CSSS 512: Lab 6

Panel Data Models with Few Time Periods

2018-5-18

Agenda

Examining the time series

```
# Clear memory
rm(list=ls())

# Load libraries
library(plm)             # Econometrics package for linear panel models
library(nlme)            # Estimation of mixed effects models
library(lme4)            # Alternative package for mixed effects models
library(tseries)         # For ADF unit root test
library(simcf)           # For panel functions and simulators
library(tile)            # For visualization of model inference
library(RColorBrewer)    # For nice colors
library(MASS)            # For mvrnorm()
source("helperCigs.R")   # For graphics functions

# Load cigarette consumption data (Jonathan Gruber, MIT)
# Variables (see codebook):
# state year    cpi pop packpc  income  tax avgprs  taxs
data <- read.csv("cigarette.csv")               #Load the dataset
data[1:5,]
```

```
##   state year   cpi     pop   packpc   income  tax   avgprs     taxs
## 1    AL 1985 1.076 3973000 116.4863 46014968 32.5 102.1817 33.34834
## 2    AL 1986 1.096 3992000 117.1593 48703940 32.5 107.9892 33.40584
## 3    AL 1987 1.136 4016000 115.8367 51846312 32.5 113.5273 33.46067
## 4    AL 1988 1.183 4024000 115.2584 55698852 32.5 120.0334 33.52509
## 5    AL 1989 1.240 4030000 109.2060 60044480 32.5 133.2560 33.65600
```

```
library(Ecdat)
help(Cigarette)
```

Examining the time series

```
# Quick inflation adjustment to 1995 dollars
inflAdjust <- function(x,cpi,year,target) {
    unique(cpi[year==target])*x/cpi
  #Multiply x with cpi in target year then divide by cpi in observed year
}
#Make adjustments to state personal income
data$income95 <- with(data, inflAdjust(income, cpi, year, 1995))
#Average state, federal, and average local excise taxes
data$tax95 <- with(data, inflAdjust(tax, cpi, year, 1995))
#Average price, including sales taxes
data$avgprs95 <- with(data, inflAdjust(avgprs, cpi, year, 1995))
#Average excise taxes, including sales taxes
data$taxs95 <- with(data, inflAdjust(taxs, cpi, year, 1995))
# Create per capita income (in k)
data$income95pc <- data$income95/data$pop
# Create pretax price, 1995 dollars
data$pretax95 <- data$avgprs95 - data$taxs95

data[1:5,]
```

```
##   state year   cpi     pop   packpc   income  tax   avgprs     taxs
## 1    AL 1985 1.076 3973000 116.4863 46014968 32.5 102.1817 33.34834
## 2    AL 1986 1.096 3992000 117.1593 48703940 32.5 107.9892 33.40584
## 3    AL 1987 1.136 4016000 115.8367 51846312 32.5 113.5273 33.46067
## 4    AL 1988 1.183 4024000 115.2584 55698852 32.5 120.0334 33.52509
## 5    AL 1989 1.240 4030000 109.2060 60044480 32.5 133.2560 33.65600
##   income95    tax95 avgprs95   taxs95 income95pc  pretax95
## 1 65173615 46.03160 144.7257 47.23314   16.40413  97.49257
## 2 67723361 45.19161 150.1601 46.45118   16.96477 103.70893
## 3 69554378 43.60035 152.3025 44.88914   17.31932 107.41336
## 4 71754049 41.86813 154.6331 43.18869   17.83152 111.44437
## 5 73796599 39.94355 163.7759 41.36431   18.31181 122.41162
```

```
attach(data)
```

Examining the time series

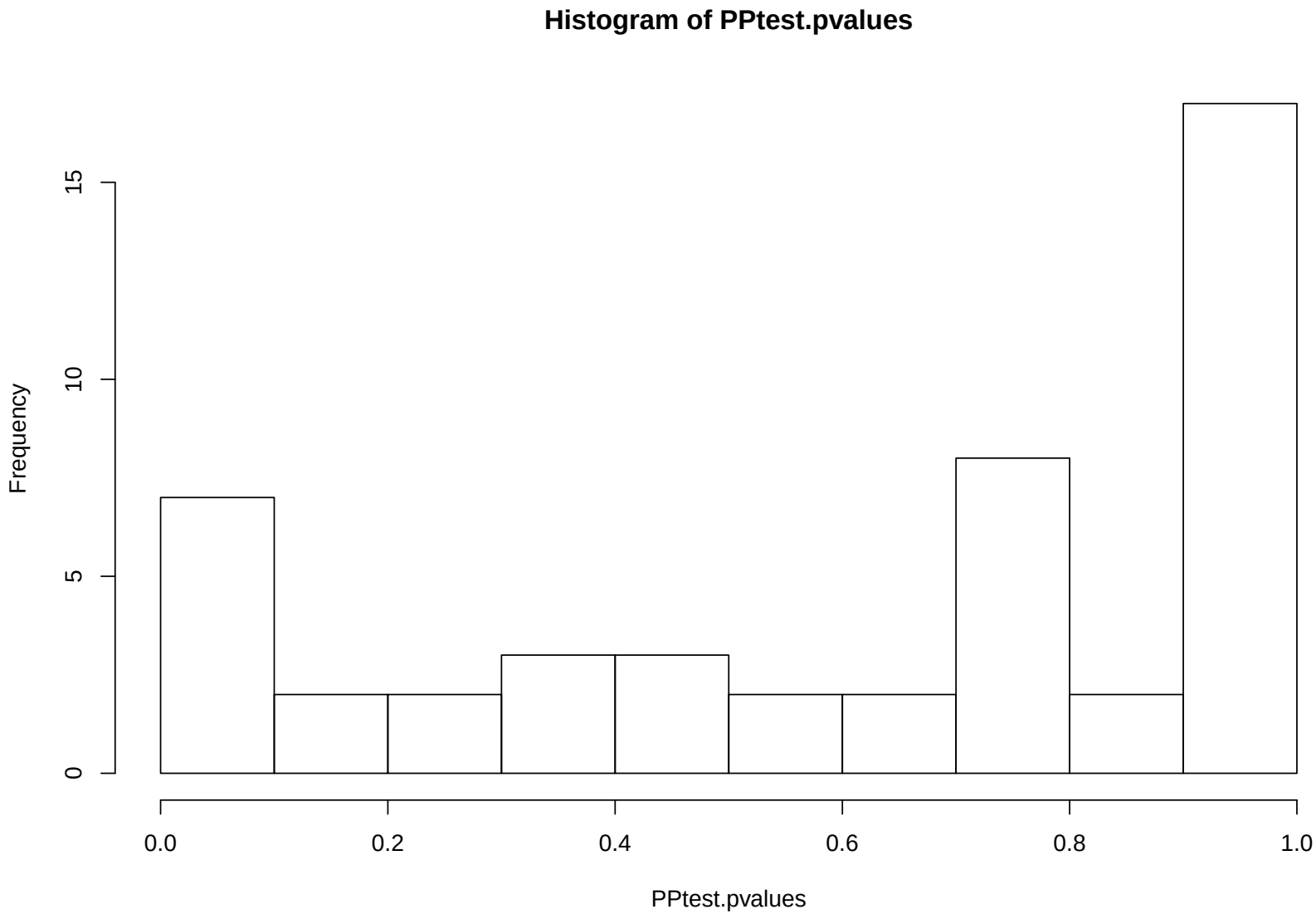
```
setwd("~/desktop/plots")
statelist <- unique(state)
# Look at the consumption time series for each state
for (i in 1:length(statelist)) {#Create a for loop from 1 to the number of states (48)
    currstate <- statelist[i]     #Make note of the state by number in the loop
    filename <- paste("tsPacksPCState",currstate,".pdf",sep="")
    #Create the file name of the plot
    pdf(filename,width=6,height=3.25)#Generate the PDF file
    plot(packpc[state==currstate],type="l",ylab="Packs Per Capita",
         #Generate the plot of packpc for the state by its number
    xlab="Year", main = paste("State",currstate) )
    dev.off()   #Turn off the PDF device
}
# Look at the ACF of consumption for each state
for (i in 1:length(statelist)) {#Create a for loop from 1 to the number of states (48)
    currstate <- statelist[i]   #Make note of the state by its number in the loop
    filename <- paste("acfPacksPCState",currstate,".pdf",sep="")
    #Create the file name of the plot
    pdf(filename,width=6,height=3.25)#Generate the PDF file
    acf(packpc[state==currstate])#Generate the ACF plot of packpc for the state by its number
    dev.off()#Turn off the PDF device
}
# Look at the PACF of consumption for each state
for (i in 1:length(statelist)) {
    currstate <- statelist[i]
    filename <- paste("acfPacksPCState",currstate,".pdf",sep="")
    pdf(filename,width=6,height=3.25)
    pacf(packpc[state==currstate])
    #Generate the PACF plot of packpc for the state by its number
    dev.off()
}
```

Examining the time series

```
# Check for a unit root in each country
PPtest.pvalues <- rep(0,length(statelist))
#Create empty vectors for PP test p-values
adftest.pvalues <- rep(0,length(statelist))
#Create empty vectors for adf test p-values

for (i in 1:length(statelist)) {#Create a for loop from 1 to the number of states
    currstate <- statelist[i]#Make note of the state by its number in the loop

    # Check PP unit root test, omitting errors due to short series
    curPP <- try(PP.test(packpc[state==currstate])$p.value)
    #Find the p-value of the PP test for the state
    if (any(class(curPP)=="try-error")) curPP <- NA
    #Make note if there is an error in the PP test, if so, fill with an NA
    PPtest.pvalues[i] <- curPP
    #Store the p-value of the PP test in the PP test vector

    curadf <- try(adf.test(packpc[state==currstate])$p.value)
    #Do the same with the adf test results
    if (any(class(curadf)=="try-error")) curadf <- NA
    adftest.pvalues[i] <- curadf
  }
```

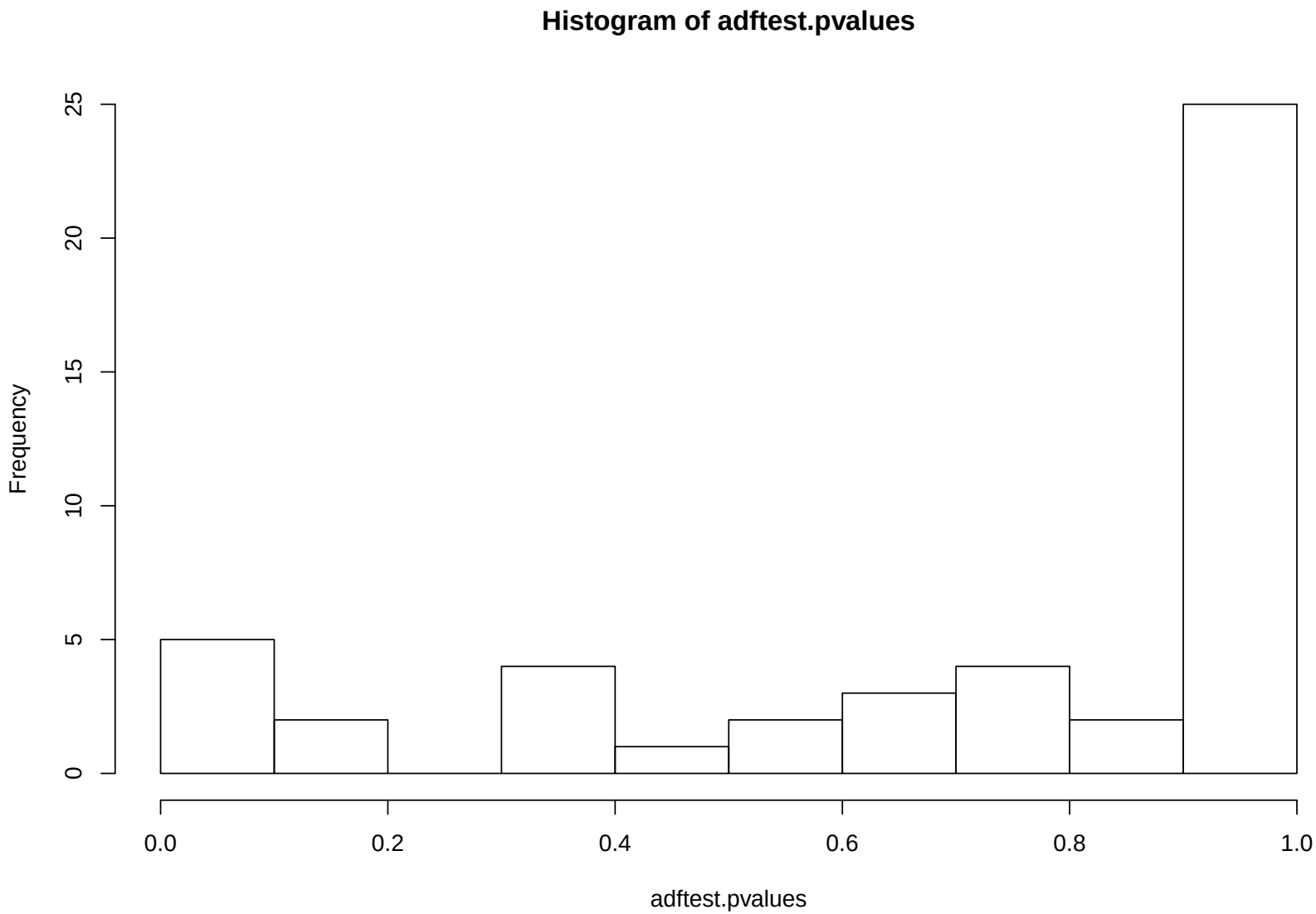
Examining the time series

```
hist(PPtest.pvalues)          # Plot a histogram of the p-values
```



