APPENDIX 1: SNEP IESF Conceptual Diagram Construction Code and Notes

Chip Heil and Emily Shumchenia - E&C Enviroscape, LLC.

3/23/2020

SNEP IESF Conceptual Diagram

Edge Bundling and Heirarchy

This document shows how parts of the **SNEP Integrated Ecosystem Services Framework (IESF) Conceptual Diagram** were created. At this stage, the major components of the diagram were created in RStudio and assembled in Adobe Illustrator. This report will show the *Edge Bundling* and the *Heirarchical Structure*.

Below you'll find pieces of the R code used to construct a **Network Diagram**. An explanation of the intent of the code will be provided for context along with citations where appropriate.

The data used in this diagram originates from the **SNEP Region IESF Relational Database** which outlines the hierarchecal structure between *Ecosystem Goods and Services*, *Beneficiaries*, and *Indicators and Metrics*.

Step 1 - Loading the R Libraries

Here we load all of the libraries that are required for the different functions that will be called in the code. Not all of these libraries are critical, but it might be helpful to have them loaded in case you want to modify the code.

```
library(ggraph)
library(igraph)
library(tidyverse)
library(RColorBrewer)
library(ggforce)
library(tidygraph)
library(circlize)
```

Step 2 - Loading the Data and Making Connections

At this stage we are loading the EXCEL CSV files that contain the individual components of the IESF Relational Database. In this case, three (3) different files were loaded based on their heirarchy, i.e., Group, Subgroup, Subgroup1, etc... These files can be defined by the user, but they need to show two (2) columns labelled "from" and "to", respectively, because they establish the hierarchechal connections. For example, the following code loads the Group to Subgroup components and connections:

```
d2 <- data.frame(read.csv("./Data files/IESF-Group.csv", header=T, as.is=T))

d2
```

```
##                          from                              to
## 1   Ecosystem Goods and Services         Provisioning Service
## 2   Ecosystem Goods and Services           Regulating Service
## 3   Ecosystem Goods and Services Habitat or Supporting Service
## 4   Ecosystem Goods and Services             Cultural Service
## 5                 Beneficiaries                 Agricultural
## 6                 Beneficiaries          Commercial/Industrial
## 7                 Beneficiaries             Education/Research
## 8                 Beneficiaries                   Government
## 9                 Beneficiaries                      Non-Use
## 10                Beneficiaries                  Recreational
## 11                Beneficiaries                   Residential
## 12                Beneficiaries                   Subsistence
## 13        Indicators and Metrics            Environmental Data
## 14        Indicators and Metrics                   Social Data
```

The following code shows all of the data files that are loaded and then combines them into one object that we've called "edges". In network diagrams, **Edges** refer to the *hierarchechal connections* between the **Groups** and **Subgroups**.

```
d1 <- data.frame(read.csv("./Data files/IESF-Origin.csv", header=T, as.is=T))
d2 <- data.frame(read.csv("./Data files/IESF-Group.csv", header=T, as.is=T))
d3 <- data.frame(read.csv("./Data files/IESF-Subgroup1.csv", header=T, as.is=T))
edges <- rbind(d1, d2, d3)
```

The next line of code creates the *relational connections* between the individual components called **Vertices**. This builds the connections between all the *Ecosystem Goods and Services* and their *Beneficiaries* and *Indicators and Metrics*.

```
connect <- data.frame(read.csv("./Data files/IESF-Edges.csv", header=T, as.is=T))
```

Step 3 - Creating the Vertices

In this step, we create the diagram's **Vertices** and their labels. Some code is included to arrange the labels appropriately, but I preferred to recreate and reorient the labels in Illustrator later.

```
## Create a vertices data.frame. One line per object of the hierarchy
vertices  <-  data.frame(
  name = unique(c(as.character(edges$from), as.character(edges$to))) ,
  value = runif(67)
)
## Add a column with the group of each name. It will be useful later to color points
vertices$group  <-  edges$from[ match( vertices$name, edges$to ) ]

## Add information concerning the label to be added: angle, horizontal adjustement
## and potential flip
```

```r
## First calculate the ANGLE of the labels
vertices$id <- NA
myleaves <- which(is.na( match(vertices$name, edges$from) ))
nleaves <- length(myleaves)
vertices$id[ myleaves ] <- seq(1:nleaves)
vertices$angle <- 90 - 360 * vertices$id / nleaves

## Then calculate the alignment of labels: right or left
## If I am on the left part of the plot, my labels have currently an angle < -90
vertices$hjust <- ifelse( vertices$angle < -90, 1, 0)

## Now flip the angle BY to make them readable
vertices$angle <- ifelse(vertices$angle < -90, vertices$angle+180, vertices$angle)
```

