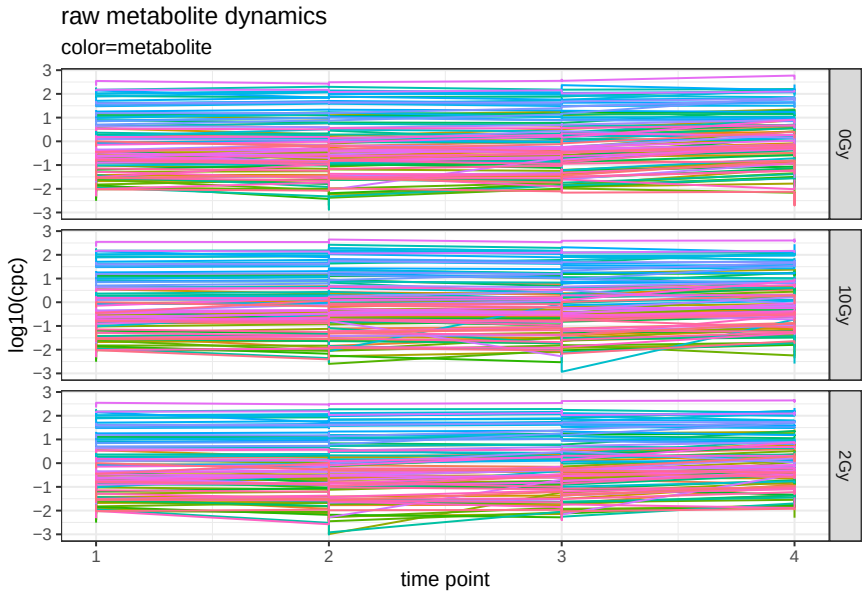
```
# plot log-transformed data
ggplot(intra, aes(x = log10(cpc))) +
  geom_density() +
  theme_bw() +
  facet_grid(cols = vars(time), rows = vars(dose)) +
  ggtitle("raw data", "cpc values")
```



In the first step in this workflow we estimate the dynamics of every single metabolite at every experimental condition (here: radiation dose). As metabolomics data is often noisy and we generally have few replicates due to high costs, a robust method is needed for the estimation of mean concentrations at every time point. For this we employ a Bayesian hierarchical model that assumes normal distributions of log-transformed metabolite concentrations. The next plot shows the raw dynamics of single metabolites.

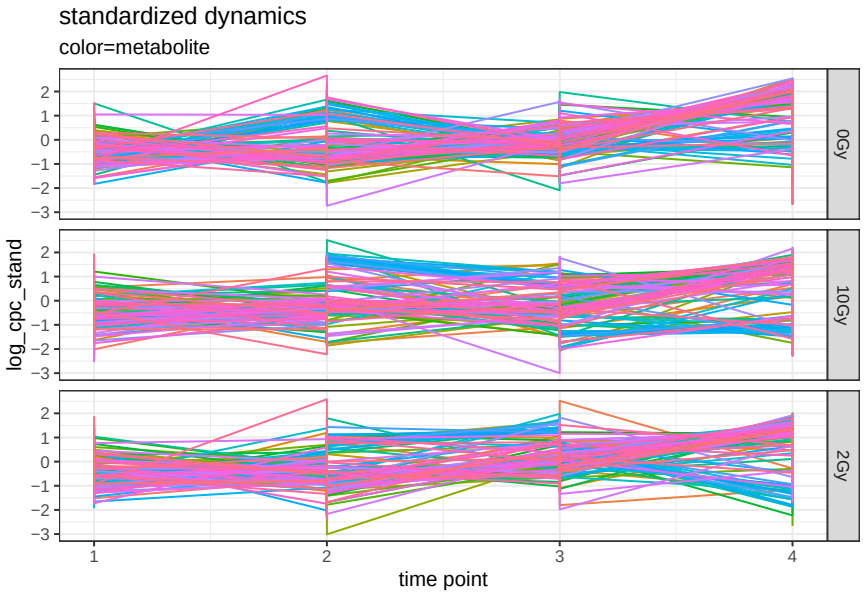
Use time as numeric variable

```
ggplot(intra) +
  geom_line(aes(x = as.numeric(as.factor(time)), y = log10(cpc),
                col = metabolite, group=metabolite)) +
  theme_bw() +
  xlab("time point") +
  theme(legend.position = "none") +
  facet_grid(rows = vars(dose)) +
  ggtitle("raw metabolite dynamics", "color=metabolite")
```



