|  |
| --- |
| NYC |
| Advanced Modeling |
| HW3 |
|  |
| **Sanyueh (Michael) Yao** |
|  |

**Exercise 1:**

**1.1: Let N be a Poisson process with intensity λ>0 with respect to and let B be a BM, both defined on the same probability space. Define the filtrations.  , . What is N’s intensity with respect to these two filtrations.**

* **With respect to filtration: **

|  |  |
| --- | --- |
| From L5, page 17, we get | , whose filtration will be Poisson |
| Hence, we can get | is F-Intensity |
| Because  (1)N and B are independent  (2) | is G-Intensity |
| Conclusion | is G-Intensity |

* **With respect to filtration: **

|  |  |
| --- | --- |
| Refer L6, page 9 , we use the same reasoning for Brownian Bridge and claim | , N is not a M’g under H |
| We mimic the logic from L6 page 10, we define | ,  , |
| Take conditional expectation |  |
| Conclusion | N’s H- intensity is given by |

**1.2: Assume that is a family of independent Poisson process with constant intensities generating the filtration Ft. Define the stopping time . What is the intensity of the {0,1}-valued process . Make sure your intensity vanish after **

|  |  |
| --- | --- |
| From the hint in BB, we approach the following | by means of independence |
| We can find intensity like Shreve’s book p478 | , where intensity vanishes after time |
| Conclusion | The intensity of the {0,1}-valued process would be |

**1.3: Under what measure. is N also has a Poisson process (constant intensity).**

|  |  |
| --- | --- |
|  | **Comment** |
|  | Under ,the intensity is. Therefore, N is still Poisson |
|  | Under , the intensity becomes . Therefore N is NOT Poisson |
|  | Under , the intensity becomes 1. Therefore N is Poisson |
|  | Under , due to the presence of . Therefore N is NOT Poisson |
|  | Under , the intensity becomes 1. Therefore N is Poisson |

**1.4: Let N be a Poisson Process and Let B be a BM. Let denote the first jump time for N. Find the density of the random variable (Hint: compute X’s characteristic function and then use the inversion with prob 4.1 in HW1)**

|  |  |
| --- | --- |
| Characteristic Function of | , |
| Define =Mt |  |
| Verify that  is Martingale (apply Ito’s lemma). Temporary replace here |  |
|  |  |
|  |  |
| From Doob’s optional sampling Them (L5, page 3) | If Mt is Martingale and is any stopping time.  Then is also a Martingale |
| Take limitation | use dominated convergence Thm |
| Two possible scenario of | (1) when :  (2) when : |
| Combine result of two scenario |  |
| Take expectation | , we want the first jump , so |
| Time on both sides (deterministic function) | We know the characteristic function for standard normal is |
| We also know that |  |

**1.5: Let N be a Poisson Process and Let B be a BM. Let , Define the process . Are B and N independent? Are W and N independent?**

* **Bt and Nt are independent**

|  |  |
| --- | --- |
| (From Shreve’s book: thm 11.2.4) If N is Poisson with intensity | is a Martingale |
| Let , then | We want both terms below to be martingale and Let them Xt |
| Goal: Show is a Martingale. Use Ito’s lemma and get | (1) |
| we will get contribution from , so | Take Exponential and get  (2) |
| Continuous component of Xt | (3) |
| We can rewrite (2) and combine (3), we get |  |
|  | . This is Martingale |
| Right now we know is Martingale. Then |  |
| Move out the deterministic part |  |
| Conclusion | From this Generating Function Method, (we have exactly product of generating function). We can conclude Nt and Bt are independent. |

* **Wt and Nt are independent**

|  |  |
| --- | --- |
|  |  |
| Conclusion | That means that Wt is a Brownian Motion  And we just proved that If W is BM and Nt is Poisson Process. They will be independent. |

