Energy as a Fundamental Right ~ EFRI ~ Simulation Model

Code

I. Simulation Preparations

Loading Libraries

Hide

```
# install.packages("plotly", dependencies=TRUE)
# install.packages("beepr")

library(readxl)
library(tidyverse)
library(ggplot2)
library(plotly)
library(hrbrthemes)
library(viridis)
library(stringr)
library(beepr)
```

Getting Input Data

Hide

```
path <- paste(getwd(),"/inputdata.xlsx", sep = "")

inputdata <- read_excel(path, sheet = "data")
```

Adding Columns for Initial Analysis

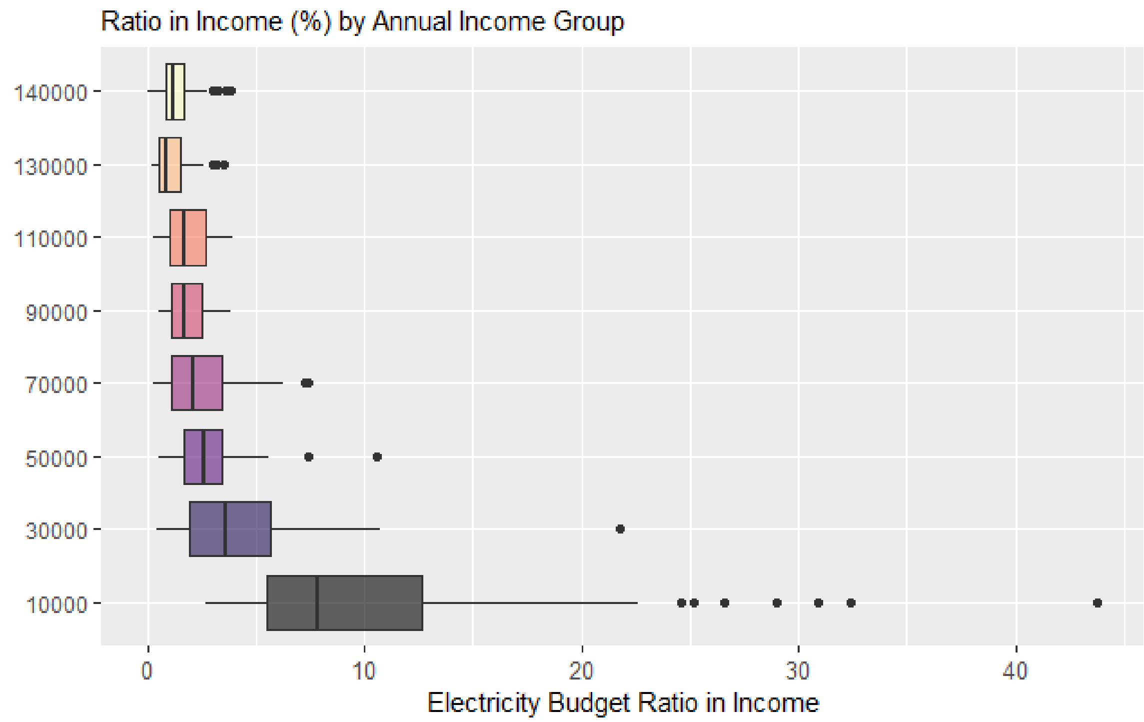
Hide

```
inputdata = cbind(inputdata,(100*inputdata$`Electricity Price`*inputdata$`Annual Electricity Consumption
`)/inputdata$`Annual Income`, as.factor(round(inputdata$`Annual Income`,0)))
names(inputdata)[length(names(inputdata))-1]<-"RII"
names(inputdata)[length(names(inputdata))]<-"Income Category"
```

Understanding Current Energy Poverty Situation

Hide

```

inputdata %>%
  ggplot( aes(x=`Income Category`, y=RII, fill=`Income Category`)) +
  geom_boxplot() +
  scale_fill_viridis(discrete = TRUE, alpha=0.6, option="A") +
  theme(axis.text.x = element_text(size=10),
        legend.position="none",
        plot.title = element_text(size=11)
  ) +
  ggtitle("Ratio in Income (%) by Annual Income Group") +
  xlab("")+
  ylab("Electricity Budget Ratio in Income")+
  coord_flip()
```



Setting Simulation Input Columns

Hide

```
attach(inputdata)
h = `Number of Residents` # Number of residents in a household
i = `Annual Income` # Annual income of a household
c = `Annual Electricity Consumption` # Annual electricity consumption of a household
p = mean(`Electricity Price`) #Defining the electricity price of the selected region per KWh
m = `Statistical Multiplier` # Statistical multiplier for expressing how many households are represented
by that survey
```

Checking Number of Simulation Cases

Hide

```
# Possible Free Electricity per Resident Cases
paste("EPR: from 0 to",trunc(sum(c * m) / sum(h * m)))

# Possible Free Electricity per Household Cases
paste("EPH: from 0 to",trunc(sum(c * m) / sum(m)))

# Total Number of Combinations to be Simulated
paste("Total Combinations (# of rows in million):",round(trunc(sum(c * m) / sum(h * m)) * trunc(sum(c *
 m) / sum(m)) /1000000, digits=1))
```

II. Simulation Implementation

EFRI Simulation Function

- h: Household size (Number of residents) - list of numbers
- i: Annual income of a household - list of numbers
- c: Annual electricity consumption of a household - list of numbers
- p: Electricity price - numeric input or list of numbers (same unit as income)
- m: Statistical multiplier (number of represented households) - list of numbers

- epov: Energy poverty threshold (% of income)
- low : Income percentile threshold for selecting low income households

Hide

```
EFRI = function(h,i,c,p,m=1,epov=0.1,low=0.2,savename="EFRI_Sim")
 {

  start.time <- Sys.time()

  epr_limit = trunc(sum(c * m) / sum(h * m)) # Upper limit for free electricity  per resident (rounded down)
  eph_limit = trunc(sum(c * m) / sum(m)) # Upper limit for free electricity  per household (rounded down)

  datalist = list()
  counter = 1

  for(epr in 0:epr_limit)
  {
    for(eph in 0:eph_limit) # Computer RAM Consideration: Divide simulations if necessary (ex: 0:4500 and 4501:9000)
    {
      if(sum(epr*h*m)+sum(eph*m) > sum(c * m)) {
        next # EFRI Constraint: Total free electricity amount cannot be more than total consumed electricity in the population
      }
      rii_list = ((((sum((p*c)*m))/(sum((c-(pmin((epr*h+eph),c)))*m)))*((c-(pmin((epr*h+eph),c)))))/i)

      psim = sum((p*c)*m)/sum((c-(pmin((epr*h+eph),c)))*m)

      riiave = mean(rii_list) # Average Ratio in Income
      riiavelow = mean(rii_list[i<=quantile(i, probs=low)])

      riimed = median(rii_list) # Median Ratio in Income
      riimedlow = median(rii_list[i<=quantile(i, probs=low)])

      npr = sum(m[rii_list>=epov]*h[rii_list>=epov])
      nph = sum(m[rii_list>=epov])

      datalist [[counter]] = c(epr,eph,round(100*psim, digits = 2),round(100*riiave, digits = 2),round(100*riiavelow, digits = 2),round(100*riimed, digits = 2),round(100*riimedlow, digits = 2),round(npr, digits = 0),round(nph, digits = 0))
      counter = counter+1
    }
  }

  results = do.call(rbind, datalist)
  colnames(results) = c("EPR", "EPH","PSIM","RIIAVE","RIIAVELOW","RIIMED","RIIMEDLOW","NPR","NPH")

  EFRI_Sim <<- results


  csvpath <- paste(getwd(),"/", savename,".csv", sep = "")
  write.csv(EFRI_Sim, csvpath, row.names = FALSE)

  end.time <- Sys.time()
  time.taken <- end.time - start.time
  print(paste("* Simulation Duration: ", round(time.taken, digits = 2)), quote=FALSE)
  print(paste("** Simulation data is saved into ", savename, ".csv file at your current directory", sep = ""), quote=FALSE)

}
```

Running the Simulation

Hide

```
EFRI(h,i,c,p,m,epov=0.1)

beep(8) # Beep Sound when the simulation ends
```

III. Preparing Data for Visualization

Checking Simulation Results

Hide

```
savename = "EFRI_Sim"
rawpath <- paste(getwd(),"/", savename, ".csv", sep = "")
rawdata = read.csv(rawpath)

nrow(rawdata)

summary(rawdata)
beep(1)
```

Understanding Data Patterns

Hide

```
rawdata[5000:6000,] %>%
  ggplot(aes(x=NPR, y=PSIM)) +
  geom_point() +
  xlab("Average Electricity Share in Low Income Budgets") +
  ylab("Price of Electricity")
```

Aggregating Data for Visualization

> The raw simulation results include large number of rows. The patterns showed us that there are better combinations in terms of electricity price. For each objective (RIIAVE, RIIAVELOW, RIIMED, RIIMEDLOW, NPR, NPH) the combination with the minimum electricity price is taken with the same objective result.
>
> The data can be aggregated in different ways. Here the **Alternative 1** is selected and used for the visualizations.

