

A primer on constructing plasticity phenotypes to classify experience-dependent development of the visual cortex

—
All R Markdowns Compiled
—

<u>Table of Contents</u>		
Markdown Number	Title	Page Number (of 108)
1	Cat&Human_Markdown	2 of 108
2	Cat_tSNE_plots_Figure7	52 of 108
3	Cat_CorrHeatmap_Figure8	63 of 108
4	Rat_Markdown	71 of 108
5	Rat_CorrHeatmap_Figure14	98 of 108

Cat&Human_Markdown

<u>Table of Contents</u>		
Section Number	Title	Page Number
1	Load Cat and Human Data	2
2	Cat Analysis	7
3	Human Analysis	24
4	Create Cat & Human Phenotypes	29
5	Create Cat Boxplots	32
6	Create Human Boxplots	36
7	Cat tSNE Analysis.	40

1.) Load Cat and Human Data

Cat & Human: Dimension Reduction Using PCA – Data Processing

Install the package, “PlasticityPhenotypes”.

```
devtools::install_github("visualneurosciencelab/PlasticityPhenotypes")
```

Load the package, “PlasticityPhenotype”.

```
library(PlasticityPhenotypes)
```

```
## Loading required package: tidyverse
```

```
## -- Attaching packages -----  
----- tidyverse 1.3.0 --
```

```
## v ggplot2 3.3.0      v purrr  0.3.3  
## v tibble  2.1.3      v dplyr  0.8.5  
## v tidyr   1.0.2      v stringr 1.4.0  
## v readr   1.3.1      v forcats 0.5.0
```

```
## -- Conflicts -----  
----- tidyverse_conflicts() --  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag()     masks stats::lag()
```

```
## Loading required package: FactoMineR
```

```
## Loading required package: factoextra
```

```
## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
```

```
## Loading required package: data.table
```

```
##
```

```
## Attaching package: 'data.table'
```

```
## The following objects are masked from 'package:dplyr':
```

```
##
```

```
##      between, first, last
```

```
## The following object is masked from 'package:purrr':
```

```
##
```

```
##      transpose
```

```
## Loading required package: ggpubr
```

```

37  ## Loading required package: magrittr
38  ##
39  ## Attaching package: 'magrittr'
40  ## The following object is masked from 'package:purrr':
41  ##
42  ##     set_names
43  ## The following object is masked from 'package:tidyr':
44  ##
45  ##     extract
46  ## Loading required package: corrplot
47  ## corrplot 0.84 loaded
48  ## Loading required package: psych
49  ##
50  ## Attaching package: 'psych'
51  ## The following objects are masked from 'package:ggplot2':
52  ##
53  ##     %+%, alpha
54  ## Loading required package: httr

```

55 Store the cat and human file paths in unique objects.

```

56 raw.cat.dev <- 'https://osf.io/45rjy//?action=download'
57 raw.cat.tsne <- 'https://osf.io/59yu6//?action=download'
58
59 raw.hum.imputed <- 'https://osf.io/6pbgr//?action=download'
60 raw.hum.nonimputed <- 'https://osf.io/azdv8//?action=download'

```

61 **Import the necessary cat data CSVs from OSF. The “raw.data” object**
62 **consists of 9 attribute columns (“id”, “Sample”, “Blot”, “Case”, “Age”,**
63 **“Condition”, “Location”, “Tissue.Piece”, “Run.Number”) and 7 protein**
64 **expression columns. The “tsne.raw.data” object consists of 8 attribute**
65 **columns (“Condition”, “Labels”, “V1”, “V2”, “Cluster.Number”,**
66 **“Reordered”, “Cluster.Name”, “Region”).**

```

67 filename <- 'cat_proteins.csv'
68 GET(raw.cat.dev, write_disk(filename, overwrite = TRUE))
69 ## Response [https://files.ca-1.osf.io/v1/resources/8a3kx/providers/osfstorag
70 e/5ed0684ec7568603012d0920?action=download&direct&version=2]
71 ## Date: 2020-06-09 21:11

```

```

72  ## Status: 200
73  ## Content-Type: text/csv
74  ## Size: 32.3 kB
75  ## <ON DISK> C:\Users\dezia\Dropbox (Kathy Murphy)\JB-ProteinAnalysisWorkflo
76  w\JB ProteinAnalysisWorkflow Markdowns\RMD files June 2020\Cat & Human Analys
77  is\cat_proteins.csv

78  raw.data <- read.csv(filename)
79  head(raw.data)

80  ##      id Sample Blot      Case Age Condition Location Tissue.Piece Run.Numb
81  er
82  ## 1 K252      5  A+C 2wk Normal   2          1          1          1          1
83  +2
84  ## 2 K252      6  A+C 2wk Normal   2          1          1          2          1
85  +2
86  ## 3 K242      3  A+C 3wk Normal   3          1          1          1          1
87  +2
88  ## 4 K242      4  A+C 3wk Normal   3          1          1          2          1
89  +2
90  ## 5 K244     15  B+D 4wk Normal   4          1          1          1          1
91  +2
92  ## 6 K244     16  B+D 4wk Normal   4          1          1          2          1
93  +2
94  ##      GluN1      GluN2A      GluN2B      GABAAa1      GABAAa3      GluA2      Syn
95  ## 1 0.3854409 0.4268017 0.7555231 0.8563928 1.526663 0.4211669 0.5734898
96  ## 2 0.3190519 0.4147973 0.6501530 0.6309027 1.361924 0.4979576 0.6161713
97  ## 3 1.1909136 1.3210631 1.7467064 1.0709774 1.477091 0.7386673 0.6625438
98  ## 4 1.2921448 1.2865798 1.6964425 1.3543666 1.851294 0.7892433 0.7016512
99  ## 5 0.8178326 1.0010536 1.0862350 0.8661382 1.504845 0.4188479 0.8361352
100 ## 6 0.8388763 0.6876145 0.8356130 1.9219633 2.037198 0.4474243 0.6970003

101 filename <- 'cat_tsne.csv'
102 GET(raw.cat.tsne, write_disk(filename, overwrite = TRUE))

103 ## Response [https://files.ca-1.osf.io/v1/resources/8a3kx/providers/osfstorag
104 e/5ed0684f17ac9e0316621448?action=download&direct&version=1]
105 ## Date: 2020-06-09 21:11
106 ## Status: 200
107 ## Content-Type: text/csv
108 ## Size: 13.9 kB
109 ## <ON DISK> C:\Users\dezia\Dropbox (Kathy Murphy)\JB-ProteinAnalysisWorkflo
110 w\JB ProteinAnalysisWorkflow Markdowns\RMD files June 2020\Cat & Human Analys
111 is\cat_tsne.csv

112 tsne.raw.data <- read.csv(filename)
113 head(tsne.raw.data)

114 ##      Condition      Labels      V1      V2 Cluster.Number Reordered
115 ## 1      Normal 5wk Normal CVF 1 -1.421038 1.926670          1          1
116 ## 2      Normal 5wk Normal CVF 2 -1.182249 3.150573          1          1

```

```

117 ## 3      Normal 5wk Normal MVF 1 -1.488770 1.005343      1      1
118 ## 4      Normal 5wk Normal MVF 2 -1.511189 1.509622      1      1
119 ## 5      Normal 5wk Normal PVF 1 -1.438462 3.133959      1      1
120 ## 6      Normal 5wk Normal PVF 2 -1.956975 2.357059      1      1
121 ##      Cluster.Name Region
122 ## 1      Normal 1      CVF
123 ## 2      Normal 1      CVF
124 ## 3      Normal 1      MVF
125 ## 4      Normal 1      MVF
126 ## 5      Normal 1      PVF
127 ## 6      Normal 1      PVF

```

128 **Import the necessary human data CSVs from OSF. The**
129 **“raw.data.imputed” contains 1 attribute column (“ids”) and 7 protein**
130 **columns. The “raw.data.nonimputed” contains 6 attribute columns**
131 **(“ids”, “Age”, “AgeBin”, “Sex”, “PMI”, “PMI.Interval”) and 7 protein**
132 **columns.**

```

133 filename <- 'human_imputed.csv'
134 GET(raw.hum.imputed, write_disk(filename, overwrite = TRUE))

135 ## Response [https://files.ca-1.osf.io/v1/resources/8a3kx/providers/osfstora
136 e/5ece9223c7568602612d38fe?action=download&direct&version=2]
137 ##   Date: 2020-06-09 21:11
138 ##   Status: 200
139 ##   Content-Type: text/csv
140 ##   Size: 1.64 kB
141 ## <ON DISK>  C:\Users\dezia\Dropbox (Kathy Murphy)\JB-ProteinAnalysisWorkflo
142 w\JB ProteinAnalysisWorkflow Markdowns\RMD files June 2020\Cat & Human Analys
143 is\human_imputed.csv

144 raw.data.imputed <- read.csv(filename)
145 head(raw.data.imputed)

146 ##      ids  Gaba1  Gaba3 Synapsin  GluA2  GluN1  NR2A  NR2B
147 ## 1  271 0.3248 1.1794   1.5397 0.7354 0.9382 2.1907 1.3247
148 ## 2 1283 0.6098 1.7641   0.5793 1.0750 1.0786 0.8077 2.5610
149 ## 3 1055 0.9530 1.3563   0.9279 1.1351 1.5934 1.3013 2.7864
150 ## 4 1296 0.5215 0.9289   0.4865 0.8936 1.4743 0.4494 0.7914
151 ## 5 1102 0.7715 1.4949   0.5107 0.8892 1.0861 1.8462 2.9910
152 ## 6  135 0.6382 1.7209   1.2124 1.2077 1.8118 1.3394 2.5131

153 filename <- 'human_nonimputed.csv'
154 GET(raw.hum.nonimputed, write_disk(filename, overwrite = TRUE))

155 ## Response [https://files.ca-1.osf.io/v1/resources/8a3kx/providers/osfstora
156 e/5ece9223aeeb6d025a085f2b?action=download&direct&version=2]
157 ##   Date: 2020-06-09 21:11

```

```

158 ## Status: 200
159 ## Content-Type: text/csv
160 ## Size: 2.34 kB
161 ## <ON DISK> C:\Users\dezia\Dropbox (Kathy Murphy)\JB-ProteinAnalysisWorkflo
162 w\JB ProteinAnalysisWorkflow Markdowns\RMD files June 2020\Cat & Human Analys
163 is\human_nonimputed.csv

164 raw.data.nonimputed <- read.csv(filename)
165 head(raw.data.nonimputed)

166 ## ids Age Gaba1 Gaba3 Synapsin GluA2 GluN1 NR2A NR2B AgeBin Se
167 x PMI
168 ## 1 271 0.05 0.3248 1.1794 NA 0.7354 0.9382 NA 1.3247 <0.3
169 M 9
170 ## 2 1157 0.05 NA NA 0.4560 NA NA NA NA <0.3
171 F 14
172 ## 3 1283 0.24 0.6098 1.7641 0.5793 1.0750 1.0786 0.8077 2.5610 <0.3
173 F 23
174 ## 4 1055 0.26 0.9530 1.3563 0.9279 1.1351 1.5934 1.3013 2.7864 <0.3
175 M 12
176 ## 5 1296 0.27 0.5215 0.9289 0.4865 0.8936 1.4743 0.4494 0.7914 <0.3
177 M 16
178 ## 6 1102 0.33 0.7715 1.4949 0.5107 0.8892 1.0861 1.8462 2.9910 <0.3
179 M 22
180 ## PMI.Interval
181 ## 1 <10
182 ## 2 10 to 15
183 ## 3 20+
184 ## 4 10 to 15
185 ## 5 15 to 20
186 ## 6 20+

```

2.) Cat Analysis

Cat: Dimension Reduction Using PCA – Additional Processing & Analysis

Rename columns in the “raw.data” object whose headers must contain special characters.

```
colnames(raw.data)[13:14] <- c('GABA\u1D00\u03b11', 'GABA\u1D00\u03b13')
```

```
head(raw.data)
```

##	id	Sample	Blot	Case	Age	Condition	Location	Tissue.Piece	Run.Numb
## 1	K252	5	A+C 2wk	Normal	2		1	1	1
## 2	K252	6	A+C 2wk	Normal	2		1	1	2
## 3	K242	3	A+C 3wk	Normal	3		1	1	1
## 4	K242	4	A+C 3wk	Normal	3		1	1	2
## 5	K244	15	B+D 4wk	Normal	4		1	1	1
## 6	K244	16	B+D 4wk	Normal	4		1	1	2
##									
## 1	0.3854409	0.4268017	0.7555231			0.8563928		1.526663	0.4211669
## 2	0.3190519	0.4147973	0.6501530			0.6309027		1.361924	0.4979576
## 3	1.1909136	1.3210631	1.7467064			1.0709774		1.477091	0.7386673
## 4	1.2921448	1.2865798	1.6964425			1.3543666		1.851294	0.7892433
## 5	0.8178326	1.0010536	1.0862350			0.8661382		1.504845	0.4188479
## 6	0.8388763	0.6876145	0.8356130			1.9219633		2.037198	0.4474243
##									
##	Syn								
## 1	0.5734898								
## 2	0.6161713								
## 3	0.6625438								
## 4	0.7016512								
## 5	0.8361352								
## 6	0.6970003								

Assign “raw.data” to the object “my.data”. While both contain the same data, the former will be used during the subcluster analysis while the latter will be used in PCA.

```
my.data <- raw.data
```

```
head(my.data)
```

```
##      id Sample Blot      Case Age Condition Location Tissue.Piece Run.Numb
er
## 1 K252      5  A+C 2wk Normal   2      1      1      1      1
+2
## 2 K252      6  A+C 2wk Normal   2      1      1      2      1
+2
## 3 K242      3  A+C 3wk Normal   3      1      1      1      1
+2
## 4 K242      4  A+C 3wk Normal   3      1      1      2      1
+2
## 5 K244     15  B+D 4wk Normal   4      1      1      1      1
+2
## 6 K244     16  B+D 4wk Normal   4      1      1      2      1
+2
##      GluN1      GluN2A      GluN2B GABA<U+1D00>a1 GABA<U+1D00>a3      GluA2
## 1 0.3854409 0.4268017 0.7555231      0.8563928      1.526663 0.4211669
## 2 0.3190519 0.4147973 0.6501530      0.6309027      1.361924 0.4979576
## 3 1.1909136 1.3210631 1.7467064      1.0709774      1.477091 0.7386673
## 4 1.2921448 1.2865798 1.6964425      1.3543666      1.851294 0.7892433
## 5 0.8178326 1.0010536 1.0862350      0.8661382      1.504845 0.4188479
## 6 0.8388763 0.6876145 0.8356130      1.9219633      2.037198 0.4474243
##      Syn
## 1 0.5734898
## 2 0.6161713
## 3 0.6625438
## 4 0.7016512
## 5 0.8361352
## 6 0.6970003
```

Rename the “1”, “2”, and “3” factor levels in “raw.data\$Location” with “CVF”, “PVF”, and “MVF”, respectively. Assign the output in the column “raw.data\$VFR”. This was done to more clearly identify where in the visual cortex an observation was taken, for the subcluster analysis.

```
raw.data$VFR <- plyr::mapvalues(raw.data$Location,
                                unique(raw.data$Location),
                                c("CVF", "PVF", "MVF"))
```

Combine the contents of “raw.data\$Case”, “raw.data\$VFR”, & “raw.data\$Tissue.Piece”, and assign them to the column “raw.data\$Labels”. This was done to create unique labels for each observation. These labels were included in the prior tSNE analysis and will be used to identify which subcluster an observation falls into.

```
raw.data$Labels <- paste(raw.data$Case,  
                        raw.data$VFR,  
                        raw.data$Tissue.Piece)
```

Centre (but do not scale) the protein columns of “my.data”, and assign them to “my.data.scaled”.

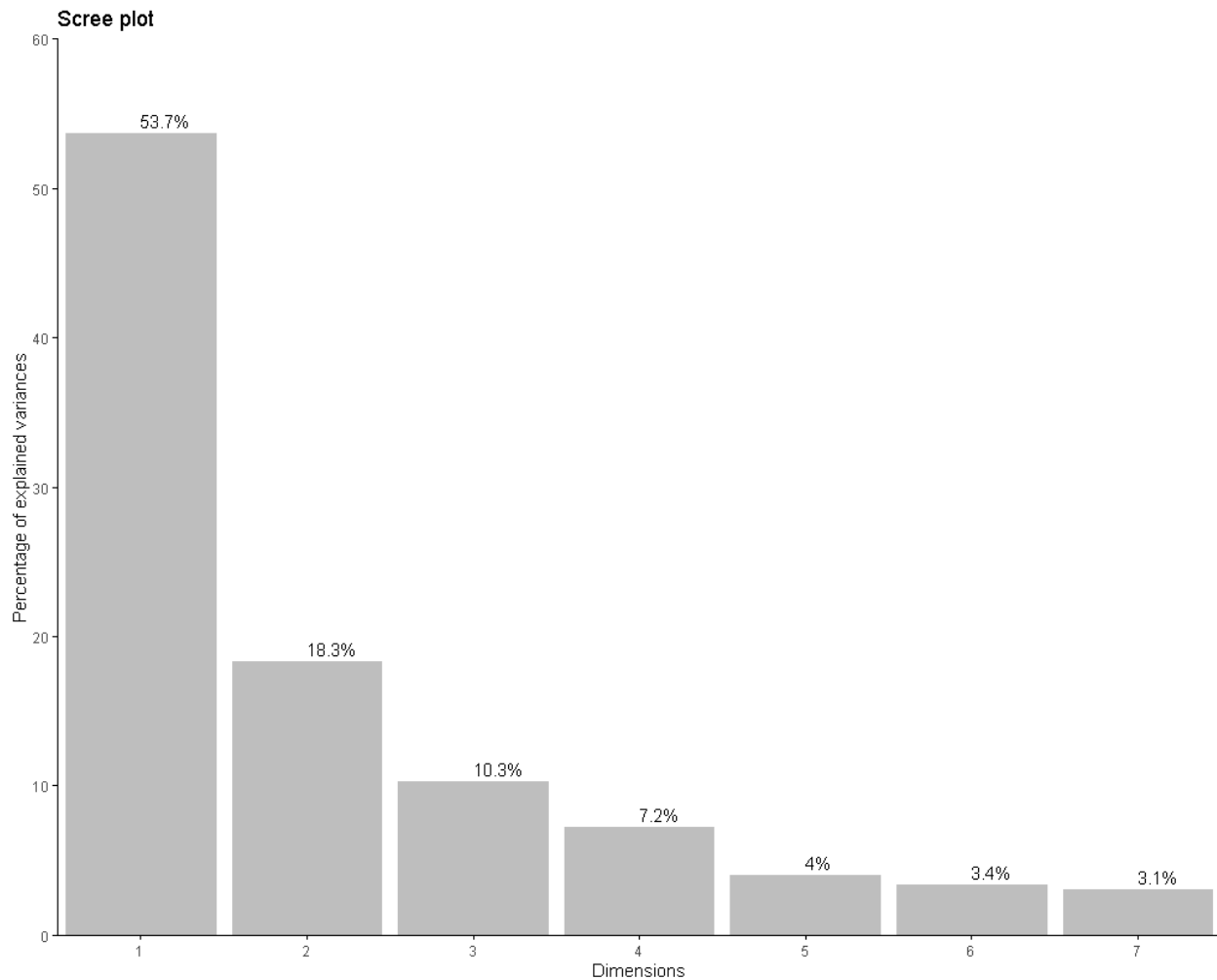
```
my.data.scaled <- scale(my.data[,10:16],  
                      center = TRUE,  
                      scale = F)
```

Perform a PCA on “my.data.scaled”.

```
pca.scaled <- PCA(my.data.scaled,  
                 ncp=ncol(my.data.scaled),  
                 scale.unit=FALSE,  
                 graph = FALSE)
```

Construct scree plot.

```
fviz_eig(pca.scaled,  
        addlabels = T,  
        ylim = c(0, 60),  
        xlim = c(0.5, 7.5),  
        ncp = 7, # Select number of principal components using 'ncp' parameter  
        barfill = "grey",  
        barcolor = "grey",  
        geom = "bar")+  
  scale_y_continuous(expand = c(0,0))+  
  scale_x_discrete(expand = c(0,0))+  
  theme(axis.line.y=element_line(),  
        axis.line.x=element_line(),  
        panel.grid=element_blank())
```



298

299 **Calculate how many components are required to maintain 80% of the**
 300 **total variance. Store the output of this function into object,**
 301 **"cum.var".**

```
302 pca.scaled$eig[,3]
303 ##    comp 1    comp 2    comp 3    comp 4    comp 5    comp 6    comp 7
304 ## 53.66586 72.00511 82.27410 89.51460 93.55403 96.92361 100.00000
305 cum.var <- cum_var(pca.eig.3 = pca.scaled$eig[,3], # "pca.scaled$eig[,3]" is
306                  # contained within object "pca"
307                  thresh = 80) # Custom threshold value
308
309 cum.var
310 ## [1] 3
```

311 Cat: Identifying Candidate Plasticity Features

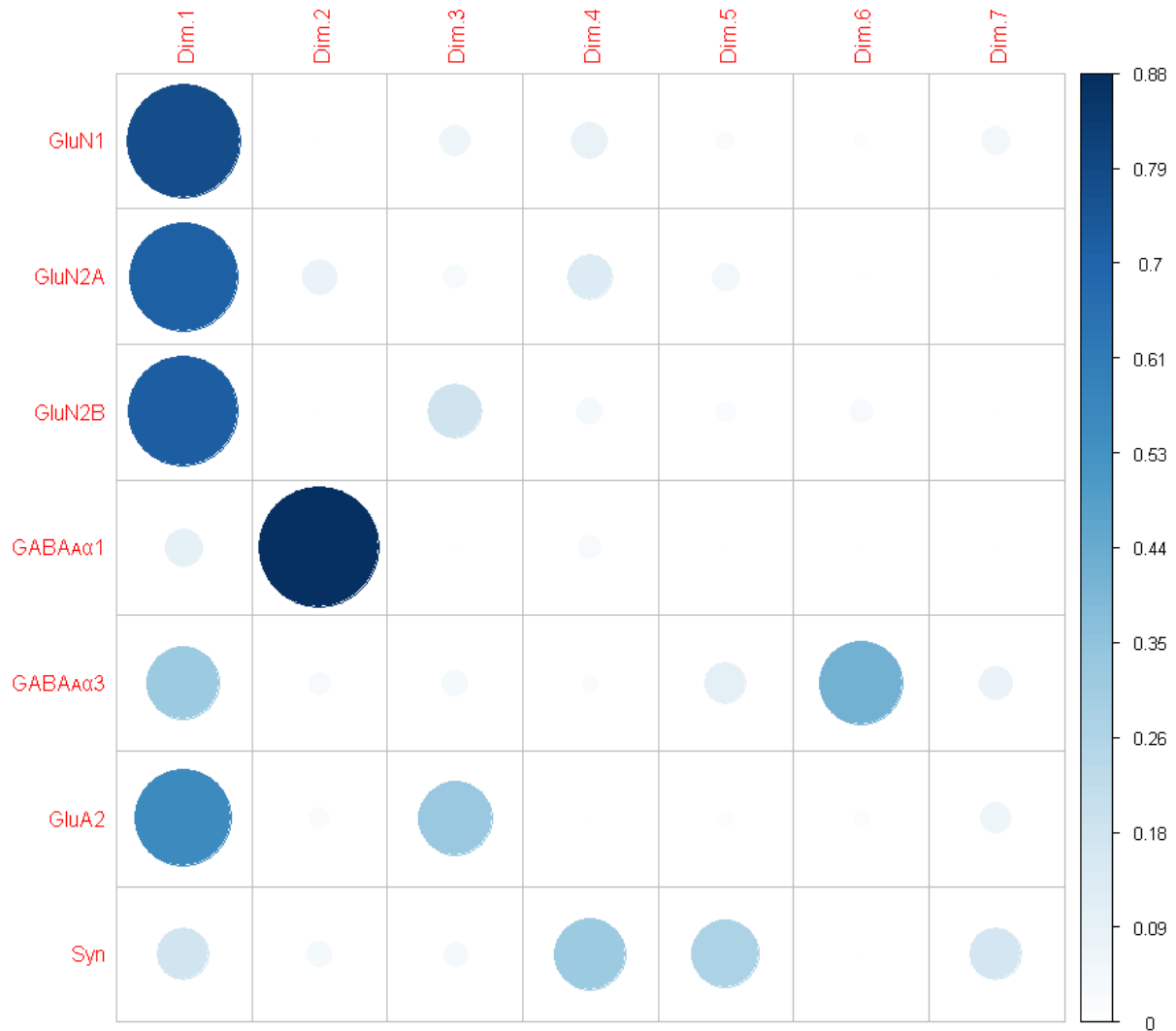
312 Construct plot of $\cos^2$ data stored in object, "pca.scaled"

```
313 head(pca.scaled$var$cos2)
```

	Dim.1	Dim.2	Dim.3	Dim.4	Dim.5
## GluN1	0.77881379	0.001235058	0.058361191	0.0806635837	0.020487897
## GluN2A	0.71044845	0.076807238	0.034590787	0.1288130896	0.044429485
## GluN2B	0.71924708	0.001258850	0.181659213	0.0409213505	0.024605760
## GABA<U+1D00>a1	0.08798193	0.875605314	0.002366629	0.0316236983	0.001141864
## GABA<U+1D00>a3	0.31884955	0.030507545	0.039562035	0.0177579594	0.102205636
## GluA2	0.55711246	0.025593577	0.328354478	0.0000216584	0.012722004

	Dim.6	Dim.7
## GluN1	0.011315701	0.0491227785
## GluN2A	0.002127357	0.0027835923
## GluN2B	0.032213898	0.0000938451
## GABA<U+1D00>a1	0.001089827	0.0001907419
## GABA<U+1D00>a3	0.419456917	0.0716603572
## GluA2	0.017388047	0.0588077725

```
328 corrplot(pca.scaled$var$cos2,is.corr=FALSE)
```



329

330 Construct plot of coord (correlation) data stored in object,
 331 "pca.scaled".

332 `head(pca.scaled$var$coord)`

```

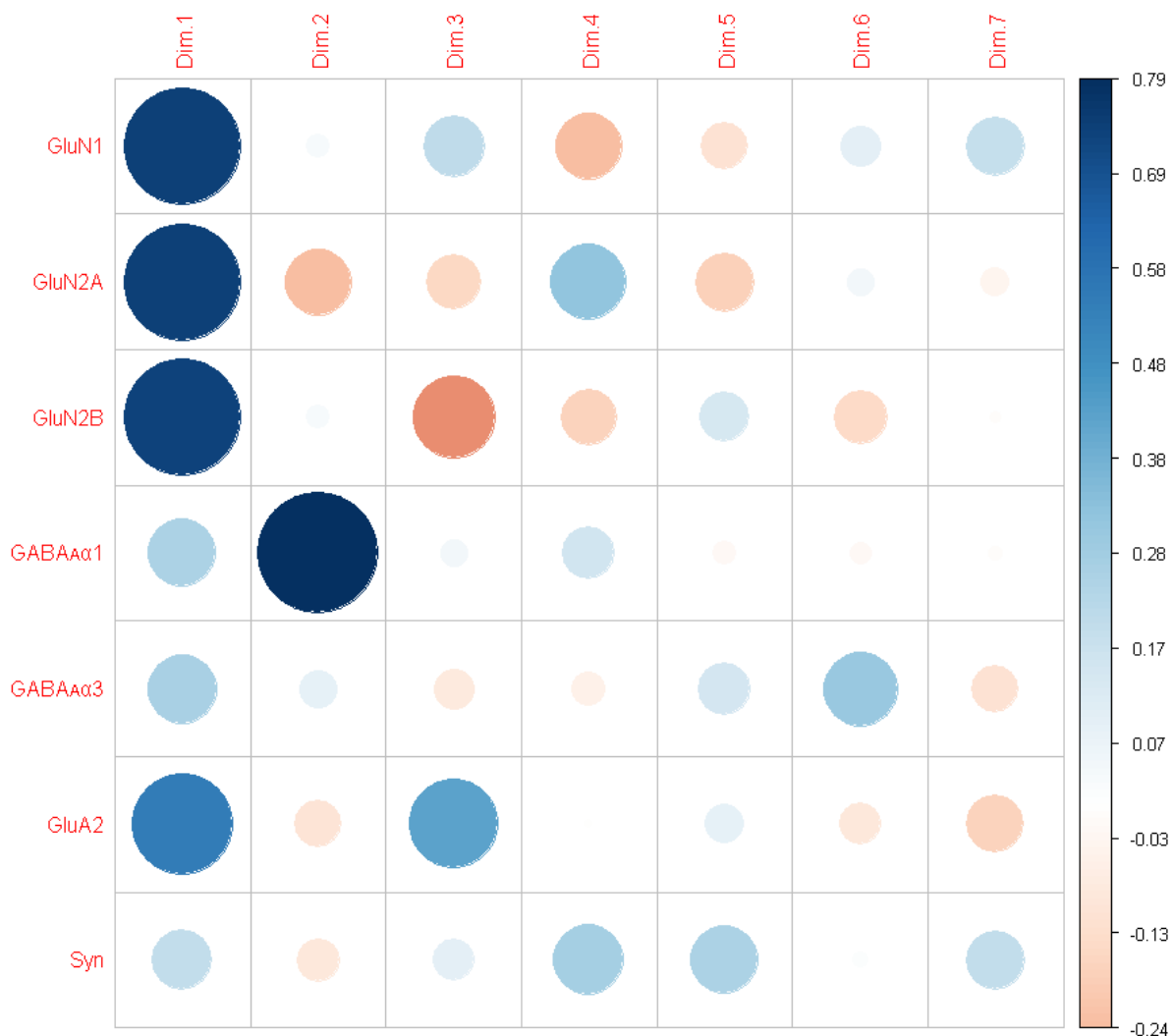
333 ##           Dim.1      Dim.2      Dim.3      Dim.4      Dim.5
334 ## GluN1      0.7372864  0.02936047  0.20182804 -0.23727839 -0.11958264
335 ## GluN2A     0.7357148 -0.24190470 -0.16233922  0.31327319 -0.18398350
336 ## GluN2B     0.7305037  0.03056119 -0.36712339 -0.17424420  0.13511443
337 ## GABA<U+1D00>a1 0.2504041  0.78994861  0.04106858  0.15012429 -0.02852673
338 ## GABA<U+1D00>a3 0.2577073  0.07971453 -0.09077645 -0.06081776  0.14590538
339 ## GluA2      0.5521184 -0.11833855  0.42386956 -0.00344250  0.08343313
340 ##           Dim.6      Dim.7
341 ## GluN1      0.08887102  0.185165851
342 ## GluN2A     0.04025904 -0.046051740
343 ## GluN2B     -0.15459840 -0.008344285

