function [Theta a v invF]=LLoneCIR\_param\_BFGS(para,Y, tau, Delta, dt, nrow,ncol,nparam)

%% initialization of the parameters for CIR model

yorig=Y;

theta=Delta(1)\*(exp(para(1)))/(1+exp(para(1)));

kappa=Delta(2)\*(exp(para(2)))/(1+exp(para(2)));

sigma=Delta(3)\*(exp(para(3)))/(1+exp(para(3)));

lambda=Delta(4)\*(exp(para(4)))/(1+exp(para(4)));

sigmai=Delta(5:end)\*(exp(para(5:end)))/(1+exp(para(5:end)));

Theta=zeros(nparam,1);

for i=1:nparam

 Theta(i)=Delta(i)\*(exp(para(i)))/(1+exp(para(i)));

end

R=eye(ncol);

for i=1:ncol

 R(i,i)=sigmai(i)^2; % eqn 9 (Bellini and Riani 2010), Diagonal GG'

end

%% system matrices initialization

C=theta\*(1-exp(-kappa\*dt)); % eqn 16 (Bellini and Riani 2010), d\_t

F=exp(-kappa\*dt); % eqn 16 (Bellini and Riani 2010), T\_t

A=zeros(1, ncol);

H=A;

%% A and B matrices

for i=1:ncol % System matrices are made for each tau

 AffineG=sqrt((kappa+lambda)^2+2\*sigma^2); % eqn 15 (Bellini and Riani 2010)

 AffineB=2\*(exp(AffineG\*tau(i))-1)/((AffineG+kappa+lambda)...

 \*(exp(AffineG\*tau(i))-1)+2\*AffineG); % eqn 14 (Bellini and Riani 2010)

 AffineA=2\*kappa\*theta/(sigma^2)\*log(2\*AffineG\*...

 exp((AffineG+kappa+lambda)\*tau(i)/2)/((AffineG+kappa+lambda)\*...

 (exp(AffineG\*tau(i))-1)+2\*AffineG)); % eqn 13 (Bellini and Riani 2010)

 A(i)=-AffineA/tau(i);

 H(i)=AffineB/tau(i);

end

%% Kalman Filter

% Step 1 Initializing the state vector

initx=theta; % eqn 66 (Bolder 2001)

initV=sigma^2\*theta/(2\*kappa); % eqn 68 (Bolder 2001)

% Starting values

AdjS=initx;

VarS=initV;

v1=zeros(nrow,ncol);

a1=zeros(nrow,1);

LL=zeros(nrow,1); % log-likelihood vector initialization

invF1=zeros(nrow\*ncol,ncol);

for i=1:nrow

 if isnan(Y(i))

 PredS=C+F\*AdjS; % eqn 75 (Bolder 2001)

 Q=theta\*sigma\*sigma\*(1-exp(-kappa\*dt))^2/(2\*kappa)+sigma\*sigma/kappa...

 \*(exp(-kappa\*dt)-exp(-2\*kappa\*dt))\*AdjS; % eqn 18 (Bellini and Riani 2010)

 VarS=F\*VarS\*F'+Q; % eqn 76 (Bolder 2001)

 % Step 2 Forecasting the Measurement equation

 PredY=A+H\*PredS; % eqn 69 (Bolder 2001)

 VarY=H'\*VarS\*H+R; % eqn 70 (Bolder 2001)

 % Step 3 Updating the inference about the state vector

 PredError=0\*H; % eqn 71 (Bolder 2001)

 PredError\_fwd1=yorig(i,:)-PredY; % eqn 71 (Bolder 2001)

 KalmanGain=0\*H; % eqn 73 (Bolder 2001) %IT IS PUT=0 for NaN

 AdjS=PredS+KalmanGain\*PredError'; % eqn 72 (Bolder 2001)

 VarS=VarS\*(1-KalmanGain\*H'); % eqn 74 (Bolder 2001)

 % Step 5 Construct the likelihood function

 DetY=det(VarY);

 LL(i)=-(ncol/2)\*log(2\*pi)-0.5\*log(DetY);

 else

 PredS=C+F\*AdjS;

 Q=theta\*sigma\*sigma\*(1-exp(-kappa\*dt))^2/(2\*kappa)+sigma\*sigma/kappa...

 \*(exp(-kappa\*dt)-exp(-2\*kappa\*dt))\*AdjS;

 VarS=F\*VarS\*F'+Q;

 % Step 2 Forecasting the Measurement equation

 PredY=A+H\*PredS;

 VarY=H'\*VarS\*H+R;

 % Step 3 Updating the inference about the state vector

 PredError\_fwd1=Y(i,:)-PredY;

 KalmanGain=VarS\*H\*inv(VarY); % Kalman Gain Matrix eqn 73 (Bolder 2001)

 AdjS=PredS+KalmanGain\*PredError\_fwd1';

 VarS=VarS\*(1-KalmanGain\*H');

 % Step 5 Construct the likelihood function

 DetY=det(VarY);

 LL(i)=-(ncol/2)\*log(2\*pi)-0.5\*log(DetY)-0.5\*PredError\_fwd1\*inv(VarY)\*PredError\_fwd1';

 end

 %sumll=-sum(LL);

 v1(i,:)=PredError\_fwd1;

 a1(i,:)=AdjS;

 invF1((i-1)\*ncol+1:i\*ncol,:)=inv(VarY);

end

v=v1(2:end,:);

a=a1(2:end,:);

invF=invF1(ncol+1:end,:);