Aggregation Alternative 1 : Sufficient Data

Hide

```
filter_ave    = aggregate(PSIM ~ RIIAVE,     data = rawdata, FUN = min)
filter_avelow = aggregate(PSIM ~ RIIAVELOW,  data = rawdata, FUN = min)
filter_med    = aggregate(PSIM ~ RIIMED,     data = rawdata, FUN = min)
filter_medlow = aggregate(PSIM ~ RIIMEDLOW,  data = rawdata, FUN = min)
filter_npr    = aggregate(PSIM ~ NPR,        data = rawdata, FUN = min)
filter_nph    = aggregate(PSIM ~ NPH,        data = rawdata, FUN = min)


agg_ave    = rawdata %>%
             filter(rawdata$PSIM == filter_ave$PSIM    | rawdata$RIIAVE    == filter_ave$RIIAVE)
agg_avelow = rawdata %>%
             filter(rawdata$PSIM == filter_avelow$PSIM | rawdata$RIIAVELOW == filter_avelow$RIIAVELOW)
agg_med    = rawdata %>%
             filter(rawdata$PSIM == filter_med$PSIM    | rawdata$RIIMED    == filter_med$RIIMED)
agg_medlow = rawdata %>%
             filter(rawdata$PSIM == filter_medlow$PSIM | rawdata$RIIMEDLOW == filter_medlow$RIIMEDLOW)
agg_npr    = rawdata %>%
             filter(rawdata$PSIM == filter_npr$PSIM    | rawdata$NPR       == filter_npr$NPR)
agg_nph    = rawdata %>%
             filter(rawdata$PSIM == filter_nph$PSIM    | rawdata$NPH       == filter_nph$NPH)

simdata = rbind(agg_ave,agg_avelow,agg_med,agg_medlow,agg_npr,agg_nph) %>% distinct()

simpath = paste(getwd(),"/", savename, "_sim.csv", sep = "")
write.csv(simdata, simpath, row.names = FALSE)

rm(filter_ave, filter_avelow, filter_med, filter_medlow, filter_npr, filter_nph, agg_ave, agg_avelow, ag
g_med, agg_medlow, agg_npr, agg_nph)

beep(1)
```

Aggregation Alternative 2 : Less Data

```
filter_ave    = aggregate(PSIM ~ RIIAVE,     data = rawdata, FUN = min)
filter_avelow = aggregate(PSIM ~ RIIAVELOW,  data = rawdata, FUN = min)
filter_med    = aggregate(PSIM ~ RIIMED,     data = rawdata, FUN = min)
filter_medlow = aggregate(PSIM ~ RIIMEDLOW,  data = rawdata, FUN = min)
filter_npr    = aggregate(PSIM ~ NPR,        data = rawdata, FUN = min)
filter_nph    = aggregate(PSIM ~ NPH,        data = rawdata, FUN = min)

agg_ave    = semi_join(rawdata, filter_ave,    by=c("RIIAVE" = "RIIAVE",       "PSIM"="PSIM"))
agg_avelow = semi_join(rawdata, filter_avelow, by=c("RIIAVELOW" = "RIIAVELOW", "PSIM"="PSIM"))
agg_med    = semi_join(rawdata, filter_med,    by=c("RIIMED" = "RIIMED",       "PSIM"="PSIM"))
agg_medlow = semi_join(rawdata, filter_medlow, by=c("RIIMEDLOW" = "RIIMEDLOW", "PSIM"="PSIM"))
agg_npr    = semi_join(rawdata, filter_npr,    by=c("NPR" = "NPR",             "PSIM"="PSIM"))
agg_nph    = semi_join(rawdata, filter_nph,    by=c("NPH" = "NPH",             "PSIM"="PSIM"))

simdata = rbind(agg_ave,agg_avelow,agg_med,agg_medlow,agg_npr,agg_nph) %>% distinct()

simpath = paste(getwd(),"/", savename, "_sim.csv", sep = "")
write.csv(simdata, simpath, row.names = FALSE)

rm(filter_ave, filter_avelow, filter_med, filter_medlow, filter_npr, filter_nph, agg_ave, agg_avelow, ag
g_med, agg_medlow, agg_npr, agg_nph)

beep(1)
```

Aggregation Alternative 3 : More Data

```
filter_ave    = aggregate(PSIM ~ RIIAVE,     data = rawdata, FUN = min)
filter_avelow = aggregate(PSIM ~ RIIAVELOW,  data = rawdata, FUN = min)
filter_med    = aggregate(PSIM ~ RIIMED,     data = rawdata, FUN = min)
filter_medlow = aggregate(PSIM ~ RIIMEDLOW,  data = rawdata, FUN = min)
filter_npr    = aggregate(PSIM ~ NPR,        data = rawdata, FUN = min)
filter_nph    = aggregate(PSIM ~ NPH,        data = rawdata, FUN = min)

agg_ave    = semi_join(rawdata, filter_ave,    by=c("PSIM"="PSIM"))
agg_avelow = semi_join(rawdata, filter_avelow, by=c("PSIM"="PSIM"))
agg_med    = semi_join(rawdata, filter_med,    by=c("PSIM"="PSIM"))
agg_medlow = semi_join(rawdata, filter_medlow, by=c("PSIM"="PSIM"))
agg_npr    = semi_join(rawdata, filter_npr,    by=c("PSIM"="PSIM"))
agg_nph    = semi_join(rawdata, filter_nph,    by=c("PSIM"="PSIM"))

simdata = rbind(agg_ave,agg_avelow,agg_med,agg_medlow,agg_npr,agg_nph) %>% distinct()

simpath = paste(getwd(),"/", savename, "_sim.csv", sep = "")
write.csv(simdata, simpath, row.names = FALSE)

rm(filter_ave, filter_avelow, filter_med, filter_medlow, filter_npr, filter_nph, agg_ave, agg_avelow, ag
g_med, agg_medlow, agg_npr, agg_nph)

beep(1)
```

Defining the Optimum Points

Hide

```
normalizer <- function(x) {
  return ((x - min(x)) / (max(x) - min(x)))
}

opt_init   = rawdata[which(rawdata$EPR == 0 & rawdata$EPH ==0),]
opt_ave    = rawdata[which.min(sqrt(normalizer(rawdata$RIIAVE)   ^2 + normalizer(rawdata$PSIM)^2)),]
opt_avelow = rawdata[which.min(sqrt(normalizer(rawdata$RIIAVELOW)^2 + normalizer(rawdata$PSIM)^2)),]
opt_med    = rawdata[which.min(sqrt(normalizer(rawdata$RIIMED)   ^2 + normalizer(rawdata$PSIM)^2)),]
opt_medlow = rawdata[which.min(sqrt(normalizer(rawdata$RIIMEDLOW)^2 + normalizer(rawdata$PSIM)^2)),]
opt_npr    = rawdata[which.min(sqrt(normalizer(rawdata$NPR)      ^2 + normalizer(rawdata$PSIM)^2)),]
opt_nph    = rawdata[which.min(sqrt(normalizer(rawdata$NPH)      ^2 + normalizer(rawdata$PSIM)^2)),]

opt_init   = cbind(opt_init,   Simulation = "INITIAL")
opt_ave    = cbind(opt_ave,    Simulation = "RIIAVE")
opt_avelow = cbind(opt_avelow, Simulation = "RIIAVELOW")
opt_med    = cbind(opt_med,    Simulation = "RIIMED")
opt_medlow = cbind(opt_medlow, Simulation = "RIIMEDLOW")
opt_npr    = cbind(opt_npr,    Simulation = "NPR")
opt_nph    = cbind(opt_nph,    Simulation = "NPH")

optdata = rbind(opt_init, opt_ave,opt_avelow, opt_med, opt_medlow, opt_npr, opt_nph)

optpath = paste(getwd(),"/", savename, "_opt.csv", sep = "")
write.csv(optdata, optpath, row.names = FALSE)

rm(opt_init,opt_ave,opt_avelow,opt_med,opt_medlow,opt_npr,opt_nph,optpath)

beep(1)
```

