Problem3

Devin Etcitty

4/25/2017

```r
library(ggplot2)
library(tidyverse)
```

```
## Loading tidyverse: tibble
## Loading tidyverse: tidyr
## Loading tidyverse: readr
## Loading tidyverse: purrr
## Loading tidyverse: dplyr

## Conflicts with tidy packages ----------------------------------------------

## filter(): dplyr, stats
## lag():    dplyr, stats
```

```r
library(scales)
```

```
##
## Attaching package: 'scales'

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

## The following object is masked from 'package:readr':
##
##     col_factor
```

```r
library(igraph)
```

```
##
## Attaching package: 'igraph'

## The following objects are masked from 'package:dplyr':
##
##     %>%, as_data_frame, groups, union

## The following objects are masked from 'package:purrr':
##
##     %>%, compose, simplify

## The following objects are masked from 'package:tidyr':
##
##     %>%, crossing

## The following object is masked from 'package:tibble':
##
##     as_data_frame

## The following objects are masked from 'package:stats':
##
##     decompose, spectrum

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

```
##     union
```

```
library(plyr)
```

```
## -------------------------------------------------------------------------

## You have loaded plyr after dplyr - this is likely to cause problems.
## If you need functions from both plyr and dplyr, please load plyr first, then dplyr:
## library(plyr); library(dplyr)

## -------------------------------------------------------------------------

##
## Attaching package: 'plyr'

## The following objects are masked from 'package:dplyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize

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

```
library(reshape2)
```

```
##
## Attaching package: 'reshape2'

## The following object is masked from 'package:tidyr':
##
##     smiths
```

Read in undweighted10.dat

First, read in the edge list for 2010 from the undweighted10.dat file

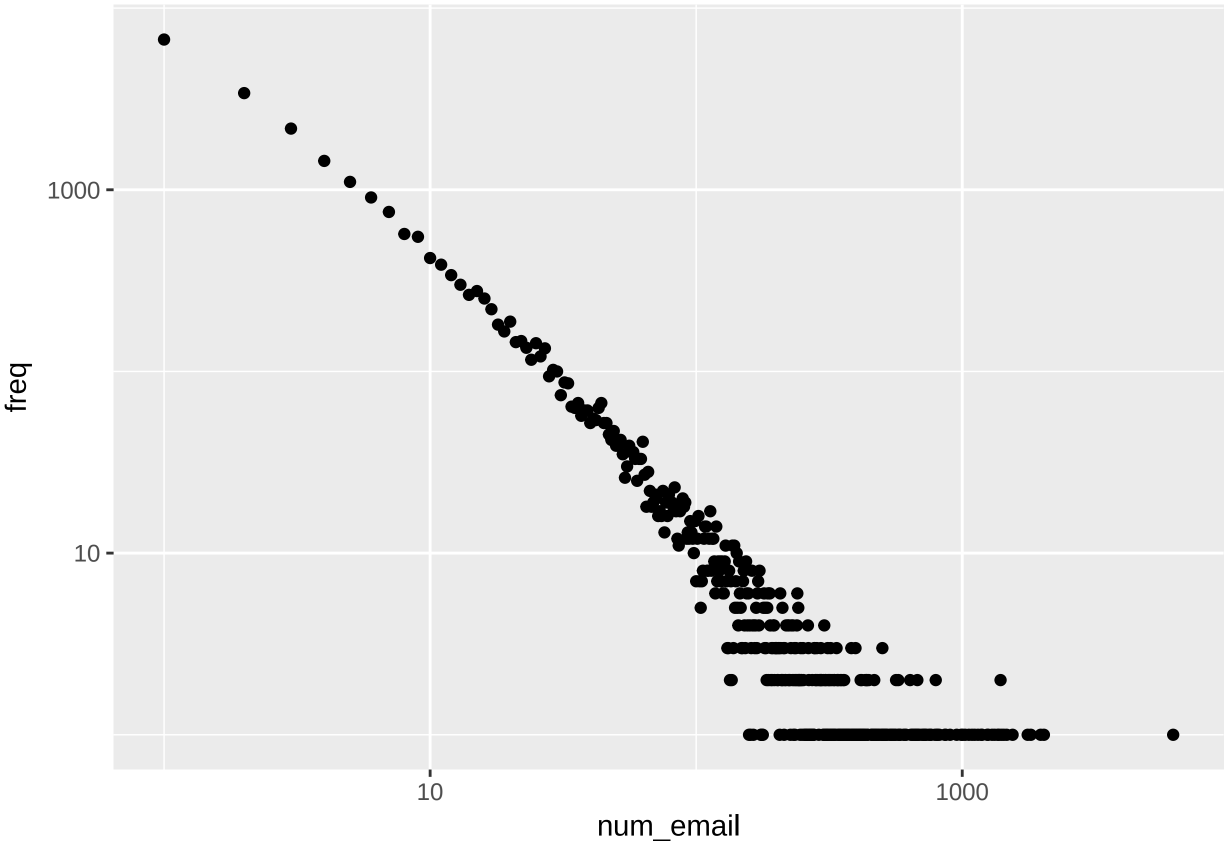
```
#setwd("~/columbia/APMA4990/msd-homework/homework/homework_3/problem_3")
setwd("~/Documents/Columbia/msd-apam4990/msd2017/homework/homework_3/problem_3")
year2010 <- read.delim("undweighted10.dat", header = FALSE, sep = ' ')
colnames(year2010) <- c('userID1', 'userID2', 'num_email')
```

