Problem 1. Numerical integration

Consider the *Gauss–Laguerre* quadrature method. This method approximates the integral of a real–valued function $f : \mathbb{R} \to \mathbb{R}$ using:

$$\int_a^b f(x)dx \approx \sum_{k=1}^{N} w_k f(x_k)$$

In this expression the numbers $\{x_1, x_2, \ldots, x_N\}$ are the roots of the Laguerre polynomial $L_N(x)$ of order N:

$$L_N(x) := \sum_{k=0}^{N} \frac{(-1)^k}{k!} \binom{N}{k} x^k$$

The weights w_k are obtained using the derivative of $L_N(x)$ evaluated at all the points $\{x_1, x_2, \ldots, x_N\}$:

$$w_k = \frac{e^{x_k}}{x_k} \left(\frac{N!}{L'_N(x_k)} \right)^2, \quad for\ k = 1,2,...,N$$

a) Using $N = 2$ please approximate numerically the following integral using the Gauss-Lagueere quadrature:

$$\int_0^4 e^{-\frac{x^2}{2}} dx$$

$$\int_0^4 e^{-\frac{x^2}{2}} dx = \frac{x^2}{2} - 2x + 1$$

$$\Rightarrow \quad x_1 = 2 + \sqrt{2}$$

$$x_2 = 2 - \sqrt{2}$$

$$L'_N(x) = x - 2$$

$$w_1 = \frac{e^{x_1}}{x_1} \left(\frac{2!}{x_1 - 2} \right)^2 = 17.8039$$

$$w_2 = \frac{e^{x_2}}{x_2} \left(\frac{2!}{x_2 - 2} \right)^2 = 6.1324$$

$$\Rightarrow \int_0^4 e^{-\frac{x^2}{2}} dx = w_1 f(x_1) + w_2 f(x_2) = 17.8039 \times 0.00294 + 6.1324 \times 0.84234 = 5.2179$$

b) Let f be a real–valued continuous and integrable function. Assume you have the code for this function written and you can make calls to it by just writing *f(x)*. Write pseudo code that will approximate $\int_0^M f(x)dx$ using the Gauss– Laguerre quadrature, where *M* is a given constant. Also assume that you can call a function polyroot that computes the roots of any given polynomial.

Find all root

For k = 0 to N do:

$$Coff = \frac{(-1)^k}{k!}\binom{N}{k}$$

Solve the root:

$$X - array = ployroot(Coff)$$

Calculate wk

For i = 1 to N do:

$$result = \sum_{i=1}^{N} Coff[i]ix$$

Final result:

For i = 1 to N do:

$$X_i = X - array[i]$$

$$w_i = \frac{e^{x_i}}{x_i}\left(\frac{N!}{result}\right)^2$$

$$final = \sum_{i=i}^{N} w_i f(x_i)$$

Final is the finally result.

Problem 2. Binomial tree

Here is the pseudo-code for the valuation of a European Put option using a multiplicative binomial tree. Please identify all the mistakes in the code below. Please indicate how to fix the mistakes.

```
Data: Parameters K, T, S, r, N, u, d
dt = T/N
p = (exp(r*dt)-d)/(u-d)
disc = exp(r*dt)        → disc=exp(-r*dt)
St[0] = S*d             → St[0]=S*d^N
for j = 1 to N do
|  St[j] = St[j-1]*u/d
end
for j = 0 to N do
|  C[j] = max (St[j] - K, 0.0)   → C[j]=max(k-St[j],0.0)
end
for i = (N-1) down to 0 do
|  for j = 0 to i do
|  |  C[j] = disc * ( (1-p) * C[j] + p * C[j+1])
|  end
end
European_put = C[0]
```

Problem 3. Option pricing using a trinomial tree

Construct a trinomial tree to price an American put option. To this end start with the following given parameters: $S_0 = 100$, $K = 120$, maturity $T = 8$ months, $r = 0$, $\delta = 0$, volatility $\sigma = 3\%$, time steps $N = 2$.

