Homework 3

February 6, 2023

1 Homework 3

1.1 FINM 37400 - 2023

1.1.1 UChicago Financial Mathematics

- Mark Hendricks
- hendricks@uchicago.edu

2 1 HBS Case: Fixed-Income Arbitrage in a Financial Crisis (C): Spread and Swap Spread in November 2008

2.1 Simplification of the setup

The date is Nov 4, 2008.

Treasury bond * Suppose the Treasury bond matures exactly 30 years later, on Nov 4, 2038 rather than May 15, 2008. * The YTM of this freshly issued treasury is 4.193% with a semiannual coupon of 4.50%, same as is given in the case. (So we're just changing the maturity date to simplify things, but keeping the market data.)

Swap * The fixed leg of the swap pays semiannually, with swap rate of 4.2560%, as given in the case. * The floating leg of the swap also pays semiannually–not quarterly–such that the payment dates are identical on both legs. Thus, it also resets the floating rate semiannually, not quarterly. * The floating rate of the swap equals the repo rate used in the trade. Thus, these two rates cancel in the financing of the trade. (No need to consider the TED spread.)

[ ]:

```
import pandas as pd
import numpy as np
import datetime
import warnings
import statsmodels.api as sm

from sklearn.linear_model import LinearRegression
import scipy.optimize as optimize
from scipy.optimize import minimize
import matplotlib.pyplot as plt

import matplotlib as mpl
%matplotlib inline
```

```python
plt.style.use('seaborn')
mpl.rcParams['font.family'] = 'serif'
import seaborn as sns
from treasury_cmds import *
```

2.2 1.1

List the projected cashflows on May 4, 2009, exactly six months into the trade, on the first coupon and swap date.

- The first exchange of payments of this trade exactly 6-months ahead require a fixed rate payment of 4.2560% and a floating rate payment of the prevailing yield 6-months prior to the exchange of cashflows. In this case the yield is 4.193%.

[ ]:

```python
t_current = pd.to_datetime('2008-11-4')
t_current
```

[ ]:

```
Timestamp('2008-11-04 00:00:00')
```

[ ]:

```python
USE_PAR_TBOND = False
# Set coupon on 30-year T bond to equal Nov 4 YTM
# Or stick with the coupon rate in the case, for the Aug 2008 T bond

YTM = [0.04193, .0436]

if USE_PAR_TBOND:
    CPNRATE = [YTM[0], .0436]
else:
    CPNRATE = [0.0450, .0436]

SWAPRATE = [.042560, .0408]

TPRICE = 105
PAR = 100

NOTIONAL = 500e6
HAIRCUT = .02

DTIME = .5
tau0 = 30
tau1 = tau0-DTIME
```

[ ]:

```python
SOFR = np.nan
```

2.2.1 Market Environment

```
[ ]: # Since the floating rate equals SOFER, CF from SWAP is just 0 - fixed payment
     summary = pd.DataFrame(index = ['Coupon Rate', 'YTM', 'Swap Rate','Spread'],␣
      ↪columns=['Nov 2008', 'May 2009'], dtype = 'float')
     summary.loc['Coupon Rate'] = CPNRATE
     summary.loc['YTM'] = YTM
     summary.loc['Swap Rate'] = SWAPRATE
     summary.loc['YTM Spread'] = summary.loc['Swap Rate']- summary.loc['YTM']
     summary.loc['coupon spread'] = summary.loc['Swap Rate'] - summary.loc['Coupon␣
      ↪Rate']
     summary.style.format('{:.2%}')
```

```
[ ]: <pandas.io.formats.style.Styler at 0x170126b3b80>
```

```
[ ]: CF = pd.DataFrame(index = ['T bond','repo','Swap (floating)','Swap (fixed)'],␣
      ↪columns=['May 2009'])
     CF.loc['repo'] = SOFR
     CF.loc['Swap (floating)'] = SOFR
     CF.loc[['T bond']] = PAR*CPNRATE[0]/2
     CF.loc[['Swap (fixed)']] = -PAR *SWAPRATE[0]/2
     CF.loc['Net Payment'] = CF.sum(axis = 0)
     CF
```

```
[ ]:                 May 2009
     T bond              2.25
     repo                 NaN
     Swap (floating)      NaN
     Swap (fixed)      -2.128
     Net Payment        0.122
```

The projected cashflows are equal to this formula 6 months ahead is equal to

$V_{swap} = Kfreq[r_k(T_{i-1}, T_t) - c_{swap}]$

- Thus as of May 4, 2009, the Net-Payment from the point of view for the fixed payer is $12, 200, 000. Mills pays the fixed leg of the swap, receives the floating rate which is equal to the repo rate used to borrow his long position in the 30-year treasury bonds. Thus, he is making a slight profit from the exchange of cashflows here.

