Hector user manual
version 1.5.1

Machiel Bos and Rui Fernandes

December 17, 2015

Contents

1 Introduction

Hector is a software package that can be used to estimate the linear trend (i.e. motion/velocity/rate) in time-series with temporal corelated noise. Trend estimation is a common task in geophysical research where one is interested in phenomena such as the increase in temperature, sea level or GNSS derived station position over time. It is well known that in most geophysical time-series the noise is correlated in time (Agnew, 1992; Beran, 1992) and this has a significant influence on the accuracy by which the linear trend can be estimated. Therefore, the use of a computer program such as Hector is advisable.

Hector assumes that the user knows what type of temporal correlated noise exists in the observations and estimates both the linear trend *and* the parameters of the chosen noise model using the Maximum Likelihood Estimation (MLE) method. Since for most observations the choice of noise model can be found from literature or by looking at the power spectral density, this is sufficient in most cases.

Instead of using Hector, one can also use the CATS software of Williams (2008) which is a good and popular choice. In fact, Hector was written from scratch in C++ with the objective to have a faster CATS. The reason is Hector is faster is that it accepts only stationary noise, with constant noise properties, and this allows the use of fast matrices operations. The obtained reduction in computation time is a factor of 10-100 compared to CATS, see Bos et al. (2013).

This manual starts in section 2 by explaining how to install Hector on your computer. In section 3 a tutorial on using Hector is presented using two examples of analysing synthetic GPS data with offsets and outliers and a real GPS data set. This is followed by sections 4 and 5 on acceptable data formats and implemented noise models respectively. Finally, a quick reference of the parameters in the control files are presented in 6.

1.1 How to cite Hector

If you find the Hector program useful, please cite it in your work as:

> Bos, M. S., Fernandes, R. M. S., Williams, S. D. P., and Bastos, L. (2013). Fast Error Analysis of Continuous GNSS Observations with Missing Data. *J. Geod.*, Vol. 87(4), 351–360, doi:10.1007/s00190-012-0605-0.

Table 1: List of programs provided by the Hector software package.

Name	Description
estimatetrend	Main program to estimate a linear trend.
estimatespectrum	Program to estimate the power spectral density from the data or residuals using the Welch periodogram method.
modelspectrum	Given a noise model and values of the noise parameters, this program computed the associated power spectral density for given frequency range.
removeoutliers	Program to find offsets and to remove outliers from the data.
simulatenoise	Program to files with synthetic coloured noise.
date2mjd	Small program to convert calender date into Modified Julian Date.
mjd2date	The inverse of date2MJD.

1.2 Main features

The main features of Hector are:

1. Correctly deals with missing data. No interpolation or zero padding of the data nor an approximation of the covariance matrix is required (as long the noise is, or has been made, stationary).

2. Allows yearly, half-yearly and other periodic signals to be included in the estimation process of the linear trend.

3. Allows the option to estimate offsets at given time epochs.

4. Includes power-law noise, ARFIMA, generalized Gauss-Markov and white noise models. Any combination of these models can be made.

5. Comes with programs to remove outliers, to make power spectral density plots and to create files with synthetic coloured noise.

2 Installation

The hector software package is intended to be run on computers with Unix-like operating systems. For the previous versions of Hector special debian binary packages were created but these were, as far as we know, never used. Also the compilation of the source code was troublesome for most users because the library names keep on changing and they are slightly different on the various Linux distributions. For that reason, for the current version 1.5.1 we only provide the static binaries (64 bit), besides the source code, from the website: http://segal.ubi.pt/hector. It is advisable to put the binaries in the directory /usr/local/bin. If the binaries are put in another directory, then make sure it's added to the `PATH` variable in your shell environment. The list of programs provides is given in Table 1.

To run the examples one also need the `Tcl` scripting language and the `gnuplot` plotting program. On an Ubuntu machine one can type:

```
sudo apt-get install pgplot tcl
```

3 Tutorial

The directory with the examples that come with the Hector package can be copied into any directory you want. By going step by step through the analysis of some example data sets the working of Hector will be explained. First, we look at some synthetic GPS data.

3.1 Example 1: Synthetic GPS time-series with spikes and offsets

In the directory `ex1` a data file called TEST.enu is stored which represents some fictional GPS position data set. The file extension 'enu' stands for East-North-Up and these three components are stored in the 2^{nd}, 3^{rd} and 4^{th} column respectively. The first column contains the Modified Julian Date (MJD) which is convenient format to make plots. You can use the programs `date2mjd` and `date2mjd` to convert between year/month/day/hour/minute/second and MJD values. These data are stored in a simple ASCII text file and can be inspected by any normal text editor. When doing so, one will detect some additional header lines:

```
# sampling period 1.0
# offset 50284.0 0
# offset 50334.0 0
# offset 50334.0 1
# offset 50784.0 0
# offset 51034.0 0
```

The first line just tells the program that the sampling period of the data is daily (T=1 day). Furthermore, there are offsets at the MJD epochs 50284, 50334, 50784 and 51034. The last 0 indicates that these only occur in the East component (0=East, 1=North, 2=Up). If an offset occurs in more than one component, the header line for the offset is repeated such as is the case for MJD=50334. Hector cannot find the time of offset, these have to be provided by the user.
After these header lines the data are listed:

```
50084.0 -17.88951 -21.47271 -19.38038
50085.0 -16.88599 -20.40868 -18.27968
50086.0 -16.84916 -20.31029 -18.14525
...
```

As explained before, each row contains the time (MJD), East, North and Up component. The east component (column 2) can be plotted in gnuplot (type gnuplot on the command line first to start this plotting program) using the command:

```
plot 'TEST.enu' using (($1-51544)/365.25+2000):2 with lines
```

The time values are converted from MJD to years. The result is shown in Figure 1 where one can detect the offsets and the presence of outliers.

