MINI-PROJECT #1 Bootstrapping Yield Curve with US Treasury Notes & Bonds

Methodology

1 Get data. Calculate full prices of US treasury notes and bonds. (in Excel)

1.1 Get data from WSJ website (https://www.wsj.com/market-data/bonds/treasuries)

1.2 Calculate the accrued interest using Excel function *ACCRINT*.

Replace issue date with the last coupon date

Replace first interest date with the next coupon date

Replace the settlement date with the quotation date + 1

1.3 Add up the quote mid price (clean price) and accrued interest, and we get full price.

2 Build the coupon matrix C, which contains the futures payments of each bonds (in Python)

Specific name was assigned to each bond with maturity and coupon rate. Additionally, all coupon dates were obtained. The next step was building coupon matrix based on coupon rates, payment dates and maturity date. There were some dates when no coupon payments were made, and these dated were dropped.

3 Handle the sparsity and singularity of the coupon matrix

The future payments for all bonds from 2031-2038 at several payment dates are the same, which leads to singularity of the coupon matrix. Data are not reliable because the US government had stopped issuing bonds for years. Therefore, we aggregate the coupon payment at the midpoint time.

4 Using regression methods to calculate the discount factor for each term

Since we have removed the singularity in coupon matrix C, we can run a regression to calculate the discount factor at payment dates.

$$C = \begin{pmatrix} c_{11} & \cdots & c_{1m} \\ & \ddots & \vdots \\ c_{n1} & & c_{nm} \end{pmatrix}_{n\times m} \quad p = \begin{pmatrix} p_1 \\ \vdots \\ p_n \end{pmatrix}_{n\times 1} \quad d = \begin{pmatrix} d_1 \\ \vdots \\ d_m \end{pmatrix}_{m\times 1}$$

$$p = Cd$$

$$d = (C^{\mathrm{T}}C)^{-1}C^{\mathrm{T}}p$$

5 Get yield from discount

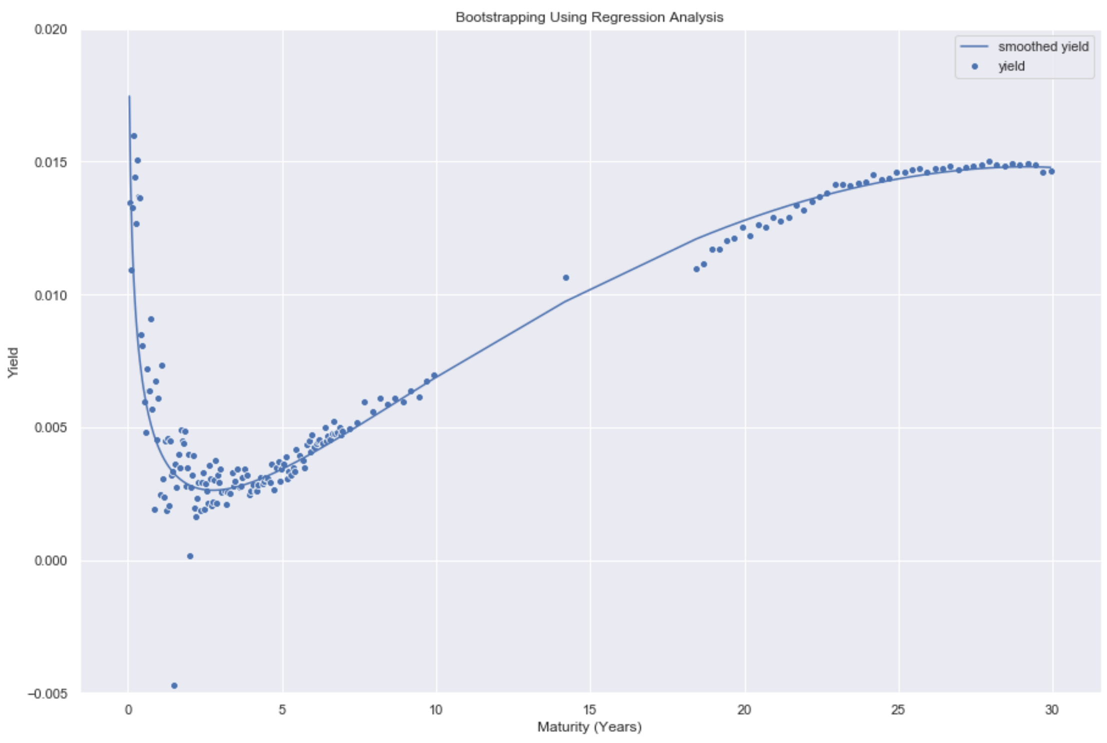
$$yield = \frac{\ln(discount\,factor)}{(-t)}$$

6 Using a regression model to fit the yield curve

$$yield = a_1 * t + a_2 * t^2 + a_3 * \sqrt{t} + a_4 * e^t + a_5 * \ln t$$

7 The results table are in Excel-OutputTable.

Results



(in Jupyter Notebook)

	discount_facotr	*days*	*years*	*yield*	*yield_fit*
9/30/2020	0.999410765	16	0.043835616	0.013445884	0.017458373
10/15/2020	0.999073362	31	0.084931507	0.01091547	0.014213789
10/31/2020	0.998293571	47	0.128767123	0.013263374	0.01224741
11/15/2020	0.997291413	62	0.169863014	0.015967345	0.010979095
11/30/2020	0.996959509	77	0.210958904	0.014434674	0.010013743
12/15/2020	0.996812384	92	0.252054795	0.012666719	0.009240921
12/31/2020	0.995557501	108	0.295890411	0.015047451	0.008562229
1/15/2021	0.99539934	123	0.336986301	0.013683867	0.008025289
1/31/2021	0.994822181	139	0.380821918	0.013631752	0.007532561
2/15/2021	0.996430567	154	0.421917808	0.008475153	0.007129519
2/28/2021	0.996307439	167	0.457534247	0.008085506	0.006817638
3/15/2021	1.010268067	182	0.498630137	-0.020487547	0.006493688
3/31/2021	0.996763099	198	0.542465753	0.005976693	0.006183841

(in Excel)