IV. Simulated Data & Plots

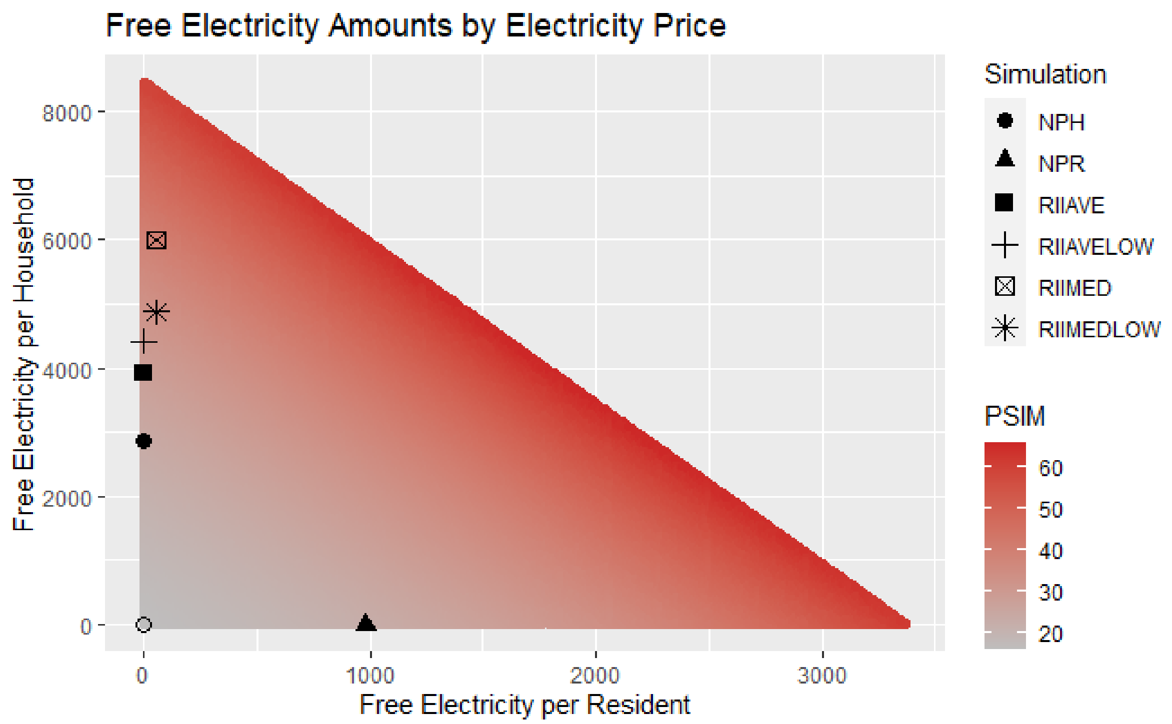
Getting the Simulated Data

Hide

```
savename = "EFRI_Sim"
simpath = paste(getwd(),"/", savename, "_sim.csv", sep = "")
optpath = paste(getwd(),"/", savename, "_opt.csv", sep = "")

simdata = read.csv(simpath)
optdata = read.csv(optpath)
```

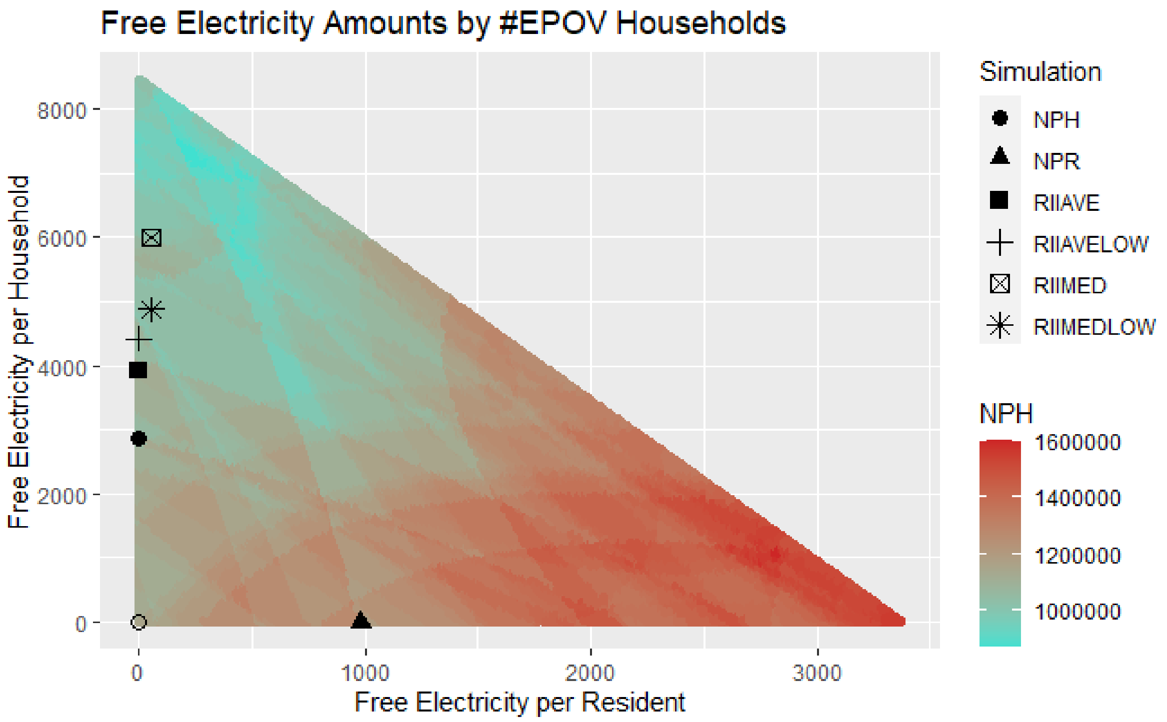
Hide

```
simdata %>%
  ggplot(aes(x=EPR, y=EPH, color=PSIM)) +
  geom_point() +
  scale_color_gradient(low="gray", high="firebrick3") +
  xlab("Free Electricity per Resident") +
  ylab("Free Electricity per Household") +
  ggtitle("Free Electricity Amounts by Electricity Price")+
  geom_point(data=optdata[2:7,],
             aes(x=EPR,y=EPH, shape= Simulation),
             color= "black",
             size=3) +
  geom_point(data=optdata[1,],
             aes(x=EPR,y=EPH),
             color= "black", shape=1,
             size=3)
```



