visualization_finance_example

data visualization practice

```
inpatientCharges$provider_zip_code <- as.character(inpatientCharges$provider_zip_code)
replace_missings <- function(x, replacement) {
  is_miss <- is.na(x)
  x[is_miss] <- replacement

  message(sum(is_miss), " missings replaced by the value ", replacement)
  x
}
reorder_size <- function(x) {
       factor(x, levels = names(sort(table(x), decreasing = TRUE)))
}
# oduble checking for missing data
sapply(inpatientCharges, function(x) sum(length(which(is.na(x)))))
```

```
##                       drg_definition                          provider_id
##                                    0                                    0
##                        provider_name              provider_street_address
##                                    0                                    0
##                        provider_city                       provider_state
##                                    0                                    0
##                    provider_zip_code hospital_referral_region_description
##                                    0                                    0
##                     total_discharges              average_covered_charges
##                                    0                                    0
##               average_total_payments            average_medicare_payments
##                                    0                                    0
```

```
num <- sapply(inpatientCharges, is.numeric)
colnames(inpatientCharges[num]) # which numeric vars
```

```
## [1] "total_discharges"          "average_covered_charges"
## [3] "average_total_payments"    "average_medicare_payments"
```

```
Hmisc::describe(inpatientCharges[num])
```

```
## inpatientCharges[num]
##
##  4  Variables      163065  Observations
## ---------------------------------------------------------------------------
## total_discharges
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##   163065        0      642    0.999    42.78    38.87       12       13
##      .25      .50      .75      .90      .95
##       17       27       49       88      123
##
## lowest :   11   12   13   14   15, highest: 1464 1487 1571 1696 3383
## ---------------------------------------------------------------------------
## average_covered_charges
##        n  missing distinct     Info     Mean      Gmd      .05      .10
```

```
##   163065        0   160236        1    36134    30304     9250    11277
##      .25      .50      .75      .90      .95
##    15947    25246    43233    72319    98723
##
## lowest :   2459.40   2521.37   2536.69   2701.72   2749.94
## highest: 613926.60 628730.40 637377.71 918023.18 929118.90
## ---------------------------------------------------------------------------
## average_total_payments
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##   163065        0   147842        1     9707     6703     3863     4234
##      .25      .50      .75      .90      .95
##     5234     7214    11286    17280    23927
##
## lowest :   2673.00   2682.64   2707.87   2708.00   2718.27
## highest: 119028.90 119113.00 131187.35 140255.26 156158.18
## ---------------------------------------------------------------------------
## average_medicare_payments
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##   163065        0   150328        1     8494     6410     2859     3223
##      .25      .50      .75      .90      .95
##     4192     6158    10057    15696    21775
##
## lowest :   1148.90   1327.23   1400.57   1594.50   1603.83
## highest: 109303.21 113462.09 130466.57 133177.26 154620.81
## ---------------------------------------------------------------------------
```

```
psych::describe(inpatientCharges[num])
```

```
##                           vars      n     mean       sd   median  trimmed
## total_discharges             1 163065    42.78    51.10    27.00    32.84
## average_covered_charges      2 163065 36133.95 35065.37 25245.82 29506.16
## average_total_payments       3 163065  9707.47  7664.64  7214.10  8181.31
## average_medicare_payments    4 163065  8494.49  7309.47  6158.46  7057.83
##                              mad    min      max    range skew kurtosis
## total_discharges            19.27   11.0   3383.0   3372.0 7.67   188.45
## average_covered_charges   16864.13 2459.4 929118.9 926659.5 4.03    31.72
## average_total_payments     3723.98 2673.0 156158.2 153485.2 3.33    17.31
## average_medicare_payments  3688.26 1148.9 154620.8 153471.9 3.36    17.95
##                              se
## total_discharges           0.13
## average_covered_charges   86.84
## average_total_payments    18.98
## average_medicare_payments 18.10
```

```
head(inpatientCharges, 2) # looking at head of inpatientCharges
```

```
## # A tibble: 2 x 12
##                            drg_definition provider_id
##                                     <chr>       <chr>
## 1 039 - EXTRACRANIAL PROCEDURES W/O CC/MCC       10001
## 2 039 - EXTRACRANIAL PROCEDURES W/O CC/MCC       10005
## # ... with 10 more variables: provider_name <chr>,
## #   provider_street_address <chr>, provider_city <chr>,
## #   provider_state <chr>, provider_zip_code <chr>,
## #   hospital_referral_region_description <chr>, total_discharges <int>,
```

```
## #   average_covered_charges <dbl>, average_total_payments <dbl>,
## #   average_medicare_payments <dbl>
```

```
hospital_state_max_count_df <- inpatientCharges %>%
  separate(hospital_referral_region_description, c("hospital_state", "hospital_city"), sep = " - ") %>%
  group_by(hospital_state) %>%
  summarize(max_covered = max(average_covered_charges), count = n(), max_disch = max(total_discharges))
```

```
inpatientCharges.tidy <-
  inpatientCharges %>%
  separate(hospital_referral_region_description, c("hospital_state", "hospital_city"), sep = " - ", rem
  separate(drg_definition, c("drg_code", "drg_description"), sep = " - ", remove = F)
```

```
library(noncensus)
data(states) # data from noncensus package
hospital_states <- states %>% select(-capital)
colnames(hospital_states) <- c("hospital_state", "hospital_state_name","hospital_state_region", "hospita
```

```
psych::describeBy(group = inpatientCharges.tidy1$hospital_state_region, x=inpatientCharges.tidy1$averag
```

```
##
##  Descriptive statistics by group
## group: Midwest
##    vars     n    mean      sd  median trimmed     mad     min      max
## X1    1 38814 7996.75 6582.44 5853.98 6701.83 3410.18 1327.23 130466.6
##       range skew kurtosis    se
## X1 129139.3  3.1    15.19 33.41
## ------------------------------------------------------------
## group: Northeast
##    vars     n    mean      sd  median trimmed     mad     min      max
## X1    1 30359 9351.39 8165.83 6808.38 7744.44 4067.66 1679.76 133177.3
##       range skew kurtosis    se
## X1 131497.5 3.36    17.08 46.87
## ------------------------------------------------------------
## group: South
##    vars     n    mean      sd  median trimmed     mad    min      max
## X1    1 67423 7743.27 6486.34 5642.11 6453.54 3281.96 1148.9 95701.42
##       range skew kurtosis    se
## X1 94552.52 3.12    14.27 24.98
## ------------------------------------------------------------
## group: West
##    vars     n     mean      sd  median trimmed     mad     min      max
## X1    1 26469 10155.09 8750.43 7448.66 8454.11 4411.98 1603.83 154620.8
##     range skew kurtosis    se
## X1 153017 3.38    17.97 53.78
```

```
inpatientCharges.tidy.v <- newcat %>% right_join(inpatientCharges.tidy1,by = "drg_definition")
```

```
library(ggsci) # ggsci because of the futurama color scheme
```

```
## Warning: package 'ggsci' was built under R version 3.4.2
```

```
theme_set(theme_bw()) # themeset bw
library(ggplot2)
library(gridExtra)
```

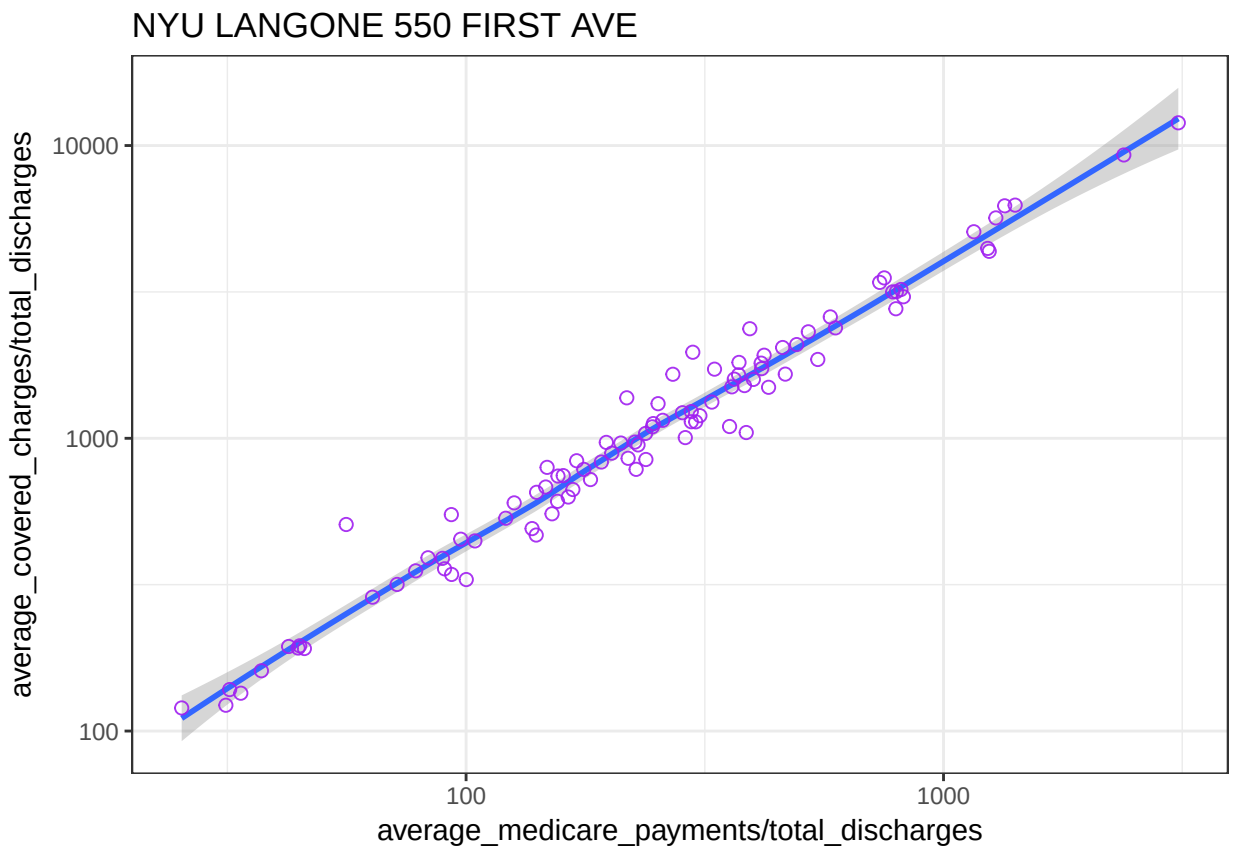
```
##
```

```
## Attaching package: 'gridExtra'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
inpatientCharges.tidy.v  %>%
  filter(provider_street_address=="550 FIRST AVENUE") %>%
  ggplot(aes(x = average_medicare_payments/total_discharges, y = average_covered_charges/total_discharg
  geom_smooth() +
   geom_point(alpha=0.9, pch=21, col="purple", size=2) +
  scale_color_futurama() + scale_x_log10() + scale_y_log10() +
  ggtitle("NYU LANGONE 550 FIRST AVE")
```

```
## `geom_smooth()` using method = 'loess'
```



Extremely busy plot/ not helpful

```
head(hospital_state_max_count_df,2) #looking at hospital_state_max_count_df
```

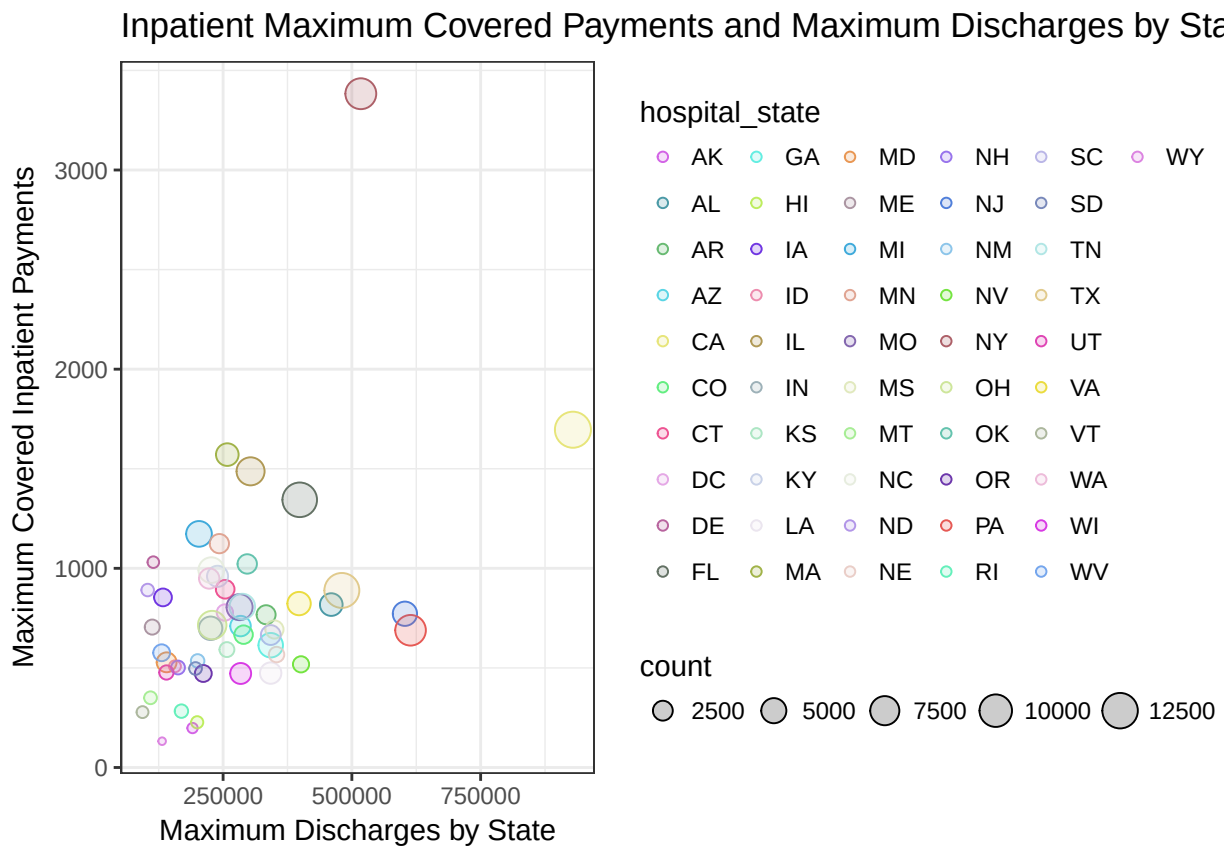
```
## # A tibble: 2 x 4
##   hospital_state max_covered count max_disch
##            <chr>       <dbl> <int>     <dbl>
## 1             AK    190709.9   231       198
## 2             AL    459985.3  3717       818
```

```
library(randomcoloR) # loading randomcolorR
```

```
## Warning: package 'randomcoloR' was built under R version 3.4.3
```

```
n <- 51
palette_51 <- distinctColorPalette(n) # creating 51 random color palettes
```

```
ggplot(hospital_state_max_count_df, aes(x = max_covered, y = max_disch, size = count, col=hospital_state
  geom_point(alpha=0.2) + ggtitle("Inpatient Maximum Covered Payments and Maximum Discharges by State")
  scale_color_manual(values=palette_51)+ guides(col = guide_legend(nrow = 10
                                                                   ), size=guide_legend(nrow=1)) +
  labs(x="Maximum Discharges by State", y="Maximum Covered Inpatient Payments")
```



