14.170: Programming for Economists

1/12/2009-1/16/2009

Melissa Dell
Matt Notowidigdo
Paul Schrimpf

Lecture 4, Introduction to Mata in Stata

Mata in Sata

- Mata is a matrix programming language that is now built into Stata. The syntax is a cross between Matlab and Stata.
- Mata is not (yet) seamlessly integrated into Stata; for more complicated projects it still might be better to export to Matlab and write Matlab code
- Examples of when to use Mata (rather than Stata or Matlab):
  - Add robust standard errors to existing Stata estimator that does not currently support it
  - Simple GMM estimator
  - Simple ML estimator (or any estimator) that would be easier to implement using matrix notation

But first ... back to Stata ML

Normal Mixture in Stata ML

```
set obs 10000
set seed 14170
local lambda = 0.25
local sigma_1 = 1
local sigma_2 = 2
local mu_1 = 1
local mu_2 = 0.5
gen type = (uniform() < `lambda')
gen v = (`mu_1' + `sigma_1'*invnorm(uniform()))     if type == 1
replace v = (`mu_2' + `sigma_2'*invnorm(uniform())) if type == 0

program define mixture_d0
 args todo b lnf
 tempvar lnf_j
 tempname lambda sigma_1 sigma_2 mu_1 mu_2
 scalar `mu_1' = `b'[1,1]
 scalar `mu_2' = `b'[1,2]
 scalar `sigma_1' = exp(`b'[1,3])
 scalar `sigma_2' = exp(`b'[1,4])
 scalar `lambda'  = normal(`b'[1,5])
 gen double `lnf_j' = ///
  `lambda' * (1/`sigma_1') * normalden(($ML_y1 - `mu_1')/`sigma_1') + ///
  (1-`lambda') * (1/`sigma_2') * normalden(($ML_y1 - `mu_2')/`sigma_2')
 mlsum `lnf' = log(`lnf_j')
end
gen mu_1 = 1
ml model d0 mixture_d0 (v = mu_1, noconstant) ///
   /mu_2 /ln_sigma_1 /ln_sigma_2 /inv_lambda
ml maximize
nlcom exp([ln_sigma_1]_b[_cons])
nlcom exp([ln_sigma_2]_b[_cons])
nlcom normal([inv_lambda]_b[_cons])
```

$$f(y) = \lambda(1/\sigma_1)\phi((y-\mu_1)/\sigma_1) + (1-\lambda)(1/\sigma_2)\phi((y-\mu_2)/\sigma_2)$$

Normal Mixture in Stata ML

```
. ml maximize

initial:       log likelihood = -27347.815
alternative:   log likelihood = -20146.623
rescale:       log likelihood = -20146.623
rescale eq:    log likelihood = -20093.115
Iteration 0:   log likelihood = -20093.115  (not concave)
Iteration 1:   log likelihood = -20037.116  (not concave)
Iteration 2:   log likelihood = -20005.921  (not concave)
Iteration 3:   log likelihood = -19999.319  (not concave)
Iteration 4:   log likelihood = -19995.661
Iteration 5:   log likelihood = -19990.202
Iteration 6:   log likelihood = -19982.384
Iteration 7:   log likelihood = -19980.125
Iteration 8:   log likelihood =  -19979.87
Iteration 9:   log likelihood = -19979.867
Iteration 10:  log likelihood = -19979.867

                                                  Number of obs   =      10000
                                                  Wald chi2(1)    =     246.71
Log likelihood = -19979.867                       Prob > chi2     =     0.0000

------------------------------------------------------------------------------
           v |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
eq1          |
        mu_1 |   .9467025    .060272    15.71   0.000     .8285715    1.064834
-------------+----------------------------------------------------------------
mu_2         |
       _cons |   .5414998   .0327479    16.54   0.000      .477315    .6056845
             .
ln_sigma_1   |
       _cons |   .0431566    .078409     0.55   0.582    -.1105223    .1968354
-------------+----------------------------------------------------------------
ln_sigma_2   |
       _cons |   .6789867   .0171358    39.62   0.000     .6454011    .7125723
-------------+----------------------------------------------------------------
inv_lambda   |
       _cons |  -.6729743   .1416098    -4.75   0.000    -.9505244   -.3954241
------------------------------------------------------------------------------
```

