Network Analysis

Shara He, Adanya Johnson

6/7/2021

Preparing Data

```
# Read in data
load("~/Desktop/School/GatesLab/NetworkAnalysis/FullMatrices.Rdata")

# Preview all data
# View(outReg)
```

Extract Submatrices for each Person

```
submatrix = list()
for (x in 1:length(outReg)) {
  submatrix = append(submatrix, list(outReg[[x]][["regression_matrix"]][1:26,1:26]))
}
# Convert all submatrices into double type
feature_names = c("energetic", "enthusiastic", "content", "irritable", "restless", "worried", "guilty",
subdf = list()
subdf_pos = list()
for (x in 1:length(outReg)) {
  subdf[[x]] = matrix(unlist(submatrix[x]), ncol = 26, byrow = FALSE, dimnames = list(feature_names, fea
  subdf_pos[[x]] = pmax(subdf[[x]], 0)
}
```

Extract Directed Graphs for each Person

```
dir_graph = list()
dir_graph_pos = list()
for(x in 1:length(outReg) ){
  dir_graph[[x]] = graph.adjacency(subdf[[x]], mode = "directed", weighted = TRUE)
  dir_graph_pos[[x]] = graph.adjacency(subdf_pos[[x]], mode = "directed", weighted = TRUE)
}
dir_graph[[19]]
```

```
## IGRAPH 7bca4ab DNW- 26 114 --
## + attr: name (v/c), weight (e/n)
## + edges from 7bca4ab (vertex names):
##  [1] energetic   ->enthusiastic energetic   ->down
##  [3] energetic   ->fatigue      energetic   ->tension
##  [5] energetic   ->ruminate     energetic   ->avoid_act
```

```
##  [7] enthusiastic->energetic    enthusiastic->positive
##  [9] enthusiastic->tension      enthusiastic->procrast
## [11] content     ->irritable    content     ->restless
## [13] content     ->worried      content     ->guilty
## [15] content     ->angry        content     ->down
## + ... omitted several edges
```

```
dir_graph_pos[[19]]
```

```
## IGRAPH 704e54e DNW- 26 73 --
## + attr: name (v/c), weight (e/n)
## + edges from 704e54e (vertex names):
##  [1] energetic   ->enthusiastic enthusiastic->energetic
##  [3] enthusiastic->positive     content     ->positive
##  [5] irritable   ->worried      irritable   ->angry
##  [7] irritable   ->threatened   irritable   ->ruminate
##  [9] restless    ->worried      restless    ->angry
## [11] restless    ->threatened   restless    ->avoid_people
## [13] worried     ->irritable    worried     ->restless
## [15] worried     ->guilty       worried     ->afraid
## + ... omitted several edges
```

Metrics for each Person

```
density = list()
overall_weight = list()
edge_weights = list()
global_efficiency = list()
variable_degrees = list()
variable_strengths = list()
betweenness = list()
cluster = list()
for(x in 1:length(outReg)) {
  density[[x]] = edge_density(dir_graph[[x]], loops = FALSE)
  overall_weight[[x]] = sum(strength(dir_graph[[x]]))
  edge_weights[[x]] = edge_attr(dir_graph_pos[[x]], "weight")

  global_efficiency[[x]] = efficiency(dir_graph_pos[[x]],
                                      type = c("global"),
                                      weights = edge_weights[[x]])
  variable_degrees[[x]] = degree(dir_graph[[x]])
  variable_strengths[[x]] = strength(dir_graph[[x]])

  betweenness[[x]] = estimate_betweenness(dir_graph_pos[[x]], cutoff=-1, weights =edge_weights[[x]])
  cluster[[x]] = cluster_walktrap(dir_graph_pos[[x]],
                                  weights = E(dir_graph_pos[[x]])$edge_weights,
                                  steps = 4)


}
print(paste0("Density of the first person's graph: ", round(density[[1]], 4)))
```

```
## [1] "Density of the first person's graph: 0.1077"
```

```r
print(paste0("Overall Weight of the first person's graph: ", round(overall_weight[[1]], 4)))
```

```
## [1] "Overall Weight of the first person's graph: 10.6119"
```

```r
print(paste0("Global Efficiency of the first person's graph: ", round(global_efficiency[[1]], 4)))
```

```
## [1] "Global Efficiency of the first person's graph: 12.5183"
```

```r
print("Degree of each variable in the first person's graph: ")
```

```
## [1] "Degree of each variable in the first person's graph: "
```

```r
print(variable_degrees[[1]])
```

```
##    energetic enthusiastic      content    irritable     restless      worried
##            4            4            3            8            5            9
##       guilty       afraid    anhedonia        angry     hopeless         down
##            1            9           12            7           11            8
##     positive      fatigue      tension  concentrate     accepted   threatened
##            7            0            7            3            1            3
##     ruminate    avoid_act     reassure     procrast        hours    difficult
##           13            3            1            2            2            8
##    unsatisfy avoid_people
##            1            8
```

```r
print("Strengths of each variable in the first person's graph: ")
```

```
## [1] "Strengths of each variable in the first person's graph: "
```

```r
print(variable_strengths[[1]])
```

```
##    energetic enthusiastic      content    irritable     restless      worried
##   0.76229416   0.96384765   0.47966325   0.83037658   0.57779129   1.20858363
##       guilty       afraid    anhedonia        angry     hopeless         down
##   0.02643344  -0.01065439   0.82426251   0.99845363   0.89364307   0.74160674
##     positive      fatigue      tension  concentrate     accepted   threatened
##   0.95930381   0.00000000   0.76755289   0.18317240   0.14779364   0.16860533
##     ruminate    avoid_act     reassure     procrast        hours    difficult
##   0.79906030   0.47042696   0.04843503  -0.06829517  -1.55552557  -0.56623424
##    unsatisfy avoid_people
##   0.20728066   0.75406944
```

```r
print("Betweenness Centrality of each variable in the first person's graph: ")
```

```
## [1] "Betweenness Centrality of each variable in the first person's graph: "
```

```r
print(betweenness[[1]])
```

```
##    energetic enthusiastic      content    irritable     restless      worried
##            0            5            0           40            1            0
##       guilty       afraid    anhedonia        angry     hopeless         down
##            0           10           49           26           85           37
##     positive      fatigue      tension  concentrate     accepted   threatened
##           46            0            2            0            0            0
##     ruminate    avoid_act     reassure     procrast        hours    difficult
##          111            0            0            0            0            0
##    unsatisfy avoid_people
##            0           51
```

```r
print("Clustering Walktrap of each variable in the first person's graph: ")
```

```
## [1] "Clustering Walktrap of each variable in the first person's graph: "
```

```r
print(cluster[[1]])
```

```
## IGRAPH clustering walktrap, groups: 11, mod: 0.27
## + groups:
##   $`1`
##   [1] "concentrate" "difficult"
##
##   $`2`
##   [1] "angry"        "down"         "ruminate"     "avoid_act"    "avoid_people"
##
##   $`3`
##   [1] "content"  "positive" "accepted"
##
##   $`4`
##   + ... omitted several groups/vertices
```

