MATH 3341: Introduction to Scientific Computing Lab

Libao Jin

University of Wyoming

February 26, 2020

Lab 05: Formatting Output

Format of Output in Command Window

"What You See Is Not What You Get"

When MATLAB performs calculations, it displays the result at a default precision. But actually, the value stored is actually in double precision.

Formatting Output using `format` command

We can specify the format of numerical output using `format type` in the command window. Here are format types available in MATLAB:

- `short/long`: Scaled fixed point format with 5 digits/15 digits for double and 7 digits for single.
- `shorte/longe`: Floating point format with 5 digits/15 digits for double and 7 digits for single.
- `shortg/longg`: Best of fixed or floating point format with 5 digits/15 digits for double and 7 digits for single.
- `shorteng/longeng`: Engineering format that has at least 5 digits/exactly 16 digits and a power that is a multiple of three
- `hex/+/rat/compact/loose`.

Example

```
%% Example 1
disp('Example 1 -- Output pi in different formats');
format
format('loose');    pi
format('compact');  pi
format('short');    pi
format('long');     pi
format('shorte');   pi
format('longe');    pi
format('shortg');   pi
format('longg');    pi
format('shorteng'); pi
format('longeng');  pi
format('rat');      pi
```

Example: Output

```
Example 1 -- Output pi in different formats

ans = 3.1416

ans = 3.1416
ans = 3.1416
ans = 3.14159265358793
ans = 3.1416e+00
ans = 3.14159265358793e+00
ans = 3.1416
ans = 3.14159265358979
ans = 3.1416e+000
ans = 3.14159265358979e+000
ans = 355/113
```

Example: More Elegant Way of Doing Repeating Tasks

Can we make the code more elegant?

Yes! Using a for-loop!

```
disp('Example 2 -- Using a for-loop to output pi in differe
% Using a cell-type variable to hold array of strings
types = {'loose', 'compact', 'short', 'long', ...
         'shorte', 'longe', 'shortg', 'longg', ...
         'shorteng', 'longeng', 'rat'};
format
for i = 1:length(types)
    format(types{i}); pi
end
```

Write Formatted Data

fprintf and sprintf

- `fprintf`: Write formatted data to text file.
  - `fprintf(fileID,formatSpec,A1,...,An)`
  - `fprintf(formatSpec,A1,...,An)`
- `sprintf`: Write formatted data to string or character vector.
  - `str = sprintf(formatSpec,A1,...,An)`

Conversion Specifications

Conversion specifications, which include a % character, a conversion character (such as d, i, o, u, x, f, e, g, c, or s), and optional flags, width, and precision fields.

Escape characters

Character	Details
`\b`	Backspace
`\f`	Form feed
`\n`	New line
`\r`	Carriage return
`\t`	Horizontal tab
`''`	Single quotation mark
`%%`	Percent character
`\\`	Backslash
`\xN`	Hexadecimal number N
`\N`	Octal number N%

For more details: `doc sprintf`

Conversion	Details
`%d` or `%i`	Base 10
`%u`	Base 10
`%o`	Base 8 (octal)
`%x`	Base 16 (hexadecimal), lowercase letters a–f
`%X`	Same as %x, uppercase letters A–F
`%f`	Fixed-point notation
`%e`	Exponential notation, such as 3.141593e+00
`%E`	Same as %e, but uppercase, such as 3.141593E+00
`%g`	The more compact of %e or %f, with no trailing zeros
`%G`	The more compact of %E or %f, with no trailing zeros
`%c`	Single character
`%s`	Character vector or string array.

For more details: `doc sprintf`

Flags	Details
-	Left-justify.
+	Right-justify text.
	Insert a space before the value.
0	Pad to field width with zeros before the value.
#	Modify selected numeric conversions: - For %o, %x, or %X, print 0, 0x, or 0X prefix. - For %f, %e, or %E, print decimal point even when precision is 0 - For %g or %G, do not remove trailing zeros or decimal point.

Examples: sprintf

```
formatSpec = string('The current time is: %d:%d %s');
A1 = 11;
A2 = 20;
A3 = 'a.m.';
str = sprintf(formatSpec,A1,A2,A3)
```

Examples: fprintf

MATLAB code:

```
x = [0:.2:1]';
A = [x exp(x)];
A_size = size(A);

file_handle = fopen('exp.txt','w');
fprintf(file_handle,'%6s %12s\n','x','exp(x)');
for i = 1:A_size(1)
    fprintf(file_handle, '%6.2f %12.8f\n', ...
                         A(i, 1), A(i, 2));
end
% fprintf(file_handle,'%6.2f %12.8f\n', A');
fclose(file_handle);
```

Examples: fprintf

Output:

```
     x          exp(x)
0.00    1.00000000
0.20    1.22140276
0.40    1.49182470
0.60    1.82211880
0.80    2.22554093
1.00    2.71828183
```

Output for LaTeX Table

What if I want to put the output in LaTeX? Recall that the table in LaTeX has the following structure:

```
\begin{table}[!hbtp]
\centering
\begin{tabular}{|r|c|l|}
\hline
Column 1 & Column 2 & Column 3 \\
\hline
Column 1 & Column 2 & Column 3 \\
Column 1 & Column 2 & Column 3 \\
\hline
\end{tabular}
\end{table}
```

So we can format our output for the use of LaTeX.

Example: fprintf for LaTeX

```
%% Example 4: fprintf for LaTeX
x = [0:.2:1]'; A = [x exp(x)]; A_size = size(A);
file_handle = fopen('exp.tex','w');
fprintf(file_handle, '\\begin{table}[!hbtp]\n');
fprintf(file_handle, '\\centering\n');
fprintf(file_handle, '\\begin{tabular}{cc}\n');
fprintf(file_handle, '\\hline\n');
fprintf(file_handle, '%6s & %12s \\\\\n','$x$','$\exp(x)$')
fprintf(file_handle, '\\hline\n');
for i = 1:A_size(1)
    fprintf(file_handle, '$%4.2f$ & $%10.8f$ \\\\\n', A(i,
end
fprintf(file_handle, '\\hline\n');
fprintf(file_handle, '\\end{tabular}\n');
fprintf(file_handle, '\\end{table}\n');
fclose(file_handle);
```

Output

```
\begin{table}[!hbtp]
\centering
\begin{tabular}{cc}
\hline
   $x$ &    $\exp(x)$ \\
\hline
$0.00$ & $1.00000000$ \\
$0.20$ & $1.22140276$ \\
$0.40$ & $1.49182470$ \\
$0.60$ & $1.82211880$ \\
$0.80$ & $2.22554093$ \\
$1.00$ & $2.71828183$ \\
\hline
\end{tabular}
\end{table}
```

Compile in LaTeX

x	$\exp(x)$
0.00	1.00000000
0.20	1.22140276
0.40	1.49182470
0.60	1.82211880
0.80	2.22554093
1.00	2.71828183

\usepackage{booktabs}

```
\begin{table}[!hbtp]
\centering
\begin{tabular}{cc}
\toprule
   $x$ &    $\exp(x)$ \\
\midrule
$0.00$ & $1.00000000$ \\
$0.20$ & $1.22140276$ \\
$0.40$ & $1.49182470$ \\
$0.60$ & $1.82211880$ \\
$0.80$ & $2.22554093$ \\
$1.00$ & $2.71828183$ \\
\bottomrule
\end{tabular}
\end{table}
```

x	$\exp(x)$
0.00	1.00000000
0.20	1.22140276
0.40	1.49182470
0.60	1.82211880
0.80	2.22554093
1.00	2.71828183