Econ. 513, Time Series Econometrics Fall 2001 Chris Sims

Notes and Exercise on Invertibility Criteria[*]

1. Exercise due Thursday, 10/25

The first problem should be very easy, though you may need a computer. The second is extra credit — you don't have to do it.

(1) Determine, as fast as possible, which of the following are fundamental MA operators. State in each case how you reached your conclusion.

(a) $1 + 2L + 3L^2 + 4L^3$

This is easy: since $4 > 1$, this must have a root inside the unit circle and can't be a fundamental MA operator.

(b) $1 + 2L + 3L^2 + 2L^3 + L^4$

There are some things you can say about this without slogging through to find the roots. Since it is symmetric about its L^2 coefficient, the inverse of every root is also a root. Thus if it has *any* roots not on the unit circle, it must have roots inside the unit circle. But this doesn't eliminate the possibility that it has all its roots on the unit circle, which would leave it a fundamental MA operator. And this proves to be the case. Though there may be clever ways to recognize this directly, I just found the roots in Matlab, which turn out to be $.5 \pm i\sqrt{3}/2$, with each root repeated. These are the two complex cube roots of 1, and therefore of course do lie on the unit circle.

(c) $I + \begin{bmatrix} .8 & -.7 \\ .7 & .8 \end{bmatrix} L$

This is also easy, since the determinant of the coefficient on L is $1.13 > 1$, which means this can't be a fundamental MA operator.

(d) $\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} + \begin{bmatrix} 3 & 2 \\ 4 & 1 \end{bmatrix} L + \begin{bmatrix} 4 & 3 \\ 2 & 1 \end{bmatrix} L^2$

Here the determinants of the first and last coefficient matrices match, so the quick check gives no answer. Probably the quickest way to get Matlab to give the answer is to enter the three matrix coefficients into Matlab as `A1, A2, A3,` form `B1=-A1\A2, B2=-A1\B3,` and then ask Matlab for `eig([B1 B2; eye(2) zeros(2)])`. If the operator is to be fundamental, all the eigenvalues of this matrix, which is the coefficient matrix when the system is stacked to first-order form, must be less than one. In fact, three of the four roots are greater than one.

[*]Copyright 2001 by Christopher A. Sims. This document may be reproduced for educational and research purposes, so long as the copies contain this notice and are retained for personal use or distributed free.

(2) Find the fundamental moving average representation corresponding to this non-fundamental one:

$$y(t) = \varepsilon(t) + \begin{bmatrix} 1.1 & 0 \\ 0 & .8 \end{bmatrix} \varepsilon(t-1)$$
$$\operatorname{Var}(\varepsilon(t)) = \begin{bmatrix} 2 & 1 \\ 1 & 1 \end{bmatrix} .$$

This is a hard problem that you have not seen in lecture how to solve. One way to proceed might have been to fill out the covariance matrix of $y(t-1), \dots, y(t-q)$ for a large value of q, using the fact that we can calculate $R_y(t)$ from the representation given. Then the least squares projection formula can be used to calculate the autoregressive coefficients for y, and this autoregressive operator can then be inverted (e.g. by forming the stacked matrix equivalent and summing powers of it, using $(I-A)^{-1} = I + A + A^2 + \dots$.) This of course gives only an approximate answer, depending on how many terms are used on forming the AR projection and how many are summed in taking the inverse.

An exact answer can be obtained by the method described in the new, improved notes on connecting the Jordan form to the properties of a system. The answer in this case is that the fundamental MA representation is

$$y(t) = \eta(t) + \begin{bmatrix} .9091 & .1909 \\ 0 & .8 \end{bmatrix} \eta(t-1)$$
$$\operatorname{Var}(\eta(t)) = \begin{bmatrix} 2.21 & 1 \\ 1 & 1 \end{bmatrix} .$$

In case you are interested, a Matlab program that does the root-flip is below. Note that, as explained in the revised notes, this program works on the assumption that $A(L)A'(L^{-1})$ is the autocovariance function, so it cannot be applied directly to the MA operator in the $I + A_1 L$ form. We have to first find W (by Choleski decomposition, e.g.) such that $W'W = \operatorname{Var}(\varepsilon(t))$ and then use $W' + AW'L$ as the moving average operator.

```
function  Aa=mpolflipr(A,z0)
%function  Aa=mpolflipr(A,z0)
% A is an s-1 order square matrix polynomial with a root at z0.
% Aa has a root at 1/z0 instead, and satisfies Aa(z)*Aa’(1/z)=A(z)*A’(1/z).
% The algorithm follows Rozanov’s Stationary Random Processes, p.46-47.
realsmall=1e-12;
[n,m,s]=size(A);
zv=z0.^[0:s-1];
Az=prodt(A,zv,3,2);
% prodt gives a tensor product, i.e. prodt(A,B,j,k),
% with A n-dimensional and B m-dimensional, yields an
% n+m-2 dimensionalmatrix whose elements are sums of
% the products of elements of A and B with the sums
% taken along the j’th and k’th dimensions (which must
% therefore be of the same length).  prodt(A,B,2,1)=A*B
% for 2d matrices.
[u,d,v]=svd(Az);
if ~(abs(d(n,n))<realsmall)
   error(sprintf(’p(z0)=%g~=0’,d(n,n)))
else
   Aa=prodt(A,v,2,1);
   Aa=permute(Aa,[1 3 2]);
   Aa(:,n,:)=prodt(Aa(:,n,:),toeplitz([1 zeros(1,s-1)],1../zv),3,1);
   Aa(:,n,:)=...
      (Aa(:,n,:)-conj(z0)*cat(3,zeros(n,1,1),Aa(:,n,1:end-1 )))/abs(z0);
end
% Steps below make Aa real, if possible (i.e. if Aa(:,:,1)*Aa(:,:,1)’
% is real.) These steps are never invoked if A and z0 are both real.
if norm(imag(Aa(:,:,1)))>n*realsmall
   sig=Aa(:,:,1)*Aa(:,:,1)’;
   if ~(norm(imag(sig))>n*realsmall)
      sig=real(sig);
      [w,err]=chol(sig);
      if err==0
         Aa=prodt(Aa,Aa(:,:,1)’/w,2,1);
         Aa=permute(Aa,[1 3 2]);
         Aa=real(Aa);
      end
   end
end
```