**Exercise 2 (Jump Diffusion Models):**

**2.1: Express **

|  |  |
| --- | --- |
|  |  |
| Conclusion |  |

**2.2:Find a predictable process such that is a Q Martingale**

|  |  |
| --- | --- |
| From text book thm 11.3.1: To get Martingale from compound Poisson Process:  First: Get the mean of compound process |  |
| We know that |  |
| Thm 11.3.1 | is a Martingale |
| Verify | so this is indeed a Martingale |
| Conclusion | The whole Compensator is , its corresponding integrand with integration from 0 to t should be |

**2.3: The stock price dynamics are defined by . Where is strictly positive adapted process and B is BM under Q. Explain why S is a Q martingale and explain what happens with S at the i’th jump time for N (what happens when N jumps from i-1 to i?**

* **Why S is a Q Martingale.**

|  |  |
| --- | --- |
| Reasons | (1) Since B is Brownian Motion under Q , thus it is a Martingale |
|  | (2)  is Martingale |
| Conclusion | S is a Q Martingale |

* **What happens to S at ith jump time for N**

|  |  |
| --- | --- |
| At time | Given the dynamics (0,1), when N jumps form i-1 to I (to , ). can be obtained by scaling with . We will get |
| Limitation about | To ensure S remains strictly positive, we need to be supported on (-1,). That is the reason why the problem let be a family iid with common density on (-1,∞)  is the relative jump for the ith jump |

**2.4: Solve for S in (0,1):**

**(a) Find explicit expression for St**

|  |  |
| --- | --- |
| From 2.2 | (1) |
| M’s continuous part | (2) |
| Recall that | (3) |
| Combine (1)(2)(3) |  |
| Conclusion |  |

**(b)Find dynamics for the R.P. . Find a,b, and s.t.**

|  |  |
| --- | --- |
| We can let |  |
| From 2.4.(a) we can get | **(1)** |
| By 2.3 | **(2)** |
|  | , where we find its continuous part is  (3) |
| From (1)(2)(3)  So we can get |  |
| Conclusion |  |

**2.5: Refer the expression Construct a measure such **

|  |  |
| --- | --- |
| Based definition above | , Zt is defined under Q. |
| : | is Q Martingale |
| t=0 just like lecture 4 slide 14 (we can remove the conditional expectation and get the following) |  |

**2.6: Find the predicable process and such that ,are both Martingales under the measure defined in the previous question.**

**(a) What is N’s intensity under ?**

**(b) Is N still a Poisson process under ?**

**(c) What is the density function for ,i=1,2….under ?**

|  |  |
| --- | --- |
| PART I:  Combine thm 11.6.5 & 11.6.11 on page 498.  The Radon-Nikodym derivative process Z(t) can be written as |  |
| (Above) it suggests that if Y1,Y2….are not discrete but instead have a common density f(y), then we can change the measure so that Q(t) has intensity and Y1,Y2…have a different density by using the Radon-Nikodym derivative process |  |
| So given we can write change-of-measure Martingale as | e (1)  If we match what we have in 2.4, we can get |
| PART II:  We want to match the jump part of Z & Jump part from 2.4, | (2) |
| From (1) and (2) , we can conclude that | (3)  (4) |
| Combine (3)and (4), we can get | (5)  (6) |
| Verify (5) to see whether that is valid density |  |
| Conclusion |  |
| **(a) What is N’s intensity under ?** | From (5), we get |
| **(b) Is N still a Poisson process under ?** | YES, N still a Poisson process under**.** Where we get (1)  (2)  ,where |
| **(c) What is density function for ,i=1,2….under ?** | From (5), we get |

**2.7:**

**(a) Show Q-characteristic fcn:**

**(b) Show -characteristic fcn:**

**(a) Show**

|  |  |
| --- | --- |
| We know from 2.4 that |  |
| We can rewrite X as | Where , |
| By independence |  |