looking at MCC / CC

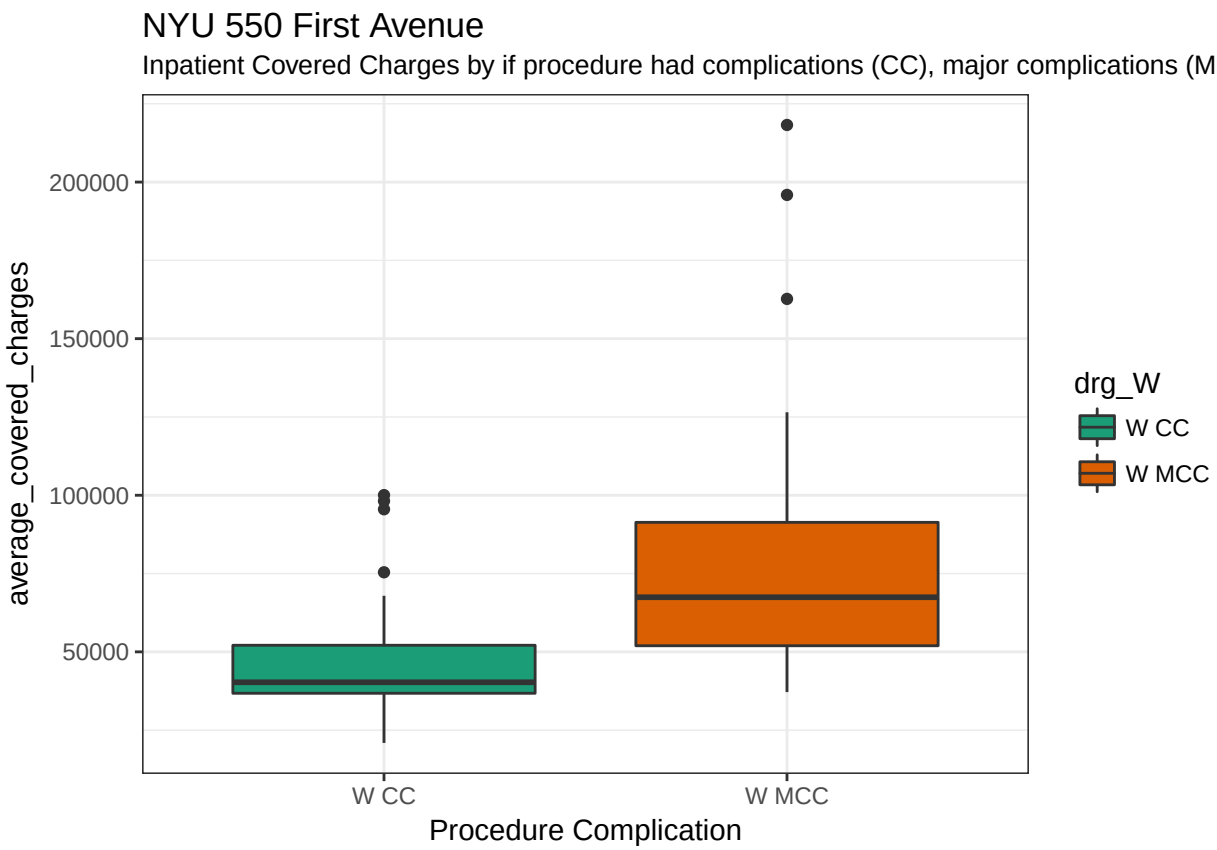
```
colnames(inpatientCharges.tidy.v)
```

```
##  [1] "drg_W"
##  [2] "drg_definition"
##  [3] "drg_code"
##  [4] "drg_description"
##  [5] "provider_id"
##  [6] "provider_name"
##  [7] "provider_street_address"
##  [8] "provider_city"
##  [9] "provider_state"
## [10] "provider_zip_code"
## [11] "hospital_referral_region_description"
## [12] "hospital_state"
## [13] "hospital_city"
## [14] "total_discharges"
## [15] "average_covered_charges"
```

```
## [16] "average_total_payments"
## [17] "average_medicare_payments"
## [18] "hospital_state_name"
## [19] "hospital_state_region"
## [20] "hospital_state_division"
## [21] "hospital_state_area"
## [22] "hospital_state_population"
## [23] "hospital_pop_area"
```

```
inpatient_region_total_discharges_summary <- psych::describeBy(group= inpatientCharges.tidy.v$hospital_s
inpatient_region_average_medicare_payments_summary <- psych::describeBy(group= inpatientCharges.tidy.v$
inpatient_region_average_medicare_payments_summary <- psych::describeBy(group= inpatientCharges.tidy.v$
```

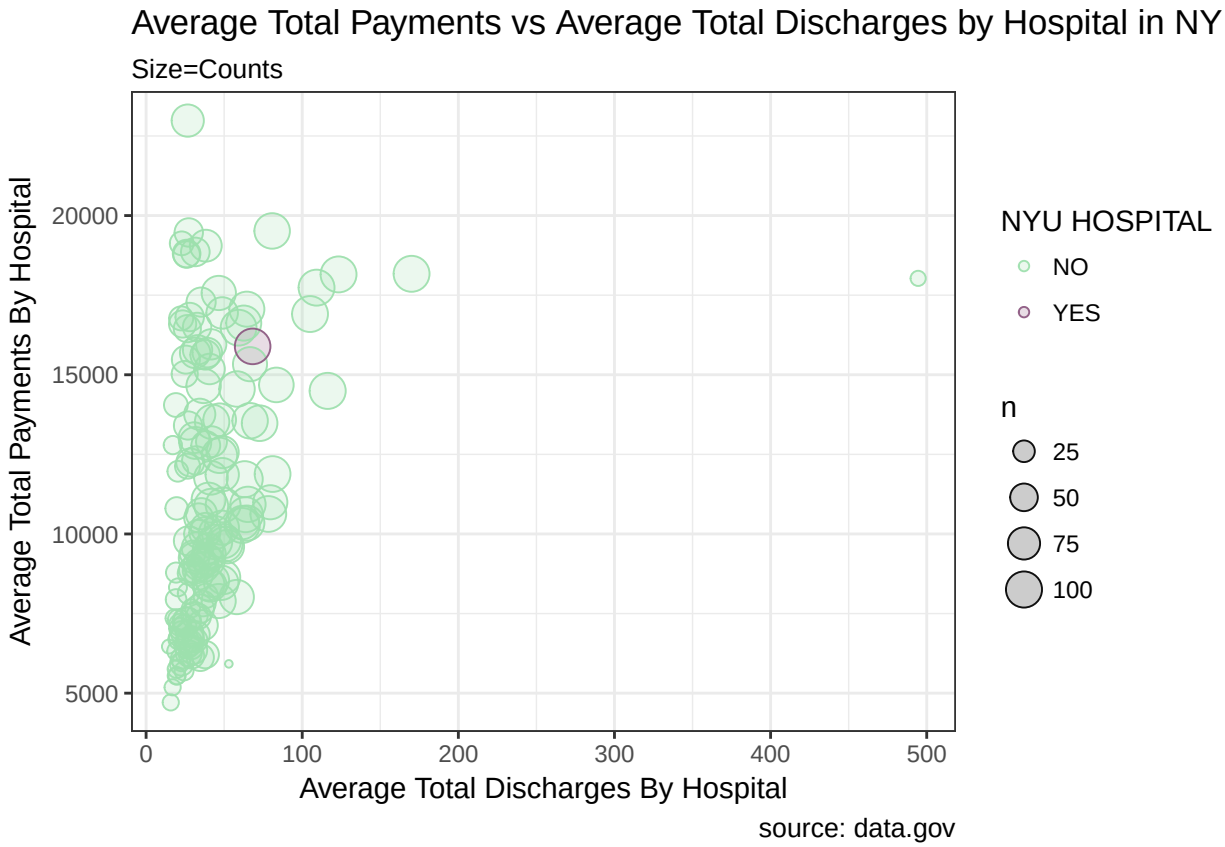
```
df.t <- inpatientCharges.tidy.v
df.t %>% filter(provider_street_address=="550 FIRST AVENUE") %>%
  filter(drg_W == "W MCC" | drg_W=="W CC") %>%
  ggplot(aes(x=drg_W, y=average_covered_charges, fill=drg_W)) + geom_boxplot()+ scale_fill_brewer(palet
  labs(title="NYU 550 First Avenue",
subtitle="Inpatient Covered Charges by if procedure had complications (CC), major complications (MCC)",
```



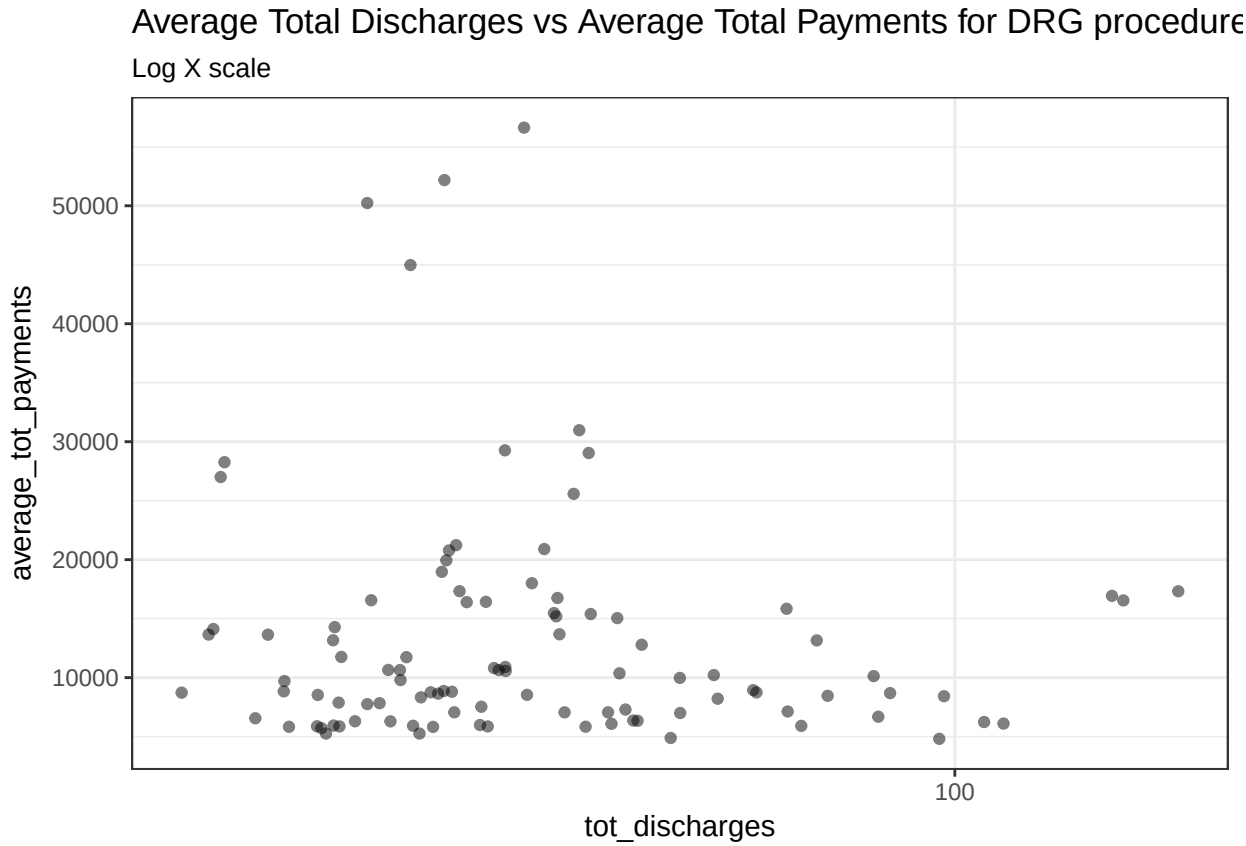
If procedure had MCC vs W CC, then higher covered charges

```
part1 <- inpatientCharges.tidy.v%>% filter(hospital_state=="NY")  %>%
  select(hospital_state, drg_code, provider_name)
inpatientCharges.tidy.v %>% filter(hospital_state=="NY") %>% group_by(provider_name) %>% summarize(tot_
  geom_count(alpha=0.2) +geom_count(alpha=0.92, pch=21) +
    scale_color_manual(values=c("#9DE0AD","#8C5881")) +
  labs(title="Average Total Payments vs Average Total Discharges by Hospital in NY",
       subtitle="Size=Counts", caption="source: data.gov",
```

```
      y="Average Total Payments By Hospital",
      x="Average Total Discharges By Hospital")
```




```
inpatientCharges.tidy.v %>% filter(hospital_state=="NY") %>% group_by(drg_code) %>% summarize(tot_discha
  geom_point(alpha=0.5) +
  scale_x_log10() +
  labs(title="Average Total Discharges vs Average Total Payments for DRG procedure in NY",
       subtitle="Log X scale")
```




```
df.manhattan.inpatient.expenses<- df.t %>% filter(hospital_referral_region_description== "NY - Manhattan
df.manhattan.inpatient.expenses %>% head(2)
```

```
## # A tibble: 2 x 23
##        drg_W                            drg_definition drg_code
##        <chr>                                     <chr>    <chr>
## 1 W/O CC/MCC 039 - EXTRACRANIAL PROCEDURES W/O CC/MCC      039
## 2 W/O CC/MCC 039 - EXTRACRANIAL PROCEDURES W/O CC/MCC      039
## # ... with 20 more variables: drg_description <chr>, provider_id <chr>,
## #   provider_name <chr>, provider_street_address <chr>,
## #   provider_city <chr>, provider_state <chr>, provider_zip_code <chr>,
## #   hospital_referral_region_description <chr>, hospital_state <chr>,
## #   hospital_city <chr>, total_discharges <int>,
## #   average_covered_charges <dbl>, average_total_payments <dbl>,
## #   average_medicare_payments <dbl>, hospital_state_name <chr>,
## #   hospital_state_region <fctr>, hospital_state_division <fctr>,
## #   hospital_state_area <dbl>, hospital_state_population <dbl>,
## #   hospital_pop_area <dbl>
```

```
multi.fun(df.manhattan.inpatient.expenses$provider_city)
```

```
##                freq percentage
## BROOKLYN        877  47.975930
## LONG BEACH       42   2.297593
## NEW YORK        746  40.809628
## STATEN ISLAND   163   8.916849
```

```
numeric.manhattan <- sapply(df.manhattan.inpatient.expenses, is.numeric)
df.manhat.cat <- df.manhattan.inpatient.expenses[!numeric.manhattan]
df.manhat.cat %>% select(provider_id, provider_name) %>% head(2)
```

```
## # A tibble: 2 x 2
##   provider_id                     provider_name
##         <chr>                             <chr>
## 1      330024               MOUNT SINAI HOSPITAL
## 2      330028 RICHMOND UNIVERSITY MEDICAL CENTER
```

```
df.manhattan.inpatient.expenses$NYU=factor(ifelse(df.manhattan.inpatient.expenses$provider_name=="NYU H

df.manhattan.inpatient.expenses$withcomplicationsorwithout <- gsub("W CC/MCC", "W COMPLICATIONS", df.ma
df.manhattan.inpatient.expenses$withcomplicationsorwithout <- gsub("W MCC", "W COMPLICATIONS", df.manha
df.manhattan.inpatient.expenses$withcomplicationsorwithout <- gsub("W CC", "W COMPLICATIONS", df.manhat
df.manhattan.inpatient.expenses$withcomplicationsorwithout <- gsub("W/O CC/MCC", "W/O COMPLICATIONS", d
df.manhattan.inpatient.expenses$withcomplicationsorwithout <- gsub("W/O MCC", "W/O COMPLICATIONS", df.m
df.manhattan.inpatient.expenses$withcomplicationsorwithout <- gsub("WNA", "DOES NOT APPLY TO PROCEDURE"
multi.fun(df.manhattan.inpatient.expenses$drg_W)
```

