MAT1856/APM466 Assignment 1

Chia Wei Kit Samuel, Student #: 1009767646

January, 2023

Fundamental Questions - 25 points

1.

   (a) Governments issue bonds instead of simply printing more money to control inflation, maintain transparency and accountability, ensure market discipline, and diversify funding sources.

   (b) A flattening of the long-term part of the yield curve could occur if investors expect the central bank to keep short-term interest rates low for a prolonged period, reducing the difference between short- and long-term rates.

   (c) Quantitative easing is a monetary policy tool used by the Fed to combat the economic downturn caused by the COVID-19 pandemic by purchasing large quantities of debt securities and other financial assets, lowering long-term interest rates and providing liquidity to the financial system to stimulate the economy.

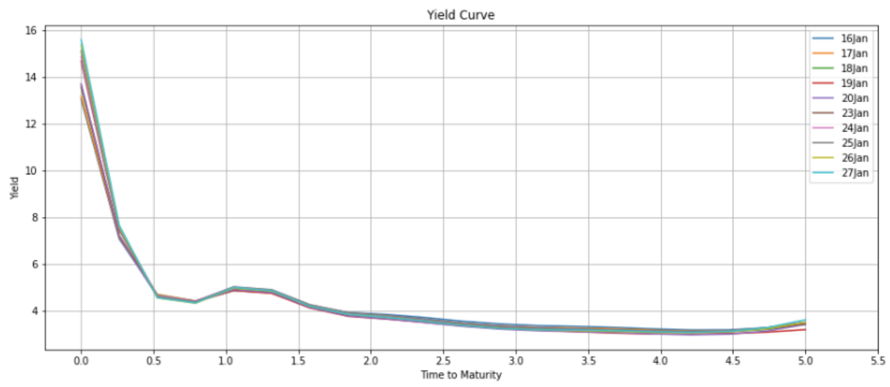
2. Bonds chosen = 'CAN 1.75 Mar 23', 'CAN 0.25 Aug 23', 'CAN 2.25 Mar 24', 'CAN 1.50 Sep 24', 'CAN 1.25 Mar 25', 'CAN 0.50 Sep 25', 'CAN 0.25 Mar 26', 'CAN 1.00 Sep 26', 'CAN 1.25 Mar 27', 'CAN 2.75 Sep 27'. We should choose bonds with consistently spaced maturity dates to construct the 0-5 year yield and spot curves. Since the government of Canada issues all of its bonds with a semi-annual (6-month) coupon, we would ideally choose 2 bonds per year in our 5-year curves, each with a maturity date exactly 6 months apart. Based on the bonds available, we pick the government-issued bonds maturing in March and September every year from 2023 to 2027. However, no bond maturing in September 2023 exists thus we choose 'CAN 0.25 Aug 23', the nearest month, to replace that.

3. The eigenvalues and eigenvectors of the covariance matrix of several stochastic processes from a stochastic curve represent the "principal components" which tell us about the characteristics of the stochastic curve. This is known as Principal Component Analysis (PCA). The eigenvectors are the directions along which the processes have the most variability, and the corresponding eigenvalues represent the amount of variability in those directions. So, if we have a large eigenvalue, it means there's a lot of variability in the corresponding eigenvector direction, and if we have a small eigenvalue, it means there's less variability in that direction. By analyzing the eigenvalues and eigenvectors of the covariance matrix, we can determine the most important factors that drive the variability in the stochastic processes.

Empirical Questions - 75 points

4.

   (a) Interpolation technique used: scipy.interpolate.CubicSpline method. This will allow us to interpolate the data with a piecewise cubic polynomial, modelling the yield curve as a function with different definitions across different ranges.



The above is an inverted yield curve which occurs when the interest rates on long-term debt instruments are lower than the interest rates on short-term debt instruments, which is the opposite of the typical relationship between rates and maturity. This phenomenon is considered a strong predictor of a future recession, as investors often demand a higher return for longer-term investments due to increased uncertainty. An inverted yield curve can also indicate that the market is expecting lower future economic growth and inflation.

