Bootstrap VAR

`bootstrap_VAR.m`

by Jaromir Benes

15 March 2014

Summary

Bootstrap is a simple yet powerful method to assess sampling uncertainty in the estimated characteristics of, or simulation results based on, parametric models, such as VARs or SVARs. Resample from an estimated VAR object, plot bootstrapped histograms for the estimated coefficients and the VAR autocorrelation function, and generate confidence intervals for parameter uncertainty in out-of-sample simulations.

Contents

1 Clear Workspace

```
clear;
close all;
clc;
```

2 Load Estimated VAR and Dates

Load the estimated VAR object and the VAR data, which also include historical residuals [1]. Load dates defining the start and end of the historical data sample [2].

```
load estimate_simple_VAR.mat v vd; [1]
load read_data.mat startHist endHist; [2]
```

3 Create 500 Bootstraped Data Sets

Resample from the estimated residuals using "wild" bootstrap to generate a total of 500 random data sets that [4]. "Wild" bootstrap is more robust to heteroscedasticity in the estimated residuals than the regular (Efron) bootstrap. Note that the data are resampled not from `startHist`, but from `startHist+p`, where `p` is the order of the VAR [3], and hence also the number of initial conditions that need to be obtained from the input database. Attempts to resample from `startHist` would result in `NaN` initial conditions.

The output database, `bd`, contains variables and residuals, each with 500 columns [5], corresponding to 500 draws from the empirical distribution described by the VAR model. The database will be subsequently used to estimate 500 VAR parameterizations.

The total number of draws, 500, is rather small to keep the execution of this tutorial file fast. Increase `N` to obtain a larger bootstrapped sample, and more robust results.

```
N = 500;

rng(0);

p = get(v,'order'); [3]
bd = resample(v,vd,startHist+p:endHist,N,'wild=',true); [4]

disp('Bootstrapped data');
disp(bd); [5]
```

```
Bootstrapped data
     r: [87x500 tseries]
    pp: [87x500 tseries]
    yy: [87x500 tseries]
    mm: [87x500 tseries]
```

4 Estimate 500 VAR Models from Bootstrapped Data

Create an empty VAR object with the same variable names 6. Use the bootstrapped database, where each variable has 500 columns, to estimate 500 parameterizations of the p-th order VAR object 7. The new VAR object with multiple parameterizations is called `vv`. Some of the parameterizations may turn out to be non-stationary; these would not produce well-behaved ACF needed in the next step. To remove all nonstationary parameterization from the VAR object, first get a true-false index indicating which parameterizations are stationary 8, and then use the index to keep only the stationary ones in the object.

```
yList = get(v,'yList');
vv = VAR(yList); 6

vv = estimate(vv,bd,startHist:endHist, ...
    'order=',p,'const=',false); 7
disp(vv);

inx = isstationary(vv); 8
disp('Total number of stable parameterisations');
fprintf('%g out of %g\n',sum(inx),length(inx));

disp('Remove explosive parameterisations');
vv = vv(inx); 9
disp(vv);
```

```
        VAR(2) object: [500] parameterisation(s)
        variable names: 'r' 'pp' 'yy' 'mm'
        instruments: empty
        comment: ''
        user data: empty

Total number of stable parameterisations
500 out of 500
Remove explosive parameterisations
        VAR(2) object: [500] parameterisation(s)
        variable names: 'r' 'pp' 'yy' 'mm'
        instruments: empty
        comment: ''
        user data: empty
```

5 Compare Original and Bootstrapped Transition Matrices

Use the function `get` 10 to retrieve the estimated transition matrices from both the original VAR object, `v`, and the boostrap VAR object, `vv`. The size of the transition matrices in the original VAR object and in the boostrap VAR are, respectively, as follows:

[11] A: $4 \times 4 \times 2$

[12] AA: $4 \times 4 \times 2 \times N$

where 4 is the number of variables, 2 is the order of the VAR (the first page is the coefficient matrix on the first lag, the second page on the second lag), and N is that number of bootstrapped parameterisations after we remove the nonstationary ones.

