

```

1 '''
2 Summary: This notebook is a highly specialized variation of the PCA/SVA pipeline that focuses on
3 preprocessing optimization and batch-correction diagnostics. Its defining feature is the manipulation
4 of the SVA design matrices—specifically transforming the continuous age parameter into binned categorical
5 factors (ageL, ageM, ageH) to better model latent covariates. It actively plots SVA residuals and runs
6 linear models to ensure surrogate variables are not inadvertently capturing the primary biological diagnosis,
7 before passing the cleanly corrected data into PCA loading extraction and subsequent Reactome pathway analysis.
8 '''

```

```

1 %load_ext rpy2.ipynthon
2 from google.colab import drive
3 drive.mount('/content/drive')

```

```

1 %R -o DER_01 -o DER_03a -o DER_03b -o DER_tmp -o capstone
2
3 if (!requireNamespace("BiocManager", quietly = TRUE))
4   install.packages("BiocManager")
5
6 BiocManager::install("WGCNA")
7 library(WGCNA)
8 #BiocManager::install("zFPKM")
9 #library(zFPKM)
10 #BiocManager::install("CancerSubtypes")
11 #library(CancerSubtypes)
12 #BiocManager::install("DESeq2")
13 #library(DESeq2)
14
15 install.packages("tidyverse")
16 library(tidyverse)
17
18 capstone_tmp <- read.csv("/content/drive/My Drive/datasets/PEC_capstone_data_map_clinical.csv")
19 capstone <- capstone_tmp[, c(1, 2, 6, 7, 8)]
20
21 #DER_tmp <- read.delim("/content/drive/My Drive/datasets/Controls_only_PEC_Gene_expression_matrix_normalized.txt")
22 DER_tmp <- read.delim("/content/drive/My Drive/datasets/DER-02_PEC_Gene_expression_matrix_TPM.txt")
23 #DER_03_tmp <- DER_tmp[,1]
24 #sql_peaks_tmp <- gsub("\\.\\d+", "", DER_03_tmp)
25 #DER_tmp[,1] <- gsub("c(", "", sql_peaks_tmp, fixed="TRUE")
26
27 DER_01 <- read.delim("/content/drive/My Drive/datasets/DER-02_PEC_Gene_expression_matrix_TPM.txt")
28 #DER_01 <- read.delim("/content/drive/My Drive/datasets/Controls_only_PEC_Gene_expression_matrix_normalized.txt")
29 DER_03 <- DER_01[,1]
30 sql_peaks <- gsub("\\.\\d+", "", DER_03)
31 DER_01[,1] <- gsub("c(", "", sql_peaks, fixed="TRUE")
32 DER_03a <- DER_01[,1] #__ gene names
33 DER_03b <- colnames(DER_01)[-1] #__ sample names
34 DER_01 <- t(scale(t(DER_01)[-1]))) #__ scaling the data (otherwise all of the highly expressed genes
35 #will cluster together even if they have different patterns
36 #among the samples)
37

```

```

1 import numpy #_____ Divide tissues and subjects to brain/peripheral
2 import numpy as np # and CTL/SCZ, respectively.
3 import pandas as pd
4 from sklearn.preprocessing import MinMaxScaler
5 from sklearn.feature_selection import VarianceThreshold
6 from sklearn.preprocessing import StandardScaler
7 from scipy.stats import gaussian_kde
8
9 %R library(data.table)
10 %R BrainSeq <- fread('https://raw.githubusercontent.com/LieberInstitute/brainseq_phase2/master/BrainSeq_Phase2_phenotype_c
11 %R head(BrainSeq)
12 %R E <- readRDS('/content/drive/My Drive/datasets/expression.rds')
13 %R gdt <- readRDS('/content/drive/My Drive/datasets/gdt.rds')
14 %R samp <- fread('https://storage.googleapis.com/gtex_analysis_v7/annotations/GTEX_v7_Annotations_SampleAttributesDS.txt')
15 %R subj <- fread('https://storage.googleapis.com/gtex_analysis_v7/annotations/GTEX_v7_Annotations_SubjectPhenotypesDS.txt')
16 %R samp[, SUBJID := gsub('([^-]*)-([^-]*)-.*', '\\1-\\2', SAMPID)]
17 %R sdt <- merge(samp, subj, by='SUBJID')
18 %R sdt <- merge(samp, subj, by='SUBJID')[SAMPID %in% rownames(E)]
19 #_____
20
21 %R -o BrainSeq
22 %R -o DER_01

```

```

23                                     # _____ Check for bias due to Tissue/Sex/Race.
24 #1. ____BrainSeq[~BrainSeq.Region.str.contains("HIPPO")] #____DLPFC/HIPPO # Replace each line to the line below "3.____
25 #2. ____BrainSeq[~BrainSeq.Race.str.contains("AA")] #____CAUC/AA
26 #3. ____BrainSeq[~BrainSeq.Sex.str.contains("F")] #____M/F
27 BrSeq = pd.DataFrame(BrainSeq)
28 capstone = pd.DataFrame(capstone)
29
30 DER01 = pd.DataFrame(DER_01)
31 DER01.columns = DER_03b
32 DER01 = np.transpose(DER01)
33 DER01.columns = DER_03a
34 DER01 = np.transpose(DER01)
35
36 # _____ I. BrainSeq DATASET(from Lieber Institute)
37
38 arr20 = np.array([])
39 arr20 = BrSeq['BrNum']
40 arr21 = np.array([])
41 arr21 = DER01.columns
42 xy21, x21_ind, y21_ind = np.intersect1d(arr20, arr21,
43                                     return_indices=True)
44 data1 = []
45 for i in x21_ind:
46     data1.append([BrSeq['BrNum'][i], BrSeq['Region'][i], BrSeq['Dx'][i],
47                 BrSeq['Sex'][i], BrSeq['Race'][i], BrSeq['RIN'][i], BrSeq['Age'][i]])
48 DER01_BrSeq = pd.DataFrame(data1, columns=['A', 'B', 'C', 'D', 'E', 'F', 'G'])
49 #DER01_BrSeq = DER01_BrSeq[~DER01_BrSeq.B.str.contains("HIPPO")]
50 # _____
51 %R -o sdt
52 GTx = pd.DataFrame(sdt)
53
54 arr22 = np.array([])
55 arr22 = GTx['SUBJID']
56 xy22, x22_ind, y22_ind = np.intersect1d(arr22, arr21,
57                                     return_indices=True)
58 data2 = []
59 for i in x22_ind:
60     data2.append([GTx['SUBJID'][i], GTx['SMTSD'][i]])
61 df_GTx = pd.DataFrame(data2, columns=['A', 'B'])
62 # _____
63 #FER01 = DER01.columns.to_series().str.contains('CMC')
64 #y23_ind = np.array([i for i, x in enumerate(FER01) if x])
65 #FER02 = DER01.columns.to_series().str.contains('Br')
66 #y24_ind = np.array([i for i, x in enumerate(FER02) if x])
67 #gtx_br = [*y23_ind, *y24_ind]
68
69 #gtx_br = [*y21_ind, *y23_ind, *y22_ind]
70 gtx_br = [*y21_ind, *y22_ind]
71
72 n13_train_scz = pd.DataFrame(DER01)
73 n14_train_scz = n13_train_scz.values[:,]
74 n15_train_scz = n14_train_scz[:,gtx_br] #_____ Filter samples
75
76 # _____ II. CAPSTONE DATASET(from Kelsey)
77 '''
78 arr19a = np.array([])
79 arr19a = capstone['Synapse..individualID']
80 arr19b = np.array([])
81 arr19b = capstone['resource.psychencode.org..individualID']
82 arr21 = np.array([])
83 arr21 = DER01.columns
84 xy22, x22_ind, y22_ind = np.intersect1d(arr19b, arr21,
85                                     return_indices=True)
86 gtx_cr = [*y22_ind]
87
88 data2 = []
89 for i in x22_ind:
90     data2.append([capstone['resource.psychencode.org..individualID'][i],
91                 capstone['diagnosis'][i], capstone['sex'][i], capstone['ethnicity'][i]])
92 capstone_BrSeq2 = pd.DataFrame(data2, columns=['A', 'B', 'C', 'D'])
93
94 BipD = np.where(capstone_BrSeq2.B.str.contains("Bipolar Disorder"))
95 AutD = np.where(capstone_BrSeq2.B.str.contains("Autism Spectrum Disorder"))
96 AffD = np.where(capstone_BrSeq2.B.str.contains("Affective Disorder"))
97 BPD = np.where(capstone_BrSeq2.B.str.contains("BP"))
98
99 Dis_ind = [*BipD, *AutD, *AffD, *BPD]

