Contents

1 FIGURE 3 – BACCALA ET AL. (2016) DTF: UNIFIED ASYMPTOTIC THEORY

DESCRIPTION:

Routine **figure3_example1_dtf_qqplots_ns50_500.m** publish

Linear bivariate VAR(2) model

LA Baccala, DY Takahashi, K Sameshima (2016) Directed Transfer Function: Unified Asymptotic Theory and Some of its Implications. *IEEE Transactions on Biomedical Engineering* **PP**.

```
http://dx.doi.org/10.1109/TBME.2016.2550199
```

Example 1: Simple example VAR(2) without feedback

Contents

Function for quantile plots

```
function [] = figure3_example1_dtf_qqplots_ns50_500(m, ns, nDiscard, metric,
   pairs, connectivity);
```

```
% function [res, resa] = example1_dtf_qqplots(m, ns, nDiscard, metric, pairs,
   connectivity);
%
% input: m - number of simulation
%        ns - data length
%        nDiscard - initial discarded points
%        metric - DTF type metric
%        pairs - pair of variables
%        connectivity - 0: not connected; and 1: connected
%
% output:  res       - results of simulation
%          resa
```

Generate Fig. 3a quantile plots

```
format compact
if nargin < 1 || ~exist('m','var') || isempty(m), m = 2000; end;
if nargin < 2, nd = [50 500]; end;
if nargin < 3, nDiscard = 2000; end;
if nargin < 4, metric = 'euc'; end;
```

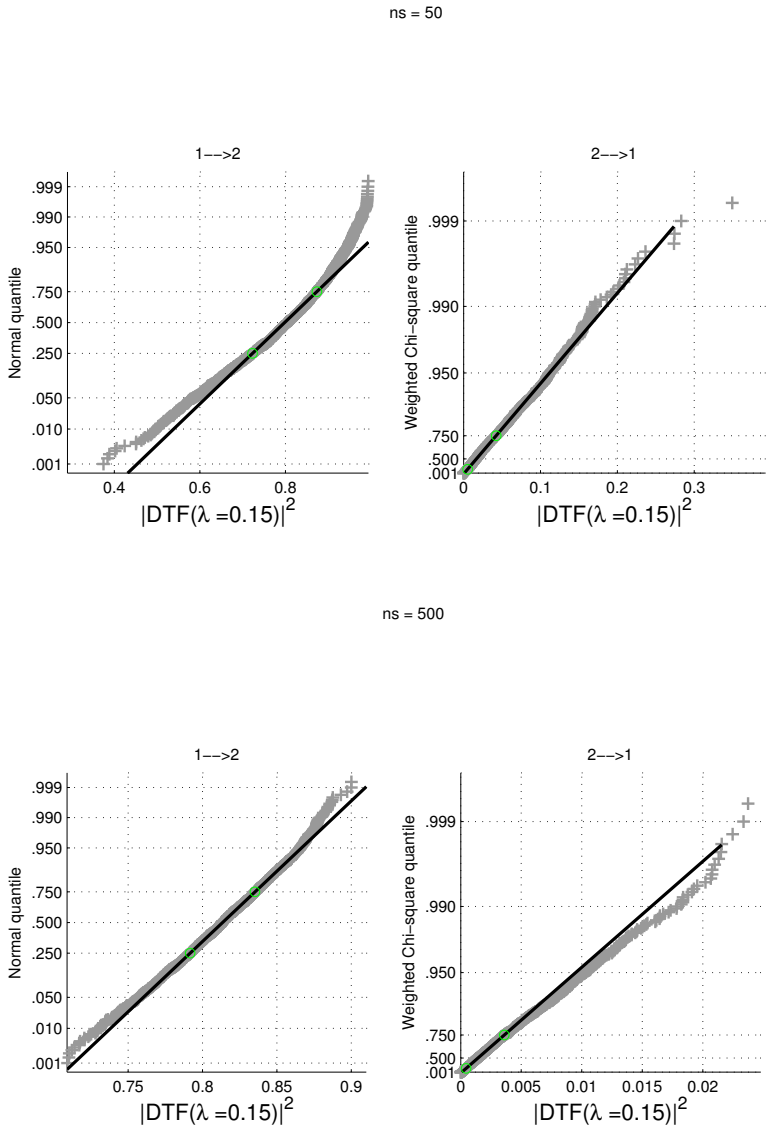
```
pairs = [[2,1]; [1,2]];
connectivity = [1 0]; % 0 = not connected; 1 = connected.

kFreq = 4; % Fourth frequency, i.e. $\lambda = 0.15$.
strFreq = '0.15';

for ns = nd,
```

```
   disp(' ');
   disp(['m = ' int2str(m) '; ns = ' int2str(ns) '; nDiscard = ' ...
      int2str(nDiscard) '; metric = ' metric '.'])

   % Performing m Monte Carlo simulations
   [res, resa] = example1_dtf_monte_carlo(m, ns, nDiscard, metric);

   [h] = qqplots2(res, resa, pairs, connectivity, 'DTF', kFreq, '0.15');

   set(h,'Name', ['[Asymptotic DTF] Fig 3. Example 1 DTF Q-Q plots, ' ...
      'ns = ' int2str(ns)])

   [ax,h1] = suplabel(['ns = ' int2str(ns)]  ,'t');
   if ~is_octave, snapnow; end;
```

```
m = 2000; ns = 50; nDiscard = 2000; metric = euc.
```

```
m = 2000; ns = 500; nDiscard = 2000; metric = euc.
Ao(:,:,1) =
          1.34             0
          0.50         -0.50
Ao(:,:,2) =
         -0.90             0
             0             0
R =
         10.75          3.28          7.59          1.03
          3.28          2.73          3.74          0.27
          7.59          3.74         10.75          3.28
          1.03          0.27          3.28          2.73
                                Using Nuttall-Strand algorithm.
iter  100
iter  200
iter  300
iter  400
iter  500
iter  600
iter  700
iter  800
iter  900
iter 1000
iter 1100
iter 1200
iter 1300
iter 1400
iter 1500
iter 1600
iter 1700
```

```
iter 1800
iter 1900
iter 2000
--------------------------------------------------------------------------
               Original DTF and asymptotic statistics
==========================================================================
qqnorm
qqChi2
```



