PS8_Kaiyue Wu

In [594…

```
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd
from scipy.stats import norm
import scipy.stats
from scipy.stats import ncx2

new_line = '\n'
pd.options.mode.chained_assignment = None
import warnings
warnings.filterwarnings('ignore')

sns.set(font_scale=1.5, rc={'text.usetex' : True,})
%config InlineBackend.figure_format='retina'
```

1 Vasicek model

$dr_t = \kappa(\bar{r} - r)dt + \sigma dW$

In [2]:

```
r0 = 0.05
sigma = 0.1
kappa = 0.82
r_bar = 0.05
```

(a) Monte Carlo Simulation for pure discount bond, with Face Value of $1,000, maturing in T = 0.5 years

In [3]:

```
T = 0.5
t=0
FV=1000
```

In [4]:

```
def VasicekR(r0,T,path,steps):

    """
    simulate rt
    """
    dt = T/steps
    rts = np.zeros((path,steps+1))

    # initialize r0
    rts[:,0] = r0

    for i in range(1,steps+1):
        dWt = np.sqrt(dt)*np.random.normal(0,1,path)
        rts[:,i] = rts[:,i-1] + kappa*(r_bar - rts[:,i-1])*dt + sigma*dWt
```

```python
    return rts
# VasicekR(r0=0.05,T=0.5,path=10,steps=10)
```

In [264…

```python
def PureDiscountBond(r0,T,path,FV,t):

    steps = int(365*(T - t))
    dt = (T - t)/steps
    r = VasicekR(r0,T,path,steps)

    Euler = np.zeros(path)

    for i in range(path):
        Euler[i] = -sum(r[i,1:]*dt)

    price = np.mean(FV*np.exp(Euler))

    return price
```

In [265…

```python
PDB = PureDiscountBond(r0=0.05,T=0.5,t=0,path=100000,FV=1000)
```

In [266…

```python
print(f"The Pure Discount Bond price is ${PDB}")
```

```
The Pure Discount Bond price is $975.42742434474
```

(b) Monte Carlo Simulation to find the price of a coupon-paying bond, with Face Value of 1,000, paying semiannual coupons of 30, maturing in T = 4 years

In [233…

```python
FV = 1000
c = np.append(np.repeat(30,7),1030)
T = 4
time = np.arange(0.5,4.5,0.5)
```

In [376…

```python
# def CouponPayingBond(r0,T,t,path,FV):
#     """
#     Coupon Paying Bond
#     """
#     steps = int(365*(T - t))
#     dt = (T - t)/steps
#     r = VasicekR(r0,(T - t),path,steps)
#     time_steps = np.array([int(i*365) for i in time])

#     Euler = np.zeros((path,len(time)))
#     for i in range(path):
#         for j in range(len(time)):
#             Euler[i,j] = c[j]*np.exp(-sum(r[i,:(time_steps[j]+1)]*dt))

#     EP = np.sum(Euler,axis=1).mean()

#     return EP
```

In [319…

```python
def CouponPayingBond(r0,T,t,path,FV):
```

```
    """
    Coupon Paying Bond
    """
    steps = int(365*(T - t))
    dt = (T - t)/steps
    r = VasicekR(r0,(T - t),path,steps)
    r = r[:,-1]

    Euler = list(map(lambda coupon,ctime:PureDiscountBond(r0=r,T=ctime,path=path
                c,time))

    EP = sum(Euler).mean()

    return EP
```

In [320…

```
CPB = CouponPayingBond(r0=0.05,T=4,t=0,path=1000,FV=1000)
```

In [152…

```
print(f"The Coupon Paying Bond price is ${round(CPB,4)}")
```

```
The Coupon Paying Bond price is $1053.2217
```

(c) Use Monte Carlo Simulation to find the price of a European Call Option on the Pure Discount Bond of part (a). The option expires in 3 months and has a strike price of K = 980. Use the explicit formula for the underlying bond price (use explicit formula only for the bond price).