We define dynamics as deviations at the observed time points from the metabolite's mean concentration over time. As the raw concentrations of metabolites can differ by orders of magnitudes from each other, and we want to be able to compare dynamics of metabolites with each other, we standardize each metabolite at each radiation dose to a mean of zero and a standard deviation of one.

```
intra <- intra %>%
  group_by(dose, metabolite) %>%
  mutate(log_cpc_stand = ((log_cpc - mean(log_cpc)) / sd(log_cpc)))
ggplot(intra) +
  geom_line(aes(
    x = as.numeric(as.factor(time)),
    y = log_cpc_stand, col = metabolite
  )) +
  theme_bw() +
  xlab("time point") +
  theme(legend.position = "none") +
  facet_grid(rows = vars(dose)) +
  ggtitle("standardized dynamics", "color=metabolite")
```



Now we can finally model the dynamics.

We employ a Bayesian hierarchical model with con= metabolite concentrations, m= metabolite, c= experimental condition and t= time point ID:

$$
\begin{aligned}
\log(con_{m,c,t}) &\sim \text{normal}(\mu_{m,c,t}, \sigma_{m,c,t}) \\
\mu_{m,c,t} &\sim \text{normal}(0, 2) \\
\sigma_{m,c,t} &\sim \text{exponential}(\lambda_{m,c}) \\
\lambda_{m,c} &\sim \text{exponential}(2)
\end{aligned}
$$

The code below shows how to fit the model and how to extract the diagnostic criteria from the model fits.

Model dynamics

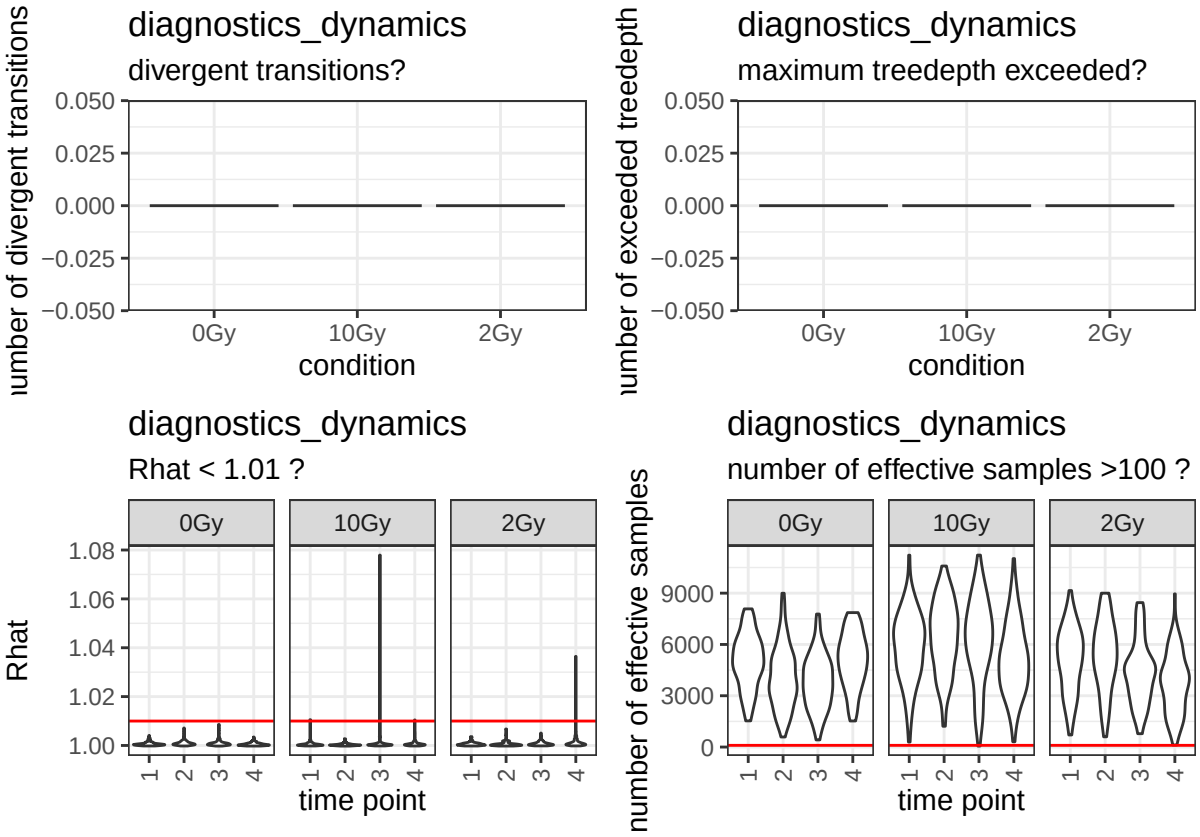
```
# # fit model
# fits_dynamics <- fit_dynamics_model(
#   data = intra, cpc = "log_cpc_stand",
#   condition = "dose", max_treedepth = 14,
#   adapt_delta = 0.999, iter = 4000, cores = 7)

# # extract diagnostics
# diagnostics_dynamics <- extract_diagnostics_dynamics(
#   data = intra, iter = 4000,
#   fits = fits_dynamics
# )

# diagnostics_dynamics[["plot_divergences"]]
# diagnostics_dynamics[["plot_treedepth_error"]]
# diagnostics_dynamics[["plot_rhat"]]
```

```
# diagnostics_dynamics[["plot_neff"]]

# # PPCs can be accessed with
# diagnostic_estimates[["plot_PPC_0Gy"]]
# diagnostic_estimates[["plot_PPC_2Gy"]]
# diagnostic_estimates[["plot_PPC_10Gy"]]
```



