PHDBA 239FC Part 2: Problem Set 1

Due: Friday, March 25, 2022 by 9:30 a.m.

INSTRUCTIONS: Before you begin, read Stanton (2022a) (on bCourses in directory `handouts`) in detail. The best way to become comfortable with the tools described here is to use them, so doing so is required for this problem set. Let me know how these instructions work for you, and if anything could be clearer (or is just plain wrong...).

Get installed

-18. If you do not already have one, set up an account on GitHub.[1]

-17. If you do not already have one, set up an account on Overleaf.[2]

-16. Install `Git` on your personal computer.[3]

-15. Install a full LaTeX distribution on your personal computer (see Stanton, 2013).[4]

-14. Install the full Anaconda Python distribution (version 3.8) on your personal computer.[5] This includes Jupyter.

-13. (Optional) If you want to use Julia, install it on your personal computer.[6]

-12. Install GNU Make on your personal computer.[7]

-11. Install an Integrated Development Environment (IDE) on your personal computer. I recommend VS Code because it is easy to use, popular, actively developed, and has good support for both Python and Julia. For Python alone, I also like PyCharm and Spyder (which is automatically installed as part of the Anaconda Python distribution). However, neither of these has very good (or, in the case of Spyder, any) support for Julia.

[1] As students, you are entitled to a free GitHub Pro account plus some other benefits (see `https://education.github.com/pack`).

[2] As U.C. Berkeley students, you are entitled to a free Overleaf Professional account (see `https://www.overleaf.com/edu/berkeley`).

[3] This can be a command-line version (I use the MacPorts version on my Mac; you can download Git for Windows at `https://git-scm.com/download/win`), or you may prefer a GUI-based client such as GitKraken or GitHub Desktop.

[4] You can get by using only Overleaf, but you will be much more productive if you edit your files locally. First, this allows you to use the editor of your choice instead of the (rather limited) editor provided by Overleaf. Second, Overleaf doesn't have a concept of directories, so any special files you need for every project (including your master bibliography file) need to be copied from project to project, while on your own machine they can just be installed once.

[5] See `https://www.anaconda.com/products/individual`.

[6] See `https://julialang.org/downloads/`.

[7] I use the MacPorts version on my Mac. For Windows, the easiest way to install GNU Make (and lots of other Unix tools) is via Cygwin.

Learn the Basics of your Tools

-10. Read Stanton (2022a) cover to cover. If you are not familiar with LaTeX, also read Stanton (2013) cover to cover. Both are available on bCourses in directory `handouts`. Then skim the references listed in Stanton (2022a) to learn the basics of the other tools you've just installed (and so you know where to look when you need help later). Mainly, you'll learn the tools by using them.

Set Things Up

-9. Create a new GitHub repository for this problem set (see `https://github.com/new`) called `PHD239FC-PS1-<name>`, where for `<name>` you should substitute your last name (so in my case, the repository would be called `PHD239FC-PS1-stanton`). Make the repository private (you probably don't want the rest of the world to be able to see it), click `Create Repository`, then *immediately*. . .

-8. Follow either the first or second set of instructions that pop up after you've clicked `Create Repository` to create a corresponding repository on your personal computer, linked to the repository you just created on GitHub. For example, I would create a directory called, say, `/courses/PHD239FC/hw1`, then I would go to directory `hw1` and type:

```
echo "# HW 1" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin git@github.com:rhstanton/PHD239FC-PS1-stanton.git
git push -u origin main
```

-7. Share your new GitHub repository with me (at email `rhstanton@berkeley.edu`).

-6. On your personal computer, create subdirectories `notebooks` and `code` underneath `hw1` to put your notebooks and final Python/Julia code in, respectively.

-5. If you will be using Jupyter notebooks (which I strongly recommend for initial exploration, even if at some later point you decide to switch to code you can run from the command line), also install nbstripout to make Git play nicely with your notebooks.

-4. Create an Overleaf project that will contain the write-up for your problem set.[8]

-3. Share your new Overleaf project with me (at email `rhstanton@berkeley.edu`).

-2. Now go to the `hw1` directory on your personal computer, and create a submodule that will contain just the write-up. To do this,

[8]In Overleaf, click "New Project" → "Blank Project", then select a descriptive name for the project, e.g., "Write-up for PHD239FC problem set 1."