Examining the time series

```
hist(adftest.pvalues)          # Plot a histogram of the p-values
```



Examining the time series

```r
# Alternative model specifications
model1 <- packpc ~ income95pc + avgprs95
model2 <- packpc ~ income95pc + pretax95 + taxs95
model3 <- log(packpc) ~ log(income95pc) + log(avgprs95)

# Simple linear models
lm.res1 <- lm(model1, data)
lm.res2 <- lm(model2, data)
lm.res3 <- lm(model3, data)

summary(lm.res1)
```

```
##
## Call:
## lm(formula = model1, data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -50.675 -10.238  -0.840   8.998  63.772
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 199.52434    6.56981  30.370  < 2e-16 ***
## income95pc    1.09830    0.26496   4.145 3.96e-05 ***
## avgprs95     -0.66467    0.03656 -18.182  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 18 on 525 degrees of freedom
## Multiple R-squared:  0.3966, Adjusted R-squared:  0.3943
## F-statistic: 172.5 on 2 and 525 DF,  p-value: < 2.2e-16
```

Examining the time series

```
summary(lm.res2)
```

```
##
## Call:
## lm(formula = model2, data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -49.882  -9.468  -0.588   8.744  66.532
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 191.71631    7.13804  26.858  < 2e-16 ***
## income95pc    1.15300    0.26415   4.365 1.53e-05 ***
## pretax95     -0.54863    0.05616  -9.768  < 2e-16 ***
## taxs95       -0.80264    0.06256 -12.831  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 17.89 on 524 degrees of freedom
## Multiple R-squared:  0.4049, Adjusted R-squared:  0.4015
## F-statistic: 118.9 on 3 and 524 DF,  p-value: < 2.2e-16
```

Examining the time series

```
summary(lm.res3)
```

```
##
## Call:
## lm(formula = model3, data = data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.67369 -0.09012  0.00698  0.09820  0.41951
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)      9.68686    0.28810  33.623  < 2e-16 ***
## log(income95pc)  0.24371    0.05367   4.541 6.96e-06 ***
## log(avgprs95)   -1.12181    0.06037 -18.582  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1696 on 525 degrees of freedom
## Multiple R-squared:  0.4036, Adjusted R-squared:  0.4013
## F-statistic: 177.6 on 2 and 525 DF,  p-value: < 2.2e-16
```

Fixed effects model

```
# Check for time invariant variables:
pvar(data)
```

```
## no time variation:       state
## no individual variation: year cpi
```

```
# "within" option tells plm to do fixed effects
# Note that if you want to add year fixed effects then set effect="time" and for both state
# and year fixed effects set effect effect="twoway"
plm.res1 <- plm(packpc ~ income95pc + pretax95 + taxs95,  data = data, model="within", effect="twoway")
```

Fixed effects model

```r
summary(plm.res1)
```

```
## Twoways effects Within Model
##
## Call:
## plm(formula = packpc ~ income95pc + pretax95 + taxs95, data = data,
##     effect = "twoway", model = "within")
##
## Balanced Panel: n=48, T=11, N=528
##
## Residuals :
##     Min.  1st Qu.   Median  3rd Qu.     Max.
## -16.5000  -1.9400   0.0468   2.1800  18.1000
##
## Coefficients :
##             Estimate Std. Error  t-value  Pr(>|t|)
## income95pc  0.969966   0.410602   2.3623 0.0185709 *
## pretax95   -0.188551   0.051420  -3.6669 0.0002738 ***
## taxs95     -0.481852   0.033595 -14.3429 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Total Sum of Squares:    13258
## Residual Sum of Squares: 8257.1
## R-Squared:      0.3772
## Adj. R-Squared: 0.29718
## F-statistic: 94.2779 on 3 and 467 DF, p-value: < 2.22e-16
```

Fixed effects model

```
# Some tests for serial correlation of errors (needed because we have a linear regression
# with lags of the dependent variable on the RHS
# the standard LM test (note we could specify order)
pbgtest(plm.res1)
```

```
##
##  Breusch-Godfrey/Wooldridge test for serial correlation in panel
##  models
##
## data:  packpc ~ income95pc + pretax95 + taxs95
## chisq = 129.6, df = 11, p-value < 2.2e-16
## alternative hypothesis: serial correlation in idiosyncratic errors
```

Fixed effects model

```
## Robust var-cov matrix alternatives for fixed effects models...
robust <- "None"   # Choose var-cov estimator here
if (robust=="None") vc <- vcov(plm.res1)
if (robust=="Arellano") vc <- vcovHC(plm.res1)
# Arellano (1987) heteroskedastic and serial correlation robust VC
if (robust=="BeckKatz") vc <- vcovBK(plm.res1)# Beck and Katz (1995) panel corrected VC
if (robust=="DriscollKraay") vc <- vcovSCC(plm.res1)# Driscoll and Kraay panel corrected VC

# Extract model results
pe.res1 <- coef(plm.res1) # Point estimates of parameters
vc.res1 <- vc   # Var-cov matrix of point estimates
se.res1 <- sqrt(diag(vc.res1))# std erros of point estimates
tstat.res1 <- abs(pe.res1/se.res1)# t-statistics
df.res1 <- rep(plm.res1$df.residual, length(tstat.res1))  # residual degrees of freedom
pval.res1 <- 2*pt(tstat.res1, df.res1, lower.tail=FALSE)  # p-values
fe.res1 <- fixef(plm.res1)# the (removed) fixed effects by group
resid.res1 <- resid(plm.res1)# Residuals
```

Random effects model

```
# Estimate a random effects AR(I)MA(p,q) model using lme (Restricted ML)
lme.res1 <- lme(# A formula object including the response,
                # the fixed covariates, and any grouping variables
                fixed = packpc ~ income95pc + pretax95 + taxs95,
                # i.e. response variable and explanatory variables

                # The random effects component
                random = ~ 1 | state,
                # 1 indicates the intercept and state indicates the grouping

                # The TS dynamics: specify the time & group variables,
                # and the order of the ARMA(p,q) process
                correlation = corARMA(form = ~ year | state,
                                      p = 1,  # AR(p) order
                                      q = 0   # MA(q) order
                                      )
                )
```

Random effects model

```
# Extract model results
pe.res1 <- fixed.effects(lme.res1)          # Point estimates of fixed effects
vc.res1 <- vcov(lme.res1)                   # Var-cov matrix of fixed effects estimates
se.res1 <- sqrt(diag(vc.res1))              # std erros of fixed effects estimates
re.res1 <- random.effects(lme.res1)         # "Estimated" random effects by group
ll.res1 <- logLik(lme.res1)                 # Log-likelihood at maximum
resid.res1 <- resid(lme.res1)               # Residuals
aic.res1 <- AIC(lme.res1)                   # Akaike Information Criterion
```

Random effects model

```
summary(lme.res1)
```

```
## Linear mixed-effects model fit by REML
##  Data: NULL
##       AIC     BIC    logLik
##   3253.21 3283.04 -1619.605
##
## Random effects:
##  Formula: ~1 | state
##         (Intercept) Residual
## StdDev:  0.01127294 20.92621
##
## Correlation Structure: AR(1)
##  Formula: ~year | state
##  Parameter estimate(s):
##       Phi
## 0.9764735
## Fixed effects: packpc ~ income95pc + pretax95 + taxs95
##                 Value Std.Error  DF    t-value p-value
## (Intercept) 173.08136  8.574965 477  20.184499  0.0000
## income95pc   -1.05746  0.387602 477  -2.728198  0.0066
## pretax95     -0.14537  0.024800 477  -5.861684  0.0000
## taxs95       -0.46630  0.040769 477 -11.437827  0.0000
##  Correlation:
##             (Intr) incm95 prtx95
## income95pc -0.856
## pretax95   -0.099 -0.223
## taxs95     -0.160 -0.097 -0.035
##
## Standardized Within-Group Residuals:
##         Min          Q1         Med          Q3         Max
## -2.79963472 -0.57137010 -0.08122771  0.45749547  3.97887775
##
## Number of Observations: 528
```

Dynamic panel data models

```
# Panel based diagnostics available in the plm library
# (This package recently expanded to contain many many panel data tests
#  for serial correlation, fixed effects, and unit roots)

# First, create a plm data frame (special data frame that "knows" the
# unit variable and time variable
pdata <- pdata.frame(data, index=c("state", "year"))
pdata[1:3,]
```

```
##         state year   cpi     pop   packpc   income  tax   avgprs     taxs
## AL-1985    AL 1985 1.076 3973000 116.4863 46014968 32.5 102.1817 33.34834
## AL-1986    AL 1986 1.096 3992000 117.1593 48703940 32.5 107.9892 33.40584
## AL-1987    AL 1987 1.136 4016000 115.8367 51846312 32.5 113.5273 33.46067
##         income95    tax95 avgprs95   taxs95 income95pc  pretax95
## AL-1985 65173615 46.03160 144.7257 47.23314   16.40413  97.49257
## AL-1986 67723361 45.19161 150.1601 46.45118   16.96477 103.70893
## AL-1987 69554378 43.60035 152.3025 44.88914   17.31932 107.41336
```

```
# Do an panel unit root test on the undifferenced cigarette data;
# there are many options; see ?purtest

# Note:  for some reason this isn't working
#purtest(packpc~1, data=pdata, test="ips")
```

Dynamic panel data models

```r
# Estimate Arellano-Bond GMM for fixed effects with lagged DV
#
# pgmm needs formulas in a specific format:
# 1. in the first part of the RHS, include lags of DV and covariates, as shown
# 2. in the second part, include the panel data instruments (99 here means use
#    up to the 99th lag of the difference as an instrument)
# 3. in an optional (not shown) third part of the RHS, include any other instruments
#
# note that pgmm formulas construct lag() properly for panel data,
# though lag() usually doesn't
pgmmformula.1a <- packpc ~ lag(packpc, 1) + income95pc + avgprs95 | lag(packpc, 2:99)

# We'll run GMM with only unit fixed effects,
# but we could include period fixed effects as well by setting effect to "two-way"
# (often a good practice in short T panels)
pgmm.res1a <- pgmm(pgmmformula.1a,
                   data = pdata,
                   effect = "individual",
                   # should consider two-way for small T
                   transformation = "d")
# should do ld if T=3, d for difference GNN and ld for system GMM
```

Dynamic panel data models

```
summary(pgmm.res1a)
```

```
## Oneway (individual) effect One step model
##
## Call:
## pgmm(formula = pgmmformula.1a, data = pdata, effect = "individual",
##     transformation = "d")
##
## Balanced Panel: n=48, T=11, N=528
##
## Number of Observations Used:  432
##
## Residuals
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -25.4300  -2.5080   0.1463   0.1238   2.7380  25.6000
##
## Coefficients
##                  Estimate Std. Error z-value  Pr(>|z|)
## lag(packpc, 1)  0.638987   0.055342 11.5462 < 2.2e-16 ***
## income95pc     -0.475568   0.486760 -0.9770    0.3286
## avgprs95       -0.180791   0.027799 -6.5035 7.848e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Sargan Test: chisq(44) = 47.99763 (p.value=0.31401)
## Autocorrelation test (1): normal = -3.948861 (p.value=7.8524e-05)
## Autocorrelation test (2): normal = -0.5688819 (p.value=0.56944)
## Wald test for coefficients: chisq(3) = 2496.07 (p.value=< 2.22e-16)
```

Dynamic panel data models

```
# Good Sargan test, Good AR(2) test
# (Sargan test has a null of the instruments as a group being exogenous)
# (The residuals of the differenced equations should exhibit AR(1) but not AR(2) behavior)


# Let's consider alternative sets of instruments; concern: distant lags are weak instruments
pgmmformula.1b <- packpc ~ lag(packpc, 1) + income95pc + avgprs95 | lag(packpc, 2:5)
pgmm.res1b <- pgmm(pgmmformula.1b,
                   data = pdata,
                   effect = "individual",    # should consider two-way for small T
                   transformation = "d")     # should do ld if T=3
```

Dynamic panel data models

```r
# Poor Sargan test, Good AR(2) test
summary(pgmm.res1b)
```

```
## Oneway (individual) effect One step model
##
## Call:
## pgmm(formula = pgmmformula.1b, data = pdata, effect = "individual",
##     transformation = "d")
##
## Balanced Panel: n=48, T=11, N=528
##
## Number of Observations Used:  432
##
## Residuals
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -25.6900  -2.5360   0.1483   0.1217   2.6740  25.5800
##
## Coefficients
##                  Estimate Std. Error z-value  Pr(>|z|)
## lag(packpc, 1)  0.650350   0.055545 11.7086 < 2.2e-16 ***
## income95pc     -0.325994   0.497744 -0.6549    0.5125
## avgprs95       -0.181297   0.027242 -6.6550 2.834e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Sargan Test: chisq(29) = 42.46279 (p.value=0.050998)
## Autocorrelation test (1): normal = -3.907354 (p.value=9.3312e-05)
## Autocorrelation test (2): normal = -0.5460146 (p.value=0.58506)
## Wald test for coefficients: chisq(3) = 2503.149 (p.value=< 2.22e-16)
```