Computing Metric Summary Statistics

```r
density_mean = mean(unlist(density))
density_sd = sd(unlist(density))
density_range = range(unlist(density))

overall_weight_mean = mean(unlist(overall_weight))
overall_weight_sd = sd(unlist(overall_weight))
overall_weight_range = range(unlist(overall_weight))

global_efficiency_mean = mean(unlist(global_efficiency))
global_efficiency_sd = sd(unlist(global_efficiency))
global_efficiency_range = range(unlist(global_efficiency))

variable_degrees_mat = do.call(rbind, variable_degrees)
variable_degrees_mean = apply(variable_degrees_mat, 2, mean)
variable_degrees_sd = apply(variable_degrees_mat, 2, sd)
variable_degrees_range = apply(variable_degrees_mat, 2, range)

variable_strengths_mat = do.call(rbind, variable_strengths)
variable_strengths_mean = apply(variable_strengths_mat, 2, mean)
variable_strengths_sd = apply(variable_strengths_mat, 2, sd)
variable_strengths_range = apply(variable_strengths_mat, 2, range)

betweenness_mat = do.call(rbind, betweenness)
betweenness_mean = apply(betweenness_mat, 2, mean)
betweenness_sd = apply(betweenness_mat, 2, sd)
betweenness_range = apply(betweenness_mat, 2, range)

# store all summary stats in a df, build out table functionalities

net_stat = data.frame("Density" = c(density_mean, density_sd, density_range), "Overall Weight" = c(overa
row.names(net_stat) = c("Mean", "Standard Deviation", "Minimum", "Maximum")
```

```
# attr(net_stat, "row.names")
typeof(row.names(net_stat))
```

```
## [1] "character"
```

```
kt = knitr::kable(x =net_stat, row.names =TRUE, col.names = c("Density", "Overvall Weight", "Global Effi
# net_stat %>%
kt
```

	Density	Overvall Weight	Global Efficiency
Mean	0.1651923	10.90231	24.557177
Standard Deviation	0.0643016	10.72964	33.724137
Minimum	0.0461538	-32.41157	1.450913
Maximum	0.3230769	33.45451	187.580449

```
# kable_styling(kt, latex_options = "striped", full_width = F)
```