Compare the transition matrix from the original VAR, v, and the mean calculated across all boostrap transition matrices from vv [14]. The mean is calculated across 4th dimension [13], which is where the individual parameterizations are reported.

```
A = get(v,'A*'); [10]
AA = get(vv,'A*');

size(A) [11]
size(AA) [12]

meanAA = mean(AA,4); [13]

disp(A(:,:,1))
disp(meanAA(:,:,1)) [14]
```

```
ans =
     4     4     2
ans =
     4     4     2   500
    1.5382   -0.0583    0.0310   -0.0028
    0.2041    0.2333    0.0381    0.0012
   -0.4046   -0.8197    0.2376   -0.0543
   -3.9568    2.1695   -0.1406    0.2290
    1.5009   -0.0619    0.0308   -0.0025
    0.1920    0.2048    0.0404    0.0005
   -0.4349   -0.8467    0.2053   -0.0563
   -3.9437    2.2422   -0.1379    0.2009
```

6 Compare Original and Bootstrapped Autocorrelations

Calculate the autocovariance and autocorrelation functions for the original and the boostrap VAR objects [15]. The size of the ACF matrices is as follows:

[16] C and R: $4 \times 4 \times 2$

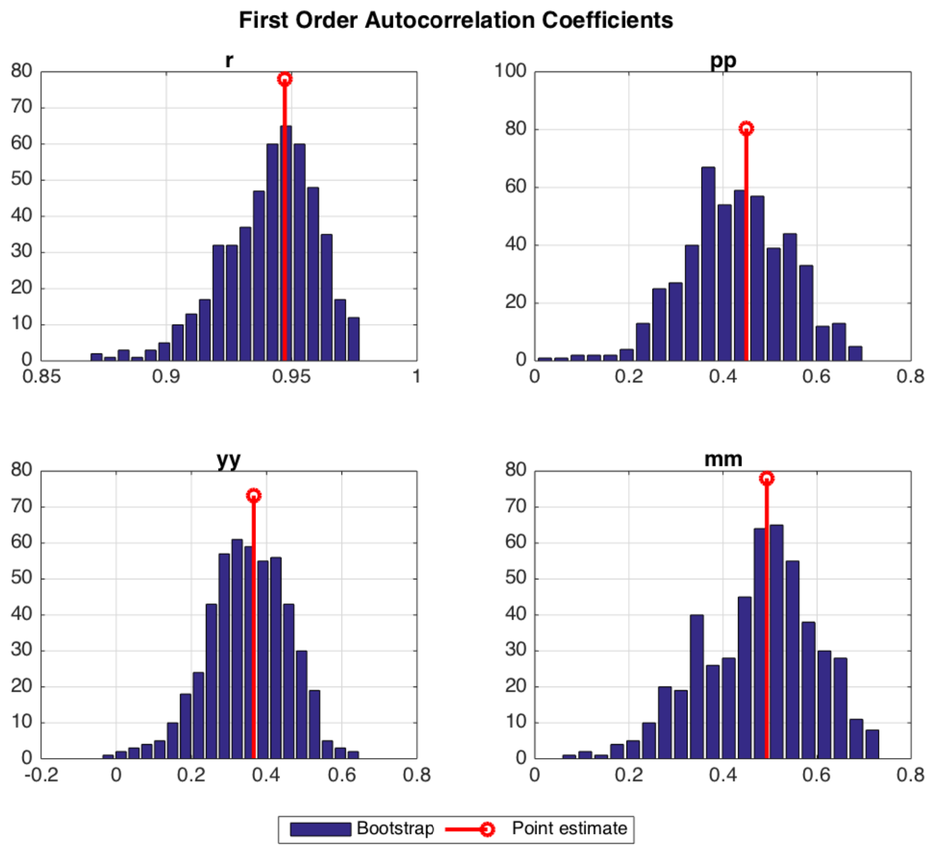
[17] CC and RR: $4 \times 4 \times 2 \times N$

where 4 is the number of variables, 2 relates to the order up to which we request the ACF (the option `'order='` is 1, which means the contemporaneous and first-order covariances and correlations will be returned), and N is the number of parameterizations in the bootstrapped VAR object, `vv`.