Step 4 - Creating the Diagram

The code below creates connections between the different **Subgroup1** components (vertices) of the relational database and plots the diagram. The diagram does not yet show the complete hierarchy, but the colors of the vertices do correspond to the **Group** level.

```r
## Create a graph object
mygraph <- igraph::graph_from_data_frame(edges, vertices=vertices)

## The connection object must refer to the ids of the leaves:
from  <-  match( connect$from, vertices$name)
to  <-  match( connect$to, vertices$name)

ggraph(mygraph, layout = 'dendrogram', circular = TRUE) +
  ## creating the nodes for the vertices and coloring them based on the upper level group
  geom_node_point(aes(filter = leaf, x = x*1.00, y=y*1.00, colour=group, size=0.99,
                      alpha=0.2)) +
  ## bundling the connections
  geom_conn_bundle(data = get_con(from = from, to = to), alpha=0.5, width=0.5,
                   show.legend = FALSE, aes(colour=..index..), tension=0.6) +
  ## setting the color palette of the connections
  scale_edge_colour_distiller(palette = "BuGn") +
  ## creating the text for the vertices
  geom_node_text(aes(x = x*1.15, y=y*1.15, filter = leaf, label=name), size=2,  alpha=1) +
  coord_fixed() +
  theme_no_axes() +
  scale_size_continuous( range = c(0.1,10) ) +
  scale_y_continuous(breaks = NULL) +
  theme_void() +
  theme(
    legend.position="none",
    plot.margin=unit(c(0,0,0,0),"cm"),
