Working from Self-driving Car - Results

Georg Hirte

May 08, 2023

#1 Introduction

This markdown file prepares the results of the Monte-Carlo Simulation to the revised paper "Working form self-driving car" resubmitted to TRA. We use GAMS to perform the simulations. Here, we prepare the results for use in the paper.

Initialization

Define function for read in data

Read parameter data

function to read variables with two sets (not "a" or "b")

```
mydataVARcr.frame <- function(gdxfile) {
  colnames <- c("id_i","id_c","id_r");
  id_r <- rgdx.set(gdxfile,"ll",names = colnames,compress=TRUE,ts=TRUE)
  id_c <- rgdx.set(gdxfile,"ctr",names = colnames,compress=TRUE,ts=TRUE)
  id_i <- rgdx.set(gdxfile,"i",names = colnames,compress=TRUE,ts=TRUE)
  alpha.frame <- rgdx.param(gdxfile,"calc_alpha",
                            names = c(colnames[2],colnames[3],"alpha"),
                            compress=TRUE,ts=TRUE,squeeze=FALSE)
  epsilon.frame <- rgdx.param(gdxfile,"calc_epsilon",
                              names = c(colnames[2],colnames[3],"epsilon"),
                              compress=TRUE,ts=TRUE,squeeze=FALSE)
  vc.frame <- rgdx.param(gdxfile,"calc_hvc",
                         names = c(colnames[2],colnames[3],"vc"),
                         compress=TRUE,ts=TRUE,squeeze=FALSE)
  indVx.frame <- rgdx.param(gdxfile,"calc_indVx",
                            names = c(colnames[2],colnames[3],"indVx"),
                            compress=TRUE,ts=TRUE,squeeze=FALSE)
  Ubar.frame <- rgdx.param(gdxfile,"calc_Ubar",
                           names = c(colnames[2],colnames[3],"Ubar"),
                           compress=TRUE,ts=TRUE,squeeze=FALSE)
  wBstar.frame <- rgdx.param(gdxfile,"calc_wBstar",
                             names = c(colnames[2],colnames[3],"wBstar"),
                             compress=TRUE,ts=TRUE,squeeze=FALSE)
  xv.frame <- rgdx.param(gdxfile,"calc_xv",
                         names = c(colnames[2],colnames[3],"xv"),
                         compress=TRUE,ts=TRUE,squeeze=FALSE)
  pb.frame <- rgdx.param(gdxfile,"calc_pb",
                         names = c(colnames[2],colnames[3],"pb"),
                         compress=TRUE,ts=TRUE,squeeze=FALSE)
```

```
  chgmargcost.frame <- rgdx.param(gdxfile,"firm_chg_marg_costs",names = c(colnames[2],colnames[3],"chgma
  chgmargprofit.frame <- rgdx.param(gdxfile,"firm_chg_marg_profit",names = c(colnames[2],colnames[3],"cl
  chgmargprod.frame <- rgdx.param(gdxfile,"firm_chg_margprod",names = c(colnames[2],colnames[3],"chgmarg
  costA.frame <- rgdx.param(gdxfile,"firm_marg_costsA",
                            names = c(colnames[2],colnames[3],"costA"),
                            compress=TRUE,ts=TRUE,squeeze=FALSE)
  costB.frame <- rgdx.param(gdxfile,"firm_marg_costsB",
                            names = c(colnames[2],colnames[3],"costB"),
                            compress=TRUE,ts=TRUE,squeeze=FALSE)
  Deltac.frame <- rgdx.param(gdxfile,"firm_marg_Deltac",
                             names = c(colnames[2],colnames[3],"Deltac"),
                             compress=TRUE,ts=TRUE,squeeze=FALSE)
  Deltafc.frame <- rgdx.param(gdxfile,"firm_marg_Deltafc",
                              names = c(colnames[2],colnames[3],"Deltafc"),
                              compress=TRUE,ts=TRUE,squeeze=FALSE)
  Deltamc.frame <- rgdx.param(gdxfile,"firm_marg_Deltamc",
                              names = c(colnames[2],colnames[3],"Deltamc"),
                              compress=TRUE,ts=TRUE,squeeze=FALSE)

  FrameVARrc.frame <- merge(alpha.frame,epsilon.frame,by=c("id_r", "id_c"))
  FrameVARrc.frame <- merge(FrameVARrc.frame,vc.frame,by=c("id_r", "id_c"))
  FrameVARrc.frame <- merge(FrameVARrc.frame,indVx.frame,by=c("id_r", "id_c"))
  FrameVARrc.frame <- merge(FrameVARrc.frame,Ubar.frame,by=c("id_r", "id_c"))
  FrameVARrc.frame <- merge(FrameVARrc.frame,wBstar.frame,by=c("id_r", "id_c"))
  FrameVARrc.frame <- merge(FrameVARrc.frame,xv.frame,by=c("id_r", "id_c"))
  FrameVARrc.frame <- merge(FrameVARrc.frame,pb.frame,by=c("id_r", "id_c"))
  FrameVARrc.frame <- merge(FrameVARrc.frame,costA.frame,by=c("id_r", "id_c"))
  FrameVARrc.frame <- merge(FrameVARrc.frame,costB.frame,by=c("id_r", "id_c"))
  FrameVARrc.frame <- merge(FrameVARrc.frame,Deltac.frame,by=c("id_r", "id_c"))
  FrameVARrc.frame <- merge(FrameVARrc.frame,Deltafc.frame,by=c("id_r", "id_c"))
  FrameVARrc.frame <- merge(FrameVARrc.frame,Deltamc.frame,by=c("id_r", "id_c"))
  FrameVARrc.frame <- merge(FrameVARrc.frame,chgmargcost.frame,by=c("id_r", "id_c"))
  FrameVARrc.frame <- merge(FrameVARrc.frame,chgmargprofit.frame,by=c("id_r", "id_c"))
  FrameVARrc.frame <- merge(FrameVARrc.frame,chgmargprod.frame,by=c("id_r", "id_c"))
  return(FrameVARrc.frame)
}



### parameters (variables in GAMS simulation) with id_i =a" and "b"
mydataVARab.frame <- function(gdxfile) {
  colnames <- c("id_i","id_c","id_r");
  id_r <- rgdx.set(gdxfile,"ll",names = colnames,compress=TRUE,ts=TRUE)
  id_c <- rgdx.set(gdxfile,"ctr",names = colnames,compress=TRUE,ts=TRUE)
  id_i <- rgdx.set(gdxfile,"i",names = colnames,compress=TRUE,ts=TRUE)
  effort.frame <- rgdx.param(gdxfile,"calc_effort",
                        names = c(colnames[1],colnames[2],colnames[3],"effort"),
                        compress=TRUE,ts=TRUE,squeeze=FALSE)
  ell.frame <- rgdx.param(gdxfile,"calc_ell",
                          names = c(colnames[1],colnames[2],colnames[3],"ell"),
                          compress=TRUE,ts=TRUE,squeeze=FALSE)
  income.frame <- rgdx.param(gdxfile,"calc_income",
```

```
                  names = c(colnames[1],colnames[2],colnames[3],"income"),
                  compress=TRUE,ts=TRUE,squeeze=FALSE)
  incomeB.frame <- as.data.frame(income.frame[effort.frame$id_i =="b" | income.frame$id_i =="b",])
  indV.frame <- rgdx.param(gdxfile,"calc_indV",
                           names = c(colnames[1],colnames[2],colnames[3],"indV"),
                           compress=TRUE,ts=TRUE,squeeze=FALSE)
  v.frame <- rgdx.param(gdxfile,"calc_v",
                        names = c(colnames[1],colnames[2],colnames[3],"v"),
                        compress=TRUE,ts=TRUE,squeeze=FALSE)
  vo.frame <- rgdx.param(gdxfile,"calc_vo",
                         names = c(colnames[1],colnames[2],colnames[3],"vo"),
                         compress=TRUE,ts=TRUE,squeeze=FALSE)
  wB.frame <- rgdx.param(gdxfile,"calc_wB",
                         names = c(colnames[1],colnames[2],colnames[3],"wB"),
                         compress=TRUE,ts=TRUE,squeeze=FALSE)
  xB.frame <- rgdx.param(gdxfile,"calc_xB",
                         names = c(colnames[1],colnames[2],colnames[3],"xB"),
                         compress=TRUE,ts=TRUE,squeeze=FALSE)
  z.frame <- rgdx.param(gdxfile,"calc_z",
                        names = c(colnames[1],colnames[2],colnames[3],"z"),
                        compress=TRUE,ts=TRUE,squeeze=FALSE)
  tkm.frame <- rgdx.param(gdxfile,"calc_tkm",
                          names = c(colnames[1],colnames[2],colnames[3],"tkm"),
                          compress=TRUE,ts=TRUE,squeeze=FALSE)
  expwage.frame <- rgdx.param(gdxfile,"firm_avg_expwage",
                      names = c(colnames[1],colnames[2],colnames[3],"expwage"),
                              compress=TRUE,ts=TRUE,squeeze=FALSE)
#  avgsqm.frame <- rgdx.param(gdxfile,"firm_avg_sqm",
#  names = c(colnames[1],colnames[2],colnames[3],"avgsqm"),
#  compress=TRUE,ts=TRUE,squeeze=FALSE)
  firmlabord.frame <- rgdx.param(gdxfile,"firm_labor_demand",
                     names = c(colnames[1],colnames[2],colnames[3],"firmlabord"),
                                 compress=TRUE,ts=TRUE,squeeze=FALSE)
#  firmsqm.frame <- rgdx.param(gdxfile,"firm_sqm",
#  names = c(colnames[1],colnames[2],colnames[3],"firmsqm"),
# compress=TRUE,ts=TRUE,squeeze=FALSE)
  firmproduction.frame <- rgdx.param(gdxfile,"firm_production",
            names = c(colnames[1],colnames[2],colnames[3],"firmproduction"),
                                     compress=TRUE,ts=TRUE,squeeze=FALSE)
  margprod.frame <- rgdx.param(gdxfile,"firm_margprod",
                      names = c(colnames[1],colnames[2],colnames[3],"margprod"),
                               compress=TRUE,ts=TRUE,squeeze=FALSE)
