COMMENTS ON THE PHILLIPS CURVE EXERCISE

The numerical answers to this exercise varied widely, though most followed an apparently reasonable approach.

One reason for variation was that some converted the problem to a standard LQ dynamic optimization, requiring solving a Riccati equation (which reproduces the main computational work done by `gensys`). This was OK, but it made the simplest way to frame the problem be the version in which the optimizer does not see the current Phillips curve shock when choosing u and π. Since the simplest version of the problem in the `gensys` framework is the version in which the optimizer does see the current shock, this is one reason for discrepancies.

A second reason is that some added a constant term to the Phillips curve and some didn't. Real Phillips curves do have a constant, but in the examples I went through in class for this problem I left out the constant. For calculating `g0` and `g1`, the constant doesn't matter, and the `G1` produced by `gensys` does not depend on the constant, but of course the estimated Phillips curve itself is likely to be quite different with a constant.

A Phillps curve without a constant in this problem's setup implies that zero inflation is compatible with zero unemployment, except for the shocks, so it is likely to make solutions more unrealistic.

There are in fact no existence or uniqueness problems in applying `gensys` to this problem. I have not had time to trace through the code in the exercise answers to locate errors, but students who thought they had found problems with existence and uniqueness are almost surely mistaken.

Setting the problem up as a Bellman equation and taking FOC's could also work in principle, but it leads to a set of equations that are the same as those that are solved by `gensys`, but with the role of Lagrange multipliers taken over by unknown partial derivatives of the value function. There is still a need to match unstable roots to forward-looking first order conditions, for which something like `gensys` is needed.

I obtained numerical solutions myself for regressions in both directions for a few values of θ, using `gensys`. With unemployment on the left in the regression, the coefficients on inflation are extremely small, all through the sample. The optimal solution therefore, unsurprisingly, pushes inflation to near zero while leaving unemployment almost the same as the historical series. This is true both for $\theta = 1$. With $\theta = .1$, inflation is pushed to large negative values for most of the sample period. This is because the response of unemployment to inflation is estimated as positive, so pushing inflation down reduces unemployment. Nonetheless, because the coefficients on inflation are so small, the unemployment time series is still almost identical to the raw data.

With unemployment on the right, the regression estimates a small coefficient on current unemployment, and a much larger coefficient on the lagged change in unemployment. That is, at lags 0, 1 and 2 the coefficients are small, then large, nearly equal, and opposite-signed.

Date: October 28, 2017.

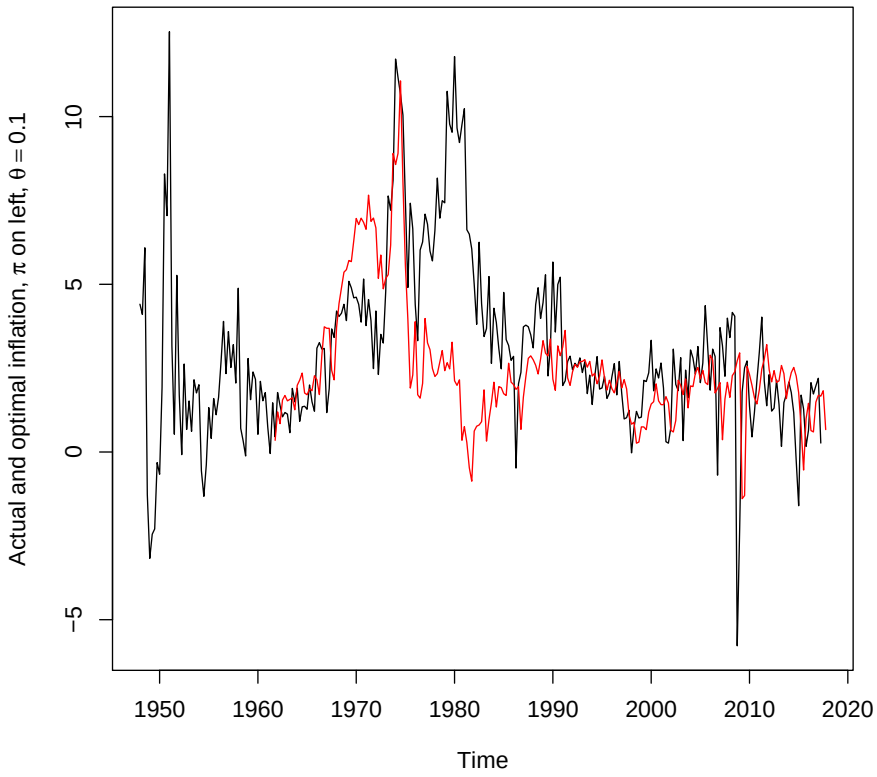
The estimated coefficient on the change in lagged unemployment is around 1.5 at the start (1960:II) and drops steadily to around .4 at the end of the sample. With $\theta = 1$, the optimal solution has unemployment above its historic value through much of the sample and inflation well below its historical values. With $\theta = .1$ the optimal solution allows inflation to rise to its first peak in the early 79's but then pushes unemployment up to high levels that bring inflation down and eliminate the 1979-80 peak.