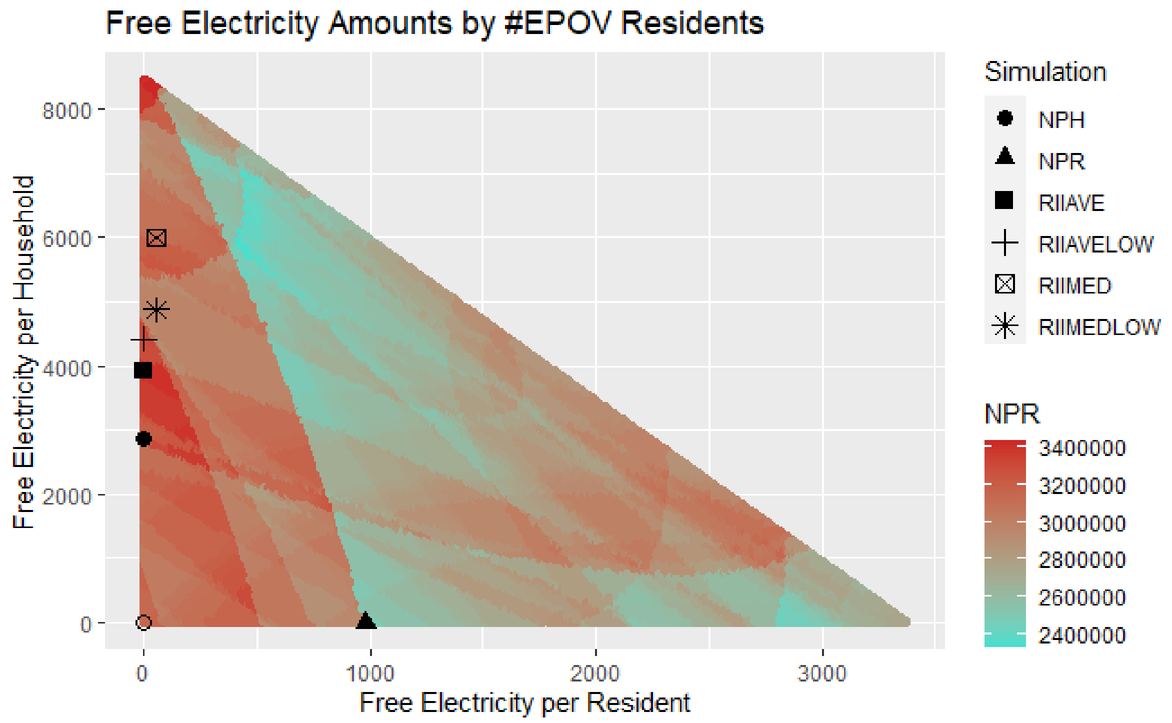
Hide

```
simdata %>%
  ggplot(aes(x=EPR, y=EPH, color=NPH)) +
  geom_point() +
  scale_color_gradient(low="turquoise", high="firebrick3") +
  xlab("Free Electricity per Resident") +
  ylab("Free Electricity per Household") +
  ggtitle("Free Electricity Amounts by #EPOV Households")+
  geom_point(data=optdata[2:7,],
             aes(x=EPR,y=EPH, shape= Simulation),
             color= "black",
             size=3) +
  geom_point(data=optdata[1,],
             aes(x=EPR,y=EPH),
             color= "black", shape=1,
             size=3)
```



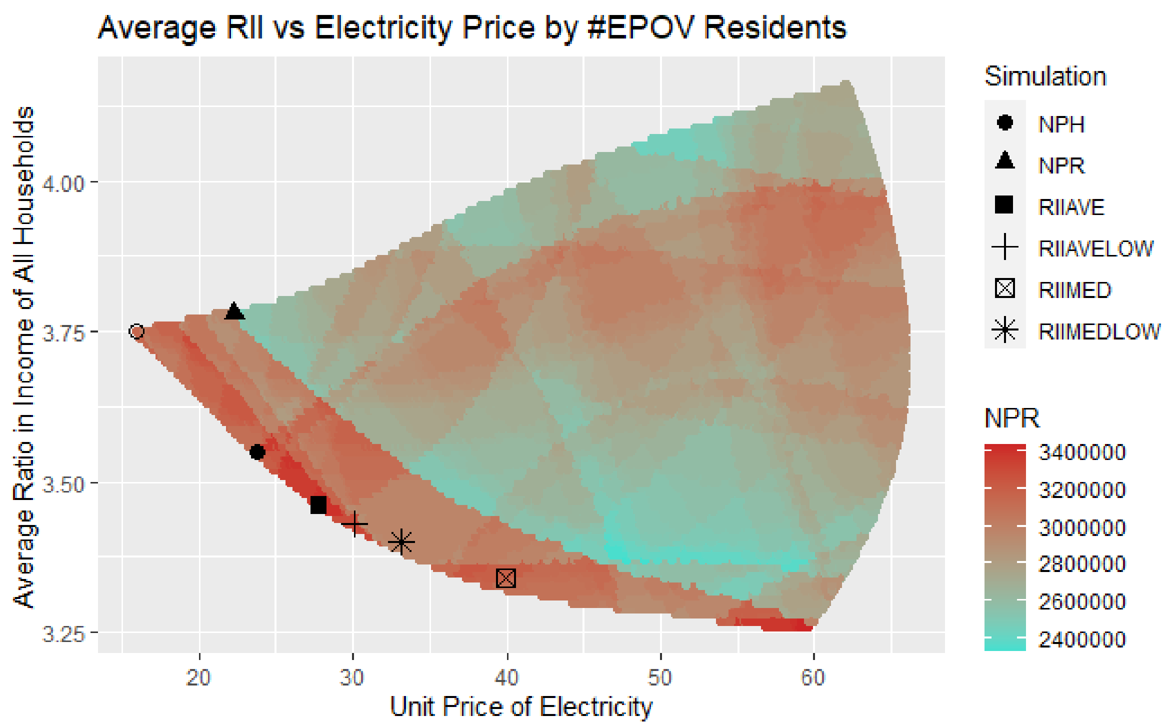
Hide

```
simdata %>%
  ggplot(aes(x=EPR, y=EPH, color=NPR)) +
  geom_point() +
  scale_color_gradient(low="turquoise", high="firebrick3") +
  xlab("Free Electricity per Resident") +
  ylab("Free Electricity per Household") +
  ggtitle("Free Electricity Amounts by #EPOV Residents")+
  geom_point(data=optdata[2:7,],
             aes(x=EPR,y=EPH, shape= Simulation),
             color= "black",
             size=3) +
  geom_point(data=optdata[1,],
             aes(x=EPR,y=EPH),
             color= "black", shape=1,
             size=3)
```



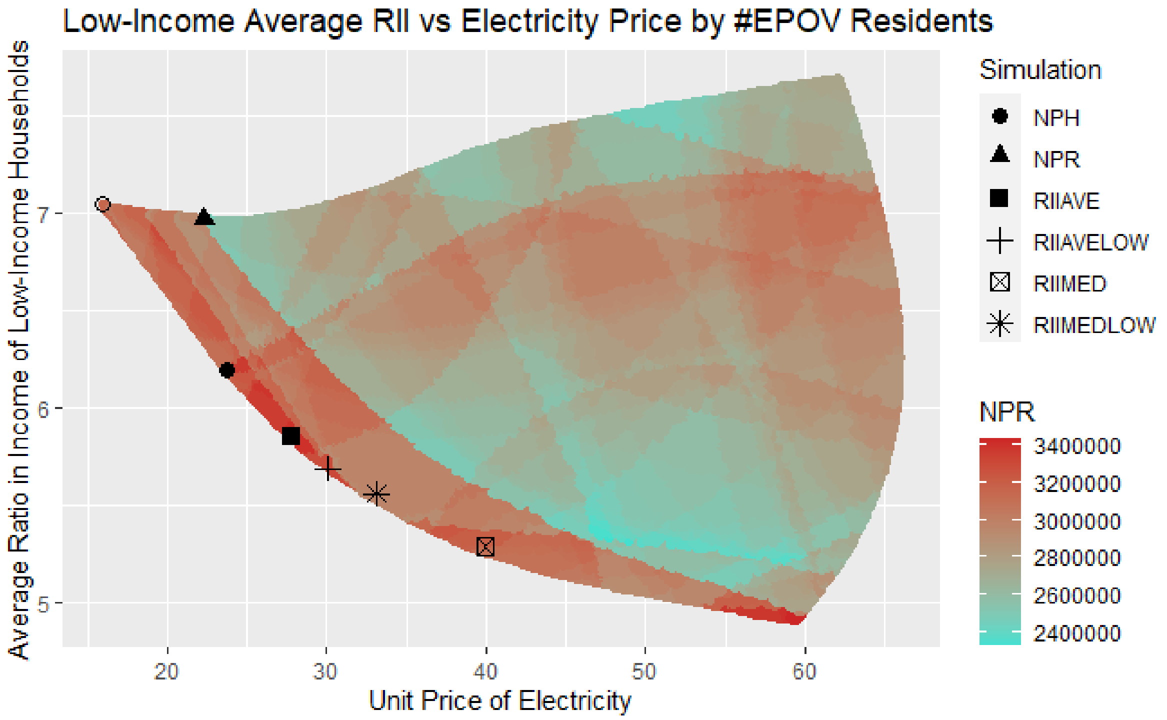
Hide

```
simdata %>%
  ggplot(aes(x=PSIM, y=RIIAVE, color=NPR)) +
  geom_point() +
  scale_color_gradient(low="turquoise", high="firebrick3") +
  xlab("Unit Price of Electricity") +
  ylab("Average Ratio in Income of All Households") +
  ggtitle("Average RII vs Electricity Price by #EPOV Residents")+
  geom_point(data=optdata[2:7,],
             aes(x=PSIM,y=RIIAVE, shape= Simulation),
             color= "black",
             size=3) +
  geom_point(data=optdata[1,],
             aes(x=PSIM,y=RIIAVE),
             color= "black", shape=1,
             size=3)
```