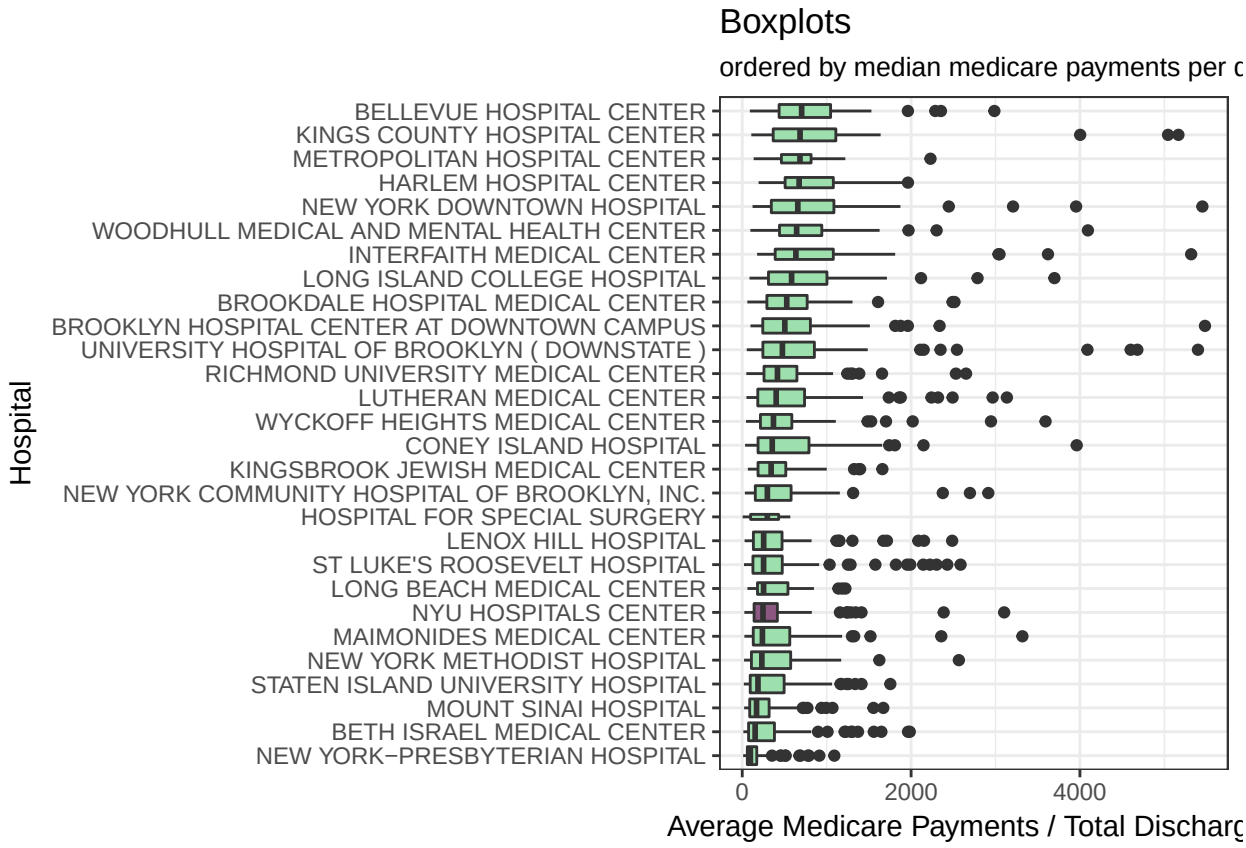
```
##            freq percentage
## W CC        409  22.374179
## W CC/MCC     22   1.203501
## W MCC       453  24.781182
## W/O CC/MCC  275  15.043764
## W/O MCC     471  25.765864
## WNA         198  10.831510
```

```
multi.fun(df.manhattan.inpatient.expenses$withcomplicationsorwithout)
```

```
##                             freq percentage
## DOES NOT APPLY TO PROCEDURE  198   10.83151
## W COMPLICATIONS              884   48.35886
## W/O COMPLICATIONS            746   40.80963
```

How does NYU compare to other hospitals inpatient expenses?

```
df.manhattan.inpatient.expenses %>% mutate(avg_med_pay_per_disch = round(average_medicare_payments/total
    scale_alpha_manual(values=c(0.2,0.1)) +
    scale_fill_manual(values=c("#9DE0AD","#8C5881")) +
  labs(x="Hospital", y="Average Medicare Payments / Total Discharges",
       title = "Boxplots",
       subtitle=
         "ordered by median medicare payments per discharge") +
   theme(legend.position = "none")
```

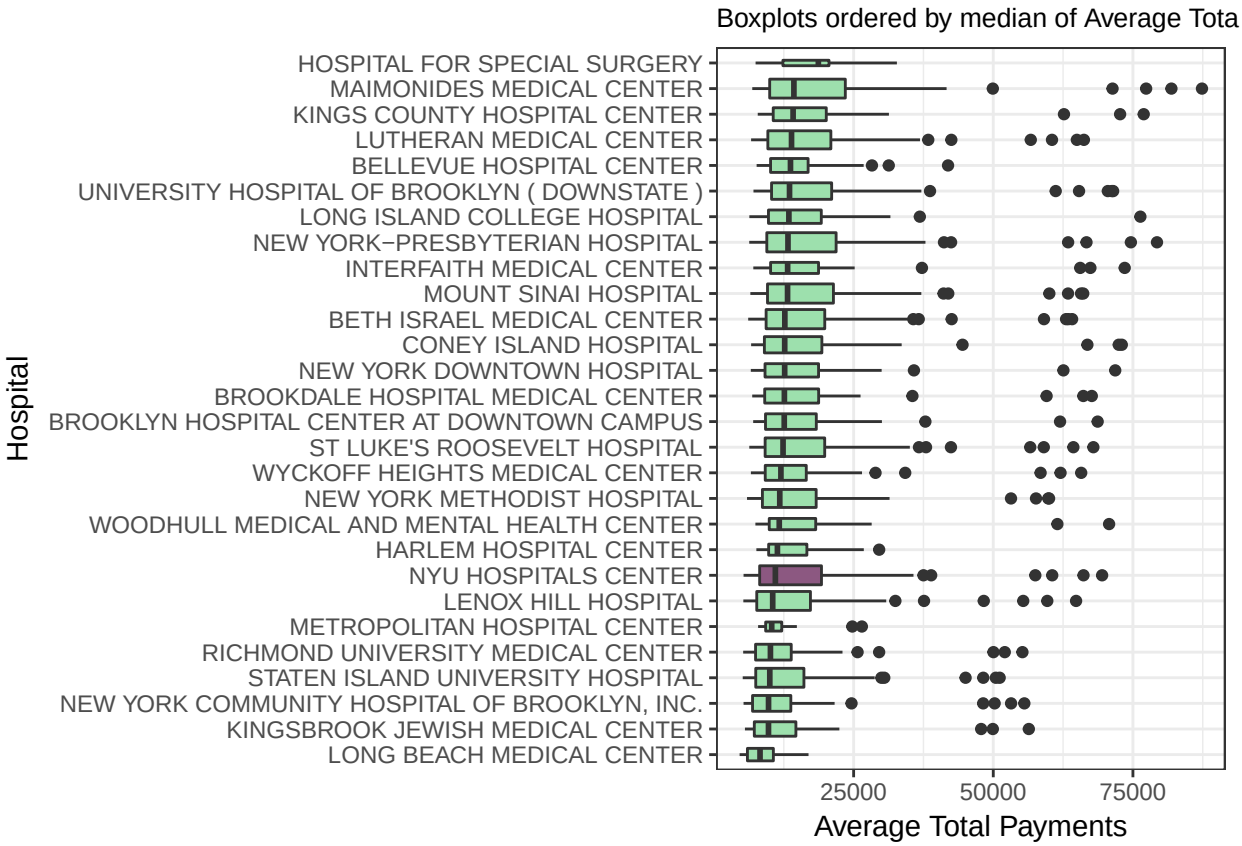



```
df.manhattan.inpatient.expenses %>% group_by(provider_name) %>% summarize(count=n(), average_disch = ro
```

```
## # A tibble: 2 x 3
##                provider_name count average_disch
##                        <chr> <int>         <dbl>
## 1   BELLEVUE HOSPITAL CENTER    52            25
## 2 BETH ISRAEL MEDICAL CENTER    99           105
```

```
# %>% left_join(df.manhattan.inpatient.expenses)
```

```
df.manhattan.inpatient.expenses %>% ggplot(aes(x=reorder(provider_name, average_total_payments, FUN=medi
    scale_alpha_manual(values=c(0.2,0.1)) +
    scale_fill_manual(values=c("#9DE0AD","#8C5881")) +
  labs(x="Hospital", y="Average Total Payments",
       subtitle=
         "Boxplots ordered by median of Average Total Payments") +
  theme(legend.position = "none")
```

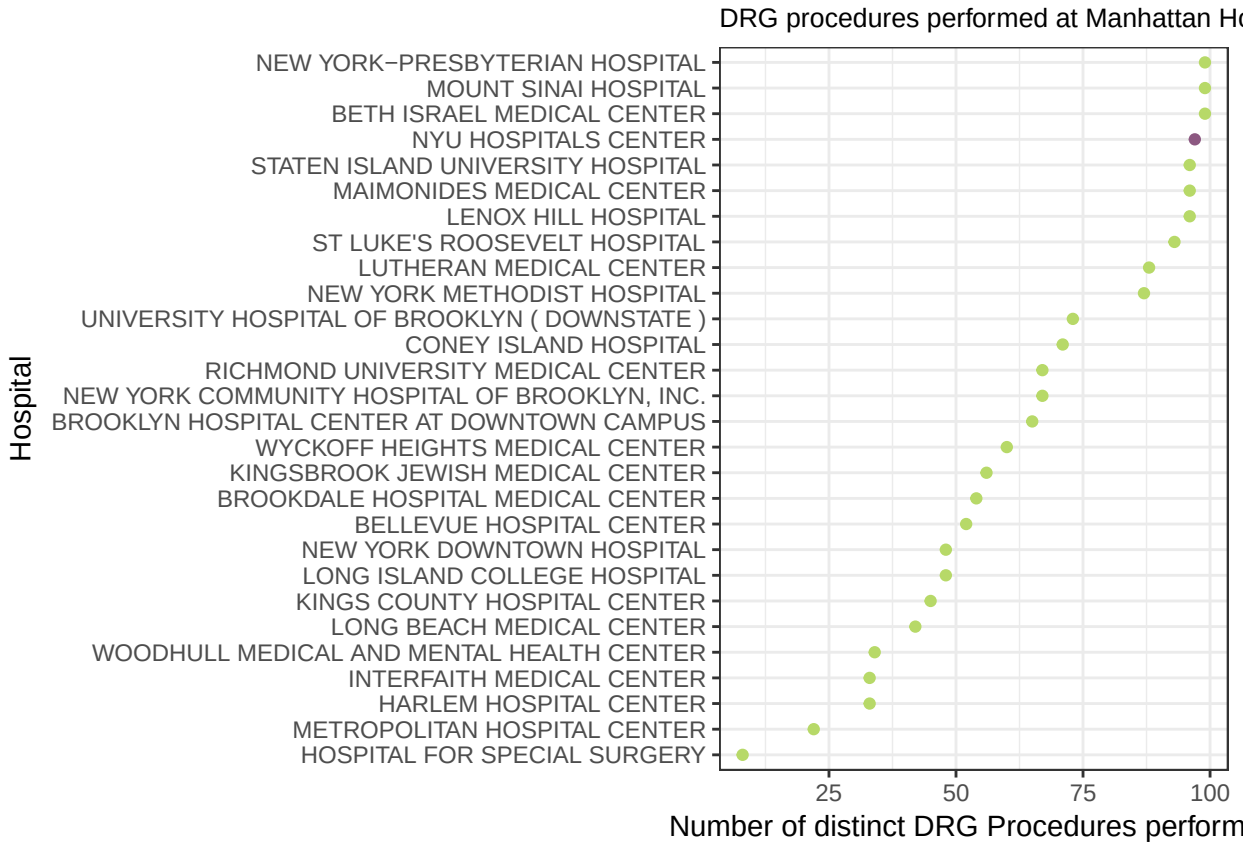



```
describeBy_total_discarges <- psych::describeBy(df.manhattan.inpatient.expenses$total_discharges, df.ma
 NYU.TOTAL.DISCHARGES <- data.frame(describeBy_total_discarges$`NYU HOSPITALS CENTER`)
 rownames(NYU.TOTAL.DISCHARGES) <- c("total.discharges.NYU")
 numeric <- sapply(df.t, is.numeric)
df.manhattan.inpatient.expenses %>% select(total_discharges, average_covered_charges, average_medicare_
```

```
## # A tibble: 1,828 x 5
##    total_discharges average_covered_charges average_medicare_payments
##               <int>                   <dbl>                     <dbl>
##  1               29                22169.41                   9504.37
##  2               12                32246.66                   9321.33
##  3               49                32142.36                   9417.73
##  4               17                29515.52                   7907.64
##  5               18                21700.11                   8458.88
##  6               17                31149.82                  10065.58
##  7               11                28410.00                  10957.72
##  8               33                37047.06                   8141.36
##  9               16                22077.12                   9091.00
## 10               21                20655.14                   9007.47
## # ... with 1,818 more rows, and 2 more variables:
## #   average_total_payments <dbl>, provider_name <chr>
```