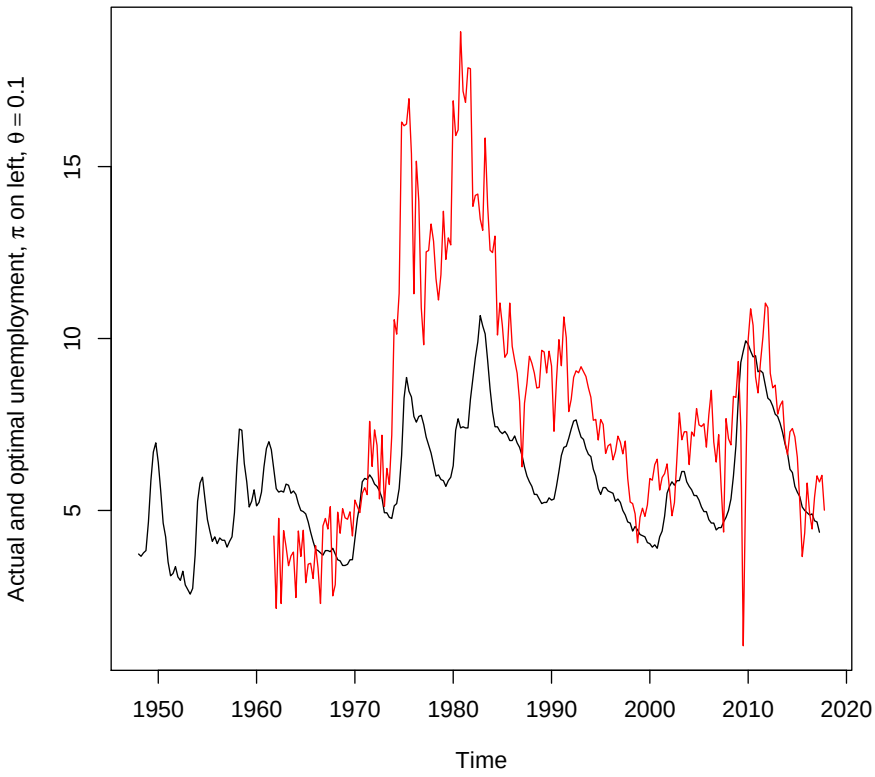
R code that produces these results is available as a separate text file, along with the two (nearly identical) equation system text files specifying the u-on-lleft and pi-on-left models. The difference between the two models is only in the normalization of the Phillips curve equation, In principle the same model could have been used for both cases, but undoing the normalization for the estimation creates a source of confusion and error, so I used separate models.

