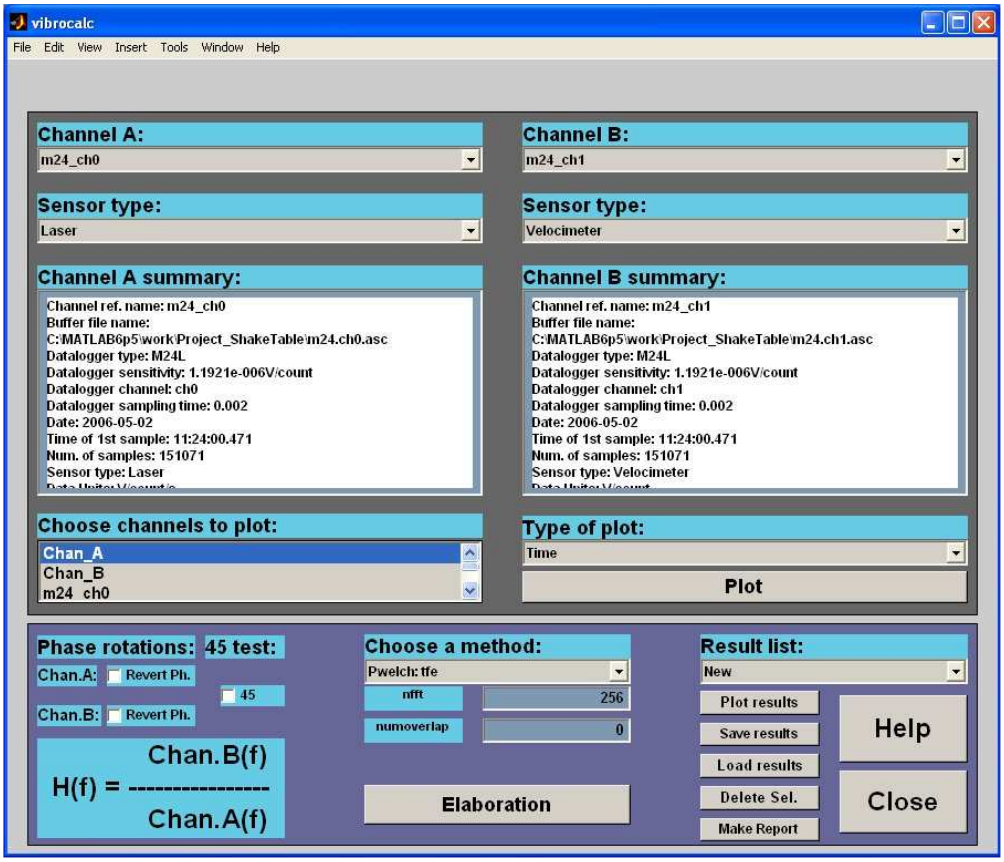
Istituto Nazionale di Oceanografia e di Geofisica Sperimentale
Centro di Ricerche Sismologiche di Udine

VIBROCALC
Relazione tecnica: Seismic sensor frequency response elaboration software.



Rapporto redatto da: D. Zuliani, E. Diez

Il Direttore del Dipartimento CRS: Dr. E. Priolo

Rel. OGS-076/2006/CRS-017
29 agosto 2006

Objective:

To describe how VIBROCALC works. VIBROCALC is a computer program written using MATLAB graphical interface for frequency response calculations (module and phase) of seismic sensors, geophones, seismometers and accelerometers.

Besides, examples of the use of the program for different types of sensors are shown.

The VIBROCALC software is part of the calibration system used at the “Centre di Ricerche Seismologiche (CRS)”, of the “Geophysical Observatory Sperimentale (OGS)”, for the seismic sensors calibration, in order to obtain the seismometer response curve (module and phase) of the Friuli seismic monitoring network sensors.