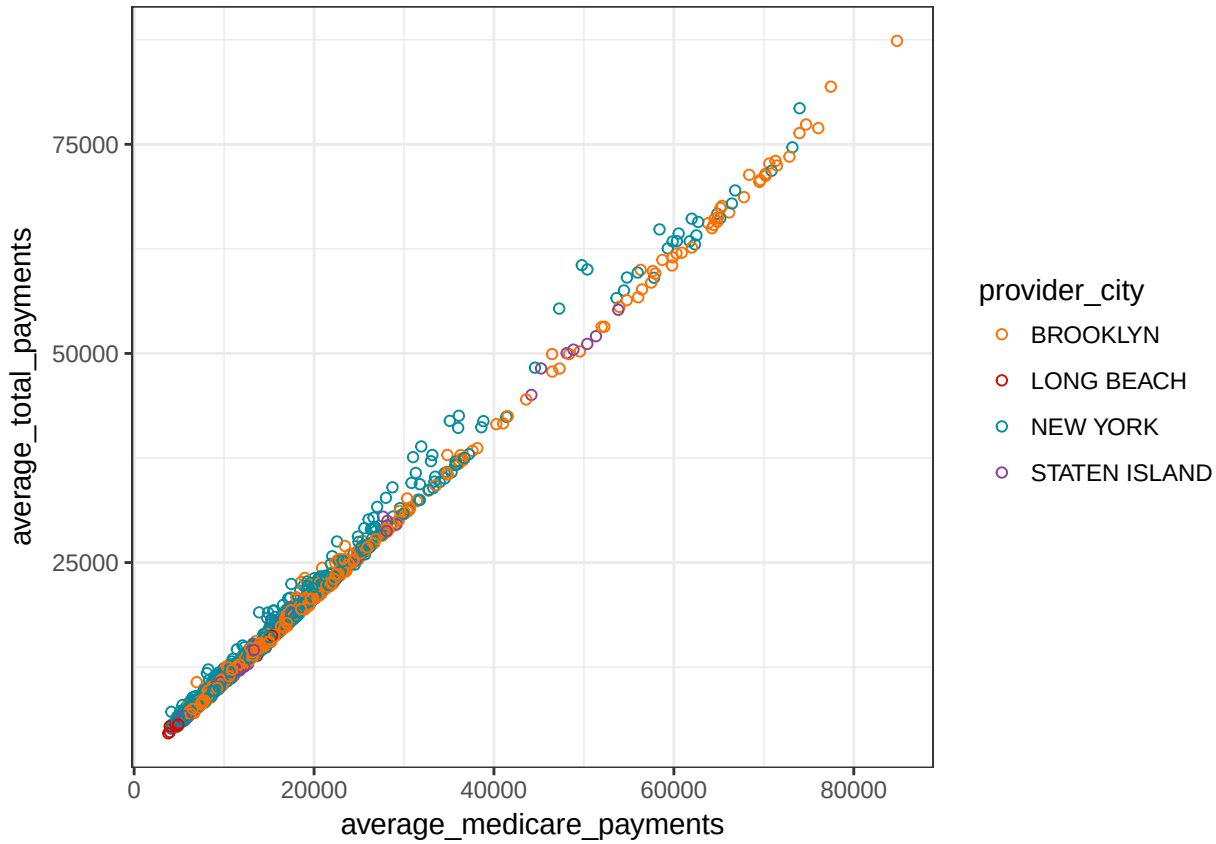
```
manhattan.hospital.procedures <- df.manhattan.inpatient.expenses %>% group_by(provider_name) %>% tally()
df.manhattan.inpatient.expenses %>% select(NYU, provider_name) %>% distinct() %>% right_join(manhattan.
  ggplot(aes(reorder(provider_name, n), n, col=NYU)) + geom_point() + coord_flip() +
    scale_color_manual(values=c("#B7D968","#8C5881")) +
  labs(subtitle="DRG procedures performed at Manhattan Hospitals", y="Number of distinct DRG Procedures
  theme(legend.position =
```

```
          "none")
```




```
# df.manhattan.inpatient.expenses %>% ggplot(aes(x=reorder(provider_name,total_discharges, FUN=median),
#     scale_alpha_manual(values=c(0.2,0.1)) +
#     scale_fill_manual(values=c("#9DE0AD","#8C5881")) +
#   labs(x="Hospital", y="Average Total Payments",
#        subtitle=
#          "Boxplots ordered by max total discharges")
```

```
manhattan_plot <- ggplot(df.manhattan.inpatient.expenses, aes(x = average_medicare_payments, y = averag
# using provider city as color
manhattan_plot+ geom_jitter(aes(color = provider_city), pch=21) + scale_color_futurama()
```




```
df.550.first.ave <- df.manhattan.inpatient.expenses %>% filter(provider_street_address ==
                                                    "550 FIRST AVENUE")

df.550.first.ave %>%
  group_by(withcomplicationsorwithout) %>%
  summarize(max_covered = max(average_covered_charges), count = n(), max_disch = max(total_discharges))
```

```
## # A tibble: 3 x 4
##    withcomplicationsorwithout max_covered count max_disch
##                         <chr>       <dbl> <int>     <dbl>
## 1 DOES NOT APPLY TO PROCEDURE    259989.3     9       179
## 2             W COMPLICATIONS    218238.0    46       208
## 3          W/O COMPLICATIONS    125392.1    42       689
```

```
sapply(df.550.first.ave[,c("average_medicare_payments","total_discharges","average_covered_charges","ave
```

```
##         average_medicare_payments total_discharges average_covered_charges
## Min.                     4138.42         11.00000                16660.37
## 1st Qu.                  6706.01         28.00000                28830.06
## Median                   9557.69         43.00000                40696.74
## Mean                    14125.23         68.23711                59737.77
## 3rd Qu.                 16211.93         71.00000                73773.51
## Max.                    66812.53        689.00000               259989.32
##         average_total_payments
## Min.                  5295.11
## 1st Qu.               8177.05
## Median               10988.55
## Mean                 15891.07
```

```
## 3rd Qu.               19242.83
## Max.                  69484.14
```

```
drg_W_df.550.first.ave <- psych::describeBy(df.550.first.ave$average_total_payments, group=df.550.first
drg_W_df.550.first.ave$WNA
```

```
##    vars n     mean       sd  median  trimmed      mad     min      max
## X1    1 9 26135.81 25442.67 15187.7 26135.81 14666.75 5295.11 69484.14
##       range skew kurtosis      se
## X1 64189.03 0.77    -1.22 8480.89
```

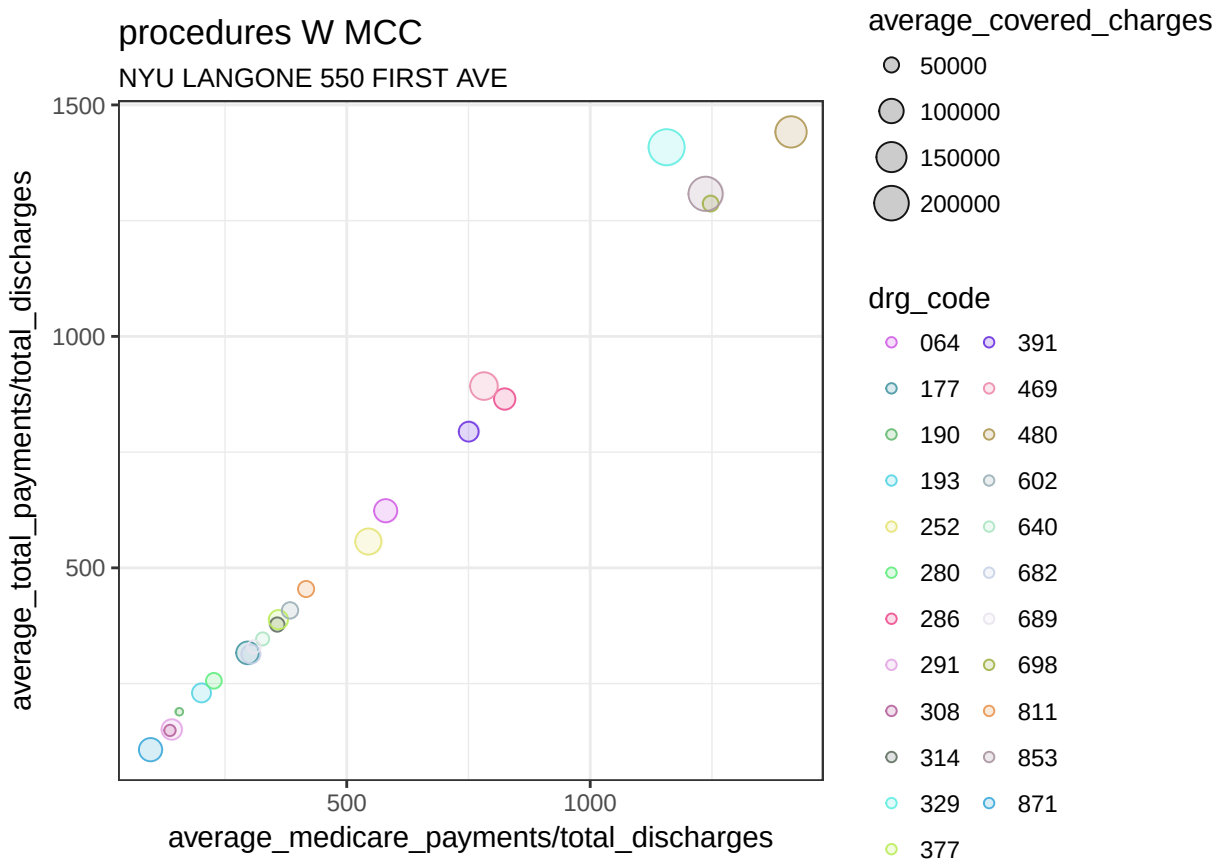
```
df.manhattan.inpatient.expenses %>%
  group_by(provider_name) %>%
  summarize(max_covered_charges = max(average_covered_charges), max_medicare_payments= max(average_medi
```

```
## # A tibble: 28 x 6
##                                   provider_name max_covered_charges
##                                           <chr>               <dbl>
##  1                    BELLEVUE HOSPITAL CENTER            49034.04
##  2                  BETH ISRAEL MEDICAL CENTER           180414.77
##  3           BROOKDALE HOSPITAL MEDICAL CENTER           109070.92
##  4 BROOKLYN HOSPITAL CENTER AT DOWNTOWN CAMPUS            74254.86
##  5                       CONEY ISLAND HOSPITAL            66430.25
##  6                      HARLEM HOSPITAL CENTER            32832.18
##  7                HOSPITAL FOR SPECIAL SURGERY            78468.56
##  8                   INTERFAITH MEDICAL CENTER           121017.94
##  9                KINGS COUNTY HOSPITAL CENTER            83421.57
## 10            KINGSBROOK JEWISH MEDICAL CENTER           135872.96
## # ... with 18 more rows, and 4 more variables:
## #   max_medicare_payments <dbl>, count <int>, max_discharges <dbl>,
## #   max_total_payments <dbl>
```

```
x <- sapply(df.550.first.ave, is.numeric)
colnames(df.550.first.ave[x])
```

```
## [1] "total_discharges"          "average_covered_charges"
## [3] "average_total_payments"    "average_medicare_payments"
## [5] "hospital_state_area"       "hospital_state_population"
## [7] "hospital_pop_area"
```

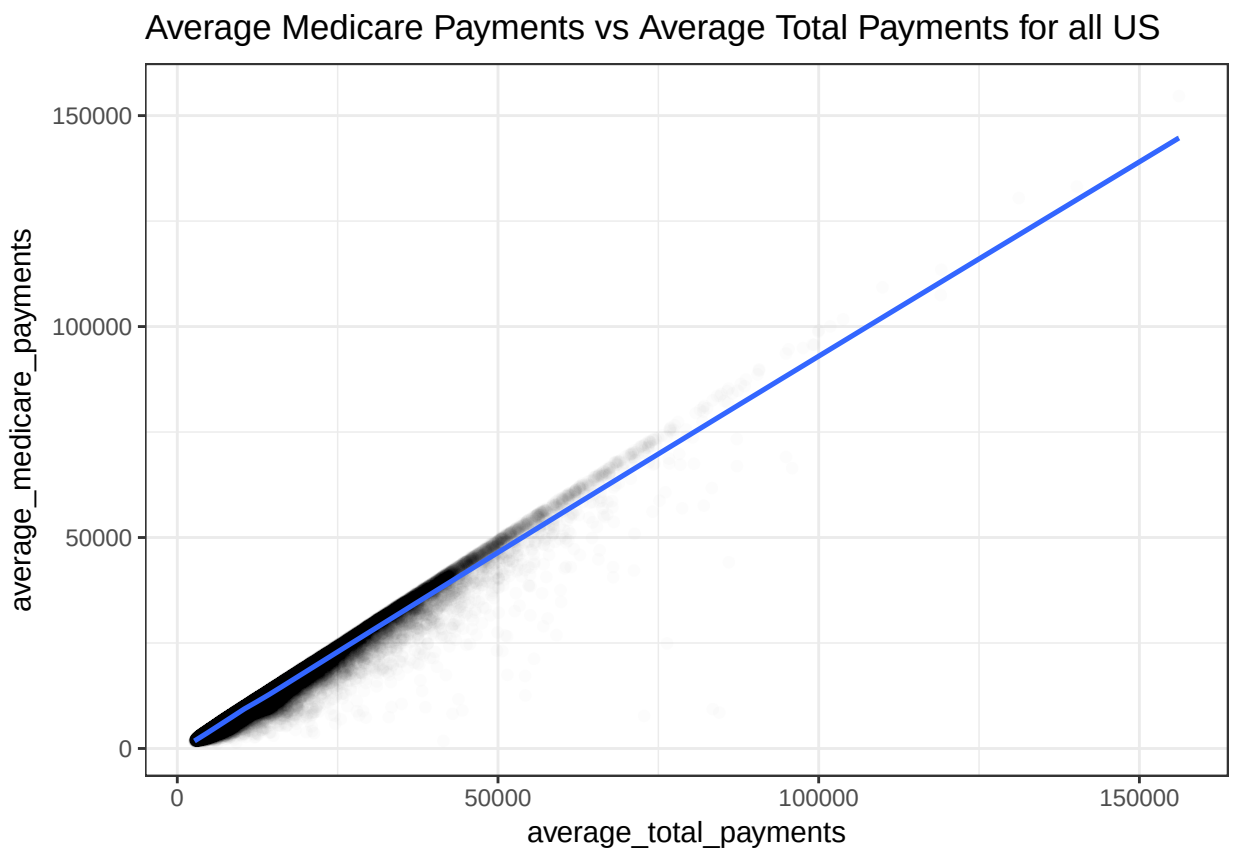
```
df.550.first.ave %>% filter(drg_W=="W MCC") %>% ggplot(aes(average_medicare_payments/total_discharges,
```




```
wes_palettes <- list(
  BottleRocket1 = c("#A42820", "#5F5647", "#9B110E", "#3F5151", "#4E2A1E", "#550307", "#0C1707"),
  BottleRocket2 = c("#FAD510", "#CB2314", "#273046", "#354823", "#1E1E1E"),
  Rushmore1 = c("#E1BD6D", "#EABE94", "#0B775E", "#35274A" ,"#F2300F"),
  Royal1 = c("#899DA4", "#C93312", "#FAEFD1", "#DC863B"),
  Royal2 = c("#9A8822", "#F5CDB4", "#F8AFA8", "#FDDDA0", "#74A089"),
  Zissou1 = c("#3B9AB2", "#78B7C5", "#EBCC2A", "#E1AF00", "#F21A00"),
  Darjeeling1 = c("#FF0000", "#00A08A", "#F2AD00", "#F98400", "#5BBCD6"),
  Darjeeling2 = c("#ECCBAE", "#046C9A", "#D69C4E", "#ABDDDE", "#000000"),
  Chevalier1 = c("#446455", "#FDD262", "#D3DDDC", "#C7B19C"),
  FantasticFox1 = c("#DD8D29", "#E2D200", "#46ACC8", "#E58601", "#B40F20"),
  Moonrise1 = c("#F3DF6C", "#CEAB07", "#D5D5D3", "#24281A"),
  Moonrise2 = c("#798E87", "#C27D38", "#CCC591", "#29211F"),
  Moonrise3 = c("#85D4E3", "#F4B5BD", "#9C964A", "#CDC08C", "#FAD77B"),
  Cavalcanti1 = c("#D8B70A", "#02401B", "#A2A475", "#81A88D", "#972D15"),
  GrandBudapest1 = c("#F1BB7B", "#FD6467", "#5B1A18", "#D67236"),
  GrandBudapest2 = c("#E6A0C4", "#C6CDF7", "#D8A499", "#7294D4")
)
gbp =c("#733080","#F4BAC8","#A40607","#7288B9","#F0C595","#733080")
nyu_cols_maybe <- c(gbp, wes_palettes$FantasticFox1, wes_palettes$GrandBudapest1, wes_palettes$GrandBuda
```

