FOSE1025 — Scientific Computing

Week 7 Lecture 1: Cleaning Data

Diego Mollá

FOSE1025 2023H1

Abstract

In this lecture we will focus in the step of data cleaning, with particular emphasis on text data. We will look at various tools that both Excel and MATLAB provide to help cleaning raw data and process text: convert types, parse text, split text, filter data.

Update April 4, 2023

Contents

Reading

- These notes
- Readings listed in iLearn — Week 7

1 Cleaning Text Data in Excel

Text as Unstructured Data

- Much of the information you find is input in text.
- People can understand text very easily . . .
- . . . but not machines!
- Text is often called a kind of *unstructured data.*
- Excel and MATLAB can help find structure from text.

Watch these LinkedIn Learning Videos for Excel

Macquarie University students have access to free videos from LinkedIn Learning. To access these, login using your student credentials. Watch these videos:

- Use Text Functions (5min):

  https://www.linkedin.com/learning/excel-2016-cleaning-up-your-data/use-text-functions

- Use CONCAT and TEXTJOIN functions to combine data (9min):

  https://www.linkedin.com/learning/excel-2016-cleaning-up-your-data/use-the-concat-and-textjoin-functions-to-combine-data

- Combine data via concatenation (9min)

  https://www.linkedin.com/learning/excel-2016-cleaning-up-your-data/combine-data-via-concatenation

- Split data into columns (5min)

  https://www.linkedin.com/learning/excel-2016-cleaning-up-your-data/split-data-into-columns-with-the-text-to-columns-feature

- Using Flash Fill (9min)

  https://www.linkedin.com/learning/excel-2016-cleaning-up-your-data/use-flash-fill-for-faster-combining-and-splitting

Some Useful Text Functions

CH-05.xlsx From https://www.linkedin.com/learning/excel-2016-cleaning-up-your-data/use-text-functions

Name	Description
LOWER	Converts all text to lowercase
PROPER	Capitalizes only letters than start the entry or follow a space or punctuation
UPPER	Converts all text to uppercase
REPLACE	Replaces characters within text, based on content, not on character position
SUBSTITUTE	Replaces characters within text, based on character position, not on content
REPT	Repeats text a given number of times
LEFT	Returns the leftmost characters from a text value
MID	Returns a specific number of characters from a text string starting at the position you specify
RIGHT	Returns the rightmost characters from a text value
FIND	Finds one text value within another (case-sensitive)
SEARCH	Finds one text value within another (not case-sensitive)
EXACT	Checks to see if two text values are identical
LEN	Returns the number of characters in a text string
TEXT	Formats a number and converts it to text
VALUE	Converts a text argument to a number
CLEAN	Removes all nonprintable characters from text
TRIM	Removes spaces from text
CONCATENATE	Joins several text items into a cell (on older Excel versions)
CONCAT	Joins several text items into a cell (on newer Excel versions)
DOLLAR	Converts a number to text, using the $ (dollar) currency format
FIXED	Formats a number as text with a fixed number of decimals
TEXTJOIN	Joins several text items into a cell using a delimiter

These are only some of the functions that can work with text. At the lecture, we will apply some of them to the file CH-05.xlsx.

The functions shown in bold are the ones that you will need to know for in-class test 3.

Concatenating Text

Several ways to concatenate text:

- Using the `&` operator

```
=A1 & " " & B1
```

- CONCAT (in Excel versions from 2016, Mobile, Web)

```
=CONCAT("Stream population for ", A2, " ",
          A3, " is ", A4, "/mile.")
=CONCAT(B2:C8)
```
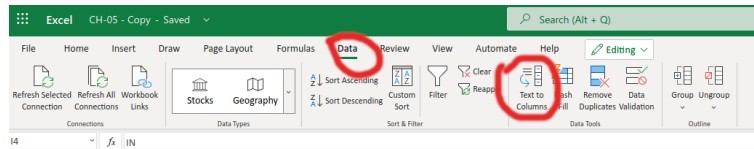
- CONCATENATE (in older Excel versions)

- TEXTJOIN (in Excel versions from 2019, Web — joins text using a text delimiter)

```
=TEXTJOIN(" ",TRUE, "The", "sun", "will", "come",
          "up", "tomorrow.")
=TEXTJOIN(", ", TRUE, A2:A8)
```

Parsing Text Using Text to Columns Feature

- Some columns have complex text that needs to be parsed.