In [66]:

```
K = 980
t = 3/12
T = 0.5
FV = 1000
```

In [322…

```
def EuroCall_explicit(r0,T,t,path,FV):

#     np.random.seed(0)

    steps = int(365*(T - t))
    dt = (T - t)/steps
    rt = VasicekR(r0,T,path,steps)

    B = 1/kappa*(1-np.exp(-kappa*(T-t)))
    A = np.exp((r_bar-sigma**2/(2*kappa**2))*(B-(T-t))-sigma**2/(4*kappa)*B**2)

    PtT = A*np.exp(-B*rt) * FV

    PtT = PtT[:,-1]

    payoff = np.maximum(PtT - K,0)

    disc = np.exp(-rt[:,-1]*(T-t))

    price = sum(payoff*disc)/path

    return price
```

In [323…

```
p_ex = EuroCall_explicit(r0=0.05,T=0.5,t=3/12,path=100000,FV=1000)
```

In [324…

```
print(f"The Pure Discount Bond price with the explicit formula is ${p_ex}")
```

```
The Pure Discount Bond price with the explicit formula is $9.936223362197568
```

(d) Use Monte Carlo Simulation to find the price of a European Call Option on the coupon-paying bond in part (b). The option expires in 3 months and has a strike price of $K = 980$. Use Monte Carlo simulation for pricing the underlying bond (no explicit formula can be used in this part).

In [ ]:

```
def EuroCall_b_MC(r0,T,t,path,FV):

    steps = int(365*(T - t))
    dt = (T - t)/steps
    rt = VasicekR(r0,(T - t),path,steps)


    P = CouponPayingBond(rt[:,-1],T,t,path,FV)

    payoff = np.maximum(P-K,0)
    disc = np.exp(-rt[:,-1]*(T-t))

    price = sum(payoff*disc)/path

    return price
```

In [336…

```
price_b = EuroCall_b_MC(r0=0.05,T=0.5,t=3/12,path=10000,FV=1000)
print(f"The Pure Discount Bond price with the Monte Carlo Simulation is ${price_
```

```
The Pure Discount Bond price with the Monte Carlo Simulation is $80.984799816647
13
```

(e) Find the price of a European Call option of part (d) by using the explicit formula for the underlying bond price, and reconcile the findings here with the ones of part (d).

In [182…

```
r0 = 0.05
sigma = 0.1
kappa = 0.82
r_bar = 0.05
t = 3/12
T = 0.5
```

In [ ]:

```
def EuroCall_b_explicit(r0,T,t,path,FV):

    steps = int(365*(T - t))
    dt = T/steps
    rt = VasicekR(r0,T,path,steps)
```

```
    c = np.append(np.repeat(30,7),1030)
    T = 3/12
    Ti = np.array([i*(6/12) for i in range(1,9,1)])


    r_star = r0
    max_iter = 1000

    for k in range(max_iter):
        Pi = np.array([])
        for i in range(len(Ti)):
            B = 1/kappa*(1-np.exp(-kappa*(Ti[i]-T)))
            A = np.exp((r_bar-sigma**2/(2*kappa**2))*(B-(Ti[i]-T))-sigma**2/(4*k
            Pi = np.append(Pi,A*np.exp(-B*r_star) )

        F = sum(Pi*c)
        if F==K: break
        elif F-K > 0.005:
            r_star +=0.001
        elif F-K > -0.005:
            r_star -=0.001

    Ki = np.array([])
    for i in range(len(Ti)):
        B = 1/kappa*(1-np.exp(-kappa*(Ti[i]-T)))
        A = np.exp((r_bar-sigma**2/(2*kappa**2))*(B-(Ti[i]-T))-sigma**2/(4*kappa
        Ki = np.append(Ki,A*np.exp(-B*r_star) )

    PTi = np.array([])
    for i in range(len(Ti)):
        B = 1/kappa*(1-np.exp(-kappa*(Ti[i]-T)))
        A = np.exp((r_bar-sigma**2/(2*kappa**2))*(B-(Ti[i]-T))-sigma**2/(4*kappa
        PTi = np.append(PTi,A*np.exp(-B*r0) )

    B = 1/kappa*(1-np.exp(-kappa*(T-t)))
    A = np.exp((r_bar-sigma**2/(2*kappa**2))*(B-(T-t))-sigma**2/(4*kappa)*B**2)
    PT = A*np.exp(-B*r0)


    sigma_p = sigma*(1-np.exp(-kappa*(Ti-T)))/kappa*np.sqrt((1-np.exp(-2*kappa*(
    d1 = np.log(PTi/(Ki*PT))/sigma_p + sigma_p/2
    d2 = d1 - sigma_p

    C = PTi*norm.cdf(d1) - Ki*PT*norm.cdf(d2)
    price = sum(c*C)
    return price
```

In [370…

```
price_b_explicit  = EuroCall_b_explicit(r0=0.05,T=0.5,t=3/12,path=1000,FV=1000)
```

In [371…

```
print(f"The Pure Discount Bond price with Explicit Method is ${price_b_explicit}
```

```
The Pure Discount Bond price with Explicit Method is $81.74101353120759
```

