M6 Excel Topics

Topics in Insurance, Risk, and Finance [1]

Professor Benjamin Avanzi

10 July 2024

 [1] References: (**EE19?**) and (**AE23?**) | → latest slides

Why Excel?

- Excel is great to spread data and calculations in a tabular form, and have a visual overview.
- Most financial modelling is done in Excel (at least initially).
- In actuarial work, many more advanced codes (in R, Python, C++, C#, VBA, ...) often starts with someone playing around in Excel, and once proof of concept is approved, this moves to proper coding.
- It is assessed in CM2-B (!).

1. Introduction and Assumed Knowledge
   - Why Excel?
   - Issues with Excel
   - Assumed knowledge
   - Some keyboard shortcuts

Issues with Excel

Excel is notoriously problematic in certain areas:

- Lack of transparency - one can't see the formulas unless you click in a cell; alternatively, you can't see the formulas and understand where the numbers come from unless you are in Excel (a problem for reports, presentations etc).
- Mistakes can be tiny but have huge consequences in the end (the job of Excel auditor actually exists!).
- Lack of good documentation capability (as opposed to code); this makes collaboration and audit difficult, and creates an operational risk (e.g. builder leaves).

- Lack of rigour in the construction of a model (input, assumptions, intermediary calculations, output).
- Can't handle (seriously) large data sets.
- Lack of good and flexible data cleaning and manipulation capabilities.
- Sometimes code is a lot easier (e.g. flip a vector around, sum over a diagonal, . . .).

I know there are counter arguments for all of those, but this presupposes you know what the solutions are. You'll learn some of those here!

Assumed knowledge

See `prerequisite knowledge on the website`. Some extracts:

- Autofill: Chapter 3, p. 105-116, and p. 298-301
- Named ranges and constants: Chapter 7, page 312-332
- Absolute and mixed cell references ($): Chapter 7, pages 332-342
- New Excel 2019 functions (IFS, MAXIFS, MINIFS): Chapter 8, pages 381-398
- Formula Auditing: Chapter 9, p. 436-439
- Paste special (incl, e.g. `Transpose`): Chapter 11, pages 518-530

Page references are for (**EE19?**), see `link here`.

Also, see tab `Assumed Knowledge` in the `module 6 spreadsheet`.

Some keyboard shortcuts

There are many keyboard shortcuts that can help you work with Excel efficiently. Some (but not all) useful shortcuts are:

- ctrl + shift + arrow key: Extend the selection of cells to the last non-blank cell towards the specified direction
- ctrl + PageDown/PageUp: Move to the next/previous sheet
- shift + F11: Insert new worksheet
- F4: Cycles through combinations of absolute and relative references

See Microsoft - `Keyboard shortcuts in Excel` for list of all keyboard shortcuts.

2 Data Wrangling and Exploratory Data Analysis

- Data Wrangling
- Pivot tables
- Pivot charts

Data Wrangling

- Sometimes we need to work on data from an external source.
- The data could be messy to work with.
  - There could be non-printable characters such as tabs (`\t`), new lines (`\n`) in the original data.
  - We may want to process the strings to retrieve information.
  - Data formats are inconsistent.
- It is important to clean our data before analysis!

Some data cleaning steps

- Editing Texts
  - `CLEAN()` removes any non-printable characters. Check `List of ASCII values` for ASCII codes for non-printable characters.
  - We can use `SUBSTITUE()` to replace characters.
- Merging and splitting columns with functions like `LEFT()`, `RIGHT()` and `MID()` and `SEARCH()`. Wildcard characters `?` and `*` are also useful.
- Removing duplicate rows: `Data > Data Tools > Remove Duplicates`.
- See `Data wrangling` tab in the `module 6 spreadsheet`.
- There are much more functions and steps can be used to clean your data - see `Microsoft - Top ten ways to clean your data`.

2 Data Wrangling and Exploratory Data Analysis

- Data Wrangling
- Pivot tables
- Pivot charts

Pivot tables

- Pivot tables are often considered as very difficult to master, but they are not that difficult to start with.
- Example (in `module 6 spreadsheet`): FIFA WWC
  - Insert / Pivot Table.
  - See how you can use variables as filters, rows, or columns. Move them around.
  - See how columns can display other things than `Sum`, such as `Count`, `Average`, `Max`, `Min`, or `Product`
- Note there are recommended Pivot Tables (automated recommendation within Excel); in the case of FIFA WWC it is not very helpful.
- Reference: Chapter 15 of (**EE19?**).

