Contents

1 **FIGURE 12 – BACCALA ET AL. (2016) DTF: UNIFIED ASYMPTOTIC THEORY** 1
Function for quantile plots 1
Generate Figure 4 quantile plots 1
Firgure 12 from article, Baccala et al (2016) 3

FIGURE 12 – BACCALA ET AL. (2016) DTF: UNIFIED ASYMPTOTIC THEORY

DESCRIPTION:

Routine **figure12_example4_dtf_qqplots_ns50_500.m** publish

Linear trivariate VAR(1) 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 4: Reachability model VAR(1)

Contents

- Function for quantile plots
- Generate Figure 4 quantile plots
- Firgure 12 from article, Baccala et al (2016)

Function for quantile plots

```
function [] = figure12_example4_dtf_qqplots_ns50_500(m, nd, nDiscard, metric,
   pairs, connectivity);
```

```
% Example 4 - Generate quantile plots from dtf estimate.
%
% function [res, resa] = example3_dtf_qqplots(m, nd, nDiscard, metric, pairs,
   connectivity);
%
% input: m - number of simulation
%        nd - data lengths
%        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 Figure 4 quantile plots

```
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 = 'info'; end;
```

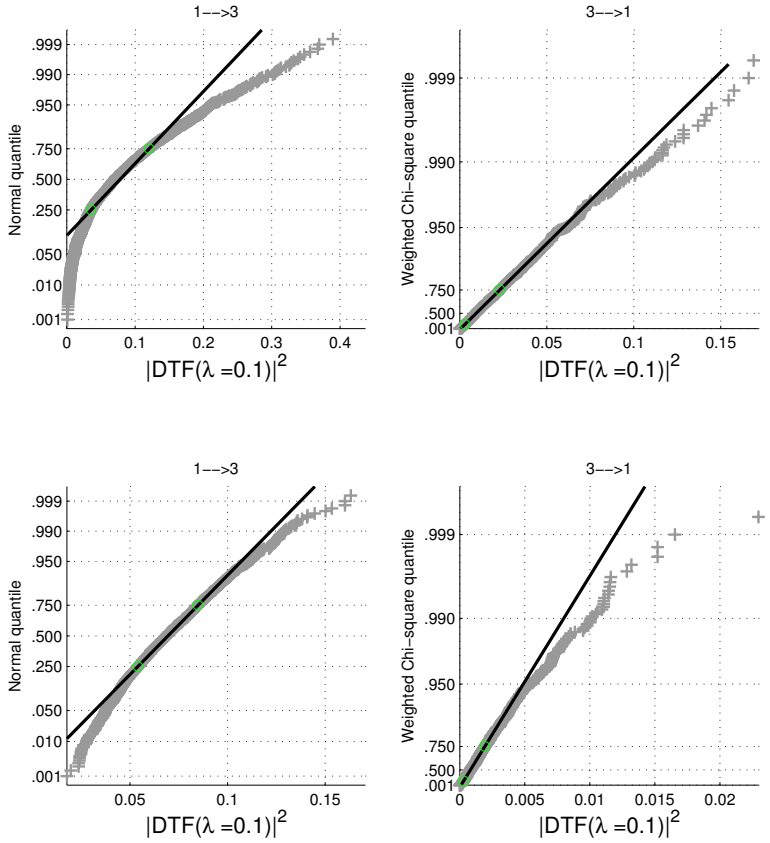
```
if nargin < 5 pairs = [[3,1]; [1,3]]; end;
if nargin < 6,
   connectivity = [1 0]; % 0 = not connected; 1 = connected.
end;

metric = 'euc';
kFreq = 3; % Take the second frequency.
strFreq = '0.1';
k = length(nd); if k > 2, nd=nd(1:2); end;

for ns = nd,
```

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

   % Performing m Monte Carlo simulations
   [res, resa] = example4_dtf_monte_carlo(m,ns,nDiscard,metric,kFreq);

   [h] = qqplots2(res,resa,pairs,connectivity,'DTF',kFreq,strFreq);
   set(h,'Name', ['[Asymptotic DTF] Fig 12. Example 4 DTF Q-Q plots, ' ...
                  'ns = ' int2str(ns)])
   if ~is_octave, snapnow; end;
```