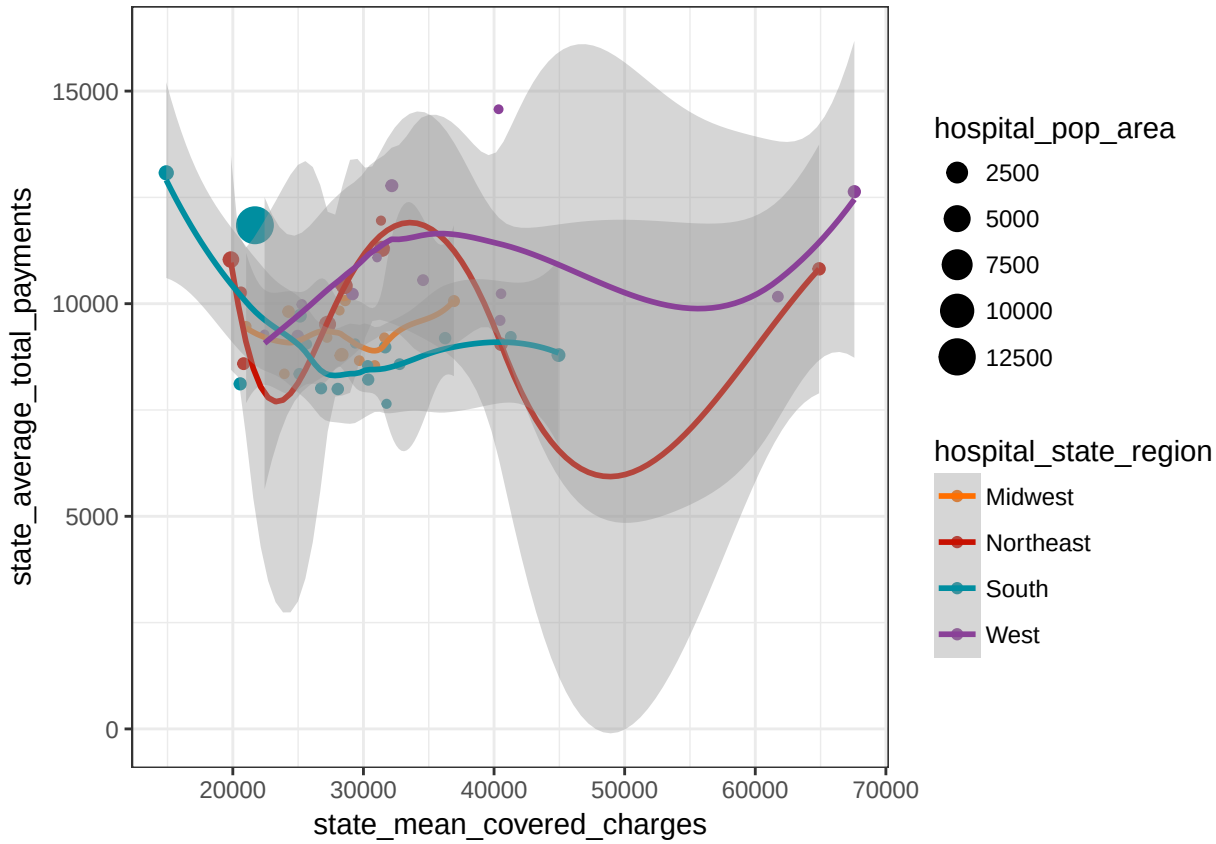
```
ggplot(df.t, aes(x = average_total_payments, y = average_medicare_payments)) +
geom_point(alpha=0.01) + ggtitle("Average Medicare Payments vs Average Total Payments for all US") + ge
```

```
## `geom_smooth()` using method = 'gam'
```



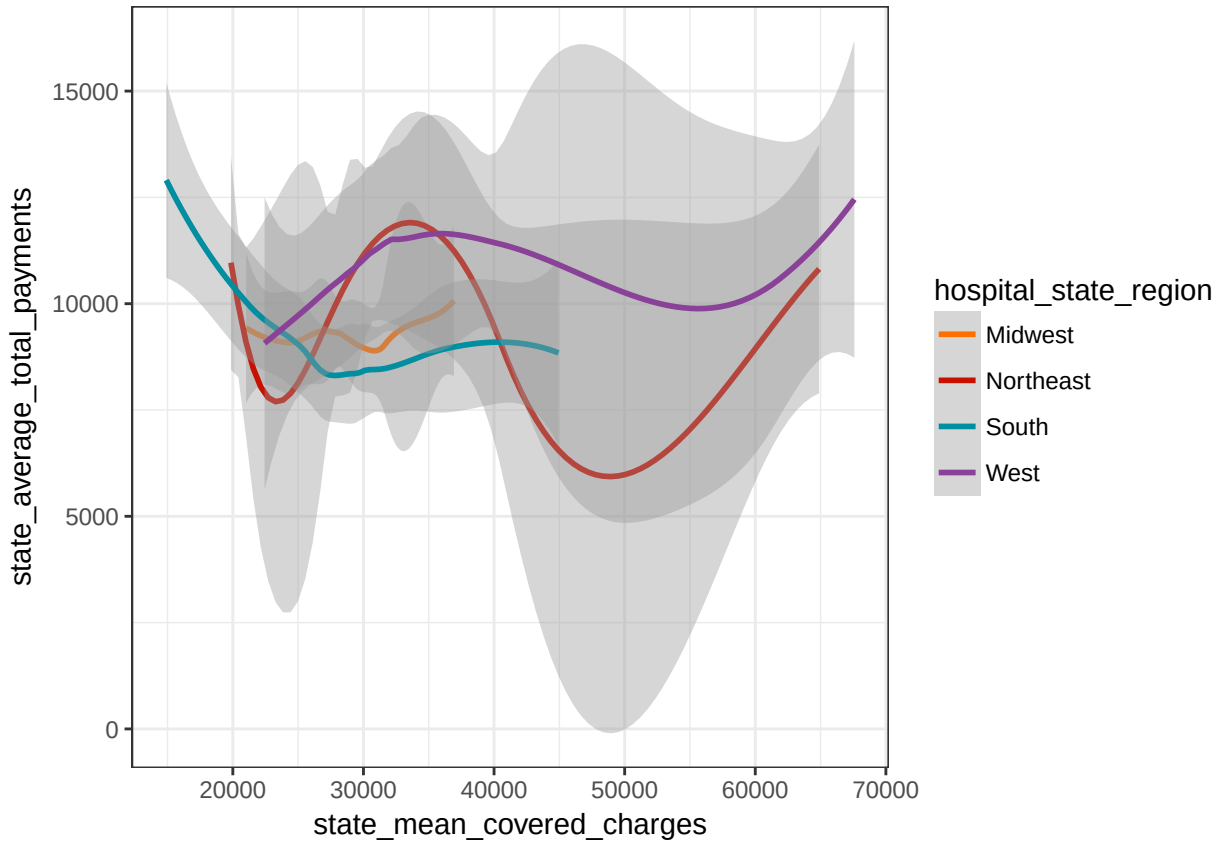

```
hospital_states$hospital_state <- as.character(hospital_states$hospital_state)
state_averages_df <-
  df.t %>%
  group_by(hospital_state) %>%
  summarize(state_mean_covered_charges = mean(average_covered_charges), state_mean_total_discharges = me
mean(average_total_payments), state_average_medicare_payments = mean(average_medicare_payments)

) %>% left_join(hospital_states, by="hospital_state") %>% mutate(state_medicare_payments_over_total_payn
```

```
ggplot(state_averages_df, aes(x = state_mean_covered_charges, y = state_average_total_payments,color = h
  geom_point(aes(size=hospital_pop_area)) +
  geom_smooth(method="loess") + scale_color_futurama()
```

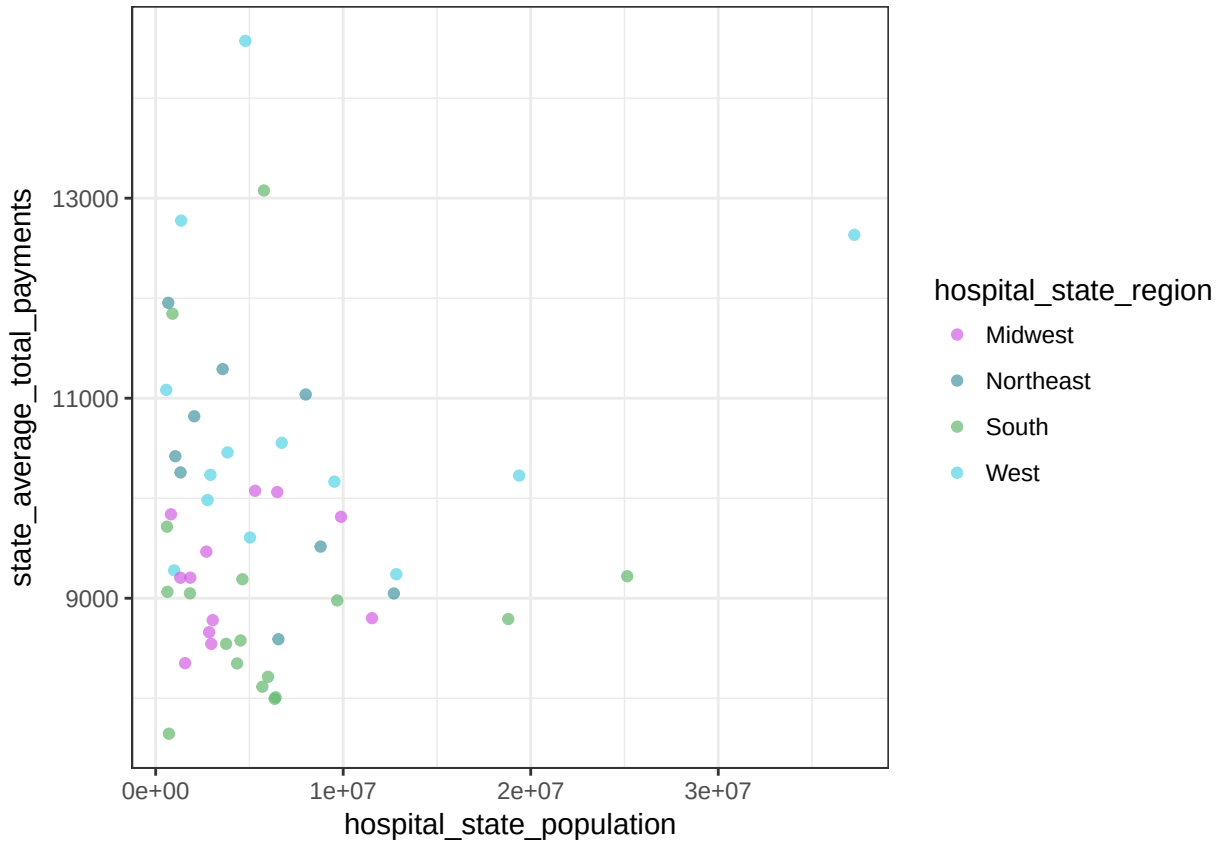



```
ggplot(state_averages_df, aes(x = state_mean_covered_charges, y = state_average_total_payments,color = h
  # geom_point() +
  geom_smooth() + scale_color_futurama()
```