  netprofits.frame <- rgdx.param(gdxfile,"firm_NetProfits",
        names = c(colnames[1],colnames[2],colnames[3],"netprofits"),
                                 compress=TRUE,ts=TRUE,squeeze=FALSE)
  worker_travelcost.frame <- rgdx.param(gdxfile,"zemployee_travelcost",
            names = c(colnames[1],colnames[2],colnames[3],"worker_travelcost"),
            compress=TRUE,ts=TRUE,squeeze=FALSE)
  worker_traveltime.frame <- rgdx.param(gdxfile,"zemployee_traveltime",
            names = c(colnames[1],colnames[2],colnames[3],"worker_traveltime"),
            compress=TRUE,ts=TRUE,squeeze=FALSE)
  VOT.frame <- rgdx.param(gdxfile,"zemployee_VOT",
                          names = c(colnames[1],colnames[2],colnames[3],"VOT"),
```

```
                          compress=TRUE,ts=TRUE,squeeze=FALSE)
  # generate dataframe with all variables with "a" and "b"
  FrameVARab.frame <- merge(effort.frame,ell.frame,by=c("id_i","id_r", "id_c"))
  FrameVARab.frame <- merge(FrameVARab.frame,income.frame,by=c("id_i","id_r", "id_c"))
  FrameVARab.frame <- merge(FrameVARab.frame,incomeB.frame,by=c("id_i","id_r", "id_c"))
  FrameVARab.frame <- merge(FrameVARab.frame,indV.frame,by=c("id_i","id_r", "id_c"))
  FrameVARab.frame <- merge(FrameVARab.frame,v.frame,by=c("id_i","id_r", "id_c"))
  FrameVARab.frame <- merge(FrameVARab.frame,vo.frame,by=c("id_i","id_r", "id_c"))
  FrameVARab.frame <- merge(FrameVARab.frame,wB.frame,by=c("id_i","id_r", "id_c"))
  FrameVARab.frame <- merge(FrameVARab.frame,xB.frame,
                            by=c("id_i","id_r", "id_c"))
  FrameVARab.frame <- merge(FrameVARab.frame,z.frame,
                            by=c("id_i","id_r", "id_c"))
  FrameVARab.frame <- merge(FrameVARab.frame,tkm.frame,
                            by=c("id_i","id_r", "id_c"))
  FrameVARab.frame <- merge(FrameVARab.frame,expwage.frame,
                            by=c("id_i","id_r", "id_c"))
#  FrameVARab.frame <- merge(FrameVARab.frame,avgsqm.frame,by=c("id_i","id_r", "id_c"))
  FrameVARab.frame <- merge(FrameVARab.frame,firmlabord.frame,
                            by=c("id_i","id_r", "id_c"))
#  FrameVARab.frame <- merge(FrameVARab.frame,firmsqm.frame,by=c("id_i","id_r", "id_c"))
  FrameVARab.frame <- merge(FrameVARab.frame,firmproduction.frame,
                            by=c("id_i","id_r", "id_c"))
  FrameVARab.frame <- merge(FrameVARab.frame,margprod.frame,
                            by=c("id_i","id_r", "id_c"))
  FrameVARab.frame <- merge(FrameVARab.frame,netprofits.frame,
                            by=c("id_i","id_r", "id_c"))
  FrameVARab.frame <- merge(FrameVARab.frame,worker_travelcost.frame,
                            by=c("id_i","id_r", "id_c"))
  FrameVARab.frame <- merge(FrameVARab.frame,worker_traveltime.frame,
                            by=c("id_i","id_r", "id_c"))
  FrameVARab.frame <- merge(FrameVARab.frame,VOT.frame,
                            by=c("id_i","id_r", "id_c"))
  return(FrameVARab.frame)
}

### df1.frame
df1.frame <- function(gdxfile) {
  MobileVARcr.frame <- mydataVARcr.frame(gdxfile)
  MobilePARAM.frame <- mydataPARAM.frame(gdxfile)
  MobileVARab.frame <- mydataVARab.frame(gdxfile)
  # generate file with only contract 'b'
  Mobileonlyb.frame <-
    as.data.frame(MobileVARab.frame[MobileVARab.frame$id_i =="a" | MobileVARab.frame$id_i =="b",])
  Mobileb.frame <- merge(MobilePARAM.frame, MobileVARcr.frame,
                         by=c("id_r", "id_c"))
  Mobileb.frame <- merge(Mobileb.frame, Mobileonlyb.frame,
                         by=c("id_r", "id_c"))
  # produce data.frame for table
  df1.frame <- Mobileb.frame
  return(df1.frame)
}
```

```
#gdxfile = "OutMobileWork_eff.gdx"
#nameparam = "eff"
#nameidc = "DE"
dfprintidc <- function(gdxfile,nameparam,nameidc) {
  df.frame <- df1.frame(gdxfile)
  df2.frame <- select(filter(df.frame, id_c == nameidc),
                      c(id_r,id_c,all_of(nameparam),
                        alpha,v,vc,vo,xB,wage,wB,pb,Deltac))
  df2.frame <- df2.frame %>% arrange(id_r) #sort data
  xtab1 <- xtable(df2.frame[1:10, 3:10])
  text0 <- nameparam
  text1 = "Parameter"
  text2 <- nameidc
  text1 <- paste(text1,text0,text2)
  textn0 = "./DataParam"
  textn1 <- nameidc
  textn2 = ".txt"
  textn <- paste(textn0,text0,textn1,textn2,sep="")
  textl1 <- nameidc
  textl01 = " "
  textl0 = "tab: "
  textlab <- paste(textl0,text0,textl01,textl1,sep="")
  pprint <- print(xtable(xtab1, caption = text1, label=textlab)
                  ,floating = TRUE, table.placement = "htb",
                  size = "footnotesize", latex.environments = "center",
                  math.style.negative = "TRUE", file = textn)
  return(pprint)
}



# print elasticites in Tex table Germany
#c("eff","beta","sqmemployee","rmonth","dB","dA","sp","xbar","gm","gx","gh","tb","wage")
#gdxfile = "OutMobileWork_eff.gdx"
#nameparam = "eff"
#nameidc = "DE"
dfprintElast <- function(gdxfile,nameparam,nameidc) {
  df.frame <- df1.frame(gdxfile)
  df2.frame <- select(filter(df.frame, id_c == nameidc),
                      c(id_r,id_c,all_of(nameparam), alpha,v,vc,vo,xB))
  df2.frame <- df2.frame %>% arrange(id_r) #sort data
  df2elast.frame <- data.frame(matrix(ncol = 8, nrow = 10))
  colnames(df2elast.frame) <- c("id_r","id_c",nameparam,"e_alpha","e_v","e_vc",
                                "e_vo", "e_xB")
  df2elast.frame <- data.frame(matrix(vector(),10,8,dimnames=list(c(),
              c("id_r","id_c",nameparam,"e_alpha","e_v","e_vc","e_vo", "e_xB"))),
                               stringsAsFactors=F)
  df2elast.frame$id_r <-  df2.frame$id_r
  df2elast.frame$id_c <-  df2.frame$id_c
  df2elast.frame[,3] <-  df2.frame[,3]
  for(i in 2:10){
    df2elast.frame$e_alpha[i] =
      ((df2.frame$alpha[i]-df2.frame$alpha[i-1])/df2.frame$alpha[i-1])/
```

```
      ((df2.frame[i,3]-df2.frame[i-1,3])/df2.frame[i-1,3])
    df2elast.frame$e_v[i] =
      ((df2.frame$v[i]-df2.frame$v[i-1])/df2.frame$v[i-1])/
      ((df2.frame[i,3]-df2.frame[i-1,3])/df2.frame[i-1,3])
    df2elast.frame$e_vc[i] =
      ((df2.frame$vc[i]-df2.frame$vc[i-1])/df2.frame$vc[i-1])/
      ((df2.frame[i,3]-df2.frame[i-1,3])/df2.frame[i-1,3])
    df2elast.frame$e_vo[i] =
      ((df2.frame$vo[i]-df2.frame$vo[i-1])/df2.frame$vo[i-1])/
      ((df2.frame[i,3]-df2.frame[i-1,3])/df2.frame[i-1,3])
    df2elast.frame$e_xB[i] =
      ((df2.frame$xB[i]-df2.frame$xB[i-1])/df2.frame$xB[i-1])/
      ((df2.frame[i,3]-df2.frame[i-1,3])/df2.frame[i-1,3])
  }
  str(df2elast.frame)
  # add median
  df2elast.frame <- rbind(df2elast.frame,c("median",nameidc,nameparam,
                          median(df2elast.frame$e_alpha,sort=TRUE,na.rm=TRUE),
                          median(df2elast.frame$e_v,sort=TRUE,na.rm=TRUE),
                          median(df2elast.frame$e_vc,sort=TRUE,na.rm=TRUE),
                          median(df2elast.frame$e_vo,sort=TRUE,na.rm=TRUE),
                          median(df2elast.frame$e_xB,sort=TRUE,na.rm=TRUE)))

# transform to numeric
  df2elast.frame$e_alpha <- as.numeric(df2elast.frame$e_alpha)
  df2elast.frame$e_v <- as.numeric(df2elast.frame$e_v)
  df2elast.frame$e_vc <- as.numeric(df2elast.frame$e_vc)
  df2elast.frame$e_vo <- as.numeric(df2elast.frame$e_vo)
  df2elast.frame$e_xB <- as.numeric(df2elast.frame$e_xB)

  number_hook <- function(x) {
  ifelse(abs(x) < 100 & abs(x) > 0.000,
         prettyNum(x, small.mark = ",", digits = 2), trunc(x))
  }
  xtable :: xtable(dplyr::mutate_if(df2elast.frame, is.numeric, number_hook))
  xtab1 <- xtable(dplyr::mutate_if(df2elast.frame, is.numeric, number_hook))
  #xtab1 <- xtable(df2elast.frame[2:10, 1:8],digits=2)
  text0 <- nameparam
  text1 = "Parameter "
  text2 <- nameidc
  text1 <- paste(text1,text0,text2)
  textn0 = "./ParamElast"
  textn1 <- nameidc
  textn2 = ".txt"
  textn <- paste(textn0,text0,textn1,textn2,sep="")
  textl1 = " "
  textl2 <- nameidc
  textl0 = "tab: "
  textlab <- paste(textl0,text0,textl1,sep="")
  pprintelast <- print(xtable(xtab1, caption = text1, label=textlab,digits=2)
                  ,floating = TRUE, table.placement = "htb",
                  size = "footnotesize", latex.environments = "center",
                  math.style.negative = "TRUE", file = textn)
```

```
  return(pprintelast)
}
```

