Milestone II

Sachin Shubham

4/8/2021

Disciplinary Clusters (CIP)

Correspondence with topical (SA's) cluster as fraction of MeSH, by 5-year period

```
library(reshape2)
```

```
## Warning: package 'reshape2' was built under R version 4.0.4
```

```
library(ggplot2)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

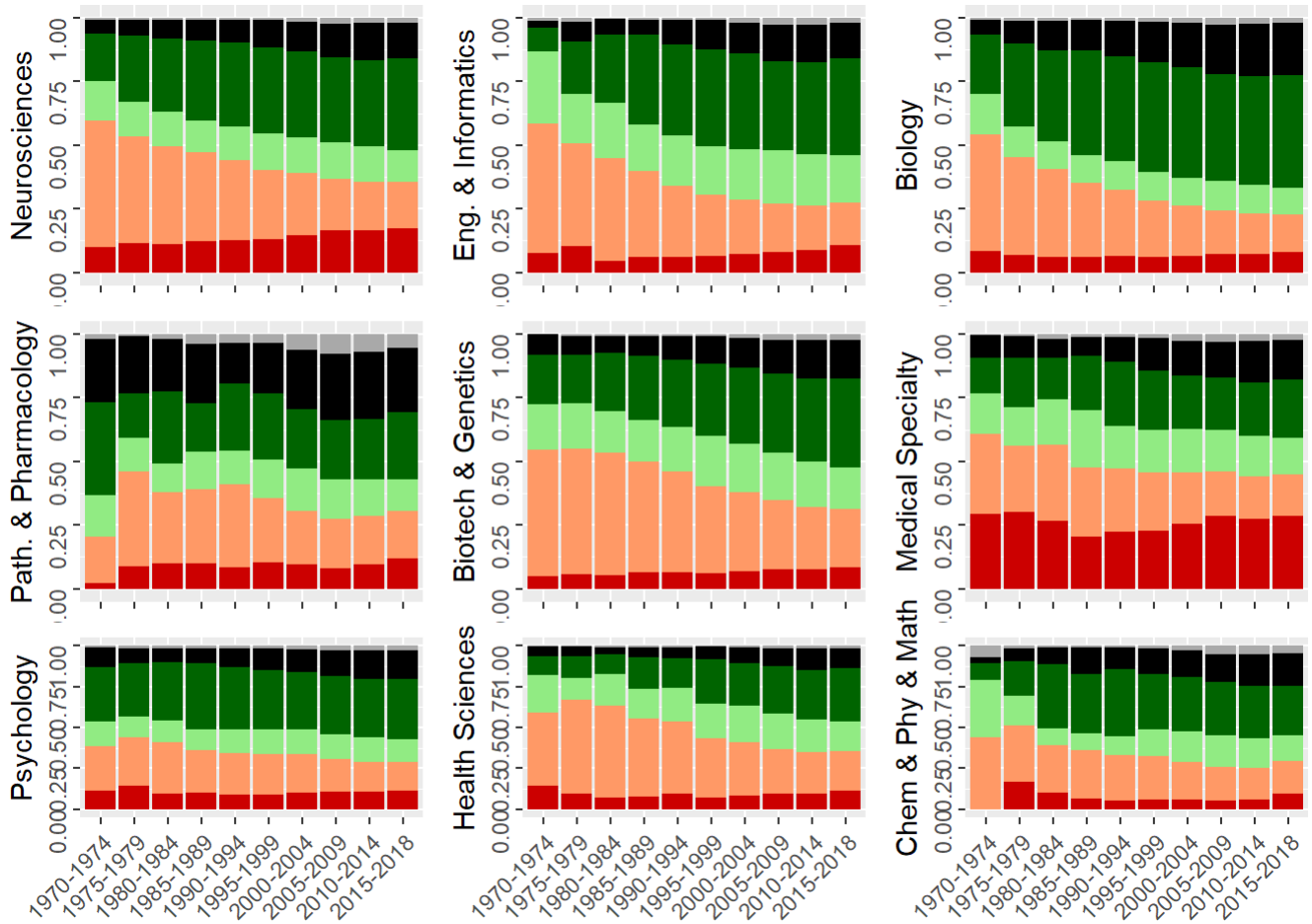
```
library(ggpubr)
```

```
## Warning: package 'ggpubr' was built under R version 4.0.5
```

```
library(grid)
setwd("D:/Statistical Methods/Project")
df1<-read.csv("ArticleLevel-RegData-ALLSA_Xc_1_NData_655386_LONGXCIP2.csv")

# Group by Years
df1 %>%
  mutate(year_group = case_when(
    (Yp %in% c(1970,1971,1972,1973,1974)) ~ "1970-1974",
    (Yp %in% c(1975,1976,1977,1978,1979)) ~ "1975-1979",
    (Yp %in% c(1980,1981,1982,1983,1984)) ~ "1980-1984",
    (Yp %in% c(1985,1986,1987,1988,1989)) ~ "1985-1989",
    (Yp %in% c(1990,1991,1992,1993,1994)) ~ "1990-1994",
    (Yp %in% c(1995,1996,1997,1998,1999)) ~ "1995-1999",
    (Yp %in% c(2000,2001,2002,2003,2004)) ~ "2000-2004",
    (Yp %in% c(2005,2006,2007,2008,2009)) ~ "2005-2009",
    (Yp %in% c(2010,2011,2012,2013,2014)) ~ "2010-2014",
    (Yp %in% c(2015,2016,2017,2018))      ~ "2015-2018",
  )) -> dfyg