Dynamic panel data models

```
# Keeping just the most recent two instruments makes no substantive difference
pgmmformula.1c <- packpc ~ lag(packpc, 1) + income95pc + avgprs95 | lag(packpc, 2:3)
pgmm.res1c <- pgmm(pgmmformula.1c,
                   data = pdata,
                   effect = "individual",    # should consider two-way for small T
                   transformation = "d")     # should do ld if T=3
```

Dynamic panel data models

```r
# Poor Sargan test, Good AR(2) test
summary(pgmm.res1c)
```

```
## Oneway (individual) effect One step model
##
## Call:
## pgmm(formula = pgmmformula.1c, data = pdata, effect = "individual",
##     transformation = "d")
##
## Balanced Panel: n=48, T=11, N=528
##
## Number of Observations Used:  432
##
## Residuals
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -25.9200  -2.4790   0.1232   0.1159   2.7110  25.6000
##
## Coefficients
##                  Estimate Std. Error z-value  Pr(>|z|)
## lag(packpc, 1)  0.660475   0.052854 12.4963 < 2.2e-16 ***
## income95pc     -0.258571   0.456052 -0.5670    0.5707
## avgprs95       -0.175368   0.026891 -6.5215  6.96e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Sargan Test: chisq(16) = 40.04482 (p.value=0.00076694)
## Autocorrelation test (1): normal = -3.831746 (p.value=0.00012724)
## Autocorrelation test (2): normal = -0.4941905 (p.value=0.62117)
## Wald test for coefficients: chisq(3) = 2405.391 (p.value=< 2.22e-16)
```

Dynamic panel data models

```
# Slight difference with one instrument, but not substatively noteworthy?
pgmmformula.1d <- packpc ~ lag(packpc, 1) + income95pc + avgprs95 | lag(packpc, 2)
pgmm.res1d <- pgmm(pgmmformula.1d,
                   data = pdata,
                   effect = "individual",    # should consider two-way for small T
                   transformation = "d")     # should do ld if T=3
```

Dynamic panel data models

```r
# Poor Sargan test, Good AR(2) test
summary(pgmm.res1d)
```

```
## Oneway (individual) effect One step model
##
## Call:
## pgmm(formula = pgmmformula.1d, data = pdata, effect = "individual",
##     transformation = "d")
##
## Balanced Panel: n=48, T=11, N=528
##
## Number of Observations Used:  432
##
## Residuals
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -26.8300  -2.6300   0.2236   0.1076   2.6800  25.6300
##
## Coefficients
##                  Estimate Std. Error z-value  Pr(>|z|)
## lag(packpc, 1)  0.700990   0.051462 13.6216 < 2.2e-16 ***
## income95pc      0.112174   0.447504  0.2507    0.8021
## avgprs95       -0.164193   0.027435 -5.9848 2.167e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Sargan Test: chisq(8) = 27.75947 (p.value=0.00052221)
## Autocorrelation test (1): normal = -3.658942 (p.value=0.00025326)
## Autocorrelation test (2): normal = -0.3566186 (p.value=0.72138)
## Wald test for coefficients: chisq(3) = 2723.818 (p.value=< 2.22e-16)
```

Dynamic panel data models

```
# Try system GMM with all lags
pgmm.res1e <- pgmm(pgmmformula.1a,
                   data = pdata,
                   effect = "individual",    # should consider two-way for small T
                   transformation = "ld")     # should do ld if T=3
```

Dynamic panel data models

```r
# Good Sargan test, Good AR(2) test
summary(pgmm.res1e)
```

```
## Oneway (individual) effect One step model
##
## Call:
## pgmm(formula = pgmmformula.1a, data = pdata, effect = "individual",
##     transformation = "ld")
##
## Balanced Panel: n=48, T=11, N=528
##
## Number of Observations Used:  912
##
## Residuals
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -35.0200  -2.5250   0.1318   0.2344   2.8280  29.0100
##
## Coefficients
##                   Estimate Std. Error z-value Pr(>|z|)
## lag(packpc, 1)  0.9372190  0.0149757 62.5826   <2e-16 ***
## income95pc      0.2033459  0.1245084  1.6332   0.1024
## avgprs95       -0.0021312  0.0098748 -0.2158   0.8291
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Sargan Test: chisq(55) = 47.79775 (p.value=0.74372)
## Autocorrelation test (1): normal = -3.451448 (p.value=0.00055759)
## Autocorrelation test (2): normal = 0.5944316 (p.value=0.55222)
## Wald test for coefficients: chisq(3) = 109661.9 (p.value=< 2.22e-16)
```

Dynamic panel data models

```
# Try system GMM with only recent lag
pgmm.res1f <- pgmm(pgmmformula.1d,
                   data = pdata,
                   effect = "individual",    # should consider two-way for small T
                   transformation = "ld")     # should do ld if T=3
```

Dynamic panel data models

```r
# Poor Sargan test, Good AR(2) test
summary(pgmm.res1f)
```

```
## Oneway (individual) effect One step model
##
## Call:
## pgmm(formula = pgmmformula.1d, data = pdata, effect = "individual",
##     transformation = "ld")
##
## Balanced Panel: n=48, T=11, N=528
##
## Number of Observations Used:  912
##
## Residuals
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -35.0100  -2.7770   0.1193   0.2195   2.8830  28.6200
##
## Coefficients
##                   Estimate  Std. Error z-value Pr(>|z|)
## lag(packpc, 1)  0.91605377  0.01475300 62.0927  < 2e-16 ***
## income95pc      0.29563062  0.14667917  2.0155  0.04385 *
## avgprs95       -0.00075664  0.01219834 -0.0620  0.95054
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Sargan Test: chisq(19) = 41.05134 (p.value=0.0023757)
## Autocorrelation test (1): normal = -3.490677 (p.value=0.0004818)
## Autocorrelation test (2): normal = 0.6062896 (p.value=0.54432)
## Wald test for coefficients: chisq(3) = 138138.9 (p.value=< 2.22e-16)
```

Dynamic panel data models

```
# Try difference GMM with two way effects
pgmm.res1g <- pgmm(pgmmformula.1a,
                   data = pdata,
                   effect = "twoways",   # should consider two-way for small T
                   transformation = "d")    # should do ld if T=3
```

Dynamic panel data models

```r
# Good Sargan test, Good AR(2) test, Wald supports 2-way
summary(pgmm.res1g)
```

```
## Twoways effects One step model
##
## Call:
## pgmm(formula = pgmmformula.1a, data = pdata, effect = "twoways",
##     transformation = "d")
##
## Balanced Panel: n=48, T=11, N=528
##
## Number of Observations Used:  432
##
## Residuals
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -18.940  -1.890  -0.259   0.000   1.824  20.430
##
## Coefficients
##                 Estimate Std. Error z-value  Pr(>|z|)
## lag(packpc, 1)  0.252415   0.117744  2.1438   0.03205 *
## income95pc      1.062384   0.674055  1.5761   0.11500
## avgprs95       -0.285703   0.060572 -4.7168 2.396e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Sargan Test: chisq(44) = 45.92782 (p.value=0.39225)
## Autocorrelation test (1): normal = -3.571843 (p.value=0.00035448)
## Autocorrelation test (2): normal = 0.02846648 (p.value=0.97729)
## Wald test for coefficients: chisq(3) = 35.6544 (p.value=8.8603e-08)
## Wald test for time dummies: chisq(9) = 70.99219 (p.value=9.7258e-12)
```

Dynamic panel data models

```
# Try system GMM with two way effects
pgmm.res1h <- pgmm(pgmmformula.1a,
                   data = pdata,
                   effect = "twoways",   # should consider two-way for small T
                   transformation = "ld")    # should do ld if T=3
```

Dynamic panel data models

```
# Good Sargan test, Good AR(2) test, Wald supports 2-way
summary(pgmm.res1h)
```

```
## Twoways effects One step model
##
## Call:
## pgmm(formula = pgmmformula.1a, data = pdata, effect = "twoways",
##     transformation = "ld")
##
## Balanced Panel: n=48, T=11, N=528
##
## Number of Observations Used:  912
##
## Residuals
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## -31.82000  -2.37500  -0.03745   0.00000   2.18300  27.94000
##
## Coefficients
##                  Estimate Std. Error z-value  Pr(>|z|)
## lag(packpc, 1)  0.912506   0.035086 26.0074 < 2.2e-16 ***
## income95pc     -0.016144   0.110832 -0.1457  0.884186
## avgprs95       -0.101336   0.032190 -3.1481  0.001644 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Sargan Test: chisq(55) = 45.80834 (p.value=0.80677)
## Autocorrelation test (1): normal = -3.133862 (p.value=0.0017252)
## Autocorrelation test (2): normal = 0.7322591 (p.value=0.46401)
## Wald test for coefficients: chisq(3) = 2942.747 (p.value=< 2.22e-16)
## Wald test for time dummies: chisq(9) = 87.03407 (p.value=6.3969e-15)
```

Dynamic panel data models

```
# Aside: Note that the year fixed effects estimates show a downward trend in smoking,
# but with large CIs

yrs <- coef(pgmm.res1g)[4:12]                    #Extract the year fixed effects from model 1g
yrs.se <- sqrt(diag(vcovHC(pgmm.res1g)))[4:12]   #Extract the standard errors of the year fe
yrsTrace <- scatter(x=1987:1995,                 #X values
                    y=yrs,                       #Y values
                    ylower=yrs-2*yrs.se,         #Upper bound of CI
                    yupper=yrs+2*yrs.se,         #Lower bound of CI
                    fit=list(method="wls", weights=1/yrs.se^2),
                    pch=1, size=.8,
                    plot=1
                    )

tile(yrsTrace,
     width = list(null=5),       # widen plot area for visibility
     output = list(file="yearEffectsModel1g", width=5.5),
     limits = c(1986.5,1995.5,-22,8),
     yaxis=list(major=FALSE),
     xaxistitle = list(labels="Year"),
     yaxistitle = list(labels="Estimated year effects (95% CI)"),
     height=list(plot="golden")
     )
```

Dynamic panel data models

```
####
# Now consider last two models with alternative specifications
pgmmformula.2a <- packpc ~ lag(packpc, 1) + income95pc + pretax95 +
  taxs95 | lag(packpc, 2:99)

pgmmformula.3a <- log(packpc) ~ lag(log(packpc), 1) + log(income95pc) +
  log(avgprs95) | lag(log(packpc), 2:99)

pgmmformula.4a <- log(packpc) ~ lag(log(packpc), 1) + log(income95pc) +
  log(pretax95) + log(taxs95) | lag(log(packpc), 2:99)
```

Model 2: Unique tax effects

```
# Try difference GMM with two way effects
pgmm.res2g <- pgmm(pgmmformula.2a,
                   data = pdata,
                   effect = "twoways",    # should consider two-way for small T
                   transformation = "d")     # should do ld if T=3


# Try system GMM with two way effects
pgmm.res2h <- pgmm(pgmmformula.2a,
                   data = pdata,
                   effect = "twoways",    # should consider two-way for small T
                   transformation = "ld")     # should do ld if T=3
```

Model 2: Unique tax effects

```
# Good Sargan test, Good AR(2) test, Wald supports 2-way
summary(pgmm.res2g)
```

```
## Twoways effects One step model
##
## Call:
## pgmm(formula = pgmmformula.2a, data = pdata, effect = "twoways",
##     transformation = "d")
##
## Balanced Panel: n=48, T=11, N=528
##
## Number of Observations Used:  432
##
## Residuals
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -18.2600  -1.9650  -0.1188   0.0000   1.7690  20.1100
##
## Coefficients
##                  Estimate Std. Error z-value  Pr(>|z|)
## lag(packpc, 1)  0.267324   0.119980  2.2281   0.02588 *
## income95pc       0.780413   0.692055  1.1277   0.25946
## pretax95        -0.027143   0.065641 -0.4135   0.67923
## taxs95          -0.407912   0.062596 -6.5166 7.192e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Sargan Test: chisq(44) = 46.51024 (p.value=0.36939)
## Autocorrelation test (1): normal = -3.467009 (p.value=0.00052628)
## Autocorrelation test (2): normal = 0.444432 (p.value=0.65673)
## Wald test for coefficients: chisq(4) = 63.13155 (p.value=6.3668e-13)
## Wald test for time dummies: chisq(9) = 100.6487 (p.value=< 2.22e-16)
```

Model 2: Unique tax effects

```
# Good Sargan test, Good AR(2) test, Wald supports 2-way
summary(pgmm.res2h)
```

```
## Twoways effects One step model
##
## Call:
## pgmm(formula = pgmmformula.2a, data = pdata, effect = "twoways",
##     transformation = "ld")
##
## Balanced Panel: n=48, T=11, N=528
##
## Number of Observations Used:  912
##
## Residuals
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## -31.81000  -2.33200  -0.06877   0.00000   2.15400  28.00000
##
## Coefficients
##                  Estimate Std. Error z-value  Pr(>|z|)
## lag(packpc, 1)  0.920875   0.032453 28.3758 < 2.2e-16 ***
## income95pc     -0.032675   0.104547 -0.3125  0.754629
## pretax95       -0.073863   0.034405 -2.1469  0.031803 *
## taxs95         -0.104354   0.036213 -2.8817  0.003956 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Sargan Test: chisq(56) = 48 (p.value=0.76774)
## Autocorrelation test (1): normal = -3.108691 (p.value=0.0018792)
## Autocorrelation test (2): normal = 0.7719112 (p.value=0.44017)
## Wald test for coefficients: chisq(4) = 3309.815 (p.value=< 2.22e-16)
## Wald test for time dummies: chisq(9) = 81.95308 (p.value=6.6084e-14)
```