2 Data Wrangling and Exploratory Data Analysis

- Data Wrangling
- Pivot tables
- Pivot charts

Pivot charts

- A pivot chart can be created from the Pivot Table but also directly from the data.
- A major difference with start charts is that you it will be somewhat "interactive" - there will be buttons you can use to alter the chart.
- Example (in `module 6 spreadsheet`): FIFA WWC
  - Insert / Pivot Chart;
  - In the example I changed the style of graph to "Combo" to allow for the two different scales;
  - I also included a "slicer" (click on chart / insert slicer), in order to easily filter by squad.
- Reference: Chapter 15 of (**EE19?**).

3 Dynamic Arrays

- Spilling
- Think in vectors
- The # sign
- New formulas
- Lookup functions

Spilling

- One advantage of programs like R are the easy use and manipulation of vectors.
- Excel can do similar things, and the vectors are called arrays. This is new (post Office 365), and is a bit of a game changer.
- Before Office 365, Excel was incapable of depositing results beyond just 1 cell. This is called "spilling".
- Note Excell will need the required space to spill.
- Main reference is (**AE23?**) - we'll only introduce this here.

Example

- Here we introduce array formulas.
  - If you calculate the sum of an array you'll get a single number.
  - The result of an array formula (such as `LEN()`), when you input an array, will give you an array.
- See `module 6 spreadsheet`:
  - `LEN()` gives an array.
  - You could then get the sum without having to put the array anywhere: `SUM(LEN(B3:B6))`.
  - Note that when I wrote the above, it became `SUM(LEN(B3#))` automatically - more on that later.
- `SUM(LEN(B3:B6))` will work in any version of Excel because it requires only one cell to output, but not `LEN(B3:B6)` as it requires several cells ("spilling").
- Note you can spill named ranges, too!

3 Dynamic Arrays

- Spilling
- Think in vectors
- The # sign
- New formulas
- Lookup functions

Think in vectors

- Once you understand you can create vectors and either display or manipulate them, Excel becomes a lot more powerful.
- You can also use arrays in arguments of known formulas such as `VLOOKUP()`,
  - For instance `VLOOKUP(.,.,{2,5})` will return value from the 2nd and 5th columns row-wise.
  - If you use `VLOOKUP(.,.,{2;5})` (with the semicolon) they will display columnwise.
  - See examples in `module 6 spreadsheet`.

3 Dynamic Arrays

- Spilling
- Think in vectors
- The # sign
- New formulas
- Lookup functions

The # sign

- The # sign when added to a reference to a cell where a dynamic array is written will duplicate that array (and spilled results).
- It is shorter, but also it will dynamically change the size of the array
  - This can be desired or not.
  - See example in `module 6 spreadsheet`.

3 Dynamic Arrays

- Spilling
- Think in vectors
- The # sign
- New formulas
- Lookup functions

New formulas

- There are a number of new formulas which were available in coding languages like `R` for a long time, which can be useful, and which are now available in Excel
  - for instance `SEQUENCE()`, `UNIQUE()`, `FILTER()`, `RANDARRAY()`...
- Some of those are exemplifed in `module 6 spreadsheet`.
- You are encouraged to browse through. They really bring data handling in Excel a little closer to coded languages such as `R`.

3 Dynamic Arrays

- Spilling
- Think in vectors
- The # sign
- New formulas
- Lookup functions

Lookup functions

- Lookup functions are essential for Excel users.
- Useful when you need to search within a single row or column to find a corresponding value in the same position in a second row or column.
- Multiple lookup functions are available.
- All of them works with arrays of lookup values!

VLOOKUP()

- Available for all versions of Excel, so it is reliable when sharing spreadsheets.
- Some drawbacks of VLOOKUP():
  - Lookup Column Must Be the First Column
  - Can be confusing when specifying the return column
- HLOOKUP() is the horizontal version of VLOOKUP().
- Check this video to see how it works.

XLOOKUP()

- An upgraded version for VLOOKUP() released in 2020 - not available in Excel 2016 and Excel 2019.
  - Robust lookup function: You can search any direction.
  - Can return multiple arrays as well.
  - The lookup array does not have to be the first column/row.
  - Different search modes available: from first to last, last to first, etc.
  - An `if not found` argument to allow combinations with functions like IFERROR, IFNA.
- Watch `this video` to see how it works.