UDF: calcprob

function calculate probabilities * Monte Carlo Table * calculate all variables as between 0-1 (shares) * sort them, calculate the empirical cumulative density function * and print probabilities

Function to generade ECDF

UDF: dfmcframe

generate data frame from Monte Carlo results

UDF dfprintMonteCarlo

print latex table for MC

```
dfprintMonteCarlo <- function(gdxfile,nameparam, nameidc,asupport) {
    #dfmc.frame <- dfmc.frame %>% arrange(id_r) #sort data
  # prepare data for output table
  dfmc.frame <- dfmcframe(gdxfile,nameparam, nameidc,asupport)
  value <- c(0,0.1,0.2,0.25, 0.3,0.4,0.5,0.6,0.7,0.75,0.8,0.9,1,"median","mean")
  digit = 4
  namevar <- dfmc.frame$alpha
  alpha <- calcprob(namevar,digit,1)
  # call function to calculate probabilities and cumulative density ecdf
  namevar <- dfmc.frame$share_v
  sv <- calcprob(namevar,digit,1)
  namevar <- dfmc.frame$share_vc
  svc <- calcprob(namevar,digit,1)
  namevar <- dfmc.frame$share_vo
  svo <- calcprob(namevar,digit,1)
  namevar <- dfmc.frame$share_xsum
  sxsum <- calcprob(namevar,digit,1)
  namevar <- dfmc.frame$share_x
  sx <- calcprob(namevar,digit,1)
  namevar <- dfmc.frame$share_xo
  sxo <- calcprob(namevar,digit,1)


  #xtab1.frame <- data.frame(value,alpha,sv,svc,svo,sxsum,sx,sxo)
  xtab1.frame <- data.frame(value,alpha,sv,svc,svo,sx,sxo)
  summary(xtab1.frame)
  xtab1 <- xtable(xtab1.frame, rownames=F,)

  if (nameidc =="DE") {
    text1 <- "Monte-Carlo Simulation: DE, ["
  } else {
    text1 <- "Monte-Carlo Simulation: US, ["
  }
  text2 <- asupport
  text3 <- "]"
  text1 <- paste(text1,text2,text3)
  if (nameidc =="DE") {
```

```
    textn0 <- "./DataMC_DE.txt"
  } else {
    textn0 <- "./DataMC_US.txt"
  }
  #textn <- paste(textn0)
  pprintMC <- print(xtable(xtab1,  caption = text1), include.rownames=F
                    ,floating = TRUE, latex.environments = "center", file = textn0)
  return(pprintMC)
}
```