#Need to make a nested list so i can call on index to create a new list
my_list <- vector(mode = "list", length = 9)

for (i in  c(1:9)){
  df_CIP1 <- subset(dfyg, dfyg[17+i] >0)
  df_CIP1 %>%
    group_by(year_group) %>%
    summarise(
      SA1 = sum(SA1 * df_CIP1[17+i]),
      SA2 = sum(SA2 * df_CIP1[17+i]),
      SA3 = sum(SA3 * df_CIP1[17+i]),
      SA4 = sum(SA4 * df_CIP1[17+i]),
      SA5 = sum(SA5 * df_CIP1[17+i]),
      SA6 = sum(SA6 * df_CIP1[17+i])
    ) -> df_CIP1_grp


df_CIP1_grp$SA_sum <- df_CIP1_grp$SA1 + df_CIP1_grp$SA2 + df_CIP1_grp$SA3+
  df_CIP1_grp$SA4 + df_CIP1_grp$SA5 + df_CIP1_grp$SA6

df_CIP1_grp$SA1_Fraction<-df_CIP1_grp$SA1/df_CIP1_grp$SA_sum
df_CIP1_grp$SA2_Fraction<-df_CIP1_grp$SA2/df_CIP1_grp$SA_sum
df_CIP1_grp$SA3_Fraction<-df_CIP1_grp$SA3/df_CIP1_grp$SA_sum
df_CIP1_grp$SA4_Fraction<-df_CIP1_grp$SA4/df_CIP1_grp$SA_sum
df_CIP1_grp$SA5_Fraction<-df_CIP1_grp$SA5/df_CIP1_grp$SA_sum
df_CIP1_grp$SA6_Fraction<-df_CIP1_grp$SA6/df_CIP1_grp$SA_sum


df_CIP1_grp$SA_sum<-NULL

df_CIP1_grp<-df_CIP1_grp[-11,]

# Rename Columns from SA to actual name
```

```
colnames(df_CIP1_grp)<-c("year_group",
                         "P&P",
                         "A&O",
                         "Ph&Pr",
                         "H",
                         "T&E",
                         "T&IS",
                         "P&P_Frac",
                         "A&O_Frac",
                         "Ph&Pr_Frac",
                         "H_Frac",
                         "T&E_Frac",
                         "T&IS_Frac")

my_list[[i]] <- df_CIP1_grp


ylabs = c("Neurosciences","Biotech & Genetics","Medical Specialty",
          "Eng. & Informatics", "Biology", "Psychology",
          "Health Sciences","Path. & Pharmacology",
          "Chem & Phy & Math")

}

plot_list <- vector(mode = "list", length = 9)

  for (i in  c(1:9) ) {
    mlt_df_CIP1_grp <-melt(my_list[[i]], id="year_group")

    f <- mlt_df_CIP1_grp[61:120,]

    gg_plot<-ggplot(f)

    gg_plot1<-gg_plot + geom_bar(aes(x=year_group, y=value,
                                     fill=forcats::fct_rev(variable)),
                                     stat="identity")



    gg_plot12<-gg_plot1+theme(axis.text.x = element_text(angle = 45, hjust = 1),
                              axis.title.x=element_blank(),legend.position="none",
                              axis.text.y = element_text(angle = 90, hjust= 1 ))

    plot_list[[i]]<-gg_plot12+ylab(colnames(df_CIP1[17+i])+
                    scale_fill_discrete(name="Subject Areas:"))+
                    scale_fill_manual(values = rev(c("#cc0000","#ff9966",
                                  "#91eb83","darkgreen","black","darkgrey")))

    plot_list[[i]]<- plot_list[[i]] + labs(y = ylabs[i])

    if(i %in% c(1,2,3,4,5,8)){
      plot_list[[i]]<-plot_list[[i]]+theme(axis.text.x = element_blank())
    }
```

```
}

ggarrange(
  plot_list[[1]],
  plot_list[[4]],
  plot_list[[5]],
  plot_list[[8]],
  plot_list[[2]],
  plot_list[[3]],
  plot_list[[6]],
  plot_list[[7]],
  plot_list[[9]],
  ncol = 3,
  nrow = 3,
  labels = NULL,
  label.x = 0,
  label.y = 1,
  hjust = -0.5,
  vjust = 1.5,
  font.label = list(size = 14, color = "black", face = "bold", family = NULL),
  align = c("none", "h", "v", "hv"),
  widths = 1,
  heights = 1,
  legend = NULL,
  common.legend = FALSE,
  legend.grob = NULL
)
```



CIP-SA Coupling in Mono-Domain Articles (2009-2018)

