JPMorgan Chase Quant Finance Mentorship Program 2024

General Instructions:

This document contains two case studies each with equal score (100 marks each).

You are expected to attempt both case studies and submit the solution by the deadline 11:59 PM 21st January 2024.

1. Each of these case studies introduce the basics of a problem and pose a few questions on them. A question might have multiple sub-parts, all of which need to be answered. The scores for the sub-parts have been indicated against the question. The solution format for each question has been specified alongside the question and the final submission format is described at the next page of this document.

2. You are free to use online resources to improve your understanding of the problem statement.

3. Hand-written answers are also accepted but need to be collated in a single pdf while submitting. Loose snapshots will not be accepted.

4. You are supposed to work individually on this case study. Collusion/collaboration of any kind will not be tolerated.

5. Only 1 submission per candidate is allowed.

Plagiarism of any kind will not be tolerated.

Final Submission Format:

Create a well formatted word/pdf document describing your solution to each question, clearly enumerated and in the same order as the questions. Include your personal details such as name, institute name, branch and year in the first page.

For each question, if the expected solution is a written answer/ plot / table or pseudo code, include it in line in this document. Name this document in the format.

{FirstName}_{LastName}_{CollegeName(short)}_JPMCQuantMentorshipCaseStudy If

the expected solution is a program, state the file name of the program.

Instructions for program submissions:

Any programming language you are comfortable with is permitted. You are permitted to use standard libraries in your chosen language.

a. The program submitted must compulsorily have a main function that must call all the test cases described in the question when run.

b. The program must be commented and intelligible.

c. The program's file name must be of the format: {CANDIDATE_NAME}_{QUESTION_NO_SUB_PART}_MAIN.

d. If there are additional modules that are imported in the script with the main function, name those files as {CANDIDATE_NAME}_Module_{ SCRIPT_NAME}.

e. You may include a short description of your program in the solution document if you wish.

f. Compress the PDF and all code files in a zip and name it 'JPMC_Quant_Mentorship_CaseStudy_<YourFirstName>_<YourLastName>' and mail it to jpmqrmentorship.mumbai@jpmorgan.com with the subject 'JPMC_Quant_Mentorship_CaseStudy_Submission_<YourFirstName>_<YourLastName>'

Derivatives

Theory:

Currencies:

Currency describes the money or official means of payment in a country or region. The best-known currencies include the U.S. dollar, Euro, Japanese Yen, British pound and Swiss franc. Currency symbols exist for most currencies, such as ₹, $, €, ¥ or £. Different financial markets, however, use ISO (International Organization for Standardization) codes to identify currencies. Some of these ISO codes are INR (Rupee), USD (U.S. Dollar), JPY (Japanese Yen), EUR (Euro) and GBP (British Pound).

International trade and investment involving billions of dollars would not be possible without the ability to buy and sell foreign currencies. Currencies must be bought and sold because the INR or USD is not the acceptable means of payment in most other countries. Investors, tourists, importers and exporters must exchange dollars for foreign currencies, and vice versa.

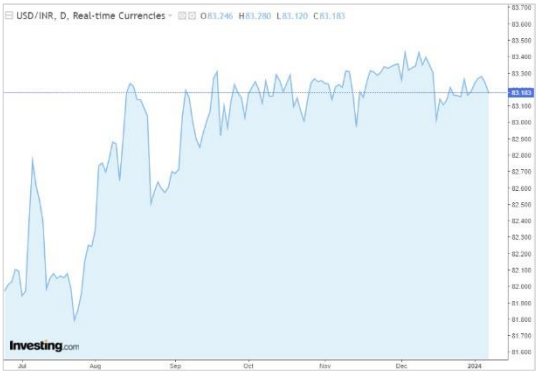
Foreign Exchange Markets:

The trading of currencies takes place in foreign exchange markets whose primary function is to facilitate foreign trade and investment. Foreign exchange (Forex or FX) is the conversion of one currency into another at a specific rate known as the foreign exchange rate. The conversion rates for almost all currencies are constantly floating as they are driven by the market forces of supply and demand. Higher the demand, greater the price; and higher the supply, lower the price.

The foreign exchange market is a decentralized market where all currency exchange trades occur. It is the largest (in terms of trading volume) and the most liquid market (can be efficiently or easily converted into ready cash without affecting its market price) in the world.

Foreign Exchange Rates

An exchange rate is a rate at which one currency will be exchanged for another currency and affects trade and the movement of money between countries.



If the USD/INR currency pair is 83.33, that means it costs 83.33 Indian Rupee to get 1 U.S. dollar. In USD/INR, the first currency listed (USD) always stands for one unit of that currency; the exchange rate shows how much of the second currency (INR) is needed to purchase that one unit of the first (USD).

This rate tells you how much it costs to buy one U.S. dollar using the Indian Rupee. To find out how much it costs to buy one Indian Rupee using U.S. dollars, use the following formula: 1/exchange rate: 1 / 83.33=0.012

It costs 0.012 U.S. dollars to buy one Indian Rupee. This price would be reflected by the INR/USD pair. In this instance, the position of the currencies has switched.

Exchange rates have what is called a spot rate, or cash value, which is the current market value at which a currency pair can be bought or sold. Alternatively, an exchange rate may have a forward value, which is based on expectations for the currency to rise or fall versus its spot price.

Forward rate values may fluctuate due to changes in expectations for future interest rates in one country versus another. If traders speculate that the eurozone will ease monetary policy versus the U.S., they may buy the dollar versus the euro, resulting in a downward trend in the value of the euro.

Spot Exchange Rate Example

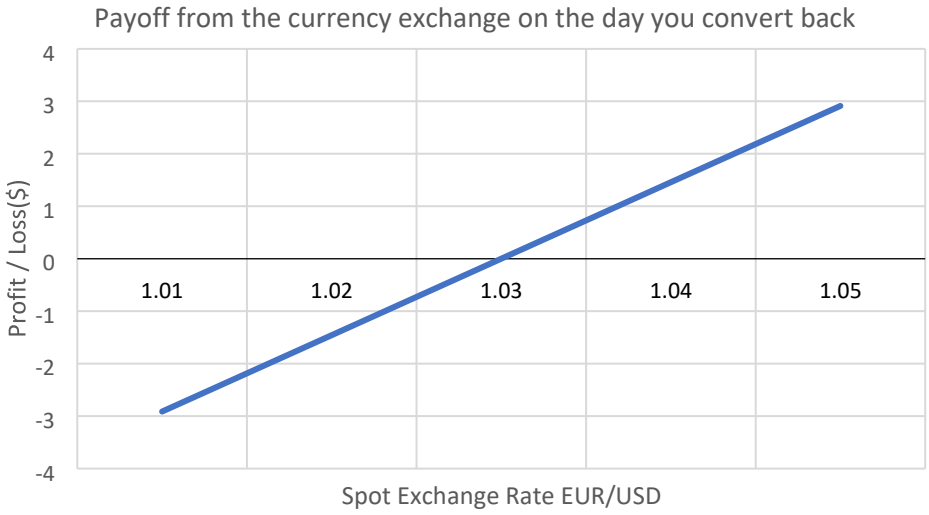
A traveler to Germany from the U.S. wants 150 USD worth of EUR when arriving in Germany. The sell rate is the rate at which a traveler sells foreign currency in exchange for local currency. The buy rate is the rate at which one buys foreign currency back from travelers to exchange it for local currency.

If the current exchange rate (EUR/USD) is 1.03, $150 will net €145.63 in return.

In this case, the equation is: dollars ÷ exchange rate = euro

$150 ÷ 1.03 = €145.631

In three months, after the trip is over, suppose he didn't spend any money and wants to convert the money back to dollars. If the exchange rate has dropped to 1.01, the change from euros to dollars will be *€145.631 x 1.01 = $147.087* and you would have made a loss of ≈ $3. Likewise, if the exchange rate rises to 1.05, the change from euro to dollar would be *€145.631 x 1.05 = $152.912* and would have made a profit of ≈ $3. This can be used to make money!



Are spot exchanges the only way to participate in these markets? Read further sections to find out.

Derivatives:

In the previous example, you went on the trip with EUR/USD exchange rate 1.03 and came back with exchange rate 1.01 and bared a loss. This presents an opportunity to use this platform to make profit. You

can purchase a certain currency in the hope of the exchange rate rising, so that you can sell it at a higher rate than what you bought for. But nobody can guarantee on what happens three years from now.