* PART I: Continuous Part

|  |  |
| --- | --- |
|  |  |

* PART II: Jump Part

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |

* Combine Part I & Part II

|  |  |
| --- | --- |
|  |  |

**(b) Show**

|  |  |
| --- | --- |
| We can still can X as | Where , |
| By independence |  |

* PART I: Continuous Part

|  |  |
| --- | --- |
|  |  |

* PART II: Jump Part

|  |  |
| --- | --- |
|  |  |
|  | From result of 2.1, we get  From result of 2.6 (a), we know that N has intensity under is  From result of 2.6 (c):  If Our goal is get |
|  | We actually can get    , where |
| : combine ref 1,2 for | , where we replay log(y+1)=x 🡪 dy=exp(x)dx |
|  | : we replace , |
|  |  |

* Combine Part I & Part II

|  |  |
| --- | --- |
| is indeed |  |

**2.8: Report ATM call price for for Merton Model**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Solid | Dashed | Long-Dashed |
| T = 0.1 | 2.7578 | 2.6567 | 2.6955 |
| T = 0.5 | 6.3299 | 5.9556 | 6.0877 |
| T = 1 | 8.9849 | 8.4183 | 8.6154 |
| T = 2 | 12.7102 | 11.8857 | 12.1707 |
| T = 5 | 19.9977 | 18.6911 | 19.1416 |

* + - * **Code is attached in the Appendix Section**

**2.9: Create a plot with K on the horizontal axis and on the vertical axis and plot the value of agrees with the outcome of Merton’s model**

****

**2.10: Compute XT using Heston model with jump process under both and **

Gatheral’s book P66 gives the equation:



Where



And C, D are noted in Gatheral’s book.

Alternatively, we can deduct the equations are the same as the ones we deduct below.





In HW2 we have

|  |  |  |
| --- | --- | --- |
|  | **Under Q** | **Under :** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

So we know that

Continuous Part:

****

****

And here

X can be replaced by ****

From 2.7 we know the jump parts. Add them in we will have the current answers.

|  |  |
| --- | --- |
|  |  |
|  |  |

Using Gatheral’s notations we have





**2.11: Transfer the numbers in the Table 5.5 in Gatheral to match in this model**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | K = 80 | K = 90 | K = 100 | K = 110 | K = 120 |
| T = 0.1 | 20.0033 | 10.0831 | 1.6574 | -0.0035 | 0.0056 |
| T = 0.5 | 20.3099 | 11.2119 | 3.8555 | 0.412 | 0.022 |
| T = 1 | 21.0698 | 12.5946 | 5.6859 | 1.4939 | 0.2521 |
| T = 2 | 22.7142 | 14.9603 | 8.5609 | 3.9817 | 1.4647 |
| T = 5 | 27.1213 | 20.5025 | 14.868 | 10.2927 | 6.7812 |

**Exercise 3:**

**3.1: Report that minimized the squared error. (The detail code are includes in the Appendix)**

|  |  |
| --- | --- |
|  | 0.018824, 0.37496, 0.053682, 0.26082, -0.6691 |
| Corresponding minimal squared error | 0.10388 |
| We also noticed that, in an alternative route of calculation, we have the following : |  |
| Side Note | This theoretically should give us a better result. However, because the Matlab implementation of mathematical models are slightly different. Thus it would give us a result with minor bias.  Besides, because producing an alternative route of implementation can be time-consuming, we decide not to take that process. |

**3.2: For the optimal value of , plot showing both implied BS volatility of the call prices in Table 1 and implied BS of the calibrate Heston Model**

|  |  |
| --- | --- |
| T=0.1 |  |
| T=1 |  |
| Comment | For T=1 Heston model results are very close to Table Prices results.  For T=0.1 the differences are slightly bigger. |