Evaluate results

PARAMETER VARIATION

Calculate results for single parameter variations

```
select1 <- c("eff","beta","sqmemployee","rmonth","dB","dA","sp","xbar","gm",
             "gx","gh","tb","wage")
for(nameparam in select1){
    text0 <- "OutMobileWork_"
    text1 <- nameparam
    text2 <- ".gdx"
    gdxfile <- paste(text0,text1,text2,sep="")
#    gdxfile = "OutMobileWork_eff.gdx"
#  nameparam <- select1
  for(nameidc in c("DE","US")) {
    dfpDE <- dfprintidc(gdxfile,nameparam,nameidc)
    dfpelastDE <- dfprintElast(gdxfile,nameparam,nameidc)
  }
}
```

```
# print results for policy parameters
select1 <- c("tauw","rho","tauq","taup","taud","taus","tauf","tauc")
for(nameparam in select1){
    text0 <- "OutMobileWork_"
    text1 <- nameparam
    text2 <- ".gdx"
    gdxfile <- paste(text0,text1,text2,sep="")
#    gdxfile = "OutMobileWork_eff.gdx"
#  nameparam <- select1
  for(nameidc in c("DE","US")) {
    dfpDE <- dfprintidc(gdxfile,nameparam,nameidc)
    dfpelastDE <- dfprintElast(gdxfile,nameparam,nameidc)
  }
}
```

Summarize Results of MC Simulations

Calculate probabilities

Plot cumulative densities

Evaluate results: Regressions and Interactions

```
# For dev version - install package to make tables1

library(MASS)
#library(knitr)
#library(texreg)
library(censReg)
```

```
## Lade nötiges Paket: maxLik
```

```
## Lade nötiges Paket: miscTools
```

```
##
## Please cite the 'maxLik' package as:
## Henningsen, Arne and Toomet, Ott (2011). maxLik: A package for maximum likelihood estimation in R. C
##
## If you have questions, suggestions, or comments regarding the 'maxLik' package, please use a forum o
## https://r-forge.r-project.org/projects/maxlik/
```

```
##
## Please cite the 'censReg' package as:
## Henningsen, Arne (2017). censReg: Censored Regression (Tobit) Models. R package version 0.5. http://
##
## If you have questions, suggestions, or comments regarding the 'censReg' package, please use a forum
## https://r-forge.r-project.org/projects/sampleselection/
```

```
library(stargazer)
```

```
##
## Please cite as:
```

```
##  Hlavac, Marek (2022). stargazer: Well-Formatted Regression and Summary Statistics Tables.
```

```
##  R package version 5.2.3. https://CRAN.R-project.org/package=stargazer
```

```
stargazer(dfmcDE2.frame)
```

% Table created by stargazer v.5.2.3 by Marek Hlavac, Social Policy Institute. E-mail: marek.hlavac at gmail.com % Date and time: Di, Mai 16, 2023 - 12:31:41

```
stargazer(dfmcUS2.frame)
```

% Table created by stargazer v.5.2.3 by Marek Hlavac, Social Policy Institute. E-mail: marek.hlavac at gmail.com % Date and time: Di, Mai 16, 2023 - 12:31:41

```
# add rent per sqm/employee to dataframe
dfmcDE2.frame <- mutate(dfmcDE2.frame, r = rmonth*sqmemployee)
# add rent per sqm/employee to dataframe
dfmcUS2.frame <- mutate(dfmcUS2.frame, r = rmonth*sqmemployee)

regshare_vDE <- censReg(share_v~log(wage) + log(tkm) + log(eff) + log(xbar)
                 + log(gm)
                 + beta
                 + log(r) + log(gx) + log(gh)
                 + log(sp)  + dB,  left=0, right=1, data = dfmcDE2.frame)
summary(regshare_vDE)
```

Call: censReg(formula = share_v ~ log(wage) + log(tkm) + log(eff) + log(xbar) + log(gm) + beta + log(r) + log(gx) + log(gh) + log(sp) + dB, left = 0, right = 1, data = dfmcDE2.frame)

Table 1:

Statistic	N	Mean	St. Dev.	Min	Max
adistrib	10,000	45.000	0.000	45	45
alpha	10,000	0.726	0.277	0.000	1.000
share_v	10,000	0.477	0.199	0.182	1.000
share_vc	10,000	0.134	0.128	0.000	0.776
share_vo	10,000	0.343	0.156	0.182	1.000
share_xsum	10,000	80.892	3,192.876	0.004	273,974.400
share_x	10,000	0.109	0.227	0.000	1.000
share_xo	10,000	80.782	3,192.868	0.004	273,974.400
v	10,000	3.817	1.588	1.455	8.000
vc	10,000	1.076	1.027	0.000	6.208
vo	10,000	2.741	1.248	1.455	8.000
xbar	10,000	21.099	31.716	0.00000	332.863
xB	10,000	0.644	2.484	0.000	83.144
xv	10,000	16.453	17.569	0.002	204.247
wage	10,000	176.552	98.403	24.590	1,108.667
wB	10,000	149.269	101.469	5.000	1,129.978
pb	10,000	2.255	2.217	0.000	7.016
Deltac	10,000	1.106	10.269	−56.423	43.424
tkm	10,000	1.353	4.820	0.016	26.731
expwage	10,000	123.888	102.978	−31.522	1,107.473
hours	10,000	8.000	0.000	8	8
tauw	10,000	0.484	0.000	0.484	0.484
tauf	10,000	0.000	0.000	0	0
taus	10,000	0.000	0.000	0	0
taud	10,000	0.000	0.000	0	0
tauc	10,000	0.000	0.000	0	0
taup	10,000	0.000	0.000	0	0
tauq	10,000	0.500	0.000	0.500	0.500
rho	10,000	0.200	0.000	0.200	0.200
eff	10,000	1.998	0.409	1.009	2.983
beta	10,000	−0.178	20.495	−211.878	187.541
dA	10,000	10.000	0.000	10	10
dB	10,000	9.971	2.048	5.011	14.936
gm	10,000	0.757	0.201	0.306	1.208
gx	10,000	0.047	0.005	0.038	0.055
gh	10,000	4.005	1.709	1.040	7.000
rmonth	10,000	21.391	8.412	8.274	45.285
sp	10,000	0.498	0.206	0.007	0.993
sqmemployee	10,000	26.830	4.659	18.807	34.898
tb	10,000	3.026	0.573	2.036	4.019

Table 2:

Statistic	N	Mean	St. Dev.	Min	Max
adistrib	10,000	60.000	0.000	60	60
alpha	10,000	0.687	0.304	0.000	1.000
share_v	10,000	0.391	0.166	0.184	1.000
share_vc	10,000	0.051	0.056	0.000	0.801
share_vo	10,000	0.340	0.159	0.182	1.000
share_xsum	10,000	173,443.400	12,125,841.000	0.038	1,192,170,945.000
share_x	10,000	0.102	0.266	0.000	1.000
share_xo	10,000	173,443.300	12,125,841.000	0.023	1,192,170,945.000
v	10,000	3.129	1.331	1.473	8.000
vc	10,000	0.408	0.449	0.000	6.408
vo	10,000	2.721	1.270	1.454	8.000
xbar	10,000	30.688	55.487	0.000	668.988
xB	10,000	0.799	5.765	0.000	259.296
xv	10,000	62.594	86.212	0.001	888.484
wage	10,000	205.760	127.099	3.939	1,100.362
wB	10,000	194.824	128.760	5.000	1,137.274
pb	10,000	5.948	4.140	0.000	10.501
Deltac	10,000	−6.516	17.360	−128.986	96.961
tkm	10,000	2.707	7.254	0.007	26.689
expwage	10,000	163.325	130.025	−51.918	1,081.960
hours	10,000	8.000	0.000	8	8
tauw	10,000	0.202	0.000	0.202	0.202
tauf	10,000	0.000	0.000	0	0
taus	10,000	0.000	0.000	0	0
taud	10,000	0.000	0.000	0	0
tauc	10,000	0.000	0.000	0	0
taup	10,000	0.000	0.000	0	0
tauq	10,000	0.500	0.000	0.500	0.500
rho	10,000	0.200	0.000	0.200	0.200
eff	10,000	2.010	0.405	1.027	2.986
beta	10,000	−0.263	24.549	−157.191	159.641
dA	10,000	10.000	0.000	10	10
dB	10,000	10.032	2.026	5.116	14.921
gm	10,000	0.743	0.215	0.261	1.226
gx	10,000	0.047	0.014	0.024	0.071
gh	10,000	7.687	0.194	7.247	8.125
rmonth	10,000	91.125	30.299	22.842	159.669
sp	10,000	0.500	0.204	0.004	0.994
sqmemployee	10,000	18.557	5.376	9.298	27.868
tb	10,000	1.077	1.952	0.055	13.475

