

## Contents

Acknowledgement . . . . .	1
Square Waves from Sine Waves . . . . .	1
Add an Odd Harmonic and Plot It . . . . .	2
Note About Gibbs Phenomenon . . . . .	3
Listings . . . . .	3

## Listings

1 listing of exampleB script . . . . .	3
2 listing of publish_mpl_examples script . . . . .	7

## exampleB.m : file for publish\_mpl showing extra options

As exampleA.m with the only difference that the listings get their own caption (and therefore the section headers are omitted). The listings are presented in a 'lstlistoflistings' by specifying this in the `publish` options (see example4) in `publish_mapl_examples`.

## Acknowledgement

This file is adapted from the `fourier_demo2.m` file that is included in MATLAB and can be copied in the current directory with

```
copyfile(fullfile(matlabroot,'help','techdoc',...  
'matlab_env','examples','fourier_demo2.m'),'.','f')
```

## Square Waves from Sine Waves

The Fourier series expansion for a square-wave is made up of a sum of odd harmonics, as shown here by the plots in figure 1 on page 2 (1 harmonic), figure 2 on page 4 (5 harmonics) and figure 3 on page 5 (9 harmonics).

```
1 if exist('avalue','var')  
2     fprintf('print the value passed to this script: %f\n',avalue)  
3 else  
4     fprintf('no value passed to this script\n')  
5 end
```

```
print the value passed to this script: 2.000000
```

## Add an Odd Harmonic and Plot It

```
1 t = 0:.1:pi*4;  
2 k = 1 ;  
3 y = sin(k*t)/k;  
4 figure(k)  
5 plot(t,y);  
6 title(sprintf('MATLAB caption: plot when k=%.0f',k))
```

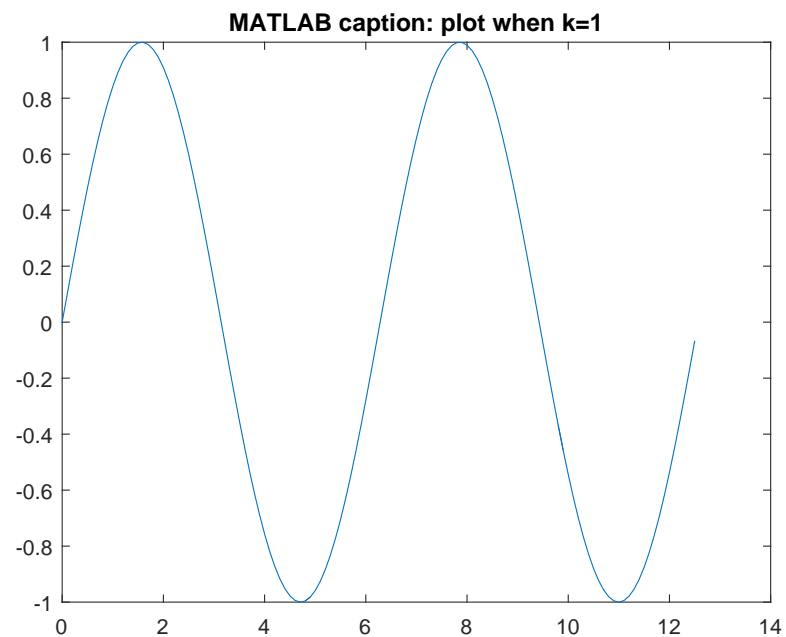


Figure 1: first harmonic

In each iteration of the for loop add an odd harmonic to  $y$ . As  $k$  increases, the output approximates a square wave with increasing accuracy.

Perform the following mathematical operation at each iteration:

$$y = y + \frac{\sin kt}{k}$$

Display some of the plots:

```
1 for k = 3:2:9
2     y = y + sin(k*t)/k;
3     if mod(k,4)==1
4         figure(k)
5         plot(t,y)
6         title(sprintf('MATLAB caption: plot when k=%.0f',k))
7     end
8 end
```

## Note About Gibbs Phenomenon

Even though the approximations are constantly improving, they will never be exact because of the Gibbs phenomenon, or ringing.

## Listings

Listing 1: listing of exampleB script

```
1 %% exampleB.m : file for publish_mpl showing extra options
2 % As exampleA.m with the only difference that the listings
3 % get their own caption (and therefore the section headers
4 % are omitted). The listings are presented in a 'lstlistoflistings'
5 % by specifying this in the |publish| options (see example4) in
6 % publish_mapl_examples.
7 %% Acknowledgement
8 % This file is adapted from the |fourier_demo2.m| file
9 % that is included in MATLAB and can be copied in
10 % the current directory with
```

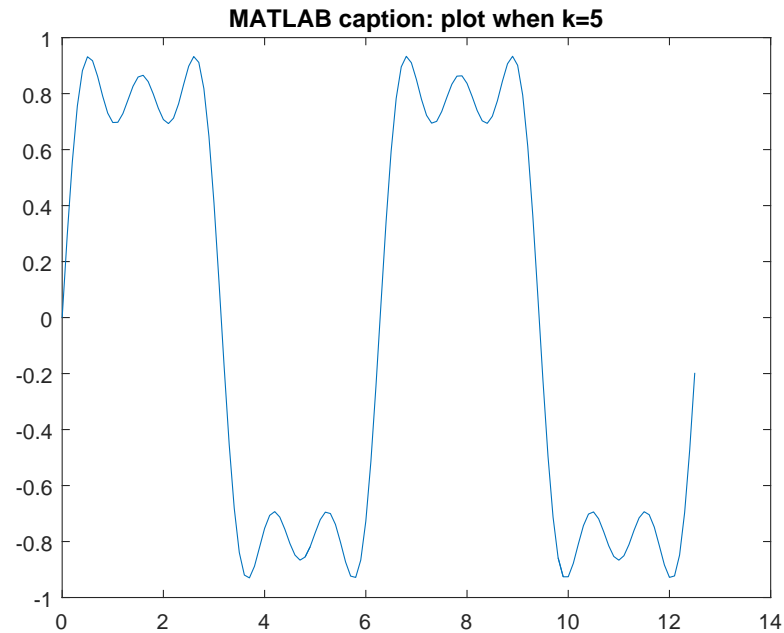


Figure 2: sum of first 5 harmonics

```

11 %%
12 %
13 %   copyfile(fullfile(matlabroot,'help','techdoc',...
14 %   'matlab_env','examples','fourier_demo2.m'),'.','f')
15 %% Square Waves from Sine Waves
16 % <latex>
17 % % The actual function to publish starts now
18 % % This text block is changed to a latex block to show the caption and reference capabilities
19 % %
20 % % the following statements insert the references to the plots:
21 % The Fourier series expansion for a square-wave is
22 % made up of a sum of odd harmonics, as shown here
23 % by the plots in figure \ref{exampleB_01.eps} on page \pageref{exampleB_01.eps} (1 harmonic),
24 % figure \ref{exampleB_02.eps} on page \pageref{exampleB_02.eps} (5 harmonics) and

```

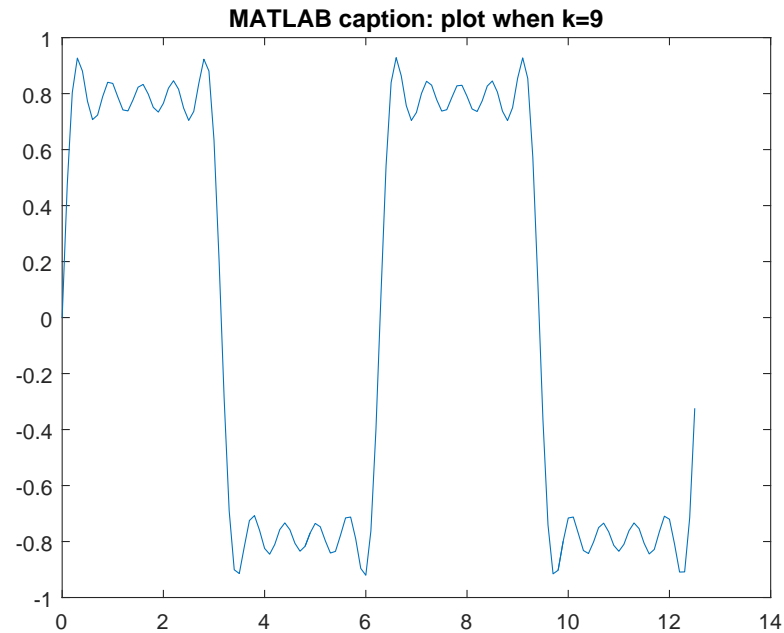


Figure 3: sum of first 9 harmonics

```

25 % figure \ref{exampleB_03.eps} on page \pageref{exampleB_03.eps} (9 harmonics).
26 % %
27 % % the following statements define the captions of the plots:
28 % \global\def\captionA{first harmonic}
29 % \global\def\captionB{sum of first 5 harmonics}
30 % \global\def\captionC{sum of first 9 harmonics}
31 % </latex>
32 if exist('avalue','var')
33     fprintf('print the value passed to this script: %f\n',avalue)
34 else
35     fprintf('no value passed to this script\n')
36 end
37 %% Add an Odd Harmonic and Plot It
38 t = 0:.1:pi*4;

```

```

39 k    = 1 ;
40 y    = sin(k*t)/k;
41 figure(k)
42 plot(t,y);
43 title(sprintf('MATLAB caption: plot when k=%.0f',k))
44
45 %%
46 % In each iteration of the for loop add an odd
47 % harmonic to y. As _k_ increases, the output
48 % approximates a square wave with increasing accuracy.
49 %
50 % Perform the following mathematical operation
51 % at each iteration:
52 %
53 % $$ y = y + \frac{\sin kt}{k} $$
54 %
55 % Display some of the plots:
56 %
57
58 for k = 3:2:9
59     y = y + sin(k*t)/k;
60     if mod(k,4)==1
61         figure(k)
62         plot(t,y)
63         title(sprintf('MATLAB caption: plot when k=%.0f',k))
64     end
65 end
66
67 %% Note About Gibbs Phenomenon
68 % Even though the approximations are constantly
69 % improving, they will never be exact because of the
70 % Gibbs phenomenon, or ringing.
71
72 %% Listings
73 % <latex>
74 % % assuming m-file in directory one level higher than tex dir (using the standard html subdirectory)
75 % % assuming numbers and framed are not set in \usepackage and they are wanted

```

```

76 % % \lstinputlisting[frame=single,numbers=left]{../exampleB.m}
77 % % assuming numbers and framed are set in \usepackage and they are not wanted
78 % % \lstinputlisting[frame=none,numbers=none]{../exampleB.m}
79 % % assuming numbers and framed are set in \usepackage are set and wanted
80 % \lstinputlisting[caption=listing of exampleB script]{../exampleB.m}
81 % </latex>
82 %%
83 % <latex>
84 % \lstinputlisting[caption=listing of publish\_mpl\_examples script]{../publish_mpl_examples.m}
85 % </latex>

```

Listing 2: listing of publish\_mpl\_examples script

```

1  addpath('../code')
2  %% example1: -> pdf
3  % Use the function to create pdf-file.
4  % This is the same as using the publish user interface.
5  mycode = { ...                                % example of code to execute (two lines)
6          'avalue = 2;' ...
7          'exampleA' ...
8          } ;
9  pstruct = struct( ...                          % publish options
10     'format' , 'pdf' , ...                      % output format
11     'call' , {mycode} , ...                    % code to execute (defined above)
12     'newname' , 'exampleA1.pdf' );              % new name of output file
13  newname = publish_mpl('exampleA', pstruct) ;   % produce the output file (pdf)
14
15  %% example2: -> latex
16  % Use the function to create tex-file
17  % with as much as possible the same layout
18  % as the original tex file but with references, captions
19  % and listings
20  mycode = { ...                                % example of code to execute (one line)
21          'exampleA' ...
22          } ;
23  pstruct = struct( ...                          % publish options
24     'format' , 'latex' , ...                    % output format latex using the new xsl file

```

```

25     'call' , {mycode} , ...
26     'orientation', 'portrait', ...
27     'newname' , 'exampleA2.tex' , ...
28     'prettifier_options' , '' ) ;
    ' )

29
30 newname = publish_mpl('exampleA', pstruct) ;
31
32 %% example3: -> latex
33 % Same as example2 but the layout is landscape and
34 % the MATLAB code will be in frames with numbers.
35 mycode = { ...
36     'exampleA' ...
37     } ;
38 pstruct = struct( ...
39     'format' , 'latex' , ...
40     'call' , {mycode} , ...
41     'newname' , 'exampleA3.tex');
42 % 'orientation', 'landscape', ...
43 % 'prettifier_options', 'framed,numbered' , ...
44
45 newname = publish_mpl('exampleA', pstruct) ;
46 %% example4: -> latex
47 % same as example3 but listings have their own
48 % caption in exampleB and they are listed by
49 % setting 'makelstlistoflistings' to true
50 mycode = { ...
51     'exampleB' ...
52     } ;
53 pstruct = struct( ...
54     'format' , 'latex' , ...
55     'call' , {mycode} , ...
56     'newname' , 'exampleB1.tex' , ...
57     'pdfauthor', 'han@hanoostdijk.nl' , ...
58     'makelstlistoflistings', true);
59 newname = publish_mpl('exampleB', pstruct) ;
60
    % code to execute (defined above)
    % overwrite orientation (default 'landscape')
    % new name of output file
    % overwrite prettify options (default 'framed,numbered')

    % produce the output file (tex)

    % example of code to execute (one line)

    % publish options
    % output format latex using the new xsl file
    % code to execute (defined above)
    % new name of output file
    % use default orientation ('landscape')
    % use default prettify options ('framed,numbered')

    % produce the output file (tex)

    % example of code to execute (one line)

    % publish options
    % output format latex using the new xsl file
    % code to execute (defined above)
    % new name of output file
    % insert a pdf option
    % create lstlistoflistings
    % produce the output file (tex)

```



```

61 %% example5: -> latex
62 % same as example4 but now with a regular LaTeX contents
63 % by setting 'maketableofcontents' to true
64 mycode = { ... % example of code to execute (one line)
65     'exampleB' ...
66 } ;
67 pstruct = struct( ... % publish options
68     'format' , 'latex' , ... % output format latex using the new xsl file
69     'call' , {mycode} , ... % code to execute (defined above)
70     'newname' , 'exampleB2.tex' , ... % new name of output file
71     'pdfauthor' , 'han@hanoostdijk.nl' , ... % insert a pdf option
72     'maketableofcontents' , true , ... % create tableofcontents
73     'makelstlistoflistings' , true); % create lstlistoflistings
74 newname = publish_mpl('exampleB', pstruct) ; % produce the output file (tex)
75
76 %% example6: -> xml
77 % same as example3 but now to xml format
78 mycode = { ... % example of code to execute (one line)
79     'exampleB' ...
80 } ;
81 pstruct = struct( ... % publish options
82     'format' , 'xml' , ... % output format latex using the new xsl file
83     'call' , {mycode} , ... % code to execute (defined above)
84     'newname' , 'exampleB3.xml' , ... % new name of output file
85     'pdfauthor' , 'han@hanoostdijk.nl' , ... % insert a pdf option
86     'makelstlistoflistings' , true); % create lstlistoflistings
87 newname = publish_mpl('exampleB', pstruct) ; % produce the output file (tex)

```