MF728_hw1-Copy2

February 9, 2024

0.1 MF 728: Fixed Income

0.1.1 Problem set # 1

1. Yield Curve Construction: Consider the following table of USD swap rates:

Term	Rate (%)
1Y	2.8438
2Y	3.060
3Y	3.126
4Y	3.144
5Y	3.150
7Y	3.169
10Y	3.210
30Y	3.237

Note: You may assume that these swaps pay coupons semi-annually (every 6 months). For simplicity, you may use a year fraction of 0.5 in all swap coupon payments.

(a) Extract the constant forward rate for the first year that enables you to match the 1Y market swap rate.

[1]:
```
import numpy as np
from scipy.optimize import fsolve

# Convert the original inner function into a standalone function
def swap_present_value(f1, sw1):
    # Calculate the present value (PV)
    PV = 0.5 * sw1 / (1 + 0.5 * f1) + 0.5 * sw1 / (1 + 0.5 * f1)**2
    # Return the difference between PV and the calculated forward rates
    return PV - 0.5 * f1 / (1 + 0.5 * f1) - 0.5 * f1 / (1 + 0.5 * f1)**2

# Modified swap_to_forward_1Y function, uses the args parameter to pass the␣
 ↪additional sw1 parameter
def swap_to_forward_1Y(sw1):
    # fsolve now calls the swap_present_value function, passing sw1 as a␣
 ↪parameter
    forward_rate = fsolve(swap_present_value, x0=sw1, args=(sw1,))
    return forward_rate[0]
```

[2]:
```
f1 = swap_to_forward_1Y(2.8438 / 100)
print(f1)
```

```
0.028437999999999998
```

(b) Holding this first year forward rate fixed, find the forward rate from one year to two years that enables you to match the two year swap (while also matching the one year).

[3]:
```
# Standalone function to calculate the present value difference for a two-year swap
def swap_present_value(f2, sw2, f1):
    # Calculate the present value of swap payments
    PV1 = sw2 / (1 + 0.5 * f1) + sw2 / (1 + 0.5 * f1)**2 + sw2 / ((1 + 0.5 * f1)**2 * (1 + 0.5 * f2)) + sw2 / ((1 + 0.5 * f1)**2 * (1 + 0.5 * f2)**2)
    # Calculate the present value of forward rate payments
    PV2 = f1 / (1 + 0.5 * f1) + f1 / (1 + 0.5 * f1)**2 + f2 / ((1 + 0.5 * f1)**2 * (1 + 0.5 * f2)) + f2 / ((1 + 0.5 * f1)**2 * (1 + 0.5 * f2)**2)
    # Return the difference in present values
    return PV1 - PV2

# Function to solve for the second year's forward rate
def swap_to_forward_2Y(sw2, f1):
    # Use fsolve to solve for the second year's forward rate
    forward_rate_2Y = fsolve(swap_present_value, x0=sw2, args=(sw2, f1))
    return forward_rate_2Y[0]

# Use the forward rate f1 calculated in the previous step
f1 = 2.8438 / 100  # The forward rate for the first year given in the example
# Calculate the forward rate for the second year
f2 = swap_to_forward_2Y(3.060 / 100, f1)
# Calculate the discount factor for the second year
D1 = 1 / ((1 + 0.5 * f1)**2)  # Calculating D1 from the previous step
D2 = D1 / ((1 + 0.5 * f2)**2)

f2, D2
```

[3]:
```
(0.03283113038626899, 0.9410092691367571)
```

(c) Continue this process and extract piecewise constant forward rates for the entire curve. Comment on the forward rates vs. the swap rates.

[4]:
```
# Standalone function to calculate the present value difference for a three-year swap
def swap_present_value_3Y(f3, sw3, f1, f2):
    # Calculate the present value of swap payments
```

```
    PV1 = sw3 / (1 + 0.5 * f1) + sw3 / (1 + 0.5 * f1)**2 + sw3 / ((1 + 0.5 *␣
 ↪f1)**2 * (1 + 0.5 * f2)) + sw3 / ((1 + 0.5 * f1)**2 * (1 + 0.5 * f2)**2) +␣
 ↪sw3 / ((1 + 0.5 * f1)**2 * (1 + 0.5 * f2)**2 * (1 + 0.5 * f3)) + sw3 / ((1 +␣
 ↪0.5 * f1)**2 * (1 + 0.5 * f2)**2 * (1 + 0.5 * f3)**2)
    # Calculate the present value of forward rate payments
    PV2 = f1 / (1 + 0.5 * f1) + f1 / (1 + 0.5 * f1)**2 + f2 / ((1 + 0.5 *␣
 ↪f1)**2 * (1 + 0.5 * f2)) + f2 / ((1 + 0.5 * f1)**2 * (1 + 0.5 * f2)**2) + f3␣
 ↪/ ((1 + 0.5 * f1)**2 * (1 + 0.5 * f2)**2 * (1 + 0.5 * f3)) + f3 / ((1 + 0.5␣
 ↪* f1)**2 * (1 + 0.5 * f2)**2 * (1 + 0.5 * f3)**2)
    # Return the difference in present values
    return PV1 - PV2

# Function to solve for the third year's forward rate
def swap_to_forward_3Y(sw3, f1, f2):
    # Use fsolve to solve for the third year's forward rate
    forward_rate_3Y = fsolve(swap_present_value_3Y, x0=sw3, args=(sw3, f1, f2))
    return forward_rate_3Y[0]

# Use the forward rates f1 and f2 calculated in the previous years
sw3 = 3.126 / 100

# Calculate the forward rate for the third year
f3 = swap_to_forward_3Y(sw3, f1, f2)

# Calculate the discount factor for the third year
D3 = D2 / ((1 + 0.5 * f3)**2)

f3, D3
```

[4]:
```
(0.03264530551203542, 0.9110258289278597)
```

[5]:
```
# Standalone swap_present_value function
def swap_present_value_4Y(f4, sw4, f1, f2, f3):
    # Calculate the present value of swap payments
    PV = sw4/(1+0.5*f1) + sw4/(1+0.5*f1)**2 + sw4/((1+0.5*f1)**2*(1+0.5*f2)) +␣
 ↪sw4/((1+0.5*f1)**2*(1+0.5*f2)**2) + sw4/((1+0.5*f1)**2*(1+0.5*f2)**2*(1+0.
 ↪5*f3)) + sw4/((1+0.5*f1)**2*(1+0.5*f2)**2*(1+0.5*f3)**2) + sw4/((1+0.
 ↪5*f1)**2*(1+0.5*f2)**2*(1+0.5*f3)**2*(1+0.5*f4)) + sw4/((1+0.5*f1)**2*(1+0.
 ↪5*f2)**2*(1+0.5*f3)**2*(1+0.5*f4)**2)
    # Calculate the present value of forward rate payments
    PV2 = f1/(1+0.5*f1) + f1/(1+0.5*f1)**2 + f2/((1+0.5*f1)**2*(1+0.5*f2)) + f2/
 ↪((1+0.5*f1)**2*(1+0.5*f2)**2) + f3/((1+0.5*f1)**2*(1+0.5*f2)**2*(1+0.5*f3))␣
 ↪+ f3/((1+0.5*f1)**2*(1+0.5*f2)**2*(1+0.5*f3)**2) + f4/((1+0.5*f1)**2*(1+0.
 ↪5*f2)**2*(1+0.5*f3)**2*(1+0.5*f4)) + f4/((1+0.5*f1)**2*(1+0.5*f2)**2*(1+0.
 ↪5*f3)**2*(1+0.5*f4)**2)
    # Return the difference in present values
    return PV - PV2
```

```
# Updated swap_to_forward_4Y function
def swap_to_forward_4Y(sw4, f1, f2, f3):
    # Use a lambda function to pass additional arguments to swap_present_value
    forward_rate_4Y = fsolve(lambda f4: swap_present_value_4Y(f4, sw4, f1, f2,␣
 ↪f3), sw4)
    return forward_rate_4Y[0]

sw4 = 3.144 / 100
f4 = swap_to_forward_4Y(sw4, f1, f2, f3)

# Calculate the discount factor for the fourth year
D4 = D3 / ((1 + 0.5 * f4)**2)

f4,D4
```

[5]: (0.03201590223021914, 0.8825442234289774)