```

```

344 ## GABA<U+1D00>a1 -0.02786913 -0.011659171
345 ## GABA<U+1D00>a3  0.29558171 -0.122172473
346 ## GluA2           -0.09754077 -0.179381694
347 corrplot(pca.scaled$var$coord, is.corr=F)

```



348

349 **Create PCA amplitude plots for each important component. Use**
350 **amplitude plots to identify relevant plasticity features.**

```

351 pca.scaled$var$coord
352 ##          Dim.1      Dim.2      Dim.3      Dim.4      Dim.5
353 ## GluN1      0.7372864  0.02936047  0.20182804 -0.23727839 -0.11958264
354 ## GluN2A      0.7357148 -0.24190470 -0.16233922  0.31327319 -0.18398350
355 ## GluN2B      0.7305037  0.03056119 -0.36712339 -0.17424420  0.13511443
356 ## GABA<U+1D00>a1 0.2504041  0.78994861  0.04106858  0.15012429 -0.02852673
357 ## GABA<U+1D00>a3 0.2577073  0.07971453 -0.09077645 -0.06081776  0.14590538

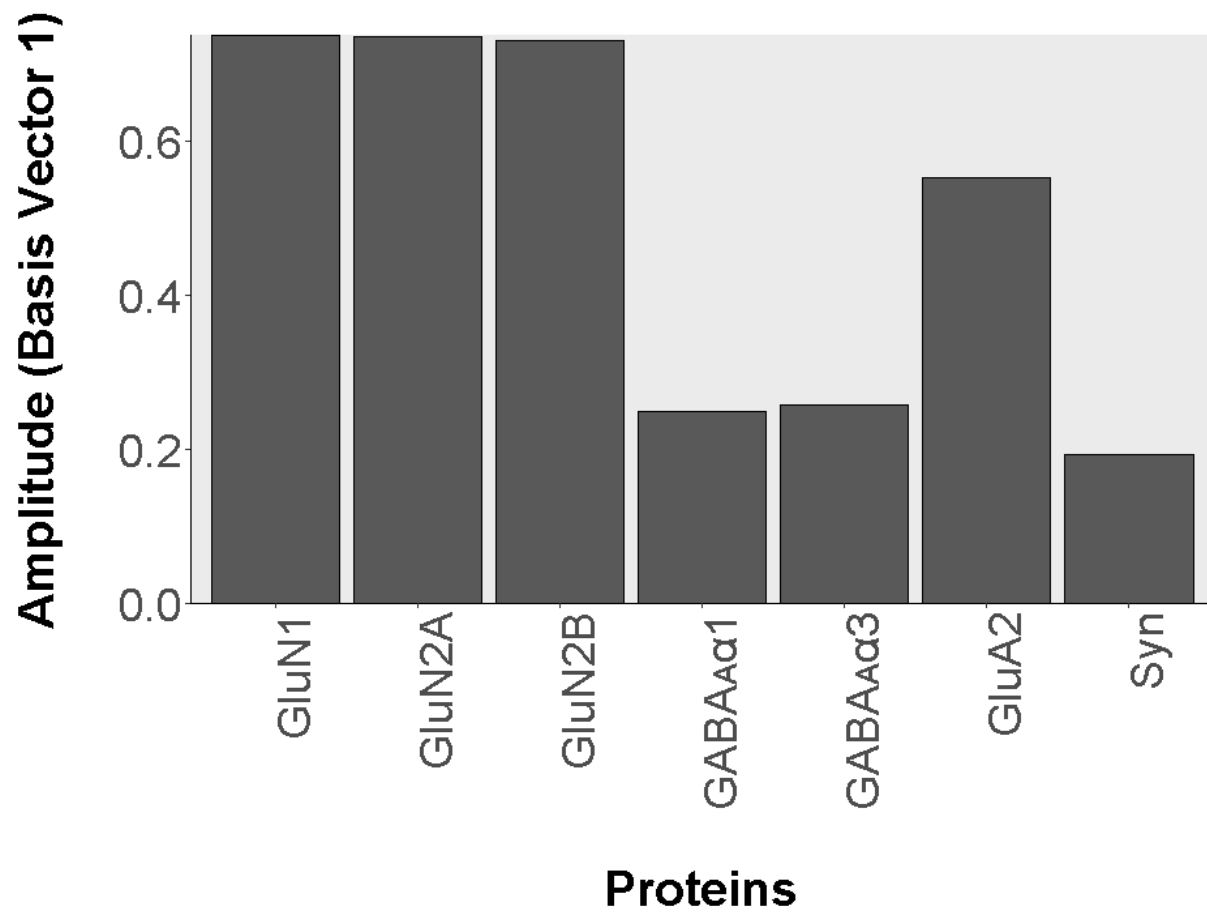
```

```

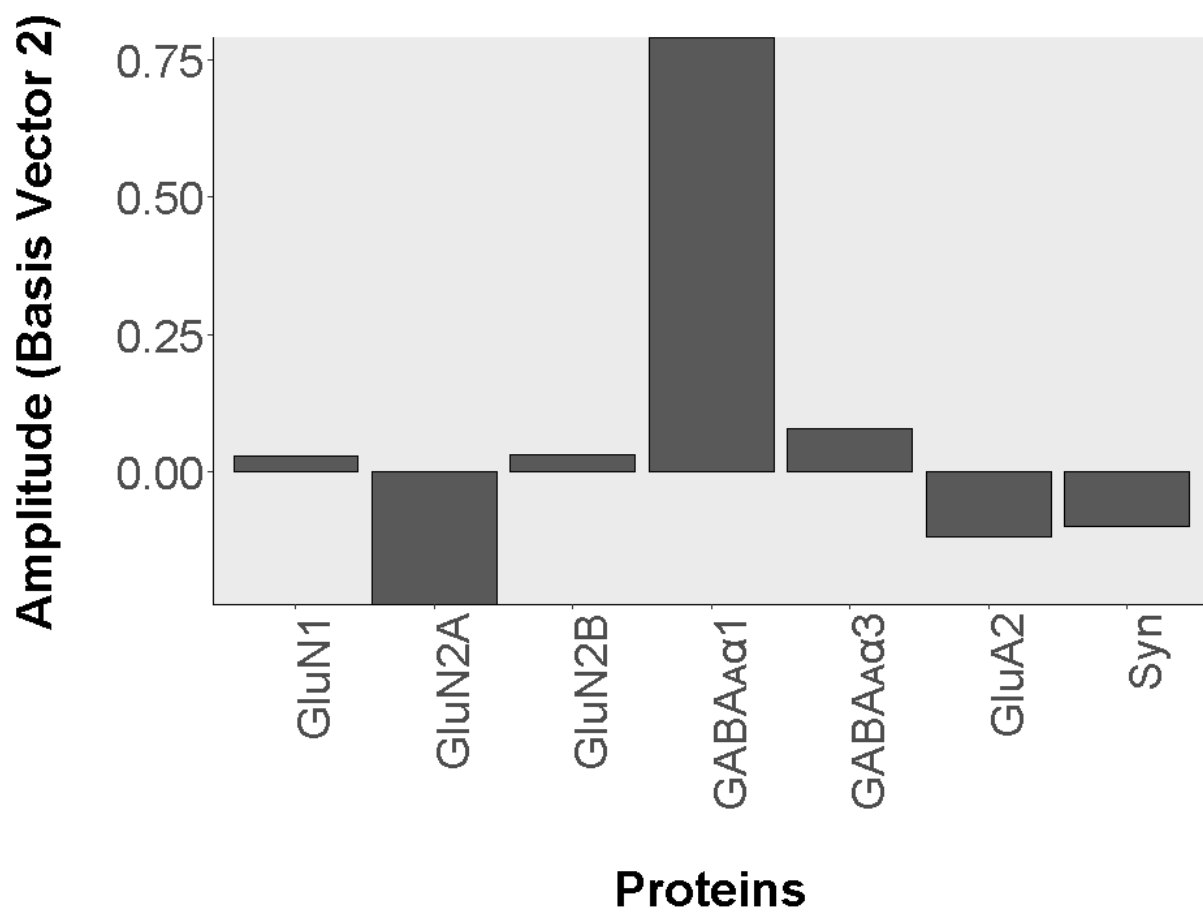
358 ## GluA2      0.5521184 -0.11833855  0.42386956 -0.00344250  0.08343313
359 ## Syn        0.1942073 -0.09789409  0.09216506  0.26636772  0.24871271
360 ##           Dim.6      Dim.7
361 ## GluN1      0.08887102  0.185165851
362 ## GluN2A     0.04025904 -0.046051740
363 ## GluN2B     -0.15459840 -0.008344285
364 ## GABA<U+1D00>a1 -0.02786913 -0.011659171
365 ## GABA<U+1D00>a3  0.29558171 -0.122172473
366 ## GluA2     -0.09754077 -0.179381694
367 ## Syn        0.01262123  0.190005138

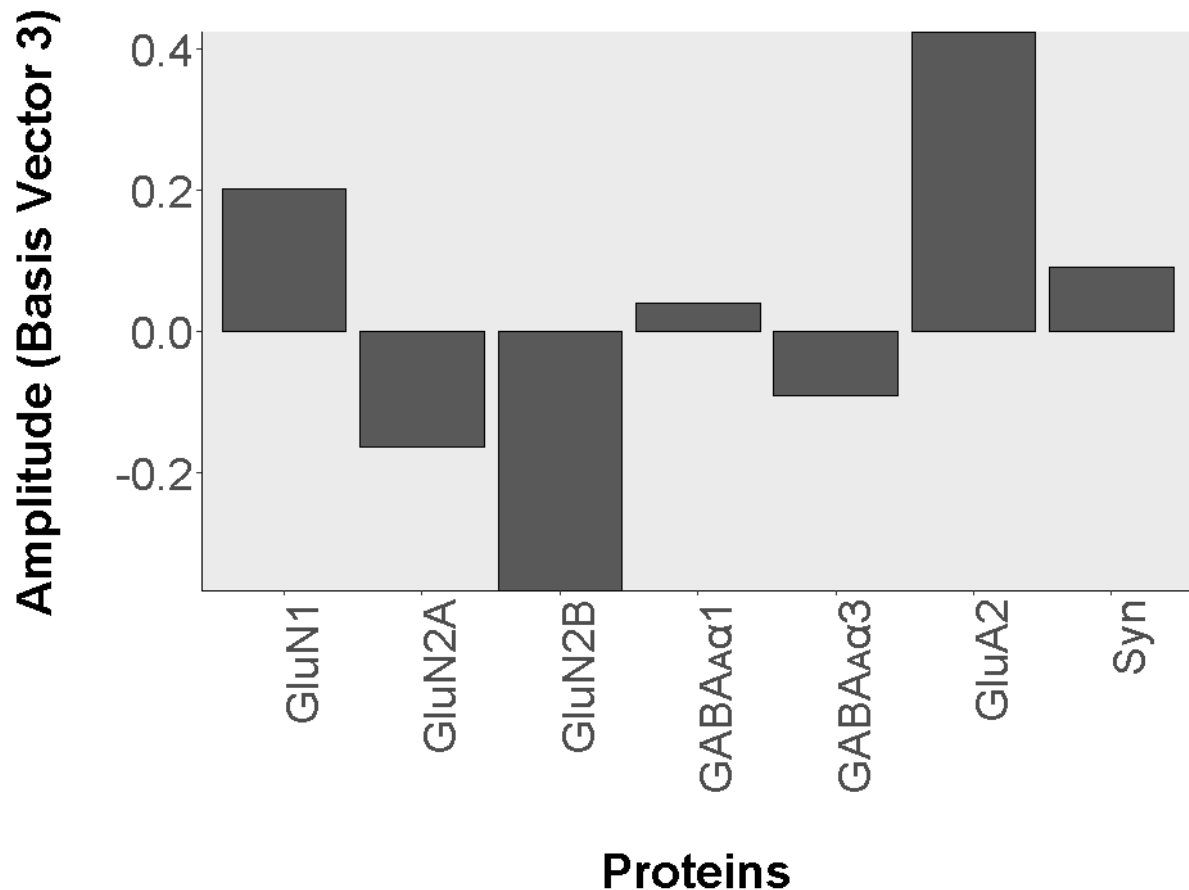
368 amplitude_plots(cum.var = cum.var, # Output of "cum_var()" function
369                  pca.var.coord = pca.scaled$var$coord) # "pca.scaled$var$coord
370 " is contained within object "pca"

```



371





373

374 **Cat: Application of the Heuristics to Identify Candidate Plasticity**
 375 **Features for Cat V1**

376 **Create a dataframe called “NewFeatures” to store the subsequently-**
 377 **calculated plasticity features.**

```
378 NewFeatures <- data.frame(matrix(nrow = nrow(my.data),
379                                ncol = 0))
```

380 **Calculate plasticity features.**

```
381 NewFeatures$'Protein Sum' <- rowSums(my.data[,c('GABA\u003b11',
382                                                'GABA\u003b13',
383                                                'Syn',
384                                                'GluA2',
385                                                'GluN1',
386                                                'GluN2A',
387                                                'GluN2B')])
```

388

```

389     prot_sums <- NewFeatures$'Protein Sum'
390
391 NewFeatures$'GluR Sum' <- rowSums(my.data[,c('GluA2',
392                                             'GluN1',
393                                             'GluN2A',
394                                             'GluN2B')])
395
396     glutr_sum <- NewFeatures$'GluR Sum'
397
398 NewFeatures$'GABA\u1D00R Sum' <- rowSums(my.data[,c('GABA\u1D00\u03b11',
399                                                     'GABA\u1D00\u03b13')])
400
401     gaba_sum <- NewFeatures$'GABA\u1D00R Sum'
402
403     ave.glut <- NewFeatures$'GluR Sum'/ncol(my.data[,c('GluA2',
404                                                         'GluN1',
405                                                         'GluN2A',
406                                                         'GluN2B')])
407
408     ave.gaba <- NewFeatures$'GABA\u1D00R Sum'/ncol(my.data[,c('GABA\u1D00\u03b11',
409                                                                 'GABA\u1D00\u03b13')])
410
411 NewFeatures$'GABA\u1D00R:GluR' <- (ave.glut - ave.gaba)/
412                                     (ave.glut + ave.gaba)
413
414 NewFeatures$'GABA\u1D00\u03b11:GluN2A' <- (my.data$GluN2A - my.data$'GABA\u1D00\u03b11')/
415                                             (my.data$'GABA\u1D00\u03b11' + my.d
416 ata$GluN2A)
417
418 NewFeatures$'GluN2B:GluN2A' <- (my.data$GluN2A - my.data$GluN2B)/
419                                     (my.data$GluN2B + my.data$GluN2A)
420
421 NewFeatures$'GABA\u1D00\u03b13:GABA\u1D00\u03b11' <- (my.data$'GABA\u1D00\u03b11' - my.data$'GABA\u1D00\u03b13')/
422                                             (my.data$'GABA\u1D00\u03b11' + my.data$'GABA\u1D00\u03b13')
423
424 NewFeatures$'GluN2B:GluA2' <- (my.data$GluA2 - my.data$GluN2B)/
425                                     (my.data$GluN2B + my.data$GluA2)
426
427 NewFeatures$'GluN2A:GluA2' <- (my.data$GluA2 - my.data$GluN2A)/
428                                     (my.data$GluN2A + my.data$GluA2)
429
430
431
432
433

```

434 **Bind the “raw.data\$Labels” column to the “NewFeatures” data frame,**
435 **and store the output in a new object, “NewFeatures.tsne”. The order**
436 **of observations in “NewFeatures” match that of “raw.data” object.**
437 **This was done so that each observation for the plasticity features had**
438 **a corresponding unique subcluster label.**

```
439 NewFeatures.tsne <- cbind(raw.data$Labels, NewFeatures)
```

440 Cat: Validating Candidate Plasticity Features

441 **Reassign “pca.scaled\$ind\$coord” object to “PCA.scores”.**

```
442 PCA.scores <- pca.scaled$ind$coord
```

443 **Perform a Bonferroni-corrected, pairwise Pearson’s correlation**
444 **against PCA scores and plasticity features. Store the output in the**
445 **object, “corr.scores.bf”.**

```
446 corr.scores.bf <- corr.test(PCA.scores[,1:cum.var],  
447                             NewFeatures,  
448                             use="pairwise",  
449                             method="pearson",  
450                             adjust="bonferroni")
```

```
451 ## Warning in abbreviate(colnames(r), minlength = minlength): abbreviate used  
452 with  
453 ## non-ASCII chars  
454  
455 ## Warning in abbreviate(colnames(r), minlength = minlength): abbreviate used  
456 with  
457 ## non-ASCII chars
```

458 **Store a matrix of correlation coefficients in the object,**
459 **“corr.scores.rval”.**

```
460 corr.scores.rval <- corr.scores.bf$r
```

```
461  
462 corr.scores.rval  
  
463 ##          Protein Sum    GlutR Sum    GABA<U+1D00>R Sum    GABA<U+1D00>R:GlutR  
464 ## Dim.1    0.98329175    0.99125848         0.47835172         0.5350370  
465 ## Dim.2    0.13405963   -0.10803230         0.81872766        -0.6241021  
466 ## Dim.3    0.03943816    0.03461785        -0.04679652         0.1217347  
467 ##          GABA<U+1D00>a1:GluN2A GluN2B:GluN2A    GABA<U+1D00>a3:GABA<U+1D00  
468 >a1
```

```

469 ## Dim.1          0.43709583    0.04388945          -0.1764722
470 ## Dim.2          -0.74316024   -0.42092044          0.5042847
471 ## Dim.3          -0.07014937    0.33804970          0.1940967
472 ##           GluN2B:GluA2 GluN2A:GluA2
473 ## Dim.1   -0.05781758   -0.1125934
474 ## Dim.2   -0.20885858    0.1716322
475 ## Dim.3    0.79782256    0.6431236

```

476 **Store a matrix of adjusted p-values for each correlation coefficient in**
477 **the object, “corr.scores.bfpval”.**

```

478 corr.scores.bfpval <- corr.scores.bf$p
479
480 corr.scores.bfpval

```

	Protein Sum	GlutR Sum	GABA<U+1D00>R Sum	GABA<U+1D00>R:GlutR
## Dim.1	1.523827e-205	2.847155e-244	6.278714e-16	1.242786e-20
## Dim.2	6.786802e-01	1.000000e+00	2.847804e-67	4.277009e-30
## Dim.3	1.000000e+00	1.000000e+00	1.000000e+00	1.000000e+00

```

486 ##           GABA<U+1D00>a1:GluN2A GluN2B:GluN2A      GABA<U+1D00>a3:GABA<U+1D00
487 >a1
488 ## Dim.1          5.148524e-13   1.000000e+00          8.369023e-02
489 ## Dim.2          7.994234e-49   5.632186e-12          5.681145e-18
490 ## Dim.3          1.000000e+00   1.871047e-07          3.023620e-02
491 ##           GluN2B:GluA2 GluN2A:GluA2
492 ## Dim.1   1.000000e+00  1.000000e+00
493 ## Dim.2   1.201851e-02  1.089777e-01
494 ## Dim.3   2.164192e-61  1.567638e-32

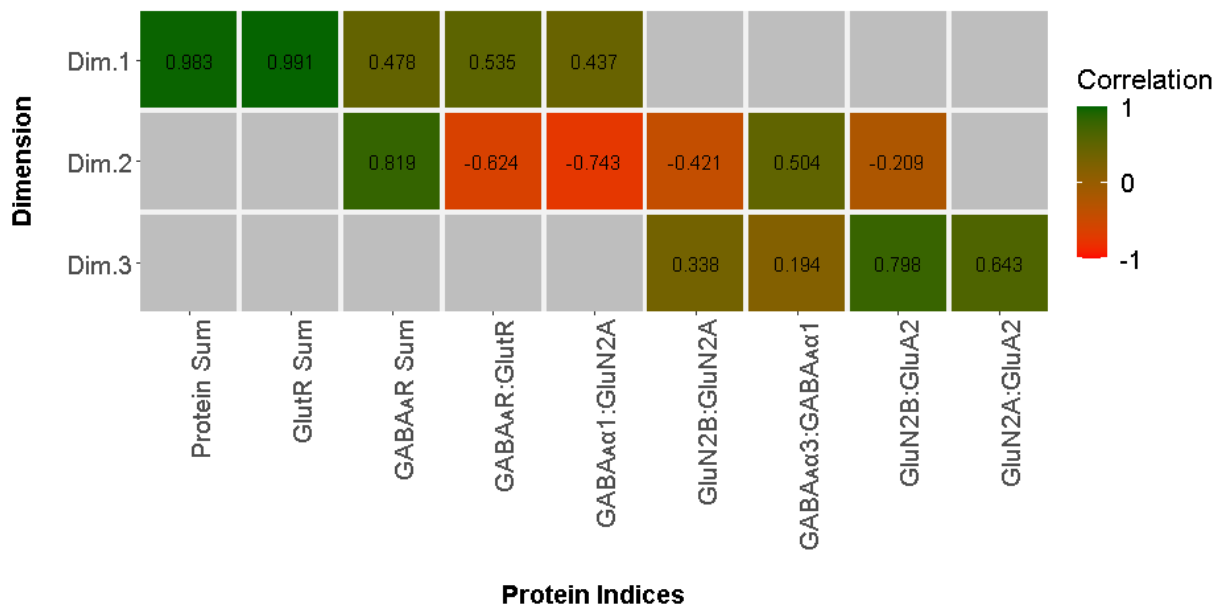
```

495 **Construct plasticity feature matrix.**

```

496 feature_matrix(
497   corr.scores.pval = corr.scores.bfpval, # Matrix of adjusted p-values
498   corr.scores.rval = corr.scores.rval, # Matrix of correlation coefficients
499   thresh = 0.05) # Significance threshold (acceptable values range from 0 - 1
500 )
501 ## Warning: Removed 12 rows containing missing values (geom_text).

```



Cat: Using Plasticity Features to Construct a Plasticity Phenotype – Data Processing

Bind “NewFeatures” data frame with the nine attribute columns of “my.data”. Assign this data frame to the object, “merged.data_all”. This was done because “NewFeatures” was calculated from “my.data”, so the order of observations are maintained between the two data frames. Furthermore, the attributes are necessary to subset the cat developmental data and thus create the phenotype and boxplots.

```
merged.data_all <- cbind(my.data[,1:9],NewFeatures)
```

Subset “merged.data_all” to display only data for the normal development condition (where it is coded as a 1 in the “merged.data_all\$Condition” column). Store the output in the object, “merged.data”.

```
merged.data <- subset(merged.data_all,
                      merged.data_all$Condition == 1)
```

519 **Store the character strings of the plasticity feature columns contained**
520 **within “merged.data”, to allow for easy selection of those columns in**
521 **subsequent analysis. Store these strings in the object, “NewFeatCol”.**

```
522 NewFeatCol <- colnames(merged.data)[10:ncol(merged.data)]
```

523 **Calculate the median of each plasticity feature in**
524 **“merged.features[,NewFeatsCol]” across all levels of**
525 **“list(merged.features\$Case)”.** Store the output in object, “cat.dev”.

```
526 cat.dev <- aggregate(merged.data[,NewFeatCol],  
527                      list(merged.data$Case),  
528                      median)
```

529 **Reorder the rows of “cat.dev” based on the specified order of the**
530 **“cat.dev\$Group.1” levels. Reassign this data to “cat.dev”.**

```
531 cat.dev <- cat.dev[match(c("2wk Normal",  
532                           "3wk Normal",  
533                           "4wk Normal",  
534                           "5wk Normal",  
535                           "6wk Normal",  
536                           "8wk Normal",  
537                           "12wk Normal",  
538                           "16wk Normal",  
539                           "24wk Normal"),  
540                        cat.dev$Group.1),]
```

541 **Rename the levels of “cat.dev\$Group.1” and reassign those names to**
542 **“cat.dev\$Group.1”. This is done to customize the appearance of the**
543 **age groups that appear on the cat developmental phenotype.**

```
544 cat.dev$Group.1 <- plyr::mapvalues(cat.dev$Group.1,  
545                                   unique(cat.dev$Group.1),  
546                                   c("2wk",  
547                                   "3wk",  
548                                   "4wk",  
549                                   "5wk",  
550                                   "6wk",  
551                                   "8wk",  
552                                   "12wk",  
553                                   "16wk",  
554                                   "Adult"))
```

```
555 Assign the the contents of the “Group.1” column as row names for the  
556 “cat.dev” data frame.  
557 rownames(cat.dev) <- cat.dev[, 'Group.1']
```

3.) Human: Analysis

Human: Translational Research Using Plasticity Phenotypes

Merge “raw.data.imputed” and “raw.data.nonimputed” by shared “ids” column. Store the output in the object, “raw.data.allproteins”.

```
raw.data.allproteins <- merge(raw.data.imputed,
                             raw.data.nonimputed[,c("ids",
                                                      "Age",
                                                      "AgeBin",
                                                      "Sex")],
                             by.x = "ids",
                             by.y = "ids")

head(raw.data.allproteins)
```

##	ids	Gaba1	Gaba3	Synapsin	GluA2	GluN1	NR2A	NR2B	Age	AgeBin	Sex
## 1	3	1.0395	1.5813	3.8900	1.8893	1.0565	5.1471	4.8389	5.39	5 to 11	M
## 2	103	1.3457	1.2796	1.7013	1.4039	1.1076	3.0127	1.9369	2.21	1 to 4	F
## 3	135	0.6382	1.7209	1.2124	1.2077	1.8118	1.3394	2.5131	0.33	<0.3	M
## 4	166	0.4611	0.8364	0.7692	1.2135	1.9642	0.3896	0.8482	0.37	0.3 to 1	F
## 5	271	0.3248	1.1794	1.5397	0.7354	0.9382	2.1907	1.3247	0.05	<0.3	M
## 6	451	0.8640	1.3357	0.9809	1.3880	0.9515	2.3311	3.0684	4.56	1 to 4	M

Reorder “raw.data.allproteins” so that all relevant attribute columns precede the protein columns. Store this reordered data frame in the object, “syn.prots”.

```
syn.prots <- raw.data.allproteins [,c("ids",
                                       "Age",
                                       "AgeBin",
                                       "Gaba1",
                                       "Gaba3",
                                       "Synapsin",
                                       "GluA2",
                                       "GluN1",
                                       "NR2A",
                                       "NR2B")]
```

Assign “syn.prots” to the new object, “my.data” but convert the “syn.prots\$ids” column to row names.

```
my.data <- column_to_rownames(syn.prots, var = "ids")
```

Rename protein data columns of “my.data” that contain special characters.

```
colnames(my.data) <- c('Age',
                        'AgeBin',
                        "GABA\u1D00\u03b11",
                        "GABA\u1D00\u03b13",
                        "Syn",
                        "GluA2",
                        "GluN1",
                        "GluN2A",
                        "GluN2B")
```

```
head(my.data)
```

```
##      Age  AgeBin GABA<U+1D00>a1 GABA<U+1D00>a3      Syn  GluA2  GluN1  GluN2
A
## 3   5.39   5 to 11           1.0395           1.5813 3.8900 1.8893 1.0565 5.147
1
## 103 2.21    1 to 4           1.3457           1.2796 1.7013 1.4039 1.1076 3.012
7
## 135 0.33     <0.3           0.6382           1.7209 1.2124 1.2077 1.8118 1.339
4
## 166 0.37 0.3 to 1           0.4611           0.8364 0.7692 1.2135 1.9642 0.389
6
## 271 0.05     <0.3           0.3248           1.1794 1.5397 0.7354 0.9382 2.190
7
## 451 4.56    1 to 4           0.8640           1.3357 0.9809 1.3880 0.9515 2.331
1
##      GluN2B
## 3   4.8389
## 103 1.9369
## 135 2.5131
## 166 0.8482
## 271 1.3247
## 451 3.0684
```

Human: Application of the Heuristics to Identify Candidate Plasticity Features

Create a data frame called “NewFeatures” to store the plasticity features.

```
NewFeatures <- data.frame(matrix(nrow = nrow(my.data),
                                  ncol = 0))
```

