Customize MATLAB publish with XSL-stylesheet

Valery Khamenya

August 6, 2013

Contents

Introduction

This is a short example on how to use Matlab "publish" function to create *comprehensive* reports. The information on MATLAB human-friendly markup features could be found here.

publish `main_rep.m` file with:

```
opts.format='latex'
opts.stylesheet='custom_mxdom2latex.xsl'
publish('main_rep', opts)
```

Does your target reading audience need to see the MATLAB source code? If no, consider to set `opts.showCode=false` in addition before calling `publish` function above.

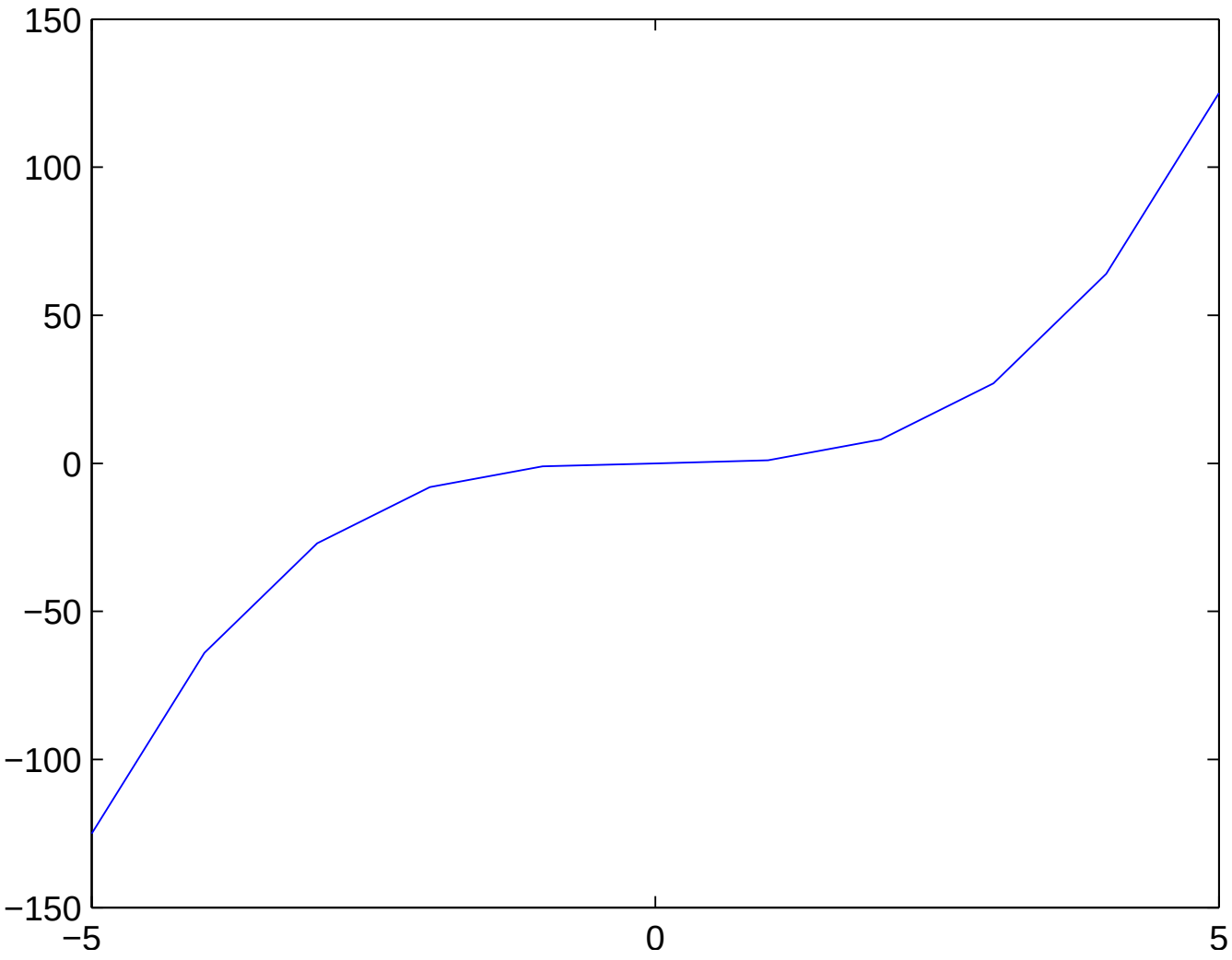
The call of MATLAB publish function causes the execution of `main_rep.m`. The graphs generated during the exucution are automatically saved in `html` directory, beside of this, the execution creates a DOM-object representing the syntaxical elements of `main_rep.m` file along with the results obtained during the execution. The DOM-object is then automatically transformed according to `custom_mxdom2latex.xsl` (see also XSL-Transformations). The resulting transformation is then saved as `main_rep.tex` text file in `html` directory as well. The `main_rep.tex` should be then processed according to the installed LaTeX implementation. For example, one could trigger this like `pdflatex main_rep` as for MikTeX.

Examples

Graphs and their descriptions generated in a loop

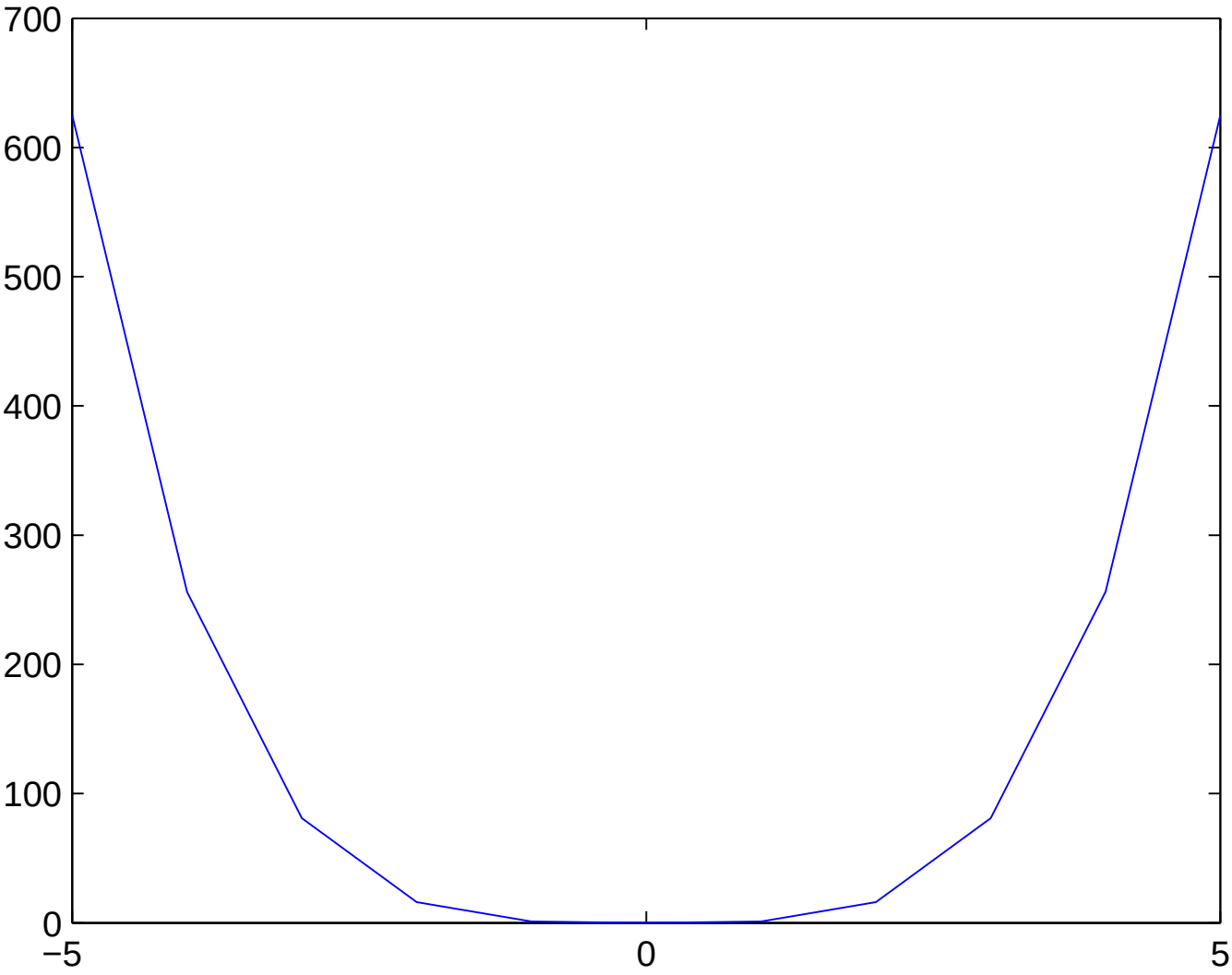
Here is an example how the report parts could be generated within a loop. Also, the comment sections right before the function declaration will be published even if the `opts.showCode=false` . Thus, one should rather tune up the comment style either for `opts.showCode=false` or for `opts.showCode=true`. The same is true for the current paragraph that was generated from such a comment section right before the `main` function of the `main_rep.m`.

Albeit this sentence is syntactically placed as a comment within a loop (see `main_rep.m`), it will appear only once in a final report. In oreder to get LaTeX parts to be generated within a loop, one could use MATLAB disp function. Here is an example of a $f(x) = x^i, i = 3$ plot and this description text generated in the loop.

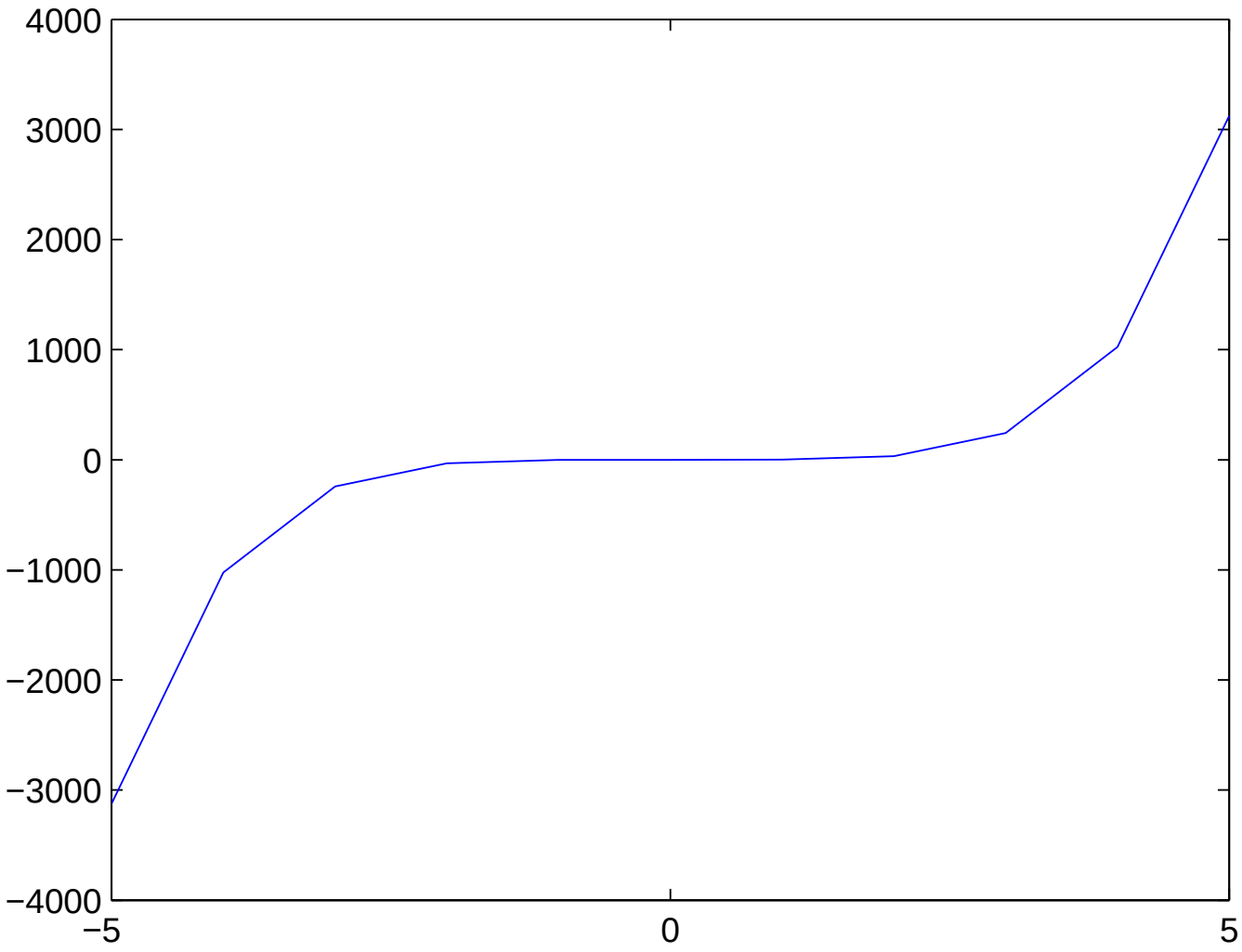


Here is an example of the conditional part of the report that is placed in the loop only if $i = 3$.

Here is an example of a $f(x) = x^i, i = 4$ plot and this description text generated in the loop.



Here is an example of a $f(x) = x^i, i = 5$ plot and this description text generated in the loop.



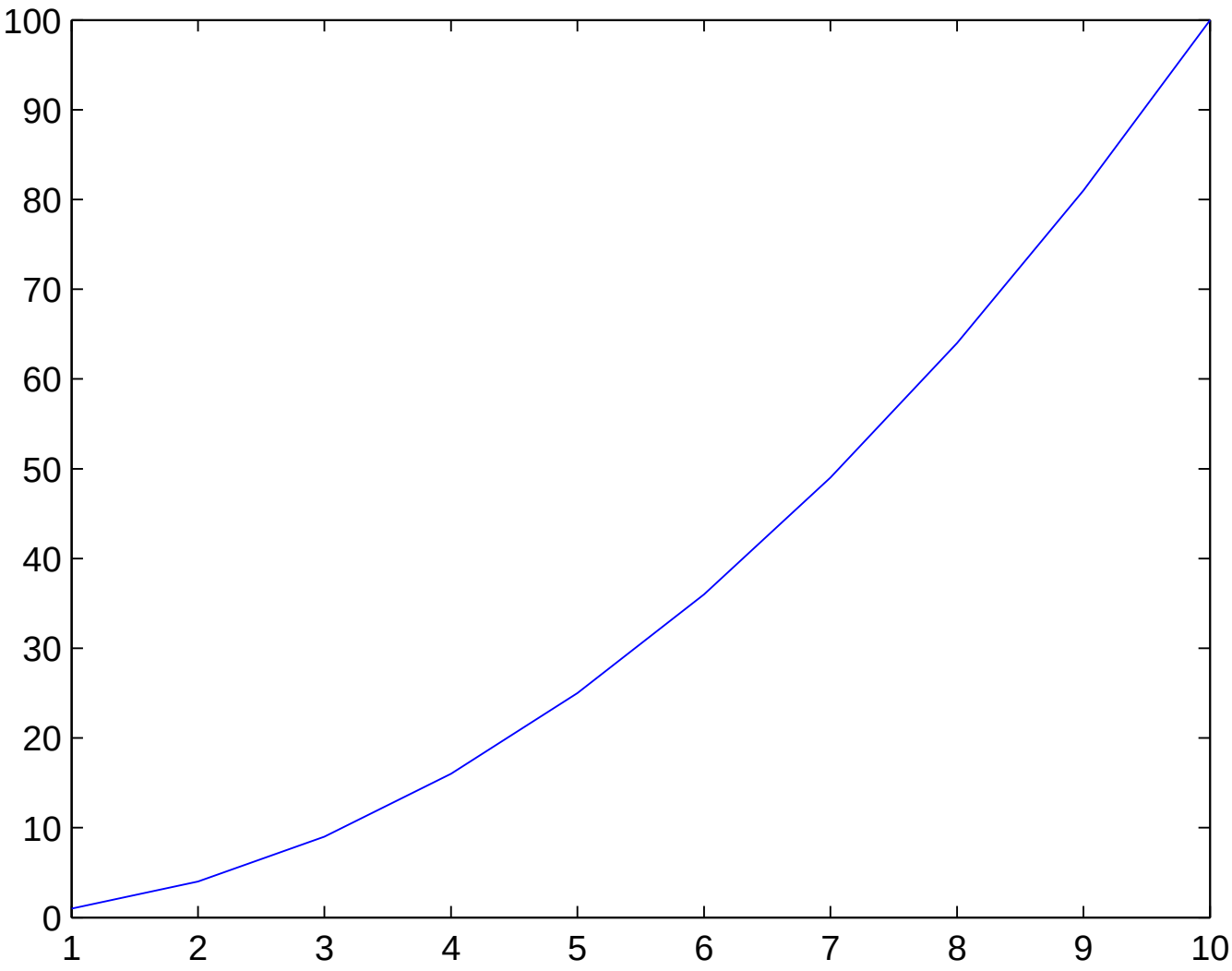

```
x =

    -5    -4    -3    -2    -1     0     1     2     3     4     5
```

Subcalls of the functions from the same file

This paragraph is generated from a comment section right before the `local_sub_func` function definition. It is generated independently on whether the `local_sub_func` is called or not.

The same is true even for any comment section *within* the body of `local_sub_func`, like this paragraph.

Here is the graph of $f(x) = x^2$ and it was generated from local_sub_func:


100
90
80
70
60
50
40
30
20
10
0
1
2
3
4
5
6
7
8
9
10

Subcalls of the functions from other files

Here is the graph of $f(x) = sin(x)$ and it was generated from external_sub_func defined in the other_file.m (this paragraph is generated using disp, because comment parsing is done only for the top-level .m-file passed to MATLAB publish function.):