[7]:
```
# Standalone swap_present_value function
def swap_present_value_5Y(f5, sw5, f1, f2, f3, f4):
    PV = sw5*sum((1+0.5*f1)**-i for i in range(1, 3)) + sw5/((1+0.5*f1)**2*(1+0.
 ↪5*f2)) + sw5/((1+0.5*f1)**2*(1+0.5*f2)**2) + sw5/((1+0.5*f1)**2*(1+0.
 ↪5*f2)**2*(1+0.5*f3)) + sw5/((1+0.5*f1)**2*(1+0.5*f2)**2*(1+0.5*f3)**2) + sw5/
 ↪((1+0.5*f1)**2*(1+0.5*f2)**2*(1+0.5*f3)**2*(1+0.5*f4)) + sw5/((1+0.
 ↪5*f1)**2*(1+0.5*f2)**2*(1+0.5*f3)**2*(1+0.5*f4)**2) + sw5/((1+0.
 ↪5*f1)**2*(1+0.5*f2)**2*(1+0.5*f3)**2*(1+0.5*f4)**2*(1+0.5*f5)) + sw5/((1+0.
 ↪5*f1)**2*(1+0.5*f2)**2*(1+0.5*f3)**2*(1+0.5*f4)**2*(1+0.5*f5)**2)
    PV2 = f1/(1+0.5*f1) + f1/(1+0.5*f1)**2 + f2/((1+0.5*f1)**2*(1+0.5*f2)) + f2/
 ↪((1+0.5*f1)**2*(1+0.5*f2)**2) + f3/((1+0.5*f1)**2*(1+0.5*f2)**2*(1+0.5*f3))␣
 ↪+ f3/((1+0.5*f1)**2*(1+0.5*f2)**2*(1+0.5*f3)**2) + f4/((1+0.5*f1)**2*(1+0.
 ↪5*f2)**2*(1+0.5*f3)**2*(1+0.5*f4)) + f4/((1+0.5*f1)**2*(1+0.5*f2)**2*(1+0.
 ↪5*f3)**2*(1+0.5*f4)**2) + f5/((1+0.5*f1)**2*(1+0.5*f2)**2*(1+0.5*f3)**2*(1+0.
 ↪5*f4)**2*(1+0.5*f5)) + f5/((1+0.5*f1)**2*(1+0.5*f2)**2*(1+0.5*f3)**2*(1+0.
 ↪5*f4)**2*(1+0.5*f5)**2)
    return PV - PV2

# Standalone swap_present_value function
def swap_to_forward_5Y(sw5, f1, f2, f3, f4):
    # Use a lambda function to pass additional arguments to swap_present_value
    forward_rate_5Y = fsolve(lambda f5: swap_present_value_5Y(f5, sw5, f1, f2,␣
 ↪f3, f4), sw5)
    return forward_rate_5Y[0]

f5 = swap_to_forward_5Y(0.0315, f1, f2, f3, f4)

# Calculate the discount factor for the fifth year
D5 = D4 / (1 + 0.5 * f5)**2
```

```
f5,D5
```

[7]: (0.031760049949048276, 0.8551683806917486)

[9]:
```
# Standalone swap_present_value function
def swap_present_value_7Y(f7, sw7, f1, f2, f3, f4, f5, D1, D2, D3, D4, D5):
    PV = (sw7 * sum((1 + 0.5 * f1) ** -i for i in range(1, 3)) +
          sw7 * D1 * sum((1 + 0.5 * f2) ** -i for i in range(1, 3)) +
          sw7 * D2 * sum((1 + 0.5 * f3) ** -i for i in range(1, 3)) +
          sw7 * D3 * sum((1 + 0.5 * f4) ** -i for i in range(1, 3)) +
          sw7 * D4 * sum((1 + 0.5 * f5) ** -i for i in range(1, 3)) +
          sw7 * D5 * sum((1 + 0.5 * f7) ** -i for i in range(1, 5)))
    PV2 = (f1 * sum((1 + 0.5 * f1) ** -i for i in range(1, 3)) +
           f2 * D1 * sum((1 + 0.5 * f2) ** -i for i in range(1, 3)) +
           f3 * D2 * sum((1 + 0.5 * f3) ** -i for i in range(1, 3)) +
           f4 * D3 * sum((1 + 0.5 * f4) ** -i for i in range(1, 3)) +
           f5 * D4 * sum((1 + 0.5 * f5) ** -i for i in range(1, 3)) +
           f7 * D5 * sum((1 + 0.5 * f7) ** -i for i in range(1, 5)))
    return PV - PV2

# Updated swap_to_forward_7Y function
def swap_to_forward_7Y(sw7, f1, f2, f3, f4, f5, D1, D2, D3, D4, D5):
    # Use a lambda function to pass additional arguments to swap_present_value
    forward_rate_7Y = fsolve(lambda f7: swap_present_value_7Y(f7, sw7, f1, f2,␣
 ↪f3, f4, f5, D1, D2, D3, D4, D5), sw7)
    return forward_rate_7Y[0]

f7 = swap_to_forward_7Y(0.03169, f1, f2, f3, f4, f5, D1, D2, D3, D4, D5)

# Calculate D7
D7 = D5 / (1 + f7 * 0.5) ** 4

f7,D7
```

[9]: (0.03222150590023792, 0.802208843979025)

[10]:
```
def swap_present_value_10Y(f10, sw10, f1, f2, f3, f4, f5, f7, D1, D2, D3, D4,␣
 ↪D5, D7):
    PV = (sw10 * sum((1 + 0.5 * f1) ** -i for i in range(1, 3)) +
          sw10 * D1 * sum((1 + 0.5 * f2) ** -i for i in range(1, 3)) +
          sw10 * D2 * sum((1 + 0.5 * f3) ** -i for i in range(1, 3)) +
          sw10 * D3 * sum((1 + 0.5 * f4) ** -i for i in range(1, 3)) +
          sw10 * D4 * sum((1 + 0.5 * f5) ** -i for i in range(1, 3)) +
          sw10 * D5 * sum((1 + 0.5 * f7) ** -i for i in range(1, 5)) +
          sw10 * D7 * sum((1 + 0.5 * f10) ** -i for i in range(1, 7)))
    PV2 = (f1 * sum((1 + 0.5 * f1) ** -i for i in range(1, 3)) +
```

```
            f2 * D1 * sum((1 + 0.5 * f2) ** -i for i in range(1, 3)) +
            f3 * D2 * sum((1 + 0.5 * f3) ** -i for i in range(1, 3)) +
            f4 * D3 * sum((1 + 0.5 * f4) ** -i for i in range(1, 3)) +
            f5 * D4 * sum((1 + 0.5 * f5) ** -i for i in range(1, 3)) +
            f7 * D5 * sum((1 + 0.5 * f7) ** -i for i in range(1, 5)) +
            f10 * D7 * sum((1 + 0.5 * f10) ** -i for i in range(1, 7)))
    return PV - PV2

def swap_to_forward_10Y(sw10, f1, f2, f3, f4, f5, f7, D1, D2, D3, D4, D5, D7):
    forward_rate_10Y = fsolve(lambda f10: swap_present_value_10Y(f10, sw10, f1,␣
 ↪f2, f3, f4, f5, f7, D1, D2, D3, D4, D5, D7), sw10)
    return forward_rate_10Y[0]

f10 = swap_to_forward_10Y(0.0321, f1, f2, f3, f4, f5, f7, D1, D2, D3, D4, D5,␣
 ↪D7)

D10 = D7 / (1 + f10 * 0.5) ** 6
f10,D10
```