Calculate plasticity features determined in section 2 (Cat Analysis).

```
NewFeatures$'Protein Sum' <-
  rowSums(my.data[,c('GABA\u1D00\u03b11',
                     'GABA\u1D00\u03b13',
                     'Syn',
                     'GluA2',
                     'GluN1',
                     'GluN2A',
                     'GluN2B')]))

prot_sums <-
  NewFeatures$'Protein Sum'

NewFeatures$'GlutR Sum' <-
  rowSums(my.data[,c('GluA2',
                     'GluN1',
                     'GluN2A',
                     'GluN2B')]))

glutr_sum <- NewFeatures$'GlutR Sum'

NewFeatures$'GABA\u1D00R Sum' <-
  rowSums(my.data[,c('GABA\u1D00\u03b11',
                     'GABA\u1D00\u03b13')]))

gaba_sum <-
  NewFeatures$'GABA\u1D00R Sum'

ave.glut <-
  NewFeatures$'GlutR Sum' /
  ncol(my.data[,c('GluA2',
                  'GluN1',
                  'GluN2A',
                  'GluN2B')]))

ave.gaba <-
  NewFeatures$'GABA\u1D00R Sum' /
  ncol(my.data[,c('GABA\u1D00\u03b11',
                  'GABA\u1D00\u03b13')]))

NewFeatures$'GABA\u1D00R:GlutR' <-
  (ave.glut - ave.gaba) /
  (ave.glut + ave.gaba)

NewFeatures$'GABA\u1D00\u03b11:GluN2A' <-
  (my.data$GluN2A - my.data$'GABA\u1D00\u03b11') /
  (my.data$'GABA\u1D00\u03b11' + my.data$GluN2A)
```

```

682 NewFeatures$'GluN2B:GluN2A' <-
683   (my.data$GluN2A - my.data$GluN2B)/
684   (my.data$GluN2B + my.data$GluN2A)
685
686 NewFeatures$'GABA\u1D00\u03b13:GABA\u1D00\u03b111' <-
687   (my.data$'GABA\u1D00\u03b111' - my.data$'GABA\u1D00\u03b13')/
688   (my.data$'GABA\u1D00\u03b111' + my.data$'GABA\u1D00\u03b13')
689
690 NewFeatures$'GluN2B:GluA2' <-
691   (my.data$GluA2 - my.data$GluN2B)/
692   (my.data$GluN2B + my.data$GluA2)
693
694 NewFeatures$'GluN2A:GluA2' <-
695   (my.data$GluA2 - my.data$GluN2A)/
696   (my.data$GluN2A + my.data$GluA2)

```

697 Human: Using Plasticity Features to Construct a Plasticity 698 Phenotype – Data Processing

699 **Bind the “Age” and “AgeBin” columns of “my.data” with the**
700 **“NewFeatures” data frame. Store this new data frame into the object,**
701 **“merged.features”.**

```

702 merged.features <- cbind(data.frame(my.data[c('Age', 'AgeBin')]), NewFeatures
703 )

```

704 **Assign the column names for the plasticity features within**
705 **“merged.features” to the new object, “NewFeatsCol”. This will allow**
706 **for the subsetting of the plasticity feature data to be easier.**

```

707 NewFeatsCol <- colnames(merged.features)[3:ncol(merged.features)]

```

708 **Calculate the median of each plasticity feature in**
709 **“merged.features[,NewFeatsCol]” across all levels of**
710 **“list(merged.features\$AgeBin)”. Store content in the object,**
711 **“hum.dev”.**

```

712 hum.dev <- aggregate(merged.features[,NewFeatsCol],
713                      list(merged.features$AgeBin),
714                      median)

```

Assign and order the factor levels of the “hum.dev\$Group.1”. Arrange the factors in ascending order from “<0.3” to “55+”. This was done to specify the order of age-bins along the X-axis of the phenotype.

```
hum.dev$Group.1 <- factor(hum.dev$Group.1,  
  levels = c('<0.3',  
             '0.3 to 1',  
             '1 to 4',  
             '5 to 11',  
             '12 to 20',  
             '21 to 55',  
             '55+'),  
  ordered = TRUE)
```

Reorder “hum.dev” rows according to the previously assigned factor order of “hum.dev\$Group.1”.

```
hum.dev <- hum.dev[match(c("<0.3",  
  "0.3 to 1",  
  "1 to 4",  
  "5 to 11",  
  "12 to 20",  
  "21 to 55",  
  "55+"),  
  hum.dev$Group.1),]
```

Assign the the contents of the “Group.1” column as row names for the “hum.dev” data frame.

```
rownames(hum.dev) <- hum.dev[, 'Group.1']
```

4.) Create Cat & Human Phenotypes

Cat & Human: Using Plasticity Features to Construct a Plasticity Phenotype – Creating Phenotypes

Combine the plasticity feature columns in the human and cat data frames into a list, and store this list in the object, “df_list”.

```
df_list <- list(hum.dev[,2:ncol(hum.dev)],  
               cat.dev[,2:ncol(cat.dev)])
```

Construct a plasticity phenotype from the human and cat developmental data

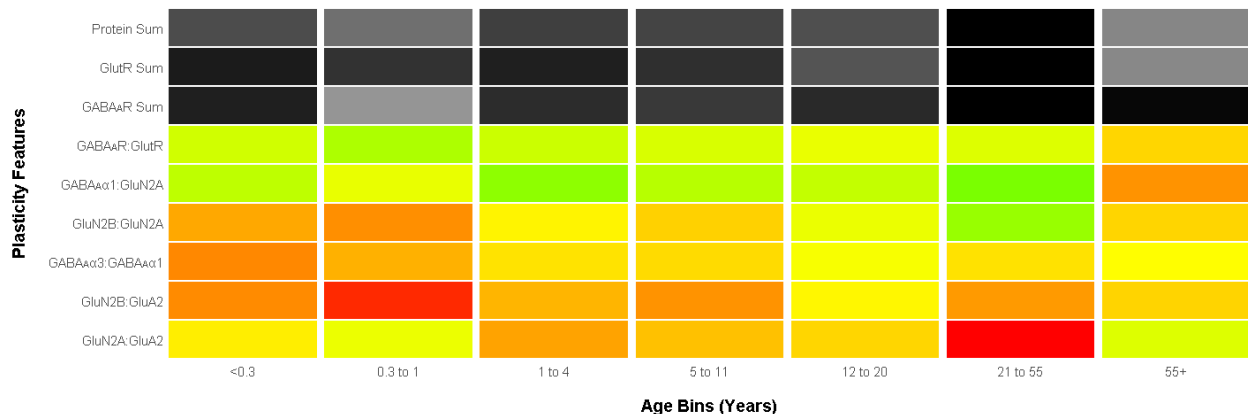
```
plasticity_phenotype(df_list = df_list,  
                     first_index_column = 4,  
                     group_label = c('\nAge Bins (Years)',  
                                     '\nAge (Weeks)'),  
                     translation = 'absolute')
```

	<0.3	0.3 to 1	1 to 4	5 to 11	12 to 20	21 to
##	55					
## Protein Sum	#4C4C4C	#6F6F6F	#3F3F3F	#444444	#4F4F4F	#0000
## GlutR Sum	#1B1B1B	#323232	#1F1F1F	#2F2F2F	#545454	#0000
## GABA<U+1D00>R Sum	#1F1F1F	#959595	#2C2C2C	#393939	#292929	#00
## GABA<U+1D00>R:GlutR	#D0FF00	#ADFF00	#CBFF00	#D8FF00	#E AFF00	#DD
## GABA<U+1D00>a1:GluN2A	#BEFF00	#E9FF00	#8EFF00	#B7FF00	#C3FF00	#7A
## GluN2B:GluN2A	#FFA800	#FF8F00	#FFF400	#FFD100	#EBFF00	#98FF
## GABA<U+1D00>a3:GABA<U+1D00>a1	#FF8800	#FFB100	#FFE300	#FFDB00	#F8FF00	#
## GluN2B:GluA2	#FF8B00	#FF2900	#FFB500	#FF9300	#FFF700	#FF9A
## GluN2A:GluA2	#FFEE00	#ECFF00	#FFA300	#FFC100	#FFD600	#FF00
##	55+					
## Protein Sum	#868686					
## GlutR Sum	#888888					
## GABA<U+1D00>R Sum	#070707					
## GABA<U+1D00>R:GlutR	#FFD600					
## GABA<U+1D00>a1:GluN2A	#FF9300					

```

780 ## GluN2B:GluN2A #FFD500
781 ## GABA<U+1D00>a3:GABA<U+1D00>a1 #FFFE00
782 ## GluN2B:GluA2 #FFD400
783 ## GluN2A:GluA2 #DDFF00
784 ## 2wk 3wk 4wk 5wk 6wk 8wk
785 ## Protein Sum #EDEDED #C0C0C0 #C7C7C7 #969696 #AEAFAE #000000
786 ## GlutR Sum #F6F6F6 #CECECE #DCDCDC #B0B0B0 #C3C3C3 #000000
787 ## GABA<U+1D00>R Sum #D1D1D1 #2A2A2A #1C1C1C #363636 #252525 #000000
788 0
789 ## GABA<U+1D00>R:GlutR #FF7800 #FFE200 #FFAB00 #EEFF00 #FFFC00 #9FFF0
790 0
791 ## GABA<U+1D00>a1:GluN2A #FF8100 #F9FF00 #FF8C00 #E3FF00 #FBFF00 #D4FF0
792 0
793 ## GluN2B:GluN2A #FFAC00 #FFC500 #FFE300 #FDFF00 #F7FF00 #FFFF00
794 ## GABA<U+1D00>a3:GABA<U+1D00>a1 #FFD100 #FFC700 #E6FF00 #FFF100 #E6FF00 #F0F
795 F00
796 ## GluN2B:GluA2 #FFBF00 #FF6F00 #FF7500 #FFE400 #FF9200 #FFDB00
797 ## GluN2A:GluA2 #EDFF00 #FFAA00 #FFAC00 #FFE400 #FF9100 #EBFF00
798 ## 12wk 16wk Adult
799 ## Protein Sum #414141 #D5D5D5 #D8D8D8
800 ## GlutR Sum #444444 #DADADA #E0E0E0
801 ## GABA<U+1D00>R Sum #747474 #CFCFCF #CBCBCB
802 ## GABA<U+1D00>R:GlutR #94FF00 #C4FF00 #F6FF00
803 ## GABA<U+1D00>a1:GluN2A #8FFF00 #C6FF00 #FFF400
804 ## GluN2B:GluN2A #DDFF00 #ADFF00 #9DFF00
805 ## GABA<U+1D00>a3:GABA<U+1D00>a1 #EDFF00 #64FF00 #64FF00
806 ## GluN2B:GluA2 #EEFF00 #E8FF00 #AEFF00
807 ## GluN2A:GluA2 #FFF000 #FFBA00 #FFE200
808 ## [[1]]

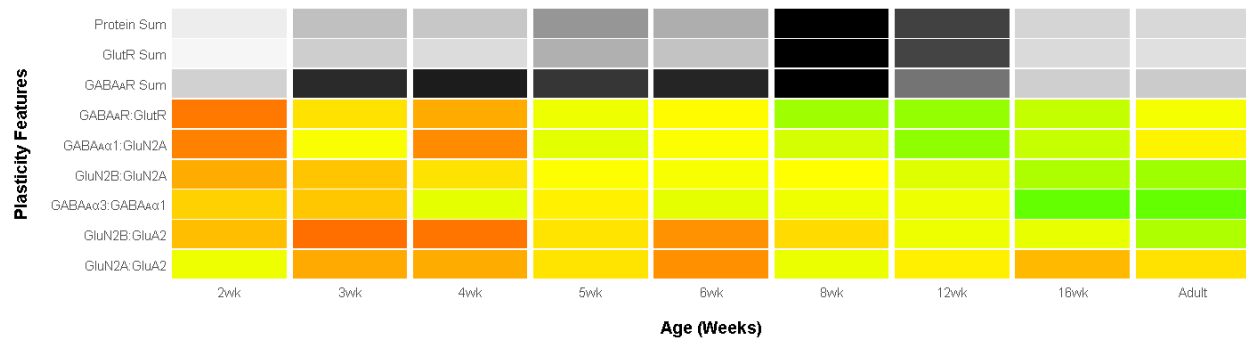
```



```

809
810 ##
811 ## [[2]]

```



812

813 **Store the human and cat colour-codes in their respective “hum.cols”**
 814 **and “cat.cols” objects.**

815 `hum.cols <- phen.cols[[1]]`

816 `cat.cols <- phen.cols[[2]]`

5.) Create Cat Boxplots

Cat: Using Plasticity Features to Construct a Plasticity Phenotype – Creating Boxplots

Assign a subset of columns – specifically, the “Case” column, and the column headers for each plasticity feature stored within “NewFeatCol”) – from “merged.data” that are necessary to create the cat developmental phenotype boxplots. Store these data in the object, “feats_df”.

```
feats_df <- as.data.frame(merged.data[,c("Case",NewFeatCol)])
```

Rename levels of 'feats_df\$Case' and reassign them to "feats_df\$Case". This is done to customize the appearance of the age groups that appear on the cat developmental boxplots.

```
feats_df$Case <- plyr::mapvalues(feats_df$Case,  
                                unique(feats_df$Case),  
                                c("2wk",  
                                  "3wk",  
                                  "4wk",  
                                  "5wk",  
                                  "6wk",  
                                  "8wk",  
                                  "12wk",  
                                  "16wk",  
                                  "24wk"))
```

Assign an order to the levels of “feats_df\$Case”, and then reassign to “feats_df\$Case”. The age groups will appear in ascending order, from “2wk” to “24wk”.

```
feats_df$Case <- factor(feats_df$Case,  
                        levels = c("2wk",  
                                    "3wk",  
                                    "4wk",  
                                    "5wk",  
                                    "6wk",  
                                    "8wk",  
                                    "12wk",  
                                    "16wk",  
                                    "24wk"))
```

```

852         "24wk"),
853         ordered = TRUE)

```

854 **Divide all data points in a “feats_df” sum by the median value of that**
855 **corresponding sum for the “2wk” age group in the object, “cat.dev”.**
856 **Store these normalized data into own “sum” object.**

```

857 sum1 <- feats_df[, 'Protein Sum']/
858         cat.dev[cat.dev$Group.1 == "2wk",
859                 'Protein Sum']
860
861 sum2 <- feats_df[, 'GlutR Sum']/
862         cat.dev[cat.dev$Group.1 == "2wk",
863                 'GlutR Sum']
864
865 sum3 <- feats_df[, 'GABA\u1D00R Sum']/
866         cat.dev[cat.dev$Group.1 == "2wk",
867                 'GABA\u1D00R Sum']

```

868 **Reassign the objects “sum1”, “sum2”, & “sum3” to their corresponding**
869 **sums column in “feats_df”.**

```

870 feats_df[, 'Protein Sum'] <- sum1
871 feats_df[, 'GlutR Sum'] <- sum2
872 feats_df[, 'GABA\u1D00R Sum'] <- sum3

```

873 **Assign “feats_df” to “feats_df2”. This is done to ensure the contents**
874 **of “feats_df” aren’t mistakenly overwritten.**

```

875 feats_df2 <- feats_df

```

876 **Append each column name for the sums in “feats_df” with “\n**
877 **(Normalized to 2wk)”. Store the output in the corresponding columns**
878 **of “feats_df2”.**

```

879 colnames(feats_df2)[2:4] <- paste0(colnames(feats_df)[2:4],
880                                     "\n (Normalized to 2wk)")

```

881 **Append each column name for the indices in “feats_df” with “\n”.**
882 **Store the output in the corresponding columns of “feats_df2”.**

```

883 colnames(feats_df2)[5:ncol(feats_df2)] <- paste0(colnames(feats_df)[5:ncol(feats_df2)],
884                                                     "\n")
885

```

886 **Create individual boxplots colour-coded according to the cat**
 887 **developmental phenotype.**

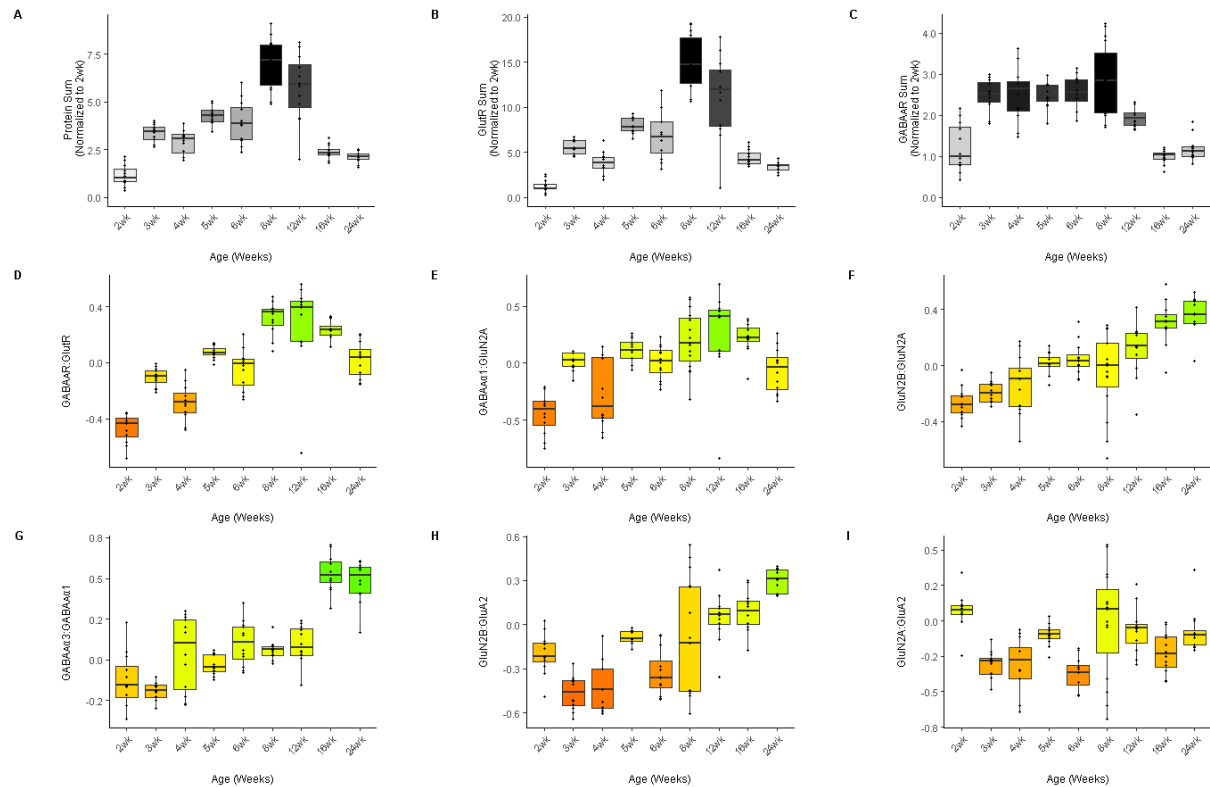
```
888 head(feats_df2)

889 ## Case Protein Sum\n (Normalized to 2wk) GlutR Sum\n (Normalized to 2wk)
890 ## 1 2wk 2.146473 2.519942
891 ## 2 2wk 1.949199 2.384410
892 ## 3 3wk 3.562481 6.331554
893 ## 4 3wk 3.893973 6.416518
894 ## 5 4wk 2.834671 4.211410
895 ## 6 4wk 3.240314 3.559622
896 ## GABA<U+1D00>R Sum\n (Normalized to 2wk) GABA<U+1D00>R:GlutR\n
897 ## 1 2.175087 -0.411126707
898 ## 2 1.818913 -0.358526315
899 ## 3 2.325699 -0.009787201
900 ## 4 2.925903 -0.117370427
901 ## 5 2.164067 -0.175800645
902 ## 6 3.613645 -0.476217884
903 ## GABA<U+1D00>a1:GluN2A\n GluN2B:GluN2A\n GABA<U+1D00>a3:GABA<U+1D00>a1\n
904 ## 1 -0.33478252 -0.27802972 -0.28126504
905 ## 2 -0.20666095 -0.22100155 -0.36682639
906 ## 3 0.10454911 -0.13874683 -0.15938105
907 ## 4 -0.02566762 -0.13739844 -0.15501567
908 ## 5 0.07225582 -0.04080959 -0.26938471
909 ## 6 -0.47300707 -0.09716108 -0.02910578
910 ## GluN2B:GluA2\n GluN2A:GluA2\n
911 ## 1 -0.2841498 -0.00664508
912 ## 2 -0.1325616 0.09110914
913 ## 3 -0.4055886 -0.28275343
914 ## 4 -0.3649694 -0.23958523
915 ## 5 -0.4434222 -0.41003249
916 ## 6 -0.3025545 -0.21161415

917 phenotype_boxplots(feature_df = feats_df2 , # Boxplot data frame
918 phenotype_cols = cat.cols, # Phenotype colour-code data frame
919
920 first_index_column = 4, # Index number of first index column in "feats_df2" (indexes begin at 0)
921
922 group_label = "\nAge (Weeks)", # X-axis label
923 point_size = 0.6, # Desired size of geom_jitter points
924 point_alpha = 1, # Desired transparency of geom_jitter points
925
926 aspect_ratio = 5/7, # Desired aspect ratio of each boxplot
927 text_size = 8) # Desired X- & Y-axis text size for each boxplot
928 xplot
```

Create a 3 x 3 matrix that displays all of the colour-coded boxplots on a single figure.

```
ggarrange(plotlist = plot_list,
  nrow = 3,
  ncol = 3,
  labels = LETTERS[1:9],
  font.label = list(size = 10),
  vjust = 1)
```



6.) Create Human Boxplots

Human: Using Plasticity Features to Construct a Plasticity Phenotype – Creating Boxplots

Assign all but the first column of “merged.features” (merged.features\$Age”) to “feats_df”.

```
feats_df <- merged.features[, -1]
```

Assign an order to the levels of “feats_df\$AgeBin”, and then reassign to “feats_df\$AgeBin”. Arrange the factors in ascending order from “<0.3” to “55+”. This was done to specify the order of the age-bins along the X-axis of the boxplots.

```
feats_df$AgeBin <- factor(feats_df$AgeBin,  
                          levels = c('<0.3',  
                                     '0.3 to 1',  
                                     '1 to 4',  
                                     '5 to 11',  
                                     '12 to 20',  
                                     '21 to 55',  
                                     '55+'),  
                          ordered = TRUE)
```

Assign “feats_df” to “feats_df2”.

```
feats_df2 <- feats_df
```

Divide all data points in a “feats_df” sum by the median value of that corresponding sum for the “<0.3” age-bin in the “hum.dev” object. Store these normalized data in their own “sum” objects.

```
sum1 <- feats_df[, 'Protein Sum']/  
  unlist(hum.dev[hum.dev$Group.1 == '<0.3',  
               'Protein Sum'])  
  
sum2 <- feats_df[, 'GlutR Sum']/  
  unlist(hum.dev[hum.dev$Group.1 == '<0.3',  
               'GlutR Sum'])  
  
sum3 <- feats_df[, 'GABA\u1D00R Sum']/
```

```

971 unlist(hum.dev[hum.dev$Group.1 == '<0.3',
972          'GABA\u1D00R Sum'])

```

Reassign objects “sum1”, “sum2”, & “sum3” to their respective sums column in “feats_df2”.

```

975 feats_df2[, 'Protein Sum'] <- sum1
976 feats_df2[, 'GlutR Sum'] <- sum2
977 feats_df2[, 'GABA\u1D00R Sum'] <- sum3

```

Assign “feats_df2” to “feats_df3”.

```

979 feats_df3 <- feats_df2

```

Append each column name for the sums in “feats_df2” with “\n (Normalized to <0.3)”. Store the output in the corresponding columns of “feats_df3”.

```

983 colnames(feats_df3)[2:4] <- paste0(colnames(feats_df2)[2:4],
984                                     "\n (Normalized to <0.3)")

```

Append each column name for the indices in “feats_df2” with “\n”. Store the output in the corresponding columns of “feats_df3”.

```

987 colnames(feats_df3)[5:ncol(feats_df3)] <- paste0(colnames(feats_df2)[5:ncol(f
988 eats_df3)], "\n")

```

Create individual boxplots that are colour-coded according to the human developmental phenotype.

```

991 head(feats_df3)

```

```

992 ##      AgeBin Protein Sum\n (Normalized to <0.3)
993 ## 3      5 to 11                2.1525040
994 ## 103     1 to 4                1.3050246
995 ## 135     <0.3                 1.1562073
996 ## 166    0.3 to 1              0.7176489
997 ## 271     <0.3                 0.9114702
998 ## 451     1 to 4                1.2089166
999 ##      GlutR Sum\n (Normalized to <0.3) GABA<U+1D00>R Sum\n (Normalized to <0
1000 .3)
1001 ## 3                        2.0967993                1.1455
1002 ## 296                      1.2097642                1.1474
1003 ## 103                      1.1142459                1.0311
1004 ## 966
1005 ## 135

```

```

1006 428
1007 ## 166 0.7159419 0.5671
1008 263
1009 ## 271 0.8413594 0.6574
1010 732
1011 ## 451 1.2548238 0.9614
1012 704
1013 ## GABA<U+1D00>R:GluR\n GABA<U+1D00>a1:GluN2A\n GluN2B:GluN2A\n
1014 ## 3 0.4231569 0.66395112 0.03086321
1015 ## 103 0.1738949 0.38247981 0.21735090
1016 ## 135 0.1858294 0.35457120 -0.30465931
1017 ## 166 0.2596819 -0.08404843 -0.37049604
1018 ## 271 0.2660112 0.74176108 0.24634465
1019 ## 451 0.2751269 0.45917186 -0.13654968
1020 ## GABA<U+1D00>a3:GABA<U+1D00>a1\n GluN2B:GluA2\n GluN2A:GluA2\n
1021 ## 3 -0.20673077 -0.4383936 -0.46299244
1022 ## 103 0.02517807 -0.1595426 -0.36426210
1023 ## 135 -0.45894621 -0.3508385 -0.05170586
1024 ## 166 -0.28924855 0.1771839 0.51394174
1025 ## 271 -0.56814253 -0.2860541 -0.49735142
1026 ## 451 -0.21443833 -0.3770757 -0.25358286

1027 phenotype_boxplots(feature_df = feats_df3, # Boxplot data frame
1028 phenotype_cols = hum.cols, # Phenotype colour-code data fr
1029 ame
1030 first_index_column = 4, # Index number of first index colu
1031 mn in "feats_df4" (indexes begin at 0)
1032 group_label = "\nAge Bins (Years)", # X-axis label
1033 point_size = 0.7, # Desired size of geom_jitter points
1034 point_alpha = 1, # Desired transparency of geom_jitter poi
1035 nts
1036 aspect_ratio = 5/7) # Desired aspect ratio of each boxplot

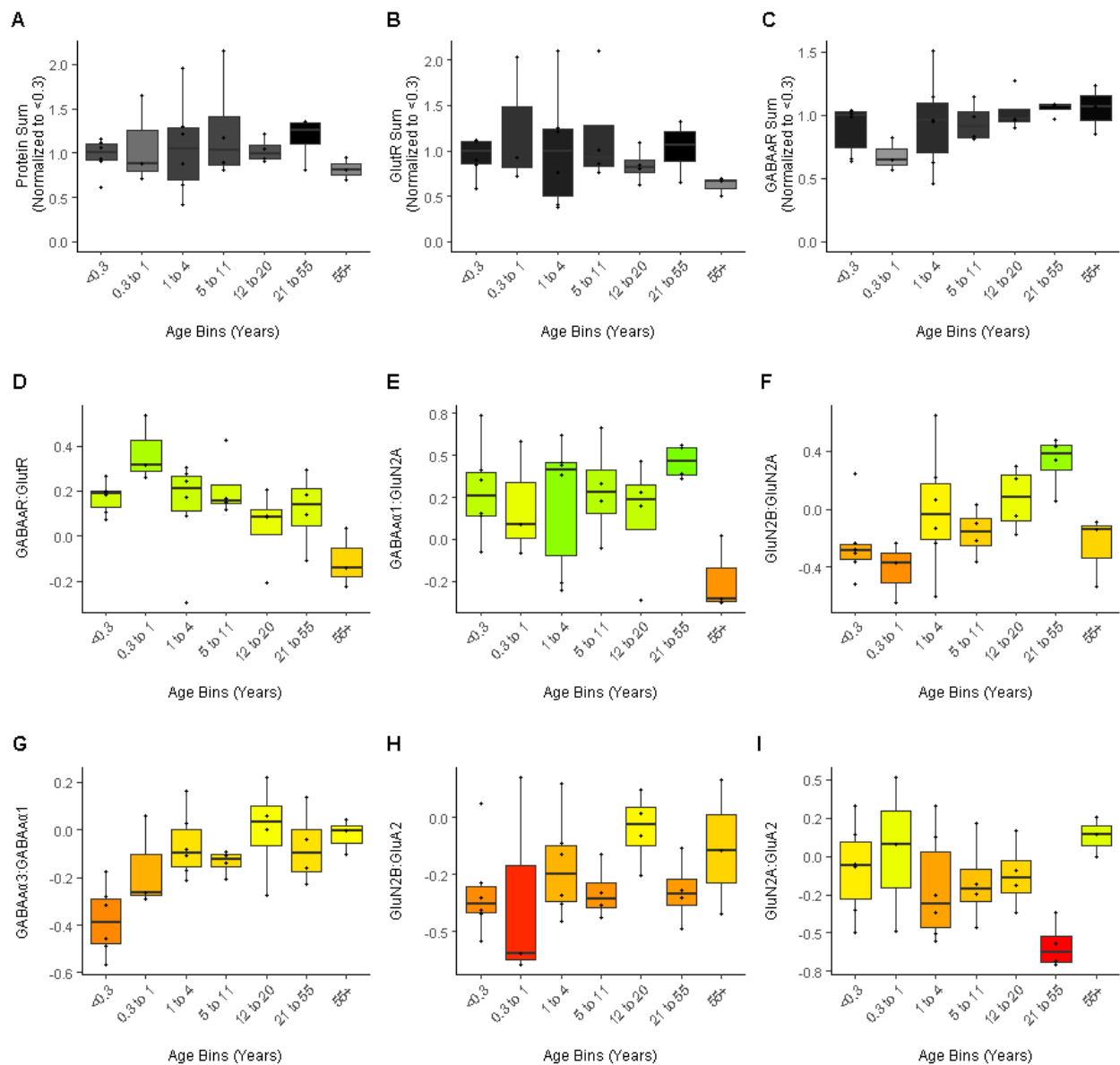
```