Observations: Total Left-censored Uncensored Right-censored 10000 0 9698 302

Coefficients: Estimate Std. error t value Pr(> t)
(Intercept) 9.456e-01 3.235e-02 29.229 <2e-16 ***log(wage) 8.002e-04 2.126e-03 0.376 0.7067***
log(tkm) 1.090e-01 2.244e-03 48.564 <2e-16 ** log(eff) -6.829e-01 4.470e-03 -152.776 <2e-16 ***log(xbar) 1.016e-01 1.406e-03 72.302 <2e-16 ***** log(gm) -3.344e-03 3.305e-03 -1.012 0.3115
beta 9.937e-05 4.608e-05 2.156 0.0311 *
log(r) -2.757e-04 2.165e-03 -0.127 0.8987
log(gx) -1.082e-02 8.697e-03 -1.244 0.2136
log(gh) 1.319e-03 1.878e-03 0.703 0.4823
log(sp) -1.072e-03 1.742e-03 -0.616 0.5381
dB -3.123e-04 4.613e-04 -0.677 0.4983
logSigma -2.363e+00 7.236e-03 -326.516 <2e-16 *** — Signif. codes: 0 ‘***0.001**’ *0.01* ” 0.05 ‘.’ 0.1 ’ ’ 1

Newton-Raphson maximisation, 9 iterations Return code 8: successive function values within relative tolerance limit (reltol) Log-likelihood: 8786.028 on 13 Df

```
regshare_vDEm <- margEff(regshare_vDE ,vcov = NULL,
   calcVCov = TRUE, returnJacobian = FALSE)
summary(regshare_vDEm)
```

```
       Marg. Eff.  Std. Error   t value Pr(>|t|)
```

log(wage) 8.0016e-04 2.1261e-03 0.3763 0.70667
log(tkm) 1.0898e-01 2.2440e-03 48.5645 < 2e-16 ***log(eff) -6.8289e-01 4.4699e-03 -152.7755 < 2e-16***
log(xbar) 1.0164e-01 1.4057e-03 72.3020 < 2e-16 ** *log(gm) -3.3445e-03 3.3045e-03 -1.0121 0.31152*
beta 9.9366e-05 4.6081e-05 2.1563 0.03108
log(r) -2.7570e-04 2.1649e-03 -0.1274 0.89867
log(gx) -1.0816e-02 8.6975e-03 -1.2436 0.21366
log(gh) 1.3193e-03 1.8778e-03 0.7026 0.48234
log(sp) -1.0724e-03 1.7416e-03 -0.6157 0.53808
dB -3.1233e-04 4.6126e-04 -0.6771 0.49834
— Signif. codes: 0 ‘***0.001**’ *0.01* ” 0.05 ‘.’ 0.1 ’ ’ 1

```
regshare_vcDE <- censReg(share_vc~log(wage) + log(tkm) + log(eff) + log(xbar)
                 + log(gm)
                 + beta
                 + log(r) + log(gx) + log(gh)
                 + log(sp)  + dB,  left=0, right=1, data = dfmcDE2.frame)
 summary(regshare_vcDE)
```

Call: censReg(formula = share_vc ~ log(wage) + log(tkm) + log(eff) + log(xbar) + log(gm) + beta + log(r) + log(gx) + log(gh) + log(sp) + dB, left = 0, right = 1, data = dfmcDE2.frame)

Observations: Total Left-censored Uncensored Right-censored 10000 527 9473 0

Coefficients: Estimate Std. error t value Pr(> t)
(Intercept) 1.769e-01 2.217e-02 7.979 1.48e-15 ***log(wage) -1.354e-03 1.460e-03 -0.928 0.3535***
log(tkm) 1.424e-01 1.654e-03 86.099 < 2e-16 ** log(eff) 6.097e-03 3.020e-03 2.019 0.0435 *
log(xbar) 1.314e-01 1.055e-03 124.520 < 2e-16 ***log(gm) -3.454e-03 2.264e-03 -1.526 0.1271***
beta 2.409e-04 3.176e-05 7.585 3.32e-14 log(r) 1.339e-03 1.484e-03 0.902 0.3668
log(gx) -1.450e-03 5.957e-03 -0.243 0.8076
log(gh) 1.730e-03 1.285e-03 1.347 0.1780
log(sp) 2.302e-04 1.192e-03 0.193 0.8469
dB 4.472e-05 3.161e-04 0.141 0.8875
logSigma -2.750e+00 7.308e-03 -376.296 < 2e-16 *** — Signif. codes: 0 ‘***0.001**’ *0.01* ” 0.05 ‘.’ 0.1 ’ ’ 1

Newton-Raphson maximisation, 9 iterations Return code 8: successive function values within relative tolerance

limit (reltol) Log-likelihood: 12222.13 on 13 Df

```
regshare_vcDEm <- margEff(regshare_vcDE ,vcov = NULL,
   calcVCov = TRUE, returnJacobian = FALSE)
summary(regshare_vcDEm)
```

```
       Marg. Eff.  Std. Error  t value  Pr(>|t|)
```

log(wage) -1.3241e-03 1.4271e-03 -0.9279 0.35349
log(tkm) 1.3928e-01 1.6266e-03 85.6249 < 2.2e-16 ***log(eff) 5.9607e-03 2.9524e-03 2.0189 0.04352*** **log(xbar) 1.2845e-01 1.0396e-03 123.5583 < 2.2e-16** *log(gm) -3.3767e-03 2.2135e-03 -1.5255 0.12716 beta 2.3555e-04 3.1052e-05 7.5855 3.597e-14* ** log(r) 1.3093e-03 1.4508e-03 0.9024 0.36684
log(gx) -1.4182e-03 5.8248e-03 -0.2435 0.80765
log(gh) 1.6918e-03 1.2562e-03 1.3468 0.17808
log(sp) 2.2505e-04 1.1652e-03 0.1931 0.84685
dB 4.3726e-05 3.0904e-04 0.1415 0.88749
— Signif. codes: 0 ‘***’ ***0.001*** ‘**’ *0.01* ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

```
regshare_voDE <- censReg(share_vo ~ log(wage) + log(tkm) + log(eff) + log(xbar)
                + log(gm)
                 + beta
                 + log(r) + log(gx) + log(gh)
                 + log(sp) + dB,  right=1, data = dfmcDE2.frame)
summary(regshare_voDE)
```

Call: censReg(formula = share_vo ~ log(wage) + log(tkm) + log(eff) + log(xbar) + log(gm) + beta + log(r) + log(gx) + log(gh) + log(sp) + dB, right = 1, data = dfmcDE2.frame)

Observations: Total Left-censored Uncensored Right-censored 10000 0 9996 4

Coefficients: Estimate Std. error t value Pr(> t)
(Intercept) 7.898e-01 1.984e-02 39.807 < 2e-16 ***log(wage) 1.020e-03 1.304e-03 0.782 0.434*** ***log(tkm) -5.554e-03 1.377e-03 -4.033 5.51e-05*** log(eff) -6.770e-01 2.707e-03 -250.075 < 2e-16 ***log(xbar) -4.612e-03 8.622e-04 -5.349 8.82e-08*** log(gm) -6.029e-04 2.027e-03 -0.297 0.766
beta -2.722e-05 2.827e-05 -0.963 0.336
log(r) -1.805e-03 1.328e-03 -1.359 0.174
log(gx) -2.831e-03 5.335e-03 -0.531 0.596
log(gh) 1.969e-04 1.152e-03 0.171 0.864
log(sp) -1.555e-03 1.068e-03 -1.455 0.146
dB -1.147e-04 2.829e-04 -0.405 0.685
logSigma -2.849e+00 7.074e-03 -402.765 < 2e-16 *** — Signif. codes: 0 ‘***’ ***0.001*** ‘**’ *0.01* ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Newton-Raphson maximisation, 10 iterations Return code 1: gradient close to zero (gradtol) Log-likelihood: 14287.05 on 13 Df

```
regshare_voDEm <- margEff(regshare_voDE ,vcov = NULL,
   calcVCov = TRUE, returnJacobian = FALSE)
summary(regshare_voDEm)
```

```
       Marg. Eff.  Std. Error   t value  Pr(>|t|)
```