Theoretically speaking, the exchange price can go as low as 0.515 and you can lose half your money. Now what if there was some way which you could use to floor your losses? This is where derivatives come in. 'Options, Futures and Other Derivatives', an extremely popular book on derivatives by John C. Hull, defines a derivative to be a financial instrument whose value depends on (or derives from) the values of other, more basic, underlying variables. A forex option, for example, is a derivative whose value is dependent on the price of a foreign exchange rate. Let's look at some famous derivatives –

Futures:

A future contract is an agreement between two parties to buy or sell an asset at a certain time in the future at a certain price. Once you enter a future contract, you are obligated to buy (i.e. take a long position) or sell (i.e. take a short position) at a pre-specified price (called futures price) on a certain date in the future (3 months in our example), irrespective of the prevailing market price (also called the spot price) on the expiry date.

Currency futures are an exchange-traded futures contract that specify the price in one currency at which another currency can be bought or sold at a future date. They can be used to hedge other trades or currency risks, or to speculate on price movements in currencies.

Foreign exchange futures contracts comprise several components outlined below:

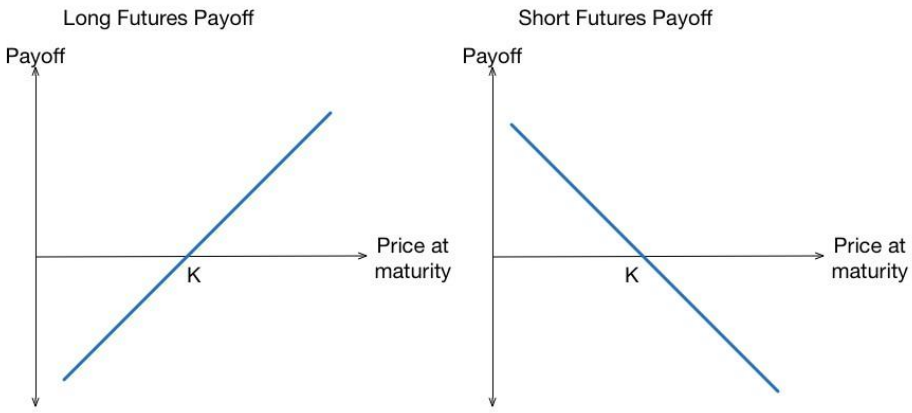
Underlying Asset – This is the specified currency exchange rate.
Expiration Date – For cash-settled futures, this is the last time it is settled. For physically delivered futures this is the date the currencies are exchanged
Size – Contracts sizes are standardized. For example, a euro currency contract is standardized to 125,000 euros.
Margin Requirement – To enter into a futures contract, an initial margin is required. A maintenance margin will also be established and if the initial margin falls below this point, a margin call will happen, meaning the trader or investor must deposit money to bring it above the maintenance margin.

Let us now look at an example that involves currency futures. Say you purchase 8 future Euro contracts (€125,000 per contract) at 0.9 US$ per €. This implies that at the time of maturity, you can buy 0.9 US$ per € for principal of 8*€125,000. If the exchange rate ends up >= 0.9 US$/€ or <0.9 US$/€, you can still buy the contract at 0.9 US$/€ and realize a profit of the difference in the price.

At the end of the day, the settlement price has moved to 0.92 US$/€. How much have you lost or profited? The price has increased meaning you have profited. The calculation to determine how much you have profited is as follows:

(0.92 US$/€ – 0.90 US$/€) x €125,000 x 8 = 20,000 US$

In general, the payoff from a long position in a future contract on one unit of currency is S_T–K where K is the strike exchange rate and S is the spot exchange rate of the currency pair at maturity of the contract (left). Similarly, the payoff from a short position in a future contract on one unit of currency is K –S_T (right). Both these pay offs can be positive or negative as can also be seen in the diagrams below



Forwards

A forward contract is an obligation to buy or sell a certain asset:

- At a specified price (forward price)
- At a specified time (contract maturity or expiration date)
- Typically, not traded on exchanges.

Sellers and buyers of forward contracts are involved in a forward transaction – and are both obligated to fulfill their end of the contract at maturity. Forward contracts are regarded as over-the-counter instruments. While this nature of forward contracts makes it easier to customize terms, the lack of a centralized clearinghouse also gives rise to a higher degree of default risk.

Forwards are very similar to future contracts and their payoff graphs also look the same. Like futures, forwards can be used for hedging trades or currency risks, or for speculation on price movements.

Options:

Going back to our earlier example, your goal is to place a bet that the exchange rate goes up after 3 months, and at the same time, you want to floor your losses. A currency option (also known as a forex option) is a contract that gives the buyer the right, but not the obligation, to buy or sell a certain currency at a specified exchange rate on or before a specified date. For this right, a premium is paid to the seller.

Basic terminology for currency options:

Premium – The upfront cost of purchasing a currency exchange option.
Strike Price – The strike (or exercise price) is the price at which the option holder has the right to buy or sell a currency.
Expiry Date – The trade's expiry date is the last date on which the rights attached to an option may be exercised.
Delivery Date – The date when the currency exchange will take place, if the option is exercised.

There are two main types of options, calls and puts.

Call options:

They provide the holder the right (but not the obligation) to purchase an underlying asset at a specified price (the strike price), for a certain period of time. If the stock fails to meet the strike price before the expiration date, the option expires and becomes worthless. Investors buy calls when they think the share

price of the underlying security will rise or sell a call if they think it will fall. You can buy a call option by paying a small premium to the seller of the option.

Imagine you bought a call option after paying a small premium. As the holder of the call option, you now have the right to buy the US$ at ₹80, 3 months from now, even if the market price is greater than ₹80. If the exchange rate ends up <= ₹80, you can let your option expire without any action, and the maximum amount you can lose is the initial premium that you paid to buy the option (so the loss is floored)!

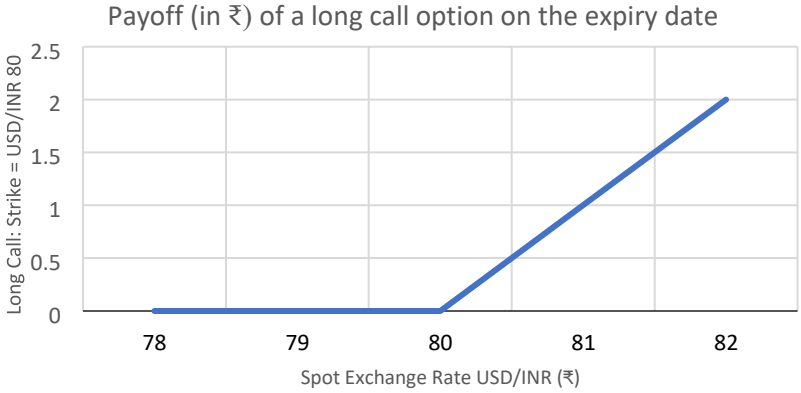
In the above example, you bought a call option in the hope that the exchange rates of the underlying currency pair will go up. What if you wanted to place a bet on the exchange rate going down? In that case, you can sell a call option! You will get paid a small premium by the buyer of the call option, and if your estimate is correct and the stock price does go down, the buyer won't exercise the option and you will get to keep the initial premium paid to you!

One fascinating point to note here is that you don't actually need to convert any currency while selling a call option. When you sell a call option, your only obligation is to give the buyer the underlying exchange at expiry at the predetermined strike price, that too only if the buyer chooses to exercise the call option. And so, you have the liberty to not have the underlying exchange currency with you until the expiry date arrives! Thus, your initial investment is very minimal unlike the earlier case where you were exchanging currency at the current market price today and hoping that it would increase in price 3 months later.

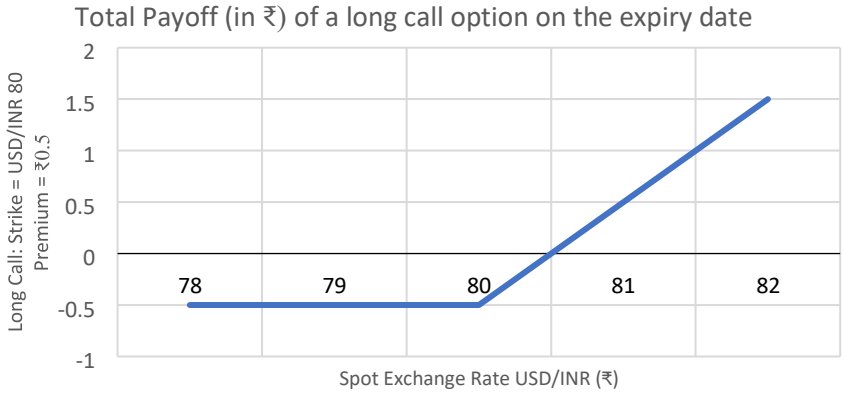
Note: 'Buying' of an asset is termed as a 'long position' and 'Selling' of an asset is termed as a 'short position'.