Make a plot of the distribution of edge weights for the entire network Use a log-log scale and comment on the result. What does it tell you about the distribution of tie strength?

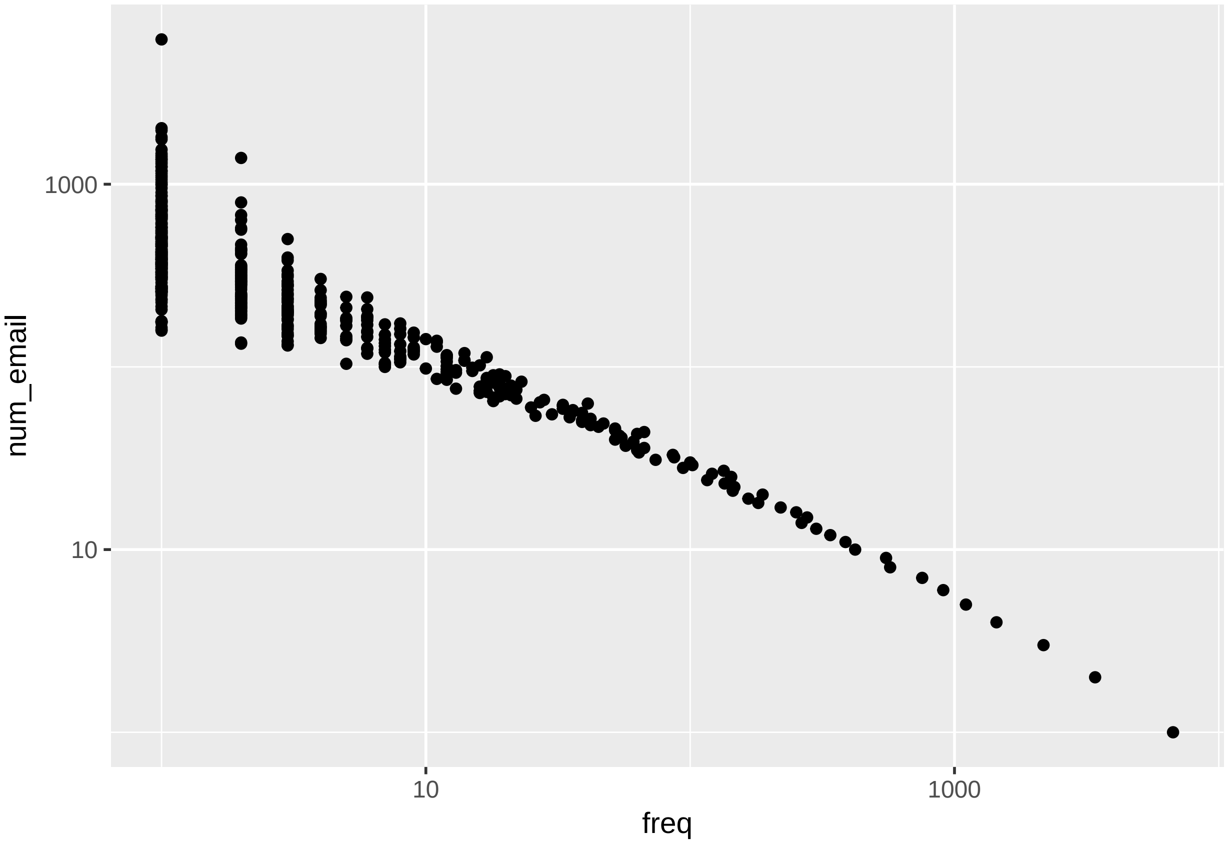
```
count_df <- year2010 %>%
  count('num_email')

#count_df
```