Normal Mixture in Stata ML

```
. nlcom exp([ln_sigma_1]_b[_cons])

       _nl_1:  exp([ln_sigma_1]_b[_cons])

------------------------------------------------------------------------------
           v |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
       _nl_1 |   1.044101   .0818669    12.75   0.000     .8836451    1.204558
------------------------------------------------------------------------------

. nlcom exp([ln_sigma_2]_b[_cons])

       _nl_1:  exp([ln_sigma_2]_b[_cons])

------------------------------------------------------------------------------
           v |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
       _nl_1 |   1.971879   .0337898    58.36   0.000     1.905652    2.038105
------------------------------------------------------------------------------

. nlcom normal([inv_lambda]_b[_cons])

       _nl_1:  normal([inv_lambda]_b[_cons])

------------------------------------------------------------------------------
           v |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
       _nl_1 |   .2504818   .0450463     5.56   0.000     .1621928    .3387709
------------------------------------------------------------------------------
```

GMM in Stata ML

- In principle, Stata ML can be used to implement any estimator based on maximization of an objective function.
- Thus we can use Stata ML to implement NLLS or GMM estimators
  - BENEFIT: Simple to code; can re-use well-known Stata syntax and helper functions
    - Particularly useful for panel data estimators (egen, bysort, etc.)
  - COST: Mata is better if moment conditions are basd on matrix algebra

GMM-OLS

$$g(\beta) = \mathrm{E}[X'\varepsilon] = 0$$

$$\beta_{GMM} = \arg\min_{\beta} g(\beta)' W g(\beta)$$

$$\hat{\varepsilon}_i = y_i - X_i \beta$$

$$\hat{g}(\beta) = \frac{1}{N} \sum_{i=1}^{N} X_i' \hat{\varepsilon}_i$$

$$\hat{W} = I$$

$$\hat{\beta}_{GMM} = \arg\min_{\beta} \left(\frac{1}{N} \sum_{i=1}^{N} X_i' \hat{\varepsilon}_i \right)' \left(\frac{1}{N} \sum_{i=1}^{N} X_i' \hat{\varepsilon}_i \right)$$

GMM-OLS = OLS

$$\hat{\beta}_{GMM-OLS} = \arg\min_{\beta} \frac{1}{N^2} \left(X'(y - X\beta) \right)' \left(X'(y - X\beta) \right)$$

$$= \arg\min_{\beta} \frac{1}{N^2} \left(X'y - X'X\beta \right)' \left(X'y - X'X\beta \right)$$

$$0 = \left(X'y - X'X\beta \right)' (-X'X) + \left(X'y - X'X\beta \right)' (-X'X)$$

$$0 = X'y - X'X\beta$$

$$\hat{\beta}_{GMM-OLS} = (X'X)^{-1} X'y$$

GMM in Stata ML

```
program drop _all
program define mygmm
 args todo b lnf
 tempvar xb e sum
 mleval `xb' = `b', eq(1)
 gen `e' = $ML_y1 - `xb'
 matrix vecaccum Xe = `e' $xlist
 matrix m = Xe' / _N
 matrix obj = m' * m
 mlsum `lnf' = -1 * obj[1,1] if _n == 1
end

clear
set obs 100
set seed 14170
gen x1 = invnorm(uniform())
gen y = 1 + x1 + invnorm(uniform())
global xlist = "x1"
reg y x1
ml model d0 mygmm (y = x1)
ml maximize
```