(b) Based on the general formula

$$P = \sum_i p_i e^{-r(t_i)t_i}$$

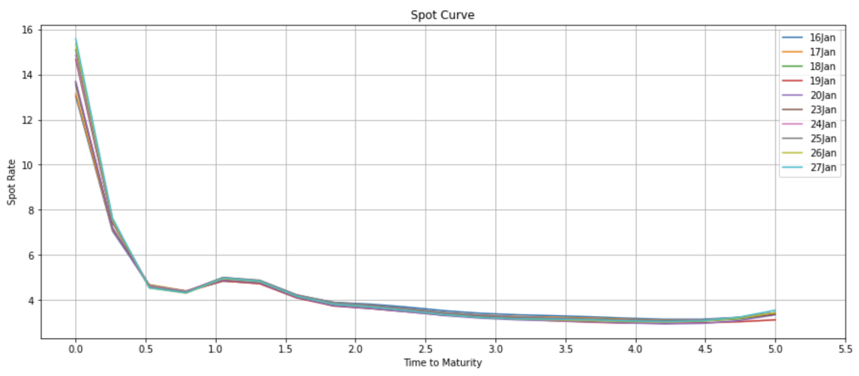
where P = price, p_i = cashflow at time i, $r(t_i)$ = spot yield at time i, t_i = number of years to time i.
The spot yield at time n can be found by the equation

$$r(t_n) = \frac{lg(p_n) - lg(P - \sum_{i=1}^{n-1} p_i e^{-r(t_i)*t_i})}{t_n}$$

With a list of bonds in ascending order of maturity dates, each exactly 6 months in between, we use the boostrapping technique to calculate the spot rate for each bond sequentially as the equation will depend on the spot rates we calculated for each bond before.

Algorithm 1 calculate_spot_rate(bonds)

```
1: Assign empty list as spot_rates
2: for i in length(bonds): do
3:     if i == 0 then
4:         $Y_i = -\frac{log(P/facevalue+coupon)}{t_n}$
5:     end if
6:     if $i > 0$ then
7:         $Y_i = \frac{lg(p_n)-lg(P-\sum_{i=1}^{n-1} p_i e^{-r(t_i)*t_i})}{t_n}$, where $r(t_i)$ = spot_rates[i]
8:     end if
9:     spot_rates.append($Y_i$)
10: end for
11: return spot_rates
```



(c) A forward interest rate acts as a discount rate for a single payment from a further future date to a closer future date.

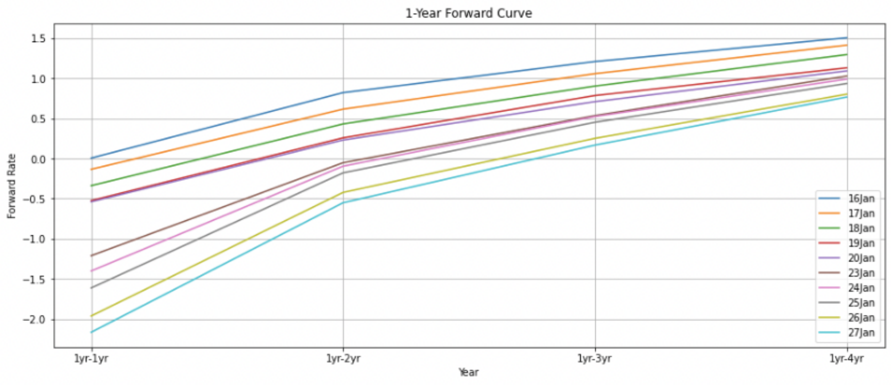
$$e^{r_{01}*1} * e^{f_{1t}(t-1)} = e^{r_{0t}*t}$$

$$f_{1t} = \frac{r_{0t} * t - r_{01}}{t-1}$$

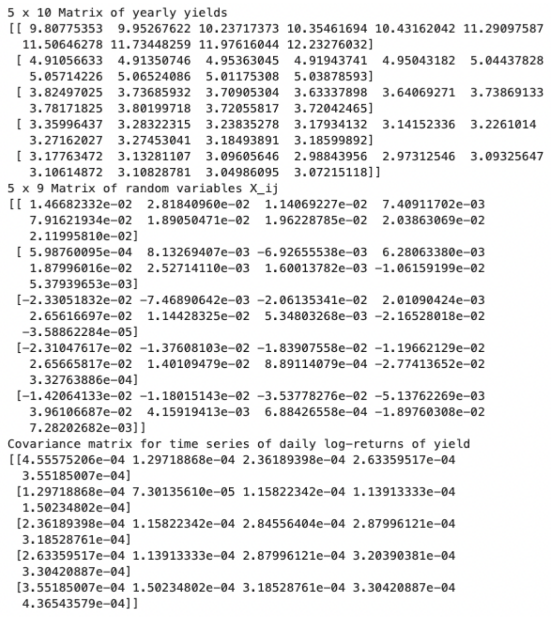
Where r_{0n} : spot rate from year 0 to year n , f_{1n} : forward rate from year 1 to year n.