See the below diagrams to see how the payoff looks like on the date of expiry for the call option:

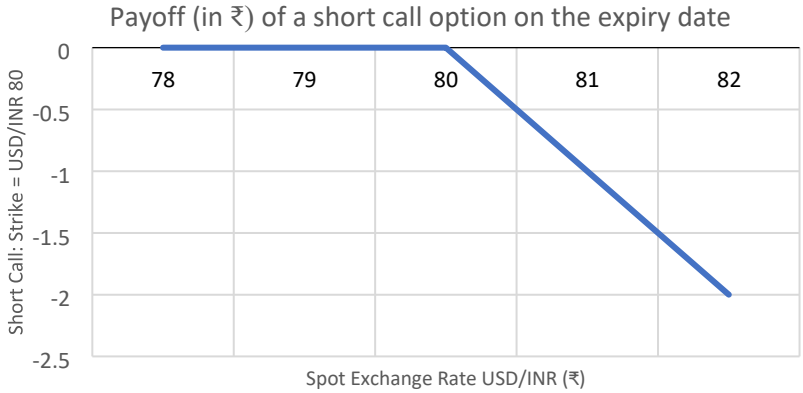
a. Long call with strike = USD/INR ₹80 has the payoff as C = max (exchange rate at expiry – strike, 0)



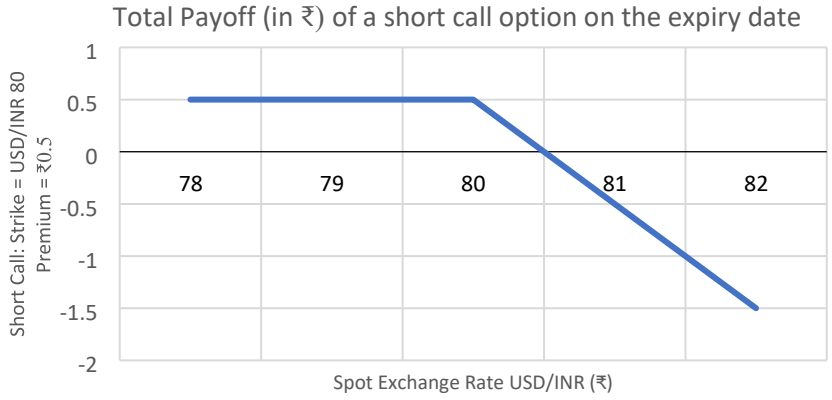
b. Assuming that the initial premium to buy the call option is ₹0.5, the above payoff graph gets shifted down by the premium amount.



c. Similarly, short call (selling a call option) with strike= USD/INR ₹80 has the payoff of long call reflected on the x-axis



d. As this time, you received the initial premium (assume ₹0.5) from the buyer of the option, the payoff graph will get shifted up by the premium amount.



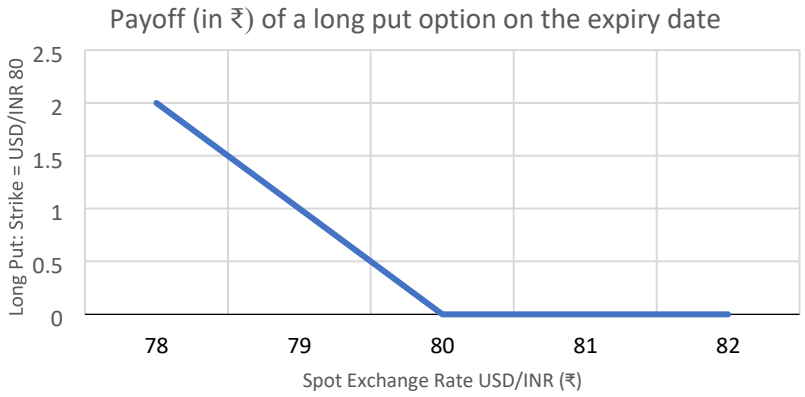
Put options:

Just like call options, a put option give the holder the right to sell an underlying asset at a specified price (the strike price). You can buy a put option by paying a small premium to the seller of the option.

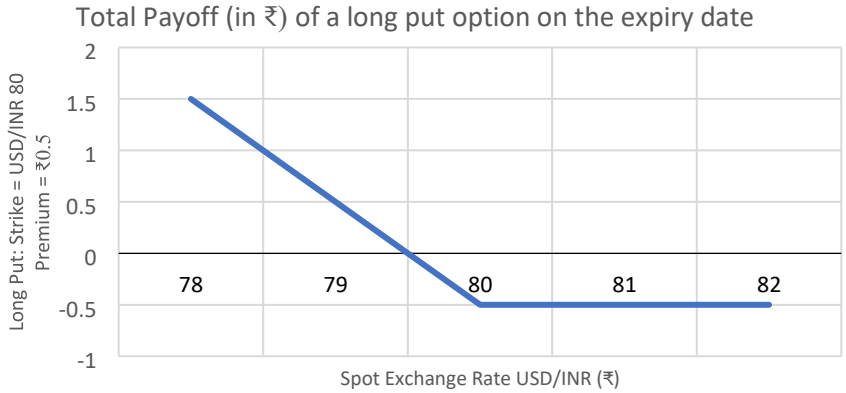
Imagine you bought a put option after paying a small premium. As the holder of the put option, you now have the right to exchange INR to USD at ₹80 per USD, 3 months from now, even if the market price is <₹80. If the stock price ends up >= ₹80, you can let your option expire without any action, and the maximum amount you can lose is the initial premium that you paid to buy the option.

See the below diagrams to see how the payoff looks like on the date of expiry for the put option:

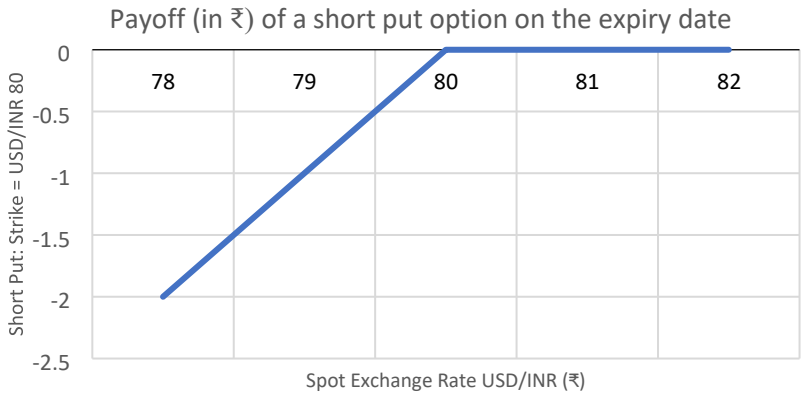
a. Long put with strike = USD/INR ₹80 has the payoff as P = max (strike - exchange rate at expiry, 0)



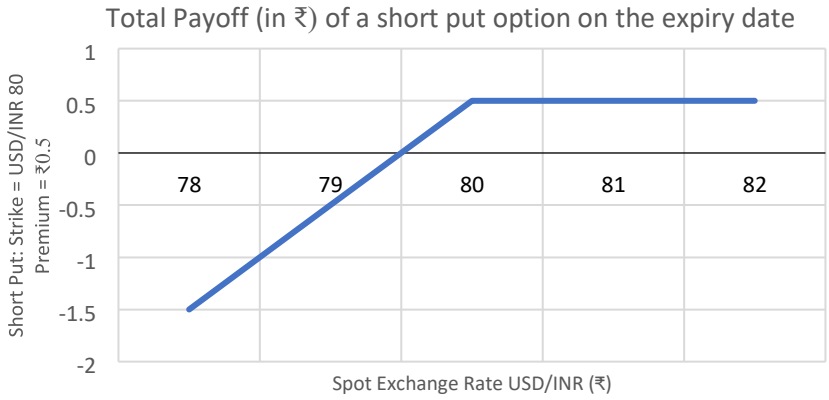
b. Assuming that the initial premium to buy the put option is ₹0.5, the above payoff graph gets shifted down by the premium amount.