1037 **Create a 3 x 3 matrix displaying all colour-coded boxplots on a single figure.**

```

1039 ggarrange(plotlist = plot_list,
1040 nrow = 3,
1041 ncol = 3,
1042 labels = LETTERS[1:9],
1043 font.label = list(size = 12),
1044 vjust = 1)

```



7.) Cat tSNE Analysis

Cat: Using Plasticity Phenotypes for Clustering of Experience-Dependent Changes in V1 – Data Processing

Rename the first column of “NewFeature.tsne” to “Labels”.

```
colnames(NewFeatures.tsne)[1] <- 'Labels'
```

Merge “NewFeatures.tsne” and “tsne.raw.data[,c(‘Labels’,‘Cluster.Name’)]” by their mutual “Labels” column, and store the output in “tsne.processed.data”. This was done to map the unique subcluster labels onto the corresponding observation, so that the subcluster each observation fell into (stored in “tsne.raw.data\$Cluster.Name”) could be determined.

```
tsne.processed.data <- merge(NewFeatures.tsne,
                             tsne.raw.data[,c('Labels',
                                                'Cluster.Name')],
                             by.x = 'Labels',
                             by.y = 'Labels')
```

Assign the “tsne.processed.data” data frame to the object, “tsne.processed.data2”, except the “Labels” column in “tsne.processed.data” which becomes the row names of “tsne.processed.data2”.

```
tsne.processed.data2 <- column_to_rownames(tsne.processed.data, "Labels")
```

Reorder “tsne.processed.data2” columns and store the output in the object, “tsne.processed.data3”. The first column is which subcluster each observation falls into (“Cluster.Name”), followed by the sums and indices calculated for each observation.

```
tsne.processed.data3 <- tsne.processed.data2[,c(colnames(tsne.processed.data2)
)[10],
                                                colnames(tsne.processed.data2)
)[1:9]]
```

1076 **Count how many instances of each subcluster level are within the**
1077 **“tsne.processed.data3” and store the output in the object,**
1078 **“enough.counts”.**

```
1079 enough.counts <- plyr::count(tsne.processed.data3$Cluster.Name)
```

1080 **Assign the labels of select subclusters (determined according to Balsor**
1081 **et al. 2019) to the object, “clustered.groups”.**

```
1082 clustered.groups <- c("Normal 1",  
1083                       "LT BV 1",  
1084                       "MD 1",  
1085                       "LT BV 5",  
1086                       "LT BV 4",  
1087                       "RO 2",  
1088                       "ST BV 3",  
1089                       "MD 3",  
1090                       "ST BV 5",  
1091                       "ST BV 1",  
1092                       "LT BV 6",  
1093                       "BD 3",  
1094                       "BD 6")
```

1095 **Subset ‘enough.counts’ to retain subcluster labels in the**
1096 **“enough.counts\$x” column that are also contained in the object,**
1097 **“clustered.groups”.**

```
1098 enough.counts <- subset(enough.counts,  
1099                        enough.counts$x %in% clustered.groups)
```

1100 **Rename the select subcluster labels stored within**
1101 **“tsne.processed.data3\$Cluster.Name” to include the visual cortex**
1102 **location of each sample, and store the output in the new column,**
1103 **“tsne.processed.data3\$Cluster.Name2”.**

```
1104 tsne.processed.data3$Cluster.Name2 <- plyr::mapvalues(tsne.processed.data3$Cl  
1105 uster.Name,  
1106                                                    unique(tsne.processed.data3$Cluster  
1107 .Name),  
1108                                                    c( 'RO 2\n C,P,M' ,  
1109 'LT BV 1\n C,P,M' ,  
1110 'LT BV 4\n P,M' ,  
1111 'LT BV 5\n P,M' ,  
1112 'ST BV 3\n C,P,M' ,  
1113 'ST BV 1\n C,P,M' ,
```

```

1114 'ST BV 5\n P' ,
1115 'ST BV 6' ,
1116 'ST BV 4' ,
1117 'LT BV 6\n P' ,
1118 'BD 3\n C,P,M' ,
1119 'BD 5' ,
1120 'BD 6\n P' ,
1121 'LT BV 3' ,
1122 'MD 3\n C,P' ,
1123 'MD 1\n P,M' ,
1124 'MD 4' ,
1125 'Normal 1\n C,P,M'))

```

1126 **Assign a factor order to the levels of**
 1127 **“tsne.processed.data3\$Cluster.Name2”. This ordering was done**
 1128 **according to (Balsor et al., 2019) and was necessary to properly order**
 1129 **the subcluster levels when displayed on the phenotype.**

```

1130 tsne.processed.data3$Cluster.Name2 <- factor(
1131     tsne.processed.data3$Cluster.Name2,
1132     levels = c('Normal 1\n C,P,M',
1133                'LT BV 1\n C,P,M',
1134                'MD 1\n P,M',
1135                'LT BV 5\n P,M',
1136                'LT BV 4\n P,M',
1137                'RO 2\n C,P,M',
1138                'ST BV 3\n C,P,M',
1139                'MD 3\n C,P',
1140                'ST BV 5\n P',
1141                'ST BV 1\n C,P,M',
1142                'LT BV 6\n P',
1143                'BD 3\n C,P,M',
1144                'BD 6\n P'),
1145     ordered = TRUE)

```

1146 **Assign a factor order to the levels of**
 1147 **“tsne.processed.data3\$Cluster.Name” columns. This ordering was**
 1148 **done according to (Balsor et al., 2019) and was necessary to properly**
 1149 **order the subcluster levels when displayed on the phenotype**
 1150 **boxplots.**

```

1151 tsne.processed.data3$Cluster.Name <- factor(
1152     tsne.processed.data3$Cluster.Name,
1153     levels = c('Normal 1',
1154                'LT BV 1',

```

```

1155         'MD 1',
1156         'LT BV 5',
1157         'LT BV 4',
1158         'RO 2',
1159         'ST BV 3',
1160         'MD 3',
1161         'ST BV 5',
1162         'ST BV 1',
1163         'LT BV 6',
1164         'BD 3',
1165         'BD 6'),
1166         ordered = TRUE)

```

1167 **Subset the “tsne.processed.data3” data frame based on items in**
 1168 **“tsne.processed.data3\$Cluster.Name” that are also present in the**
 1169 **“enough.counts\$x” column. Store the output in the object,**
 1170 **“tsne.processed.data4”.**

```

1171 tsne.processed.4 <- subset(tsne.processed.data3,
1172                           tsne.processed.data3$Cluster.Name %in% enough.coun
1173 ts$x)

```

1174 **Calculate the median values of each plasticity feature across all levels**
 1175 **of “tsne.processed.data4\$Cluster.Name2”. Store the output in the**
 1176 **object, “cat.subcl_0.5”.**

```

1177 cat.subcl_0.5 <- aggregate(tsne.processed.4[,NewFeatCol],
1178                           list(tsne.processed.4$Cluster.Name2),
1179                           median)

```

1180 **Store “cat.subcl_0.5” in “cat.subcl”. This was done to avoid**
 1181 **potentially overwriting the contents of “cat.subcl_0.5”.**

```

1182 cat.subcl <- cat.subcl_0.5

```

1183 **Reverse the signs of**
 1184 **“cat.subcl_0.5\$'GABA\u1D00\u03b13:GABA\u1D00\u03b11” and**
 1185 **store the output in “cat.subcl\$'GABAD003b13:GABAD003b11”.**

```

1186 cat.subcl$'GABA\u1D00\u03b13:GABA\u1D00\u03b11' <- cat.subcl_0.5$'GABA\u1D00\u
1187 u03b13:GABA\u1D00\u03b11'*-1

```

Rename the 8th column of the “cat.subcl”.

```
colnames(cat.subcl)[8] <- 'GABA\u1D00\u03b1:GABA\u1D00\u03b13'
```

Cat: Constructing and Visualizing the Plasticity Phenotypes for the Subclusters – Create Phenotype

Assign the contents of the “Group.1” column as row names for the “cat.subcl” data frame.

```
rownames(cat.subcl) <- cat.subcl$Group.1
```

Create a plasticity phenotype using the “cat.subcl” data.

```
head(cat.subcl)
```

```
##                               Group.1 Protein Sum GlutR Sum GABA<U+1D00>R Sum
## Normal 1\n C,P,M Normal 1\n C,P,M    10.054874  6.240234          2.664731
## LT BV 1\n C,P,M   LT BV 1\n C,P,M     8.983626  5.654809          2.008049
## MD 1\n P,M          MD 1\n P,M      9.672284  6.299899          2.644214
## LT BV 5\n P,M      LT BV 5\n P,M     6.088109  3.781250          1.228423
## LT BV 4\n P,M      LT BV 4\n P,M     8.208065  5.690774          1.635767
## RO 2\n C,P,M       RO 2\n C,P,M     8.391937  4.463141          1.982746
##                               GABA<U+1D00>R:GluR GABA<U+1D00>a1:GluN2A GluN2B:GluN2A
## Normal 1\n C,P,M              0.06713220              0.11370947  0.009268495
## LT BV 1\n C,P,M              0.17430524              0.01721096 -0.164778667
## MD 1\n P,M                  0.08728357              -0.02049818 -0.308250943
## LT BV 5\n P,M              0.20686317              0.27958360  0.023554669
## LT BV 4\n P,M              0.27555796              0.14812194 -0.222851584
## RO 2\n C,P,M              0.08007062              0.04127925  0.286154842
##                               GABA<U+1D00>a1:GABA<U+1D00>a3 GluN2B:GluA2 GluN2A:GluA2
## Normal 1\n C,P,M              0.04220642      -0.09389056 -0.08669236
## LT BV 1\n C,P,M             -0.20718232      -0.25534466 -0.14229348
## MD 1\n P,M                 -0.17115154      -0.32646982 -0.02025748
## LT BV 5\n P,M              0.14884904      -0.14511606 -0.05281835
## LT BV 4\n P,M             -0.08800405      -0.36991967 -0.19746244
## RO 2\n C,P,M             -0.07850049       0.42592757  0.15801685
```

```
plasticity_phenotype(df_list = list(cat.subcl[, -1]), # Median values data frame
```

```
                        first_index_column = 4, # Index number of first index column in "cat.subcl" (indexes begin at 0)
```

```
                        group_label = "\nSubclusters", # X-axis labels
```

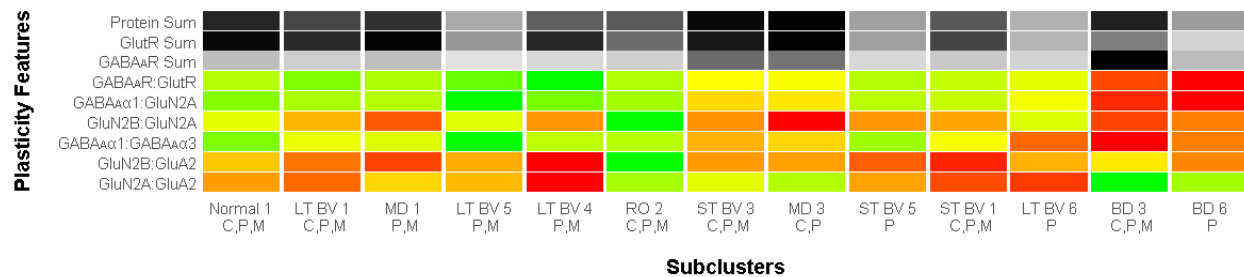
```
                        translation = 'local') # Desired colour-scale
```

```
##                               Normal 1\n C,P,M LT BV 1\n C,P,M MD 1\n P,M
## Protein Sum                    #252525          #464646          #303030
## GlutR Sum                      #090909          #292929          #040404
## GABA<U+1D00>R Sum              #BEBEBE          #CECECE          #BFBFBF
```

```

1228 ## GABA<U+1D00>R:GluR          #B5FF00          #7DFF00          #ACFF00
1229 ## GABA<U+1D00>a1:GluN2A          #86FF00          #AAFF00          #B7FF00
1230 ## GluN2B:GluN2A                  #E8FF00          #FFB800          #FF5900
1231 ## GABA<U+1D00>a1:GABA<U+1D00>a3  #7DFF00          #ECFF00          #DEFF00
1232 ## GluN2B:GluA2                    #FFC700          #FF7500          #FF4400
1233 ## GluN2A:GluA2                    #FF9E00          #FF6900          #FFD500
1234 ##                                LT BV 5\n P,M LT BV 4\n P,M RO 2\n C,P,M
1235 ## Protein Sum                     #AAAAAA          #5F5F5F          #595959
1236 ## GlutR Sum                       #969696          #272727          #6C6C6C
1237 ## GABA<U+1D00>R Sum               #E1E1E1          #D7D7D7          #CFCFCF
1238 ## GABA<U+1D00>R:GluR              #66FF00          #00FF00          #AFFF00
1239 ## GABA<U+1D00>a1:GluN2A          #00FF00          #77FF00          #A2FF00
1240 ## GluN2B:GluN2A                  #E1FF00          #FF9500          #00FF00
1241 ## GABA<U+1D00>a1:GABA<U+1D00>a3  #00FF00          #BDFF00          #B8FF00
1242 ## GluN2B:GluA2                    #FFAE00          #FF0000          #00FF00
1243 ## GluN2A:GluA2                    #FFBB00          #FF0000          #A5FF00
1244 ##                                ST BV 3\n C,P,M MD 3\n C,P ST BV 5\n P
1245 ## Protein Sum                     #090909          #000000          #A0A0A0
1246 ## GlutR Sum                       #181818          #000000          #9F9F9F
1247 ## GABA<U+1D00>R Sum               #6B6B6B          #747474          #D8D8D8
1248 ## GABA<U+1D00>R:GluR              #FDFF00          #FAFF00          #B4FF00
1249 ## GABA<U+1D00>a1:GluN2A          #FFD900          #FFE700          #B9FF00
1250 ## GluN2B:GluN2A                  #FF9400          #FF0000          #FF9500
1251 ## GABA<U+1D00>a1:GABA<U+1D00>a3  #FFB300          #FFD600          #9DFF00
1252 ## GluN2B:GluA2                    #FF9B00          #FF9F00          #FF5D00
1253 ## GluN2A:GluA2                    #E3FF00          #B1FF00          #FFA200
1254 ##                                ST BV 1\n C,P,M LT BV 6\n P BD 3\n C,P,M BD 6\n
1255 P
1256 ## Protein Sum                     #585858          #B1B1B1          #202020  #9C9C9C
1257 C
1258 ## GlutR Sum                       #454545          #B5B5B5          #7E7E7E  #D2D2D2
1259 2
1260 ## GABA<U+1D00>R Sum               #C7C7C7          #D2D2D2          #000000  #BBB
1261 BBB
1262 ## GABA<U+1D00>R:GluR              #C3FF00          #E8FF00          #FF4900  #FF0
1263 000
1264 ## GABA<U+1D00>a1:GluN2A          #C6FF00          #F4FF00          #FF2800  #FF0
1265 000
1266 ## GluN2B:GluN2A                  #FFA600          #DAFF00          #FF4400  #FF800
1267 0
1268 ## GABA<U+1D00>a1:GABA<U+1D00>a3  #F9FF00          #FF6700          #FF0000  #F
1269 F8000
1270 ## GluN2B:GluA2                    #FF2400          #FFB000          #FFEA00  #FF860
1271 0
1272 ## GluN2A:GluA2                    #FF4D00          #FF3900          #00FF00  #A1FF0
1273 0
1274 ## [[1]]

```



1275

1276 **Assign the tSNE phenotype colour-code to the object “tsne.cols”.**

1277 `tsne.cols <- phen.cols[[1]]`

1278 **Rename the 8th column of the “tsne.processed.4”.**

1279 `colnames(tsne.processed.4)[8] <- 'GABA\u1D00\u03b11:GABA\u1D00\u03b13'`

1280 **Store the “tsne.processed.4” object into the new object,**
 1281 **“tsne.processed.5”. This was done to avoid potentially overwriting**
 1282 **the contents of “tsne.processed.4”.**

1283 `tsne.processed.5 <- tsne.processed.4`

1284 **Append a subset of the column headers of “tsne.processed.5” with an**
 1285 **“\n”.**

1286 `colnames(tsne.processed.5)[2:10] <- paste0(colnames(tsne.processed.4)[c(2:10)`
 1287 `],“\n”)`

1288 **Cat: Constructing and Visualizing the Plasticity Phenotypes for**
 1289 **the Subclusters – Create Boxplots**

1290 **Create individual colour-coded boxplots for cat subcluster data.**

1291 `head(tsne.processed.5[, -ncol(tsne.processed.5)])`

1292 `##` Cluster.Name Protein Sum\n GlutR Sum\n GABA<U+1D00>R Sum\n
 1293 `## 18d RO CVF 1` RO 2 9.326704 5.307563 2.142932
 1294 `## 18d RO CVF 2` RO 2 10.556702 5.316230 2.694749
 1295 `## 18d RO MVF 1` RO 2 9.658435 5.497712 2.100966
 1296 `## 18d RO MVF 2` RO 2 8.259729 4.771556 1.727323
 1297 `## 18d RO PVF 1` RO 2 9.216756 4.490773 2.830239
 1298 `## 18d RO PVF 10` RO 2 8.754209 5.089096 1.522142
 1299 `##` GABA<U+1D00>R:GlutR\n GABA<U+1D00>a1:GluN2A\n GluN2B:GluN2A\n

```

1300 n
1301 ## 18d RO CVF 1 0.10649975 0.11897191 0.280133
1302 5
1303 ## 18d RO CVF 2 -0.00684382 -0.02566512 0.303105
1304 7
1305 ## 18d RO MVF 1 0.13359050 0.06702033 0.292176
1306 2
1307 ## 18d RO MVF 2 0.16008720 0.09124979 0.333386
1308 8
1309 ## 18d RO PVF 1 -0.11522762 -0.17862038 0.257593
1310 6
1311 ## 18d RO PVF 10 0.25140999 0.29200612 0.367567
1312 0
1313 ## GABA<U+1D00>a1:GABA<U+1D00>a3\n GluN2B:GluA2\n GluN2A:GluA2\
1314 n
1315 ## 18d RO CVF 1 0.02462795 0.3838222 0.116180
1316 6
1317 ## 18d RO CVF 2 0.05244266 0.4242947 0.139074
1318 9
1319 ## 18d RO MVF 1 0.06432353 0.5487068 0.305509
1320 6
1321 ## 18d RO MVF 2 0.16977656 0.5336853 0.243649
1322 5
1323 ## 18d RO PVF 1 0.10214623 0.4249325 0.187907
1324 2
1325 ## 18d RO PVF 10 0.03174592 0.4604927 0.111859
1326 2

1327 phenotype_boxplots(feature_df = tsne.processed.5[, -ncol(tsne.processed.5)], #
1328 Boxplot data frame
1329 phenotype_cols = tsne.cols, # Phenotype colour-code data f
1330 rame
1331 first_index_column = 4, # Index number of first index colu
1332 mn in "tsne.processed.5[, -ncol(tsne.processed.5)]" (indexes begin at 0)
1333 group_label = "\nSubclusters", # X-axis label
1334 point_size = 0.7, # Desired size of geom_jitter points
1335 point_alpha = 0.5, # Desired transparency of geom_jitter p
1336 oints
1337 aspect_ratio = 9/10) # Desired aspect ratio of each boxplo
1338 t

```

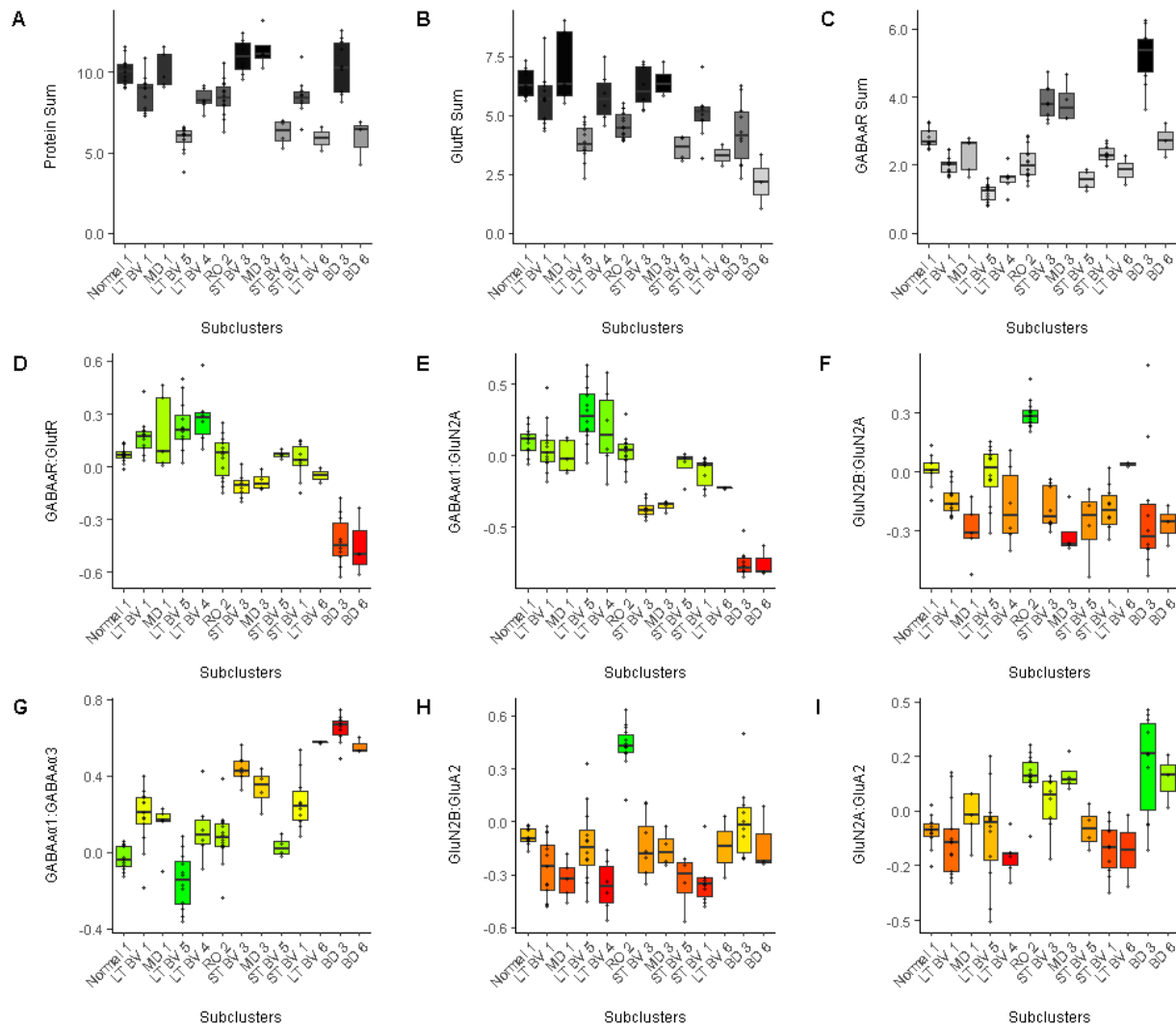
1339 **Create a single figure displaying all colour-coded boxplots for the**
1340 **subcluster phenotypes.**

```

1341 ggarrange(plotlist = plot_list,
1342           nrow = 3,
1343           ncol = 3,
1344           labels = LETTERS[1:9],
1345           font.label = list(size = 12))

```

1346



1347

1348 **Combine contents of “tsne.processed.4\$Cluster.Name2” and numbers**
 1349 **from 1 to the total number of rows in the “tsne.processed.4” data**
 1350 **frame. Assign these combined identifiers as the row names of the**
 1351 **“tsne.processed.4” object.**

1352 `rownames(tsne.processed.4) <- paste(tsne.processed.4$Cluster.Name2, 1:nrow(tsn`
 1353 `e.processed.4))`

1354 Cat: Interpretation of the plasticity phenotype and cluster
1355 analysis for classifying experience-dependent changes in V1

1356 Create an ORA_phenotype using the object, "tsne.processed.4[,2:10]"
1357 with default percentile values: "c(0.25,0.75)".

1358 `head(tsne.processed.4[,2:10])`

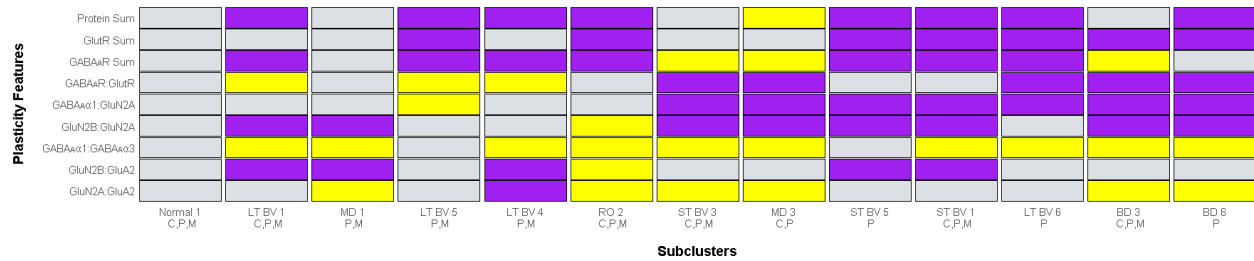
```
1359 ##          Protein Sum GlutR Sum GABA<U+1D00>R Sum GABA<U+1D00>R:GlutR
1360 ## RO 2\n C,P,M 1      9.326704  5.307563          2.142932          0.10649975
1361 ## RO 2\n C,P,M 2     10.556702  5.316230          2.694749          -0.00684382
1362 ## RO 2\n C,P,M 3      9.658435  5.497712          2.100966          0.13359050
1363 ## RO 2\n C,P,M 4      8.259729  4.771556          1.727323          0.16008720
1364 ## RO 2\n C,P,M 5      9.216756  4.490773          2.830239          -0.11522762
1365 ## RO 2\n C,P,M 6      8.754209  5.089096          1.522142          0.25140999
1366 ##          GABA<U+1D00>a1:GluN2A GluN2B:GluN2A
1367 ## RO 2\n C,P,M 1          0.11897191          0.2801335
1368 ## RO 2\n C,P,M 2         -0.02566512          0.3031057
1369 ## RO 2\n C,P,M 3          0.06702033          0.2921762
1370 ## RO 2\n C,P,M 4          0.09124979          0.3333868
1371 ## RO 2\n C,P,M 5         -0.17862038          0.2575936
1372 ## RO 2\n C,P,M 6          0.29200612          0.3675670
1373 ##          GABA<U+1D00>a1:GABA<U+1D00>a3 GluN2B:GluA2 GluN2A:GluA2
1374 ## RO 2\n C,P,M 1          0.02462795          0.3838222          0.1161806
1375 ## RO 2\n C,P,M 2          0.05244266          0.4242947          0.1390749
1376 ## RO 2\n C,P,M 3          0.06432353          0.5487068          0.3055096
1377 ## RO 2\n C,P,M 4          0.16977656          0.5336853          0.2436495
1378 ## RO 2\n C,P,M 5          0.10214623          0.4249325          0.1879072
1379 ## RO 2\n C,P,M 6          0.03174592          0.4604927          0.1118592
```

```
1380 ORA_phenotype(
1381   features_df_row = tsne.processed.4[,2:10], # Data frame for bootstrap anal
1382   ysis
1383   condition_list = as.list(c('Normal 1\n C,P,M',
1384     'LT BV 1\n C,P,M',
1385     'MD 1\n P,M',
1386     'LT BV 5\n P,M',
1387     'LT BV 4\n P,M',
1388     'RO 2\n C,P,M',
1389     'ST BV 3\n C,P,M',
1390     'MD 3\n C,P',
1391     'ST BV 5\n P',
1392     'ST BV 1\n C,P,M',
1393     'LT BV 6\n P',
1394     'BD 3\n C,P,M',
1395     'BD 6\n P')), # List of subclusters as they appear in ro
1396   w names of "features_df_row"
1397   reference_group = 'Normal 1\n C,P,M', # Name of reference group as it appe
```

```

1398 ars in the row names of "features_df_row"
1399 group_label = "\nSubclusters") # X-axis Label

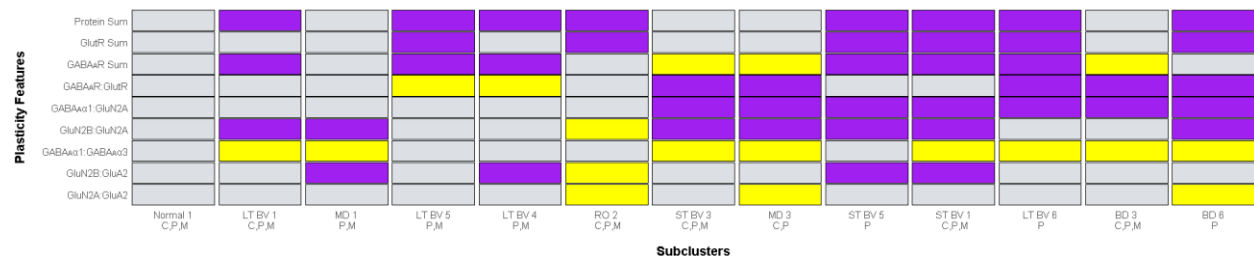
```



```

1400
1401 Create an ORA_phenotype using the object, "tsne.processed.4[,2:10]"
1402 with percentile values: "c(0.10,0.90)".
1403 ORA_phenotype(
1404   features_df_row = tsne.processed.4[,2:10], # Data frame for bootstrap anal
1405   ysis
1406   condition_list = as.list(c('Normal 1\n C,P,M',
1407     'LT BV 1\n C,P,M',
1408     'MD 1\n P,M',
1409     'LT BV 5\n P,M',
1410     'LT BV 4\n P,M',
1411     'RO 2\n C,P,M',
1412     'ST BV 3\n C,P,M',
1413     'MD 3\n C,P',
1414     'ST BV 5\n P',
1415     'ST BV 1\n C,P,M',
1416     'LT BV 6\n P',
1417     'BD 3\n C,P,M',
1418     'BD 6\n P')), # List of subclusters as they appear in ro
1419   w names of "features_df_row"
1420   reference_group = 'Normal 1\n C,P,M', # Name of reference group as it appe
1421   ars in the row names of "features_df_row"
1422   group_label = "\nSubclusters", # X-axis Label
1423   percentiles = c(0.10,0.90) # Thresholds to use for experimental subcluste
1424   rs when performing ORA against reference subclusters
1425 )

