MATH3090 Assignment 2

April 22, 2024

MATH3090 Assignment 2

Student: Hugo Burton (s4698512) Date Due: Tuesday April 23 @ 1pm

[1]:
```python
from colorama import Fore, Style

import bond
import swap
import table

from lattice import BinNode, BinLattice
import interest
import display as dsp
from IPython.display import Latex, display
```

Question 1 [3 Marks]

You have just invested in a 3-year coupon paying bond with 8% semi-annual coupons and a face value $F = \$100,000$. Suppose the coupon-paying bond yield curve is flat at 9%.

Part A [1 Mark]

Calculate the present value and the (absolute value of) duration $|D|$ of the bond.

We have the following from the question

$$\begin{align} y &= 9\% \;\; \text{annually} \tag{1} \\ F &= \$100,000 \tag{2} \\ T &= 3 \;\; \text{years} \tag{3} \\ c &= 8\% \tag{4} \\ n &= 2 \;\; \text{(semiannual)} \tag{5} \end{align}$$

Then, we can derive

$$
\begin{aligned}
C &= c \cdot F \qquad (6)\\
&= 0.08 \cdot \$100,000 \qquad (7)\\
&= \$8,000 \qquad (8)\\
\Rightarrow \frac{C}{2} &= \$4,000 \qquad (9)
\end{aligned}
$$

Now

$$|D| = \sum_{t=1}^{T} \frac{t \cdot \mathrm{PV}_t}{B(y)}$$

where

$$\mathrm{PV}_t := \begin{cases} \frac{\frac{C}{n}}{\left(1+\frac{y}{n}\right)^t}, & t \in [1, T-1] \\ \frac{\frac{C}{n}+F}{\left(1+\frac{y}{n}\right)^T}, & t = T \end{cases}$$

For the duration, we have

$$|D| = \sum_{t=1}^{T} \frac{t \cdot \mathrm{PV}_t}{B(y)} \qquad (10)$$

$$(11)$$

All of these values can be computed using code below:

[2]:
```
# Calculate the present values

T = 3
n = 2
y = 0.09
c = 0.08
F = 100_000

present_values = bond.present_values_coupon_bearing_bond_discrete(
    F, T, c, y, n)

present_val = sum(present_values)

print(f"{Fore.LIGHTGREEN_EX}Present value: ${present_val:.4f}{Style.
  ↪RESET_ALL}\n")

# Bond Duration
```

```
bond_duration = bond.bond_duration_discrete(F, T, c, y, n)

print(f"{Fore.LIGHTGREEN_EX}Bond Duration {bond_duration:.4f} years{Style.
↪RESET_ALL}\n")

print(f"{Fore.LIGHTRED_EX}Please refer to code included in submission for␣
↪working. This also applies to the remaining questions.{Style.RESET_ALL}")
```

```
Present value: $97421.0638

Bond Duration 2.7217 years

Please refer to code included in submission for working. This also applies
to the remaining questions.
```

Part B [1 Mark]

Now calculate the value of the bond in $|D|$ years time.

The value of a bond in $|D|$ years time is given by the formula from slide 44 w4

$$B_D = \frac{\text{FV}}{(1+\frac{y}{2})^{T-|D|}}$$

where

$$\begin{aligned}
\text{FV} &\quad \text{is the face value of the bond (including the last coupon payment)} && (12)\\
y &\quad \text{is the yield to maturity or interest rate} && (13)\\
|D| &\quad \text{is the duration of the bond} && (14)\\
T &\quad \text{is the time to maturity of the bond} && (15)\\
B_D &\quad \text{is the value of the bond at time } |D| && (16)
\end{aligned}$$