Algorithm 2 calculate_forward_rate(march_spot_rates)

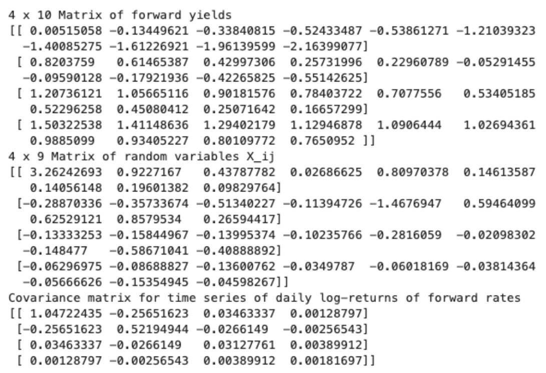
```
   Assign empty list as forward_rates
2: $r_{01}$ = march_spot_rates[0]
   **for** i in range(1, length(march_spot_rates)): **do**
4:     forward_rate = (march_spot_rates[i] * (i+1) - $r_{01}$) / i
       march_spot_rates.append(forward_rate)
6: **end for**
   **return** march_spot_rates
```



5. For daily log-returns of yield, we create a 5x10 matrix representing the yearly YTM (taken in march) for each of the 10 days we collected data. We can then calculate the random variables $X_{i,j}$ for matrix X by **X = np.log(yearly_yields[1:]/yearly_yields[:-1])**, where X is a 5 x 9 matrix. We then use np.cov() to find the 5 x5 covariance matrix. Similarly for forward rates, we have a 4 x 10 matrix, 10 days for 4 forward rates, 4 x 9 matrix X, 4 x 4 covariance matrix.



```
5 x 10 Matrix of yearly yields
[[ 9.80775353  9.95267622 10.23717373 10.35461694 10.43162042 11.29097587
  11.50646278 11.73448259 11.97616044 12.23276032]
 [ 4.91056633  4.91350746  4.95363045  4.91943741  4.95043182  5.04437828
   5.05714226  5.06524086  5.01175308  5.03878593]
 [ 3.82497025  3.73685932  3.70905304  3.63337898  3.64069271  3.73869133
   3.78171825  3.80199718  3.72055817  3.72042465]
 [ 3.35996437  3.28322315  3.23835278  3.17934132  3.14152336  3.2261014
   3.27162027  3.27453041  3.18493891  3.18599892]
 [ 3.17763472  3.13281107  3.09605646  2.98843956  2.97312546  3.09325647
   3.10614872  3.10828781  3.04986095  3.07215118]]
5 x 9 Matrix of random variables X_ij
[[ 1.46682332e-02  2.81840960e-02  1.14069227e-02  7.40911702e-03
   7.91621934e-02  1.89050471e-02  1.96228785e-02  2.03863069e-02
   2.11995810e-02]
 [ 5.98760095e-04  8.13269407e-03 -6.92655538e-03  6.28063380e-03
   1.87996016e-02  2.52714110e-03  1.60013782e-03 -1.06159199e-02
   5.37939653e-03]
 [-2.33051832e-02 -7.46890642e-03 -2.06135341e-02  2.01090424e-03
   2.65616697e-02  1.14428325e-02  5.34803268e-03 -2.16528018e-02
  -3.58862284e-05]
 [-2.31047617e-02 -1.37608103e-02 -1.83907558e-02 -1.19662129e-02
   2.65665817e-02  1.40109479e-02  8.89114079e-04 -2.77413652e-02
   3.32763886e-04]
 [-1.42064133e-02 -1.18015143e-02 -3.53778276e-02 -5.13762269e-03
   3.96106687e-02  4.15919413e-03  6.88426558e-04 -1.89760308e-02
   7.28202682e-03]]
Covariance matrix for time series of daily log-returns of yield
[[4.55575206e-04 1.29718868e-04 2.36189398e-04 2.63359517e-04
  3.55185007e-04]
 [1.29718868e-04 7.30135610e-05 1.15822342e-04 1.13913333e-04
  1.50234802e-04]
 [2.36189398e-04 1.15822342e-04 2.84556404e-04 2.87996121e-04
  3.18528761e-04]
 [2.63359517e-04 1.13913333e-04 2.87996121e-04 3.20390381e-04
  3.30420887e-04]
 [3.55185007e-04 1.50234802e-04 3.18528761e-04 3.30420887e-04
  4.36543579e-04]]
```