MATCH(), XMATCH()

- Both functions work similarly by finding the index of your lookup value within the lookup array.
- XMATCH() is more robust by providing new match mode and search mode.
- By combining with INDEX(), it returns the value instead of the index inside the lookup array.
- Check the lookup tab in module 6 spreadsheet for comparisons between these lookup functions. Reference: Murray, 2022 (Chapter 7, 11)

Dynamic References

- Sometimes we have to work with references that changes position or size over time.
- For instance, we need to adjust our range of references for each development period when modelling with chain ladder.
- This is tedious to do manually if we have a large dataset!
- Dynamic Referencing will be useful here.

4 Dynamic References

- OFFSET()
- INDIRECT()
- ADDRESS()

4 Dynamic References

- OFFSET()
- INDIRECT()
- ADDRESS()

4 Dynamic References

- OFFSET()
- INDIRECT()
- ADDRESS()

ADDRESS()

Reference: Murray, 2022 (Chapter 11)

5 General etiquette and tools

- Etiquette
- Principles

Etiquette

You should build your spreadsheet with (at least) the following **objectives** in mind:

1. so as to minimise chances or error (accuracy);
2. so as to minimise unnecessary calculations (efficiency);
3. so as to make the structure as clear as possible (transparency);
4. so as to make updates, changes and extensions possible and easy (extendibility);
5. so as to allow someone else to use it easily (user friendliness);
6. so as to allow someone else to verify it easily (auditability).

Those are, of course, interconnected. We could add more (for instance, automation of data input via an API, automation of communication objects such as charts, etc. . .).

5 General etiquette and tools

- Etiquette
- Principles

Principles

There is no single way to achieve the objectives above, but there are a number of **principles** that one could list, and that will contribute to meeting those objectives:

- Have a separate tab that collects all your assumptions that are valid for the whole spreadsheet (1, 3, 4, 5, 6).
  - Include some explanations about the source/justification of those assumptions.
  - Give your assumptions names (for instance, the technical rate of interest to calculate life insurance could be called `techint` or similar, for ease of later reference, and to make formulas more easily readable).
- Also include your data sets in separate tabs (1, 3, 4, 5, 6).
  - Name your data (including columns and/or rows if possible; e.g. `FIFA WWC` in the spreadsheet; see also "Named ranges and constants": Chapter 7, page 312-332).

- Consider colouring / contouring input and output differently (3, 5)
  - This is not always advisable, but if you have a large model with relatively few input (e.g. purchase price and interest rate for a property mortgage schedule) and/or few outputs (e.g. NPV) then this achieves many of the objectives.
- Use named ranges and variables as much as possible (1, 3, 5, 6), unless the variable is going to be used only once.
- Use more advanced formulas if shorter, and avoid too much nesting (1, 3, 4, 5, 6).

6 Tools for auditing

- Dependents and precendents
- Show formulas

Dependents and precendents

Use of dependents / precedents, e.g.:

1. go to tab `Dependents - Precedents - PPCI`;
2. click on one of the outstanding loss projected amounts;
3. make sure the formula tool tab is live;
4. click on the "trace precedents" sequentially.

This will highlight dependence of each cell to previous cells, sequentially, with arrows.

This is useful for

- understanding the structure of a spreadsheet;
- check formulas (audit);
- debug issues.

(Formula Auditing: Chapter 9, p. 436-439)

6 Tools for auditing

- Dependents and precendents
- Show formulas

Show formulas

- The "Formulas" tool tab should have a "Show formula" tile. This will replace all numeric values by formulas.
  - This is helpful to check what numbers are hard coded, and which are results of calculations. Together with dependents, it helps seeing if everything is as dynamic as it should.
- If you want a formula to be shown all the time start with an apostrophe `'`, and the formula will show as text.
- Note also the `FORMULATEXT()` formula which is a dynamic array formula requiring spilling (see later section "Dynamic Arrays"!).

(Formula Auditing: Chapter 9, p. 436-439)

Next steps

- All of (**AE23?**) is relevant, but take it as a cook book for the assignment. You can go as far as you wish.
- Chapters 16, 17, 18, and 19 of (**EE19?**) are out of scope.
- However, macros and VBA (which is Chapter 19) are essential components of Excel
  - I strongly encourage you to get started. Start by recording a macro, then play around with the code.
  - VBA allows more efficient calculations via compiled code, and is a powerful addition to Excel.
- Fun fact: the 2021 `Excel World Champion` is an actuary: Andrew Ngai, now Director at Taylor Fry.