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

```
m = 2000; ns = 500; nDiscard = 2000; metric = euc.
pr_ =
    metric: 'euc'
      fixp: 1.00
      maxp: 1.00
        ss: 0
        nf: 10.00
         v: 0
R =
          1.33          0.00          0.24
          0.00          1.33          0.24
          0.24          0.24          1.33
Ao =
          0.35          0.35             0
         -0.35          0.35             0
             0          0.50             0
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

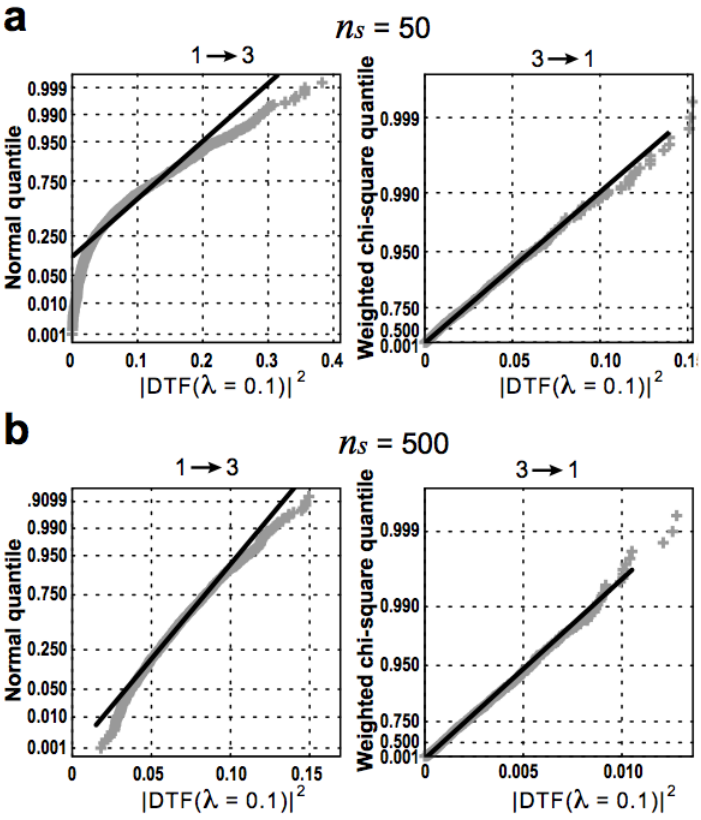
```
  eval(['print -depsc2 -painters Fig12_example4_dtf_qqplots_ns' ...
        int2str(ns) '.eps'])
```

```
end;
```

Firgure 12 from article, Baccala et al (2016)

Figure 12, from article.

```
function [res, resa] = example4_dtf_monte_carlo(m, ns, nDiscard, metric, nf)
%Perform Monte Carlo simulation.
%
```




```
%function [res, resa] = example6_monte_carlo(m = 10000, ns = 20000,
%                       nDiscard = 2000, metric = 'euc')
%
% input: m - number of simulation
%        ns - data length
%        nDiscard - initial discarded points
%        metric - type PDC or DTF
%        nf - number of frequency
%
% output:  res       - results used by qqplots.m
%          resa      -

if nargin == 0,
   m=2000
   ns = 50
   nDiscard = 2000;
   metric = 'info';
   nf = 5;
end;
global pr_

pr_.metric = metric   % Asymptotic statistics metric
pr_.fixp = 1;         % True
pr_.maxp = 1;  IP = 1;% max IP
pr_.ss = 0;           % False
pr_.nf = 10;           % Number of frequency points.
pr_.v = 0;            % False
```

```
n = 3;                  % Number of variables?
if nargin < 3,
   nDiscard = 2000;
end;
[Ao, eo] = model2();
R = auto_theo(Ao,eo)
sumA = 0;
sume = 0;
Ao
res = zeros(m,n,n,pr_.nf);

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_example4(ns, nDiscard, flgRandomize); % Ao & eo are
      truth values
   % of the VAR model
   % MVAR estimation:
   [du,e,A,pb,B,ef,eb,vaic,Vaicv] = mvar(data,IP,1,5);
   %    alg=1;       Nutall-Strand
   %    criterion=5; Fixed IP
   %    maxIP = IP;  IP value itself
   res(i,:,:,:) = abs(dtf_alg_A(A, e, pr_.nf,pr_.metric));

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

   pr_.v = 0; %False;

end;

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

resa = zeros(4,n,n,pr_.nf);

data = ar_data_example6(Ao, eo, ns, nDiscard);
alpha = 0.05;

%c = asymp_dtf3_theo(data, Am, em, pr_.nf,pr_.metric,alpha);
c = asymp_dtf_theo(data, Ao, eo, pr_.nf,pr_.metric,alpha);

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

% eval(['save fig12_monte_carlo_example4_' date])

%%=========================================================================
function [A,e]= model2(dummy)
```

```
% Adapted from Baccala & Sameshima 2001
e = eye(3);
A = zeros(3,3,1);
A(1,1,1) = 0.25*sqrt(2);
A(1,2,1) = 0.25*sqrt(2);
A(2,1,1) = -0.25*sqrt(2);
A(2,2,1) = 0.25*sqrt(2);
A(3,2,1) = 0.5;

%%=========================================================================
function c = ar_data_example6(A, er, m, nDiscard)
% function c = ar_data(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;
x1=zeros(1,N); x2=zeros(1,N);x3=zeros(1,N);
% SZ=[1 5 .30;5 100 2;.30 2 1];SZ=chol(SZ);
ei=randn(3,N);
% ei=SZ'*ei;

for t=1:2,
   x1(t)=randn(1); x2(t)=randn(1);x3(t)=randn(1);
end;

for t=2:N,
   x1(t) = 0.25*sqrt(2)*x1(t-1) + 0.25*sqrt(2)*x2(t-1) + ei(1,t);
   x2(t) = -0.25*sqrt(2)*x1(t-1) + 0.25*sqrt(2)*x2(t-1) + ei(2,t);
   x3(t) = 0.5*x2(t-1) + ei(3,t);
end;

y=[x1' x2' x3']; % data must be organized column-wise
c=y(nDiscard+1:nDiscard+m,:)';
```

```
        pr_ =
    metric: 'euc'
      fixp: 1.00
      maxp: 2.00
```

```
        ss: 0
        nf: 10.00
         v: 0
R =
          1.33          0.00          0.24
          0.00          1.33          0.24
          0.24          0.24          1.33
Ao =
          0.35          0.35             0
         -0.35          0.35             0
             0          0.50             0
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
```