[10]:
```
(0.033225985137036694, 0.7266972400950599)
```

[11]:
```
# Standalone swap_present_value function
def swap_present_value_30Y(f30, sw30, f1, f2, f3, f4, f5, f7, f10, D1, D2, D3,␣
 ↪D4, D5, D7, D10):
    PV = (sw30 * sum((1 + 0.5 * f1) ** -i for i in range(1, 3)) +
          sw30 * D1 * sum((1 + 0.5 * f2) ** -i for i in range(1, 3)) +
          sw30 * D2 * sum((1 + 0.5 * f3) ** -i for i in range(1, 3)) +
          sw30 * D3 * sum((1 + 0.5 * f4) ** -i for i in range(1, 3)) +
          sw30 * D4 * sum((1 + 0.5 * f5) ** -i for i in range(1, 3)) +
          sw30 * D5 * sum((1 + 0.5 * f7) ** -i for i in range(1, 5)) +
          sw30 * D7 * sum((1 + 0.5 * f10) ** -i for i in range(1, 7)) +
          sw30 * D10 * sum((1 + 0.5 * f30) ** -i for i in range(1, 41)))
    PV2 = (f1 * sum((1 + 0.5 * f1) ** -i for i in range(1, 3)) +
           f2 * D1 * sum((1 + 0.5 * f2) ** -i for i in range(1, 3)) +
           f3 * D2 * sum((1 + 0.5 * f3) ** -i for i in range(1, 3)) +
           f4 * D3 * sum((1 + 0.5 * f4) ** -i for i in range(1, 3)) +
           f5 * D4 * sum((1 + 0.5 * f5) ** -i for i in range(1, 3)) +
           f7 * D5 * sum((1 + 0.5 * f7) ** -i for i in range(1, 5)) +
           f10 * D7 * sum((1 + 0.5 * f10) ** -i for i in range(1, 7)) +
           f30 * D10 * sum((1 + 0.5 * f30) ** -i for i in range(1, 41)))
    return PV - PV2

# Refactored swap_to_forward_30Y function
def swap_to_forward_30Y(sw30, f1, f2, f3, f4, f5, f7, f10, D1, D2, D3, D4, D5,␣
 ↪D7, D10):
    forward_rate_30Y = fsolve(lambda f30: swap_present_value_30Y(f30, sw30, f1,␣
 ↪f2, f3, f4, f5, f7, f10, D1, D2, D3, D4, D5, D7, D10), sw30)
```

```
    return forward_rate_30Y[0]

# Using the refactored function to calculate f30
f30 = swap_to_forward_30Y(0.03237, f1, f2, f3, f4, f5, f7, f10, D1, D2, D3, D4,␣
 ↪D5, D7, D10)

# Calculating D30
D30 = D10 / (1 + f30 * 0.5) ** 20
f30,D30
```

[11]:
```
(0.03258650990142389, 0.5259841369171494)
```

[12]:
```
f_list=[f1,f2,f3,f4,f5,f7,f10,f30]
f_list
```

[12]:
```
[0.028437999999999998,
 0.03283113038626899,
 0.03264530551203542,
 0.03201590223021914,
 0.031760049949048276,
 0.03222150590023792,
 0.033225985137036694,
 0.03258650990142389]
```

(d) Compute the fair market, breakeven swap rate of a 15Y swap. That is, find the swap rate that equates the present values of the fixed and floating legs.

[13]:
```
def sw15(f1, f2, f3, f4, f5, f7, f10, f30, D1, D2, D3, D4, D5, D7, D10):
    # Calculate the numerator of PV
    numerator = (
        f1 * sum((1 + 0.5 * f1) ** -i for i in range(1, 3)) +
        f2 * D1 * sum((1 + 0.5 * f2) ** -i for i in range(1, 3)) +
        f3 * D2 * sum((1 + 0.5 * f3) ** -i for i in range(1, 3)) +
        f4 * D3 * sum((1 + 0.5 * f4) ** -i for i in range(1, 3)) +
        f5 * D4 * sum((1 + 0.5 * f5) ** -i for i in range(1, 3)) +
        f7 * D5 * sum((1 + 0.5 * f7) ** -i for i in range(1, 5)) +
        f10 * D7 * sum((1 + 0.5 * f10) ** -i for i in range(1, 7)) +
        f30 * D10 * sum((1 + 0.5 * f30) ** -i for i in range(1, 11))
    )

    # Calculate the denominator of PV
    denominator = (
        sum((1 + 0.5 * f1) ** -i for i in range(1, 3)) +
        D1 * sum((1 + 0.5 * f2) ** -i for i in range(1, 3)) +
        D2 * sum((1 + 0.5 * f3) ** -i for i in range(1, 3)) +
        D3 * sum((1 + 0.5 * f4) ** -i for i in range(1, 3)) +
        D4 * sum((1 + 0.5 * f5) ** -i for i in range(1, 3)) +
        D5 * sum((1 + 0.5 * f7) ** -i for i in range(1, 5)) +
```

```
        D7 * sum((1 + 0.5 * f10) ** -i for i in range(1, 7)) +
        D10 * sum((1 + 0.5 * f30) ** -i for i in range(1, 11))
    )

    # Calculate PV by dividing the numerator by the denominator
    PV = numerator / denominator

    return PV
sw15(f1, f2, f3, f4, f5, f7, f10, f30, D1, D2, D3, D4, D5, D7, D10)
```

[13]: 0.03223672517257003

(e) Compute discount factors. Compute zero rates by finding the constant rate that leads to the calibrated discount factors. Comment on the differences in the zero rates and swap rates.

[14]:
```
D_list=[D1,D2,D3,D4,D5,D7,D10,D30]
zero_list = [np.log(D)/-T for D,T in zip(D_list,[1,2,3,4,5,7,10,30])]
zero_list
```

[14]:
```
[0.028237716361644027,
 0.030401144570267404,
 0.031061343281080404,
 0.03123659497399354,
 0.03129137859064855,
 0.0314837572929969,
 0.031924533922686626,
 0.021416140821712493]
```

Differences between Zero Rates and Swap Rates:

- Level of Rates: The zero rates and swap rates follow a similar trend up to the 10Y maturity, where zero rates increase with maturity, reflecting a normal upward-sloping yield curve. However, for the 30Y maturity, the zero rate significantly drops compared to the swap rate, indicating a discrepancy in long-term interest rate expectations or market conditions.
- Market Expectations and Liquidity: Swap rates are influenced by the expectations of future interest rates, credit risk, and liquidity in the swap market. In contrast, zero rates are purely derived from the current market prices of zero-coupon bonds or derived from bond prices, reflecting the market's view on future interest rates without the swap market's credit and liquidity considerations.

(f) Shift all forward rates up 100 basis points and re-calculate the breakeven swap rates for each benchmark point. Generate a table of new swap rates. Are these rates equivalent to having shifted the swap rates directly?