- On Overleaf, click "Menu" → "Git" and copy the URL that appears after "git clone", e.g.,

  ```
  https://git.overleaf.com/5e1e5dc63fe82300013bf0e7
  ```

- On your personal computer, go to the directory `hw1` and run the command

  ```
  git submodule add https://git.overleaf.com/5e1e5dc63fe82300013bf0e7 writeup
  ```

- This will create a directory `hw1/writeup` containing a git (sub)repository that is linked to the Overleaf document.
- Now when you run `git push` in directory `hw1` or any subdirectory other than `writeup`, it will push your edits to the repository on GitHub. If you run `git push` in the `writeup` subdirectory, it will push the changes to just the write-up to the Overleaf document (and if you or your coauthors are connected to Overleaf, you'll see your changes suddenly appear in front of them).

-1. Create a Jupyter notebook in subdirectory `notebooks` called `hw1-<name>.ipynb`, where again you should replace `<name>` with your last name. To do this:

- Go to the subdirectory `notebooks` and run the command

  ```
  jupyter notebook
  ```

- A browser window should pop up with a Jupyter window open.
- Click on "New" and select what language you want your notebook to be in (Python 3 or Julia).[9]
- A new window will open, displaying a blank new notebook called something like `Untitled`. Before doing anything else, click "File" → "Rename", and enter the new name `hw1-<name>`.
- Save the (currently blank) notebook file by clicking "File" → "Save and Checkpoint".

0. At the command line in subdirectory `notebooks`, tell Git to keep track of the new notebook file with the command

```
git add hw1-<name>.ipynb
```

Now you're ready to get going! Do your exploratory coding in `notebooks/hw1-<name>.ipynb` and write up your solutions in `writeup/main.tex`.[10] Eventually, you should also create final Python or Julia programs to run the code for each question, and a Makefile to go along with them, all stored in `hw1/code`. Keep track of everything using Git by

- running `git add` to add each new file you create e.g., `Q1.py`, `Makefile`.
- running `git commit` on a regular basis to commit your changes (locally).

[9] If Julia does not appear as an option, you need to install `IJulia`; follow the instructions at https://github.com/JuliaLang/IJulia.jl.

[10] I suggest you practice editing `main.tex` both directly on Overleaf and on your local machine.

- running `git push` both in the main project and in `hw1/writeup` to push your edits to the remote repositories.

1. You take out a 30-year, fixed-rate mortgage for \$1,000,000. The interest rate on the mortgage loan is 5.25% (APR, compounded monthly), and it requires you to make equal payments at the end of each of the next 360 months (i.e., the first payment is 1 month from today and the last payment is 360 months from today).

   (a) What is the amount of each monthly payment?

   (b) What is the remaining balance on the loan immediately after the 100th loan payment?

   (c) How much interest in total will you pay during year 10 of the mortgage (i.e., what is the total amount of interest included in the 12 payments ending with the payment 120 months from today)?

2. The annually compounded one-, two- and three-year spot rates, $r_1(0,1)$, $r_1(0,2)$ and $r_1(0,3)$, are 5%, 6% and 7% respectively.

   (a) What are the first three annually compounded (one-year) forward rates for years 1, 2 and 3, $f_1(0,0,1)$, $f_1(0,1,2)$ and $f_1(0,2,3)$?

   (b) If no explicit forward-rate agreements existed, how would you use a combination of spot lending and borrowing to effectively borrow \$1,000 a year from today, and pay it back 3 years from today (plus interest)?

3. The annually compounded one-, two- and three-year forward rates, $f_1(0,0,1)$, $f_1(0,1,2)$ and $f_1(0,2,3)$, are 11%, 13% and 17% respectively. What are the annually compounded one, two and three-year spot rates, $r_1(0,1)$, $r_1(0,2)$ and $r_1(0,3)$?

4. We defined a bond's yield to maturity in class. An alternative measure that is sometimes quoted is a bond's *current yield*, defined as its coupon rate (as a percentage) divided by its current price (quoted per \$100 face value). So a bond with a 5% annual coupon rate and a current price of \$102 would have a current yield of

$$\frac{5}{102} = 4.0902\%.$$

   Prove that for a discount bond (i.e., price $<$ \$100), YTM $>$ current yield $>$ coupon rate, and for a premium bond (i.e., price $>$ \$100), YTM $<$ current yield $<$ coupon rate.

The next four problems involve fitting yield curves. In principle, to obtain the input data for your analysis, use Bloomberg to download LIBOR/swap rates from Feb. 28, 2019 (see Appendix A for detailed instructions), making sure to use *mid-quotes*, i.e., (Bid + Ask) / 2, in your analysis. **In practice, I have provided the input data for your analysis as file `libor_rates_input_022819.xlsx`. which I downloaded from Bloomberg prior to 2020.**[11]

[11] I am providing the data rather than asking you to download it for two reasons: i. to save you some time, and ii. because Bloomberg changed the way they did their calculations some time around 2020, and it is impossible to match their newer output data as accurately.

