```
data=xlsread('C:\Users\***.xlsx','returns','B2:H**');
returns=data(:,1);

T=length(returns);
i=0;
for ii=1:5;
    jj=0;
    Mdl = arima('ARLags',1:ii,'Variance',egarch(1,1));
    Mdl.Distribution = 't';
    [~,EstParam,logL]=estimate(Mdl,returns);
    nparams = sum(any(EstParam))
    [aic,bic]=aicbic(logL,nparams,T);
    i=i+1;
    crit(i,1)=aic;
    crit(i,2)=bic;
    crit(i,3)=ii;
    crit(i,4)=jj;
end

T=length(returns);
for jj=1:5;
    ii=0;
    Mdl = arima('MALags',1:jj,'Variance',egarch(1,1));
    Mdl.Distribution = 't';
    [~,EstParam,logL]=estimate(Mdl,returns);
    nparams = sum(any(EstParam))
    [aic,bic]=aicbic(logL,nparams,T);
    i=i+1;
    crit(i,1)=aic;
    crit(i,2)=bic;
    crit(i,3)=ii;
    crit(i,4)=jj;
end

Mdl = arima('ARLags',1:5,'MALags',1:5,'Variance',egarch(1,1));
Mdl.Distribution = 't';
EstMdl = estimate(Mdl,returns);
[res,v,logL] = infer(EstMdl,returns);
residuals=res./sqrt(v);

%t
U = tdis_cdf(residuals,EstMdl.Distribution.DoF);

%norm
U = normcdf(residuals,mean(residuals),sqrt(var(residuals)));

%skew t
pd=fitdist(residuals,'tLocationScale');
lower = [2, -0.99];
upper = [Inf, 0.99 ];
theta0 = [pd.nu;0];
options = optimset('Display','off','TolCon',10^-12,'TolFun',10^-4,'TolX',10^-↙
6,'DiffMaxChange',Inf,'DiffMinChange',0,'Algorithm','active-set');
theta1 = fmincon('skewtdis_LL',theta0,[],[],[],[],lower,upper,[],options,residuals);
```

```
outSKEWT = theta1'
hess=hessian_2sided('skewtdis_LL',theta1,residuals);
stderrors = sqrt(diag((hess)^(-1)))
U = skewtdis_cdf(residuals,outSKEWT(1,1),outSKEWT(1,2));

%test U
[h1_U,p1_U,~,~]=lbqtest(U-mean(U),'Lags',1:10);
[h2_U,p2_U,~,~]=lbqtest((U-mean(U)).^2,'Lags',1:10);
[h3_U,p3_U,~,~]=lbqtest((U-mean(U)).^3,'Lags',1:10);
[h4_U,p4_U,~,~]=lbqtest((U-mean(U)).^4,'Lags',1:10);
h_U=[h1_U',h2_U',h3_U',h4_U'];
p_U=[p1_U',p2_U',p3_U',p4_U']

[h_res,p_res]=lbqtest(residuals,'lags',1:10)
[h_residuals,p_residuals]=archtest(residuals,'lags',1:10)
[ks_h,ks_p]=kstest(U,[U,unifcdf(U,0,1)])

%Copula
options = optimset('Algorithm','interior-↙
point','Display','iter','Hessian','bfgs','MaxFunEvals',1000);
options = optimset(options,'FinDiffType','central','MaxIter',1500,'TolCon',10^-↙
12,'TolFun',10^-6,'TolX',10^-6);

%import U data
data=xlsread('C:\Users\***.xlsx','U','B2:D***');
u=data(:,2);v=data(:,3);
%lower and upper bounds use [-25,25] in order to be consistent.
% Note: Patton uses [-5,5], sometimes [-25,25]. Cadidios uses others[-40,40].