- Excel can parse the text of a column and split it into several columns.
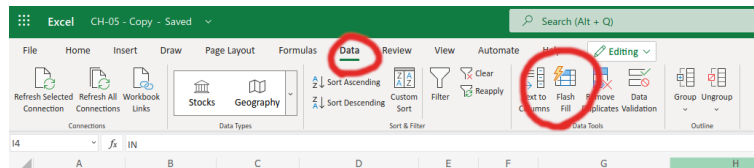
- On Excel Online, look at the "Data" tab

CH-05.xlsx; watch the video *https://www.linkedin.com/learning/excel-2016-cleaning-up-your-data/split-data-into-columns-with-the-text-to-columns-feature*

The Magic of Flash Fill

- Flash Fill is one of Excel's most powerful and least known features.

- Uses AI techniques to try to predict how you want to parse the text.

- Looks like magic, but sometimes might not work for your task.

CH-05.xlsx; watch the video *https://www.linkedin.com/learning/excel-2016-cleaning-up-your-data/use-flash-fill-for-faster-combining-and-splitting*

2   Cleaning Data in MATLAB

MATLAB's Column Types
https://au.mathworks.com/help/matlab/data-types.html

- All values of a MATLAB table column are of the same type.

- Common types in MATLAB are:

  Numeric

  - `double` — This is the default numerical type. It is what is called *double-precision floating point*.
  - There are other types that you can use to represent integers (with or without sign) and other numerical types.

  Text

  - `string` — Starting in MATLAB's version R2016b, this is the preferred way to store text. It's called "string arrays".

– `char` — Available in all MATLAB versions but not recommended from MATLAB version R2016b. It's called "character arrays".

Dates and Time

  – `datetime` — MATLAB stores both dates and times using this format.
  – We will look at MATLAB's dates and times in a subsequent lecture.

Categorical

  – Use this type (instead of, say, `string`), if you know that the column has a finite set of possible values.
  – For example, `C = categorical({'R','G','B','B','G','B'})` creates a categorical array with six elements that belong to the categories `R`, `G`, or `B`.

Examining the Type of a Table Column
File: mlb_players.csv

- MATLAB's `summary` function gives a summary of a table.

- It reports various information, including the types of all of its columns.

Try This

1. Generate a live script that imports the file `mlb_players.csv` and stores the generated table in the variable `mlb_players`.

2. Add this command to the live script (without ";" at the end):

```
summary(mlb_players)
```

The output should look like this:

```
Variables:

Name: 1035x1 string

Team: 1035x1 categorical

    Values:

        ANA             35
        ARZ             28
        ATL             37
        BAL             35
        BOS             36
        CHC             36
        CIN             36
        CLE             35
        COL             35
        CWS             33
        DET             37
        FLA             32
```

```
                HOU                34
                KC                 35
                LA                 33
                MIN                33
                MLW                35
                NYM                38
                NYY                32
                OAK                37
                PHI                36
                PIT                35
                SD                 33
                SEA                34
                SF                 34
                STL                32
                TB                 33
                TEX                35
                TOR                34
                WAS                36
                NumMissing          1

    Position: 1035x1 categorical

        Values:

                Catcher                 76
                Designated Hitter       18
                First Baseman           55
                Outfielder             194
                Relief Pitcher         315
                Second Baseman          58
                Shortstop               52
                Starting Pitcher       221
                Third Baseman           45
                NumMissing               1

    Height_inches_: 1035x1 double

        Values:

                Min                67
                Median             74
                Max                83
                NumMissing          1

    Weight_lbs_: 1035x1 double

        Values:

                Min               150
                Median            200
                Max               290
                NumMissing          2

    Age: 1035x1 double

        Values:

                Min               20.9
                Median            27.925
                Max               48.52
                NumMissing          1
```
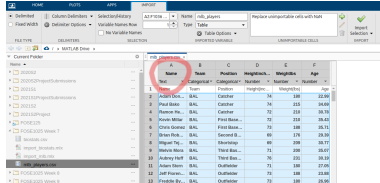
In this output you can see the type of each column. For columns with categorical data, it will list the number of values in each category. And for columns with numerical data, it will show the minimum, median,