[15]:
```
def for_up_swr(f_list, basis_list):

    for i in range(len(f_list)):
        f_list[i] += basis_list[i]
```

```
  f1, f2, f3, f4, f5, f7, f10, f30 = f_list

  sw1 = f1 * sum((1+0.5*f1)**-i for i in range(1,3)) / sum((1+0.5*f1)**-i for i in range(1,3))

  D1 = 1 / (1+0.5*f1)**2

  sw2 = f1 * sum((1+0.5*f1)**-i for i in range(1,3)) + f2 * D1 * sum((1+0.5*f2)**-i for i in range(1,3))
  sw2 /= sum((1+0.5*f1)**-i for i in range(1,3)) + D1 * sum((1+0.5*f2)**-i for i in range(1,3))

  D2 = D1 / (1+0.5*f2)**2

  sw3 = f1 * sum((1+0.5*f1)**-i for i in range(1,3)) + f2 * D1 * sum((1+0.5*f2)**-i for i in range(1,3)) + f3 * D2 * sum((1+0.5*f3)**-i for i in range(1,3))
  sw3 /= sum((1+0.5*f1)**-i for i in range(1,3)) + D1 * sum((1+0.5*f2)**-i for i in range(1,3)) + D2 * sum((1+0.5*f3)**-i for i in range(1,3))

  D3 = D2 / (1+0.5*f3)**2

  sw4 = f1 * sum((1+0.5*f1)**-i for i in range(1,3)) + f2 * D1 * sum((1+0.5*f2)**-i for i in range(1,3)) + f3 * D2 * sum((1+0.5*f3)**-i for i in range(1,3)) + f4 * D3 * sum((1+0.5*f4)**-i for i in range(1,3))
  sw4 /= sum((1+0.5*f1)**-i for i in range(1,3)) + D1 * sum((1+0.5*f2)**-i for i in range(1,3)) + D2 * sum((1+0.5*f3)**-i for i in range(1,3)) + D3 * sum((1+0.5*f4)**-i for i in range(1,3))

  D4 = D3 / (1+0.5*f4)**2

  sw5 = f1 * sum((1+0.5*f1)**-i for i in range(1,3)) + f2 * D1 * sum((1+0.5*f2)**-i for i in range(1,3)) + f3 * D2 * sum((1+0.5*f3)**-i for i in range(1,3)) + f4 * D3 * sum((1+0.5*f4)**-i for i in range(1,3)) + f5 * D4 * sum((1+0.5*f5)**-i for i in range(1,3))
  sw5 /= sum((1+0.5*f1)**-i for i in range(1,3)) + D1 * sum((1+0.5*f2)**-i for i in range(1,3)) + D2 * sum((1+0.5*f3)**-i for i in range(1,3)) + D3 * sum((1+0.5*f4)**-i for i in range(1,3)) + D4 * sum((1+0.5*f5)**-i for i in range(1,3))

  D5 = D4 / (1+0.5*f5)**2
```

```
    sw7 = f1 * sum((1+0.5*f1)**-i for i in range(1,3)) + f2 * D1 * sum((1+0.
↪5*f2)**-i for i in range(1,3)) + f3 * D2 * sum((1+0.5*f3)**-i for i in␣
↪range(1,3)) + f4 * D3 * sum((1+0.5*f4)**-i for i in range(1,3)) + f5 * D4 *␣
↪sum((1+0.5*f5)**-i for i in range(1,3)) + f7 * D5 * sum((1+0.5*f7)**-i for i␣
↪in range(1,5))
    sw7 /= sum((1+0.5*f1)**-i for i in range(1,3)) + D1 * sum((1+0.5*f2)**-i␣
↪for i in range(1,3)) + D2 * sum((1+0.5*f3)**-i for i in range(1,3)) + D3 *␣
↪sum((1+0.5*f4)**-i for i in range(1,3)) + D4 * sum((1+0.5*f5)**-i for i in␣
↪range(1,3)) + D5 * sum((1+0.5*f7)**-i for i in range(1,5))

    D7 = D5 / (1+0.5*f7)**4

    sw10 = f1 * sum((1+0.5*f1)**-i for i in range(1,3)) + f2 * D1 * sum((1+0.
↪5*f2)**-i for i in range(1,3)) + f3 * D2 * sum((1+0.5*f3)**-i for i in␣
↪range(1,3)) + f4 * D3 * sum((1+0.5*f4)**-i for i in range(1,3)) + f5 * D4 *␣
↪sum((1+0.5*f5)**-i for i in range(1,3)) + f7 * D5 * sum((1+0.5*f7)**-i for i␣
↪in range(1,5)) + f10 * D7 * sum((1+0.5*f10)**-i for i in range(1,7))
    sw10 /= sum((1+0.5*f1)**-i for i in range(1,3)) + D1 * sum((1+0.5*f2)**-i␣
↪for i in range(1,3)) + D2 * sum((1+0.5*f3)**-i for i in range(1,3)) + D3 *␣
↪sum((1+0.5*f4)**-i for i in range(1,3)) + D4 * sum((1+0.5*f5)**-i for i in␣
↪range(1,3)) + D5 * sum((1+0.5*f7)**-i for i in range(1,5)) + D7 * sum((1+0.
↪5*f10)**-i for i in range(1,7))

    D10 = D7 / (1+0.5*f10)**6

    sw30 = f1 * sum((1+0.5*f1)**-i for i in range(1,3)) + f2 * D1 * sum((1+0.
↪5*f2)**-i for i in range(1,3)) + f3 * D2 * sum((1+0.5*f3)**-i for i in␣
↪range(1,3)) + f4 * D3 * sum((1+0.5*f4)**-i for i in range(1,3)) + f5 * D4 *␣
↪sum((1+0.5*f5)**-i for i in range(1,3)) + f7 * D5 * sum((1+0.5*f7)**-i for i␣
↪in range(1,5)) + f10 * D7 * sum((1+0.5*f10)**-i for i in range(1,7)) + f30 *␣
↪D10 * sum((1+0.5*f30)**-i for i in range(1,41))
    sw30 /= sum((1+0.5*f1)**-i for i in range(1,3)) + D1 * sum((1+0.5*f2)**-i␣
↪for i in range(1,3)) + D2 * sum((1+0.5*f3)**-i for i in range(1,3)) + D3 *␣
↪sum((1+0.5*f4)**-i for i in range(1,3)) + D4 * sum((1+0.5*f5)**-i for i in␣
↪range(1,3)) + D5 * sum((1+0.5*f7)**-i for i in range(1,5)) + D7 * sum((1+0.
↪5*f10)**-i for i in range(1,7)) + D10 * sum((1+0.5*f30)**-i for i in␣
↪range(1,41))

    return sw1,sw2,sw3,sw4,sw5,sw7,sw10,sw30

basis_list = [0.01 for i in range(8)]
```

[16]:
```
import pandas as pd

original_swaps = [2.8438, 3.060, 3.126, 3.144, 3.150, 3.169, 3.210, 3.237]
original_swaps = [i/100 for i in original_swaps]
```

```
modified_swaps = for_up_swr(f_list, basis_list)

rates = {
    'Term': [1, 2, 3, 4, 5, 7, 10, 30],
    'Original': [round(r, 4) for r in original_swaps],
    'Modified': [round(r, 4) for r in modified_swaps]
}

df = pd.DataFrame(rates)
print("\nOriginal vs Modified Swap Rates:\n")
print(df)
```

```
Original vs Modified Swap Rates:

   Term  Original  Modified
0     1    0.0284    0.0384
1     2    0.0306    0.0406
2     3    0.0313    0.0412
3     4    0.0314    0.0414
4     5    0.0315    0.0415
5     7    0.0317    0.0417
6    10    0.0321    0.0421
7    30    0.0324    0.0423
```

The modified swap rates produced by for_up_swr() are equivalent to having directly shifted the swap rates by 0.01 basis points. The end result is the same in both cases.

Shifting the forward rates and recalculating is equivalent to shifting the swap rates directly.

(g) Consider a bearish steepener to the swap rates, that is perform the following shifts on each swap rate:

Term	Rate change (bps)
1Y	+0
2Y	+0
3Y	+0
4Y	+5
5Y	+10
7Y	+15
10Y	+25
30Y	+50

[17]:
```
swap_list=[0.028438,0.03060,0.03126,0.03144,0.03150,0.03169,0.03210,0.03237]

basis_list=[0,0,0,0.0005,0.001,0.0015,0.0025,0.005]
```

```
sw1,sw2,sw3,sw4,sw5,sw7,sw10,sw30=[sw+basis for sw,basis in␣
 ↪zip(swap_list,basis_list)]
sw1,sw2,sw3,sw4,sw5,sw7,sw10,sw30
```

[17]:
```
(0.028438,
 0.0306,
 0.03126,
 0.03194,
 0.0325,
 0.033190000000000004,
 0.0346,
 0.03737)
```

(h) Re-run your bootstrapping procedure with this new curve. Comment on the changes to the forward rates.

[18]:
```
# Convert the original inner function into a standalone function
def swap_present_value(f1, sw1):
    # Calculate the present value (PV)
    PV = 0.5 * sw1 / (1 + 0.5 * f1) + 0.5 * sw1 / (1 + 0.5 * f1)**2
    # Return the difference between PV and the forward rate calculations
    return PV - 0.5 * f1 / (1 + 0.5 * f1) - 0.5 * f1 / (1 + 0.5 * f1)**2

# Modified swap_to_forward_1Y function, uses the args parameter to pass the␣
 ↪additional sw1 parameter
def swap_to_forward_1Y(sw1):
    # fsolve now calls the swap_present_value function, passing sw1 as a␣
 ↪parameter
    forward_rate = fsolve(swap_present_value, x0=sw1, args=(sw1,))
    return forward_rate[0]
```

[19]:
```
f1 = swap_to_forward_1Y(2.8438 / 100)
print(f1)
D1 = 1 / (1 + 0.5*f1)**2
print(D1)
```

```
0.028437999999999998
0.9721572416487895
```

[21]:
```
# Standalone function to calculate the present value difference for a two-year␣
 ↪swap
def swap_present_value(f2, sw2, f1):
    # Calculate the present value of swap payments
    PV1 = sw2 / (1 + 0.5 * f1) + sw2 / (1 + 0.5 * f1)**2 + sw2 / ((1 + 0.5 *␣
 ↪f1)**2 * (1 + 0.5 * f2)) + sw2 / ((1 + 0.5 * f1)**2 * (1 + 0.5 * f2)**2)
    # Calculate the present value of forward rate payments
    PV2 = f1 / (1 + 0.5 * f1) + f1 / (1 + 0.5 * f1)**2 + f2 / ((1 + 0.5 *␣
 ↪f1)**2 * (1 + 0.5 * f2)) + f2 / ((1 + 0.5 * f1)**2 * (1 + 0.5 * f2)**2)
```

```
    # Return the difference in present values
    return PV1 - PV2

# Function to solve for the second year's forward rate
def swap_to_forward_2Y(sw2, f1):
    # Use fsolve to solve for the second year's forward rate
    forward_rate_2Y = fsolve(swap_present_value, x0=sw2, args=(sw2, f1))
    return forward_rate_2Y[0]

# Calculate the forward rate for the second year
f2 = swap_to_forward_2Y(sw2, f1)
# Calculate the discount factor for the second year
D2 = D1 / ((1 + 0.5 * f2)**2)

f2, D2
```