Graph

```
ggplot(count_df, aes(x=num_email, y=freq)) +
  geom_point() +
  scale_x_log10() +
  scale_y_log10()
```



```r
ggplot(count_df, aes(x=freq, y =num_email)) +
  geom_point() +
  scale_x_log10() +
  scale_y_log10()
```



```
power2 <- 2^(0:10)
power2 <- append(power2, 0, 0)
power2
```

```
##  [1]    0    1    2    4    8   16   32   64  128  256  512 1024
```

```
final_df  <- data.frame(nodes=integer(),
                 edges=integer(),
                 num_connected_components=integer(),
                 percent_of_nodes_in_max_component=double(),
                 ave_path_distance=double())

for(i in power2) {
  df <- year2010 %>%
    filter(num_email > i)

  g <- graph_from_data_frame(df, directed = FALSE, vertices = NULL)

  nodes <- gorder(g)
  edges <- gsize(g)

  num_connect_comps <- no.clusters(g)
  clus <- clusters(g)

  test <- groups(clus)

  d <- adply(test, '1', length)
  max_num_nodes_comp <- max(d$V1)

  percent_connect_comp <- max_num_nodes_comp / nodes

  avg_distance <- mean_distance(g)

  results = c(nodes, edges, num_connect_comps, percent_connect_comp, avg_distance)

  final_df <- rbind(final_df, i = results)

}
```

```
colnames(final_df) <- c("nodes", "edges", "num_connect_comps",
                     "percent_connect_comp", "avg_distance")
```