**APPENDEX : CODE**

**2.8:**

|  |  |
| --- | --- |
| **Note** | **Code** |
| HW3Q2\_8.m | %hw3\_2\_8.m  clear all;  T=[0.1,0.5,1,2,5];  sig=[0.2,0.2,0.2];  lam=[0.5,1,1];  alpha=[-0.15,-0.07,-0.07];  delta=[0.05,0,0.05];    s0=100;  K=100;    du=1;    c=zeros(5,3);  for i=1:5  for j=1:3  beta = exp(alpha(j)+0.5\*delta(j)\*delta(j))-1;    integ = 0;  for u=0.5:du:100  C = 1i\*u\*(-0.5\*sig(j)\*sig(j)\*T(i)-beta\*lam(j)\*T(i));  D = -0.5\*u\*u\*sig(j)\*sig(j)\*T(i);  E = lam(j)\*T(i)\*(exp(1i\*u\*alpha(j)-0.5\*u\*u\*delta(j)\*delta(j))-1);  psi = exp(C + D + E);  integ = integ + imag(exp(-1i\*u\*log(K/s0))\*psi)/u\*du;  end  Q1=1/2+1/pi\*integ;    integ = 0;  psi = 0;  for u=0.5:du:100  C = 1i\*u\*(0.5\*sig(j)\*sig(j)\*T(i)-beta\*lam(j)\*T(i));  D = -0.5\*u\*u\*sig(j)\*sig(j)\*T(i);  E = lam(j)\*T(i)\*(exp(1i\*(u-1i)\*alpha(j)-0.5\*delta(j)\*delta(j)\*(u-1i)\*(u-1i))-1-beta);  psi = exp(C + D) \* exp(E);  integ = integ + imag(exp(-1i\*u\*log(K/s0))\*psi)/u\*du;  end  Q2=1/2+1/pi\*integ;  c(i,j)=s0\*Q2-K\*Q1;  end  end    c |

**2.9:**

|  |  |
| --- | --- |
| **Note** | **Code** |
| HW3Q2\_9.m | %hw3\_2\_9c.m  clear all;  T=[0.1,1,5];  sig=[0.2,0.2,0.2];  lam=[0.5,1,1];  alpha=[-0.15,-0.07,-0.07];  delta=[0.05,0,0.05];    s0=100;  K=(80:0.1:120);    du=1;    c=zeros(1,1);  for k=1:length(K)  for i=1:3  for j=3:3  beta = exp(alpha(j)+0.5\*delta(j)\*delta(j))-1;    integ = 0;  for u=0.5:du:100  C = 1i\*u\*(-0.5\*sig(j)\*sig(j)\*T(i)-beta\*lam(j)\*T(i));  D = -0.5\*u\*u\*sig(j)\*sig(j)\*T(i);  E = lam(j)\*T(i)\*(exp(1i\*u\*alpha(j)-0.5\*u\*u\*delta(j)\*delta(j))-1);  psi = exp(C + D + E);  integ = integ + imag(exp(-1i\*u\*log(K(k)/s0))\*psi)/u\*du;  end  Q1=1/2+1/pi\*integ;    integ = 0;  psi = 0;  for u=0.5:du:100  C = 1i\*u\*(0.5\*sig(j)\*sig(j)\*T(i)-beta\*lam(j)\*T(i));  D = -0.5\*u\*u\*sig(j)\*sig(j)\*T(i);  E = lam(j)\*T(i)\*(exp(1i\*(u-1i)\*alpha(j)-0.5\*delta(j)\*delta(j)\*(u-1i)\*(u-1i))-1-beta);  psi = exp(C + D) \* exp(E);  integ = integ + imag(exp(-1i\*u\*log(K(k)/s0))\*psi)/u\*du;  end  Q2=1/2+1/pi\*integ;  c(i,k)=s0\*Q2-K(k)\*Q1;  end  end  end    r=0;  ImpliedVol = zeros(3,length(K));  options = optimset('fzero');  options = optimset(options, 'TolX', 1e-8, 'Display', 'off');  for i = 1:length(K)  for j = 1:3  try  v0 = fzero(@(v0) ObjFcn(v0,s0,K(i),T(j),r,c(j,i)),[0.0001 5],options);  catch  v0 = NaN;  end  ImpliedVol(j,i) = v0;  end  end    plot(K,ImpliedVol); |