[21]:
```
(0.032831130386268975, 0.9410092691367571)
```

[22]:
```
def swap_present_value_3Y(f3, sw3, f1, f2):
    PV1 = sw3 / (1 + 0.5 * f1) + sw3 / (1 + 0.5 * f1)**2 + sw3 / ((1 + 0.5 *␣
 ↪f1)**2 * (1 + 0.5 * f2)) + sw3 / ((1 + 0.5 * f1)**2 * (1 + 0.5 * f2)**2) +␣
 ↪sw3 / ((1 + 0.5 * f1)**2 * (1 + 0.5 * f2)**2 * (1 + 0.5 * f3)) + sw3 / ((1 +␣
 ↪0.5 * f1)**2 * (1 + 0.5 * f2)**2 * (1 + 0.5 * f3)**2)
    PV2 = f1 / (1 + 0.5 * f1) + f1 / (1 + 0.5 * f1)**2 + f2 / ((1 + 0.5 *␣
 ↪f1)**2 * (1 + 0.5 * f2)) + f2 / ((1 + 0.5 * f1)**2 * (1 + 0.5 * f2)**2) + f3␣
 ↪/ ((1 + 0.5 * f1)**2 * (1 + 0.5 * f2)**2 * (1 + 0.5 * f3)) + f3 / ((1 + 0.5␣
 ↪* f1)**2 * (1 + 0.5 * f2)**2 * (1 + 0.5 * f3)**2)
    return PV1 - PV2

def swap_to_forward_3Y(sw3, f1, f2):
    forward_rate_3Y = fsolve(swap_present_value_3Y, x0=sw3, args=(sw3, f1, f2))
    return forward_rate_3Y[0]


f3 = swap_to_forward_3Y(sw3, f1, f2)

D3 = D2 / ((1 + 0.5 * f3)**2)

f3, D3
```

[22]:
```
(0.03264530551203545, 0.9110258289278597)
```

[23]:
```
def swap_present_value_4Y(f4, sw4, f1, f2, f3):
```

```
    PV = sw4/(1+0.5*f1) + sw4/(1+0.5*f1)**2 + sw4/((1+0.5*f1)**2*(1+0.5*f2)) +␣
 ↪sw4/((1+0.5*f1)**2*(1+0.5*f2)**2) + sw4/((1+0.5*f1)**2*(1+0.5*f2)**2*(1+0.
 ↪5*f3)) + sw4/((1+0.5*f1)**2*(1+0.5*f2)**2*(1+0.5*f3)**2) + sw4/((1+0.
 ↪5*f1)**2*(1+0.5*f2)**2*(1+0.5*f3)**2*(1+0.5*f4)) + sw4/((1+0.5*f1)**2*(1+0.
 ↪5*f2)**2*(1+0.5*f3)**2*(1+0.5*f4)**2)
    PV2 = f1/(1+0.5*f1) + f1/(1+0.5*f1)**2 + f2/((1+0.5*f1)**2*(1+0.5*f2)) + f2/
 ↪((1+0.5*f1)**2*(1+0.5*f2)**2) + f3/((1+0.5*f1)**2*(1+0.5*f2)**2*(1+0.5*f3))␣
 ↪+ f3/((1+0.5*f1)**2*(1+0.5*f2)**2*(1+0.5*f3)**2) + f4/((1+0.5*f1)**2*(1+0.
 ↪5*f2)**2*(1+0.5*f3)**2*(1+0.5*f4)) + f4/((1+0.5*f1)**2*(1+0.5*f2)**2*(1+0.
 ↪5*f3)**2*(1+0.5*f4)**2)
    return PV - PV2


def swap_to_forward_4Y(sw4, f1, f2, f3):
    forward_rate_4Y = fsolve(lambda f4: swap_present_value_4Y(f4, sw4, f1, f2,␣
 ↪f3), sw4)
    return forward_rate_4Y[0]

f4 = swap_to_forward_4Y(sw4, f1, f2, f3)

D4 = D3 / ((1 + 0.5 * f4)**2)

f4,D4
```

[23]:
```
(0.03411899988499989, 0.8807202227901273)
```

[24]:
```
def swap_present_value_5Y(f5, sw5, f1, f2, f3, f4):
    PV = sw5*sum((1+0.5*f1)**-i for i in range(1, 3)) + sw5/((1+0.5*f1)**2*(1+0.
 ↪5*f2)) + sw5/((1+0.5*f1)**2*(1+0.5*f2)**2) + sw5/((1+0.5*f1)**2*(1+0.
 ↪5*f2)**2*(1+0.5*f3)) + sw5/((1+0.5*f1)**2*(1+0.5*f2)**2*(1+0.5*f3)**2) + sw5/
 ↪((1+0.5*f1)**2*(1+0.5*f2)**2*(1+0.5*f3)**2*(1+0.5*f4)) + sw5/((1+0.
 ↪5*f1)**2*(1+0.5*f2)**2*(1+0.5*f3)**2*(1+0.5*f4)**2) + sw5/((1+0.
 ↪5*f1)**2*(1+0.5*f2)**2*(1+0.5*f3)**2*(1+0.5*f4)**2*(1+0.5*f5)) + sw5/((1+0.
 ↪5*f1)**2*(1+0.5*f2)**2*(1+0.5*f3)**2*(1+0.5*f4)**2*(1+0.5*f5)**2)
    PV2 = f1/(1+0.5*f1) + f1/(1+0.5*f1)**2 + f2/((1+0.5*f1)**2*(1+0.5*f2)) + f2/
 ↪((1+0.5*f1)**2*(1+0.5*f2)**2) + f3/((1+0.5*f1)**2*(1+0.5*f2)**2*(1+0.5*f3))␣
 ↪+ f3/((1+0.5*f1)**2*(1+0.5*f2)**2*(1+0.5*f3)**2) + f4/((1+0.5*f1)**2*(1+0.
 ↪5*f2)**2*(1+0.5*f3)**2*(1+0.5*f4)) + f4/((1+0.5*f1)**2*(1+0.5*f2)**2*(1+0.
 ↪5*f3)**2*(1+0.5*f4)**2) + f5/((1+0.5*f1)**2*(1+0.5*f2)**2*(1+0.5*f3)**2*(1+0.
 ↪5*f4)**2*(1+0.5*f5)) + f5/((1+0.5*f1)**2*(1+0.5*f2)**2*(1+0.5*f3)**2*(1+0.
 ↪5*f4)**2*(1+0.5*f5)**2)
    return PV - PV2

def swap_to_forward_5Y(sw5, f1, f2, f3, f4):

    forward_rate_5Y = fsolve(lambda f5: swap_present_value_5Y(f5, sw5, f1, f2,␣
 ↪f3, f4), sw5)
```

```
    return forward_rate_5Y[0]

f5 = swap_to_forward_5Y(sw5, f1, f2, f3, f4)


D5 = D4 / (1 + 0.5 * f5)**2

f5,D5
```

[24]: (0.034936952865818174, 0.8507384140726433)

[25]:
```
def swap_present_value_7Y(f7, sw7, f1, f2, f3, f4, f5, D1, D2, D3, D4, D5):
    PV = (sw7 * sum((1 + 0.5 * f1) ** -i for i in range(1, 3)) +
          sw7 * D1 * sum((1 + 0.5 * f2) ** -i for i in range(1, 3)) +
          sw7 * D2 * sum((1 + 0.5 * f3) ** -i for i in range(1, 3)) +
          sw7 * D3 * sum((1 + 0.5 * f4) ** -i for i in range(1, 3)) +
          sw7 * D4 * sum((1 + 0.5 * f5) ** -i for i in range(1, 3)) +
          sw7 * D5 * sum((1 + 0.5 * f7) ** -i for i in range(1, 5)))
    PV2 = (f1 * sum((1 + 0.5 * f1) ** -i for i in range(1, 3)) +
           f2 * D1 * sum((1 + 0.5 * f2) ** -i for i in range(1, 3)) +
           f3 * D2 * sum((1 + 0.5 * f3) ** -i for i in range(1, 3)) +
           f4 * D3 * sum((1 + 0.5 * f4) ** -i for i in range(1, 3)) +
           f5 * D4 * sum((1 + 0.5 * f5) ** -i for i in range(1, 3)) +
           f7 * D5 * sum((1 + 0.5 * f7) ** -i for i in range(1, 5)))
    return PV - PV2

def swap_to_forward_7Y(sw7, f1, f2, f3, f4, f5, D1, D2, D3, D4, D5):
    forward_rate_7Y = fsolve(lambda f7: swap_present_value_7Y(f7, sw7, f1, f2,␣
 ↪f3, f4, f5, D1, D2, D3, D4, D5), sw7)
    return forward_rate_7Y[0]

f7 = swap_to_forward_7Y(sw7, f1, f2, f3, f4, f5, D1, D2, D3, D4, D5)


D7 = D5 / (1 + f7 * 0.5) ** 4

f7,D7
```