```

```

100 Dis_ind_flat = [item for sublist in Dis_ind for item in sublist]
101
102 capstone_BrSeq2 = capstone_BrSeq2[~capstone_BrSeq2.B.str.contains("Bipolar Disorder")]
103 capstone_BrSeq2 = capstone_BrSeq2[~capstone_BrSeq2.B.str.contains("Autism Spectrum Disorder")]
104 capstone_BrSeq2 = capstone_BrSeq2[~capstone_BrSeq2.B.str.contains("Affective Disorder")]
105 capstone_BrSeq2 = capstone_BrSeq2[~capstone_BrSeq2.B.str.contains("BP")]
106
107 gj = []
108 for i in range(len(capstone_BrSeq2)):
109     if len(list(capstone_BrSeq2['A'])[i])==0 :
110         gj.append(i)
111     if len(list(capstone_BrSeq2['B'])[i])==0 :
112         gj.append(i)
113     if len(list(capstone_BrSeq2['C'])[i])==0 :
114         gj.append(i)
115     if len(list(capstone_BrSeq2['D'])[i])==0 :
116         gj.append(i)
117
118 res = []
119 [res.append(x) for x in gj if x not in res];
120
121 mask = np.ones(len(capstone_BrSeq2), bool)
122 mask[res] = False
123 capstone_BrSeq2_new = capstone_BrSeq2.iloc[mask]
124
125 len(capstone_BrSeq2_new)
126
127 n13_train_scz = pd.DataFrame(DER01)
128 n14_train_scz = n13_train_scz.values[:,]
129 n15_train_scz = n14_train_scz[:,gtx_cr] #_____ Filter samples
130 Dis_mask = np.ones(n15_train_scz.shape[1], bool)
131 Dis_mask[Dis_ind_flat] = False
132 n15_train_scz = n15_train_scz[:,Dis_mask]
133 n15_train_scz = n15_train_scz[:,mask]
134 '''
135 #_____
136
137 #n13_train_scz_min = n15_train_scz.astype(float) #_____1. No transformation
138
139 ###medianValue = median(logData); #_____2. Median centering data before log2-trans
140 ###medianCtrData = logData-medianValue
141
142 n13_train_scz_min = np.log(n15_train_scz.astype(float)+1) #_____3. Log2 transformation
143
144 #n13_train_scz_min <- zFPKM(n15_train_scz.astype(float)) #_____4. Z-scale transformation (in R, from zFPK
145
146 n13_train_scz_min = np.transpose(n13_train_scz_min) #_____ Shape the array as (samples, genes)
147
148 n13_train_scz_min[np.isnan(n13_train_scz_min)] = 0 #(57820, 414) #_____ Replace NAs with zeroes
149 pd.DataFrame(n13_train_scz_min).isnull().sum()
150
151 print(n13_train_scz_min.shape)
152 jdx = np.argwhere(np.all(n13_train_scz_min[... ,:] == 0, axis=0)) #_____ Remove zero columns
153 n13_train_scz_min = np.delete(n13_train_scz_min, jdx, axis=1)
154 print(n13_train_scz_min.shape)
155
156
157 f = VarianceThreshold().fit(n13_train_scz_min)
158 n13_train_scz_min = n13_train_scz_min[:, f.variances_ > 0.0] #._____
159 print(n13_train_scz_min.shape)
160
161
162 kdx = jdx.tolist() #_____ Final gene list after zero column removal
163 ldx = [item for sublist in kdx for item in sublist]
164 ER_TMM = []
165 DER_tmp.rename(columns = {'gene_id':'GeneName'}, inplace = True)
166 for i in range(len(DER_tmp['GeneName'])[ldx]):
167     ER_TMM.append(DER_tmp['GeneName'][ldx][i])
168 crr1 = np.array([])
169 crr1 = DER_tmp['GeneName']
170 crr2 = np.array([])
171 crr2 = ER_TMM
172 c22, c1_ind, c2_ind = np.intersect1d(crr1, crr2,
173                                     return_indices=True)
174 crr1[c1_ind] = 0
175 crr1 = [i for i in crr1 if i != 0]
176