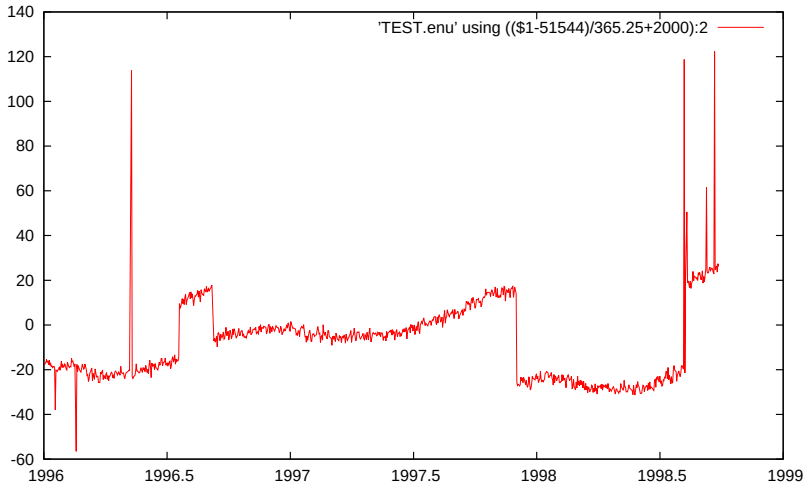

Figure 1: The raw data of the East component of TEST.enu.

3.1.1 Removal of outliers

To remove the outliers we need to run `removeoutliers` which requires a control file called `removeoutliers.ctl`. Hector uses various control files which are simple text files and the rows with the keywords can occur in any order. If Hector cannot find a keyword, then it will complain unless the keyword is optional. If the keyword optional and omitted then its default value will be used. Another control file can be specified on the command line. For example:

```
removeoutliers othercontrolfile.ctl
```

The contents of `removeoutliers.ctl` in the `ex1` directory is:

```
DataFile            TEST.enu
DataDirectory       ./
OutputFile          TEST_pre.mom
component           East
interpolate         no
seasonalsignal      yes
halfseasonalsignal  no
estimateoffsets     yes
IQ_factor           3.0
PhysicalUnit        mm
ScaleFactor         1.0
```

The first line gives the name of the DataFile with the raw data which is TEST.enu in our case. The second line gives the directory where this file can be found and the third live contains the required name of the file with the preprocessed data (outliers removed): TEST_pre.mom. The output will always be in mom-format which stands for **M**JD, **O**bservations, **M**odel. The last column is optional and since `removeoutliers`

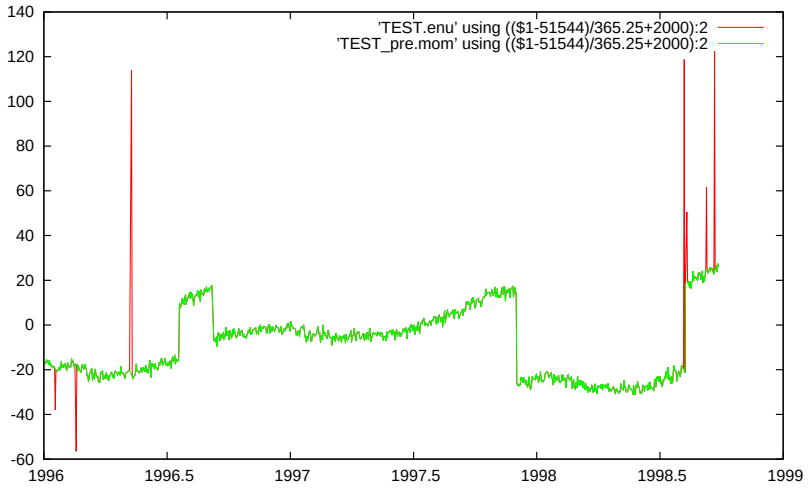

Figure 2: The raw and preprocessed data of the East component stored in TEST.enu and TEST_pre.mom.

only replaces the raw observations with the preprocessed observations, no third column will be added. See section 4 for more details on the acceptable data format.

`removeoutliers` fits a linear trend to the raw data using ordinary least-squares and afterwards subtracts this linear trend from the observations to create residuals. These residuals are ordered by size and the interquartile range is computed (this is the value of the residual at 75% of the sorted array minus the value of the residual at 25% of the sorted array). Any residual with a value less than 3 times this interquartile range below or above the median is considered to be an outlier (Langbein and Bock, 2004). This factor of 3 is set by the keyword IQ_factor and can be changed by the user.

In this control file one must also give the physical unit of the data. This information is not essential but reminds the user to think about the unit of the data and if some scaling is required. Such scaling is set by the keyword 'ScaleFactor' which is 1.0 in this case. This keyword is optional and if omitted then a default value of 1.0 will be assumed.

The linear trend is estimated assuming a white noise model and, as can be seen from the `removeoutliers.ctl` file, a seasonal (i.e. yearly) signal is also included in the estimation process. Offsets are also estimated. On the other hand, no half-seasonal signal is estimated nor any other periodic signal and the missing data are not interpolated. The keyword 'periodicsignals' is optional and can be omitted.

The results of `removeoutliers`, stored in TEST_pre.mom, are shown in Figure 2 which have been generated with `gnuplot` using the command:

```
plot 'TEST.enu' using (($1-51544)/365.25+2000):2 with lines,\
     'TEST_pre.mom' using (($1-51544)/365.25+2000):2 with lines
```

3.1.2 Estimation of the linear trend

Now that the outliers have been removed, we can estimate the linear trend. The parameters that control this analysis are by default given in the file `estimatetrend.ctl`.

As before, another name for the control file can be specified on the command line. The contents of `estimatetrend.ctl` is:

```
DataFile            TEST_pre.mom
DataDirectory       ./
OutputFile          TEST_out.mom
interpolate         no
seasonalsignal      yes
halfseasonalsignal  no
estimateoffsets     yes
NoiseModels         PowerlawApprox White
LikelihoodMethod    AmmarGrag
PhysicalUnit        mm
ScaleFactor         1.0
```

Again, there is the keyword DataFile which should be given by the name of the input file which is TEST_pre.mom in this case because we are now going to use the preprocessed observations. These preprocessed observations together with the estimated trend in the third column, are written to the file name given after the keyword OutputFile. As before, the data are not interpolated. However, a seasonal signal and offsets are estimated. The chosen noise models are a combination of power-law noise (appriximated) plus white noise. Alternatively we could have chosen from: ARMA, ARFIMA or GGM (Generalized Gauss Markov), FlickerGGM and RandomWalkGGM for the noise models. More information about them is given in section 5. Note that 'PowerlawApprox' should give very similar results as 'GGM' when you fix the `GGM_1mphim` value, see section 5.