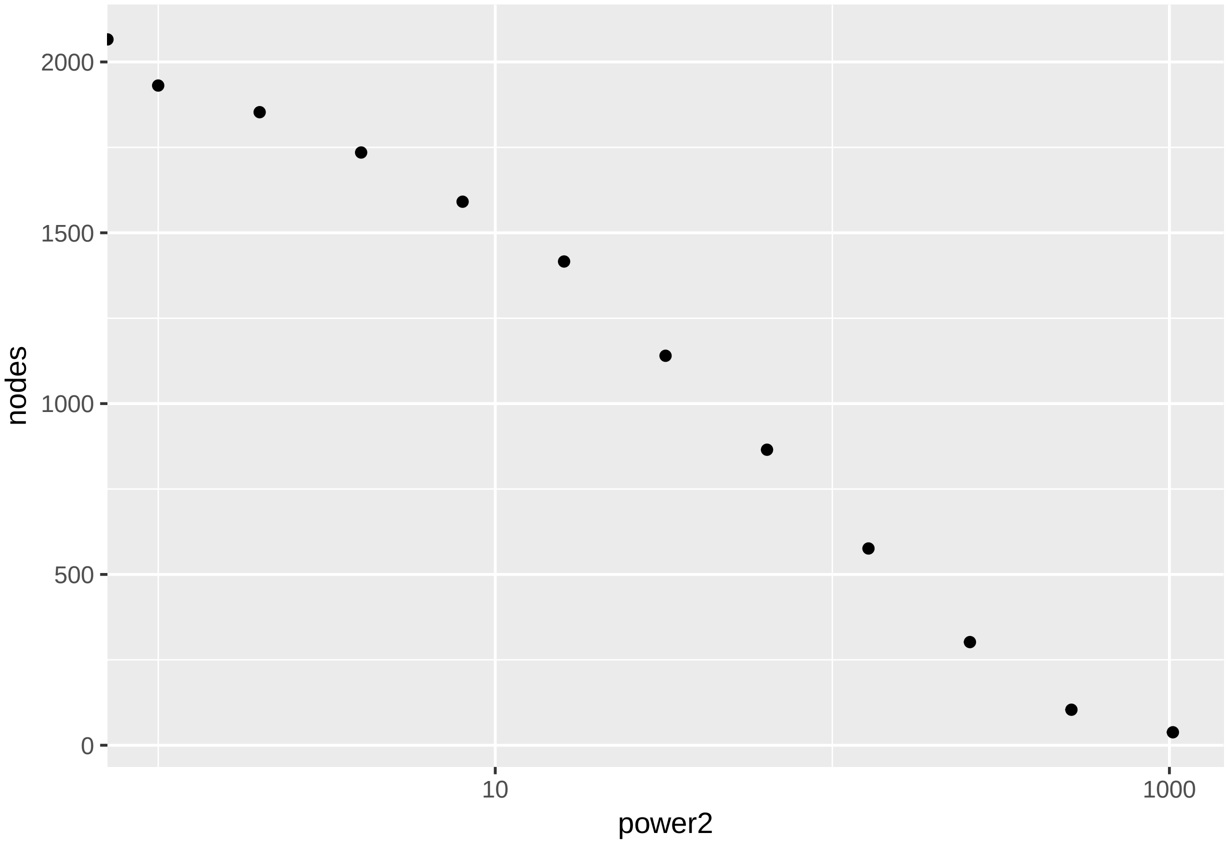
```
final_df <- cbind(power2, final_df)
```

```
final_df
```

```
##     power2 nodes edges num_connect_comps percent_connect_comp avg_distance
## i        0  2066 25124                 2            0.9990319     3.053320
## i1       1  1931 18410                 3            0.9979285     3.241222
## i2       2  1853 15005                 2            0.9989207     3.391726
## i3       4  1735 11391                 4            0.9965418     3.585476
## i4       8  1591  8053                 5            0.9949717     3.871058
## i5      16  1416  5262                 6            0.9929379     4.287107
## i6      32  1140  3109                 8            0.9824561     4.941018
## i7      64   865  1607                22            0.9317919     6.610826
## i8     128   576   729                45            0.7048611     9.447853
## i9     256   302   276                66            0.1059603     3.088959
## i10    512   104    79                31            0.1057692     1.873016
## i11   1024    38    24                14            0.1315789     1.368421
```

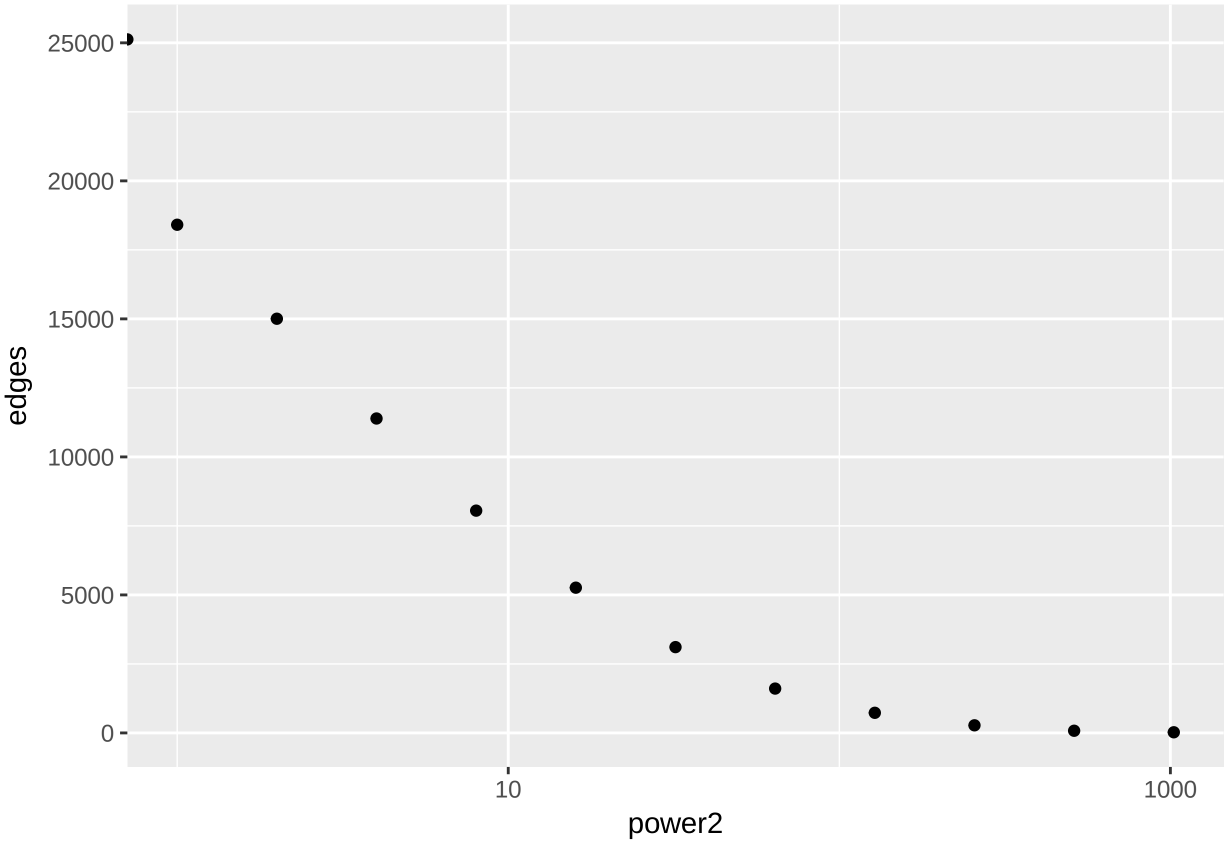
```
ggplot(final_df, aes(x=power2, y=nodes)) +
  scale_x_log10() +
  geom_point()
```

```
## Warning: Transformation introduced infinite values in continuous x-axis
```