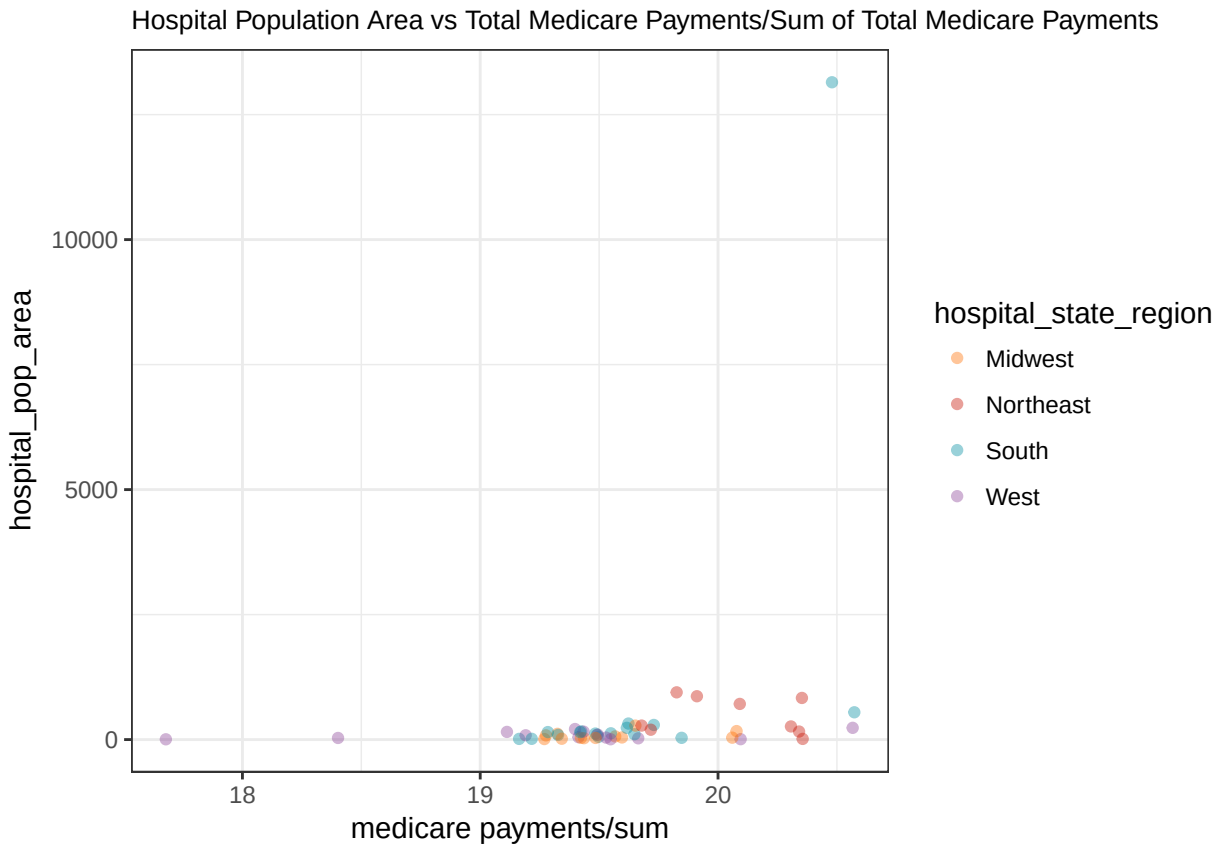
```
## `geom_smooth()` using method = 'loess'
```




```
state_averages_df %>% ggplot(aes(hospital_state_population, state_average_total_payments, col=hospital_s
```



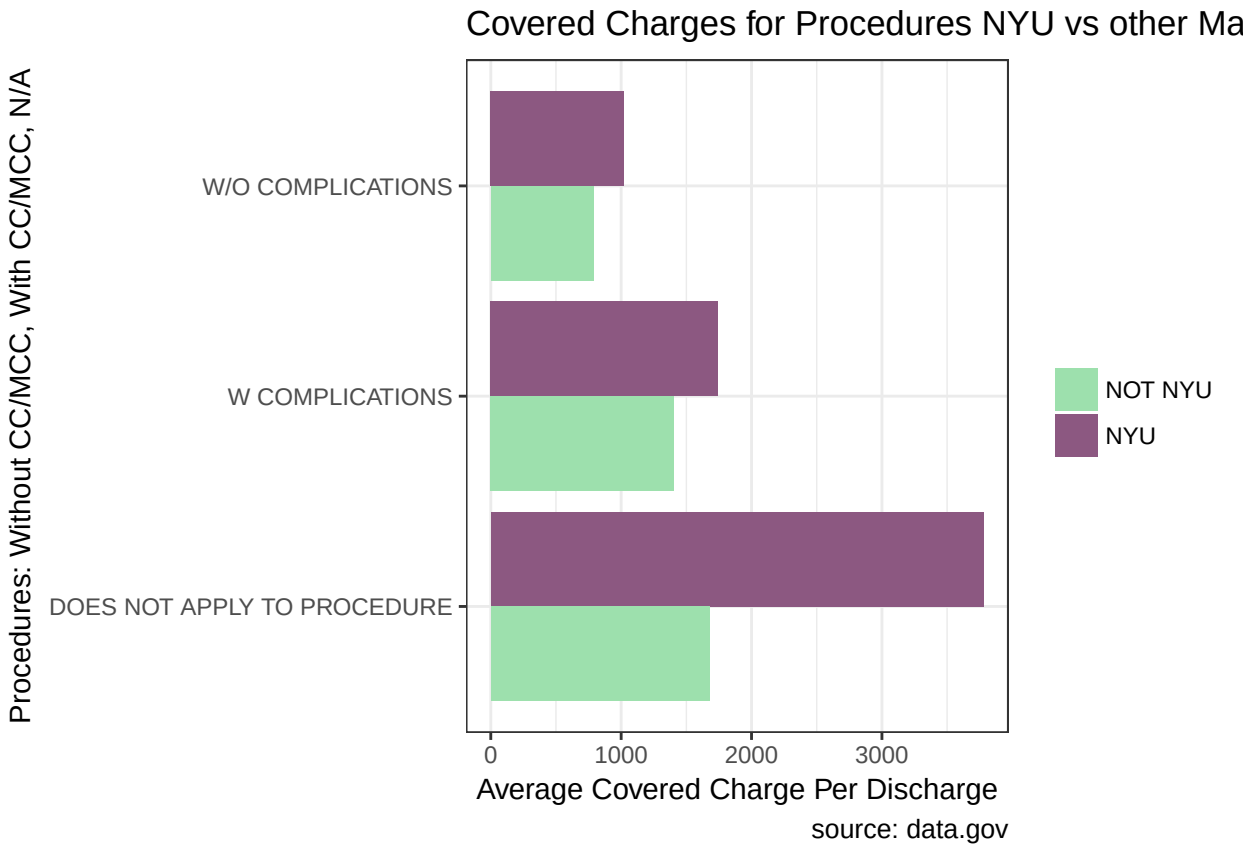

```
state_averages_df %>% ggplot(aes(x = 1000*state_medicare_payments_over_total_payments/sum(state_medicar
  labs(x="medicare payments/sum", subtitle="Hospital Population Area vs Total Medicare Payments/Sum of
```




```
df.manhattan.inpatient.expenses %>% group_by(withcomplicationsorwithout, NYU) %>%
  summarise(mean_covered_charge_per_disch = mean(average_covered_charges/total_discharges))
```

```
## # A tibble: 6 x 3
## # Groups:   withcomplicationsorwithout [?]
##    withcomplicationsorwithout     NYU mean_covered_charge_per_disch
##                         <chr>  <fctr>                         <dbl>
## 1 DOES NOT APPLY TO PROCEDURE NOT NYU                     1678.9996
## 2 DOES NOT APPLY TO PROCEDURE     NYU                     3780.7482
## 3             W COMPLICATIONS NOT NYU                     1404.9787
## 4             W COMPLICATIONS     NYU                     1742.7239
## 5           W/O COMPLICATIONS NOT NYU                      786.1939
## 6           W/O COMPLICATIONS     NYU                     1020.8330
```

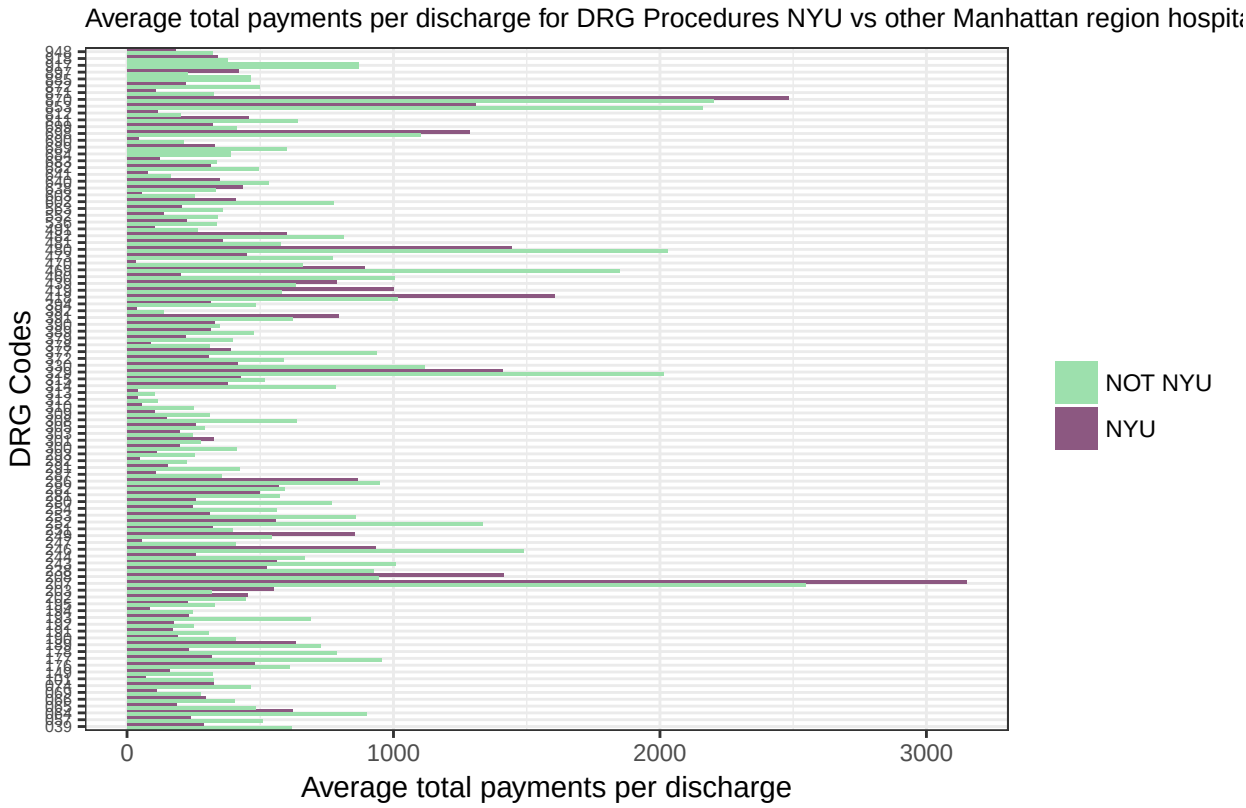
```
df.manhattan.inpatient.expenses %>% group_by(withcomplicationsorwithout, NYU) %>%
  summarise(mean_covered_charge_per_disch = mean(average_covered_charges/total_discharges)) %>%
  ggplot(aes(factor(withcomplicationsorwithout), mean_covered_charge_per_disch, fill=NYU)) + geom_col(p
  scale_fill_manual(name="", values=c("#9DE0AD","#8C5881")) + coord_flip() +
  labs(title="Covered Charges for Procedures NYU vs other Manhattan region hospitals", y="Average Cover
```




```
df.manhattan.inpatient.expenses %>% group_by(drg_code, NYU) %>%
  summarise(avg_total_payments_per_disch = mean(average_total_payments/total_discharges)) %>% head(10)
```

```
## # A tibble: 10 x 3
## # Groups:   drg_code [5]
##    drg_code     NYU avg_total_payments_per_disch
##       <chr>  <fctr>                        <dbl>
##  1      039 NOT NYU                     616.0401
##  2      039     NYU                     285.5406
##  3      057 NOT NYU                     508.9051
##  4      057     NYU                     237.4776
##  5      064 NOT NYU                     899.9452
##  6      064     NYU                     622.5697
##  7      065 NOT NYU                     483.0742
##  8      065     NYU                     186.2466
##  9      066 NOT NYU                     404.8173
## 10      066     NYU                     294.5214
```

```
df.manhattan.inpatient.expenses %>% group_by(drg_code, NYU) %>%
  summarise(avg_total_payments_per_disch = mean(average_total_payments/total_discharges)) %>%
  ggplot(aes(factor(drg_code), avg_total_payments_per_disch, fill=NYU)) + geom_col(position="dodge") +
  scale_fill_manual(name="", values=c("#9DE0AD","#8C5881")) + coord_flip() +
  labs(subtitle="Average total payments per discharge for DRG Procedures NYU vs other Manhattan region
```

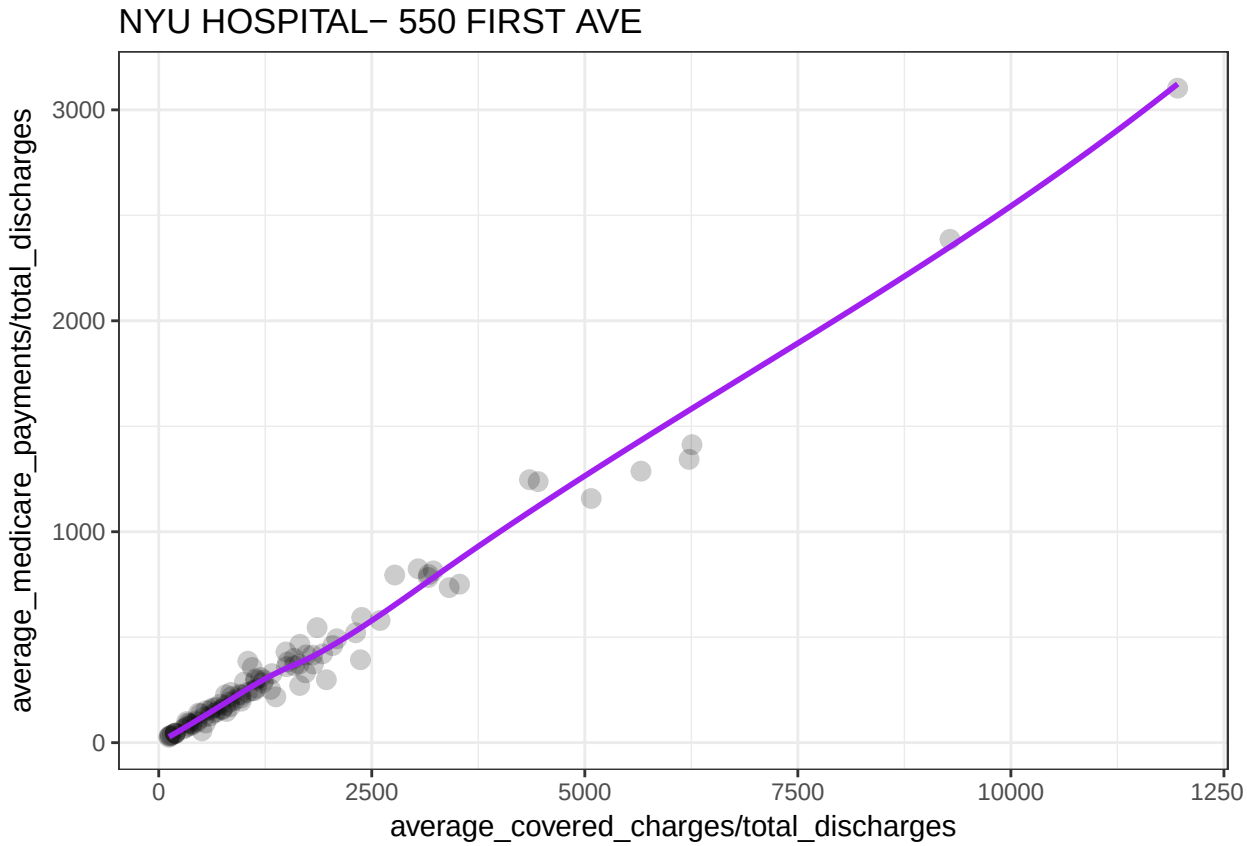



```
summary(df.550.first.ave$average_covered_charges)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   16660   28830   40697   59738   73774  259989
```