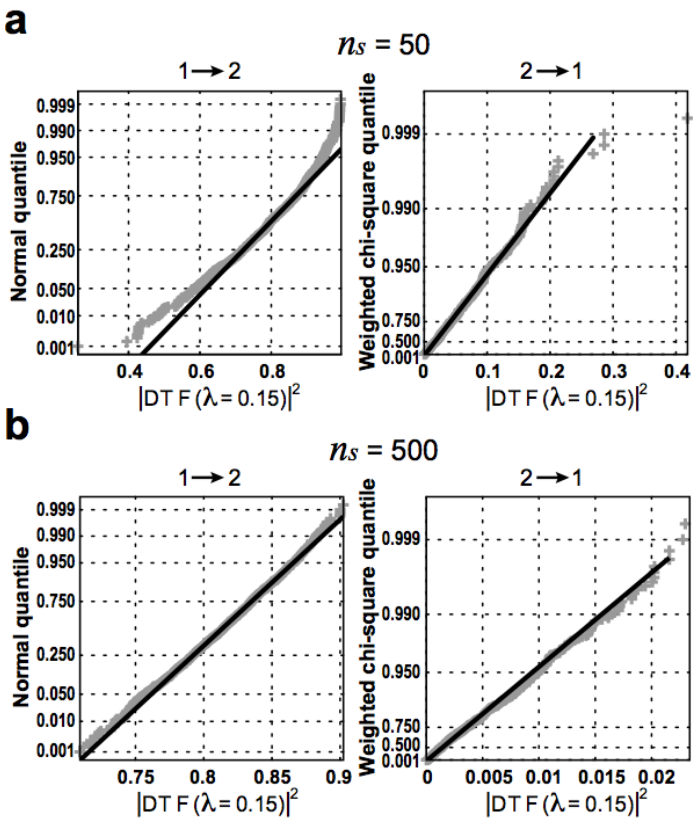
Uncomment the command line bellow to generate an eps output file

```
%eval(['print -depsc2 -painters Fig3_example1_dtf_qqplots_ns' int2str(ns) '.eps
    '])
```

```
end;
```

Firgure 3 from article, Baccala et al (2016)

