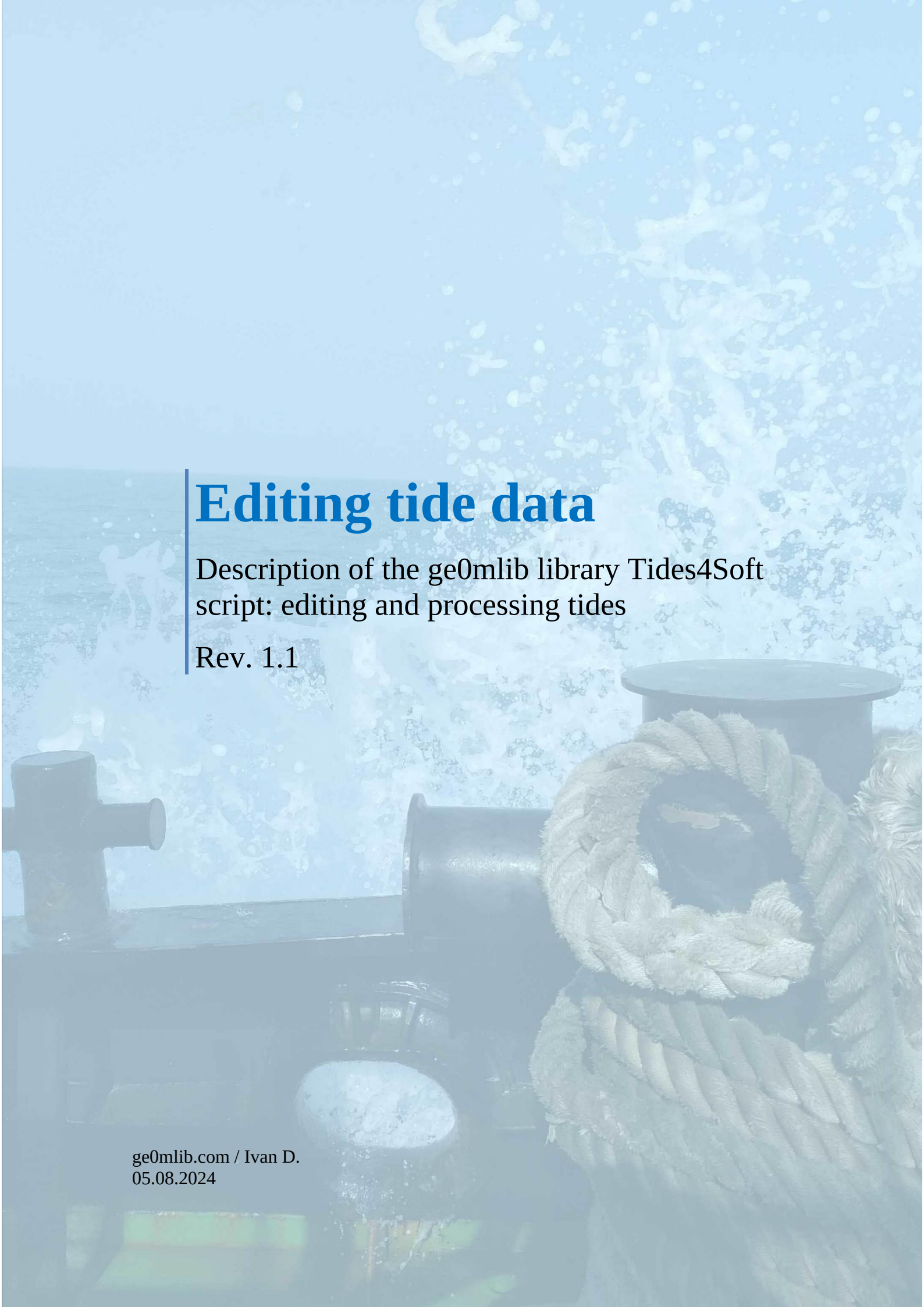




Editing tide data

Description of the ge0mlib library Tides4Soft
script: editing and processing tides

Rev. 1.1

Contents

Introduction 3

1. Data formats 4

2. Script commands 6

Appendix 1. Tides4Soft script code 11

Revision History

Version	Date	Description of changes
1.0	2022-08-14	Document created // Ivan D.
1.1	2024-08-05	Document revised // Ivan D.