GMM in Stata ML

```
. reg y x1

      Source |       SS       df       MS              Number of obs =     100
-------------+------------------------------           F(  1,    98) =  169.67
       Model |  144.560432     1  144.560432           Prob > F      =  0.0000
    Residual |  83.495656     98  .851996489           R-squared     =  0.6339
-------------+------------------------------           Adj R-squared =  0.6301
       Total |  228.056088    99  2.30359685           Root MSE      =  .92304

------------------------------------------------------------------------------
           y |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
          x1 |   1.098961   .0843677    13.03   0.000     .9315356    1.266386
       _cons |    .975923   .0927796    10.52   0.000     .7918049    1.160041
------------------------------------------------------------------------------

. ml model d0 mygmm (y = x1)

. ml maximize

initial:       log likelihood =  -3.272676
alternative:   log likelihood = -2.2676663
rescale:       log likelihood = -1.7688444
Iteration 0:   log likelihood = -1.7688444
Iteration 1:   log likelihood = -8.070e-17
Iteration 2:   log likelihood = -2.064e-32

                                                  Number of obs   =        100
                                                  Wald chi2(1)    =       3.42
Log likelihood = -2.064e-32                       Prob > chi2     =     0.0645

------------------------------------------------------------------------------
           y |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
          x1 |   1.098961   .5943889     1.85   0.064    -.0660201    2.263941
       _cons |    .975923   .7174338     1.36   0.174    -.4302215    2.382067
------------------------------------------------------------------------------
```

GMM-OLS standard errors

$$G' = \frac{\partial g(\beta)}{\partial \beta'}$$

$$\Psi = E[mm']$$

$$V_{GMM} = \frac{1}{N}\left(G'G\right)^{-1} G'\Psi G\left(G'G\right)^{-1}$$

$$\hat{\varepsilon}_i = y_i - X_i \beta$$

$$\hat{g}(\beta) = \frac{1}{N}\sum_{i=1}^{N} X_i'(y_i - X_i \beta)$$

$$\frac{\partial \hat{g}(\beta)}{\partial \beta'} = -\frac{X'X}{N}$$

$$\Psi = E[(X'\varepsilon)(X'\varepsilon)'] = E[\varepsilon^2 X'X]$$

$$\hat{\Psi} = \hat{\sigma}_{\varepsilon}^2 \frac{X'X}{N}$$

$$\hat{V}_{GMM} = \hat{\sigma}_{\varepsilon}^2 (X'X)^{-1}$$

Mata in Sata

- How to learn more about Mata? Type the following into Stata:
  - help [M-0] intro
  - help [M-4] intro
    - help [M-4] manipulation
    - help [M-4] matrix
    - help [M-4] scalar
    - help [M-4] statistical
    - help [M-4] string
    - help [M-4] io
    - help [M-4] stata
    - help [M-4] programming

OLS in Mata

```
clear
set obs 200
set seed 1234
set more off
gen x = invnorm(uniform())
gen y = 1 + 2 * x + 0.1*invnorm(uniform())

** enter Mata
mata

x = st_data(., ("x"))
cons = J(rows(x), 1, 1)
X = (x, cons)
y = st_data(., ("y"))
X
beta_hat = (invsym(X'*X))*(X'*y)
e_hat = y - X * beta_hat
s2 = (1 / (rows(X) - cols(X))) * (e_hat' * e_hat)

V_ols = s2 * invsym(X'*X)
se_ols = sqrt(diagonal(V_ols))
beta_hat
se_ols

/** leave mata **/
end
regress y x
```

OLS in Mata

```
: beta_hat
                  1
    +----------------+
  1 |  2.010107289   |
  2 |  .9869115353   |
    +----------------+

: se_ols
                  1
    +----------------+
  1 |  .0068861654   |
  2 |  .0071979239   |
    +----------------+

:
: /** leave mata **/
: end
--------------------------------------------------------------------------------

. regress y x

      Source |       SS       df       MS              Number of obs =     200
-------------+------------------------------           F(  1,   198) =85208.63
       Model |  882.641221     1  882.641221           Prob > F      =  0.0000
    Residual |  2.05100074   198  .01035859           R-squared     =  0.9977
-------------+------------------------------           Adj R-squared =  0.9977
       Total |  884.692222   199  4.44568956           Root MSE      =  .10178

------------------------------------------------------------------------------
           y |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
           x |   2.010107   .0068862   291.91   0.000     1.996528    2.023687
       _cons |   .9869115   .0071979   137.11   0.000     .9727171    1.001106
------------------------------------------------------------------------------
```