```

```

177 crr1 = pd.DataFrame(crr1).iloc[f.variances_ > 0.0] #_____ 2
178 #print(pd.DataFrame(crr1).iloc[transform.get_support(indices=True)].shape) #_____ Gene Names(columns)
179
180
181 '''
182 for i in range(n13_train_scz_min.shape[0]): #_____ Z-FPKM(log2) transformation (in Python)
183     n13_train_tmp = n13_train_scz_min[:,i]
184     kernel = gaussian_kde(n13_train_tmp)
185     xi = np.linspace(n13_train_tmp.min(), n13_train_tmp.max(), 100)
186     yi = kernel.evaluate(xi)
187     mu = xi[np.argmax(yi)]
188     U = n13_train_scz_min[n13_train_scz_min > mu].mean()
189     sigma = (U - mu) * np.sqrt(np.pi / 2)
190     n13_train_scz_min[:,i] = (n13_train_tmp - mu) / sigma
191 '''
192
193 %R -i crr1
194 %R -i n13_train_scz_min #_____ Remove zero columns
195 %%R n13_train_scz_min <- n13_train_scz_min[, colSums(n13_train_scz_min != 0) > 0]
196 %%R print(dim(n13_train_scz_min))
197
198 %%R -i n13_train_scz_min #_____ z-transformation (in R)
199 #zscore<- function(x){
200 #     z<- (x - mean(x)) / sd(x)
201 #     return(z)
202 #}
203 #print(dim(n13_train_scz_min))
204 #n13_train_scz_min <- zscore(n13_train_scz_min)
205 #n13_train_scz_min <- n13_train_scz_min[, colSums(n13_train_scz_min != 0) > 0]
206 #print(dim(n13_train_scz_min))
207
208 ## PS: The FPKM counts are already normalised but are absolutely not suitable for
209 ## cross-sample comparisons, i.e., not suitable for differential expression analysis.
210 #_____
211
212 #_____ re-transform from log2(FPKM + z) ->
213 # -> FPKM ->
214 # -> zFPKM scores
215 #z=0.1
216 #exp.fpkM <- 2^expr
217 #exp.fpkM.original <- exp.fpkM - z
218 #exp.zfpkm <- zFPKM(exp.fpkM.original)
219 ##filter out lowly expressed genes
220 #thres <- (ncol(exp.zfpkm) * 30) / 100
221 ##filter all expression values that have absolute zfpkm score above 3.0 in more than 70% of the samples
222 #expr.zfpkm.filtered <- exp.zfpkm[(rowSums(abs(exp.zfpkm) > 3.0)) > thres, ]
223
224 #_____ Traits (CTL/SCZ)
225
226 data4 = [] #I. BrainSeq DATASET(from Lieber Institute)
227 for i in DER01_BrSeq['B']:
228     if i == 'DLPPFC':
229         k = 0
230     if i == 'HIPPO':
231         k = 1
232     data4.append(k)
233 DER01_BrSeq6 = pd.DataFrame(data4, columns=['region'])
234
235 data4 = []
236 for i in DER01_BrSeq['C']:
237     if i == 'Control':
238         k = 0
239     if i == 'Schizo':
240         k = 1
241     data4.append(k)
242 DER01_BrSeq1 = pd.DataFrame(data4, columns=['diag'])
243
244 data4 = []
245 for i in DER01_BrSeq['D']:
246     if i == 'F':
247         k = 0
248     if i == 'M':
249         k = 1
250     data4.append(k)
251 DER01_BrSeq2 = pd.DataFrame(data4, columns=['sex'])
252
253 data4 = []

```

```

254 for i in DER01_BrSeq['E']:
255     if i == 'CAUC':
256         k = 1
257     else:
258         k = 0
259     data4.append(k)
260 DER01_BrSeq3 = pd.DataFrame(data4, columns=['ethn'])
261
262 data4 = []
263 for i in DER01_BrSeq['F']:
264     data4.append(i)
265 DER01_BrSeq4 = pd.DataFrame(data4, columns=['rin'])
266
267 data4 = []
268 for i in DER01_BrSeq['G']:
269     data4.append(i)
270 DER01_BrSeq5 = pd.DataFrame(data4, columns=['age'])
271
272 DER01_BrSeq_Tot = pd.concat([DER01_BrSeq1, DER01_BrSeq2, DER01_BrSeq3,
273                             DER01_BrSeq4, DER01_BrSeq5, DER01_BrSeq6], axis=1)
274
275 n13_trait_path1 = []
276 n13_trait_path1 = pd.DataFrame(DER01_BrSeq_Tot)
277
278 n13_trait_path = n13_trait_path1
279 %R -i n13_trait_path
280
281 #
282 '''
283
284 #II. CAPSTONE DATASET(from Kelsey)
285 #pd.DataFrame(capstone_BrSeq2_new['B']).value_counts()
286 #pd.DataFrame(capstone_BrSeq2_new['C']).value_counts()
287 #pd.DataFrame(capstone_BrSeq2_new['D']).value_counts()
288
289 data4 = []
290 for i in capstone_BrSeq2_new['B']:
291     if i == 'Control':
292         k = 0
293     elif i == 'Schizophrenia':
294         k = 1
295     data4.append(k)
296 DER01_BrFin1 = pd.DataFrame(data4, columns=['diag'])
297
298 data4 = []
299 for i in capstone_BrSeq2_new['C']:
300     if i == 'F':
301         k = 0
302     elif i == 'M':
303         k = 1
304     data4.append(k)
305 DER01_BrFin2 = pd.DataFrame(data4, columns=['sex'])
306
307 data4 = []
308 for i in capstone_BrSeq2_new['D']:
309     if i == 'CAUC':
310         k = 1
311     else:
312         k = 0
313     data4.append(k)
314 DER01_BrFin3 = pd.DataFrame(data4, columns=['ethn'])
315
316 DER01_BrFin = pd.concat([DER01_BrFin1, DER01_BrFin2, DER01_BrFin3], axis=1)
317
318 n13_trait_path1 = []
319 n13_trait_path1 = pd.DataFrame(DER01_BrFin)
320
321 n13_trait_path = n13_trait_path1
322 %R -i n13_trait_path
323
324 #
325 #_____ Clear unused variables
326 del DER01_BrSeq
327 del DER01_BrFin
328 del DER01
329 del n13_train_scz
330 del n14_train_scz
331 del n15_train_scz
332
333

```

```

1 # _____ Defining number of CTLs and SCZs
2 # before removing outliers!!!!!!!!!!
3 # Use it as input to the PCA below
4
5
6 #I. BrainSeq DATASET(from Lieber Institute)
7 DER01_BrSeq = pd.DataFrame(data1, columns=['A', 'B', 'C', 'D', 'E', 'F', 'G'])
8 #DER01_BrSeq = DER01_BrSeq[~DER01_BrSeq.B.str.contains("HIPPO")]
9
10 FER05 = DER01_BrSeq['C'].str.contains('Con')
11 z23_ind = np.array([i for i, x in enumerate(FER05) if x])
12
13 FER06 = DER01_BrSeq['C'].str.contains('Sc')
14 z24_ind = np.array([i for i, x in enumerate(FER06) if x])
15
16 gtx_all = [*z23_ind, *z24_ind]
17 #print(DER01_BrSeq.values[gtx_all])
18
19 n13_pca_scz_min = n13_train_scz_min[gtx_all,]
20 n13_pca_path = n13_trait_path.iloc[gtx_all]
21 #n13_pca_path = n13_trait_path['D'][gtx_all]
22 %R -i n13_pca_path
23
24 DER01_BrSeq['C'].value_counts() #_____ CTLs and SCZs
25
26 # _____
27 '''
28 #II. CAPSTONE DATASET(from Kelsey)
29 FER05 = capstone_BrSeq2_new['B'].str.contains('Con')
30 z23_ind = np.array([i for i, x in enumerate(FER05) if x])
31
32 FER06 = capstone_BrSeq2_new['B'].str.contains('Sc')
33 z24_ind = np.array([i for i, x in enumerate(FER06) if x])
34
35 gtx_all = [*z23_ind, *z24_ind]
36 #print(DER01_BrFin1.values[gtx_all])
37
38 n13_pca_scz_min = n13_train_scz_min[gtx_all,]
39 n13_pca_path = n13_trait_path.iloc[gtx_all]
40 #n13_pca_path = n13_trait_path['diag'][gtx_all]
41 %R -i n13_pca_path
42
43 #DER01_BrFin['diag'].value_counts() #_____ CTLs and SCZs
44 capstone_BrSeq2_new['B'].value_counts()
45 '''