This software, works connected to a shake table according to the block diagram shown in the figure 1:

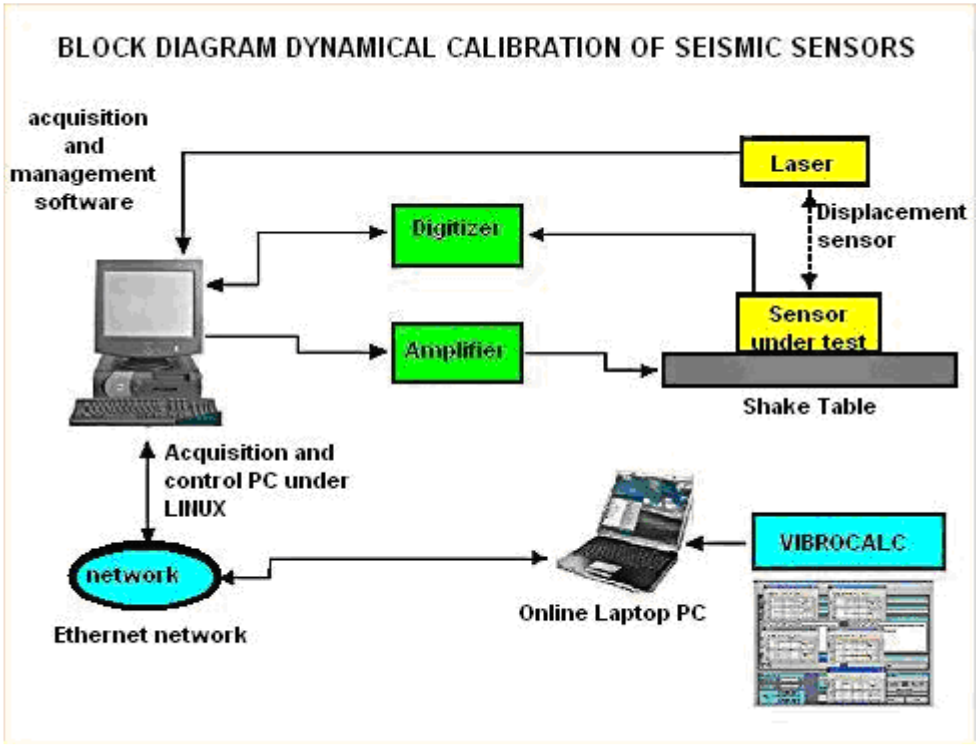


Figure 1.Block diagram of calibration direct method developed and in use at CRS.

This block diagram and all its part are described within the technical report ¨OGS-11/2005/CRS-3¨ (Paolo Di Bartolomeo, Fausto Ponton, Cristiano Urban, David Zuliani), the current report is limited to the final step of the system, the elaboration software.

VIBROCALC, is a script written under MATLAB 6.5 platform, that permits the creation of relatively friendly user graphic interface and compatible (in form and design), with the WINDOWS environment.

The script yields the transfer function between two input signals called channel A and channel B. In case of the system implemented at CRS, they refer to files with extension ASC, which include the signals, previously digitized, one, the sensor coil output under test and another, to the laser position sensor that controls the effective displacement of the seismometer.

Both inputs is referring to correspond to ASCII files which are recorded in a specific locations of the PC that contains the shake table management software and are available for any computer installed around the network.

The structure of the input channels is shown in the figures 2a and 2b:

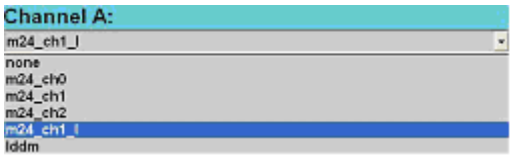


Figure 2a.Channel A selection

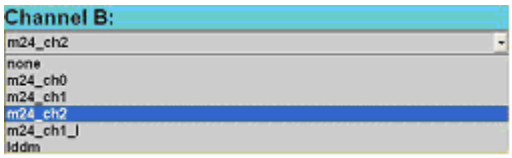


Figure 2b.Channel B selection

In both cases, it is possible to select any type of signal available from the digitizer; this means sensor components under test or sensor laser output. When no channel selected, the system will not work correctly and an error message will be appear.