```
library(dplyr)
library(LICORS)
```

```
## Warning: package 'LICORS' was built under R version 4.0.4
```

```
library(scales)
library(networkD3)
```

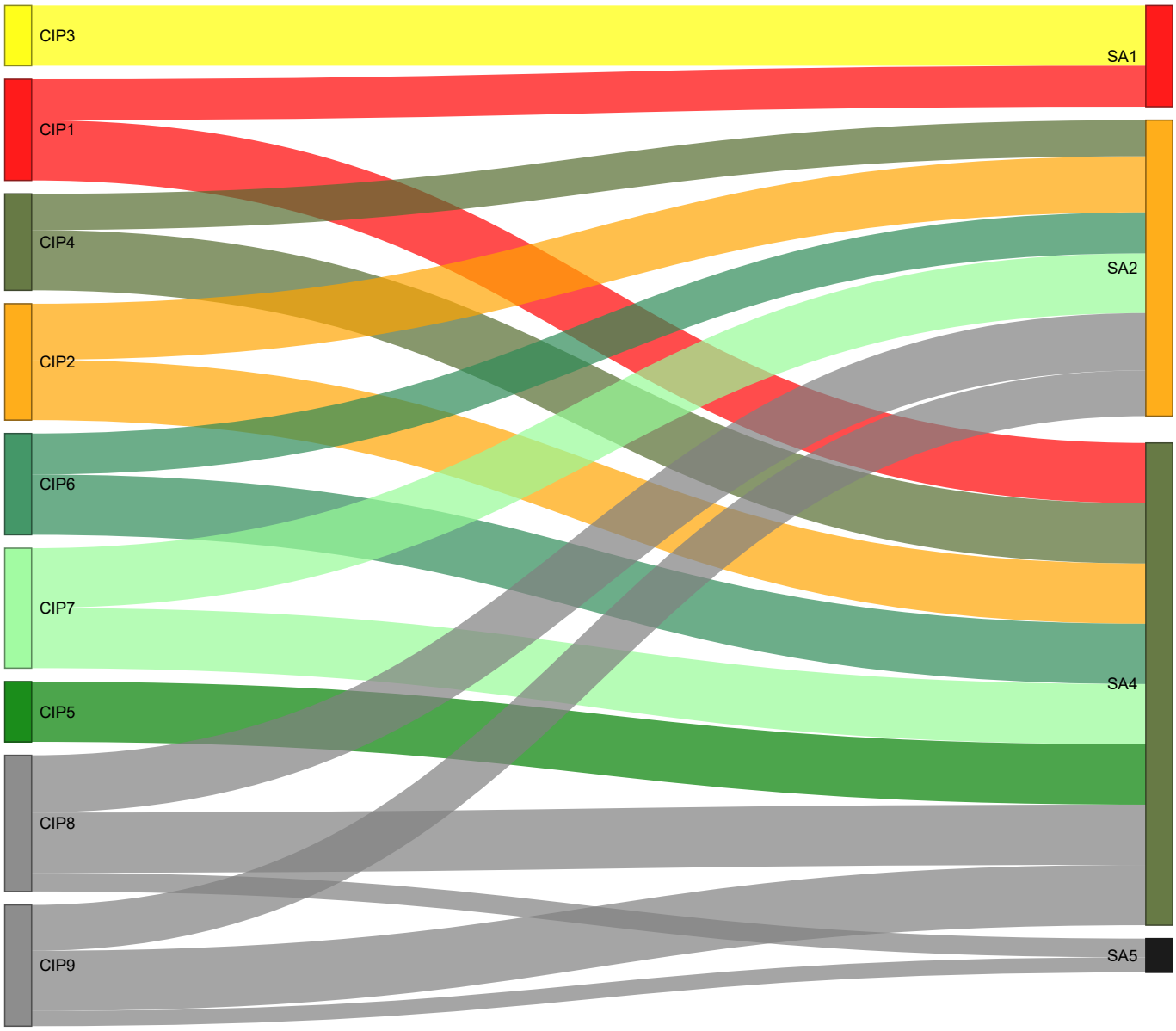
```
## Warning: package 'networkD3' was built under R version 4.0.4
```

```
# ArticleLevel-RegData-ALLSA_Xc_1_NData_655386_LONGXCIP2.csv
setwd("D:/Statistical Methods/Project")
data_csv<-read.csv("ArticleLevel-RegData-ALLSA_Xc_1_NData_655386_LONGXCIP2.csv")