The code is also reproduced on the next page, in case you want to read it over rather than use it directly. It requires the R `gensys` package, which is available on my subversion server (svn://sims.princeton.edu/R/toolbox), and the `csolve` package, which is also available there. These packages are also available via http at sims.princeton.edu/yftp/gensys and sims.princeton.edu/yftp/csolve.

```
pceqP <- read.eqsys("pcexeqP.txt")
## pceqU <- read.eqsys("pcexeqU.txt")
# up is a time series object with columns u and dp.  u is unemployment rate in %
## and dp is inflation in % at annual rates, computed as 400*diff(log(p)), where p
## is the chained PCE deflator.  up starts in 1948, ends in 2017:II (quarterly)
cf <- matrix(0, 226, 6)
for (it in 1:(242-16)) {
upt <- window(up,end=1957 + (it - 1)/ 4 + 4)
xt <- lagts(upt, 0:2)
lsout <- lsfit(x=xt[ , c(1:3, 5:6)], y=xt[ , 4]) #regression with dp on the left
## lsout <- lsfit(x=xt[ , c(2:3, 4:6)], y=xt[ , 1]) #regression with u on the left
cf[it, ] <- lsout$coefficients
}

param <- cf[ , c(5:6, 2:4, 1)]      #these are coefficients in the order used by the
## param <- cf[ , c(2:6, 1)]              #equation system.  upper for dp lhs, lower for u lhs
##  calculating residuals for time-t data, time-(t-1) est'd coefficients
epc <- vector("numeric", 225)
epc <- xt[-(1:51),c(1:3, 5:6) ] * cf[=226, 2:6]
## epc <- xt[-(1:51),2:6] * cf[-226, 2:6]
epc <- apply(epc, 1, sum)
epc <- xt[-(1:51), 4] - epc - cf[-226, 1]
## epc <- xt[-(1:51), 1] - epc - cf[-226, 1]
## epc's start date is 1960.25, end 2017
yhat <- matrix(0, 225, 6)            #225 x 6
## pceqP or U has been read in with read.eqsys()
pcPd <- g0g1d(pceqP)                    #produces new system, with derivative function
paramx <- cbind(param[ , -6], bet=.99, thet=.1) # need to add beta and theta, remove constant
dimnames(paramx)[[2]][1:5] <- c("a1","a2","b0", "b1","b2")
for (it in 1:225) {
pcPg0g1 <- g0g1eval(pcUd, x=c(u=0,pi=0,w=0,v=0,x=0,lm=0), param=paramx[it, ])
## constructs coefficent matrices for gensys, here treating steady state as 0 (since doesn't matter
## for linear system)
gout <- with(pcPg0g1, gensys(g0, g1, c0=matrix(c(0, 0,0, cf[it,1], 0,0), 6, 1), psi=Psi, pi=Pi))
## extract constant from cf matrix for input to gensys.  Ph. curve is 4th equation.
if(!identical(gout$eu,c(1,1))) print(time(cf)[it-1]) # flag any existence or uniqueness problem
yhat[it, ] <- with(gout, G1 %*% c(xt[51  + it, c(1,4,2,5)], 0, 0) + C + impact * epc[it])
## we store the entire state vector for each period.
}
yhat <- ts(Re(yhat), start=1961.75, freq=4) # gensys leaves +0i terms that make its output complex
dimnames(yhat)[[2]] <- c("u", "dp", "ul", "dpl", "lambda", "Elambda")
```

```
ludef
w - ul
## w is lagged unemployment

lpidef
v - pil
## v is lagged inflation

elmdef*
xl - lm
##  x is E_t[lm(t+1)]

phcurve
pi - a1 * pil - a2 * vl - b0 * u - b1 * ul - b2 * wl - epc

ufoc*
-ul + b0 * lml + bet * b1 * lm +  bet^2 * b2 * x
## this assumes epc is observed when u (and pi) are set

pifoc*
-thet * pil -  lml + bet * lm * a1 + bet^2 * x * a2


vlist
u pi w v x lm

param
a1 a2 b0 b1 b2 bet thet

shock
epc
```


Actual and optimal inflation, π on left, θ = 0.1
10
5
0
−5
1950
1960
1970
1980
1990
2000
2010
2020
Time


Actual and optimal unemployment, π on left, θ = 0.1
5
10
15
1950
1960
1970
1980
1990
2000
2010
2020
Time