```



Cat_tSNE_plots_Figure7

<u>Table of Contents</u>		
Section Number	Title	Page Number
1	Load Cat tSNE Data	53
2	Clustering of Experience-Dependent Changes in V1 Using a Plasticity Phenotype	56
2a	Create tSNE scatter plot: Plain	56
2b	Create tSNE scatter plot: Elbow Plot	57
2c	Create tSNE scatter plot: Cluster-Coded	58
2d	Create tSNE scatter plot: Region-Coded	59
2e	Create tSNE scatter plot: Condition- & Region-Coded	61

1.) Load Cat tSNE Data

Load relevant packages and data files

Install the package, "PlasticityPhenotypes".

```
devtools::install_github("visualneurosciencelab/PlasticityPhenotypes")
```

Load the package, "PlasticityPhenotypes".

```
library(PlasticityPhenotypes)
```

```
## Loading required package: tidyverse
```

```
## -- Attaching packages ----- tidyverse  
1.3.0 --
```

```
## v ggplot2 3.3.0      v purrr  0.3.3
```

```
## v tibble  2.1.3      v dplyr  0.8.5
```

```
## v tidyr   1.0.2      v stringr 1.4.0
```

```
## v readr   1.3.1      v forcats 0.5.0
```

```
## -- Conflicts ----- tidyverse_confli  
cts() --
```

```
## x dplyr::filter() masks stats::filter()
```

```
## x dplyr::lag()     masks stats::lag()
```

```
## Loading required package: FactoMineR
```

```
## Loading required package: factoextra
```

```
## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
```

```
## Loading required package: data.table
```

```
##
```

```
## Attaching package: 'data.table'
```

```
## The following objects are masked from 'package:dplyr':
```

```
##
```

```
##      between, first, last
```

```
## The following object is masked from 'package:purrr':
```

```
##
```

```
##      transpose
```

```
## Loading required package: ggpubr
```

```

36  ## Loading required package: magrittr
37  ##
38  ## Attaching package: 'magrittr'
39  ## The following object is masked from 'package:purrr':
40  ##
41  ##     set_names
42  ## The following object is masked from 'package:tidyr':
43  ##
44  ##     extract
45  ## Loading required package: corrplot
46  ## corrplot 0.84 loaded
47  ## Loading required package: psych
48  ##
49  ## Attaching package: 'psych'
50  ## The following objects are masked from 'package:ggplot2':
51  ##
52  ##     %+%, alpha
53  ## Loading required package: httr

```

54 **Store file paths in unique objects.**

```

55 raw.cat.tsne <- 'https://osf.io/59yu6//?action=download'
56 wss <- 'https://osf.io/t73fc//?action=download'

```

57 **Import cat subcluster/tSNE data from OSF, and store them in the**
58 **object, “tsne.raw.data”. This data frame consists of 6 attributes**
59 **(“Condition”, “Labels”, “Cluster.Number”, “Reordered”,**
60 **“Cluster.Name”, & “Region”), and 2 tSNE X & Y coordinates. The**
61 **object “elbow.data” consists of one column for ascending cluster**
62 **numbers (‘k’) and another for the within groups sums of squares of**
63 **each cluster (“ss”).**

```

64 filename <- 'cat_tsne.csv'
65 GET(raw.cat.tsne, write_disk(filename, overwrite = TRUE))
66 ## Response [https://files.ca-1.osf.io/v1/resources/8a3kx/providers/osfstora
67 ## e/5ed0684f17ac9e0316621448?action=download&direct&version=1]
68 ##   Date: 2020-06-03 18:00
69 ##   Status: 200

```

```

70  ## Content-Type: text/csv
71  ## Size: 13.9 kB
72  ## <ON DISK> C:\Users\dezia\Dropbox (Kathy Murphy)\JB-ProteinAnalysisWorkflo
73  w\JB ProteinAnalysisWorkflow Markdowns\RMD files\Cat Analysis\cat_tsne.csv

74  tsne.raw.data <- read.csv(filename)
75  head(tsne.raw.data)

76  ## Condition Labels V1 V2 Cluster.Number Reordered
77  ## 1 Normal 5wk Normal CVF 1 -1.421038 1.926670 1 1
78  ## 2 Normal 5wk Normal CVF 2 -1.182249 3.150573 1 1
79  ## 3 Normal 5wk Normal MVF 1 -1.488770 1.005343 1 1
80  ## 4 Normal 5wk Normal MVF 2 -1.511189 1.509622 1 1
81  ## 5 Normal 5wk Normal PVF 1 -1.438462 3.133959 1 1
82  ## 6 Normal 5wk Normal PVF 2 -1.956975 2.357059 1 1
83  ## Cluster.Name Region
84  ## 1 Normal 1 CVF
85  ## 2 Normal 1 CVF
86  ## 3 Normal 1 MVF
87  ## 4 Normal 1 MVF
88  ## 5 Normal 1 PVF
89  ## 6 Normal 1 PVF

90  filename <- 'and stored in the file wss.csv'
91  GET(wss, write_disk(filename, overwrite = TRUE))

92  ## Response [https://files.ca-1.osf.io/v1/resources/8a3kx/providers/osfstora
93  ge/5ed7dce7237cd3009d54b26e?action=download&direct&version=1]
94  ## Date: 2020-06-03 18:00
95  ## Status: 200
96  ## Content-Type: text/csv
97  ## Size: 236 B
98  ## <ON DISK> C:\Users\dezia\Dropbox (Kathy Murphy)\JB-ProteinAnalysisWorkflo
99  w\JB ProteinAnalysisWorkflow Markdowns\RMD files\Cat Analysis\wss.csv

100  elbow.data <- read.csv(filename)
101  head(elbow.data)

102  ## k ss
103  ## 1 1 2744.0456
104  ## 2 2 1407.4805
105  ## 3 3 737.2642
106  ## 4 4 642.9166
107  ## 5 5 316.1098
108  ## 6 6 228.6830

```

2.) Clustering of Experience-Dependent Changes in V1 Using a Plasticity Phenotype

Subset “tsne.raw.data” to only return observations where there are no NAs in the “Cluster.Number” column.

```
tsne.raw.data <- subset(tsne.raw.data,  
  is.na(tsne.raw.data$Cluster.Number) == F)
```

Set a standard size for data points and axis label size across all tSNE scatter plots.

```
ptsize = 5  
txtsize = 20
```

Set a standard size for data points, axis tick labels, and axis titles on the elbow plot.

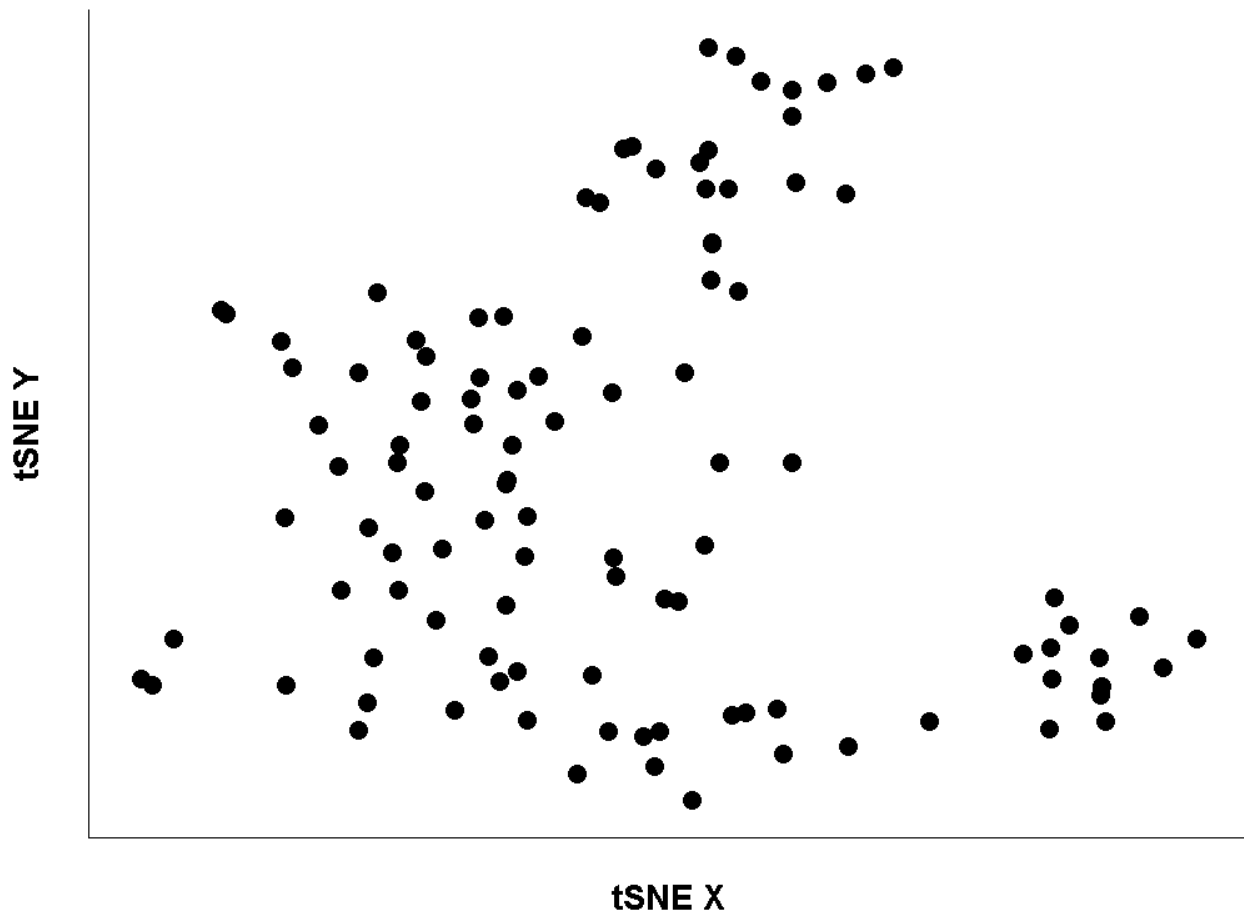
```
#data point size for elbow plot = "ptsize"  
txtsize.elb = 12  
titlesize.elb = 14
```

Create tSNE scatterplots

a.) Plain: Create a tSNE scatter plot with no mapped aesthetics.

```
noncolor <- ggplot(tsne.raw.data,  
  aes(x=V1,  
    y=V2))+  
  geom_point(size = ptsize)+  
  xlab("\ntSNE X")+  
  ylab("tSNE Y\n")+  
  theme_classic()+  
  theme(axis.text.x = element_blank(),  
    axis.text.y = element_blank(),  
    axis.ticks.x=element_blank(),  
    axis.ticks.y=element_blank(),  
    axis.title.x = element_text(size = txtsize, face =  
'bold'),  
    axis.title.y = element_text(size = txtsize, face =  
'bold'),  
    legend.position = "none")
```

```
142
143 print(noncolor)
```



```
144
```

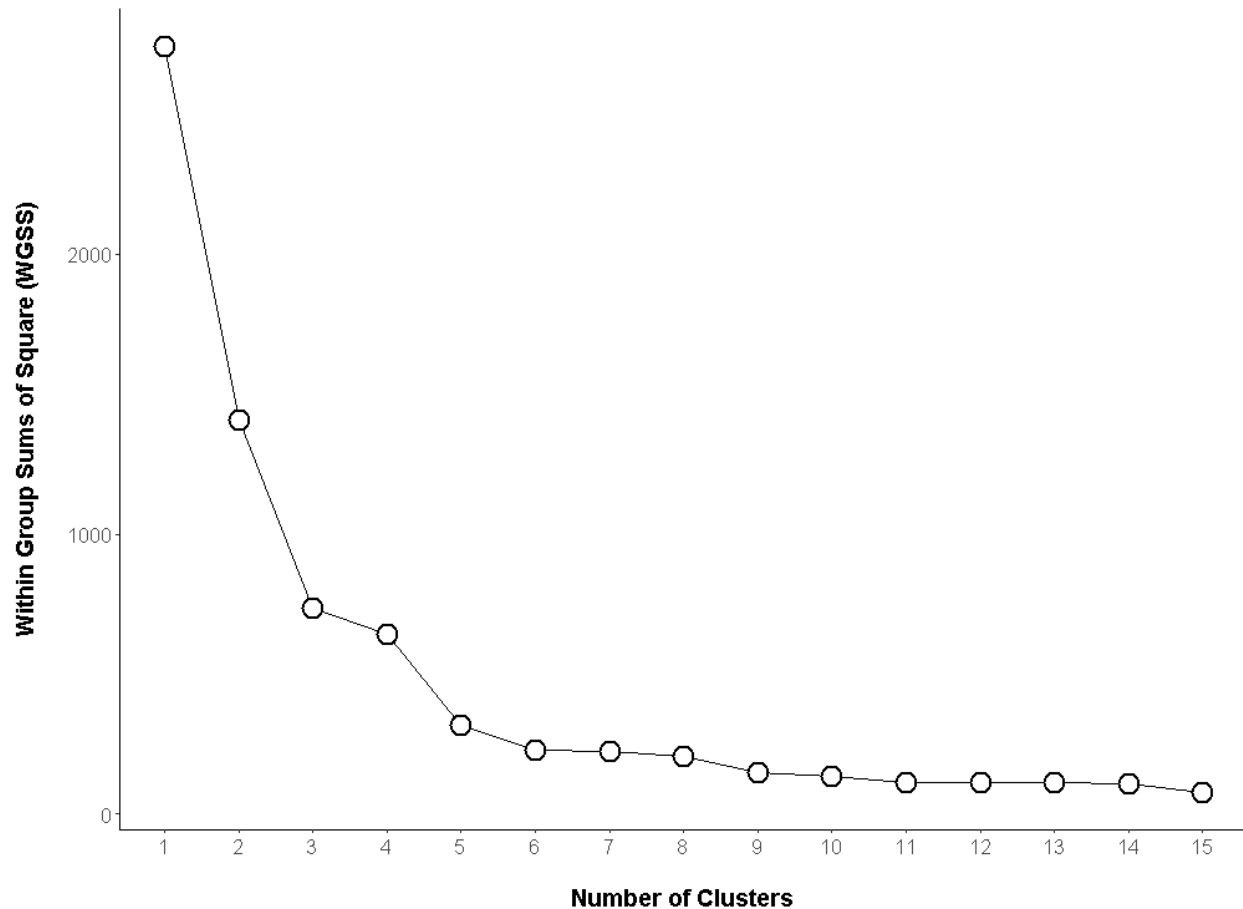
145 **b.) Elbow Plot: Create an elbow plot that identifies how many**
 146 **clusters are required to minimize the within group sums of**
 147 **squares.**

```
148 elb.plot <- ggplot(elbow.data, aes(x = factor(k),
149                                   y = ss,
150                                   group = 1)) +
151   xlab("\nNumber of Clusters") +
152   ylab("Within Group Sums of Square (WGSS)\n") +
153   theme_classic() +
154   geom_line() +
155   geom_point(shape=21, fill="white",
156             color="black",
157             size=ptsize,
158             stroke = 1.3)+
159   theme(axis.text.x = element_text(size = txtsize.elb),
160         axis.text.y = element_text(size = txtsize.elb),
161         axis.title.x = element_text(size = titlesize.elb, face = "bold"),
```

```

162     axis.title.y = element_text(size = titlesize.elb, face = "bold"))
163 print(elb.plot)

```



164

c.) Cluster-Coded: Create a tSNE scatter plot where points are coloured according to their cluster membership.

165

166

167

168

169

170

171

172

173

174

175

176

177

178

179

180

181

```

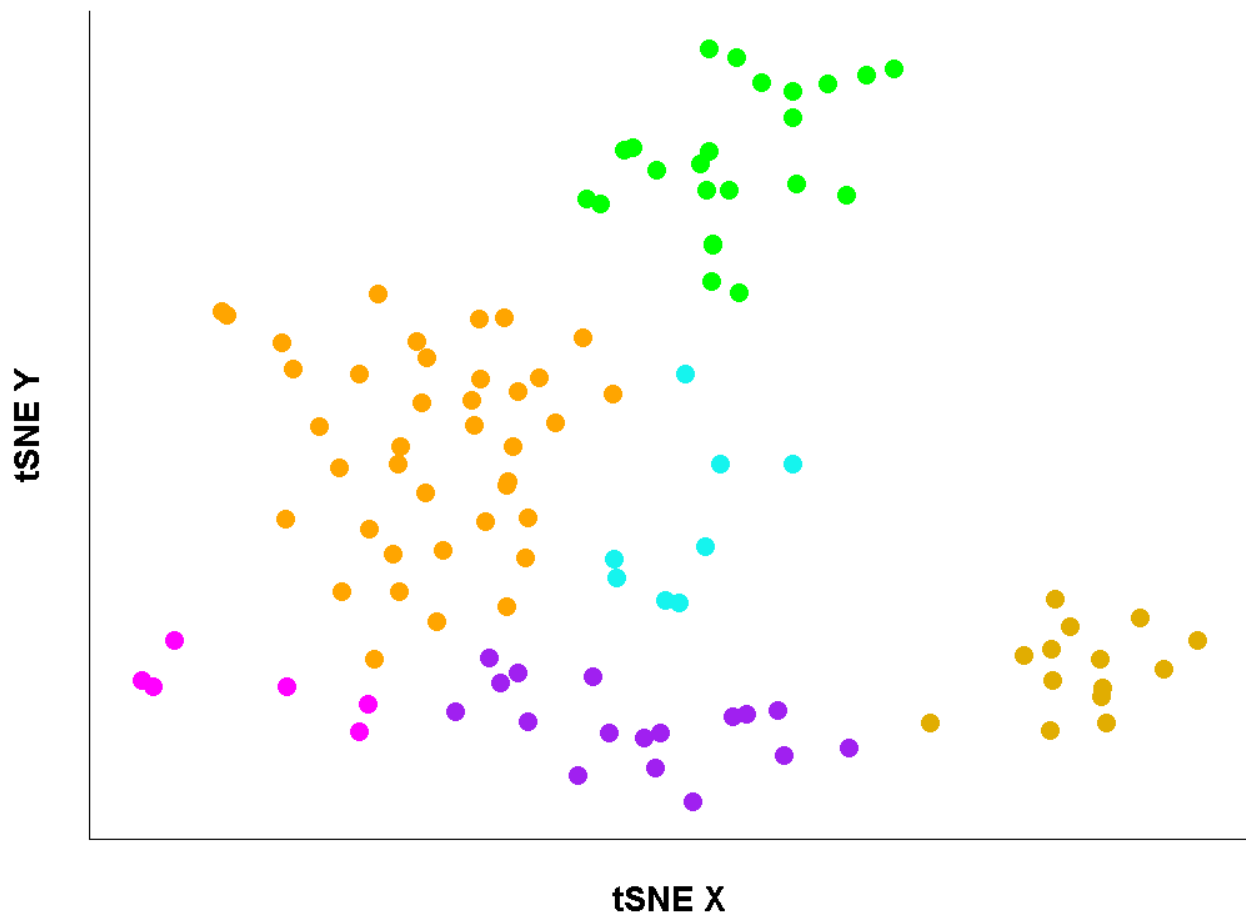
color <- ggplot(tsne.raw.data,
               aes(x=V1,
                  y=V2,
                  color = as.factor(Cluster.Number)))+
  geom_point(size = ptsize)+
  xlab("\ntSNE X")+
  ylab("tSNE Y\n")+
  theme_classic()+
  theme(axis.text.x = element_blank(),
        axis.text.y = element_blank(),
        axis.ticks.x=element_blank(),
        axis.ticks.y=element_blank(),
        axis.title.x = element_text(size = txtsize, face = 'bold'),
        axis.title.y = element_text(size = txtsize, face = 'bold'))

```

```

182 d')
183     ,legend.position = "none"
184     )+scale_color_manual(values = c('orange',
185                                     'green',
186                                     '#15F4EE',
187                                     '#E1AD01',
188                                     'purple',
189                                     '#FF00FF'))
190 print(color)

```



191

192 **d.) Region-Coded: Create a tSNE scatter plot where points are**
 193 **shaped according to the region of the visual cortex they were**
 194 **sampled from.**

```

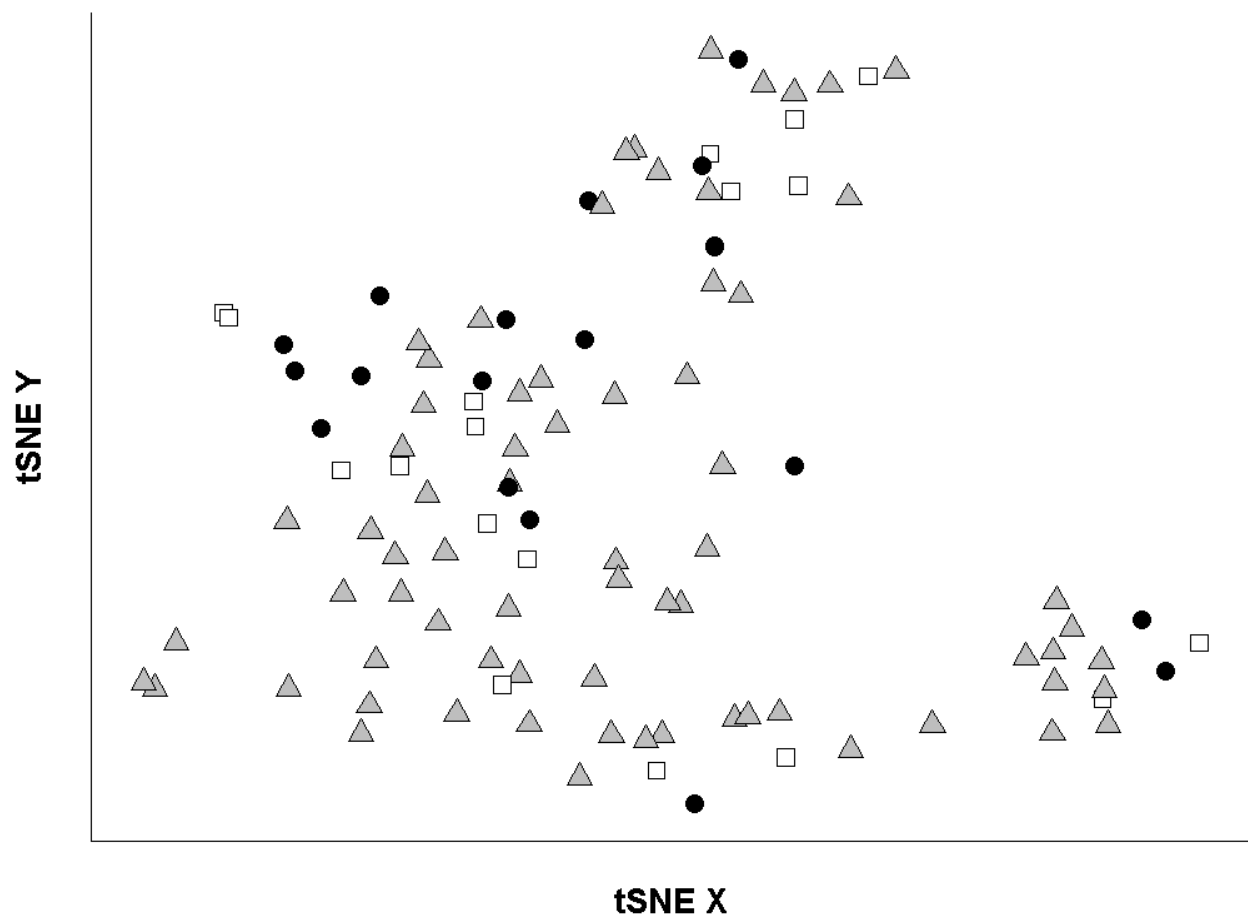
195 shapes <- ggplot(tsne.raw.data,
196                  aes(x=V1,
197                      y=V2,
198                      shape = Region,
199                      fill = Region))+
200   geom_point(size = ptsize)+
201   xlab("\ntSNE X")+

```

```

202     ylab("tSNE Y\n")+
203     theme_classic()+
204     theme(axis.text.x = element_blank(),
205           axis.text.y = element_blank(),
206           axis.ticks.x=element_blank(),
207           axis.ticks.y=element_blank(),
208           axis.title.x = element_text(size = txtsize, f
209 ace = 'bold'),
210           axis.title.y = element_text(size = txtsize, f
211 ace = 'bold'),
212           legend.position = "none")+
213     scale_shape_manual(values = c(21,22,24))+
214     scale_fill_manual(values = c('black','white','grey'))
215 print(shapes)

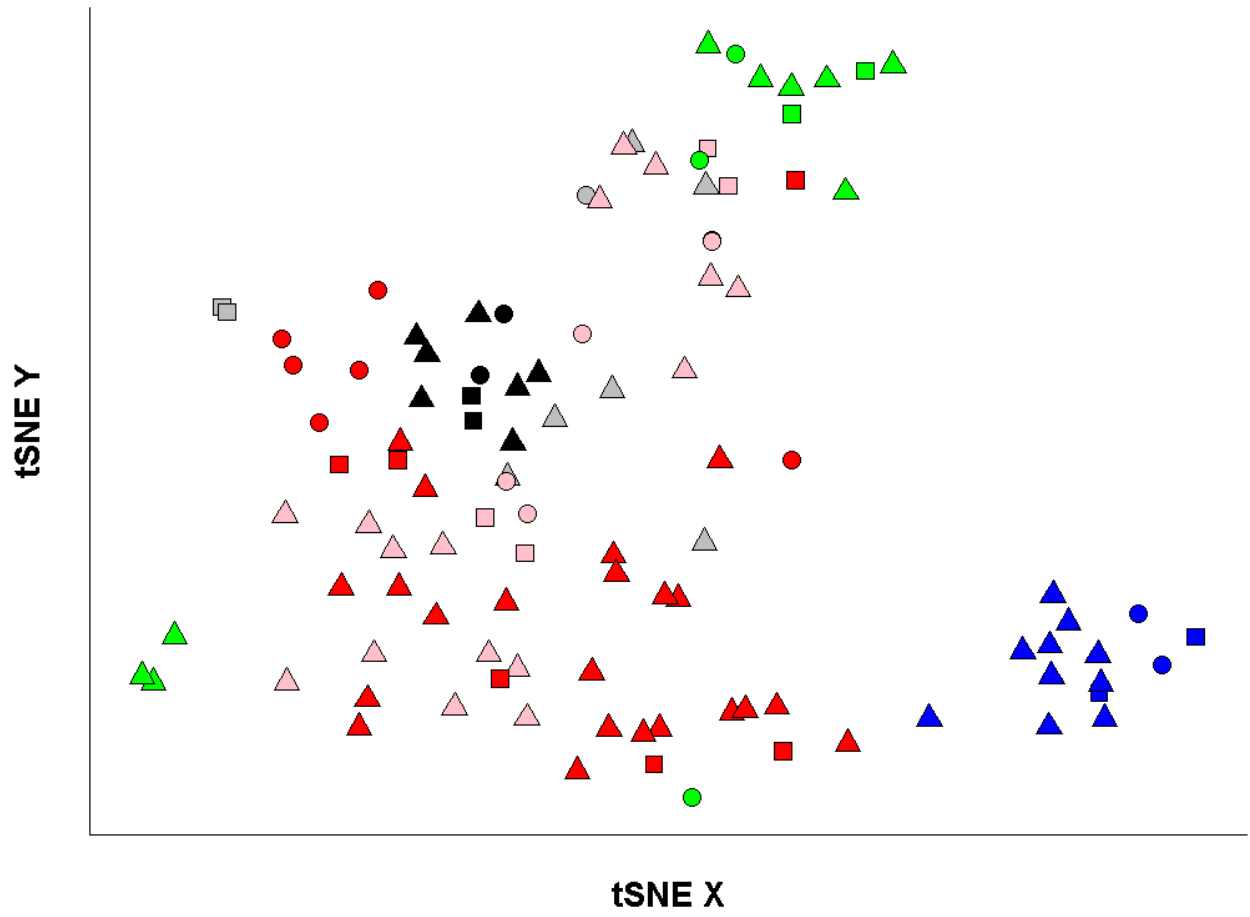
```



216

e.) Condition- & Region-Coded: Create a tSNE scatter plot where points are shaped according to the region of the visual cortex they were sampled from, and coloured based on which condition they were in.

```
col.shapes <- ggplot(tsne.raw.data,
  aes(x=V1,
      y=V2,
      shape = Region,
      fill = Condition))+
  geom_point(size = ptsize)+
  xlab("\ntSNE X")+
  ylab("tSNE Y\n")+
  theme_classic()+
  theme(axis.text.x = element_blank(),
        axis.text.y = element_blank(),
        axis.ticks.x=element_blank(),
        axis.ticks.y=element_blank(),
        axis.title.x = element_text(size = txtsize, f
ace = 'bold'),
        axis.title.y = element_text(size = txtsize, f
ace = 'bold'),
        legend.position = "none")+
  scale_shape_manual(values = c(21,22,24))+
  scale_fill_manual(values = c('green',
                                'red',
                                'grey',
                                'black',
                                'blue',
                                'pink'))
print(col.shapes)
```



247

Cat_CorrHeatmap_Figure8

<u>Table of Contents</u>		
Section Number	Title	Page Number
1	Load Cat Correlation Data	64
2	Process the Cat tSNE Correlation Data	67
3	Identifying and Exploring Subclusters using Plasticity Features and Phenotypes – Create Heatmap	68

5

1.) Load Cat Correlation Data

6

Install the package, “PlasticityPhenotypes”.