“robust” OLS in Mata

```
clear
set obs 200
set seed 1234
set more off
gen x = invnorm(uniform())
gen y = 1 + 2 * x + x * x * invnorm(uniform())

mata
x_vars = st_data(., ("x"))
cons = J(rows(x_vars), 1, 1)
X = (x_vars , cons)
y = st_data(., ("y"))
X
beta_hat = (invsym(X'*X))*(X'*y)
e_hat = y - X * beta_hat
sandwich_mid = J(cols(X), cols(X), 0)
n = rows(X)
for (i=1; i<=n; i++) {
  sandwich_mid =sandwich_mid+(e_hat[i,1]*X[i,.])'*(e_hat[i,1]*X[i,.])
}
V_robust = (n/(n-cols(X)))*invsym(X'*X)*sandwich_mid*invsym(X'*X)
se_robust = sqrt(diagonal(V_robust))
beta_hat
se_robust
end
reg y x, robust
```

$$V_{robust} = \frac{N}{N-K} * (X' * X)^{-1} * \left(\sum_{i=1}^{N} (\hat{\varepsilon}_i * x_i)' * (\hat{\varepsilon}_i * x_i) \right) * (X' * X)^{-1}$$

“robust” OLS in Mata

```
: beta_hat
                  1
    +----------------+
  1 |   2.233123557  |
  2 |   .8948731562  |
    +----------------+

: se_robust
                  1
    +----------------+
  1 |   .2186835826  |
  2 |   .1145941403  |
    +----------------+

: end
------------------------------------------------------------------------------

. reg y x, robust

Linear regression                                      Number of obs =     200
                                                       F(  1,   198) =  104.28
                                                       Prob > F      =  0.0000
                                                       R-squared     =  0.6806
                                                       Root MSE      =  1.6068

------------------------------------------------------------------------------
             |               Robust
           y |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
           x |   2.233124   .2186836    10.21   0.000     1.801876    2.664371
       _cons |   .8948732   .1145941     7.81   0.000     .6688915    1.120855
------------------------------------------------------------------------------
```

Fixed Effects OLS (LSDV)

$$y = X\beta + \varepsilon$$

$$P_w = I_N \otimes i_T (i_T' i_I)^{-1} i_T'$$

$$M_w = I_{N \times T} - P_w$$

$$M_w y = M_w X\beta + M_w \varepsilon$$

$$\begin{aligned}
\beta_{FE} &= ((M_w X)' M_w X)^{-1} ((M_w X)' M_w y) \\
&= (X' M_w' M_w X)^{-1} (X' M_w' M_w y) \\
&= (X' M_w M_w X)^{-1} (X' M_w M_w y) \\
&= (X' M_w X)^{-1} (X' M_w y)
\end{aligned}$$

OLS FE in Mata

```
clear
set obs 100
local N = 10
gen id = 1+floor((_n - 1)/10)
bys id: gen fe = 5*invnorm(uniform())
by id: replace fe = fe[1]
gen x = invnorm(uniform())
gen y = 1.2 * x + fe + invnorm(uniform())

mata
X = st_data(., ("x"))
y = st_data(., ("y"))
I_N = I(`N')
I_NT = I(rows(X))
i_T = J(`N',1,1)
P w = I N # (i T*invsym(i T'*i T)*i T')
M_w = I_NT - P_w
beta = invsym(X'*M_w*X)*(X'*M_w*y)
e_hat = M_w*y - (M_w*X)*beta
s2 = (1 / (rows(X) - cols(X) - `N')) * (e_hat' * e_hat)
V = s2 * invsym(X'*M_w*X)
se = sqrt(diagonal(V))
beta
se
end
reg y x
areg y x, absorb(id)
```

