TeraSoft 鈦思科技 The Power of Driver for the Best!!

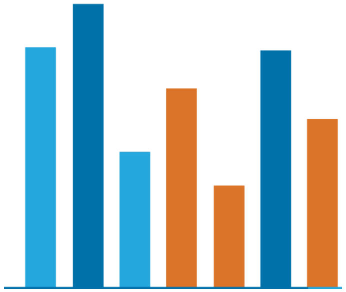
MATLAB for Data Analysis

Application Engineer

Jeffrey Liu

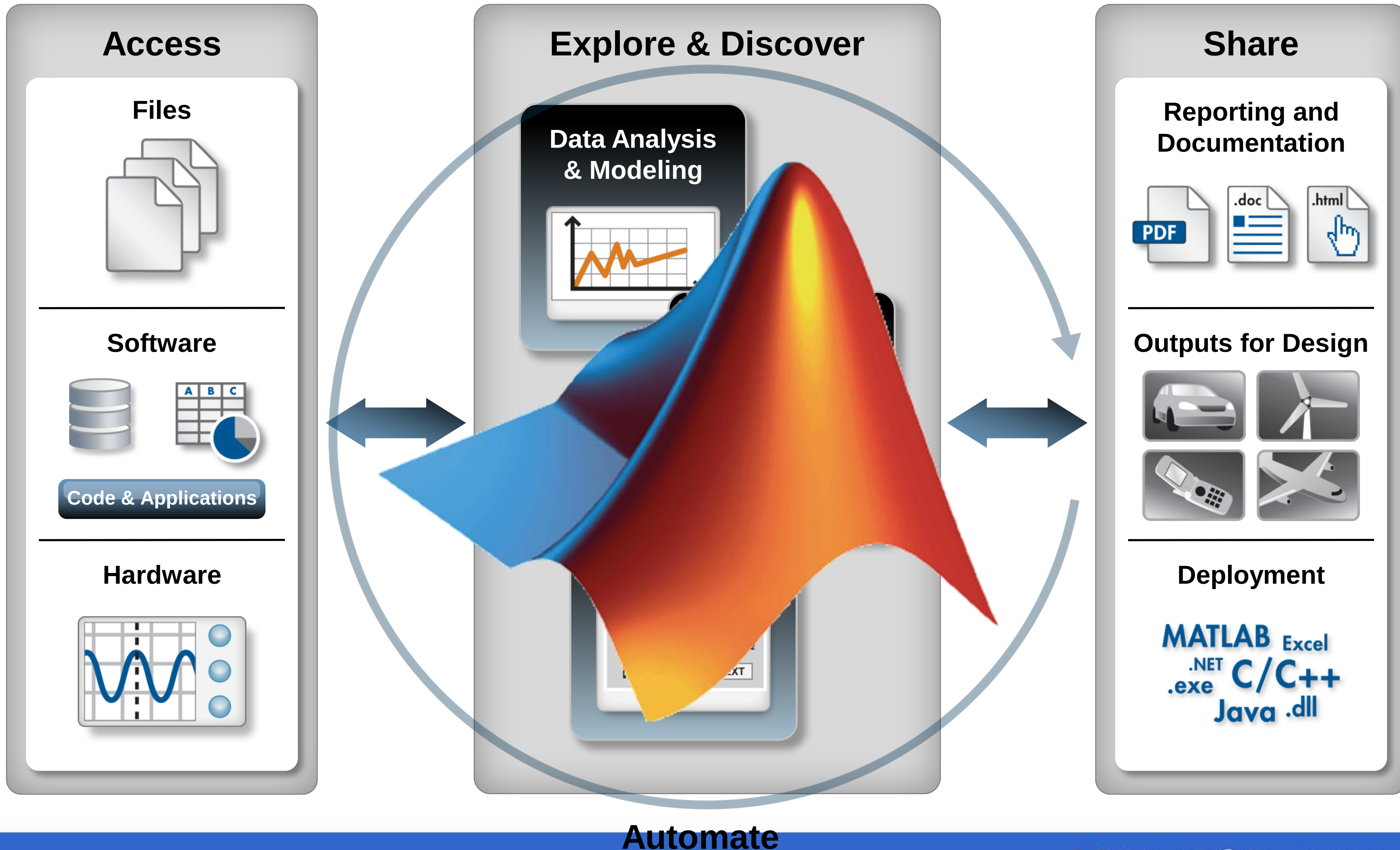
Outline

- Importing Data to MATLAB
  - Create data in MATLAB
  - Working with data files
- Basic data analysis
  - Descriptive statistics
  - Split-apply workflow
- Data Types
  - Dataset, table
  - data types



- Fit data to models
  - Curve fitting toolbox
  - Statistics models
- Data pre-processing
  - Dealing with missing value
  - Locating data
  - Merging data
- Publish your code

Technical Computing Workflow



Random Number generation

- MATLAB
  - rand
    - uniform distribution generation between 0~1
  - randi
    - Uniformly distributed pseudorandom integers
  - randn
    - Standard normal distribution generator
- Statistics Toolbox
- Random seed
  - rng
  - rng shuffle