7

```
devtools::install_github("visualneurosciencelab/PlasticityPhenotypes")
```

8

Load the package, “PlasticityPhenotypes”.

9

```
library(PlasticityPhenotypes)
```

10

```
## Loading required package: tidyverse
```

11

```
## -- Attaching packages -----
```

12

```
## v ggplot2 3.3.0      v purrr  0.3.3
```

13

```
## v tibble  2.1.3      v dplyr  0.8.5
```

14

```
## v tidyr   1.0.2      v stringr 1.4.0
```

15

```
## v readr   1.3.1      v forcats 0.5.0
```

16

```
## -- Conflicts --- tidyverse_conflicts() --
```

17

```
## x dplyr::filter() masks stats::filter()
```

18

```
## x dplyr::lag()    masks stats::lag()
```

19

```
## Loading required package: FactoMineR
```

20

```
## Loading required package: factoextra
```

21

```
## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
```

22

23

```
## Loading required package: data.table
```

24

```
##
```

25

```
## Attaching package: 'data.table'
```

26

```
## The following objects are masked from 'package:dplyr':
```

27

```
##
```

28

```
##      between, first, last
```

29

```
## The following object is masked from 'package:purrr':
```

30

```
##
```

31

```
##      transpose
```

32

```
## Loading required package: ggpubr
```

33

```
## Loading required package: magrittr
```

34

```
##
```

35

```
## Attaching package: 'magrittr'
```

```
36  ## The following object is masked from 'package:purrr':
37  ##
38  ##      set_names
39  ## The following object is masked from 'package:tidyr':
40  ##
41  ##      extract
42  ## Loading required package: corrplot
43  ## corrplot 0.84 loaded
44  ## Loading required package: psych
45  ##
46  ## Attaching package: 'psych'
47  ## The following objects are masked from 'package:ggplot2':
48  ##
49  ##      %+%, alpha
50  ## Loading required package: httr
51  library(gplots)
52  ##
53  ## Attaching package: 'gplots'
54  ## The following object is masked from 'package:stats':
55  ##
56  ##      lowess
57  library(RColorBrewer)
58  library(Hmisc)
59  ## Loading required package: lattice
60  ## Loading required package: survival
61  ## Loading required package: Formula
62  ##
63  ## Attaching package: 'Hmisc'
64  ## The following object is masked from 'package:psych':
65  ##
66  ##      describe
67  ## The following objects are masked from 'package:dplyr':
68  ##
69  ##      src, summarize
```

```

70 ## The following objects are masked from 'package:base':
71 ##
72 ##     format.pval, units

```

73 Store the file path in a unique object.

```

74 raw.cat.corr <- "https://osf.io/5d4kt/?action=download"

```

75 Import the necessary CSV from OSF. The “raw.data” object consists of 76 10 columns – 1 attributes column (“Cluster.Name”) and 9 plasticity 77 feature columns.

```

78 filename <- 'cat_correlation.csv'
79 GET(raw.cat.corr, write_disk(filename, overwrite = TRUE))

80 ## Response [https://files.ca-1.osf.io/v1/resources/8a3kx/providers/osfstorag
81 e/5ecd57e117ac9e01fa62506b?action=download&direct&version=1]
82 ##   Date: 2020-06-02 19:45
83 ##   Status: 200
84 ##   Content-Type: text/csv
85 ##   Size: 17.9 kB
86 ## <ON DISK> C:\Users\dezia\Dropbox (Kathy Murphy)\JB-ProteinAnalysisWorkflo
87 w\JB ProteinAnalysisWorkflow Markdowns\RMD files\Cat Analysis\cat_correlation
88 .csv

89 raw.data <- read.csv(filename, row.names = 1)
90 head(raw.data)

91 ##           Cluster.Name Protein.Sum.  GlutR.Sum.  GABA.U.1D00.R.Sum.
92 ## 18d RO CVF 1           RO 2      9.326704    5.307563          2.142932
93 ## 18d RO CVF 2           RO 2     10.556702    5.316230          2.694749
94 ## 18d RO MVF 1           RO 2      9.658435    5.497712          2.100966
95 ## 18d RO MVF 2           RO 2      8.259729    4.771556          1.727323
96 ## 18d RO PVF 1           RO 2      9.216756    4.490773          2.830239
97 ## 18d RO PVF 10          RO 2      8.754209    5.089096          1.522142
98 ##           GABA.U.1D00.R.GlutR.  GABA.U.1D00.a1.GluN2A.  GluN2B.GluN2A.
99 ## 18d RO CVF 1           0.10649975          0.11897191          0.2801335
100 ## 18d RO CVF 2          -0.00684382          -0.02566512          0.3031057
101 ## 18d RO MVF 1           0.13359050          0.06702033          0.2921762
102 ## 18d RO MVF 2           0.16008720          0.09124979          0.3333868
103 ## 18d RO PVF 1          -0.11522762          -0.17862038          0.2575936
104 ## 18d RO PVF 10          0.25140999          0.29200612          0.3675670
105 ##           GABA.U.1D00.a1.GABA.U.1D00.a3.  GluN2B.GluA2.  GluN2A.GluA2.
106 ## 18d RO CVF 1           0.02462795          0.3838222          0.1161806
107 ## 18d RO CVF 2           0.05244266          0.4242947          0.1390749
108 ## 18d RO MVF 1           0.06432353          0.5487068          0.3055096
109 ## 18d RO MVF 2           0.16977656          0.5336853          0.2436495
110 ## 18d RO PVF 1           0.10214623          0.4249325          0.1879072
111 ## 18d RO PVF 10          0.03174592          0.4604927          0.1118592

```

2.) Process the Cat tSNE Correlation Data

Store contents of “raw.data\$Cluster.Name” as a data frame in the object, “ids.3”.

```
ids.3 <- as.data.frame(raw.data$Cluster.Name)
```

Rename “ids.3” column header.

```
colnames(ids.3) <- 'sample.ids'
```

Calculate the median of each feature across subclusters, and store in ‘med.df’.

```
all.df <- group_by(raw.data, ids.3$sample.ids)
```

```
med.df <- summarise_at(all.df, vars(colnames(raw.data)[-1]), median)
```

Rename the first column of “med.df” to “subcluster”. Transform the “med.df\$subcluster” from a column to individual row names. Store the output into the object, “med.df.2”.

```
colnames(med.df)[1] <- 'subcluster'
```

```
med.df.2 <- column_to_rownames(med.df, var = 'subcluster')
```

Transpose the “med.df.2”, and store in “t.med.df”.

```
t.med.df <- as.matrix(t(med.df.2))
```

3.) Exploring Subclusters using Plasticity Phenotypes – Create Heatmap

Perform a Pearson's correlation against all subclusters to each other, and store in "data.correlation".

```
data.correlation <- rcorr(as.matrix(t.med.df),  
                          type = "pearson")
```

Save correlations coefficients as a matrix in "CorMat".

```
CorMat <- as.matrix(data.correlation$r)
```

Create custom color palette for heatmap.

```
my_palette <- colorRampPalette(c(  
  "blue",  
  "#00CED1",  
  "green",  
  "yellow",  
  "orange",  
  "red"))(n = 6000)
```

Sets heatmap margins.

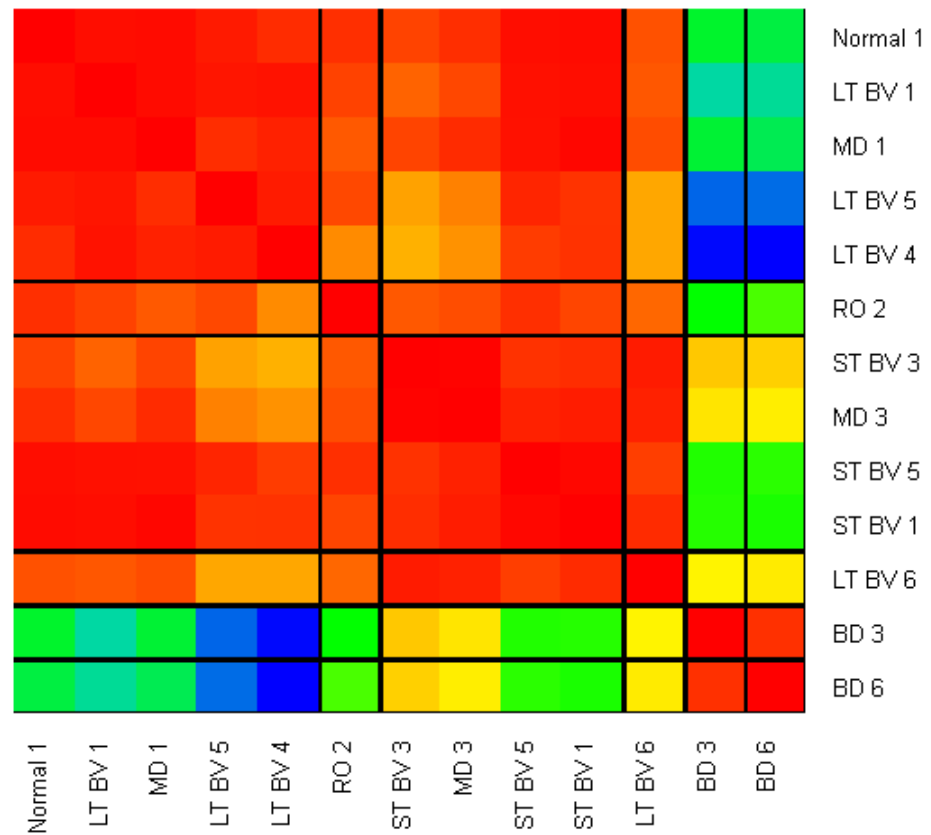
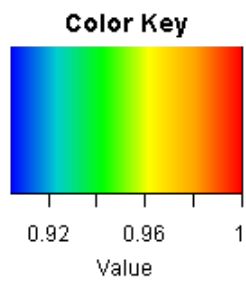
```
par(oma=c(rep(7,4)))
```

Create a character vector specifying the order of subclusters, called "clust_order".

```
clust_order <- c("Normal 1",  
  "LT BV 1",  
  "MD 1",  
  "LT BV 5",  
  "LT BV 4",  
  "RO 2",  
  "ST BV 3",  
  "MD 3",  
  "ST BV 5",  
  "ST BV 1",  
  "LT BV 6",  
  "BD 3",  
  "BD 6")
```

164 **Creates heatmap.**

```
165 heatmap.2(CorMat[clust_order,  
166           clust_order],  
167           cexRow = 1,  
168           Colv = F,  
169           Rowv = F,  
170           colsep=c(5,  
171                   6,  
172                   10,  
173                   11,  
174                   12,  
175                   13),  
176           rowsep=c(5,  
177                   6,  
178                   10,  
179                   11,  
180                   12,  
181                   13),  
182           cexCol=1,  
183           sepcolor="black",  
184           density.info="none",  
185           trace="none",  
186           col = my_palette,  
187           revc = T,  
188           symm=T)  
  
189 ## Warning in heatmap.2(CorMat[clust_order, clust_order], cexRow = 1, Colv =  
190 F, :  
191 ## Discrepancy: Rowv is FALSE, while dendrogram is `both'. Omitting row dendo  
192 gram.  
  
193 ## Warning in heatmap.2(CorMat[clust_order, clust_order], cexRow = 1, Colv =  
194 ## F, : Discrepancy: Colv is FALSE, while dendrogram is `column'. Omitting co  
195 lumn  
196 ## dendrogram.  
  
197 ## Warning in plot.window(...): "revc" is not a graphical parameter  
198 ## Warning in plot.xy(xy, type, ...): "revc" is not a graphical parameter  
199 ## Warning in title(...): "revc" is not a graphical parameter
```



Rat_Markdown

<u>Table of Contents</u>		
Section Number	Title	Page Number
1	Load Rat Data	72
2	Rat Analysis	75
3	Rat Phenotype	91
4	Create Rat Boxplots	93
5	Rat Plasticity Phenotype Bootstrap Analysis	96

1.) Load Rat Data

Rat: Dimension Reduction using PCA – Data Processing

Install the package, “PlasticityPhenotypes”.

```
devtools::install_github("visualneurosciencelab/PlasticityPhenotypes")
```

Load the package, “PlasticityPhenotypes”.

```
library(PlasticityPhenotypes)

## Loading required package: tidyverse

## -- Attaching packages ----- tidyverse 1.3.0 --

## v ggplot2 3.3.0      v purrr  0.3.3
## v tibble  2.1.3      v dplyr  0.8.5
## v tidyr   1.0.2      v stringr 1.4.0
## v readr   1.3.1      v forcats 0.5.0

## -- Conflicts ----- tidyverse
## _conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()

## Loading required package: FactoMineR

## Loading required package: factoextra

## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa

## Loading required package: data.table

##
## Attaching package: 'data.table'

## The following objects are masked from 'package:dplyr':
##
##   between, first, last

## The following object is masked from 'package:purrr':
##
##   transpose

## Loading required package: ggpubr

## Loading required package: magrittr
```

```

37  ##
38  ## Attaching package: 'magrittr'

39  ## The following object is masked from 'package:purrr':
40  ##
41  ##   set_names

42  ## The following object is masked from 'package:tidyr':
43  ##
44  ##   extract

45  ## Loading required package: corrplot
46  ## corrplot 0.84 loaded

47  ## Loading required package: psych
48  ##
49  ## Attaching package: 'psych'

50  ## The following objects are masked from 'package:ggplot2':
51  ##
52  ##   %+%, alpha

53  ## Loading required package: httr

```

54 **Store the file path in a unique object.**

```

55  raw.rat <- 'https://osf.io/d5pzt/?action=download'

```

56 **Import the necessary CSV. The “raw.data” object consists of 5 57 attribute columns (“ID”, “Condition”, “Region”, “Hemisphere”, & 58 “Run”), and 10 columns of distinct proteins’ expression.**

```

59  filename <- 'rat_protein.csv'
60  GET(raw.rat, write_disk(filename, overwrite = TRUE))

61  ## Response [https://files.ca-1.osf.io/v1/resources/8a3kx/providers/osfstorag
62  e/5ece9211aeeb6d025a085f0d?action=download&direct&version=3]
63  ##   Date: 2020-06-09 23:53
64  ##   Status: 200
65  ##   Content-Type: text/csv
66  ##   Size: 12 kB
67  ## <ON DISK> C:\Users\dezia\Dropbox (Kathy Murphy)\JB-ProteinAnalysisWorkflo
68  w\JB ProteinAnalysisWorkflow Markdowns\RMD files June 2020\Rat Analysis\Curre
69  nt\rat_protein.csv

70  raw.data <- read.csv(filename)
71  head(raw.data)

```

72	##	Case_Code	Condition	Region	Hemisphere	Run	GluA2	GluN1	GluN2A
73	## 1	231	Prozac_N	VC	Cont	1	1.0365590	0.5287500	0.9606986
74	## 2	232	Prozac_N	VC	Cont	1	1.1490010	0.9744765	0.9863363
75	## 3	233	Prozac_N	VC	Cont	1	1.1218811	1.5784559	1.5190979
76	## 4	234	Prozac_N	VC	Cont	1	1.0465851	1.0153453	0.8924184
77	## 5	235	Prozac_N	VC	Cont	1	0.8387192	0.7336301	0.3954039
78	## 6	236	Prozac_N	VC	Cont	1	1.2681864	1.1609523	1.0274127
79	##	GluN2B	GABAAa1	GABAAa3	PSD95	Gephyrin	VGLUT	VGAT	
80	## 1	0.6609931	0.6081925	1.0732763	0.5933443	1.0182930	0.6543510	0.6877782	
81	## 2	0.8524244	1.0276800	1.0655188	2.4078841	2.5542893	0.8957831	0.9694615	
82	## 3	0.8402657	1.8131427	1.1594168	0.6794247	1.1658637	0.8531914	0.9891639	
83	## 4	0.6999245	1.1133509	0.8682642	1.0641081	2.0030711	0.6892650	0.8166134	
84	## 5	0.4312522	0.8327478	0.5771284	0.9394761	1.4427779	0.3932702	0.6479182	
85	## 6	1.3597457	1.2145710	1.0280006	0.6317986	0.9063774	0.9291042	1.2881240	

2.) Rat Analysis

Rat: Dimension Reduction Using PCA – Additional Processing & Analysis

Rename the levels of “raw.data\$Condition” into a manner consistent with (Beshara et al., 2015), and reassign to "raw.data\$Condition".

```
raw.data$Condition <- plyr::mapvalues(raw.data$Condition,  
                                     unique(raw.data$Condition),  
                                     c('flx',  
                                       'flx + 1wk MD',  
                                       'normal',  
                                       '1wk MD'))
```

Rename protein data columns of “raw.data” that contain special characters.

```
colnames(raw.data)[10:11] <- c("GABA\u1D00\u03b11",  
                              'GABA\u1D00\u03b13')
```

Retrieve cases without missing values in “raw.data[complete.cases(raw.data),]”, and reassign them to the object, “syn.prots”.

```
syn.prots <- raw.data[complete.cases(raw.data),]
```

Assign the protein columns of “syn.prots” to the new object, “my.data”.

```
my.data <- syn.prots[,c("GluA2",  
                       "GluN1",  
                       "GluN2A",  
                       "GluN2B",  
                       "GABA\u1D00\u03b11",  
                       "GABA\u1D00\u03b13",  
                       "Gephyrin",  
                       "PSD95",  
                       "VGLUT",  
                       "VGAT")]
```

Concatenate attribute columns of “syn.prots” and assign as distinct row names of “my.data”.

```
rownames(my.data) <- paste(syn.prots$Case_Code,  
                           syn.prots$Condition,  
                           syn.prots$Region,  
                           syn.prots$Hemisphere,  
                           syn.prots$Run, sep = " ")
```

Centre data and scale the contents of “my.data”, and assign them to the object, “my.data.scaled”.

```
my.data.scaled <- scale(my.data,  
                        center = TRUE,  
                        scale = TRUE)
```

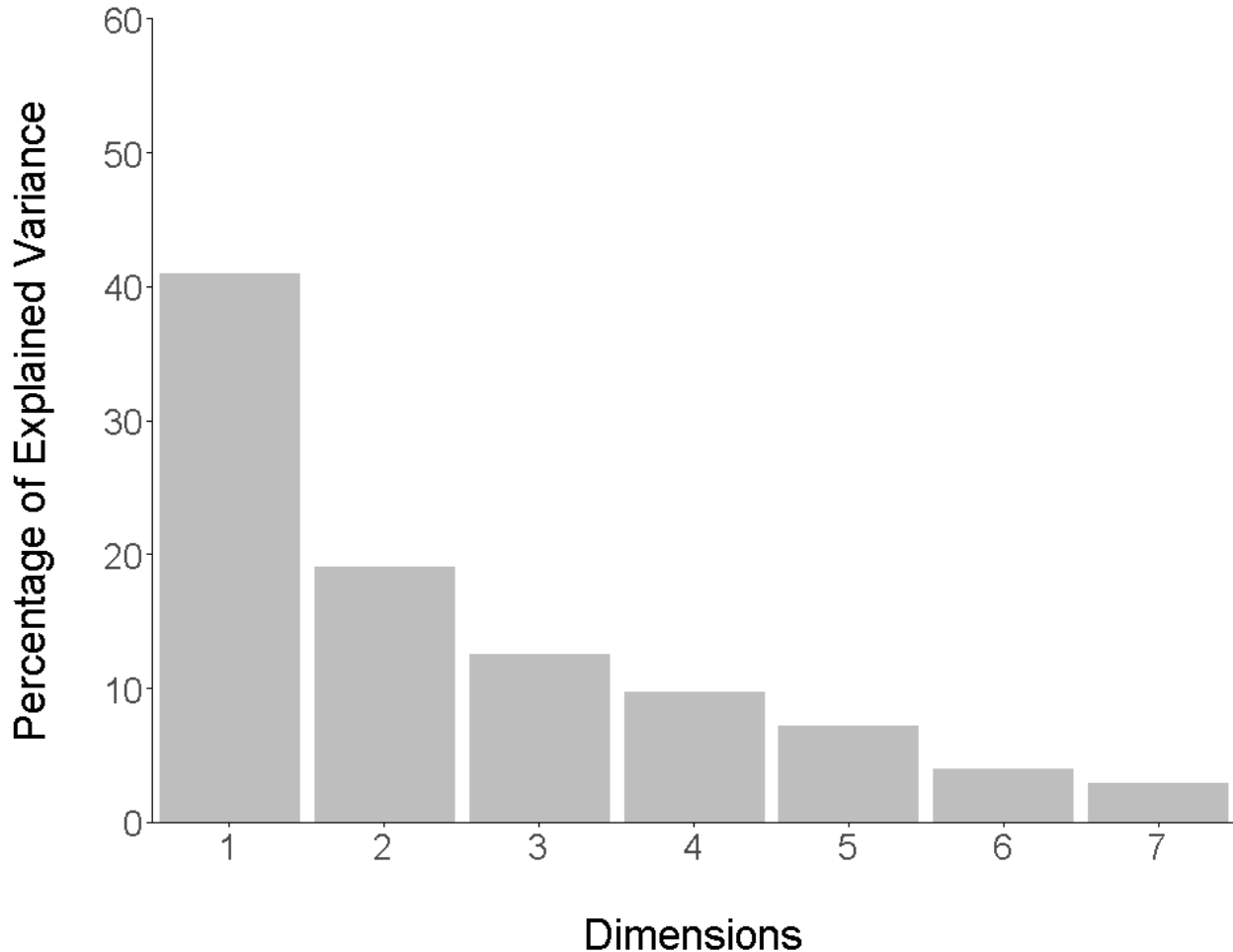
Perform PCA on “my.data.scaled”, and store in the object, “pca.scaled”.

```
pca.scaled <- PCA(my.data.scaled,  
                 ncp=ncol(my.data.scaled),  
                 scale.unit=FALSE,  
                 graph = FALSE)
```

Construct scree plot.

```
fviz_eig(pca.scaled,  
         ylim = c(0, 60),  
         xlim = c(0.5, 7.5),  
         ncp = 7, # Select number of principal components using 'ncp' parameter  
         barfill = "grey",  
         barcolor = "grey",  
         geom = "bar", text = TRUE  
         )+  
  scale_y_continuous(expand = c(0,0))+  
  scale_x_discrete(expand = c(0,0))+  
  theme(axis.line.y=element_line(),  
        axis.line.x=element_line(),  
        panel.grid=element_blank(),  
        axis.text.x = element_text(size = 20),  
        axis.text.y = element_text(size = 20),  
        text = element_text(size = 25)  
  )+  
  xlab("\nDimensions")+  
  ylab("Percentage of Explained Variance\n")+  
  ggtitle("Scree Plot\n")
```

Scree Plot



157

Dimensions

158

Rat: Identifying Candidate Plasticity Features

159

Construct plot of the $\cos^2$ data stored in the object, "pca.scaled".

160

```
pca.scaled$var$cos2
```

161

	Dim.1	Dim.2	Dim.3	Dim.4	Dim.5
## GluA2	0.76474184	0.030976705	0.004973224	3.038472e-02	0.004758510
## GluN1	0.74240651	0.002971197	0.022233983	6.479174e-03	0.088372864
## GluN2A	0.44898824	0.090426240	0.239947534	9.384183e-03	0.088170565
## GluN2B	0.48360386	0.179034865	0.069430784	2.961603e-03	0.144965377
## GABA<U+1D00>a1	0.06119843	0.072946897	0.040036126	7.031178e-01	0.112828020
## GABA<U+1D00>a3	0.63841652	0.010636313	0.144280232	6.277124e-07	0.009637956
## Gephyrin	0.15196515	0.686260112	0.054970569	1.328582e-02	0.036682070
## PSD95	0.06444387	0.825600954	0.017532551	2.944617e-02	0.005298097
## VGLUT	0.72309081	0.002824324	0.066308229	3.061224e-02	0.025621447
## VGAT	0.01576539	0.008665241	0.593194429	1.474058e-01	0.207785267

162

163

164

165

166

167

168

169

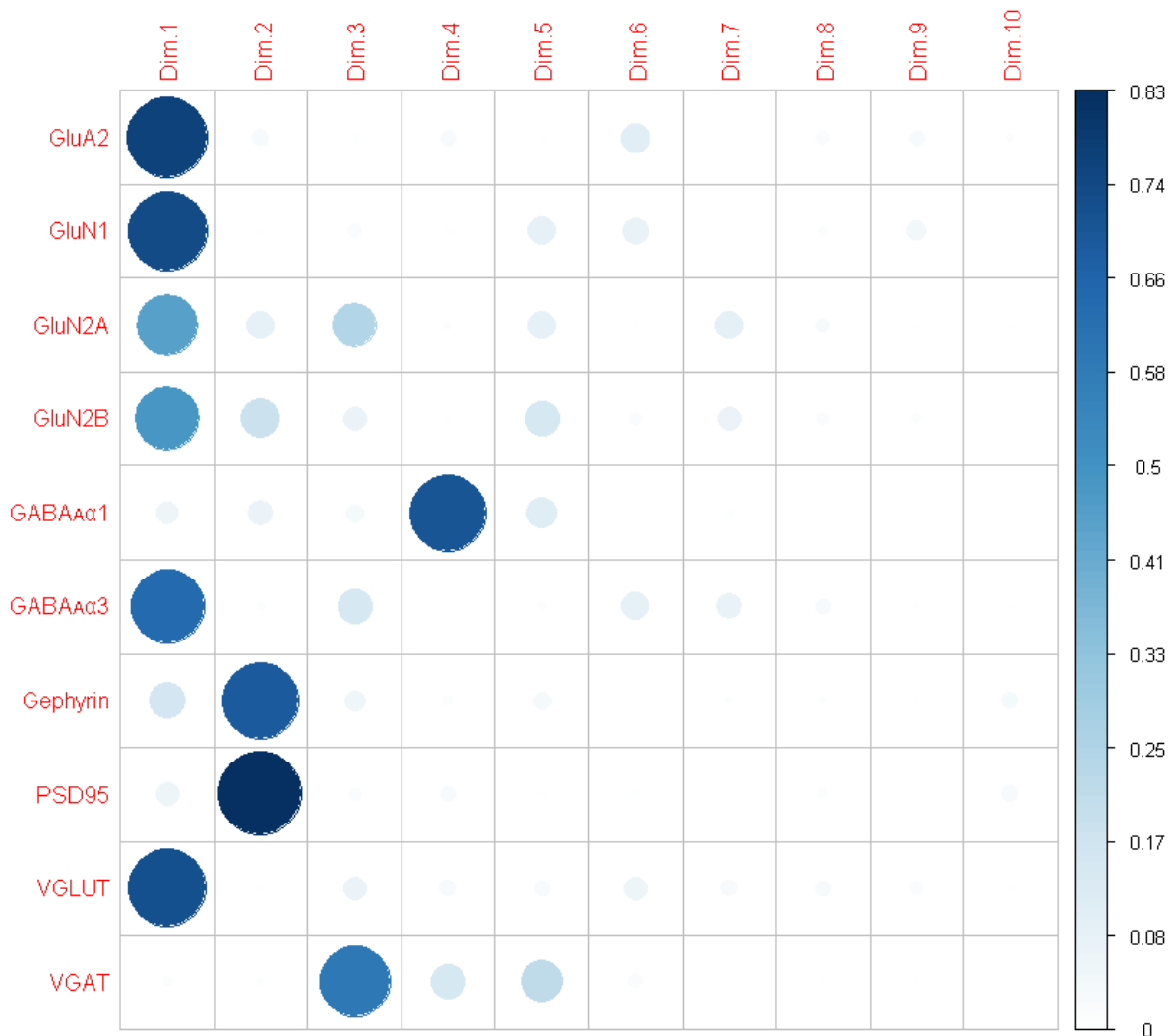
170

171

```

172 ##          Dim.6          Dim.7          Dim.8          Dim.9          Dim
173 .10
174 ## GluA2          0.1032742691 0.0009677872 0.0222832077 0.0291530529 8.486683e
175 -03
176 ## GluN1          0.0814380106 0.0001801498 0.0111629190 0.0434232666 1.331923e
177 -03
178 ## GluN2A          0.0005045302 0.0940226829 0.0251046288 0.0014452982 2.006101e
179 -03
180 ## GluN2B          0.0195524884 0.0680180183 0.0176440849 0.0131112895 1.677633e
181 -03
182 ## GABA<U+1D00>a1 0.0026940292 0.0061893560 0.0003928626 0.0003983653 1.98077
183 9e-04
184 ## GABA<U+1D00>a3 0.0885853506 0.0767075293 0.0261080521 0.0029823856 2.64503
185 0e-03
186 ## Gephyrin          0.0032132792 0.0092461556 0.0094056857 0.0010813466 3.388981e
187 -02
188 ## PSD95          0.0077746916 0.0002620205 0.0159275404 0.0016589820 3.205513e
189 -02
190 ## VGLUT          0.0632995867 0.0317297777 0.0304004013 0.0208645253 5.248660e
191 -03
192 ## VGAT          0.0243190008 0.0003393269 0.0020497922 0.0004751795 6.018825e
193 -07
194 corrplot(pca.scaled$var$cos2,is.corr=FALSE)

```



Construct a correlation plot of coord data stored in the object, "pca.scaled".