For our experimental data set this might take up to 10 minutes per radiation dose depending on how many cores are used.

This returns a list of model fits that are named by the experimental condition ("0Gy","2Gy","10Gy"). With extract_diagnostics_dynamics() we can extract all the diagnostic criteria of Bayesian models (rhat, neff, divergences, max_treedepth) and visualize them.

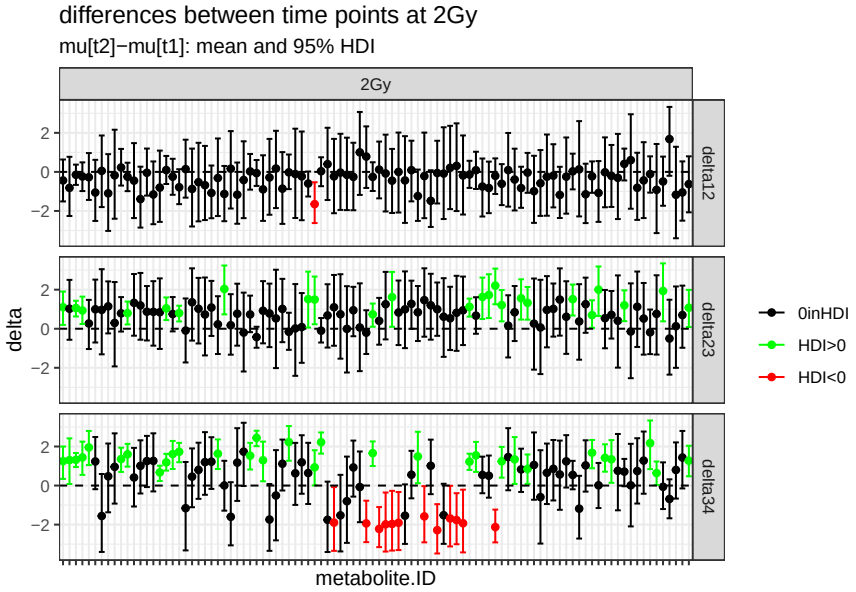
After checking the diagnostic criteria and the PPC we can extract the estimates:

```
# #extract estimates
# estimates_dynamics <- extract_estimates_dynamics(
#   data = intra, fits = fits_dynamics,
#   iter = 4000
# )
```

We get two major outputs: 1) the estimation of concentration differences between two subsequent time points of each metabolite at each experimental condition 2) the dynamic profiles of each metabolites at each experimental condition.