**2.11:**

|  |  |
| --- | --- |
| **Note** | **Code** |
| HW3Q2\_11.m | clear all;    v=0.0158;  v\_bar=0.0439;  eta=0.3038;  rho=-0.6974;  lambda=0.5394;  lambdaJ=0.1308;  delta=0.0967;  alpha=-0.1151;    s0=100;  K=[80,90,100,110,120];  T=[0.1,0.5,1,2,5];    du=1;    c=zeros(5,1);  for k=1:length(K)  for i=1:5  integ = 0;  for u=0.5:du:100  sigma = eta;  gamma = (eta^2)/2;  alpha2 = -0.5\*u\*(1i+u);  beta = lambda - 1i\*u\*rho\*sigma;  d = sqrt(beta^2 - 4\*alpha2\*gamma);  rMinus = (beta-d)/(2\*gamma);  rPlus = (beta+d)/(2\*gamma);  g = rMinus/rPlus;  beta2 = exp(alpha+0.5\*delta\*delta)-1;    D = rMinus \* ((1-exp(-d\*T(i)))/(1-g\*exp(-d\*T(i))));  C = lambda\*(rMinus\*T(i)-(2/sigma^2)\*log((1-g\*exp(-d\*T(i)))/(1-g)));  E = -1i\*u\*beta2\*lambdaJ\*T(i);  F = lambdaJ\*T(i)\*(exp(1i\*u\*alpha-0.5\*u\*u\*delta\*delta)-1);    psi = exp(C\*v\_bar + D\*v + E + F);  integ = integ + imag(exp(-1i\*u\*log(K(k)/s0))\*psi)/u\*du;  end  Q1=1/2+1/pi\*integ;    integ = 0;  psi = 0;  for u=0.5:du:100  sigma = eta;  gamma = (eta^2)/2;  alpha2 = 0.5\*u\*(1i-u);  beta = lambda - 1i\*u\*rho\*sigma - sigma\*rho;  d = sqrt(beta^2 - 4\*alpha2\*gamma);  rMinus = (beta-d)/(2\*gamma);  rPlus = (beta+d)/(2\*gamma);  g = rMinus/rPlus;    D = rMinus \* ((1-exp(-d\*T(i)))/(1-g\*exp(-d\*T(i))));  C = lambda\*(rMinus\*T(i)-(2/sigma^2)\*log((1-g\*exp(-d\*T(i)))/(1-g)));  E = 1i\*u\*(-beta2\*lambdaJ\*T(i));  F = lambdaJ\*T(i)\*(exp(1i\*(u-1i)\*alpha-0.5\*delta\*delta\*(u-1i)\*(u-1i))-1-beta2);    psi = exp(C\*v\_bar + D\*v + E + F);  integ = integ + imag(exp(-1i\*u\*log(K(k)/s0))\*psi)/u\*du;  end  Q2=1/2+1/pi\*integ;  c(i,k)=s0\*Q2-K(k)\*Q1;  end  end  c |