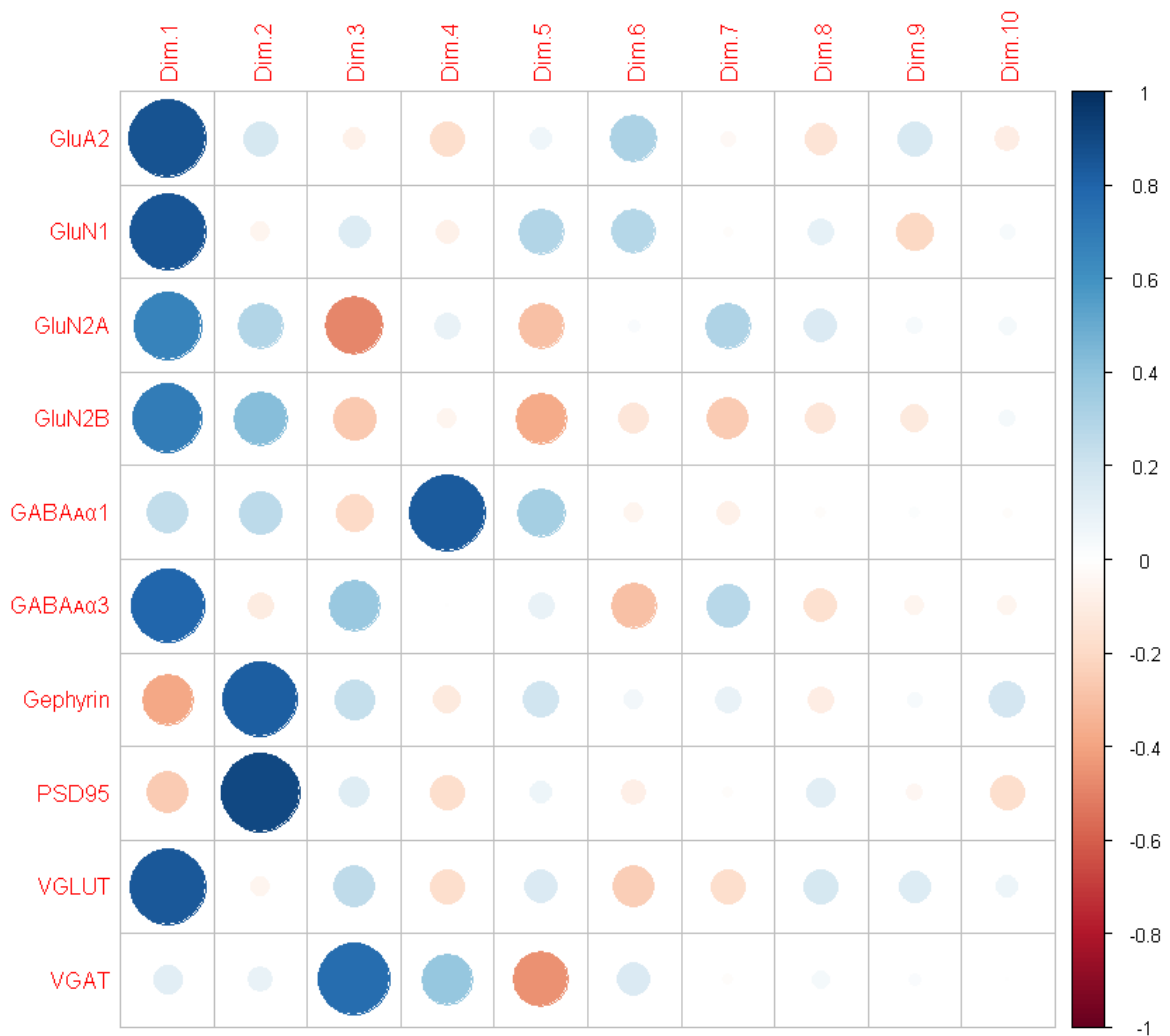
```
head(pca.scaled$var$coord)
```

	Dim.1	Dim.2	Dim.3	Dim.4	Dim.5
GluA2	0.8687981	0.17485540	-0.07006166	-0.1731765424	0.06853255
GluN1	0.8560169	-0.05415358	0.14813922	-0.0799689248	0.29533907
GluN2A	0.6657005	0.29875052	-0.48665319	0.0962408982	-0.29500083
GluN2B	0.6908859	0.42036857	-0.26178061	-0.0540660776	-0.37826275
GABA<U+1D00>a1	0.2457715	0.26832729	-0.19878676	0.8330584803	0.33371053
GABA<U+1D00>a3	0.7938044	-0.10246062	0.37736781	-0.0007871216	0.09753352
	Dim.6	Dim.7	Dim.8	Dim.9	Dim.10
GluA2	0.31926955	-0.03090661	-0.14830312	0.16963031	-0.09152304
GluN1	0.28351433	-0.01333455	0.10496640	-0.20702495	0.03625776
GluN2A	0.02231542	0.30463357	0.15741218	0.03776941	0.04449773

```

210 ## GluN2B      -0.13891926 -0.25910358 -0.13196568 -0.11375857  0.04069209
211 ## GABA<U+1D00>a1 -0.05156590 -0.07815993 -0.01969163  0.01982906 -0.01398233
212 ## GABA<U+1D00>a3 -0.29569392  0.27515691 -0.16052721 -0.05425544 -0.05109481
213 corrpplot(pca.scaled$var$coord, is.corr= TRUE)

```



214

215 **Calculate how many components are required to maintain 80% of the**
 216 **total variance. Store the output in the object, “cum.var”.**

```

217 pca.scaled$eig[,3]
218 ##      comp 1      comp 2      comp 3      comp 4      comp 5      comp 6      comp 7      c
219 omp 8
220 ## 40.94621  60.04963  72.57871  82.30949  89.55069  93.49725  96.37387  97.
221 97867
222 ##      comp 9      comp 10
223 ## 99.12460 100.00000

```

```

224 cum.var <- cum_var(pca.eig.3 = pca.scaled$eig[,3], # "pca.scaled$eig[,3]" is
225 contained within object "pca"
226                 thresh = 80) # Custom threshold value (in units percentage
227 )
228
229 cum.var
230 ## [1] 4

```

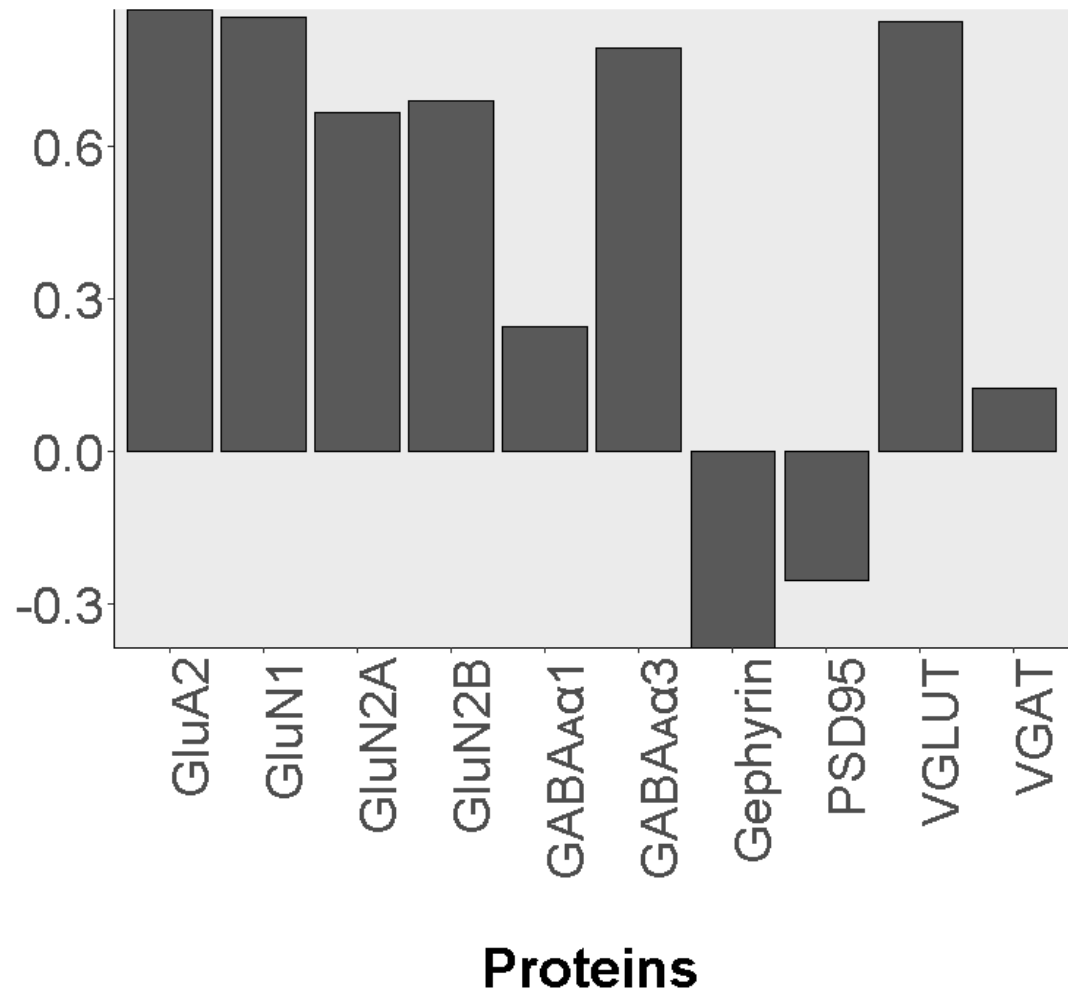
231 **Create PCA amplitude plots for each important component. Use**
 232 **amplitude plots to identify relevant plasticity features.**

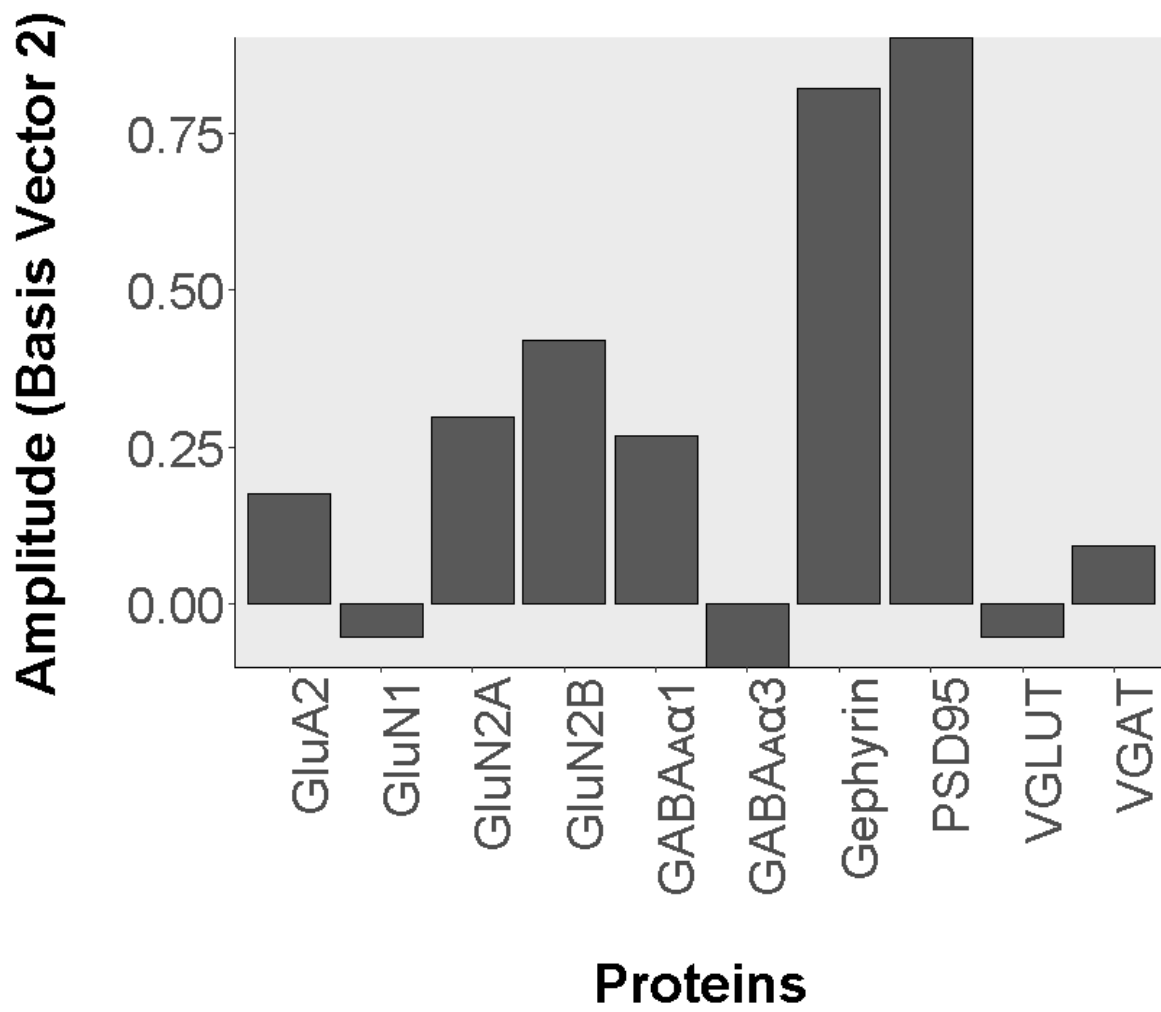
```

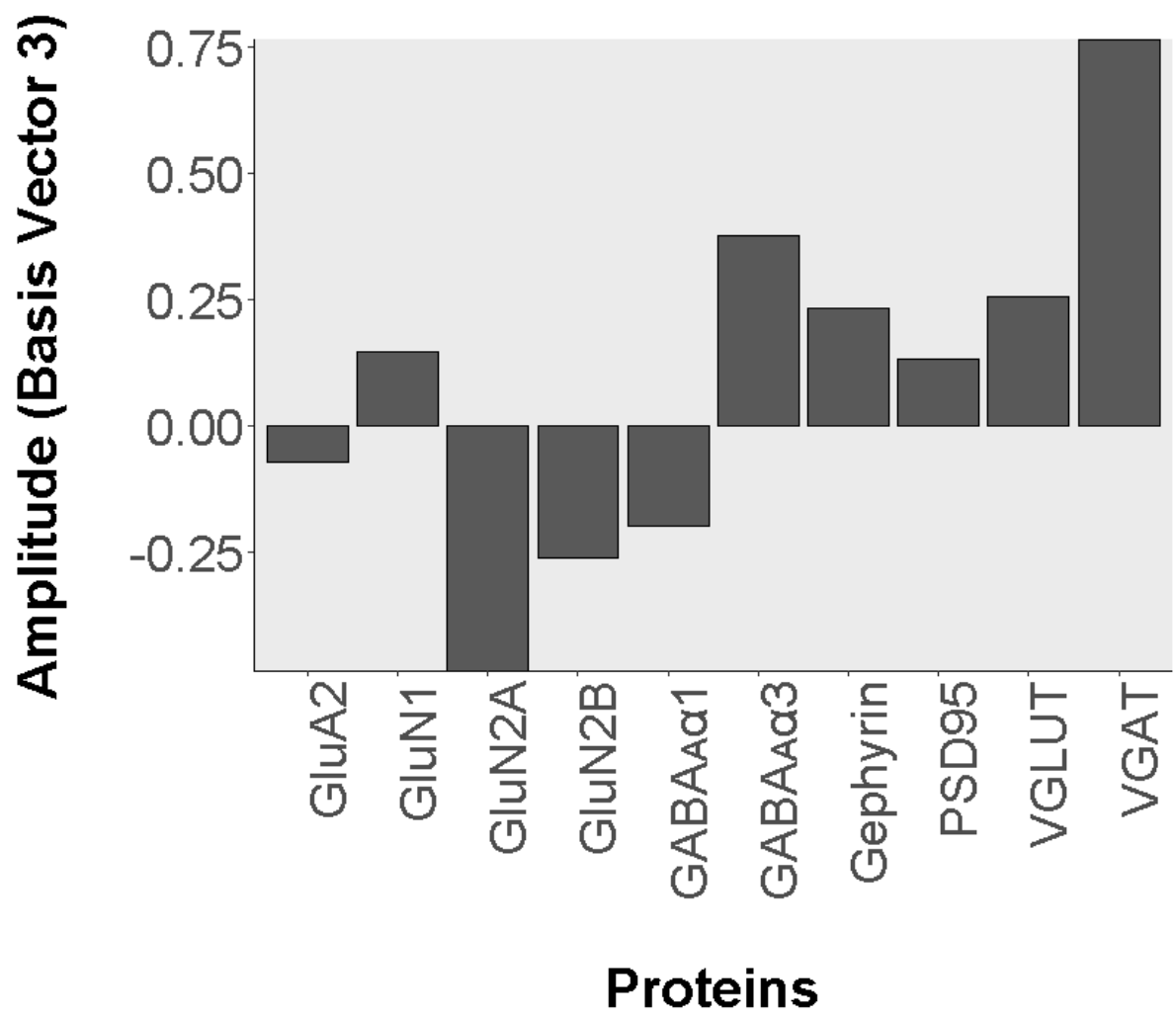
233 amplitude_plots(cum.var = cum.var, # Output of 'cum_var()' function
234                pca.var.coord = pca.scaled$var$coord) # "pca.scaled$var$coord
235 " is contained within object 'pca'

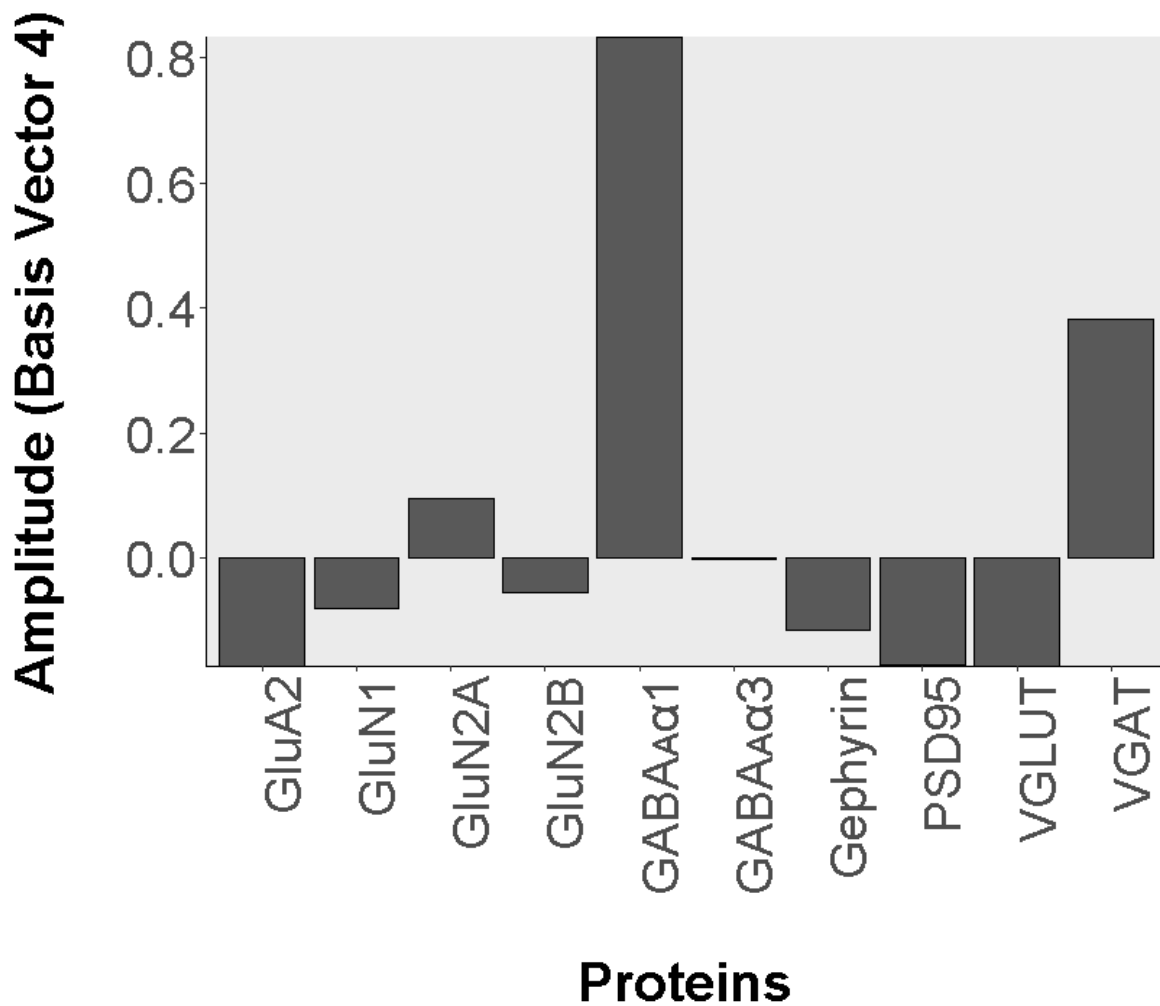
```

Amplitude (Basis Vector 1)









239

240 [Rat: Application of the Heuristics to Identify Candidate Plasticity](#)
 241 [Features for Rat V1](#)

242 **Create dataframe, “NewFeatures”, to store plasticity features.**

```
243 NewFeatures <- data.frame(matrix(nrow = nrow(my.data),
244                                ncol = 0))
```

245 **Calculate plasticity features.**

```
246 NewFeatures$'Receptors Sum' <- rowSums(my.data[,1:6])
247
248     ave.rec <- NewFeatures$'Receptors Sum' / ncol(my.data[,1:6])
249
250 NewFeatures$'VGLUT' <- my.data$VGLUT
251
252 NewFeatures$'Scaffolding Sum' <- rowSums(my.data[,7:8])
```

```

253
254     ave.scaf <- NewFeatures$'Scaffolding Sum'/ncol(my.data[,7:8])
255
256 NewFeatures$'Receptors:Scaffolding' <- (ave.rec - ave.scaf)/
257     (ave.rec + ave.scaf)
258
259 NewFeatures$'VGAT' <- my.data$VGAT
260
261 NewFeatures$"GABA\u1D00\u03b13:GABA\u1D00\u03b11" <-
262     (my.data$'GABA\u1D00\u03b11' - my.data$'GABA\u1D00\u03b13')/
263     (my.data$'GABA\u1D00\u03b11' + my.data$'GABA\u1D00\u03b13')
264
265 NewFeatures$'GluN2A:GluN2B' <- (my.data$GluN2A - my.data$GluN2B)/
266     (my.data$GluN2A + my.data$GluN2B)
267
268 NewFeatures$'GABA\u1D00\u03b11' <- my.data$'GABA\u1D00\u03b11'
269
270 NewFeatures$'VGLUT:VGAT' <- (my.data$VGLUT - my.data$VGAT)/
271     (my.data$VGLUT + my.data$VGAT)

```

272 **Specify order of “NewFeature” columns in the object “NewFeatsCol”.**

```

273 NewFeatsCol <- c("VGAT", "VGLUT", 'Receptors Sum',
274     'Scaffolding Sum', "GABA\u1D00\u03b11",
275     "GABA\u1D00\u03b13:GABA\u1D00\u03b11" ,
276     "GluN2A:GluN2B",
277     'Receptors:Scaffolding',
278     "VGLUT:VGAT")

```

279 **Rat: Validating Candidate Plasticity Features**

280 **Reassign “pca.scaled\$ind\$coord” object to “PCA.scores”.**

```

281 PCA.scores <- pca.scaled$ind$coord

```

282 **Perform a Bonferroni-corrected, pairwise Pearson’s correlation**
283 **against the PCA scores and plasticity features. Store the results in the**
284 **object, “corr.scores.bf”.**

```

285 corr.scores.bf <- corr.test(PCA.scores[,1:cum.var],
286     NewFeatures[,NewFeatsCol],
287     use="pairwise",
288     method="pearson",
289     adjust="bonferroni")
290
291 ## Warning in abbreviate(colnames(r), minlength = minlength): abbreviate used
    with

```

```

292 ## non-ASCII chars
293
294 ## Warning in abbreviate(colnames(r), minlength = minlength): abbreviate used
295 with
296 ## non-ASCII chars
297
298 ## Warning in abbreviate(colnames(r), minlength = minlength): abbreviate used
299 with
300 ## non-ASCII chars

```

301 Store correlation coefficients in the object, “corr.scores.rval”.

```

302 corr.scores.rval <- corr.scores.bf$r
303
304 corr.scores.rval

```

	VGAT	VGLUT Receptors	Sum	Scaffolding	Sum	GABA<U+1D00>a
1						
## Dim.1	0.12556031	0.85034747	0.8912394	-0.3283217		0.2473832
## Dim.2	0.09308728	-0.05314437	0.2785196	0.9026116		0.2700868
## Dim.3	0.77019116	0.25750384	-0.1797255	0.1862151		-0.2000903
## Dim.4	0.38393459	-0.17496353	0.2982607	-0.1506626		0.8385212
##	GABA<U+1D00>a3:GABA<U+1D00>a1 GluN2A:GluN2B Receptors:Scaffolding					
g						
## Dim.1		-0.2040768	0.11725395			0.6212029
## Dim.2		0.3033732	-0.03396026			-0.6809633
## Dim.3		-0.4347533	-0.37020779			-0.2181736
## Dim.4		0.6250398	0.17726754			0.1624328
##	VGLUT:VGAT					
## Dim.1	0.47533836					
## Dim.2	-0.08083291					
## Dim.3	-0.50327931					
## Dim.4	-0.35495764					

322 Store adjusted p-values of correlation coefficients in the object, “corr.scores.bfpval”.

```

324 corr.scores.bfpval <- corr.scores.bf$p
325
326 corr.scores.bfpval

```

	VGAT	VGLUT Receptors	Sum	Scaffolding	Sum	GABA<U+1D00>a1
## Dim.1	1.000000e+00	4.733524e-21	6.508887e-26	1.279815e-01		1.000000e+00
## Dim.2	1.000000e+00	1.000000e+00	5.103404e-01	1.279791e-27		6.307897e-01
## Dim.3	9.569439e-15	8.555534e-01	1.000000e+00	1.000000e+00		1.000000e+00

```

335 ## Dim.4 2.040493e-02 1.000000e+00 3.031827e-01 1.000000e+00 6.535877e-
336 20
337 ##          GABA<U+1D00>a3:GABA<U+1D00>a1 GluN2A:GluN2B Receptors:Scaffolding
338 g
339 ## Dim.1          1.000000e+00          1.000000          5.978854e-08
340 ## Dim.2          2.634037e-01          1.000000          3.406416e-10
341 ## Dim.3          2.802401e-03          0.0331084          1.000000e+00
342 ## Dim.4          4.434046e-08          1.000000          1.000000e+00
343 ##          VGLUT:VGAT
344 ## Dim.1 0.0004503966
345 ## Dim.2 1.0000000000
346 ## Dim.3 0.0001110059
347 ## Dim.4 0.0553288372

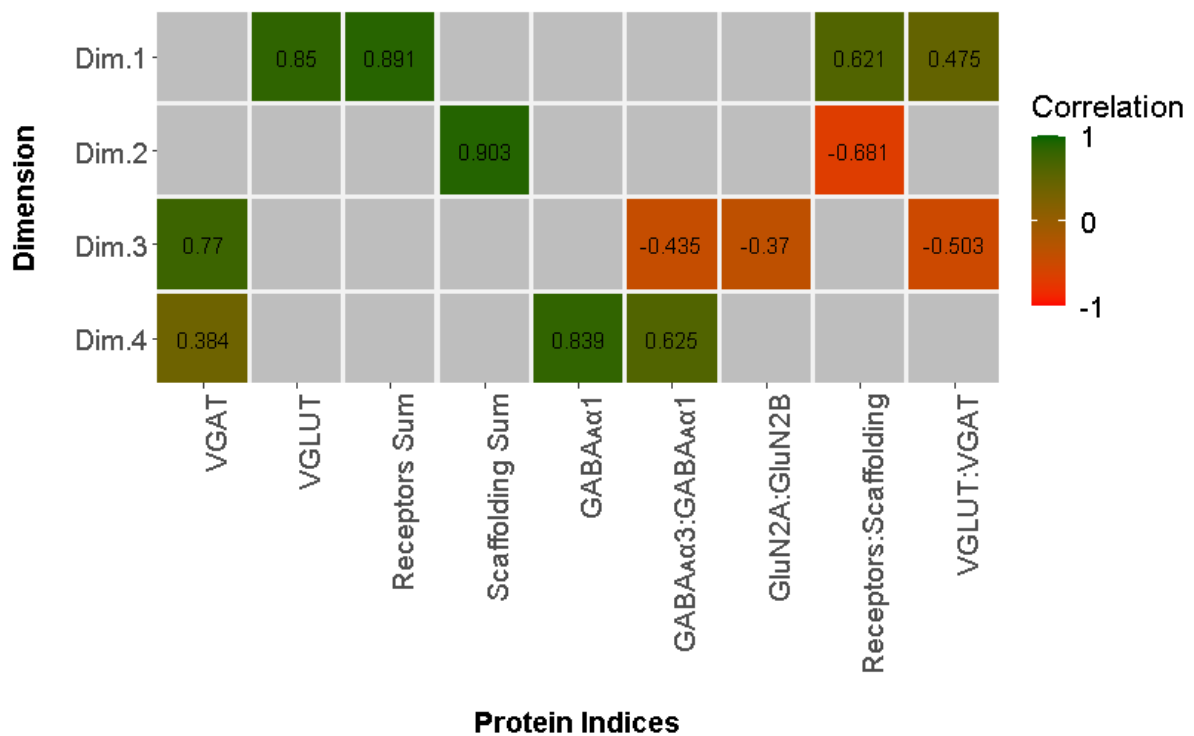
```

348 Construct plasticity feature matrix.

```

349 feature_matrix(
350   corr.scores.pval = corr.scores.bfpval, # Matrix of adjusted p-values
351   corr.scores.rval = corr.scores.rval, # Matrix of correlation coefficients
352   thresh = 0.05) # Significance threshold (acceptable values range from 0 - 1
353 )
354 ## Warning: Removed 23 rows containing missing values (geom_text).