a) What is a suitable choice for Δx to obtain a stable scheme? Calculate Δx

According to restriction idea,

$\Delta x \geq \sigma\sqrt{3\Delta t}$

We assume $\Delta x = \sigma\sqrt{3\Delta t}$,

$$T = \frac{8}{12} = \frac{2}{3},\ N = 2,\ \Delta t = \frac{T}{N} = \frac{1}{3} \ \Rightarrow$$

$$\Delta x = 0.03 \times \sqrt{3 \times \frac{1}{3}} = 0.03$$

b) With your choice of Δx calculate p_u, p_m and p_d.

$$D = r - \frac{\sigma^2}{2} = 0 - \frac{0.03^2}{2} = -0.00045$$

$$p_u = \frac{1}{2}\left(\frac{\sigma^2\Delta t + D^2\Delta t^2}{\Delta x^2} + \frac{D\Delta t}{\Delta x}\right)$$

$$= \frac{1}{2}\left(\frac{0.03^2 \times \frac{1}{3} + (-0.00045)^2 \times (\frac{1}{3})^2}{0.03^2} + \frac{-0.00045 \times \frac{1}{3}}{0.03}\right)$$

$$= \frac{1}{2}(0.3889 - 0.005) = 0.19195$$

$$p_m = 1 - \frac{\sigma^2\Delta t + D^2\Delta t^2}{\Delta x^2}$$

$$= 1 - \frac{0.03^2 \times \frac{1}{3} + (-0.00045)^2 \times (\frac{1}{3})^2}{0.03^2}$$

$$= 1 - 0.3889 = 0.6111$$

$$p_d = \frac{1}{2}\left(\frac{\sigma^2\Delta 4 + D^2\Delta t^2}{\Delta x^2} - \frac{D\Delta t}{\Delta x}\right)$$

$$= \frac{1}{2}\left(\frac{0.03^2 \times \frac{1}{3} + (-0.00045)^2 \times (\frac{1}{3})^2}{0.03^2} - \frac{-0.00045 \times \frac{1}{3}}{0.03}\right)$$

$$= \frac{1}{2} \times (0.3889 + 0.005) = 0.19695$$

c) Calculate the American Put option price using the tree. It will be helpful to draw a diagram of the trinomial tree containing the probabilities and the stock values.

$Discount\ factor = e^{(-r\times\Delta t)} = 1$

		100*exp(0.06) max(120-100*exp(0.06),0)=14
	100*exp(0.03) 1*(0.19195*14+0.6111*17+0.19695*20)	100*exp(0.03) max(120-100*exp(0.03),0)=17
100	100 1*(0.19195*17+0.6111*20+0.19695*23)	200 max(120-100,0)=20
	100*exp(-0.03) 1*(0.19195*20+0.6111*23+0.19695*26)	100*exp(-0.03) max(120-100*exp(-0.03),0)=23
		exp(-0.06) max(120-100*exp(-0.06),0)=26

$option\ price = 1 \times (0.19195 \times (0.19195 * 14 + 0.6111 * 17 + 0.19695 * 20)$
$+0.6111 \times (0.19195 * 17 + 0.6111 * 20 + 0.19695 * 23) + 0.19695 \times$
$(0.19195 * 20 + 0.6111 * 23 + 0.19695 * 26))=$
$1 \times (3.26641315 + 12.231165 + 4.63082626) = 20.1284059$