5. Finish the fitting example that we went over in class (slides 34–38) by calculating the third forward rate, f_3.

6. Fit a yield curve to the input data you just downloaded, assuming that

   - The *simple interest rate* r_s is a continuous function of maturity, $r_s(t)$.
   - $r_s(t)$ is piecewise linear, with kinks at the maturity dates of the instruments in the input data set.
   - $r_s(t)$ is constant outside the range of maturities in the input data. I.e., if $t_{\min}$ and $t_{\max}$ are, respectively, the shortest and longest maturities in the input sample,

   $$r_s(t) = \begin{cases} r_s(t_{\min}) & \text{if } t \leq t_{\min}, \\ r_s(t_{\max}) & \text{if } t \geq t_{\max}. \end{cases}$$

   [This is Bloomberg's default "method 1" (see Bloomberg, 2021, pp. 7–8). When using this method, assume simple rates are quoted using the Actual/360 day-count.]

   Your goal is to match the zero rates and forward rates calculated by Bloomberg, which I've provided for you.

   Using your fitted yield curve, generate a table showing, at 3-month intervals from 3/4/2019 to 12/4/2068 (use the dates in the output spreadsheet you just downloaded from Bloomberg):

   (a) Your fitted discount factor.
   (b) Your fitted zero rate.
   (c) Bloomberg's fitted zero rate
   (d) Your fitted 3-month forward rate
   (e) Bloomberg's fitted 3-month forward rate
   (f) The difference between your fitted zero rate and Bloomberg's
   (g) The difference between your fitted forward rate and Bloomberg's

   Report all numbers to 9 decimal places, and report the numbers in columns (b)–(g) as percentages (e.g., 2.615130000%). **Your numbers should be identical to Bloomberg's to at least this degree of precision. In other words, the numbers in columns (f) and (g) should all be $\pm$0.000000000%**. To achieve this, you should be aware of the Bloomberg quoting conventions:

   - Zero rates: semi-annually compounded and use 30I/360 day-count.
   - Forward rates: simple compounding and use Actual/360 day-count.

   Note that this is not easy! There are a lot of details to get right to get the numbers to match exactly. In particular, if you manage to match only to, say, 7 decimal places, e.g., a difference of 0.000000026%, your calculations are *not* 100% right…

   [To make this a *lot* easier, read Stanton (2022b) (provided as a handout for this problem set) in great detail. This is a pretty detailed discussion of the

data and the compounding and timing assumptions you will need to make in order to get this to work.]

7. Fit the 5-parameter model of Nelson and Siegel (1987) to your input data. Generate the same table as in Question 6, but including only columns (a), (b) and (d).

8. Repeat Question 7, this time using the model of Svensson (1994).

References

Bloomberg, 2021, Building the Bloomberg interest rate curve: Definitions and methodology, IDOC 2064159.

Nelson, Charles R., and Andrew F. Siegel, 1987, Parsimonious modeling of yield curves, *Journal of Business* 60, 473–489.

Stanton, Richard, 2013, Writing finance papers using LaTeX, http://faculty.haas.berkeley.edu/stanton/texintro/.

Stanton, Richard, 2022a, Managing your research, Working paper, U. C. Berkeley.

Stanton, Richard, 2022b, Notes on fitting the Bloomberg #23 LIBOR/swap curve.

Svensson, Lars E. O., 1994, Estimating and interpreting forward rates: Sweden 1992–1994, Working Paper 4871, NBER.