OLS FE in Mata

```
: beta
  1.286490436

: se
  .1018454395

: end
-------------------------------------------------------------------------------

. reg y x

      Source |       SS       df       MS              Number of obs =     100
-------------+------------------------------           F(  1,    98) =   14.24
       Model |    235.9108     1    235.9108           Prob > F      =  0.0003
    Residual |  1624.07248    98  16.5721681           R-squared     =  0.1268
-------------+------------------------------           Adj R-squared =  0.1179
       Total |  1859.98328    99  18.7877099           Root MSE      =  4.0709

------------------------------------------------------------------------------
           y |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
           x |   1.464391   .3881261     3.77   0.000     .6941675    2.234615
       _cons |   .5269774   .4070931     1.29   0.199    -.2808856     1.33484
------------------------------------------------------------------------------

. areg y x, absorb(id)

Linear regression, absorbing indicators                Number of obs =     100
                                                       F(  1,    89) =  159.56
                                                       Prob > F      =  0.0000
                                                       R-squared     =  0.9530
                                                       Adj R-squared =  0.9477
                                                       Root MSE      =  .99123

------------------------------------------------------------------------------
           y |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
           x |    1.28649   .1018454    12.63   0.000     1.084126    1.488855
       _cons |   .5277863   .0991239     5.32   0.000     .3308292    .7247434
-------------+----------------------------------------------------------------
          id |          F(9, 89) =    173.771   0.000          (10 categories)
```

Bootstrapping With Mata (BROKEN!)

```
clear
set seed 14170
set obs 50
set more off
local B = 10000
set matsize `B'
matrix betas = J(`B', 1, 0)

gen x = invnormal(uniform())
gen y = x + invnormal(uniform())

forvalues b = 1/`B' {
  preserve
  bsample
  mata
  x = st_data(., ("x"))
  cons = J(rows(x), 1, 1)
  y = st_data(., ("y"))
  X = (x, cons)
  beta_hat = invsym(cross(X,X)) * cross(X,y)
  st_matrix("b", beta_hat)
  end
  matrix betas[`b',1] = b[1,1]
  restore
}
regress y x
drop _all
svmat betas
summ
```

Bootstrapping With Mata (BROKEN!)

```
.
. forvalues b = 1/`B' {
  2.   preserve
  3.   bsample
  4.   mata
  5.   x = st_data(., ("x"))
  6.   cons = J(rows(x), 1, 1)
  7.   y = st_data(., ("y"))
  8.   X = (x, cons)
  9.   beta_hat = invsym(cross(X,X)) * cross(X,y)
 10.   st_matrix("b", beta_hat)
 11.   end
--Break--
r(1);

end of do-file
```

Bootstrapping With Mata (GOOD!)

```
clear
set seed 14170
set obs 50
set more off
local B = 10000
set matsize `B'
matrix betas = J(`B', 1, 0)
gen x = invnormal(uniform())
gen y = x + invnormal(uniform())

forvalues b = 1/`B' {
  preserve
  bsample
  quietly do helper.do
  matrix betas[`b',1] = b[1,1]
  restore
}
regress y x
drop _all
svmat betas
summ
```

(helper.do file)

```
mata
x = st_data(., ("x"))
cons = J(rows(x), 1, 1)
y = st_data(., ("y"))
X = (x, cons)
beta_hat = invsym(cross(X,X)) * cross(X,y)
st_matrix("b", beta_hat)
end
```

Bootstrapping With Mata (GOOD!)

```
. regress y x

      Source |       SS       df       MS              Number of obs =      50
-------------+------------------------------           F(  1,    48) =   37.67
       Model |  40.9289568     1  40.9289568           Prob > F      =  0.0000
    Residual |  52.1553759    48  1.08657033           R-squared     =  0.4397
-------------+------------------------------           Adj R-squared =  0.4280
       Total |  93.0843327    49  1.89968026           Root MSE      =  1.0424

------------------------------------------------------------------------------
           y |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
           x |   .7953509   .1295903     6.14   0.000     .5347922     1.05591
       _cons |   .1628063   .1477362     1.10   0.276    -.1342372    .4598498
------------------------------------------------------------------------------

. drop _all

. svmat betas
number of observations will be reset to 10000
Press any key to continue, or Break to abort
obs was 0, now 10000

. summ

    Variable |       Obs        Mean    Std. Dev.       Min        Max
-------------+--------------------------------------------------------
      betas1 |     10000    .7908434    .1224051   .2737004   1.275181
```