MATLAB®

`rand`
`randi`
`randn`

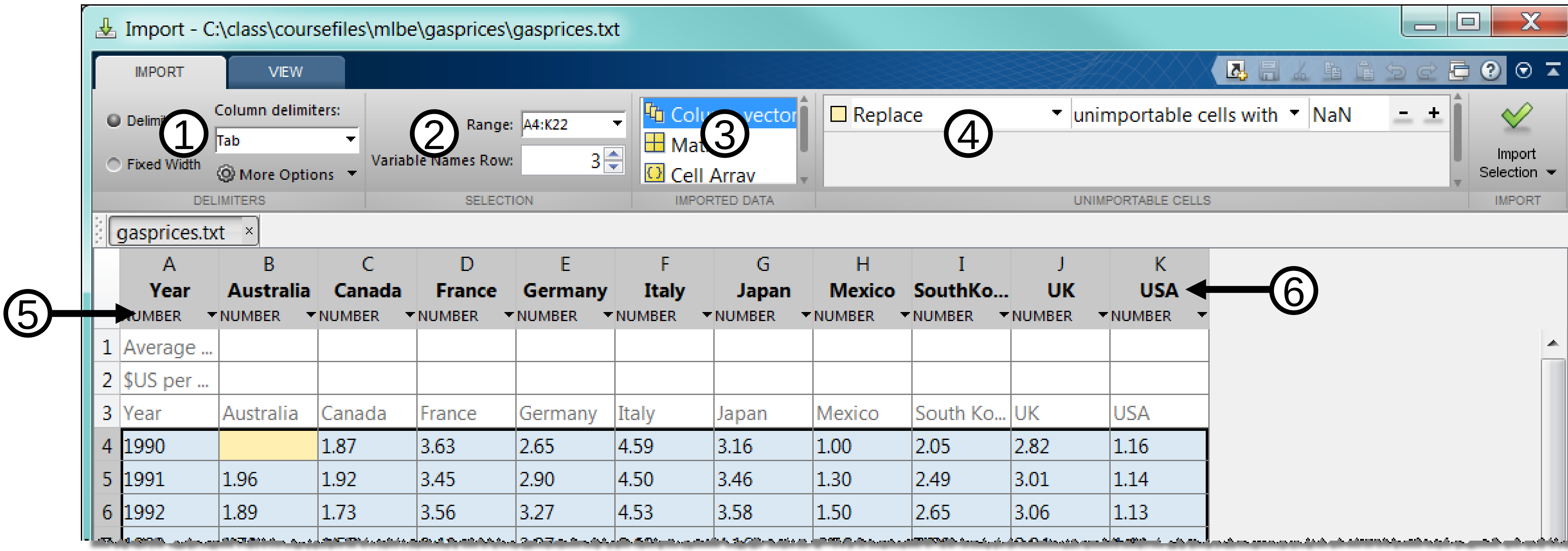
Statistics Toolbox™

betarnd
binornd
chi2rnd
ncx2rnd
exprnd
evrnd
frnd
ncfrnd
gamrnd
...

`random`

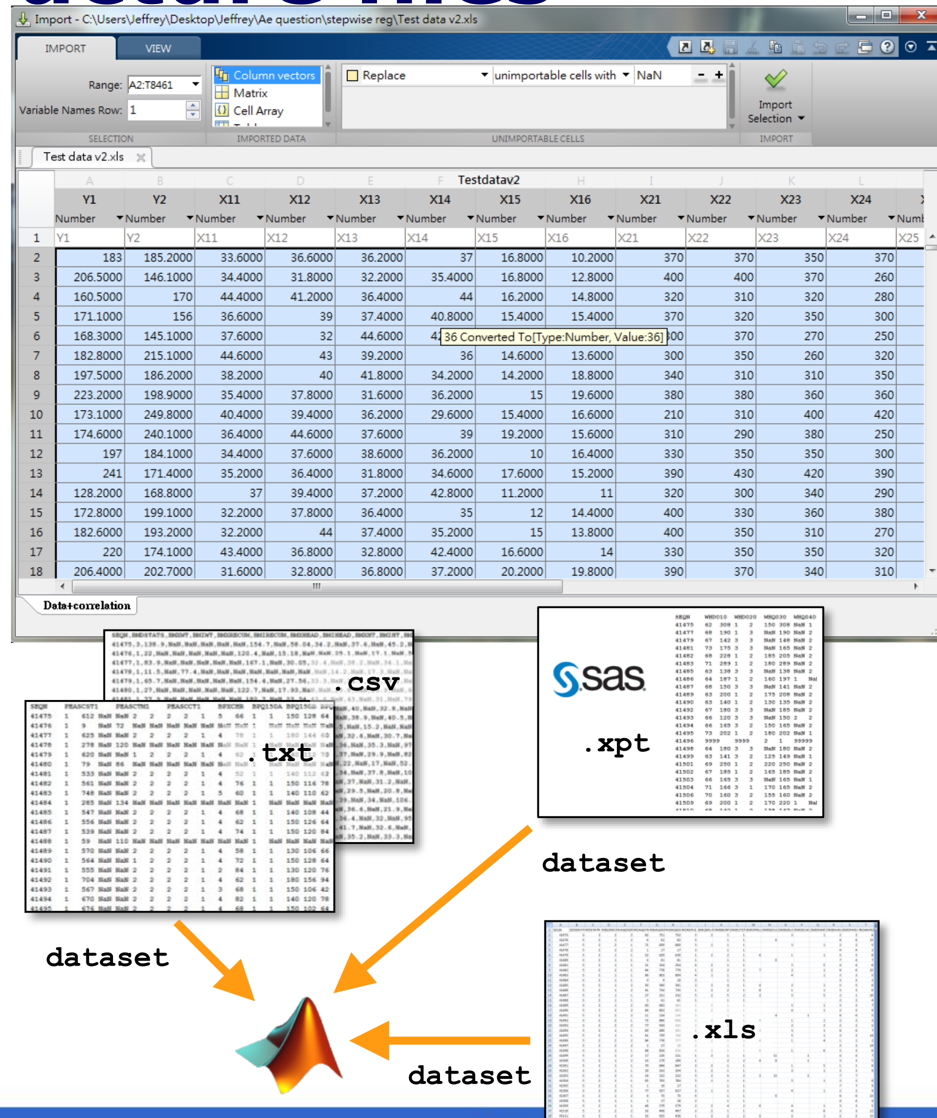
Interactively Importing Text Files and Spreadsheets

- The Import Tool attempts to interpret the contents of spreadsheets and delimited text files, and automatically set several options.



Reading Fixed structure files

- Files
  - Excel, text, csv, or binary
  - .xpt in SAS
  - .mat
  - Multimedia, scientific
  - Web, XML
- Command
  - load
  - csvread
  - xlsread
  - textread
  - textscan



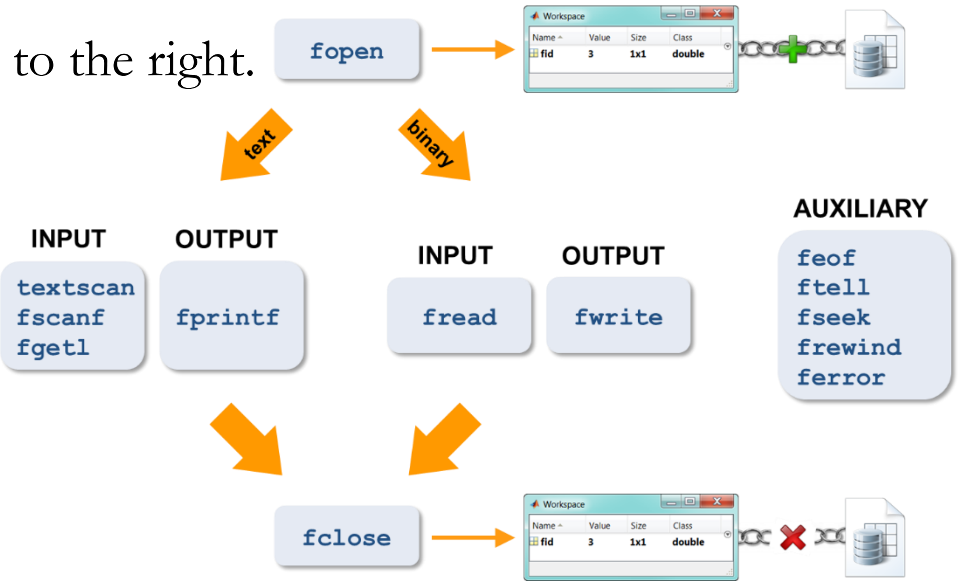
Low-Level File I/O

1. Open the file

>> fid = fopen('gasprices.csv','r');

2. Perform the file operations

Functions for reading and writing text and binary files are shown in the diagram to the right.



