Student Id: 9910821

Task 1

Formula for the value h given by

$$h = \frac{X_r - f(r,t,T)}{\sqrt{v^2(t,T)}}. \tag{1}$$

Task 2

Value of the put option $V(r,t,T)$ is

$$V(r,t,T) = 0.549 \tag{2}$$

with parameters $r = 0.052$, $t = 0$, $T = 2$, $X_r = 0.06$, $\kappa = 0.0944$, $\theta = 0.0616$, $\sigma = 0.0317$.

Task 3

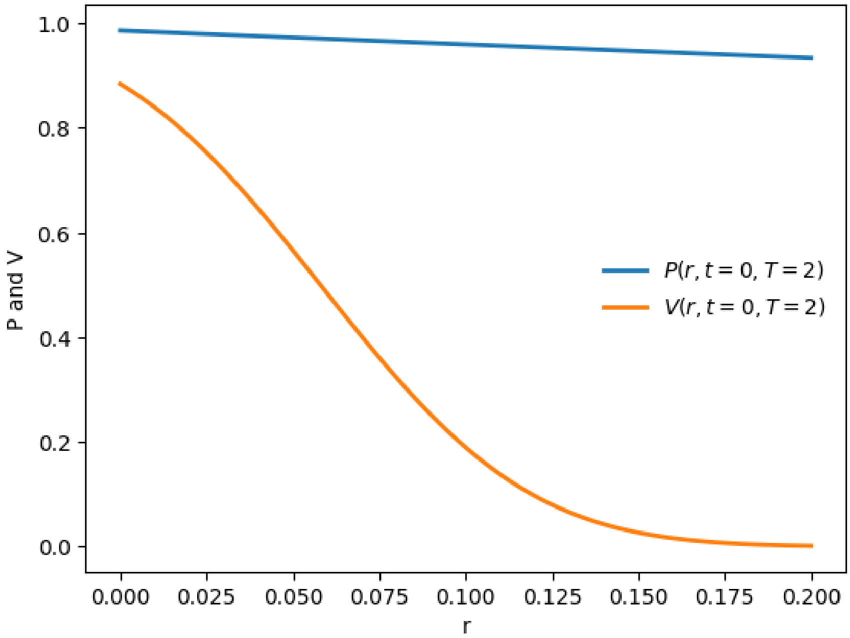

Figure 1: A plot of the functions of bond price P and the option price V for different interest rates r.

Program listings

Pricing Generator Listing

```
#include <iostream>
#include <iomanip>
#include <fstream>
#include <cmath>
```

```cpp
using namespace std;

double normalDistribution(double x)
{
    return 0.5*erfc(-x / sqrt(2.));
}
double qFunc(double t, double T, double kappa, double theta, double sigma)
{
    return (0.25*sigma*sigma/(kappa*kappa))*pow(1-exp(-kappa*(T-t)),3.);
}
double kSquaredFunc(double t, double T, double kappa, double theta, double sigma)
{
    return 0.5*sigma*sigma/pow(kappa,3.)
        *( 2.*exp(-kappa*(T-t)) - 3.* exp(-2.*kappa*(T-t))
        + 4.*kappa*(T-t) + 1. );
}
double nFunc(double r, double t, double T, double kappa, double theta, double sigma)
{
    return r*(T-t) + (theta-r)*(1.-exp(-kappa*(T-t)))/(2. * kappa);
}
double mFunc(double r, double t, double T, double kappa, double theta, double sigma)
{
    return r*exp(-kappa*(T-t)) + (1-exp(-kappa*(T-t)))*theta;
}
double vSquaredFunc(double t, double T, double kappa, double theta, double sigma)
{
    return (sigma*sigma/(3.*kappa))*(1-exp(-3.*kappa*(T-t)));
}
double fFunc(double r, double t, double T, double kappa, double theta, double sigma)
{
    return mFunc(r,t,T,kappa,theta,sigma) - 0.5*qFunc(t,T,kappa,theta,sigma);
}
//H function used by normalising z=xr-f/v where xr is N(f,v^2)
double hFunc(double r, double t, double T, double kappa, double theta, double sigma,double Xr)
{
    return (Xr-fFunc(r,t,T,kappa,theta,sigma))/sqrt(vSquaredFunc(t,T,kappa,theta,sigma));
}
double PFunc(double r, double t, double T, double kappa, double theta, double sigma)
{
    return exp((2/3)*kSquaredFunc(t,T,kappa,theta,sigma) - 0.25*nFunc(r,t,T,kappa,theta,sigma));
}
double VFunc(double r, double t, double T, double kappa, double theta, double sigma,double Xr)
{
    return PFunc(r, t, T, kappa, theta, sigma)*normalDistribution(hFunc(r, t, T, kappa, theta, sigma,Xr));
}

```

```cpp
int main()
{
    double r0 = 0.052, t = 0., T = 2., kappa = 0.0944, theta = 0.0616,
   sigma = 0.0317, Xr=0.06;

    double rMin = 0., rMax = 0.2;
    double n = 100.;
    double dr = (rMax - rMin) / n;
    cout << VFunc(r0, t, T, kappa, theta, sigma,Xr) << endl;
    ofstream output("./Assignment_2/test.csv");
    for (int i = 0; i <= 100; i++)
    {
        double r = rMin + i*dr;
        output << r << "," << PFunc(r, t, T, kappa, theta, sigma);
        output << "," << VFunc(r, t, T, kappa, theta, sigma,Xr) << endl;
    }
}
```

Graphing Package Listing

```python
import matplotlib.pyplot as plt
import csv

with open('test.csv', newline='\n') as csvfile:
    reader = csv.DictReader(csvfile,fieldnames=['r','p','v'],quoting=csv.
   QUOTE_NONNUMERIC)
    allData={'r':[],'p':[],'v':[]}
    for row in reader:
        allData['r'].append(row['r'])
        allData['p'].append(row['p'])
        allData['v'].append(row['v'])
    plt.figure()
    plt.plot(allData['r'],allData['p'],label=r'$P(r,t=0,T=2)$',linewidth=2)
    plt.plot(allData['r'],allData['v'],label=r'$V(r,t=0,T=2)$',linewidth=2)
    plt.xlabel('r')
    plt.ylabel('P and V')
    plt.legend(loc='center right',fancybox=False, framealpha=0.0)
    plt.savefig('Solution/plot.png',bbox_inches='tight', transparent="True"
   , pad_inches=0)
```