Perhaps due to input errors, I have calculated many times, but the results are not completely consistent, so the option price obtained by using the code to calculate and verify again is 20. The results are as follows.

```
TRINOMIAL_TREE <- function(S0,r,sigma,K,T,dvd,N,C_or_P){
  # set parameter
  dt <- T/N
  v <- r-dvd-(1/2)*sigma^2
  dx <- 0.03
  pu <- 0.19195
  pm <- 0.6111
  pd <- 0.19695
  discount_factor <- exp(-r*dt)
  C_or_P_factor <- ifelse(C_or_P,1,-1)

  stock_matrix <- matrix(nrow=(2*N+1), ncol=N+1, byrow=FALSE)
  stock_matrix[N+1,1] <- S0
  for (j in 2:(N+1)-1){
    for (i in (N+1-j+1):(N+1+j-1)){
      stock_matrix[i+1,j+1] <- stock_matrix[i,j]*exp(dx)
      stock_matrix[i,j+1] <- stock_matrix[i,j]
      stock_matrix[i-1,j+1] <- stock_matrix[i,j]*exp(-dx)
    }
  }
```

```
  option_matrix <- stock_matrix
  for (i in 1:(2*N+1)){
    option_matrix[i,N+1] <- max(0, C_or_P_factor*(stock_matrix[i,N+1]-K))
  }
  for (j in N:1){
    for (i in (N+1-j+1):(N+1+j-1)){
      option_matrix[i,j] <- discount_factor*(pu*option_matrix[i-1,j+1]+
                                               pm*option_matrix[i, j+1]+
                                               pd*option_matrix[i+1,j+1])
      for (i in (2*N+1):1){
        option_matrix[i,j] <- max(option_matrix[i,j],
                                  C_or_P_factor*(stock_matrix[i,N+1]- K))
      }
    }
  }
  print(option_matrix[N+1,1])
}
TRINOMIAL_TREE(S0=100,r=0.00,sigma=0.03,K=120,T=30,dvd=0.00,N=2,C_or_P=FALSE)
```

```
  [1] 20
```

Problem 4.Finite Difference method for PDE

We know that an option price under a certain stochastic model satisfies the following PDE:

$$\frac{\partial V}{\partial t} + 2tan(S)\frac{\partial V}{\partial S} + S^2\frac{\partial^2 V}{\partial S^2} - rV = 0$$

Assume you have an equidistant grid with points of the form $(i, j) = (i\Delta t, j\Delta x)$, where $i \in \{1,2,...,N\}$ and $j \in \{-N_S, N_S\}$. Let $V_{i,j} = (i\Delta t, j\Delta x)$.

a) Discretize the derivatives and give the finite difference equation for an Explicit scheme. Use the notation introduced above. Please simplify the final expressions so that the unknown quantity is on the left hand side and all the known quantities are on the right hand side.

$$\frac{\partial V}{\partial t} = \frac{V_{i+1,j} - V_{i,j}}{\Delta t}$$

$$\frac{\partial V}{\partial x} = \frac{V_{i+1,j+1} - V_{i+1,j-1}}{2\Delta x}$$

$$\frac{\partial^2 V}{\partial x^2} = \frac{V_{i+1,j+1} + V_{i+1,j-1} - 2V_{i+1,j}}{\Delta x^2}$$

$$\Rightarrow \frac{V_{i+1,j}-V_{i,j}}{\Delta t}+2tan(S)\frac{V_{i+1,j+1}-V_{i+1,j-1}}{2\Delta x}+S^2\frac{V_{i+1,j+1}+V_{i+1,j-1}-2V_{i+1,j}}{\Delta x^2}=rV_{i+1,j}$$

$$V_{i,j}=V_{i+1,j}+tan(S)\frac{V_{i+1,j+1}-V_{i+1,j-1}}{\Delta x}\Delta t+S^2\frac{V_{i+1,j+1}+V_{i+1,j-1}-2V_{i+1,j}}{\Delta x^2}\Delta t-rV_{i+1,j}\Delta t$$

$$V_{i,j}=\left(1-\frac{2S^2}{\Delta x^2}\Delta t-r\Delta t\right)V_{i+1,j}+\left(\frac{tan(S)}{\Delta x}\Delta t+\frac{S^2}{\Delta x^2}\right)V_{i+1,j+1}+\left(-\frac{tan(S)}{\Delta x}\Delta t+\frac{S^2}{\Delta x^2}\Delta t\right)V_{i+1,j-1}$$

b) Derive the discretized equation for the Implicit scheme. Please simplify the final expressions so that the unknown quantities are on the left hand side and the known quantity is on the right hand side. You do not need to solve the corresponding matrix equation.