Again, this can be computed in code as follows

[3]:
```
n = 2
val_at_d = bond.bond_value_at_time(bond_duration, F, T, c, y, n)

print(f"{Fore.LIGHTGREEN_EX}The bond value at |D| is: {val_at_d:.4f}{Style.
↪RESET_ALL}")
```

```
Time step 1, Year 0.5, Cashflow:   4000.0, Beta:  1.1027, Reinvestment value
4410.9352
Time step 2, Year 1.0, Cashflow:   4000.0, Beta:  1.0787, Reinvestment value
4314.9179
Time step 3, Year 1.5, Cashflow:   4000.0, Beta:  1.0552, Reinvestment value
4220.9907
Time step 4, Year 2.0, Cashflow:   4000.0, Beta:  1.0323, Reinvestment value
```

```
4129.1080
Time step 5, Year 2.5, Cashflow:   4000.0, Beta:  1.0098, Reinvestment value
4039.2255
Time step 6, Year 3.0, Cashflow: 104000.0, Beta:  0.9878, Reinvestment value
102733.7882
--------------------------------------------------
The bond value at |D| is: 123848.9655
```

Part C [1 Mark]

Suppose that, immediately after buying the bond, the yield curve shifted up to be flat at 10%. Now calculate the value of the bond again in $|D|$ years time under the new yield curve (don't calculate D again). Compare your answer with what you obtained in (b).

[4]:
```
# Compute bond value with new interest rate of 10% and compare values
n = 2
y_new = 0.10

val_at_d_yield_shift = bond.bond_value_at_time(bond_duration, F, T, c, y_new, n)

print(f"f{Fore.LIGHTGREEN_EX}The new bond value at |D| with {Fore.
↪LIGHTMAGENTA_EX}y = {y_new:.2f}"+\
      f"{Fore.LIGHTCYAN_EX} is ${val_at_d_yield_shift:.4f}.{Style.RESET_ALL}\n")

# Compare differences

difference = val_at_d_yield_shift - val_at_d
relation = "higher" if difference > 0 else "lower"
print(f"The difference in bond value is ${difference:.3f}. In other words, the␣
↪bond is now valued ${abs(difference):.3f} \n"+ \
      f"{relation} with the yield curve at 10% compared with when it was at 9%.
↪")
```

```
Time step 1, Year 0.5, Cashflow:   4000.0, Beta:  1.1145, Reinvestment value
4457.9612
Time step 2, Year 1.0, Cashflow:   4000.0, Beta:  1.0876, Reinvestment value
4350.5247
Time step 3, Year 1.5, Cashflow:   4000.0, Beta:  1.0614, Reinvestment value
4245.6774
Time step 4, Year 2.0, Cashflow:   4000.0, Beta:  1.0358, Reinvestment value
4143.3568
Time step 5, Year 2.5, Cashflow:   4000.0, Beta:  1.0109, Reinvestment value
4043.5022
Time step 6, Year 3.0, Cashflow: 104000.0, Beta:  0.9865, Reinvestment value
102597.4076
--------------------------------------------------
fThe new bond value at |D| with y = 0.10 is $123838.4300.
```

```
The difference in bond value is $-10.536. In other words, the bond is now valued
$10.536
lower with the yield curve at 10% compared with when it was at 9%.
```

We see the new value with $y = 10\%$ is lower by $\sim \$10.5$. Although the reinvestment value of the coupons before time T is higher when the interest rate is 10%, the majority of the cashflow (i.e. the final payment $C + F$ is yet to come, and being in the future, this cashflow is now discounted at a higher rate now that $y = 10\%$, compared with the previous 9%. As a result, the value of the bond at $|D|$ is actually slightly lower even though interest rates are higher.

Question 2 [7 Marks]

Assume that you observe the following yield curve for government's coupon paying bonds.

- There are a total of 20 bonds
- For the k-th bond, $k = 1, ..., 20$, the maturity is k years.
- The face value is $F = \$100,000$ and the coupon rate for the k-th bond, $k = 1, ..., 20$, is $c = 4\%$. Let $C = cF$.
- The price of the bonds $(P(k), k = 1, 2, ..., 20)$ are given by

$$
\begin{aligned}
&[P(1), P(2), ..., P(20)] && (17)\\
=&[99412, 97339, 94983, 94801, 94699, 94454, 93701, 93674, 93076, 92814, && (18)\\
&\quad 91959, 91664, 87384, 87329, 86576, 84697, 82642, 82350, 82207, 81725]. && (19)
\end{aligned}
$$

- Denote by $y_{0,k}$ the spot zero-coupon bond yield curve, and by $y_{k-1,k}$ the implied one-year forward rates.

Assume that all coupon payments are made annually. Use continuous compounding.