2 CIR Model

$$dr = \kappa(\bar{r} - r)dt + \sigma\sqrt{r}dW$$

In [133…

```
r = 0.05
sigma=0.12
kappa=0.92
r_bar =0.055
K=980
FV=1000
T=0.5
S=1
t0=0
```

In [134…

```
def CIR_R(path,steps,r0,T):

    dt = T/steps
    r = np.zeros((path,steps+1))
    r[:,0] = r0

    for i in range(1,steps+1):
        dWt = np.sqrt(dt)*np.random.normal(0,1,path)
        r[:,i] = r[:,i-1] + kappa*(r_bar - r[:,i-1])*dt + sigma*np.sqrt(r[:,i-1]

    return r
```

In [134…

```
def PureDiscountBond_CIR(r0,T,path,S):



    steps = int(365*(S - T))
    dt = (S - T)/steps
    r = CIR_R(r0=r0,T=(S - T),path=path,steps=steps)

    Euler = np.zeros(path)

    for i in range(path):
        Euler[i] = -sum(r[i,1:]*dt)

    price = np.mean(FV*np.exp(Euler))


    return price
```

(a) Use Monte Carlo Simulation to find at time t = 0 the price c(t, T, S) of a European Call option, with strike price of K = 980 and expiration in T = 0.5 years on a Pure Discount Bond with Face Value of 1,000, that matures in S = 1 year:

In [134…

```
def EuroCall_CIR(r0,T,t,path,S):


    steps = int(365*(S - T))
    dt = (S - T)/steps
    r = CIR_R(r0=r0,T=(S - T),path=path,steps=steps)
```

```
    BondPrice = np.array([PureDiscountBond_CIR(r0=temp_r,T=T,path=path,S=S) for

    payoff = np.maximum(BondPrice - K,0)

    disc = np.exp(-r[:,-1]*(S-T))

    # dicount price at T=0.5 to t=0
    price = np.mean(payoff*disc)
    price *= np.exp(-r0*(T-t))

    return price
```

In [134…

```
Price_a_CIR = EuroCall_CIR(r0=0.05,T=0.5,t=0,S=1,path=1000)
```

In [134…

```
print(f"The Pure Discount Bond price with Monte Carlo Simulation is ${round(Pric
```

```
The Pure Discount Bond price with Monte Carlo Simulation is $0.4241
```

(b) Use the Implicit Finite-Difference Method to find at time t = 0 the price c(t, T, S) of a European Call option, with strike price of S = 980 and expiration in T = 0.5 years on a Pure Discount Bond with Face Value of 1,000, which matures in S = 1 year.

In [ ]:

```
def Pumd(dt, sigma, r_bar, dr, kappa, grid_r):

    Pu = -(1/2)*dt*((sigma**2 * grid_r)/(dr**2) + (kappa*(r_bar - grid_r))/(dr))
    Pm = 1 + (sigma**2 * dt * grid_r)/(dr**2) + grid_r*dt
    Pd = -(1/2)*dt*((sigma**2 * grid_r)/(dr**2) - (kappa*(r_bar - grid_r))/(dr))

    return Pu, Pm, Pd
```

In [151…

```
def CIR_Implict(r0,T,t,path,S,M,N):


    dt  = T/M
    dr = 0.01
    grid = np.zeros((M,N+1))
    A = np.zeros((N-2,N-2))
    Bi = np.zeros(M-1)
    rgrid = np.zeros(N+1)

    for j in range(N):
        rgrid[j] = j*dr

    Pu,Pm,Pd = Pumd(dt=dt, sigma=sigma, r_bar=r_bar, dr=dr, kappa=kappa, grid_r=

    for i in range(M):
        for j in range(N+1):
            grid[i,j] = PureDiscountBond_CIR(rgrid[j],T,path,S)

    A[0,[0,1]] = [Pm[N-1],Pd[N-1]]
    j=0
    for i in range(1,len(A)-1):
        A[i,[j,j+1,j+2]] = [Pu[N-j-2],Pm[N-j-2],Pd[N-j-2]]
```

```
        j+=1
    A[-1,[-2,-1]] = [Pu[1],Pm[1]]

    index = np.where(rgrid==0.05)[0][0]


    Ci = np.maximum(grid[:,-2]-980,0)
    Ci=Ci[1:-1]
    B = np.zeros_like(Ci)
    B[1:-1] = Ci[1:-1]

    B[-1] = PureDiscountBond_CIR(r0=rgrid[-1],T=T,path=M,S=2)  - PureDiscountBon
    B[0]=0

    j=0
    for i in range(N):
        Ci = np.dot(np.linalg.inv(A),B)
        B = np.zeros_like(Ci)
        B[1:-1]= Ci[1:-1]
#          B[-1] = PureDiscountBond_CIR(r0=rgrid[-1],T=T,path=M,S=2)  - PureDiscc
#          j+=1

    option = Ci[index]
    option *= np.exp(-r0*(T-t))

    return option
```

