

MATH 3341: Introduction to Scientific Computing Lab

Melissa Butler

University of Wyoming

September 20, 2021



Lab 05: Formatting Output and \LaTeX



The background of the slide features a large, faint watermark of the University of Wyoming seal. The seal is circular with a rope-like border. Inside the border, the words "UNIVERSITY OF WYOMING" are written in an arc at the top, and "1886" is at the bottom. In the center of the seal is an open book with the word "EQUALITY" written below it.

Format Numerical Output



format: Format Numerical Output

- `format('short')` or `format short`: scaled fixed point format with 5 digits. Example:

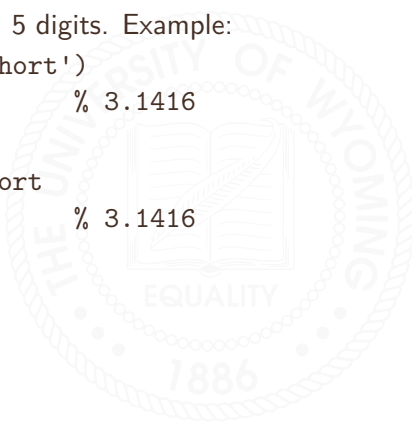
```
format('short')
```

```
pi                % 3.1416
```

```
or
```

```
format short
```

```
pi                % 3.1416
```



format: Format Numerical Output

- `format('short')` or `format short`: scaled fixed point format with 5 digits. Example:

```
format('short')
```

```
pi                % 3.1416
```

```
or
```

```
format short
```

```
pi                % 3.1416
```

- `format('long')` or `format long`: scaled fixed point format with 15 digits for double precision number and 7 digits for single precision number. Example:

```
format('long')
```

```
pi                % 3.141592653589793
```

```
or
```

```
format long
```

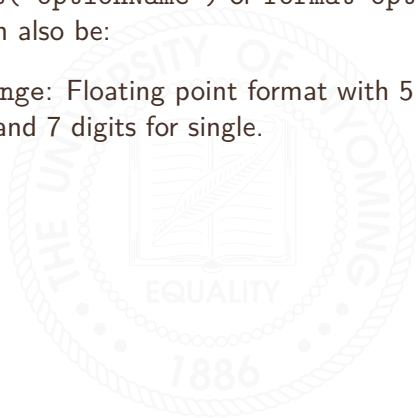
```
pi                % 3.141592653589793
```



format: Format Numerical Output

In short, `format('optionName')` or `format optionName`, `optionName` can also be:

- `shorte/longe`: Floating point format with 5 digits/15 digits for double and 7 digits for single.



format: Format Numerical Output

In short, `format('optionName')` or `format optionName`, `optionName` can also be:

- `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.



format: Format Numerical Output

In short, `format('optionName')` or `format optionName`, `optionName` can also be:

- `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



format: Format Numerical Output

In short, `format('optionName')` or `format optionName`, `optionName` can also be:

- `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
- `rat`: Approximation by ratio of small integers. Example:

```
format('rat')
```

```
pi % 355/113
```

```
abs(pi - 355/113) == 0 % logical 0 (false)
```



format: Format Numerical Output

In short, `format('optionName')` or `format optionName`, `optionName` can also be:

- `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
- `rat`: Approximation by ratio of small integers. Example:

```
format('rat')
```

```
pi % 355/113
```

```
abs(pi - 355/113) == 0 % logical 0 (false)
```

- `compact/loose`: Remove/Add line-feeds between outputs.