and maximum value.

Text or string?

Depending on what part of MATLAB you use, it may say that a column type is "Text" or "string". Both are the same!

If using the text import tool you may see that the column type is "text".

If using `summary` you may see that the same column type is "string".

```
>> summary(mlb_players)

Variables:

    Name: 1035×1 string

    Team: 1035×1 categorical
```

Setting the Type in a Table Column

File: `mlb_players.csv`

- A common problem with MATLAB (and Excel) is that the default settings when reading a CSV file might not be correct.

  - For example, by default, `readtable` may store text as a character array, not a string array.

- If we use MATLAB's import tool we can specify the data type (see lecture week 6).

  - Check how the generated script defines options to the `readtable` function.

- We can also change the data type *after* the table has been created.

```
mlb.Team = categorical(mlb.Team);
mlb.Name = string(mlb.Name);
```

In the example `mlb.Team = categorical(mlb.Team);`:

- `mlb.Team` indicates the column with name `Team` which is stored in the table with name `mlb`.

- `categorical(mlb.Team)` returns a column vector where the type of the elements is categorical.

- `mlb.Team = ...` means that the `Team` column of the table `mlb` is assigned the result on the right-hand side of the = (which, in our case, is the contents of the same column that has been converted to the categorical type).

Filtering Data in an Array

- MATLAB can identify what values meet a particular condition.

- For example, to find what elements in an array "ages" are larger than 10:

```
>> ages = [1 2 5 34 2 32];
>> ages > 10
ans =
  1x6 logical array
  0   0   0   1   0   1
```

- The result is a filter represented as a *logical array*: each element is either 0 ("false") or 1 ("true").

- We can now select all elements whose corresponding logical array indicates true.

```
>> ages(ages > 10)
ans =
  34   32
```

The MATLAB examples above (and below) have been executed in MATLAB's Command Window. The prompt >> indicates that we have typed a MATLAB command. Then, any text that is not preceded with >> indicates MATLAB's response.

Filtering Data in a Table
File trees.csv

- The same process can be used to select rows whose columns meet a particular condition.

```
>> trees.Girth_in_ > 15
ans =
  31x1 logical array
0 0 0 ... 1 1 1
>> wide_trees = trees(trees.Girth_in_ > 15, :)
```

- We can combine multiple filters by using Boolean operators.

- Can you tell what's the output of the following?

```
>> trees = readtable("trees.csv");
>> filtera = trees.Girth_in_ > 10;
>> filterb = trees.Girth_in_ < 15;
>> filterc = trees.Height_ft_ > 70;
>> result = trees(filtera & filterb | filterc, :)
```

Take-home Messages

Excel

- Fixing problems from manual data input.

- Importing text.

- Text to columns feature.

- Flash Fill.

MATLAB

- Changing data types.

- Text functions.

- Filtering data.

What's Next

Assessments this week
There are no assessments due this week.

Next weeks

- *2 weeks without classes, time to catch up!*

- The project will be released during the break

  - 30% of the unit assessment
  - Submit by Wed 17 May

- Week 8 lecture: Transforming Data

  - *Tuesday 25 April: Anzac Day*
  - *The Tuesday lecture and SGTA will move to another day.*
  - *We will send an announcement with the details.*