

Data mining made easy, reproducible and open-source

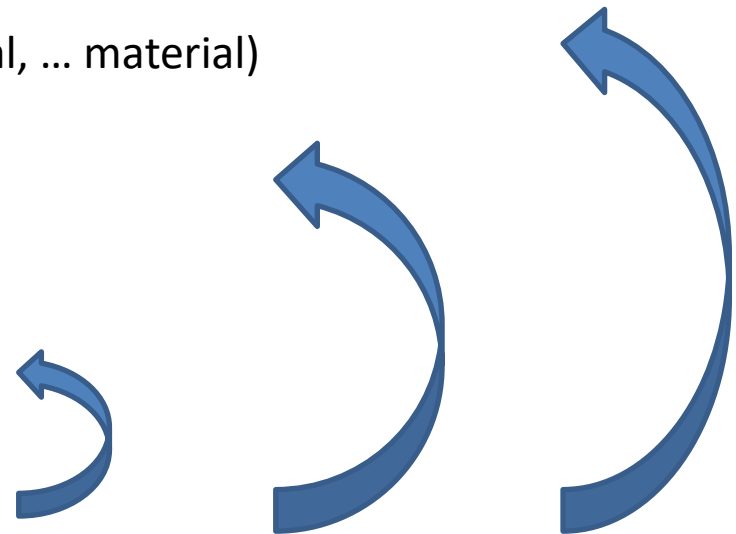
deRSE19 - 2019 June 05

Shany Gefen-Treves & Alexander Bartholomäus

(Life) Science way of data

The way of data in life science publication process

- *Unknown*
- Ideas how to answer / clarify / solve *Unknown*
- Obtain / collect material (biological, physical, ... material)
- Get raw data
- Process data
- Create data collection
- Ask questions
- Format / restructure data
- Visualize data
- More new questions



- Enough questions answered:
Publish and share data and conclusions

Reproducibility on the way of data

- *Unknown*
 - Ideas how to answer / clarify / solve *Unknown*
 - Obtain / collect material (biological, physical, ... material)
 - Get raw data
 - Process data
 - Create data collection
 - Ask questions
 - **Format / restructure data**
 - **Visualize data**
 - More new questions
 - Enough questions answered:
Publish and share data and conclusions
- make the results usable by the community
- create reliability and enhance the „correctness“ of the results

Our need and vision




Microbiologist need

- Custom data analysis need
- Simple adjustment of plots and parameters

Bioinformatic vision

- Interactive data analysis
- Simple usage and publishing
- Good review possibility
- Strong reproducibility
- Easy reuse and modification of code and analysis

R / Rstudio / Shiny

- R 
 - Well-known scientific programming language
 - Number 7 most popular programming language (June 2019 PYPL-index <http://pypl.github.io/PYPL.html>)
- RStudio 
 - R optimized IDE
 - Intuitive and simple to generate reports
- Shiny (RStudio) 
 - R package to enable easy R based web programming
 - Flexible to use (standalone, R markdown, dashboards)

R / RStudio / Shiny – demo

The screenshot displays the RStudio interface with a Shiny application code file named `app.R`. The code is as follows:

```
1 # This is a Shiny web application. You can run the application by clicking
2 # the 'Run App' button above.
3
4 # Find out more about building applications with shiny here:
5 #
6 # http://shiny.rstudio.com/
7 #
8
9 library(shiny)
10
11 # Define UI for application that draws a histogram
12 ui <- fluidPage(
13   # Application title
14   titlePanel("Old Faithful Geyser Data"),
15   # Sidebar with a slider input for number of color
16   sidebarLayout(
17     sidebarPanel(
18       sliderInput("color",
19                 "color:",
20                 min = 1,
21                 max = 8,
22                 value = 1)
23     ),
24     # Show a plot of the generated distribution
25     mainPanel(
26       plotOutput("distPlot")
27     )
28   )
29 )
30
31 # Define server logic required to draw a histogram
32 server <- function(input, output) {
33
34   output$distPlot <- renderPlot({
35     # generate bins based on input$bins from ui.R
36     x <- faithful[, c(1,2)]
37
38     # another plot
39     plot(x, col = input$color, pch = 19)
40   })
41
42 # Run the application
43 shinyApp(ui = ui, server = server)
44
45
46
47
48
49
50
```

The console shows the R startup message:

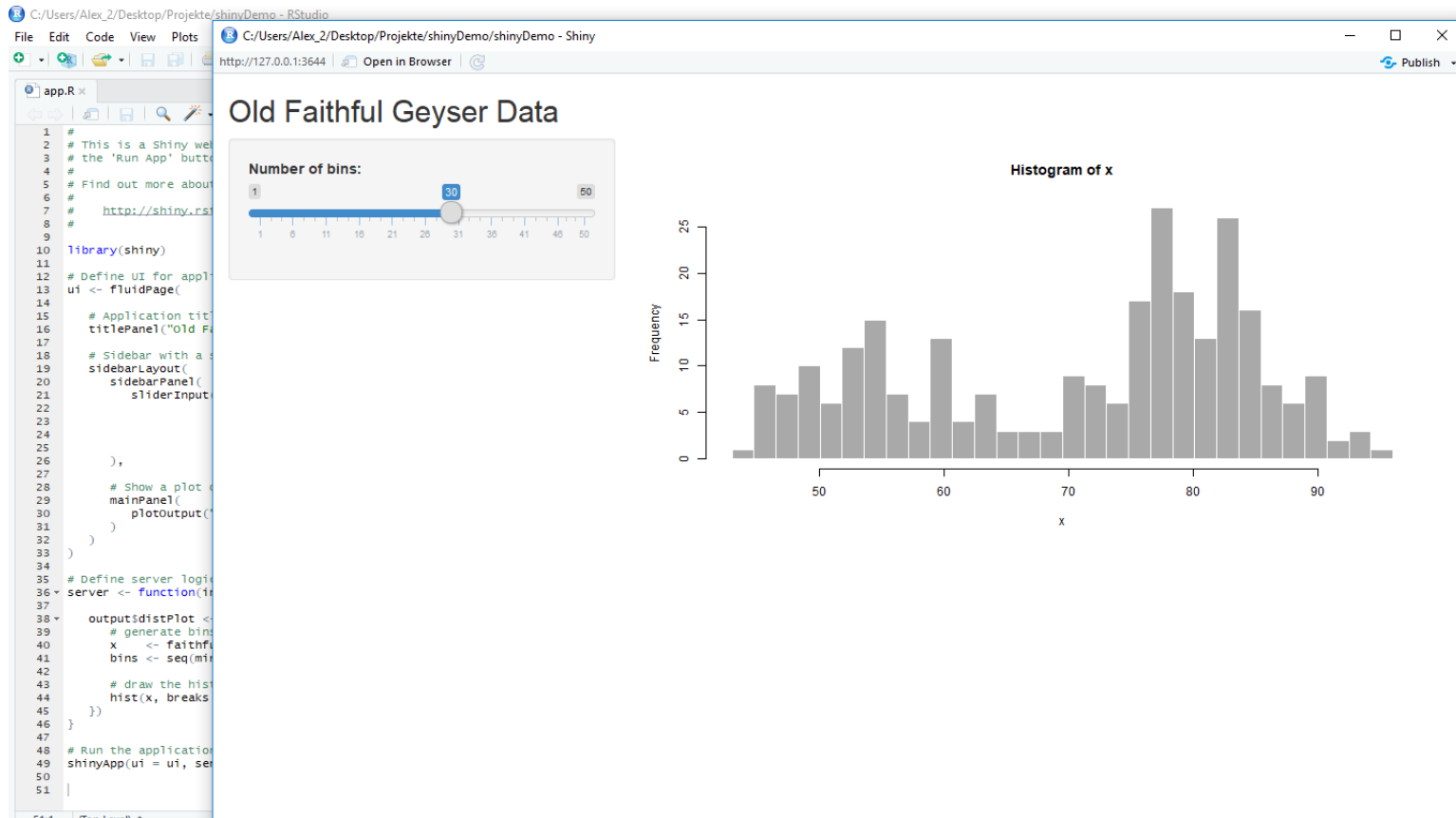
```
C:/Users/Alex_2/Desktop/Projekte/shinyDemo/
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> |
```

R / RStudio / Shiny – demo



R / RStudio / Shiny – demo

The screenshot displays the RStudio environment with a Shiny application running. The application window is titled "Old Faithful Geyser Data" and features a slider input for "Number of bins" set to 7. The histogram, titled "Histogram of x", shows the frequency distribution of the data. The x-axis is labeled "x" and ranges from approximately 45 to 95, with major ticks at 50, 60, 70, 80, and 90. The y-axis is labeled "Frequency" and ranges from 0 to 70, with major ticks every 10 units. The histogram consists of 7 bars of equal width (10 units each). The approximate frequencies for each bin are: 25, 45, 28, 22, 68, 68, and 15.

```
1 # This is a Shiny web application. You can run the application by clicking the 'Run App' button above.
2 # Find out more about Shiny apps here: http://shiny.rstudio.com/
3 # http://shiny.rstudio.com/
4
5 library(shiny)
6
7 # Define UI for application with fluidPage()
8 ui <- fluidPage(
9   # Application title
10  titlePanel("Old Faithful Geyser Data"),
11  # Sidebar with a slider input for the number of bins
12  sidebarLayout(
13    sidebarPanel(
14      sliderInput("bins", "Number of bins:", min = 1, max = 50, value = 7)
15    ),
16    # Show a plot of the histogram of the data
17    mainPanel(
18      plotOutput("distPlot")
19    )
20  )
21
22 # Define server logic
23 server <- function(input, session) {
24   output$distPlot <- renderPlot({
25     # generate bins
26     x <- faithful$waiting
27     bins <- seq(min(x), max(x), length.out = input$bins)
28     # draw the histogram
29     hist(x, breaks = bins)
30   })
31 }
32
33 # Run the application
34 shinyApp(ui = ui, server = server)
```