```

```

1 %R -i n13_pca_scz_min #_____ Delete R variables, save files
2 # for WGCNA analysis
3 rm(list=ls()[! ls() %in% c("n13_pca_scz_min", "n13_pca_path", "crr1")])
4 ls()
5
6 #save(n13_pca_scz_min, file = '/content/drive/My Drive/datasets/n13_pca_scz_min.RData')
7 #save(n13_pca_path, file = '/content/drive/My Drive/datasets/n13_pca_path.RData')
8

```

```

1 %R -i n13_pca_scz_min
2 # _____ plot PCA before removing outliers!!!!!!!!!!
3
4 library(ggplot2)
5
6 n13_pca_scz_min <- n13_pca_scz_min[, which(apply(n13_pca_scz_min, 2, var) != 0)]#_Remove zero variance columns from the new
7 genes=paste("gene",seq(1:dim(n13_pca_scz_min)[2]), sep="")
8 colnames(n13_pca_scz_min)=genes
9 row.names(n13_pca_scz_min)=c(paste0("ctl_",seq(1:261)), paste0("scz",seq(1:153))) #___Python command: DER01_BrSeq['C'].val
10 condition1=rep(c("ctl"), each=261)
11 condition2=rep(c("scz"), each=153)
12 condition1 <- append(condition1, condition2)
13 length(condition1)
14
15 #pca_data=prcomp(n13_train_scz_min, center = TRUE, scale = TRUE)
16 #pca_data_perc=round(100*pca_data$sdev^2/sum(pca_data$sdev^2),1)
17 #df_pca_data=data.frame(PC1 = pca_data$x[,1], PC2 = pca_data$x[,2], sample = row.names(n13_train_scz_min), condition1=condi
18 #ggplot(df_pca_data, aes(PC1,PC2, color = condition1))+
19 # geom_point(size=8)+

```

```

20 #      labs(x=paste0("PC1 (",pca_data_perc[1],")"), y=paste0("PC2 (",pca_data_perc[2],")"))
21
22                                     #____ PCA for Z-scaled data
23 pca_data=prcomp(n13_pca_scz_min, center = TRUE, scale = F)
24 pca_data_perc=round(100*pca_data$sdev^2/sum(pca_data$sdev^2),1)
25 df_pca_data=data.frame(PC1 = pca_data$x[,1], PC2 = pca_data$x[,2], sample = row.names(n13_pca_scz_min), condition1=condition1)
26
27                                     #_____ Plot samples
28 ggplot(df_pca_data, aes(PC1,PC2, color = condition1))+
29     geom_point(size=2)+
30     labs(x=paste0("PC1 (",pca_data_perc[1],")"), y=paste0("PC2 (",pca_data_perc[2],")"))
31
32                                     #_____ Plot genes
33 #ggplot(df_pca_data, aes(PC1,PC2, color = sample))+geom_point(size=2)+
34 #     labs(x=paste0("PC1 (",pca_data_perc[1],")"), y=paste0("PC2 (",pca_data_perc[2],")"))+
35 #     theme(legend.position = "none")
36
37 #df_pca_data[df_pca_data$PC1>300000,]                                     #____ How many patients/controls
38 #sum(df_pca_data$PC1>300000)                                           # as outliers in the PC1 axis
39
40 #_____ PCA using FactoMineR package
41
42 ##install.packages("FactoMineR")
43 ##library(FactoMineR)
44 ##install.packages("factoextra")
45 ##library(factoextra)
46 #pca_data <- as.data.frame(n13_train_scz_min)
47 #pca_data$group <- c(rep('ctl',261),rep('scz',153))
48 #pca <- PCA(pca_data[,1:(ncol(pca_data)-1)], graph = F)
49 #fviz_pca_ind(pca,
50 #     geom.ind = "point",
51 #     col.ind = pca_data$group,
52 #     palette = c("#00AFBB", "#E7B800"),
53 #     addEllipses = TRUE,
54 #     legend.title = "Groups")
55

```

```
1 # _____ WGCNA _____
```

```
1 # _____ WGCNA _____
```

```
1 # _____ WGCNA _____
```

```

1 %%R
2
3 sampleTree = hclust(dist(n13_pca_scz_min), method = "average");
4 par(cex = 0.6);
5 par(mar = c(0,4,2,0))
6 plot(sampleTree, main = "Sample clustering to detect outliers",
7     sub="", xlab="", cex.lab = 1.5,cex.axis = 1.5, cex.main = 2)
8
9 abline(h = 160, col = "red");
10 clust = cutreeStatic(sampleTree, cutHeight = 160, minSize = 1)
11 table(clust)
12 keepSamples1 = (clust==1)
13 print(sum(keepSamples1))
14 keepSamples2 = (clust==2)
15 print(sum(keepSamples2))
16
17 keepSamples <- list()
18 keepSamples <- c(keepSamples, keepSamples1+keepSamples2)
19 length(keepSamples)
20
21 dim(n13_pca_scz_min)
22 datExpr = n13_pca_scz_min[which(keepSamples != 0),]
23 print(dim(datExpr))
24
25 nGenes <- ncol(datExpr)
26 nSamples <- nrow(datExpr)
27
28 datTraits <- n13_pca_path[which(keepSamples != 0),]
29 print(dim(datTraits))
30

```

```

1 # _____ defining new number of CTLs and SCZs after
2 # Use it as input to the PCA below
3
4 #I. BrainSeq DATASET(from Lieber Institute)
5 #DER01_BrSeq = pd.DataFrame(data1, columns=['A', 'B', 'C', 'D', 'E', 'F', 'G'])
6 #R -i DER01_BrSeq
7 #R -o DER02_BrSeq DER02_BrSeq <- DER01_BrSeq[which(keepSamples != 0),]
8 #pd.DataFrame(DER02_BrSeq['C']).value_counts()
9
10 %R -o datTraits
11 print(pd.DataFrame(datTraits['diag']).value_counts())
12
13 '''
14 #II. CAPSTONE DATASET(from Kelsey)
15 %R -i capstone_BrSeq2_new
16 %R -o capstone_BrSeq3_new capstone_BrSeq3_new <- capstone_BrSeq2_new[which(keepSamples != 0),]
17 pd.DataFrame(capstone_BrSeq3_new['B']).value_counts()
18 '''