Part A [1 Mark]

Show that

$$y_{0,k} = \frac{1}{k} \log \left(\frac{C + F}{P(k) - C \sum_{j=1}^{k-1} e^{-y_{0,j} \times j}} \right), \qquad 1 \leq k \leq 20.$$

In general, we have for bond k the cashflow

$$\underbrace{C + C + \cdots + C}_{T-1} + C + F$$

over the lifespan of the bond. Therefore, with exception to the final coupon payment, there are $T - 1$ coupon payments. Denote

$$
\begin{aligned}
V_t &:= \frac{C}{e^{y_{0,t}\cdot t}}, \quad \forall t \in [1, T-1] && (20)\\
&= Ce^{y_{0,t}\cdot t}, \quad \forall t \in [1, T-1] && (21)
\end{aligned}
$$

as the value of coupon payment $t \in [1, T-1]$ for any of the k bonds in the question. Next, the stripped price of bond k can be written as follows. Note $T = k$ in our example, hence we can perform the variable substitution $T = k$.

$$
\begin{aligned}
P'(k) &:= P(k) - \sum_{j=1}^{T-1} V_j, \quad \forall k \in [1, 20] && (22)\\
&= P(k) - \sum_{j=1}^{k-1} Ce^{y_{0,j}\cdot j}, \quad \forall k \in [1, 20] && (23)
\end{aligned}
$$

Now equate the stripped price of the bond, $P'(k)$ with the equivalent zero-coupon bond (maturing at $T = k$ years with rate $y_{0,T} = y_{0,k}$ to compute the spot zero-coupon yield rate.

$$
\begin{aligned}
\frac{C+F}{e^{y_{0,T}\cdot T}} &= P'(k) && (24)\\
\frac{C+F}{e^{y_{0,k}\cdot k}} &= P(k) - \sum_{j=1}^{k-1} Ce^{y_{0,j}\cdot j} && (25)\\
e^{y_{0,k}\cdot k} &= \frac{C+F}{P(k) - C\sum_{j=1}^{k-1} e^{y_{0,j}\cdot j}} && (26)\\
y_{0,k}\cdot k &= \ln\left(\frac{C+F}{P(k) - C\sum_{j=1}^{k-1} e^{y_{0,j}\cdot j}}\right) && (27)\\
y_{0,k} &= \frac{1}{k}\ln\left(\frac{C+F}{P(k) - C\sum_{j=1}^{k-1} e^{y_{0,j}\cdot j}}\right), \quad \forall k \in [1, 20] && (28)
\end{aligned}
$$

As given in the question. Note the question uses log, though I prefer to use ln to specify this is the natural log which is derived from using continuous compounding.

Part B [2 Marks]

Implement a Matlab/Python program to compute spot zero-coupon bond yield curve $y_{0,k}$ and the implied one-year forward rates $y_{k-1,k}$. Submit Table 1 filled with computed value.

[5]:
```
bond_prices = [
    99412,
    97339,
    94983,
    94801,
```

```python
    94699,
    94454,
    93701,
    93674,
    93076,
    92814,
    91959,
    91664,
    87384,
    87329,
    86576,
    84697,
    82642,
    82350,
    82207,
    81725,
]

num_bonds = 20
assert(len(bond_prices) == num_bonds)
F = 100_000
c = 0.04
n = 1  # annual
T = [k for k in range(1, num_bonds + 1)]

spot_rates, forward_rates = bond.recursive_zero_coupon_yield_continuous(
    bond_prices, F, T, c, n
)

col_heads = ["Time Step", "Year", "Spot Rate", "Forward Rate"]
col_spaces = [10, 6, 11, 14]
col_decimals = [None, None, 5, 5]

table_data = []

for i in range(len(T)):
    table_data.append([i+1, T[i], spot_rates[i], forward_rates[i]])

table_str = table.generate_table(col_heads, col_spaces, table_data,
 ↪col_decimals)

dsp.printmd(table_str)
```

