Generating the results of Table 2 involves a number of Eviews programs, displayed below.

The raw simulation data is generated by comStatics04.prg. In this program for each 0,z,h (where \( z = \frac{h}{3} \)) a separate working file is created. I actually ran this a separate batches on a number of machines. Then program consolidate04.prg merges all this simulated data into one data file (Eviews working file).

Program define1.prg defines each of the non-linear models and estimates them over all the simulated data. In some cases finding good starting values involved some tricks like first estimating a simplified version of the model. The first time it was run commands like \( c = \text{smooth1}_c \) were commented out. Thereafter the "param" commands were commented out, and comments like "\( c = \text{smooth1}_c \)" were turned back to commands (comment markers removed). Likewise any simplified versions were commented out. So then the program could be run multiple times, each of them updating the coefficient vector saved, (such as smooth1_c).

Then programs big_forecastA.prg through big_forecastD.prg estimates the models in the two subsamples, saving the resulting coefficient vectors for use in the next program.

Finally, program big_forecast.prg uses coefficients as estimated in one part of the sample to forecast in the other part. It then creates a table called "forecast1" in which forecasting and re-estimation results are saved, which are displayed in Table 2 of the paper.
program com_statics04
Eviews program com_statics04.prg

```
" program com_statics04
" note standard Miller-Orr with NORMAL net receipts. Will allow fractional Z,H
" Note this creates separate workfiles for each z, so not using sss
" Static variation in b
" Aug 11, 2006

"WFCreate(wf=com_static04) u 20
rndseed(default = "mt4")
!neach = 100000
For %week 10
!sets = 0
!bcount = 1
For %z 11.2 13.7
"3.61  3.86  4.23  4.61  4.98  5.41  5.95  6.44  7.24  8.2  9.3  11.2  13.7  20.5
WFCreate(wf=com_static04_{%z}) u 100000
smpl 1 10000
!bigstart = !sets*!neach +1 "will be recording !neach obs for every Z
!sets = !sets +1
matrix(10000,2) allM
!z = {%z}
!H = 3*!z
!PZ = 2/(!H)
!slope1 = !PZ/(!z)
!slope2 = !PZ/(2*!z )
gen set1 = !z*(4/3) "at mean if not otherwise modified
gen set2 = !z*(4/3)

!smplstart = 0
!smplend = 0
For !x = !z to 0 step -1 "initialize to just above zero
!smplstart = !smplend +1
!smplend = !smplstart +@floor( (!x)*!slope1*10000 ) -1
If !smplstart <= !smplend Then
smpl !smplstart !smplend
gen set1 = !x
gen set2 = !x
Endif
Next
"still using above !smplstart and end
For !x = !z+1 to !H "initialize through H
!smplstart = !smplend +1
!smplend = !smplstart +@floor( (!PZ -(!x-!z)*!slope2)*10000 ) -1
If !smplstart <= !smplend Then
smpl !smplstart !smplend
gen set1 = !x
gen set2 = !x
Endif
Next

smpl 1 10000
" now begin to shuffle, pair, assign net, record number of adjustments and adjust to targets, sum
" will want to see if number of adjustments correlated with sum being high or low.
For !ini = 1 to 1000 "simulate a bunch of rounds to initialize
stom(set1,vec1)
stom(set2,vec2) "converted series to vectors
colplace(allM,vec1,1) "place vectors in a matrix
colplace(allM,vec2,2)
```
program com_statics04
vector allV = @vec(allM) ""collapse to one big vector 20,000 elements
allV = @permute(allV) ""resort randomly
vec1 = @subextract(allV,1,1,10000,1) ""place first half of bigV back into vec1
vec2 = @subextract(allV,10001,1,20000,1) ""place second half of bigV back into vec2
mtos(vec1,set1) ""now put randomized vector back into series
mtos(vec2,set2)

gener trans = nrnd ""series for random net receipts
gener set1 = set1 +trans
gener set2 = set2 -trans ""assign net receipts