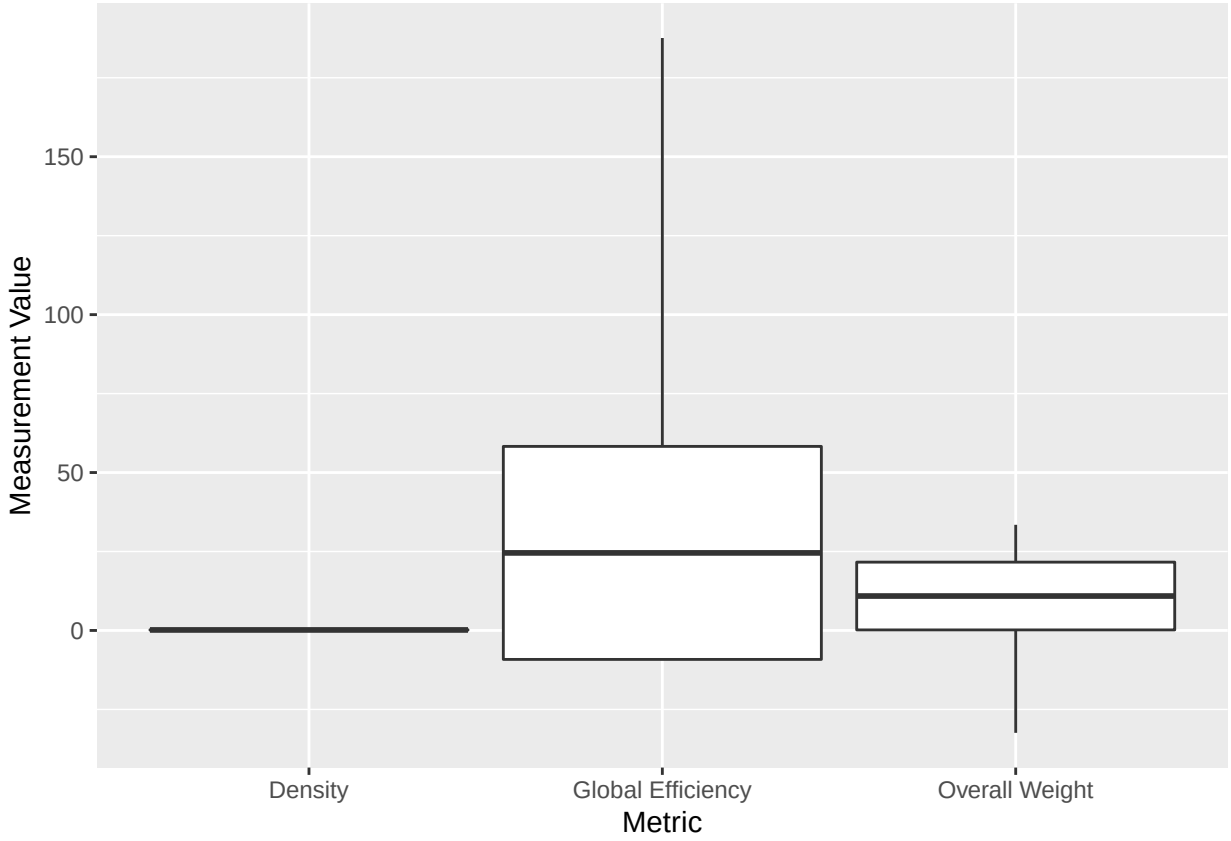
Plots and Distribution

Boxplot Distribution Across each Network-Wide Metric Using Summary Statistics

```
network_stat = data.frame("mean" = c(density_mean, overall_weight_mean, global_efficiency_mean), "sd" =

p = ggplot(network_stat, aes(x = as.factor(group))) +
  geom_boxplot(aes(
      lower = mean - sd,
      upper = mean + sd,
      middle = mean,
      ymin = minimum,
      ymax = maximum),
      stat = "identity")

p + xlab("Metric") +
  ylab("Measurement Value")
```