c. Similarly, short put (selling a put option) with strike= USD/INR ₹80 has the payoff of long put reflected on the x-axis



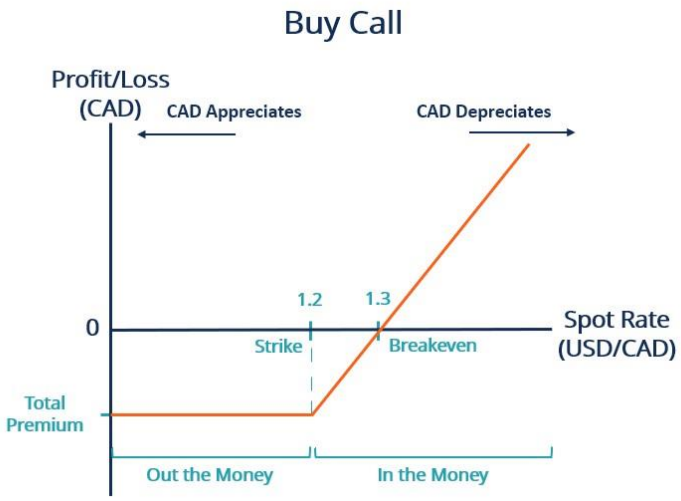
d. As this time, you received the initial premium (assume ₹0.5) from the buyer of the option, the payoff graph will get shifted up by the premium amount.



American vs. European Options:

An American option can be exercised any point up to the expiration date whereas a European option can only be exercised at maturity.

An option that would be profitable to exercise at the current exchange rate is said to be in-themoney. Conversely, an out-of-money option is one that would not be profitable to exercise at the current exchange rate. An option whose strike price is same as the spot exchange rate is termed as at-the-money.



Now let us consider a more realistic scenario where a trader holds more than one instrument in his portfolio. The trader holds a combination of different instruments to hedge against the market risk for different scenarios.

An appropriate hedging strategy can minimize the exposure to currency fluctuations and provide stability to future earnings and expected cash flows. The objective of a proper hedge is to eliminate the uncertainty of futures transactions denominated in a foreign currency, not to maximize profits from currency speculation. A successful hedge will therefore not produce excess returns but will protect the hedger against losses resulting from unfavorable exchange rate fluctuations.

Consider the below given combination to understand how the payoffs of the following combinations look like and then we will try constructing the payoffs of few combinations on our own.

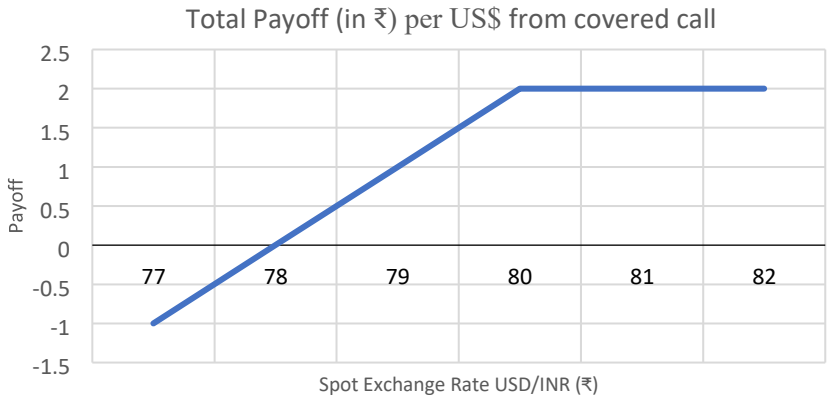
Covered call is a strategy in which the seller of a call option also holds an equivalent amount of the underlying security. The investor holding the long position in an asset sells call options on same asset to generate income. This strategy is often employed by those who wish to hold the underlying asset for long time but do not expect major increment in price in near future and is ideal for people not expecting major fluctuations in prices of the underlying asset.

Let us understand how we design the payoff of the covered call. Let us assume you have purchased exchanged currency USD to INR with rate USD/INR ₹80, hoping that the rate will go up to ₹81 in next 6 months with no significant volatility expected. To balance of the profits, you parallelly sell 1 call option with the strike price of USD/INR ₹81 expiring in next 6 months. Assume that the premium on this call is ₹2 per $ in contract. Your pay off in the future will depend on the price of stock in next 6 months.

A) **Exchange rate remains at USD/INR ₹80:** In this case, the call option will not be exercised as the strike rate matches market rate. As the rate remains same, we don't get any return from forex but earn the premium per USD on the call.

B) **Exchange rate increases to USD/INR ₹81**: If the exchange rate increases to USD/INR ₹81 after 6 months, the call will be exercised, and you will receive ₹81 per $ and ₹2 per $ from premium.

C) **Exchange rate decreases to USD/INR ₹79:** This case is similar to the scenario 1. In this case we lose ₹1 per $ but this loss will be reduced by the premium of ₹2 per $ that we receive from the call option.



Problem Statements:

Total Marks: 100

There are three questions in the case study. The weightage for each question has been mentioned individually, along with the format of the solution that is expected for each question. A common instruction for all questions is to show your work; your journey is almost as important as your destination!

Q1. Forward and futures contracts are examples of derivatives that involve the agreement between two parties to buy and sell an underlying asset at a specific price by a certain date. These contracts serve a similar function of hedging risk that can arise due to uncertainty in future prices. Hence, they can be confused with each other but there are certain differences between them.

Answer the following questions. Solutions are expected in any format (pdf, images, handwritten, doc, etc.)

(a) Can you list three of these differences? (6 marks)
(b) Which contract is exposed to counterparty risk? (4 marks)

Q2. Suppose on January 1, an American company makes a sale for which it will receive €140,000 on March 1. The firm will want to convert these euros into dollars, so it is exposed to the risk that the euro can fall below its current spot price ($1.4890) before March. However, the firm wants to minimize the risk of a fluctuating euro by using a hedging strategy (using a forward/futures or an options contract). Help the firm choose the correct hedging strategy between buying/selling a future or an options contract.

Assume a declining exchange rate scenario where the spot price falls to $1.460 on March 1. Use options with strike price just above and below the January 1 spot exchange rate ($1.48 and $1.50). Refer to the following table:

Price	**January 1**	**March 1**
Spot Rate	$1.4890	$1.460
Futures Price	1.4910	1.4640
$1.48 strike call/put price	0.0055	0.0251
$1.50 strike call/put price	0.0120	0.0430

(a) Plot graphs to show how the net profit/loss varies with the exchange rate for long and short futures (10 marks)
(b) Plot graphs to show how the net profit/loss varies with the exchange rate for short put options at both strikes. (10 marks)
(c) What is the difference between a long put and a short call option. Plot the expected payoffs associated with them to explain your answer. (12 marks)
(d) Which hedging strategy should the firm employ? Or should the firm remain unhedged? Plot the expected payoffs associated with them to explain your answer. (15 marks)

In the above graphs, mark the strike rate, premium of the options/future price and the expected payoff using the above exchange rate scenario.

We don't expect you to code each plot but how you came to the plot is most important so please make sure that you submit the idea behind the curve. Solutions can be submitted in any format (pdf, images, handwritten, pseudocode, code files, etc.)

Q3. Suppose Akshita is a trader who expects a rise in the price of an asset and wishes to profit from this increment. She can buy a call option at a strike price K_1 for her purpose. She also pays a premium for buying this call option equal to C_1. The asset is currently trading at price P_1.

(a) Plot the expected payoffs for this call option with P_1=\$35, K_1=\$35, C_1=\$3. Is it an in-the-money, out-of-the-money, or at-the-money option? (10 marks)

(b) Suppose Akshita wishes to reduce her possible risk from this purchase and decides to get creative with her strategy. She decides to sell the same number of calls of the same asset at a higher strike price K_2 (that is, $K_2>K_1$). Akshita receives a premium equal to C_2 for selling this call option. Both options have the same expiration date. Plot the expected payoff for this strategy with P_1=\$35, K_1=\$35, C_1=\$3, K_2=\$37, C_2=\$2. (20 marks)

(c) By how much should the price of the underlying asset rise for Akshita to break even in the case mentioned in (b)? (8 marks)

(d) What are the limitations of Akshita's strategy mentioned in (b)? (5 marks)

The aim of this case study is to give you a glimpse into how trading strategies are designed. Don't fret, let's start now to deep dive into the world of trading. We don't expect you to code each plot but how you came to the plot is most important so please make sure that you submit the idea behind the curve that you make. Solutions are expected in any format (pdf, images, handwritten, code files, etc.)