log(wage) 1.0195e-03 1.3040e-03 0.7818 0.4343
log(tkm) -5.5542e-03 1.3772e-03 -4.0330 5.547e-05 ***log(eff) -6.7695e-01 2.7070e-03 -250.0749 < 2.2e-16*** log(xbar) -4.6125e-03 8.6223e-04 -5.3495 9.016e-08 *** log(gm) -6.0295e-04 2.0268e-03 -0.2975 0.7661
beta -2.7220e-05 2.8273e-05 -0.9628 0.3357
log(r) -1.8045e-03 1.3278e-03 -1.3590 0.1742
log(gx) -2.8313e-03 5.3348e-03 -0.5307 0.5956
log(gh) 1.9694e-04 1.1515e-03 0.1710 0.8642

log(sp) -1.5546e-03 1.0685e-03 -1.4550 0.1457
dB -1.1466e-04 2.8288e-04 -0.4053 0.6852
— Signif. codes: 0 ‘’ ***0.001*** ’’ *0.01* ’’ 0.05 ‘.’ 0.1 ’ ’ 1

```
regalphaDE2 <- censReg(alpha~log(wage) + log(tkm) + log(eff) + log(xbar)
                + log(gm)
                + beta
                + log(r)  + log(gx) + log(gh)
                + log(sp)  + dB,  left=0, right=1, data = dfmcDE2.frame)
summary(regalphaDE2)
```

Call: censReg(formula = alpha ~ log(wage) + log(tkm) + log(eff) + log(xbar) + log(gm) + beta + log(r) + log(gx) + log(gh) + log(sp) + dB, left = 0, right = 1, data = dfmcDE2.frame)

Observations: Total Left-censored Uncensored Right-censored 10000 285 6642 3073

Coefficients: Estimate Std. error t value Pr(> t)
(Intercept) 0.1629857 0.0585210 2.785 0.00535 ** log(wage) 0.0514803 0.0041976 12.264 < 2e-16 ***log(tkm) 0.1335448 0.0039325 33.960 < 2e-16*** log(eff) 0.0731949 0.0079345 9.225 < 2e-16 ***log(xbar) 0.1300357 0.0025232 51.535 < 2e-16*** log(gm) 0.0383272 0.0059198 6.474 9.52e-11 ***beta 0.0171307 0.0001208 141.827 < 2e-16*** log(r) 0.0691874 0.0038993 17.744 < 2e-16 ***log(gx) -0.0299434 0.0156021 -1.919 0.05496 .***
log(gh) -0.0602393 0.0034156 -17.637 < 2e-16 log(sp) 0.0054654 0.0031256 1.749 0.08036 .
dB -0.0055952 0.0008258 -6.776 1.24e-11 ***logSigma -1.8841496 0.0089137 -211.376 < 2e-16*** — Signif. codes: 0 ‘’ ***0.001*** ’’ *0.01* ’’ 0.05 ‘.’ 0.1 ’ ’ 1

Newton-Raphson maximisation, 9 iterations Return code 2: successive function values within tolerance limit (tol) Log-likelihood: 1483.232 on 13 Df

```
regalphaDE2m <- margEff(regalphaDE2 ,vcov = NULL,
   calcVCov = TRUE, returnJacobian = FALSE)
summary(regalphaDE2m)
```

```
       Marg. Eff.  Std. Error  t value  Pr(>|t|)
```

log(wage) 0.04675261 0.00378938 12.3378 < 2.2e-16 ***log(tkm) 0.12128061 0.00355304 34.1343 < 2.2e-16*** log(eff) 0.06647300 0.00721614 9.2117 < 2.2e-16 ***log(xbar) 0.11809376 0.00226586 52.1188 < 2.2e-16*** log(gm) 0.03480736 0.00537450 6.4764 9.836e-11 ***beta 0.01555745 0.00010311 150.8880 < 2.2e-16*** log(r) 0.06283349 0.00353735 17.7629 < 2.2e-16 ***log(gx) -0.02719351 0.01416894 -1.9192 0.05498 .***
log(gh) -0.05470719 0.00310030 -17.6458 < 2.2e-16 log(sp) 0.00496352 0.00283846 1.7487 0.08038 .
dB -0.00508138 0.00074990 -6.7761 1.304e-11 *** — Signif. codes: 0 ‘’ ***0.001*** ’’ *0.01* ’’ 0.05 ‘.’ 0.1 ’ ’ 1

```
# print results for Germany into table
stargazer(regalphaDE2,regshare_vDE,regshare_vcDE,regshare_voDE,
          title="Regression Results (Germany)",#algin=TRUE,
          dep.var.labels=c("$alpha$","$v/H$","$v_c/H$","$v_o$"),
         no.space=TRUE)
```

% Table created by stargazer v.5.2.3 by Marek Hlavac, Social Policy Institute. E-mail: marek.hlavac at gmail.com % Date and time: Di, Mai 16, 2023 - 12:31:48

```
# results for US
regshare_vUS <- censReg(share_v~log(wage) + log(tkm) + log(eff) + log(xbar)
                + log(gm)
                + beta
                + log(r) + log(gx) + log(gh)
                + log(sp) + dB,  left=0, right=1, data = dfmcUS2.frame)
summary(regshare_vUS)
```

Table 3: Regression Results (Germany)

	Dependent variable:			
	alpha	v/H	v_c/H	v_o
	(1)	(2)	(3)	(4)
log(wage)	0.051^{***}	0.001	-0.001	0.001
	(0.004)	(0.002)	(0.001)	(0.001)
log(tkm)	0.134^{***}	0.109^{***}	0.142^{***}	-0.006^{***}
	(0.004)	(0.002)	(0.002)	(0.001)
log(eff)	0.073^{***}	-0.683^{***}	0.006^{**}	-0.677^{***}
	(0.008)	(0.004)	(0.003)	(0.003)
log(xbar)	0.130^{***}	0.102^{***}	0.131^{***}	-0.005^{***}
	(0.003)	(0.001)	(0.001)	(0.001)
log(gm)	0.038^{***}	-0.003	-0.003	-0.001
	(0.006)	(0.003)	(0.002)	(0.002)
beta	0.017^{***}	0.0001^{**}	0.0002^{***}	-0.00003
	(0.0001)	(0.00005)	(0.00003)	(0.00003)
log(r)	0.069^{***}	-0.0003	0.001	-0.002
	(0.004)	(0.002)	(0.001)	(0.001)
log(gx)	-0.030^{*}	-0.011	-0.001	-0.003
	(0.016)	(0.009)	(0.006)	(0.005)
log(gh)	-0.060^{***}	0.001	0.002	0.0002
	(0.003)	(0.002)	(0.001)	(0.001)
log(sp)	0.005^{*}	-0.001	0.0002	-0.002
	(0.003)	(0.002)	(0.001)	(0.001)
dB	-0.006^{***}	-0.0003	0.00004	-0.0001
	(0.001)	(0.0005)	(0.0003)	(0.0003)
logSigma	-1.884^{***}	-2.363^{***}	-2.750^{***}	-2.849^{***}
	(0.009)	(0.007)	(0.007)	(0.007)
Constant	0.163^{***}	0.946^{***}	0.177^{***}	0.790^{***}
	(0.059)	(0.032)	(0.022)	(0.020)
Observations	10,000	10,000	10,000	10,000
Log Likelihood	1,483.232	8,786.028	12,222.130	14,287.050
Akaike Inf. Crit.	$-2,940.464$	$-17,546.060$	$-24,418.250$	$-28,548.090$
Bayesian Inf. Crit.	$-2,846.730$	$-17,452.320$	$-24,324.520$	$-28,454.360$
Note:				*p<0.1; **p<0.05; ***p<0.01