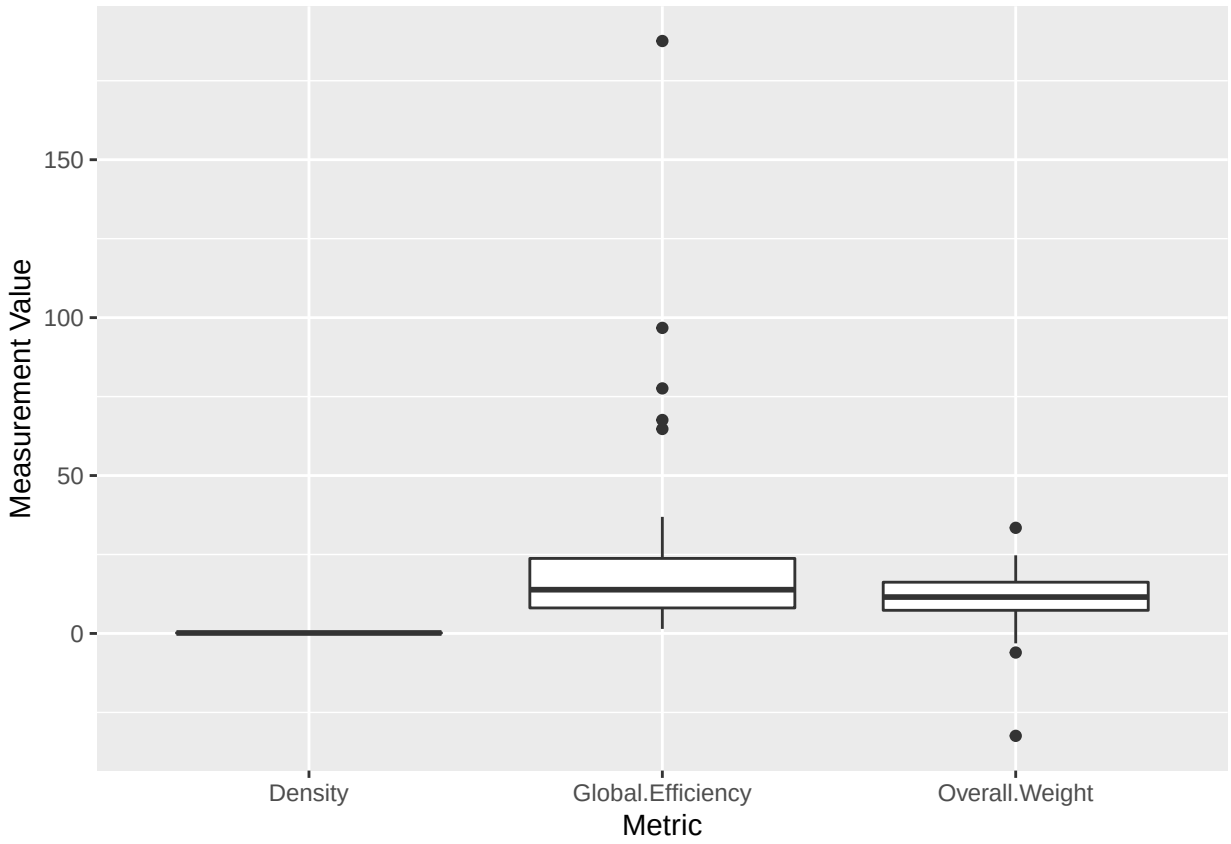


Distribution Across each Network-Wide Metric Using Data

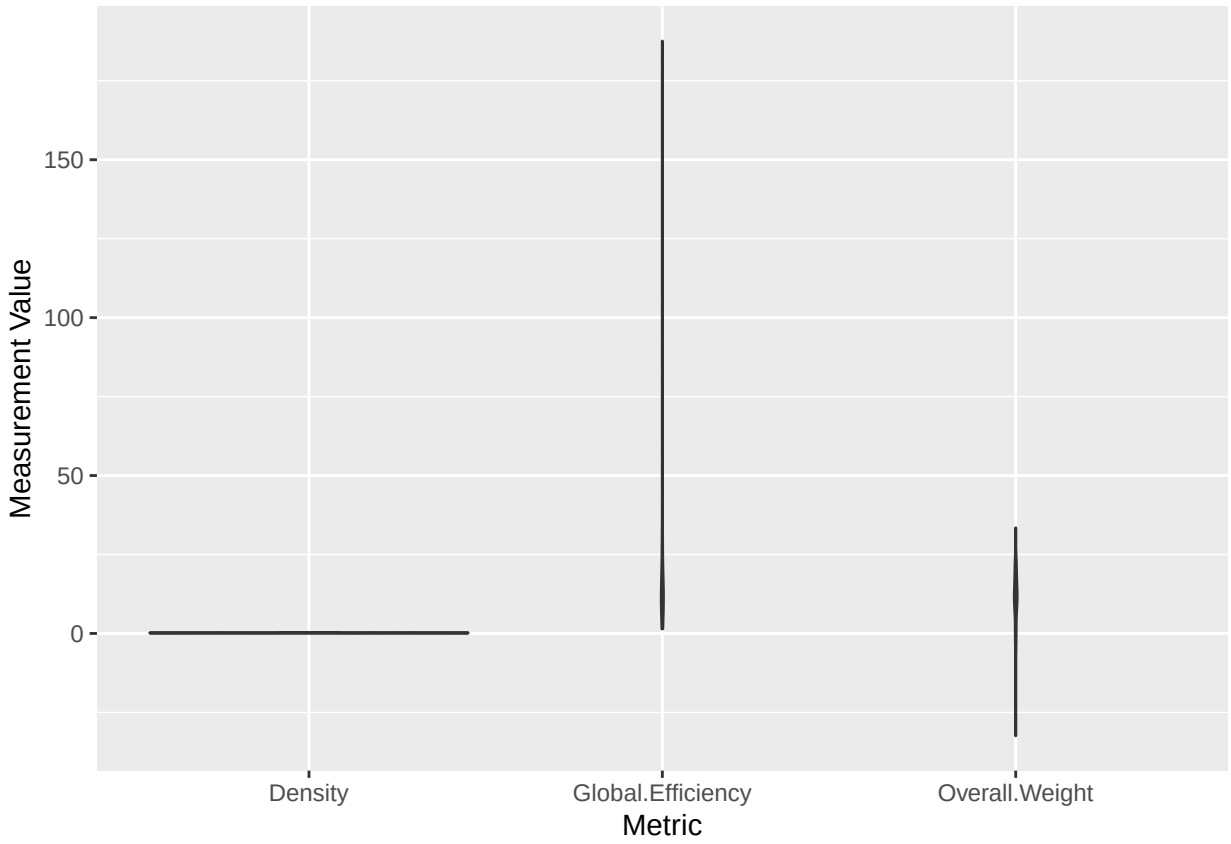
```
# par(mfrow=3)

# Combine in network-wide metrics into a dataframe
network_df = data.frame("Density" = unlist(density), "Overall Weight" = unlist(overall_weight), "Global

# Group Boxplot
stack(network_df) %>%
  dplyr::mutate(ordered_ind = factor(ind,
                                     levels = c("Density", "Global.Efficiency", "Overall.Weight"))) %>%
  ggplot(aes(x = ordered_ind, y = values)) +
  geom_boxplot() + xlab("Metric") +
  ylab("Measurement Value")
```

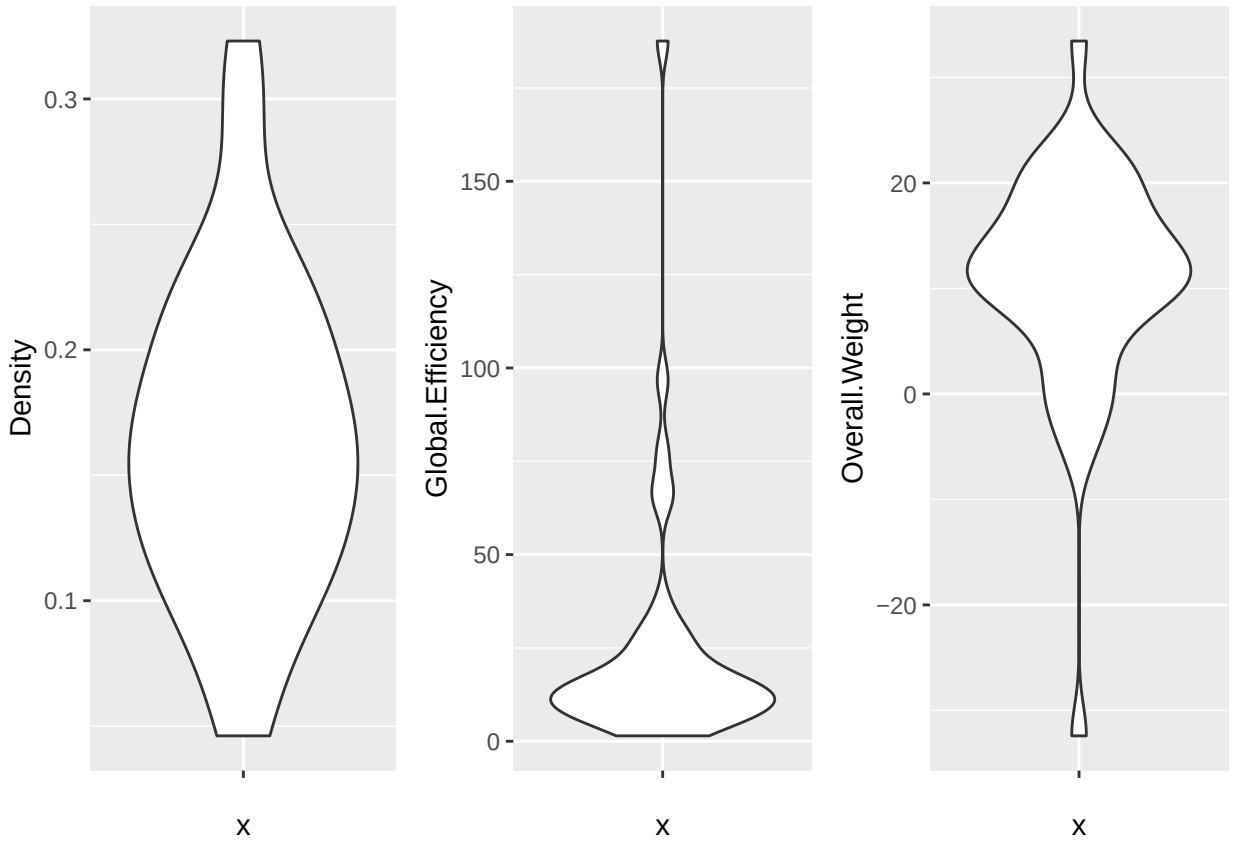



```
# Group Violin Plot
stack(network_df) %>%
  dplyr::mutate(ordered_ind = factor(ind,
                                     levels = c("Density", "Global.Efficiency", "Overall.Weight"))) %>%
  ggplot(aes(x = ordered_ind, y = values)) +
  geom_violin() + xlab("Metric") +
  ylab("Measurement Value")
```