3. Close the file

>> fid = fclose(fid);

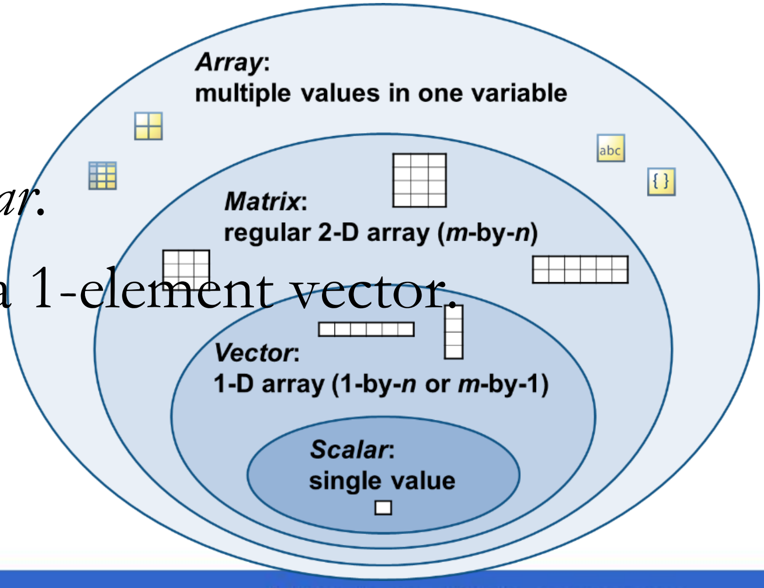
Vectors, Matrices, and Arrays

- A table of numbers with *m* rows and *n* columns is referred to as an *m*-by-*n* matrix.

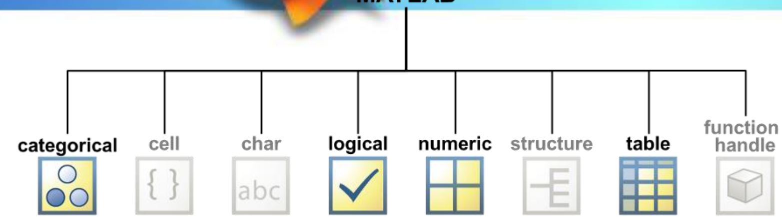
- A *vector* is a single row or a single column of numbers. It is therefore a special case of a matrix, where either *m* or *n* is equal to 1.

- A single number is referred to as a *scalar*.

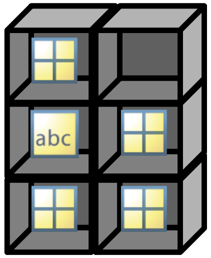
  It is a 1-by-1 matrix and, equivalently, a 1-element vector.



Cell array



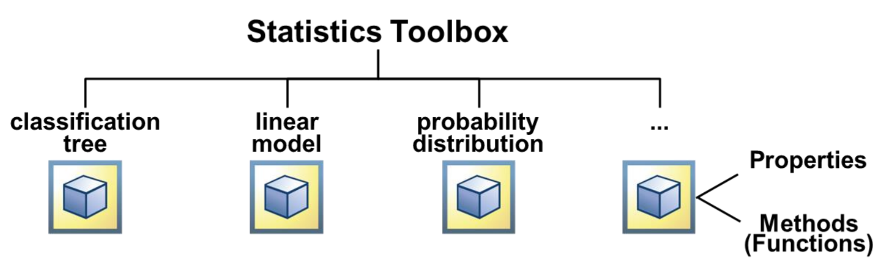
- Cell arrays are an easy way to assemble data of dissimilar types and sizes into a single "container of containers."

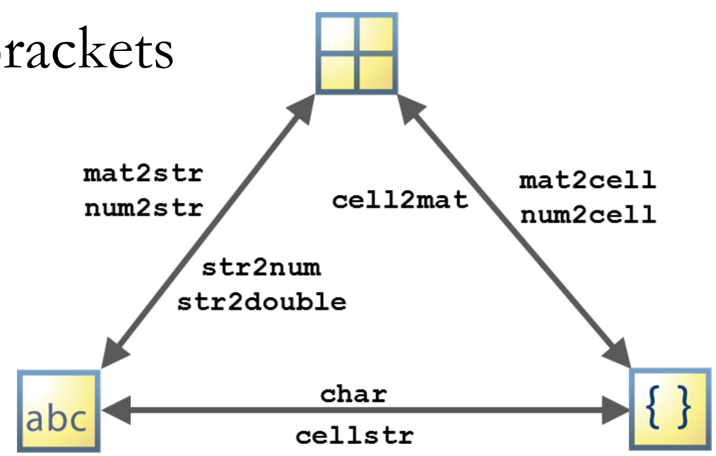


- They are typically used to store strings of different length.

- To concatenate elements into a cell array,

  use curly braces ({}) instead of square brackets





What Is a Table ?