```



355

356 Rat: Using Plasticity Features to Construct a Plasticity
357 Phenotype – Data Processing

358 **Reorder the protein columns of “NewFeatures” according to the order**
359 **of column names stored in “NewFeatsCol”. Assign this reordered data**
360 **frame to itself.**

```
361 NewFeatures <- NewFeatures[,NewFeatsCol]
```

362 **Assign row names of “my.data” to row names of “NewFeatures”. This**
363 **is done to allow for the future creation of the bootstrap phenotype.**

```
364 rownames(NewFeatures) <- rownames(my.data)
```

365 **Reassign the “NewFeatures” row names as their own “ids” column.**
366 **This is done to allow for the subsequent merging of “NewFeatures”**
367 **with “syn.prots”.**

```
368 NewFeatures$ids <- rownames(NewFeatures)
```

369 **Store “my.data” row names as their own column in “syn.prots”. This**
370 **is done to allow for the subsequent merging of “NewFeatures” with**
371 **“syn.prots”.**

```
372 syn.prots$ids <- row.names(my.data)
```

373 **Merge “syn.prots” and “NewFeatures” data frame by their shared**
374 **“ids” column, and store the output in the object, “merged.data”.**

```
375 merged.data <- merge(NewFeatures,  
376                      syn.prots[,c("Case_Code",  
377                                   "Condition",  
378                                   "Region",  
379                                   "Hemisphere",  
380                                   "Run",  
381                                   "ids")],  
382                      by.x = "ids",  
383                      by.y = "ids")
```

384 Calculate the median of each plasticity feature in
385 “merged.features[,NewFeatsCol]” across all levels of
386 “list(merged.data\$Condition)”. Store the output in the object,
387 “meds”.

```
388 meds <- aggregate(merged.data[,NewFeatsCol],  
389                   list(merged.data$Condition),  
390                   median)
```

391 Reorder the rows of the "meds" data frame according to the specified
392 order of it "Group.1" column. Reassign the output into the object,
393 "meds".

```
394 meds <- meds[match(c("normal",  
395                     "1wk MD",  
396                     "flx + 1wk MD",  
397                     "flx"),  
398                  meds$Group.1),]
```

399 Assign contents of “Group.1” column to row names of object, “meds”.
400 rownames(meds) <- meds\$Group.1

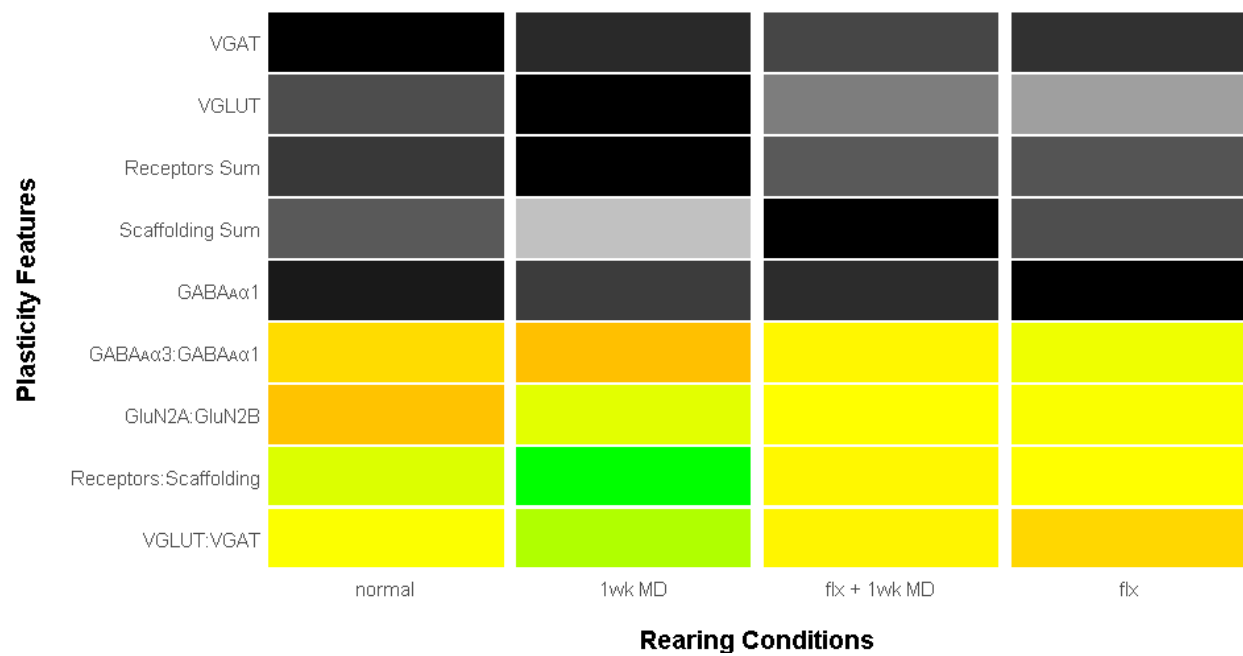
3.) Rat Phenotype

Rat: Using Plasticity Features to Construct a Plasticity Phenotype – Creating Phenotypes

Create a plasticity phenotype. Function assumes that the first few columns will be sums grouped together, followed by all indices grouped together.

```
plasticity_phenotype(  
    df_list = list(meds[, -1]), # Median values data frame  
    first_index_column = 6, # Index number of first index column in "meds" (in  
    dexes begin at 0)  
    group_label = "\nRearing Conditions", # X-axis Label  
    translation = 'absolute') # Desired colour-scale
```

```
##              normal 1wk MD flx + 1wk MD      flx  
## VGAT           #000000 #292929          #464646 #313131  
## VGLUT          #4D4D4D #000000          #7D7D7D #9F9F9F  
## Receptors Sum  #383838 #000000          #595959 #545454  
## Scaffolding Sum #595959 #C1C1C1          #000000 #4E4E4E  
## GABA<U+1D00>a1  #191919 #3C3C3C          #2C2C2C #000000  
## GABA<U+1D00>a3:GABA<U+1D00>a1 #FFDC00 #FFC000          #FFF700 #EFFF00  
## GluN2A:GluN2B  #FFC300 #E3FF00          #FFFF00 #FAFF00  
## Receptors:Scaffolding #DCFF00 #00FF00          #FFF800 #FFFF00  
## VGLUT:VGAT     #FCFF00 #B0FF00          #FFF500 #FFD700  
## [[1]]
```



424

425 **Assign the rat phenotype's colour-code to the object, "rat.cols".**

426 `rat.cols <- phen.cols[[1]]`

4.) Create Rat Boxplots

Rat: Using Plasticity Features to Construct a Plasticity Phenotype – Creating Boxplots

Append contents of “NewFeatsCol” with an “\n” and store in “NewFeatsCol2”. This was done to avoid overwriting “NewFeatsCol”.

```
NewFeatsCol2 <- paste0(NewFeatsCol, "\n")
```

Store “merged.data” into “merged.data2”. This was done to avoid overwriting “merged.data”.

```
merged.data2 <- merged.data
```

Assign contents of “NewFeatsCol2” as the plasticity feature column headers of “merged.data2”.

```
colnames(merged.data2)[2:10] <- NewFeatsCol2
```

Order the levels of “merged.data2\$Condition” and reassign the output into “merged.data2\$Condition”.

```
merged.data2$Condition <- factor(merged.data2$Condition,
                                c("normal",
                                  "1wk MD",
                                  "flx + 1wk MD",
                                  "flx"),
                                ordered = T)
```

Create multiple individual boxplots based on the rat fluoxetine plasticity phenotype.

```
head(merged.data2[,c("Condition", NewFeatsCol2)])
```

```
## Condition    VGAT\n    VGLUT\n Receptors Sum\n Scaffolding Sum\n## 1      flx 0.6877782 0.6543510      4.868469      1.611637\n## 2      flx 1.0696206 0.8939942      6.091951      1.880370\n## 3      flx 1.7799586 0.7829019      4.992009      1.868406\n## 4      flx 0.9694615 0.8957831      6.055437      4.962173\n## 5      flx 1.0398267 0.9726280      5.514919      2.049094\n## 6      flx 2.7974660 0.9304105      7.646494      2.363624\n## GABA<U+1D00>a1\n GABA<U+1D00>a3:GABA<U+1D00>a1\n GluN2A:GluN2B\n## 1      0.6081925      -0.27659380      0.18481041
```

```

459 ## 2      1.2728010      0.04431119      0.13280069
460 ## 3      0.6445332      -0.28039590      -0.07655649
461 ## 4      1.0276800      -0.01807704      0.07282728
462 ## 5      0.9234916      -0.02424368      -0.03311775
463 ## 6      2.3551698      0.30931242      -0.03443097
464 ## Receptors:Scaffolding\n VGLUT:VGAT\n
465 ## 1      0.003458318  -0.02490615
466 ## 2      0.038424886  -0.08944036
467 ## 3      -0.057864918  -0.38904056
468 ## 4      -0.421693303  -0.03950063
469 ## 5      -0.054223255  -0.03339143
470 ## 6      0.037701603  -0.50083619

471 phenotype_boxplots(feature_df = merged.data2[,c("Condition",NewFeatsCol2)], #
472 Boxplot data frame
473 phenotype_cols = rat.cols, # Phenotype colour-code data fr
474 ame
475 max_sum = c(1.5,
476             max(merged.data2$'VGLUT\n'),
477             max(merged.data2$'Receptors Sum\n'),
478             max(merged.data2$'Scaffolding Sum\n'),
479             2), # Maximum Y-axis values for feature sums'
480 boxplots
481 group_label = " ", # X-axis Label
482 first_index_column = 6, # Index number of first index colu
483 mn in "merged.data2[,c("Condition",NewFeatsCol2)]" (indexes begin at 0)
484 point_size = 1.5, # Desired size of geom_jitter points
485 point_alpha = 1, # Desired transparency of geom_jitter poi
486 nts
487 aspect_ratio = 5/7, # Desired aspect ratio of each boxplot
488 text_size = 8) # Desired X- & Y-axis text size for each bo
489 xplot

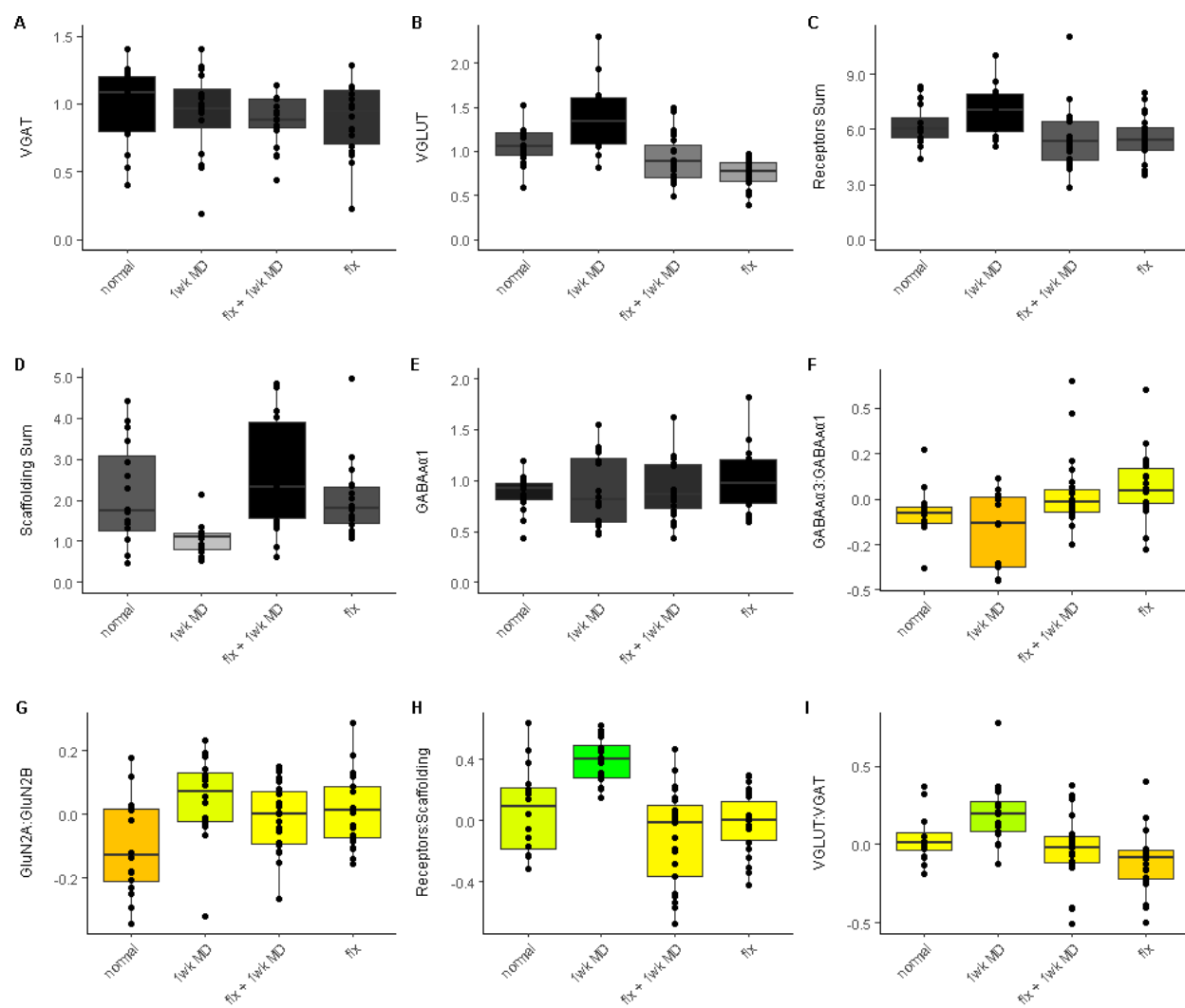
```

490 **Create a 3 x 3 matrix displaying all colour-coded boxplots on a single**
491 **figure.**

```

492 ggarrange(plotlist = plot_list,
493           nrow = 3,
494           ncol = 3,
495           labels = LETTERS[1:9],
496           font.label = list(size = 10),
497           vjust = 1)

```



5.) Rat Plasticity Phenotype Bootstrap Analysis

500 **Create a phenotype displaying which features in which experimental**
 501 **groups are significantly higher than, lower than, or statistically non-**
 502 **significant in comparison to the “reference_group”.**

503 `head(NewFeatures[,NewFeatsCol])`

```
504 ##
505 ## 231 flx VC Cont 1 0.6877782 0.6543510 4.868469 1.611637
506 ## 232 flx VC Cont 1 0.9694615 0.8957831 6.055437 4.962173
507 ## 233 flx VC Cont 1 0.9891639 0.8531914 8.032260 1.845288
508 ## 234 flx VC Cont 1 0.8166134 0.6892650 5.635888 3.067179
509 ## 235 flx VC Cont 1 0.6479182 0.3932702 3.808882 2.382254
510 ## 236 flx VC Cont 1 1.2881240 0.9291042 7.058869 1.538176
511 ##
512 ## 231 flx VC Cont 1 0.6081925 -0.27659380
513 ## 232 flx VC Cont 1 1.0276800 -0.01807704
514 ## 233 flx VC Cont 1 1.8131427 0.21992021
515 ## 234 flx VC Cont 1 1.1133509 0.12368031
516 ## 235 flx VC Cont 1 0.8327478 0.18130631
517 ## 236 flx VC Cont 1 1.2145710 0.08319483
518 ##
519 ## 231 flx VC Cont 1 0.18481041 0.003458318 -0.02490615
520 ## 232 flx VC Cont 1 0.07282728 -0.421693303 -0.03950063
521 ## 233 flx VC Cont 1 0.28771834 0.183989663 -0.07380365
522 ## 234 flx VC Cont 1 0.12088717 -0.240314546 -0.08456751
523 ## 235 flx VC Cont 1 -0.04336542 -0.304672207 -0.24457434
524 ## 236 flx VC Cont 1 -0.13921701 0.209394110 -0.16192279
```

```
525 bootstrap_phenotype(features_df_row = NewFeatures[,NewFeatsCol], # Data frame  

526 for bootstrap analysis  

527 condition_list = c(' normal ',  

528 ' 1wk MD ',  

529 ' flx + 1wk MD ',  

530 ' flx '), # List of experimental con  

531 ditions as they appear in row names of "features_df_row"  

532 reference_group = ' normal ', # Name of reference group  

533 as it appears in the row names of "features_df_row"  

534 group_label = "\nRearing Conditions") # X-axis Label
```

Plasticity Features

VGAT				
VGLUT				
Receptors Sum				
Scaffolding Sum				
GABA α 1				
GABA α 3:GABA α 1				
GluN2A:GluN2B				
Receptors:Scaffolding				
VGLUT:VGAT				
	normal	1wk MD	flx + 1wk MD	flx

Rearing Conditions

535

Rat_CorrHeatmap_Figure14

<u>Table of Contents</u>		
Section Number	Title	Page Number
1	Load Rat Correlation Data	99
2	Process the Rat Correlation Data	103
3	Studying Fluoxetine-enhanced Plasticity in Adult Rat V1 Using a Plasticity Phenotype – Create Heatmap	106

1.) Load Rat Correlation Data

Install the package, "PlasticityPhenotypes".

```
devtools::install_github("visualneurosciencelab/PlasticityPhenotypes")
```

Load the package, "PlasticityPhenotypes".

```
library(PlasticityPhenotypes)
```

```
## Loading required package: tidyverse
```

```
## -- Attaching packages ----- tidyverse 1
## .3.0 --
```

```
## v ggplot2 3.3.0      v purrr  0.3.3
```

```
## v tibble  2.1.3      v dplyr  0.8.5
```

```
## v tidyr   1.0.2      v stringr 1.4.0
```

```
## v readr   1.3.1      v forcats 0.5.0
```

```
## -- Conflicts ----- tidyverse_conflic
ts() --
```

```
## x dplyr::filter() masks stats::filter()
```

```
## x dplyr::lag()     masks stats::lag()
```

```
## Loading required package: FactoMineR
```

```
## Loading required package: factoextra
```

```
## Welcome! Want to learn more? See two factoextra-related books at https://g
oo.gl/ve3WBa
```

```
## Loading required package: data.table
```

```
##
```

```
## Attaching package: 'data.table'
```

```
## The following objects are masked from 'package:dplyr':
```

```
##
```

```
##      between, first, last
```

```
## The following object is masked from 'package:purrr':
```

```
##
```

```
##      transpose
```

```
## Loading required package: ggpubr
```

```
## Loading required package: magrittr
```

```
##
```

```
## Attaching package: 'magrittr'
```

```
38 ## The following object is masked from 'package:purrr':
39 ##
40 ##   set_names
41 ## The following object is masked from 'package:tidyr':
42 ##
43 ##   extract
44 ## Loading required package: corrplot
45 ## corrplot 0.84 loaded
46 ## Loading required package: psych
47 ##
48 ## Attaching package: 'psych'
49 ## The following objects are masked from 'package:ggplot2':
50 ##
51 ##   %+%, alpha
52 ## Loading required package: httr
53 library(gplots)
54 ##
55 ## Attaching package: 'gplots'
56 ## The following object is masked from 'package:stats':
57 ##
58 ##   lowess
59 library(RColorBrewer)
60 library(Hmisc)
61 ## Loading required package: lattice
62 ## Loading required package: survival
63 ## Loading required package: Formula
64 ##
65 ## Attaching package: 'Hmisc'
66 ## The following object is masked from 'package:psych':
67 ##
68 ##   describe
69 ## The following objects are masked from 'package:dplyr':
70 ##
71 ##   src, summarize
```

```

72 ## The following objects are masked from 'package:base':
73 ##
74 ##     format.pval, units

```

75 Store the file path in a unique object.

```

76 raw.rat.corr <- "https://osf.io/z2asv/?action=download"

```

77 Import the necessary CSV from OSF. The “raw.data” object consists of 78 10 columns - 1 attributes column (“ids”) and 7 plasticity feature 79 columns.

```

80 filename <- 'rat_correlation.csv'
81 GET(raw.rat.corr, write_disk(filename, overwrite = TRUE))

82 ## Response [https://files.ca-1.osf.io/v1/resources/8a3kx/providers/osfstorag
83 e/5ecd3179c7568601d42d2039?action=download&direct&version=1]
84 ##   Date: 2020-06-02 20:17
85 ##   Status: 200
86 ##   Content-Type: text/csv
87 ##   Size: 13.2 kB
88 ## <ON DISK> C:\Users\dezia\Dropbox (Kathy Murphy)\JB-ProteinAnalysisWorkflo
89 w\JB ProteinAnalysisWorkflow Markdowns\RMD files\Rat Analysis\rat_correlation
90 .csv

91 raw.data <- read.csv(filename)
92 head(raw.data)

```

```

93 ##           ids      VGAT      VGLUT Receptors.Sum
94 ## 1 231 fluoxetine VC Cont 1 0.6877782 0.6543510      4.868469
95 ## 2 232 fluoxetine VC Cont 1 0.9694615 0.8957831      6.055437
96 ## 3 233 fluoxetine VC Cont 1 0.9891639 0.8531914      8.032260
97 ## 4 234 fluoxetine VC Cont 1 0.8166134 0.6892650      5.635888
98 ## 5 235 fluoxetine VC Cont 1 0.6479182 0.3932702      3.808882
99 ## 6 236 fluoxetine VC Cont 1 1.2881240 0.9291042      7.058869
100 ## Scaffolding.Sum GABA.U.1D00.a1 GABA.U.1D00.a1.GABA.U.1D00.a3 GluN2A.GluN
101 2B
102 ## 1      1.611637      0.6081925      -0.27659380      0.184810
103 41
104 ## 2      4.962173      1.0276800      -0.01807704      0.072827
105 28
106 ## 3      1.845288      1.8131427      0.21992021      0.287718
107 33
108 ## 4      3.067179      1.1133509      0.12368031      0.120887
109 17
110 ## 5      2.382254      0.8327478      0.18130631     -0.043365
111 42
112 ## 6      1.538176      1.2145710      0.08319483     -0.139217
113 01

```

114	##	Receptors.Scaffolding	VGLUT.VGAT				ids.1
115	## 1	0.003458318	-0.02490615	231	fluoxetine	VC	Cont 1
116	## 2	-0.421693303	-0.03950063	232	fluoxetine	VC	Cont 1
117	## 3	0.183989663	-0.07380365	233	fluoxetine	VC	Cont 1
118	## 4	-0.240314546	-0.08456751	234	fluoxetine	VC	Cont 1
119	## 5	-0.304672207	-0.24457434	235	fluoxetine	VC	Cont 1
120	## 6	0.209394110	-0.16192279	236	fluoxetine	VC	Cont 1

2.) Process the Rat Correlation Data

Split the character strings in “raw.data\$ids” into 5 different columns, and assign them in the object, “ids”. Rename the “ids” column headers.

```
ids <- as.data.frame(str_split_fixed(raw.data$ids,
                                     " ",
                                     5))

colnames(ids) <- c('id', 'condition', 'location', 'hemisphere', 'run')

head(ids)

##    id condition location hemisphere run
## 1 231 fluoxetine      VC          Cont    1
## 2 232 fluoxetine      VC          Cont    1
## 3 233 fluoxetine      VC          Cont    1
## 4 234 fluoxetine      VC          Cont    1
## 5 235 fluoxetine      VC          Cont    1
## 6 236 fluoxetine      VC          Cont    1
```

Rename the factor levels of “ids\$condition” to be consistent with (Beshara et al., 2015) and reassign to itself.

```
ids$condition <- plyr::mapvalues(ids$condition,
                                 unique(ids$condition),
                                 c("flx",
                                   "flx + 1wk MD",
                                   "normal",
                                   "1wk MD"))
```

Split the “ids” object into 5 columns (“V1 - V5”). Concatenate the “id” and “condition” columns and separate them using a “-”. Assign these newly created character strings in the novel column, “ids\$rat_id”, and store full output in the object, “ids2”.

```
ids.2 <- unite(ids, rat_id,
               c(id,
                 condition),
               sep = "-",
               remove = FALSE)
```

156 **Select both the “rat_id” and “runs” column of “ids.2”, and store in the**
157 **object, “ids.3”.**

```
158 ids.3 <- ids.2[,c('rat_id','run')]
```

159 **Bind plasticity feature columns of “raw.data” and “ids.3\$run” column,**
160 **and store in the object, “raw.data.2”.**

```
161 raw.data.2 <- cbind(raw.data[, -ncol(raw.data)],  
162                     ids.3$run)
```

163 **Rename the last column of “raw.data.2” to “run”.**

```
164 colnames(raw.data.2)[ncol(raw.data.2)] <- "run"
```

165 **Calculate the median of each feature across multiple runs, and store**
166 **in “med.df”.**

```
167 all.df <- group_by(raw.data.2[, -ncol(raw.data.2)],  
168                   ids.3$rat_id)  
169  
170  
171 med.df <- summarise_at(all.df,  
172                       vars(colnames(raw.data.2)[2:10]),  
173                       median)
```

174 **Rename the first column of “med.df” to “rat_id”. Transform**
175 **“med.df\$rat_id” from a column to individual row names. Store the**
176 **output into the object, “med.df.2”.**

```
177 colnames(med.df)[1] <- 'rat_id'  
178  
179 med.df.2 <- column_to_rownames(med.df,  
180                               var = 'rat_id')
```

181 **Transpose the “med.df.2”, and store in “t.med.df”.**

```
182 t.med.df <- as.matrix(t(med.df.2))
```

183 **Create empty data frame to store the “ids”, “treatments”, and**
184 **“conditions” columns of “med.df.2” and store in “id.labels”.**

```
185 id.labels <- as.data.frame(matrix(nrow = nrow(med.df.2), ncol = 2))
```

186 **Rename columns of “id.labels”.**

```
187 colnames(id.labels) <- c("ids",  
188                          "groups")
```

189 **Assign rownames of “med.df.2” to ‘ids’ column of ‘ids.labels’.**

```
190 id.labels$ids <- rownames(med.df.2)
```

191 **Assign experimental conditions of each animal within the row names
192 of “med.df.2” to the “groups” column of “id.labels”.**

```
193 id.labels$groups <- as.data.frame(  
194   str_split_fixed(rownames(med.df.2), "-", 4))[, 2]
```

3.) Studying Fluoxetine-enhanced Plasticity in Adult Rat V1 Using a Plasticity Phenotype – Create Heatmap

Assign colors to each animal condition of “ids.labels” (flx = “#008000”/green, flx + 1wk MD = “#FF0000”/red, normal = “#000000”/black, 1wk MD = “#D3D3D3”/grey).

```
id.labels$colors <- plyr::mapvalues(id.labels$groups,  
                                   unique(id.labels$groups),  
                                   c("#008000",  
                                     "#FF0000",  
                                     "#000000",  
                                     "#D3D3D3"))
```

Perform a Pearson’s correlation against all animals to each other, and store in ‘data.correlation’.

```
data.correlation <- rcorr(as.matrix(t.med.df),  
                          type = "pearson")
```

Save the correlations coefficients as a matrix in “CorMat”.

```
CorMat <- as.matrix(data.correlation$r)
```

Convert the correlation matrix into a distance matrix (stored in “distance.row”), and perform hierarchical clustering upon this distance matrix. Store this hierarchical clustered object into “cluster.row”.

```
distance.row <- dist(as.matrix(CorMat),  
                    method = "euclidean")  
  
cluster.row <- hclust(distance.row,  
                     method = "ward.D2")
```

Transform “cluster.row” into a dendrogram which is stored in “dd”.

```
dd <- as.dendrogram(cluster.row)
```

224 **Create a color palette for the heatmap.**

```
225 nb.cols <- 25
226
227 mycolors <- colorRampPalette(brewer.pal(8, "YlOrRd"))(nb.cols)
```

228 **Sets the heatmap's margins.**

```
229 par(oma=c(rep(7,4)))
```

230 **Create the heatmap.**

```
231 heatmap.2(CorMat,
232           cexRow=1,
233           Colv = dd,
234           Rowv = dd,
235           cexCol=1,
236           breaks = seq(0.9,1, length.out = 26),
237           trace="none",
238           col = mycolors,
239           RowSideColors =
240             as.character(id.labels$colors),
241           ColSideColors =
242             as.character(id.labels$colors),
243           symm=T,
244           colsep=c(2,11,16),
245           rowsep=c(28 - c(2,11,16)),
246           sepcolor = 'black')
```

