Problem 1

1)

The model we have to estimate can be written as.

$$y_t = G(z_{t-1})y_{0t} + (1 - G(z_{t-1}))y_{1t} \tag{1}$$

Where y_t is the short term interest rate, z_{t-1} are predetermined variables, $G()$ is the mixing function and y_{0t}, y_{1t} are two latent variables which determine y_t. In this case, because we are estimating an LSTAR, $z_{t-1} = (y_{t-1})$, that is, y_{t-1} is all we need to now what the mixing function is . We have to estimate a model with four lags, so

$$y_{0t} = b_0 + \phi_{0,a}y_{t-1} + \phi_{0,b}y_{t-2} + \phi_{0,c}y_{t-3} + \phi_{0,d}y_{t-4} + \sigma_0\varepsilon_t$$
$$y_{1t} = b_1 + \phi_{1,a}y_{t-1} + \phi_{1,b}y_{t-2} + \phi_{1,c}y_{t-3} + \phi_{1,d}y_{t-4} + \sigma_1\varepsilon_t$$

Coefficents for both latent variables are allowed to be different, including the variance of the shock, which in our case will be a student's t. In this case, as we are estimating an LSTAR, the mixing function is given by the logistic function:

$$G(z_{t-1}) = \frac{e^{-\gamma(y_{t-1}-k)}}{1 + e^{-\gamma(y_{t-1}-k)}} \tag{2}$$

With all this said, we have to estimate this model. In order to do so, we will use Gauss.
Once we have installed Gauss (see instructions on Campus Virtual), we should extract the files in the folder 'GAUSS_PS5' (which can be found in Campus Virtual) somewhere. Then we should specify to Gauss that that folder will be our working directory. In order to do so, we go to *File/Change working directory* and choose the folder we have created earlier. It is important to have all files within one single folder, otherwise GAUSS may have problems reading it. Once we have set the working directory, we go to *File/Open* and choose the file 'L-E-STAR', which is the code we'll be using for this problem.
Before running it, it's good practise to take a look at the code to see what it is actually doing. While this is not strictly necessary, we'll do so in order to learn where we could change the code if we had to use it on different data.

First, we set the number of observations. This is important, because GAUSS needs the dimension of matrices to be pre-specified when loading data. If we used a different series, we

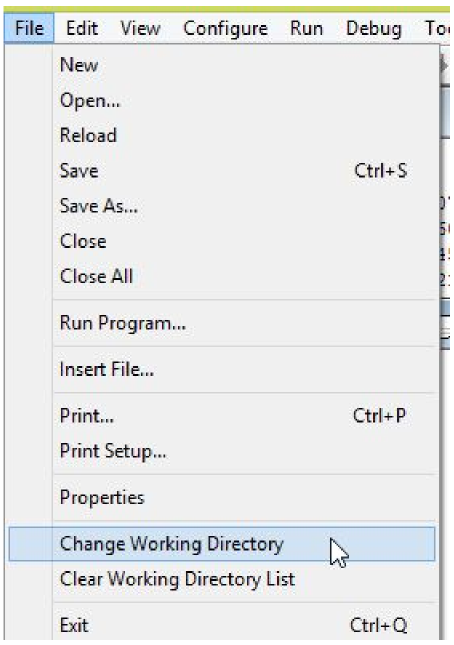


Figura 1: Change Working directory

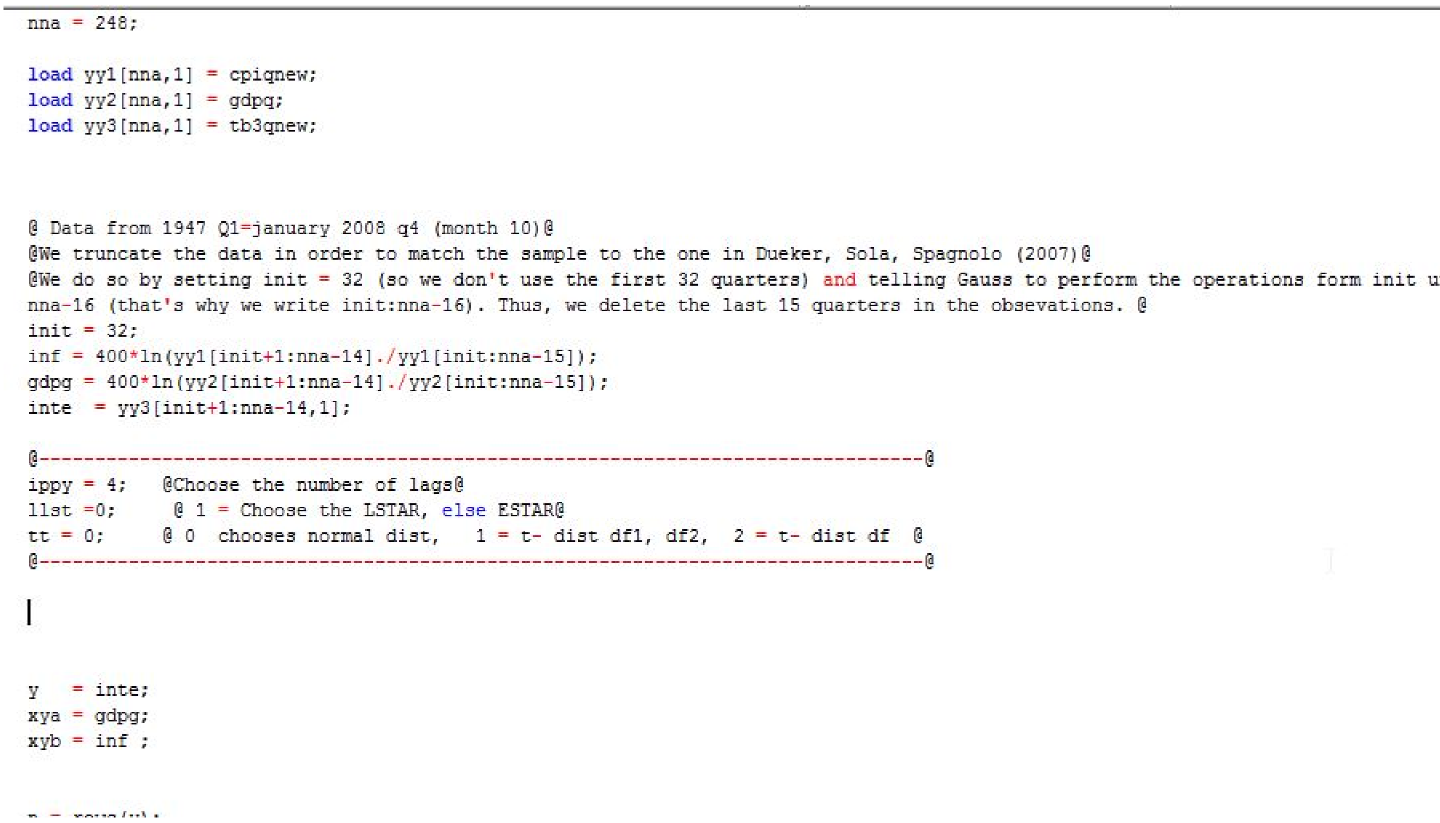

```
nna = 248;

load yy1[nna,1] = cpiqnew;
load yy2[nna,1] = gdpq;
load yy3[nna,1] = tb3qnew;



@ Data from 1947 Q1=january 2008 q4 (month 10)@
@We truncate the data in order to match the sample to the one in Dueker, Sola, Spagnolo (2007)@
@We do so by setting init = 32 (so we don't use the first 32 quarters) and telling Gauss to perform the operations form init u
nna-16 (that's why we write init:nna-16). Thus, we delete the last 15 quarters in the obsevations. @
init = 32;
inf = 400*ln(yy1[init+1:nna-14]./yy1[init:nna-15]);
gdpg = 400*ln(yy2[init+1:nna-14]./yy2[init:nna-15]);
inte  = yy3[init+1:nna-14,1];

@------------------------------------------------------------------------------@
ippy = 4;   @Choose the number of lags@
llst =0;     @ 1 = Choose the LSTAR, else ESTAR@
tt = 0;     @ 0  chooses normal dist,   1 = t- dist df1, df2,   2 = t- dist df  @
@------------------------------------------------------------------------------@



y   = inte;
xya = gdpg;
xyb = inf ;


n = rows(y);
```

Figura 2: First lines of the file

would have to adjust the number `nna`, which is the number of obs. We load three data series: *cpiqnew* (USA quarterly CPI index (in levels)), *gdpq* (quarterly GDP from USA) and *tb3qnew* (3-month T-Bill interest rate).

The code then perform some transformation of the DATA in order to make it usable: it diffe-

rences the first two series to get inflation and GDP growth, and correct the number of obs of the third series. We also truncate the data in order to match the sample used in Dueker, Sola, Spagnolo (2007). This is important, because this kind of models are 'taylor-made' to forecast certain features with a specific data set.
Then we have to specify estimation settings. We set the number of lags to 4, set *llst = 1* in order to use the LSTAR specficitacion, and choose to use student's t distributed errors, with the same degrees of freedom for both states (had we chosen the option $tt = 1$, then we would be allowing the df of both errors to be different. This could be useful if, for example, in one state there were a larger chance of outliers, therefore we would like the df of this state to be lower than in the other, close-to-gaussian-inovations state.).

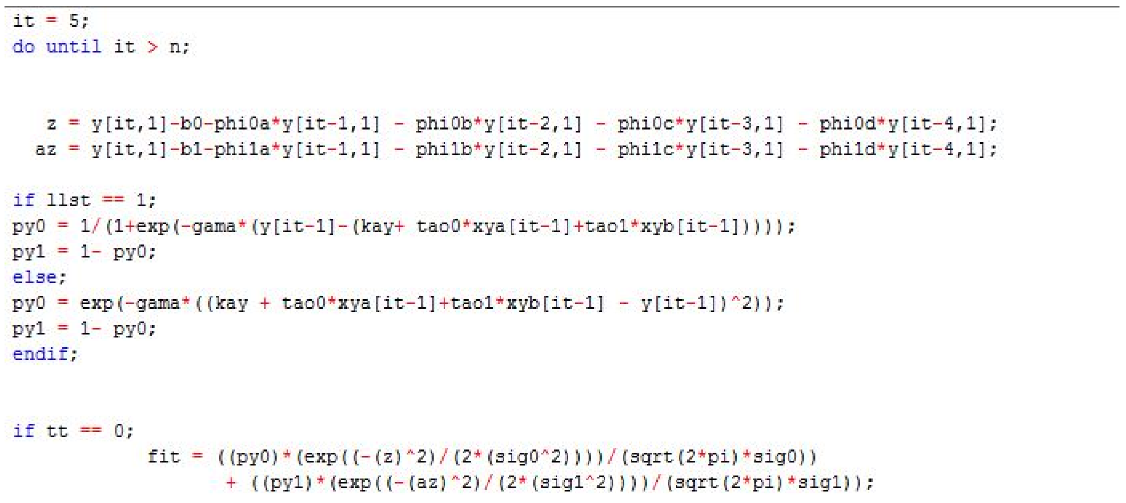
The data is then renamed: y, the short term interest rate, will be the variable we want to explain. We won't be using the other two variables for the moment (we'll se why, and then when and how we DO use them in Problem 2). The code then goes on. It good to notice some other things that can help to understand the code better in the case you want to change it:

- There are lots of *if* closes. The code is written this way to allow for several specfications. The source of this is the fact that we can change the number of lags or the distribution of innovations solely by changing a number at the beginning. *If* closes are there to let the program know that, for example, if we chose to use 4 lags, then the parameter space has to be of a different dimension than if we chose 2 lags.