format: Format Numerical Output

```
%% Example 1
```

```
disp('Example 1 -- Output pi in different formats');
```

```
format
```

```
format('loose');    pi
```

```
format('compact'); pi    % 3.1416
```

```
format('short');    pi    % 3.1416
```

```
format('long');     pi    % 3.141592653589793
```

```
format('shorte');   pi    % 3.1416e+00
```

```
format('longe');    pi    % 3.141592653589793e+00
```

```
format('shortg');   pi    % 3.1416
```

```
format('longg');    pi    % 3.14159265358979
```

```
format('shorteng'); pi    % 3.1416e+000
```

```
format('longeng');  pi    % 3.14159265358979e+000
```

```
format('rat');      pi    % 355/113
```



format: Format Numerical Output

Example 1 -- Output pi in different formats

```
ans = 3.1416
```

```
ans = 3.1416
```

```
ans = 3.1416
```

```
ans = 3.141592653589793
```

```
ans = 3.1416e+00
```

```
ans = 3.141592653589793e+00
```

```
ans = 3.1416
```

```
ans = 3.14159265358979
```

```
ans = 3.1416e+000
```

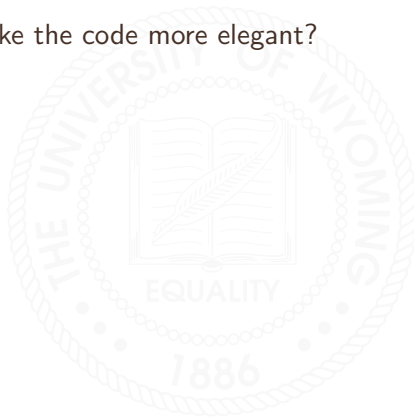
```
ans = 3.14159265358979e+000
```

```
ans = 355/113
```



format: Format Numerical Output

- Can we make the code more elegant?



format: Format Numerical Output

- Can we make the code more elegant?
- Yes! Using a for-loop!

```
disp('Example 2 -- Output pi in different formats');  
% Using a cell array to hold option names  
format      % reset to default format  
options = {'loose', 'compact', 'short', 'long', ...  
           'shorte', 'longe', 'shortg', 'longg', ...  
           'shorteng', 'longeng', 'rat'};  
for i = 1:length(options)  
    format(options{i});  
    pi  
end
```



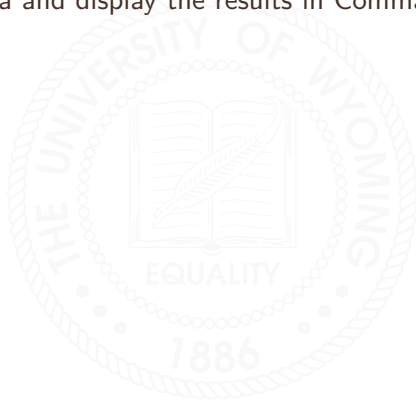
The background of the slide features a large, faint watermark of the University of Wyoming seal. The seal is circular with a rope-like border. Inside the border, the words "UNIVERSITY OF WYOMING" are written in an arc at the top, and "1886" is at the bottom. In the center of the seal is an open book with the word "EQUALITY" written below it.

Format String Output



fprintf and sprintf

- `fprintf(formatSpec, variable1, ..., variableN)`:
Format data and display the results in Command Window.



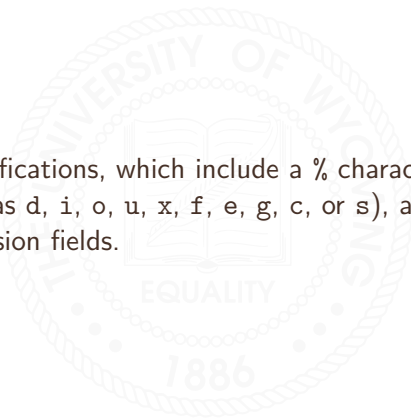
fprintf and sprintf

- `fprintf(formatSpec, variable1, ..., variableN)`:
Format data and display the results in Command Window.
- `strVariable = sprintf(formatSpec, variable1, ..., variableN)`: Write formatted data to a variable. Example:
% Libao Jin for floating point numbers
`piStr1 = sprintf('%f', pi)` % '3.141593'
% display 8 decimal places
`piStr2 = sprintf('%.8f', pi)` % '3.14159265'
% set string length to 12 by prepending spaces
`piStr3 = sprintf('%12.8f', pi)` % ' 3.14159265'
% set string length to 12 by appending spaces
`piStr4 = sprintf('%-12.8f', pi)` % '3.14159265 '
% Libao Jin for integers
`piStr5 = sprintf('%d', int32(pi))` % '3'
% set string length to 6 by prepending spaces
`piStr6 = sprintf('%6d', int32(pi))` % ' 3'



fprintf and sprintf - 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.