A Downloading data from Bloomberg

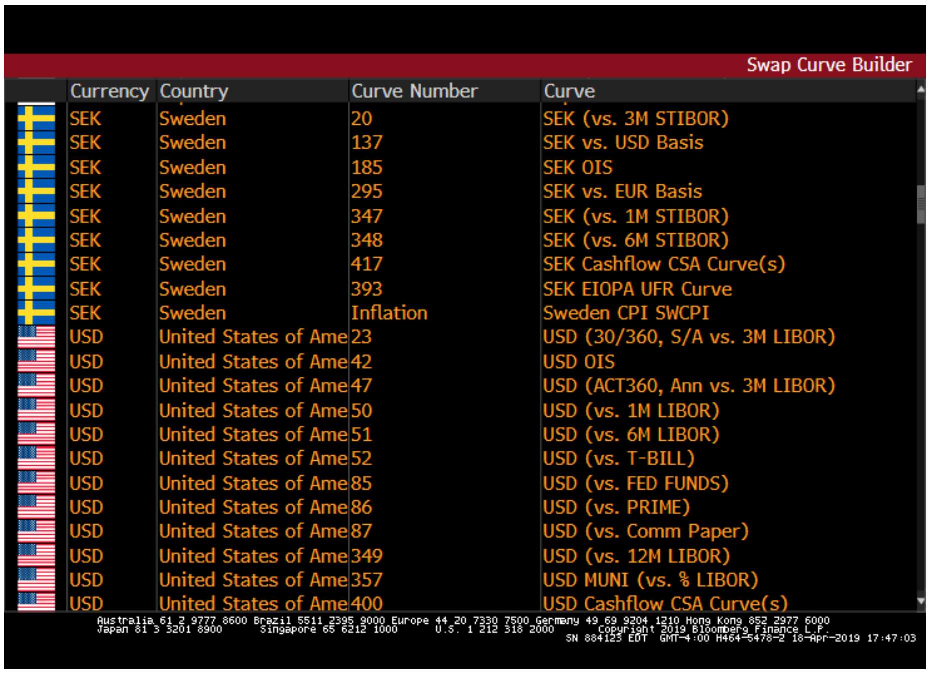
LIBOR curves are typically estimated from quotes on three different types of instrument:

1. LIBOR deposit rates for short maturities.
2. Rates calculated from Eurodollar futures for intermediate maturities.
3. Swap rates for longer maturities.

For this problem set, you need to download rates for February 28, 2019 (corresponding settlement date: March 4, 2019) by using Bloomberg's swap-curve-builder command, `ICVS`:[12]

- Run the command `ICVS` to display the curve-selection screen (see Figure 1).
- Pick curve number 23, `USD (30/360, S/A vs. 3M LIBOR)`, to display the curve-construction screen (see Figure 2), showing yields for today's date plus a graph of Bloomberg's fitted yield curve (see Bloomberg, 2021, pp. 3–6).
- Change the trade date (at the top-right of the screen) to 2/28/19 and press `Enter` to change to the desired date.
- Download the displayed data to Excel (at higher precision than shown on the screen) by selecting `Actions` → `Export to Excel`. The columns you care about are:
  - `Term`: Number of months/years or actual date.
  - `Unit`: Months (MO), years (YR) or maturity date (ACTDATE)
  - `Bid` and `Ask`: Bid and ask yields on the security.
  - `Rate Type`: The type of interest rate:
    - `Cash Rates`: Deposit rate
    - `Contiguous Futures`: Rate calculated from Eurodollar futures contract.
    - `Swap Rates`: LIBOR swap rate.
  - `Daycount`: The day-count convention being used.
  - `Freq`: Compounding/payment frequency.

[12] Further information about `ICVS`, and about the methods used by Bloomberg for interpolating yield curves, can be found in Bloomberg (2021). While Bloomberg allows the user to customize exactly which quotes to use at each maturity, we shall just use the default settings.