[25]: (0.035134972565236094, 0.7934931028923888)

[26]:
```
def swap_present_value_10Y(f10, sw10, f1, f2, f3, f4, f5, f7, D1, D2, D3, D4,␣
 ↪D5, D7):
    PV = (sw10 * sum((1 + 0.5 * f1) ** -i for i in range(1, 3)) +
          sw10 * D1 * sum((1 + 0.5 * f2) ** -i for i in range(1, 3)) +
          sw10 * D2 * sum((1 + 0.5 * f3) ** -i for i in range(1, 3)) +
          sw10 * D3 * sum((1 + 0.5 * f4) ** -i for i in range(1, 3)) +
          sw10 * D4 * sum((1 + 0.5 * f5) ** -i for i in range(1, 3)) +
```

```
          sw10 * D5 * sum((1 + 0.5 * f7) ** -i for i in range(1, 5)) +
          sw10 * D7 * sum((1 + 0.5 * f10) ** -i for i in range(1, 7)))
    PV2 = (f1 * sum((1 + 0.5 * f1) ** -i for i in range(1, 3)) +
           f2 * D1 * sum((1 + 0.5 * f2) ** -i for i in range(1, 3)) +
           f3 * D2 * sum((1 + 0.5 * f3) ** -i for i in range(1, 3)) +
           f4 * D3 * sum((1 + 0.5 * f4) ** -i for i in range(1, 3)) +
           f5 * D4 * sum((1 + 0.5 * f5) ** -i for i in range(1, 3)) +
           f7 * D5 * sum((1 + 0.5 * f7) ** -i for i in range(1, 5)) +
           f10 * D7 * sum((1 + 0.5 * f10) ** -i for i in range(1, 7)))
    return PV - PV2

def swap_to_forward_10Y(sw10, f1, f2, f3, f4, f5, f7, D1, D2, D3, D4, D5, D7):
    forward_rate_10Y = fsolve(lambda f10: swap_present_value_10Y(f10, sw10, f1,␣
 ↪f2, f3, f4, f5, f7, D1, D2, D3, D4, D5, D7), sw10)
    return forward_rate_10Y[0]

f10 = swap_to_forward_10Y(sw10, f1, f2, f3, f4, f5, f7, D1, D2, D3, D4, D5, D7)

D10 = D7 / (1 + f10 * 0.5) ** 6
f10,D10
```

[26]:
```
(0.03853787653877865, 0.7076368151287791)
```

[27]:
```
# Standalone swap_present_value function
def swap_present_value_30Y(f30, sw30, f1, f2, f3, f4, f5, f7, f10, D1, D2, D3,␣
 ↪D4, D5, D7, D10):
    PV = (sw30 * sum((1 + 0.5 * f1) ** -i for i in range(1, 3)) +
          sw30 * D1 * sum((1 + 0.5 * f2) ** -i for i in range(1, 3)) +
          sw30 * D2 * sum((1 + 0.5 * f3) ** -i for i in range(1, 3)) +
          sw30 * D3 * sum((1 + 0.5 * f4) ** -i for i in range(1, 3)) +
          sw30 * D4 * sum((1 + 0.5 * f5) ** -i for i in range(1, 3)) +
          sw30 * D5 * sum((1 + 0.5 * f7) ** -i for i in range(1, 5)) +
          sw30 * D7 * sum((1 + 0.5 * f10) ** -i for i in range(1, 7)) +
          sw30 * D10 * sum((1 + 0.5 * f30) ** -i for i in range(1, 41)))
    PV2 = (f1 * sum((1 + 0.5 * f1) ** -i for i in range(1, 3)) +
           f2 * D1 * sum((1 + 0.5 * f2) ** -i for i in range(1, 3)) +
           f3 * D2 * sum((1 + 0.5 * f3) ** -i for i in range(1, 3)) +
           f4 * D3 * sum((1 + 0.5 * f4) ** -i for i in range(1, 3)) +
           f5 * D4 * sum((1 + 0.5 * f5) ** -i for i in range(1, 3)) +
           f7 * D5 * sum((1 + 0.5 * f7) ** -i for i in range(1, 5)) +
           f10 * D7 * sum((1 + 0.5 * f10) ** -i for i in range(1, 7)) +
           f30 * D10 * sum((1 + 0.5 * f30) ** -i for i in range(1, 41)))
    return PV - PV2

# Refactored swap_to_forward_30Y function
def swap_to_forward_30Y(sw30, f1, f2, f3, f4, f5, f7, f10, D1, D2, D3, D4, D5,␣
 ↪D7, D10):
```

```
    forward_rate_30Y = fsolve(lambda f30: swap_present_value_30Y(f30, sw30, f1,␣
 ↪f2, f3, f4, f5, f7, f10, D1, D2, D3, D4, D5, D7, D10), sw30)
    return forward_rate_30Y[0]

# Using the refactored function to calculate f30
f30 = swap_to_forward_30Y(sw30, f1, f2, f3, f4, f5, f7, f10, D1, D2, D3, D4,␣
 ↪D5, D7, D10)

# Calculating D30
D30 = D10 / (1 + f30 * 0.5) ** 20
f30,D30
```

[27]:
```
(0.03978366720532815, 0.47723043447913865)
```

[28]:
```
f_list_new = [f1,f2,f3,f4,f5,f7,f10,f30]
f_list_new
```

[28]:
```
[0.028437999999999998,
 0.032831130386268975,
 0.03264530551203545,
 0.03411899988499989,
 0.034936952865818174,
 0.035134972565236094,
 0.03853787653877865,
 0.03978366720532815]
```

After applying the bearish steepener adjustments to the swap rates and recalculating the forward rates, here are the key points:

1. **No Change in Short Term**: The forward rates for the 1-year and 2-year terms didn't change, reflecting stability in the short-term interest rate expectations.
2. **Increases in Longer Terms**: Starting from the 4-year term forward, rates increased, more noticeably for the longer maturities (7Y, 10Y, and 30Y). This change is due to the applied steepener, indicating expectations of rising rates or inflation over time.
3. **Steepening Curve**: The curve steepened, especially in the longer end, showing a market expectation of higher interest rates or economic growth/inflation in the future.
4. **Impact of Bearish Steepener**: The adjustments suggest a market view that anticipates higher future risks or inflationary pressures, especially affecting longer-term financial planning and strategies.

(i) Consider a bull steepener to the swap rates, that is perform the following shifts on each swap rate:

Term	Rate change (bps)
1Y	-50
2Y	-25

Term	Rate change (bps)
3Y	-15
4Y	-10
5Y	-5
7Y	+0
10Y	+0
30Y	+50

Print the new swap rates.

[39]:
```
swap_list=[0.028438,0.03060,0.03126,0.03144,0.03150,0.03169,0.03210,0.03237]
basis_list=[-0.005,-0.0025,-0.0015,-0.001,-0.0005,0,0,0]

sw1,sw2,sw3,sw4,sw5,sw7,sw10,sw30=[sw + basis for sw,basis in␣
 ↪zip(swap_list,basis_list)]
sw1,sw2,sw3,sw4,sw5,sw7,sw10,sw30
```

[39]:
```
(0.023438,
 0.0281,
 0.02976,
 0.030440000000000002,
 0.031,
 0.03169,
 0.0321,
 0.03237)
```

(j) Re-run your bootstrapping procedure with this new curve. Comment on the changes to the forward rates.