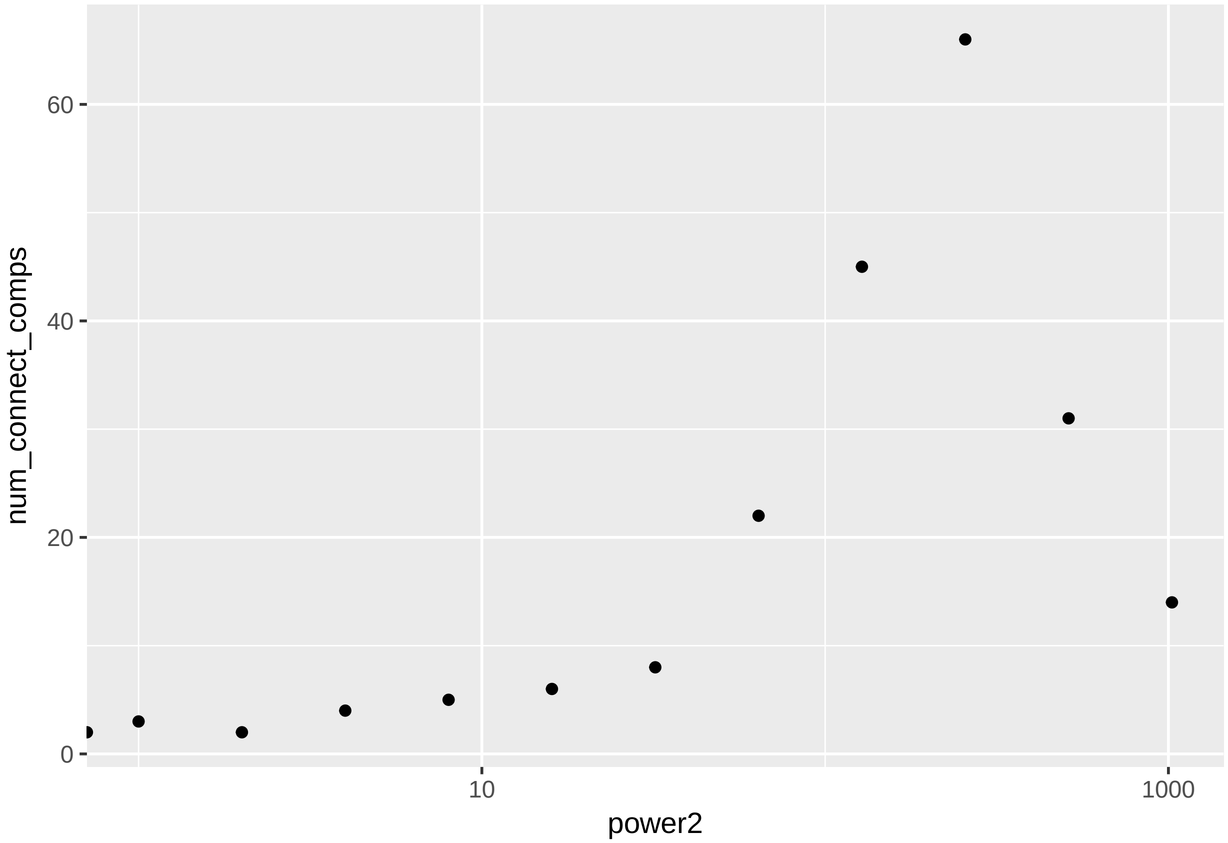
```
ggplot(final_df, aes(x=power2, y=edges)) +
  scale_x_log10() +
  geom_point()
```

```
## Warning: Transformation introduced infinite values in continuous x-axis
```