```

```

1 %%R
2 # _____ plot PCA after removing outliers!!!!!!!!!!
3
4 library(ggplot2)
5
6 crr1 <- crr1[which(apply(datExpr, 2, var) != 0),]
7 datExpr <- datExpr[, which(apply(datExpr, 2, var) != 0)]#_Remove zero variance columns from the new dataset
8 save(datExpr, file = '/content/drive/My Drive/datasets/datExpr.RData')
9 save(crr1, file = '/content/drive/My Drive/datasets/crr1.RData')
10
11 genes=paste("gene",seq(1:dim(datExpr)[2]), sep="")
12 colnames(datExpr)=genes
13 row.names(datExpr)=c(paste0("ctl_",seq(1:246)), paste0("scz",seq(1:138))) #__Python command: DER01_BrSeq['C'].value_count
14 condition1=rep(c("ctl"), each=246)
15 condition2=rep(c("scz"), each=138)
16 condition1 <- append(condition1, condition2)
17 length(condition1)
18
19 #pca_data=prcomp(datExpr, center = TRUE, scale = TRUE)
20 #pca_data_perc=round(100*pca_data$sdev^2/sum(pca_data$sdev^2),1)
21 #df_pca_data=data.frame(PC1 = pca_data$x[,1], PC2 = pca_data$x[,2], sample = row.names(datExpr), condition1=condition1)
22 #ggplot(df_pca_data, aes(PC1,PC2, color = condition1))+
23 #   geom_point(size=8)+
24 #   labs(x=paste0("PC1 (",pca_data_perc[1],")"), y=paste0("PC2 (",pca_data_perc[2],")"))
25
26 #_____ PCA for Z-scaled data
27 #pca_data=prcomp(datExpr, center = TRUE, scale = F)
28 #pca_data_perc=round(100*pca_data$sdev^2/sum(pca_data$sdev^2),1)
29 #df_pca_data=data.frame(PC1 = pca_data$x[,1], PC2 = pca_data$x[,2], sample = row.names(datExpr), condition1=condition1)
30
31 #_____Plot samples
32 #ggplot(df_pca_data, aes(PC1,PC2, color = condition1))+
33 #   geom_point(size=2)+
34 #   labs(x=paste0("PC1 (",pca_data_perc[1],")"), y=paste0("PC2 (",pca_data_perc[2],")"))
35
36 #_____Plot genes
37 #ggplot(df_pca_data, aes(PC1,PC2, color = sample))+geom_point(size=2)+
38 #   labs(x=paste0("PC1 (",pca_data_perc[1],")"), y=paste0("PC2 (",pca_data_perc[2],")"))+
39 #   theme(legend.position = "none")
40 #_____ PCA using FactoMineR package
41
42 ##install.packages("FactoMineR")
43 ##library(FactoMineR)
44 ##install.packages("factoextra")
45 ##library(factoextra)
46 #pca_data <- as.data.frame(datExpr)
47 #pca_data$group <- c(rep('ctl',250),rep('scz',139))
48 #pca <- PCA(pca_data[,1:(ncol(pca_data)-1)], graph = F)
49 #fviz_pca_ind(pca,
50 #   geom.ind = "point",
51 #   col.ind = pca_data$group,
52 #   palette = c("#00AFBB", "#E7B800"),
53 #   addEllipses = TRUE,
54 #   legend.title = "Groups")
55

```

```

1 %%R
2 # _____ Matrix factorization (age parameter as thr
3
4 #load(file = '/content/drive/My Drive/datasets/datExpr.RData')
5 #load(file = '/content/drive/My Drive/datasets/datTraits.RData')
6 #load(file = '/content/drive/My Drive/datasets/crr1.RData')
7
8 datTraits <- n13_pca_path[which(keepSamples != 0),]
9
10 xs=quantile(datTraits$age,c(0,1/3,2/3,1))
11 xs[1]=xs[1]-.00005
12 datTraits <- datTraits %>% mutate(category=cut(age, breaks=xs,
13   labels=c("low","middle","high"),ordered_result = TRUE))
14
15 datExpr = t(datExpr)
16 datTraits$diag <- as.factor(datTraits$diag)
17 datTraits$ethn <- as.factor(datTraits$ethn)
18 datTraits$sex <- as.factor(datTraits$sex)
19 datTraits$age <- as.factor(as.numeric(as.factor(datTraits$category)))
20 datTraits$region <- as.factor(datTraits$region)
21
22 out = binarizeCategoricalVariable(datTraits$age,
23   includePairwise = FALSE,
24   includeLevelVsAll = TRUE);
25 datTraits$ageL <- as.factor(out[,1])
26 datTraits$ageM <- as.factor(out[,2])
27 datTraits$ageH <- as.factor(out[,3])
28 colnames(datExpr) <- c(rownames(datTraits[,c(-4:-5,-7)] ))
29 rownames(datExpr) <- t(crr1)
30 datExpr = t(datExpr)
31
32 datTraits<-datTraits[,c(-4:-5,-7)]
33 print(head(datTraits))
34

```

```

1 %%R                                     # _____ SVA analysis   (AG3/5)
2
3
4 if (!requireNamespace('BiocManager', quietly = TRUE))
5   install.packages('BiocManager')
6
7 BiocManager::install('sva')
8 library(sva)
9
10 #mod0 = model.matrix(~as.factor(ethn)+as.factor(sex)+as.factor(age)+as.factor(region), data=datTraits) # _____
11 #mod = model.matrix(~as.factor(diag)+as.factor(ethn)+as.factor(sex)+as.factor(age)+as.factor(region), data=datTraits)
12
13 mod0 = model.matrix(~as.factor(ethn)+as.factor(sex)+as.factor(ageL)+as.factor(ageH)+as.factor(region), data=datTraits) #_
14 mod = model.matrix(~as.factor(diag)+as.factor(ethn)+as.factor(sex)+as.factor(ageL)+as.factor(ageH)+as.factor(region), data=
15
16 n.sv = num.sv(t(datExpr),mod,method="leek")
17 svobj = sva(t(datExpr),mod,mod0,n.sv=n.sv)
18
19 cleanY = function(y, mod, svobj) {
20   X = cbind(mod, svobj)
21   Hat = solve(t(X) %*% X) %*% t(X)
22   beta = (Hat %*% t(y))
23   rm(Hat)
24   gc()
25   P = ncol(mod)
26   return(y - t(as.matrix(X[, -c(1:P)])) %*% beta[-c(1:P),]))
27 }
28
29 datExpr_cln <- cleanY(t(datExpr), mod, svobj$sv)
30
31 # _____
32 #mod0 = model.matrix(~as.factor(Gender)+as.factor(Population), data=pheno)
33 #mod = model.matrix(~as.factor(inversion_genotype)+as.factor(Gender)+as.factor(Population), data=pheno)
34 #n.sv = num.sv(edata,mod,method="leek")
35 #svobj = sva(edata,mod,mod0,n.sv=n.sv)
36