- Real action will not take place in this code. Circa line 120, the code calls other functions (*ttstant22*) which contain the likelihood functions to be optimized. Then, it uses the optimizer in the line 139. The output of this optimization is x (estimated parameters), f (likelihood function evaluated at the optimum), g (gradient vector, should be close to zero) and h (evaluated Hessian matrix, can be used to estimate the standard errors.)

- the rest of the code computes the standard errors (there are several specifications to be chosen here), and prints the output. Once more, there are several *if* closes because the output printed will depend on the options we choose.

We can also take a closer look at the likelihood file, *ttstant22*. In this case, the code firstly defines all parameters to be estimated, and output to be produced (such as p-values). Then it defines the likelihood. It is interesting to note how this is done. The code writes the deviations



```
it = 5;
do until it > n;


   z = y[it,1]-b0-phi0a*y[it-1,1] - phi0b*y[it-2,1] - phi0c*y[it-3,1] - phi0d*y[it-4,1];
  az = y[it,1]-b1-phi1a*y[it-1,1] - phi1b*y[it-2,1] - phi1c*y[it-3,1] - phi1d*y[it-4,1];

if llst == 1;
py0 = 1/(1+exp(-gama*(y[it-1]-(kay+ tao0*xya[it-1]+tao1*xyb[it-1]))));
py1 = 1- py0;
else;
py0 = exp(-gama*((kay + tao0*xya[it-1]+tao1*xyb[it-1] - y[it-1])^2));
py1 = 1- py0;
endif;


if tt == 0;
            fit = ((py0)*(exp((-(z)^2)/(2*(sig0^2))))/(sqrt(2*pi)*sig0))
                   + ((py1)*(exp((-(az)^2)/(2*(sig1^2))))/(sqrt(2*pi)*sig1));
```

Figura 3: Writing the likelihood

of each observation from the values that the model would predict. then uses this to compute the densities. Finally, it computes the sum of log-densities in order to get the log-likelihood.

With a better picture of what the code is doing let's move on to estimation. We have to estimate a LSTAR (therefore, we choose that option with *llst=1*) with 4 lags (therefore we set ippy=4) and with t-student distribution (we use option 2 for convinience). We run the code by clicking in the arrow shown in the figure.

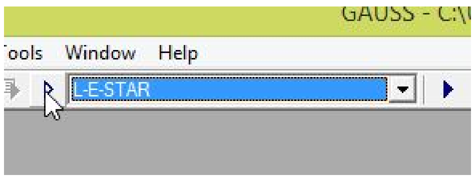
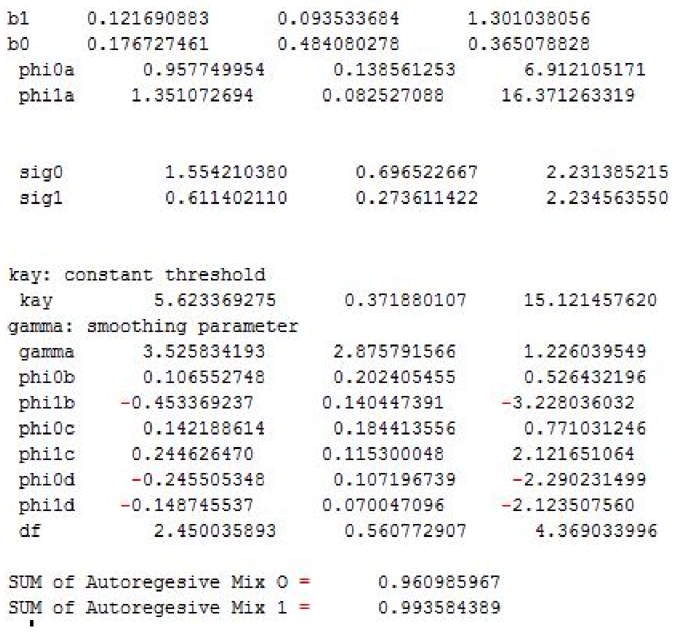


Figura 4: Run the file

The output is shown in the figure. The code also shows us a plot of the threshold (which in this case is constant), the value of the estimated mixing function, and plots of original series.

The first column is the estimated coefficent, the second one is its standard error, and the third one shows t-stats.

Coefficent	value	SE	t-stat
b_1	0.121690883	0.093533684	1.301038056
b_0	0.176727461	0.484080278	0.365078828
$\phi_{0,a}$	0.957749954	0.138561253	6.912105171
$\phi_{1,a}$	1.351072694	0.082527088	16.371263319
σ_0	1.554210380	0.696522667	2.231385215
σ_1	0.611402110	0.273611422	2.234563550
k	5.623369275	0.371880107	15.121457620
γ	3.525834193	2.875791566	1.226039549
$\phi_{0,b}$	0.106552748	0.202405455	0.526432196
$\phi_{1,b}$	-0.453369237	0.140447391	-3.228036032
$\phi_{0,c}$	0.142188614	0.184413556	0.771031246
$\phi_{1,c}$	0.244626470	0.115300048	2.121651064
$\phi_{0,d}$	-0.245505348	0.107196739	-2.290231499
$\phi_{1,d}$	-0.148745537	0.070047096	-2.123507560
df	2.450035893	0.560772907	4.369033996

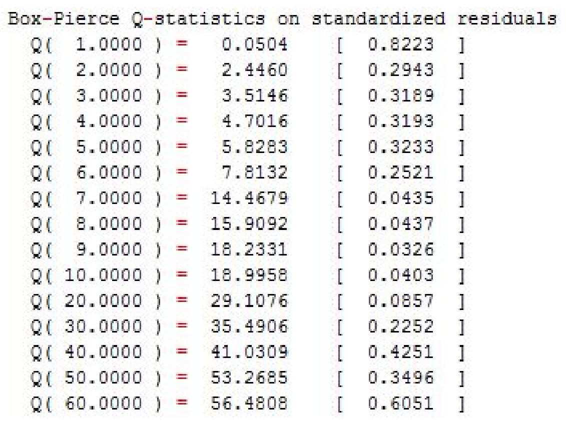


```
b1      0.121690883      0.093533684      1.301038056
b0      0.176727461      0.484080278      0.365078828
 phi0a      0.957749954      0.138561253      6.912105171
 phi1a     1.351072694      0.082527088     16.371263319


 sig0        1.554210380      0.696522667      2.231385215
 sig1        0.611402110      0.273611422      2.234563550


kay: constant threshold
 kay         5.623369275      0.371880107     15.121457620
gamma: smoothing parameter
 gamma       3.525834193      2.875791566      1.226039549
 phi0b       0.106552748      0.202405455      0.526432196
 phi1b    -0.453369237      0.140447391     -3.228036032
 phi0c       0.142188614      0.184413556      0.771031246
 phi1c      0.244626470      0.115300048      2.121651064
 phi0d      -0.245505348      0.107196739     -2.290231499
 phi1d     -0.148745537      0.070047096     -2.123507560
 df          2.450035893      0.560772907      4.369033996

SUM of Autoregesive Mix 0 =      0.960985967
SUM of Autoregesive Mix 1 =      0.993584389
```

Figura 5: Estimation output

2)

These are also shown in the estimation output. Here we see that we have a problem in the



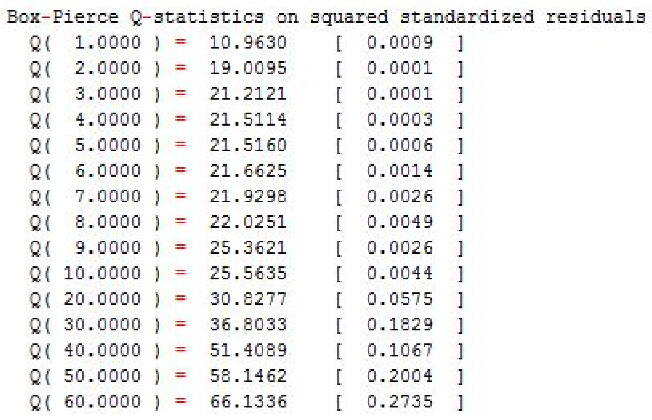
```
Box-Pierce Q-statistics on standardized residuals
  Q(  1.0000 ) =   0.0504    [  0.8223  ]
  Q(  2.0000 ) =   2.4460    [  0.2943  ]
  Q(  3.0000 ) =   3.5146    [  0.3189  ]
  Q(  4.0000 ) =   4.7016    [  0.3193  ]
  Q(  5.0000 ) =   5.8283    [  0.3233  ]
  Q(  6.0000 ) =   7.8132    [  0.2521  ]
  Q(  7.0000 ) =  14.4679    [  0.0435  ]
  Q(  8.0000 ) =  15.9092    [  0.0437  ]
  Q(  9.0000 ) =  18.2331    [  0.0326  ]
  Q( 10.0000 ) =  18.9958    [  0.0403  ]
  Q( 20.0000 ) =  29.1076    [  0.0857  ]
  Q( 30.0000 ) =  35.4906    [  0.2252  ]
  Q( 40.0000 ) =  41.0309    [  0.4251  ]
  Q( 50.0000 ) =  53.2685    [  0.3496  ]
  Q( 60.0000 ) =  56.4808    [  0.6051  ]
```

Figura 6: Box-Pierce Q-stats for residuals

9th and 10th lag. For square residuals, the Q-stats are shown in the figure. We reject all the null of no correlation for all lags, therefore we have a problem modelling the variance as well.