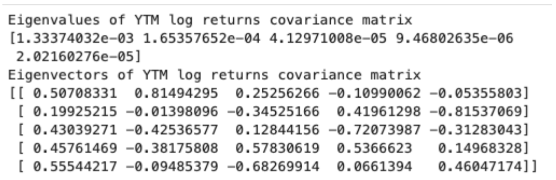


```
4 x 10 Matrix of forward yields
[[ 0.00515058 -0.13449621 -0.33840815 -0.52433487 -0.53861271 -1.21039323
  -1.40085275 -1.61226921 -1.96139599 -2.16399077]
 [ 0.8203759   0.61465387  0.42997306  0.25731996  0.22960789 -0.05291455
  -0.09590128 -0.17921936 -0.42265825 -0.55142625]
 [ 1.20736121  1.05665116  0.90181576  0.78403722  0.7077556   0.53405185
   0.52296258  0.45080412  0.25071642  0.16657299]
 [ 1.50322538  1.41148636  1.29402179  1.12946878  1.0906444   1.02694361
   0.9885099   0.93405227  0.80109772  0.7650952 ]]
4 x 9 Matrix of random variables X_ij
[[ 3.26242693  0.9227167   0.43787782  0.02686625  0.80970378  0.14613587
   0.14056148  0.19601382  0.09829764]
 [-0.28870336 -0.35733674 -0.51340227 -0.11394726 -1.4676947   0.59464099
   0.62529121  0.8579534   0.26594417]
 [-0.13333253 -0.15844967 -0.13995374 -0.10235766 -0.2816059  -0.02098302
  -0.148477   -0.58671041 -0.40888892]
 [-0.06296975 -0.08688827 -0.13600762 -0.0349787  -0.06018169 -0.03814364
  -0.05666626 -0.15354945 -0.04598267]]
Covariance matrix for time series of daily log-returns of forward rates
[[ 1.04722435 -0.25651623  0.03463337  0.00128797]
 [-0.25651623  0.52194944 -0.0266149  -0.00256543]
 [ 0.03463337 -0.0266149   0.03127761  0.00389912]
 [ 0.00128797 -0.00256543  0.00389912  0.00181697]]
```

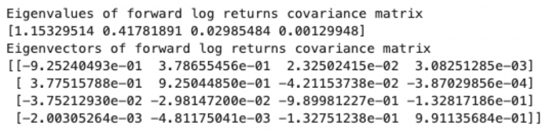
6. The first eigenvector corresponds to the directions of maximum variance in the data and its associated eigenvector indicates the percent of total variance explained by the eigenvector which can be calculated by

$$x = \frac{v}{\sum_i v_i}$$

where x is the percent of total variance and v_i is the ith eigenvalue.



```
Eigenvalues of YTM log returns covariance matrix
[1.33374032e-03 1.65357652e-04 4.12971008e-05 9.46802635e-06
 2.02160276e-05]
Eigenvectors of YTM log returns covariance matrix
[[ 0.50708331  0.81494295  0.25256266 -0.10990062 -0.05355803]
 [ 0.19925215 -0.01398096 -0.34525166  0.41961298 -0.81537069]
 [ 0.43039271 -0.42536577  0.12844156 -0.72073987 -0.31283043]
 [ 0.45761469 -0.38175808  0.57830619  0.5366623   0.14968328]
 [ 0.55544217 -0.09485379 -0.68269914  0.0661394   0.46047174]]
```



```
Eigenvalues of forward log returns covariance matrix
[1.15329514 0.41781891 0.02985484 0.00129948]
Eigenvectors of forward log returns covariance matrix
[[-9.25240493e-01  3.78655456e-01  2.32502415e-02  3.08251285e-03]
 [ 3.77515788e-01  9.25044850e-01 -4.21153738e-02 -3.87029856e-04]
 [-3.75212930e-02 -2.98147200e-02 -9.89981227e-01 -1.32817186e-01]
 [-2.00305264e-03 -4.81175041e-03 -1.32751238e-01  9.91135684e-01]]
```

References and GitHub Link to Code

References

- Campbell, B. (n.d.). Par Curve, Spot Curve, and Forward Curve. Financial exam help 123. Retrieved January 30, 2023, from http://www.financialexamhelp123.com/par-curve-spot-curve-and-forward-curve/

- Jaadi, Z. (n.d.). A step-by-step explanation of principal component analysis (PCA). Built In. Retrieved January 30, 2023, from https://builtin.com/data-science/step-step-explanation-principal-component-analysis

- KumarI, A. (2020, August 7). Eigenvalues amp; eigenvectors with python examples. Data Analytics. Retrieved January 30, 2023, from https://vitalflux.com/eigenvalues-eigenvectors-python-examples/

GitHub: https://github.com/Sampie314/yieldcurve-apm466a1-repo

Overleaf Link: https://www.overleaf.com/read/zdjncknkfwjs