Time Step	Year	Spot Rate	Forward Rate
1	1	0.04512	0.04512
2	2	0.05313	0.06115

Time Step	Year	Spot Rate	Forward Rate
3	3	0.05735	0.06577
4	4	0.05336	0.04138
5	5	0.05078	0.04048
6	6	0.04938	0.04238
7	7	0.04938	0.04940
8	8	0.04816	0.03960
9	9	0.04815	0.04805
10	10	0.04766	0.04327
11	11	0.04815	0.05309
12	12	0.04783	0.04424
13	13	0.05324	0.11824
14	14	0.05232	0.04032
15	15	0.05250	0.05501
16	16	0.05431	0.08140
17	17	0.05637	0.08946
18	18	0.05585	0.04686
19	19	0.05518	0.04314
20	20	0.05507	0.05307

Part C [3 Marks]

Suppose you enter into a 20-year vanilla fixed-for-floating swap on a notional principal of $\$1,000,000$ where you pay the fixed rate of 6.5% and the counter-party pays the yield curve plus 1%.

Code in Matlab/Python a program to compute the swap value. Submit a table of results, similar to the table on L5.15.

[6]:
```
notional = 1_000_000    # $
fixed_rate = 0.065      # %
floating_spread = 0.01 # %

# We have spot and forward rates from part b

swap_values, _, swap_table_str = swap.compute_swap_values(
    notional, T, n, spot_rates, forward_rates, fixed_rate, floating_spread)

dsp.printmd(swap_table_str)

sum_swap = sum(swap_values)
print("Sum Swap:", sum_swap)
```

n	$y_{0,n}$	$y_{n-1,n}$	Fixed Payment	Floating Payment	Fixed - Floating	PV @ Spot
1	0.0451	0.0451	65000	55118.068	9881.932	9445.986
2	0.0531	0.0611	65000	71146.044	-6146.044	-5526.444
3	0.0573	0.0658	65000	75771.530	-10771.530	-9069.081

n	$y_{0,n}$	$y_{n-1,n}$	Fixed Payment	Floating Payment	Fixed - Floating	PV @ Spot
4	0.0534	0.0414	65000	51384.704	13615.296	10998.661
5	0.0508	0.0405	65000	50484.174	14515.826	11260.882
6	0.0494	0.0424	65000	52383.880	12616.120	9381.006
7	0.0494	0.0494	65000	59399.118	5600.882	3963.932
8	0.0482	0.0396	65000	49602.285	15397.715	10474.349
9	0.0481	0.0481	65000	58050.421	6949.579	4505.689
10	0.0477	0.0433	65000	53269.991	11730.009	7282.981
11	0.0482	0.0531	65000	63087.086	1912.914	1126.292
12	0.0478	0.0442	65000	54243.640	10756.360	6059.069
13	0.0532	0.1182	65000	128243.007	-63243.007	-31651.977
14	0.0523	0.0403	65000	50320.258	14679.742	7056.606
15	0.0525	0.0550	65000	65009.234	-9.234	-4.201
16	0.0543	0.0814	65000	91396.748	-26396.748	-11071.017
17	0.0564	0.0895	65000	99459.396	-34459.396	-13215.787
18	0.0558	0.0469	65000	56863.576	8136.424	2977.600
19	0.0552	0.0431	65000	53135.906	11864.094	4158.471
20	0.0551	0.0531	65000	63067.572	1932.428	642.326

```
Sum Swap: 18795.343538847617
```

Part E [1 Mark]

Test with different fixed rates and provide a better approximation of the swap rate so that the swap value is near zero (you do not need to develop a new code).