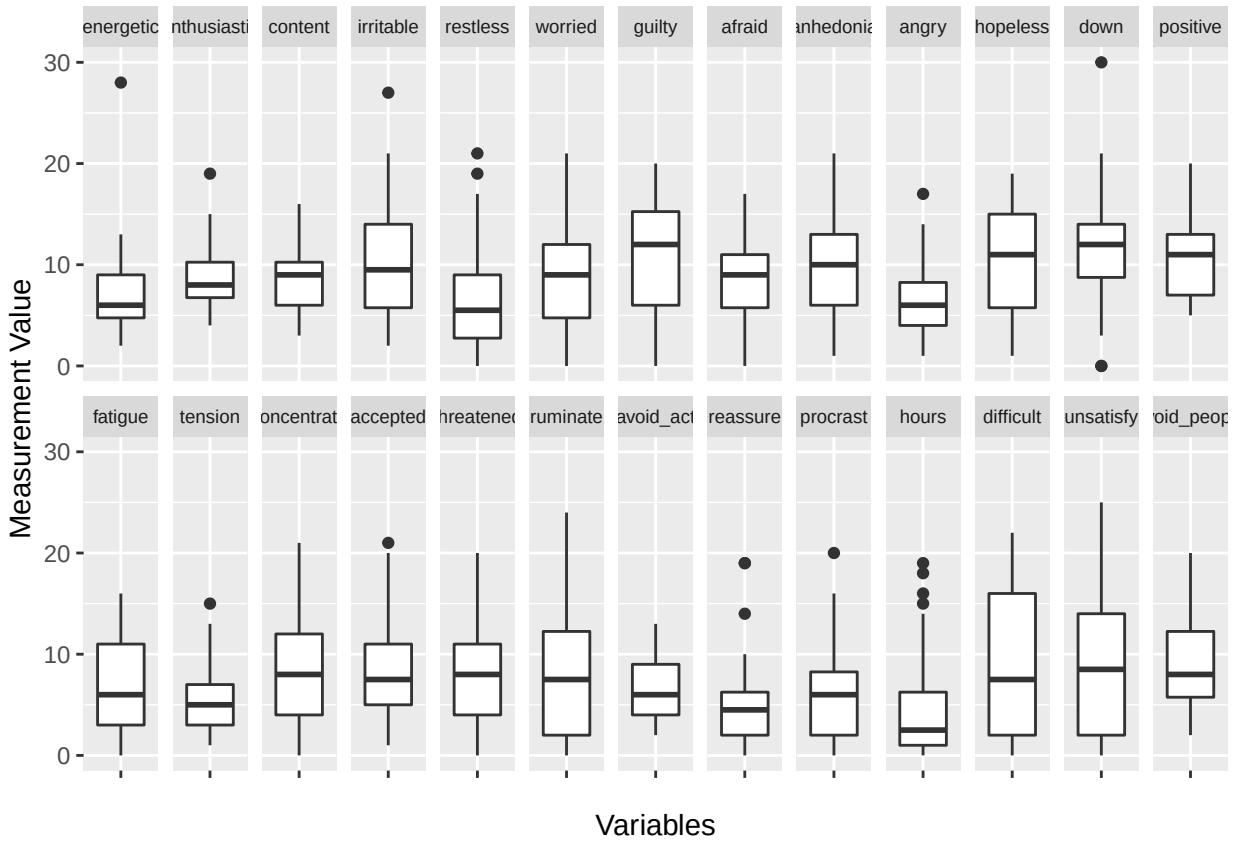

```
# Violin Plots

density_plot = ggplot(network_df, aes(y = Density, x = "")) + geom_violin()
efficiency_plot = ggplot(network_df, aes(y = Global.Efficiency, x = "")) + geom_violin()
overall_weight_plot = ggplot(network_df, aes(y = Overall.Weight, x = "")) + geom_violin()
grid.arrange(density_plot, efficiency_plot, overall_weight_plot, ncol = 3)
```