In [151…

```
price_cir_im = CIR_Implict(r0=r0,T=0.5,t=0,path=1000,S=1,M=10,N=10)
```

In [152…

```
print(f"The Pure Discount Bond price of CIR with Implicit Method is ${round(pric
```

```
The Pure Discount Bond price of CIR with Implicit Method is $0.4564
```

(c) Compute the price c(t, T, S) of the European Call option above using the explicit formula, and compare it to your findings in parts (a) and (b) and comment on your findings.

In [663…

```
def AB_CIR(rt, T, t, FV):

    h1 = np.sqrt(kappa**2 + 2*sigma**2)
    h2 = (kappa + h1)/2
    h3 = (2*kappa*r_bar)/sigma**2

    A = ((h1 * np.exp(h2*(T - t)))/(h2 * (np.exp(h1*(T - t)) - 1) + h1))**h3
    B = (np.exp(h1*(T - t)) - 1)/(h2 * (np.exp(h1*(T - t)) - 1) + h1)

    return FV * A*np.exp(-B*rt)
```

In [666…

```
def CIR_Explict(r0,T,t,path,S,K):

    K= K/FV
    steps = int(365*(S - T))
    dt = (S - T)/steps
```

```
    PT=AB_CIR(r0, T, 0, FV)/FV
    PS=AB_CIR(r0, S, 0, FV)/FV

    h1 = np.sqrt(kappa**2 + 2*sigma**2)
    h2 = (kappa + h1)/2
    h3 = (2*kappa*r_bar)/sigma**2

    A = ((h1 * np.exp(h2*(S - T)))/(h2 * (np.exp(h1*(S - T)) - 1) + h1))**h3
    B = (np.exp(h1*(S - T)) - 1)/(h2 * (np.exp(h1*(S - T)) - 1) + h1)

    theta = np.sqrt(kappa**2 + 2*sigma**2)
    phi = (2 * theta)/(sigma**2 * (np.exp(theta*(T - 0)) - 1))
    si = (kappa + theta)/sigma**2
    r_star = np.log(A/K)/B

    C = (PS* ncx2.cdf(2*r_star*(phi + si + B), (4*kappa*r_bar)/(sigma**2), (2*ph
                  /(phi + si + B))
                    - K * PT * ncx2.cdf(2*r_star*(phi + si),
                                        (4*kappa*r_bar)/(sigma**2),
                                        (2*phi**2 * r0 * np.exp(theta*(T-0))
    C = FV*C

    return C
```

In [668…

```
price_cir_ex = CIR_Explict(r0=r0,T=T,t=0,path=10,S=S,K=980)
```

In [669…

```
print(f"The Pure Discount Bond price of CIR with Explicit Method is ${round(pric
```

```
The Pure Discount Bond price of CIR with Explicit Method is $0.3941
```

3. G2++ model

In [777…

```
x0 = 0
y0 = 0
phi_0 = 0.03
r0 = 0.03
rho =0.7
a = 0.1
b = 0.3
sigma =0.03
eta = 0.08
pht_t = 0.03

K = 950
T = 0.5
S = 1
FV = 1000
```

In [738…

```
def bivariate_normal(rho):

    z1 = np.random.normal(0,1, 1)
    z2 = np.random.normal(0,1, 1)

    Z1 = z1
```

```
    Z2 = rho*z1+np.sqrt(1-rho**2)*z2

    return Z1[0],Z2[0]
# bivariate_normal(rho)
```

In [707…

```
def G2_r(x0,y0,rho,T,t,path):

    steps = int(365*(T-t))
    dt = T/steps

    x = np.zeros((path,steps+1))
    y = np.zeros((path,steps+1))
    r = np.zeros((path,steps+1))

    x[:,0] = x0
    y[:,0] = y0


    for i in range(1,steps+1):
        dw = bivariate_normal(rho)
        dw1 = dw[0]
        dw2 = dw[1]
        x[:,i] = x[:,i-1] -a*x[:,i-1]*dt + sigma*np.sqrt(dt)*dw1
        y[:,i] = x[:,i-1] -b*y[:,i-1]*dt + eta*np.sqrt(dt)*dw2

    r = x + y + phi

    return r,x,y
# G2_r(x0,y0,rho,T,t,10)
```