Figure 3, from article.




```
%=========================================================================
```

```
function [res, resa] = example1_dtf_monte_carlo(m, ns, nDiscard, metric)
% Perform Monte Carlo simulation.
%
% function [res, resa] = example1_monte_carlo(m = 10000, ns = 20000,
%                        nDiscard = 2000, metric = 'euc')
%
% input: m - number of simulation
%        ns - data length
%        nDiscard - initial discarded points
%        metric - type DTF
%
% output:  res   - results used by qqplots.m
%          resa  -

maxp = 2;  IP = 2; % max IP
nf = 10;           % Number of frequency points.
                   % {0,0.05,0.1,0.15,0.2,0.25,0.3,0.35,0.4,0.45}

n = 2;             % Number of variables?

if nargin < 3,
   nDiscard = 2000;
   metric = 'euc';
end;
[Ao, eo] = example1();
```

```
Ao
R = auto_theo(Ao,eo)

sumA = 0;
sume = 0;
res = zeros(m,n,n,nf);
alg = 1;
criterion = 5;

blancos = '                                        ';
disp([blancos 'Using Nuttall-Strand algorithm.'])
flgRandomize = 1;

for i = 1:m,       % m simulations
   if mod(i,100) == 0, fprintf('iter %4.0f\n', i); end;

   % Generate data sample.
   data = fbaccala2016_example1( ns, nDiscard, flgRandomize );
%    data = ar_data_example1(Ao, eo, ns, nDiscard); % Ao & eo are truth values
%    %                                               % of the VAR model

   % MVAR estimation:
   [du,e,A,pb,B,ef,eb,vaic,Vaicv] = mvar(data,IP,alg,criterion);
   %    alg=1;       Nutall-Strand
   %    criterion=5; Fixed IP
   %    maxIP = IP;  IP value itself
   res(i,:,:,:) = abs(dtf_alg_A(A, e, nf, metric));

   sumA = sumA + A;
   sume = sume + e;
end;

% Average A & e
Am = sumA/m;
em = sume/m;

resa = zeros(4,n,n,nf);
alpha = 0.05;

data = ar_data_example1(Ao, eo, ns, nDiscard);

c = asymp_dtf_theo(data, Am, em, nf, metric, alpha); % Simulation average
%c = asymp_dtf_theo(data, Ao, eo, nf, metric, alpha);  % Theoretical values

resa(1,:,:,:) = c.dtf;             %
resa(2,:,:,:) = sqrt(c.varass1);  % r[1] == std
resa(3,:,:,:) = c.patden*ns;      % r[6] == patden
resa(4,:,:,:) = c.patdf;          % r[5] == patdf

% eval(['save fig3_monte_carlo_example1_ns' int2str(ns) '_m' int2str(m) '_'
   date])

%%=========================================================================
function [A,e]= example1(dummy)

e = eye(2);
```

```
A = zeros(2,2,2);
A(1,1,1) = 2*0.95*cos(pi/4);
A(1,1,2) = -(0.95)^2;
A(2,1,1) = 0.5;
A(2,2,1) = -0.5;

%%=========================================================================
function c = ar_data_example1(A, er, m, nDiscard)
% function c = ar_data_example1(A, er, m, nDiscard)
%    Simulate ar-model from A matrix
%
%       Input:
%         A(n, n, p) - AR model (n - number of signals, p - model order)
%         er(n) - variance of innovations
%         m - length of simulated time-series
%
%       Output:
%         data(n, m) - simulated time-series
%

if ndims(A) == 2,
   [n,du]=size(A);
   p=1;
elseif ndims(A) == 3,
   [n du p] = size(A);
else
   error('A matrix dimension > 3.');
end;

randn('state', sum(100*clock))

N = m + nDiscard + p;

% Variables initialization
ei = randn(2,N);
x1 = zeros(1,N);
x2 = zeros(1,N);

for t=1:2,
   x1(t) = randn(1); x2(t) = randn(1);
end;

for t = 3:N,
   x1(t) = 2*0.95*cos(pi/4)*x1(t-1) - (.95)^2*x1(t-2) + ei(1,t);
   x2(t) = 0.5*x1(t-1) - 0.5*x2(t-1) + ei(2,t);
end;

y = [x1' x2']; % data must be organized column-wise
c = y(nDiscard+1:nDiscard+m,:)';
```

```
        Ao(:,:,1) =
          1.34            0
          0.50        -0.50
Ao(:,:,2) =
         -0.90            0
             0            0
R =
```

```
         10.75          3.28          7.59          1.03
          3.28          2.73          3.74          0.27
          7.59          3.74         10.75          3.28
          1.03          0.27          3.28          2.73
                                Using Nuttall-Strand algorithm.
iter  100
iter  200
iter  300
iter  400
iter  500
iter  600
iter  700
iter  800
iter  900
iter 1000
iter 1100
iter 1200
iter 1300
iter 1400
iter 1500
iter 1600
iter 1700
iter 1800
iter 1900
iter 2000
------------------------------------------------------------------------
               Original DTF and asymptotic statistics
========================================================================
qqnorm
qqChi2
```