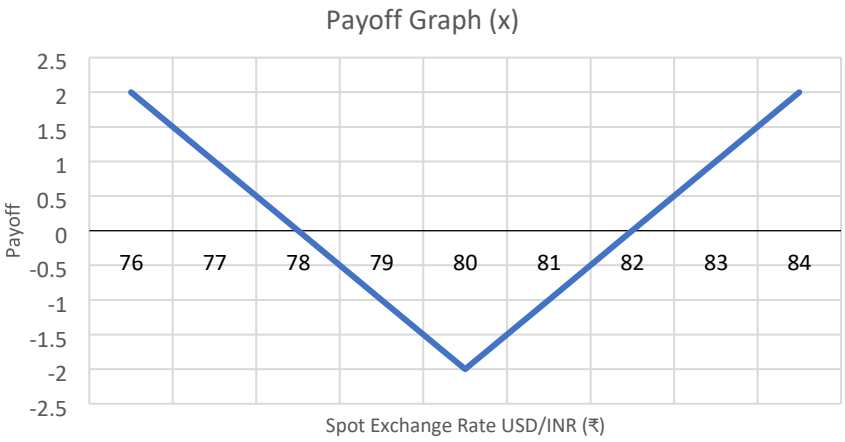
Bonus Question:

The following problem will not be evaluated, it is intended for encouraging a deeper understanding of derivatives.

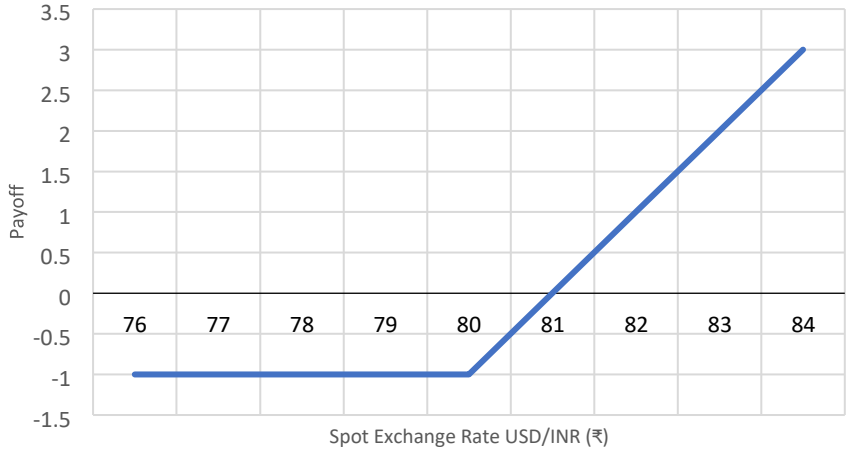
We will try to understand how trading strategies are formulated.

Let us look at an example. Here, the trading is done for USD/INR exchange. The current exchange rate is USD/INR ₹80 and the option premium call/put is ₹1. We want to achieve the payoff described in plot (x).

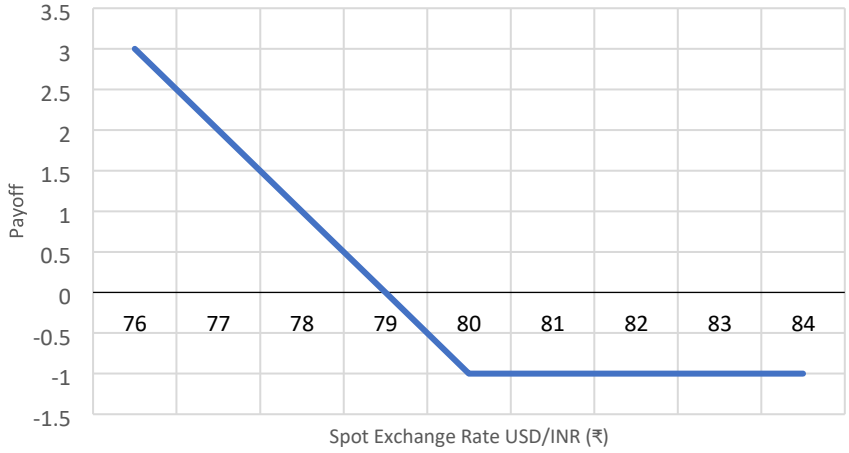
For this, we will try to use a combination of derivatives.



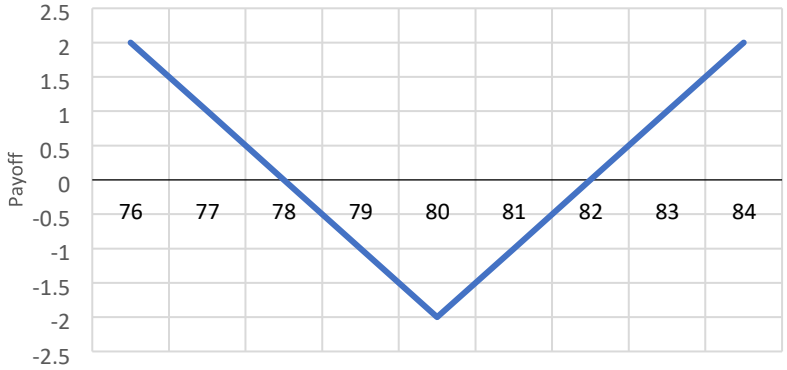
1. First, we can observe that the final payoff is in upward growing direction, indicating a long call option. Also, that the graph is symmetric at the center with minima at ₹80. With a premium of ₹1, Lets first take a long call option at strike ₹80.



2. The starting part of the graph is a decreasing graph, resembling a long put option. So, we take a long put option at strike ₹80, with a premium of ₹1.



3. Looking at the two graphs, it can be noticed that if we simply add them both, ₹80 will become a minima as both graphs will put its payoff down to -₹2. And due to the flat parts of the plots, the forward declining and the backward growing directions will be retained. So lets try to see the payoff of buying a long call and a long put option at strike ₹80. (Shown on the right). It matches with the intended payoff!



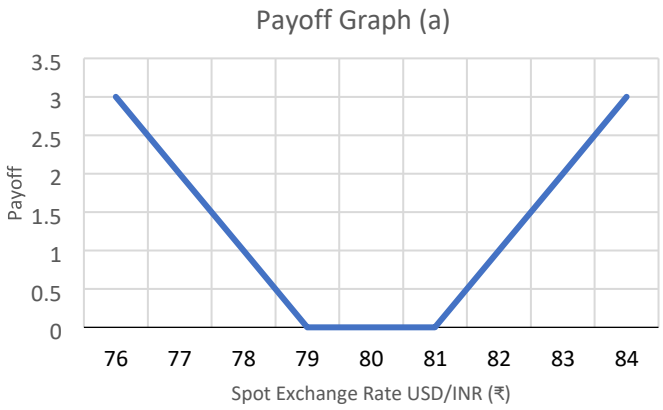
4. This seems like a good strategy when we expect the price of the underlying asset to move substantially. The payoff will be positive if the exchange rate increases or decreases from ₹80, and payoff will be higher if the change in rate is higher.

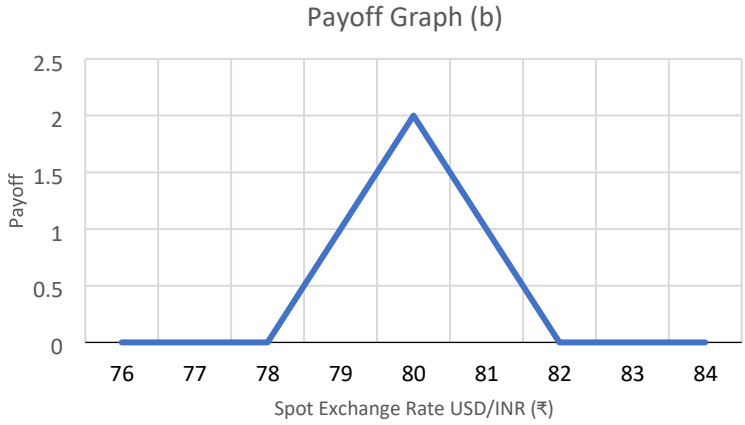
Now try to solve the following problem.

Taking the same USD/INR exchange, the current exchange rate USD/INR ₹80 and the option premium call/put is ₹1, look at the plots (a) and (b), and answer the following questions.

i. Find the combination of derivatives used to obtain these payoffs. ii. Reproduce the plots given in (a) and (b) using your answer in (i).
iii. What are the benefits of using such trading strategies.