Model 3: Elasticity specification

```
# Try difference GMM with only unit fixed effects
pgmm.res3a <- pgmm(pgmmformula.3a,
                   data = pdata,
                   effect = "individual",    # should consider two-way for small T
                   transformation = "d")     # should do ld if T=3


# Try system GMM with all lags
pgmm.res3e <- pgmm(pgmmformula.3a,
                  data = pdata,
                  effect = "individual",    # should consider two-way for small T
                  transformation = "ld")     # should do ld if T=3


# Try difference GMM with two way effects
pgmm.res3g <- pgmm(pgmmformula.3a,
                   data = pdata,
                   effect = "twoways",    # should consider two-way for small T
                   transformation = "d")     # should do ld if T=3


# Try system GMM with two way effects
pgmm.res3h <- pgmm(pgmmformula.3a,
                   data = pdata,
                   effect = "twoways",    # should consider two-way for small T
                   transformation = "ld")     # should do ld if T=3
```

Model 3: Elasticity specification

```r
# Good Sargan test, Good AR(2) test
summary(pgmm.res3a)
```

```
## Oneway (individual) effect One step model
##
## Call:
## pgmm(formula = pgmmformula.3a, data = pdata, effect = "individual",
##     transformation = "d")
##
## Balanced Panel: n=48, T=11, N=528
##
## Number of Observations Used:  432
##
## Residuals
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## -0.263000 -0.024090  0.002736  0.001092  0.027680  0.216400
##
## Coefficients
##                       Estimate Std. Error z-value  Pr(>|z|)
## lag(log(packpc), 1)  0.674051   0.061660 10.9317 < 2.2e-16 ***
## log(income95pc)     -0.066044   0.111880 -0.5903     0.555
## log(avgprs95)       -0.305656   0.043073 -7.0963 1.281e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Sargan Test: chisq(44) = 46.02872 (p.value=0.38824)
## Autocorrelation test (1): normal = -4.050332 (p.value=5.1145e-05)
## Autocorrelation test (2): normal = 0.05678179 (p.value=0.95472)
## Wald test for coefficients: chisq(3) = 2140.806 (p.value=< 2.22e-16)
```

Model 3: Elasticity specification

```r
# Good Sargan test, Good AR(2) test
summary(pgmm.res3e)
```

```
## Oneway (individual) effect One step model
##
## Call:
## pgmm(formula = pgmmformula.3a, data = pdata, effect = "individual",
##     transformation = "ld")
##
## Balanced Panel: n=48, T=11, N=528
##
## Number of Observations Used:  912
##
## Residuals
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## -0.366800 -0.022700  0.002000  0.001797  0.025150  0.278300
##
## Coefficients
##                       Estimate Std. Error  z-value Pr(>|z|)
## lag(log(packpc), 1)  0.9860119  0.0094198 104.6742   <2e-16 ***
## log(income95pc)     -0.0057655  0.0154370  -0.3735   0.7088
## log(avgprs95)        0.0110743  0.0077600   1.4271   0.1535
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Sargan Test: chisq(55) = 47.59902 (p.value=0.75038)
## Autocorrelation test (1): normal = -3.352045 (p.value=0.00080217)
## Autocorrelation test (2): normal = 0.982078 (p.value=0.32606)
## Wald test for coefficients: chisq(3) = 4737094 (p.value=< 2.22e-16)
```

Model 3: Elasticity specification

```
# Good Sargan test, Good AR(2) test, Wald supports 2-way
summary(pgmm.res3g)
```

```
## Twoways effects One step model
##
## Call:
## pgmm(formula = pgmmformula.3a, data = pdata, effect = "twoways",
##     transformation = "d")
##
## Balanced Panel: n=48, T=11, N=528
##
## Number of Observations Used:  432
##
## Residuals
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## -0.181000 -0.019490 -0.001678  0.000000  0.017810  0.204300
##
## Coefficients
##                     Estimate Std. Error z-value  Pr(>|z|)
## lag(log(packpc), 1)  0.33315    0.14500  2.2976   0.02158 *
## log(income95pc)      0.18131    0.18166  0.9981   0.31825
## log(avgprs95)       -0.62271    0.11127 -5.5965 2.188e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Sargan Test: chisq(44) = 43.30027 (p.value=0.5015)
## Autocorrelation test (1): normal = -3.620058 (p.value=0.00029454)
## Autocorrelation test (2): normal = 0.5219654 (p.value=0.60169)
## Wald test for coefficients: chisq(3) = 45.27325 (p.value=8.0946e-10)
## Wald test for time dummies: chisq(9) = 71.0946 (p.value=9.2856e-12)
```

Model 3: Elasticity specification

```r
# Good Sargan test, Good AR(2) test, Wald supports 2-way
summary(pgmm.res3h)
```

```
## Twoways effects One step model
##
## Call:
## pgmm(formula = pgmmformula.3a, data = pdata, effect = "twoways",
##     transformation = "ld")
##
## Balanced Panel: n=48, T=11, N=528
##
## Number of Observations Used:  912
##
## Residuals
##       Min.    1st Qu.     Median       Mean    3rd Qu.       Max.
## -0.3422000 -0.0233100  0.0006081  0.0000000  0.0224300  0.2716000
##
## Coefficients
##                       Estimate Std. Error z-value  Pr(>|z|)
## lag(log(packpc), 1)  0.9454089  0.0286753 32.9694 < 2.2e-16 ***
## log(income95pc)     -0.0072777  0.0214627 -0.3391  0.734544
## log(avgprs95)       -0.1650673  0.0494761 -3.3363  0.000849 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Sargan Test: chisq(55) = 44.19932 (p.value=0.85117)
## Autocorrelation test (1): normal = -3.066312 (p.value=0.0021672)
## Autocorrelation test (2): normal = 1.1093 (p.value=0.2673)
## Wald test for coefficients: chisq(3) = 4820.892 (p.value=< 2.22e-16)
## Wald test for time dummies: chisq(9) = 90.44993 (p.value=1.3225e-15)
```

Model 4: Elasticity specification, components of price

```
# Try difference GMM with two way effects
pgmm.res4g <- pgmm(pgmmformula.4a,
                   data = pdata,
                   effect = "twoways",    # should consider two-way for small T
                   transformation = "d")     # should do ld if T=3

# Try system GMM with two way effects
pgmm.res4h <- pgmm(pgmmformula.4a,
                   data = pdata,
                   effect = "twoways",    # should consider two-way for small T
                   transformation = "ld")     # should do ld if T=3
```

Model 4: Elasticity specification, components of price

```
# Good Sargan test, Good AR(2) test, Wald supports 2-way
summary(pgmm.res4g)
```

```
## Twoways effects One step model
##
## Call:
## pgmm(formula = pgmmformula.4a, data = pdata, effect = "twoways",
##     transformation = "d")
##
## Balanced Panel: n=48, T=11, N=528
##
## Number of Observations Used:  432
##
## Residuals
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## -0.159800 -0.019740 -0.001557  0.000000  0.018140  0.198300
##
## Coefficients
##                       Estimate Std. Error z-value  Pr(>|z|)
## lag(log(packpc), 1)  0.309885   0.135805  2.2818    0.0225 *
## log(income95pc)       0.160518   0.195323  0.8218    0.4112
## log(pretax95)        -0.086999   0.078535 -1.1078    0.2680
## log(taxs95)          -0.287494   0.045624 -6.3013 2.951e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Sargan Test: chisq(44) = 44.84174 (p.value=0.43636)
## Autocorrelation test (1): normal = -3.654349 (p.value=0.00025784)
## Autocorrelation test (2): normal = 1.083759 (p.value=0.27847)
## Wald test for coefficients: chisq(4) = 70.23767 (p.value=2.0222e-14)
## Wald test for time dummies: chisq(9) = 70.48873 (p.value=1.2211e-11)
```

Model 4: Elasticity specification, components of price

```
# Good Sargan test, Good AR(2) test, Wald supports 2-way
summary(pgmm.res4h)
```

```
## Twoways effects One step model
##
## Call:
## pgmm(formula = pgmmformula.4a, data = pdata, effect = "twoways",
##     transformation = "ld")
##
## Balanced Panel: n=48, T=11, N=528
##
## Number of Observations Used:  912
##
## Residuals
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## -0.356300 -0.022980  0.000891  0.000000  0.022530  0.272700
##
## Coefficients
##                       Estimate Std. Error z-value  Pr(>|z|)
## lag(log(packpc), 1)  0.953854   0.025812 36.9544 < 2.2e-16 ***
## log(income95pc)     -0.013174   0.019295 -0.6828  0.494750
## log(pretax95)       -0.107924   0.040645 -2.6553  0.007925 **
## log(taxs95)         -0.042752   0.014618 -2.9246  0.003450 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Sargan Test: chisq(56) = 46.48103 (p.value=0.81385)
## Autocorrelation test (1): normal = -3.05496 (p.value=0.0022509)
## Autocorrelation test (2): normal = 1.104899 (p.value=0.2692)
## Wald test for coefficients: chisq(4) = 5525 (p.value=< 2.22e-16)
## Wald test for time dummies: chisq(9) = 77.97589 (p.value=4.0745e-13)
```

Simulate conditional forecasts

```
# Forecast for 3 years from 1996 to 1998
periods.out <- 3
sims <- 1000

# How big a change in price to simulate?
# How about "double" the average tax in the most recent year?
summary(pdata$taxs95[pdata$year==1995])
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   34.44   48.75   59.84   61.87   74.78  112.60
```

```
# The average (and median) tax is about 60 cents/pack
sd(pdata$taxs95[pdata$year==1995])
```

```
## [1] 18.47741
```

Simulate conditional forecasts

```
# A 60 cent increase would also be about 3 sd's,
# and raise the tax to a bit more than the max observed

# Other possibilities:
# (2) A 10 cent increase
# (3) Raise every state to the max observed for any state in 1995 (112.60 cents)


# Construct the year dummies
yearfe <- makeFEdummies(pdata$year)          # Construct the dummies for each year
yearfe <- yearfe[,3:ncol(yearfe)]            # Why drop first 2 col's?
yearlist <- unique(pdata$year)               # List all the years
yearlist <- yearlist[3:length(yearlist)]     # List the years less the first two
colnames(yearfe) <- paste0("y",yearlist)     # Create names for the year dummies

# Construct formulas -- without year dummies (1a)
formula.1a <- packpc ~ income95pc + avgprs95 -1          #with Income and Price as covariates

# Construct formulas -- without year dummies but with intercept (1e)
formula.1e <- packpc ~ income95pc + avgprs95
```

Simulate conditional forecasts

```
# Construct formulas -- with year dummies (1g)
formula <- "packpc ~ income95pc + avgprs95 -1"      #Initial formula with no intercept
datayearfe <- cbind(pdata,yearfe)                   #Combine pdata variables with the year dummies
datayearfe[1:5,]
```

```
##   state year   cpi     pop   packpc   income  tax   avgprs     taxs
## 1    AL 1985 1.076 3973000 116.4863 46014968 32.5 102.1817 33.34834
## 2    AL 1986 1.096 3992000 117.1593 48703940 32.5 107.9892 33.40584
## 3    AL 1987 1.136 4016000 115.8367 51846312 32.5 113.5273 33.46067
## 4    AL 1988 1.183 4024000 115.2584 55698852 32.5 120.0334 33.52509
## 5    AL 1989 1.240 4030000 109.2060 60044480 32.5 133.2560 33.65600
##   income95    tax95 avgprs95   taxs95 income95pc  pretax95 y1987 y1988
## 1 65173615 46.03160 144.7257 47.23314   16.40413  97.49257     0     0
## 2 67723361 45.19161 150.1601 46.45118   16.96477 103.70893     0     0
## 3 69554378 43.60035 152.3025 44.88914   17.31932 107.41336     1     0
## 4 71754049 41.86813 154.6331 43.18869   17.83152 111.44437     0     1
## 5 73796599 39.94355 163.7759 41.36431   18.31181 122.41162     0     0
##   y1989 y1990 y1991 y1992 y1993 y1994 y1995
## 1     0     0     0     0     0     0     0
## 2     0     0     0     0     0     0     0
## 3     0     0     0     0     0     0     0
## 4     0     0     0     0     0     0     0
## 5     1     0     0     0     0     0     0
```

Simulate conditional forecasts

```
yearfenames <- NULL
for (i in 1:ncol(yearfe)) {
  formula <- paste0(formula,"+ y",yearlist[i]," ")          #Add the year dummies to the initial formula
  yearfenames <- c(yearfenames,paste0("y",yearlist[i]))     #Make a vector of names for the years
}
names(datayearfe) <- c(names(data),yearfenames)
formula.1g <- as.formula(formula)
formula.1g
```

```
## packpc ~ income95pc + avgprs95 - 1 + y1987 + y1988 + y1989 +
##     y1990 + y1991 + y1992 + y1993 + y1994 + y1995
```

```
# Construct formulas -- with year dummies and intercept (1h)
formula <- "packpc ~ income95pc + avgprs95"          #Initial formula without the year dummies
datayearfe <- cbind(pdata,yearfe)                    #Combine pdata variables with the year dummies

yearfenames <- NULL
for (i in 1:ncol(yearfe)) {
  formula <- paste0(formula,"+ y",yearlist[i]," ")  #Add the year dummies to the initial formula
  yearfenames <- c(yearfenames,paste0("y",yearlist[i])) #Make a vector of names for the years
}
names(datayearfe) <- c(names(data),yearfenames)
formula.1h <- as.formula(formula)

# Population in 1995 in average state
avgpop1995 <- mean(pdata$pop[pdata$year==1995])
```