# filter out years 2009 to 2018
year_2009_2018<-filter(data_csv, Yp >= 2009 & Yp <= 2018)


# Filter out IRegionRefined
IRegionRefinedp<-filter(year_2009_2018, IRegionRefinedp > 0 & IRegionRefinedp < 4)

# Filter out where both NEUROLONGXSAp & NEUROLONGXCIPp == 0
df_mono = year_2009_2018 %>% filter(NEUROLONGXSAp == 0 & NEUROLONGXCIPp == 0)


mono_mat = matrix(0L, nrow = 9, ncol = 6)
# mono matrix
for(i in 1:nrow(df_mono)){
  row = df_mono[i,]
  vSA = c(row$SA1, row$SA2, row$SA3, row$SA4, row$SA5, row$SA6)
  vCIP = c(row$CIP3, row$CIP1, row$CIP4, row$CIP2, row$CIP6, row$CIP7, row$CIP5, row$CIP8, row$C
IP9)
  vSA = round(vSA / sum(vSA),2)
  for(k in which(vCIP > 0)){
    for(j in 1:6){
      mono_mat[[k,j]] = mono_mat[[k,j]] + vSA[j]
    }

  }

}
print(mono_mat)
```

```
##          [,1]     [,2]     [,3]     [,4]    [,5]   [,6]
##  [1,]  6787.17  3618.41  2759.66  3857.22  327.66 119.34
##  [2,] 19934.66 18673.17 11479.54 29265.15 1590.53 432.47
##  [3,]  1275.54  2339.73  2539.63  3785.82  383.36 183.64
##  [4,]  3282.75  9598.81  6297.16 10326.54 1269.26 454.74
##  [5,]  4267.80  7378.05  4826.99 10700.34 1188.72 419.28
##  [6,]  1560.95  3585.80  2471.23  3617.39  329.15  96.85
##  [7,]  6939.61 11540.33  7540.50 25982.80 2206.71 618.29
##  [8,]   856.00  1388.11   811.67  1451.48  624.57 249.43
##  [9,]   573.83  1527.55  1151.15  1889.12  762.46 378.54
```

```
m = mono_mat
for(i in 1:9){
  row = mono_mat[i,]
  # m[i,] = sapply(row, function(X) {(X - min(row))/(max(row)-min(row))})
  m[i,] = rescale(row, to=c(0,1))

}
mm_b = apply(m, 2, function(x) {ifelse(x > 0.0, round(x,2), 0)})
mm = rescale(mm_b, to=c(0,0.02))


nodes = data.frame("name" = c("CIP3", "CIP1", "CIP4", "CIP2", "CIP6", "CIP7", "CIP5", "CIP8", "C
IP9", "SA1", "SA2", "", "SA4", "SA5", ""))
links = as.data.frame(matrix(c(0,9,  mm[1,1],
                               1,9,  mm[2,1],
                               1,12, mm[2,4],
                               2,10, mm[3,2],
                               2,12, mm[3,4],
                               3,10, mm[4,2],
                               3,12, mm[4,4],
                               4,10, mm[5,2],
                               4,12, mm[5,4],
                               5,10, mm[6,2],
                               5,12, mm[6,4],
                               6,12, mm[7,4],
                               7,10, mm[8,2],
                               7,12, mm[8,4],
                               7,13, mm[8,5],
                               8,10, mm[9,2],
                               8,12, mm[9,4],
                               8,13, mm[9,5]
                               ), byrow = TRUE, ncol = 3))
names(links) = c("source", "target", "value")
links$group <- as.factor(c("type_0","type_1","type_1","type_2", "type_2","type_3","type_3", "typ
e_4","type_4","type_5","type_5","type_6","type_7","type_7","type_7","type_8","type_8","type_8"))
node_color <- 'd3.scaleOrdinal() .domain(["CIP3", "CIP1", "CIP4", "CIP2", "CIP6", "CIP7", "CIP
5", "CIP8", "CIP9", "SA1", "SA2", "SA3", "SA4", "SA5", "SA6", "type_0", "type_1", "type_2", "typ
e_3", "type_4", "type_5", "type_6", "type_7", "type_8", "type_12"]) .range(["yellow", "red", "da
rkolivegreen" , "orange", "seagreen", "palegreen", "green", "gray", "gray", "red", "orange", "li
ghtgreen", "darkolivegreen", "black", "gray", "yellow", "red", "darkolivegreen" , "orange", "sea
green", "palegreen", "green", "gray", "gray","white"])'




p = sankeyNetwork(Links = links, Nodes = nodes,
                  Source = "source", Target = "target",
                  Value = "value", NodeID = "name",
                  fontSize= 12, nodeWidth = 20,
                  height = 800, width = "100%",
                  colourScale=node_color,
                  LinkGroup="group",
```

```
                iterations = 0,
                nodePadding=10)