Plot first-order autocorrelation coefficient for each variable. The first-order autocorrelation coefficient for the i-th variable are found in `R(i,i,2)` or `RR(i,i,2,:)` [18].

```
[C,R] = acf(v,'order=',1); [15]
[CC,RR] = acf(vv,'order=',1);

size(C) [16]
size(CC) [17]

figure();
for i = 1 : 4
    subplot(2,2,i);
    Ri = R(i,i,2); [18]
    RRi = RR(i,i,2,:);
    [y,x] = hist(RRi(:),20);
    bar(x,y);
    hold all;
    stem(Ri,1.2*max(y),'color','red','linewidth',2);
    grid on;
    title(yList{i},'interpreter','none');
end

grfun.bottomlegend('Bootstrap','Point estimate');

grfun.ftitle('First Order Autocorrelation Coefficients');
```

```
ans =
     4     4     2
ans =
     4     4     2   500
```



7 Simulate Data Out Of Sample

Simulate the historical data 3 years into the future [19], using first the original VAR, `v` [20], and then the bootstrapped VAR, `vv` [21]. The bootstrapped VAR has N different parameterizations, and so the output database, `ff`, will contain series with N columns each [22]. The initial condition for the simulations are though the same for all these simulations.

```
startSim = endHist + 1; [19]
endSim = endHist + 12;

f = forecast(v,vd,startSim:endSim,'meanOnly=',true); [20]
ff = forecast(vv,vd,startSim:endSim,'meanOnly=',true); [21]

disp(ff); [22]
```

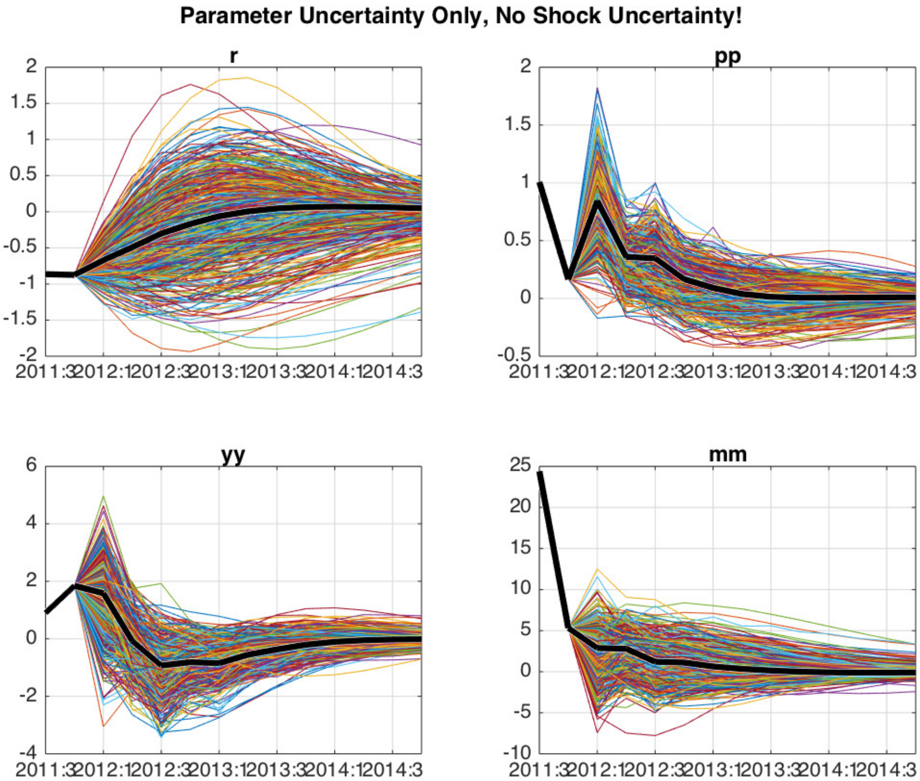
```
     r: [14x500 tseries]
    pp: [14x500 tseries]
```

```
        yy: [14x500 tseries]
        mm: [14x500 tseries]
     res_r: [12x500 tseries]
    res_pp: [12x500 tseries]
    res_yy: [12x500 tseries]
    res_mm: [12x500 tseries]
```

8 Plot Confidence Intervals for Parameter Uncertainty

Plot all N simulated paths from the bootstrapped VAR [23] against the point simulation from the original VAR. Use a little trick to make sure the point simulation is clearly visible: first, plot it as a very thick white line [24], and then again as a somewhat thinner (but still fairly thick) black line [25].