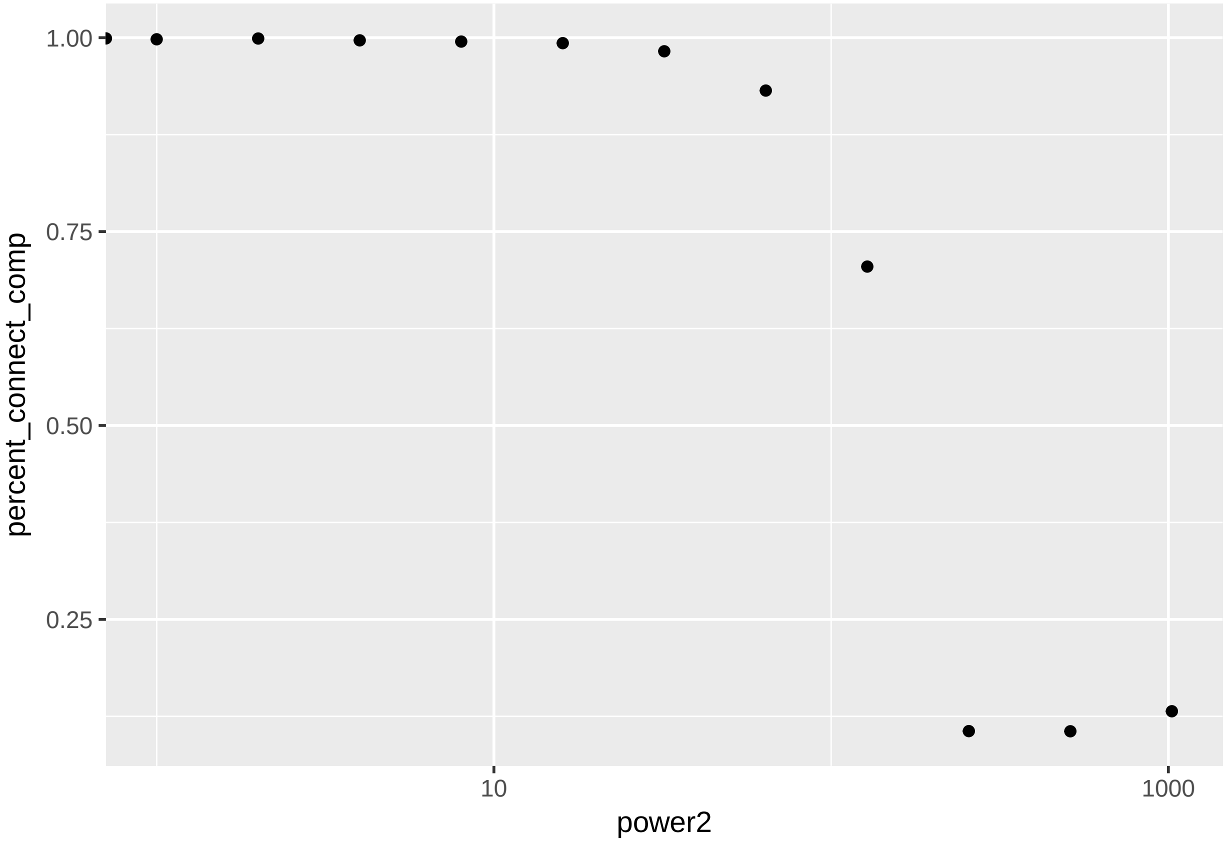
```
ggplot(final_df, aes(x=power2, y=num_connect_comps)) +
  scale_x_log10() +
  geom_point()
```

```
## Warning: Transformation introduced infinite values in continuous x-axis
```