The chosen method for the Likelihood computation is 'AmmarGrag' which is explained in more detail in Bos et al. (2013). The keyword LikelihoodMethod is optional and can be omitted. If it is omitted, then the default method is 'AmmarGrag' is the percentage of missing data are less than 50% of the total observation period. Otherwise the Full Covariance matrix ('FullCov') method is used.

Note that if the ScaleFactor is not 1 in the file `removeoutliers.ctl`, then you probably want to set it to 1 in `estimatetrend.ctl` to avoid applying the scaling twice. Again, this keyword is optional and a default value of 1 is assumed when this keyword is not provided. The information of the offsets can be given in another file by using the keyword 'OffsetFile' and specifying the 'component' keyword, see section 7.

The program `estimatetrend` shows the following on the screen:

```
**********************************
    estimatetrend, version 1.5.1
**********************************
0) PowerlawApprox
1) White
generator type: mt19937
seed = 0
first value = 4293858116
Data format: MJD, Observations, Model
Filename              : ./TEST_pre.mom
```

```
Number of observations: 1000
Percentage of gaps    : 10.7

----------------
  AmmarGrag
----------------
No quadratic term is included.

Number of CPU's used (threads) = 1

    1    0.55000     0.55000   f()= 1738.477356  size=0.265
...
   35    0.42454     0.41335   f()= 1724.688587  size=0.000
converged to minimum at
   36    0.42455     0.41337   f()= 1724.688587  size=0.000

Likelihood value
--------------------
min log(L)=-1724.689
AIC       =3453.377
BIC       =3462.966

PowerlawApprox:
fraction  = 0.42455
sigma     = 3.55346 mm/yr^0.20669
d         = 0.4134 +/- 0.1041
kappa     = -0.8267 +/- 0.2082

White:
fraction  = 0.57545
sigma     = 1.22192 mm
No noise parameters to show

STD of the innovation noise: 1.61078
bias : 0.872 +/- 0.966 mm (at 1997/5/15, 12:0:0.000)
trend: 16.440 +/- 0.724 mm/year
cos yearly : 4.097 +/- 0.294 mm
sin yearly : -3.927 +/- 0.298 mm
offset at 50284.0000 :   24.46 +/-  0.70 mm
offset at 50334.0000 :  -24.65 +/-  0.77 mm
offset at 50784.0000 :  -39.86 +/-  0.72 mm
offset at 51034.0000 :   38.53 +/-  0.74 mm
--> TEST_out.mom
Total computing time: 3.00000 sec
```

The first lines are self explanatory. The number of CPU's used is also shown to make sure that Multi-Threading on Multi-Core Processors is working. In this case only 1 CPU

is used. The next few lines show the minimization steps of the negative value of the natural logaritm of the likelihood (which is thus maximised!).

Nowadays the quality of the chosen noise models in describing the noise in the data is evaluated using the Akaike Information Criteria (AIC) and the Baysian Information Criteria (BIC). These values are shown below the value of the natural logarithm of the Likelihood value.

Since we are using white and power-law noise, two noise amplitudes need to be estimated. The parameter (ϕ_1), determines the distribution of white and power-law noise given as fractions, see section 5 for more details. I call the main driving noise the innovation noise which for this case has an standard deviation of 1.6108 mm. Taking note that parameter ϕ determines the distribution of variances and that $1 - 0.425 = 0.575$, the standard deviation of the white noise is:

$$\sigma_{wn} = \sqrt{0.575} \times 1.6108 = 1.222 \tag{1}$$

In Hector the covariance matrix is not scaled by the factor $\Delta T^{-\kappa/2}$, where ΔT is the sampling period in years (Williams, 2003). However, to facilitate comparison with amplitude values for power-law noise quoted in the literature, we divide the estimated amplitude by $\Delta T^{-\kappa/4}$:

$$\sigma_{pl} = \frac{\sqrt{0.425} \times 1.6108}{(1/365.25)^{(0.25 \times 0.8267)}} = 3.553 \tag{2}$$

The second noise parameter of the power-law noise is the spectral index d which is $-1/2$ times the more often used parameter κ in other papers on GPS time series. The values of d and ϕ_1 need to be determined using the numerical minimisation scheme and their values during each step (36 steps are needed before convergence has been reached) are shown in the output.

For white noise there is no additional parameter to estimate so, that is why there is this line "No noise parameters to show" in the output in the white noise section. More details on the noise models are given in section 5.

The rest of the lines show the estimated values of the model such as nominal bias (also known at intercept at t_0 and which is equal to the estimated value at t_0), linear trend and a seasonal signal. To get the most accurate estimate of the linear trend, the linear trend in the design matrix has a zero mean. To explain this better, assume that we have 5 observations. The design matrix $\mathbf{H}$ looks like:

$$\mathbf{H} = \begin{pmatrix} 1 & -2 \\ 1 & -1 \\ 1 & 0 \\ 1 & 1 \\ 1 & 2 \end{pmatrix} \tag{3}$$

The first column will estimate the nominal bias, the second the linear trend. The two columns are orthogonal since $\mathbf{H}^T\mathbf{H}$ produces a diagonal matrix. Thus, the estimation of the nominal bias is not influenced by the estimation of the linear trend which is beneficial for the accuracy. It also means that the nominal bias corresponds to the value of the model at the time at row 3 (half of the time-series). hector notes this time.