1) differences between two time points

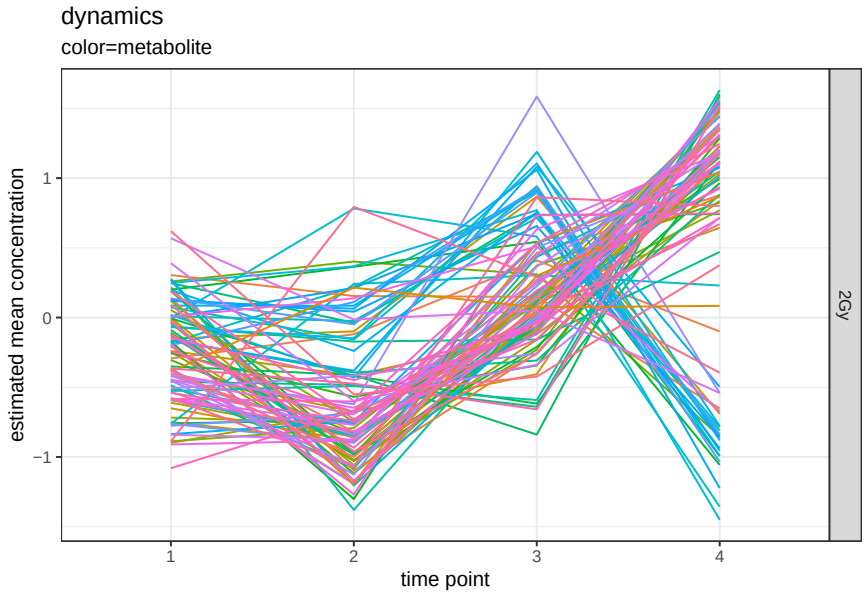
```
# 1) the differences between two time points
# set numer of time points
#estimates_dynamics[["plot_timepoint_differences"]]
```



If the 95% highest density interval (HDI) of the posterior does not include zero we can state that there is probably a difference in mean concentrations between two time points. If the 95% HDI lies below zero we have a decrease in metabolite concentrations between the two time points, if lies above zero we have an increase in concentrations between time points.

2) dynamic profiles

```
# 2) dynamic profiles
#estimates_dynamics[["plot_dynamics"]]
```

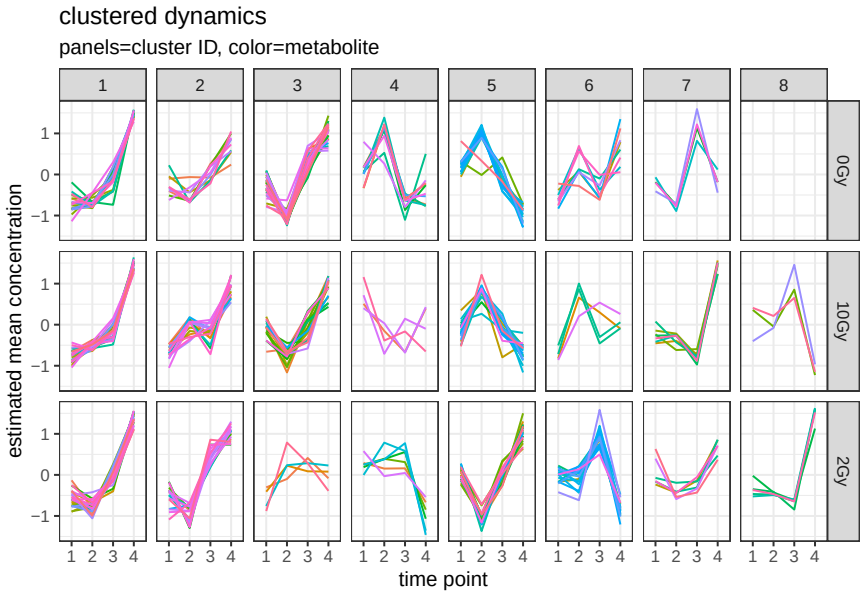


So we now have dynamic profiles of the metabolites at each radiation dose. What do we do with this? We could cluster these dynamics vectors, to investigate if we find groups of metabolites that have similar dynamics.