Call: censReg(formula = share_v ~ log(wage) + log(tkm) + log(eff) + log(xbar) + log(gm) + beta + log(r) + log(gx) + log(gh) + log(sp) + dB, left = 0, right = 1, data = dfmcUS2.frame)

Observations: Total Left-censored Uncensored Right-censored 10000 0 9832 168

Coefficients: Estimate Std. error t value Pr(> t)
(Intercept) 9.395e-01 6.496e-02 14.462 < 2e-16 ***
log(wage) -1.137e-03 1.140e-03 -0.997 0.3187
log(tkm) 7.018e-03 9.391e-04 7.474 7.8e-14 log(eff) -6.912e-01 3.713e-03 -186.170 < 2e-16 ***
log(xbar) 1.300e-02 6.798e-04 19.130 < 2e-16 log(gm) -2.402e-03 2.454e-03 -0.979 0.3275
beta 2.731e-05 3.168e-05 0.862 0.3886
log(r) -1.345e-03 1.597e-03 -0.842 0.3997
log(gx) -3.399e-04 2.513e-03 -0.135 0.8924
log(gh) -2.879e-02 3.080e-02 -0.935 0.3499
log(sp) -1.996e-03 1.443e-03 -1.383 0.1667
dB -7.423e-04 3.840e-04 -1.933 0.0533 .
logSigma -2.555e+00 7.177e-03 -355.966 < 2e-16 *** — Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Newton-Raphson maximisation, 9 iterations Return code 8: successive function values within relative tolerance limit (reltol) Log-likelihood: 10887.47 on 13 Df

```
regshare_vUSm <- margEff(regshare_vUS ,vcov = NULL,
   calcVCov = TRUE, returnJacobian = FALSE)
summary(regshare_vUSm)
```

```
       Marg. Eff.  Std. Error   t value  Pr(>|t|)
```

log(wage) -1.1371e-03 1.1404e-03 -0.9971 0.31874
log(tkm) 7.0181e-03 9.3906e-04 7.4736 8.46e-14 ***
log(eff) -6.9124e-01 3.7130e-03 -186.1699 < 2.2e-16
log(xbar) 1.3004e-02 6.7976e-04 19.1299 < 2.2e-16 *** log(gm) -2.4024e-03 2.4537e-03 -0.9791 0.32757
beta 2.7310e-05 3.1676e-05 0.8622 0.38863
log(r) -1.3446e-03 1.5966e-03 -0.8422 0.39972
log(gx) -3.3989e-04 2.5132e-03 -0.1352 0.89242
log(gh) -2.8793e-02 3.0802e-02 -0.9348 0.34992
log(sp) -1.9958e-03 1.4434e-03 -1.3828 0.16677
dB -7.4231e-04 3.8405e-04 -1.9329 0.05328 .
— Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```
regshare_vcUS <- censReg(share_vc ~ log(wage) + log(tkm) + log(eff) + log(xbar)
                + log(gm)
                + beta
                + log(r) + log(gx) + log(gh)
                + log(sp) + dB,  left=0, right=1, data = dfmcUS2.frame)
 summary(regshare_vcUS)
```

Call: censReg(formula = share_vc ~ log(wage) + log(tkm) + log(eff) + log(xbar) + log(gm) + beta + log(r) + log(gx) + log(gh) + log(sp) + dB, left = 0, right = 1, data = dfmcUS2.frame)

Observations: Total Left-censored Uncensored Right-censored 10000 782 9218 0

Coefficients: Estimate Std. error t value Pr(> t)
(Intercept) 1.159e-01 4.074e-02 2.846 0.004428 ** log(wage) -1.763e-03 7.137e-04 -2.470 0.013523 *
log(tkm) 1.820e-02 7.261e-04 25.071 < 2e-16 ***
log(eff) 1.058e-02 2.317e-03 4.567 4.95e-06 log(xbar) 2.405e-02 5.608e-04 42.886 < 2e-16 ***
log(gm) 1.544e-03 1.537e-03 1.004 0.315284
beta 7.743e-05 2.000e-05 3.871 0.000109 log(r) 6.714e-04 9.993e-04 0.672 0.501658
log(gx) -2.763e-04 1.574e-03 -0.175 0.860715
log(gh) -2.683e-02 1.933e-02 -1.388 0.165062
log(sp) -4.643e-04 9.048e-04 -0.513 0.607840

dB -2.817e-04 2.405e-04 -1.171 0.241499
logSigma -3.039e+00 7.406e-03 -410.401 < 2e-16 *** — Signif. codes: 0 ‘’ ***0.001*** ’’ *0.01* ’’ 0.05 ‘.’ 0.1 ’ ’ 1

Newton-Raphson maximisation, 10 iterations Return code 8: successive function values within relative tolerance limit (reltol) Log-likelihood: 14419.99 on 13 Df

```
regshare_vcUSm <- margEff(regshare_vcUS ,vcov = NULL,
   calcVCov = TRUE, returnJacobian = FALSE )
summary(regshare_vcUSm )
```

```
       Marg. Eff.  Std. Error t value  Pr(>|t|)
```

log(wage) -1.4710e-03 5.9558e-04 -2.4698 0.0135350 *
log(tkm) 1.5191e-02 6.0710e-04 25.0220 < 2.2e-16 ***log(eff) 8.8308e-03 1.9338e-03 4.5665 5.018e-06***
log(xbar) 2.0072e-02 4.6989e-04 42.7152 < 2.2e-16 ***log(gm) 1.2884e-03 1.2830e-03 1.0042 0.3153069***
beta 6.4613e-05 1.6690e-05 3.8714 0.0001089 log(r) 5.6032e-04 8.3394e-04 0.6719 0.5016694
log(gx) -2.3053e-04 1.3138e-03 -0.1755 0.8607186
log(gh) -2.2389e-02 1.6128e-02 -1.3882 0.1650974
log(sp) -3.8747e-04 7.5507e-04 -0.5132 0.6078512
dB -2.3511e-04 2.0074e-04 -1.1712 0.2415324
— Signif. codes: 0 ‘’ ***0.001*** ’’ *0.01* ’’ 0.05 ‘.’ 0.1 ’ ’ 1

```
regshare_voUS <- censReg(share_vo~log(wage) + log(tkm) + log(eff) + log(xbar)
                + log(gm)
                + beta
                + log(r) + log(gx) + log(gh)
                + log(sp) + dB,  left=0, right=1, data = dfmcUS2.frame)
summary(regshare_voUS)
```

Call: censReg(formula = share_vo ~ log(wage) + log(tkm) + log(eff) + log(xbar) + log(gm) + beta + log(r) + log(gx) + log(gh) + log(sp) + dB, left = 0, right = 1, data = dfmcUS2.frame)

Observations: Total Left-censored Uncensored Right-censored 10000 0 9967 33

Coefficients: Estimate Std. error t value Pr(> t)
(Intercept) 8.283e-01 5.148e-02 16.090 <2e-16 ***log(wage) -3.263e-04 9.036e-04 -0.361 0.7180***
log(tkm) -1.014e-03 7.444e-04 -1.362 0.1731
log(eff) -6.963e-01 2.933e-03 -237.418 <2e-16 log(xbar) -6.626e-04 5.389e-04 -1.230 0.2188
log(gm) -3.274e-03 1.944e-03 -1.684 0.0922 .
beta 1.336e-05 2.511e-05 0.532 0.5947
log(r) -1.320e-03 1.265e-03 -1.044 0.2966
log(gx) -2.647e-04 1.992e-03 -0.133 0.8943
log(gh) -3.127e-03 2.441e-02 -0.128 0.8981
log(sp) -1.777e-03 1.144e-03 -1.554 0.1203
dB -4.121e-04 3.043e-04 -1.354 0.1756
logSigma -2.787e+00 7.093e-03 -392.941 <2e-16 *** — Signif. codes: 0 ‘’ ***0.001*** ’’ *0.01* ’’ 0.05 ‘.’ 0.1 ’ ’ 1

Newton-Raphson maximisation, 10 iterations Return code 1: gradient close to zero (gradtol) Log-likelihood: 13571.35 on 13 Df