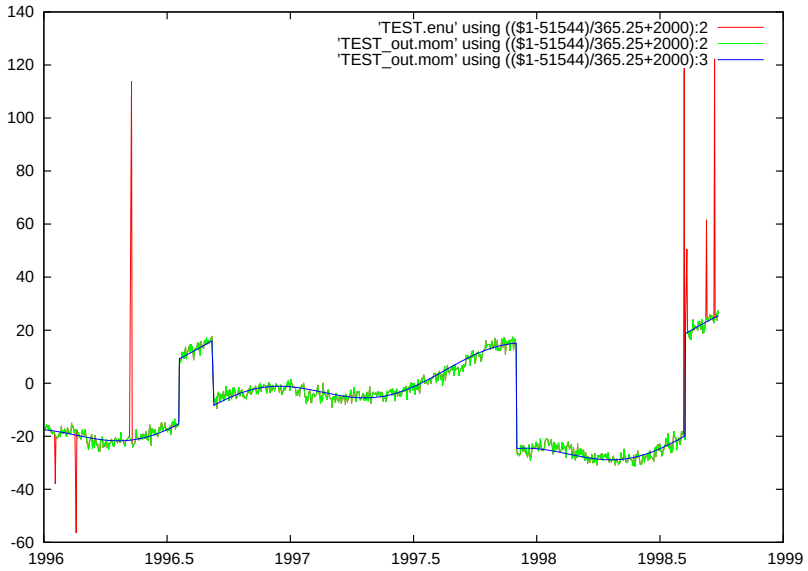

Figure 3: The raw, filtered data and the estimated model of the East component stored in TEST.enu and TEST_out.mom.

If another reference epoch for the nominal bias is required, then this date can be provided after the keyword 'ReferenceEpoch', using year, month and day. For example, a reference epoch of 1 January 2008 is given by:

```
ReferenceEpoch          2008 1 1
```

Also shown in the output are the values of the estimated offsets. The results of `estimatetrend`, stored in TEST_out.mom, are shown in Figure 3 which have been generated in `gnuplot` with the command:

```
plot 'TEST.enu' using (($1-51544)/365.25+2000):2 with lines,\
     'TEST_out.mom' using (($1-51544)/365.25+2000):2 with lines,\
     'TEST_out.mom' using (($1-51544)/365.25+2000):3 with lines
```

3.1.3 Plotting the power spectral density

We have used a power-law plus white noise model in our estimation process. To verify if this is correct, it is good to make a power spectral density (PSD) plot of the residuals (i.e. the difference between observations minus the estimated linear trend and additional offsets and periodic signals). This can be done using the program `estimatespectrum` which computes a Welch periodogram, stored in the file `estimatespectrum.out`. As usual, the behaviour of this program is controlled by the file `estimatespectrum.ctl`:

```
DataFile            TEST_out.mom
DataDirectory       ./
interpolate         no
```

```
ScaleFactor          1.0
WindowFunction       Parzen
Fraction             0.1
```

The contents of `estimatespectrum.out` is shown in Figure 4. By default the time-series is devided into 4 parts by `estimatespectrum`. Since 50% overlap is used, there are 7 segments in total and in this case the length of each segment is 250 data points. The first and last 10% of each segment (set by the keyword "Fraction") is smoothed to zero using a Parzen window function. The window function is set by the keyword "WindowFunction". Another choice is "Hann". If more segments are required, to get a better averaging of the periodograms but at the cost of a smaller frequency range, one can specify the number of divisions of the data on the command line. For example:

```
estimatespectrum 8
```

which will divided the time-series into 8 pieces, creating 15 segments due to the 50% overlap used. The area underneath the (one-sided) power spectral density plot should be equal to the variance of the time-series (Buttkus, 2000). This area underneath the PSD plot has been computed by simply assuming that each point represents a bar of width $1/\Delta t$ and adding them all up. Next, it is important to note the begin and end value of the frequency range.

```
----------------------
   EstimateSpectrum
----------------------

Data format: MJD, Observations, Model
Filename              : ./TEST_out.mom
Number of observations: 1000
Percentage of gaps    : 10.7
Number of data points n : 1000
Number of data used   N : 1000
Number of segments    K : 7
Length of segments    L : 250
Total variance in signal (time domain): 3.297
Total variance in signal (spectrum)   : 3.123
freq0: 4.6296e-08
freq1: 5.7870e-06
```

The PSD of our estimated noise model can be computed using the program `modelspectrum` which has no control file and which saves its output in `modelspectrum.out`. However, note that it gets information on the required set of noise models from the file `estimatetrend.ctl`. Normally one makes a PSD after estimating the trend so this should not be an inconvenience. However, note that one should use the 'Powerlaw' noisemodel instead of the 'PowerlawApprox' model for making power spectrum plots. The user must manually enter the requested values for the noise parameters and provide the begin and end value of the frequency range. For our example, the input looks like:

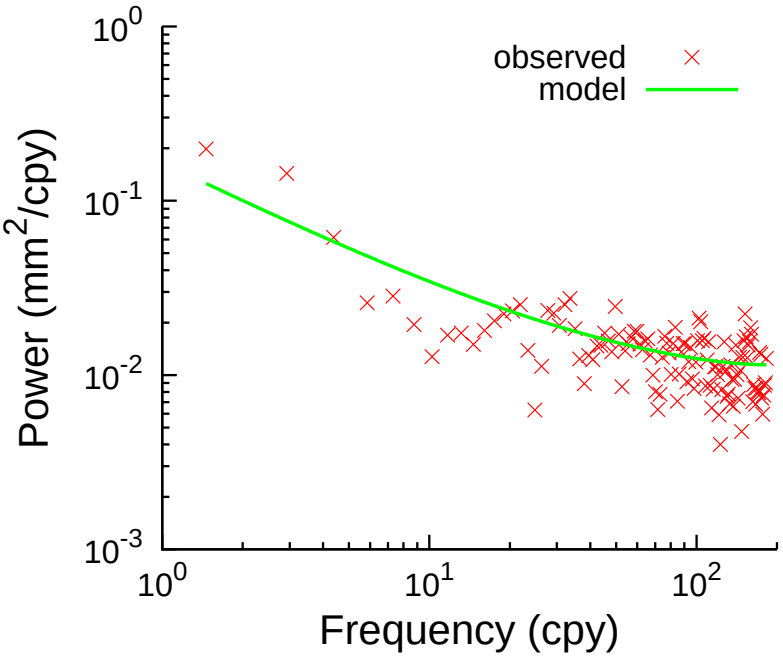

Figure 4: The PSD of the residuals and used noise model.

```
---------------
ModelSpectrum
---------------

Enter the standard deviation of the innovation noise: 1.6190
Enter the sampling period in hours: 24
Enter fraction for model Powerlaw: 0.490
Enter fraction for model White: 0.5103

Powerlaw:
Enter value of fractional difference d:0.3831

White:
1) Linear or 2) logarithmic scaling of frequency?: 2
Enter freq0 and freq1: 4.6296e-08 5.7870e-06
```

The contents of `estimatespectrum.out` and `modelspectrum.out` are plotted in Figure 4.