dynamic clusters

For the sake of demonstration we only show the results of the clustering analysis here. In practice the optimal number of clusters should be determined by clustering criteria such as Gap statistics and average silhouette.

```
temp <- cluster
temp <- temp %>% pivot_longer(
  cols = c(mu1_mean, mu2_mean, mu3_mean, mu4_mean),
  names_to = "timepoint", values_to = "mu_mean"
)
ggplot(temp, aes(
  x = as.factor(as.numeric(as.factor(timepoint))),
  y = mu_mean, group = metabolite, col=metabolite
)) +
  geom_line() +
  xlab("time point") +
  ylab("estimated mean concentration") +
  theme_bw() +
  theme(legend.position = "none") +
  facet_grid(rows = vars(condition), cols = vars(cluster)) +
  ggtitle("clustered dynamics", "panels=cluster ID, color=metabolite")
```



As we can see metabolites show different dynamics in different radiation doses. Can we quantify the biological function of these dynamic clusters?

Over-representation analysis of functional modules in dynamic clusters

To quantify the possible biological function of these dynamic clusters we retrieved from the KEGG-database the following information (with package KEGGREST): 1) to which functional modules our experimental metabolites are annotated and 2) which metabolites are annotated to functional modules in general.

The functional modules of the KEGG-database are organised in three hierarchies: upper, middle and lower. Here we will do functional analysis on the middle hierarchy. To facilitate analysis the data frames "metabolite_modules", which holds the information about experimental metabolites, and "modules_compounds", which holds the information about which metabolites are in general annotated to functional modules, were prepared. We load both data sets and can inspect the documentation.

```
data("metabolite_modules")
#help("metabolite_modules")
head(metabolite_modules)
```