- Many data sets conform to a particular tabular arrangement with the following properties:
  - Each column of data has the same type (text, numeric, logical (T/F), etc.).
  - Different columns, however, can have different types.
  - Each column typically has a unique name.
  - Each column has the same number of rows

	A	B	C	D	E	F	G	H	I	J	K	L
1	Date	AAPL	CSCO	DELL	EBAY	HPQ	IBM	INTC	MSFT	ORCL	YHOO	DJIA
2	1/3/2008	0.0005	0.0079	-0.0283	0.0107	0.0000	0.0020	-0.0272	0.0042	0.0272	0.0050	0.0010
3	1/4/2008	-0.0794	-0.0238	-0.0708	-0.0480	-0.0576	-0.0366	-0.0845	-0.0284	-0.0479	-0.0289	-0.0198
4	1/7/2008	-0.0135	0.0004	-0.0383	-0.0279	-0.0330	-0.0107	0.0092	0.0067	0.0099	0.0009	0.0021
5	1/8/2008	-0.0366	-0.0272	-0.0248	-0.0142	-0.0488	-0.0249	-0.0275	-0.0341	-0.0507	-0.0249	-0.0188
6	1/9/2008	0.0465	0.0314	0.0091	-0.0047	0.0285	0.0074	0.0218	0.0292	0.0215	-0.0022	0.0115
7	1/10/2008	-0.0077	0.0000	0.0005	0.0163	0.0200	0.0162	-0.0093	-0.0032	0.0032	0.0656	0.0092
8	1/11/2008	-0.0304	-0.0142	-0.0086	-0.0227	-0.0075	-0.0228	-0.0247	-0.0123	-0.0271	-0.0308	-0.0194
9	1/14/2008	0.0347	0.0161	0.0327	-0.0153	0.0248	0.0525	[illegible]	[illegible]	[illegible]	[illegible]	0.0135
10	1/15/2008	-0.0560	-0.0169	-0.0260	-0.0448	-0.0237	-0.0107	[illegible]	[illegible]	[illegible]	[illegible]	-0.0219
11	1/16/2008	-0.0572	-0.0275	-0.0106	0.0050	-0.0159	-0.0020	[illegible]	[illegible]	[illegible]	[illegible]	-0.0028
12	1/17/2008	0.0078	-0.0331	0.0087	0.0011	-0.0307	-0.0052	[illegible]	[illegible]	[illegible]	[illegible]	-0.0249
13	1/18/2008	0.0029	-0.0012	0.0105	0.0074	0.0173	0.0225	[illegible]	[illegible]	[illegible]	[illegible]	-0.0049
14	1/22/2008	-0.0361	-0.0343	-0.0402	-0.0396	-0.0238	-0.0213	[illegible]	[illegible]	[illegible]	[illegible]	-0.0106
15	1/23/2008	-0.1126	0.0232	-0.0020	0.0609	0.0109	0.0471	0.0710	-0.0009	0.0246	0.0075	0.0247
16	1/24/2008	-0.0253	0.0440	0.0426	-0.0627	0.0386	0.0076	0.0349	0.0405	0.0000	0.0806	0.0088
17	1/25/2008	-0.0421	-0.0369	-0.0501	-0.0130	-0.0260	-0.0226	-0.0339	-0.0094	-0.0161	0.0115	-0.0139
18	1/28/2008	0.0000	-0.0041	0.0144	0.0015	0.0014	0.0044	0.0144	-0.0067	-0.0010	-0.0543	0.0144
19	1/29/2008	0.0117	-0.0017	0.0103	-0.0283	-0.0096	0.0106	0.0103	-0.0037	-0.0094	0.0014	0.0078
20	1/30/2008	0.0049	0.0124	-0.0063	0.0069	0.0037	-0.0043	0.0092	-0.0123	0.0099	-0.0884	-0.0030
21	1/31/2008	0.0238	0.0057	-0.0193	0.0222	0.0041	0.0137	0.0196	0.0123	0.0137	0.0068	0.0165
22	2/1/2008	-0.0120	0.0178	[illegible]	[illegible]	[illegible]	[illegible]	0.0313	-0.0682	0.0063	0.3918	0.0073
23	2/4/2008	-0.0158	-0.0459	[illegible]	[illegible]	[illegible]	[illegible]	0.0265	-0.0086	-0.0235	0.0329	-0.0085
24	2/5/2008	-0.0175	-0.0238	[illegible]	[illegible]	[illegible]	[illegible]	0.0523	-0.0378	-0.0477	-0.0120	-0.0297

Each row is a set of observations

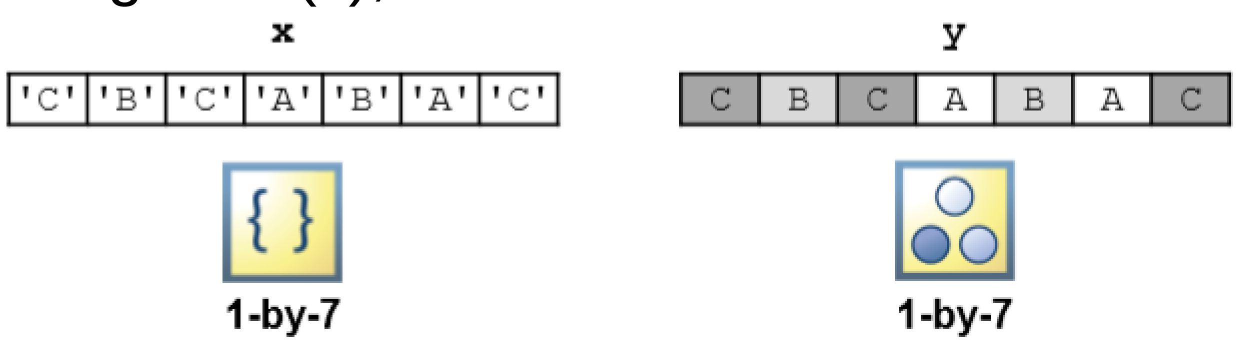
Each column is a named variable

Categorical array

- When text labels intended to represent finite set of possibilities, cell array in unnecessary, Instead, you can use *categorical* array

>> x = {‘C’, ‘B’, ‘C’, ‘A’, ‘B’, ‘A’, ‘C’};

>> y = categorical(x);



Extracting Data from a Table

- Index into variables in a table using dot (.) notation to reference the variables by name:

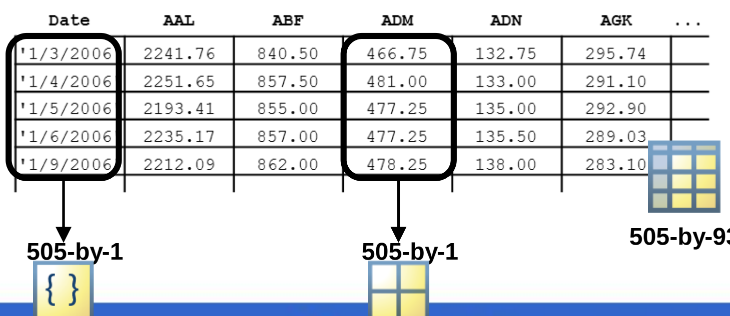
```
>> dates = stockPrices.Date;
>> admPrices = stockPrices.ADM;
```

- You can also use curly braces ({}) to index into variables in a table. You can index numerically

```
>> twoFin = stockPrices{:,[4 5]};
```

or by name

```
>> twoFin = stockPrices{:,{'ADM','ADN'}};
```



Merging Data from two tables

- Use "join" to merge "dataset" variables
  - Import files with dataset array

```
>> Variables = dataset('XLSFile', 'filename');
```

  - Merge two datasets

```
>> New = join(A,B, 'Keys', 'date', 'Type', 'outer',
'MergeKeys',true);
```

- Use "innerjoin", "outerjoin" to combine "table" variables
  - Import files with table array

```
>> Variables = readrable(date, data, 'var_name');
```

  - Merge two tables

✓ innerjoin, outerjoin

Exercise 1

- Merging two tables by time in excel file.