Forecast: Model 1a, +60

```
# Recall model 1a: packpc ~ lag(packpc, 1) + income95pc + avgprs95 | lag(packpc, 2:99)
# Difference GMM with state fixed effects

# Simulate parameters
simparam.1a <- mvrnorm(sims, coefficients(pgmm.res1a), vcovHC(pgmm.res1a))
#Sample parameters from an mvrnorm
simphis.1a <- simparam.1a[,1]
#Extract the simulated phis
simbetas.1a <- simparam.1a[,2:ncol(simparam.1a)]
#Extract the simulated betas

simphis.1a[1:2]
```

```
## [1] 0.5492740 0.5911616
```

```
simbetas.1a[1:2,]
```

```
##      income95pc   avgprs95
## [1,] -0.4444569 -0.2239691
## [2,] -1.1680507 -0.1897537
```

Forecast: Model 1a, +60

```
# Make matrix of hypothetical x's:
# Assume an average state raised taxes 60 cents starting 1996
#

# Make matrix of hypothetical x's: covariates
xhyp.1a <- cfMake(formula.1a,datayearfe, periods.out)
#With mean packpc, income, and price for the forecast period

# pgmm uses covariates in differenced form
# so we want most of them to be 0 (no change)
# exceptions:
# (1) changes in covariates of interest
# (2) time dummies aren't differenced
xhyp.1a$x <- xhyp.1a$xpre <- 0*xhyp.1a$x
xhyp.1a <- cfChange(xhyp.1a, "avgprs95", x=60, scen=1)

# We can "ignore" the state fixed effects for now and add them later
# because model is total linear
```

Forecast: Model 1a, +60

```
# Create baseline scenario
xbase.1a <- xhyp.1a
xbase.1a$x <- xbase.1a$xpre

# We need a lag of the price per pack
lagY.1a <- NULL # Hypothetical previous change in Y for simulation
for (i in 1:length(pgmm.res1a$model))       #For 1 to 48
    lagY.1a <- c(lagY.1a, as.data.frame(pgmm.res1a$model[[i]])["1995",]$packpc)
#Hypothetical change in packpc for each state in 1995
lagY.1a <- mean(lagY.1a, na.rm=TRUE)    #Find the mean of these hypothetical previous changes

# Hypothetical initial level of Y for simulation
initialY <- mean(pdata$packpc[pdata$year==1995], na.rm=TRUE)    #The mean of packpc in 1995
```

Forecast: Model 1a, +60

```
# Simulate expected values of Y (on original level scale)
# out to periods.out given hypothetical future values of X,
# initial lags of the change in Y, and an initial level of Y
sim.ev1a <- ldvsimev(xhyp.1a,                # The matrix of hypothetical x's
                     simbetas.1a,            # The matrix of simulated betas
                     ci=0.95,              # Desired confidence interval
                     constant=NA,          # NA indicates no constant!
                     phi=simphis.1a,             # estimated AR parameters; length must match lagY
                     lagY=lagY.1a,           # lags of y, most recent last
                     transform="diff",     # "log" to undo log transformation,
                                           # "diff" to under first differencing
                                           # "difflog" to do both
                     initialY=initialY     # for differenced models, the lag of the level of y
                     )
```

Forecast: Model 1a, +60

```
# Simulate expected values of Y given no change in covariates
sim.base1a <- ldvsimev(xbase.1a,                # The matrix of hypothetical x's
                       simbetas.1a,            # The matrix of simulated betas
                       ci=0.95,              # Desired confidence interval
                       constant=NA,          # NA indicates no constant!
                       phi=simphis.1a,              # estimated AR parameters; length must match lagY
                       lagY=lagY.1a,           # lags of y, most recent last
                       transform="diff",    # "log" to undo log transformation,
                                           # "diff" to under first differencing
                                           # "difflog" to do both
                       initialY=initialY    # for differenced models, the lag of the level of y
                       )
```

Forecast: Model 1a, +60

```
# Simulate first differences in y
# out to periods.out given hypothetical future values of x, xpre,
# and initial lags of the change in y
sim.fd1a <- ldvsimfd(xhyp.1a,              # The matrix of hypothetical x's
                     simbetas.1a,          # The matrix of simulated betas
                     ci=0.95,              # Desired confidence interval
                     constant=NA,          # Column containing the constant
                                           # set to NA for no constant
                     phi=simphis.1a,       # estimated AR parameters; length must match lagY
                     lagY=lagY.1a,         # lags of y, most recent last
                     transform="diff",     # Model is differenced
                     #initialY=initialY    # Redundant in this case (fd of linear differenced Y)
                     )
```

Forecast: Model 1a, +60

```r
# Compute revenue effects
# Below is a rough attempt; it would be better to directly simulate these quantities
# It would also be better to wrap this in a function, to avoid typos in copy.paste.edit

# Population in 1995 in average state
avgpop1995 <- mean(pdata$pop[pdata$year==1995])

# Lost revenues from reduced consumption, dollars pc
revLost.1a <- lapply(sim.fd1a, function(x) mean(pdata$taxs95[pdata$year==1995])*x/100)
#Multiply change in consumption by mean tax revenues in 1995 and divide by 100 (for dollars)
revLost.1a
```

```
## $pe
##            [,1]
## [1,]  -6.774258
## [2,] -11.047238
## [3,] -13.762235
##
## $lower
##            [,1]
## [1,]  -8.844776
## [2,] -13.767698
## [3,] -16.654914
##
## $upper
##            [,1]
## [1,]  -4.713463
## [2,]  -8.106940
## [3,] -10.439280
##
## $se
##          [,1]
## [1,] 1.053755
## [2,] 1.457078
## [3,] 1.597410
```

Forecast: Model 1a, +60

```
# Added revenue from higher taxes on remaining consumption, dollars pc
# Sensitive to (implicit) consumption trend assumptions
revGain.1a <- lapply(sim.ev1a, function(x) 60*x/100)
#Multiply expected consumption by 60 cents and divide by 100 (for dollars)
revGain.1a
```

```
## $pe
##          [,1]
## [1,] 51.72390
## [2,] 47.89887
## [3,] 45.47208
##
## $lower
##          [,1]
## [1,] 49.64215
## [2,] 45.11712
## [3,] 42.54319
##
## $upper
##          [,1]
## [1,] 53.82550
## [2,] 50.89983
## [3,] 48.84296
##
## $se
##          [,1]
## [1,] 1.055502
## [2,] 1.480022
## [3,] 1.634110
```

Forecast: Model 1a, +60

```
# Net change in revenue, dollars pc
revNet.1a <- list(pe=revLost.1a$pe + revGain.1a$pe,
                  #Lost revenues from reduced consumption plus added revenues from higher taxes
                  lower=revLost.1a$lower + revGain.1a$lower,    #Lower bound
                  upper=revLost.1a$upper + revGain.1a$upper)    #Upper bound
revNet.1a
```

```
## $pe
##          [,1]
## [1,] 44.94964
## [2,] 36.85164
## [3,] 31.70984
##
## $lower
##          [,1]
## [1,] 40.79738
## [2,] 31.34942
## [3,] 25.88828
##
## $upper
##          [,1]
## [1,] 49.11203
## [2,] 42.79289
## [3,] 38.40368
```

Forecast: Model 1a, +60

```
# Total change in state revenue, in millions of dollars
revNetState.1a <- lapply(revNet.1a, function(x) avgpop1995*x/1000000)
#Multiply state population by net change pc and divide by one million
revNetState.1a
```

```
## $pe
##          [,1]
## [1,] 243.9175
## [2,] 199.9740
## [3,] 172.0723
##
## $lower
##          [,1]
## [1,] 221.3854
## [2,] 170.1164
## [3,] 140.4818
##
## $upper
##          [,1]
## [1,] 266.5046
## [2,] 232.2140
## [3,] 208.3961
```

Forecast: Model 1e, +60

```
# Recall model 1e: packpc ~ lag(packpc, 1) + income95pc + avgprs95 | lag(packpc, 2:99)
# System GMM with state fixed effects

# Simulate parameters
simparam.1e <- mvrnorm(sims, coefficients(pgmm.res1e), vcovHC(pgmm.res1e))
#Sample model parameters
simphis.1e <- simparam.1e[,1]
#Extract the phis
simbetas.1e <- simparam.1e[,2:ncol(simparam.1e)]
#Extract the betas

# System GMM does NOT difference the covariates

# Make matrix of hypothetical x's:
# Assume an average state raised taxes 60 cents starting 1996
#

# Make matrix of hypothetical x's: covariates
xhyp.1e <- cfMake(formula.1a, datayearfe, periods.out)
#With mean packpc, income, and price for the forecast period

# system pgmm uses covariates in *level* form
# -> back to our usual use of simcf; note apply to all 3 periods!
xhyp.1e <- cfChange(xhyp.1e, "avgprs95", x=60 + mean(pdata$avgprs95), scen=1:3)
#Add 60 cents to the avg price per pack
```

Forecast: Model 1e, +60

```
# State fixed effects are not removed from the covariates,
# but from the instruments (so we can ignore them here)

# Create baseline scenario
xbase.1e <- xhyp.1e
xbase.1e$x <- xbase.1e$xpre

# We need a lag of the price per pack, now in levels
# But the code above to extract it from the pgmm object won't work!
lagY.1e <- mean(pdata$packpc[pdata$year==1995], na.rm=TRUE)      #average packpc in 1995

# Hypothetical initial level of Y for simulation
initialY <- mean(pdata$packpc[pdata$year==1995], na.rm=TRUE)     #average packpc in 1995
```

Forecast: Model 1e, +60

Forecast: Model 1e, +60

```
# Simulate expected values of Y (on original level scale)
# out to periods.out given hypothetical future values of X,
# initial lags of the change in Y, and an initial level of Y
sim.ev1e <- ldvsimev(xhyp.1e,                # The matrix of hypothetical x's
                     simbetas.1e,            # The matrix of simulated betas
                     ci=0.95,              # Desired confidence interval
                     constant=NA,          # NA indicates no constant!
                     phi=simphis.1e,             # estimated AR parameters; length must match lagY
                     lagY=lagY.1e,           # lags of y, most recent last
                     transform="none",    # NOTE: System GMM is not differenced!
                     initialY=initialY
                     )
```

Forecast: Model 1e, +60

```
# Simulate expected values of Y given no change in covariates
sim.base1e <- ldvsimev(xbase.1e,                # The matrix of hypothetical x's
                       simbetas.1e,            # The matrix of simulated betas
                       ci=0.95,              # Desired confidence interval
                       constant=NA,          # NA indicates no constant!
                       phi=simphis.1e,             # estimated AR parameters; length must match lagY
                       lagY=lagY.1e,           # lags of y, most recent last
                       transform="none"    # NOTE: System GMM is not differenced!
                       )
```

Forecast: Model 1e, +60

```
# Simulate first differences in y
# out to periods.out given hypothetical future values of x, xpre,
# and initial lags of the change in y
sim.fd1e <- ldvsimfd(xhyp.1e,              # The matrix of hypothetical x's
                     simbetas.1e,          # The matrix of simulated betas
                     ci=0.95,              # Desired confidence interval
                     constant=NA,          # Column containing the constant
                                           # set to NA for no constant
                     phi=simphis.1e,       # estimated AR parameters; length must match lagY
                     lagY=lagY.1e,         # lags of y, most recent last
                     transform="none"      # NOTE: System GMM is not differenced!
                     )
```

Forecast: Model 1e, +60

```
# Simulate relative risks in y
# out to periods.out given hypothetical future values of x, xpre,
# and initial lags of the change in y
sim.rr1e <- ldvsimrr(xhyp.1e,              # The matrix of hypothetical x's
                     simbetas.1e,          # The matrix of simulated betas
                     ci=0.95,              # Desired confidence interval
                     constant=NA,          # Column containing the constant
                                           # set to NA for no constant
                     phi=simphis.1e,       # estimated AR parameters; length must match lagY
                     lagY=lagY.1e,         # lags of y, most recent last
                     transform="none"      # NOTE: System GMM is not differenced!
                     )
```

Forecast: Model 1e, +60

```
# Compute revenue effects
# Below is a rough attempt; it would be better to directly simulate these quantities
# It would also be better to wrap this in a function, to avoid typos in copy.paste.edit

# Population in 1995 in average state
avgpop1995 <- mean(pdata$pop[pdata$year==1995])

# Lost revenues from reduced consumption, dollars pc
revLost.1e <- lapply(sim.fd1e, function(x) mean(pdata$taxs95[pdata$year==1995])*x/100)
#Multiply change in consumption by mean tax revenues in 1995 and divide by 100 (for dollars)

# Added revenue from higher taxes on remaining consumption, dollars pc
# Sensitive to (implicit) consumption trend assumptions
revGain.1e <- lapply(sim.ev1e, function(x) 60*x/100)
#Multiply expected consumption by 60 cents and divide by 100 (for dollars)

# Net change in revenue, dollars pc
revNet.1e <- list(pe=revLost.1e$pe + revGain.1e$pe,
                  #Lost revenues from reduced consumption plus added revenues from higher taxes
                  lower=revLost.1e$lower + revGain.1e$lower,    #Lower bound
                  upper=revLost.1e$upper + revGain.1e$upper)    #Upper bound

# Total change in state revenue, in millions of dollars
revNetState.1e <- lapply(revNet.1e, function(x) avgpop1995*x/1000000)
#Multiply state population by net change pc and divide by one million
```