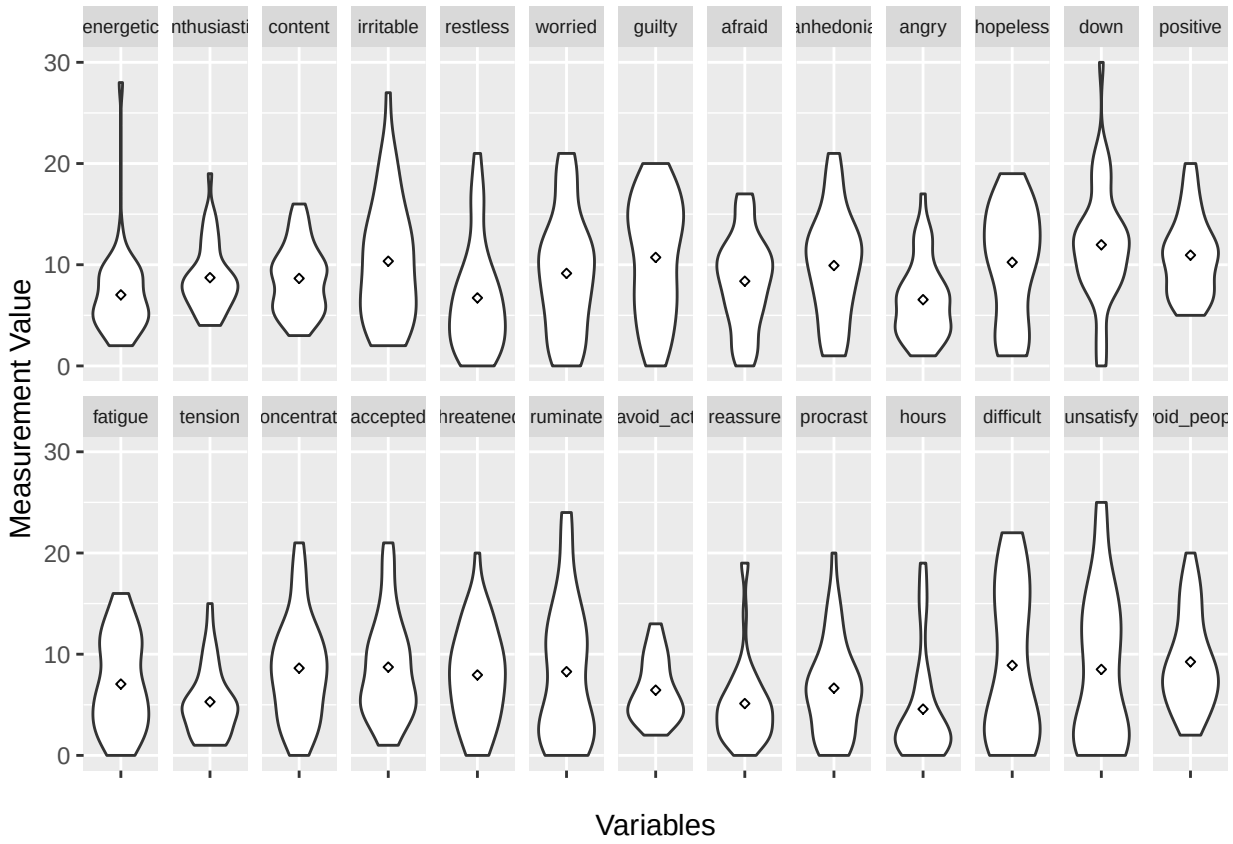


Distributions Across each Variable

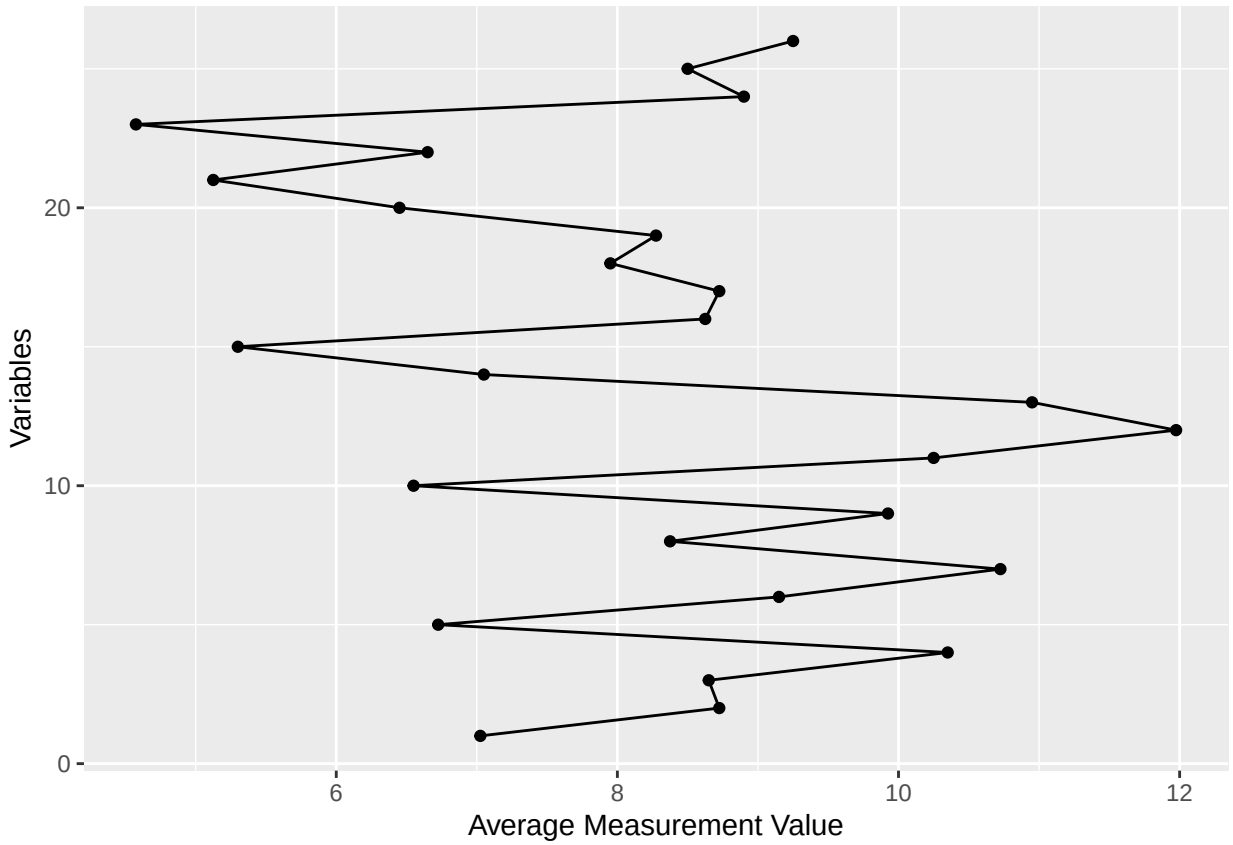
```
# Boxplot
ggplot(stack(data.frame(variable_degrees_mat)), aes(x ="" , y = values)) + geom_boxplot() + facet_wrap(
```