In [705…

```
def PureDiscountBondG2(t, T, FV,path,rho,x0,y0):

    steps = int(365*(T-t))
    dt = T/steps

    r = G2_r(x0,y0,rho,T,t,path)[0]

    Euler = np.zeros(path)

    for i in range(path):
        Euler[i] = -sum(r[i,1:]*dt)

    price = np.mean(FV*np.exp(Euler))

    return price
# PureDiscountBondG2(t, T, FV,10,rho)
```

In [ ]:

```
def G2_MC(t, T, S, FV,path,rho,x0,y0):

    steps = int(365*(T-t))
    dt = T/steps

    r,x,y = G2_r(x0,y0,rho,T,t,path)[0],G2_r(x0,y0,rho,T,t,path)[1],G2_r(x0,y0,r

    p = np.zeros(path)
    for i in range(path):
```

```
        p[i] = PureDiscountBondG2(t=T, T=S, FV=FV,path=path,rho=rho,x0=x[:,-1],y

    payoff = np.maximum(K-p,0)

    Euler = np.zeros(path)

    for i in range(path):
        Euler[i] = -sum(r[i,1:]*dt)

    price = np.mean(payoff*np.exp(Euler))
    # dicount price at T=0.5 to t=0
    price *= np.exp(-r0*(T-t))

    return price
```

In [736…

```
g2_mc = G2_MC(t, T,S, FV,1000,rho,x0,y0)
```

In [739…

```
print(f"The Put Option Price of G2 model by Monte Carlo Simulation is ${round(g2
```

```
The Put Option Price of G2 model by Monte Carlo Simulation is $1.4175
```

In [768…

```
def G2_ZCB_Explicit(t,T,a,b,sigma,rho,eta,FV):

    part1 = (sigma**2/ a**2)*( T - t
                              + (2/a)*np.exp(-a*(T - t))
                              - (1/(2*a)) *np.exp(-2*a*(T - t))
                              - (3/(2*a)))

    part2 = (eta**2/b**2) *(T - t
                            + (2/b)*np.exp(-b * (T - t))
                            - (1/(2*b)) *np.exp(-2 * b * (T - t))
                            - (3/(2 * b)))

    part3 = 2*rho*(sigma*eta)/(a*b)*(T - t
                                     + (np.exp(-a*(T - t)) - 1)/a
                                     + (np.exp(-b*(T - t)) - 1)/b
                                     -(np.exp(-(a + b)*(T - t))-1)/(a + b))
    total = part1 + part2 + part3

    P = np.exp(-phi*(T - t)
               - ((1 - np.exp(-a * (T - t)))/a) * x0
               - ((1 - np.exp(-b * (T - t))/b)*y0)
               + (1/2)* total) * FV

    return P
# G2_ZCB_Explicit(t,T,a,b,sigma,rho,eta,FV)
```

In [789…

```
def G2_EuroPut_Explicit(t,T,S,a,b,sigma,rho,eta,FV,K):

    K /= FV

    part1 = sigma**2/(2* a**3)*(1 - np.exp(-a * (S - T)))**2 * (1 - np.exp(-2 *a

    part2 = eta**2/(2*b**3)*(1 - np.exp(-b* (S - T)))**2*(1 - np.exp(-2*b*(T - t
```

```
    part3 = (2 *rho *sigma *eta/(a*b*(a + b))*(1-np.exp(-a*(S - T)))*(1 - np.exp


    total = np.sqrt(part1 + part2 + part3)

    PS = G2_ZCB_Explicit(0,S,a,b,sigma,rho,eta,FV)
    PT = G2_ZCB_Explicit(0,T,a,b,sigma,rho,eta,FV)

    price = (-PS*scipy.stats.norm.cdf(np.log(K*PT/PS)/total - total/2)
           + PT*K*scipy.stats.norm.cdf(np.log(K*PT/PS)/total +total/2))

    return price
```

In [790…

```
price_g2_ex = G2_EuroPut_Explicit(0,0.5,1,a,b,sigma,rho,eta,FV,K)
```

In [792…

```
print(f"The Put Option Price of G2 model by Explicit Method is ${round(price_g2_
```

```
The Put Option Price of G2 model by Explicit Method is $1.861
```