Figures list

Figure 0.1 MatLab window screenshots with command session examples for the Tides4Soft script 3

Figure 2.1 Input files 6

Figure 2.2 Graph of the downloaded tide data 6

Figure 2.3 Graph of the “rarefied” data.....7

Figure 2.4 MatLab graphics window for interactive outliers removal 8

Figure 2.5 Window showing the tool removing measurements from a selected rectangular area 8

Figure 2.6 Window showing the graphs displaying the editing results.....9

Figure 2.7 Original data (blue dots) and processed data interpolated to 1 minute interval (red dots).....10

Figure 2.8 Contents of the folder containing Silas files format.....10

Introduction

The Tides4Soft script was created to edit text files containing tide data (often such editing is done in Excel, which is time-consuming for the processor). The script implements the following processing steps:

- loading all files with tides from a specific folder into MatLab and combining them into a single data array;
- array decimation for quick smoothing of tide data;
- interactive removal of outliers using the gMapPickHandleNan tool,
- smoothing with a given width of the sliding window,
- interpolation of values removed at the previous step,
- data interpolation to a 1-minute step,
- saving processed tide data in a text file.

To run the script, one must install MatLab 2018a software (or a later version) and the ge0mLib library.

Processing results are saved in formats readable by DugInsight or Silas software (tdx format). In *Figure 0.1* there are MatLab window screenshots with command session examples for the Tides4Soft script (for editing text files with tide data).

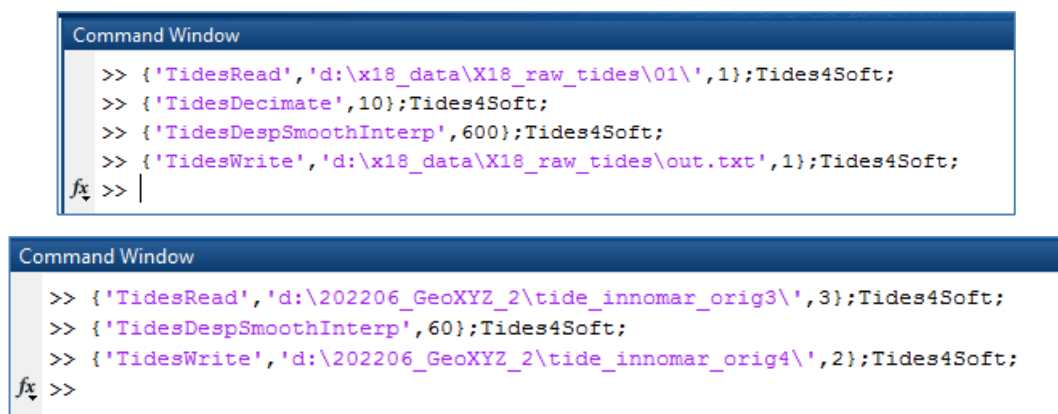


Figure 0.1 MatLab window screenshots with command session examples for the Tides4Soft script

1. Data formats

The script code implements three formats for the source data:

- 1) data received from USV without recording the registration date in a text file;
- 2) data received from the USV with the registration date recorded in a text file;
- 3) data in the format most commonly used in tidal stations.

Example of the “1” format for source data (for USV without recording the registration date)

#<Time> HHmmss.zzz, Latitude WGS-84 - COG - Latitude WGS-84, Longitude WGS-84 - COG - Longitude WGS-84, Height WGS-84 - COG - Height WGS-84, Easting - COG - Easting, Northing - COG - Northing, Height - COG - Height

161248.237,5520.17000858,00134.00310710,54.775,580051.32,6100123.86,8.44

161249.001,5520.17000347,00134.00310369,54.777,580051.33,6100123.85,8.44

The last column contains the tide value (most often in centimeters).

The registration date for the “1” format is read from the name of the file containing the data; in this case, characters containing the date are “cut out” from the file name by the fName(end-16:end-9) operator. File names example:

X18_Tide_20220715_0000.log; X18_Tide_20220716_0000.log

Example of the “2” format for source data (for USV with the registration date)

#<Time> HHmmss.zzz,<Date> dd,<Date> MM,<Date> yyyy, Latitude WGS-84 - COG - Latitude WGS-84, Longitude WGS-84 - COG - Longitude WGS-84, Height WGS-84 - COG - Height WGS-84, Easting - COG - Easting, Northing - COG - Northing, Height - COG - Height

000001.002,19,07,2022,5516.94135841,00117.68878421,46.600,608305.37,6127537.33,0.87

000002.002,19,07,2022,5516.94081535,00117.68899275,46.623,608305.17,6127536.32,0.89

Example of the “3” format for source data

23:27:38,2.39

23:28:08,2.60

The registration date for the “3” format is read from the name of the file containing the data; in this case, characters containing the date are “cut out” from the file name by the fName(end-11:end-4) operator. File names example:

tide_innomar_20220512.txt; tide_innomar_20220513.txt

Attention! Lines not conforming to the format must be removed from the input files before using the script. Example of lines not conforming to the “3” format:

13:57:32,

13:58:02,

The script code implements two formats for processed data:

- 1) format readable by DugInsight software;
- 2) tdx format readable by Silas software.

Example of the “1” format for processed data (for DugInsight):

1612:00 15 7 2022 4.74

1613:00 15 7 2022 4.74

Example of the “2” format for processed data (tdx format for Silas):

%08:40 1,974

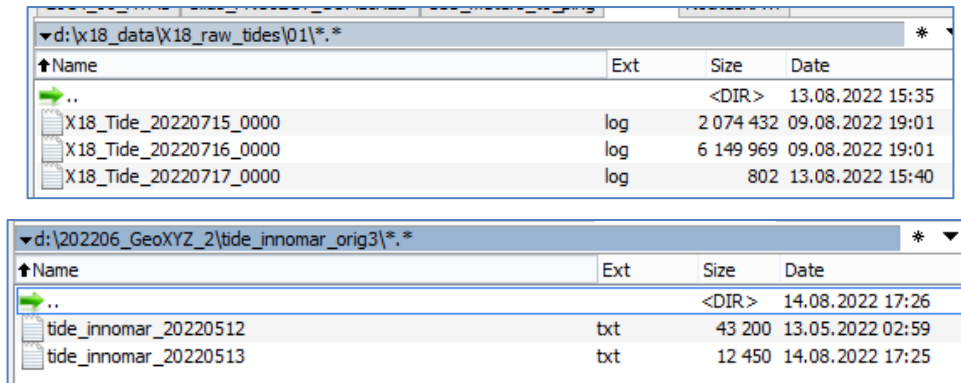
%08:41 2.007

The processed data is placed in text files created separately for each day. Example of processed file names:

[Tide_20220512.tdx](#); [Tide_20220513.tdx](#)

2. Script commands

Before running the script, one must place the files with the source data in a separate folder. An example of folder contents for the “1” and “3” formats is shown in *Figure 2.1*.



Name	Ext	Size	Date
..		<DIR>	13.08.2022 15:35
X18_Tide_20220715_0000	log	2 074 432	09.08.2022 19:01
X18_Tide_20220716_0000	log	6 149 969	09.08.2022 19:01
X18_Tide_20220717_0000	log	802	13.08.2022 15:40

Name	Ext	Size	Date
..		<DIR>	14.08.2022 17:26
tide_innomar_20220512	txt	43 200	13.05.2022 02:59
tide_innomar_20220513	txt	12 450	14.08.2022 17:25

Figure 2.1 Input files

For the “1” and “3” data formats, the recording date is read by the script from the file name. The position of the date characters in the file name is calculated as fName(end-16:end-9) for the “1” format and as fNameN(end-11:end-4) for the “3” format.

The script commands implementing sequential processing steps are described below.

1) Loading files containing source data is performed using a script command such as

```
>> {'TidesRead','d:\x18_data\X18_raw_tides\01',1};Tides4Soft;
```

where

'TidesRead' – script command name;

'd:\x18_data\X18_raw_tides\01' – name of the folder containing the source data;

1 – source data format number (1, 2 or 3).

After loading the data, the script plots a tide graph along the time axis (*Figure 2.2*).

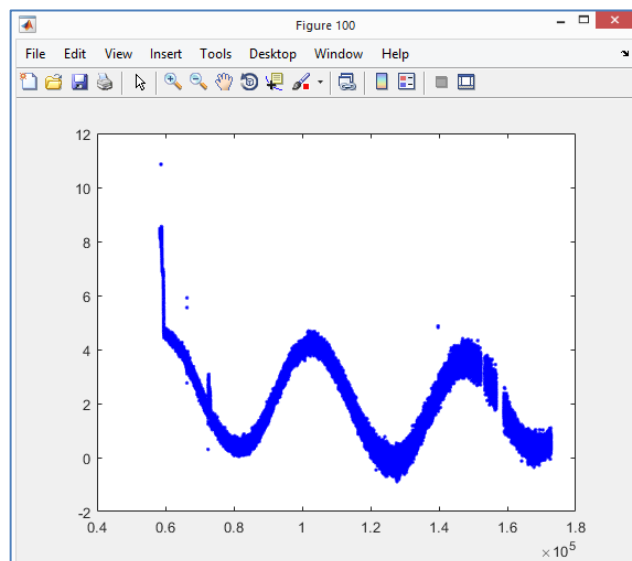


Figure 2.2 Graph of the downloaded tide data

2) If it is necessary to reduce the amount of data, one needs to use a script command such as

```
>> {'TidesDecimate',10};Tides4Soft;
```

where

'TidesDecimate' – script command name;

10 – step at which the data is saved (for the given command example, every 10th value from a series of tides will be saved).

After executing the command, the script plots a graph containing “rarefied” data ([Figure 2.3](#)).

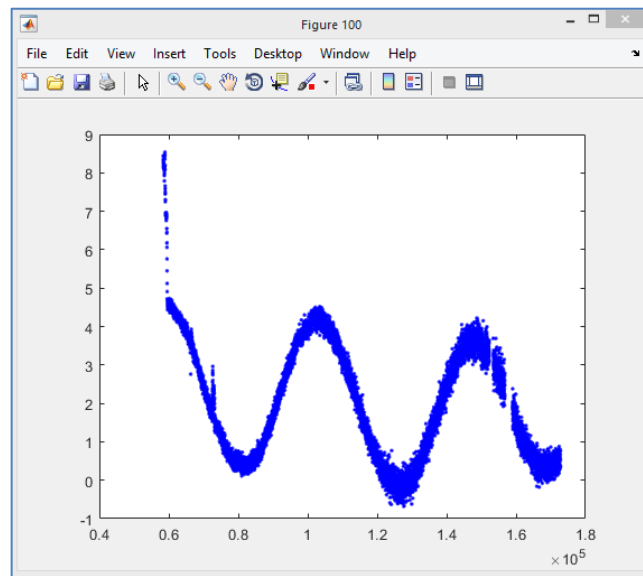


Figure 2.3 Graph of the “rarefied” data

3) To remove outliers, smooth and interpolate the removed values, one needs to use a script command such as

```
>> {'TidesDespSmoothInterp',600};Tides4Soft;
```

where

'TidesDespSmoothInterp' – script command name;

600 – sliding window width for smoothing tide data.

After running the command for execution, the MatLab graphics window opens to interactively remove outliers ([Figure 2.4](#))

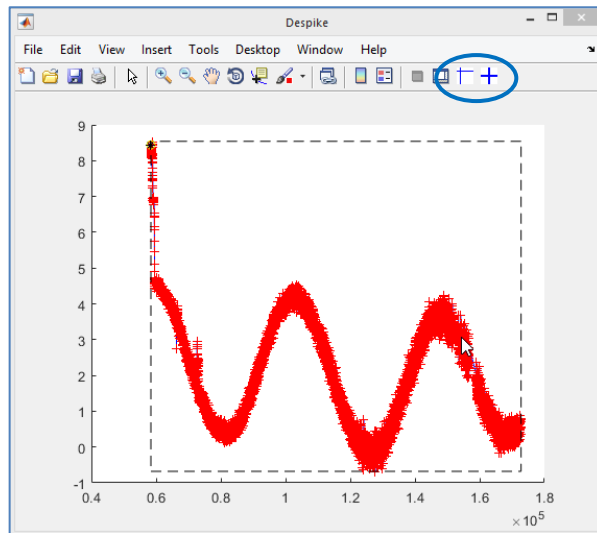


Figure 2.4 MatLab graphics window for interactive outliers removal

Outliers can be removed by two tools (the icons are circled in blue in [Figure 2.4](#)). The first tool, removing measurements from a selected rectangular area, works as shown in [Figure 2.5](#).

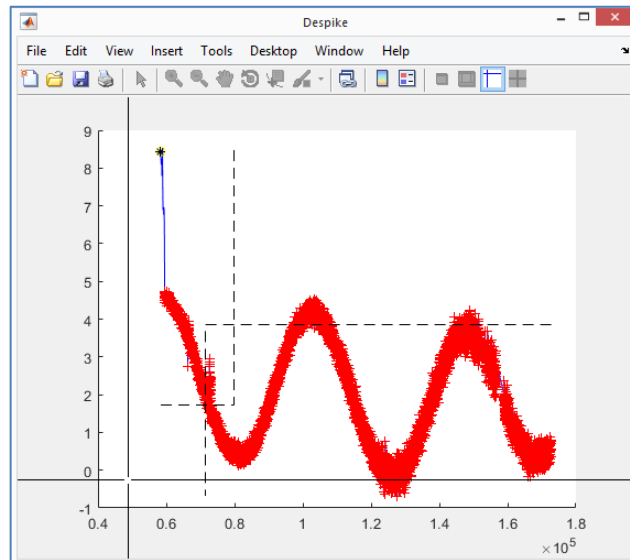


Figure 2.5 Window showing the tool removing measurements from a selected rectangular area

When operating the outlier removal window (by selecting a rectangular area), the following keys are used:

Clicking the icon – go to the “editing” mode (removing outliers);

Left mouse button – editing mode, set the first border of the selected area (the first part of the rectangle or the first point of the curve);

Right mouse button – editing mode, set the second border of the selected area (the second part of the rectangle or the second point of the curve);

Mouse wheel – editing mode, “delete” points in the selected area (the Y coordinate value is set to NaN);

z – editing mode, cancel the action;

x – editing mode, return the canceled action;

q – exit editing mode;

Backspace (or any key) – close the outlier removal window (the key is pressed after exiting the “editing” mode);

After exiting the “editing” mode (pressing the “q” key), one can change the scale of the graph or shift it by clicking on the corresponding icons (circled in blue in [Figure 2.6](#)).

After closing the outlier removal window, the graphs displaying the editing results are plotted in the MatLab graphics window ([Figure 2.6](#)). The window contains the following graphs:

Blue dots – original data;

Red line – smoothed data;

Blue line – the result of the smoothed data interpolation, where the values deleted in editing mode are “restored” by linear interpolation.

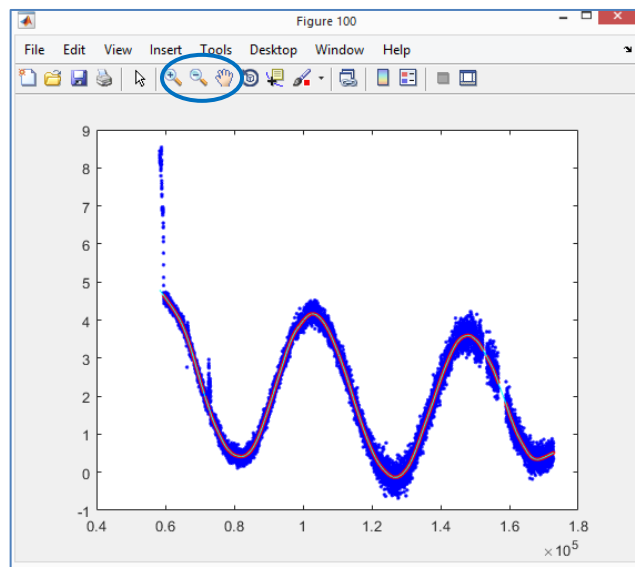


Figure 2.6 Window showing the graphs displaying the editing results

4) Saving processing results in DugInsight format is performed using a script command such as

```
>> {'TidesWrite','d:\x18_data\X18_raw_tides\out.txt',1};Tides4Soft;
```

where

'TidesWrite' – script command name;

'd:\x18_data\X18_raw_tides\out.txt' – file name for data output;

1 – number of the format corresponding to DugInsight.

Saving processing results in tdx (Silas) format is performed using a script command such as

```
>> {'TidesWrite','d:\202206_GeoXYZ_2\tide_innomar_orig4',2};Tides4Soft;
```

where

'TidesWrite' – script command name;

'd:\202206_GeoXYZ_2\tide_innomar_orig4' – folder name for data output;

2 – number of the format corresponding to tdx (Silas).

After running the command for execution, the tide data step is brought to a one minute interval and then the data is saved to a file. After executing the command, the script plots the graphs shown in [Figure 2.7](#):

Blue dots – original data;
Red dots – processed data in 1-minute steps.

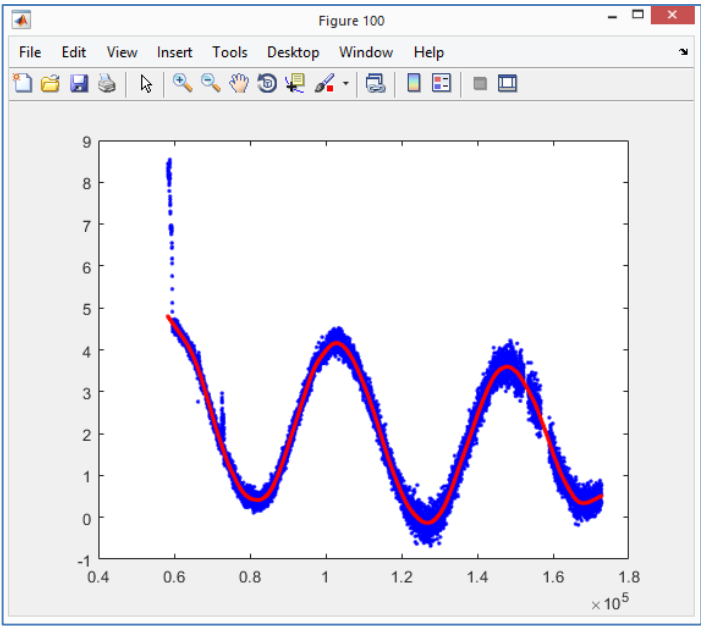


Figure 2.7 Original data (blue dots) and processed data interpolated to 1 minute interval (red dots)

For the “1” processed data format, all data is uploaded to one file. For the “2” processed data format (tdx - Silas), files are created for each day separately (**Figure 2.8**).

Name	Ext	Size	Date
..		<DIR>	14.08.2022 17:31
Tide_20220512	tdx	18 720	14.08.2022 18:19
Tide_20220513	tdx	5 395	14.08.2022 18:19

Figure 2.8 Contents of the folder containing Silas files format

The saved files can be used to load tide data into DugInsight or Silas software.

Appendix 1. Tides4Soft script code

```
%script Tides4Soft;
%Edit tides and transform from equipment data format to software data format (DugInsight, Silas)
%==== Input format 1 (USV data format) >>>
%#<Time> HHmmss.zzz, Latitude WGS-84 - COG - Latitude WGS-84, Longitude WGS-84 - COG - Longitude WGS-84, Height WGS-84 - COG - Height WGS-84, Easting - COG - Easting , Northing - COG - Northing, Height - COG - Height
%161248.237,5520.17000858,00134.00310710,54.775,580051.32,6100123.86,8.44
%161249.001,5520.17000347,00134.00310369,54.777,580051.33,6100123.85,8.44
%Input file names: X18_Tide_20220715_0000.log; X18_Tide_20220716_0000.log
%The date must be in the file name's position: fName(end-16:end-9)
%==== Input format 2 (USV data format) >>>
%#<Time> HHmmss.zzz,<Date> dd,<Date> MM,<Date> yyyy, Latitude WGS-84 - COG - Latitude WGS-84, Longitude WGS-84 - COG - Longitude WGS-84, Height WGS -84 - COG - Height WGS-84, Easting - COG - Easting, Northing - COG - Northing, Height - COG - Height
%000001.002,19,07,2022,5516.94135841,00117.68878421,46.600,608305.37,6127537.33,0.87
%000002.002,19,07,2022,5516.94081535,00117.68899275,46.623,608305.17,6127536.32,0.89
%Input file names: X18_Tide_20220715_0000.log; X18_Tide_20220716_0000.log
%==== Input format 3 >>>
%23:27:38,2.39
%23:28:08,2.60
%Input file names: tide_innomar_20220512.txt; tide_innomar_20220513.txt
%The date must be in the file name's position: fNameN(end-11:end-4)
%====Output format 1 (DugInsight data format) >>>
%1612:00 15 7 2022 4.74
%1613:00 15 7 2022 4.74
%====Output format 2 (Silas tdx format) >>>
%08:40 1.974
%08:41 2.007
%The files create for each day separately. Output file names example: Tide_20220512.tdx; Tide_20220513.tdx
%==== Commands example >>>
%{'TidesRead','d:\x18_data\X18_raw_tides\01\','1'};Tides4Soft; -- Read tides files from folder;
%{'TidesDecimate',10};Tides4Soft; -- Decimate tides for quick smoothing (use each 10 values only);
%{'TidesDespSmoothInterp',600};Tides4Soft; -- Manual despiking, smoothing in a window 600 points and interpolation for gaps;
%{'TidesWrite','d:\x18_data\X18_raw_tides\out.txt'};Tides4Soft; -- Create and write file for DugInsight with data-per-minute
%==== Keys for manual despiking (use icon to select edit tool) >>>
%LeftMouseButton - first selection element (first part of rectangle or first point at curve);
%RightMouseButton - second selection element (second part of rectangle or second point at curve);
%MiddleMouseButton - "delete" points in selected area (set to NaN Y-coordinate value);
%z - undo; x - redo; q - Exit from edit mode. Backspace (or any key) - finish manual despiking.

gKey=ans;
ifstrcmp(gKey{1},'TidesRead')%Read tides files from folder
%{'TidesRead','d:\x18_data\X18_raw_tides\01\','1'};Tides4Soft;
%{'TidesRead','d:\tide_innomar_orig3\','3'};Tides4Soft;
tryfName=gKey{2};catch,fName=input('Input folder name=');end;tryff=gKey{3};catch,ff=input('Input format type=');end
dz=dir(fName);dz([dz.isdir])=[];fName=[repmat(fName,length(dz),1) char(dz.name)];TD=[];
for nn=1:size(fName,1)
fNameN=deblank(fName(nn,:));
switchff
case1
```

```

[~,tmp]=gDataTxtRead(fNameN,1,7,'%f',',','\r\n');
GpsTime=gNavTime2Time('HMS2Sd',tmp(:,1));GpsDay= repmat(gNavTime2Time('YMD2Dx',str2num(fNameN(end-16:end-9))),size(GpsTime));
GpsLat=gNavAng2Ang('DM2D',tmp(:,2));GpsLon=gNavAng2Ang('DM2D',tmp(:,3));
TD0=struct('GpsDay',GpsDay,'GpsTime',GpsTime,'GpsLat',GpsLat,'GpsLon',GpsLon,'GpsHgtGeoid',tmp(:,4),'GpsE',tmp(:,5),'GpsN',tmp(:,6),'Tide',tmp(:,7));
    case2
[~,tmp]=gDataTxtRead(fNameN,1,10,'%f',',','\r\n');
GpsTime=gNavTime2Time('HMS2Sd',tmp(:,1));GpsDay=gNavTime2Time('YMD32Dx',tmp(:,4),tmp(:,3),tmp(:,2));
GpsLat=gNavAng2Ang('DM2D',tmp(:,5));GpsLon=gNavAng2Ang('DM2D',tmp(:,6));
TD0=struct('GpsDay',GpsDay,'GpsTime',GpsTime,'GpsLat',GpsLat,'GpsLon',GpsLon,'GpsHgtGeoid',tmp(:,7),'GpsE',tmp(:,8),'GpsN',tmp(:,9),'Tide',tmp(:,10));
    case3
fid=fopen(fNameN,'r');tmp=fscanf(fid,'%d:%d:%d,%f',[4 inf]);fclose(fid);
GpsTime=gNavTime2Time('HMS32Sd',tmp(1,:),tmp(2,:),tmp(3,:));GpsDay= repmat(gNavTime2Time('YMD2Dx',str2num(fNameN(end-11:end-4))),size(GpsTime));
TD0=struct('GpsDay',GpsDay,'GpsTime',GpsTime,'Tide',tmp(4,:));
    otherwise,error('Incorrect format number!');
    end
    if isempty(TD),TD=TD0;else,TD=gFieldsRowAppend(TD,TD0,numel(TD.GpsTime));end
    end
[TD.GpsDay,TD.GpsTime]=gNavDayCheck(TD.GpsDay,TD.GpsTime);
[~,TD.t]=gNavTime2Time('DxSd2DmS',TD.GpsDay,TD.GpsTime);L=[false diff(TD.t)==0];TD=gFieldsRowSet(TD,numel(TD.GpsTime),L,[]);
TD.Mask=true(size(TD.GpsTime));TD.TideSm=nan(size(TD.GpsTime));TD.TideSmInt=nan(size(TD.GpsTime));
figure(100);plot(TD.t,TD.Tide,'.b');
clearvars fName fid ff dz nn fNameN tmp GpsTime GPSDay GPSLat GPSLon TD0 L;
end
if strcmp(gKey{1},'TidesDecimate')%Use each N values only (decimate tides for quick smoothing)
    %{'TidesDecimate',10};Tides4Soft;
    try dK=gKey{2};catch,dK=input('Decimate step=');end
TD=gFieldsRowGet(TD,numel(TD.GpsTime),1:dK:numel(TD.GpsTime));
figure(100);plot(TD.t,TD.Tide,'.b');
clearvars dK;
end
if strcmp(gKey{1},'TidesDespSmoothInterp')%Despike manually, smoothing and interpolation
    %{'TidesDespSmoothInterp',600};Tides4Soft;
    try smW=gKey{2};catch,smW=input('Smooth window size=');end
a=figure('Name','Despike','NumberTitle','off');p=gMapPickHandleNan(TD.t,TD.Tide,a);pause;mask=~get(p,'UserData');close(a);TD.Mask(mask)=false;
TideSm=smooth(TD.t(TD.Mask),TD.Tide(TD.Mask),smW,'lowess';TD.TideSm(TD.Mask)=TideSm;
TD.TideSmInt=interp1(TD.t(TD.Mask),TD.TideSm(TD.Mask),TD.t,'spline','extrap');
figure(100);plot(TD.t,TD.Tide,'.b');hold on;plot(TD.t,TD.TideSm,'.r');plot(TD.t,TD.TideSmInt,'-c');hold off;
clearvars smW a p mask TideSm;
end
if strcmp(gKey{1},'TidesWrite')%Create and write file for software with value-per-minute
    %{'TidesWrite','d:\x18_data\x18_raw_tides\out.txt',1};Tides4Soft;
    %{'TidesWrite','d:\tide_innomar_orig4\',2};Tides4Soft;
    try fName=gKey{2};catch,fName=input('Output file or folder name=');end;try ff=gKey{3};catch,ff=input('Output format type=');end
t=fix(TD.t(1)./60)*60:60:TD.t(end);[~,di]=min(abs(t-TD.t(1)));d=TD .GpsDay(di);
TideSmInt=interp1(TD.t,TD.TideSmInt,t,'spline','extrap');figure(100);plot(TD.t,TD.Tide,'.b');hold on;plot(t,TideSmInt,'.r');
[GpsDay,GpsTime]=gNavTime2Time('DmS2DxSd',d,t);
    switch ff
        case1
[Y,Mm,D]=gNavTime2Time('Dx2YMD3',GpsDay);[H,M,S]=gNavTime2Time('Sd2HMS3',GpsTime);
fid=fopen(fName,'w');fprintf(fid,'%02d%02d:%02d\t%d\t%d\t%d\t%0.2f\t\t\r\n',[H;M;S;D;Mm;Y;TideSmInt]);fclose(fid);
        case2

```

```

L=[1 find(diff(GpsDay))+1 numel(GpsDay)+1];
for nn=1:numel(L)-1
[YMD]=gNavTime2Time('Dx2YMD',GpsDay(L(nn)));[H,M,S]=gNavTime2Time('Sd2HMS3',GpsTime(L(nn):L(nn+1)-1));
fid=fopen([fName'\Tide_'num2str(YMD)'.tdx'],'w');fprintf(fid,'%02d:%02d %0.3f\r\n',[H;M;TideSmInt(L(nn):L(nn+1)-1)]);fclose(fid);
end
otherwise,error('Incorrect format number!');
end
clearvars fName ff t di d TideSmInt GPSDay GpsTime Y mm D H M S L nn YMD fid;
end

% mail@ge0milib.com 08/13/2022

```