

This first section of code needs to be added to the program that creates the Mata library called libbayes as described in the posting 'Creating a Mata library'

```

void mixMNormal(
  real matrix Y, /* nxp data matrix */
  real matrix MU, /* initial means pxr */
  real matrix T, /* initial precisions px(pr) */
  real vector P, /* initial probabilities */
  real scalar r, /* components in mixture */
  real matrix M, /* means of priors on mean pxr */
  real matrix B, /* prec matrix of priors on mean px(pr) */
  real matrix R, /* scale matrices of priors on prec px(pr) */
  real vector K, /* df of priors on prec lxr */
  real vector C, /* allocation nx1 values each 1 to r */
  real vector ALPHA, /* Dirichlet prior parameter lxr */
  real scalar burnin,
  real scalar updates,
  string scalar filename
)
{
  p = cols(Y)
  n = rows(Y)
  fh = fopen(filename,"w")
  outLine = ""
  for(s=1;s<=r;s++) {
    for(j=1;j<=p;j++) {
      if( s == 1 & j == 1) outLine = outLine + "mul_1"
      else outLine = outLine + ",mu" + strofreal(s) + "_" + strofreal(j)
    }
    for(i=1;i<=p;i++) {
      for(j=1;j<=p;j++) {
        outLine = outLine + ",T" + strofreal(s) + "_" + strofreal(i) + "_" + strofreal(j)
      }
    }
    outLine = outLine + ",p" + strofreal(s)
  }
  for(i=1;i<=n;i++) {
    outLine = outLine + ",c" + strofreal(i)
  }
  fput(fh,outLine)
  for(iter=1;iter<=burnin+updates;iter++) {
    /* Update mu for each component */
    for(s=1;s<=r;s++) {
      j1 = (s-1)*p+1
      j2 = s*p
      Ms = M[.,s]
      Ts = T[|1,j1 \ p,j2|]
      Bs = B[|1,j1 \ p,j2|]
      ST = Bs
      SM = Bs*Ms
      for(i=1;i<=n;i++) {
        if( C[i] == s ) {
          ST = ST + Ts
          SM = SM + Ts*Y[i,.]
        }
      }
      V = invsym(ST)
      MU[.,s] = rMNormal(V*SM,cholesky(V))
    }
    /* Update T for each component */
    for(s=1;s<=r;s++) {
      j1 = (s-1)*p+1
      j2 = s*p
      MUs = MU[.,s]
      Rs = R[|1,j1 \ p,j2|]
      SR = Rs
      df = K[s]
      for(i=1;i<=n;i++) {
        if( C[i] == s ) {
          SR = SR + (Y[i,.]'-MUs)*(Y[i,.]'-MUs)'
          df = df + 1
        }
      }
      T[|1,j1 \ p,j2|] = rWishart(cholesky(invsym(SR)),df)
    }
  }
}

```

```

/* update membership */
for(i=1;i<=n;i++) {
  D = J(r,1,0)
  maxD = 0
  for(s=1;s<=r;s++) {
    j1 = (s-1)*p+1
    j2 = s*p
    MUs = MU[.,s]
    Ts = T[|1,j1 \ p,j2|]
    D[s] = log(det(Ts))-(Y[i,.]'-MUs)'*Ts*(Y[i,.]'-MUs)
    if( s == 1 | D[s] > maxD ) maxD = D[s]
  }
  sD = 0
  for(s=1;s<=r;s++) {
    D[s] = P[s]*exp(D[s]-maxD)
    sD = sD + D[s]
  }
  for(s=1;s<=r;s++) {
    D[s] = D[s]/sD
  }
  C[i] = rCategorical(D)
}
/* update probabilities */
N = ALPHA
for(i=1;i<=n;i++) {
  N[C[i]] = N[C[i]] + 1
}
P = rDirichlet(N)
/* write to file */
if( iter > burnin ) {
  outLine = ""
  for(s=1;s<=r;s++) {
    for(j=1;j<=p;j++) {
      if( s == 1 & j == 1 ) outLine = outLine + strofreal(MU[j,s])
      else outLine = outLine + "," + strofreal(MU[j,s])
    }
    for(i=1;i<=p;i++) {
      for(j=1;j<=p;j++) {
        outLine = outLine + "," + strofreal(T[i,(s-1)*p+j])
      }
    }
    outLine = outLine + "," + strofreal(P[s])
  }
  for(i=1;i<=n;i++) {
    outLine = outLine + "," + strofreal(C[i])
  }
  fput(fh,outLine)
}
fclose(fh)
}
mata mlib add libbayes mixMNormal()

```

This code gives the Stata program for calling the Mata function. You could either place it at the head of the analysis program or copy it to an ado file called mixmnormal.ado and allow Stata to load it whenever mixmnormal is referred to.

```
program mixmnormal
    syntax varlist , MU(string) T(string) P(string) n(integer) ///
        M(string) B(string) R(string) DF(string) C(string) ALPHA(string) ///
        BUrnin(integer) Updates(integer) Filename(string)

    tempname Y
    mata: mata clear
    qui putmata `Y'=(`varlist')
    local list "`mu' `r' `df' `m' `b' `t' `c' `alpha' `p'"
    foreach item of local list {
        mata : `item' = st_matrix("`item'")
    }
    cap erase "`filename'"
    mata:
    mixMNormal(`Y', `mu', `t', `p', `n', `m', `b', `r', `df', `c', `alpha', `burnin', `updates', "`filename'")
end
```