3)

The plots are also made when running the code. We note that the threshold is constant, which is natural, since we estimated it that way. On the other hand, we can see the plot of the mixing function. This is interesting to see, as we can infer from here how important is each



```
Box-Pierce Q-statistics on squared standardized residuals
  Q(  1.0000 ) =  10.9630    [  0.0009  ]
  Q(  2.0000 ) =  19.0095    [  0.0001  ]
  Q(  3.0000 ) =  21.2121    [  0.0001  ]
  Q(  4.0000 ) =  21.5114    [  0.0003  ]
  Q(  5.0000 ) =  21.5160    [  0.0006  ]
  Q(  6.0000 ) =  21.6625    [  0.0014  ]
  Q(  7.0000 ) =  21.9298    [  0.0026  ]
  Q(  8.0000 ) =  22.0251    [  0.0049  ]
  Q(  9.0000 ) =  25.3621    [  0.0026  ]
  Q( 10.0000 ) =  25.5635    [  0.0044  ]
  Q( 20.0000 ) =  30.8277    [  0.0575  ]
  Q( 30.0000 ) =  36.8033    [  0.1829  ]
  Q( 40.0000 ) =  51.4089    [  0.1067  ]
  Q( 50.0000 ) =  58.1462    [  0.2004  ]
  Q( 60.0000 ) =  66.1336    [  0.2735  ]
```

Figura 7: Box-Pierce Q-stats for sq. residuals

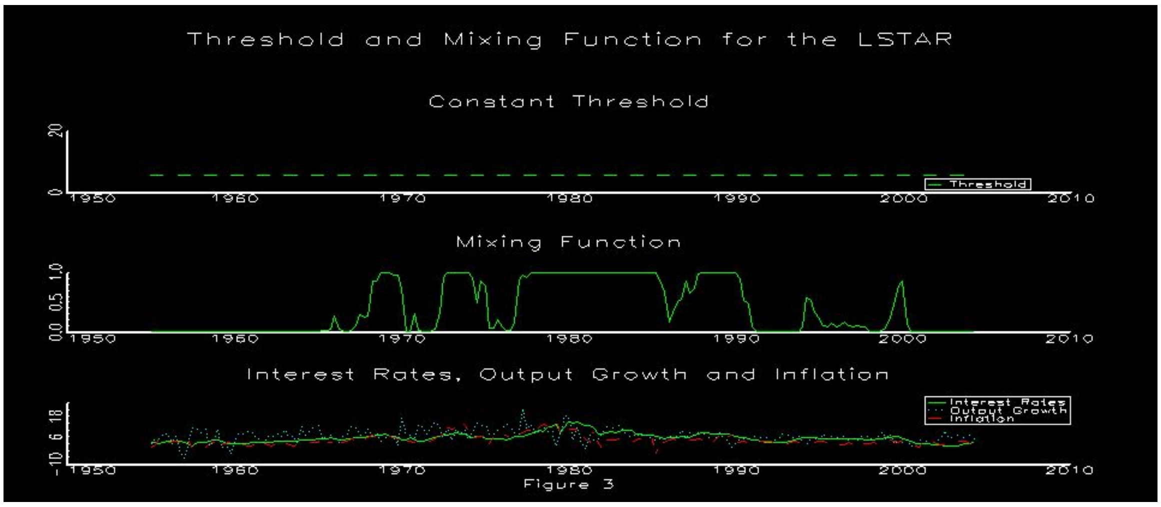
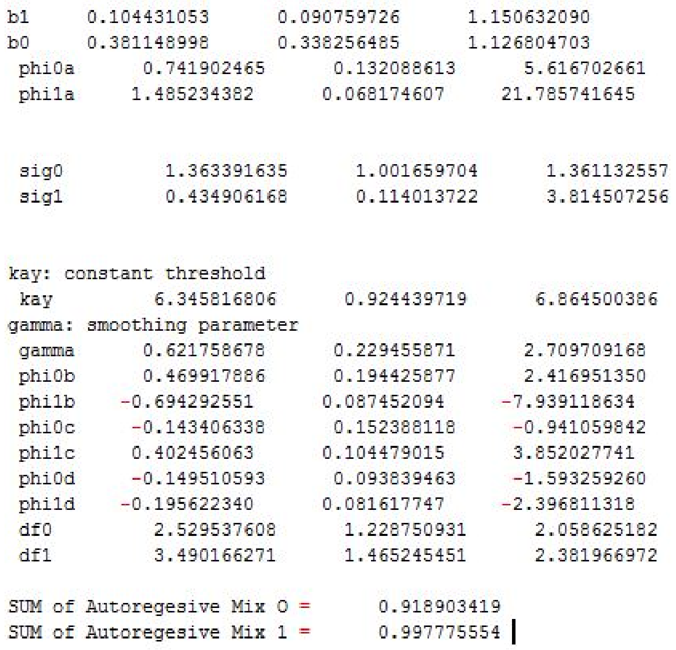


Figura 8: Threshold, mixing function, and variables values

latent variable at a given time. In this case, a higher value of the mixing funtion seems to be associated to a regime of higher interest rates.

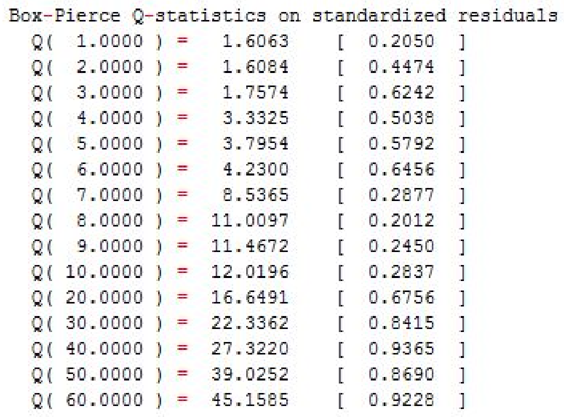
4)

We just have to change the corresponding option in the code. To unrestrict the dof's, we choose $tt = 1$ and run the code again. Estimation output, Q-stats and graphs are all shown in figures below. Results do not differ substantially from those obtained before. In this case, none of the p-values of the Q-stats for residuals are below 5 %, so residuals seem clear. We have problems again with square residuals. We note also that the sum of coefficients of the AR terms is close to 1, that is, there is almost a unit root, which could be a huge problem.



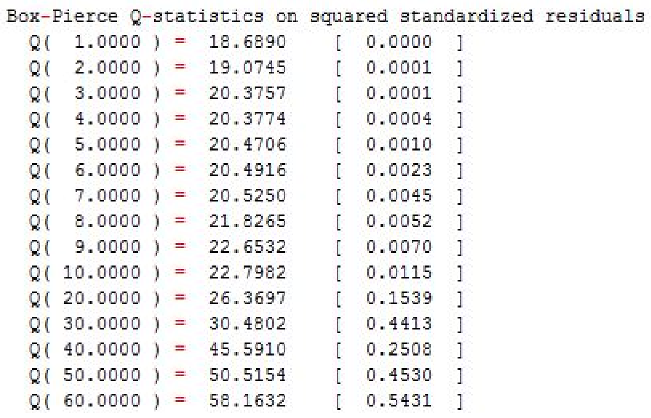
```
b1     0.104431053      0.090759726      1.150632090
b0     0.381148998      0.338256485      1.126804703
 phi0a      0.741902465      0.132088613      5.616702661
 phi1a     1.485234382      0.068174607     21.785741645


 sig0         1.363391635      1.001659704      1.361132557
 sig1         0.434906168      0.114013722      3.814507256


kay: constant threshold
 kay         6.345816806      0.924439719      6.864500386
gamma: smoothing parameter
 gamma      0.621758678      0.229455871      2.709709168
 phi0b      0.469917886      0.194425877      2.416951350
 phi1b    -0.694292551      0.087452094     -7.939118634
 phi0c     -0.143406338      0.152388118     -0.941059842
 phi1c     0.402456063      0.104479015      3.852027741
 phi0d     -0.149510593      0.093839463     -1.593259260
 phi1d    -0.195622340      0.081617747     -2.396811318
 df0        2.529537608      1.228750931      2.058625182
 df1        3.490166271      1.465245451      2.381966972

SUM of Autoregesive Mix 0 =       0.918903419
SUM of Autoregesive Mix 1 =       0.997775554
```

Figura 9: Estimation output. Note that de df of the errors are estimated separately



```
Box-Pierce Q-statistics on standardized residuals
  Q(  1.0000 ) =    1.6063    [  0.2050  ]
  Q(  2.0000 ) =    1.6084    [  0.4474  ]
  Q(  3.0000 ) =    1.7574    [  0.6242  ]
  Q(  4.0000 ) =    3.3325    [  0.5038  ]
  Q(  5.0000 ) =    3.7954    [  0.5792  ]
  Q(  6.0000 ) =    4.2300    [  0.6456  ]
  Q(  7.0000 ) =    8.5365    [  0.2877  ]
  Q(  8.0000 ) =   11.0097    [  0.2012  ]
  Q(  9.0000 ) =   11.4672    [  0.2450  ]
  Q( 10.0000 ) =   12.0196    [  0.2837  ]
  Q( 20.0000 ) =   16.6491    [  0.6756  ]
  Q( 30.0000 ) =   22.3362    [  0.8415  ]
  Q( 40.0000 ) =   27.3220    [  0.9365  ]
  Q( 50.0000 ) =   39.0252    [  0.8690  ]
  Q( 60.0000 ) =   45.1585    [  0.9228  ]
```

Figura 10: Q-stats for the residuals



```
Box-Pierce Q-statistics on squared standardized residuals
  Q(  1.0000 ) =  18.6890    [  0.0000  ]
  Q(  2.0000 ) =  19.0745    [  0.0001  ]
  Q(  3.0000 ) =  20.3757    [  0.0001  ]
  Q(  4.0000 ) =  20.3774    [  0.0004  ]
  Q(  5.0000 ) =  20.4706    [  0.0010  ]
  Q(  6.0000 ) =  20.4916    [  0.0023  ]
  Q(  7.0000 ) =  20.5250    [  0.0045  ]
  Q(  8.0000 ) =  21.8265    [  0.0052  ]
  Q(  9.0000 ) =  22.6532    [  0.0070  ]
  Q( 10.0000 ) =  22.7982    [  0.0115  ]
  Q( 20.0000 ) =  26.3697    [  0.1539  ]
  Q( 30.0000 ) =  30.4802    [  0.4413  ]
  Q( 40.0000 ) =  45.5910    [  0.2508  ]
  Q( 50.0000 ) =  50.5154    [  0.4530  ]
  Q( 60.0000 ) =  58.1632    [  0.5431  ]
```