$$V_{i+1,j}=rV_{i,j}\Delta t+V_{i,j}-tan(S)\frac{V_{i+1,j+1}-V_{i+1,j-1}}{\Delta x}\Delta t-S^2\frac{V_{i+1,j+1}+V_{i+1,j-1}-2V_{i+1,j}}{\Delta x^2}\Delta t$$

$$V_{i+1,j}=\left(1+\frac{2S^2}{\Delta x^2}\Delta t+r\Delta t\right)V_{i,j}-\left(\frac{tan(S)}{\Delta x}\Delta t+\frac{S^2}{\Delta x^2}\right)V_{i,j+1}+\left(\frac{tan(S)}{\Delta x}\Delta t-\frac{S^2}{\Delta x^2}\Delta t\right)V_{i,j-1}$$

$$A=-\left(\frac{tan(S)}{\Delta x}\Delta t+\frac{S^2}{\Delta x^2}\right)$$

$$B=\left(1+\frac{2S^2}{\Delta x^2}\Delta t+r\Delta t\right)$$

$$C=\left(\frac{tan(S)}{\Delta x}\Delta t-\frac{S^2}{\Delta x^2}\Delta t\right)$$

For a Call option, denote $j = 2Nj + 1$ is the total segments of one column:

$$V_{i+1,N_j+1}-V_{i+1,N_j}=\Delta S=S_{i+1,N_j+1}-S_{i+1,N_j}$$

$$V_{i+1,2}-V_{i+1,1}=0$$

For a Put option, denote N is the total nots of one column:

$$V_{i+1,N_j}-V_{i+1,N_j-1}=0$$

$$V_{i+1,2}-V_{i+1,1}=\Delta S=-\left(S_{i+1,N_j+1}-S_{i+1,Nj}\right)$$

For call option:

$$\begin{bmatrix} 1 & -1 & 0 & 0 & \dots & 0 & 0 \\ A & B & C & 0 & \dots & 0 & 0 \\ 0 & A & B & C & \dots & 0 & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & \ddots & \ddots & B & C & 0 \\ 0 & 0 & 0 & \ddots & A & B & C \\ 0 & 0 & 0 & \dots & 0 & 1 & -1 \end{bmatrix} \times \begin{bmatrix} V_{i,N_j} \\ V_{i,N_j-1} \\ V_{i,N_j-2} \\ \vdots \\ V_{i,-N_j+2} \\ V_{i,-N_j+1} \\ V_{i,-N_j} \end{bmatrix} = \begin{bmatrix} \Delta S \\ V_{i+1,N_j-1} \\ V_{i+1,N_j-2} \\ \vdots \\ V_{i+1,-N_j+2} \\ V_{i+1,-N_j+1} \\ 0 \end{bmatrix}$$

For put option:

$$\begin{bmatrix} 1 & -1 & 0 & 0 & \dots & 0 & 0 \\ A & B & C & 0 & \dots & 0 & 0 \\ 0 & A & B & C & \dots & 0 & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & \ddots & \ddots & B & C & 0 \\ 0 & 0 & 0 & \ddots & A & B & C \\ 0 & 0 & 0 & \dots & 0 & 1 & -1 \end{bmatrix} \times \begin{bmatrix} V_{i,N_j} \\ V_{i,N_j-1} \\ V_{i,N_j-2} \\ \vdots \\ V_{i,-N_j+2} \\ V_{i,-N_j+1} \\ V_{i,-N_j} \end{bmatrix} = \begin{bmatrix} 0 \\ V_{i+1,N_j-1} \\ V_{i+1,N_j-2} \\ \vdots \\ V_{i+1,-N_j+2} \\ V_{i+1,-N_j+1} \\ \Delta S \end{bmatrix}$$