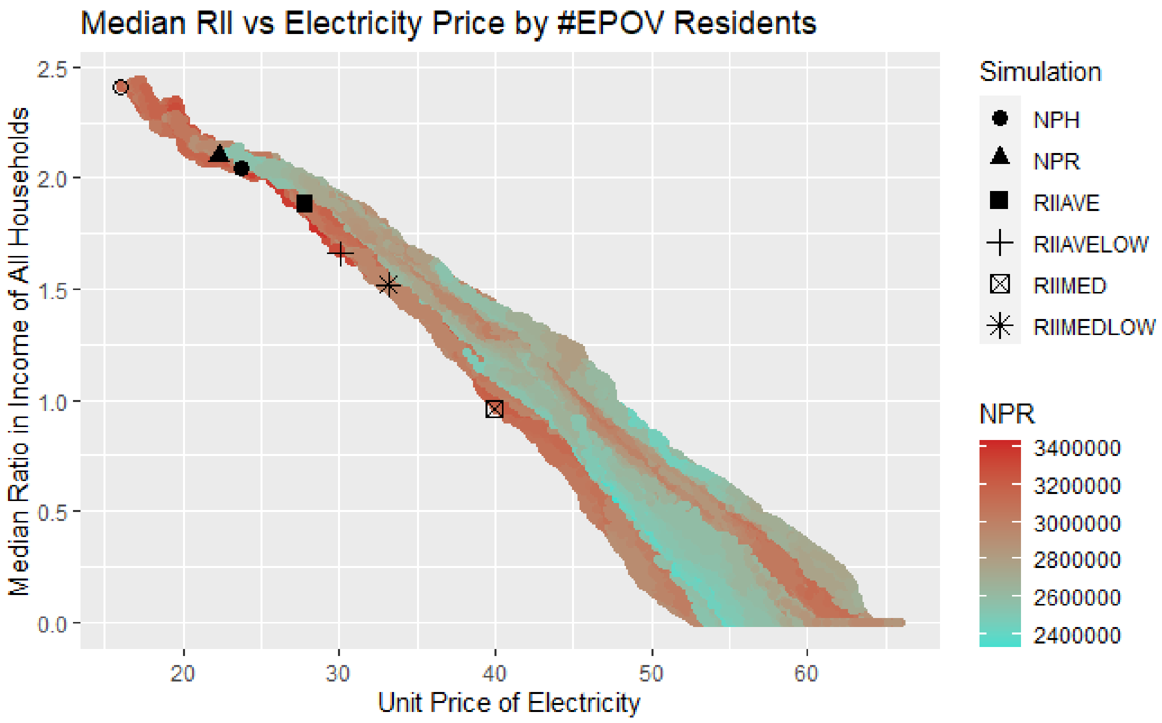
Hide

```
simdata %>%
  ggplot(aes(x=PSIM, y=RIIAVELOW, color=NPR)) +
  geom_point() +
  scale_color_gradient(low="turquoise", high="firebrick3") +
  xlab("Unit Price of Electricity") +
  ylab("Average Ratio in Income of Low-Income Households") +
  ggtitle("Low-Income Average RII vs Electricity Price by #EPOV Residents")+
  geom_point(data=optdata[2:7,],
             aes(x=PSIM,y=RIIAVELOW, shape= Simulation),
             color= "black",
             size=3) +
  geom_point(data=optdata[1,],
             aes(x=PSIM,y=RIIAVELOW),
             color= "black", shape=1,
             size=3)
```



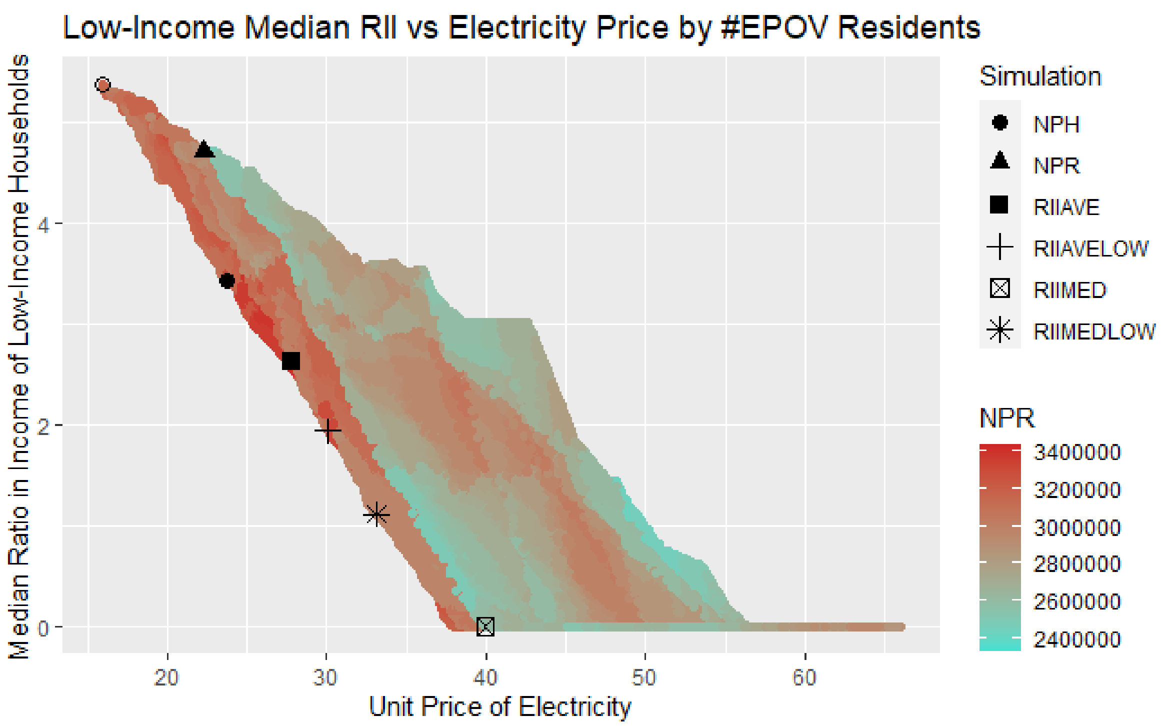
Hide

```
simdata %>%
  ggplot(aes(x=PSIM, y=RIIMED, color=NPR)) +
  geom_point() +
  scale_color_gradient(low="turquoise", high="firebrick3") +
  xlab("Unit Price of Electricity") +
  ylab("Median Ratio in Income of All Households") +
  ggtitle("Median RII vs Electricity Price by #EPOV Residents")+
  geom_point(data=optdata[2:7,],
             aes(x=PSIM,y=RIIMED, shape= Simulation),
             color= "black",
             size=3) +
  geom_point(data=optdata[1,],
             aes(x=PSIM,y=RIIMED),
             color= "black", shape=1,
             size=3)
```