3.2 Example 2: The monthly PSMSL tide gauge data at Cascais

In the directory ex2 we have stored the monthly tide gauge data of Cascaise, downloaded from PSMSL (`http://www.psmsl.org/data/obtaining/stations/52.php`). This time-series has no outliers so we can directly estimate the linear trend. The control file `estimatetrend.ctl` is:

```
DataFile            52.rlrdata
DataDirectory       ./
OutputFile          52_out.mom
QuadraticTerm       no
interpolate         no
seasonalsignal      yes
halfseasonalsignal  yes
estimateoffsets     no
NoiseModels         ARMA
PhysicalUnit        mm
AR_p                1
MA_q                0
```

Here we are using the ARMA noise model. To be precise, there is 1 AR coefficient (set by the AR_p keyword) and 0 MA coefficients (set by the MA_q keyword). Thus, we can shorten our notation of ARMA(1,0) to AR(1). If we now run `estimatetrend` we obtain a linear trend of 1.270 $\pm$ 0.075 mm/yr. If we now change the noise model to AR(5), ARFIMA with AR_p=1 and MA_q=0 and GGM we obtain trends of 1.277 $\pm$ 0.103, 1.253 $\pm$ 0.175 and 1.259 $\pm$ 0.201 mm/yr which all have lower BIC and AIC values than the AR(1) noise model. Using `modelspectrum` and `estimatespectrum` one can produce the power spectral density plot shown in Figure 5. Note that the sampling time in hours is 730.5 hours. Furthermore, since only one noise model is used each time, the fraction is always 1. The controlfile `estimatespectrum.ctl` is:

```
DataFile            52_out.mom
DataDirectory       ./
interpolate         no
firstdifference     no
```

This provides us with the frequency range of 1.1317e-09 to 1.9013e-07 Hz which needs to be fed into `modelspectrum`.

In sea level research one is sometimes also interested in the acceleration. It is possible to estimate this by setting the optional keyword 'QuadraticTerm' to yes in `estimatetrend.ctl`. It's default value is no. If we do this, then we obtain, using the GGM noise model, an acceleration of 0.007 $\pm$ 0.012 mm/yr^2.

3.3 Example 3: Creating synthetic coloured noise

In order to perform Monte Carlo simulations, one must create time-series with synthetic coloured noise. This task can be performed with the program `simulatenoise`. It is based on the method described by Kasdin (1995) where an impulse response, different for each noise model, is convoluted with a white noise time-series. The result is our desired synthetic noise time-series. As usual, the convolution is performed using FFT. There might be some spin-up effects because implicitly it is assumed that the noise is zero before the first observation. To migitate this problem, the keyword 'TimeNoiseStart' can have a large number, normally 1000 is enough, to specify the amount of extra points before the first observations need to be modelled, see also section 5.3.

In the directory `ex3` the control file `simulatenoise.ctl` is given:

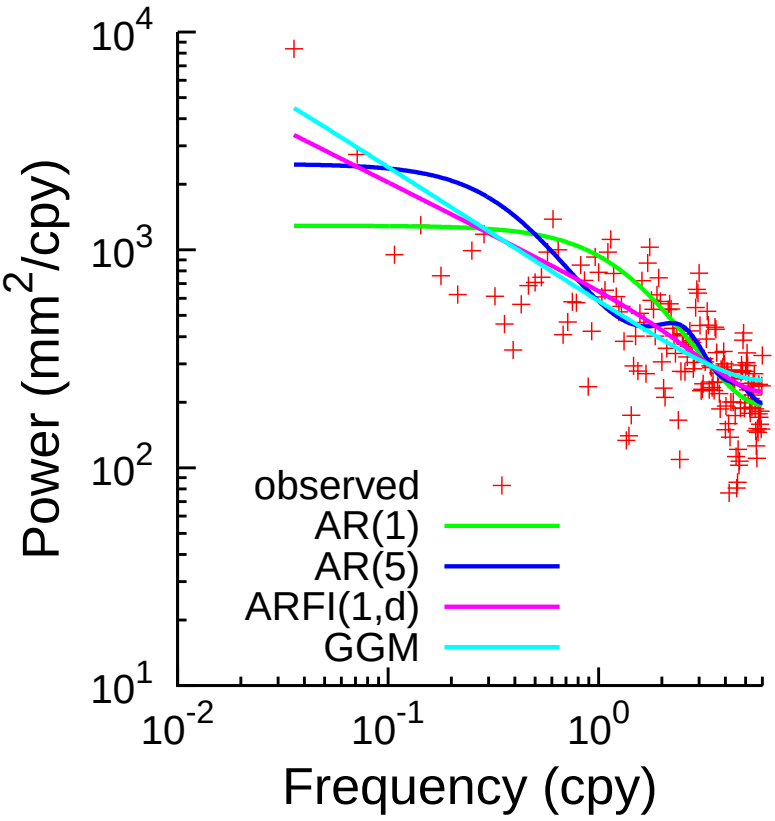

Figure 5: The power spectral density plot of tide gauge data at Cascais.

```
SimulationDir           ./
SimulationLabel         test_base
NumberOfSimulations     10
NumberOfPoints          5000
SamplingPeriod          1
TimeNoiseStart          1000
NoiseModels             Flicker White
PhysicalUnit            mm
```

Some of these keywords are new. For example, 'SimulationDir' specifies in which directory the created files should be stored. The keyword 'SimulationLabel' specifies the base name of those files. The next keyword tells hector how many simulation runs are required. The filenames will in this case be: test_base0.mom, test_base1.mom, ..., test_base9.mom.

The keyword 'NumberOfPoints' specifies the number of points in the the time-series and the keyord 'TimeNoiseStart' was already discussed above.

When `simulatenoise` is run, it will ask the user to manually enter the parameter values of the chosen noise model. In this case it will ask the values of ϕ and d.

4 Acceptable data format

Hector can accept various data formats which are described in this section. All of the are plain ASCII files and the time should always be increasing and the time step should be constant. The data format is specified by the extension of the filename. For example, the file name TEST.enu has the extension 'enu'.