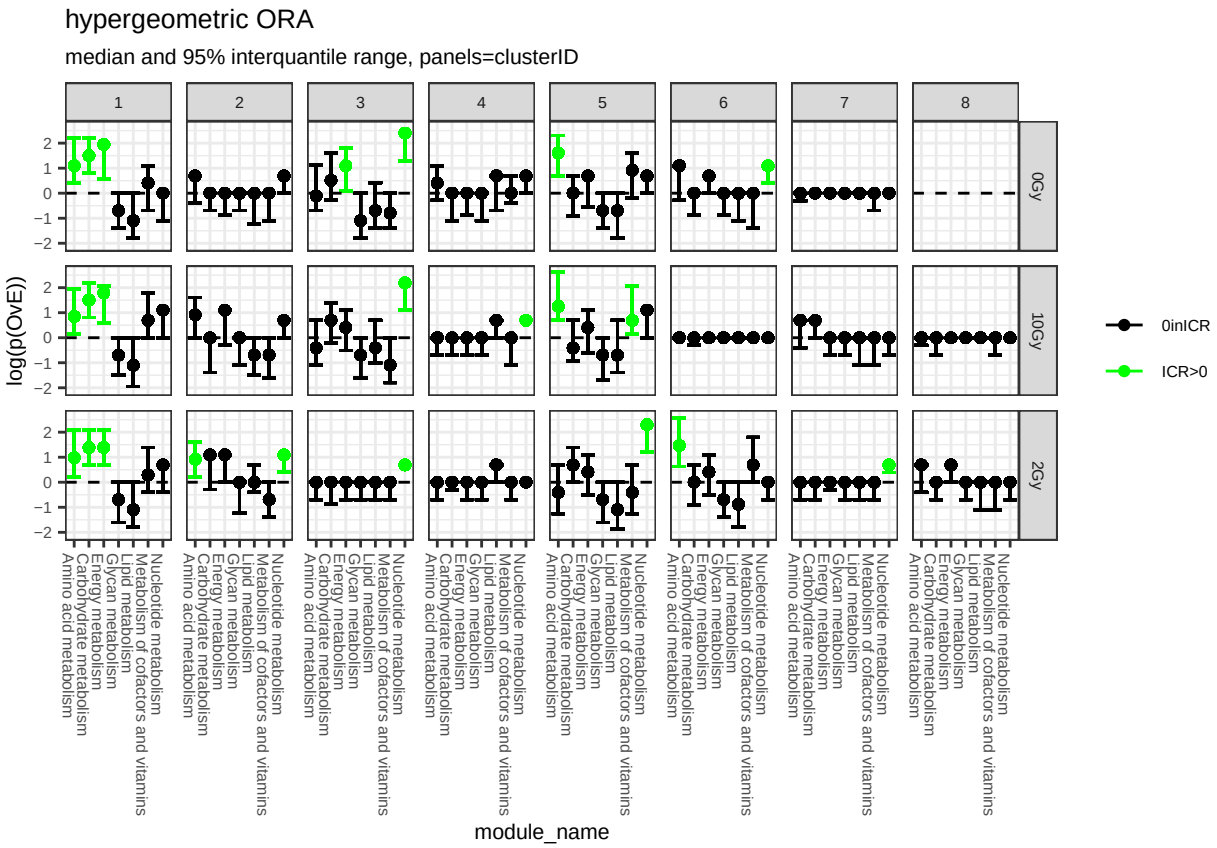
```
## # A tibble: 6 x 8
##    ...1 metabolite  KEGG  module_id module_name upper_hierarchy middle_hierarchy
##   <dbl> <chr>       <chr> <chr>     <chr>       <chr>           <chr>
## 1     1 1-Aminocyc~ C012~ M00368    Ethylene b~ Pathway modules Amino acid meta~
## 2     2 2-Aminomuc~ C022~ M00038    Tryptophan~ Pathway modules Amino acid meta~
## 3     3 2-Phosphog~ C006~ M00001    Glycolysis~ Pathway modules Carbohydrate me~
## 4     4 2-Phosphog~ C006~ M00002    Glycolysis~ Pathway modules Carbohydrate me~
## 5     5 2-Phosphog~ C006~ M00003    Gluconeoge~ Pathway modules Carbohydrate me~
## 6     6 2-Phosphog~ C006~ M00346    Formaldehy~ Pathway modules Energy metaboli~
## # i 1 more variable: lower_hierarchy <chr>
```

```
data("modules_compounds")
#help("modules_compounds")
head(modules_compounds)
```

```
## # A tibble: 6 x 6
##    ...1 module_id kegg_id upper_hierarchy middle_hierarchy       lower_hierarchy
##   <dbl> <chr>     <chr>   <chr>           <chr>                  <chr>
## 1     2 M00001    C00267  Pathway modules Carbohydrate metaboli~ Central carboh~
## 2     3 M00001    C00668  Pathway modules Carbohydrate metaboli~ Central carboh~
## 3     4 M00001    C05345  Pathway modules Carbohydrate metaboli~ Central carboh~
## 4     5 M00001    C05378  Pathway modules Carbohydrate metaboli~ Central carboh~
## 5     6 M00001    C00111  Pathway modules Carbohydrate metaboli~ Central carboh~
## 6     7 M00001    C00118  Pathway modules Carbohydrate metaboli~ Central carboh~
```

For the functional analysis we employ a hypergeometric model. We consider a functional module as over-expressed in a cluster if the 95% inter-quantile range (ICR) of the log-transformed probabilities of OvEs lies above zero. OvE refers to the ratio of observed metabolites in a cluster being mapped to a functional module over the number of expected metabolites in a cluster being mapped to a module under the assumption of a hypergeometric distribution (=drawing without replacement).

```
# ORA <- ORA_hypergeometric(background = modules_compounds,
#                           annotations = metabolite_modules,
#                           clusters = cluster,
#                           tested_column = "middle_hierarchy")
# ORA["ORA"]
```



Great, we can now see which functional module is over- (green points and error-bars) or under-represented (none in this example) in which dynamic cluster! For instance in cluster 1 at all radiation doses the modules "Energy metabolism", "Carbohydrate metabolism" and "Amino acid metabolism" are over-represented.