[7]:
```python
# I know the question says you don't need to develop new code here but I wrote a small script to find the optimal swap rate :)!

lb = 0.06
ub = 0.07
itv = 0.001

eps = 10
closest_sum_swap = None
closest_fixed_rate = None

while closest_sum_swap is None or closest_sum_swap > eps:
    fixed_rates = [lb + itv * k for k in range(int((ub - lb) / itv) + 1)]

    sum_swap_rates = []
    for fixed_rate in fixed_rates:
        swap_values, _, swap_table_str = swap.compute_swap_values(
```

```
            notional, T, n, spot_rates, forward_rates, fixed_rate,␣
↪floating_spread)
        sum_swap_values = sum(swap_values)

        sum_swap_rates.append(sum_swap_values)

    closest_index = -1
    for i, sum_swap in enumerate(sum_swap_rates):

        if closest_sum_swap is None or abs(sum_swap) < closest_sum_swap:
            closest_sum_swap = abs(sum_swap)
            closest_fixed_rate = fixed_rates[i]
            closest_index = i

    if closest_index >= 0:
        # if we didn't find a better rate, we need to make the interval more␣
↪granular
        lb = fixed_rates[closest_index - 1]
        ub = fixed_rates[closest_index + 1]
    itv /= 2
    print(
        f"fixed rate: {closest_fixed_rate}, sum of swap values:␣
↪{closest_sum_swap}")

print(
    f"\n{Fore.LIGHTGREEN_EX}FOUND: closest rate: {closest_fixed_rate:.6f}, sum␣
↪of swap values: {closest_sum_swap:.4f} ~ {0}{Style.RESET_ALL}")

# We can also see the table as in L5.15 for this new rate below
print(f"\n\n{Fore.LIGHTCYAN_EX}Table showing values at each cashflow for the␣
↪optimal swap rate: {closest_fixed_rate}{Style.RESET_ALL}")
dsp.printmd(swap_table_str)
```

```
fixed rate: 0.063, sum of swap values: 5447.49588805529
fixed rate: 0.0635, sum of swap values: 613.213968670438
fixed rate: 0.0635, sum of swap values: 613.213968670438
fixed rate: 0.0635, sum of swap values: 613.213968670438
fixed rate: 0.0634375, sum of swap values: 144.374763420368
fixed rate: 0.0634375, sum of swap values: 144.374763420368
fixed rate: 0.063453125, sum of swap values: 45.02241960239809
fixed rate: 0.063453125, sum of swap values: 45.02241960239809
fixed rate: 0.06344921875, sum of swap values: 2.326876153360942

FOUND: closest rate: 0.063449, sum of swap values: 2.3269 ~ 0
```

Table showing values at each cashflow for the optimal swap rate:

0.06344921875

n	$y_{0,n}$	$y_{n-1,n}$	Fixed Payment	Floating Payment	Fixed - Floating	PV @ Spot
1	0.0451	0.0451	63469	55118.068	8350.682	7982.288
2	0.0531	0.0611	63469	71146.044	-7677.294	-6903.324
3	0.0573	0.0658	63469	75771.530	-12302.780	-10358.315
4	0.0534	0.0414	63469	51384.704	12084.046	9761.692
5	0.0508	0.0405	63469	50484.174	12984.576	10072.991
6	0.0494	0.0424	63469	52383.880	11084.870	8242.409
7	0.0494	0.0494	63469	59399.118	4069.632	2880.215
8	0.0482	0.0396	63469	49602.285	13866.465	9432.711
9	0.0481	0.0481	63469	58050.421	5418.329	3512.918
10	0.0477	0.0433	63469	53269.991	10198.759	6332.251
11	0.0482	0.0531	63469	63087.086	381.664	224.717
12	0.0478	0.0442	63469	54243.640	9225.110	5196.515
13	0.0532	0.1182	63469	128243.007	-64774.257	-32418.340
14	0.0523	0.0403	63469	50320.258	13148.492	6320.529
15	0.0525	0.0550	63469	65009.234	-1540.484	-700.881
16	0.0543	0.0814	63469	91396.748	-27927.998	-11713.236
17	0.0564	0.0895	63469	99459.396	-35990.646	-13803.049
18	0.0558	0.0469	63469	56863.576	6605.174	2417.225
19	0.0552	0.0431	63469	53135.906	10332.844	3621.754
20	0.0551	0.0531	63469	63067.572	401.178	133.349

Question 3 [6 Marks]

Assume annual time periods, $T = 3$, a binomial model of the yield curve, and $y_{0,1} = 2\%$. Suppose over the whole forward rate lattice that the next period's forward rates can either go up by a factor of $u = 1.3$ with a probability of $p = 60\%$ or down by a factor of $d = 0.9$. (For example $y(u) = y_{0,1} \times u = 0.02 \times 1.3 = 0.026$ or 2.6%, $y(uu) = y_{0,1} \times u \times u$ and so on.) Use discrete compounding.