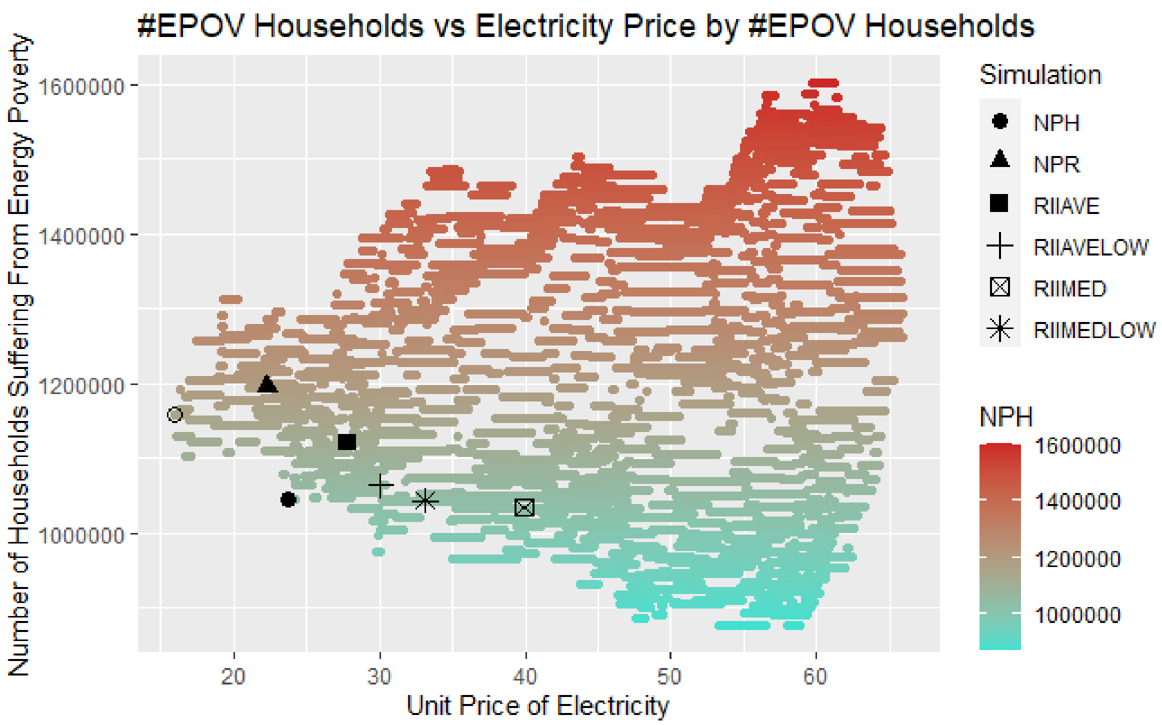
Hide

```
simdata %>%
  ggplot(aes(x=PSIM, y=RIIMEDLOW, color=NPR)) +
  geom_point() +
  scale_color_gradient(low="turquoise", high="firebrick3") +
  xlab("Unit Price of Electricity") +
  ylab("Median Ratio in Income of Low-Income Households") +
  ggtitle("Low-Income Median RII vs Electricity Price by #EPOV Residents")+
  geom_point(data=optdata[2:7,],
             aes(x=PSIM,y=RIIMEDLOW, shape= Simulation),
             color= "black",
             size=3) +
  geom_point(data=optdata[1,],
             aes(x=PSIM,y=RIIMEDLOW),
             color= "black", shape=1,
             size=3)
```



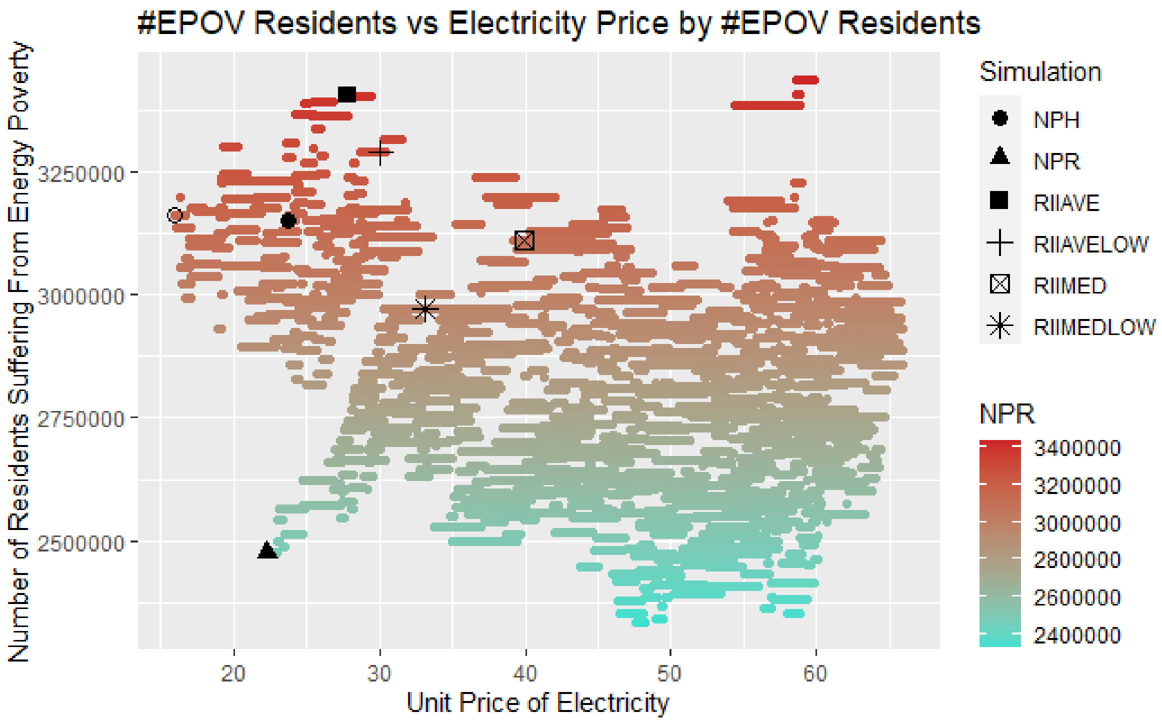
Hide

```
simdata %>%
  ggplot(aes(x=PSIM, y=NPH, color=NPH)) +
  geom_point() +
  scale_color_gradient(low="turquoise", high="firebrick3") +
  xlab("Unit Price of Electricity") +
  ylab("Number of Households Suffering From Energy Poverty") +
  ggtitle("#EPOV Households vs Electricity Price by #EPOV Households")+
  geom_point(data=optdata[2:7,],
             aes(x=PSIM,y=NPH, shape= Simulation),
             color= "black",
             size=3) +
  geom_point(data=optdata[1,],
             aes(x=PSIM,y=NPH),
             color= "black", shape=1,
             size=3)
```



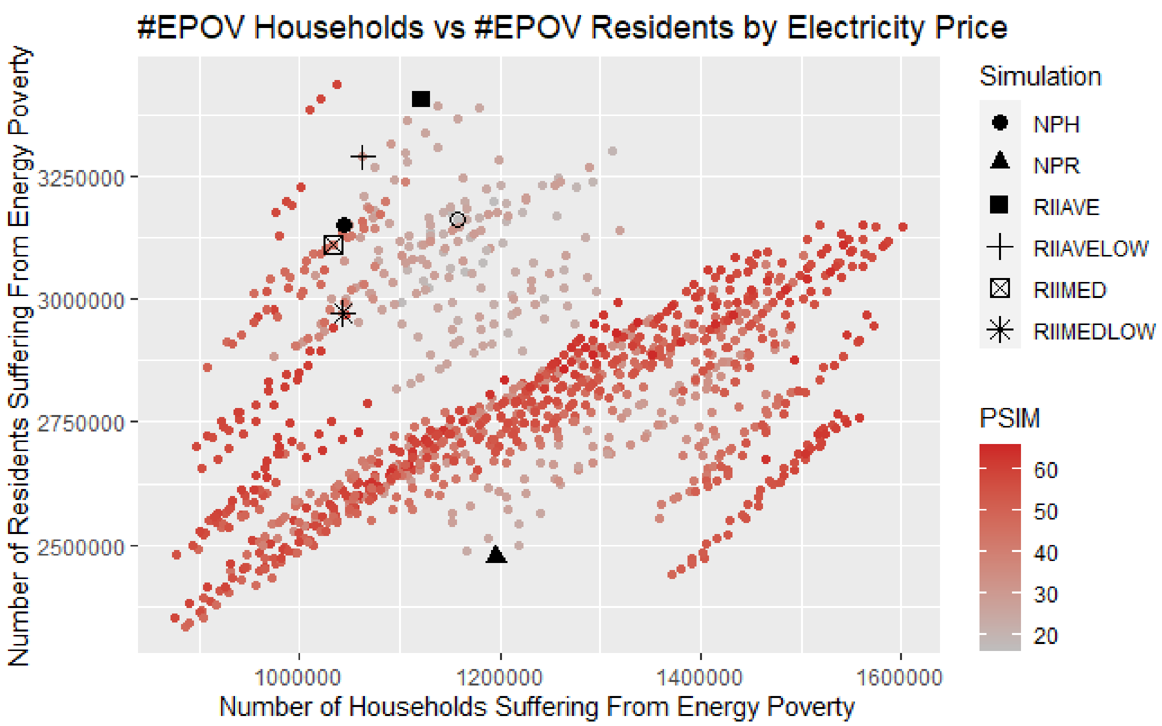
Hide

```
simdata %>%
  ggplot(aes(x=PSIM, y=NPR, color=NPR)) +
  geom_point() +
  scale_color_gradient(low="turquoise", high="firebrick3") +
  xlab("Unit Price of Electricity") +
  ylab("Number of Residents Suffering From Energy Poverty") +
  ggtitle("#EPOV Residents vs Electricity Price by #EPOV Residents")+
  geom_point(data=optdata[2:7,],
             aes(x=PSIM,y=NPR, shape= Simulation),
             color= "black",
             size=3) +
  geom_point(data=optdata[1,],
             aes(x=PSIM,y=NPR),
             color= "black", shape=1,
             size=3)
```