T1

A	B	C
3	9	4

T2

A	B	C
1	7	2
5	8	3

[]

A	B	C
3	9	4
1	7	2
5	8	3

innerjoin

A	B	C
3	9	4
1	7	2
5	8	3

T3

A	X	Y	Z
5	2	2	9
3	1	7	8
2	6	1	5

A	B	C	X	Y	Z
3	9	4	1	7	8
5	8	3	2	2	9

outerjoin

A	B	C
3	9	4
1	7	2
5	8	3

A	X	Y	Z
5	2	2	9
3	1	7	8
2	6	1	5

A	B	C	X	Y	Z
3	9	4	1	7	8
1	7	2			
5	8	3	2	2	9
2			6	1	5

Dealing with missing value

- Possible strategies

Ignore Delete Replace

Avoiding NaNs in Calculations

- Several functions are designed to ignore NaNs in calculations

```
nancov      nanmedian   nansum
nanmax      nanmin      nanvar
nanmean     nanstd
```

- if a column contains all NaNs, ignoring them will result in applying the desired function to an empty array.

3	2	4	NaN
1	NaN	2	1
NaN	NaN	6	1

↓ nanmean

2	2	4	1

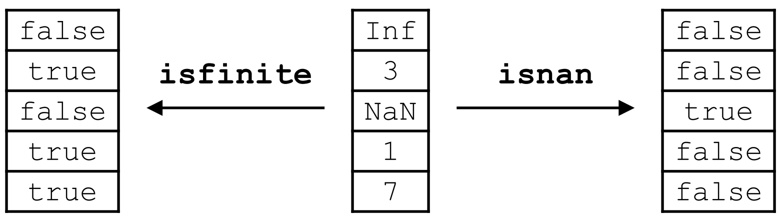
3	NaN	4	NaN
1	NaN	2	1
NaN	NaN	6	1

↓ nanmean

2	NaN	4	1

Locating Missing(Other) Values

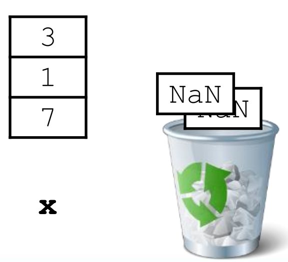
- MATLAB provides numerous “is*” functions that take an array as input and return a logical output that signifies if the input has a certain characteristic.

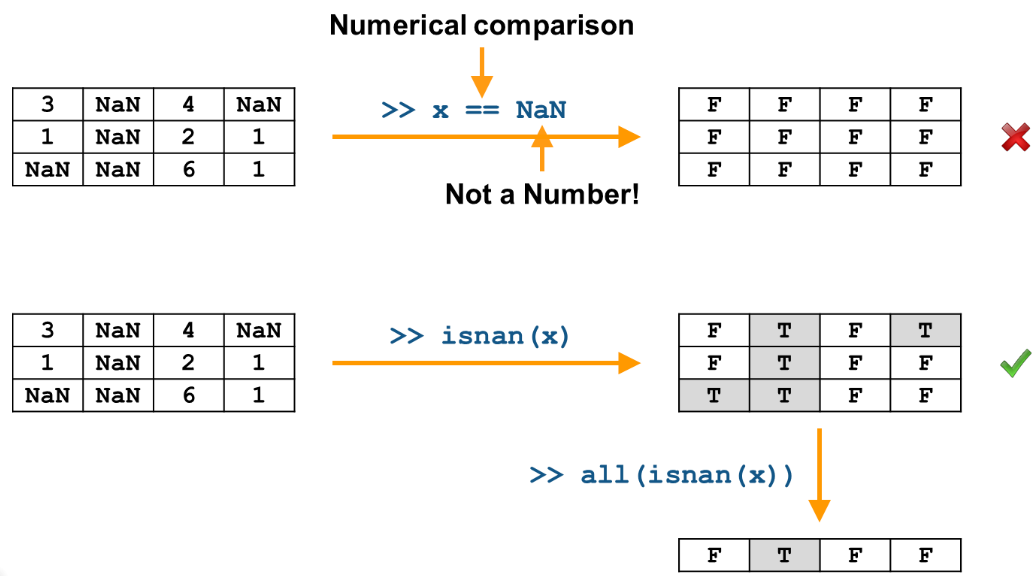


- You can use logical indexing to remove elements from an array.