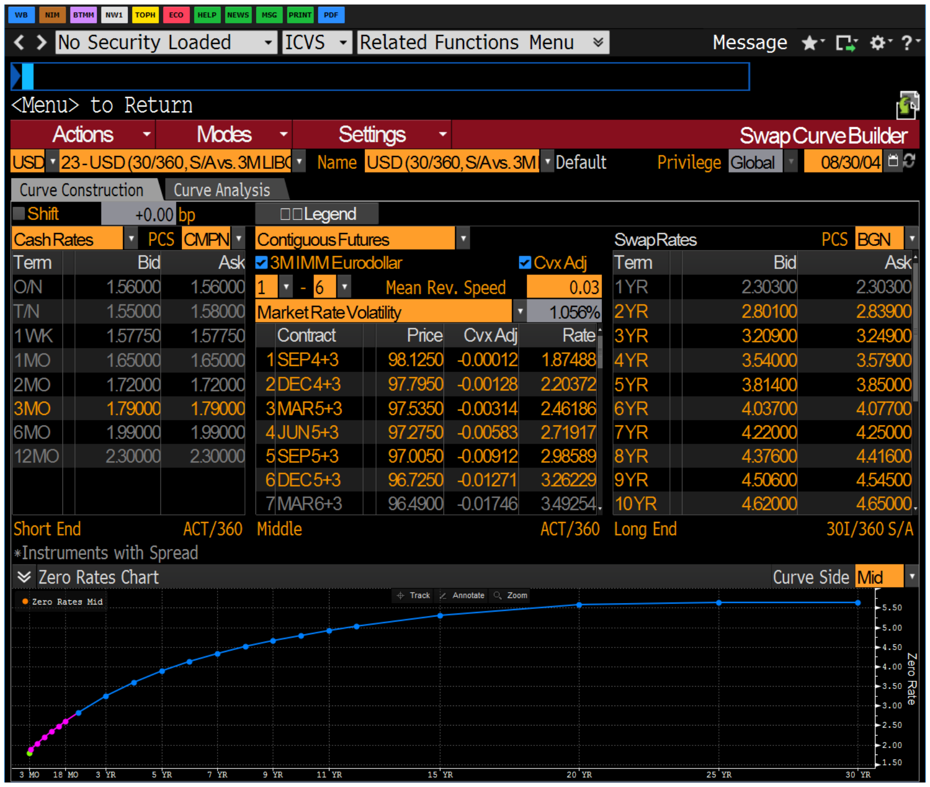


Swap Curve Builder

	Currency	Country	Curve Number	Curve
	SEK	Sweden	20	SEK (vs. 3M STIBOR)
	SEK	Sweden	137	SEK vs. USD Basis
	SEK	Sweden	185	SEK OIS
	SEK	Sweden	295	SEK vs. EUR Basis
	SEK	Sweden	347	SEK (vs. 1M STIBOR)
	SEK	Sweden	348	SEK (vs. 6M STIBOR)
	SEK	Sweden	417	SEK Cashflow CSA Curve(s)
	SEK	Sweden	393	SEK EIOPA UFR Curve
	SEK	Sweden	Inflation	Sweden CPI SWCPI
	USD	United States of Ame	23	USD (30/360, S/A vs. 3M LIBOR)
	USD	United States of Ame	42	USD OIS
	USD	United States of Ame	47	USD (ACT360, Ann vs. 3M LIBOR)
	USD	United States of Ame	50	USD (vs. 1M LIBOR)
	USD	United States of Ame	51	USD (vs. 6M LIBOR)
	USD	United States of Ame	52	USD (vs. T-BILL)
	USD	United States of Ame	85	USD (vs. FED FUNDS)
	USD	United States of Ame	86	USD (vs. PRIME)
	USD	United States of Ame	87	USD (vs. Comm Paper)
	USD	United States of Ame	349	USD (vs. 12M LIBOR)
	USD	United States of Ame	357	USD MUNI (vs. % LIBOR)
	USD	United States of Ame	400	USD Cashflow CSA Curve(s)

Australia 61 2 9777 8600 Brazil 5511 2395 9000 Europe 44 20 7330 7500 Germany 49 69 9204 1210 Hong Kong 852 2977 6000
Japan 81 3 3201 8900 Singapore 65 6212 1000 U.S. 1 212 318 2000 Copyright 2019 Bloomberg Finance L.P.
SN 884123 EDT GMT-4:00 H464-5478-2 18-Apr-2019 17:47:03

Figure 1: Bloomberg ICVS curve-selection screen



<Menu> to Return

Swap Curve Builder

USD 23-USD (30/360, S/A vs. 3M LIBO Name USD (30/360, S/A vs. 3M Default Privilege Global 08/30/04

Curve Construction | Curve Analysis

Shift +0.00 bp ☐☐Legend

Cash Rates PCS CMPN

Term	Bid	Ask
O/N	1.56000	1.56000
T/N	1.55000	1.58000
1 WK	1.57750	1.57750
1MO	1.65000	1.65000
2MO	1.72000	1.72000
3MO	1.79000	1.79000
6MO	1.99000	1.99000
12MO	2.30000	2.30000

Short End ACT/360

Contiguous Futures

☑ 3M IMM Eurodollar ☑ Cvx Adj

1 - 6 Mean Rev. Speed 0.03

Market Rate Volatility 1.056%

	Contract	Price	Cvx Adj	Rate
1	SEP4+3	98.1250	-0.00012	1.87488
2	DEC4+3	97.7950	-0.00128	2.20372
3	MAR5+3	97.5350	-0.00314	2.46186
4	JUN5+3	97.2750	-0.00583	2.71917
5	SEP5+3	97.0050	-0.00912	2.98589
6	DEC5+3	96.7250	-0.01271	3.26229
7	MAR6+3	96.4900	-0.01746	3.49254

Middle ACT/360

Swap Rates PCS BGN

Term	Bid	Ask
1YR	2.30300	2.30300
2YR	2.80100	2.83900
3YR	3.20900	3.24900
4YR	3.54000	3.57900
5YR	3.81400	3.85000
6YR	4.03700	4.07700
7YR	4.22000	4.25000
8YR	4.37600	4.41600
9YR	4.50600	4.54500
10YR	4.62000	4.65000

Long End 30I/360 S/A

*Instruments with Spread

Zero Rates Chart Curve Side Mid

Figure 2: Bloomberg ICVS curve-construction screen