  ) +
  guides(size=FALSE) +
  guides(alpha=FALSE) +
  labs(colour="") +

  expand_limits(x = c(-1.5, 1.5), y = c(-1.5, 1.5))
```
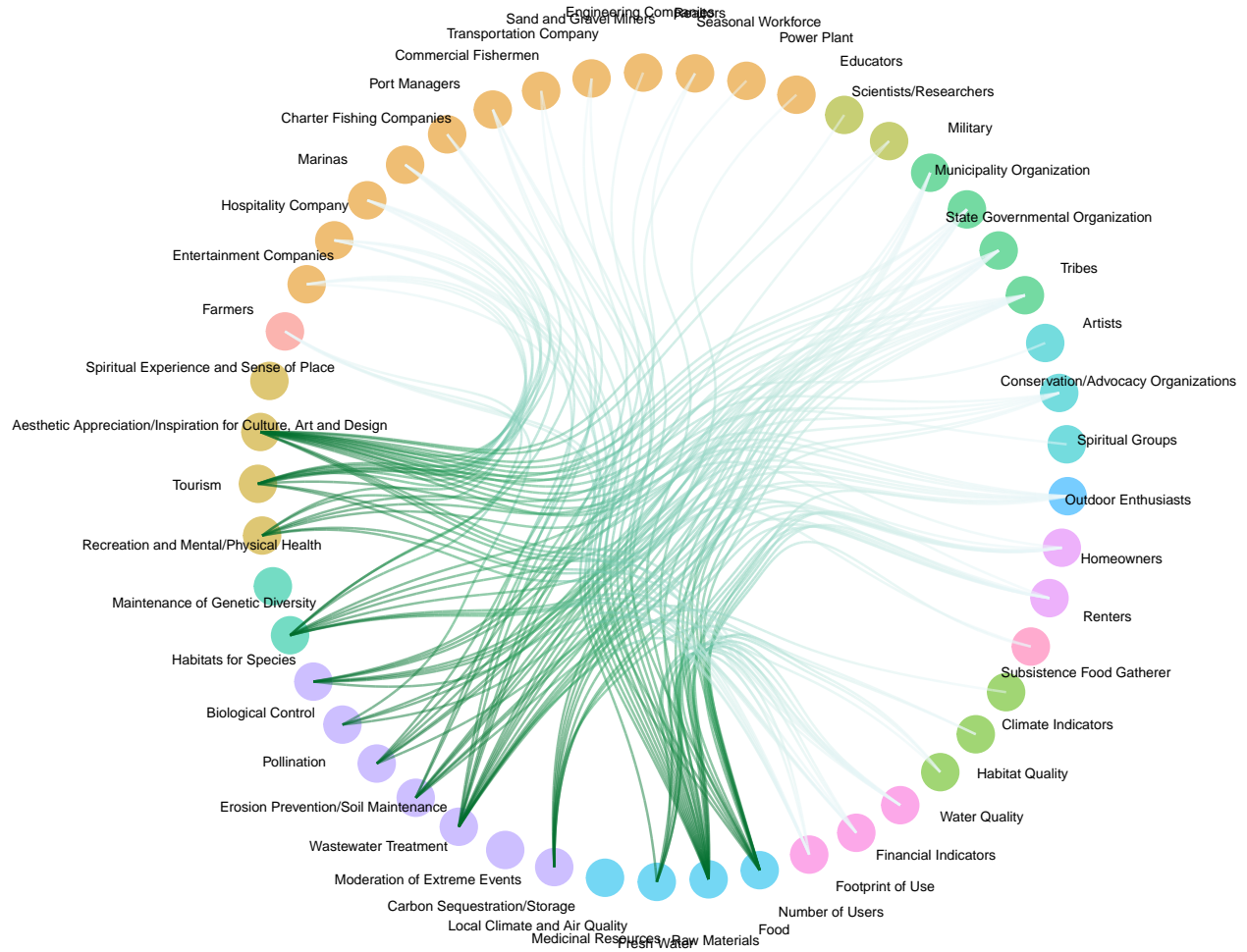
In the next chunk of code, we create *arc bars* to represent the upper levels of the hierarchy (e.g., **Group** and **Subgroup**).

```
d2$amount <- c(4, 7, 2, 4, 1, 12, 2, 4, 3, 1, 2, 1, 3, 3)
d3$amount1 <-rep(c(1), each=49)
d1$amount2 <- c(17, 26, 6)

ggraph(mygraph, layout = 'dendrogram', circular = TRUE) +
  ## arc_bar for Group
  geom_arc_bar(aes(x0 = 0, y0 = 0, r0 = 1.4, r = 1.6, amount = amount2, fill = d1$to),
               alpha = 0.2, data = d1, stat = 'pie', show.legend = FALSE) +
  ## arc_bar for Subgroup
  geom_arc_bar(aes(x0 = 0, y0 = 0, r0 = 1.2, r = 1.4, amount = amount, fill = d2$to),
               alpha = 0.2, data = d2, stat = 'pie', show.legend = FALSE) +
  #arc_bar for subgroup1
  geom_arc_bar(aes(x0 = 0, y0 = 0, r0 = 1.0, r = 1.2, amount = amount1, fill = d3$to),
               alpha = 0.2, data = d3, stat = 'pie', show.legend = FALSE) +
  coord_fixed() +
  theme_no_axes() +
  scale_size_continuous( range = c(0.1,10) ) +
  scale_y_continuous(breaks = NULL) +
```
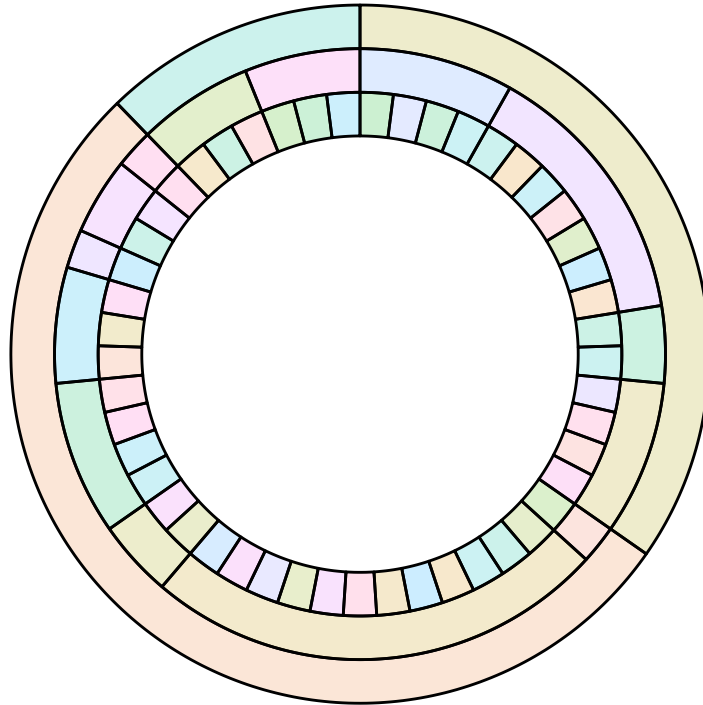
```
theme_void() +
theme(
  legend.position="none",
  plot.margin=unit(c(0,0,0,0),"cm"),
) +
guides(size=FALSE) +
guides(alpha=FALSE) +
labs(colour="") +

expand_limits(x = c(-1.5, 1.5), y = c(-1.5, 1.5))
```