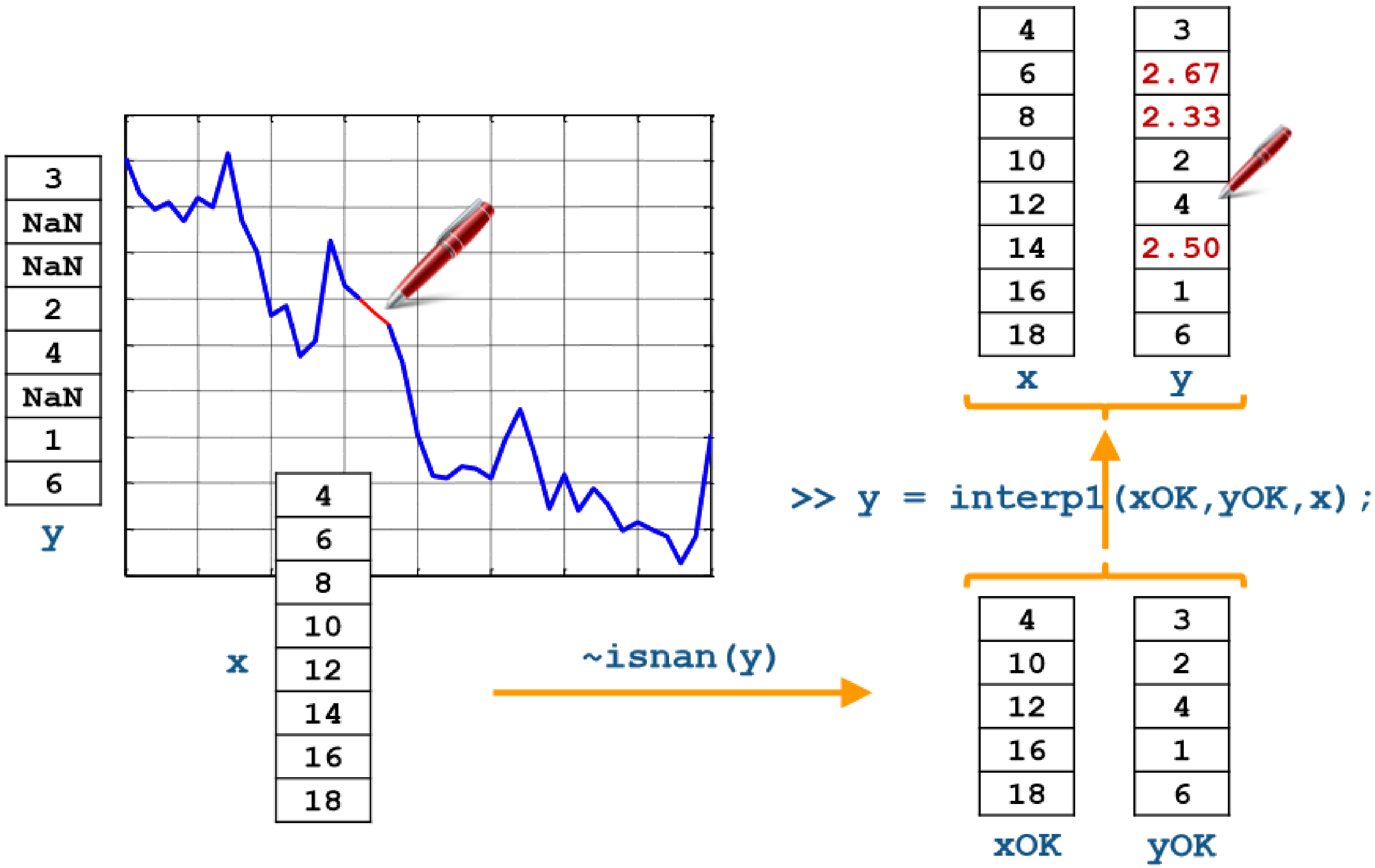
  >> idx = isnan(x);

  >> x(idx) = [];





Replacing Missing Values in Matrices



Test Your Knowledge

1. (Select all that apply.) If x is a (numeric) vector, which commands will remove all the NaNs from x?

   A. x = x(isnan(x));

   B. x = x(x ~= NaN);

   C. x(isnan(x)) = [];

   D. x(x == NaN) = [];

   E. x = x(isfinite(x));

2. Suppose x is a 6-by-3 (numeric) matrix of ones. Three values in the first column are missing (NaN), as are six values in the third column. What will >> nanmean(x) return?

   A. 1 B. [0.5 1 0]

   C. [0.5 1 NaN]

   D. [1 1 NaN]

   E. [1 1 0]

   F. An error message.

Calculate the data by group

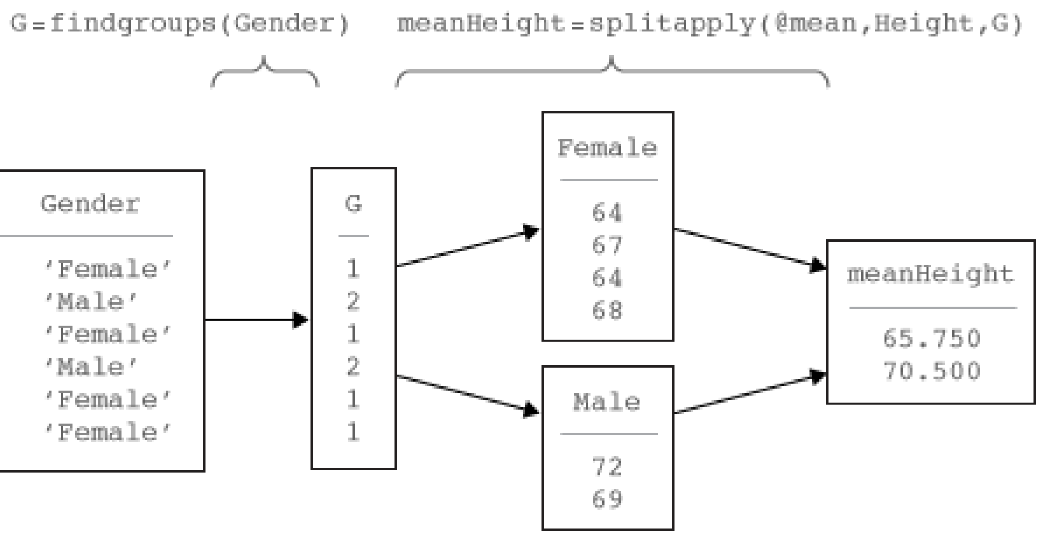
- Function 'grpstats' : Summary statistics organized by group
  - Syntax : statarray = grpstats(tbl,groupvar,whichstats)