For the mom and enu-format, the sampling period in days can be specified in the header as follows:

```
# sampling period 1.0
```

If this information is missing, then hector tries to estimate the sampling period from the first few observations. The sampling periods it can detect automatically are: 0.5 hour, 1 hour, 1 day and 7 days. Note that this sampling period must always be given in days!

4.1 mom-format

This format expects 2 or 3 columns. The first column contains the time in MJD, the second the Observations. The third column is optional and should contain the estimated Model. Missing data are allowed.

4.2 enu-format

This format is similar to the mom-format but has four columns. The first column contains the time in MJD, the second to the fourth column contain the East, North and Up component. Missing data are allowed.

4.3 neu-format

This format is used by SOPAC (`http://sopac.ucsd.edu/`) and accepted by the CATS software. The first column contains the time as year-fractions and columns two to four contain the North, East and Up component in metres. Missing data are allowed. The unit of these files is normally metres and it is convenient to convert these to millimetres using the keyword ScaleFactor in `removeoutliers.ctl`. The year-fractions are converted inside Hector to MJD using the formula:

$$MJD = floor(365.25 * (T - 1970) + 40587 + 0.1) - 0.5 \quad (4)$$

This implies that only sampling periods which are an integer multiple of 1 day are acceptable. Hector can read the slightly different offset headers which are of the form, see the CATS manual (Williams, 2008):

```
# offset 2003.45479 7
```

Note however that if an external file with offset information is used, then the expected format is of the form day-month-year NaN NaN NaN. where the last three parameters stand for East, North and Up. NaN indicates that an offset needs to be estimated. A normal number such as 0, tells the program no offset needs to be estimated for that component.

4.4 rlrdata-format

Hector can read PSMSL's monthly data format, see `http://www.psmsl.org/`. To create an evenly spaced data set inside Hector, each month is assumed to take exactly 30.4375 days, equal to 730.5 hours.

5 Implemented Noise Models

Hector can use various types of noise models and in addition, accepts various combinations of them with power-law plus white noise being the most popular for GPS time-series. Williams (2008) wrote the covariance matrix $\mathbf{C}$ of this particular combination as:

$$\mathbf{C} = \sigma^2 \left(\cos^2 \phi \, \mathbf{I} + \sin^2 \phi \, \mathbf{E}(d) \right) \quad (5)$$

where $\mathbf{I}$ is the unit matrix (equal to the covariance matrix for unit white noise) and $\mathbf{E}$ the covariance matrix for power-law noise which depends on the spectral index d. The distribution of the magnitudes of both noise models is controlled by the parameter ϕ. The total variance of the noise is set by σ^2 (This is called the 'innovation' noise in the output). This has been generalized to:

$$\mathbf{C} = \sigma^2 \left(\phi_1 \mathbf{E}_1 + (1-\phi_1)\phi_2 \mathbf{E}_2 + (1-\phi_1)(1-\phi_2)\phi_3 \mathbf{E}_3 + \ldots \right. \\ \left. (1-\phi_1)(1-\phi_2)\ldots\phi_N \mathbf{E}_{N+1} \right) \quad (6)$$

for $N+1$ noise models. All ϕ-parameters vary between 0 and 1. As was noted in section 1, only stationary noise is accepted. This creates a Toeplitz covariance matrix and only the first column of the covariance matrix needs to be stored. This column vector will be denoted by γ.

5.1 White noise

For white noise the covariance matrix is just the unit matrix The first column of the covariance matrix $\mathbf{C}$, with $\sigma = 1$, is:

$$\gamma_i = 1 \qquad \text{for } i = 0 \tag{7}$$
$$= 0 \qquad i \neq 0 \tag{8}$$

Its one-sided power spectrum density is:

$$S(f) = 2\frac{1}{f_s} \tag{9}$$

where f_s is the sampling frequency in Hz. If you integrate this from zero frequency to the Nyquist frequency, you get the variance that is observed in the time-series, as it should be.

5.2 Power-law noise

For power-law noise the first column of the covariance matrix is:

$$\gamma_i = \frac{\Gamma(d+i)\Gamma(1-2d)}{\Gamma(d)\Gamma(1+i-d)\Gamma(1-d)} \tag{10}$$

Its one-sided power spectrum density, with $\sigma = 1$, is:

$$S(f) = 2\frac{1}{f_s}\frac{1}{(2\sin(\pi f/f_s))^{2d}} \tag{11}$$

The power-law noise model is stationary up to d=0.5. For GNSS time series which have a d value close to 0.5 it is therefore, advisable to use PowerlawApprox or GGM with `GGM_1mphi` fixed to be extend the range of possible d values to above 0.5.

When the simulatenoise program is used one can specify 'Flicker' and 'RandomWalk' noise models because the limitation to stationary models does not apply.

5.3 Power-law approximated

Bos et al. (2013) introduced a Toeplitz approximation for power-law noise which can be chosen using the label "PowerlawApprox" after the keyword Noisemodels. In addition, one must specify the number of days before the start of the observation when the noise is assumed to have started after the keyword "TimeNoiseStart". A value of 1000 is a good first guess. Note that nowadays we prefer to use the Generalized Gauss Markov noise model with ϕ close to 1. This also approximates power-law noise and even works for random walk.

5.4 ARFIMA and ARMA

The definition of the ARFIMA noise model is Sowell (1992):

$$\Phi(L)(1-L)^d z_t = \Theta(L)\epsilon_t \tag{12}$$

L is the backshift operator ($Lx_i = x_{i-1}$), z_t is the residual at time t (observation minus modelled signal) and ϵ is a white noise signal. In other words, Eq. (12) says that the residuals in the observations can be produced by applying some transformations on a white noise process. The operators Φ and Θ are defined as, see Hosking (1981):

$$\Phi(L) = 1 - \phi_1 L - \phi_2 L^2 - \ldots - \phi_p L^p \tag{13}$$

$$\Theta(L) = 1 + \theta_1 L + \theta_2 L^2 + \ldots + \theta_q L^q \tag{14}$$

This definition is implemented in Hector but note that the definition of the signs before the ϕ coefficients in Φ are positive in Sowell (1992). To complicate matters further, the coefficients of Θ are negative in the formula's of Zinde-Wash (1988).