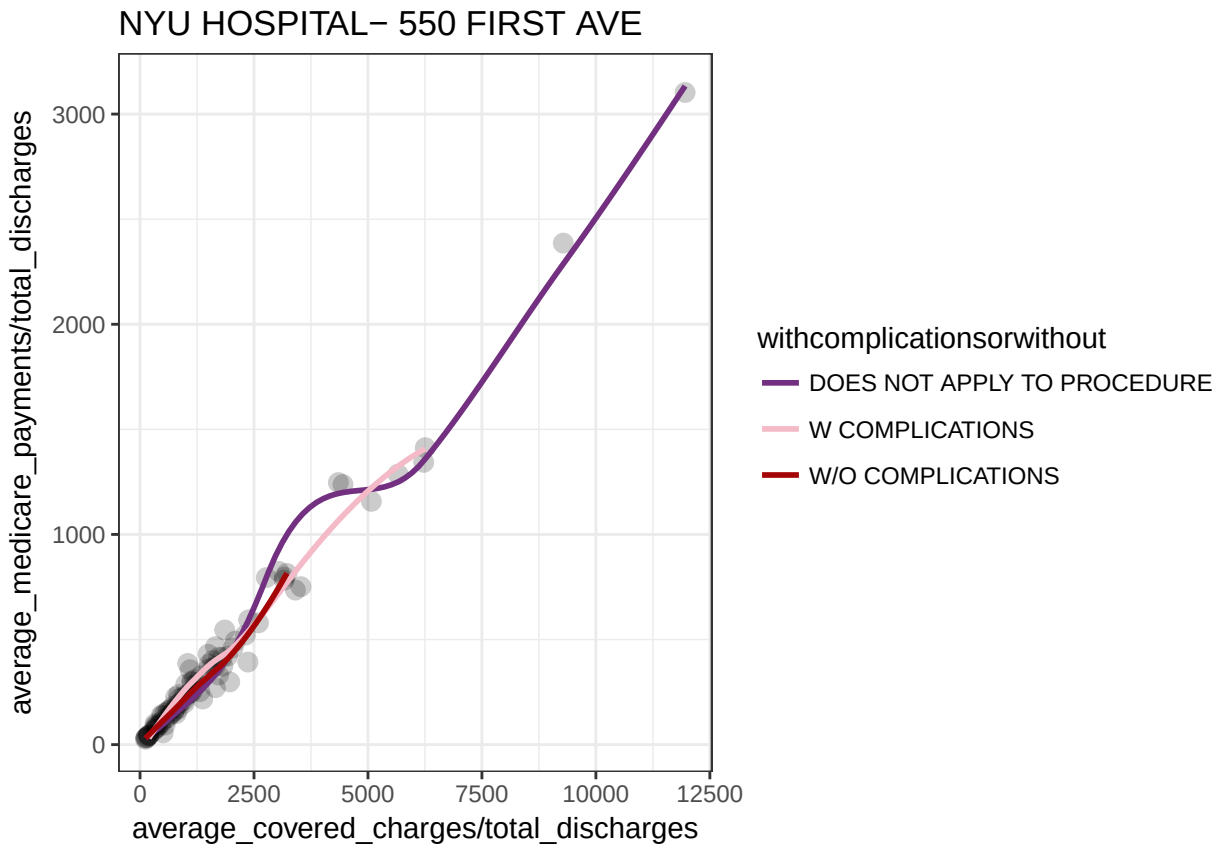
```
nyu_color_scale <- scale_color_manual(values=nyu_cols_maybe)
# 1st ave plot
firstave_plot <- ggplot(df.550.first.ave, aes(x = average_covered_charges/total_discharges, y = average_

dia_plot <- firstave_plot + geom_point(alpha=0.2, size=3) + ggtitle("NYU HOSPITAL- 550 FIRST AVE")

dia_plot + geom_smooth(se = FALSE, col="purple")
```

```
## `geom_smooth()` using method = 'loess'
```

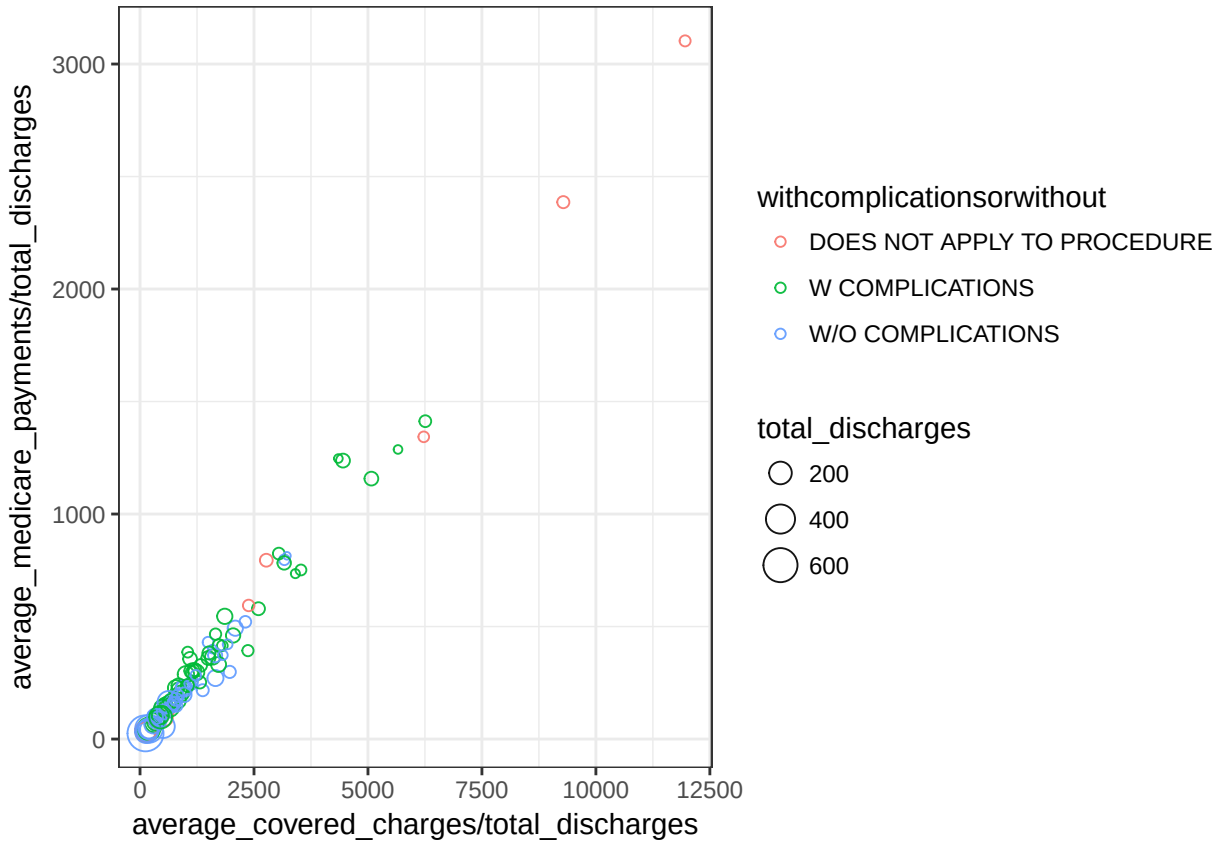



```
dia_plot + geom_smooth(aes(col = withcomplicationsorwithout), se = FALSE) +nyu_color_scale
```

```
## `geom_smooth()` using method = 'loess'
```




```
df.550.first.ave%>%
ggplot( aes(x = average_covered_charges/total_discharges, y = average_medicare_payments/total_discharges
  aes(col=withcomplicationsorwithout, size=total_discharges), pch=21)
```




```
summary(df.550.first.ave$average_covered_charges/df.550.first.ave$total_discharges)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   119.9   552.1  1046.2  1619.2  1816.0 11957.1
```

```
summary(df.550.first.ave$average_total_payments)
```

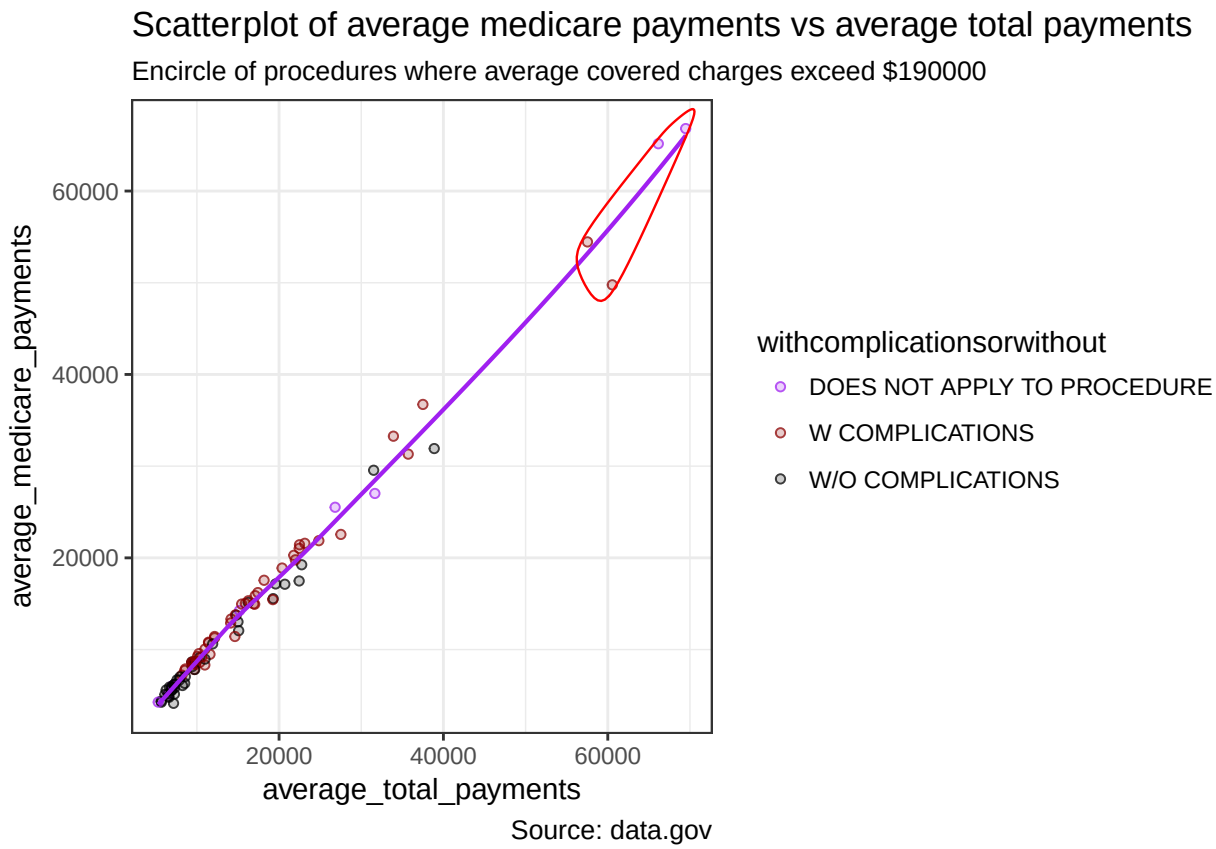
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    5295    8177   10989   15891   19243   69484
```

```
summary(df.550.first.ave$average_medicare_payments)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    4138    6706    9558   14125   16212   66813
```

```
library(ggalt)
payments_select <- df.550.first.ave[
   df.550.first.ave$average_covered_charges> 190000,]
   # &
   #                         df.550.first.ave$average_medicare_payments > 35000, ]

ggplot(df.550.first.ave, aes(x = average_total_payments, y = average_medicare_payments)) + geom_point(a
  aes(col=withcomplicationsorwithout)) +
  geom_point(alpha=0.7, size=1.3, pch=21,
  aes(col=withcomplicationsorwithout)) +
      # , size=total_discharges)) +   # draw points
  geom_smooth(method="loess", se=F, col="purple", size=.75) +
  # xlim(c(0, 0.1)) +
  # ylim(c(0, 500000)) +   # draw smoothing line
```