2.3 1.2

What is the duration of… * the T-bond * the swap

Remember that… * the swap can be decomposed into a fixed-rate bond and a floating-rate note * a floating-rate note has duration equal to the time until the next reset. Thus, at initialization, it has duration equal to 0.5 years.

Is the duration for the "paying-fixed" swap positive or negative? Is it bigger or smaller in magnitude than the T-bond?

For this problem, calculate the Macauley duration and the dollar (Macauley) duration.

[ ]:
```
def duration_closed_form(tau, ytm, cpnrate = None, freq = 2):
    if cpnrate is None:
        cpnrate = ytm
    y = ytm/freq
    c = cpnrate/freq
    T = tau*freq

    if cpnrate == ytm:
        duration = (1+y)/y * (1-1/(1+y)**T)
    else:
        duration = (1+y)/y - (1+y+T*(c-y))/(c*((1+y)**T-1)+y)

    duration /= freq

    return duration
```

[ ]:
```
duration_df = pd.DataFrame(dtype = float, index = ['T bond', 'fixed leg', 'floating leg'], columns = ['duration'])
duration_df.loc['T bond', 'duration'] = duration_closed_form(30, summary.loc['YTM', 'Nov 2008'], summary.loc['Coupon Rate','Nov 2008'])
duration_df.loc['fixed leg','duration'] = duration_closed_form(30,summary.loc['Swap Rate', 'Nov 2008'])
duration_df.loc['floating leg','duration'] = 0.5
duration_df['Dollar Duration'] = duration_df['duration']*np.array([TPRICE,PAR,PAR])
duration_df.loc['swap'] = duration_df.loc['fixed leg'] - duration_df.loc['floating leg']
duration_df.loc['net'] = duration_df.loc['T bond'] - duration_df.loc['swap']
```

[ ]:
```
duration_df
```

[ ]:
```
               duration  Dollar Duration
T bond        17.083633      1793.781472
fixed leg     17.212744      1721.274445
floating leg   0.500000        50.000000
swap          16.712744      1671.274445
net            0.370889       122.507027
```

Match the dollar by ensuring that the ratio of swap contracts equals the ratio of the dollar duration. Mills is buying $500M notional value of swap contracts in the case

[ ]:
```
hedge_ratio = duration_df.loc['swap','Dollar Duration']/duration_df.loc['T bond','Dollar Duration']
contracts = pd.DataFrame(NOTIONAL*np.array([hedge_ratio/TPRICE,-1/PAR]), index = ['T bond','swap'], columns=['positions'])
```

```
[ ]: contracts
```

```
[ ]:            positions
     T bond  4.436689e+06
     swap   -5.000000e+06
```

2.4 1.3

What hedge ratio should be used to balance the notional size of the Treasury bond with the notional size of the swap, such that it is a duration-neutral position?

Specifically, if the trader enters the swap paying fixed on $500 million notional, how large of a position should they take in the Treasury bond?

2.5 1.4

Suppose it is May 4, 2009, exactly six months after putting the trade on.

The spread is at -28 bps due to… * The YTM on a new 30-year bond has risen to 4.36% * The swap rate on a new 30-year swap has dropped to 4.08%

Explain conceptually how this movement impacts the components of the trade.

2.6 1.5

Calculate the value of the position on May 4, 2009, immediately after the first coupon and swap payments and swap reset.

- Calculate the revised price of the Treasury bond by assuming you can apply the (May 4) 30-year YTM as a discount rate to the 29.5 year bond. (We are just using this for a rough approximation. You know that good pricing would require a discount curve, but let's not get bogged down with that here.)
- Calculate the value of the swap by decomposing it into a fixed-rate bond and a floating-rate bond.
  - The 29.5 year fixed-rate leg is priced using the (May 4) 30-year swap rate as a discount rate.
  - The floating-rate leg is priced at par given that floating-rate notes are par immediately after resets.

Note

You are being asked to calculate these valuations using the exact formula between price, cashflows, and YTM discount rate. We are not simply approximating with duration, as we already know the position was set up with zero dollar duration.