```
regshare_voUSm <- margEff(regshare_voUS ,vcov = NULL,
   calcVCov = TRUE, returnJacobian = FALSE  )
summary(regshare_voUSm)
```

```
       Marg. Eff.  Std. Error   t value Pr(>|t|)
```

log(wage) -3.2629e-04 9.0356e-04 -0.3611 0.71802
log(tkm) -1.0140e-03 7.4438e-04 -1.3622 0.17315
log(eff) -6.9630e-01 2.9328e-03 -237.4175 < 2e-16 *** log(xbar) -6.6262e-04 5.3887e-04 -1.2297 0.21885

log(gm) -3.2736e-03 1.9443e-03 -1.6837 0.09227 .
beta 1.3356e-05 2.5106e-05 0.5320 0.59474
log(r) -1.3204e-03 1.2651e-03 -1.0437 0.29667
log(gx) -2.6466e-04 1.9916e-03 -0.1329 0.89429
log(gh) -3.1274e-03 2.4410e-02 -0.1281 0.89806
log(sp) -1.7772e-03 1.1439e-03 -1.5536 0.12031
dB -4.1214e-04 3.0433e-04 -1.3543 0.17568
— Signif. codes: 0 ‘’ ***0.001*** ’’ *0.01* ’’ 0.05 ‘.’ 0.1 ’ ’ 1

```
regalphaUS2  <- censReg(alpha~log(wage) + log(tkm) + log(eff) + log(xbar)
                 + log(gm)
                 + beta
                 + log(r) + log(gx) + log(gh)
                 + log(sp)  + dB,  left=0, right=1, data = dfmcUS2.frame)
summary(regalphaUS2)
```

Call: censReg(formula = alpha ~ log(wage) + log(tkm) + log(eff) + log(xbar) + log(gm) + beta + log(r) + log(gx) + log(gh) + log(sp) + dB, left = 0, right = 1, data = dfmcUS2.frame)

Observations: Total Left-censored Uncensored Right-censored 10000 565 6671 2764

Coefficients: Estimate Std. error t value Pr(> t)
(Intercept) -0.3622700 0.1425643 -2.541 0.011050 *
log(wage) 0.0347551 0.0026088 13.322 < 2e-16 ***log(tkm) -0.0182636 0.0019743 -9.251 < 2e-16*** log(eff) 0.0279217 0.0081799 3.413 0.000641 ***log(xbar) 0.0110621 0.0014176 7.803 6.03e-15*** log(gm) 0.0792612 0.0053658 14.772 < 2e-16 ***beta 0.0176531 0.0001159 152.327 < 2e-16*** log(r) 0.1721397 0.0035232 48.859 < 2e-16 ***log(gx) -0.0159057 0.0055038 -2.890 0.003853*** *log(gh) -0.1607890 0.0676897 -2.375 0.017531*
log(sp) 0.0132892 0.0031505 4.218 2.46e-05 ***dB -0.0071190 0.0008446 -8.429 < 2e-16*** logSigma -1.8844865 0.0088743 -212.353 < 2e-16 *** — Signif. codes: 0 ‘’ ***0.001*** ’’ *0.01* ’’ 0.05 ‘.’ 0.1 ’ ’ 1

Newton-Raphson maximisation, 9 iterations Return code 1: gradient close to zero (gradtol) Log-likelihood: 1586.704 on 13 Df

```
regalphaUS2m <- margEff(regalphaUS2 ,vcov = NULL,
   calcVCov = TRUE, returnJacobian = FALSE )
summary(regalphaUS2m )
```

```
       Marg. Eff.  Std. Error  t value  Pr(>|t|)
```

log(wage) 0.03281581 0.00244978 13.3954 < 2.2e-16 ***log(tkm) -0.01724455 0.00186411 -9.2508 < 2.2e-16*** log(eff) 0.02636373 0.00772632 3.4122 0.000647 ***log(xbar) 0.01044488 0.00133811 7.8057 6.661e-15*** log(gm) 0.07483851 0.00506325 14.7807 < 2.2e-16 ***beta 0.01666810 0.00010184 163.6644 < 2.2e-16*** log(r) 0.16253451 0.00331291 49.0610 < 2.2e-16 ***log(gx) -0.01501821 0.00519669 -2.8900 0.003861*** *log(gh) -0.15181712 0.06391329 -2.3754 0.017551*
log(sp) 0.01254771 0.00297426 4.2188 2.478e-05 ***dB -0.00672175 0.00079724 -8.4313 < 2.2e-16*** — Signif. codes: 0 ‘’ ***0.001*** ’’ *0.01* ’’ 0.05 ‘.’ 0.1 ’ ’ 1

```
stargazer(regalphaUS2,regshare_vUS,regshare_vcUS,regshare_voUS,
          title="Regression Results (U.S.)",#algin=TRUE,
          dep.var.labels=c("$alpha$","$v$","$v_c$","$v_o$"),
          no.space=TRUE)
```

% Table created by stargazer v.5.2.3 by Marek Hlavac, Social Policy Institute. E-mail: marek.hlavac at gmail.com % Date and time: Di, Mai 16, 2023 - 12:31:56

Table 4: Regression Results (U.S.)

	Dependent variable:			
	alpha	v	v_c	v_o
	(1)	(2)	(3)	(4)
log(wage)	0.035^{***}	-0.001	-0.002^{**}	-0.0003
	(0.003)	(0.001)	(0.001)	(0.001)
log(tkm)	-0.018^{***}	0.007^{***}	0.018^{***}	-0.001
	(0.002)	(0.001)	(0.001)	(0.001)
log(eff)	0.028^{***}	-0.691^{***}	0.011^{***}	-0.696^{***}
	(0.008)	(0.004)	(0.002)	(0.003)
log(xbar)	0.011^{***}	0.013^{***}	0.024^{***}	-0.001
	(0.001)	(0.001)	(0.001)	(0.001)
log(gm)	0.079^{***}	-0.002	0.002	-0.003^{*}
	(0.005)	(0.002)	(0.002)	(0.002)
beta	0.018^{***}	0.00003	0.0001^{***}	0.00001
	(0.0001)	(0.00003)	(0.00002)	(0.00003)
log(r)	0.172^{***}	-0.001	0.001	-0.001
	(0.004)	(0.002)	(0.001)	(0.001)
log(gx)	-0.016^{***}	-0.0003	-0.0003	-0.0003
	(0.006)	(0.003)	(0.002)	(0.002)
log(gh)	-0.161^{**}	-0.029	-0.027	-0.003
	(0.068)	(0.031)	(0.019)	(0.024)
log(sp)	0.013^{***}	-0.002	-0.0005	-0.002
	(0.003)	(0.001)	(0.001)	(0.001)
dB	-0.007^{***}	-0.001^{*}	-0.0003	-0.0004
	(0.001)	(0.0004)	(0.0002)	(0.0003)
logSigma	-1.884^{***}	-2.555^{***}	-3.039^{***}	-2.787^{***}
	(0.009)	(0.007)	(0.007)	(0.007)
Constant	-0.362^{**}	0.939^{***}	0.116^{***}	0.828^{***}
	(0.143)	(0.065)	(0.041)	(0.051)
Observations	10,000	10,000	10,000	10,000
Log Likelihood	1,586.704	10,887.470	14,420.000	13,571.350
Akaike Inf. Crit.	$-3{,}147.409$	$-21{,}748.950$	$-28{,}813.990$	$-27{,}116.700$
Bayesian Inf. Crit.	$-3{,}053.674$	$-21{,}655.210$	$-28{,}720.260$	$-27{,}022.970$
Note:				*p<0.1; **p<0.05; ***p<0.01

Instructions

Use results from ParamElast..De.txt and PramElast..US.txt (line 11 is the median) to get the elasticities for policy parameters

Use data from DataMC_Us.txt and DataMC_DE.txt for result table in the paper.

Use marginal effects from .pdf file for marginal effects tables in the paper.

Convert .RMD into .md file

knitr::knit("WFCTRAResults20230420.RMD")

Convert .md file into .tex file

Use tables from .tex file for paper text