**3.1:**

|  |  |
| --- | --- |
| **Note** | **Code** |
| HW3Q3a.m | clear;    %first index is T  %second index is K    obsPrice = zeros(5,5);  obsPrice(1,1) = 20.0087;  obsPrice(1,2) = 10.0863;  obsPrice(1,3) = 1.6517;  obsPrice(1,4) = 0.0024;  obsPrice(1,5) = 0.0001;  obsPrice(2,1) = 20.3092;  obsPrice(2,2) = 11.2117;  obsPrice(2,3) = 3.8561;  obsPrice(2,4) = 0.4113;  obsPrice(2,5) = 0.0223;  obsPrice(3,1) = 21.0696;  obsPrice(3,2) = 12.5945;  obsPrice(3,3) = 5.6858;  obsPrice(3,4) = 1.4939;  obsPrice(3,5) = 0.2518;  obsPrice(4,1) = 22.7139;  obsPrice(4,2) = 14.9601;  obsPrice(4,3) = 8.5607;  obsPrice(4,4) = 3.9815;  obsPrice(4,5) = 1.4644;  obsPrice(5,1) = 27.1208;  obsPrice(5,2) = 20.5021;  obsPrice(5,3) = 14.8677;  obsPrice(5,4) = 10.2924;  obsPrice(5,5) = 6.7808;    s0 = 100;  K=[80,90,100,110,120];  T=[0.1,0.5,1,2,5];  %v0, kappa,theta,beta,rho  piVector=[0.018824,0.37496,0.053682,0.26082,-0.6691];    r = 0;  options = optimset('MaxFunEvals',10000);  piVector = fminsearch(@(piVector) objFun(piVector,r,T,s0,K,obsPrice), piVector, options);  v0 = piVector(1);  kappa = piVector(2);  theta = piVector(3);  beta = piVector(4);  rho = piVector(5);  disp([num2str(v0) ',' num2str(kappa) ',' num2str(theta) ',' num2str(beta) ',' num2str(rho)]); |
| HestonPrice.m | %HestonPrice.m    function call = HestonPrice(kappa,theta,sig,rho,v0,r,T,s0,K)  call = s0\*HestonP(kappa,theta,sig,rho,v0,r,T,s0,K,1) - K\*exp(-r\*T)\*HestonP(kappa,theta,sig,rho,v0,r,T,s0,K,2);    function retP = HestonP(kappa,theta,sig,rho,v0,r,T,s0,K,type)  retP = 1/2 + 1/pi\*quad(@HestonPIntg,0,100,[],[],kappa,theta,sig,rho,v0,r,T,s0,K,type);    function retI = HestonPIntg(phi,kappa,theta,sig,rho,v0,r,T,s0,K,type)  retI = real(exp(-1i\*phi\*log(K)).\*Hestf(phi,kappa,theta,sig,rho,v0,r,T,s0,type)./(1i\*phi));    function retf = Hestf(phi,kappa,theta,sig,rho,v0,r,T,s0,type)  if type == 1  u = 0.5;  b = kappa - rho\*sig;  else  u = -0.5;  b = kappa;  end  x = log(s0);  a = kappa \* theta;  d = sqrt((rho\*sig\*phi.\*1i-b).^2 - sig^2\*(2\*u\*phi.\*1i-phi.^2));  g = (b-rho\*sig\*phi\*1i+d) ./ (b-rho\*sig\*phi\*1i-d);  C = r\*phi.\*1i\*T + (a/sig^2).\*((b-rho\*sig\*phi\*1i+d)\*T - 2\*log((1-g.\*exp(d\*T))./(1-g)));  D = (b-rho\*sig\*phi\*1i+d)./sig^2 .\* ((1-exp(d\*T))./(1-g.\*exp(d\*T)));  retf = exp(C + D\*v0 + 1i\*phi\*x); |
| objFun.m | function delta = objFun(piVector,r,T,s0,K,c)  v0 = piVector(1);  kappa = piVector(2);  theta = piVector(3);  beta = piVector(4);  rho = piVector(5);  SquaredError = 0;  PriceDiffSum = 0;  myC = c;  for i=1:5  for j=1:5  myC(i,j) = HestonPrice(T(i),s0,K(j),v0,theta,kappa,beta,rho,r);  PriceDiff = c(i,j) - myC(i,j);  if myC(i,j)<0  SquaredError = SquaredError + 1000;  end  if myC(i,j)<(s0-K(j))  SquaredError = SquaredError + 1000;  end  SquaredError = SquaredError + PriceDiff\*PriceDiff;  PriceDiffSum = PriceDiffSum+PriceDiff;  end  end    if abs(rho) >= 1  SquaredError = SquaredError + 1000;  end  if beta < 0.00001  SquaredError = SquaredError + 1000;  end  if v0 < 0.01  SquaredError = SquaredError + 1000;  end  if theta <= 0.000001  SquaredError = SquaredError + 1000;  end  if kappa <= 0.00001  SquaredError = SquaredError + 1000;  end  %myC  disp([num2str(SquaredError) ': ' num2str(PriceDiffSum) ',' num2str(v0) ',' num2str(kappa) ',' num2str(theta) ',' num2str(beta) ',' num2str(rho)]);  delta = SquaredError; |
| objFun2.m | %objFcn2.m    function delta = objFun2(piVector,r,T,s0,K,c)  v0 = piVector(1);  kappa = piVector(2);  theta = piVector(3);  beta = piVector(4);  rho = piVector(5);  SquaredError = 0;  PriceDiffSum = 0;  myC = c;  for i=1:5  for j=1:5  myC(i,j) = HestonPrice(kappa,theta,beta,rho,v0,r,T(i),s0,K(j));  PriceDiff = c(i,j) - myC(i,j);  if myC(i,j)<=0.0001  SquaredError = SquaredError + 1000;  end  if myC(i,j)<=(s0-K(j)+0.0001)  SquaredError = SquaredError + 1000;  end  %if (i==1 && j==1) || (i==1 && j==5) || ...  % (i==3 && j==3) || ...  % (i==5 && j ==1) || (i==5 && j == 5)  SquaredError = SquaredError + PriceDiff\*PriceDiff;  %end  PriceDiffSum = PriceDiffSum+PriceDiff;  end  end  if myC(1,1) <20.001  SquaredError = SquaredError + 1000;  end  c = HestonPrice(kappa,theta,beta,rho,v0,r,0.1,s0,118);  if c <= 0  SquaredError = SquaredError + 1000;  end  if abs(rho) >= 1  SquaredError = SquaredError + 1000;  end  if beta < 0.00001  SquaredError = SquaredError + 1000;  end  if v0 < 0.001  SquaredError = SquaredError + 1000;  end  if theta <= 0.000001  SquaredError = SquaredError + 1000;  end  if kappa <= 0.00001  SquaredError = SquaredError + 1000;  end  myC  disp([num2str(SquaredError) ': ' num2str(PriceDiffSum) ',' num2str(v0) ',' num2str(kappa) ',' num2str(theta) ',' num2str(beta) ',' num2str(rho)]);  %disp(['squared error ' num2str(SquaredError)]);  delta = SquaredError; |
| ObjFcn.m | function delta = ObjFcn(volatility, s0, K, T, r, CallPrice)    BSprice = BSPrice(s0, K, T, r, volatility);  delta = CallPrice - BSprice; |
| BSPrice.m | function BlackScholesPrice = BSPrice(s0,K,T,r,sigma)    F=s0.\*exp(r.\*T);  d1=log(F./K)./(sigma.\*sqrt(T))+sigma.\*sqrt(T)/2;  d2=log(F./K)./(sigma.\*sqrt(T))-sigma.\*sqrt(T)/2;  BlackScholesPrice = exp(-r.\*T).\*(F.\*normcdf(d1)-K.\*normcdf(d2)); |

