Econometrics with Financial Applications: Workshop Four

In this class we'll estimate a vector error correction model and then write a program with rolling windows to show the stability of the cointegrating rank\vector.

1. Download three series: *indpro* (ind. prod), *pce* (cons) and *m2real* (M2):

```
wfcreate m 1970m1 2014m9
fetch FRED::series
```

2. Again plot the three series:

```
graph plottedseries.line log(indpro) log(pce) log(m2real)
plottedseries.seetelem(1) axis(right)
```

3. Run a series of ADF tests to determine their stationarity.
4. Generate three new variables and group them:

```
group threeserieslogged lindpro lpce lm2real
```

5. To run Johansen cointegration tests, click *View→Cointegration Test→Johansen Test* from VAR or Group objects. While k is typically chosen through an IC on a VAR in first differences, try the tests for various lag lengths:

```
freeze(cointtest{!k}) threeserieslogged.coint(s,!k)
```

6. Now try the VAR in first differences as before:

```
var firstdifferences.ls 1 4 d(lindpro) d(lpce) d(lm2real)
firstdifferences.laglen(12)
```

7. Lets now consider how stable this relationship is by estimating the test as a rolling window:

```
vector(240) ranks
for !k = 1 to 240
smpl @first+!k @last-240+!k
freeze(vecmtable!k) threeserieslogged.coint(s,4)
ranks(!k) = vecmtable!k(13,4)
next
freeze(cointegratingranks) ranks.line
```

   Experiment with window size? What can we conclude? What was our hypothesis? Does money affect long run production or consumption?

8. Lets test whether money does not enter into the long run relationship:

```
var firstvecm
firstvecm.append(coint) b(1,3)=0
firstvecm.ec(c,1,restrict) 1 1 lpce lindpro lm2real
```

   First a VEC, 'firstvecm' is declared, then a restriction is appended to 'firstvecm', finally 'firstvecm' is estimated with that restriction imposed. Alternatively, this can be imposed on a standard VECM: *Proc→Specify/Estimate→VECM Restrictions→Impose Restrictions(B(1,3)=0).*

9. View the cointegrating graph (*View→Cointegrating Graph*).
10. Consider the program on the following page for how to run all of these commands in a batch environment.

Program for Workshop Four

```
wfcreate m 1959m1 2014m09
fetch fred::indpro
fetch fred::m2real
fetch fred::pce
genr lindpro=log(indpro)
genr lpce=log(pce)
genr lm2real=log(m2real)
graph loggedseries.line lindpro lpce lm2real
loggedseries.setelem(1) axis(right)
for %series lindpro lpce lm2real
    for %det const trend
        freeze({%series}_{%det}_d0) {%series}.uroot(adf,{%det},dif=0)
        freeze({%series}_{%det}_d1) {%series}.uroot(adf,{%det},dif=1)
        freeze({%series}_{%det}_d2) {%series}.uroot(adf,{%det},dif=2)
    next
    freeze({%series}_d0) {%series}.correl
    freeze({%series}_d1) d({%series},1).correl
    freeze({%series}_d2) d({%series},2).correl
next
group threeserieslogged lindpro lpce lm2real
vector(240) ranks
for !j = 1 to 240
    smpl @first+!j @last-240+!j
    var varfirstdifferences.ls 1 4 d(lpce) d(lm2real) d(lindpro)
    varfirstdifferences.laglen(12,mname=matrixlags{!j})
    !lags=matrixlags{!j}(14,5)
    freeze(vecmtable!j) threeserieslogged.coint(s,!lags)
    ranks(!j)=vecmtable!j(13,4)
next
freeze(cointegratingranks) ranks.line
smpl @all
var firstvecm
firstvecm.append(coint) b(1,3)=0
firstvecm.ec(c,1,restrict) 1 1 lpce lindpro lm2real
```