[30]:
```
def swap_present_value(f1, sw1):
    PV = 0.5 * sw1 / (1 + 0.5 * f1) + 0.5 * sw1 / (1 + 0.5 * f1)**2
    return PV - 0.5 * f1 / (1 + 0.5 * f1) - 0.5 * f1 / (1 + 0.5 * f1)**2

def swap_to_forward_1Y(sw1):
    forward_rate = fsolve(swap_present_value, x0=sw1, args=(sw1,))
    return forward_rate[0]

f1 = swap_to_forward_1Y(sw1)
print(f1)
D1 = 1 / (1 + 0.5*f1)**2
print(D1)
```

```
0.023438
0.9769676601655329
```

[31]:
```
def swap_present_value(f2, sw2, f1):
    PV1 = sw2 / (1 + 0.5 * f1) + sw2 / (1 + 0.5 * f1)**2 + sw2 / ((1 + 0.5 *␣
 ↪f1)**2 * (1 + 0.5 * f2)) + sw2 / ((1 + 0.5 * f1)**2 * (1 + 0.5 * f2)**2)
```

```
    PV2 = f1 / (1 + 0.5 * f1) + f1 / (1 + 0.5 * f1)**2 + f2 / ((1 + 0.5 *␣
 ↪f1)**2 * (1 + 0.5 * f2)) + f2 / ((1 + 0.5 * f1)**2 * (1 + 0.5 * f2)**2)
    return PV1 - PV2


def swap_to_forward_2Y(sw2, f1):
    forward_rate_2Y = fsolve(swap_present_value, x0=sw2, args=(sw2, f1))
    return forward_rate_2Y[0]


f2 = swap_to_forward_2Y(sw2,f1)

D2 = D1 / ((1 + 0.5 * f2)**2)

f2, D2
```

[31]: (0.03290535948429921, 0.9455965031506183)

[32]:
```
def swap_present_value_3Y(f3, sw3, f1, f2):
    PV1 = sw3 / (1 + 0.5 * f1) + sw3 / (1 + 0.5 * f1)**2 + sw3 / ((1 + 0.5 *␣
 ↪f1)**2 * (1 + 0.5 * f2)) + sw3 / ((1 + 0.5 * f1)**2 * (1 + 0.5 * f2)**2) +␣
 ↪sw3 / ((1 + 0.5 * f1)**2 * (1 + 0.5 * f2)**2 * (1 + 0.5 * f3)) + sw3 / ((1 +␣
 ↪0.5 * f1)**2 * (1 + 0.5 * f2)**2 * (1 + 0.5 * f3)**2)
    PV2 = f1 / (1 + 0.5 * f1) + f1 / (1 + 0.5 * f1)**2 + f2 / ((1 + 0.5 *␣
 ↪f1)**2 * (1 + 0.5 * f2)) + f2 / ((1 + 0.5 * f1)**2 * (1 + 0.5 * f2)**2) + f3␣
 ↪/ ((1 + 0.5 * f1)**2 * (1 + 0.5 * f2)**2 * (1 + 0.5 * f3)) + f3 / ((1 + 0.5␣
 ↪* f1)**2 * (1 + 0.5 * f2)**2 * (1 + 0.5 * f3)**2)

    return PV1 - PV2

def swap_to_forward_3Y(sw3, f1, f2):
    forward_rate_3Y = fsolve(swap_present_value_3Y, x0=sw3, args=(sw3, f1, f2))
    return forward_rate_3Y[0]

f3 = swap_to_forward_3Y(sw3, f1, f2)

D3 = D2 / ((1 + 0.5 * f3)**2)

f3, D3
```

[32]: (0.033243751341449745, 0.9149280790879107)

[33]:
```
def swap_present_value_4Y(f4, sw4, f1, f2, f3):
```

```
    PV = sw4/(1+0.5*f1) + sw4/(1+0.5*f1)**2 + sw4/((1+0.5*f1)**2*(1+0.5*f2)) +␣
 ↪sw4/((1+0.5*f1)**2*(1+0.5*f2)**2) + sw4/((1+0.5*f1)**2*(1+0.5*f2)**2*(1+0.
 ↪5*f3)) + sw4/((1+0.5*f1)**2*(1+0.5*f2)**2*(1+0.5*f3)**2) + sw4/((1+0.
 ↪5*f1)**2*(1+0.5*f2)**2*(1+0.5*f3)**2*(1+0.5*f4)) + sw4/((1+0.5*f1)**2*(1+0.
 ↪5*f2)**2*(1+0.5*f3)**2*(1+0.5*f4)**2)
    PV2 = f1/(1+0.5*f1) + f1/(1+0.5*f1)**2 + f2/((1+0.5*f1)**2*(1+0.5*f2)) + f2/
 ↪((1+0.5*f1)**2*(1+0.5*f2)**2) + f3/((1+0.5*f1)**2*(1+0.5*f2)**2*(1+0.5*f3))␣
 ↪+ f3/((1+0.5*f1)**2*(1+0.5*f2)**2*(1+0.5*f3)**2) + f4/((1+0.5*f1)**2*(1+0.
 ↪5*f2)**2*(1+0.5*f3)**2*(1+0.5*f4)) + f4/((1+0.5*f1)**2*(1+0.5*f2)**2*(1+0.
 ↪5*f3)**2*(1+0.5*f4)**2)
    return PV - PV2


def swap_to_forward_4Y(sw4, f1, f2, f3):
    forward_rate_4Y = fsolve(lambda f4: swap_present_value_4Y(f4, sw4, f1, f2,␣
 ↪f3), sw4)
    return forward_rate_4Y[0]

f4 = swap_to_forward_4Y(sw4, f1, f2, f3)


D4 = D3 / ((1 + 0.5 * f4)**2)

f4,D4
```

[33]:
```
(0.03261670364494173, 0.8858005940260291)
```

[34]:
```
def swap_present_value_5Y(f5, sw5, f1, f2, f3, f4):
    PV = sw5*sum((1+0.5*f1)**-i for i in range(1, 3)) + sw5/((1+0.5*f1)**2*(1+0.
 ↪5*f2)) + sw5/((1+0.5*f1)**2*(1+0.5*f2)**2) + sw5/((1+0.5*f1)**2*(1+0.
 ↪5*f2)**2*(1+0.5*f3)) + sw5/((1+0.5*f1)**2*(1+0.5*f2)**2*(1+0.5*f3)**2) + sw5/
 ↪((1+0.5*f1)**2*(1+0.5*f2)**2*(1+0.5*f3)**2*(1+0.5*f4)) + sw5/((1+0.
 ↪5*f1)**2*(1+0.5*f2)**2*(1+0.5*f3)**2*(1+0.5*f4)**2) + sw5/((1+0.
 ↪5*f1)**2*(1+0.5*f2)**2*(1+0.5*f3)**2*(1+0.5*f4)**2*(1+0.5*f5)) + sw5/((1+0.
 ↪5*f1)**2*(1+0.5*f2)**2*(1+0.5*f3)**2*(1+0.5*f4)**2*(1+0.5*f5)**2)
    PV2 = f1/(1+0.5*f1) + f1/(1+0.5*f1)**2 + f2/((1+0.5*f1)**2*(1+0.5*f2)) + f2/
 ↪((1+0.5*f1)**2*(1+0.5*f2)**2) + f3/((1+0.5*f1)**2*(1+0.5*f2)**2*(1+0.5*f3))␣
 ↪+ f3/((1+0.5*f1)**2*(1+0.5*f2)**2*(1+0.5*f3)**2) + f4/((1+0.5*f1)**2*(1+0.
 ↪5*f2)**2*(1+0.5*f3)**2*(1+0.5*f4)) + f4/((1+0.5*f1)**2*(1+0.5*f2)**2*(1+0.
 ↪5*f3)**2*(1+0.5*f4)**2) + f5/((1+0.5*f1)**2*(1+0.5*f2)**2*(1+0.5*f3)**2*(1+0.
 ↪5*f4)**2*(1+0.5*f5)) + f5/((1+0.5*f1)**2*(1+0.5*f2)**2*(1+0.5*f3)**2*(1+0.
 ↪5*f4)**2*(1+0.5*f5)**2)
    return PV - PV2


def swap_to_forward_5Y(sw5, f1, f2, f3, f4):
```

```
    forward_rate_5Y = fsolve(lambda f5: swap_present_value_5Y(f5, sw5, f1, f2,␣
 ↪f3, f4), sw5)
    return forward_rate_5Y[0]

f5 = swap_to_forward_5Y(sw5, f1, f2, f3, f4)

D5 = D4 / (1 + 0.5 * f5)**2

f5,D5
```