This code gives the program for creating the plots given in the posting.

```
*-----
* read data and set seed to ensure that
* we can reproduce exactly the same analysis
*-----
set scheme sj
program drop _all
use fish.dta, clear
set seed 430197
*-----
* Priors
*-----
matrix R = (3,3,3,3,3,3)
matrix df = (3,3,3,3,3,3)
matrix M = (6,6,6,6,6,6)
matrix B = (0.1,0.1,0.1,0.1,0.1,0.1)
matrix ALPHA = (20,20,10,5,1,1)
*-----
* initial values
*-----
matrix T = (1,1,1,1,1,1)
matrix C = J(1,256,1)
forvalues i=1/256 {
    matrix C[1,`i'] = 1 + int(6*runiform())
}
matrix P = J(1,6,1/6)
matrix MU = M
*-----
* fit mixture model
*-----
mixmnormal fish , mu(MU) t(T) p(P) n(6) m(M) b(B) r(R) df(df) c(C) alpha(ALPHA) ///
    burnin(2500) updates(5000) filename("temp.csv")
insheet using temp.csv, clear
mcmcstats m* t* p*
forvalues i=1/6 {
    gen sd`i' = 1/sqrt(t`i'_1_1)
}
mcmcstats m* sd* p*
*-----
* plot illustrate fits
*-----
set obs 1001
range y 0 15 1001
foreach iter of numlist 1000 2000 3000 4000 5000 {
    cap drop f`iter'
    gen f`iter' = 0 in 1/1001
    forvalues s=1/6 {
        di %8.4f p`s'[`iter'] %8.3f mu`s'_1[`iter'] %8.3f 1/sqrt(t`s'_1_1[`iter'])
        qui replace f`iter' = f`iter' +
        p`s'[`iter']*normalden(y,mu`s'_1[`iter'],1/sqrt(t`s'_1_1[`iter']))
    }
}
merge 1:1 _n using fish.dta
histogram fish , start(2) width(0.25) xtitle(Fish length) ///
    addplot((line f1000 y, lpat(solid) lcol(blue)) (line f2000 y, lpat(solid) lcol(red)) ///
    (line f3000 y, lpat(solid) lcol(green)) (line f4000 y, lpat(solid) lcol(black)) ///
    (line f5000 y, lpat(solid) lcol(orange)) ) leg(off)
*-----
* create the trace plot
*-----
rename mu6_1 mu1
rename mu1_1 mu2
rename mu2_1 mu3
rename mu4_1 mu4
rename mu3_1 mu5
rename mu5_1 mu6
mcmctrace mu1 mu2 mu3 mu4 mu5 mu6
```

The fish length data

2.875	2.875	2.875	2.875	2.875	2.875	3.125	3.125
3.125	3.125	3.125	3.125	3.125	3.375	3.375	3.375
3.375	3.375	3.375	3.375	3.375	3.375	3.625	3.625
3.625	3.875	3.875	3.875	4.125	4.125	4.125	4.125
4.375	4.375	4.375	4.375	4.375	4.375	4.625	4.625
4.625	4.625	4.625	4.625	4.625	4.625	4.625	4.625
4.625	4.875	4.875	4.875	4.875	4.875	4.875	4.875
4.875	4.875	4.875	4.875	4.875	4.875	4.875	4.875
4.875	4.875	4.875	4.875	4.875	4.875	4.875	4.875
4.875	4.875	4.875	5.125	5.125	5.125	5.125	5.125
5.125	5.125	5.125	5.125	5.125	5.125	5.125	5.125
5.125	5.125	5.125	5.125	5.125	5.125	5.125	5.125
5.125	5.125	5.125	5.375	5.375	5.375	5.375	5.375
5.375	5.375	5.375	5.375	5.375	5.375	5.375	5.375
5.375	5.375	5.375	5.375	5.375	5.625	5.625	5.625
5.625	5.625	5.625	5.625	5.625	5.625	5.625	5.625
5.625	5.625	5.625	5.625	5.625	5.625	5.625	5.625
5.625	5.625	5.625	5.625	5.625	5.875	5.875	5.875
5.875	5.875	5.875	5.875	5.875	5.875	5.875	5.875
5.875	5.875	5.875	6.125	6.125	6.125	6.125	6.125
6.125	6.125	6.125	6.125	6.125	6.125	6.375	6.375
6.375	6.375	6.375	6.375	6.375	6.375	6.625	6.625
6.625	6.625	6.875	6.875	6.875	6.875	6.875	6.875
6.875	7.125	7.125	7.125	7.125	7.125	7.125	7.125
7.125	7.125	7.125	7.125	7.375	7.375	7.375	7.375
7.375	7.375	7.375	7.375	7.375	7.375	7.375	7.625
7.625	7.625	7.625	7.625	7.625	7.625	7.625	7.625
7.625	7.625	7.875	7.875	7.875	7.875	7.875	7.875
7.875	7.875	7.875	8.125	8.125	8.125	8.125	8.125
8.125	8.375	8.375	8.375	8.375	8.625	8.625	8.625
8.875	8.875	8.875	9.125	9.125	9.375	9.375	9.625
9.625	9.625	9.625	9.875	9.875	9.875	10.125	10.125
10.375	10.375	10.625	10.875	11.375	11.875	12.375	12.625