```

```

1 %%R
2
3 fit1 = lm.fit(mod,datExpr)
4 modsv = cbind(mod,svobj$sv)

```

```

5 fitsv = lm.fit(modsv,datExpr)
6
7 plot(fitsv$coefficients[2,], fit1$coefficients[2,],col=2,
8      xlab="SVA",ylab="linear model",xlim=c(-1,1),ylim=c(-1,1))
9 abline(c(0,1),col=1,lwd=3)
10

```

```

1 %%R
2
3 summary(lm(svobj$sv ~ datTraits$diag))
4

```

```

1 %%R
2
3 boxplot(svobj$sv[,3] ~ datTraits$diag)
4 points(svobj$sv[,3] ~ jitter(as.numeric(datTraits$diag)),col=as.numeric(datTraits$diag))
5

```

```

1 %%R
2 # _____ plot PCA after SVA analysis!!!!!!!!!!!!
3
4 datExpr_cln <- t(datExpr_cln)
5
6 library(ggplot2)
7
8 #crr1 <- crr1[which(apply(datExpr_cln, 2, var) != 0),]
9 #datExpr_cln <- datExpr_cln[,which(apply(datExpr_cln, 2, var) != 0)]#_Remove zero variance columns from the new dataset
10
11 genes=paste("gene",seq(1:dim(datExpr_cln)[2]), sep="")
12 colnames(datExpr_cln)=genes
13 row.names(datExpr_cln)=c(paste0("ctl_",seq(1:246)), paste0("scz",seq(1:138))) #__Python command: DER01_BrSeq['C'].value_c
14 condition1=rep(c("ctl"), each=246)
15 condition2=rep(c("scz"), each=138)
16 condition1 <- append(condition1, condition2)
17 length(condition1)
18
19 #_____ PCA for Z-scaled data
20 pca_data=prcomp(datExpr_cln, center = TRUE, scale. = F)
21 pca_data_perc=round(100*pca_data$sdev^2/sum(pca_data$sdev^2),1)
22 df_pca_data=data.frame(PC1 = pca_data$x[,1], PC2 = pca_data$x[,2], sample = row.names(datExpr_cln), condition1=condition1)
23
24 #_____Plot samples
25 ggplot(df_pca_data, aes(PC1,PC2, color = condition1))+
26   geom_point(size=2)+
27   labs(x=paste0("PC1 (",pca_data_perc[1],")"), y=paste0("PC2 (",pca_data_perc[2],")"))

```

```

1 %%R
2
3 library(devtools)
4 install_github("vqv/ggbiplot")
5 library(ggbiplot)
6 library(ggplot2)
7
8 # Plot the PCA plots
9 hvPCA1<-ggbiplot(pca_data,choices=c(1,2),scale=0,groups=datTraits$diag, ellipse=T,var.axes=F) + scale_color_manual(values=c(
10 hvPCA2<-ggbiplot(pca_data,choices=c(1,2),scale=0,groups=datTraits$ethn, ellipse=T,var.axes=F) + scale_color_manual(values=c(
11 hvPCA3<-ggbiplot(pca_data,choices=c(1,2),scale=0,groups=datTraits$sex, ellipse=T,var.axes=F) + scale_color_manual(values=c(
12 hvPCA4<-ggbiplot(pca_data,choices=c(1,2),scale=0,groups=datTraits$region, ellipse=T,var.axes=F) + scale_color_manual(values=
13
14
15 #hvPCA5<-ggbiplot(pca_data,choices=c(1,2),scale=0,groups=datTraits$age, ellipse=T,var.axes=F) + scale_color_manual(values=c(
16 #hvPCA5<-ggbiplot(pca_data,choices=c(1,2),scale=0,groups=datTraits$age, ellipse=T,var.axes=F) + scale_color_manual(values=c(
17
18