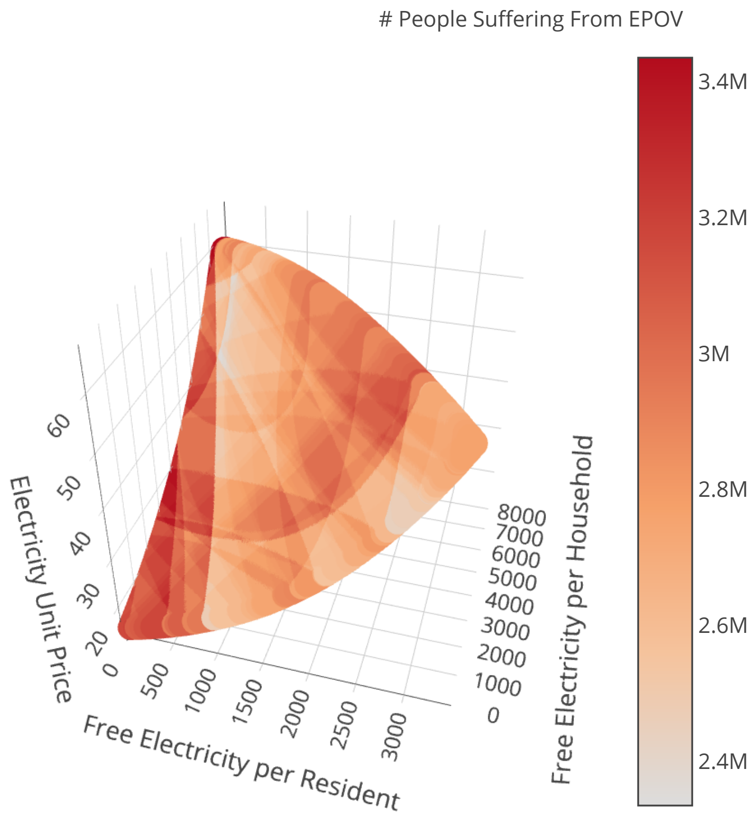
Hide

```
simdata %>%
  ggplot(aes(x=NPH, y=NPR, color=PSIM)) +
  geom_point() +
  scale_color_gradient(low="gray", high="firebrick3") +
  xlab("Number of Households Suffering From Energy Poverty") +
  ylab("Number of Residents Suffering From Energy Poverty") +
  ggtitle("#EPOV Households vs #EPOV Residents by Electricity Price")+
  geom_point(data=optdata[2:7,],
             aes(x=NPH,y=NPR, shape= Simulation),
             color= "black",
             size=3) +
  geom_point(data=optdata[1,],
             aes(x=NPH,y=NPR),
             color= "black", shape=1,
             size=3)
```



Hide

```
sim3d = plot_ly(simdata, x = ~EPR, y = ~EPH, z = ~PSIM,
                         marker = list(color = ~NPR, colorscale = c('#FFE1A1', '#683531'), showscale = T
RUE))
sim3d = sim3d %>% add_markers()
sim3d = sim3d %>% layout(scene = list(xaxis = list(title = 'Free Electricity per Resident'),
                                      yaxis = list(title = 'Free Electricity per Household'),
                                      zaxis = list(title = 'Electricity Unit Price')),
                                annotations = list(
                                  x = 1.13,
                                  y = 1.05,
                                  text = '# People Suffering From EPOV',
                                  xref = 'paper',
                                  yref = 'paper',
                                  showarrow = FALSE
                                ))

sim3d
```



V. Initial vs Simulation Comparison

Selecting the Simulation Case for Comparison

The optimum cases are written into the *optdata* table with the following order:

1. **INITIAL**: Initial case **without granted free electricity**
2. **RIIAVE**: The case where the **population average "ratio in income"** is minimized
3. **RIIAVELOW**: The case where the **low-income average "ratio in income"** is minimized
4. **RIIMED**: The case where the **population median"ratio in income"** is minimized
5. **RIIMEDLOW**: The case where the **low-income median"ratio in income"** is minimized
6. **NPR**: The case where the number of **residents with energy poverty** is minimized
7. **NPH**: The case where the number of **households with energy poverty** is minimized

Hide

```
case = 6 # NPR: The case that minimized number of residents with energy poverty

EPR = optdata[case,"EPR"]
EPH = optdata[case,"EPH"]
PRICEsim = optdata[case,"PSIM"]

EPR
```

```
[1] 982
```

Hide

```
EPH
```

```
[1] 0
```

Hide

```
PRICEsim
```

```
[1] 22.3
```

Adding Necessary Columns to the Initial Data Table

- **PRICEsim**: New Electricity Price after providing free electricity
- **KWHgrant**: Total Granted Free Electricity for a household
- **KWHgrcons**: Consumed Amount of Free Electricity for a household
- **KWHpaid**: Amount of Paid Electricity for the households where the free electricity amount is exceeded
- **RIIsim**: New Ratio in Income for a household

Hide

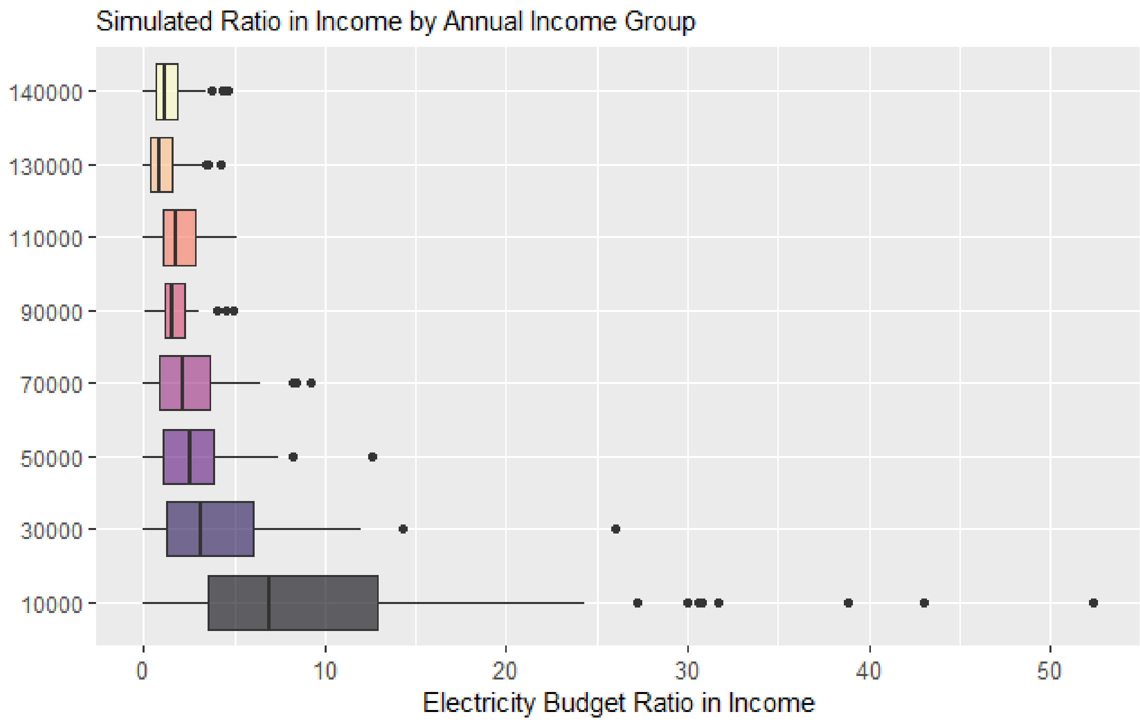
```
comparisondata = cbind(inputdata,PRICEsim)
  names(comparisondata)[length(names(comparisondata))]<-"PRICEsim"

comparisondata = cbind(comparisondata,EPR*comparisondata$`Number of Residents` + EPH)
  names(comparisondata)[length(names(comparisondata))]<-"KWHgrant"

  comparisondata = cbind(comparisondata,pmin(comparisondata$KWHgrant,comparisondata$`Annual Electricity
 Consumption`))
 names(comparisondata)[length(names(comparisondata))]<-"KWHgrcons"

 comparisondata = cbind(comparisondata,comparisondata$`Annual Electricity Consumption` - comparisondata
$KWHgrcons)
 names(comparisondata)[length(names(comparisondata))]<-"KWHpaid"

 comparisondata = cbind(comparisondata,(comparisondata$PRICEsim*comparisondata$KWHpaid)/comparisondata$`
Annual Income`)
 names(comparisondata)[length(names(comparisondata))]<-"RIIsim"

 colnames(comparisondata)
```

```
 [1] "Electricity Price"              "Annual Electricity Consumption"
 [3] "Number of Residents"            "Annual Income"
 [5] "Statistical Multiplier"         "RII"
 [7] "Income Category"                "PRICEsim"
 [9] "KWHgrant"                       "KWHgrcons"
[11] "KWHpaid"                        "RIIsim"
```