Feel free to use future/option derivatives, at any strike rate (Not bizarre strike rates, which are very far from the current exchange rate).





Note that there may not be a unique solution to this question.

We don't expect you to code each plot but how you came to the plot is most important so please make sure that you submit the idea behind the curve that you make. Solutions are expected in any format (pdf, images, handwritten, code files, etc.)

Programming

1 Introduction to hierarchical clustering

Clustering algorithms aim to assign objects to clusters such that similar objects go to similar clusters, for some notion of similarity.
Hierarchical clustering aims to cluster objects on a level-by-level basis (like a tree) where lower levels contain more granular clusters and clusters become coarser as we move towards the root node, the root node having a single cluster containing all objects and leaf nodes being clusters of individual objects. The idea behind is that even in a group of similar objects some objects will be more similar to each other so we can divide the group further.

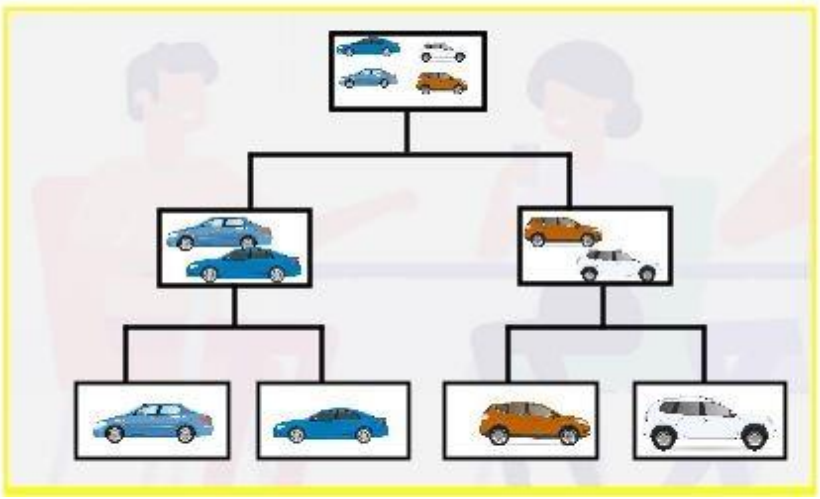


Figure 1: First level clustering of blue cars and SUVs, and then 2nd level clustering of all cars

Hierarchical clustering can happen in two directions:

- Divisive: We start with all objects in a cluster and then make more granular subgroups based on similarity.
- Agglomerative: We start with all objects in their own cluster and then group them based on similarity.

For this case study we will be looking to implement agglomerative clustering on stocks based on the similarity of their returns. Doing hierarchical clustering analysis for stocks can expose non-trivial market structure, i.e. we might find that some stocks that we might not expect to be ”similar” actually are and we can thus use that information to more effectively inform our positions.

2 Algorithm overview

We start with an $N \times N$ distance matrix, i.e. a symmetric matrix of pairwise distances. Below is an example matrix (Figure 2) that we will use to demonstrate the algorithm. We want to start grouping objects on the basis of ”similarity”, the distance matrix we construct is based on our notion of similarity, i.e. the less distance between two entries the more ”similar” they are.
We will first identify the minimum element of the distance matrix which is $d(P_3,P_6)$. Our first cluster

	P1	P2	P3	P4	P5	P6
P1	0					
P2	0.23	0				
P3	0.22	0.15	0			
P4	0.37	0.20	0.15	0		
P5	0.34	0.14	0.28	0.29	0	
P6	0.23	0.25	0.11	0.22	0.39	0

Figure 2: Note that diagonal elements are zero and only lower triangular elements are reported because the matrix is symmetric

	P1	P2	P3	P4	P5	P6
P1	0					
P2	0.23	0				
P3	0.22	0.15	0			
P4	0.37	0.20	0.15	0		
P5	0.34	0.14	0.28	0.29	0	
P6	0.23	0.25	0.11	0.22	0.39	0

Figure 3: P3 and P6 are the closest objects

will be formed by combining P_3 and P_6. Now that we have identified what cluster is to be formed we have to update the matrix with distances of the remaining objects to the newly formed cluster, as shown in Figure 4.

We calculated the distances to the newly formed cluster using the formula $d(C_1, C_2) = \min_{p \in C_1, q \in C_2} d(p, q)$ where C_1 and C_2 are clusters (single objects can be treated as clusters of size 1), basically the distance between two clusters is the same as the distance between their closest members. The distance formula is called the linkage function, we can have different linkage functions some examples are:

- Single linkage: $d(C_1,C_2) = \min_{p \in C_1, q \in C_2} d(p,q)$ this is the one we will be using in our example.
- Average linkage: $\frac{\sum_{p \in C_1, q \in C_2} d(p,q)}{n_1 \cdot n_2}$ where n_1 and n_2 are the size of clusters C_1 and C_2 respectively.
  We will be using this linkage function in later problem statements.

Note that in the above $d(p,q)$ denotes the distance as in the original $N \times N$ distance matrix.

Now, we repeat this procedure of picking out the smallest entry and making a new cluster from the objects corresponding to them. This continues as shown in Figures 5-8. We continue till there is only one cluster left containing all the objects.

	P1	P2	P3,P6	P4	P5
P1	0				
P2	0.23	0			
P3,P6	0.22	0.15	0		
P4	0.37	0.20	0.15	0	
P5	0.34	0.14	0.28	0.29	0

Figure 4: New matrix is of $N-1 \times N-1$ dimension. For example $d((P_3,P_6),P_2) = min(d(P_3,P_2),d(P6,P2)) = min(0.15,0.25) = 0.15$

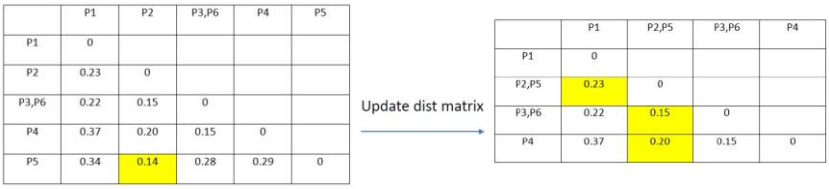


Figure 5: Smallest entry corresponds to $(P2,P5)$ so we make them a cluster. For example $d((P_3,P_6),(P_2,P_5)) = min(d(P_3,P_2),d(P_3,P_5),d(P_6,P_2),d(P_6,P_5)) = min(0.15,0.28,0.25,0.39) = 0.15$

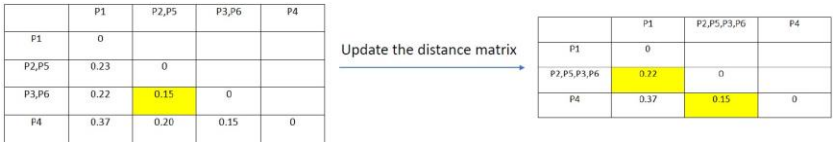


Figure 6: Smallest entry corresponds to $((P_2,P_5),(P_3,P_6))$ so we make them a cluster.

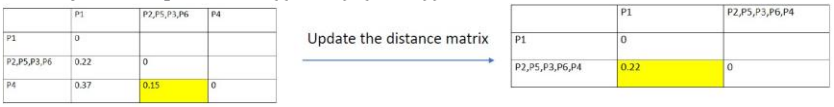


Figure 7: Smallest entry corresponds to $((P_2,P_3,P_5,P_6),P_4)$ so we make them a cluster.

	P1	P2,P5,P3,P6,P4
P1	0	
P2,P5,P3,P6,P4	0.22	0

Figure 8: Smallest entry corresponds to $((P_2,P_3,P_4,P_5,P_6),P_1)$ so we make them a cluster. Algorithm will terminate here since all elements are now part of the same cluster.