Forecast: Model 1g, +60

```
# Recall model 1g: packpc ~ lag(packpc, 1) + income95pc + avgprs95 | lag(packpc, 2:99)
# Difference GMM with state and year fixed effects

# Simulate parameters
simparam.1g <- mvrnorm(sims, coefficients(pgmm.res1g), vcovHC(pgmm.res1g))      #Sample parameters
simphis.1g <- simparam.1g[,1]                                                    #Extract the phis
simbetas.1g <- simparam.1g[,2:ncol(simparam.1g)]                                 #Extract the betas

# Make matrix of hypothetical x's:
# Assume an average state raised taxes 60 cents starting 1996
#
# Issues -- we need to somehow include the state and year FEs:
#           Let's set the state to be an "average" state in 1995,
#           and year to be like the last year (1995)

# Make matrix of hypothetical x's: covariates
xhyp.1g <- cfMake(formula.1g, datayearfe, periods.out)      #Including the year fixed effects

# pgmm uses covariates in differenced form
# so we want most of them to be 0 (no change)
# exceptions:
# (1) changes in covariates of interest
# (2) differenced time dummies require special care
xhyp.1g$x <- xhyp.1g$xpre <- 0*xhyp.1g$x
xhyp.1g <- cfChange(xhyp.1g, "avgprs95", x=60, scen=1)      #Assume tax is raised 60 cents in 1996
```

Forecast: Model 1g, +60

```
# We can "ignore" the state fixed effects for now and add them later
# because model is total linear
# Create baseline scenario
xbase.1g <- xhyp.1g
xbase.1g$x <- xbase.1g$xpre
xbase.1g
```

```
## $x
##   packpc income95pc avgprs95 y1987 y1988 y1989 y1990 y1991 y1992 y1993
## 1      0          0        0     0     0     0     0     0     0     0
## 2      0          0        0     0     0     0     0     0     0     0
## 3      0          0        0     0     0     0     0     0     0     0
##   y1994 y1995
## 1     0     0
## 2     0     0
## 3     0     0
##
## $xpre
##   packpc income95pc avgprs95 y1987 y1988 y1989 y1990 y1991 y1992 y1993
## 1      0          0        0     0     0     0     0     0     0     0
## 2      0          0        0     0     0     0     0     0     0     0
## 3      0          0        0     0     0     0     0     0     0     0
##   y1994 y1995
## 1     0     0
## 2     0     0
## 3     0     0
##
## $model
## packpc ~ income95pc + avgprs95 - 1 + y1987 + y1988 + y1989 +
##     y1990 + y1991 + y1992 + y1993 + y1994 + y1995
##
## attr(,"class")
## [1] "list"           "counterfactual"
```

Forecast: Model 1g, +60

```
xhyp.1g
```

```
## $x
##   packpc income95pc avgprs95 y1987 y1988 y1989 y1990 y1991 y1992 y1993
## 1      0          0       60     0     0     0     0     0     0     0
## 2      0          0        0     0     0     0     0     0     0     0
## 3      0          0        0     0     0     0     0     0     0     0
##   y1994 y1995
## 1     0     0
## 2     0     0
## 3     0     0
##
## $xpre
##   packpc income95pc avgprs95 y1987 y1988 y1989 y1990 y1991 y1992 y1993
## 1      0          0        0     0     0     0     0     0     0     0
## 2      0          0        0     0     0     0     0     0     0     0
## 3      0          0        0     0     0     0     0     0     0     0
##   y1994 y1995
## 1     0     0
## 2     0     0
## 3     0     0
##
## $model
## packpc ~ income95pc + avgprs95 - 1 + y1987 + y1988 + y1989 +
##     y1990 + y1991 + y1992 + y1993 + y1994 + y1995
##
## attr(,"class")
## [1] "list"           "counterfactual"
```

Forecast: Model 1g, +60

```
# We need a lag of the price per pack
lagY.1g <- NULL # Hypothetical previous change in Y for simulation

pgmm.res1g$model[1]
```

```
## $AL
##           packpc lag(packpc, 1)  income95pc   avgprs95
## 1987 -1.3226623      0.6730347 0.354547306   2.142378  1  0  0  0  0  0  0
## 1988 -0.5782090     -1.3226623 0.512205963   2.330570 -1  1  0  0  0  0  0
## 1989 -6.0524902     -0.5782090 0.480287962   9.142863  0 -1  1  0  0  0  0
## 1990  2.5389175     -6.0524902 0.148464583   3.489286  0  0 -1  1  0  0  0
## 1991 -4.7301254      2.5389175 0.042664143  13.691496  0  0  0 -1  1  0  0
## 1992 -0.1118164     -4.7301254 0.465548804  10.345649  0  0  0  0 -1  1  0
## 1993 -1.9451370     -0.1118164 0.006317597 -17.561808  0  0  0  0  0 -1  1
## 1994 -1.5314713     -1.9451370 0.419305971 -13.036122  0  0  0  0  0  0 -1
## 1995 -2.3408889     -1.5314713 0.288875052  -2.333091  0  0  0  0  0  0  0
##
## 1987  0 0
## 1988  0 0
## 1989  0 0
## 1990  0 0
## 1991  0 0
## 1992  0 0
## 1993  0 0
## 1994  1 0
## 1995 -1 1
```

Forecast: Model 1g, +60

```
for (i in 1:length(pgmm.res1g$model))
    lagY.1g <- c(lagY.1g, as.data.frame(pgmm.res1g$model[[i]])["1995",]$packpc)
#Store change in packpc 1995 for each state

lagY.1g <- mean(lagY.1g, na.rm=TRUE)
#Find the mean for all packpc changes in 1995

# Hypothetical initial level of Y for simulation
pdata$packpc[pdata$year==1995]
```

```
##   AL-1995   AR-1995   AZ-1995   CA-1995   CO-1995   CT-1995   DE-1995
## 101.08543 111.04297  71.95417  56.85931  82.58292  79.47219 124.46660
##   FL-1995   GA-1995   IA-1995   ID-1995   IL-1995   IN-1995   KS-1995
##  93.07455  97.47462  92.40160  74.84978  83.26508 134.25835  88.75344
##   KY-1995   LA-1995   MA-1995   MD-1995   ME-1995   MI-1995   MN-1995
## 172.64778 105.17613  76.62064  77.47355 102.46978  81.38825  82.94530
##   MO-1995   MS-1995   MT-1995   NC-1995   ND-1995   NE-1995   NH-1995
## 122.45028 105.58245  87.15957 121.53806  79.80697  87.27071 156.33675
##   NJ-1995   NM-1995   NV-1995   NY-1995   OH-1995   OK-1995   OR-1995
##  80.37137  64.66887  93.52612  70.81732 111.38010 108.68011  92.15575
##   PA-1995   RI-1995   SC-1995   SD-1995   TN-1995   TX-1995   UT-1995
##  95.64309  92.59980 108.08275  97.21923 122.32005  73.07931  49.27220
##   VA-1995   VT-1995   WA-1995   WI-1995   WV-1995   WY-1995
## 105.38687 122.33475  65.53092  92.46635 115.56883 112.23814
```

```
initialY <- mean(pdata$packpc[pdata$year==1995], na.rm=TRUE)
#Set the initial mean value of pack in 1995 across states
```

Forecast: Model 1g, +60

```
# Simulate expected values of Y (on original level scale)
# out to periods.out given hypothetical future values of X,
# initial lags of the change in Y, and an initial level of Y
sim.ev1g <- ldvsimev(xhyp.1g,                # The matrix of hypothetical x's
                     simbetas.1g,            # The matrix of simulated betas
                     ci=0.95,              # Desired confidence interval
                     constant=NA,          # NA indicates no constant!
                     phi=simphis.1g,             # estimated AR parameters; length must match lagY
                     lagY=lagY.1g,           # lags of y, most recent last
                     transform="diff",     # "log" to undo log transformation,
                                           # "diff" to under first differencing
                                           # "difflog" to do both
                     initialY=initialY     # for differenced models, the lag of the level of y
                     )
```

Forecast: Model 1g, +60

```
# Simulate expected values of Y given no change in covariates
sim.base1g <- ldvsimev(xbase.1g,                # The matrix of hypothetical x's
                       simbetas.1g,           # The matrix of simulated betas
                       ci=0.95,             # Desired confidence interval
                       constant=NA,         # NA indicates no constant!
                       phi=simphis.1g,             # estimated AR parameters; length must match lagY
                       lagY=lagY.1g,          # lags of y, most recent last
                       transform="diff",    # "log" to undo log transformation,
                                           # "diff" to under first differencing
                                           # "difflog" to do both
                       initialY=initialY    # for differenced models, the lag of the level of y
                       )
```

Forecast: Model 1g, +60

```
# Simulate first differences in y
# out to periods.out given hypothetical future values of x, xpre,
# and initial lags of the change in y
sim.fd1g <- ldvsimfd(xhyp.1g,              # The matrix of hypothetical x's
                     simbetas.1g,          # The matrix of simulated betas
                     ci=0.95,              # Desired confidence interval
                     constant=NA,          # Column containing the constant
                                           # set to NA for no constant
                     phi=simphis.1g,       # estimated AR parameters; length must match lagY
                     lagY=lagY.1g,         # lags of y, most recent last
                     transform="diff",     # Model is differenced
                     #initialY=initialY    # Redundant in this case (fd of linear differenced Y)
                     )
```

Forecast: Model 1g, +60

```
# Compute revenue effects
# Below is a rough attempt; it would be better to directly simulate these quantities
# It would also be better to wrap this in a function, to avoid typos in copy.paste.edit

# Population in 1995 in average state
avgpop1995 <- mean(pdata$pop[pdata$year==1995])

# Lost revenues from reduced consumption, dollars pc
revLost.1g <- lapply(sim.fd1g, function(x) mean(pdata$taxs95[pdata$year==1995])*x/100)
#Multiply change in consumption by mean tax revenues in 1995 and divide by 100 (for dollars)

# Added revenue from higher taxes on remaining consumption, dollars pc
# Note this is sensitive to assumptions about consumption trends embodied by year effects
revGain.1g <- lapply(sim.ev1g, function(x) 60*x/100)
#Multiply expected consumption by 60 cents and divide by 100 (for dollars)

# Net change in revenue, dollars pc
revNet.1g <- list(pe=revLost.1g$pe + revGain.1g$pe,
                  #Lost revenues from reduced consumption plus added revenues from higher taxes
                  lower=revLost.1g$lower + revGain.1g$lower, #Lower bound
                  upper=revLost.1g$upper + revGain.1g$upper) #Upper bound

# Total change in state revenue, in millions of dollars
revNetState.1g <- lapply(revNet.1g, function(x) avgpop1995*x/1000000)
#Multiply state population by net change pc and divide by one million
```

Forecast: Model 1h, +60

```
# Recall model 1h: packpc ~ lag(packpc, 1) + income95pc + avgprs95 | lag(packpc, 2:99)
# System GMM with state and year fixed effects

# Simulate parameters
simparam.1h <- mvrnorm(sims, coefficients(pgmm.res1h), vcovHC(pgmm.res1h))
#Sample parameters
simphis.1h <- simparam.1h[,1]
#Extract the phis
simbetas.1h <- simparam.1h[,2:ncol(simparam.1h)]
#Extract the betas

# System GMM does NOT difference the covariates
# -> with 2-way effects, the model has a constant,
# which pgmm() puts in an odd place
simbetas.1h <- cbind(simbetas.1h[,3], simbetas.1h[,-3])
# Move the constant to the front of the matrix!
```

Forecast: Model 1g, +60

```
# Make matrix of hypothetical x's:
# Assume an average state raised taxes 60 cents starting 1996
#
# Issues -- we need to somehow include the state and year FEs:
#           Let's set the state to be an "average" state in 1995,
#           and year to be like the last year (1995)

# Make matrix of hypothetical x's: covariates
xhyp.1h <- cfMake(formula.1h, datayearfe, periods.out)
#Create hypothetical matrix with covariates at their mean

# system pgmm uses covariates in *level* form
# -> back to our usual use of simcf; note apply to all 3 periods!
xhyp.1h <- cfChange(xhyp.1h, "avgprs95", x=60 + mean(pdata$avgprs95), scen=1:3)
#Assume tax raises price by 60 cents
```