    tbl : data in table or dataset array

    groupvar : Column name for grouping in tbl

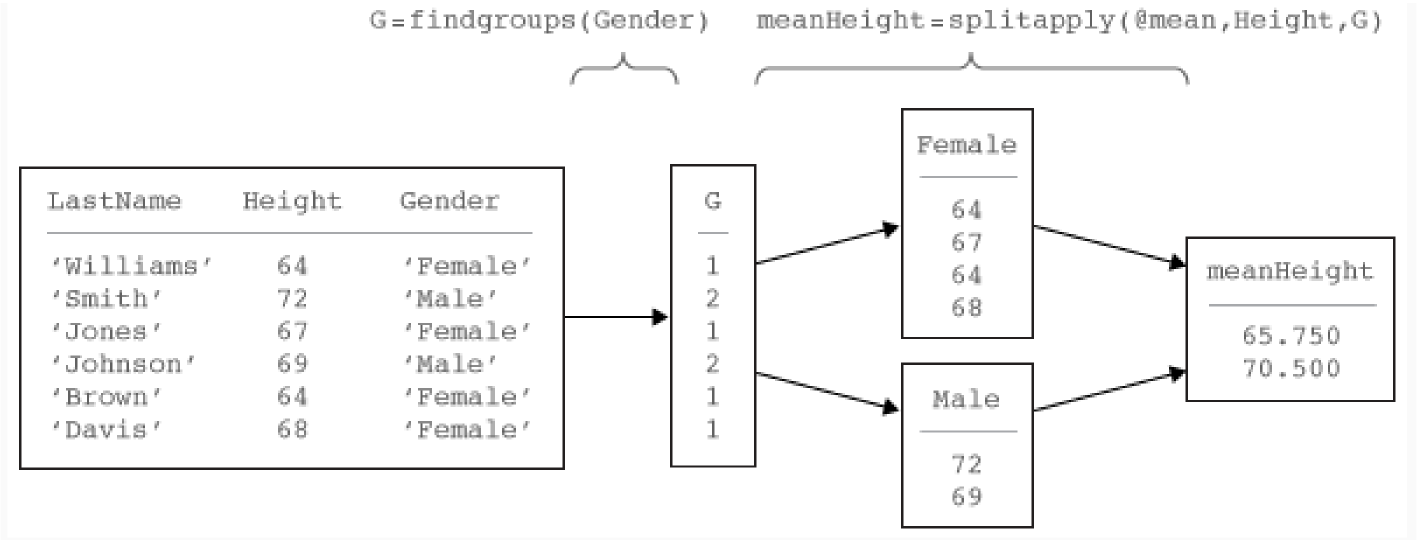
    whichstats : Types of summary statistics (numel, std, max, sum…)

- New Split-apply workflow



Exercise 2

- 在bank-full.csv中依照三種類別(工作、婚姻、購買與否)區分客戶，並計算類別中的平均收入



LinearModel Variables

- Given data in vectors `x` (predictor) and `y` (response), you can perform a least-squares linear regression using `fitlm`:

```
>> linmodel = fitlm(x,y)
```

- You can visually inspect the fit using the `plot` method:

```
>> plot(linmodel)
```

- Evaluate the fitted model at chosen predictor values using the predict method:

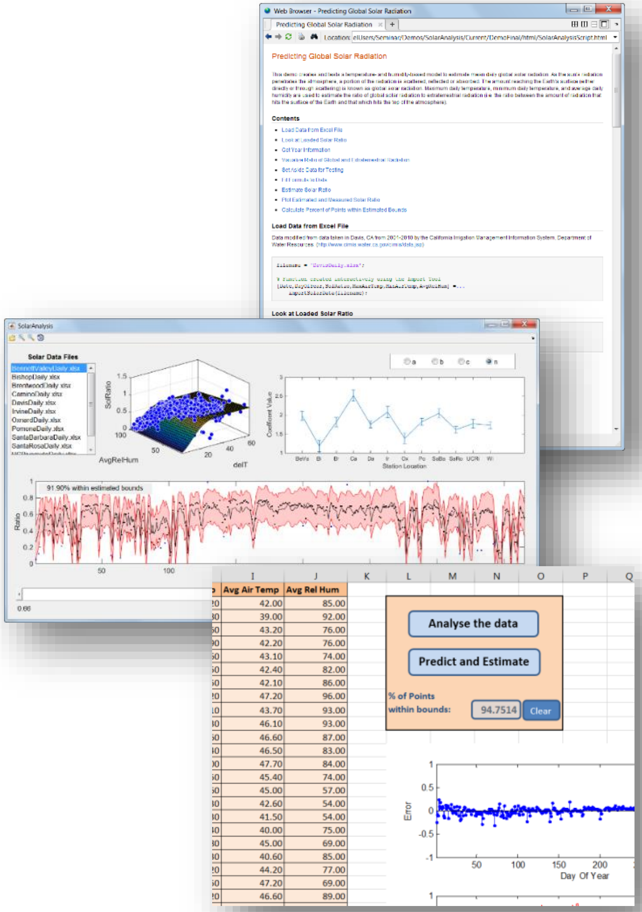
```
>> predict(linmodel,xnew)
```

- This value is one of the properties of a LinearModel variable:

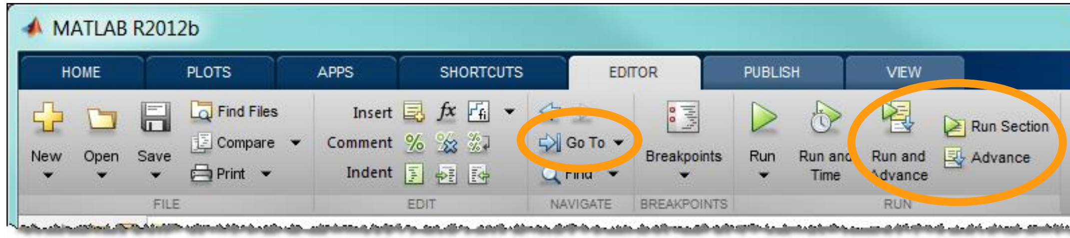
```
>> linmodel.Rsquared
```

Sharing Results from MATLAB

- Automatically generate reports
- Create and package applications
- Deploy to other environments



Code Sections



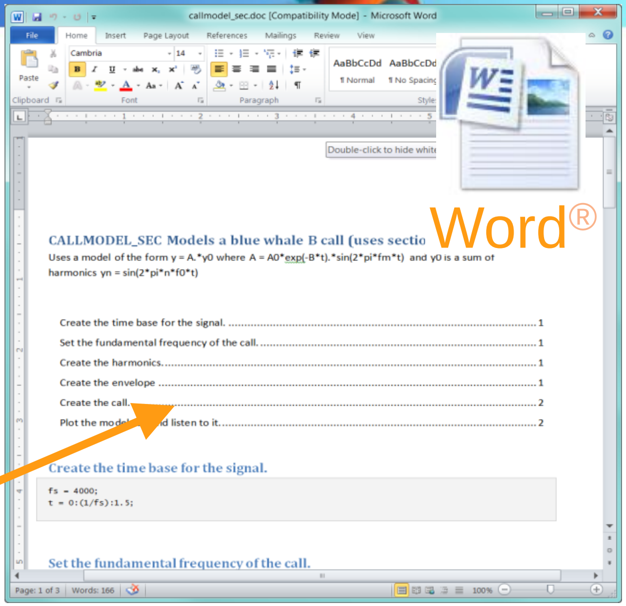
- Create the time base for the signal
- Set the fundamental frequency of the call
- Create the harmonics
- Create the envelope
- Create the call
- Plot the model call and listen to it