p
```

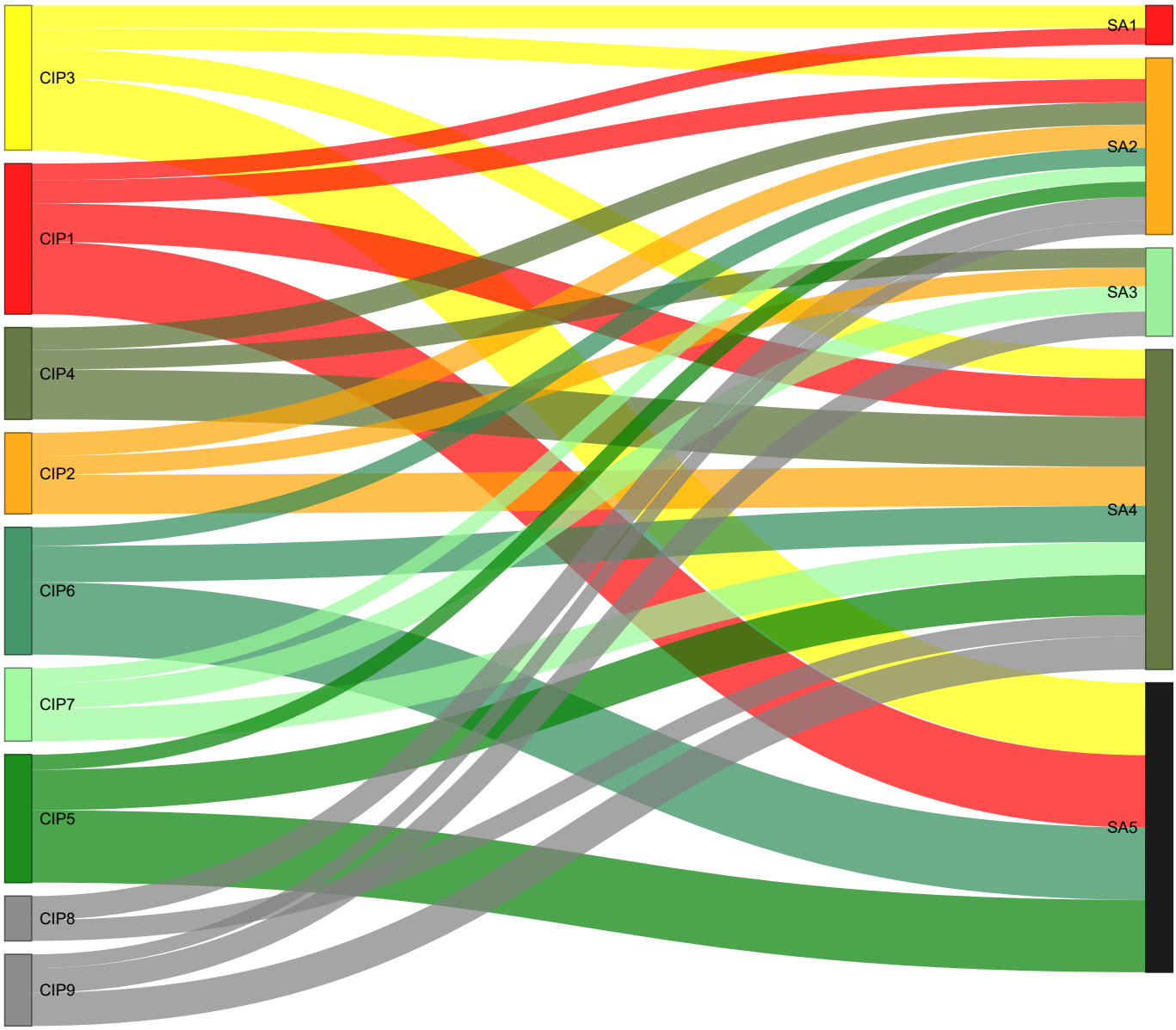


CIP-SA Coupling in Cross-Domain Articles (2009-2018)

```
library(dplyr)
library(LICORS)
library(scales)
library(networkD3)

# ArticleLevel-RegData-ALLSA_Xc_1_NData_655386_LONGXCIP2.csv
setwd("D:/Statistical Methods/Project")
data_csv<-read.csv("ArticleLevel-RegData-ALLSA_Xc_1_NData_655386_LONGXCIP2.csv")


# filter out years 2009 to 2018
year_2009_2018<-filter(data_csv, Yp >= 2009 & Yp <= 2018)

# Filter out IRegionRefined
IRegionRefinedp<-filter(year_2009_2018, IRegionRefinedp > 0 &
                          IRegionRefinedp < 4)


# Filter out where both NEUROLONGXSAp & NEUROLONGXCIPp == 1
df_XD = year_2009_2018 %>% filter(NEUROLONGXSAp == 1 & NEUROLONGXCIPp == 1)


xd_mat = matrix(0L, nrow = 9, ncol = 6)
# XD matrix
for(i in 1:nrow(df_XD)){
  row = df_XD[i,]
  vSA = c(row$SA1, row$SA2, row$SA3, row$SA4, row$SA5, row$SA6)
  vCIP = c(row$CIP3, row$CIP1, row$CIP4, row$CIP2, row$CIP6, row$CIP7, row$CIP5, row$CIP8, row$C
IP9)
  vSA = round(vSA / sum(vSA),2)
  for(k in which(vCIP > 0)){
    for(j in 1:6){
      xd_mat[[k,j]] = xd_mat[[k,j]] + vSA[j]
    }

  }

}


x = xd_mat
for(i in 1:9){
  row = xd_mat[i,]
  x[i,] = rescale(row, to=c(0,1))