```
  geom_encircle(aes(x=average_total_payments, y = average_medicare_payments),
                data=payments_select,
                color="red",
                size=1.2,
                expand=0.02) +   # encircle
  labs(title="Scatterplot of average medicare payments vs average total payments",
       subtitle="Encircle of procedures where average covered charges exceed $190000",
       caption="Source: data.gov") + scale_color_manual(values=c("purple","darkred", "black"))
```



From R cookbook:

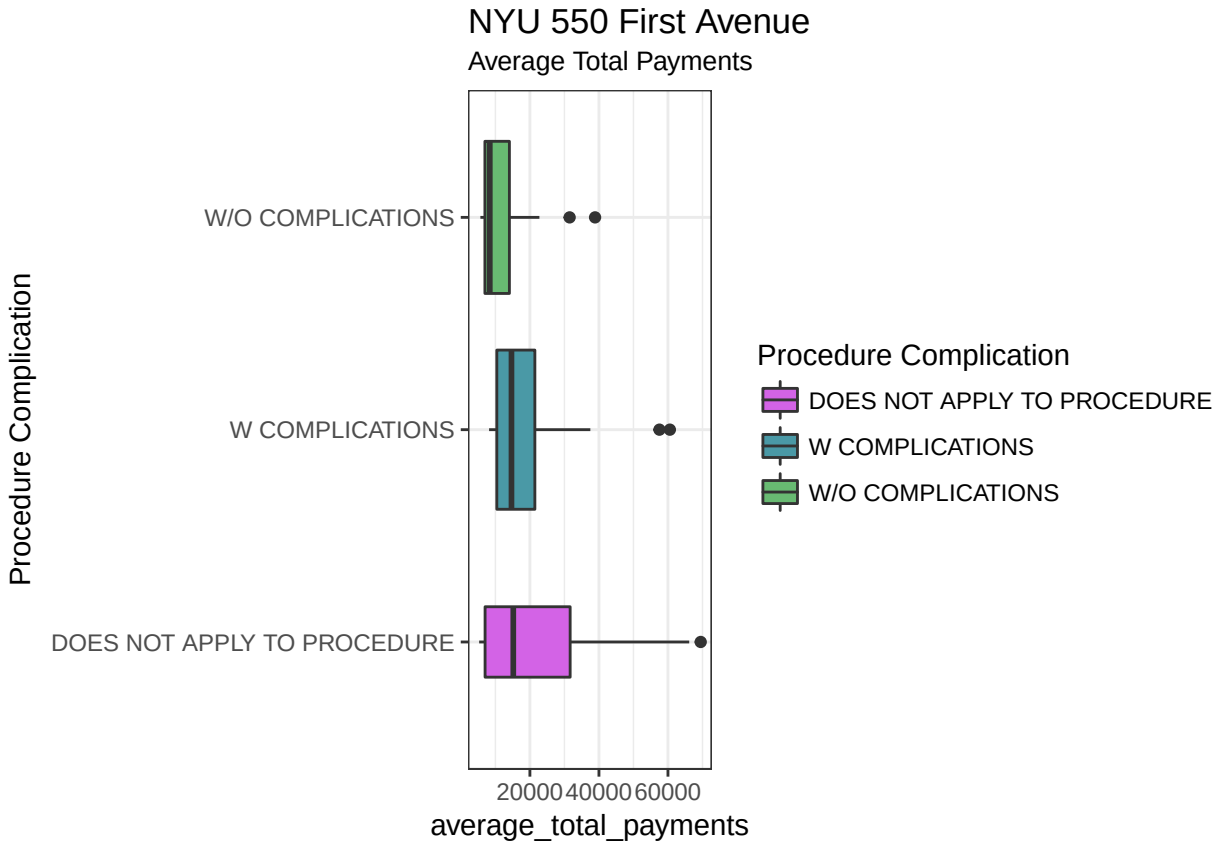
```
## Gives count, mean, standard deviation, standard error of the mean, and confidence interval (default 9
##   data: a data frame.
##   measurevar: the name of a column that contains the variable to be summariezed
##   groupvars: a vector containing names of columns that contain grouping variables
##   na.rm: a boolean that indicates whether to ignore NA's
##   conf.interval: the percent range of the confidence interval (default is 95%)
summarySE <- function(data=NULL, measurevar, groupvars=NULL, na.rm=FALSE,
                      conf.interval=.95, .drop=TRUE) {
    library(plyr)

    # New version of length which can handle NA's: if na.rm==T, don't count them
    length2 <- function (x, na.rm=FALSE) {
        if (na.rm) sum(!is.na(x))
        else       length(x)
    }

    # This does the summary. For each group's data frame, return a vector with
```

```
    # N, mean, and sd
    datac <- ddply(data, groupvars, .drop=.drop,
      .fun = function(xx, col) {
        c(N    = length2(xx[[col]], na.rm=na.rm),
          mean = mean   (xx[[col]], na.rm=na.rm),
          sd   = sd     (xx[[col]], na.rm=na.rm)
        )
      },
      measurevar
    )

    # Rename the "mean" column
    datac <- rename(datac, c("mean" = measurevar))

    datac$se <- datac$sd / sqrt(datac$N)  # Calculate standard error of the mean

    # Confidence interval multiplier for standard error
    # Calculate t-statistic for confidence interval:
    # e.g., if conf.interval is .95, use .975 (above/below), and use df=N-1
    ciMult <- qt(conf.interval/2 + .5, datac$N-1)
    datac$ci <- datac$se * ciMult

    return(datac)
}
## Norms the data within specified groups in a data frame; it normalizes each
## subject (identified by idvar) so that they have the same mean, within each group
## specified by betweenvars.
##   data: a data frame.
##   idvar: the name of a column that identifies each subject (or matched subjects)
##   measurevar: the name of a column that contains the variable to be summariezed
##   betweenvars: a vector containing names of columns that are between-subjects variables
##   na.rm: a boolean that indicates whether to ignore NA's
normDataWithin <- function(data=NULL, idvar, measurevar, betweenvars=NULL,
                           na.rm=FALSE, .drop=TRUE) {
    library(plyr)

    # Measure var on left, idvar + between vars on right of formula.
    data.subjMean <- ddply(data, c(idvar, betweenvars), .drop=.drop,
     .fun = function(xx, col, na.rm) {
        c(subjMean = mean(xx[,col], na.rm=na.rm))
      },
      measurevar,
      na.rm
    )

    # Put the subject means with original data
    data <- merge(data, data.subjMean)

    # Get the normalized data in a new column
    measureNormedVar <- paste(measurevar, "_norm", sep="")
    data[,measureNormedVar] <- data[,measurevar] - data[,"subjMean"] +
                               mean(data[,measurevar], na.rm=na.rm)
```

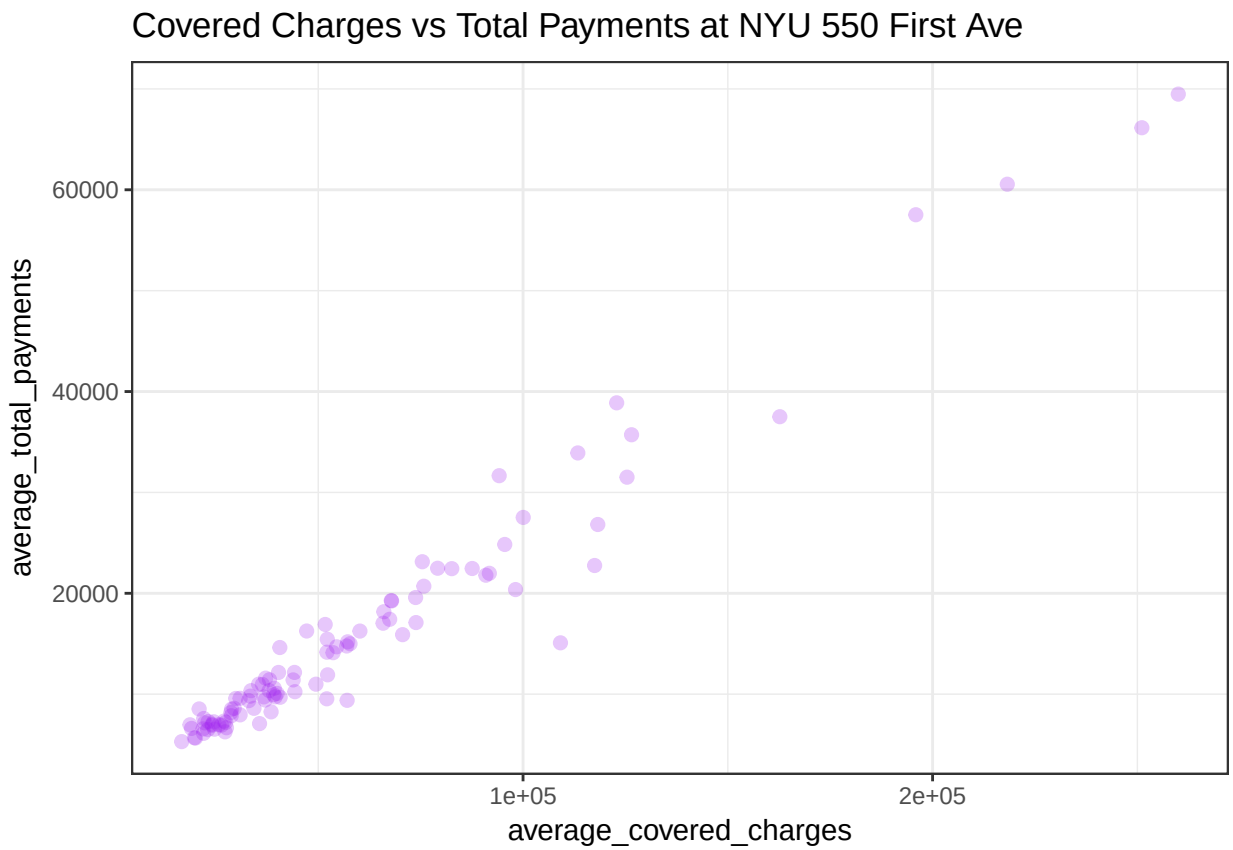
```
    # Remove this subject mean column
    data$subjMean <- NULL

    return(data)
}
## Summarizes data, handling within-subjects variables by removing inter-subject variability.
## It will still work if there are no within-S variables.
## Gives count, un-normed mean, normed mean (with same between-group mean),
##   standard deviation, standard error of the mean, and confidence interval.
## If there are within-subject variables, calculate adjusted values using method from Morey (2008).
##   data: a data frame.
##   measurevar: the name of a column that contains the variable to be summariezed
##   betweenvars: a vector containing names of columns that are between-subjects variables
##   withinvars: a vector containing names of columns that are within-subjects variables
##   idvar: the name of a column that identifies each subject (or matched subjects)
##   na.rm: a boolean that indicates whether to ignore NA's
##   conf.interval: the percent range of the confidence interval (default is 95%)
summarySEwithin <- function(data=NULL, measurevar, betweenvars=NULL, withinvars=NULL,
                            idvar=NULL, na.rm=FALSE, conf.interval=.95, .drop=TRUE) {

  # Ensure that the betweenvars and withinvars are factors
  factorvars <- vapply(data[, c(betweenvars, withinvars), drop=FALSE],
    FUN=is.factor, FUN.VALUE=logical(1))

  if (!all(factorvars)) {
    nonfactorvars <- names(factorvars)[!factorvars]
    message("Automatically converting the following non-factors to factors: ",
            paste(nonfactorvars, collapse = ", "))
    data[nonfactorvars] <- lapply(data[nonfactorvars], factor)
  }

  # Get the means from the un-normed data
  datac <- summarySE(data, measurevar, groupvars=c(betweenvars, withinvars),
                     na.rm=na.rm, conf.interval=conf.interval, .drop=.drop)

  # Drop all the unused columns (these will be calculated with normed data)
  datac$sd <- NULL
  datac$se <- NULL
  datac$ci <- NULL

  # Norm each subject's data
  ndata <- normDataWithin(data, idvar, measurevar, betweenvars, na.rm, .drop=.drop)

  # This is the name of the new column
  measurevar_n <- paste(measurevar, "_norm", sep="")

  # Collapse the normed data - now we can treat between and within vars the same
  ndatac <- summarySE(ndata, measurevar_n, groupvars=c(betweenvars, withinvars),
                      na.rm=na.rm, conf.interval=conf.interval, .drop=.drop)

  # Apply correction from Morey (2008) to the standard error and confidence interval
  #  Get the product of the number of conditions of within-S variables
  nWithinGroups    <- prod(vapply(ndatac[,withinvars, drop=FALSE], FUN=nlevels,
```

```
                            FUN.VALUE=numeric(1)))
  correctionFactor <- sqrt( nWithinGroups / (nWithinGroups-1) )

  # Apply the correction factor
  ndatac$sd <- ndatac$sd * correctionFactor
  ndatac$se <- ndatac$se * correctionFactor
  ndatac$ci <- ndatac$ci * correctionFactor

  # Combine the un-normed means with the normed results
  merge(datac, ndatac)
}
```

```
  ggplot(df.550.first.ave,aes(x=withcomplicationsorwithout, y=average_total_payments, fill=withcomplica
  labs(title="NYU 550 First Avenue",
subtitle="Average Total Payments", x="Procedure Complication") +
  coord_flip()
```



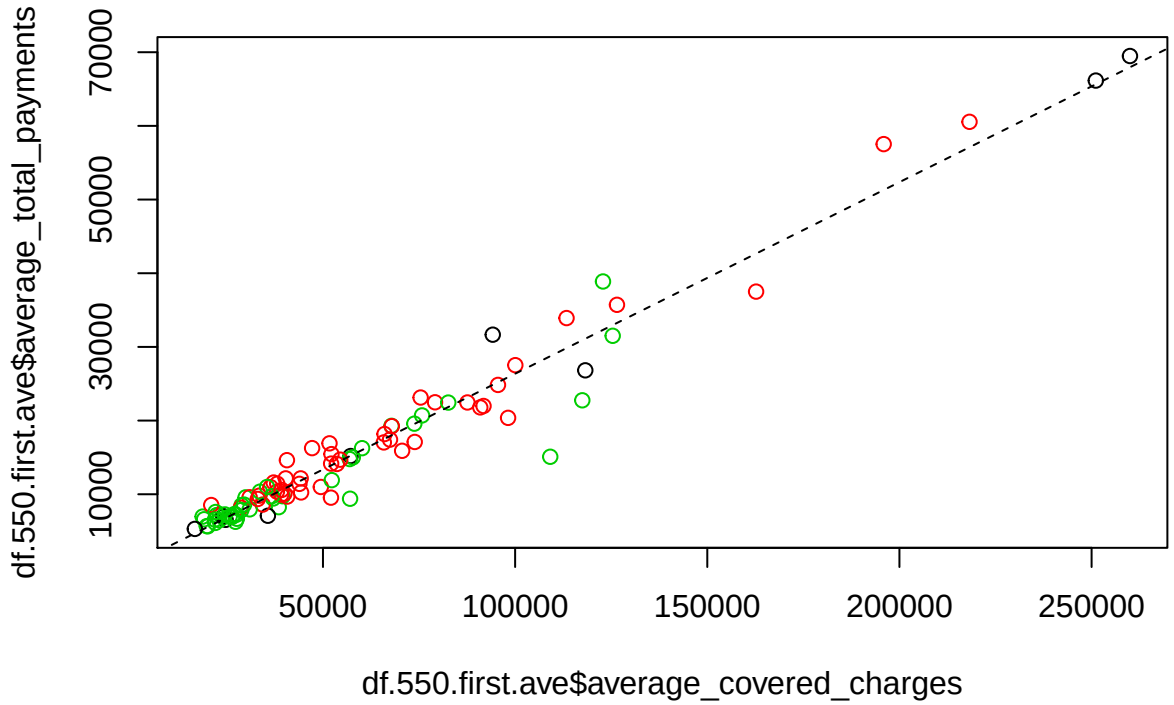

```
df.550.first.ave %>% ggplot(aes(average_covered_charges, average_total_payments)) + geom_point(col="pur
  labs(title="Covered Charges vs Total Payments at NYU 550 First Ave")
```




```
firstaveModel <- lm(average_total_payments ~ average_covered_charges, data = df.550.first.ave)
```

using base package plot

```
df.550.first.ave$withcomplicationsorwithout <- as.factor(df.550.first.ave$withcomplicationsorwithout)
plot(df.550.first.ave$average_covered_charges, df.550.first.ave$average_total_payments, col = df.550.fi
abline(firstaveModel, lty = 2)
```




```
# install.packages("CerioliOutlierDetection")
```