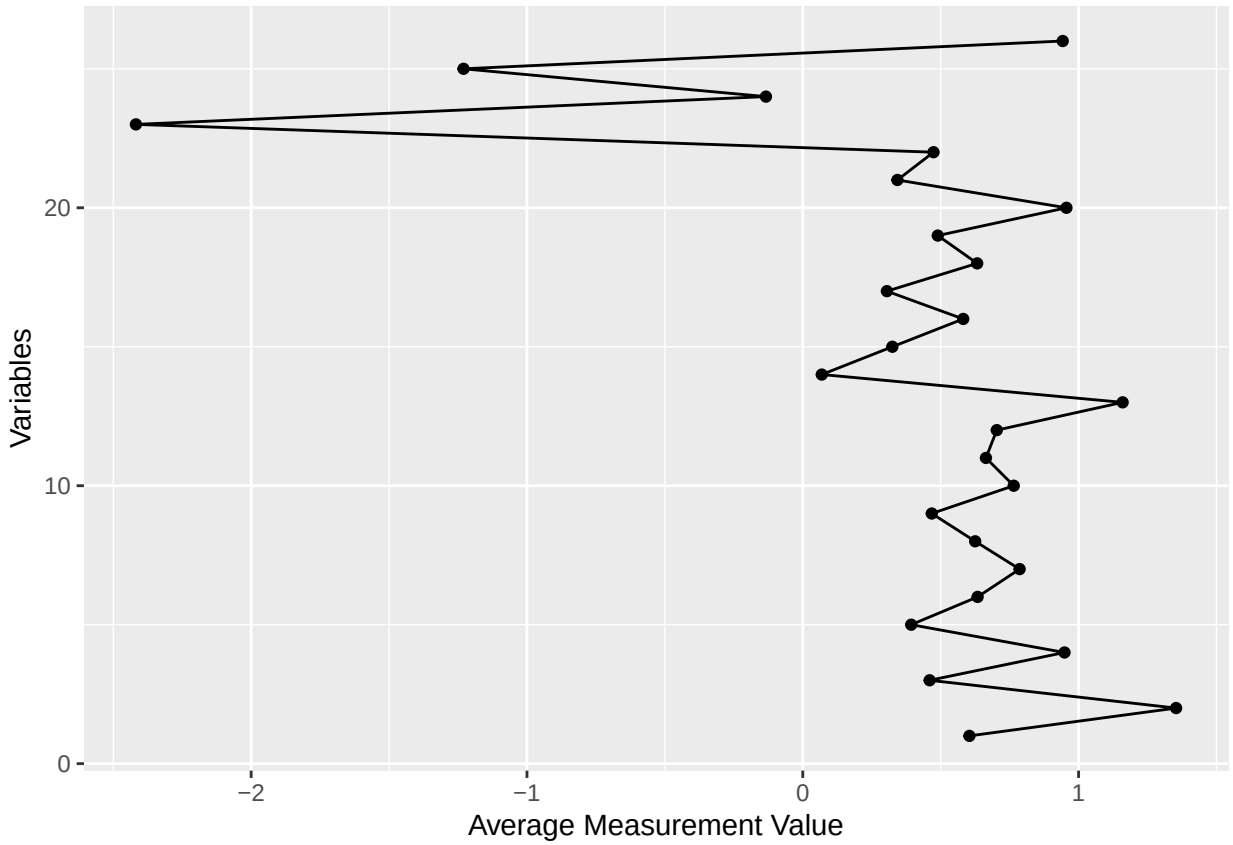

```
# Violin Plot
ggplot(stack(data.frame(variable_degrees_mat)), aes(x ="" , y = values)) + geom_violin() + facet_wrap(~
```



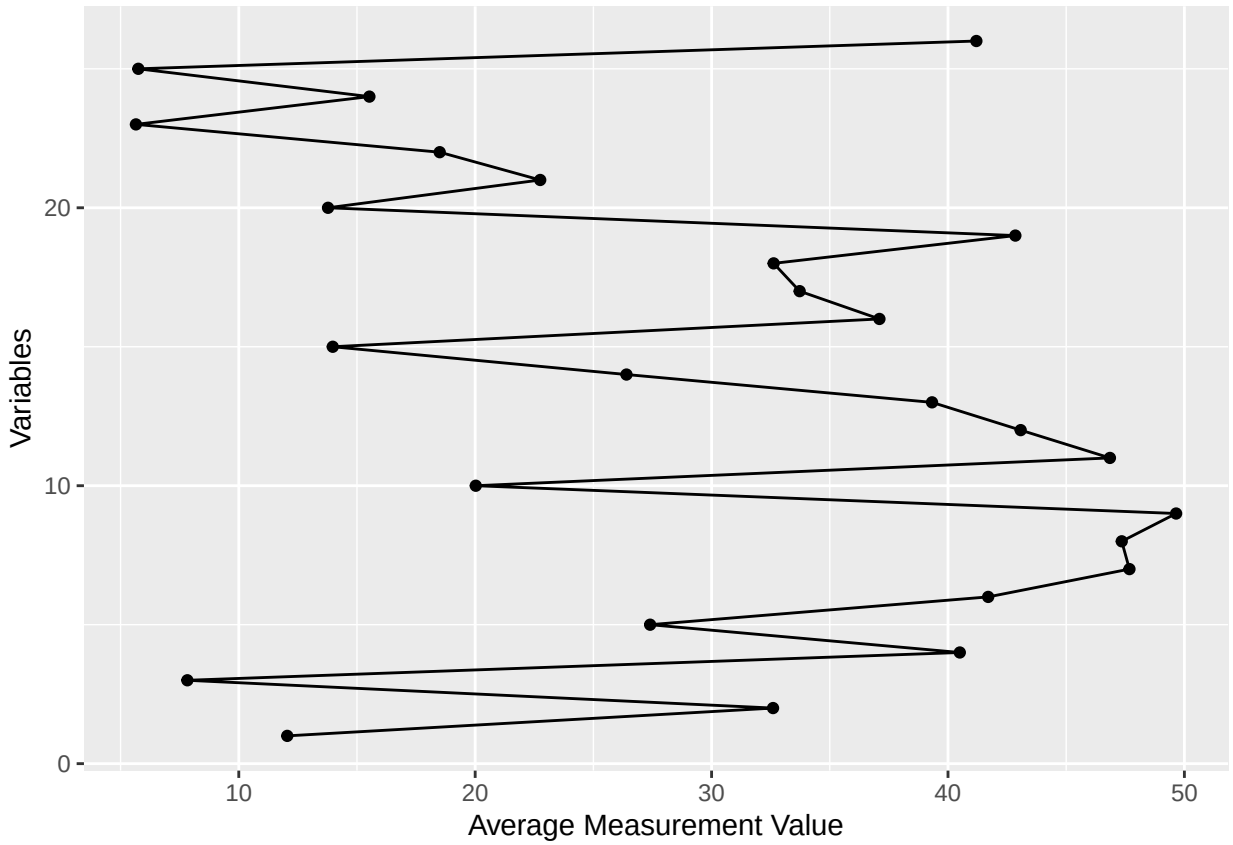

```
ggplot(data.frame(variable_degrees_mean, variable = seq(1,26)), aes(x = variable_degrees_mean, y = varia
```




```
ggplot(data.frame(variable_strengths_mean, variable = seq(1,26)), aes(x = variable_strengths_mean, y = v
```