Part A [4 Marks]

Construct the forward rate lattice and the zero coupon bond yield curve $y_{0,2}$ and $y_{0,3}$.

[8]:
```
# Maturity
T = 3

# zero spot yield rate for time step 0 to time step 1
```

```python
y_0_1 = 0.02

# Probability of increase / decrease
p = 0.6
q = 1 - p

# Increase / Decrease factors
u = 1.3
d = 0.9

head_node = BinNode(y_0_1, 0, None, None, None)
forward_lattice = BinLattice(head_node)

forward_lattice.construct_bin_lattice(u, d, T)

print(f"{Fore.LIGHTCYAN_EX}Forward Lattice{Style.RESET_ALL}")
print(forward_lattice)

print(f"{Fore.LIGHTMAGENTA_EX}Yield curve lattice{Style.RESET_ALL}")
# y_{0, 2}
T_y = 2
p_lattice = forward_lattice.construct_p_lattice(T_y, p, q)

print(f"{Fore.GREEN}y_{0, 2} P lattice{Style.RESET_ALL}")
print(p_lattice)
y_0_2 = bond.spot_rate_from_p_lattice(p_lattice)
print(f"{Fore.LIGHTCYAN_EX}Spot rate: {y_0_2:.4f}{Style.RESET_ALL}")

print("-"*70)

# y_{0, 3}
T_y = 3
p_lattice = forward_lattice.construct_p_lattice(
    T_y, p, q)

print(f"{Fore.GREEN}y_{0, 3} P lattice{Style.RESET_ALL}")
print(p_lattice)
y_0_3 = bond.spot_rate_from_p_lattice(p_lattice)
print(f"{Fore.LIGHTCYAN_EX}Spot rate: {y_0_3:.4f}{Style.RESET_ALL}")
```

```
Forward Lattice
                   | 0.02000 |
              | 0.01800 | 0.02600 |
         | 0.01620 | 0.02340 | 0.03380 |
 | 0.01458 | 0.02106 | 0.03042 | 0.03042 | 0.04394 |

Yield curve lattice
y_(0, 2) P lattice
```

```
         | 0.95855 |
 | 0.98232 | 0.97466 |

Spot rate: 0.0214
----------------------------------------------------------------------
y_(0, 3) P lattice
              | 0.93432 |
         | 0.96258 | 0.94662 |
 | 0.98406 | 0.97714 | 0.96731 |

Spot rate: 0.0229
```

Part B [2 Marks]

Construct the 1-period forward rates $y_{1,2}$ and $y_{2,3}$, which are embedded in this zero coupon bond yield curve (we already have $y_{0,1}$).

[9]:
```python
# y_{1, 2}

y_1_2 = interest.zero_coupon_forward_rate_discrete(y_0_1, y_0_2, 1, 2)
print(f"y_{{1, 2}}: {y_1_2:.4f}")

# y_{2, 3}

y_2_3 = interest.zero_coupon_forward_rate_discrete(y_0_2, y_0_3, 2, 3)
print(f"y_{{2, 3}}: {y_2_3:.4f}")
```

```
y_{1, 2}: 0.0228
y_{2, 3}: 0.0259
```

Question 4 [3 Marks]

Consider the payoff at maturity T in Figure 1. Show how to construct this payoff using European calls with the same maturity only (you can use any combination of European calls with any strike price). You must state long/short, strike prices as well as the number of units. In addition, express the current value of the (replicating) portfolio in terms of the current prices of strike-K European calls $C_0(K), K > 0$.

To construct a perfectly replicating portfolio of EU call options, each call option will have 2 parameters: it's strike price, K_i, and the number of call options x_i. If x_i is negative, this is a short position, and hence a positive value is a long position.