Forecast: Model 1g, +60

```
# The current trend seems to start in 1993; we will average over the
# the last three years of year effects:
xhyp.1h <- cfChange(xhyp.1h, "y1987", x=0, xpre=0, scen=1:3)
xhyp.1h <- cfChange(xhyp.1h, "y1988", x=0, xpre=0, scen=1:3)
xhyp.1h <- cfChange(xhyp.1h, "y1989", x=0, xpre=0, scen=1:3)
xhyp.1h <- cfChange(xhyp.1h, "y1990", x=0, xpre=0, scen=1:3)
xhyp.1h <- cfChange(xhyp.1h, "y1991", x=0, xpre=0, scen=1:3)
xhyp.1h <- cfChange(xhyp.1h, "y1992", x=0, xpre=0, scen=1:3)
xhyp.1h <- cfChange(xhyp.1h, "y1993", x=1/3, xpre=1/3, scen=1:3)
#Start the trend in 1993 averaged over last three years
xhyp.1h <- cfChange(xhyp.1h, "y1994", x=1/3, xpre=1/3, scen=1:3)
xhyp.1h <- cfChange(xhyp.1h, "y1995", x=1/3, xpre=1/3, scen=1:3)


# State fixed effects are not removed from the covariates,
# but from the instruments (so we can ignore them here)

# Create baseline scenario
xbase.1h <- xhyp.1h
xbase.1h$x <- xbase.1h$xpre

# We need a lag of the price per pack, now in levels
# But the code above to extract it from the pgmm object won't work!
lagY.1h <- mean(pdata$packpc[pdata$year==1995], na.rm=TRUE)
#Find the mean of packpc in 1995 across all states

# Hypothetical initial level of Y for simulation
initialY <- mean(pdata$packpc[pdata$year==1995], na.rm=TRUE)
#Find the mean of packpc in 1995 across all states
```

Forecast: Model 1g, +60

```
# Simulate expected values of Y (on original level scale)
# out to periods.out given hypothetical future values of X,
# initial lags of the change in Y, and an initial level of Y
sim.ev1h <- ldvsimev(xhyp.1h,                # The matrix of hypothetical x's
                     simbetas.1h,            # The matrix of simulated betas
                     ci=0.95,              # Desired confidence interval
                     constant=1,          # NOTE: System GMM has a constant!
                                           # You will need to note the column of the constant in simbetas
                     phi=simphis.1h,              # estimated AR parameters; length must match lagY
                     lagY=lagY.1h,           # lags of y, most recent last
                     transform="none"    # NOTE: System GMM is not differenced!
                     )
```

Forecast: Model 1g, +60

```
# Simulate expected values of Y given no change in covariates
sim.base1h <- ldvsimev(xbase.1h,                # The matrix of hypothetical x's
                       simbetas.1h,            # The matrix of simulated betas
                       ci=0.95,              # Desired confidence interval
                       constant=1,          # NOTE: System GMM has a constant!
                                           # You will need to note the column of the constant in simbetas
                       phi=simphis.1h,              # estimated AR parameters; length must match lagY
                       lagY=lagY.1h,           # lags of y, most recent last
                       transform="none"    # NOTE: System GMM is not differenced!
                       )
```

Forecast: Model 1g, +60

```
# Simulate first differences in y
# out to periods.out given hypothetical future values of x, xpre,
# and initial lags of the change in y
sim.fd1h <- ldvsimfd(xhyp.1h,              # The matrix of hypothetical x's
                     simbetas.1h,          # The matrix of simulated betas
                     ci=0.95,              # Desired confidence interval
                     constant=1,          # Column containing the constant
                                           # set to NA for no constant
                     phi=simphis.1h,       # estimated AR parameters; length must match lagY
                     lagY=lagY.1h,         # lags of y, most recent last
                     transform="none"     # NOTE: System GMM is not differenced!
                     )
```

Forecast: Model 1g, +60

```
# Simulate relative risks in y
# out to periods.out given hypothetical future values of x, xpre,
# and initial lags of the change in y
sim.rr1h <- ldvsimrr(xhyp.1h,              # The matrix of hypothetical x's
                     simbetas.1h,          # The matrix of simulated betas
                     ci=0.95,              # Desired confidence interval
                     constant=1,          # Column containing the constant
                                           # set to NA for no constant
                     phi=simphis.1h,       # estimated AR parameters; length must match lagY
                     lagY=lagY.1h,         # lags of y, most recent last
                     transform="none"     # NOTE: System GMM is not differenced!
                     )
```

Forecast: Model 1g, +60

```
# Compute revenue effects
# Below is a rough attempt; it would be better to directly simulate these quantities
# It would also be better to wrap this in a function, to avoid typos in copy.paste.edit

# Population in 1995 in average state
avgpop1995 <- mean(pdata$pop[pdata$year==1995])

# Lost revenues from reduced consumption, dollars pc
revLost.1h <- lapply(sim.fd1h, function(x) mean(pdata$taxs95[pdata$year==1995])*x/100)

# Added revenue from higher taxes on remaining consumption, dollars pc
# Note this is sensitive to assumptions about consumption trends embodied by year effects
revGain.1h <- lapply(sim.ev1h, function(x) 60*x/100)

# Net change in revenue, dollars pc
revNet.1h <- list(pe=revLost.1h$pe + revGain.1h$pe,
                  lower=revLost.1h$lower + revGain.1h$lower,
                  upper=revLost.1h$upper + revGain.1h$upper)

# Total change in state revenue, in millions of dollars
revNetState.1h <- lapply(revNet.1h, function(x) avgpop1995*x/1000000)
```

Forecast: Model 3a, +60

```
# Recall model 3a: log(packpc) ~ lag(log(packpc), 1) + log(income95pc)
# + log(avgprs95) | lag(log(packpc), 2:99)
# log-log Difference GMM with state fixed effects

# Simulate parameters
simparam.3a <- mvrnorm(sims, coefficients(pgmm.res3a), vcovHC(pgmm.res3a))
simphis.3a <- simparam.3a[,1]
simbetas.3a <- simparam.3a[,2:ncol(simparam.3a)]

# Make matrix of hypothetical x's:
# Assume an average state raised taxes 60 cents starting 1996
# Make matrix of hypothetical x's: covariates
xhyp.3a <- cfMake(formula.1a, datayearfe, periods.out)

# pgmm uses covariates in differenced form
# so we want most of them to be 0 (no change)
# exceptions:
# (1) changes in covariates of interest
# (2) time dummies aren't differenced
xhyp.3a$x <- xhyp.3a$xpre <- 0*xhyp.3a$x

# Need log version of differenced key covariate (doubling tax in avg state)
meanPrice95 <- mean(pdata$avgprs95[pdata$year==1995], na.rm=TRUE)
#Find the mean of avgprs95 across all states
meanTaxs95 <- mean(pdata$taxs95[pdata$year==1995], na.rm=TRUE)
#Find the mean of taxs95 across all states

xhyp.3a <- cfChange(xhyp.3a, "avgprs95",
                    #Change avgprs95 to log difference in mean price
                    x=log(meanPrice95+meanTaxs95) - log(meanPrice95),
                    scen=1)
```

Forecast: Model 3a, +60

```r
# We can "ignore" the state fixed effects for now and add them later
# because model is total linear

# Create baseline scenario
xbase.3a <- xhyp.3a
xbase.3a$x <- xbase.3a$xpre

# We need a lag of the price per pack
lagY.3a <- NULL # Hypothetical previous change in Y for simulation
for (i in 1:length(pgmm.res3a$model))
    lagY.3a <- c(lagY.3a, as.data.frame(pgmm.res3a$model[[i]])["1995",1])
#Find the change in packpc across all states
lagY.3a <- mean(lagY.3a, na.rm=TRUE)
#Compute the mean

# Hypothetical initial level of Y for simulation
initialY <- mean(pdata$packpc[pdata$year==1995], na.rm=TRUE)


# Simulate expected values of Y (on original level scale)
# out to periods.out given hypothetical future values of X,
# initial lags of the change in Y, and an initial level of Y
sim.ev3a <- ldvsimev(xhyp.3a,                # The matrix of hypothetical x's
                     simbetas.3a,            # The matrix of simulated betas
                     ci=0.95,             # Desired confidence interval
                     constant=NA,         # NA indicates no constant!
                     phi=simphis.3a,             # estimated AR parameters; length must match lagY
                     lagY=lagY.3a,           # lags of y, most recent last
                     transform="difflog",    # "log" to undo log transformation,
                                          # "diff" to under first differencing
                                          # "difflog" to do both
                     initialY=initialY    # for differenced models, the lag of the level of y
                     )
```

Forecast: Model 3a, +60

```
# Simulate expected values of Y given no change in covariates
sim.base3a <- ldvsimev(xbase.3a,                # The matrix of hypothetical x's
                       simbetas.3a,            # The matrix of simulated betas
                       ci=0.95,              # Desired confidence interval
                       constant=NA,          # NA indicates no constant!
                       phi=simphis.3a,              # estimated AR parameters; length must match lagY
                       lagY=lagY.3a,           # lags of y, most recent last
                       transform="difflog",    # "log" to undo log transformation,
                                             # "diff" to under first differencing
                                              # "difflog" to do both
                       initialY=initialY    # for differenced models, the lag of the level of y
                       )
```

Forecast: Model 3e, +60

```
# Recall model 3e: log(packpc) ~ lag(log(packpc), 1) + log(income95pc)
# + log(avgprs95) | lag(log(packpc), 2:99)
# log-log System GMM with state fixed effects

# Because system GMM is in levels, it is convenient to
# handle logging through the formula combined with simcf
formula.3e <- log(packpc) ~ log(income95pc) + log(avgprs95) -1

# Simulate parameters
simparam.3e <- mvrnorm(sims, coefficients(pgmm.res3e), vcovHC(pgmm.res3e))
simphis.3e <- simparam.3e[,1]
simbetas.3e <- simparam.3e[,2:ncol(simparam.3e)]

# System GMM does NOT difference the covariates

# Make matrix of hypothetical x's:
# Assume an average state raised taxes 60 cents starting 1996
#
# Make matrix of hypothetical x's: covariates
xhyp.3e <- cfMake(formula.3e, datayearfe, periods.out)  #See log transformation in formula.3e

# system pgmm uses covariates in *level* form
# -> back to our usual use of simcf; note apply to all 3 periods!
xhyp.3e <- cfChange(xhyp.3e, "avgprs95", x=60 + mean(pdata$avgprs95), scen=1:3)
```

Forecast: Model 3e, +60

```
# State fixed effects are not removed from the covariates,
# but from the instruments (so we can ignore them here)

# Create baseline scenario
xbase.3e <- xhyp.3e
xbase.3e$x <- xbase.3e$xpre

# We need a lag of the price per pack, now in logged levels
# But the code above to extract it from the pgmm object won't work!
# Getting this right is crucial
lagY.3e <- log(mean(pdata$packpc[pdata$year==1995], na.rm=TRUE))

# Hypothetical initial level of Y for simulation
# Still in linear levels
initialY <- mean(pdata$packpc[pdata$year==1995], na.rm=TRUE)
```

Forecast: Model 3e, +60

```
# Simulate expected values of Y (on original level scale)
# out to periods.out given hypothetical future values of X,
# initial lags of the change in Y, and an initial level of Y
sim.ev3e <- ldvsimev(xhyp.3e,                # The matrix of hypothetical x's
                     simbetas.3e,            # The matrix of simulated betas
                     ci=0.95,              # Desired confidence interval
                     constant=NA,          # NA indicates no constant!
                     phi=simphis.3e,             # estimated AR parameters; length must match lagY
                     lagY=lagY.3e,           # lags of y, most recent last
                     transform="log"    # NOTE: System GMM is not differenced!
                     )
```

Forecast: Model 3e, +60

```
# Simulate expected values of Y given no change in covariates
sim.base3e <- ldvsimev(xbase.3e,                # The matrix of hypothetical x's
                       simbetas.3e,           # The matrix of simulated betas
                       ci=0.95,               # Desired confidence interval
                       constant=NA,           # NA indicates no constant!
                       phi=simphis.3e,              # estimated AR parameters; length must match lagY
                       lagY=lagY.3e,          # lags of y, most recent last
                       transform="log",    # NOTE: System GMM is not differenced!
                       )
```

Forecast: Model 3e, +60

```
# Simulate first differences in y
# out to periods.out given hypothetical future values of x, xpre,
# and initial lags of the change in y
sim.fd3e <- ldvsimfd(xhyp.3e,              # The matrix of hypothetical x's
                     simbetas.3e,          # The matrix of simulated betas
                     ci=0.95,              # Desired confidence interval
                     constant=NA,          # Column containing the constant
                                           # set to NA for no constant
                     phi=simphis.3e,       # estimated AR parameters; length must match lagY
                     lagY=lagY.3e,         # lags of y, most recent last
                     transform="log"       # NOTE: System GMM is not differenced!
                     )
```

Forecast: Model 3e, +60

```
# Simulate relative risks in y
# out to periods.out given hypothetical future values of x, xpre,
# and initial lags of the change in y
sim.rr3e <- ldvsimrr(xhyp.3e,              # The matrix of hypothetical x's
                     simbetas.3e,          # The matrix of simulated betas
                     ci=0.95,              # Desired confidence interval
                     constant=NA,          # Column containing the constant
                                           # set to NA for no constant
                     phi=simphis.3e,       # estimated AR parameters; length must match lagY
                     lagY=lagY.3e,         # lags of y, most recent last
                     transform="log"       # NOTE: System GMM is not differenced!
                     )
```