R / RStudio / Shiny – demo

The image displays two windows from a RStudio environment. The left window, titled 'app.R', shows the source code for a Shiny web application. The code includes comments, library loading, UI definition, and server logic. The right window, titled 'Shiny', shows the rendered application. It features a title 'Old Faithful Geyser Data', a sidebar with a slider input for 'Color' (ranging from 1 to 8), and a main panel with a scatter plot of 'waiting' (y-axis, 50-90) versus 'eruptions' (x-axis, 1.5-5.0). The scatter plot shows two distinct clusters of data points.

```
1 #
2 # This is a Shiny web application. You can run the application
3 # the 'Run App' button above.
4 #
5 # Find out more about building applications with Shiny here:
6 #
7 # http://shiny.rstudio.com/
8 #
9
10 library(shiny)
11
12 # Define UI for application that draws a histogram
13 ui <- fluidPage(
14
15   # Application title
16   titlePanel("Old Faithful Geyser Data"),
17
18   # Sidebar with a slider input for number of color
19   sidebarLayout(
20     sidebarPanel(
21       sliderInput("color",
22                 "Color:",
23                 min = 1,
24                 max = 8,
25                 value = 1)
26     ),
27
28     # Show a plot of the generated distribution
29     mainPanel(
30       plotOutput("distPlot")
31     )
32   )
33 )
34
35 # Define server logic required to draw a histogram
36 server <- function(input, output) {
37
38   output$distPlot <- renderPlot({
39     # generate bins based on input$bins from ui.R
40     x <- faithful[, c(1,2)]
41
42     # another plot
43     plot(x, col = input$color, pch = 19)
44   })
45 }
46
47 # Run the application
48 shinyApp(ui = ui, server = server)
49
50
```

R / RStudio / Shiny – demo

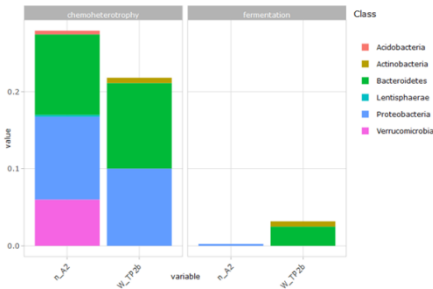
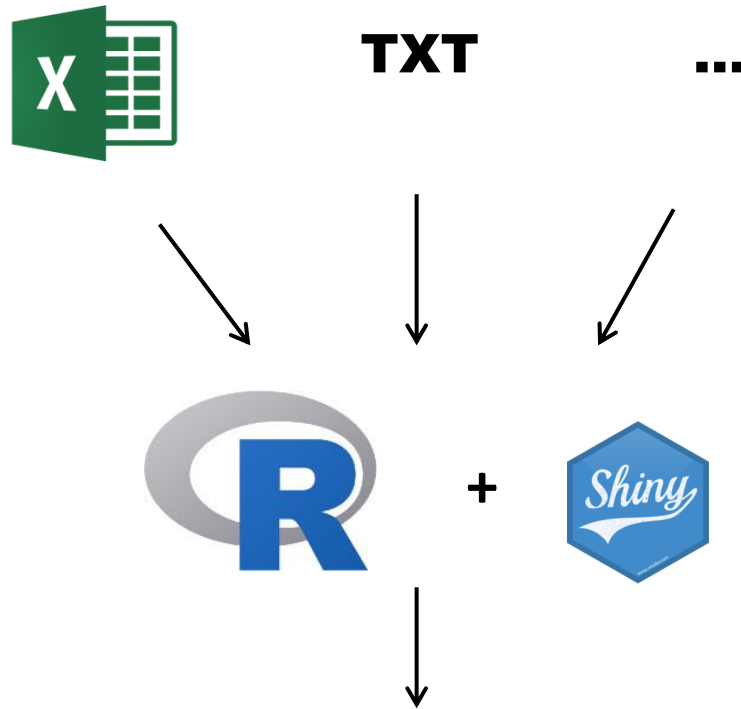
The image shows a screenshot of RStudio and a Shiny web application. The RStudio window on the left displays the source code for a Shiny application. The Shiny application window on the right displays the user interface, which includes a slider input for 'Color' and a scatter plot titled 'Old Faithful Geyser Data'.

```
1 #
2 # This is a Shiny web application. You can run the application by clicking
3 # the 'Run App' button above.
4 #
5 # Find out more about building applications with Shiny here:
6 #
7 #   http://shiny.rstudio.com/
8 #
9
10 library(shiny)
11
12 # Define UI for application that draws a histogram
13 ui <- fluidPage(
14
15   # Application title
16   titlePanel("Old Faithful Geyser Data"),
17
18   # Sidebar with a slider input for number of colors
19   sidebarLayout(
20     sidebarPanel(
21       sliderInput("color",
22                 "Color:",
23                 min = 1,
24                 max = 8,
25                 value = 1)
26     ),
27
28     # Show a plot of the generated distribution
29     mainPanel(
30       plotOutput("distPlot")
31     )
32   )
33 )
34
35 # Define server logic required to draw a histogram
36 server <- function(input, output) {
37
38   output$distPlot <- renderPlot({
39     # generate bins based on input$bins from ui.R
40     x <- faithful[, c(1,2)]
41
42     # another plot
43     plot(x, col = input$color, pch = 19)
44   })
45 }
46
47 # Run the application
48 shinyApp(ui = ui, server = server)
49
50
```

The Shiny application window displays the following UI elements:

- Title:** Old Faithful Geyser Data
- Slider Input:** Color: (range 1 to 8, value 6)
- Scatter Plot:** waiting (y-axis, 50 to 90) vs eruptions (x-axis, 1.5 to 5.0). The plot shows a positive correlation between eruptions and waiting time, with data points colored magenta.

Our tool - Biodiversity visualisier



Biodiversity visualisier – the data



Biodiversity visualiser

Data selection

Select samples (data columns)

Tide pool 1a - winter [W_TP1a] ×

Platform edge 1 - winter [W_E1] ×

Aquarium grown alga 1 [n_A1] ×

[Open sample information](#)

Cutoff (minimal abundance)

0.001



Biodiversity visualiser

Data selection

Select samples (data columns)

Tide pool 1a - winter [W_TP1a] ×
Platform edge 1 - winter [W_E1] ×
Aquarium grown alga 1 [n_A1] ×

[Open sample information](#)

Cutoff (minimal abundance)

0.001

Manage samples and groups

Plot settings

Color set to use

Color set 1

Sample order for plotting

bySelection

Interactive plot

Horizontal bar plot

More plot settings

Biodiversity visualiser

Data selection

Select samples (data columns)

- Tide pool 1a - winter [W_TP1a] ✕
- Platform edge 1 - winter [W_E1] ✕
- Aquarium grown alga 1 [n_A1] ✕

[Open sample information](#)

Cutoff (minimal abundance)

0.001

Manage samples and groups

Plot settings

Color set to use

Color set 1

Sample order for plotting

bySelection

Interactive plot

Horizontal bar plot

[More plot settings](#)

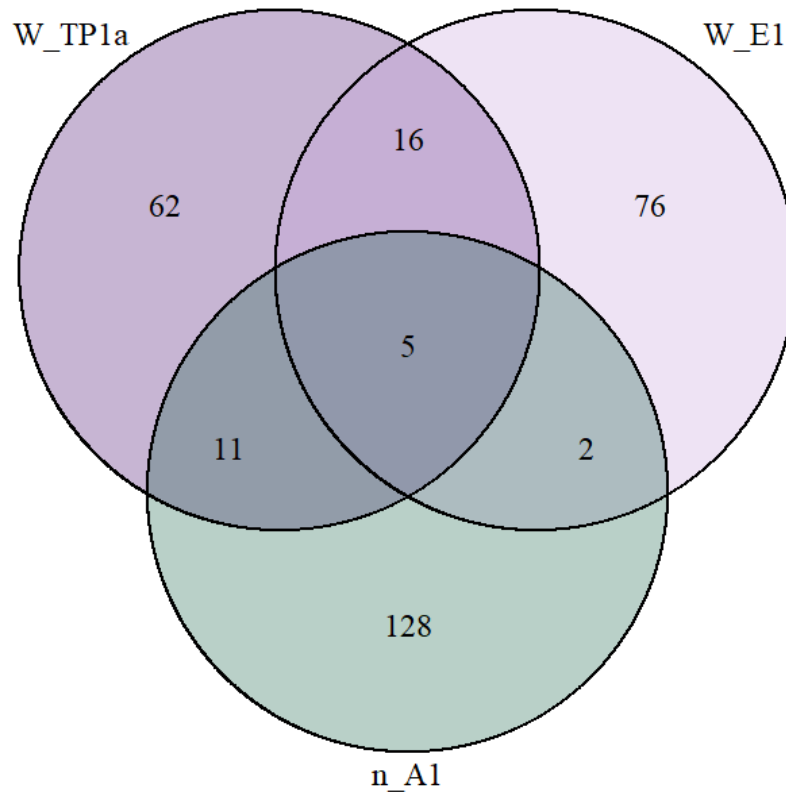
VennDiagram PCA Funct. Annotation Funct. Anno. by Group Abundance by Group Clustering Heat map

[Abundance dist.](#) [Explore annotation](#) [Data tables](#)

Venn diagram showing the overlap of OTUs for selected samples. Select 2 to 5 samples to compare

Size of numbers **Size of labels**

1 2 3 4 5 5 1 2 3 4 5



Biodiversity visualiser

- VennDiagram
 - PCA
 - Funct. Annotation**
 - Funct. Anno. by Group
 - Abundance by Group
 - Clustering
 - He
- Abundance dist.
 - Explore annotation
 - Data tables

Select annotation classes

 Stacked bar plot?

Data selection

Select samples (data columns)

- Tide pool 1a - winter [W_TP1a] ✕
- Platform edge 1 - winter [W_E1] ✕
- Aquarium grown alga 1 [n_A1] ✕

[Open sample information](#)

Cutoff (minimal abundance)

0.001

 Manage samples and groups

Plot settings

Color set to use

Color set 1

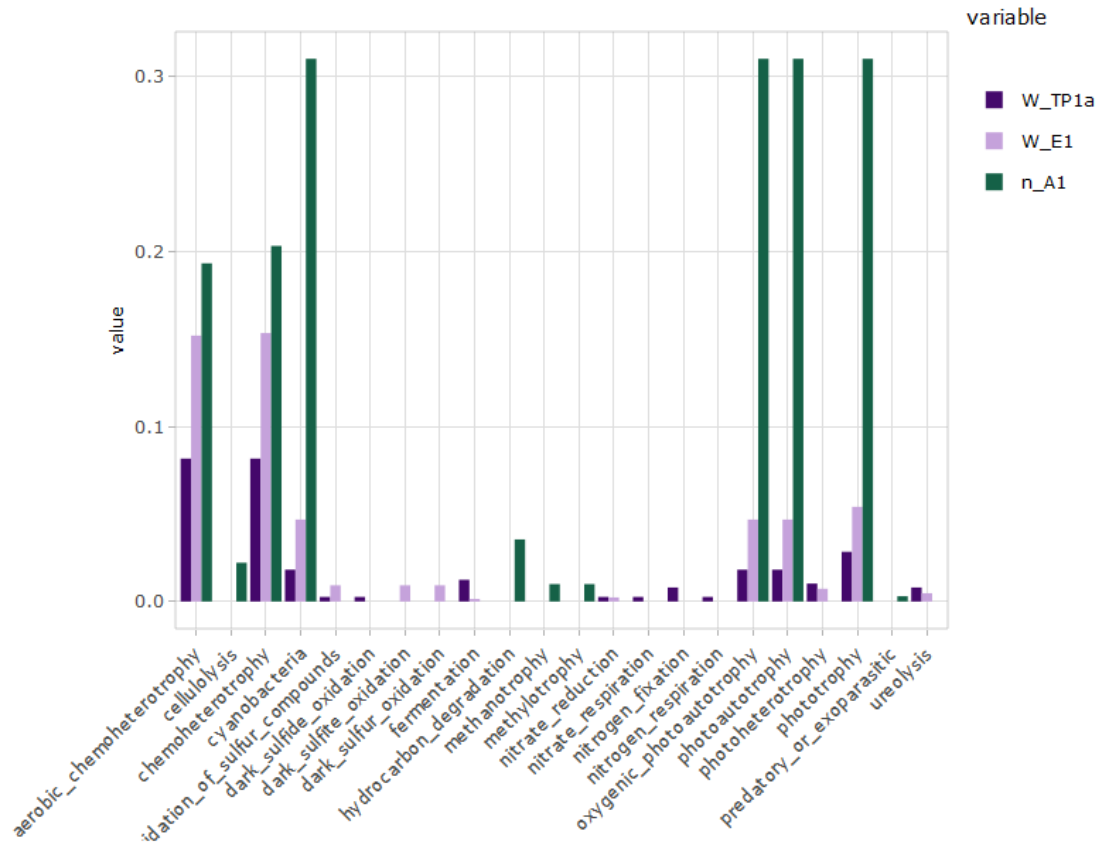
Sample order for plotting

bySelection

 Interactive plot

 Horizontal bar plot

[More plot settings](#)



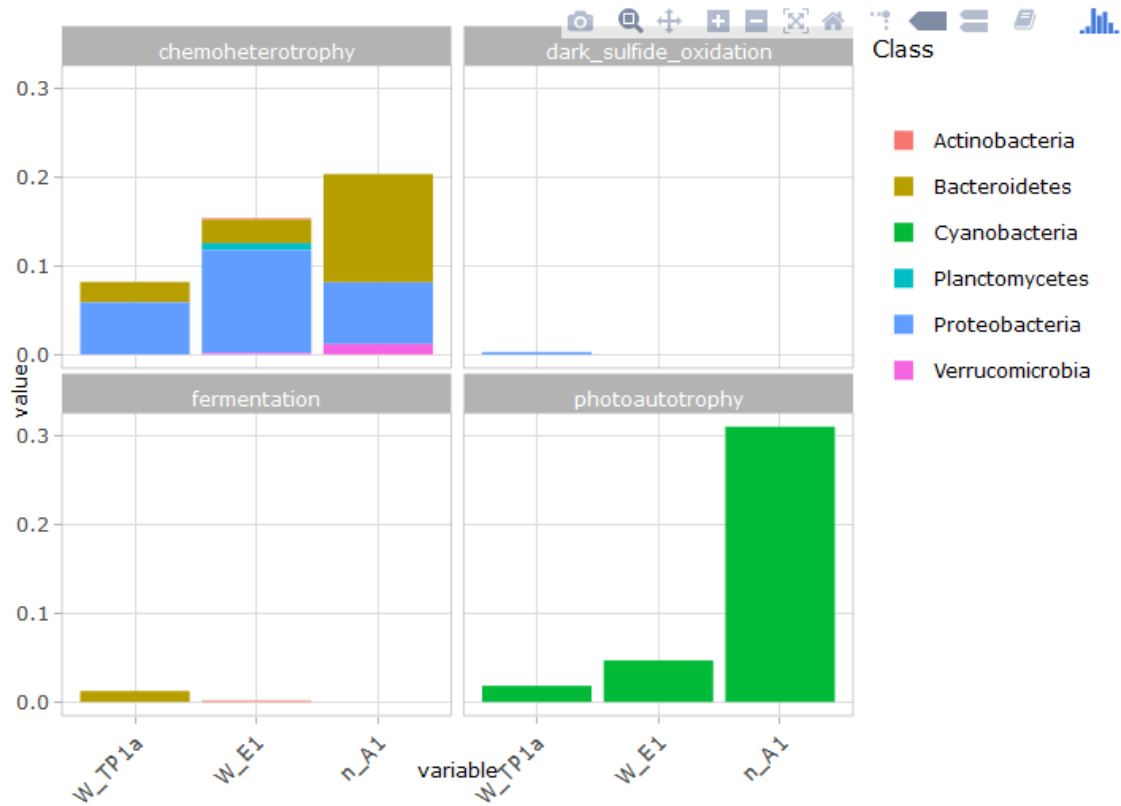
Select annotation classes

chemoheterotrophy ×
 fermentation ×
 photoautotrophy ×
 dark_sulfide_oxidation ×

Select sub class

Phylum ▾

Stacked bar plot?