This is how the agglomeration happens. Now, based on the chronology of merging we can construct a "dendrogram" this is is just a pictorial representation of the cluster forming process. For our example the dendrogram looks like Figure 9. We can make the "hierarchical" distance matrix using the dendrogram by following the steps below, let's say we want to find the distance between P_i and P_j:

- We first look for the LCA (Lowest Common Ancestor) corresponding to P_i and P_j; For ex- for 4

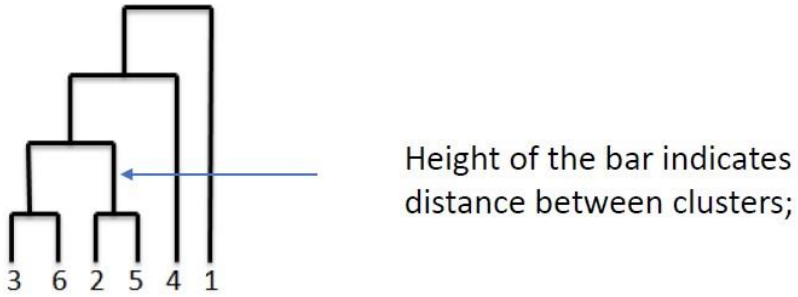


Figure 9: First 2 clusters formed were (P_2,P_5) and (P_3,P_6) so they are merged first. Then (P_2,P_5) and (P_3,P_6) are merged, and so on.

and 6 the LCA would be the cluster formed by merging (P_2,P_3,P_5,P_6) and P_4.

- Once we have the LCA, the "hierarchical" distance between P_i and P_j would be the same as the distance between C_i and C_j where C_i and C_j are the clusters that were merged to form the LCA; Continuing the example- $d_H(P_4,P_6) = d((P_2,P_3,P_5,P_6),P_4) = 0.15$ where d_H is the hierarchical distance.

3 Submission Instructions

- Your final submission will be in the form of a zip folder with name "{FirstName} _{LastName} {CollegeName} Programming", inside this folder there should be 5 sub-folders, 1 corresponding to each problem statement; below are the details on what these folders should contain:
  - First folder should be named "PS1" and should contain 2 .pdf files containing your solutions for the two subparts of Problem Statement 1, i.e. one each corresponding to the single and average linkage functions, you can submit the .pdf file by scanning handwritten notes or other ways.
  - Second folder should be named "PS2" and should contain the implementation, in the language of your choice, for Problem Statemen 2. The program should be able to read from "inp.txt" and write to "inv.txt" files that will be added into the same folder at the time of evaluation.
  - Third folder should be named "PS3" and should contain the implementation, in the languauge of your choice, for Problem Statement 3. The program should be able to read from "inp.txt" and write to "intermediateDistMats.txt" files that will be added into the same folder at the time of evaluation.
  - Fourth folder should be named "PS4" and should contain the implementation, in the language of your choice, for Problem Statement 4. It should be able to read from "inp.txt" and write to "dendrogram.txt" files that will be added in the same directory at the time of evaluation.
  - Fifth folder should be named "PS5" and should contain the implementation, in the language of your choice, for Problem Statement 5. It should be able to read from the "inp.txt" and write to "hDist.txt" files that will be added in the same directory at the time of evaluation.
- You are NOT required to submit the input and output files that you are using to test your implementation locally.

4 Problem Statement 1 [25 Points]

For stocks we construct the distance matrix using the correlation of their daily returns. The correlation of daily returns is a measure of their similarity, we can then transform the correlation to a distance√

metric using some transformation; for ex- $d = 1 - \rho$ where ρ is the correlation and d is the distance. Your task for this problem statement is to carry out the hierarchical clustering procedure by hand for a given 5 × 5 distance matrix as below (Figure 10), and report the following:

- The dendrogram structure implied by the matrix.
- The pairwise ”hierarchical” distance matrix between the objects calculated using the methodology described in the Algorithm overview section.

Repeat both of the above using single and average linkage functions. So you have to report 2 dendrograms 1 each corresponding to the linkage functions as well as 2 distance matrices each.

	0	1	2	3	4
0	0.000	0.967	1.035	1.083	0.956
1	0.967	0.000	1.129	1.116	0.966
2	1.035	1.129	0.000	0.443	1.002
3	1.083	1.116	0.443	0.000	1.072
4	0.956	0.966	1.002	1.072	0.000

Figure 10: Initial matrix for Problem statement 1. In practice, these numbers are based on some transformation of the correlation of daily stock returns.

5 Problem Statement 2 [12 Points]

For this problem statement you have to implement some utility I/O functions that you will require in further problem statements. You have to implement a program that reads from a file named ”inp.txt”, this file will contain an $N \times N$ array where numbers in a row are separated by commas and rows are separated by newline (”\n”) characters, additionally the first row will have row labels (0,1,2..,$N - 1$) and first column will have the column labels, again (0,1,2..,$N - 1$); the element at top left corner (0,0) position will be a ”#” character, see example to understand input format. The program should then create one output file named ”inv.txt” that should contain the inverse of the matrix you have read, written in the same format as the matrix present in ”inp.txt”, see example to understand input and output formatting.

5.1 Example

Let’s say ”inp.txt” contains:

```
#,0,1,2,3,4
0,4.904,-0.011,-0.768,0.944,-1.156
1,-0.011,0.571,-0.032,-0.141,-0.13
2,-0.768,-0.032,0.768,-0.371,0.151
3,0.944,-0.141,-0.371,0.844,-0.319
4,-1.156,-0.13,0.151,-0.319,0.656
```

Then ”inv.txt” should contain:

```
#,0,1,2,3,4
0,0.417,0.157,0.255,-0.074,0.67
1,0.157,2.181,0.413,0.738,0.972
2,0.255,0.413,1.858,0.783,0.485
3,-0.074,0.738,0.783,2.05,0.832
4,0.67,0.972,0.485,0.832,3.191
```

Some things to keep in mind:

- You can use the numpy.linalg.inv function to invert the matrix. Note that this is the *ONLY* problem statement in which you can use some library function, for the rest of the problem statements you are prohibited from using library functions from numpy, sklearn, scipy, etc.
- For the output file "inv.txt", you have to round the inverse matrix to 3 decimal places you can do this using the "numpy.round", i.e. something like "finalOutputMat = numpy.round(inverseMat, 3)" will give you the inverse matrix rounded to 3 decminal places and then you have to print this matrix to "inv.txt" in the format specified above.

Points breakdown is as follows:

- 10 points for correctness of outputs. Note that the outputs will be validated by an automated script so please keep the rounding rule in mind.
- 2 points for code hygiene, i.e. suitable variable naming, readability, modularity, etc.

6 Problem Statement 3 [21 Points]

You have to implement a program that reads from a file named "inp.txt", this file will contain an $N \times N$ array in the same format as Problem Statement 2, you are encouraged to import the utility I/O function from the solution for Problem Statement 2. The program should then create one output file named "intermediateDistMats.txt", below are the details about what these files should contain. See last section "Summary of things to be submitted" for folder structure, naming conventions, etc.
Using the array in "inp.txt" as the input distance matrix, your program needs to perform agglomerative hierarchical clustering using the single linkage function as described in the "Algorithm Overview" section and produce the following:

- The sequence of intermediate matrices that will occur during the process of agglomerative clustering, as explained in the Algorithm Overview section.
- The rows and column labels should be such that:
  - Members of clusters are sorted within the cluster, i.e. (0,1,5) is a valid label but (1,0,5) is not.
  - Clusters are sorted lexicographically, i.e. if labels are 0,(2,3,5),(1,4) then correct order of printing should be 0,(1,4),(2,3,5). Note that ordering of labels also affect the ordering of the corresponding rows and columns so be careful.
- There should be no white spaces between consecutive matrices, only consecutive rows should be separated by newline ("\n") character. (The "#" character will denote the beginning of a new matrix and will be used by the automatic checking script)
- The output matrix should be rounded to 3 decimal places.

Example below will demonstrate the formatting rules. Since an automated script will be used to evaluate the output for this problem these rules must be strictly followed.