Forecast: Model 3e, +60

```
# Compute revenue effects
# Below is a rough attempt; it would be better to directly simulate these quantities
# It would also be better to wrap this in a function, to avoid typos in copy.paste.edit

# Population in 1995 in average state
avgpop1995 <- mean(pdata$pop[pdata$year==1995])

# Lost revenues from reduced consumption, dollars pc
revLost.3e <- lapply(sim.fd3e, function(x) mean(pdata$taxs95[pdata$year==1995])*x/100)

# Added revenue from higher taxes on remaining consumption, dollars pc
# Sensitive to (implicit) consumption trend assumptions
revGain.3e <- lapply(sim.ev3e, function(x) 60*x/100)

# Net change in revenue, dollars pc
revNet.3e <- list(pe=revLost.3e$pe + revGain.3e$pe,
                  lower=revLost.3e$lower + revGain.3e$lower,
                  upper=revLost.3e$upper + revGain.3e$upper)

# Total change in state revenue, in millions of dollars
revNetState.3e <- lapply(revNet.3e, function(x) avgpop1995*x/1000000)
```

Forecast: Model 3g, +60

```
# Recall model 3g: log(packpc) ~ lag(log(packpc), 1) + log(income95pc)
#+ log(avgprs95) | lag(log(packpc), 2:99)
# log log Difference GMM with state and year fixed effects

simparam.3g <- mvrnorm(sims, coefficients(pgmm.res3g), vcovHC(pgmm.res3g))
simphis.3g <- simparam.3g[,1]
simbetas.3g <- simparam.3g[,2:ncol(simparam.3g)]

# Make matrix of hypothetical x's:
# Assume an average state raised taxes 60 cents starting 1996
#
# Issues -- we need to somehow include the state and year FEs:
#           Let's set the state to be an "average" state in 1995,
#           and year to be like the last year (1995)

# Make matrix of hypothetical x's: covariates
# Still use the 1g formula (no logs) -- we will handle logging manually
#  to get the differences of logs right
xhyp.3g <- cfMake(formula.1g, datayearfe, periods.out)
```

Forecast: Model 3g, +60

```
# pgmm uses covariates in differenced form
# so we want most of them to be 0 (no change)
# exceptions:
# (1) changes in covariates of interest
# (2) time dummies aren't differenced
xhyp.3g$x <- xhyp.3g$xpre <- 0*xhyp.3g$x

# Need log version of differenced key covariate (doubling tax in avg state)
meanPrice95 <- mean(pdata$avgprs95[pdata$year==1995], na.rm=TRUE)
meanTaxs95 <- mean(pdata$taxs95[pdata$year==1995], na.rm=TRUE)

xhyp.3g <- cfChange(xhyp.3g, "avgprs95",
                    x=log(meanPrice95+meanTaxs95) - log(meanPrice95),
                    scen=1)

xhyp.3g <- cfChange(xhyp.3g, "y1995", x=1, xpre=1, scen=1:3)
```

Forecast: Model 3g, +60

```r
# We can "ignore" the state fixed effects for now and add them later
# because model is total linear

# Create baseline scenario
xbase.3g <- xhyp.3g
xbase.3g$x <- xbase.3g$xpre

# We need a lag of the price per pack
lagY.3g <- NULL # Hypothetical previous change in Y for simulation
for (i in 1:length(pgmm.res3g$model))
    lagY.3g <- c(lagY.3g, as.data.frame(pgmm.res3g$model[[i]])["1995",1])
lagY.3g <- mean(lagY.3g, na.rm=TRUE)


initialY <- mean(pdata$packpc[pdata$year==1995], na.rm=TRUE)
               # Hypothetical initial level of Y for simulation
```

Forecast: Model 3g, +60

```
# Simulate expected values of Y (on original level scale)
# out to periods.out given hypothetical future values of X,
# initial lags of the change in Y, and an initial level of Y
sim.ev3g <- ldvsimev(xhyp.3g,                # The matrix of hypothetical x's
                     simbetas.3g,            # The matrix of simulated betas
                     ci=0.95,              # Desired confidence interval
                     constant=NA,          # NA indicates no constant!
                     phi=simphis.3g,             # estimated AR parameters; length must match lagY
                     lagY=lagY.3g,           # lags of y, most recent last
                     transform="difflog",    # "log" to undo log transformation,
                                           # "diff" to under first differencing
                                           # "difflog" to do both
                     initialY=initialY     # for differenced models, the lag of the level of y
                     )
```

Forecast: Model 3g, +60

```
# Simulate expected values of Y given no change in covariates
sim.base3g <- ldvsimev(xbase.3g,                # The matrix of hypothetical x's
                       simbetas.3g,            # The matrix of simulated betas
                       ci=0.95,              # Desired confidence interval
                       constant=NA,          # NA indicates no constant!
                       phi=simphis.3g,              # estimated AR parameters; length must match lagY
                       lagY=lagY.3g,           # lags of y, most recent last
                       transform="difflog",    # "log" to undo log transformation,
                                             # "diff" to under first differencing
                                             # "difflog" to do both
                       initialY=initialY    # for differenced models, the lag of the level of y
                       )
```

Forecast: Model 3g, +60

```
# Below is a rough attempt; it would be better to directly simulate these quantities
# It would also be better to wrap this in a function, to avoid typos in copy.paste.edit

# Population in 1995 in average state
avgpop1995 <- mean(pdata$pop[pdata$year==1995])

# Lost revenues from reduced consumption, dollars pc
revLost.3g <- lapply(sim.fd1g, function(x) mean(pdata$taxs95[pdata$year==1995])*x/100)

# Added revenue from higher taxes on remaining consumption, dollars pc
# Note this is sensitive to assumptions about consumption trends embodied by year effects
revGain.3g <- lapply(sim.ev1g, function(x) 60*x/100)

# Net change in revenue, dollars pc
revNet.3g <- list(pe=revLost.3g$pe + revGain.3g$pe,
                  lower=revLost.3g$lower + revGain.3g$lower,
                  upper=revLost.3g$upper + revGain.3g$upper)

# Total change in state revenue, in millions of dollars
revNetState.3g <- lapply(revNet.3g, function(x) avgpop1995*x/1000000)
```

Forecast: Model 3h, +60

```
# Recall model 3h: log(packpc) ~ lag(log(packpc), 1) + log(income95pc)
# + log(avgprs95) | lag(log(packpc), 2:99)
# log log System GMM with state and year fixed effects

# Because system GMM is in levels, it is convenient to
# handle logging through the formula combined with simcf
formula <- "log(packpc) ~ log(income95pc) + log(avgprs95)"
datayearfe <- cbind(pdata,yearfe)
yearfenames <- NULL                                    #Create an empty vector of the year names
for (i in 1:ncol(yearfe)) {
  formula <- paste0(formula,"+ y",yearlist[i]," ")     #Add year names to formula
  yearfenames <- c(yearfenames,paste0("y",yearlist[i]))
}
names(datayearfe) <- c(names(data),yearfenames)        #Add year names to datayearfe

formula.3h <- as.formula(formula)

# Simulate parameters
simparam.3h <- mvrnorm(sims, coefficients(pgmm.res3h), vcovHC(pgmm.res3h))
simphis.3h <- simparam.3h[,1]
simbetas.3h <- simparam.3h[,2:ncol(simparam.3h)]

# System GMM does NOT difference the covariates
# -> the model has a constant, which pgmm() puts in an odd place
# Move the constant to the front of the matrix!
simbetas.3h <- cbind(simbetas.3h[,3], simbetas.3h[,-3])
```

Forecast: Model 3h, +60

```
# Make matrix of hypothetical x's:
# Assume an average state raised taxes 60 cents starting 1996
#
# Issues -- we need to somehow include the state and year FEs:
#           Let's set the state to be an "average" state in 1995,
#           and year to be like the last year (1995)

# Make matrix of hypothetical x's: covariates
xhyp.3h <- cfMake(formula.3h, datayearfe, periods.out)


# system pgmm uses covariates in *level* form
# -> back to our usual use of simcf; let simcf handle logging here
xhyp.3h <- cfChange(xhyp.3h, "avgprs95", x=60 + mean(pdata$avgprs95), scen=1:3)
xhyp.3h <- cfChange(xhyp.3h, "y1987", x=0, xpre=0, scen=1:3)
xhyp.3h <- cfChange(xhyp.3h, "y1988", x=0, xpre=0, scen=1:3)
xhyp.3h <- cfChange(xhyp.3h, "y1989", x=0, xpre=0, scen=1:3)
xhyp.3h <- cfChange(xhyp.3h, "y1990", x=0, xpre=0, scen=1:3)
xhyp.3h <- cfChange(xhyp.3h, "y1991", x=0, xpre=0, scen=1:3)
xhyp.3h <- cfChange(xhyp.3h, "y1992", x=0, xpre=0, scen=1:3)
xhyp.3h <- cfChange(xhyp.3h, "y1993", x=0, xpre=0, scen=1:3)
xhyp.3h <- cfChange(xhyp.3h, "y1994", x=0, xpre=0, scen=1:3)
xhyp.3h <- cfChange(xhyp.3h, "y1995", x=1, xpre=1, scen=1:3)

# State fixed effects are not removed from the covariates,
# but from the instruments (so we can ignore them here)
```

Forecast: Model 3h, +60

```
# Create baseline scenario
xbase.3h <- xhyp.3h
xbase.3h$x <- xbase.3h$xpre

# We need a lag of the price per pack, now in logged levels
# But the code above to extract it from the pgmm object won't work!
# Getting this right is crucial
lagY.3h <- log(mean(pdata$packpc[pdata$year==1995], na.rm=TRUE))

# Hypothetical initial level of Y for simulation
# Still in linear levels
initialY <- mean(pdata$packpc[pdata$year==1995], na.rm=TRUE)

# Simulate expected values of Y (on original level scale)
# out to periods.out given hypothetical future values of X,
# initial lags of the change in Y, and an initial level of Y
sim.ev3h <- ldvsimev(xhyp.3h,                # The matrix of hypothetical x's
                     simbetas.3h,            # The matrix of simulated betas
                     ci=0.95,              # Desired confidence interval
                     constant=1,          # NOTE: System GMM with two-way effects has a constant!
                                           # You will need to note the column of the constant in simbetas
                     phi=simphis.3h,             # estimated AR parameters; length must match lagY
                     lagY=lagY.3h,           # lags of y, most recent last
                     transform="log"    # NOTE: System GMM is not differenced!
                     )
```

Forecast: Model 3h, +60

```
# Simulate expected values of Y given no change in covariates
sim.base3h <- ldvsimev(xbase.3h,                # The matrix of hypothetical x's
                       simbetas.3h,            # The matrix of simulated betas
                       ci=0.95,              # Desired confidence interval
                       constant=1,          # NOTE: System GMM with two-way effects has a constant!
                                           # You will need to note the column of the constant in simbetas
                       phi=simphis.3h,              # estimated AR parameters; length must match lagY
                       lagY=lagY.3h,           # lags of y, most recent last
                       transform="log"    # NOTE: System GMM is not differenced!
                       )
```

Forecast: Model 3h, +60

```
# Simulate first differences in y
# out to periods.out given hypothetical future values of x, xpre,
# and initial lags of the change in y
sim.fd3h <- ldvsimfd(xhyp.3h,              # The matrix of hypothetical x's
                     simbetas.3h,          # The matrix of simulated betas
                     ci=0.95,              # Desired confidence interval
                     constant=1,          # Column containing the constant
                                           # set to NA for no constant
                     phi=simphis.3h,       # estimated AR parameters; length must match lagY
                     lagY=lagY.3h,         # lags of y, most recent last
                     transform="log"    # NOTE: System GMM is not differenced!
                     )
```

Forecast: Model 3h, +60

```
# Simulate relative risks in y
# out to periods.out given hypothetical future values of x, xpre,
# and initial lags of the change in y
sim.rr3h <- ldvsimrr(xhyp.3h,              # The matrix of hypothetical x's
                     simbetas.3h,          # The matrix of simulated betas
                     ci=0.95,              # Desired confidence interval
                     constant=1,          # Column containing the constant
                                           # set to NA for no constant
                     phi=simphis.3h,       # estimated AR parameters; length must match lagY
                     lagY=lagY.3h,         # lags of y, most recent last
                     transform="log"    # NOTE: System GMM is not differenced!
                     )
```

Forecast: Model 3h, +60

```
# Compute revenue effects
# Below is a rough attempt; it would be better to directly simulate these quantities
# It would also be better to wrap this in a function, to avoid typos in copy.paste.edit

# Population in 1995 in average state
avgpop1995 <- mean(pdata$pop[pdata$year==1995])

# Lost revenues from reduced consumption, dollars pc
revLost.3h <- lapply(sim.fd3h, function(x) mean(pdata$taxs95[pdata$year==1995])*x/100)

# Added revenue from higher taxes on remaining consumption, dollars pc
# Note this is sensitive to assumptions about consumption trends embodied by year effects
revGain.3h <- lapply(sim.ev3h, function(x) 60*x/100)

# Net change in revenue, dollars pc
revNet.3h <- list(pe=revLost.3h$pe + revGain.3h$pe,
                  lower=revLost.3h$lower + revGain.3h$lower,
                  upper=revLost.3h$upper + revGain.3h$upper)

# Total change in state revenue, in millions of dollars
revNetState.3h <- lapply(revNet.3h, function(x) avgpop1995*x/1000000)


# Make plots of expected values, first differences, and percent changes
# using custom tile code in helperCigs.R

# Hypothetical initial level of Y for simulation
initialY <- mean(pdata$packpc[pdata$year==1995], na.rm=TRUE)
```