GMM-OLS review

$$g(\beta) = \mathrm{E}[X'\varepsilon] = 0$$

$$\beta_{GMM} = \arg\min_{\beta} g(\beta)' W g(\beta)$$

$$\hat{\varepsilon}_i = y_i - X_i \beta$$

$$\hat{g}(\beta) = \frac{1}{N} \sum_{i=1}^{N} X_i' \hat{\varepsilon}_i$$

$$\hat{W} = I$$

$$\hat{\beta}_{GMM} = \arg\min_{\beta} \left(\frac{1}{N} \sum_{i=1}^{N} X_i' \hat{\varepsilon}_i \right)' \left(\frac{1}{N} \sum_{i=1}^{N} X_i' \hat{\varepsilon}_i \right)$$

GMM in Mata

```
clear
set obs 100
set seed 14170
gen x = invnorm(uniform())
gen y = 1 + 2 * x + invnorm(uniform())

mata
mata clear
x_vars = st_data(., ("x"))
cons = J(rows(x_vars), 1, 1)
X = (x_vars , cons)
y = st_data(., ("y"))
X
data = (y, X)
void ols_gmm0(todo,betas,data,Xe,S,H){
 y =  data[1...,1]
 X =  data[1...,2..3]
 e = y - X * (betas')
 Xe = (X'*e/rows(X))'*(X'*e/rows(X))
}
S = optimize_init()
optimize_init_evaluator(S, &ols_gmm0())
optimize_init_evaluatortype(S, "v0")
optimize init which(S, "min")
optimize_init_params(S, J(1,2,3))
optimize_init_argument(S, 1, data)
```

```
p = optimize(S)
gmm_V = ///
 (1/(rows(X)-cols(X))) * ///
 (y-X*p')'*(y-X*p') * ///
 invsym(X' * X)
gmm_se = sqrt(diagonal(gmm_V))

P
gmm_se
end

reg y x
```

```
:
: p = optimize(S)
Iteration 0:   f(p) =   62415.188
Iteration 1:   f(p) =   5.737e-14
Iteration 2:   f(p) =   3.105e-24

: gmm_V = ///
>  (1/(rows(X)-cols(X))) * ///
>  (y - X * p')'*(y - X*p') * ///
>  invsym(X' * X)

: gmm_se = sqrt(diagonal(gmm_V))

: p
                 1              2
    +-----------------------------+
  1 |  2.098960624    .9759229598  |
    +-----------------------------+

: gmm_se
                 1
    +---------------+
  1 |  .0843677198  |
  2 |  .0927795947  |
    +---------------+

: end
-------------------------------------------------------------------------------

.
. reg y x

      Source |       SS       df       MS              Number of obs =     100
-------------+------------------------------           F(  1,    98) =  618.95
       Model |  527.343707     1  527.343707           Prob > F      =  0.0000
    Residual |  83.4956566    98  .851996496           R-squared     =  0.8633
-------------+------------------------------           Adj R-squared =  0.8619
       Total |  610.839364    99  6.17009459           Root MSE      =  .92304

------------------------------------------------------------------------------
           y |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
           x |   2.098961   .0843677    24.88   0.000     1.931536    2.266386
       _cons |    .975923   .0927796    10.52   0.000     .7918049    1.160041
------------------------------------------------------------------------------
```