smpl if set1 <= 0 ""reallocate if breach interval
gener set1 = !z +set1 ""without this get too much piled at Z
smpl 1 10000
smpl if set1 <= 0 ""reallocate if breach interval
gener set1 = !z +set1 ""without this get too much piled at Z
smpl 1 10000
smpl if set1 >=!H
gener set1 = !z +(set1 -!H) ""without this get too much piled at Z
smpl 1 10000
smpl if set1 >=!H
gener set1 = !z +(set1 -!H) ""without this get too much piled at Z
smpl 1 10000

smpl if set2 <=0 ""reallocate if breach interval
gener set2 = !z +set2
smpl 1 10000
smpl if set2 <=0 ""reallocate if breach interval ""doing it twice
gener set2 = !z +set2
smpl 1 10000
smpl if set2 >=!H
gener set2 = !z +(set2 -!H)
smpl 1 10000
smpl if set2 >=!H
gener set2 = !z +(set2 -!H)
smpl 1 10000
Next ""For ini = 1 to 1000 ""simulate a bunch of rounds to initialize

Next ""For ini = 1 to 1000 ""simulate several rounds to initialize

Now record data, every week'th round

smpl 1 10000
gener portrigger = na ""for recording portion of all adjusted
gener portrigger0 = na ""for recording portion of lower breaches
gener portriggerH = na ""for recording portion of upper breachers!
!trigger0 = 0 ""need to initialize
!triggerH = 0 ""need to initialize
!record = 1
For !round = 1 to !neach
!trigger = 0
For !week = 1 to {%week} ""recording holdings only every {}th period, recording sum of transfers over the week
stom(set1,vec1)
stom(set2,vec2) ""converted series to vector
colplace(allM,vec1,1) ""place vectors in a matrix
colplace(allM,vec2,2)
vector allV = @vec(allM) ""collapse to one big vector 20,000 elements
allV = @permute(allV) ""resort randomly
vec1 = @subextract(allV,1,1,10000,1) ""place first half of bigV back into vec1
vec2 = @subextract(allV,10001,1,20000,1) ""place second half of bigV back into vec2

gener trans = nrnd
gener set1 = set1 +trans
gener set2 = set2 -trans ""assign net receipts
program com_statics04

smpl if set1 <= 0 "allocate if breach interval
!trigger0 = !trigger0 + @obs(set1)
gener set1 = !z + set1
smpl 1 10000
smpl if set1 <= 0 "allocate if breach interval 2nd time
!trigger0 = !trigger0 + @obs(set1)
gener set1 = !z + set1
smpl 1 10000

smpl if set1 >= !H
!triggerH = !triggerH + @obs(set1)
gener set1 = !z + (set1 - !H)
smpl 1 10000
smpl if set1 >= !H "2nd time
!triggerH = !triggerH + @obs(set1)
gener set1 = !z + (set1 - !H)
smpl 1 10000

smpl if set2 <= 0 "allocate if breach interval
!trigger0 = !trigger0 + @obs(set2)
gener set2 = !z + set2
smpl 1 10000
smpl if set2 <= 0 "allocate if breach interval 2nd time
!trigger0 = !trigger0 + @obs(set2)
gener set2 = !z + set2
smpl 1 10000

smpl if set2 >= !H
!triggerH = !triggerH + @obs(set2)
gener set2 = !z + (set2 - !H)
smpl 1 10000
smpl if set2 >= !H "2nd time
!triggerH = !triggerH + @obs(set2)
gener set2 = !z + (set2 - !H)
smpl 1 10000

!sumn = 0
Next "For !week = 1 to
"!sss = !bigstart + !round - 1 only use if making one big workfile
"smpl !sss !sss
smpl !round !round
gener porTrigger = (!trigger0 + !triggerH)/20000
!trigger0 = !trigger0/20000
!triggerH = !triggerH/20000
gener Z = %Z
!trigger0 = 0
!triggerH = 0
smpl 1 10000
!sumn = @sum(set1) + @sum(set2)
"smpl !sss !sss
smpl !round !round
gener total = !sumn
smpl 1 10000
Next "For !round = 1 to