The value of the integers p and q are set by the keywords AR_p and MA_q respectively in `estimatetrend.ctl`. It is advised to use values for p smaller than 5 to ensure that the MLE procedure always start with coefficients values for $\phi_1, \ldots, \phi_p$ of $\Phi(L)$ that produce stationary noise. If p and q are zero, then one obtains again a pure power-law noise process. We have implemented the method of Doornik and Ooms (2003) to compute the first column of the covariance matrix. For the special case when $d = 0$, we use the equations of Zinde-Wash (1988) and can be selected by using the name 'ARMA' after the keyword Noisemodels. For sea level research the first order auto-regressive noise model is a popular choice: ARMA(1,0). For pure ARMA noise models faster Maximum Likelihood Methods exist, see for example Brockwell and Davis (2002), but Hector will give the same result.

5.5 Generalized Gauss Markov noise model

Langbein (2004) took the first order Gauss Markov noise model depending on the parameter ϕ and modified with an additional parameter, d, to create power-law noise with a slope of $2d$ in the power density spectrum which flattens to white noise at the very low and very high frequencies. The analytical expression for the autocovariance vector (with $\sigma = 1$) for this noise model is:

$$\gamma_i = \frac{\Gamma(d+i)(1-\phi)^i}{\Gamma(d)\Gamma(1+i)} \, {}_2F_1(d, d+i; 1+i; (1-\phi)^2) \tag{15}$$

This noise model can be used using the name 'GGM' after the keyword Noisemodels in `estimatetrend.ctl`. Bos et al. (2014) provide some additional formula's.

The $1 - \phi$ parameter can be held fixed a priori by adding to the control file:

```
GGM_1mphi            6.881e-6
```

where 6.881e-6 is the value you want to give this parameter and is small enough to be a good representation for power-law noise for time series shorter than 25 years. It is close to the smallest value you can use.

5.6 Flicker noise and Random Walk noise

Flicker noise and Random walk noise are simply two types of power-law noise where the spectral index d has the fixed value of 0.5 and 1.0 respectively. However, as was noted

in the introduction, Hector can only deal with stationary noise because that results in Toeplitz covariance matrices that allow fast inversion techniques. This can be achieved by using the Generalized Gauss Markov noise model with a small value for the $1 - \phi$ parameter. Example:

```
NoiseModels          FlickerGGM
GGM_1mphi            6.881e-6
```

6 Quick reference for the control files

6.1 removeoutliers.ctl

This file is read by `removeoutliers`.

Keyword	Value(s)
DataFile	name of file with observations
DataDirectory	directory where file with observations is stored
OffsetFile	name of file with offset information (optional)
OutputFile	name of file with observations *and* estimated model in .mom format
component	only required for the .enu and .neu format or when an OffsetFile is being used. Select East, North or Up
interpolate	yes|no
QuadraticTerm	yes|no (optional, default=no)
seasonalsignal	yes|no
halfseasonalsignal	yes|no
periodicsignals	a sequence of numbers reprenting the period in days (optional)
estimateoffsets	yes|no
ScaleFactor	a number to scale the observations (optional, default=1)
PhysicalUnit	the physical unit of the observations
IQ_factor	the number used to scale the interquartile range

6.2 estimatetrend.ctl

This file is read by `estimatetrend` and by `modelspectrum` although the latter only reads the keyword NoiseModels and, if necessary, the keywords AR_p, MA_q and TimeNoiseStart. The program `modelspectrum` creates a file called `modelspectrum.out`.

Keyword	Value(s)
DataFile	name of file with observations
DataDirectory	directory where file with observations is stored
OffsetFile	name of file with offset information (optional)
OutputFile	name of file with observations *and* estimated model in .mom format
component	only required for the .enu and .neu format or when an OffsetFile is being used. Select East, North or Up
interpolate	yes|no
QuadraticTerm	yes|no (optional, default=no)
seasonalsignal	yes|no
halfseasonalsignal	yes|no
periodicsignals	a sequence of numbers, separated by spaced, representing the period of the periodic signals in days (optional)
estimateoffsets	yes|no
ScaleFactor	a number to scale the observations (optional, default=1)
PhysicalUnit	the physical unit of the observations
NoiseModels	chose any set from: White, FlickerGGM, RandomWalkGGM, Powerlaw, PowerlawApprox, ARFIMA, ARMA and GGM.
LikelihoodMethod	chose one from: AmmarGrag or FullCov (optional, default=Ammargrag if percentage of missing data is less than 50% of the whole time-series. Otherwise FullCov method is used.
AR_p	number of AR coefficients (only for ARFIMA or ARMA)
MA_q	number of MA coefficients (only for ARFIMA or ARMA)
TimeNoiseStart	number of days before the start of the observations when it is assumed that the noise started (only for PowerlawApprox)

6.3 estimatespectrum.ctl

This file is read by `estimatespectrum` which creates a file called `estimatespectrum.out`.

Keyword	Value(s)
DataFile	name of file with observations
DataDirectory	directory where file with observations is stored
interpolate	yes|no
ScaleFactor	a number to scale the observations (optional, default=1)
WindowFunction	Parzen|Hann (Parzen is default)
Fraction	number between 0 and 1 (0.1 is default).

6.4 simulatenoise.ctl

This file is read by `simulatenoise`.

Keyword	Value(s)
SimulationDir	directory where created files will be stored
SimulationLabel	base name of the created files
NumberOfSimulations	number of simulations
NumberOfPoints	length of each simulation
SamplingPeriod	specifies the time step in days
TimeNoiseStart	number of points that need to be included before 1st observation has been made.
NoiseModels	chose any set from: White, Flicker, RandomWalk, Powerlaw, PowerlawApprox, ARFIMA, ARMA and GGM.

7 Advanced Features

7.1 Offsets

The information about the time of an offset can also be provided in an another file instead of in the header of the file with the observations. In this case one must specify the keyword 'OffsetFile' and its name in the control file. One must also not forget to specify the component in the control file. Since most people prefer simple day-month-year format, the format for the offsets in the external file is a bit different. Using the offset data shown before, we have:

```
20- 7-1996   NaN   0   0
 8- 9-1996   NaN NaN   0
 2-12-1997   NaN   0   0
 9- 8-1998   NaN   0   0
```

This external file with time of offsets can be used with `removeoutliers` and `estimatetrend`.