From the Discussion 1 notebook, we have this formula expressing a bond's price as a function of the coupon, c, and the YTM, y_j.

$

$$P_j(t,T,c) = \sum_{i=1}^{n-1} \frac{100\left(\frac{c}{2}\right)}{\left(1+\frac{y_j}{2}\right)^{2(T_i-t)}} + \frac{100\left(1+\frac{c}{2}\right)}{\left(1+\frac{y_j}{2}\right)^{2(T-t)}}$$

$

[ ]:

```
def price_treasury_ytm(time_to_maturity, ytm, cpnrate, freq = 2, face = 100):
    c = cpnrate/freq
    y = ytm/freq
    tau = round(time_to_maturity*freq)
    pv = 0
    for i in range(1, tau):
        pv += 1/(1+y)**i
    pv = c*pv + (1+c)/(1+y)**tau
    pv *= face
    return pv
```

[ ]:

```
summary
```

[ ]:

```
                Nov 2008  May 2009
Coupon Rate      0.04500    0.0436
YTM              0.04193    0.0436
Swap Rate        0.04256    0.0408
Spread               NaN       NaN
YTM Spread       0.00063   -0.0028
coupon spread   -0.00244   -0.0028
```

[ ]:

```
prices = pd.DataFrame(index = ['T bond','swap'], dtype = float, columns=['Nov 2008'])



if USE_PAR_TBOND:
    prices.loc['T bond','Nov 2008'] = price_treasury_ytm(tau0, summary.loc['YTM','Nov 2008'], summary.loc['Coupon Rate','Nov 2008'])
else:
    prices.loc['T bond','Nov 2008'] = TPRICE

prices.loc['swap','Nov 2008'] = PAR - PAR

prices.loc['T bond','May 2009'] = price_treasury_ytm(tau1, summary.loc['YTM','May 2009'], summary.loc['Coupon Rate','Nov 2008'])
prices.loc['swap','May 2009'] = price_treasury_ytm(tau1, summary.loc['Swap Rate','May 2009'], summary.loc['Swap Rate','Nov 2008']) - PAR

prices.style.format('${:,.2f}')
```

[ ]:

```
<pandas.io.formats.style.Styler at 0x1701282e700>
```

2.7 1.6

Accounting for the change in value of the positions, as well as the 6-month cashflows paid on May 4, * what is the net profit and loss (pnl) of the position? * what is the return on the equity capital, considering that there was a 2% haircut (equity contribution) on the size of the initial treasury bond position.

```
[ ]: pnl = pd.DataFrame(dtype=float, index =['T bond', 'swap'], columns=['Cashflow'])
     pnl['Cashflow'] = CF.loc[['T bond', 'Swap (fixed)']].values *contracts.abs().
       ↪values
     pnl['CapitalGains'] = prices.diff(axis = 1)['May 2009'].values * contracts.
       ↪values[:,0]

     pnl['total'] = pnl.sum(axis = 1)
     pnl.loc['net'] = pnl.sum()
     pnl.style.format('${:,.2f}')
```

```
[ ]: <pandas.io.formats.style.Styler at 0x1701283c280>
```

```
[ ]: prices
```

```
[ ]:         Nov 2008    May 2009
     T bond     105.0  102.311401
     swap         0.0    3.003349
```

```
[ ]: # Calculating ROC

     capital = pd.DataFrame(prices.iloc[:,0].values * contracts.values[:,0], index =␣
       ↪['T bond', 'swap'], columns = ['Assets'])
     capital['equity'] = capital['Assets']*HAIRCUT

     capital.loc['net'] = capital.sum()
     capital['pnl'] = pnl['total']
     capital['Return'] = capital['pnl']/capital['equity']
     capital.loc[['T bond','swap'], 'Return'] = np.nan

     capital.style.format({'assets':'${:,.2f}','equity':'${:,.2f}','pnl':'${:,.
       ↪2f}','Return':'{:.2%}'},na_rep='')
```

```
[ ]: <pandas.io.formats.style.Styler at 0x17012858b50>
```

```
[ ]:
```

```
[ ]:               Assets         equity           pnl     Return  return
     T bond   4.658523e+08   9.317046e+06 -1.945929e+06 -0.208857     NaN
     swap    -0.000000e+00  -0.000000e+00 -2.565675e+07       inf     NaN
     net      4.658523e+08   9.317046e+06 -2.760268e+07 -2.962599     NaN
```