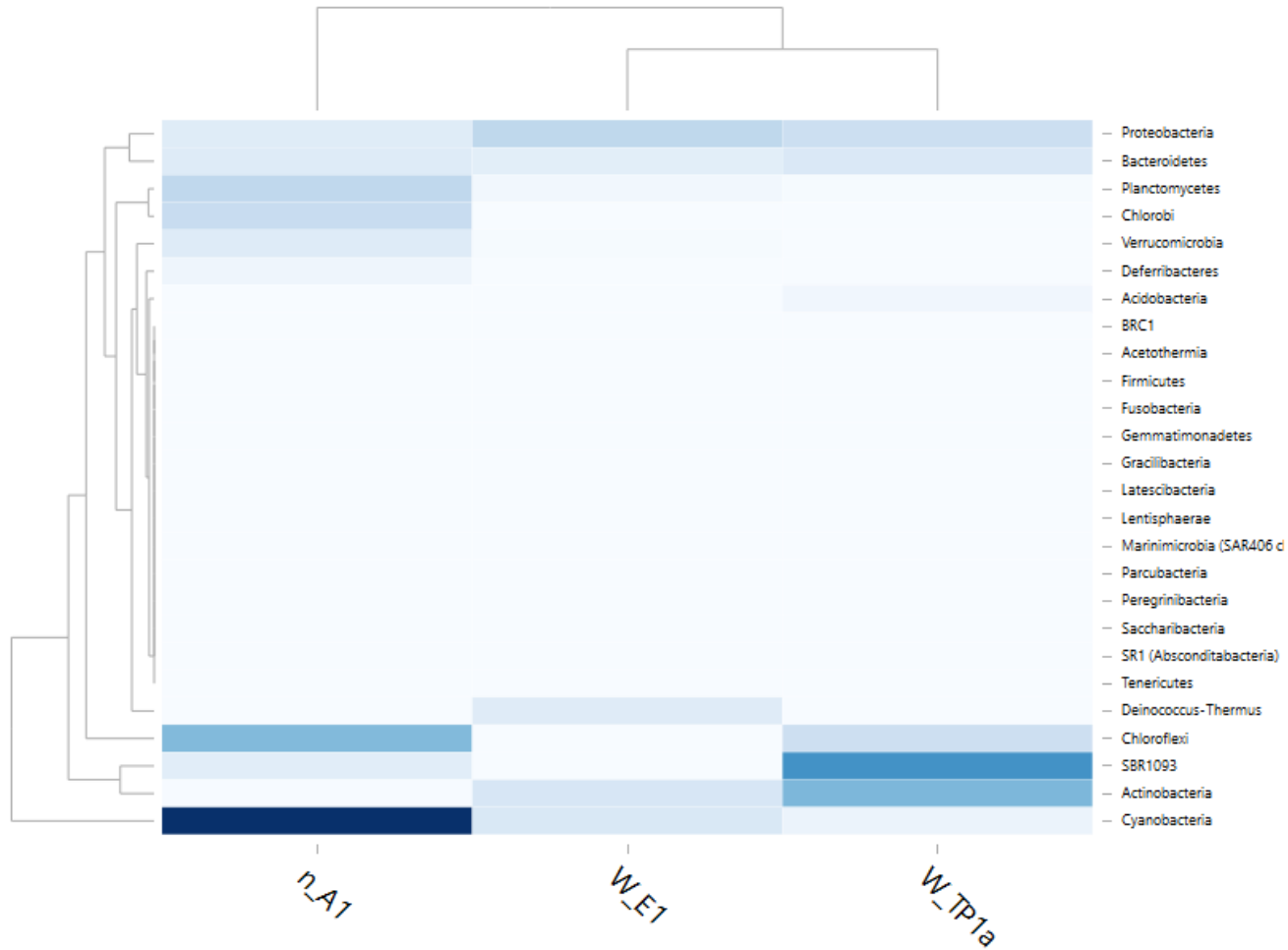


[VennDiagram](#)[PCA](#)[Funct. Annotation](#)[Funct. Anno. by Group](#)[Abundance by Group](#)[Clustering](#)[Heat map](#)[Abundance dist.](#)[Explore annotation](#)[Data tables](#)

Heat map of all non-zero rows for the selected samples

Chose group to sum

Phylum



[VennDiagram](#)[PCA](#)[Funct. Annotation](#)[Funct. Anno. by Group](#)[Abundance by Group](#)[Clustering](#)[Heat map](#)[Abundance dist.](#)[Explore annotation](#)[Data tables](#)

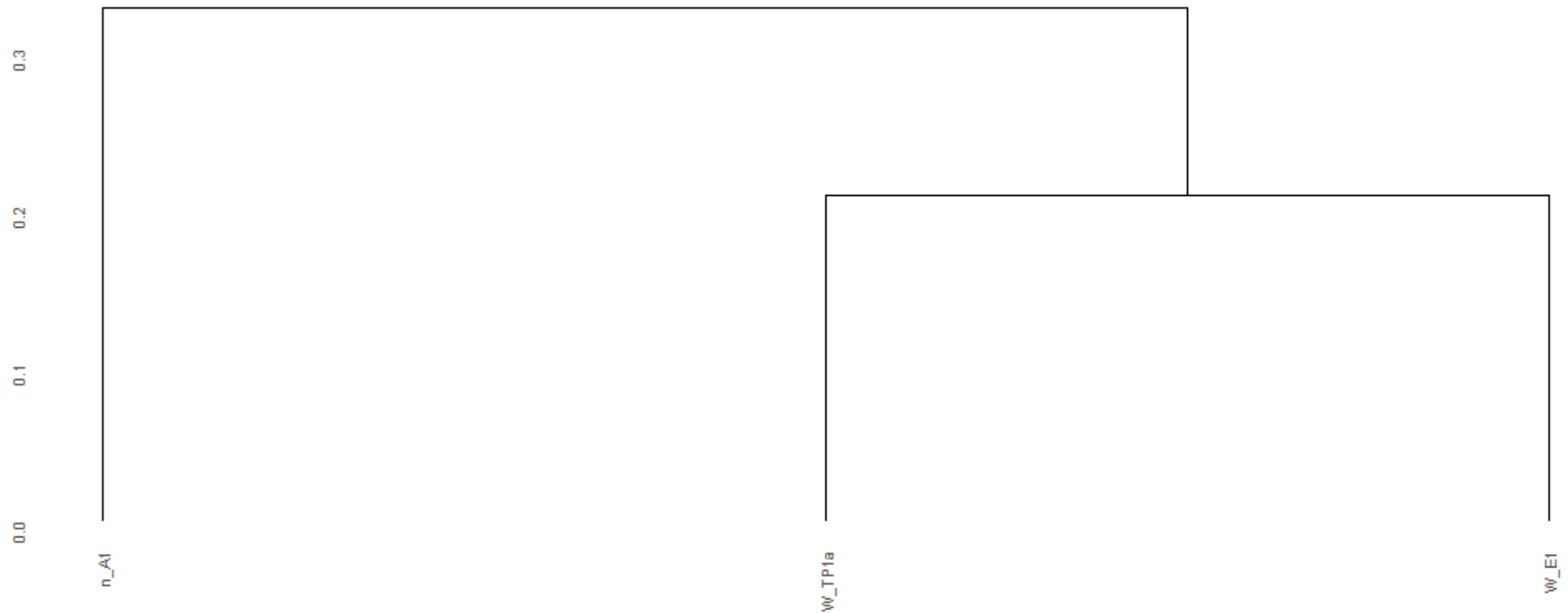
Hierarchical clustering of selected samples. Please select three or more samples.