}
XD_b = apply(x, 2, function(x) {ifelse(x > 0, round(x,2), 0)})
XD = rescale(XD_b, to=c(0,0.02))


nodes = data.frame("name" = c("CIP3", "CIP1", "CIP4", "CIP2", "CIP6", "CIP7", "CIP5", "CIP8", "C
IP9", "SA1", "SA2", "SA3", "SA4", "SA5"))
```

```
links = as.data.frame(matrix(c(0,9,  XD[1,1],
                               0,10, XD[1,2],
                               0,12, XD[1,4],
                               0,13, XD[1,5],
                               1,9,  XD[2,1],
                               1,10, XD[2,2],
                               1,12, XD[2,4],
                               1,13, XD[2,5],
                               2,10, XD[3,2],
                               2,11, XD[3,3],
                               2,12, XD[3,4],
                               3,10, XD[4,2],
                               3,11, XD[4,3],
                               3,12, XD[4,4],
                               4,10, XD[5,2],
                               4,12, XD[5,4],
                               4,13, XD[5,5],
                               5,10, XD[6,2],
                               5,11, XD[6,3],
                               5,12, XD[6,4],
                               6,10, XD[6,2],
                               6,12, XD[7,4],
                               6,13, XD[7,5],
                               7,10, XD[8,2],
                               7,12, XD[8,4],
                               8,10, XD[9,2],
                               8,11, XD[9,3],
                               8,12, XD[9,4]
                               ), byrow = TRUE, ncol = 3))
names(links) = c("source", "target", "value")
links$group <- as.factor(c("type_0","type_0","type_0","type_0", "type_1","type_1","type_1","type
_1","type_2", "type_2","type_2","type_3","type_3", "type_3","type_4","type_4","type_4","type_5",
"type_5","type_5","type_6","type_6","type_6","type_7","type_7","type_8","type_8","type_8"))


node_color <- 'd3.scaleOrdinal() .domain(["CIP3", "CIP1", "CIP4", "CIP2", "CIP6", "CIP7", "CIP
5", "CIP8", "CIP9", "SA1", "SA2", "SA3", "SA4", "SA5", "SA6", "type_0", "type_1", "type_2", "typ
e_3", "type_4", "type_5", "type_6", "type_7", "type_8", "type_12"]) .range(["yellow", "red", "da
rkolivegreen" , "orange", "seagreen", "palegreen", "green", "gray", "gray", "red", "orange", "li
ghtgreen", "darkolivegreen", "black", "gray", "yellow", "red", "darkolivegreen" , "orange", "sea
green", "palegreen", "green", "gray", "gray","white"])'


p = sankeyNetwork(Links = links,
                  Nodes = nodes,
                  Source = "source",
                  Target = "target",
                  Value = "value",
                  NodeID = "name",
                  fontSize= 12,
                  nodeWidth = 20,
                  height = 800,
                  width = "100%",
                  colourScale=node_color,
```

```
                LinkGroup="group",
                iterations = 0,
                nodePadding=10)

p
```



Difference Between SA and CIP Coupling Networks (2009-2018)

```
library(dplyr)
library(LICORS)
library(scales)
library(networkD3)

# ArticleLevel-RegData-ALLSA_Xc_1_NData_655386_LONGXCIP2.csv
setwd("D:/Statistical Methods/Project")
data_csv<-read.csv("ArticleLevel-RegData-ALLSA_Xc_1_NData_655386_LONGXCIP2.csv")

# filter out years 2009 to 2018
year_2009_2018<-filter(data_csv, Yp >= 2009 & Yp <= 2018)


# Filter out IRegionRefined
IRegionRefinedp<-filter(year_2009_2018, IRegionRefinedp > 0 & IRegionRefinedp < 4)

# Filter out both NEUROLONGXSAp & NEUROLONGXCIPp
df_mono = year_2009_2018 %>% filter(NEUROLONGXSAp == 0 & NEUROLONGXCIPp == 0)
df_XD = year_2009_2018 %>% filter(NEUROLONGXSAp == 1 & NEUROLONGXCIPp == 1)


mono_mat = matrix(0L, nrow = 9, ncol = 6)
# mono matrix
for(i in 1:nrow(df_mono)){
  row = df_mono[i,]
  vSA = c(row$SA1, row$SA2, row$SA3, row$SA4, row$SA5, row$SA6)
  vCIP = c(row$CIP3, row$CIP1, row$CIP4, row$CIP2, row$CIP6, row$CIP7, row$CIP5, row$CIP8, row$C
IP9)
  vSA = round(vSA / sum(vSA),2)
  for(k in which(vCIP > 0)){
    for(j in 1:6){
      mono_mat[[k,j]] = mono_mat[[k,j]] + vSA[j]
    }