Now we can combine the last two (2) figures to show *BOTH* the heirarchy and the connections.

```
ggraph(mygraph, layout = 'dendrogram', circular = TRUE) +
  ## bundling the connections
  geom_conn_bundle(data = get_con(from = from, to = to), alpha=0.5, width=0.5,
                   show.legend = FALSE, aes(colour=..index..), tension=0.6) +
  ## setting the color palette of the connections
  scale_edge_colour_distiller(palette = "BuGn") +
  ## creating the text for the vertices
  geom_node_text(aes(x = x*1.15, y=y*1.15, filter = leaf, label=name), size=2,  alpha=1) +
  ## arc_bar for Group
  geom_arc_bar(aes(x0 = 0, y0 = 0, r0 = 1.4, r = 1.6, amount = amount2, fill = d1$to),
               alpha = 0.2, data = d1, stat = 'pie', show.legend = FALSE) +
  ## arc_bar for Subgroup
  geom_arc_bar(aes(x0 = 0, y0 = 0, r0 = 1.2, r = 1.4, amount = amount, fill = d2$to),
               alpha = 0.2, data = d2, stat = 'pie', show.legend = FALSE) +
  #arc_bar for subgroup1
  geom_arc_bar(aes(x0 = 0, y0 = 0, r0 = 1.0, r = 1.2, amount = amount1, fill = d3$to),
               alpha = 0.2, data = d3, stat = 'pie', show.legend = FALSE) +
```
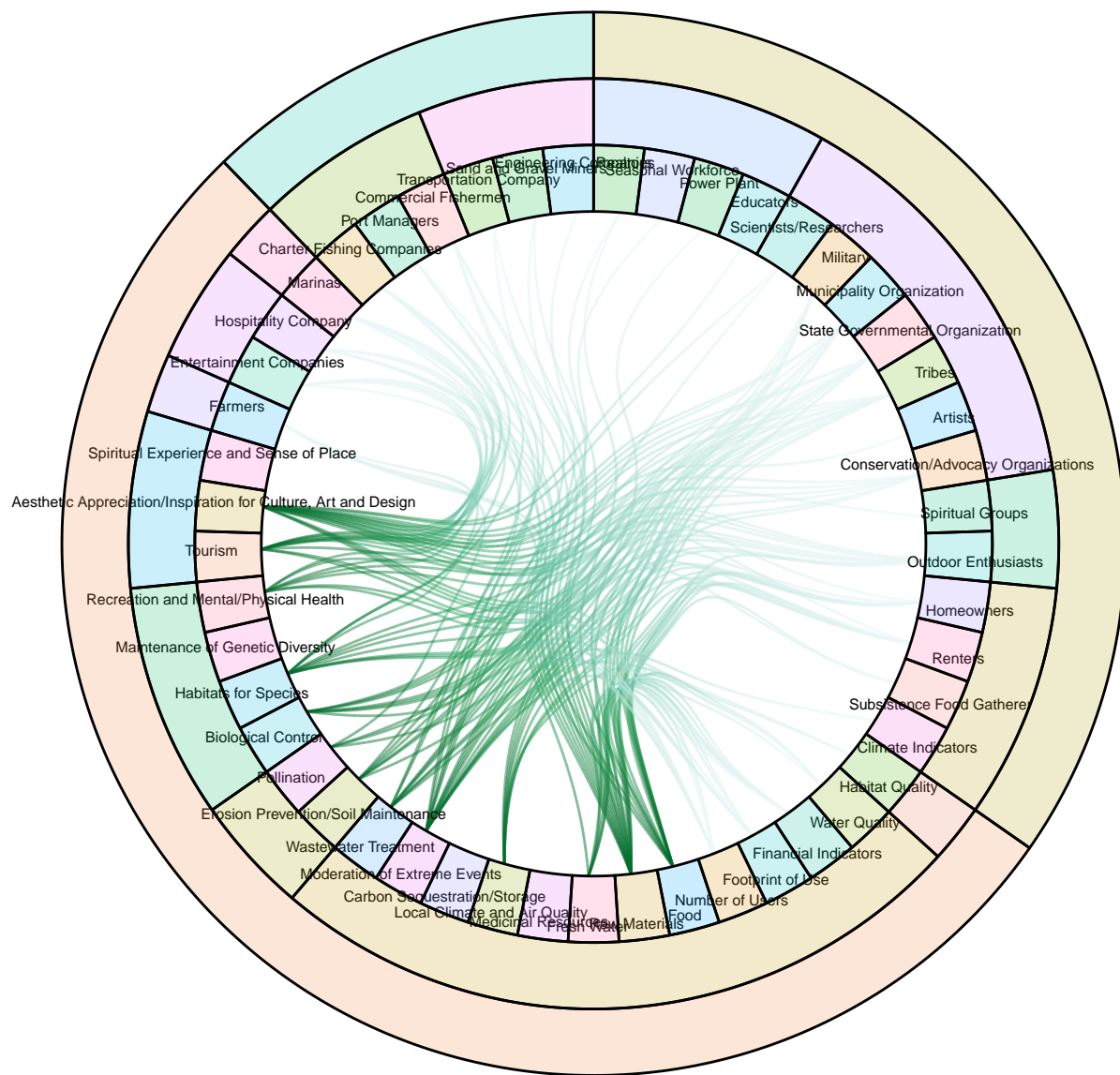
```
coord_fixed() +
theme_no_axes() +
scale_size_continuous( range = c(0.1,10) ) +
scale_y_continuous(breaks = NULL) +
theme_void() +
theme(
  legend.position="none",
  plot.margin=unit(c(0,0,0,0),"cm"),
) +
guides(size=FALSE) +
guides(alpha=FALSE) +
labs(colour="") +

expand_limits(x = c(-1.5, 1.5), y = c(-1.5, 1.5))
```