Distance measure

euclidean

Cluster method

Average linkage



 Save plot

Name

Name long

Sample

Select color1

Select color2

A color selection tool with a large square color gradient on the left and a vertical color bar on the right. A small white circle is positioned on the green part of the gradient. Below the color bar is a dropdown arrow. At the bottom are two buttons: "Save/create" and "Delete".

Table of merged sample. Select row in the table to edit, or create new.

Copy CSV Search:

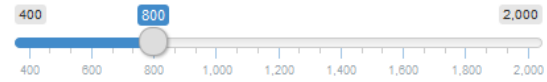
name	samples	color1	color2	symbol	namesLong
n_A1	n_A1	#156147	#366115	9	Aquarium grown alga 1
n_A2	n_A2	#156147	#1D8A66	9	Aquarium grown alga 2

Interactive plot

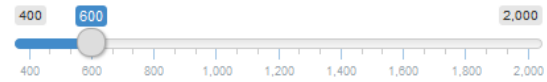
Horizontal bar plot

More plot settings

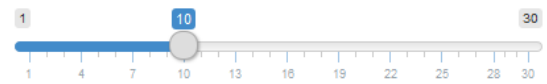
Plot width [pixel]



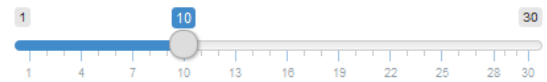
Plot height [pixel]



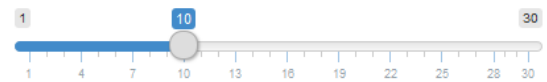
Label size



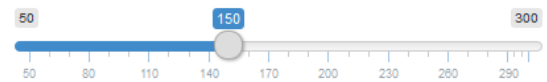
X-axis label size



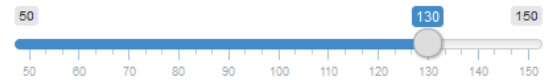
Y-axis label size



Bottom margin (for text labels)



Left margin (for axis labels)



Summary - Biodiversity visualisier

- Share and publish
 - Start locally
 - Publish on managed service
 - Publish on own server
- Interactive analysis
 - No knowledge required to use (may be to interpret) !!!
 - Simple adjust parameters, select sub-groups by clicking
- Reproducibility
 - Technical
 - Data and analysis
- Review, reusage and modification
 - Better review availability
 - Easy to reuse and modify the code or adjust analysis !!!

My experience using interactive data analysis

+++

- More users get involved
- More answered question
- Faster progress
- Deeper insights

-

- 10-20% more time needed to generate interactivity
- Projects get a bit more complex

Alternative R/Shiny software stack

- Python
- Dash (plotly)
 - Python based web framework
 - Fast growing community
 - Slightly more programming knowlegde required



Plotly: <https://plot.ly/>

Dash: <https://plot.ly/products/dash/>

Acknowledgement

Shany Gefen-Treves

Prof. Aaron Kaplan

Prof. Dirk Wagner

Prof. Dan Tchernov

Dr. Fabian Horn

Dr. Haim Treves

Dr. Daniel Lipus



Ministry of Science
and Technology



יד הנדיב
Yad Hanadiv
ياد هندیف



LEON H. CHARNEY
SCHOOL OF MARINE SCIENCES

בית הספר למדעי הים על שם ליאון צ'רני



אוניברסיטת חיפה
University of Haifa



האוניברסיטה העברית בירושלים
THE HEBREW UNIVERSITY OF JERUSALEM



Contact

Alexander Bartholomäus

info@raccoome.de

<http://raccoome.de>



Thank you for your attention!