Figura 11: Q-stats for the squared residuals

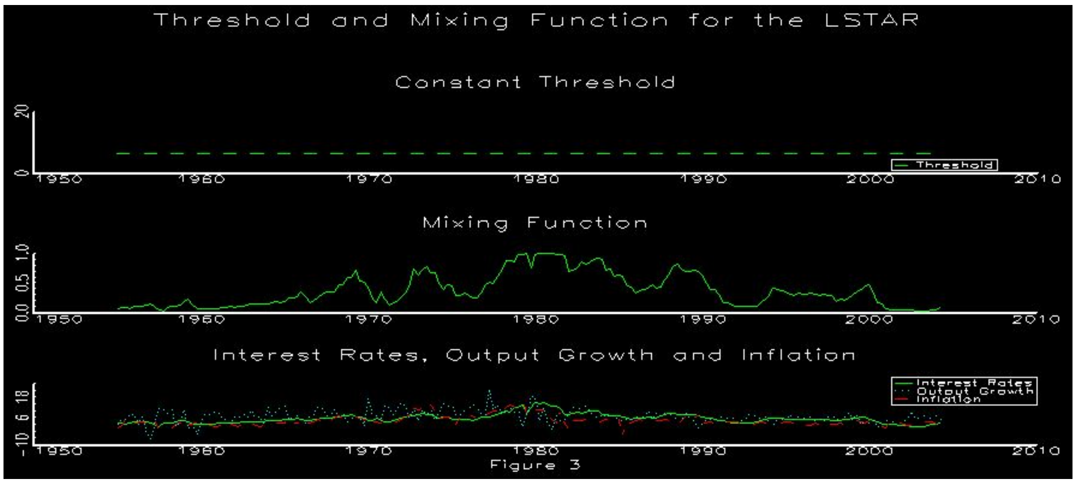
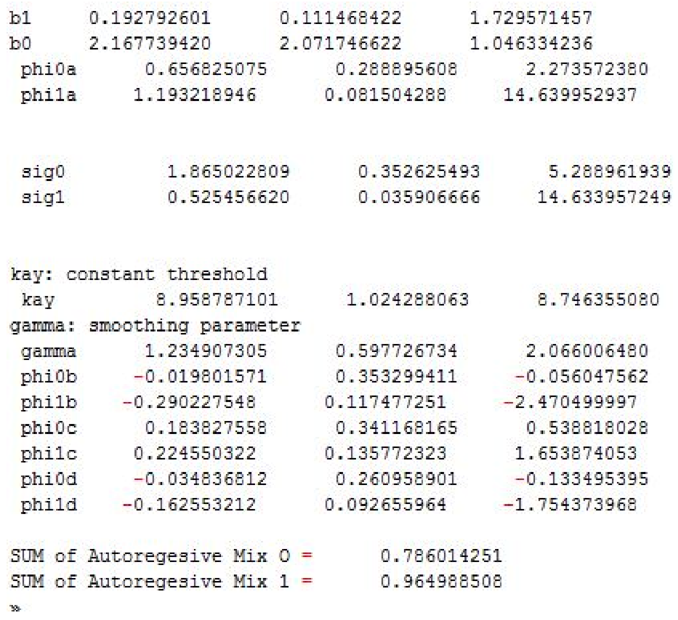


Figura 12: Plot of the threshold and mixing function

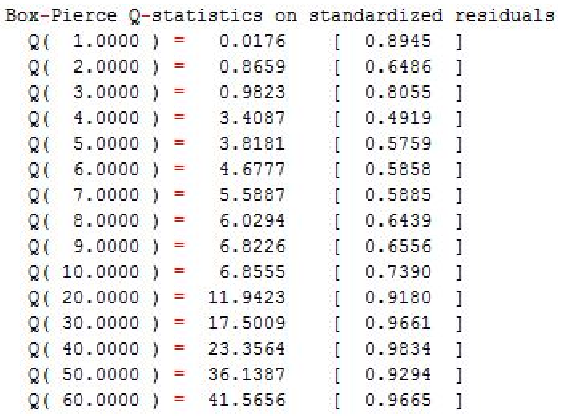
5)

We just have to change the corresponding option in the code. To use the normal distribution, we choose $tt = 0$ and run the code again. Estimation output, Q-stats and graphs are all shown in figures below. In this case, residuals and square residuals are clean. Also, there seems to be no problem with the stationarity of the AR parts. This is the specification used in Dueker, Sola, Spagnolo (2007). Our plot replicates figure 4 of the paper.



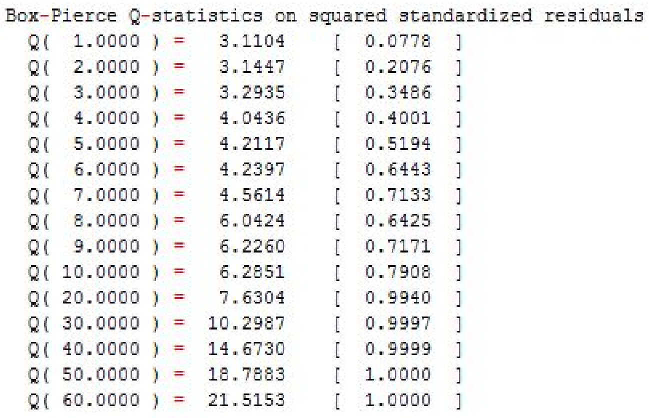
```
b1      0.192792601       0.111468422       1.729571457
b0      2.167739420       2.071746622       1.046334236
 phi0a       0.656825075       0.288895608       2.273572380
 phi1a      1.193218946       0.081504288      14.639952937


 sig0         1.865022809       0.352625493       5.288961939
 sig1         0.525456620       0.035906666      14.633957249


kay: constant threshold
 kay          8.958787101       1.024288063       8.746355080
gamma: smoothing parameter
 gamma       1.234907305       0.597726734       2.066006480
 phi0b      -0.019801571       0.353299411      -0.056047562
 phi1b     -0.290227548       0.117477251     -2.470499997
 phi0c       0.183827558       0.341168165       0.538818028
 phi1c      0.224550322       0.135772323       1.653874053
 phi0d      -0.034836812       0.260958901      -0.133495395
 phi1d     -0.162553212       0.092655964     -1.754373968

SUM of Autoregesive Mix 0 =       0.786014251
SUM of Autoregesive Mix 1 =       0.964988508
>>
```

Figura 13: Estimation output for gaussian errors



```
Box-Pierce Q-statistics on standardized residuals
  Q(  1.0000 ) =    0.0176    [  0.8945  ]
  Q(  2.0000 ) =    0.8659    [  0.6486  ]
  Q(  3.0000 ) =    0.9823    [  0.8055  ]
  Q(  4.0000 ) =    3.4087    [  0.4919  ]
  Q(  5.0000 ) =    3.8181    [  0.5759  ]
  Q(  6.0000 ) =    4.6777    [  0.5858  ]
  Q(  7.0000 ) =    5.5887    [  0.5885  ]
  Q(  8.0000 ) =    6.0294    [  0.6439  ]
  Q(  9.0000 ) =    6.8226    [  0.6556  ]
  Q( 10.0000 ) =    6.8555    [  0.7390  ]
  Q( 20.0000 ) =   11.9423    [  0.9180  ]
  Q( 30.0000 ) =   17.5009    [  0.9661  ]
  Q( 40.0000 ) =   23.3564    [  0.9834  ]
  Q( 50.0000 ) =   36.1387    [  0.9294  ]
  Q( 60.0000 ) =   41.5656    [  0.9665  ]
```

Figura 14: Q-stats for the residuals



```
Box-Pierce Q-statistics on squared standardized residuals
  Q(  1.0000 ) =    3.1104    [  0.0778  ]
  Q(  2.0000 ) =    3.1447    [  0.2076  ]
  Q(  3.0000 ) =    3.2935    [  0.3486  ]
  Q(  4.0000 ) =    4.0436    [  0.4001  ]
  Q(  5.0000 ) =    4.2117    [  0.5194  ]
  Q(  6.0000 ) =    4.2397    [  0.6443  ]
  Q(  7.0000 ) =    4.5614    [  0.7133  ]
  Q(  8.0000 ) =    6.0424    [  0.6425  ]
  Q(  9.0000 ) =    6.2260    [  0.7171  ]
  Q( 10.0000 ) =    6.2851    [  0.7908  ]
  Q( 20.0000 ) =    7.6304    [  0.9940  ]
  Q( 30.0000 ) =   10.2987    [  0.9997  ]
  Q( 40.0000 ) =   14.6730    [  0.9999  ]
  Q( 50.0000 ) =   18.7883    [  1.0000  ]
  Q( 60.0000 ) =   21.5153    [  1.0000  ]
```

Figura 15: Q-stats for the squared residuals

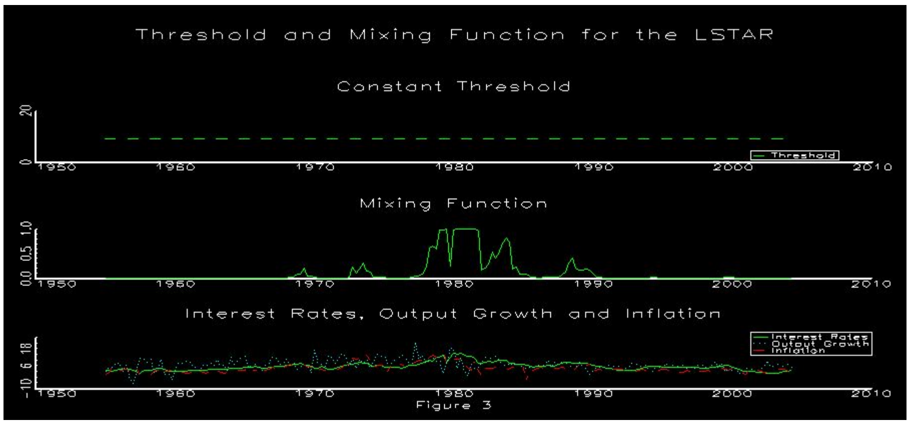
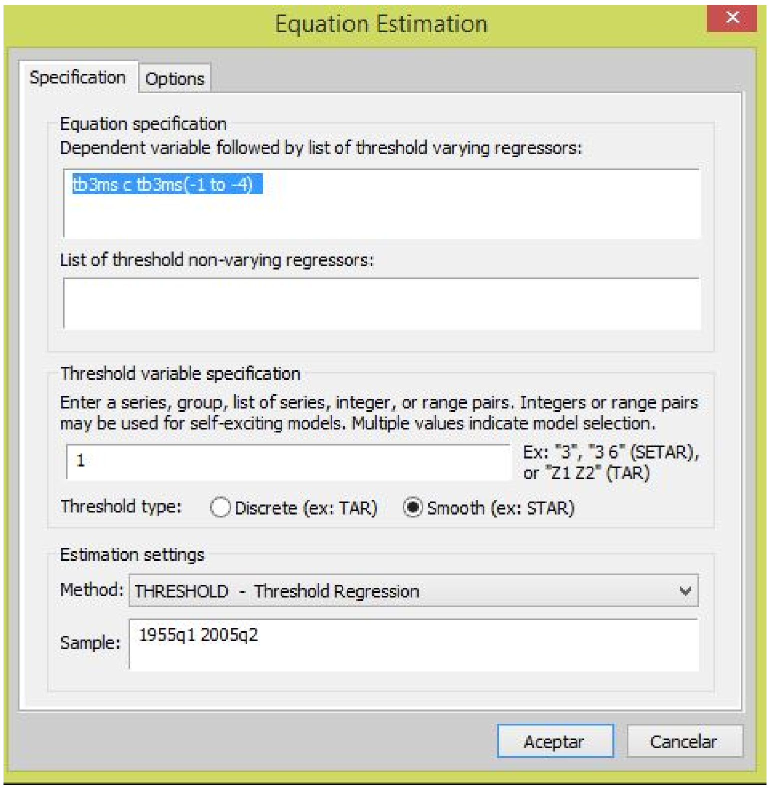


Figura 16: Plot of the threshold and mixing function

Problem 2

To estimate the model, we go to *Quick/estimate equation* and select 'Threshold' in method. We get a window like the ones shown in the figure.