From this point, the the final figure was exported as a PDF so that it could be opened in Illustrator to

insert and modify the labels and colors according to their groupings. Future work could include adding the appropriate code to accomplish both of those tasks.

Step 5 - Highlighting specific connections

This framework was created to serve a broad range of users including scientists, managers, and the general public. As such, the users may "enter" the framework from different points. For instance, Comercial Fisherman can enter from their block and see only the connections between **Ecosystem Goods and Services**, **Indictors and Metrics**, and themselves. The code below was created to highlight just those connections using a simple spreadsheet that has **Commercial Fisherman** (Beneficiaries) in the "from" column (column 1) and all the services and metrics in the "to" column (column 2). Other example data files are provided for entry points from **Water Quality** (Indictors and Metrics) and **Food** (Ecosystem Goods and Services). Future work could include adding the code to extract the connections for a given entry point and creating the data frame.

```r
## Highlighting the connections from one (1) starting point:

## There are three (3) data files included in this porject for this demonstration
## 1) connect_CF is the connections starting from Commercial Fisherman (Beneficiaries)
## 2) connect_FOOD is the connections starting from Food (Ecosystem goods and Services)
## 3) connect_WQ is the connections starting from Water Quality (Indicators and Metrics)
## the code below is just with the connections for Commercial Fisherman but can be
## modified by changing the file name

connect_CF <- data.frame(read.csv("./Data files/connect_CF.csv", header=T, as.is=T))
from_head_CF = match(connect_CF$from, vertices$name) %>% head(16)
to_head_CF = match(connect_CF$to, vertices$name) %>% head(16)

ggraph(mygraph, layout = 'dendrogram', circular = TRUE) +
  ## bundling the connections
  geom_conn_bundle(data = get_con(from = from_head_CF, to = to_head_CF), alpha = 1,
                   colour="#69b3a2", width=0.5, tension=0.9) +
  ## setting the color palette of the connections
  scale_edge_colour_distiller(palette = "BuGn") +
  ## creating the text for the vertices
  geom_node_text(aes(x = x*1.15, y=y*1.15, filter = leaf, label=name), size=2,  alpha=1) +
  ## arc_bar for Group
  geom_arc_bar(aes(x0 = 0, y0 = 0, r0 = 1.4, r = 1.6, amount = amount2, fill = d1$to),
               alpha = 0.2, data = d1, stat = 'pie', show.legend = FALSE) +
  ## arc_bar for Subgroup
  geom_arc_bar(aes(x0 = 0, y0 = 0, r0 = 1.2, r = 1.4, amount = amount, fill = d2$to),
               alpha = 0.2, data = d2, stat = 'pie', show.legend = FALSE) +
  #arc_bar for subgroup1
  geom_arc_bar(aes(x0 = 0, y0 = 0, r0 = 1.0, r = 1.2, amount = amount1, fill = d3$to),
               alpha = 0.2, data = d3, stat = 'pie', show.legend = FALSE) +
  coord_fixed() +
  theme_no_axes() +
  scale_size_continuous( range = c(0.1,10) ) +
  scale_y_continuous(breaks = NULL) +
  theme_void() +
  theme(
    legend.position="none",
    plot.margin=unit(c(0,0,0,0),"cm"),
```

```
) +
guides(size=FALSE) +
guides(alpha=FALSE) +
labs(colour="") +

expand_limits(x = c(-1.5, 1.5), y = c(-1.5, 1.5))
```