  }

}
print(mono_mat)
```

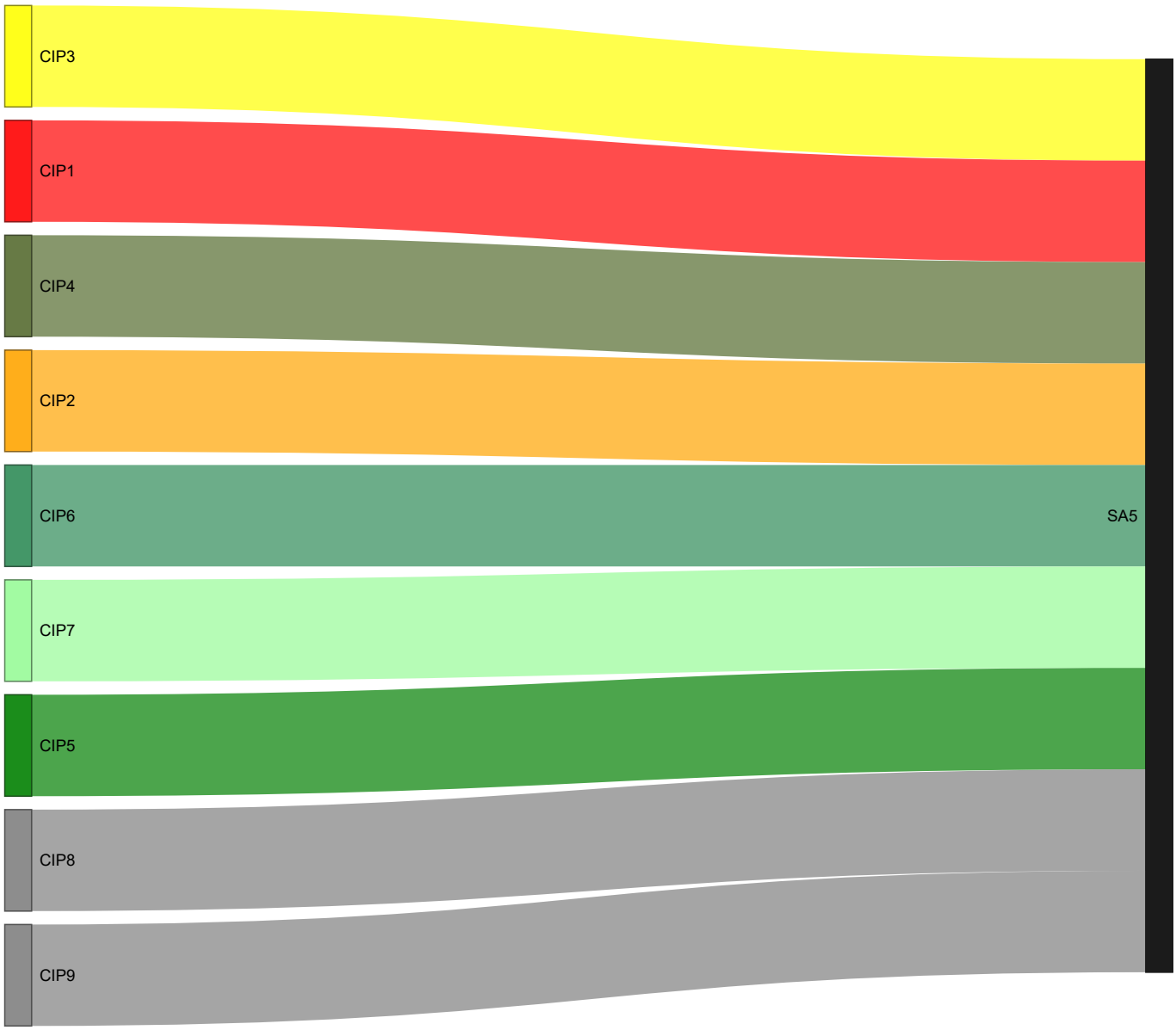
```
##           [,1]     [,2]     [,3]     [,4]    [,5]   [,6]
##  [1,]  6787.17  3618.41  2759.66  3857.22  327.66 119.34
##  [2,] 19934.66 18673.17 11479.54 29265.15 1590.53 432.47
##  [3,]  1275.54  2339.73  2539.63  3785.82  383.36 183.64
##  [4,]  3282.75  9598.81  6297.16 10326.54 1269.26 454.74
##  [5,]  4267.80  7378.05  4826.99 10700.34 1188.72 419.28
##  [6,]  1560.95  3585.80  2471.23  3617.39  329.15  96.85
##  [7,]  6939.61 11540.33  7540.50 25982.80 2206.71 618.29
##  [8,]   856.00  1388.11   811.67  1451.48  624.57 249.43
##  [9,]   573.83  1527.55  1151.15  1889.12  762.46 378.54
```

```
xd_mat = matrix(0L, nrow = 9, ncol = 6)
# xd matrix
for(i in 1:nrow(df_XD)){
  row = df_XD[i,]
  vSA = c(row$SA1, row$SA2, row$SA3, row$SA4, row$SA5, row$SA6)
  vCIP = c(row$CIP3, row$CIP1, row$CIP4, row$CIP2, row$CIP6, row$CIP7, row$CIP5, row$CIP8, row$C
IP9)
  vSA = round(vSA / sum(vSA),2)
  for(k in which(vCIP > 0)){
    for(j in 1:6){
      xd_mat[[k,j]] = xd_mat[[k,j]] + vSA[j]
    }