Comparison of clusters between different experimental conditions

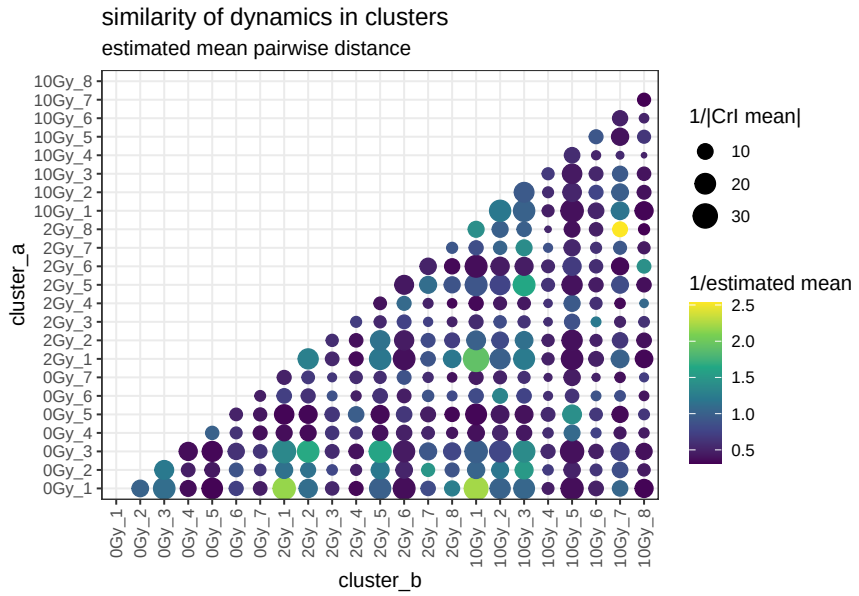
dynamics

We can not only do over-representation analysis of KEGG-functional modules but also compare dynamic clusters across different experimental conditions. For this we employ a Bayesian model that estimates the mean difference as well as the standard deviation of differences between dynamic clusters.

dist= vector of pairwise euclidean distances between each dynamic vector of cluster a and every dynamic vector of cluster b, ID= cluster pair ID

$$
\begin{aligned}
dist_{ID} &\sim \text{normal}(\mu_{ID}, \sigma_{ID}) \\
\mu_{ID} &\sim \text{normal}^{+}(2, 2) \\
\sigma_{ID} &\sim \text{exponential}(1)
\end{aligned}
$$

```
comparison_dynamics <- compare_dynamics(clusters = cluster,
                                        dynamics = c("mu1_mean", "mu2_mean",
                                                     "mu3_mean", "mu4_mean"))
comparison_dynamics[["plot_dynamic_comparison"]]
```