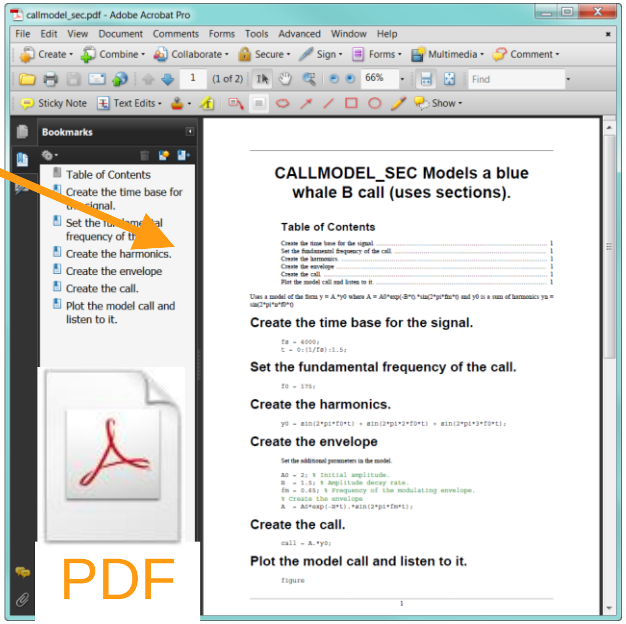
%%

```
%% Set the fundamental frequency of the call.
f0 = 175;

%% Create the harmonics.
y0 = sin(2*pi*f0*t) + sin(2*pi*2*f0*t) + sin(2*pi*3

%% Create the envelope
% Set the additional parameters in the model.
A0 = 2; % Initial amplitude.
B  = 1.5; % Amplitude decay rate.
fm = 0.65; % Frequency of the modulating envelope.
% Create the envelope
A  = A0*exp(-B*t).*sin(2*pi*fm*t);

%% Create the call.
call = A.*y0;
```

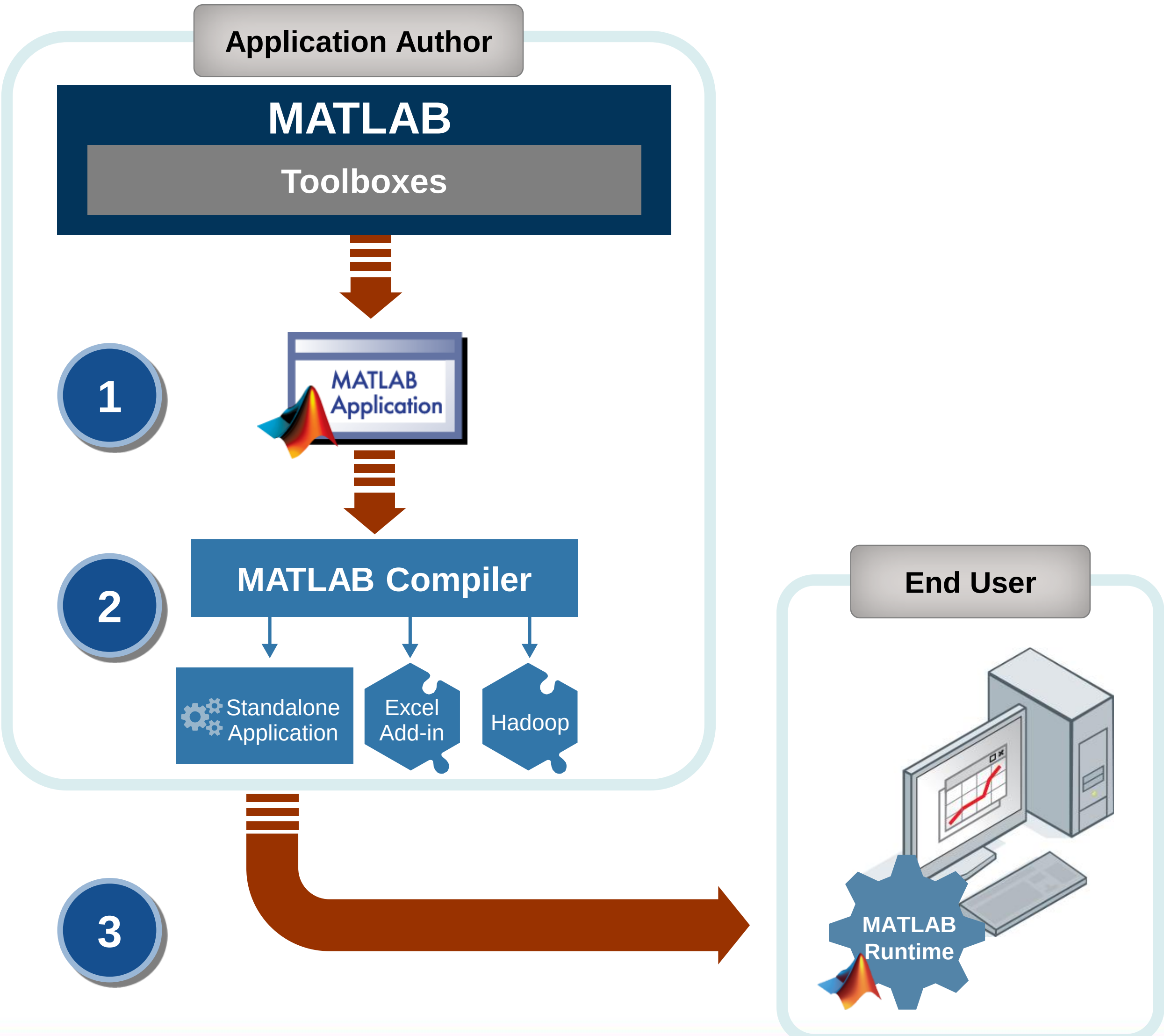
Publishing Code

```
%% Set the fundamental frequency of the call.
f0 = 175;

%% Create the harmonics.
y0 = sin(2*pi*f0*t) + sin(2*pi*2*f0*t) + sin(2*pi*3

%% Create the envelope
% Set the additional parameters in the model.
A0 = 2; % Initial amplitude.
B  = 1.5; % Amplitude decay rate.
fm = 0.65; % Frequency of the modulating envelope.
% Create the envelope
A  = A0*exp(-B*t).*sin(2*pi*fm*t);

%% Create the call.
call = A.*y0;
```





Sharing Standalone Applications



Thanks for your attention !