"smpl !bigstart !bigstart + !neach - 1 "this was for one big file
smpl 1 10000
ls total = c(1)
gener av = c(1)
!av = c(1)
"is total = c(2)*total(-1) + (1 - c(2))*av
!b2 = c(2)
smpl 1 10000
WFsave com_static04_{%z}.wf1
program com_statics04
close com_static04_{%z}.wf1
""WFselect com_static04
""smpl !bcount !bcount
""genr av = !av
""genr bM = !b2
""genr z = %z
!bcount = !bcount +1
Next""For %z
Next ""For %week 10
""WFsave com_static04
""smpl 1 14
""param 1 -10 2 0
""ls bm = exp(c(1)/(c(2)+z^2))
program consolidate04
Eviews program consolidate04.prg

"program consolidate04
"Sept 5, 2006
"consolidates individual data from programs comStatics04
"but uses only 10,001 obs from each file (each z)
!file = 1
For %z 3.61 3.86 4.23 4.61 4.98 5.41 5.95 6.4 4 7.24 8.20 9.30 11.2 13.7 20.5
WFopen com_static04_{%z}.wf1
smpl 2 10001
genr m_{!file} = total
genr mlag_{!file} = total(-1)
genr av_{!file} = av
genr z_{!file} = z
store(i) m_{!file} mlag_{!file} av_{!file} z_{!file}
close com_static04_{%z}.wf1
!file = !file +1
Next
"above stores obs of individual series

WFCreate(wf=com_static04_c) u 140030
!start = 0
!round = 1
For !file = 1 to 14
fetch m_{!file} mlag_{!file} av_{!file} z_{!file} "these are complete for smpl 2 10001
!sstart = (!start*10000) +1
!eend = !sstart +9999
!lag = ((!round-1)*10000) -1
smpl !sstart !eend
genr M = m_{!file}(-!lag)
genr M_lag = mlag_{!file}(-!lag)
genr av = av_{!file}(-!lag)
genr z = z_{!file}(-!lag)
!start = !start +1
!round = !round +1
Next
**Program define1.prg**

**Eviews program define1.prg**

```
smpl 1 140000

**c(1) = -1
**c = MBW_c
```

```
equation MBW.ls(m=1000,c=0.0000001,s) m = (EXP((10^10)*C(1)/(AV^2)))*M_LAG + (1 - (EXP((10^10)*C(1)/(AV^2))))*AV
```

```
**c = MBW_c
```

```
equation smooth1.ls(m=100,c=0.0000001,s) m = (C(1) + C(2)*(EXP((-C(3))^2)*)((10^-5)*(M_LAG - AV - 0)*(10^5) )^2 )) )/AV
```

```
**c = smooth1_c
```

```
equation smooth1r.ls(m=100,c=0.0000001,s) m = (C(1) + C(2)*(EXP((-C(3))^2)*)((10^-5)*(AV - M_LAG) - 0*(10^5) )^2 )) )/AV
```

```
**c = smooth1r_c
```

```
equation smooth2.ls(m=100,c=0.0000001,s) m = (C(1) + C(2)*(EXP((-C(3))^2)*)((10^-5)*(M_LAG - AV - 0)*(10^5) )^2 )) )/AV
```

```
**c = smooth2_c
```

```
equation smooth2r.ls(m=100,c=0.0000001,s) m = (C(1) + C(2)*(EXP((-C(3))^2)*)((10^-5)*(AV - M_LAG) - 0*(10^5) )^2 )) )/AV
```

```
**c = smooth2r_c
```

```
equation smooth3.ls(m=100,c=0.0000001,s) m = (C(1) + C(2)*(EXP((-C(3))^2)*)((10^-5)*(AV - M_LAG) + 0*(10^5) )^2 )) )/AV
```

```
**c = smooth3_c
```

```
equation smooth3r.ls(m=100,c=0.0000001,s) m = (C(1) + C(2)*(EXP((-C(3))^2)*)((10^-5)*(AV - M_LAG) + 0*(10^5) )^2 )) )/AV
```

```
**c = smooth3r_c
```

---

```
param 1 0 1 3 0.001
```

```
'ls m = (C(1) + 1*(EXP((-C(3))^2)*)((10^-5)*(M_LAG - AV - 0)*(10^5) )^2 )) )/M_LAG + (1 - (C(1) + 1*(EXP((-C(3))^2)*)((10^-5)*(M_LAG - AV - 0)*(10^5) )^2 )) )/AV
```

```
'param 2 1 4 0
```

```
'c = smooth1_c
```

```
equation smooth1.ls(m=100,c=0.0000001,s) m = (C(1) + C(2)*(EXP((-C(3))^2)*)((10^-5)*(AV - M_LAG) - C(4)*(10^5) )^2 )) )/AV
```

```
'param 1 0 2 0.01 3 0.01 4 0
```

```
'ls m = (0 + C(2)*EXP((-C(3))^2)*)((10^-5)*(M_LAG - AV - 0)*(10^5) )^2 )) )/M_LAG + (1 - (0 + C(2)*EXP((-C(3))^2)*)((10^-5)*(M_LAG - AV - 0)*(10^5) )^2 )) )/AV
```

```
'smooth1r.ls(m=2000,c=0.0000001,s)
```

```
equation smooth1r.ls(m=1000,c=0.0000001,s) m = (C(1) + C(2)*(EXP((-C(3))^2)*)((10^-5)*(AV - M_LAG) - C(4)*(10^5) )^2 )) )/AV
```

```
store(i) smooth1r
```

```
store(i) smooth1r_c
```

```
equation smooth2r.ls(m=1000,c=0.0000001,s) m = (C(1) + C(2)*(EXP((-C(3))^2)*)((10^-5)*(AV - M_LAG) + c(4)*(10^5) )^2 )) )/AV
```

```
equation smooth2r_c = c
```

```
store(i) smooth2r
```

```
store(i) smooth2r_c
```

---

**Define smooth-logistic**

```
param 1 2 0.01 3 0.01 4 0
```

```
l s(m=200,c=0.0000001,s) m = (1 - C(2)/(1 + EXP(-C(3))^2)*)((10^-5)*(M_LAG - AV + 0)) )/M_LAG + (1 - (1 - C(2)/(1 + EXP(-C(3))^2)*)((10^-5)*(M_LAG - AV + 0)) )/AV
```

```
**c = smooth2_c
```

```
equation smooth2r.ls(m=1000,c=0.0000001,s) m = (C(1) - C(2)/(1 + EXP(-C(3))^2)*)((10^-5)*(AV - M_LAG) + c(4)) )/M_LAG + (1 - (c(1) - C(2)/(1 + EXP(-C(3))^2)*)((10^-5)*(M_LAG - AV + c(4)) )/AV
```

```
equation smooth2r_c = c
```

```
store(i) smooth2r
```

```
store(i) smooth2r_c
```

---

**Define smooth-logistic**

```
param 1 2 0.01 3 0.01 4 0
```

```
l s(m=200,c=0.0000001,s) m = (1 - (c(2))^2)/(1 + EXP(-(c(3))^2))*)((10^-5)*(M_LAG - AV + 0)) )/M_LAG + (1 - (1 - (c(2))^2)/(1 + EXP(-(c(3))^2))*)((10^-5)*(M_LAG - AV + 0)) )/AV
```

```
**c = smooth2_c
```

```
equation smooth2r.ls(m=1000,c=0.0000001,s) m = (c(1) - (c(2)^2)/(1 + EXP(-(c(3))^2))*)((10^-5)*(M_LAG - AV + 0)) )/M_LAG + (1 - (1 - (c(2)^2)/(1 + EXP(-(c(3))^2))*)((10^-5)*(M_LAG - AV + 0)) )/AV
```

```
**c = smooth2r_c
```

---
program define1.prg
Equation smooth2r.ls(m=1000,c=0.0000001,s) m = \( \frac{C(1) - ((c(2)^2))/(1 + \exp(-((c(3)^2)^2)*10^{-5})*(M_LAG - AV + C(4)))}{M_LAG + (1 - ((C(1) - ((c(2)^2))/(1 + \exp(-((c(3)^2)^2)*10^{-5})*(M_LAG - AV + C(4)))))*AV} \)

coeff(751) smooth2r_c = c
store(i) smooth2r
store(i) smooth2r_c
programs big_forecastA.pr, big_forecastB, big_forecastC, big_forecastD
Eviews programs big_forecastA-D:

```eviews

"""program big_forecastA
"""workfile com_static04_c c is for consolidated
"""uses first half of smpl to forecast in second half
"""BREAKS UP AND SAVES STARTING COEFS FOR SMOOTH1
"""March 23, 2007

smpl 1 70000
c = smooth1_c
smooth1.ls(m=2000,c=0.000001,s)
coef(751) smooth1F_c = c """"F for first
smpl 70001 140000
c = smooth1_c
smooth1.ls(m=2000,c=0.000001,s)
coef(751) smooth1S_c = c """"S for second
store(i) smooth1F_c
store(i) smooth1S_c

---------------------------

"""program big_forecastB
"""workfile com_static04_c c is for consolidated
"""uses first half of smpl to forecast in second half
"""BREAKS UP AND SAVES STARTING COEFS FOR SMOOTH1R
"""March 23, 2007

smpl 1 70000
c = smooth1r_c
smooth1r.ls(m=2000,c=0.000001,s)
coef(751) smooth1rF_c = c """"F for first
smpl 70001 140000
c = smooth1r_c
smooth1r.ls(m=2000,c=0.000001,s)
coef(751) smooth1rS_c = c """"S for second
store(i) smooth1rF_c
store(i) smooth1rS_c

---------------------------------

"""program big_forecastC
"""workfile com_static04_c c is for consolidated
"""uses first half of smpl to forecast in second half
"""BREAKS UP AND SAVES STARTING COEFS FOR SMOOTH2
"""March 23, 2007

smpl 1 70000
c = smooth2_c
smooth2.ls(m=2000,c=0.000001,s)
coef(751) smooth2F_c = c """"F for first
smpl 70001 140000
c = smooth2_c
smooth2.ls(m=2000,c=0.000001,s)
coef(751) smooth2S_c = c """"S for second
store(i) smooth2F_c
store(i) smooth2S_c
```

programs big_forecastA.prg, big_forecastB, big_forecastC, big_forecastD

---------
"program big_forecastD
"workfile com_static04_c  c is for consolidated
"uses first half of smpl to forecast in second half
"BREAKS UP AND SAVES STARTING COEFS FOR smooth2r
"March 23, 2007

smpl 1 70000
  c = smooth2r_c
  smooth2r.ls(m=2000,c=0.0000001,s)
  coef(751) smooth2rF_c = c  ""F for first
smpl 70001 140000
  c = smooth2r_c
  smooth2r.ls(m=2000,c=0.0000001,s)
  coef(751) smooth2rS_c = c  ""S for second

store(i) smooth2rF_c
store(i) smooth2rS_c
```
program big_forecast
Eviews program big_forecast.prg

''''program big_forecast
''''workfile com_static04_c  c is for consolidated
''''FIRST RUN PROGRAM DEFINE1
''''THEN RAN PROGRAMS big_forecastA, big_forecastB, ...C ...D which saved near convergence starting values
''which were fetched into the working file.
''uses first half of smpl to forecast in second half
'March 23, 2007

smpl 1 70000
equation Lin.ls m = c(1)*m_lag +c(2)*av
smpl 70001 140000
genr mhat_Lin = c(1)*m_lag +c(2)*av
Lin.LS
!ser_Lin = Lin.@se
smpl 1 70000
c(1) = -1
equation MBW.ls(m=2000,c=0.0000001,s) m = (EXP((10^10)*C(1)/(AV^2)))*M_LAG + (1 -(EXP((10^10)*C(1)/(AV^2))))*AV
smpl 70001 140000
genr mhat_MBW = (EXP((10^10)*C(1)/(AV^2)))*M_LAG + (1 -(EXP((10^10)*C(1)/(AV^2))))*AV

c(1) =-1
MBW.ls(m=2000,c=0.0000001,s)
!ser_MBW = MBW.@se

smpl 1 70000
c = smooth1F_c
smooth1.ls(m=20,c=0.0000001,s)
smpl 70001 140000
genr mhat_smooth1 = (C(1) +C(2)*EXP((-C(3))*(((10^-5)*(AV -M_LAG) -C(4)*(10^5) )^2) ))) )*M_LAG + (1 - (C(1) +C(2)*EXP((-C(3))*(((10^-5)*(AV -M_LAG) -C(4)*(10^5) )^2) ))) )*AV

c = smooth1S_c
smooth1.ls(m=20,c=0.0000001,s)
!ser_smooth1 = smooth1.@se

smpl 1 70000
c = smooth1rF_c
smooth1r.ls(m=20,c=0.0000001,s)
smpl 70001 140000
genr mhat_smooth1r = (C(1) +(C(2)^2)*(EXP((-(C(3)^2)*(((10^-5)*(AV -M_LAG) -C(4)*(10^5) )^2) ))) )*M_LAG + (1- (C(1) +(C(2)^2)*(EXP((-(C(3)^2)*(((10^-5)*(AV -M_LAG) -C(4)*(10^5) )^2) ))) ))) )*AV

c = smooth1rS_c
smooth1r.ls(m=20,c=0.0000001,s)
!ser_smooth1r = smooth1r.@se

smpl 1 70000
c = smooth2F_c
smooth2.ls(m=20,c=0.0000001,s)
smpl 70001 140000
genr mhat_smooth2 = ( ( c(1) -(C(2))/((1 +EXP( -(C(3))*((10^-5)*(M_LAG -AV +c(4))) ))) )*M_LAG + (1 -(c(1) -(C(2))/((1 +EXP( -(C(3))*((10^-5)*(M_LAG -AV +c(4))) ))) ))) )*AV

c = smooth2S_c
smooth2.ls(m=20,c=0.0000001,s)
!ser_smooth2 = smooth2.@se

smpl 1 70000
c = smooth2rF_c
smooth2r.ls(m=20,c=0.0000001,s)
smpl 70001 140000
```
program big_forecast
gen mhat_smooth2r = ( C(1) -((c(2)^2))/(1 +EXP( -((c(3)^2))*(10^-5)*(M_LAG -AV +C(4))) ))*M_LAG  + (1 -( C(1) -((c(2)^2))/(1 +EXP( -((c(3)^2))*(10^-5)*(M_LAG -AV +C(4))) ))) )*AV
c = smooth2r@c_c
smooth2r.ls(m=20,c=0.0000001,s)
!ser_smooth2r = smooth2r.@se

table(7,3) forecast1
forecast1(1,1) = "model"
forecast1(1,2) = "rmse2ndhalf"
forecast1(1,3) = "SER 2ndhalf"

forecast1(2,1) = "Linear"
forecast1(3,1) = "Smooth1"
forecast1(4,1) = "Smooth1r"
forecast1(5,1) = "Smooth2"
forecast1(6,1) = "Smooth2r"
forecast1(7,1) = "Exp01"

smpl 70001 140000
gen err = m -mhat_Lin
gen err2 = err^2
!merr2 = @sum(err2)/@obs(err2)
!rmse = !merr2^0.5
forecast1(2,2) = !rmse
d err err2

gen err = m -mhat_smooth1
gen err2 = err^2
!merr2 = @sum(err2)/@obs(err2)
!rmse = !merr2^0.5
forecast1(3,2) = !rmse
d err err2

gen err = m -mhat_smooth1r
gen err2 = err^2
!merr2 = @sum(err2)/@obs(err2)
!rmse = !merr2^0.5
forecast1(4,2) = !rmse
d err err2

gen err = m -mhat_smooth2
/gen err2 = err^2
!merr2 = @sum(err2)/@obs(err2)
!rmse = !merr2^0.5
forecast1(5,2) = !rmse
d err err2

gen err = m -mhat_smooth2r
/gen err2 = err^2
!merr2 = @sum(err2)/@obs(err2)
!rmse = !merr2^0.5
forecast1(6,2) = !rmse
d err err2

gen err = m -mhat_smooth2r
/gen err2 = err^2
!merr2 = @sum(err2)/@obs(err2)
!rmse = !merr2^0.5
forecast1(6,3) = !ser_smooth2r
d err err2
program big_forecast

genr err = m - mhat_MBW
genr err2 = err^2
!merr2 = @sum(err2)/@obs(err2)
!rmse = !merr2^0.5
forecast1(7,2) = !rmse
forecast1(7,3) = !ser_MBW
d err err2