Equation specification
Dependent variable followed by list of threshold varying regressors:

```
tb3ms c tb3ms(-1 to -4)
```

List of threshold non-varying regressors:

Threshold variable specification
Enter a series, group, list of series, integer, or range pairs. Integers or range pairs may be used for self-exciting models. Multiple values indicate model selection.

```
1
```

Ex: "3", "3 6" (SETAR), or "Z1 Z2" (TAR)

Threshold type: ( ) Discrete (ex: TAR) [✓ Smooth (ex: STAR)]

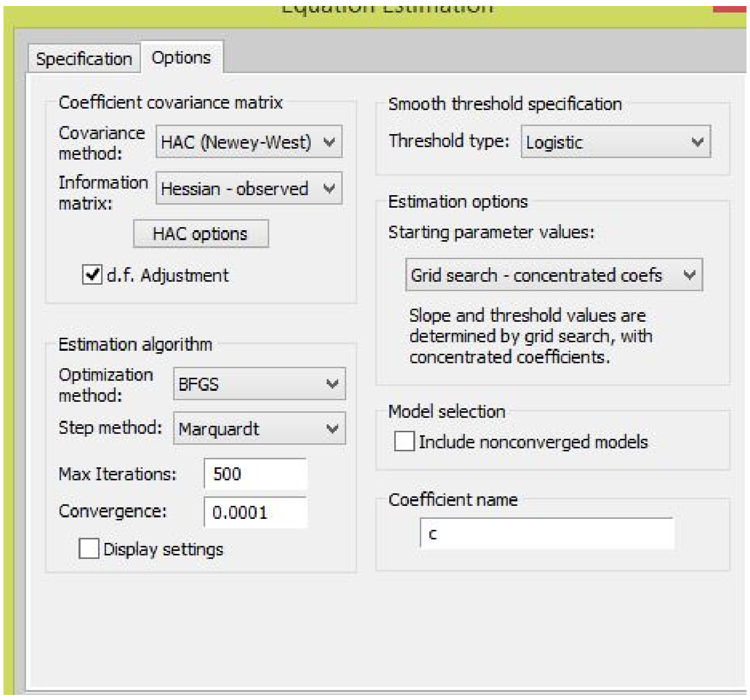
Estimation settings
Method: THRESHOLD - Threshold Regression
Sample: 1955q1 2005q2

Figura 17: Estimation specification

In the first box we have to write the dependent variable (the 3-month interest rate) followed by the **threshold varying regressors**, that is, regressors with different coefficients for both regimes. In the *Threshold vairable specification* box, we write '1' to indicate Eviews that we want to use only one lag as the variable the threshold depends upon. If we added extra lags o other variables, Eviews don't use them all. Instead, the program chooses one (the best) of the variables listed as the one the threshold depends upon.
In the 'options' section, we can choose the form of the mixing function. We also set the coefficent covariance matrix to be similar from that in the Gauss code. Therefore, we choose *HAC-Newey west*, we use the Hessian to compute the information matrix. In the optimization options, we choose 'line search' as step method. The output is given in the figure below. How do we read this? To know that, we go to *View/ representations*. There we can see what Eviews actually estimated. There, we notice that the estimated equation looks like

$$y_t = (\mu_0 + \sum_{i=1}^{4} \phi_{0,i} y_{t-i}) + G(y_{t-1})(\mu_1 + \sum_{i=1}^{4} \phi_{1,i} y_{t-i}) \tag{3}$$

With $G(y_{t-1}) = \dfrac{1}{1+e^{-\gamma(y_{t-1}-k)}}$ being the logistic function. This equation is equivalent to the one used in lectures. More importantly, the estimation method is 'non-linear least squares'. Basically, Eviews tries to minimize the sum of squared residuals. This is somewhat different



Specification | Options

Coefficient covariance matrix
Covariance method: HAC (Newey-West)
Information matrix: Hessian - observed
HAC options
✓ d.f. Adjustment

Estimation algorithm
Optimization method: BFGS
Step method: Marquardt
Max Iterations: 500
Convergence: 0.0001
☐ Display settings

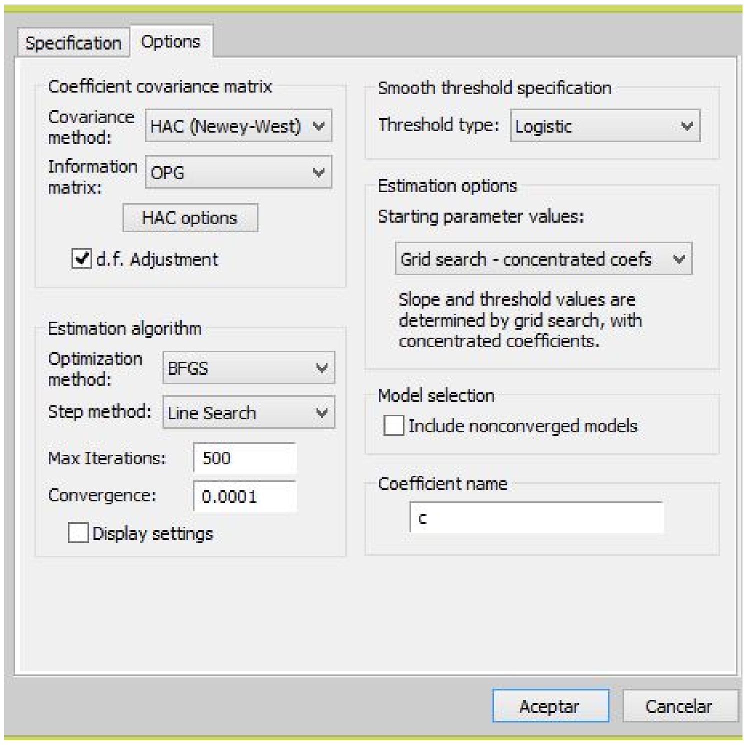
Smooth threshold specification
Threshold type: Logistic

Estimation options
Starting parameter values:
Grid search - concentrated coefs
Slope and threshold values are determined by grid search, with concentrated coefficients.

Model selection
☐ Include nonconverged models

Coefficient name
c

Figura 18: Estimation specification



Specification | Options

Coefficient covariance matrix
Covariance method: HAC (Newey-West)
Information matrix: OPG
HAC options
✓ d.f. Adjustment

Estimation algorithm
Optimization method: BFGS
Step method: Line Search
Max Iterations: 500
Convergence: 0.0001
☐ Display settings

Smooth threshold specification
Threshold type: Logistic

Estimation options
Starting parameter values:
Grid search - concentrated coefs
Slope and threshold values are determined by grid search, with concentrated coefficients.

Model selection
☐ Include nonconverged models

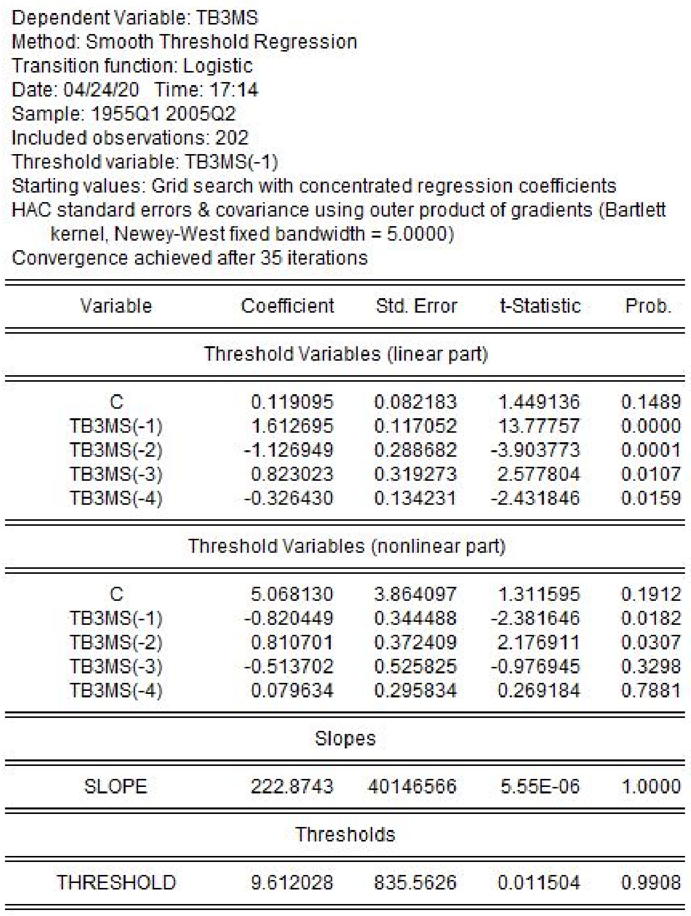
Coefficient name
c

Aceptar | Cancelar

Figura 19: Estimation options

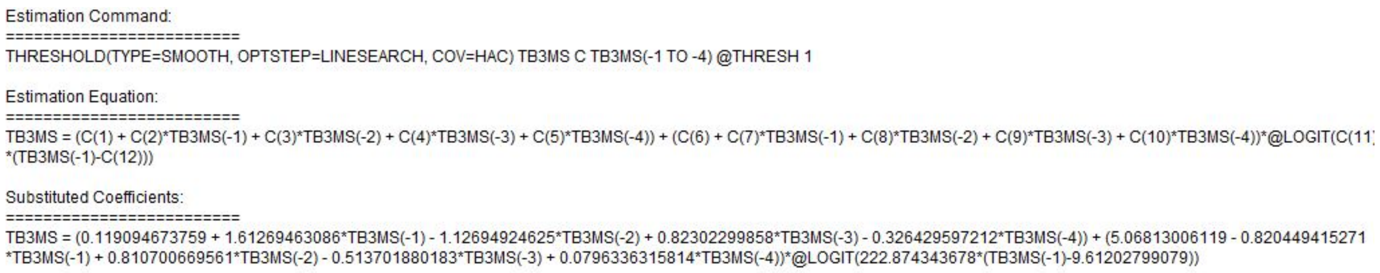
from the ML aproach taken in Problem 1 [1] We can also look the corelograms In this case, we