**3.2:**

|  |  |
| --- | --- |
| **Note** | **Code** |
| hw3\_3b.m | clear all;  TVector=[0.1,1];  k=1; %choose between T=0.1 and 1    s0=100;  KVector=[80,90,100,110,120];    %v0, kappa,theta,beta,rho  piVector=[0.018824,0.37496,0.053682,0.26082,-0.6691];  %piVector=[0.0174,1.3253,0.0354,0.3877,-0.7165];    kappa = piVector(2);  theta = piVector(3);  beta = piVector(4);  rho = piVector(5);  v0 = piVector(1);    r=0;  ImpliedVol = zeros(length(TVector),length(KVector));  c = zeros(length(TVector),length(KVector));  options = optimset('fzero');  options = optimset(options, 'TolX', 1e-8, 'Display', 'off');  for i = 1:length(KVector)  for j = 1:2  v0 = piVector(1);  T = TVector(k);  K = KVector(i);  if j==1  c(j,i) = HestonPrice(T,s0,K,v0,theta,kappa,beta,rho,r);  else  c(2,:) = [20.0087 10.0863 1.6517 0.0024 0.0001];  %c(2,:) = [21.0696 12.5945 5.6858 1.4939 0.2518];  end  try  v0 = fzero(@(v0) ObjFcn(v0,s0,KVector(i),TVector(k),r,c(j,i)),[0.0001 5],options);  catch  v0 = NaN;  end  ImpliedVol(j,i) = v0;  end  end    plot(KVector,ImpliedVol);  h = legend('Heston','Table Prices');  ImpliedVol; |