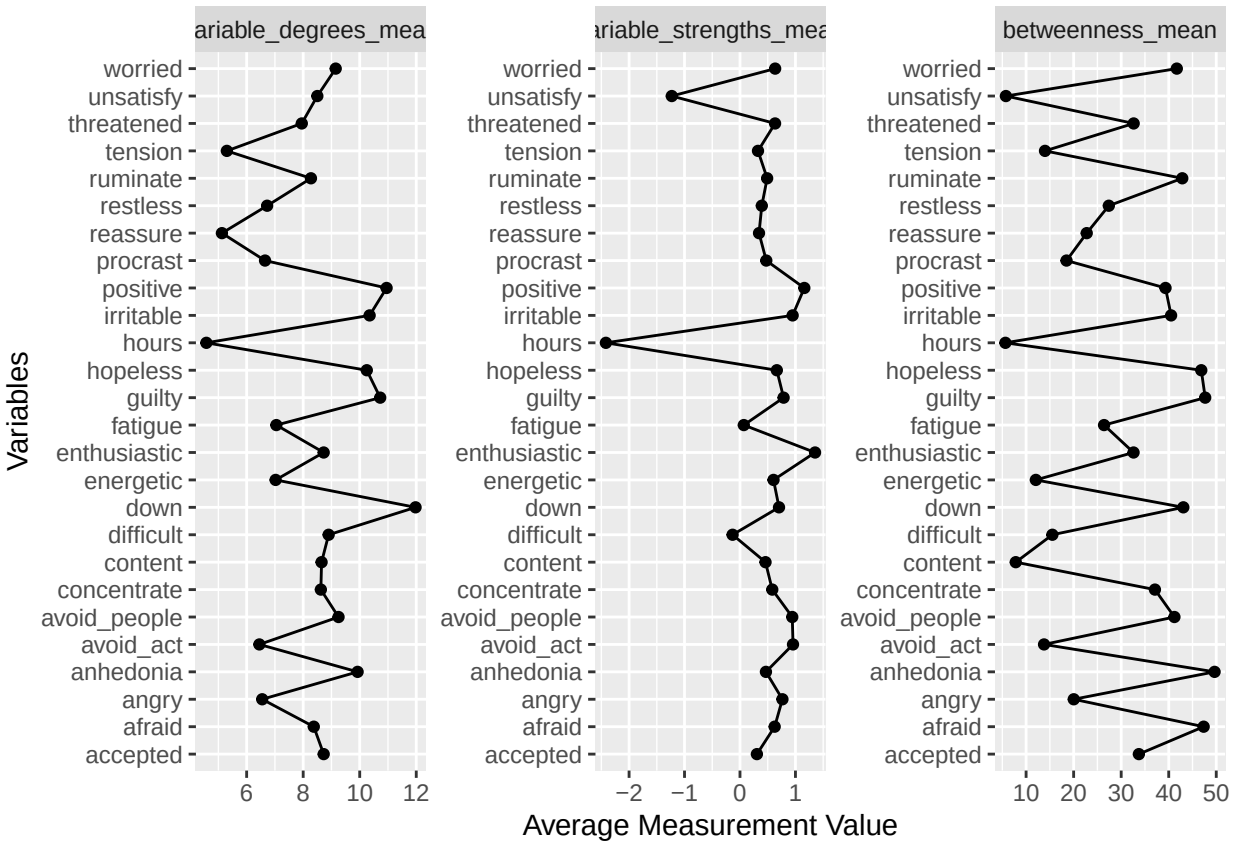



```
ggplot(data.frame(betweenness_mean, variable = seq(1,26)), aes(x = betweenness_mean, y = variable)) + g
```




```
ggplot(data.frame(stack(data.frame(variable = feature_names, variable_degrees_mean, variable_strengths_m
```

```
## Warning in stack.data.frame(data.frame(variable = feature_names,
## variable_degrees_mean, : non-vector columns will be ignored
```


ariable_degrees_mea
ariable_strengths_mea
betweenness_mean
worried
unsatisfy
threatened
tension
ruminate
restless
reassure
procrast
positive
irritable
hours
hopeless
guilty
fatigue
enthusiastic
energetic
down
difficult
content
concentrate
avoid_people
avoid_act
anhedonia
angry
afraid
accepted
Variables
6
8
10
12
−2
−1
0
1
10
20
30
40
50
Average Measurement Value