```

```

1 %%R
2
3 # Show the diagnosis plot
4 hvPCA1

```

```

1 %%R
2

```

```
3 # Show the ethnicity plot
4 hvPCA2
```

```
1 %%R
2
3 # Show the sex plot
4 hvPCA3
```

```
1 %%R
2
3 # Show the region plot
4 hvPCA4
```

```
1 %%R
2
3 datTraits1 <- n13_pca_path[which(keepSamples != 0),]
4 xs=quantile(datTraits1$age,c(0,1/3,2/3,1))
5 xs[1]=xs[1]-.00005
6 datTraits1 <- datTraits1 %>% mutate(category=cut(age, breaks=xs,
7   labels=c("low","middle","high"),ordered_result = TRUE))
8 datTraits1$diag <- as.factor(datTraits1$diag)
9 datTraits1$ethn <- as.factor(datTraits1$ethn)
10 datTraits1$sex <- as.factor(datTraits1$sex)
11 datTraits1$age <- as.factor(as.numeric(as.factor(datTraits1$category)))
12 datTraits1$region <- as.factor(datTraits1$region)
13
14 hvPCA5<-ggbiplot(pca_data,choices=c(1,2),scale=0,groups=datTraits1$age, ellipse=T,var.axes=F) + scale_color_manual(values=c
15
16 # Show the age plot
17 hvPCA5
```

```
1 %%R                                     # _____ age parameter as
2
3 datExpr <- datExpr_cln
4 datTraits <- n13_pca_path[which(keepSamples != 0),]
5
6 xs=quantile(datTraits$age,c(0,1/3,2/3,1))
7 xs[1]=xs[1]-.00005
8 datTraits <- datTraits %>% mutate(category=cut(age, breaks=xs,
9   labels=c("low","middle","high"),ordered_result = TRUE))
10
11 datExpr = t(datExpr)
12 datTraits$diag <- as.factor(datTraits$diag)
13 datTraits$ethn <- as.factor(datTraits$ethn)
14 datTraits$sex <- as.factor(datTraits$sex)
15 datTraits$age <- as.factor(as.numeric(as.factor(datTraits$category)))
16 datTraits$region <- as.factor(datTraits$region)
17
18 out = binarizeCategoricalVariable(datTraits$age,
19 includePairwise = FALSE,
20 includeLevelVsAll = TRUE);
21 datTraits$ageL <- as.factor(out[,1])
22 datTraits$ageM <- as.factor(out[,2])
23 datTraits$ageH <- as.factor(out[,3])
24 colnames(datExpr) <- c(rownames(datTraits[,c(-4:-5,-7)]) )
25 rownames(datExpr) <- t(crr1)
26 datExpr = t(datExpr)
27
28 datTraits<-datTraits[,c(-4:-5,-7)]
29 print(head(datTraits))
30
```

```
1 %%R
2 # _____ Additional PCA analysis
3
4 #load(file = '/content/drive/My Drive/datasets/m13_train_scz_min')
5 #load(file = '/content/drive/My Drive/datasets/m13_trait_path.RData')
6 #load(file = '/content/drive/My Drive/datasets/crr1.RData')
7
8 #m13_train_scz_min = t(m13_train_scz_min)
9 #colnames(m13_train_scz_min) <- c(rownames(m13_trait_path))
10 #rownames(m13_train_scz_min) <- t(crr1)
11 #m13_trait_path$diag <- as.factor(m13_trait_path$diag)
12 #m13_train_scz_min = t(m13_train_scz_min)
```

```

13 # _____ PRCOMP (& randomForest prediction)
14
15 install.packages("randomForest")
16 install.packages("caret")
17 library(caret)
18 library(randomForest)
19
20 colnames(datExpr) <- t(crr1)
21 #datTraits$diag <- as.factor(datTraits$diag)
22
23 set.seed(1234)
24 new_train_scz_min <- datExpr[sample(nrow(datExpr)),]
25 new_trait_path <- datTraits[sample(nrow(datTraits)),]
26 new_trait_path <- new_trait_path[match(rownames(new_train_scz_min), rownames(new_trait_path)),]
27 set.seed(NULL)
28
29 pca.train <- new_train_scz_min[1:(as.integer(0.7*nrow(new_train_scz_min))),]
30 pca.test <- new_train_scz_min[-(1:(as.integer(0.7*nrow(new_train_scz_min))),),]
31 pca_trait_train <- new_trait_path[1:(as.integer(0.7*nrow(new_trait_path))),]
32 pca_trait_test <- new_trait_path[-(1:(as.integer(0.7*nrow(new_trait_path))),),]
33 pca.test <- pca.test[,which(apply(pca.train, 2, var) != 0)]
34 pca.train <- pca.train[,which(apply(pca.train, 2, var) != 0)]
35 prin_comp <- prcomp(pca.train, scale. = F)
36
37 #std_dev <- prin_comp$sdev #_____ Plots
38 #pr_var <- std_dev^2
39 #prop_varex <- pr_var/sum(pr_var)
40
41 #plot(prop_varex, xlab = "Principal Component",
42 #      ylab = "Proportion of Variance Explained",
43 #      type = "b")
44
45 #plot(cumsum(prop_varex), xlab = "Principal Component",
46 #      ylab = "Cumulative Proportion of Variance Explained",
47 #      type = "b")
48
49 #_____randomForest
50 train.data <- data.frame(diag = pca_trait_train$diag, prin_comp$x)
51 train.data <- train.data[,1:31]
52
53 RFmodel_Fit <- randomForest(diag ~ .,data = train.data, method = "anova")
54 RFmodel_Fit
55 test.data <- predict(prin_comp, newdata = pca.test)
56 test.data <- as.data.frame(test.data)
57 test.data <- test.data[,1:30]
58 RFmodel_Fit.prediction <- predict(RFmodel_Fit, test.data)
59 print(confusionMatrix(RFmodel_Fit.prediction, pca_trait_test$diag)$overall[1])
60 print(RFmodel_Fit$confusion)
61
62 #_____ PCATOOLS
63
64 if (!requireNamespace('BiocManager', quietly = TRUE))
65   install.packages('BiocManager')
66
67 BiocManager::install('PCAtools')
68 library(PCAtools)
69
70 #_____ PCATOOLS(various Plotloadings and ENSEMBL to SYMBOL translation)
71
72 organism = "org.Hs.eg.db"
73 BiocManager::install(organism, character.only = TRUE, force = TRUE)
74 library(organism, character.only = TRUE)
75
76 require(org.Hs.eg.db)
77 mapping <- mapIds(
78   org.Hs.eg.db,
79   keys = colnames(datExpr),
80   column = 'SYMBOL',
81   keytype = 'ENSEMBL')
82 colnames(datExpr) <- make.unique(ifelse(is.na(mapping), colnames(datExpr), mapping))
83
84 p <- pca(t(datExpr), metadata = datTraits, center = TRUE,
85         scale = F, removeVar = 0.1)
86

```

```

1 %R                                     # _____ age parameter as
2
3 eigencorplot(p,                        #_ 3a.
4 components = getComponents(p, 1:6),
5 metavars = c("diag", "sex", "ethn", "ageL", "ageH"),
6 col = c('darkblue', 'blue2', 'black', 'red2', 'darkred'),
7 cexCorval = 0.7,
8 colCorval = 'white',
9 fontCorval = 2,
10 posLab = 'bottomleft',
11 rotLabX = 45,
12 posColKey = 'top',
13 cexLabColKey = 1.5,
14 scale = TRUE,
15 main = bquote(Principal ~ component ~ Pearson ~ r^2 ~ clinical ~ correlates),
16 plotRsquared = TRUE,
17 corFUN = 'pearson',
18 corUSE = 'pairwise.complete.obs',
19 corMultipleTestCorrection = 'BH',
20 signifSymbols = c('****', '***', '**', '*', ''),
21 signifCutpoints = c(0, 0.0001, 0.001, 0.01, 0.05, 1))

```

```

1 %R
2
3 biplot(p, showLoadings = TRUE, labSize = 5, pointSize = 5, sizeLoadingsNames = 5) #_ 1.
4

```

```

1 %R
2
3 pairsplot(p)                            #_ 2.
4

```

```

1 %R
2
3 plotloadings(p,                        #_ 5.
4   components = getComponents(p, c(1,2,3,4)),
5   rangeRetain = 0.1,
6   labSize = 3.0,
7   absolute = FALSE,
8   title = 'Loadings plot',
9   subtitle = 'Misc PCs',
10  caption = 'Top 10% variables',
11  shape = 23, shapeSizeRange = c(1, 16),
12  drawConnectors = FALSE)
13

```

```

1 %R
2
3 plotloadings(p,                        #_ 6.
4   components = getComponents(p, c(1,2,3,4,5)),
5   rangeRetain = 0.001,
6   labSize = 3.0,
7   title = 'Loadings plot',
8   subtitle = 'PC1, PC2, PC3, PC4, PC5',
9   caption = 'Top 1% variables',
10  drawConnectors = TRUE)
11

```

```

1 %R
2
3 prin_comp <- prcomp(datExpr, scale. = F)
4 expl_var <- prin_comp$sdev^2/sum(prin_comp$sdev^2)
5 barplot(expl_var[1:50], ylab="Variance",
6         names.arg=paste0("PC",seq(1:50)), col="darkred")
7
8 N_perm <- 10
9 expl_var_perm <- matrix(NA, ncol = length(prin_comp$sdev), nrow = N_perm)
10 for(k in 1:N_perm)
11 {
12   expr_perm <- apply(datExpr,2,sample)
13   PC_perm <- prcomp(expr_perm, center=TRUE, scale=FALSE)
14   expl_var_perm[k,] <- PC_perm$sdev^2/sum(PC_perm$sdev^2)
15 }
16 plot(expl_var[1:200]~seq(1:200), ylab="Variance",