Observe we know

$$K_1 = 8 \tag{29}$$

$$x_1 = 4 \tag{30}$$

since the first call option strikes at $S_t = 8$, and the slope of the first option can be derived as

$$\Delta = \frac{8-0}{10-8} \tag{31}$$
$$= \frac{8}{2} \tag{32}$$
$$= 4 \tag{33}$$

We also have the system of equations

$$\sum_{i=1}^{N} x_i = 0 \tag{34}$$
$$\sum_{i=1}^{N-1} x_i = 1 \tag{35}$$
$$\tag{36}$$

since the gradient of $S_t \in [8, 10]$ is $\Delta = 1$. And for K, we have

$$\sum_{i=1}^{N-1} x_i \left(K_N - K_i\right) = 10 \tag{37}$$
$$\sum_{i=1}^{N-2} x_i \left(K_{N-1} - K_i\right) = 8 \tag{38}$$

Easily we can observe $N = 3$ since there are only 3 call options needed to achieve the three gradients in the payoff diagram (with exception of $S_t < 8$ which is achieved by the K_1 strike price. Now solve

$$x_1 + x_2 = 1 \tag{39}$$
$$4 + x_2 = 1 \tag{40}$$
$$x_2 = -3 \tag{41}$$

$$x_1 + x_2 + x_3 = 0 \tag{42}$$
$$4 - 3 + x_3 = 0 \tag{43}$$
$$x_3 = -1 \tag{44}$$

$$x_1 \left(K_2 - K_1\right) = 8 \tag{45}$$
$$4 \left(K_2 - K_1\right) = 8 \tag{46}$$
$$K_2 = 2 + K_1 \tag{47}$$
$$K_2 = 2 + 8 \tag{48}$$
$$K_2 = 10 \tag{49}$$

$$
\begin{aligned}
x_1\left(K_3-K_1\right)+x_2\left(K_3-K_2\right) &= 10 && (50)\\
4\left(K_3-K_1\right)-3\left(K_3-2-K_1\right) &= 10 && (51)\\
4K_3-4K_1-3K_3+6+3K_1 &= 10 && (52)\\
K_3-K_1 &= 4 && (53)\\
K_3-8 &= 4 && (54)\\
K_3 &= 12 && (55)
\end{aligned}
$$

Therefore, we have

$$
x=\begin{pmatrix}4\\-3\\-1\end{pmatrix}, \qquad\qquad K=\begin{pmatrix}8\\10\\12\end{pmatrix}
$$

leading to a portfolio of call options, Θ defined as

$$
\Theta=\begin{cases}4 & \text{long call}, K_1=8\\ 3 & \text{short call}, K_2=10\\ 1 & \text{short call}, K_3=12\end{cases}
$$

This portfolio perfectly replicates the payoff diagram.

To compute the current price of this portfolio, we can simply take the weighted sum of the price of the assets in the portfolio at time 0. That is,

$$
\begin{aligned}
P_0^{\Theta} &= \sum_{i=1}^{N=3} x_i\cdot C_0(K_i) && (56)\\
&= 4\cdot C_0(K_1)-3\cdot C_0(K_2)-C_0(K_3) && (57)
\end{aligned}
$$

Question 5 [5 Marks]

Given a stock whose time-t price is S_t, consider a derivative that pays e^{S_T} at maturity T (the writer pays e^{S_T} to the holder; the holder pays nothing to the writer). We assume that there is also a (risk-free) zero-coupon bond with maturity T and face value 1, whose time-0 price is Z_0. Let C_0 be the arbitrage-free time-0 price of the derivative. Answer the following.

Part A [2 Marks]

Suppose S_T can take any positive value with strictly positive probability (under the physical probability measure P), and hence $P(S_T>M)>0$ for any $M>0$. Show that the considered derivative cannot be super-replicated if only the stock and bond are available in the market.

Suppose Θ^a super-replicates Θ^b where

$$\Theta^a = \begin{cases} 1 & \text{risk free zero-coupon bond} \\ 1 & \text{stock} \end{cases}, \qquad \Theta^b = \begin{cases} 1 & \text{derivative} \end{cases}$$

Equivalently,

$$V_t^a \geq V_t^b, \quad \forall t \in \{0, T\}$$

Rearranging this, we get

$$V_t^a - V_t^b \geq 0, \quad \forall t \in \{0, T\}$$

Now construct Θ^c such that

$$\Theta^c = \Theta^a - \Theta^b$$

Then

$$\begin{aligned} V_0^c &= V_0^a - V_0^b && (58)\\ &= [Z_0 + S_0] - [C_0] && (59)\\ &= Z_0 + S_0 - C_0 && (60) \end{aligned}$$

and

$$\begin{aligned} V_T^c &= V_T^a - V_T^b && (61)\\ &= [1 + S_T] - [e^{S_T}] && (62)\\ &= 1 + S_T - e^{S_T} && (63) \end{aligned}$$