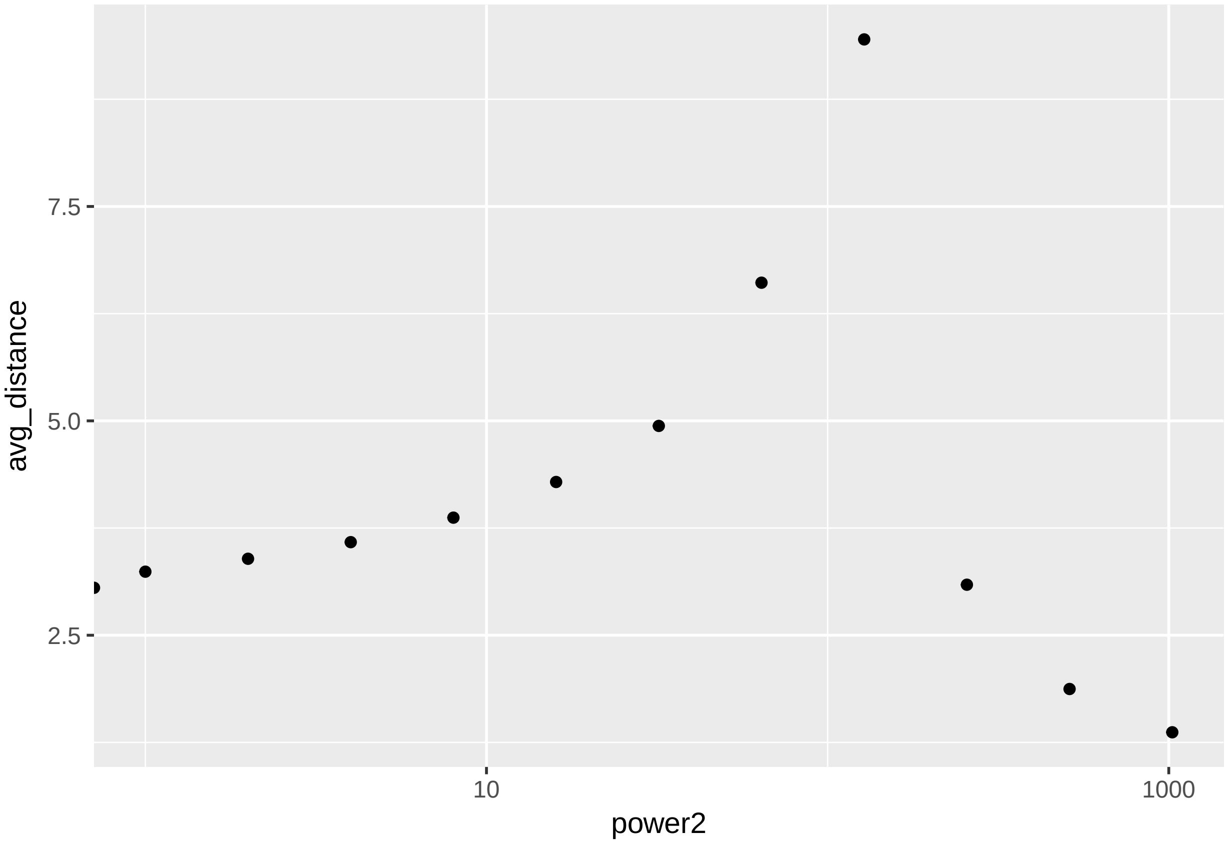
```
ggplot(final_df, aes(x=power2, y=percent_connect_comp)) +
  scale_x_log10() +
  geom_point()
```

```
## Warning: Transformation introduced infinite values in continuous x-axis
```



```
ggplot(final_df, aes(x=power2, y=avg_distance)) +
  scale_x_log10() +
  geom_point()
```

```
## Warning: Transformation introduced infinite values in continuous x-axis
```


avg_distance
7.5
5.0
2.5
10
1000
power2