[1]Acutally, the GAUSS code estimates other type of model. In that model, we have two underlying regimes and y_t is generated with probability $G(z_{t-1})$ by y_{0t} and with probability $1 - G(z_{t-1})$ by y_{1t}. That's why the likelihood is like 'the weighted sum of densities'. Ask Prof. Solá for further details.



```
Dependent Variable: TB3MS
Method: Smooth Threshold Regression
Transition function: Logistic
Date: 04/24/20   Time: 17:14
Sample: 1955Q1 2005Q2
Included observations: 202
Threshold variable: TB3MS(-1)
Starting values: Grid search with concentrated regression coefficients
HAC standard errors & covariance using outer product of gradients (Bartlett
      kernel, Newey-West fixed bandwidth = 5.0000)
Convergence achieved after 35 iterations
=============================================================================
      Variable          Coefficient    Std. Error    t-Statistic    Prob.
=============================================================================
                      Threshold Variables (linear part)
=============================================================================
         C               0.119095      0.082183      1.449136      0.1489
      TB3MS(-1)          1.612695      0.117052      13.77757      0.0000
      TB3MS(-2)         -1.126949      0.288682     -3.903773      0.0001
      TB3MS(-3)          0.823023      0.319273      2.577804      0.0107
      TB3MS(-4)         -0.326430      0.134231     -2.431846      0.0159
=============================================================================
                     Threshold Variables (nonlinear part)
=============================================================================
         C               5.068130      3.864097      1.311595      0.1912
      TB3MS(-1)         -0.820449      0.344488     -2.381646      0.0182
      TB3MS(-2)          0.810701      0.372409      2.176911      0.0307
      TB3MS(-3)         -0.513702      0.525825     -0.976945      0.3298
      TB3MS(-4)          0.079634      0.295834      0.269184      0.7881
=============================================================================
                                  Slopes
=============================================================================
        SLOPE            222.8743      40146566      5.55E-06      1.0000
=============================================================================
                                Thresholds
=============================================================================
      THRESHOLD          9.612028      835.5626      0.011504      0.9908
=============================================================================
```

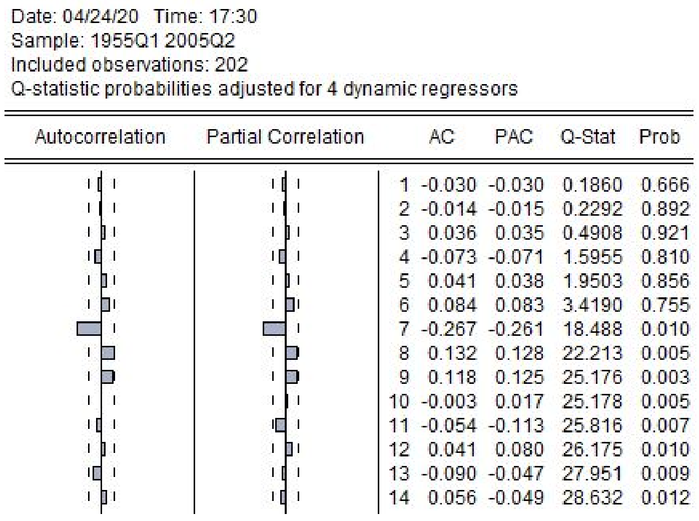
Figura 20: Estimation output



```
Estimation Command:
=========================
THRESHOLD(TYPE=SMOOTH, OPTSTEP=LINESEARCH, COV=HAC) TB3MS C TB3MS(-1 TO -4) @THRESH 1

Estimation Equation:
=========================
TB3MS = (C(1) + C(2)*TB3MS(-1) + C(3)*TB3MS(-2) + C(4)*TB3MS(-3) + C(5)*TB3MS(-4)) + (C(6) + C(7)*TB3MS(-1) + C(8)*TB3MS(-2) + C(9)*TB3MS(-3) + C(10)*TB3MS(-4))*@LOGIT(C(11)
*(TB3MS(-1)-C(12)))

Substituted Coefficients:
=========================
TB3MS = (0.119094673759 + 1.61269463086*TB3MS(-1) - 1.12694924625*TB3MS(-2) + 0.82302299858*TB3MS(-3) - 0.326429597212*TB3MS(-4)) + (5.06813006119 - 0.820449415271
*TB3MS(-1) + 0.810700669561*TB3MS(-2) - 0.513701880183*TB3MS(-3) + 0.0796336315814*TB3MS(-4))*@LOGIT(222.874343678*(TB3MS(-1)-9.61202799079))
```

Figura 21: Estimated equation

that there is unmodeled structure left in residuals. The model estimated in Gauss were slightly different, but in this case, differences seem to matter. Also, Gauss codes allow us to modify estimation settings knowing exactlly what we are doing.



```
Date: 04/24/20   Time: 17:30
Sample: 1955Q1 2005Q2
Included observations: 202
Q-statistic probabilities adjusted for 4 dynamic regressors

 Autocorrelation   Partial Correlation      AC     PAC    Q-Stat   Prob

                                        1 -0.030  -0.030  0.1860   0.666
                                        2 -0.014  -0.015  0.2292   0.892
                                        3  0.036   0.035  0.4908   0.921
                                        4 -0.073  -0.071  1.5955   0.810
                                        5  0.041   0.038  1.9503   0.856
                                        6  0.084   0.083  3.4190   0.755
                                        7 -0.267  -0.261  18.488   0.010
                                        8  0.132   0.128  22.213   0.005
                                        9  0.118   0.125  25.176   0.003
                                       10 -0.003   0.017  25.178   0.005
                                       11 -0.054  -0.113  25.816   0.007
                                       12  0.041   0.080  26.175   0.010
                                       13 -0.090  -0.047  27.951   0.009
                                       14  0.056  -0.049  28.632   0.012
```

Figura 22: Correlograms

Problem 3

1)

In this case, the threshold is state dependent: that is, it is not constant but dependant on other variables such as inflation rate or output growth. In this case, following the notation used in the preceding exercise, $z_{t-1} = (cpiqnew_{t-1}, gdpq_{t-1}, tb3qnew_{t-1})'$. In this case, the mixing function is given by:

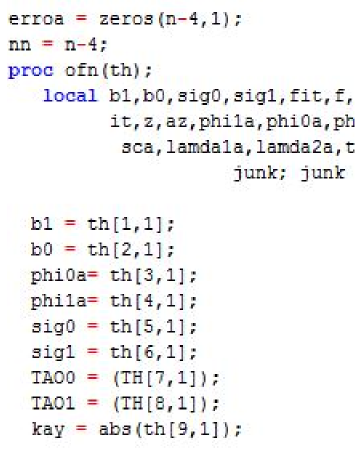
$$G(z_{t-1}) = \frac{e^{-\gamma(y_{t-1}-k-\tau_0 gdpq_{t-1}-\tau_1 tb3qnew_{t-1})}}{1+e^{-\gamma(y_{t-1}-k-\tau_0 gdpq_{t-1}-\tau_1 tb3qnew_{t-1})}} \tag{4}$$

When we open the *ttsand2* file, we note that it is similar to the one in the preceding excersise, but with an important difference: variables *tao0* and *tao1* are not longer set to zero, but initialized and used in optimization instead. Therefore, current specification differs from the one in the previous problem in that the threshold is allowed to vary as a function of other variables.

2)

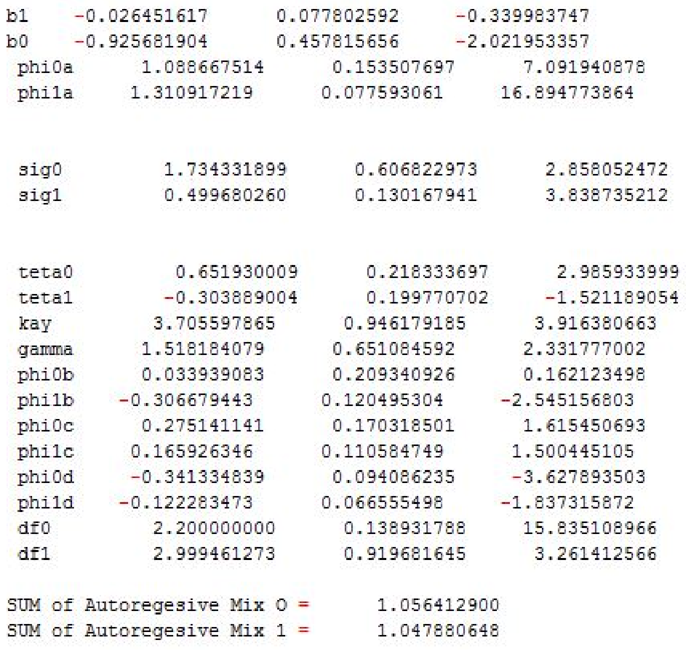
Because of what we did en Problem 1, we now that we have to set $tt = 1$ to use different dof for different student's t distributions. Results are shown in the figures

We note that in this case threshold varies. The sum of the AR coefficients for both states are larger than one, therefore, we would be estimating individual latent variable processes that are not stationary. . This could be a huge problem. There are two potential reasons: either the



```
erroa = zeros(n-4,1);
nn = n-4;
proc ofn(th);
   local b1,b0,sig0,sig1,fit,f,
         it,z,az,phi1a,phi0a,ph
          sca,lamda1a,lamda2a,t
                   junk; junk

  b1 = th[1,1];
  b0 = th[2,1];
  phi0a= th[3,1];
  phi1a= th[4,1];
  sig0 = th[5,1];
  sig1 = th[6,1];
  TAO0 = (TH[7,1]);
  TAO1 = (TH[8,1]);
  kay = abs(th[9,1]);
```

Figura 23: Starting lines of *ttstand2*. Note that *tao0* and *tao1* are now initialized and will be used in estimation.



```
b1    -0.026451617      0.077802592     -0.339983747
b0    -0.925681904      0.457815656     -2.021953357
 phi0a      1.088667514      0.153507697      7.091940878
 phi1a     1.310917219      0.077593061     16.894773864


 sig0        1.734331899      0.606822973      2.858052472
 sig1        0.499680260      0.130167941      3.838735212


 teta0         0.651930009      0.218333697      2.985933999
 teta1        -0.303889004      0.199770702     -1.521189054
 kay         3.705597865      0.946179185      3.916380663
 gamma      1.518184079      0.651084592      2.331777002
 phi0b      0.033939083      0.209340926      0.162123498
 phi1b    -0.306679443      0.120495304     -2.545156803
 phi0c      0.275141141      0.170318501      1.615450693
 phi1c     0.165926346      0.110584749      1.500445105
 phi0d     -0.341334839      0.094086235     -3.627893503
 phi1d    -0.122283473      0.066555498     -1.837315872
 df0         2.200000000      0.138931788     15.835108966
 df1         2.999461273      0.919681645      3.261412566

SUM of Autoregesive Mix 0 =      1.056412900
SUM of Autoregesive Mix 1 =      1.047880648
```

Figura 24: Estimation output for state-dependant threshold

model is wrongly specified (for exmaple, we should use less lags, or an entirely different model), or our optimization procedure stopped in a local maximum, which is not what we are looking for. We could also try with a different distribution for errors (such as gaussian errors).

