Project 5
MFE 405: ComputationalFinance
Professor Goukasian
Student: Xiahao Wang

This is a summary of the project for data visualisation, for detail implementation and result,

Note: Running the R code may take a long time, I have set the simulation to be 10000 in order to showcase that the code is able to run. The results that I have captured in this document use 100,000 simulations stated in the project question, although the difference from 10,000 and 10,000 are very small

Qn1

Implementation in R

Summary:

(a) Laguerre polynomials t = 0.5 100000 simulations with 100 steps each

```
library(data.table)
DT <- data.table(
  "Stock Price" = c("36", "40", "44"),
  "Payoff At K=2" = c(3.990018,1.375098,0.4870358),
  "Payoff At K=3" = c(4.065801,1.660266,0.605405),
  "Payoff At K=4" = c(4.182604, 1.786724,0.6324536)
)
DT
```

```
##    Stock Price Payoff At K=2 Payoff At K=3 Payoff At K=4
## 1:          36     3.9900180      4.065801     4.1826040
## 2:          40     1.3750980      1.660266     1.7867240
## 3:          44     0.4870358      0.605405     0.6324536
```

t = 1 100000 simulations with 100 steps each

```
DT1 <- data.table(
  "Stock Price" = c("36", "40", "44"),
  "Payoff At K=2" = c(3.980887, 1.355297, 0.5066987),
  "Payoff At K=3" = c(4.065187, 1.782886, 0.8470582),
  "Payoff At K=4" = c(4.253372, 2.096174, 0.556851)
)
DT1
```

```
##    Stock Price Payoff At K=2 Payoff At K=3 Payoff At K=4
## 1:          36     3.9808870     4.0651870      4.253372
## 2:          40     1.3552970     1.7828860      2.096174
## 3:          44     0.5066987     0.8470582      0.556851
```

t = 2 100000 simulations with 100 steps each

```
DT2 <- data.table(
  "Stock Price" = c("36", "40", "44"),
  "Payoff At K=2" = c(3.989478,1.427179,0.6266209),
  "Payoff At K=3" = c(4.078398,1.805302,0.9089303),
  "Payoff At K=4" = c(4.252536,2.210652,1.208593)
```

```
)
DT2
```

```
##    Stock Price Payoff At K=2 Payoff At K=3 Payoff At K=4
## 1:          36     3.9894780     4.0783980      4.252536
## 2:          40     1.4271790     1.8053020      2.210652
## 3:          44     0.6266209     0.9089303      1.208593
```

(b)Hermite polynomials t = 0.5 100000 simulations with 100 steps each

```
DT <- data.table(
  "Stock Price" = c("36", "40", "44"),
  "Payoff At K=2" = c(4.170322,1.764724,0.6194262),
  "Payoff At K=3" = c(4.1855,1.784501,0.6221356),
  "Payoff At K=4" = c(4.219372,1.795214,0.6332847)
)
DT
```

```
##    Stock Price Payoff At K=2 Payoff At K=3 Payoff At K=4
## 1:          36     4.1703220     4.1855000     4.2193720
## 2:          40     1.7647240     1.7845010     1.7952140
## 3:          44     0.6194262     0.6221356     0.6332847
```

t = 1 100000 simulations with 100 steps each

```
DT1 <- data.table(
  "Stock Price" = c("36", "40", "44"),
  "Payoff At K=2" = c(4.424671,2.272033,1.09008),
  "Payoff At K=3" = c(4.477422,2.305847,1.112054),
  "Payoff At K=4" = c(4.483433,2.317392,1.105327)
)
DT1
```

```
##    Stock Price Payoff At K=2 Payoff At K=3 Payoff At K=4
## 1:          36      4.424671      4.477422      4.483433
## 2:          40      2.272033      2.305847      2.317392
## 3:          44      1.090080      1.112054      1.105327
```

t = 2 100000 simulations with 400 steps each

```
DT2 <- data.table(
  "Stock Price" = c("36", "40", "44"),
  "Payoff At K=2" = c(4.73921, 2.80891 ,1.65054),
  "Payoff At K=3" = c(4.81431,2.88348,1.69033),
  "Payoff At K=4" = c(4.84891,2.89473, 1.695247)
)
DT2
```

```
##    Stock Price Payoff At K=2 Payoff At K=3 Payoff At K=4
## 1:          36       4.73921       4.81431      4.848910
## 2:          40       2.80891       2.88348      2.894730
## 3:          44       1.65054       1.69033      1.695247
```

(c) Monomial t = 0.5 100000 simulations with 100 steps each

```
DT <- data.table(
  "Stock Price" = c("36", "40", "44"),
  "Payoff At K=2" = c(4.15788, 1.77400, 0.62463),
  "Payoff At K=3" = c(4.1899, 1.79160,0.63149),
```

```
  "Payoff At K=4" = c(4.21016,1.79566, 0.62618)
)
DT
```

```
##    Stock Price Payoff At K=2 Payoff At K=3 Payoff At K=4
## 1:          36       4.15788       4.18990       4.21016
## 2:          40       1.77400       1.79160       1.79566
## 3:          44       0.62463       0.63149       0.62618
```

t = 1 100000 simulations with 100 steps each

```
DT1 <- data.table(
  "Stock Price" = c("36", "40", "44"),
  "Payoff At K=2" = c(4.41960,2.26678,1.09120),
  "Payoff At K=3" = c(4.46201,2.30797,1.11393),
  "Payoff At K=4" = c(4.48446,2.31708,1.10063)
)
DT1
```

```
##    Stock Price Payoff At K=2 Payoff At K=3 Payoff At K=4
## 1:          36       4.41960       4.46201       4.48446
## 2:          40       2.26678       2.30797       2.31708
## 3:          44       1.09120       1.11393       1.10063
```

t = 2 100000 simulations with 100 steps each

```
DT2 <- data.table(
  "Stock Price" = c("36", "40", "44"),
  "Payoff At K=2" = c(4.75640, 2.82641, 1.66047),
  "Payoff At K=3" = c(4.84874, 2.87321, 1.68933),
  "Payoff At K=4" = c(4.84457, 2.88137, 1.65334)
)
DT2
```

```
##    Stock Price Payoff At K=2 Payoff At K=3 Payoff At K=4
## 1:          36       4.75640       4.84874       4.84457
## 2:          40       2.82641       2.87321       2.88137
## 3:          44       1.66047       1.68933       1.65334
```

Comment: As we can observe from the result, results from Monomial and Hermit are quite similiar and converges nicely. Method Lagurre is more volatile and different as compared to the other methods and does not converge nicely as the k increases. Also we might encounter the case where the matrix A is not positive definite, hence making it impossible to invert. This is especially true when the stock price is above strike price and you might get only 1 or 0 values available to execute linear regression in the column.

There are a few ways: 1. Normalize the matrix, 2. set a very small number as threshold for determinant of the matrix and only invert the matrix when this criteria is met. Also skip the calculation when there is only one y and x for linear regression analysis. The second method turns out to be just fine.

Qn2

Set Number of simulations to be 100000, and path to be 100 each I choose to use Hermit and the method to approximate linear regression coefficient With k=3.

2(a) The price of the Europeal Forward-start options is \$3.21

2(b) The price of the American Forward-start option is \$3.42