For Θ^a to super-replicate Θ^b, we need the value of Θ^c to remain non-negative for $t \in \{0, T\}$. Consider $t = T$ and when $V_T^c = 0$ to get a lower bound.

$$\begin{aligned} 1 + S_T &= e^{S_T} && (64)\\ \ln(1 + S_T) &= S_T && (65)\\ S_T - \frac{S_T^2}{2} + \frac{S_T^3}{3} - \frac{S_T^4}{4} + \cdots &\approx S_T && (66)\\ \Rightarrow S_T &= 0 && (67) \end{aligned}$$

Since $e^{S_T} > 1 + S_T$, this shows that $\forall T \geq 0$, $(1 + S_T) \leq e^{S_T}$, and more importantly, $\forall T > 0$, $(1 + S_T) < e^{S_T}$. We know $T > 0$ as maturity must always be in the future.

Hence we have reached a contradiction. We have shown the value of Θ^c at time $t = T$ is strictly negative, meaning Θ^a does not super-replicate Θ^b. In fact, it can only sub-replicate (or potentially replicate) it.

Part B [3 Marks]

Show that

$$C_0 \geq e^{\frac{S_0}{Z_0}} Z_0$$

Consider deriving a portfolio, say Θ^d which gives the tightest lower bound to the payoff of the derivative. Trivially, this is the tangent line of the payoff of the derivative. We can derive this as follows

We know the payoff of the derivative is

$$P_T^{\text{derivative}} = e^{S_T}$$

and that its derivative (no pun intended) is $\frac{\mathrm{d}}{\mathrm{d}x} P_T^{\text{derivative}} = e^{S_T}$. Consider an arbitrary point $k \in [0, \infty)$ along this payoff curve. Now derive its tangent.

$$P = mS_T + c \tag{68}$$

$$e^k = e^k \cdot k + c \tag{69}$$

$$c = e^k (1 - k) \tag{70}$$

Therefore, the tangent line is given as

$$P_T = e^k S_T + e^k (1 - k) \tag{71}$$

where P_T is the payoff at time T. Observe this is equivlant to constructing Θ^d as follows

$$\Theta^d = \begin{cases} e^k & \text{stocks} \\ e^k (1 - k) & \text{bonds} \end{cases}$$

Such a portfolio has time-0 value of

$$V_0^d = e^k \cdot S_0 + e^k (1 - k) Z_0$$

meaning we now have the inequality

$$C_0 \geq e^k \cdot S_0 + e^k (1 - k) Z_0$$

Now, we want to find the tightest lower bound. We can do this by maximising k through calculus.

$$
\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}k} C_0 &= e^k \cdot S_0 + \left[e^k (1-k) + e^k (-1)\right] Z_0 && (72)\\
0 &= e^k S_0 + e^k [1-k-1] Z_0 && (73)\\
&= e^k S_0 - k e^k Z_0 && (74)\\
&= S_0 - k Z_0 && (75)\\
k Z_0 &= S_0 && (76)\\
k &= \frac{S_0}{Z_0} && (77)
\end{aligned}
$$

Now we have our optimal k, substitute back into our inequality from above.

$$
\begin{aligned}
C_0 &\geq e^k \cdot S_0 + e^k (1-k) Z_0 && (78)\\
&= e^{\frac{S_0}{Z_0}} \cdot S_0 + e^{\frac{S_0}{Z_0}} \left(1 - \frac{S_0}{Z_0}\right) Z_0 && (79)\\
&= e^{\frac{S_0}{Z_0}} \left[S_0 + \left(1 - \frac{S_0}{Z_0}\right) Z_0\right] && (80)\\
&= e^{\frac{S_0}{Z_0}} [S_0 + Z_0 - S_0] && (81)\\
&= e^{\frac{S_0}{Z_0}} \cdot Z_0 && (82)\\
\Rightarrow C_0 &\geq e^{\frac{S_0}{Z_0}} Z_0 && (83)
\end{aligned}
$$

As given in the question :).