GMM-IV overview (iid errors)

$$\mathrm{E}[Z'\varepsilon] = 0$$

$$\hat{\varepsilon}_i = y_i - X_i\beta$$

$$\hat{g}(\beta) = \frac{1}{N}\sum_{i=1}^{N} Z_i'\hat{\varepsilon}_i$$

$$\hat{W} = \left(\frac{Z'Z}{N}\right)^{-1}$$

$$\beta_{GMM} = \arg\min_{\beta} g(\beta)'Wg(\beta)$$

$$\hat{\beta}_{GMM} = \arg\min_{\beta}\left(\frac{1}{N}\sum_{i=1}^{N} Z_i'\hat{\varepsilon}_i\right)'\left(\frac{Z'Z}{N}\right)^{-1}\left(\frac{1}{N}\sum_{i=1}^{N} Z_i'\hat{\varepsilon}_i\right)$$

GMM in Mata (IV)

```
clear
set obs 100
set seed 14170
gen spunk = invnorm(uniform())
gen z1 = invnorm(uniform())
gen z2 = invnorm(uniform())
gen z3 = invnorm(uniform())
gen x = ///
 invnorm(uniform()) + ///
 10*spunk + ///
 z1 + z2 + z3
gen ability = ///
 invnorm(uniform())+10*spunk
gen y = ///
 2*x+ability + ///
 .1*invnorm(uniform())

mata
mata clear
x_vars = st_data(., ("x"))
Z = st_data(., ("z1","z2","z3"))
cons = J(rows(x_vars), 1, 1)
X = (x_vars)
y = st_data(., ("y"))
X
data = (y, Z, X)
```

```
void
   oiv_gmm0(todo,betas,data,mWm,S,H){
 y =  data[1...,1]
 Z =  data[1...,2..4]
 X =  data[1...,5]
 e = y - X * (betas')
 m = (1/rows(Z)) :* (Z'*e)
 mWm = (m'*(invsym(Z'Z)*rows(Z))*m)
}

S = optimize_init()
optimize_init_evaluator(S,&oiv_gmm0())
optimize_init_evaluatortype(S, "v0")
optimize_init_which(S, "min")
optimize_init_params(S, J(1,1,5))
optimize_init_argument(S, 1, data)
p = optimize(S)
p
end
ivreg y (x = z1 z2 z3), nocons
```

```
: p = optimize(S)
Iteration 0:  f(p) =  48.269141
Iteration 1:  f(p) =  .43446345
Iteration 2:  f(p) =  .43446345

: p
  2.439236459

: end
--------------------------------------------------------------------------------


.
. ivreg y (x = z1 z2 z3) , nocons

Instrumental variables (2SLS) regression

      Source |       SS       df       MS              Number of obs =     100
-------------+------------------------------           F(  1,    99) =       .
       Model |  105968.451     1  105968.451           Prob > F      =       .
    Residual |  3983.89461    99  40.2413597           R-squared     =       .
-------------+------------------------------           Adj R-squared =       .
       Total |  109952.346   100  1099.52346           Root MSE      =  6.3436

------------------------------------------------------------------------------
           y |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
           x |   2.439236   .2348737    10.39   0.000     1.973196    2.905277
------------------------------------------------------------------------------
Instrumented:  x
Instruments:   z1 z2 z3
------------------------------------------------------------------------------
```

GMM-IV = 2SLS

$$\hat{\beta}_{GMM} = \arg\min_{\beta} \frac{1}{N}\left(Z'(y - X\beta)\right)'\left(Z'Z\right)^{-1}\left(Z'(y - X\beta)\right)$$

$$= \arg\min_{\beta} \frac{1}{N}(y - X\beta)'Z\left(Z'Z\right)^{-1}Z'(y - X\beta)$$

$$= \arg\min_{\beta} \frac{1}{N}(y - X\beta)'P_Z(y - X\beta)$$

$$0 = (y - X\beta)'P_Z(-X) + (y - X\beta)'P_Z(-X)$$

$$= (y - X\beta)'P_Z X$$

$$= (y - X\beta)'(X'P_z)'$$

$$= ((X'P_z)(y - X\beta))'$$

$$= (X'P_z y - X'P_z X\beta)'$$

$$X'P_z y = X'P_z X\beta$$

$$\hat{\beta}_{GMM} = (X'P_Z X)^{-1}X'P_Z y$$

Normal Mixture using "Method of Moment-Generating Functions"