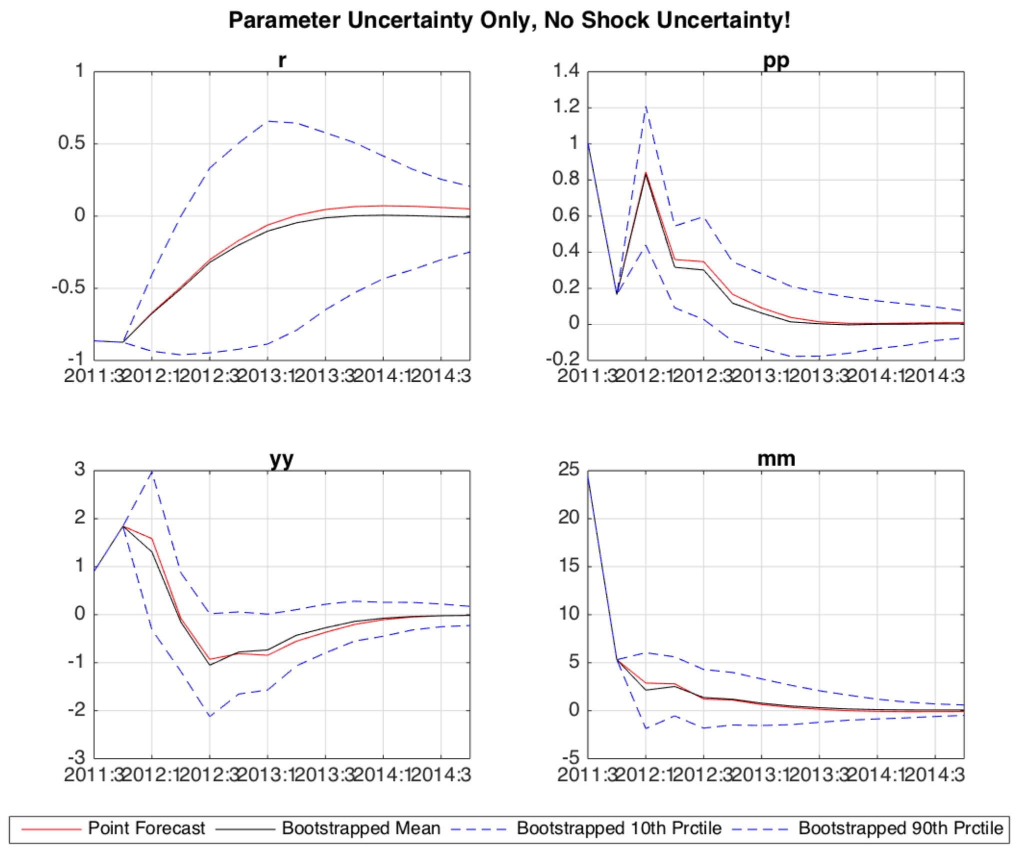
```
plotRng = startSim-2 : endSim;

figure();
for i = 1 : 4
    subplot(2,2,i,'box','on');
    hold all;
    name = yList{i};
    plot(plotRng,ff.(name)); 23
    plot(plotRng,f.(name),'color=','white','lineWidth=',10); 24
    plot(plotRng,f.(name),'color=','black','lineWidth=',3); 25
    title(name);
    grid on;
end

grfun.ftitle('Parameter Uncertainty Only, No Shock Uncertainty!');
```



Plot the point simulation against the mean [26], 10-th and 90-th [27] percentiles computed from the bootstrapped simulations. And remember that the simulations only show parameter uncertainty, and do not include future shock uncertainty.

```
figure();
for i = 1 : 4
    subplot(2,2,i,'box','on');
    hold all;
    name = yList{i};
    h = plot(plotRng,[ ...
        f.(name), ...
        mean(ff.(name),2), ... 26
        pctile(ff.(name),[10,90],2), ... 27
        ]);
    set(h,{'color'},{'red';'black';'blue';'blue'}, ...
        {'lineStyle'},{'-';'-';'--';'--'});
    title(name);
    grid on;
```

```
end

grfun.bottomlegend('Point Forecast','Bootstrapped Mean', ...
    'Bootstrapped 10th Prctile','Bootstrapped 90th Prctile');

grfun.ftitle('Parameter Uncertainty Only, No Shock Uncertainty!');
```



9 Help on IRIS Functions Used in This File

Use either `help` to display help in the command window, or `idoc` to display help in an HTML browser window.

```
help VAR
help VAR/VAR
help VAR/resample
help VAR/estimate
help VAR/get
help VAR/isstationary
```

```
help VAR/mean
help VAR/forecast
help grfun/ftitle
```