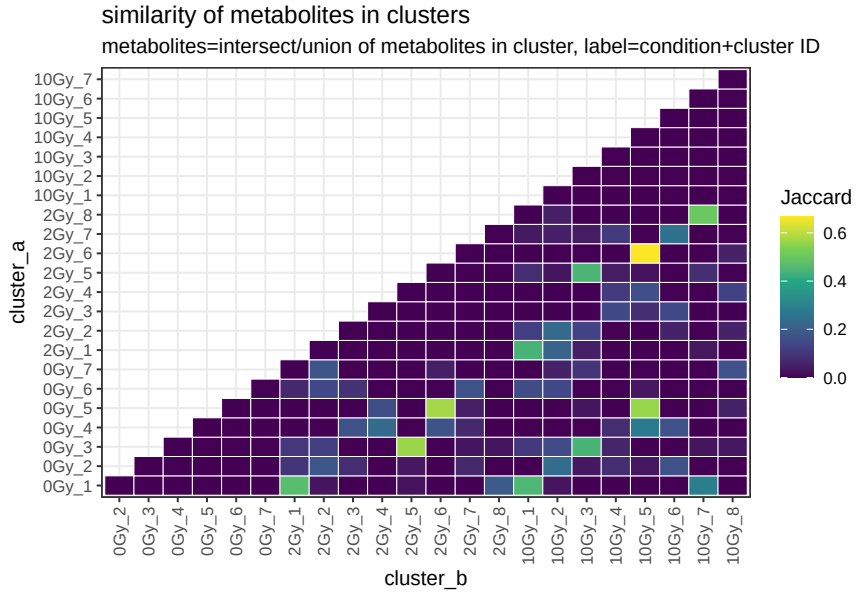


The bigger and brighter a point is the smaller is the mean distance between dynamic clusters and the smaller is the standard deviation. That means big bright points indicate high dynamic similarity with small spread. For instance, cluster 7 at 10Gy is similar to cluster 8 at 2Gy. Also cluster 1 at 0Gy is quite similar to cluster 1 at 2Gy and cluster 1 at 10Gy.

metabolites

For the comparison of metabolites between different dynamic clusters we employ the Jaccard Index.

```
comparison_metabolites <- compare_metabolites(clusters=cluster)
comparison_metabolites[["plot_metabolite_comparison"]]
```



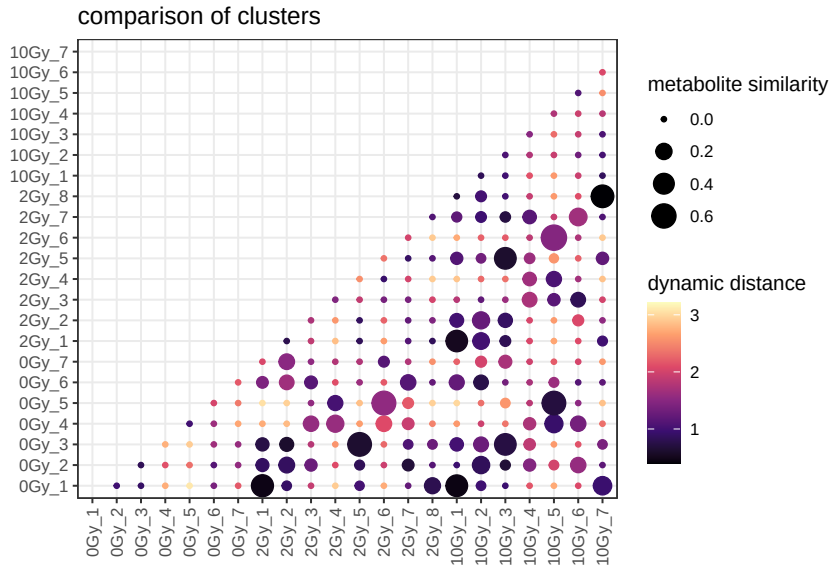
We have two clusters that are very similar in their metabolite composition (yellow square: cluster 5 at 10Gy and cluster 6 at 2Gy). If we compare that to the dynamic profiles we see that the dynamics differ between the two radiation doses.

Can we facilitate visualization?

combine both

```
dynamics <- comparison_dynamics[["estimates"]]
metabolites <- comparison_metabolites[["Jaccard"]]

temp <- left_join(dynamics,metabolites,by=c("cluster_a","cluster_b"))
x <- unique(temp$cluster_a)
ggplot(temp, aes(x = cluster_b, y = cluster_a)) +
    geom_point(aes(size = Jaccard, col = mu_mean)) +
    theme_bw() +
    scale_color_viridis_c(option = "magma") +
    scale_x_discrete(limits =x) +
    xlab("")+
    ylab("")+
    scale_y_discrete(limits = x) +
    theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust = 1)) +
    labs(col = "dynamic distance", size = "metabolite similarity")+
    ggtitle("comparison of clusters")
```



We can find two cluster pairs that have pretty similar metabolite composition but dissimilar dynamics (big purple dots). As in the comparison of metabolites before the pair cluster 5 at 10Gy and cluster 6 at 2Gy, but also cluster 6 at 2Gy and cluster 5 at 0Gy. For both pairs the ORA profiles are quite similar as expected from the similar metabolite compositions.