[34]:
```
(0.033431395628067 12, 0.856913350718001)
```

[35]:
```
def swap_present_value_7Y(f7, sw7, f1, f2, f3, f4, f5, D1, D2, D3, D4, D5):
    PV = (sw7 * sum((1 + 0.5 * f1) ** -i for i in range(1, 3)) +
          sw7 * D1 * sum((1 + 0.5 * f2) ** -i for i in range(1, 3)) +
          sw7 * D2 * sum((1 + 0.5 * f3) ** -i for i in range(1, 3)) +
          sw7 * D3 * sum((1 + 0.5 * f4) ** -i for i in range(1, 3)) +
          sw7 * D4 * sum((1 + 0.5 * f5) ** -i for i in range(1, 3)) +
          sw7 * D5 * sum((1 + 0.5 * f7) ** -i for i in range(1, 5)))
    PV2 = (f1 * sum((1 + 0.5 * f1) ** -i for i in range(1, 3)) +
           f2 * D1 * sum((1 + 0.5 * f2) ** -i for i in range(1, 3)) +
           f3 * D2 * sum((1 + 0.5 * f3) ** -i for i in range(1, 3)) +
           f4 * D3 * sum((1 + 0.5 * f4) ** -i for i in range(1, 3)) +
           f5 * D4 * sum((1 + 0.5 * f5) ** -i for i in range(1, 3)) +
           f7 * D5 * sum((1 + 0.5 * f7) ** -i for i in range(1, 5)))
    return PV - PV2

def swap_to_forward_7Y(sw7, f1, f2, f3, f4, f5, D1, D2, D3, D4, D5):
    forward_rate_7Y = fsolve(lambda f7: swap_present_value_7Y(f7, sw7, f1, f2,␣
 ↪f3, f4, f5, D1, D2, D3, D4, D5), sw7)
    return forward_rate_7Y[0]

f7 = swap_to_forward_7Y(sw7, f1, f2, f3, f4, f5, D1, D2, D3, D4, D5)


D7 = D5 / (1 + f7 * 0.5) ** 4

f7,D7
```

[35]:
```
(0.033627079512032 27, 0.8016256901458785)
```

[36]:
```
def swap_present_value_10Y(f10, sw10, f1, f2, f3, f4, f5, f7, D1, D2, D3, D4,␣
 ↪D5, D7):
    PV = (sw10 * sum((1 + 0.5 * f1) ** -i for i in range(1, 3)) +
          sw10 * D1 * sum((1 + 0.5 * f2) ** -i for i in range(1, 3)) +
          sw10 * D2 * sum((1 + 0.5 * f3) ** -i for i in range(1, 3)) +
          sw10 * D3 * sum((1 + 0.5 * f4) ** -i for i in range(1, 3)) +
```

```
          sw10 * D4 * sum((1 + 0.5 * f5) ** -i for i in range(1, 3)) +
          sw10 * D5 * sum((1 + 0.5 * f7) ** -i for i in range(1, 5)) +
          sw10 * D7 * sum((1 + 0.5 * f10) ** -i for i in range(1, 7)))
    PV2 = (f1 * sum((1 + 0.5 * f1) ** -i for i in range(1, 3)) +
           f2 * D1 * sum((1 + 0.5 * f2) ** -i for i in range(1, 3)) +
           f3 * D2 * sum((1 + 0.5 * f3) ** -i for i in range(1, 3)) +
           f4 * D3 * sum((1 + 0.5 * f4) ** -i for i in range(1, 3)) +
           f5 * D4 * sum((1 + 0.5 * f5) ** -i for i in range(1, 3)) +
           f7 * D5 * sum((1 + 0.5 * f7) ** -i for i in range(1, 5)) +
           f10 * D7 * sum((1 + 0.5 * f10) ** -i for i in range(1, 7)))
    return PV - PV2


def swap_to_forward_10Y(sw10, f1, f2, f3, f4, f5, f7, D1, D2, D3, D4, D5, D7):
    forward_rate_10Y = fsolve(lambda f10: swap_present_value_10Y(f10, sw10, f1,␣
 ↪f2, f3, f4, f5, f7, D1, D2, D3, D4, D5, D7), sw10)
    return forward_rate_10Y[0]


f10 = swap_to_forward_10Y(sw10, f1, f2, f3, f4, f5, f7, D1, D2, D3, D4, D5, D7)


D10 = D7 / (1 + f10 * 0.5) ** 6
f10,D10
```

[36]:
```
(0.03323013440399479, 0.7261600868744981)
```

[37]:
```
# Standalone swap_present_value function
def swap_present_value_30Y(f30, sw30, f1, f2, f3, f4, f5, f7, f10, D1, D2, D3,␣
 ↪D4, D5, D7, D10):
    PV = (sw30 * sum((1 + 0.5 * f1) ** -i for i in range(1, 3)) +
          sw30 * D1 * sum((1 + 0.5 * f2) ** -i for i in range(1, 3)) +
          sw30 * D2 * sum((1 + 0.5 * f3) ** -i for i in range(1, 3)) +
          sw30 * D3 * sum((1 + 0.5 * f4) ** -i for i in range(1, 3)) +
          sw30 * D4 * sum((1 + 0.5 * f5) ** -i for i in range(1, 3)) +
          sw30 * D5 * sum((1 + 0.5 * f7) ** -i for i in range(1, 5)) +
          sw30 * D7 * sum((1 + 0.5 * f10) ** -i for i in range(1, 7)) +
          sw30 * D10 * sum((1 + 0.5 * f30) ** -i for i in range(1, 41)))
    PV2 = (f1 * sum((1 + 0.5 * f1) ** -i for i in range(1, 3)) +
           f2 * D1 * sum((1 + 0.5 * f2) ** -i for i in range(1, 3)) +
           f3 * D2 * sum((1 + 0.5 * f3) ** -i for i in range(1, 3)) +
           f4 * D3 * sum((1 + 0.5 * f4) ** -i for i in range(1, 3)) +
           f5 * D4 * sum((1 + 0.5 * f5) ** -i for i in range(1, 3)) +
           f7 * D5 * sum((1 + 0.5 * f7) ** -i for i in range(1, 5)) +
           f10 * D7 * sum((1 + 0.5 * f10) ** -i for i in range(1, 7)) +
           f30 * D10 * sum((1 + 0.5 * f30) ** -i for i in range(1, 41)))
    return PV - PV2
```

```
# Refactored swap_to_forward_30Y function
def swap_to_forward_30Y(sw30, f1, f2, f3, f4, f5, f7, f10, D1, D2, D3, D4, D5,␣
 ↪D7, D10):
    forward_rate_30Y = fsolve(lambda f30: swap_present_value_30Y(f30, sw30, f1,␣
 ↪f2, f3, f4, f5, f7, f10, D1, D2, D3, D4, D5, D7, D10), sw30)
    return forward_rate_30Y[0]

# Using the refactored function to calculate f30
f30 = swap_to_forward_30Y(sw30, f1, f2, f3, f4, f5, f7, f10, D1, D2, D3, D4,␣
 ↪D5, D7, D10)

# Calculating D30
D30 = D10 / (1 + f30 * 0.5) ** 20
f30,D30
```

[37]:
```
(0.03258709705552466, 0.5255923083478515)
```

[38]:
```
f_list_new = [f1,f2,f3,f4,f5,f7,f10,f30]
f_list_new
```

[38]:
```
[0.023438,
 0.03290535948429921,
 0.033243751341449745,
 0.03261670364494173,
 0.03343139562806712,
 0.03362707951203227,
 0.03323013440399479,
 0.03258709705552466]
```

1. **Short to Mid-Term Rates Drop**: The forward rates for the short to mid-term decrease significantly, reflecting the lower adjustments in swap rates for these periods.
2. **Long-Term Rates Show Mixed Response**: Despite a large increase in the 30-year swap rate, the long-term forward rate doesn't increase as much, indicating a complex view on long-term economic conditions.
3. **Curve Steepens at Short End**: The curve steepens at the short end due to reduced rates, but this steepening doesn't extend uniformly to the long end.
4. **Market Expectations**: The adjustments suggest the market expects lower interest rates or an accommodative policy stance in the near term, with a cautious or mixed outlook for the long term.

In essence, the bull steepener adjustment leads to lower short to medium-term forward rates and a nuanced long-term rate outlook, reflecting mixed market expectations for future economic conditions.

[ ]: