StaVel

A MATLAB/GNU Octave toolbox for the computation of the velocity of a network of GNSS stations starting from a database of coordinate time series

Release 1.0 – December 2022

User's guide

Giordano Teza
Department of Physics and Astronomy, Alma Mater Studiorum University of Bologna
Viale Berti Pichat, 6/2, I-40127 Bologna, Italy
giordano.teza@unibo.it, giordano.teza@gmail.com

Arianna Pesci
Istituto Nazionale di Geofisica e Vulcanologia, Bologna, Italy
Via Creti, 1, I-40128 Bologna, Italy
arianna.pesci@bo.ingv.it

Table of contents

1. Introduction 3
1.1 The StaVel toolbox 3
1.2 Tips for a quick approach to StaVel 4
1.3 Disclaimer 4
2. Toolbox installation and program running 5
3. Input and output data 6
3.1 Input data 6
3.2 Automatic importation of GNSS time series from a database (GetNevada function) 7
3.3 tsData object and input/output data 8
3.4 Other output files 12
4. Main StaVel functions 15
4.1 Choice of StaVel general options: geneOpts function 15
4.2 The main StaVel function: StaVelMain 17
4.3 Inspecting the coordinate time series: the InspectOffsetSearch function. 20
4.4 Removal of time series segments: the tsSegmentRemoval function 21
4.5 Common Mode Error filtering: the CMEstackFiltering function 22
4.6 Exporting the StaVel results to GridStrain: the geneGSFile function 24
5. Tutorial 26
5.1 Download of time series from Nevada Geodetic Laboratory 27
5.2 Generation of tsData objects and raw mom files 27
5.3 Download of data about offsets, inspection of the results, search of possible other offsets and generation of observation mom files 28
5.3.1 Direct use of InspectOffsetSearch. 29
5.3.2 Calling InspectOffsetSearch *from* StaVelMain 31
5.3.3 Removal of time series segments 32
5.4 Outlier recognition, trend computation and evaluation of results 33
5.5 Velocity exportation to GridStrain 34
6. List of StaVel MATLAB functions 36
7. References 38

1. Introduction

1.1 The `StaVel` toolbox

`StaVel` is a MATLAB™ toolbox, described in Teza et al. (2022), which is conceived for an easy and quick calculation of the velocities of some GNSS stations starting from the corresponding coordinate time series. It is the first component of a toolbox aimed at computing the strain rate field starting from the coordinate time series of some GNSS station. The second component, whose input data are the velocities obtained by means of `StaVel`, is `GridStrain` (for more information about the second component, please see the `GridStrain` user's guide. `GridStrain` is an improved version of grid_strain, described in Teza et al., 2008). The complete workflow (`StaVel` and `GridStrain`) is shown in Fig. 1.1. A first application of these toolboxes, which at that time were not yet complete, is shown in Meschis et al. (2022). `StaVel` is described in detail in this user's guide. For more information about `GridStrain`, please refer to the corresponding user's guide.

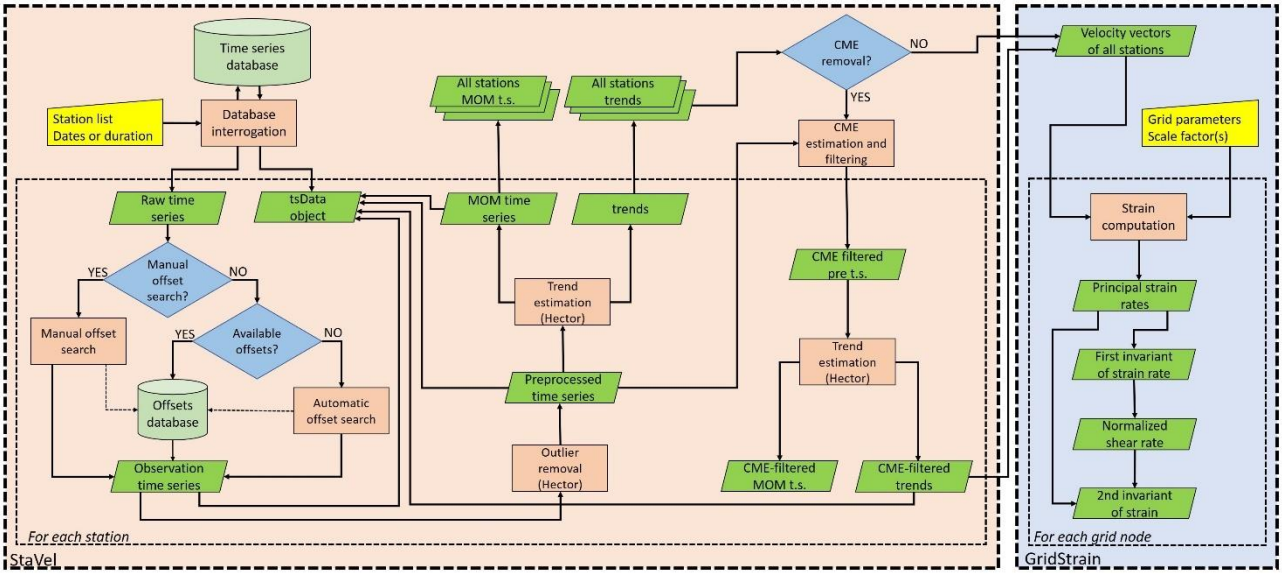

Figure 1.1. workflow of the whole process. `StaVel` is developed in order to carry out the steps inside the left, dashed rectangle (computations for each station). The computations carried out for each grid node, shown in the right, dotted rectangle, are carried out by means of `GridStrain`. The database interrogation is also carried out by means of `StaVel`.

These steps are carried out by means of `StaVel`:

- download of the time series of the selected GNSS stations from a data repository or, alternatively, access to a database of time series. Availability of time series provided by RINEX data processing is a precondition for the procedure described here;
- for each coordinate time series of each station:
  - offset recognition or offset importation from a database;
  - outlier recognition and modeling;
  - velocity computation based on Maximum Likelihood Estimation (MLE), carried out by means of the external software package Hector (Bos et al., 2013), automatically called by a MATLAB function (for other details about download and installation of Hector, please see Chapter 2);
  - (optional, if the estimation and removal of the common mode error, CME, is required):
    - × time series detrending, CME estimation and its removal from time series;
    - × MLE-based velocity computation with the CME-filtered time series.

The input and output data are described in the next chapters. An ASCII file which can directly managed by means of `GridStrain` can be generated by `StaVel` as the velocities are computed. These steps are carried out by means of `GridStrain` (please see the corresponding user's guide):

- strain rate field computation with the MLS method for one or more scale factors, including the computation of first and second invariant of the strain and of normalized shear;
- strain rate field visualization and interpretation.

The toolbox surely runs with all possible functionalities under MATLAB™ 2018a or later releases. Some actions aimed at allowing use with much older versions of MATLAB were however implemented. For example, whether a figure is an object or not is automatically verified.

The toolbox can also run under GNU Octave. In this case, some functions must be used by means of command line because the user interface controls could be incompatible with such a free package. These cases are highlighted both in this User'a Guide and in the corresponding function helps. An example of such a function is the main `StaVel` function, i.e. `StaVelMain`, described in Chapter 4. The corresponding help can be shown by typing

```
help StaVelMain
```

on the MATLAB Command Window (MCW), or the equivalent in Octave.

For any question, or also suggestion, please contact the authors (the Email addresses are shown in the cover of this user's guide).

1.2 Tips for a quick approach to `StaVel`

Given that this guide requires reading the article Teza et al. (2022), the quick approach is reading the very short Chapter 2, reading Subsection 3.1 about the input files and taking a quick look at Section 3.2, then moving onto Chapter 5 (Tutorial), reserving the rest of Chapter 3 and Chapter 4 for further information and, above all, to understand how to set the inputs of the various functions used. Moreover, it is necessary to read the facts highlighted at the conclusion of Subsection 4.2 about the main function of `StaVel`, i.e. `StaVelMain`. If the user is interested in common mode error (CME) filtering, he should refer to Subsection 4.5.

Some files are also provided for tutorial purposes. The user is invited to elaborate the data also taking into account what is highlighted in Subsection 4.2.

1.3 Disclaimer

This toolbox is free. The authors require that the use of this software be intended for scientific use only (no commercial use). If a publication whose results were obtained by means of this software is accepted for the publication, the toolbox `StaVel` and their authors must be cited.

Disclaimer: The authors of this toolbox accept no responsibility for damages resulting from the use of these products and makes no warranty or representation, either express or implied, including but not limited to, any implied warranty of merchantability or fitness for a particular purpose. This software is provided "AS IS", and the user assumes all risks when using it.

2. Toolbox installation and program running

The toolbox `StaVel` is contained in the zip file `StaVel.zip`. The files should be extracted and saved in a MATLAB directory whose name should be `StaVel`. For example, if the MATLAB work directory is:

`C:\Users\john.doe\Documents\MATLAB\`,

the toolbox can be saved and accessed with this path:

`C:\Users\john.doe\Documents\MATLAB\StaVel`.

Please note that, if MATLAB operates under an Unix environment, "/"must be used instead of "\".
If the directory `StaVel` is saved in the default directory on which MATLAB command window operates, the directory change must be carried out before the toolbox run. This change is obtained simply typing on the MCW:

```
cd StaVel
```

In order to call the program whichever is the current MCW directory, a startup file can be written by the user. For example, if the toolbox is placed in the directory '`C:\Users\john.doe\Documents\MATLAB\StaVel`', the rows

```
addpath 'C:\Users\john.doe\Documents\MATLAB\StaVel';
```

should be added to the `startup.m` script. If a file named `startup.m` is placed in the MATLAB work directory, it is automatically executed at each MATLAB start. In this way, all the defined search paths are automatically added at each MATLAB start and the directory changes to use the toolbox are unnecessary. The functionalities of these toolboxes do not depend on user's choice about the startup. More information about `addpath` function in a startup file can be found in http://www.mathworks.it/it/help/matlab/ref/ addpath.html.

Another option is the use of the Set Path dialog box, which appears by typing

```
pathtool
```

on the MCW or by selecting `Set Path` in `Home menu` of MATLAB desktop. The button `Add Folders` allows the choice of the folder and other buttons allow the choice of the folder order for the search of the files. Please see http://www.mathworks.it/it/help/matlab/matlab_env/using-the-matlab-search-path.html to have more information about the `Set Path` dialog box.

All provided functions are MATLAB `.m` files that can the opened and, if necessary, modified by the user. In this way, an expert user can modify, for example, the data saving options.

The toolbox calls an external free, open source software package, i.e. Hector (Bos et al., 2013). It is a high-performance, frequently upgraded package which runs under Linux. Hector must be separately downloaded and installed (current download page: http://segal.ubi.pt/hector/). Moreover, Hector runs under Linux regardless to the operating system (OS) used by MATLAB. If the OS is Windows, the automatic Hector call by MATLAB requires the Windows Subsystem for Linux 2 (WSL2). As Hector is installed, i.e. its executable files are placed in the chosen folder, this folder is managed by editing the corresponding row of `geneOpts` function (see Chapter 4).

Besides `StaVel` functions and scripts, some sample files are added for tutorial purposes in the file `StaVel.zip`.

3. Input and output data

3.1 Input data

Input data are coordinate time series, with time, East, North and Vertical, in this order, taken from Nevada Geodetic Laboratory (NGL) database (Blewitth et al., 2018) or from another similar database whose data are managed in the same way. The possible formats, correctly read by `StaVel`, are `tenv3`, `tenv` and `kenv`, which are described in http://geodesy.unr.edu/gps_timeseries/README_tenv3.txt for `tenv3` format, http://geodesy.unr.edu/gps_timeseries/README_tenv.txt for `tenv` format and, finally, http://geodesy.unr.edu/gps_timeseries/README_kenv.txt for the `kenv` format. All these files are ASCII files.

```
site YYMMMDD yyyy.yyyy __MJD week d reflon _e0(m) __east(m) ____n0(m) _north(m) u0(m) ____up(m) _ant(m) sig_e(m) sig_n(m) sig_u(m) __corr_en __corr_eu __corr_nu _latitude(deg) _longitude(deg) __height(m)
BOLG 22SEP04 2022.6749 59826 2226 0   11.4  -3437 -0.850160   4929405 -0.013671    99  0.613264  1.0350 0.000945 0.000876 0.002770 -0.074947  0.136729 -0.012292  44.5002195155 -348.6432215960    99.61326
BOLG 22SEP05 2022.6776 59827 2226 1   11.4  -3437 -0.851090   4929405 -0.008693    99  0.615643  1.0350 0.000946 0.000880 0.002783 -0.073537  0.135409 -0.007130  44.5002195607 -348.6432216070    99.61564
BOLG 22SEP06 2022.6804 59828 2226 2   11.4  -3437 -0.849414   4929405 -0.012486    99  0.620457  1.0350 0.000946 0.000881 0.002783 -0.073755  0.135017 -0.010965  44.5002195269 -348.6432215852    99.62046
BOLG 22SEP07 2022.6831 59829 2226 3   11.4  -3437 -0.850072   4929405 -0.009302    99  0.620006  1.0350 0.000952 0.000889 0.002811 -0.073169  0.131880 -0.022275  44.5002195559 -348.6432215928    99.62001
BOLG 22SEP08 2022.6858 59830 2226 4   11.4  -3437 -0.853486   4929405 -0.012452    99  0.609230  1.0350 0.001000 0.000929 0.002936 -0.075638  0.131966 -0.043496  44.5002195280 -348.6432216350    99.60923
BOLG 22SEP09 2022.6886 59831 2226 5   11.4  -3437 -0.849885   4929405 -0.011609    99  0.629226  1.0350 0.000986 0.000916 0.002895 -0.075019  0.134557 -0.024120  44.5002195359 -348.6432215890    99.62923
BOLG 22SEP10 2022.6913 59832 2226 6   11.4  -3437 -0.849376   4929405 -0.015859    99  0.608739  1.0350 0.000980 0.000905 0.002860 -0.076933  0.136109 -0.032276  44.5002194980 -348.6432215819    99.60874
```

Figure 3.1 Example of data from NGL database (`23-columns tenv3`)

An example of `tenv3` data is shown in Figure 3.1. In accordance with the instructions provided in the NVL site, a row should be read in this way (first row shown in Fig. 3.1):

Table 3.1 Meaning of the data of a tenv3 file

column	Sample value	Meaning
1	BOLG	station name (Bologna, Italy. Complete name: BOLG00ITA)
2	22SEP04	Date in the form YYMMMDD (4 September 2022)
3	2022.6749	decimal year in the form yyyy.yyyy
4	59826	modified Julian day (MJD)
5	2226	GPS week
6	0	day of GPS week
7	11.4	longitude (degrees) of reference meridian
8	-3437	eastings (m), integer portion (from ref. meridian)
9	-0.850160	eastings (m), fractional portion
10	4929405	northings (m), integer portion (from equator
11	-0.013671	northings (m), fractional portion
12	99	vertical (m), integer portion
13	0.613264	vertical (m), fractional portion
14	1.0350	antenna height (m) assumed from RINEX header
15	0.000945	east sigma (m)
16	0.000876	north sigma (m)
17	0.002770	vertical sigma (m)
18	-0.074947	east-north correlation coefficient
19	0.136729	east-vertical correlation coefficient
20	-0.012292	north-vertical correlation coefficient
21	44.5002195155	latitude(deg)
22	-348.6432215960	longitude(deg)
23	99.61326	height(m)

Please note that for some stations there are 20 columns instead of 23. The function for the data downloads automatically recognizes the kind of file and, in the case of a `tenv3`, the number of valid columns.

3.2 Automatic importation of GNSS time series from a database (`GetNevada` function)

Under the conditions that the input GNSS files are taken from the NGL database or from a database whose files are as in NGL, if a list of stations is defined, the corresponding `tenv3/tenv/kenv` files can be downloaded in an entirely automatic way. The function which carries out this is `GetNevada`, whose syntax is:

```
StatStatus = GetNevada(fileStations,PlateID,OutDir,AddName,Ext)
```

This function allows the time series download for the GNSS stations whose standard 4-length names (e.g. BOLG for the station BOLG00ITA, i.e. Bologna, Italy) are placed in the first column of the `.xlsx/.xls` file `fileStations`, from the second to the last row (the first row is the header) or on the only column of the ASCII file `fileStations`.
If `fileStations` is undefined or empty, it can be interactively chosen.

If `PlateID` is undefined, empty or is invalid (see below for the valid `PlateIDs`), an interactive box allows the choice of the kind of time series to be downloaded (`IGS14 ENV`, i.e. East-North-Vertical, `IGS14 XYZ`, plates `ENV`). In this case, a user interface control (Fig. 3.2) allows the choice of the plate for the velocity calculation; `IGS14 ENV` data can also be used for `StaVel` computations as well as for possible subsequent `GridStrain` computations. Moreover, for completeness `GetNevada` also allows the download of `IGS14 XYZ` data, i.e. time series in GNSS geocentric coordinates, but these data are unsuitable for `StaVel` computations.
The valid `PlateIDs` are (see also Fig.3.2):

```
'IGS14 - XYZ','IGS14 - ENV', 'AF','AN','AR','AU','BU','CA',
'CO','EU','IN','MA','NA','NB','NZ','OK','ON','PA','PM','PS',
'SA','SB','SC','SL','SO','SU','WL'.
```

The option valid `PlateID` is added in order to allow the use of this function under GNU Octave.

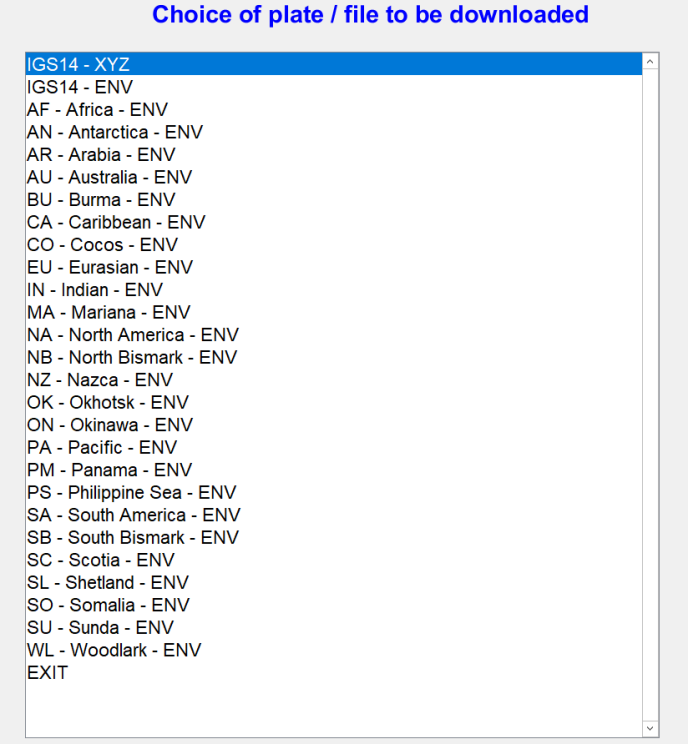

Figure 3.2 Choice of plate for `GetNevada` execution.

The successfully downloaded files have name

```
(OutDir)\(station name)(AddName)(Ext)    (Windows)
(OutDir)/(station name)(AddName)(Ext)    (Unix, MacOS)
```

were:

- If `OutDir` is undefined or empty, the downloaded files are placed in the current directory.
- If `AddName` is non-empty and its first character is not `'.'`, it is `AddName=['.' AddName]`.
  If `AddName` is empty, the possible name component is taken from the filename of Nevada database (for example, in the case of Eurasian plate the default `AddName` is `'.EU'`).
- If `Ext` is non-empty and `Ext(1)` is not `'.'`, it is `Ext=['.' Ext]`.
  If `Ext` is empty, the extension is taken from the filename of Nevada database (`.tenv3` for env files and `.txyz2` for XYZ files).

The output cell variable `StatStatus` is such that `StatStatus{k,1}` is the name of the k-th GNSS station and `StatStatus{k,2}` is true if the file download was successful and false elsewhere.

3.3 `tsData` object and input/output data

A MATLAB object of `tsData` class is defined for each continuous station managed by means of `StaVel`. Such an object is generated by using NGL data and is upgraded during the stages of velocity calculation (offset recognition, outlier removal, trend estimation). In particular, a `tsData` object is generated with the raw time series taken from a `tenv3`, `tenv` or `kenv` ASCII file; the function that recognizes the type of file and extracts these data from the file is a method of this object.

The script `tsData` is automatically called by `StaVelMain` function, which is described in Chapter 4. However, it can also be used by means of a command line. Possible syntaxes:

```
tsDataOut = tsData
tsDataOut = tsData(FILENA)
```

The output `tsData` object is generated with the data taken from a `tenv3`, `tenv` or `kenv .txt` ASCII file `FILENA` (the function that extracts these data is a method of this object). If `FILENA` is undefined or empty, the filename can be interactively managed by means of a combo box.

The properties of the generated object of `tsData` class are related to:

1) input time series (in this case, the data are managed by means of `tsData` methods);
2) time series processing (in this case, no `tsData` methods act on the data because the processing is based on Hector).

Complete list of properties related of input time series (please note that all these time series properties are defined only in the case of data taken from a `tenv3` file. In the case of `tenv` or `kenv` files some properties are undefined and, therefore, the corresponding values are empty):

Scalars/strings:

`statName`	station name (string)
`statLat`	station latitude (single value, degrees)
`statLon`	station longitude (single value, degrees)
`statHeight`	station heigh (single value, m)

Arrays:

dateS	date (string YYMMMDD)
dateyfrac	date (fractional year
MJD	modified julian date
GPSweek	GPS week
GPSday	GPS day
t	date (MATLAB serial form)
reflon	reference meridian longitude (degrees)
E0	eastings (m), integer portion (from ref. meridian
Ed	eastings (m), fractional portion
E	eastings (m), complete
N0	northings (m), integer portion (from equator)
Nd	northings (m), fractional portion
N	northings (m), complete
V0	vertical (m), integer portion
Vd	vertical (m), fractional portion
V	vertical (m), complete
antH	antenna height (m) assumed from RINEX header
sE	east sigma (m)
sN	north sigma (m)
sV	vertical sigma (m)
cEN	east-north correlation coefficient
cEV	east-vertical correlation coefficient
cNV	north-vertical correlation coefficient

Complete list of properties related to the time series processing and, therefore, are empty before such a processing:

cleanedE	cleaned time series after outlier removal - East
cleanedN	cleaned time series after outlier removal - North
cleanedV	cleaned time series after outlier removal - Vertical
offsetsE	offsets East
offsetsN	offsets North
offsetsV	offsets Vertical

The value of each property `offsetsE`, `offsetsN` and `offsetsV`, as the offset search is carried out, is an `offsets` struct variable (see below). The value of `offsetsOpts`:

- if the offsets recognition is carried out in an automatic way, it is `OptsGen.Offsets`, where `OptsGen` is the struct variable with the options for `StaVelMain` computations;
- if the offsets recognition is carried out in the manual way, it is the string `'Manual offsets recognition'`;
- if the offsets are taken from NGL database, it is the string `'Offsets taken from Nevada Geodetic Laboratory database'`;
- if the offsets are taken from another database, it is the string `'Offsets taken from file ....'`.

outliersE	outliers East
outliersN	outliers North

outliersV	outliers Vertical

The value of each property `outliersE`, `outliersN`, `outliersV`, as the outlier search is carried out, is an `outliers` struct variable (see below). The value of `outliersOpts` is `OptsGen.Outliers`, where `OptsGen` is as above.

estimatedTrendE	estimated trend East
estimatedTrendN	estimated trend North
estimatedTrendV	estimated trend Vertical
estimatedTrendOpts	estimated trend computation general options

The value of each property `estimatedTrendE`, `estimatedTrendN` and `estimatedTrendV`, as the trend estimation is carried out, is an `estimatedTrend` struct variable (see below). The value of `estimatedTrendOpts` is `OptsGen.Trend`, with `OptsGen` as above.

detrendedZeroMeanE	detrended zero mean East time series
detrendedZeroMeanN	detrended zero mean North time series
detrendedZeroMeanV	detrended zero mean Vertical time series

CMEfiltered	CME-filtered time series and data	
	CMEfiltered.t	CME filtered time series
	CMEfiltered.E	
	CMEfiltered.N	
	CMEfiltered.V	
	CMEfiltered.estimatedTrendE	`estimatedTrend` data for CME filtered time series
	CMEfiltered.estimatedTrendN	
	CMEfiltered.estimatedTrendV	
	CMEfiltered.Method	Method for CME filtering

The fields of these processing-related struct variables, also depending on the kind of struct variable, i.e. `offsets`, `outliers` or `estimatedTrend`, as well as on the user's choices, are taken from the JSON files generated by Hector. They can be

t1	initial time
t2	final time
N	actual number of days (gaps are excluded from count)
gap_percentage	percentage of gaps
K	number of estimated parameters (it is not kappa!)
Ln_L	minimum value of log-likelihood ln(L)
AIC	Akaike Information Criterion (AIC=2*k+2*ln(L))
BIC	Bayesian Information Criterion (BIC=k*ln(N)+2*ln(L))
BIC_tp	another BIC value (BIC_tp=k*ln(N/(2π))+2 ln(L))
BIC_c	another BIC value, with some extra-penalties
ln_det_I	
NoiseModel:	struct variable depending on the user's choices:

if white noise is chosen:
`NoiseModel.White`
`NoiseModel.White.sigma`
`NoiseModel.White.fraction`

if Powerlaw noise is chosen:
`NoiseModel.Powerlaw`

NoiseModel.Powerlaw.sigma
NoiseModel.Powerlaw.d
NoiseModel.Powerlaw.kappa
NoiseModel.Powerlaw.fraction

if GGM (Powerlaw) noise is chosen:
NoiseModel.GGM
NoiseModel.GGM.sigma
NoiseModel.GGM.d
NoiseModel.GGM.kappa
NoiseModel.GGM.x1_phi
NoiseModel.GGM.fraction

if FlickerGGM noise is chosen:
NoiseModel.FlickerGGM
NoiseModel.FlickerGGM.sigma
NoiseModel.FlickerGGM.d (0.5)
NoiseModel.FlickerGGM.kappa (1)
NoiseModel.FlickerGGM.x1_phi
NoiseModel.FlickerGGM.fraction

if ARMA model is chosen:
NoiseModel.ARMA
NoiseModel.ARMA.sigma
NoiseModel.ARMA.AR
NoiseModel.ARMA.MA
NoiseModel.ARMA.d
NoiseModel.ARMA.fraction

if ARFIMA model is chosen:
NoiseModel.ARFIMA
NoiseModel.ARFIMA.sigma
NoiseModel.ARFIMA.AR
NoiseModel.ARFIMA.MA
NoiseModel.ARFIMA.d
NoiseModel.ARFIMA.fraction

driving_noise	driving noise standard deviation (mm)
Trend	estimated trend (mm/y)
trend_sigma	estimated trend standard deviation (mm/y)
Sa_cos	yearly cos coefficient
Sa_cos_sigma	yearly cos coefficient standard deviation
Sa_sin	yearly sin coefficient
Sa_sin_sigma	yearly sin coefficient standard deviation
Sa_amplitude	yearly oscillation amplitude
Sa_amplitude_sigma	yearly oscillation amplitude standard deviation
Sa_phase	yearly oscillation phase
Sa_phase_sigma	yearly oscillation phase standard deviation
Ssa_cos	half-yearly cos coefficient
Ssa_cos_sigma	half-yearly cos coefficient standard deviation
Ssa_sin	half-yearly sin coefficient

Ssa_sin_sigma	half-yearly sin coefficient standard deviation
Ssa_amplitude	half-yearly oscillation amplitude
Ssa_amplitude_sigma	half-yearly oscillation amplitude standard deviation
Ssa_phase	half-yearly oscillation phase
Ssa_phase_sigma	half-yearly oscillation phase standard deviation
jumps_epochs	
jumps_sizes	Jump parameters
jumps_sigmas	

For example, the trend value and standard deviation for the East component of the `tsData` object named `tsDataBOLG` are `tsDataBOLG.estimatedTrendE.trend` and `tsDataBOLG.estimatedTrendE.trend_sigma` respectively. If power law is among the chosen noise models, the corresponding `k` is `tsDataBOLG.estimatedTrendE.NoiseModel.Powerlaw.kappa` (if power law is not a chosen noise model, the field is empty).

Since all `.m` files are available to the user, he/she can make the toolbox compatible with other kinds of input files by editing the `tsData.m` file and adding other methods to `tsData` object in order to allow a successful data reading.

Note: a `tsData` object can be read as an object only if it is loaded in an appropriate way. If such a file is not correctly accessed, it is not read as a `tsData` object but such a warning message appears: "`Warning: Variable 'ts' originally saved as a tsData cannot be instantiated as an object and will be read in as a uint32.`" In order to solve this problem and allow a correct access to a `tsData` file whichever is its folded, the function `tsDataFileIn` can be used. Its syntax is:

```
[tsDataOut,IT]=tsDataFileIn(tsDataIn)
```

This function checks if the input variable `tsDataIn` is a `tsData` object.
If this test is passed, `tsDataOut=tsDataIn` and `IT` is true.
If the test is not passed and `tsDataIn` is a char variable, the function checks if it exists a MATLAB `.mat` file with this name and carrying a `tsData` object (see below for more details about this file).
If `tsDataIn` is undefined or empty, the MATLAB .mat file carrying a `tsData` object can be managed in an interactive way.
If the `tsData` object should be extracted by a file:

- if a single variable is carried by the file, it must necessarily be a `tsData` object;
- if two or more variables are carried by the file, one and only one of them must be a `tsData` object.

If the file loading is successful, the extracted `tsData` object is `tsDataOut` and `IT` is true. If no a valid `tsData` object is extracted, `tsDataOut` is empty, `IT` is false and a warning message is shown.

3.4 Other output files

The above-described `tsData` object represents both the input data, downloaded from NGL or other compatible database, and the results of the processing. During the execution of the main function of `StaVel`, which call Hector several times, in addition to updating the `tsData` object for each station, some essential ASCII files are generated to allow Hector to proceed in the

various calculation steps. These ASCII files are stored in some folders (if these folders do not exist at the time of initialization of `StaVelMain` program, they are automatically generated).

For each processed station, as the calculation steps progress, files are generated and placed in these folders, whose names can be managed by acting on `geneOpts` function, described in the next chapter:

- Folder of downloaded ASCII ENV files (`tenv3`, `tenv` or `kenv` files). Such a folder is filled by means of the above described `GetNevada` function. A possible folder name is `tenv_EU` if the EU plate is considered;
- Folder of `tsData` objects, generated by means of `tsData.m` called by `StaVelMain`, stored as `.mat` files and upgraded at each calculations step. The default folder name is `ts_files`;
- Folder of raw mom files. A mom (MJD-Observations-Model) file is an ASCII file with the time series in the first two columns (MJD dates in the first column and data in the second one. An optional third column has the modeled data). The extensions always is `.mom`, regardless to the number of columns (an example of mom file is shown in Fig. 3.3). This folder is filled by `StaVelMain` function. The default folder name is `raw_files`;
- Folder of control files for Hector. They are ASCII files, whose extension is `.ctl`, generated by `StaVelMain`, for each step in which Hector is involved, on the basis of options initially managed by means of `geneOpts` function. The default folder filename is `ctl_files`;
- Folder of observation mom files. They are generated as the offset recognition or acquisition from a database is carried out, regardless to the fact that they are provided by Hector, called by `StaVelMain`, or by other functions called by `StaVelMain`. The file header of an observation mom file (or of a mom file subsequently generated) provides information about possible offsets (see Figure 3.3). The default folder filename is `obs_files`;
- Folder of JSON files, generated by Hector as the calculation steps progress. The JSON files are read by `StaVelMain` in order to upgrade the `tsData` objects. The default folder filename is `JSON_files`;
- Folder of Hector output files, generated by `StaVelMain` on the basis of the Hector output. At present, these files are not used for `tsData` file upgrade, but they provide a little bit more information with respect to JSON files and could be manually read by the user, if necessary. The default folder filename is `out_files`;
- Folder of pre-processed mom files, generated by Hector as the outlier recognition is carried out. The default folder filename is `pre_files`;
- Folder of results mom files, generated by Hector as the trend estimation is carried out. These mom files have three columns, where the third one has the modeled time series (they are the only three-column files; in other cases where Hector provides three-column files, the third is removed from `StaVelMain` in order to prevent problems in the next calculation steps. The default folder filename is `mom_files`.

For more information about control and mom files, please see the Hector User's Guide. Information can also be taken by a look to the `StaVelMain.m` file.

Final note for use by expert users potentially interested in customizing the `StaVel` toolbox. As regards the management of the offsets in the functions of the toolbox, the binary-like convention shown in Table 3.2 is used (please also note that the dates are in this case expressed in the MJD form). In the functions that make use of it (for example, `InspOffsetMom`), the corresponding matrices are indicated with `MosMom`. Since the toolbox can also generate `.neu`

files that might be used by CATS (Williams, 2008), the convention presented in Table 3.3 is also used; the corresponding matrices, in which the dates are presented in fractional year form, are indicated with `Mos`. For example, an offset in the East component only has code 3 with the mom convention, and code 2 with the neu convention (see the help of `writeNeuMom` and `MosMom2MosNeu` for more information).

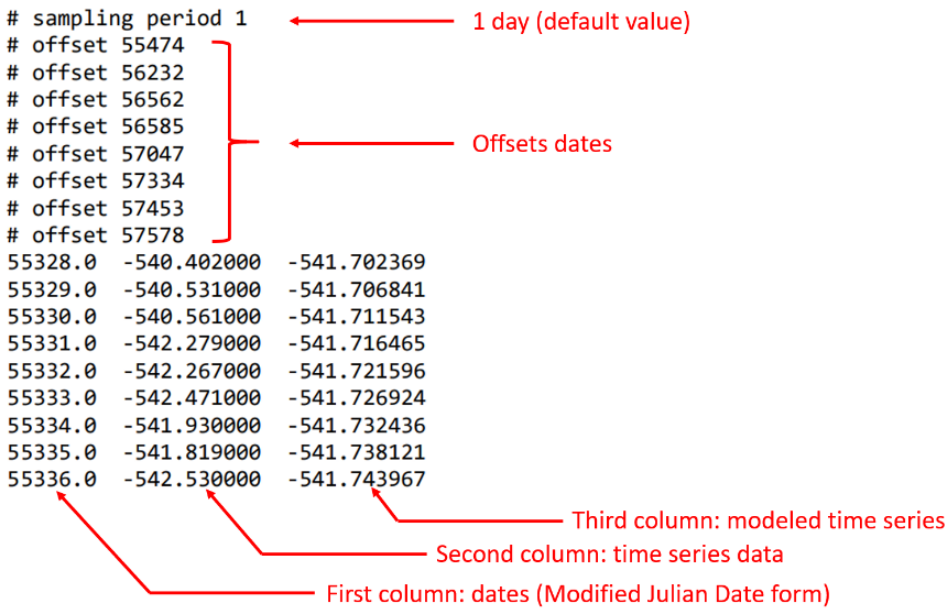

Figure 3.3 Example of a three-columns mom file.

Table 3.2 Binary convention for offsets (mom files)

East	North	Vertical	Code
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
1	0	1	5
1	1	0	6
1	1	1	7

Table 3.3 Binary convention for offsets (neu files)

North	East	Vertical	Code
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
1	0	1	5
1	1	0	6
1	1	1	7

4. Main StaVel functions

From the point of view of the execution of all actions that can be implemented within the toolbox, the main MATLAB functions are:

- GetNevada, for the download of coordinate time series from Nevada Geodetic Laboratory (NGL) o from another similar database, already described in the previous chapter;
- geneOpts, for the management of StaVel options (models to be used for data processing, choice of folders and filename common parts);
- StaVelMain, main StaVel function, which is the heart of the toolbox;
- geneGSfile, for the generation of an ASCII file compatible with GridStrain.

These functions call, where necessary, other specifically developed functions.

4.1 Choice of StaVel general options: geneOpts function

The function geneOpts allows the generation of the option struct variable Opts which is used by StaVelMain and by other functions called by StaVelMain at the various stages of the computation. The syntax is:

```
Opts=geneOpts(Components,OffsetNM,TrendNM,…
      CommonFolder,CommonAdd,Lmin,Lmax)
```

Input arguments:

- Components: vector of components, where 0 means East, 1 means North and 2 means Vertical. For example, to analyze horizontal components only, the required choice is Components=[0 1].
  If Components is undefined or empty, Components=[0 1 2] is used.
- OffsetNM: Noise models to be used for offset recognition. Allowed values are the strings:

FNWN	flicker noise and white noise
PLWN	power law and white noise
RWFNWN	random walk noise, flicker noise and white noise
WN	white noise

  If OffsetNM is undefined, empty of is no an allowed string, the default choice WN is used.
- TrendNM: Noise models to be used for trend computation. The argument must be either an allowed string (see below the list of allowed strings) for the case of a single noise model or a cell variable whose elements are allowed strings for the case of multiple noise models. The allowed strings are:

WN	white noise
GGM	generalized Gauss-Markov
fGGM	k-fixed GGM
FN	flicker noise
FNGGM	flicker noise GGM
RW	random walk
RWGGM	random walk GGM
PL	power law
PLGGM	power law GGM

MT	Matern
VA	varying annual
VSA	varying semi-annual
AR1	ARMA

If TrendNM is undefined, empty, or no valid strings are in TrendNM, the default choice TrendNM={'WN','PL'} is used.

- CommonFolder is an optional general folder, subfolder of the main StaVel folder, that will contain all the folders generated/filled by StaVelMain. If CommonFolder is undefined or empty, the general folder is the main StaVel folder.
- CommonAdd is the common string to be added to the file name. Options:
  - it is a char (included the empty char "): it is used for all the generated files. For example, the generated raw file for a station NAME is './raw_files/NAME.EU.mom' if EU is CommonAdd and './raw_files' is the raw files folder. If CommonAdd is ", in the same conditions the raw file is './raw_files/NAME.mom'.
  - it is not a char (including [], i.e. empty vector, but not empty char). No a common string is used and each generated file can have a specific added string (these strings are defined by editing the present function).
- Lmin and Lmax are the optional minimum and maximum lengths of the GNSS time series, expressed in years. If Lmin (Lmax) is undefined or empty, no minimum length (no maximum length) is considered. Good practices require Lmin ≥ 4.5 (years).

Example of Opts struct variable:

```
      dirEnv    './Syr/tenv_AF'
       dirTs    './Syr/ts_files'
      dirRaw    './Syr/raw_files'
      dirObs    './Syr/obs_files'
      dirPre    './Syr/pre_files'
      dirMom    './Syr/mom_files'
      dirCtl    './Syr/ctl_files'
      dirOut    './Syr/out_files'
     dirJSON    './Syr/JSON_files'
   dirHector    '/mnt/c/GNSS/hector'
     extTenv    '.tenv3'
  Components    [0 1 2]
      AddNeu    '.EU'
       AddTs    '.EU'
      AddRaw    '.EU'
      AddObs    '.EU'
      AddPre    '.EU'
      AddMom    '.EU'
      AddOut    '.EU'
     AddJSON    '.EU'
        Lmin    4.5000
        Lmax    []
      Offset    [1×1 struct]
     Outlier    [1×1 struct]
       Trend    [1×1 struct]
```

In order not to make it necessary to introduce a complex user interface that is not compatible with GNU Octave, the fields relating to the names of the folders of the files generated or modified directly and indirectly by `StaVelMain` are managed by acting on the program lines of the `geneOpts` function (it is recommended to always have a backup copy of this function in order to facilitate any future changes). Please open the file `geneOpts.m` to see how this function is structured and how it can be modified.

The values of the `Offset`, `Outlier` and `Trend` fields are themselves struct variables, established on the basis of the options chosen by the input options of `geneOpt` and the choices on `foldernames`. These values act on generation of the corresponding Hector control (`.ctl`) files by `StaVelMain`. An example of `Opts.Trend` is the struct variable:

```
              Interpolate    'no'
            PhysicalUnit     'mm'
             ScaleFactor     1
          Seasonalsignal     'yes'
     Halfseasonalsignal      'yes'
                    JSON     'yes'
       estimateoffsets       'yes'
           NoiseModels       'White Powerlaw'
```

(meaning of these choices: differential coordinate data expressed in mm, to provide velocities in mm/y; no interpolation for the dates where data are missing; annual and semi-annual signals are modeled; the offsets at the already known offset dates are estimated[1]; the noise models are white noise and power law noise).

4.2 The main `StaVel` function: `StaVelMain`

The main component of `StaVel` toolbox is the function `StaVelMain.m`, whose syntax is:

```
COut=StaVelMain(fileStations,OptAct,OptsGen)
```

This function, which calls the external package Hector, can perform all computations necessary to obtain the velocities of the GNSS stations listed in the Excel or ASCII file `fileStations`. Each station must be represented by its standard 4-character name. In the case of an Excel file, the station names must be placed in the first column (no more than the first column is read). In the case of an ASCII file, only a column is admitted.
If `fileStations` is undefined or empty, the filename can be interactively managed.

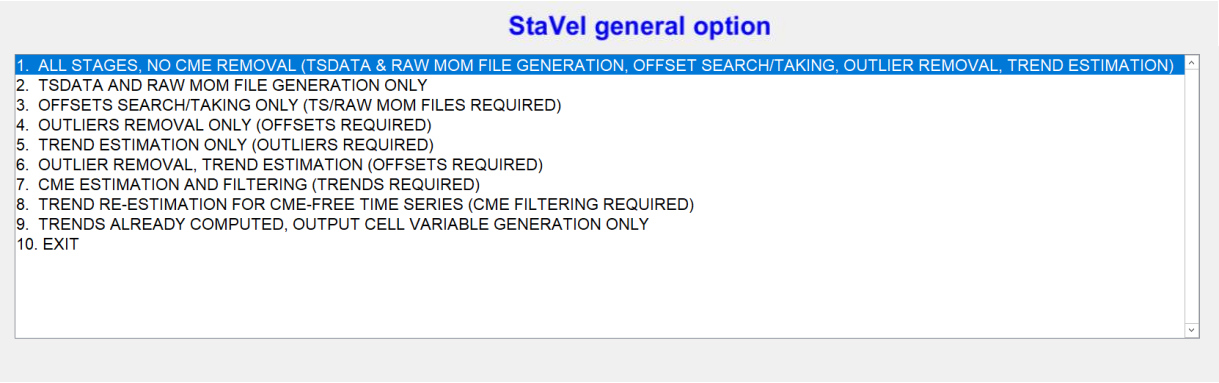

Figure 4.1 Interactive choice of General initial option for `StaVel`.

[1] Please note that “estimate offsets” does not mean “recognize the offsets”, i.e. recognize the offsets dates, but “Estimate the offsets at the dates reported in the headers of the mom files”.

The second input argument, `OptAct`, allows the choice of the operation to be carried out. There are several options for such an argument. If `OptAct` is undefined, empty or outside 1:10, the choice can be carried out in an interactive way by means of a user interface (Fig. 4.1).

If `OptAct` is a scalar in the range 1:10, the corresponding action is carried out.
If `OptAct` is 1 or 3 (chosen as an input argument or by means of interactive choice), the option about the offset management (`OptOff`) can be interactively chosen with the user interface shown in Fig. 4.2.
If `OptAct` is 7 (chosen as an input argument or by means of interactive choice), the option about the CME estimation method can be interactively chosen with the user interface in Fig. 4.3 (see Subsection 4.5 for more information about CME estimation and filtering).
If `OptAct` is 9 (chosen as an input argument or by means of interactive choice) and CME-filtered data are available, the use can generate the output variable `COut` on the basis or non-filtered or filtered data. The option can be interactively chosen (Fig. 4.4).

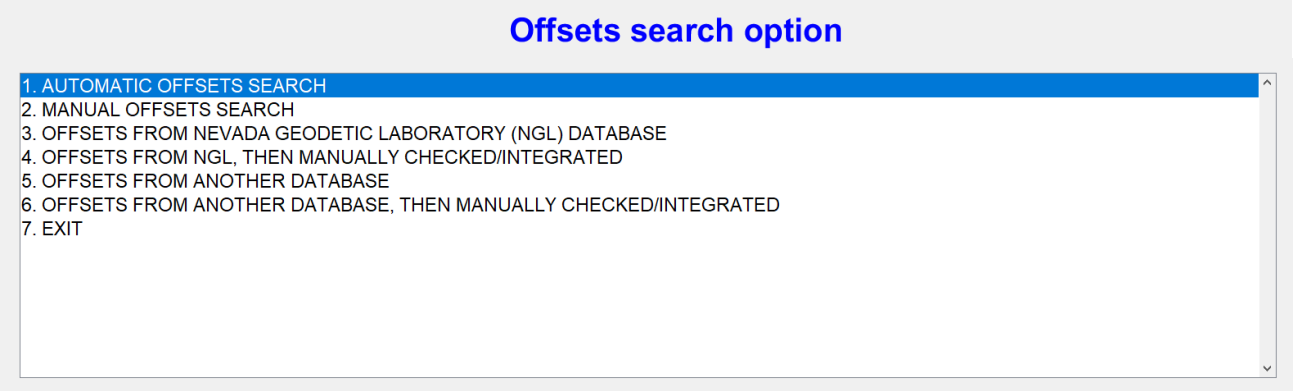

Figure 4.2 Choice of the modality for search/use of offsets if `OptAct` is 1 or 3.

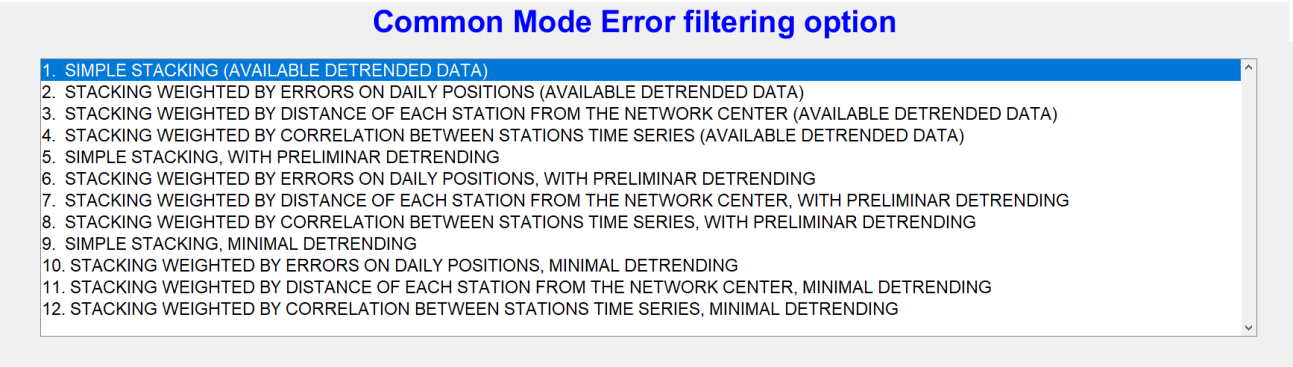

Figure 4.3 Choice of the CME filtering option if `OptAct` is 7 (for more information, please see Subsection 4.5).

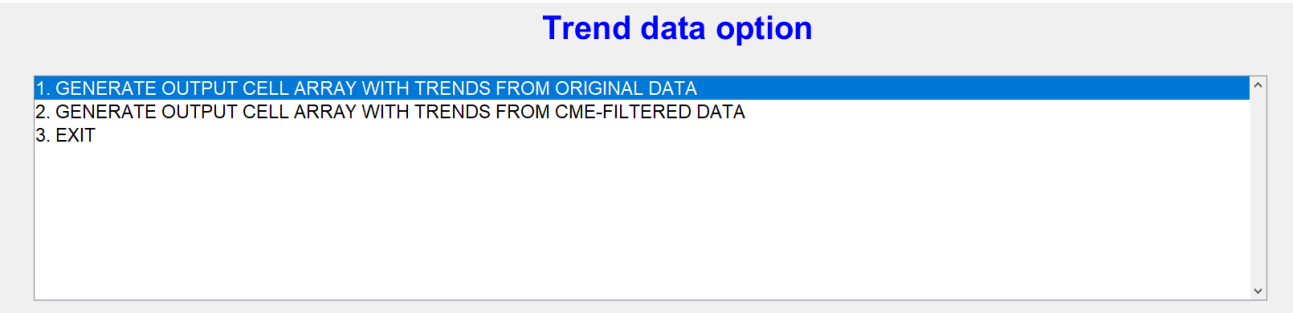

Figure 4.4 Choice for the generation of the output cell variable if `OptAct` is 9. Clearly, if the option 1 (respectively 9) is chosen, trends obtained from the original data (CME-filtered data) must be available.

If OptAct is a 2-length vector, OptAct(1) is the option 1-10 as above, and OptAct(2) is OptOff, i.e. the option for the offset management (1-6 to manage offsets, 7 to exit from StaVelMain), provided that OptAct(1) is 1 or 3, the option for the CME estimation, provided that OptAct(1) is 7, or the option for the choice of trends from original or CME-filtered data, provided that OptAct(1) is 9 and these trends exist. Clearly, the second element, i.e. OptAct(2), is actually used only if OptAct(1) is 1, 3, 7 or 9. The vector option about OptAct is introduced to allow the use of StaVelMain under GNU Octave.

OptsGen is the struct variable with the general StaVel options, generated by means of the already described geneOpts function. If OptsGen is undefined or empty, the name of the file carrying such a struct variable can be interactively managed. In this case, the file must have a field whose name is either Opts or OptsGen.

The output variable COut is non-empty if OptAct is 1, 5, 6, 8 or 9, empty elsewhere. If COut is non-empty, it is a N-by-11 cell array, where N is the number of managed stations, and the columns are:

```
[station name, initial time, final time,
station latitude, station longitude, station height (m a.s.l.),
estimated trend East, estimated trend East SD,
estimated trend North, estimated trend North SD,
estimated trend Vertical, estimated trend Vertical SD]
```

Such an output COut can be used as input of geneGSFile, described in Subsection 4.3.

If OptAct is 8, the trends are related to CME-filtered time series.

It is important to highlight these facts about `StaVelMain`:

1) Although, in principle, all operations could be implemented in fully automatic way (OptsAct 1), this is not recommended. It is recommended to at first generate the tsData and mom files for each station (option OptsAct 2), then proceed to search for the offsets (automatic or manual), or to download and verify them (OptsAct 3), when the offsets are at least verified, proceed to the recognition of the outliers and, finally, to the calculation of the trends (OptsAct 4, 5 and 6, based on the user's wishes to proceed step-by-step or to perform the two steps automatically). It is therefore recommended to start StaVelMain several times. This is also shown in the short tutorial (Chapter 5);

2) If the user decides to use StaVelMain for OptsAct after 1 ("ALL STAGES"), it is essential that the previous operations have already been performed or that in any case the necessary data are already present. This is highlighted in the user interface. For example, in order to perform OptsAct 4 and 6, the offset dates, if any, must already be known and the necessary files in the Hector observation mom files folder (and those in the preprocessed mom files folder for OptsAct 6) must already be available;

3) It is not strictly necessary that the computer is online during the implementation of the StaVelMain calculations and of the external functions and programs controlled by StaVelMain (obviously, the computer had to be online at the time of the download of the files implemented by the GetNevada function). However, if the data taken from the database is of the type tenv, kenv or tenv3 with 20 columns, the computer must be online at the time the tsData objects are generated (OptsAct 1 and 2) in order to access the longitude, latitude and ellipsoidal height data (these data are instead taken directly from the file in case of tenv3 file with 23 columns);

4) StaVelMain operates for each station considered in the list placed in filenaStations, that is, all the operations selected through OptsAct are all performed, one after the other, for each station. The generation of the tsData objects, of the raw mom files and the management of the offsets, if selected, is carried out for all the components at the same time; the subsequent phases, if selected, are instead implemented component by component, before moving on to the next station;

5) If a user wants to repeat the calculation by changing one or more parameters, he/she can redefine the options using geneOpts, save a copy of the starting data in the necessary folders and proceed with a new StaVelMain session. For example, if the user intends to recalculate the trend on the basis of different noise models or periodic signals keeping the previously processed data, he/she can copy the preprocessed mom files into a new folder, whose name will be entered in OptsGen by acting on geneOpts, and then repeat the calculation by choosing OptsAct 5 or 6;

6) The folders defined by means of geneOpts, if not existing, are generated by StaVelMain. If a folder already exists, StaVelMain does not alter its contents except when writing the files. In case of files with the same name, StaVelMain rewrites without any warning. It should be noted that the user can repeat the analysis for some stations, or introduce new stations, simply by acting on fileStations. For example, he/she can consider a new file fileStations1, with only the names of the stations to recalculate and/or add, and proceed. He/she then redefine a third file fileStations2 with all stations to implement OptsAct=9 and then export the data for GridStrain. An example of this possibility is given in Chapter 5 for tutorial purposes;

7) The CME estimation and removal is carried out by means of CMEstackFiltering, which can be called by StaVelMain is OptAct 7 is selected. However, the direct use of CMEstackFiltering offers more options with respect to the use of such a function called by StaVelMain. Please see Subsection 4.5 for more information.

4.3 Inspecting the coordinate time series: the `InspectOffsetSearch` function.

The time series inspection is an important stage of GNSS data analysis. The function InspectOffsetSearch allows the inspection of several time series and the manual check/search of the possible corresponding offsets.
This function can be used in two different ways:

- Direct use as an independent function (see below the corresponding command line);
- Use as a function called by StaVelMain. This function is called by StaVelMain if OptAct is 1 or 3 (or is a vector whose first element is 1 or 3) and the corresponding OptOff (interactively chosen or defined as second element of OptAct if this is a vector) is 4 or 6 (for the other options about OptOff, the offsets are managed in a non-interactive manner).

The syntax for the direct use is:

```
InspectOffsetSearch(fileStations,OptsGen,OptPre)
```

This function allows the visual inspection of GNSS time series on the stations listed in fileStations in accordance with the StaVelMain options stored in the struct variable OptsGen. This function requires the availability of tsData files related to the listed stations.
Input variables:

- `fileStations` carries the list of GNSS stations, represented by their standard 4-character names. This file can be either an Excel or an ASCII file. In the case of Excel file, the station names must be placed in the first column (no more than the first column is read). In the case of an ASCII file, only a column is admitted. If `filenaStations` is undefined or empty, the filename can be managed in interactive way. Let `statName` be the string array with the station names obtained from the loaded file.
- `OptsGen` is a struct variable generated by means of `geneOpts` function. For each station, the input `tsData` file is searched of the basis of the station name, `statName(k)`, and information carried by `OptsGen` in accordance with:

```
filets=fullfile(OptsGen.dirTs,…
    [statName(k) OptsGen.AddTs '.mat']).
```

  The `tsData` file is upgraded according to the detected offsets. The corresponding mom observation files, whose names are stored in `OptsGen`, are also generated and/or upgraded. If `OptsGen` is undefined or empty, the `.mat` file with such a struct variable can be interactively managed.
- If `OptPre` is undefined, empty or false, no possible already available data about offsets are used, i.e. the offsets search is carried out starting from scratch. If `OptPre` is true, for each station the offsets search starts from the offsets resulting from the corresponding `tsData` object.

It is important to underline that the `InspectOffsetSearch` function allows not only to inspect the time series and to identify/verify the dates of the offsets, but also to identify any anomalous segments of them to be taken into consideration or, if necessary, to be removed. For an example of use of such a function, including the acquisition of information about the removal of time series segments, please see the tutorial Subsection 5.3.

4.4 Removal of time series segments: the `tsSegmentRemoval` function

In some cases, it is necessary to remove one or more segments of a time series because they are affected by excessive dispersion, progressive changes in position not related to crustal kinematics or other reasons. The `tsSegmentRemoval` function allows you to do this. It should be noted that this is a low-level automation function in order to avoid sudden deletion of data. The corresponding syntax is:

```
tsSegmentRemoval(statName,OptsGen,MdateRemove,OptStr)
```

This function allows the removal of one or more segments from the time series related to the station `statName`, on the basis of the general `StaVel` options represented by `OptsGen` struct variable.

The input argument `MdateRemove` is a `Nrem-by-2` matrix, where `Nrem` is the number of time series segments to be removed, `MdateRemove(k,1)` and `MdateRemove(k,2)` are the lower and higher date limit, expressed in MATLAB serial form, for the k-th segment.

The `tsData` file and the related mom files are automatically upgraded. If the input argument `OptStr` is `'Obs'`, the observation mom file is upgraded. If `OptStr` is `'Raw'`, the raw mom file is upgraded. If `OptStr` is undefined, empty or is not `'Obs'` or `'Raw'`, the default value `'Obs'` is used.

For an example of use of `tsSegmentRemoval` please see the tutorial Subsection 5.3.

4.5 Common Mode Error filtering: the `CMEstackFiltering` function

The function `CMEstackFiltering` performs the CME evaluation and filtering. It can be called by means of `StaVelMain` function or can be used in autonomous way. At present, these CME estimation procedures are implemented in CMEstackFiltering:

- Stacking (Wdowinski et al., 1997);
- Weighted Stacking Filtering Method (Nicolaidis, 2002);
- Distance Weighted Filtering Method (He et al., 2020);
- Correlation Weighted Stacking Filtering Method (Tian and Shen, 2011).

The syntax of this function is:

```
CME=CMEstackFiltering(fileStations,OptsGen,OptStack,Ncomp,Nstaz)
```

This function carries out the CME filtering of the time series of the GNSS stations listed in the Excel or ASCII file fileStations on the basis of the chosen stacking approach (see below about possible options).

Warning: since CME filtering requires detrended, zero-mean time series, trend estimations must already be available!

If data about detrended, zero mean time series are available in a `tsData` file, they can be used (see below about the input variable OptStack). It they are unavailable, or the user wants to re-compute them, the detrended, zero-mean time series are computed and the corresponding `tsData` files are upgraded. As the calculations are completed, the `tsData` files of involved stations are automatically upgraded and the corresponding preprocessed mom files are generated.

The output `CME` is a struct variable whose fields are:

`deltat`	vector `[t1 t2]` of initial and final time (serial form)
`Ivv`	`nt-by-ns` matrix, where `nt` is the length of the daily time vector `t1->t2` and `ns` is the number of stations, where `Ivv(k,h)` is true if at the time `t(k)` there are valid values for all components of the station `h`
`CMEE`	`nt-by-1` vector (options `OptStack 1-3`, `5-7`, see below) or `nt-by-ns` matrix (`OptStack 4` or `8`) of common mode error for East component
`CMEN`	as above, North component
`CMEV`	as above, Vertical component

If the computations cannot be carried out, `CME=[]` is returned.

Each station must be represented by its standard 4-character name in `fileStation`. In the case of an Excel file, the station names must be placed in the first column (no more than the first column is read). In the case of an ASCII file, only a column is admitted. If `fileStations` is undefined or empty, the filename can be interactively managed.

`OptsGen` is the struct variable with the general `StaVel` options, generated by means of `geneOpts` function. If `OptsGen` is undefined or empty, the name of the file carrying such a struct variable can be interactively managed. In this case, the file must have a field whose name is either `Opts` or `OptsGen`.

`OptStack` is the option on stacking. The admitted values are:

1. simple stacking;
2. stacking weighted by errors on daily positions;
3. stacking weighted by distance of each station from the network center;

4. stacking weighted by correlation between stations time series;
   (note: if `OptStack` *options 1-4 are selected, for each station the existence of detrended zero-mean time series is checked. If such a time series exists, it is used for the CME estimation. If such a time series does not exist instead, it is computed like the case of options 5-8)*
5. simple stacking, with recalculation of possible existing detrended zero-mean time series;
6. stacking weighted by errors on daily positions, with recalculation of possible existing detrended zero-mean time series;
7. stacking weighted by distance of each station from the network center, with recalculation of possible existing detrended zero-mean time series;
8. stacking weighted by correlation between stations time series, with recalculation of possible existing detrended zero-mean time series;
9. simple stacking with minimal detrending, i.e. detrending based on simple least square fitting to a straight line (in this case, the Hector-based trend is not used);
10. stacking weighted by errors on daily positions with minimal detrending;
11. stacking weighted by distance of each station from the network center with minimal detrending;
12. stacking weighted by correlation between stations time series with minimal detrending.

If `OptStack` is undefined, empty or is not an admitted value, it can be interactively chosen.

`Ncomp` is the number of components. Allowed values: `2` (E-N), `3` (E-N-V). If `Ncomp` is undefined, empty or is not an allowed value, `Ncomp=3` is used.

`Nstaz` is the minimum number of stations to be used for stacking at each time. If, at a time `t(k)`, less than `Nstaz` stations have data, `CMEE(k)=0`, `CMEN(k)=0`, `CMEV(k)=0`, where `CMEE=CME.CMEE`, `CMEN=CME.CMEN` and `CMEV=CME.CMEV`. `Nstaz` must be at least 3, i.e. at a time `t(k)`, data from at least 3 stations are necessary to allow the computation of CME. If `Nstaz` is undefined, empty or lower than 3, the default `Nstaz=3` is used. If `Nstaz` is `Inf` or higher than the number of stations, `CMEs` are non-zero only at the times where all stations have data.

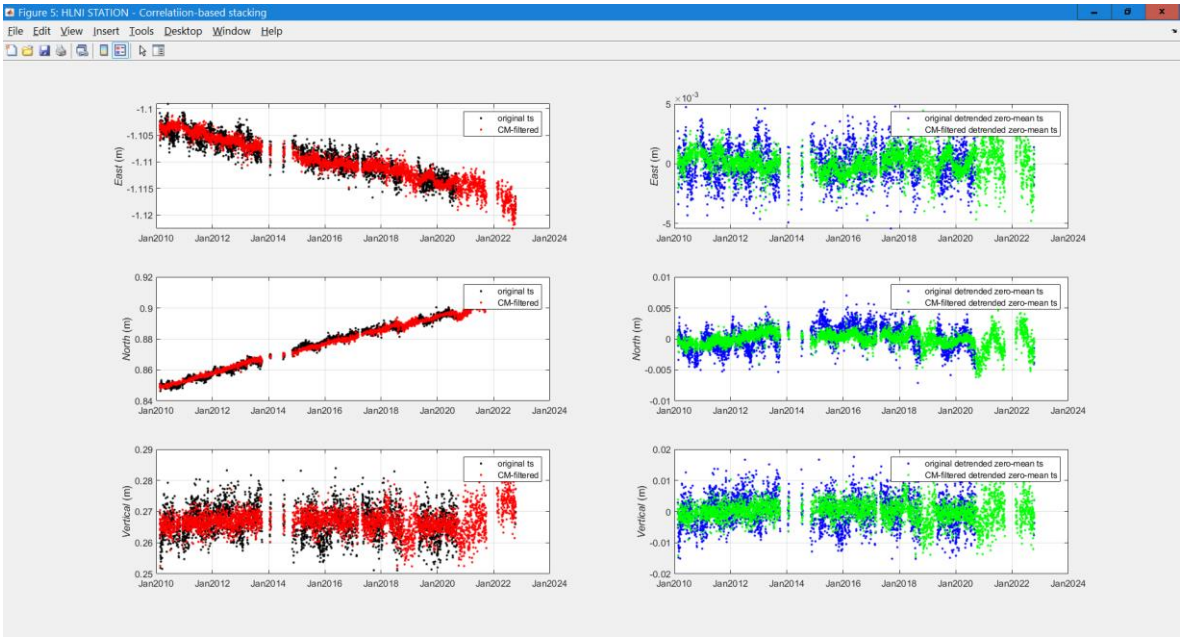

Figure 4.5 Example of plot provided by `CMEstackFiltering` function. The fact that the variance after CME filtering is lower than the one before filtering should be noted. The original and filtered time series coincide after October 2020 because data from no more than two stations are available (at a given time, the CME can be computed only if data from at least three stations are available).

Examples of plots provided by `CMEstackFiltering` function, with the option "correlation-based stacking" are shown in Figs. 4.4 and 4.5 for two stations belonging to the sample dataset included in `StaVel` toolbox. In this specific case, the limited area of the stations does not actually recommend filtering, which is only presented here for tutorial purposes. Instead, it is better to estimate and remove the CME, operating at the level of homogeneous areas, when there are many stations in a very large area. Once the homogeneous areas have been identified, each of them presumably affected by a specific CME, the user can proceed with the modeling and filtering of the time series of the stations within each of them (a trial-and-error approach could be necessary). In this way, the effects of the different CMEs which could give rise to huge uncertainties on the speed estimates will at least be attenuated.

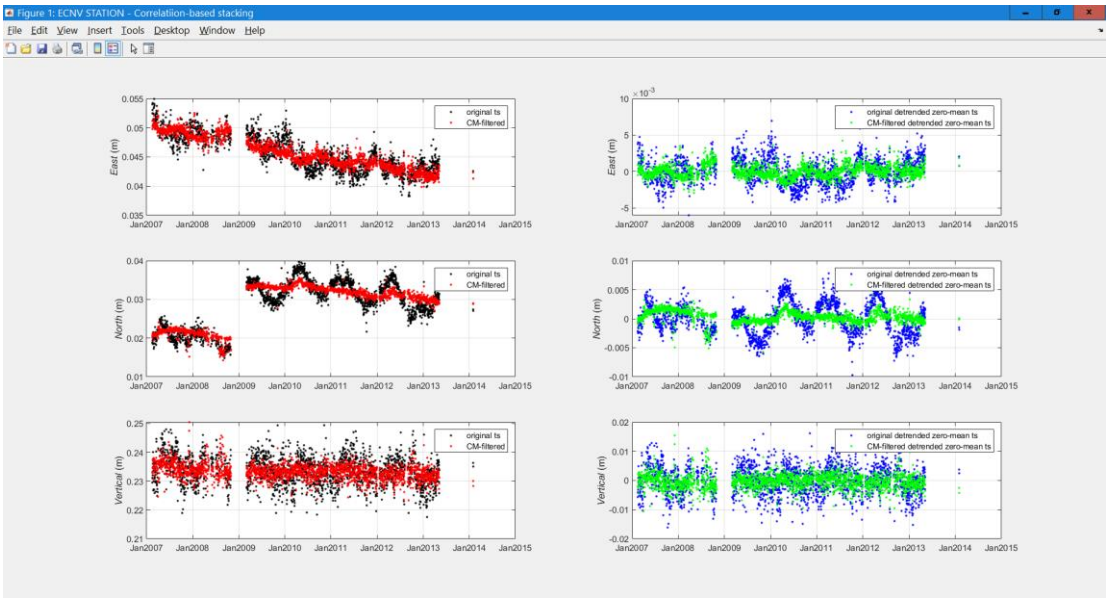

Figure 4.6 Another example of plot provided by `CMEstackFiltering` function.

4.6 Exporting the `StaVel` results to `GridStrain`: the `geneGSFile` function

As the velocity computations are completed, the results can be exported to `GridStrain` toolbox by means of geneGSFile function, whose syntax is:

```
CCompl=geneGSFile(CellIn,FilenaOut,N,DTY,UTMZzone,NameID)
```

This function generates the .txt file `FilenaOut` suitable for `GridStrain` (2D or 2.5D) on the basis of:

- Input cell data `CellIn` provided by a previous `StaVelMain` session.
  Options:
  - `CellIn` is a cell variable available in MATLAB command window (e.g. a `COut` variable provided by a `StaVelMain` session);
  - `CellIn` is a string (including or not including the extension `'.mat'`) with the filename of the MATLAB file which contains the cell variable. The file must contain no more than a field, regardless to its name;
  - `CellIn` is undefined or empty. In this case, the filename is interactively managed.
- Name of the output ASCII file `filenaOut`. If `filenaOut` is undefined or empty, the filename can be interactively managed.

- Number of coordinates to be taken into account. Valid choices are 2 (horizontal components only) and 3 (xyz coordinates). If `N` is invalid, undefined or empty, `N=2` is used.
- Time series minimum duration, in years, `dty`. The stations whose time series are shorter than `dty` are excluded from the output file. If `dty` is undefined or empty, no a minimum duration is considered.
- UTM zone expressed by the string `UTMzone` (e.g. `'33N'`). If `UTMzone` is undefined or empty, the default UTM zone which corresponds to the WGS84 coordinates is chosen.
- `NameID`. If `NameID` is undefined, empty or false, the station IDs are the station progressive numbers. If `NameID` is true, the station IDs are the station names.

5. Tutorial

In order to show how `StaVel` operates, the case of 5 GNSS stations, located in Southern Italy, for which data is available in the NGL database, is considered here. To do this, two files are added to the toolbox for tutorial purposes, i.e.:

- `SampleStations.txt`, with a list of stations;
- `SampleOpts.mat`, with a struct variable, named `OptsGen`, with the options for `StaVelMain`.

The user can use these files directly and can also modify them at will to vary the conditions and, above all, to change, if necessary, the link to the Hector folder.
Obviously, before using the `StaVel` toolbox, Hector must be installed (download page: http://segal.ubi.pt/hector/).

Note: the variable `OptsGen` saved in `SampleOpts` refers to the case in which the Hector's executables are placed in C:\GNSS\hector on a Windows machine equipped with WSL2 ("Windows subsystem for Linux"), so that these executables can be reached from MATLAB with the command `'/mnt/c/GNSS/hector/'` (note that Hector operates under Linux). The corresponding value must be modified if the path is different and, to generate other struct `OptsGen` variables, it is also necessary to act on the corresponding line of `geneOpts.m` (line 139 in the current version of the toolbox).

These steps are carried out within the tutorial:

- download of time series from NGL;
- generation of `tsData` and mom raw files through a first session of `StaVelMain`;
- use of the data available in NGL for the management of offsets with a second `StaVelMain` session;
- inspection of the time series and identification of possible stations that requires manual recognition of offsets and recognition of these offsets by using `InspectOffsetSearch` function;
- Search for outliers and trend estimation with a fourth use of `StaVelMain`;
- export of results.

Before the listed steps, the sample files are briefly discussed.
The file `SampleStations.txt` has a column with the names of five GNSS continuous stations located in Southern Italy. Information about the station ECNV can be found at http://geodesy.unr.edu/NGLStationPages/stations/ECNV.sta, and similar for the other stations:

```
ECNV
EDEN
EIIV
GALF
HLNI
```

An Excel file with the same information could also be used (in this case, more than a column are allowed, but only the first column is used).

As for the other sample file, the command line on MCW

```
load SampleOpts
```

loads the file SampleOpts and adds the struct variable OptsGen to the current workspace. A look to OptsGen shows that all folders will be placed inside the general folder Tutorial, subfolder of StaVel. The command lines required to obtain such a file where:

```
OptsGen=geneOpts([0 1],'PLWN',{'PL','WN'},'Tutorial','.EU',4.5);
save SaveOpts OptsGen
```

The user could modify the variable acting e.g. on noise models. Please see the description of geneOpts function in Chapter 4 or type help geneOpts on MCW for more information. In this case, horizontal components are considered ([0 1]), power law and white noise are the noise models used for all the Hector-based computations ('PLWN' for offset recognition and {'PL', 'WN'} for trend estimation), the general folder is 'Tutorial', all the file names contain '.EU' and the time series having at least 4.5 y length are considered.

5.1 Download of time series from Nevada Geodetic Laboratory

The first step is the download of time series from NGL. To carry out this, the command is:

```
status=GetNevada('SampleStations.txt','EU','Tutorial/tenv_files');
```

which reads the station names from the file Sample.Station.txt, uses the data related to the Eurasian plate ('EU'; if the second input argument is undefined or empty, the user interface shown in Fig. 3.2 appears to allow the choice of the plate. Clearly, the plate choice must be coherent, i.e. the corresponding NGL data must be available), and places the files in the folder (subfolder of the StaVel main folder) Tutorial/tenv_EU (if this folder does not exist, it is made by GetNevada function). For example, the name of the file related to the first station in a Windows machine is

```
Tutorial\ECNV.EU.tenv3
```

where '.EU' comes from the fact that EU (Eurasia) is the chosen plate and no an AddName input argument is defined and '.tenv3' is the default choice for the extension if the corresponding input argument is undefined as in this case (see Subsection 3.2 or type help GetNevada on the MCW for more information). Clearly, the input arguments of GetNevada function must be coherent with the StaVel general options about the Env files and the corresponding folder.
As GetNevada runs, these messages are shown by MCW:

```
Download of file http://geodesy.unr.edu/gps_timeseries/tenv3/plates/EU/ECNV.EU.tenv3 in progress...
Download of file http://geodesy.unr.edu/gps_timeseries/tenv3/plates/EU/ECNV.EU.tenv3 completed
Download of file http://geodesy.unr.edu/gps_timeseries/tenv3/plates/EU/EDEN.EU.tenv3 in progress...
Download of file http://geodesy.unr.edu/gps_timeseries/tenv3/plates/EU/EDEN.EU.tenv3 completed
Download of file http://geodesy.unr.edu/gps_timeseries/tenv3/plates/EU/EIIV.EU.tenv3 in progress...
Download of file http://geodesy.unr.edu/gps_timeseries/tenv3/plates/EU/EIIV.EU.tenv3 completed
Download of file http://geodesy.unr.edu/gps_timeseries/tenv3/plates/EU/GALF.EU.tenv3 in progress...
Download of file http://geodesy.unr.edu/gps_timeseries/tenv3/plates/EU/GALF.EU.tenv3 completed
Download of file http://geodesy.unr.edu/gps_timeseries/tenv3/plates/EU/HLNI.EU.tenv3 in progress...
Download of file http://geodesy.unr.edu/gps_timeseries/tenv3/plates/EU/HLNI.EU.tenv3 completed
```

The output variable status also shows that all the required data are successfully downloaded. Please note that, is not already available, the folder Tutorial, with the subfolder tenv_files, is generated.

5.2 Generation of `tsData` objects and raw mom files

The second step involves StaVelMain function. The option for the actions to be chosen in this case, i.e. OptsAct, is 2, to be entered in the command line or to be chosen with the user interface shown in Fig. 4.1 (GNU Octave users must necessarily use the command line with all the correctly defined input arguments). The two possibilities correspond to these command lines:

```
COut=StaVelMain('SampleStations.txt',2,OptsGen);
COut=StaVelMain('SampleStations.txt',[],OptsGen);
```

Please note that this command line

```
COut=StaVelMain;
```

is also valid in MATLAB (not in GNU Octave). In this case, all the input arguments are managed in an interactive way. Another example of valid command line is `COut= StaVelMain([],[],OptsGen)`. Other combinations of input arguments are also valid (please see Subsection 4.2 or type `help StaVelMain` for more information). Clearly, if `OptsGen` is among the input arguments, such a variable must be available in the workspace because a `geneOpts` session was carried out or the file `SampleOpts` was loaded before the specific `StaVelMain` session.

Regardless to the used command line, for the choice `OptAct=2` the `tsData` objects and the raw mom files are generated for each station listed in `SampleStations.txt` and placed in the folders chosen by means of `OptsGen`. Since no trends are estimated at this stage, the output variable `COut` is empty.

5.3 Download of data about offsets, inspection of the results, search of possible other offsets and generation of observation mom files

The NGL database also has data about offsets. Please note that in such a database the offsets are called steps. The reference page accessed by the function `NevadaAllOffset`, automatically called by `StaVelMain`, is http://geodesy.unr.edu/NGLStationPages/steps.txt. This tutorial shows the case of using the offsets available in the NGL database. The command line complete and also compatible with GNU Octave is:

```
COut=StaVelMain('SampleStations.txt',[3 3],OptsGen);
```

where `[3 3]` indicates Offsets recognition/taking (“first 3”) and download of offsets by NGL database without subsequent manual offset check (“second 3”); see below for the manual offset check by using `StaVelMain`. MATLAB users can interactively select the corresponding options. As `StaVelMain` runs with such a command, this is progressively shown on the MCW:

```
Offsets database loading in progress...
... Offsets database loaded
Station ECNV, no offsets from database
Station EDEN, offsets from database taken
Station EIIV, offsets from database taken
Station GALF, no offsets from database
Station HLNI, offsets from database taken
```

The `tsData` objects are upgraded and the observation mom files are generated. The message `no offsets from database` means that no offsets are known to NGL for the specific station. Also in this case, `COut` is empty.

It is undoubtedly advisable, and indeed necessary, to verify the validity of the information about offsets. For this reason, the next step is the inspection of the results, to be implemented using the `InspectOffsetSearch` function (please see Subsection 4.3 or type `help InspectOffsetSearch` on the MCW for more information about his function). As stated in Subsection 4.3, `InspectOffsetSearch` can the directly used, or called by StaVelMain.

5.3.1 Direct use of `InspectOffsetSearch`.

In the specific case, the command line is

```
InspectOffsetSearch('SampleStations.txt',OptsGen,1)
```

where the third input argument (`1`) allows the use of already available information on offsets, i.e., in this case, the information downloaded from NGL database (`true` is also a valid input for this option. If the user wants to exclude the available information, the third input variable should be `0`, `false`, `[]` or also undefined).
As the function runs, Fig. 5.1 appears (the figure name has the station name, in this case ECNV, and the whole time span of available data). A drop-down menu allows the choice of the zoom parameters to allow the choice of the offset dates, if this is required. The possible options are

```
ZOOM ON 14 DAYS TIME SERIES AROUND A SELECTED TIME
ZOOM ON 1 MONTH TIME SERIES AROUND A SELECTED TIME
ZOOM ON 6 MONTH TIME SERIES AROUND A SELECTED TIME
ZOOM ON ONE YEAR TIME SERIES AROUND A SELECTED TIME
FREE CHOICE OF TIME LIMITS
EXIT
```

If no offsets seem to exist, the better solution is `EXIT`. Please note that, when `EXIT` is pressed after the selection of one or more offsets, the specific `tsData` file is updated with the selected `offsetsE`, `offsetsN`, `offsetsV` properties. Similarly, the observation mom files are updated. If the observation mom files do not exist, they are generated.

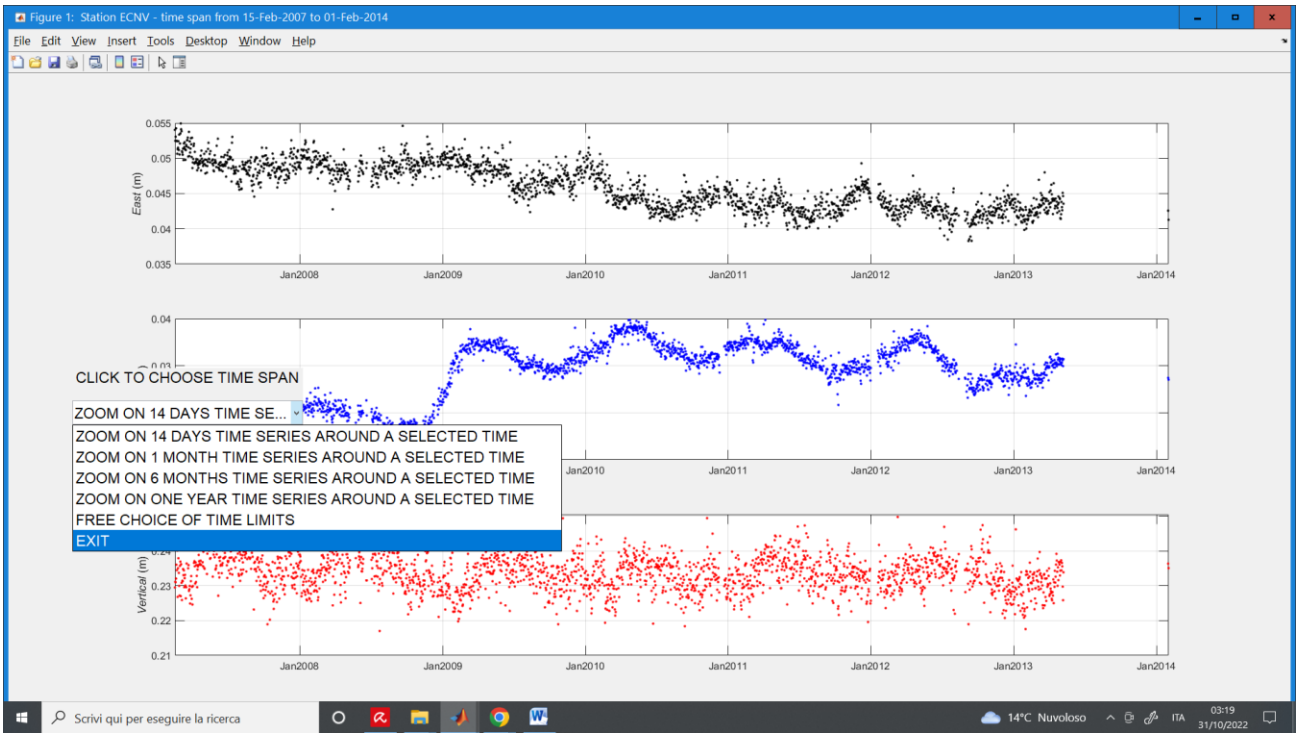

Figure 5.1 Time series related to the first station of the lists and related drop-down menu.

In the case of ECNV (first station of the sample list), no offsets are in the NGL database (if some such offsets existed, they would be shown in Fig. 5.1 in a similar way to that of Fig. 5.4 in the case of EIIV. However, an anomalous trend is observed for the N component around January 2009. This suggests the suppression of the corresponding segment of the time series of all the components and the introduction of an offset, limited to the N component, or the start time, or the termination of the

deleted segment. To do this, the user first zooms on the time series, for example using the ass option, thus obtaining Fig. 5.2, whose drop-down menu offers various options:

```
OFFSET E-N-V (7)
OFFSET E-N (6)
OFFSET E-V (5)
OFFSET E ONLY (4)
OFFSET N-V (3)
OFFSET N ONLY (2)
OFFSET V ONLY (7)
DATE ACQUISITION ONLY
NO OFFSET, NO DATE
```

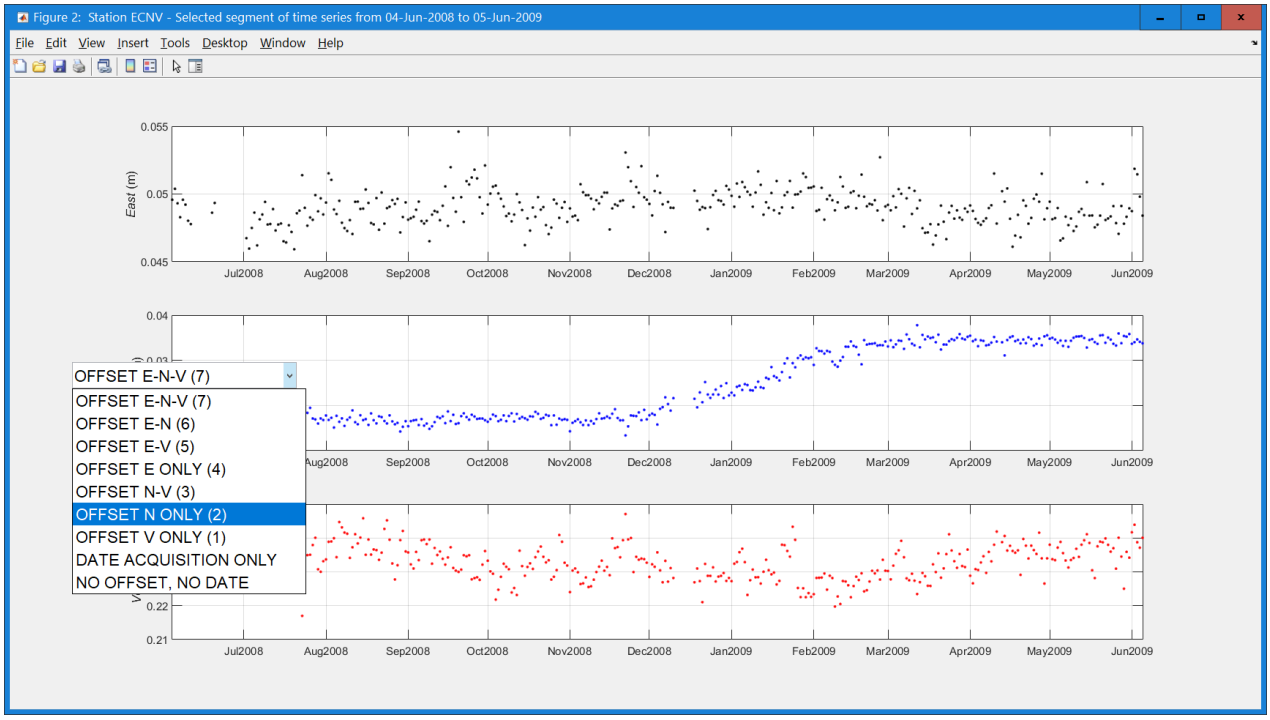

Figure 5.2 Zoom on time series and selection of the operation to be carried out.

The command `DATE ACQUISITION ONLY` allows the selection of the initial and final date of the segment to be removed (please note that the removal is not carried out in this stage; another function must be invoked, as shown below). As a selection is done, the zoom must be repeated on a figure like Fig. 5.1, where offsets, if selected, are shown as vertical lines. To allow escape is necessary, the command `NO OFFSET, NO DATE` closes the zoom figure without any change. In the specific case two dates are selected (Fig. 5.3) and an offset is recognized at the first of these dates. Therefore, three consecutive zooms are necessary.

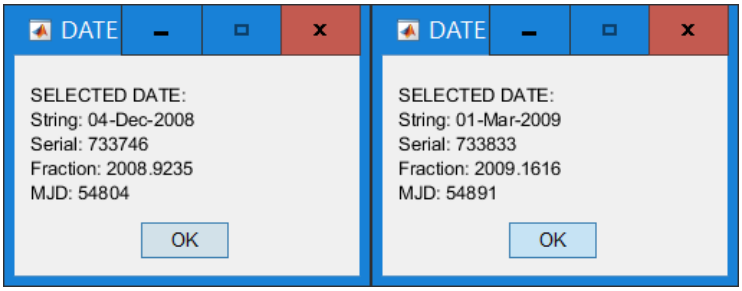

Figure 5.3 Selection of initial and final date of the time series segment to be removed

As EXIT is selected, the time series of the second station, in this case EDEN, is shown, and so on. The EDEN time series does not require any action. The time series of the third station, EIIV, is more interesting (Fig. 5.4). An offset is provided by NGL database and anomalies can be seen around January 2019.

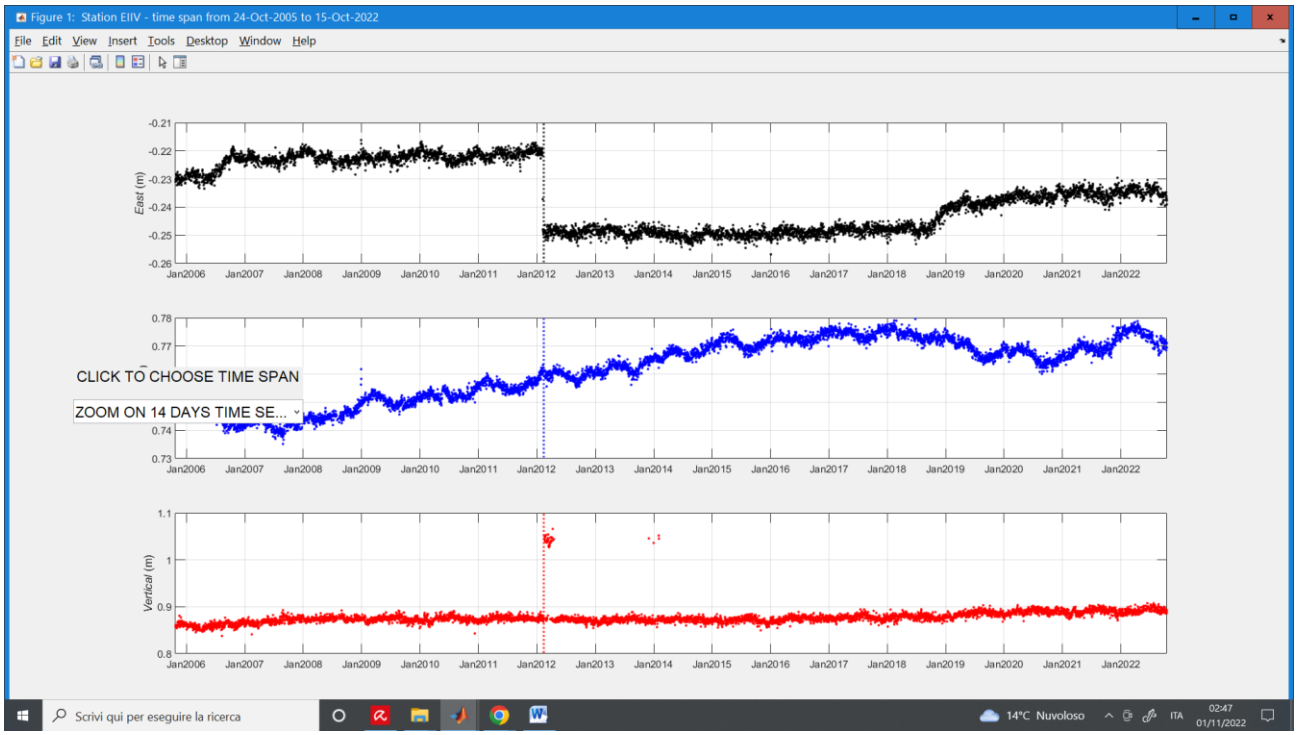

Figure 5.4 Time series of a station where an offset is already known.

5.3.2 Calling `InspectOffsetSearch` *from* `StaVelMain`

The same result can be obtained by calling `InspectOffsetSearch` from `StaVelMain`. It the user wants to ignore possible already known offsets (such a choice corresponds to `InspectOffsetSearch` with third input argument undefined, empty of false), the command line should be

```
COut=StaVelMain('SampleStations.txt',[3 1],OptsGen);
COut=StaVelMain('SampleStations.txt',[3 2],OptsGen);
COut=StaVelMain('SampleStations.txt',[3 3],OptsGen);
COut=StaVelMain('SampleStations.txt',[3 5],OptsGen);
```

or equivalent interactive options.
If the user wants to manually check and, if necessary, integrate the offsets list, the command list should be

```
COut=StaVelMain('SampleStations.txt',[3 4],OptsGen);
COut=StaVelMain('SampleStations.txt',[3 6],OptsGen);
```

for the case of NGL or another database respectively.
In this case, NGL is used and, therefore, `OptAct=[3 4]` is the choice to carry out the above described activity in the same `StaVelMain` session in which the offset data are downloaded. The direct use of `InspectOffsetSearch` function, or the use of such a function called from `StaVelMain`, simply depends on user's wishes. There are no significant differences in the two cases.

5.3.3 Removal of time series segments

As an `InspectOffsetSearch` session (independent or called by `StaVelMain`) is completed, the problematic time series segments can be removed using the `tsSegmentRemoval` function (please see Subsection 4.4 or type `help tsSegmentRemoval` on MCW for more information about this function). The latter operates exclusively at the level of a single station in order to prevent incorrect cancellations. In the case of the `ECNV` station, the delete command is

```
tsSegmentRemoval('ECNV',OptsGen,[733746 733833]);
```

or, equivalently,

```
tsSegmentRemoval('ECNV',OptsGen,[733746 733833],'Obs');
```

where the dates, expressed in the serial MATLAB form, were previously found with `InspectOffsetSearch`. It is important to underline that the `tsSegmentRemoval` function updates and saves the `tsData` files and the observation mom files (or the raw mom files if indicated in the command line)[2]. The ECNV time series after this action is shown in Fig. 5.5. Such a figure can be obtained either with a new session of `InspectOffsetSearch`, or with a session of `InspOffsetMom` is the check is required for one or two stations and a manual approach is suitable. In this second case, the command line is

```
InspOffsetMom('ECNV',OptsGen,[],1)
```

(please type `help InspOffsetMom` on MCV for more information about this function).

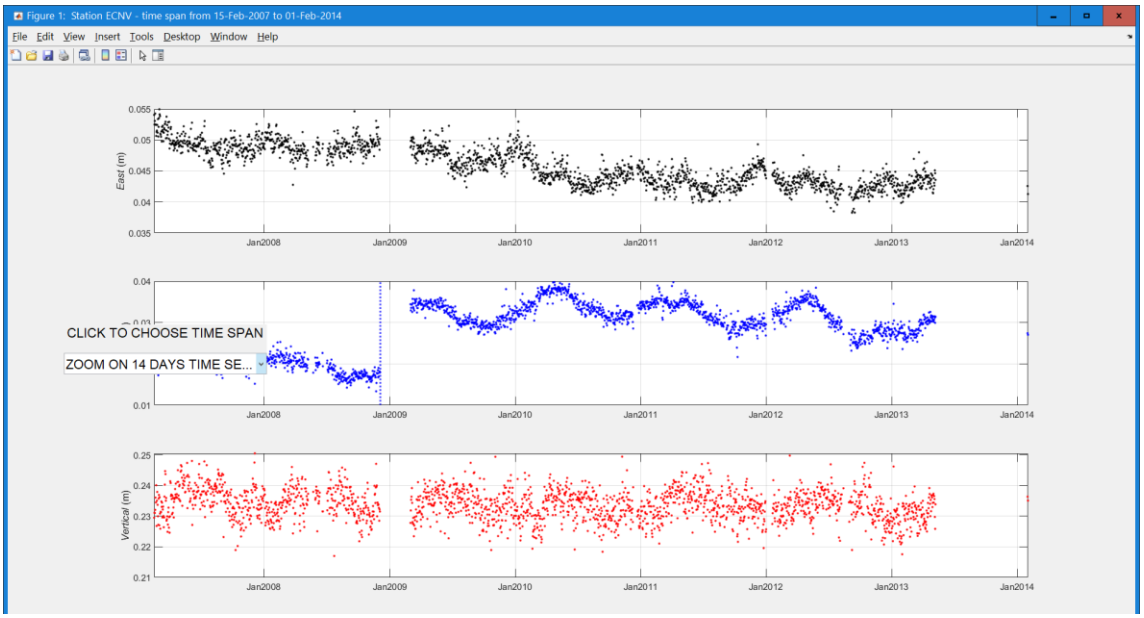

Figure 5.5 ECNV time series after segment removal and offset recognition.

A similar action is required for EIIV time series. In this case, the corresponding command line is

```
tsSegmentRemoval('EIIV',OptsGen,[737365 737515]);
```

If the user is not interested to any previously known offsets, it is possible to obtain the same results achieved by means of `InspectOffsetSearch` using `StaVelMain` directly with the option

[2] The time series inspection and possible removal of time series sections can be carried out before or after the offset recognition. In this tutorial, the second approach is preferred, in which it is possible to observe the time series and make decisions on identifying offsets and removing any problematic time series segments at the same time. Clearly, if the inspection and eventual removal of time series segments are carried out before the search for offsets, the mom files to be updated are the raw ones (in this case, the fourth input argument for the `tsSegmentRemoval` function must be the string `'Raw'`).

`OptAct = [3 2]` (or interactive equivalent), i.e. the command line is `COut=StaVelMain('SampleStations.txt',[3 2],OptsGen)`. It is emphasized that this involves the loss of any previous information on offsets. The maintenance, if required, of previous data on offsets, possibly to be integrated by manual search, is the characterizing element of the `InspectOffsetSearch` function.

5.4 Outlier recognition, trend computation and evaluation of results

The data needed to complete the station velocity calculation are now available. To do this, the required command line is (`OptAct 6`)

```
COut=StaVelMain('SampleStations.txt',6,OptsGen);
```

(or, in MATLAB, the corresponding choices can be interactively taken).
Equivalently, it is possible to proceed first to search for outliers (`OptAct 4`), and then to compute velocities (`OptAct 5`).
As velocity data are available, the `COut` output variable is filled.
As the function works, this information is shown on the MCW for each station (in this case ECNV):

```
station ECNV, component 0 (E) - outlier removal in progress...
station ECNV, component 0 (E) - outlier removed
station ECNV, component 0 (E) - trend estimation in progress...
station ECNV, component 0 (E) - trend estimated
station ECNV - E, .\Tutorial\ts_files\ECNV.EU.mat tsData file updated
station ECNV, component 1 (N) - outlier removal in progress...
station ECNV, component 1 (N) - outlier removed
station ECNV, component 1 (N) - trend estimation in progress...
station ECNV, component 1 (N) - trend estimated
station ECNV - N, .\Tutorial\ts_files\ECNV.EU.mat tsData file updated
station ECNV, component 2 (V) - outlier removal in progress...
station ECNV, component 2 (V) - outlier removed
station ECNV, component 2 (V) - trend estimation in progress...
station ECNV, component 2 (V) - trend estimated
station ECNV - V, .\Tutorial\ts_files\ECNV.EU.mat tsData file updated
station ECNV, output cell row generated
```

The `tsData` file is upgraded for each component.

In the specific case, the output cell variable `COut` is:

ECNV	733088	735631	37.5956	14.7125	476.23	-1.398	-1.059	-0.372	0.186	0.263	0.283
EDEN	734629	738809	37.5231	14.3035	732.30	-1.996	3.250	-0.182	0.129	0.162	0.240
EIIV	732609	738809	37.5136	15.0821	88.89	0.734	2.372	1.485	0.112	0.114	0.266
GALF	733005	738048	37.7107	14.5665	731.40	-1.984	4.708	0.503	0.112	0.098	0.109
HLNI	734185	738809	37.3486	14.8719	133.272	-1.022	4.566	-0.048	0.053	0.090	0.153

The 12 columns are, in the order: station name; initial time (MATLAB serial form); final time; latitude (°); longitude (°); ellipsoidal elevation (m); East velocity (mm/y); North velocity (mm/y); Vertical velocity (mm/y); East velocity standard deviation (SD) (mm/y); North velocity SD (mm/y); Vertical velocity SD (mm/y).

It is possible to evaluate the obtained results by showing the time series and the modeled trends, i.e. the content of the mom files obtained by calculating the trend. The suitable function is `showMom` (please type `help showMom` on MCW for more information about this function). To show the results for all the mom time series and the final model, the command line is

```
showMom('SampleStations.txt',OptsGen,'mom')
```

As this function runs, a figure is generated for each station. This fact must be taken into account in the case of a large number of stations (if necessary, it may be advisable to split the contents of the station file).

The results in the case of HLNI station are shown in Fig. 5.6.

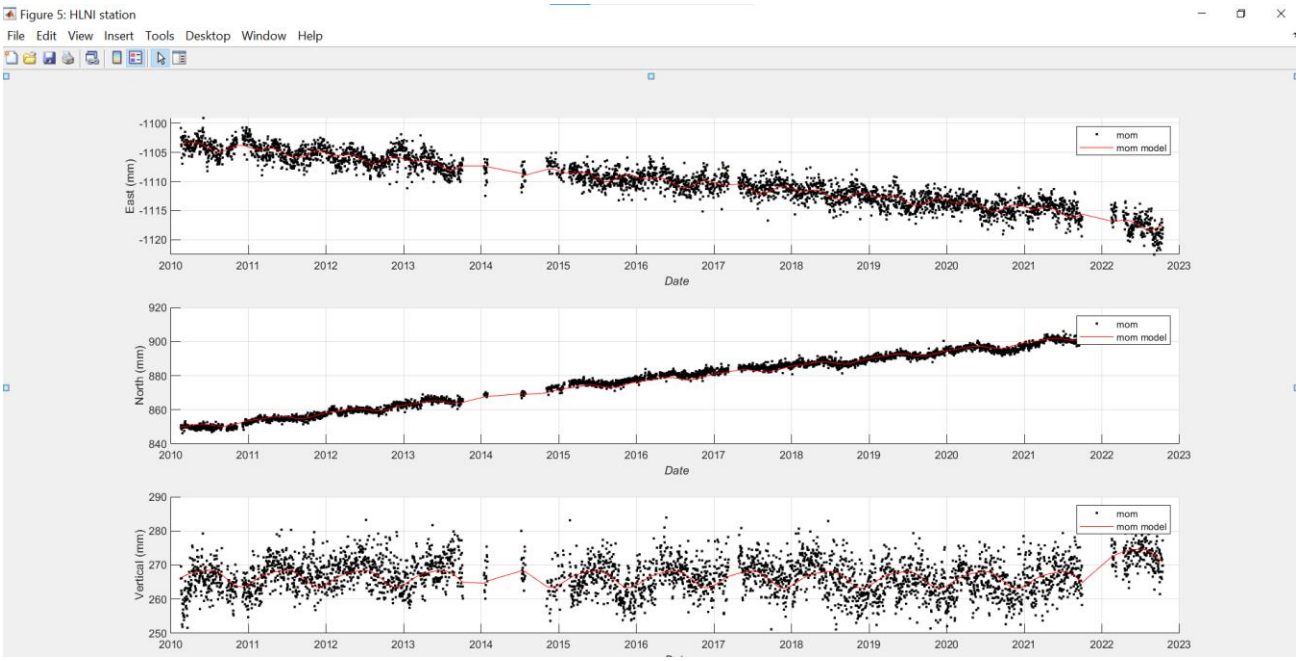

Figure 5.6 Mom time series and modeled time series for HLNI station.

The results seem to be satisfying for all the stations, except EDEN, for with a segment deletion and the introduction of offsets is required (this is left to the reader for exercise).

If the user wants to see the time series of all the mom files, including raw, observation and preprocessed, the command line is

```
showMom('SampleStations.txt',OptsGen,{'raw','obs','pre','mom'}).
```

If there are several stations and the user wants to see the time series for a station only, he/she can use `showMom` and redefine the file including the name of this station only, or use the function `showMomSingle`. In the case of HLNI, to show the mom time series and the model, i.e. to obtain the same result in Fig. 5.6 for this specific station, the command line is

```
showMomSingle('HLNI','.EU','Tutorial/mom_files','mom');
```

The different syntaxes of `showMom` and `showMomSingle` should be noted. Please refer to the corresponding helps for more information.

5.5 Velocity exportation to `GridStrain`

The obtained data can be exported to `GridStrain` by means of `geneGSFile` function (please see Subsection 4.5 or type `help geneGSFile` on the MCW). In order to generate the ASCII file 'TutorialVel.txt' with the station names (sixth input argument: is 1) starting from the cell variable provided by StaVelMain COut, with the horizontal components only (second input argument: 2), with minimum time series length 4.5 y, the command line is:

```
CCompl=geneGSFile(COut,'TutorialVel.txt',2,4.5,'33N',1);
```

This function also requires the choice of the UTM zone. This because `GridStrain` is incompatible with geographic coordinates but requires data expressed in a projected coordinate system. To obtain the results, the function `wgs2utm` by Alexandre Schimel (MetOcean Solutions Ltd, New Plymouth, New Zealand) is also used.

The output file shows these data:

```
ECNV   474617.6995  4160988.0314 -1.39845     -1.05895     0.186270     0.263388
EDEN   438454.0056  4153130.0115 -1.99602     3.249760     0.128591     0.161566
EIIV   507254.3131  4151853.8595 0.733491     2.371800     0.111879     0.113773
GALF   461792.5148  4173805.6890 -1.98372     4.707620     0.111877     0.0982456
HLNI   488658.5567  4133547.8445 -1.02225     4.565980     0.0529152    0.0902596
```

If the data are loaded within `GridStrain`, the resulting velocity vectors are in Fig. 5.7 (to obtain this figure, the files `TutorialMap.jpg` and `TutorialMap.mat` must be placed in `GridStrain` main folder together with the file generated by means of `geneGSFile`).

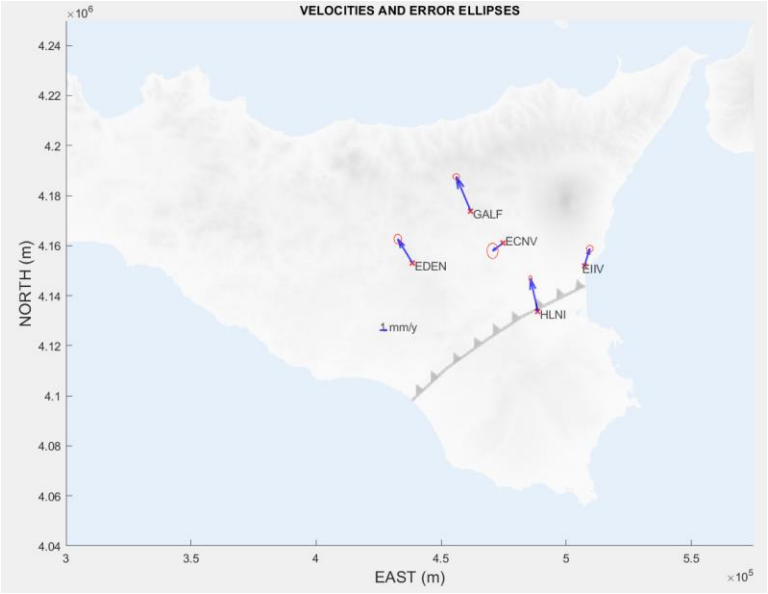

Figure 5.7 Velocities obtained with the tutorial, together with their 2-sigma (i.e. 95% confidence level) error ellipses (EU plate, WGS84UTM33 reference frame).

6. List of `StaVel` MATLAB functions

The list of MATLAB functions/scripts/objects is shown here. Besides the file name, a very brief description is shown for each function. The corresponding help can be seen by typing

```
help (function name)
```

on the MCW. For example, in the case of `StaVelMain`, the help can be accessed by typing

```
help StaVelMain
```

The main functions, which should be directly managed by the user in the standard `StaVel` computations, are highlighted with bold font. The other functions are generally called by the main functions, but can also be used in an autonomous way (please see the help for the allowed syntaxes). Please note that, if the toolbox is used under GNU Octave, some interactivities could be not allowed, i.e. the complete command lines should be used.

`addfromStruct`	Adding values to an object from a struct array
`CMEstackFiltering`	**Common mode error stack filtering**
`ctlFileEdit`	Editing a `.ctl` file (Hector control file)
`date2gpsw`	GPS week and day computation from date
`detrendZeroMean`	Computing detrended, zero mean time series
`frac2MJD`	Fractional year date to MJD conversion
`frac2serial`	Fractional year date to MATLAB serial date conversion
`geneGSfile`	**Generation of a .txt file for `GridStrain`**
`geneOpts`	**Generation of struct variable for `StaVel` options management**
`geneTrendFile`	Generates an Excel .xlsx file of GNSS station trends (this function is not used by `StaVelMain` but is proposed for those users who want to export the data to Excel).
`GetLonLat`	Get geodetic coordinates from Nevada Geodetic Laboratory (this function provides data only if the computer is online)
`GetNevada`	**Download one or more GNSS timeseries from Nevada database** (this function provides data only if the computer is online)
`gpsw2serial`	GPS week/day to serial date conversion
`InspectOffsetSearch`	**Inspection of several GNSS time series and offsets search**
`InspOffsetMom`	GNSS time series inspection and manual offset detection for a single station
`jsonRead`	Read a JSON file
`Leapy`	Search of leap years
`MJD2frac`	MJD to fractional year date conversion
`MJD2serial`	MJD to MATLAB serial year date conversion
`MosMom2MosNeu`	Conversion of a Mos matrix (i.e. an offsets matrix) from mom notation to CATS Neu notation
`MosMomGen`	Generation of a Mos matrix (mom notation) from a `tsData` object
`OptsGenLoad`	Interactive load of a `StaVel` options file
`NevadaAllSteps`	Takes all steps from Nevada database
`readStationList`	Read station list from an Excel or an ASCII file
`searchFilledFields`	Search of filled fields of a struct or an object
`serial2frac`	Date conversion from serial to fractional year

serial2MJD	Date conversion from serial to MJD
serial2YMD	Date conversion from serial to YYMMMDDD
showMom	**Show mom time series (several stations)**
showMomSingle	Show mom time series (single stations)
StaVelMain	**Main StaVel function**
TrendExt	Hector/CATS-based trend evaluation and data extraction (function not used by StaVelMain but added for the users who want use CATS instead of Hector)
TrendExtJSON	Trend extraction from a JSON file generated by Hector
tsData	Generation of a tsData object
tsDataFileIn	Access to a tsData file
tsSegmentRemoval	**Removal of one or more time series segments**
txtReadData	Header and data extraction from a txt file
txtWriteData	Header and data write to a txt file
verCoorLonLat	Verification and correction of lon/lat/height coordinates
wgs2utm	Coordinate conversion from WGS84 to UTM (external function by Alexandre Schimel, MetOcean Solutions Ltd, New Plymouth, New Zealand).
wlinfit	least square straight line fit
writeNeuMom	Write one or more .neu/.mom files for CATS/Hector
YMD2serial	Date conversion from YYMMMDD to MATLAB serial date

7. References

Blewitt, G., Hammond, W.C., Kreemer, C., 2018. Harnessing the GPS data explosion for interdisciplinary science. Eos, 99, https://doi.org/10.1029/2018EO104623.

Bos, M.S., Fernandes, R.M.S., Williams, S.D.P., Bastos, L., 2013. Fast error analysis of continuous GNSS observations with missing data. Journal of Geodesy, 87, 351–360. https://doi.org/10.1007/s00190-012-0605-0.

He, X., Yu, K., Montillet, J.-P., Xiong, C., Lu, T., Zhou, S., Ma, X., Cui, H., Ming, F., 2020. GNSS-TS-NRS: An Open-Source MATLAB-Based GNSS Time Series Noise Reduction Software. Remote Sensing, 12, 3532. https://doi.org/10.3390/rs12213532.

Meschis, M., Teza, G., Serpelloni, E., Elia, L., Lattanzi, G., Di Donato, M., Castellaro, S., 2022. Refining rates of active crustal deformation in the upper plate of subduction zones, implied by geological and geodetic data: the E-dipping West Crati fault, Southern Italy. Remote Sensing, 14, 5303. https://doi.org/10.3390/rs14215303.

Nikolaidis, R., 2002. Observation of Geodetic and Seismic Deformation with the Global Positioning System. PhD dissertation, University of California: San Diego, CA, USA. Available online at: https://www.proquest.com/pagepdf/304798351?accountid=9652 (last access: 16/11/2022).

Teza, G., Pesci, A., Galgaro, A., 2008. Grid_strain and grid_strain3: software packages for strain field computation in 2D and 3D environment. *Computers & Geosciences*, **34** (9), 1142-1153.

Teza, G., Pesci, A., Meschis, M., 2022. A MATLAB toolbox for computation of velocity and strain rate field from GNSS coordinate time series. *Annals of Geophysics*, submitted.

Tian, Y., Shen, Z., 2011. Correlation Weighted Stacking Filtering of Common-Mode Component in GPS Observation Network. *Acta Seismologica Sinica* 2011, 33, 198–208. https://doi.org/10.3969/j.issn.0253-37822011.02.007.

Wdowinski, S., Bock, Y., Zhang, J., Fang, P., Genrich, J., 1997. Southern California permanent GPS geodetic array: Spatial filtering of daily positions for estimating coseismic and postseismic displacements induced by the 1992 Landers earthquake. *Journal of Geophysical Research Solid Earth*, 102, 18057–18070. https://doi.org/10.1029/97JB01378.

Williams, S.D.P., 2008. CATS: GPS coordinate time series analysis software. GPS Solutions, 12(2), 147-153. https://doi.org/10.1007/s10291-007-0086-4.