Box-Plot of the Simulation Case

Hide

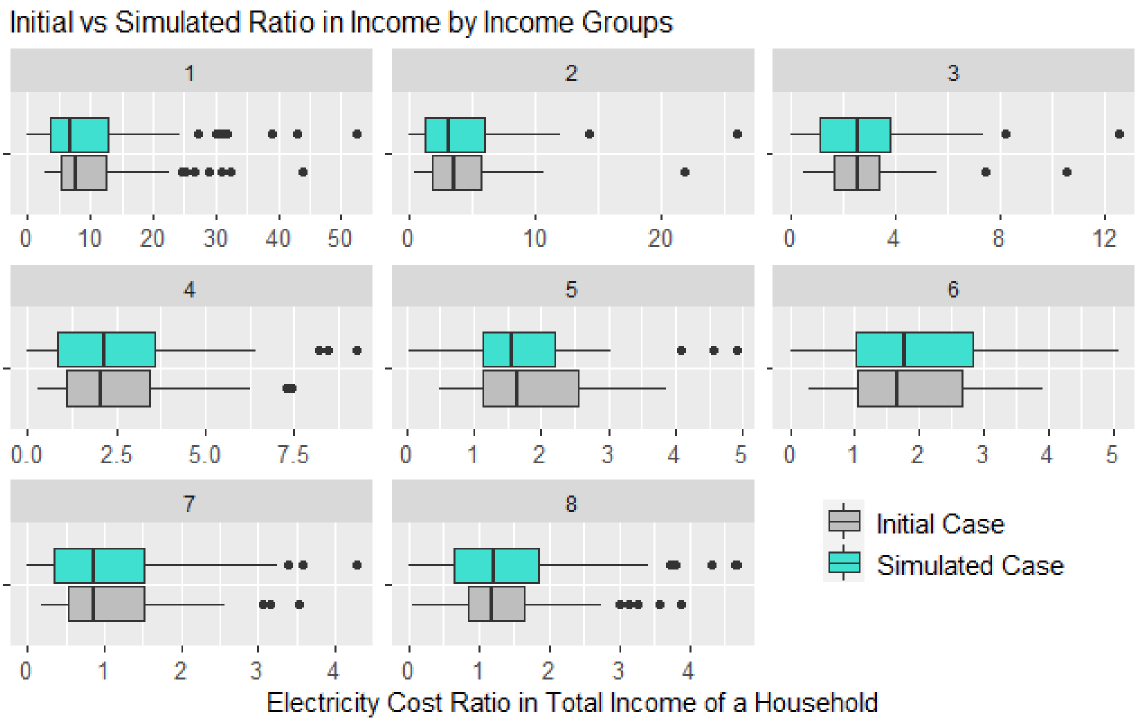
```
comparisondata %>%
   ggplot( aes(x=`Income Category`, y=RIIsim, fill=`Income Category`)) +
   geom_boxplot() +
   scale_fill_viridis(discrete = TRUE, alpha=0.6, option="A") +
   theme_get() +
   theme(axis.text.x = element_text(size=10),
         legend.position="none",
         plot.title = element_text(size=11)
   ) +
   ggtitle("Simulated Ratio in Income by Annual Income Group") +
   xlab("")+
   ylab("Electricity Budget Ratio in Income")+
   coord_flip()
```



Comparison of Each Income Level by Its Initial Situation

Hide

```
   a_ = comparisondata[,"Income Category"] %>% cbind(comparisondata[,"RII"], rep("Initial Case",nrow(com
parisondata)))
   b_ = comparisondata[,"Income Category"] %>% cbind(comparisondata[,"RIIsim"], rep("Simulated Case",nro
w(comparisondata)))

   comparisonplot = data.frame(rbind(a_,b_))
   comparisonplot[,2] = as.numeric(comparisonplot[,2])
   colnames(comparisonplot) = c("Group","RII","Case")

    comparisonplot %>%
     ggplot( aes(x=Group, y=RII, fill=Case)) +
     geom_boxplot() +
     facet_wrap(~Group, scale="free") +
     scale_fill_manual(values = c("grey", "turquoise")) +
     theme_get() +
     theme(axis.text.x = element_text(size=10),
           axis.text.y = element_blank(),
           legend.position=c(0.83,0.2),
           legend.title = element_blank(),
           legend.text = element_text(size=11),
           plot.title = element_text(size=12)
     ) +
    scale_shape_manual(values= c(1,19)) +
     ggtitle("Initial vs Simulated Ratio in Income by Income Groups") +
     xlab("")+
     ylab("Electricity Cost Ratio in Total Income of a Household")+
     coord_flip()
```



Reduction in the Number of Residents with Energy Poverty

1. Whole Population

Hide

```
EPOV_init = comparisondata %>% filter(RII>=10)
EPOV_init_total = sum(EPOV_init$`Statistical Multiplier`*EPOV_init$`Number of Residents`)

EPOV_sim = comparisondata %>% filter(RIIsim>=10)
EPOV_sim_total = sum(EPOV_sim$`Statistical Multiplier`*EPOV_sim$`Number of Residents`)

print("COMPARISON: Number of Total Energy Poor Residents")
```

```
[1] "COMPARISON: Number of Total Energy Poor Residents"
```

Hide

```
print(paste("Initial Case:",round(EPOV_init_total, digits=0)))
```

```
[1] "Initial Case: 3162044"
```

Hide

```
print(paste("Simulated Case:",round(EPOV_sim_total, digits=0)))
```

```
[1] "Simulated Case: 2475819"
```

Hide

```
print(paste("Difference (Number of Residents):",round(EPOV_init_total - EPOV_sim_total, digits=0)))
```

```
[1] "Difference (Number of Residents): 686225"
```

Hide

```
print(paste("Improvement (Number of Residents): %",100*round((EPOV_init_total - EPOV_sim_total)/EPOV_init_total, digits=4)))
```

```
[1] "Improvement (Number of Residents): % 21.7"
```

2. Low-Income Households

Hide

```
EPOV_init = LowIncome %>% filter(RII>=10)
EPOV_init_total = sum(EPOV_init$`Statistical Multiplier`*EPOV_init$`Number of Residents`)


EPOV_sim = LowIncome %>% filter(RIIsim>=10)
EPOV_sim_total = sum(EPOV_sim$`Statistical Multiplier`*EPOV_sim$`Number of Residents`)

print("COMPARISON: Number of Energy Poor Low-Income Residents")
```

```
[1] "COMPARISON: Number of Energy Poor Low-Income Residents"
```

Hide

```
print(paste("Initial Case:",round(EPOV_init_total, digits=0)))
```

```
[1] "Initial Case: 3055576"
```

Hide

```
print(paste("Simulated Case:",round(EPOV_sim_total, digits=0)))
```

```
[1] "Simulated Case: 2369351"
```

Hide

```
print(paste("Difference (Number of Residents):",round(EPOV_init_total - EPOV_sim_total, digits=0)))
```

```
[1] "Difference (Number of Residents): 686225"
```

Hide

```
print(paste("Improvement (Number of Residents): %",100*round((EPOV_init_total - EPOV_sim_total)/EPOV_init_total, digits=4)))
```

```
[1] "Improvement (Number of Residents): % 22.46"
```