$$f(y) = \lambda(1/\sigma_1)\phi((y-\mu_1)/\sigma_1) + (1-\lambda)(1/\sigma_2)\phi((y-\mu_2)/\sigma_2)$$

$$M(t) = E[e^{tx}] = e^{t\mu + t^2\sigma^2/2}$$

$$\hat{m}_{GMM} = \frac{1}{N}\sum_{i=1}^{N} e^{ty_i} - (\lambda e^{t\mu_1 + t^2\sigma_1^2/2} + (1-\lambda)e^{t\mu_2 + t^2\sigma_2^2/2}) = 0$$

Normal Mixture GMM in Mata

```
mata
mata clear
data = st_data(., ("v"))

void oiv_gmm0(todo,betas,data,mWm,S,H) {
 N = rows(data)
 ones = J(1, N, 1)
 ts = (0.1, 0.2, 0.3, 0.4, 0.5)
 m = 0
 lambda = normal(betas[1,1])
 sigma_1 = exp(betas[1,2])
 sigma_2 = exp(betas[1,3])
 for (i = 1; i<=5; i++) {
  t = ts[1,i]
  mT=ones*exp(t*data)/N
  mT=mT-lambda*exp(t*betas[1,4]+t^2*(sigma_1^2)/2)
  mT=mT-(1-lambda)*exp(t*betas[1,5]+t^2*(sigma_2^2)/2)
  m = (m, mT)
 }
 mWm = m * m'
}
S = optimize_init()
optimize_init_evaluator(S,&oiv_gmm0())
optimize_init_evaluatortype(S, "v0")
optimize_init_which(S, "min")
init = (-0.2,0,0.7,1,0.5)
optimize_init_params(S, init)
optimize_init_argument(S, 1, data)
p = optimize(S)
p = (normal(p[1,1]), exp(p[1,2]), exp(p[1,3]), p[1,4], p[1,5])
p
end
```

Normal Mixture in Stata ML

```
: p = optimize(S)
Iteration 0:  f(p) =  .00013527  (not concave)
Iteration 1:  f(p) =  .00002705  (not concave)
Iteration 2:  f(p) =   .0000244  (not concave)
Iteration 3:  f(p) =  .00002216  (not concave)
Iteration 4:  f(p) =  .00001598  (not concave)
Iteration 5:  f(p) =  .00001389  (not concave)
Iteration 6:  f(p) =  .00001052  (not concave)
Iteration 7:  f(p) =  6.580e-06  (not concave)
Iteration 8:  f(p) =  6.227e-06  (not concave)
Iteration 9:  f(p) =  5.531e-07  (not concave)
Iteration 10:  f(p) =  5.381e-07  (not concave)
Iteration 11:  f(p) =  4.961e-07  (not concave)
Iteration 12:  f(p) =  4.225e-07  (not concave)
Iteration 13:  f(p) =  4.546e-08  (not concave)
Iteration 14:  f(p) =  2.922e-08  (not concave)
Iteration 15:  f(p) =  2.846e-08  (not concave)
Iteration 16:  f(p) =  2.728e-08  (not concave)
Iteration 17:  f(p) =  2.551e-08
Iteration 18:  f(p) =  7.650e-09

: p = (normal(p[1,1]), exp(p[1,2]), exp(p[1,3]), p[1,4], p[1,5])

: p
                 1             2             3             4             5
    +-----------------------------------------------------------------------+
  1 |  .3179022401   1.084268912   2.033915384   .9800508865    .484377959  |
    +-----------------------------------------------------------------------+
```

Exercises

(A) Non-linear GMM-IV using Mata (EASY)

(B) Bootstrap standard errors of Non-linear GMM-IV estimator (MEDIUM)

(C) Test that the bootstrapped standard errors are consistent using a Monte Carlo simulation (HARD)