3)

Several differences can be noticed. First, the mixing function seems to vary quite a lot, allowing us to separate better among the two regimes. Next, we can see that the threshold now varies with the values of inflation and output, making it **state dependant**. This is intuitive

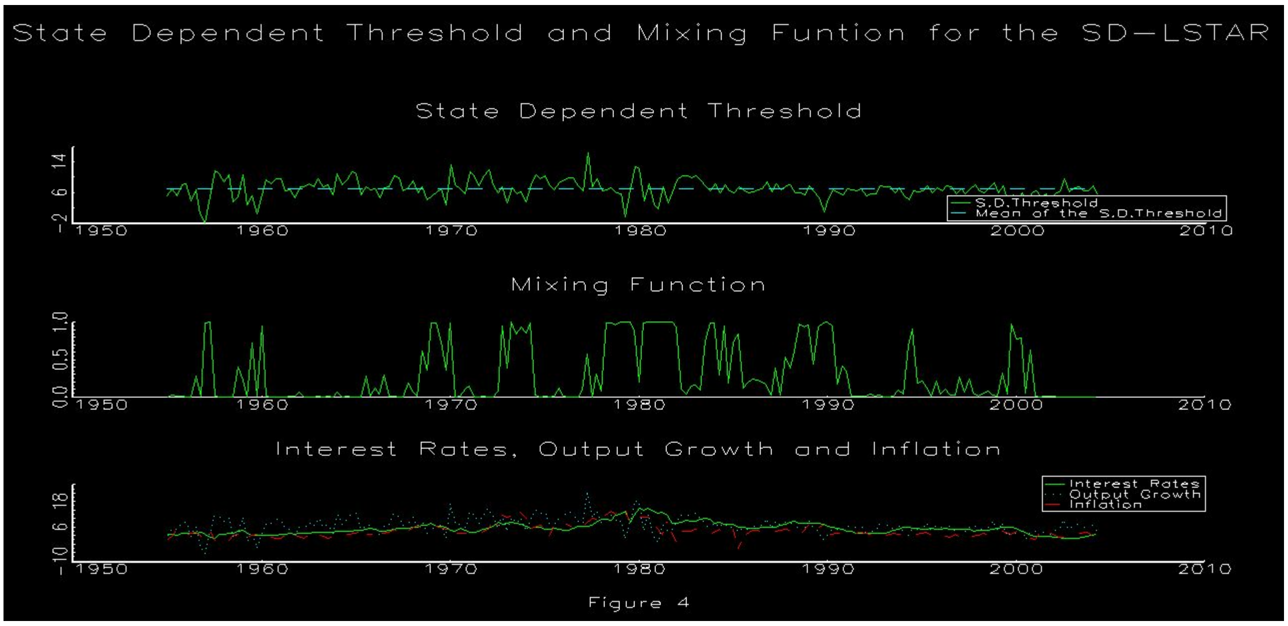


Figura 25: Plot of the threshold and mixing function

since, for example, an interest rate of 10 % may seem 'high' in inflation is 3 %, but low if inflation is 15 %. By allowing the threshold to vary, we let the model to identify 'high' and 'low' interest rates relative to other variables.

Problem 4

1)

In this case, we are working with a SDC-STAR model. The special feature of this model is the choice of the mixing function: it turns out to be related with the ex-ante probabilty of being in a specific regime. To be more specfic, we will be estimating

$$y_t = G(z_{t-1})y_{1t} + (1 - G(z_{t-1}))y_{2t} \tag{5}$$

As before. The 'SD' comes from the fact that the threshold is state dependant. The underlying states are

$$y_{i,t} = b_i \sum_{j=1}^{p} \phi_{i,j} y_{t-j} + \sigma_i \varepsilon_t$$

Two relevant things need to be added to specficiaction:

- The mixing function $G(z_{t-1})$ is given by

$$G\left(\mathbf{z}_{t-1}\right)=\frac{F_1\left(\left\{y^*_{t-1}-\mu_1-\alpha_1'\mathbf{y}_{t-1}\right\}/\sigma_1\right)}{F_1\left(\left\{y^*_{t-1}-\mu_1-\alpha_1'\mathbf{y}_{t-1}\right\}/\sigma_1\right)+1-F_2\left(\left\{y^*_{t-1}-\mu_2-\alpha_2'\mathbf{y}_{t-1}\right\}/\sigma_2\right)} \tag{6}$$

  Where F_1 and F_2 are the CDF of $u_{1,t}$ and $u_{2,t}$ respectively.

- The threshold in is state-dependant and follows the following transition equation

$$y^*_{t-1}=y^*+\delta'\mathbf{x_{t-1}} \tag{7}$$

The SDT-STARX model includes exogenous variables in the latent variable equations, that is, the regimes are characterized now as

$$y_{i,t}=b_i\sum_{j=1}^{p}\phi_{i,j}y_{t-j}+\beta_i'x_{t-1}+\sigma_i\varepsilon_{i,t}$$

Where x_{t-1} are exogenous variables.
The RC-STARX model is a restricted version of the general SDT-STARX, in which the β_i appearing in the latent variables equations are restricted to be the same, that is, exogenous variables parameters are not state dependant.

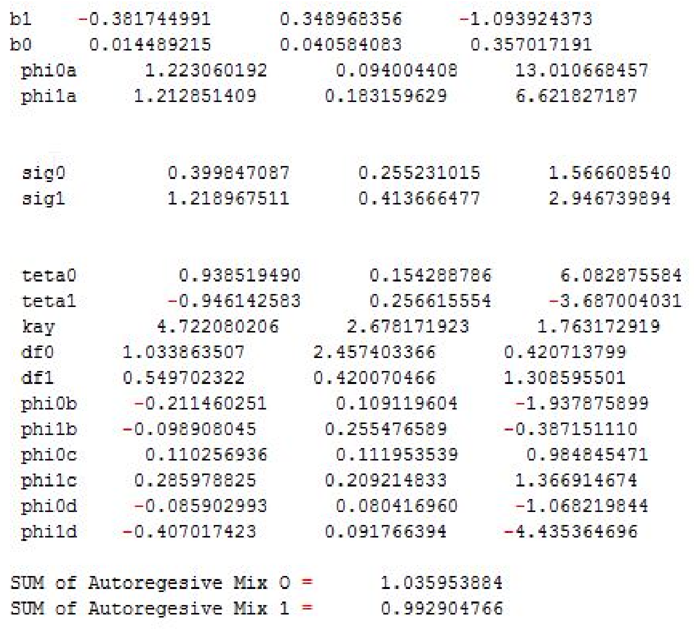
2)

To estimate we use the code given. The structure is the same as those used before. The only change is when the code calls the function to be optimized: these new functions (described in the codes *llt* and *llt1*) are the likelihood functions of this new, more general process. We have to set *refe = 0* in order to estimate the general SDC model.
The estimation output is shown

3)

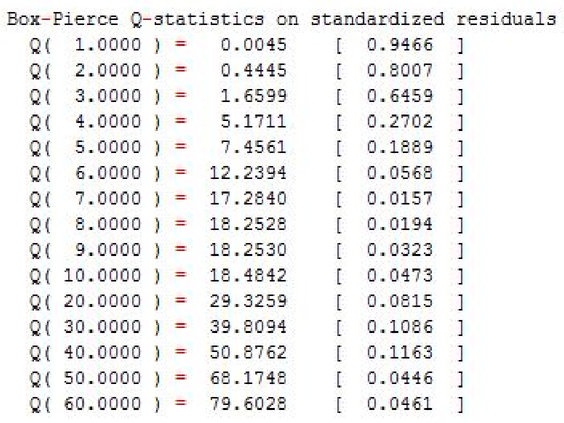
4

To estimate the RC-STARX, we choose *refe=1*. Estimation output is shown in the figures.



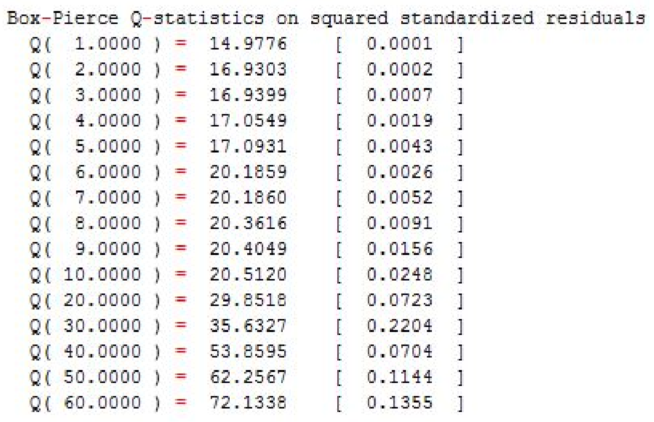
```
b1    -0.381744991      0.348968356     -1.093924373
b0     0.014489215      0.040584083      0.357017191
 phi0a      1.223060192      0.094004408      13.010668457
 phi1a     1.212851409      0.183159629      6.621827187


 sig0        0.399847087      0.255231015      1.566608540
 sig1        1.218967511      0.413666477      2.946739894


 teta0        0.938519490      0.154288786      6.082875584
 teta1       -0.946142583      0.256615554     -3.687004031
 kay        4.722080206      2.678171923      1.763172919
 df0      1.033863507      2.457403366      0.420713799
 df1      0.549702322      0.420070466      1.308595501
 phi0b     -0.211460251      0.109119604     -1.937875899
 phi1b    -0.098908045      0.255476589     -0.387151110
 phi0c      0.110256936      0.111953539      0.984845471
 phi1c     0.285978825      0.209214833      1.366914674
 phi0d     -0.085902993      0.080416960     -1.068219844
 phi1d    -0.407017423      0.091766394     -4.435364696

SUM of Autoregesive Mix 0 =      1.035953884
SUM of Autoregesive Mix 1 =      0.992904766
```