6.1 Example

Let's say "inp.txt" contains:

```
#,0,1,2,3,4,5
0,0.0,0.23,0.22,0.37,0.34,0.23
1,0.23,0.0,0.15,0.20,0.14,0.25
2,0.22,0.15,0.0,0.15,0.28,0.11
3,0.37,0.20,0.15,0.0,0.29,0.22
4,0.34,0.14,0.28,0.29,0.0,0.39
5,0.23,0.25,0.11,0.22,0.39,0.0
```

Notice that this matrix is the same as the matrix we used in the Algorithm Overview section. ”intermediateDistMats.txt” should therefore contain:

```
#,0,1,2,3,4,5
0,0.0,0.23,0.22,0.37,0.34,0.23
1,0.23,0.0,0.15,0.20,0.14,0.25
2,0.22,0.15,0.0,0.15,0.28,0.11
3,0.37,0.20,0.15,0.0,0.29,0.22
4,0.34,0.14,0.28,0.29,0.0,0.39
5,0.23,0.25,0.11,0.22,0.39,0.0
#,0,1,(2,5),3,4
0,0.0,0.23,0.22,0.37,0.34
1,0.23,0.0,0.15,0.20,0.14
(2,5),0.22,0.15,0.0,0.15,0.28
3,0.37,0.20,0.15,0.0,0.29
4,0.34,0.14,0.28,0.29,0.0
#,0,(1,4),(2,5),3
0,0.0,0.23,0.22,0.37
(1,4),0.23,0.0,0.15,0.20
(2,5),0.22,0.15,0.0,0.15
3,0.37,0.20,0.15,0
#,0,(1,2,4,5),3
0,0.0,0.22,0.37
(1,2,4,5),0.22,0.0,0.15
3,0.37,0.15,0.0
#,0,(1,2,3,4,5)
0,0.0,0.22
(1,2,3,4,5),0.22,0.0
#,(0,1,2,3,4,5)
(0,1,2,3,4,5),0.0
```

Note that in the above example the input matrix had a precision of 2 decimal places, which is why the example output is also precise upto 2 decimal places. All final test cases will have input matrices that are precise upto 3 places.
Points breakdown is as follows:

- 5 points are for the code being able to correctly print the 0th step (original matrix) and 1st step (matrix after 1st merge).
- 7 points are for the code being able to correctly print the 2nd step (matrix after 2nd merge).
- 7 points are for the code being able to correctly print the whole sequence of intermediate matrices.
- 2 points for code hygiene, i.e. suitable variable naming, readability, modularity, etc.

Also, note that there will be NO additional sample cases provided by us in the form of ”inp.txt” files, you are free to create your own. You don’t need to submit sample test cases generated by you. Additionally, you are not allowed to use numpy, sklearn, scipy, etc; your implementation should be completely original, bruteforce implementations are acceptable.

7 Problem Statement 4 [21 Points]

With the completion of Problem Statement 3 you can now track the chronology of merges and the resultant intermediate distance matrices that occured in the process of agglomerative clustering, using this chronology you now have to create the dendrogram structure and report it as described below. You have to implement a program that reads from a file named ”inp.txt”, this file will contain an $N \times N$ array in the same format as Problem Statement 2. The program should then create an output file named

”dendrogram.txt”, below are the details about what this file should contain. See last section ”Summary of things to be submitted” for folder structure, naming conventions, etc.

- Using the chronology of merges, generate the level-wise representation of the dendrogram corresponding to the data with each level in a new line, nodes at the same level are to be sorted according to the formatting rules mentioned below. This should be written to the ”dendrogram.txt” file.
- Example is provided below that you can use to understand the expected output formatting and also for testing out your code.
- You are encouraged to reuse code from Problem Statement 3.

Since an automated script will be used to evaluate the output for this problem these rules must be strictly followed:

- Data should be printed such that all the clusters have their members arranged in ascending order, i.e. (0,1,4,6) is a valid cluster but (4,1,0,6) is invalid.
- In the case of one level containing multiple clusters, or a mix of clusters and single numbers, a lexicographical order should be followed; for ex- if a level has the following members (0,2) (3,5,7) 1 4 6 then the correct order of printing them would be (0,2) 1 (3,5,7) 4 6.

7.1 Example

Let’s say ”inp.txt” contains:

```
#,0,1,2,3,4
0,0.0,1.285,0.544,1.201,0.793
1,1.285,0.0,1.246,0.752,1.140
2,0.544,1.246,0.0,1.199,0.825
3,1.201,0.752,1.199,0.0,1.123
4,0.793,1.140,0.825,1.123,0.0
```

Then ”dendrogram.txt” should contain:

```
(0,1,2,3,4)
(0,2,4) (1,3)
(0,2) 1 3 4 0
2
```

Some things to note:

- Round brackets must be used to denote clusters.
- Inside the cluster commas must be used to separate the values.
- Spaces must be used to separate clusters from clusters or clusters from values or values from values.
- Note that as mentioned in the problem statement, values within the clusters are sorted, as well as if a level contains a mix of clusters and values then a lexicographical order is followed.

Explanation: See Figure 11 for the dendrogram that is formed by the given distance matrix. The levels corresponding to each node are marked, nodes with the same level are printed in the same line sorted as described in the formatting rules.

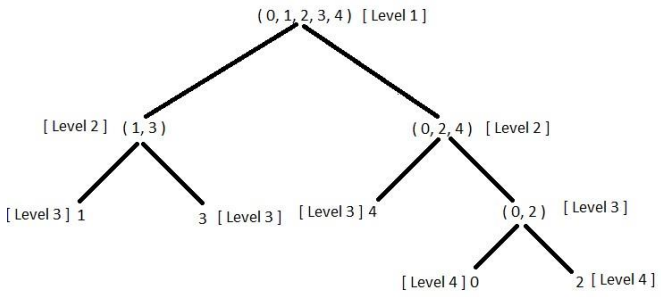


Figure 11: Dendrogram corresponding to given matrix.

Points breakdown is as follows:

- 19 points are for the correctness of the code.
- 2 points for code hygiene, i.e. suitable variable naming, readability, modularity, etc.

Also, note that there will be NO additional sample cases provided by us in the form of ”inp.txt” files, you are free to create your own. You don’t need to submit sample test cases generated by you. Additionally, you are not allowed to use numpy, sklearn, scipy, etc; your implementation should be completely original, bruteforce implementations are acceptable.

8 Problem Statement 5 [21 Points]

In Problem Statement 4, you have successfully implemented a dendrogram representation. Your task for this problem statement is to now implement the final step of our algorithm to get a hierarchical matrix, i.e. an LCA finding algorithm that will search your dendrogram representation for the pairwise LCA and populate the ”hierarchical” distance matrix as described in the Algorithm Overview section. You have to implement a program that reads from a file named ”inp.txt”, this file will contain an $N \times N$ array in the same format as previous problem statements. The program should then create an output file named ”hDist.txt”, below are the details about what this file should contain. See last section ”Summary of things to be submitted” for folder structure, naming conventions, etc.

- The ”hierarchical” distance matrix calculated as described in the Algorithm Overview. This should be written in the ”hDist.txt” file in the same format as ”inp.txt”.
- You are encouraged to reuse code from previous problems.

The following formatting rules should be kept in mind:

- You should output the hierarchical distance matrix in the same format as the input matrix is provided in ”inp.txt”.
- Also, the numbers in the output matrix should be rounded to 3 decimal places, as in below example.

8.1 Example

Let’s say ”inp.txt” contain:

```
#,0,1,2,3,4
0,0.0,1.285,0.544,1.201,0.793
1,1.285,0.0,1.246,0.752,1.140
2,0.544,1.246,0.0,1.199,0.825
3,1.201,0.752,1.199,0.0,1.123
4,0.793,1.140,0.825,1.123,0.0
```

Then ”hDist.txt” should contain:

#,0,1,2,3,4
0,0.0,1.123,0.544,1.123,0.793
1,1.123,0.0,1.123,0.752,1.123
2,0.544,1.123,0.0,1.123,0.793
3,1.123,0.752,1.123,0.0,1.123
4,0.793,1.123,0.793,1.123,0.0

This is the matrix obtained by carrying out the LCA procedure as mentioned in the Algorithm Overview section; for ex- $d_H(0,1)$, i.e the entry at (0,1) in the above matrix will be calculated as:

- LCA of 0 and 1 is (0,1,2,3,4). (Observable in Figure 11)
- The clusters that were merged to form the LCA were (1,3) and (0,2,4). Therefore, $d_H(0,1) = d((1,3),(0,2,4))$.
- According to our linkage function,

$$\begin{aligned} & d((1,3),(0,2,4)) \\ & \quad = \min\{d(0,1),d(1,2),d(1,4),d(0,3),d(2,3),d(3,4)\} \\ & \quad = \min\{1.285,1.246,1.140,1.201,1.199,1.123\} = 1.123 \end{aligned} \tag{1}$$

Points breakdown is as follows:

- 19 points are for the correctness of the code.
- 2 points for code hygiene, i.e. suitable variable naming, readability, modularity, etc.

Also, note that there will be NO additional sample cases provided by us in the form of ”inp.txt” files, you are free to create your own. You don’t need to submit sample test cases generated by you. Additionally, you are not allowed to use numpy, sklearn, scipy, etc; your implementation should be completely original, bruteforce implementations are acceptable.