```

```

17 col="green", type='o', xlab="PC")
18 lines(colMeans(expl_var_perm)[1:200]~seq(1:200),col="red")
19 legend("topright", c("original data", "data permutation"),
20 fill=c("green","red"), inset=0.02)
21
22 pval <- apply(t(expl_var_perm) >= expl_var,1,sum) / N_perm
23 plot(pval[1:200]~seq(1:200),col="darkred",type='o',
24 xlab="PC",ylab="p-value")
25 optPC<-head(which(pval>=0.05),1)-1
26 mtext(paste0("OPTIMAL PC = ",optPC))

```

```

1 ##R
2
3 install.packages("tsne")
4 ##install.packages("Rtsne")
5 library(tsne)
6 ##library(Rtsne)
7 library(ggplot2)
8
9 nComponents <- 118
10 seed <- 1234
11 set.seed(seed)
12
13 dat.filter.BCV.tsne<-tsne(pca_data$x[,1:nComponents],perplexity=20,max_iter=2000,whiten = FALSE)
14
15 tsne_plot <- data.frame(x = dat.filter.BCV.tsne[,1], y = dat.filter.BCV.tsne[,2], col = datTraits$diag)
16 ggplot(tsne_plot) + geom_point(aes(x=x, y=y, color=col, size=1))

```

```

1 ##R                                     # _____ Create dataframe with the top genes of the
2                                     # Use it as input to Reactome enrichment ana
3
4 ph_all = list()
5 pl_all = list()
6
7 p1h<-order(p$loadings[1], decreasing=T)[1:5]
8 p1l<-order(p$loadings[1], decreasing=F)[1:5]
9
10 p2h<-order(p$loadings[2], decreasing=T)[1:5]
11 p2l<-order(p$loadings[2], decreasing=F)[1:5]
12
13 p3h<-order(p$loadings[3], decreasing=T)[1:5]
14 p3l<-order(p$loadings[3], decreasing=F)[1:5]
15
16 p4h<-order(p$loadings[4], decreasing=T)[1:5]
17 p4l<-order(p$loadings[4], decreasing=F)[1:5]
18
19 p5h<-order(p$loadings[5], decreasing=T)[1:5]
20 p5l<-order(p$loadings[5], decreasing=F)[1:5]
21
22 ph_all[[1]] <- rownames(p$loadings[p1h,])
23 ph_all[[2]] <- rownames(p$loadings[p2h,])
24 ph_all[[3]] <- rownames(p$loadings[p3h,])
25 ph_all[[4]] <- rownames(p$loadings[p4h,])
26 ph_all[[5]] <- rownames(p$loadings[p5h,])
27
28 pl_all[[1]] <- rownames(p$loadings[p1l,])
29 pl_all[[2]] <- rownames(p$loadings[p2l,])
30 pl_all[[3]] <- rownames(p$loadings[p3l,])
31 pl_all[[4]] <- rownames(p$loadings[p4l,])
32 pl_all[[5]] <- rownames(p$loadings[p5l,])

```

```

1 ##R
2
3 if (!requireNamespace("BiocManager", quietly = TRUE))
4   install.packages("BiocManager")
5
6 BiocManager::install("biomaRt", force = TRUE)
7 library(biomaRt)
8
9 mart <- useMart("ENSEMBL_MART_ENSEMBL")
10 mart <- useDataset("hsapiens_gene_ensembl", mart)
11
12 #devtools::install_github(c("GuangchuangYu/DOSE", "GuangchuangYu/enrichplot", "GuangchuangYu/GOSemSim", "eliocamp/ggnewscale"))
13
14 BiocManager::install("clusterProfiler")

```

```

15 library(clusterProfiler)
16
17 BiocManager::install("ReactomePA")
18 library(ReactomePA)
19
20 #organism = "org.Hs.eg.db"
21 #BiocManager::install(organism, character.only = TRUE)
22 #library(organism, character.only = TRUE)
23

```

```

1 %%R                                     # _____ Reactome pathway gene set enrichment analy
2
3 ll = list()
4 PCAGeneSets_tmp = list()
5                                     # _____ 1.
6 PCAGeneSets_tmp <- list(ph_all[[1]],ph_all[[2]],ph_all[[3]],ph_all[[4]],ph_all[[5]],
7                       pl_all[[1]],pl_all[[2]],pl_all[[3]],pl_all[[4]],pl_all[[5]])
8
9 #pe1 = list()                                     # _____ 2.
10 #pe2 = list()
11 #pe3 = list()
12 #pe4 = list()
13 #pe5 = list()
14 #pe1[[1]]<-ph_all[[1]]
15 #pe1[[2]]<-pl_all[[1]]
16 #pe2[[1]]<-ph_all[[2]]
17 #pe2[[2]]<-pl_all[[2]]
18 #pe3[[1]]<-ph_all[[3]]
19 #pe3[[2]]<-pl_all[[3]]
20 #pe4[[1]]<-ph_all[[4]]
21 #pe4[[2]]<-pl_all[[4]]
22 #pe5[[1]]<-ph_all[[5]]
23 #pe5[[2]]<-pl_all[[5]]
24 #PCAGeneSets_tmp <- list(unlist(pe1),unlist(pe2),unlist(pe3),unlist(pe4),unlist(pe5))
25
26 PCAGeneSets_entrez_tmp = list()
27 PCAGeneSets_entrez_tmp<-lapply(PCAGeneSets_tmp,function(x){
28   bitr(x,fromType="SYMBOL",toType="ENTREZID",OrgDb="org.Hs.eg.db")$ENTREZID})
29
30 for (i in 1:length(PCAGeneSets_entrez_tmp)){
31   print("_____IV. enrichPathway_____")
32   l6 <- enrichPathway(gene = as.character(unique(sort(PCAGeneSets_entrez_tmp[[i]]))),
33                       pvalueCutoff = 0.05,
34                       qvalueCutoff = 0.1,
35                       readable = TRUE)
36   if (length(l6$ID) == 0) {
37     ll[[i]] <- "NULL"
38   }
39   else{
40     ll[[i]] <- data.frame(
41       i,
42       l6$ID,
43       l6$Description,
44       l6$GeneRatio,
45       l6$BgRatio,
46       l6$pvalue,
47       l6$p.adjust,
48       l6$qvalue,
49       l6$geneID,
50       l6$Count
51     )
52     colnames(ll[[i]]) <- c("Community#", "ID", "Description", "GeneRatio",
53                           "BgRatio", "pvalue", "p.adjust", "qvalue",
54                           "geneID","Count")
55     options(scipen=999)
56     print(ll[[i]])
57     print("#####")
58   }
59 }
60

```