Figura 26: Estimation output of the SDC-STAR model



```
Box-Pierce Q-statistics on standardized residuals
  Q(  1.0000 ) =    0.0045    [  0.9466  ]
  Q(  2.0000 ) =    0.4445    [  0.8007  ]
  Q(  3.0000 ) =    1.6599    [  0.6459  ]
  Q(  4.0000 ) =    5.1711    [  0.2702  ]
  Q(  5.0000 ) =    7.4561    [  0.1889  ]
  Q(  6.0000 ) =   12.2394    [  0.0568  ]
  Q(  7.0000 ) =   17.2840    [  0.0157  ]
  Q(  8.0000 ) =   18.2528    [  0.0194  ]
  Q(  9.0000 ) =   18.2530    [  0.0323  ]
  Q( 10.0000 ) =   18.4842    [  0.0473  ]
  Q( 20.0000 ) =   29.3259    [  0.0815  ]
  Q( 30.0000 ) =   39.8094    [  0.1086  ]
  Q( 40.0000 ) =   50.8762    [  0.1163  ]
  Q( 50.0000 ) =   68.1748    [  0.0446  ]
  Q( 60.0000 ) =   79.6028    [  0.0461  ]
```

Figura 27: Residuals Q-stats



```
Box-Pierce Q-statistics on squared standardized residuals
  Q(  1.0000 ) =   14.9776    [  0.0001  ]
  Q(  2.0000 ) =   16.9303    [  0.0002  ]
  Q(  3.0000 ) =   16.9399    [  0.0007  ]
  Q(  4.0000 ) =   17.0549    [  0.0019  ]
  Q(  5.0000 ) =   17.0931    [  0.0043  ]
  Q(  6.0000 ) =   20.1859    [  0.0026  ]
  Q(  7.0000 ) =   20.1860    [  0.0052  ]
  Q(  8.0000 ) =   20.3616    [  0.0091  ]
  Q(  9.0000 ) =   20.4049    [  0.0156  ]
  Q( 10.0000 ) =   20.5120    [  0.0248  ]
  Q( 20.0000 ) =   29.8518    [  0.0723  ]
  Q( 30.0000 ) =   35.6327    [  0.2204  ]
  Q( 40.0000 ) =   53.8595    [  0.0704  ]
  Q( 50.0000 ) =   62.2567    [  0.1144  ]
  Q( 60.0000 ) =   72.1338    [  0.1355  ]
```

Figura 28: Sq. residuals Q-stats

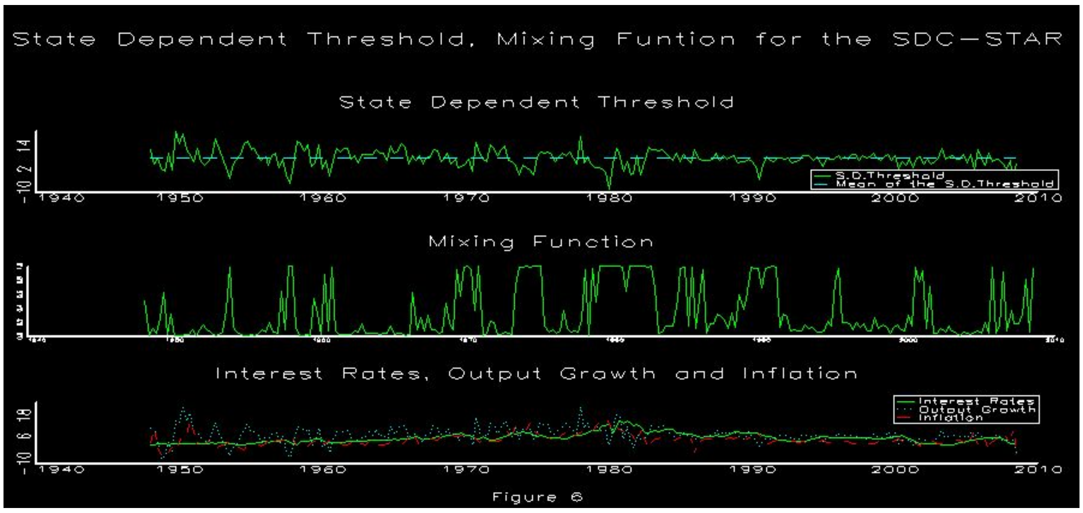
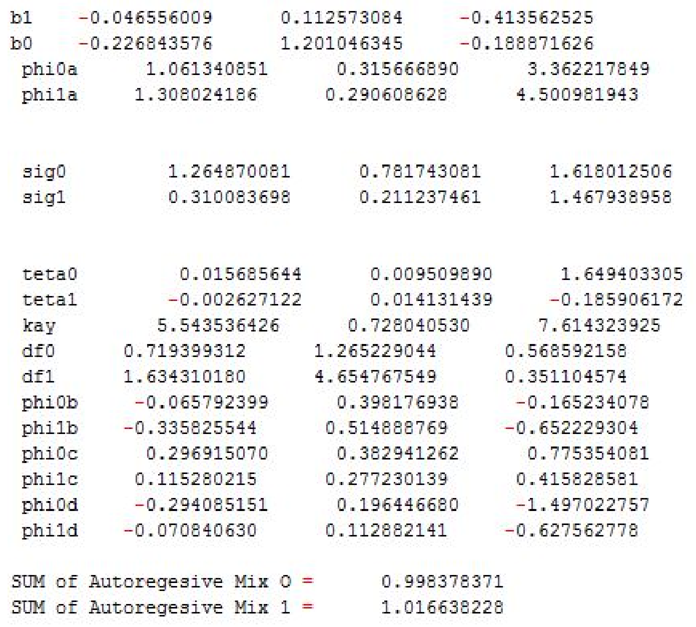
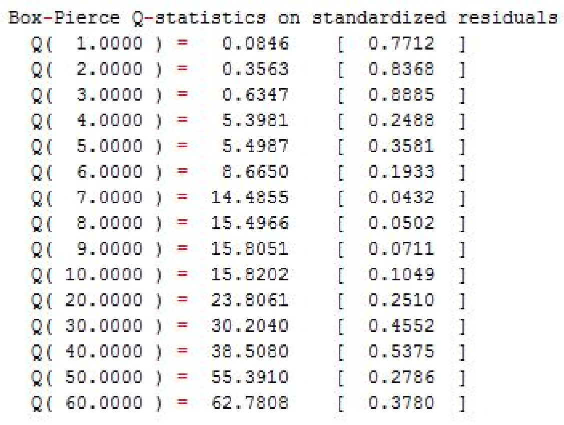


Figura 29: Time variying threshold and mixing function



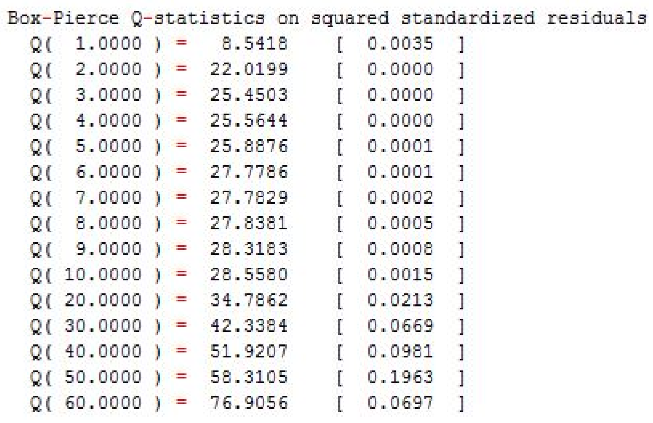
```
b1     -0.046556009       0.112573084       -0.413562525
b0     -0.226843576       1.201046345       -0.188871626
 phi0a       1.061340851       0.315666890       3.362217849
 phi1a      1.308024186       0.290608628       4.500981943


 sig0         1.264870081       0.781743081       1.618012506
 sig1         0.310083698       0.211237461       1.467938958


 teta0          0.015685644       0.009509890       1.649403305
 teta1         -0.002627122       0.014131439      -0.185906172
 kay          5.543536426       0.728040530       7.614323925
 df0       0.719399312       1.265229044       0.568592158
 df1       1.634310180       4.654767549       0.351104574
 phi0b      -0.065792399       0.398176938      -0.165234078
 phi1b     -0.335825544       0.514888769     -0.652229304
 phi0c       0.296915070       0.382941262       0.775354081
 phi1c      0.115280215       0.277230139       0.415828581
 phi0d      -0.294085151       0.196446680      -1.497022757
 phi1d     -0.070840630       0.112882141     -0.627562778

SUM of Autoregesive Mix 0 =       0.998378371
SUM of Autoregesive Mix 1 =       1.016638228
```

Figura 30: Estimation output of the RC-STARX model



```
Box-Pierce Q-statistics on standardized residuals
   Q(  1.0000 ) =    0.0846    [  0.7712  ]
   Q(  2.0000 ) =    0.3563    [  0.8368  ]
   Q(  3.0000 ) =    0.6347    [  0.8885  ]
   Q(  4.0000 ) =    5.3981    [  0.2488  ]
   Q(  5.0000 ) =    5.4987    [  0.3581  ]
   Q(  6.0000 ) =    8.6650    [  0.1933  ]
   Q(  7.0000 ) =   14.4855    [  0.0432  ]
   Q(  8.0000 ) =   15.4966    [  0.0502  ]
   Q(  9.0000 ) =   15.8051    [  0.0711  ]
   Q( 10.0000 ) =   15.8202    [  0.1049  ]
   Q( 20.0000 ) =   23.8061    [  0.2510  ]
   Q( 30.0000 ) =   30.2040    [  0.4552  ]
   Q( 40.0000 ) =   38.5080    [  0.5375  ]
   Q( 50.0000 ) =   55.3910    [  0.2786  ]
   Q( 60.0000 ) =   62.7808    [  0.3780  ]
```

Figura 31: Residuals Q-stats



```
Box-Pierce Q-statistics on squared standardized residuals
  Q(  1.0000 ) =   8.5418    [  0.0035  ]
  Q(  2.0000 ) =  22.0199    [  0.0000  ]
  Q(  3.0000 ) =  25.4503    [  0.0000  ]
  Q(  4.0000 ) =  25.5644    [  0.0000  ]
  Q(  5.0000 ) =  25.8876    [  0.0001  ]
  Q(  6.0000 ) =  27.7786    [  0.0001  ]
  Q(  7.0000 ) =  27.7829    [  0.0002  ]
  Q(  8.0000 ) =  27.8381    [  0.0005  ]
  Q(  9.0000 ) =  28.3183    [  0.0008  ]
  Q( 10.0000 ) =  28.5580    [  0.0015  ]
  Q( 20.0000 ) =  34.7862    [  0.0213  ]
  Q( 30.0000 ) =  42.3384    [  0.0669  ]
  Q( 40.0000 ) =  51.9207    [  0.0981  ]
  Q( 50.0000 ) =  58.3105    [  0.1963  ]
  Q( 60.0000 ) =  76.9056    [  0.0697  ]
```

Figura 32: Sq. residuals Q-stats