fprintf and sprintf - Conversion Specifications

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.



fprintf and sprintf - Flags

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: <ul style="list-style-type: none">- 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. |
-



fprintf and sprintf - Escape characters

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



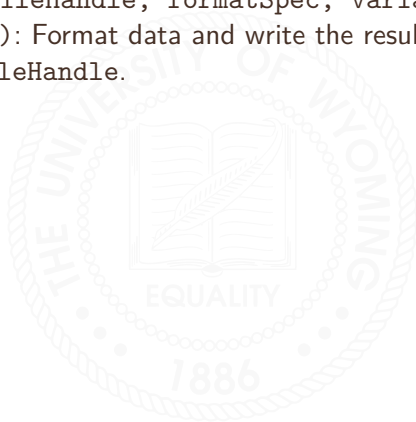
fprintf and sprintf - Example

```
%% Example 3: sprintf
hour = 11;
minute = 20;
am = 'a.m.';
currentTime = sprintf('The current time is: %d:%d %s',...
    hour, minute, am)
```



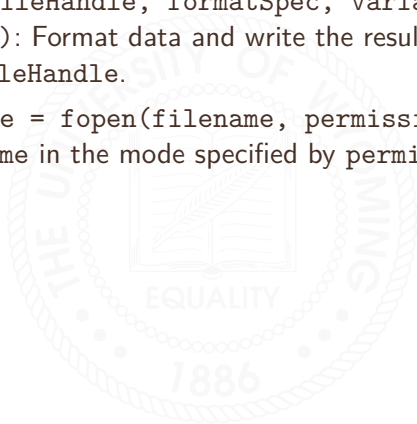
fprintf, fopen, fclose, and type

- `fprintf(fileHandle, formatSpec, variable1, ..., variableN)`: Format data and write the results to a file through `fileHandle`.



fprintf, fopen, fclose, and type

- `fprintf(fileHandle, formatSpec, variable1, ..., variableN)`: Format data and write the results to a file through `fileHandle`.
- `fileHandle = fopen(filename, permission)`: Open the file `filename` in the mode specified by `permission`.



fprintf, fopen, fclose, and type

- `fprintf(fileHandle, formatSpec, variable1, ..., variableN)`: Format data and write the results to a file through `fileHandle`.
- `fileHandle = fopen(filename, permission)`: Open the file `filename` in the mode specified by `permission`.
- `fclose(fileHandle)`: Close the file associated with `fileHandle`. Example:

```
fileHandle = fopen('./current_time.txt', 'w');  
fprintf(fileHandle, 'The current time is: %d:%d %s',...  
        11, 20, 'a.m.')
```

```
fclose(fileHandle);
```



fprintf, fopen, fclose, and type

- `fprintf(fileHandle, formatSpec, variable1, ..., variableN)`: Format data and write the results to a file through `fileHandle`.
- `fileHandle = fopen(filename, permission)`: Open the file `filename` in the mode specified by `permission`.
- `fclose(fileHandle)`: Close the file associated with `fileHandle`. Example:

```
fileHandle = fopen('./current_time.txt', 'w');  
fprintf(fileHandle, 'The current time is: %d:%d %s',...  
        11, 20, 'a.m.')
```



```
fclose(fileHandle);
```
- `type('path/filename')`: Print the contents of `filename`. Example:

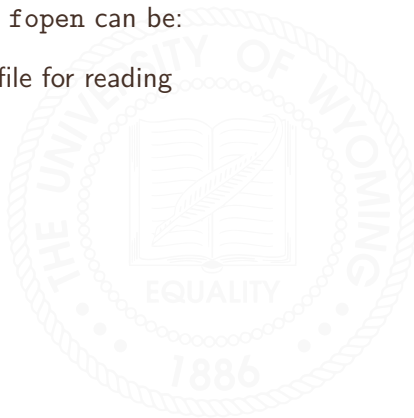
```
type('./current_time.txt')
```



fprintf, fopen, fclose, and type

permission for fopen can be:

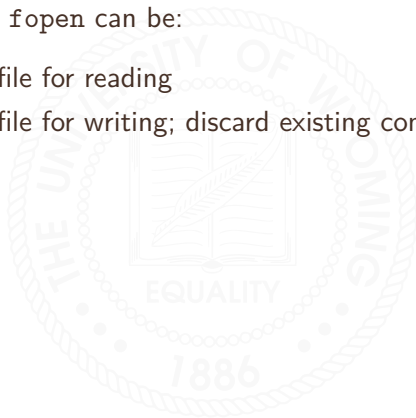
- 'r': open file for reading



fprintf, fopen, fclose, and type

permission for fopen can be:

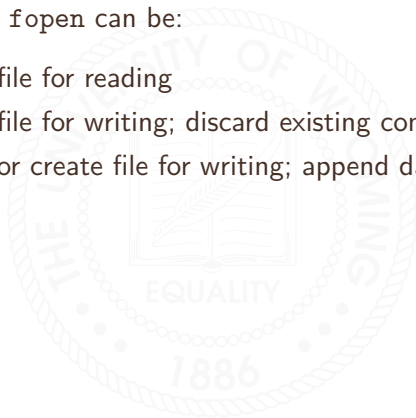
- 'r': open file for reading
- 'w': open file for writing; discard existing contents



fprintf, fopen, fclose, and type

permission for fopen can be:

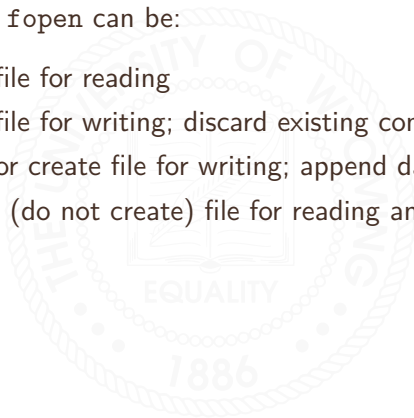
- 'r': open file for reading
- 'w': open file for writing; discard existing contents
- 'a': open or create file for writing; append data to end of file



fprintf, fopen, fclose, and type

permission for fopen can be:

- 'r': open file for reading
- 'w': open file for writing; discard existing contents
- 'a': open or create file for writing; append data to end of file
- 'r+': open (do not create) file for reading and writing



fprintf, fopen, fclose, and type

permission for fopen can be:

- 'r': open file for reading
- 'w': open file for writing; discard existing contents
- 'a': open or create file for writing; append data to end of file
- 'r+': open (do not create) file for reading and writing
- 'w+': open or create file for reading and writing; discard existing contents



fprintf, fopen, fclose, and type

permission for fopen can be:

- 'r': open file for reading
- 'w': open file for writing; discard existing contents
- 'a': open or create file for writing; append data to end of file
- 'r+': open (do not create) file for reading and writing
- 'w+': open or create file for reading and writing; discard existing contents
- 'a+': open or create file for reading and writing; append data to end of file



fprintf, fopen, fclose, and type

permission for fopen can be:

- 'r': open file for reading
- 'w': open file for writing; discard existing contents
- 'a': open or create file for writing; append data to end of file
- 'r+': open (do not create) file for reading and writing
- 'w+': open or create file for reading and writing; discard existing contents
- 'a+': open or create file for reading and writing; append data to end of file
- 'W': open file for writing without automatic flushing



fprintf, fopen, fclose, and type

permission for fopen can be:

- 'r': open file for reading
- 'w': open file for writing; discard existing contents
- 'a': open or create file for writing; append data to end of file
- 'r+': open (do not create) file for reading and writing
- 'w+': open or create file for reading and writing; discard existing contents
- 'a+': open or create file for reading and writing; append data to end of file
- 'W': open file for writing without automatic flushing
- 'A': open file for appending without automatic flushing



fprintf, fopen, fclose, and type - Example

MATLAB code:

```
%% Example 4: fprintf
x = [0:.2:1]'; fx = [x,exp(x)];
fileHandle = fopen('exp.txt','w');
fprintf(fileHandle,'%6s %12s\n','x','exp(x)');
% Libao Jin output row by row using a for-loop
for i = 1:size(fx, 1)
    fprintf(fileHandle, '%6.2f %12.8f\n', ...
            fx(i, 1), fx(i, 2));
end
fclose(fileHandle);
% View the contents of the file with the `type` command
type('exp.txt');
```



Examples: `fprintf`

Output:

x	exp(x)
0.00	1.00000000
0.20	1.22140276
0.40	1.49182470
0.60	1.82211880
0.80	2.2254093
1.00	2.71828183



fprintf - Application: Generate L^AT_EX Table

```

\begin{table}[!hbt]
\centering
\begin{tabular}{rcl}
\toprule
Column 1 & Column 2 & Column 3 \\
\midrule
Col 1      & Col 2      & Col 3 \\
\bottomrule
\end{tabular}
\end{table}

```

Column 1	Column 2	Column 3
Col 1	Col 2	Col 3



fprintf - Application: Generate \LaTeX Table - Example

```
%% Example 6: fprintf for LaTeX
x = [0:.2:1]'; fx = [x,exp(x)];
fileHandle = fopen('exp.tex','w');
fprintf(fileHandle, '\\begin{table}[!hbt] \n');
fprintf(fileHandle, '\\centering \n');
fprintf(fileHandle, '\\begin{tabular}{cc} \n');
fprintf(fileHandle, '\\toprule \n');
fprintf(fileHandle, '%6s & %12s \\ \\ \\ \n', '$x$', '$\exp(x)$');
fprintf(fileHandle, '\\midrule \n');
for i = 1:size(fx, 1)
    fprintf(fileHandle, '$%4.2f$ & $%10.8f$ \\ \\ \\ \n', fx(i),
end
fprintf(fileHandle, '\\bottomrule \n');
fprintf(fileHandle, '\\end{tabular} \n');
fprintf(fileHandle, '\\end{table} \n');
fclose(fileHandle);
```



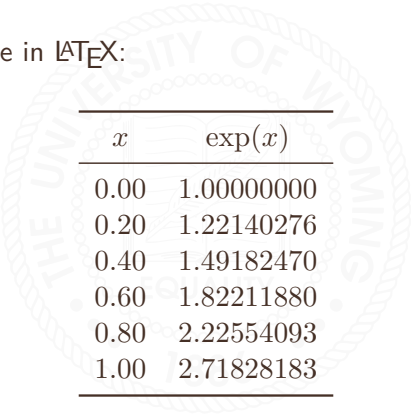
fprintf - Application: Generate L^AT_EX Table - Example

```
\begin{table}[!hbtpr]
\centering
\begin{tabular}{cc}
\toprule
    $x$ &    $\exp(x)$ \\
\midrule
$0.00$ & $1.000000000$ \\
$0.20$ & $1.22140276$ \\
$0.40$ & $1.49182470$ \\
$0.60$ & $1.82211880$ \\
$0.80$ & $2.22554093$ \\
$1.00$ & $2.71828183$ \\
\bottomrule
\end{tabular}
\end{table}
```



`fprintf` - Application: Generate \LaTeX Table - Example

Compile the table 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