%Gaussian Static
kappabarN = corrcoef12(norminv(u),norminv(v));
LL1 = NormalCopula_CL(kappabarN,[u,v]);
LL_norm=LL1
AIC_norm_static=LL1*2+1*2
%one can correct for small sample
%k=length(kappabarN)
%T=length(data)
%2*LL1+2*k+((2*k*(k+1))/(T-k-1))

%Gumbel
lower = 1.0001;
upper=5;
theta0 = 2; %Reboredo uses 15, Patton uses 2, candido uses 1.11
[kappabarGL LL2] = fmincon('gumbelCL',theta0,[],[],[],[],lower,[],[],options,[u,v]);
Tail2 = [0,2-2^(1/kappabarGL)]
LL_Gumbel=LL2
AIC_Gumbel=LL2*2+1*2
%one can correct for small sample
%k=length(kappabarGL)
%T=length(data)
%2*LL2+2*k+((2*k*(k+1))/(T-k-1))

%t
kappa1 = corrcoef12(norminv(u),norminv(v));
```

```
lower = [-0.9 , 2.1 ];
upper = [ 0.9 , 100 ];
theta0 = [kappa1;10];
[ kappa4 LL4] = fmincon('tcopulaCL',theta0,[],[],[],[],lower,upper,[],options,[u,v]);
tail4 = ones(1,2)*2*tdis_cdf(-sqrt((kappa4(2)+1)*(1-kappa4(1))/(1+kappa4(1))),kappa4(2)+1)
rho=kappa4(1)
kendall=(2/pi)*asin(kappa4(1))
LL_Student=LL4
AIC_Stduent=LL4*2+2*2
%one can correct for small sample
%k=length(kappa4)
%T=length(data)
%2*LL4+2*k+((2*k*(k+1))/(T-k-1))

%clayton
lower = 0.0001;
theta0 = 1; %Initial value
[kappabarCL LL3] = fmincon('claytonCL',theta0,[],[],[],[],lower,[],[],options,[u,v]);
Tail3 = [2^(-1/kappabarCL),0]
LL_clayton=LL3
AIC_clayton=LL3*2+1*2
%one can correct for small sample
%k=length(kappabarCL)
%T=length(data)
%2*LL3+2*k+((2*k*(k+1))/(T-k-1))

%sym
lower = [0 , 0 ]; %bound of parameter: tail dependence in between [0,1]
upper = [ 1 , 1];
theta0 = [0.25;0.25]; %Patton and Candido use [0.25;0.25], Reboredo uses [0.7;0.1]
[kappabarSJC LL5] = fmincon('sym_jc_CL',theta0,[],[],[],[],lower,upper,[],options,[u,v]);
LL_SJC=LL5
AIC_SJC=LL5*2+2*2
kappabarSJC
%one can correct for small sample
%k=length(kappabarSJC)
%T=length(data)
%2*LL5+2*k+((2*k*(k+1))/(T-k-1))

%ARMA Gaussian Dynamic
kappa0 = [log((1+kappabarN)/(1-kappabarN));0;0]; %initial values for time-varying copulas
lower = -25*ones(3,1);  %Patton use 5*ones(3,1); in theory there are no constraints, but setting loose constraints sometimes helps in the numerical optimisation
upper =  25*ones(3,1);  %Patton use -5*ones(3,1);
[kappa11 LL11] = fmincon('bivnorm_tvp1_CL',kappa0,[],[],[],[],lower,upper,[],options,[u,v],kappabarN);
[LL11, rho] = bivnorm_tvp1_CL(kappa11,[u,v],kappabarN);
LL11
LL11*2+3*2
```

```
%one can correct for small sample
%k=length(kappa11)
%T=length(data)
%2*LL11+2*k+((2*k*(k+1))/(T-k-1))

%Gumbel Dynamic
kappa0=[sqrt(kappabarGL-1);0;0]; %initial values from Patton, Candido uses kappa0=[1
-1 0];
lower = -25*ones(3,1);  %Patton use 5*ones(3,1); %this is the constain for [w,phi,
thi]; % Patton uses this. Candido set as: lower = -25*ones(3,1); upper = 25*ones
(3,1); If use Candido, mean(Tail_dynamics) is far from Tail4
upper =  25*ones(3,1); %Patton use -5*ones(3,1); % in theory there are no
constraints, but setting loose constraints sometimes helps in the numerical
optimisation
[kappa12 LL12] = fmincon('Gumbel_tvp1_CL',kappa0,[],[],[],[],lower,upper,[],options,
[u,v],kappabarGL);
[LL12 kappaGL] = Gumbel_tvp1_CL(kappa12,[u,v],kappabarGL);
Tails_Gumbel=zeros(length(u),2);
for i=1:length(u)
    kappaGumbel=kappaGL(i,:);
    Temp=2-2^(1/kappaGumbel);
    Tails_Gumbel(i,2)=Temp;
    kendall(i)=(kappaGL(i)-1)/(kappaGL(i));
end
LL12
LL12*2+3*2
%one can correct for small sample
%k=length(kappa12)
%T=length(data)
%2*LL12+2*k+((2*k*(k+1))/(T-k-1))

%Student T Dynamic
kappa1 = corrcoef12(norminv(u),norminv(v));
theta0 = [kappa11;kappa4(2)];
%theta0 = [kappa11;5];%time-varying Gaussian as inputs
lower = [-25 -25 -25 2];  % in theory there are no constraints, but setting loose
constraints sometimes helps in the numerical optimisation
upper = [25 25 25 100];
%lower = -25*ones(4,1);  % in theory there are no constraints, but setting loose
constraints sometimes helps in the numerical optimisation
%upper = 25*ones(4,1);
[ kappa14, LL14,exitflag,output,lambda,grad,hessian]= fmincon('tcopulaCL_tvp1_CL_1',
theta0,[],[],[],[],lower,upper,[],options,[u,v],kappa1);
LL14
LL14*2+4*2
%one can correct for small sample
%k=length(kappa14)
%T=length(data)
%2*LL4+2*k+((2*k*(k+1))/(T-k-1))

%Dynamic tail
psi=zeros(length(u),1);
psi(1)=kappa1;
rho(1)=kappa1;
```

```
theta=kappa14;
x = tinv(u(:,1),kappa14(4));
y = tinv(v(:,1),kappa14(4));
```