8 License and Copyright

References

Agnew, D. C. (1992). The time-domain behaviour of power-law noises. *Geophys. Res. Letters*, 19(4):333–336.

Beran, J. (1992). Statistical methods for data with long-range dependence. *Statistical Science*, 7(4):404–416.

Bos, M. S., Fernandes, R. M. S., Williams, S. D. P., and Bastos, L. (2013). Fast Error Analysis of Continuous GNSS Observations with Missing Data. *J. Geod.*, 87(4):351–360.

Bos, M. S., Williams, S. D. P., Araújo, I. B., and Bastos, L. (2014). The effect of temporal correlated noise on the sea level rate and acceleration uncertainty. *Geophysical Journal International*, 196:1423–1430.

Brockwell, P. and Davis, R. A. (2002). *Introduction to Time Series and Forecasting*. Springer-Verlag, New York, second edition edition.

Buttkus, B. (2000). *Spectral Analysis and Filter Theory in Applied Geophysics*. Springer-Verlag Berlin Heidelberg.

Doornik, J. A. and Ooms, M. (2003). Computational Aspects of Maximum Likelihood Estimation of Autoregressive Fractionally Integrated Moving Average Models. *Computational Statistics and Data Analysis*, 42:333–348.

Hosking, J. R. M. (1981). Fractional differencing. *Biometrika*, 68:165–176.

Kasdin, N. J. (1995). Discrete simulation of colored noise and stochastic processes and $1/f^{\alpha}$ power-law noise generation. *Proc. IEEE*, 83(5):802–827.

Langbein, J. (2004). Noise in two-color electronic distance meter measurements revisited. *J. Geophys. Res.*, 109(B04406).

Langbein, J. and Bock, Y. (2004). High-rate real-time GPS network at Parkfield: Utility for detecting fault slip and seismic displacements. *Geophys. Res. Letters*, 31:15.

Sowell, F. (1992). Maximum likelihood estimation of stationary univariate fractionally integrated time series models. *J. Econom.*, 53:165–188.

Williams, S. D. P. (2003). The effect of coloured noise on the uncertainties of rates from geodetic time series. *J. Geod.*, 76(9-10):483–494.

Williams, S. D. P. (2008). CATS : GPS coordinate time series analysis software. *GPS Solutions*, 12(2):147–153.

Zinde-Wash, V. (1988). Some Exact Formulae for Autoregressive Moving Average Processes. *Econometric Theory*, 4(3):384–402.

A History of changes made in the various versions of Hector

A.1 Version 1.1

1. The programs `estimatetrend`, `removeoutliers`, `estimatespectrum` and `modelspectrum` now accept the name of the control file on the command line. For example:

```
estimatetrend mycontrol.ctl
```

2. The date of the nominal bias of the whole time-series can now be set in estimatetrend.ctl using the keyword 'ReferenceEpoch'. Furthermore, in the output this date is now shown in "day/month/year hour" format instead of Modified Julian Date. If this keyword is not provided, then the default behaviour is to define the ReferenceEpoch as the midpoint between the start and end time of the observations.

3. The ability to leave out a keyword also made it possible to define other default parameters. Now it is no longer necessary to provide the ScaleFactor keyword if this is not different from 1 and the LikelihoodMethod keyword is optinal. If it is missing, then the AmmarGrag method will be used when the amount of missing data is less than 50% of the whole time-series. Otherwise the FullCov method is used.

4. The ARFIMA model had two bugs, one due to a sign error and one due to accessing arrays outside their range, which have been resolved.

5. The keyword 'MinimizingMethod' has been removed because the Nelder-Mead Simplex method is the only method available.

6. The header information about the sampling period and the time of offsets can now be given in a separate file using the keyword 'OffsetFile'.

7. It is now also possible to estimate a quadratic polynomial by setting the keyword 'QuadraticTerm' to yes.

8. The program `simulatenoise` was added.

9. Implemented the dd-mm-year NaN NaN NaN offset format for external files.

A.2 Version 1.2

1. `simulatenoise` has now correct power.

2. The ARMA and ARFIMA noise model now also accept first-difference (removed from version 1.5 onwards).

3. All programs now except ridiculously long names.

A.3 Version 1.3

1. Removed a bug which caused Hector to crash on some computers. The reason was that Nnumbers was not set to zero explicitly.

2. The parser of the control files had trouble when there was a space after the last label. For example "NoiseModels PowerlawApprox White " where there is a space behind the last word. This has now hopefully been corrected in Control.cpp

3. Using the Generalized Gauss Markov noise model did not converge in rare situations but they did occur. Now the maximum value of ϕ has been lowered from 0.9999 to 0.999.

4. The binaries are now stored in `/usr/local/bin` instead of `/usr/bin` which we hope is more following the Linux standard.

A.4 Version 1.4

1. Better command line parsing for estimatespectrum.

2. Added "# sampling period 1.0" in .enu file created by `convert_sol_files.tcl`.

3. Removed root-message in ARFIMA.cpp.

4. Removed trailing spaced bug in Control.cpp (again).

5. Improved C++ correctness (fp.getline(..)!=NULL) in Observations.cpp

A.5 Version 1.5

1. Removed the option to apply first difference to data. This option was never used and it makes the source code unnecessary complicated.

2. Added Hann window to `estimatespectrum`. You can now chose between the Hann and Parzen window function and select the percentage of data to which this window will be applied at both ends of the segment.

3. Implemented a Taylor expansion of the Hypergeometric function in GenGauss-Markov.cpp and now ϕ can be closer to 1: 0.9999. This helps to simulate pure power-law noise.

4. Changed the output of estimatetrend by elminating parameters d, innovation noise and fractions. The noise amplitudes now follow what is normally quoted in the literature.

5. The Plate Boundary Observatory changed their time series file format. We find it easier to write a little script to convert the new format to my mom-format than to update the Observations.cpp class.

6. Removed subsection "Some additional tests" since it was not read or created confusion for those who did.

A.6 Version 1.5.1

1. Removed a small bug in the class Minimizer.cpp that left the first noise parameter 0.001 larger than the optimal value (residual of computing Fisher information matrix). It could have a large effect when "Random Walk" was the first chosen noise model.