  }

}
print(xd_mat)
```

```
##         [,1]   [,2]   [,3]   [,4]   [,5]   [,6]
##  [1,] 107.03 103.99 103.94 129.29 270.55  35.11
##  [2,] 311.98 391.72 308.33 562.26 956.83 121.64
##  [3,] 244.03 316.47 287.85 583.69 802.55  94.18
##  [4,]  47.05  73.74  63.37 108.79 183.66  21.71
##  [5,]  66.22  79.42  67.69 130.16 234.92  25.65
##  [6,]  18.37  17.28  24.72  30.24  60.68   5.49
##  [7,] 135.00 300.13 204.11 442.86 710.68 106.56
##  [8,] 191.13 346.64 209.12 321.25 766.94 141.68
##  [9,]   7.35  11.50  17.04  21.38  40.63   4.84
```

```
m = mono_mat
for(i in 1:9){
  row = mono_mat[i,]
  m[i,] = rescale(row, to=c(0,1))

}
mm_b = apply(m, 2, function(x) {ifelse(x > 0.5, round(x,2), 0)})
mm = rescale(mm_b, to=c(0,0.02))


x = xd_mat
for(i in 1:9){
  row = xd_mat[i,]
  x[i,] = sapply(row, function(X) {(X - min(row))/(max(row)-min(row))})
}
XD_b = apply(x, 2, function(x) {ifelse(x > 0.5, round(x,2), 0)})
XD = rescale(XD_b, to=c(0,0.02))




## Diff between Mono and XD
diff_x_m = XD_b - mm_b



## keeping only positive(+) values
diff_x_n = apply(diff_x_m, 2, function(x) {ifelse(x > 0, round(x,2), 0)})
diff_x_mm = rescale(diff_x_n, to=c(0,0.02))

nodes = data.frame("name" = c("CIP3", "CIP1", "CIP4", "CIP2", "CIP6", "CIP7", "CIP5", "CIP8", "C
IP9", "", "", "", "", "SA5", ""))

links = as.data.frame(matrix(c(0,13,  diff_x_mm[1,5],
                               1,13,  diff_x_mm[2,5],
                               2,13,  diff_x_mm[3,5],
                               3,13,  diff_x_mm[4,5],
                               4,13,  diff_x_mm[5,5],
                               5,13,  diff_x_mm[6,5],
                               6,13,  diff_x_mm[7,5],
                               7,13,  diff_x_mm[8,5],
                               8,13,  diff_x_mm[9,5]
), byrow = TRUE, ncol = 3))
names(links) = c("source", "target", "value")
links$group <- as.factor(c("type_0","type_1","type_2","type_3","type_4","type_5","type_6","type_
7","type_8"))
node_color <- 'd3.scaleOrdinal() .domain(["CIP3", "CIP1", "CIP4", "CIP2", "CIP6", "CIP7", "CIP
5", "CIP8", "CIP9", "SA1", "SA2", "SA3", "SA4", "SA5", "SA6", "type_0", "type_1", "type_2", "typ
e_3", "type_4", "type_5", "type_6", "type_7", "type_8", "type_12"]) .range(["yellow", "red", "da
rkolivegreen" , "orange", "seagreen", "palegreen", "green", "gray", "gray", "red", "orange", "li
ghtgreen", "darkolivegreen", "black", "gray", "yellow", "red", "darkolivegreen" , "orange", "sea
green", "palegreen", "green", "gray", "gray","white"])'
```

```
p = sankeyNetwork(Links = links,
                  Nodes = nodes,
                  Source = "source",
                  Target = "target",
                  Value = "value",
                  NodeID = "name",
                  fontSize= 12,
                  nodeWidth = 20,
                  height = 800,
                  width = "100%",
                  colourScale=node_color,
                  LinkGroup="group",
                  iterations = 0,
                  nodePadding=10)

p
```



Conclusion:

Figure 2A showed fractions of articles with cross domain relationships that appeared to increase with time from 1980 -2018.

Figure 2B showed how researchers that collaborated across different disciplines and had a high amount of citations from 1999-2008 appeared to further increase in influence from 2009-2018.

Figure 3A shows specific researcher background disciplines and the fraction of article categories that a specific discipline contributed in research from the years 1970-2018 in 5-year intervals. As the years progressed, researcher disciplines that dominated in publishing articles in certain categories (such as Neuroscience backgrounds publishing Anatomy & Organism category articles in 1970) appeared to lessen to become an overall more balanced fraction of article categories published by any discipline in 2018 which corresponds to increased collaborations happening in the research being done.

Figure 3B Shows the coupling of different research backgrounds with categories of papers published. A mono-domain research article is where just one research background contributes to a published article. The first graph shows mono-domain research teams contributing to certain categories of research articles (SA). The second graph is of cross-domain articles showing researchers that collaborated across domains to publish research in a certain category. The third graph shows the difference between cross-domain and mono-domain articles resulting in showing certain areas of science that have emerged for research disciplines to collaborate in different fields of science.