The software then needs to defined, which type of sensors are being used, laser, seismometer or accelerometer, this is because of the automatic adjustment of the measurement units (figure 3). The programs also carry out the automatic adjustment of the source file length, in fact the mathematical models; used inside the script, need the same length in any combinations.

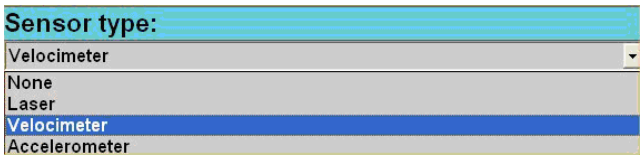


Figure 3.Type of sensor selection popup.

For each channel selected, the following values are shown:

- Reference channel name.
- Path of file.
- Type of digitizer.
- Channel of the digitizer.
- Sampling frequency used.
- Recorded dates.
- Time of the first sample.
- Number of samples.
- Type of sensor.
- Measurement units.

All these values are included in the acquired file. As it is shown in the figures 4a and 4 b, summary boxes report informative function.

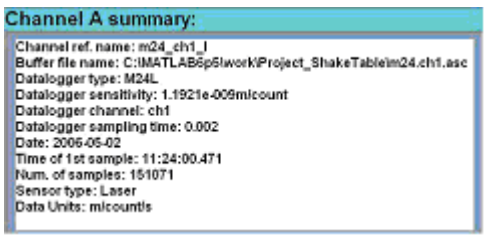


Figure 4a.Channel A summary

Figure 4b.Channel B summary

Furthermore, the program enables you to plot the different input signals, channel A and/or B or laser, or its combination (figure 5).

During the plot session you can also select the plot domain of signals in function of time or frequency. The control of the inputs allows you to know if, really, the signals concerning to sensors under test have been supplied correctly to the calculation program, (figure 6).

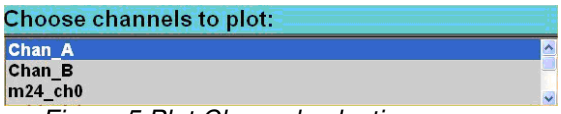


Figure 5.Plot Channel selection popup.

Figure 6.Domain selection popup.

The graphical representation of input channels are shown in figures 7, 8, 9

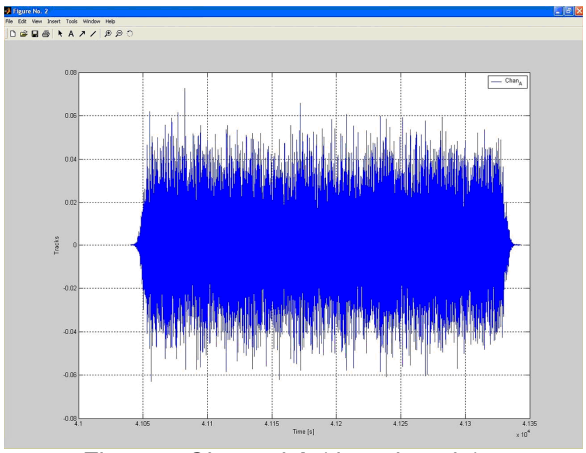


Figure 7.Channel A (time domain).

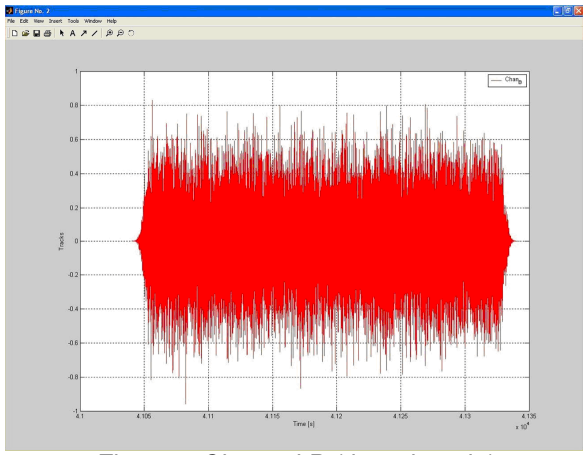


Figure 8.Channel B (time domain).

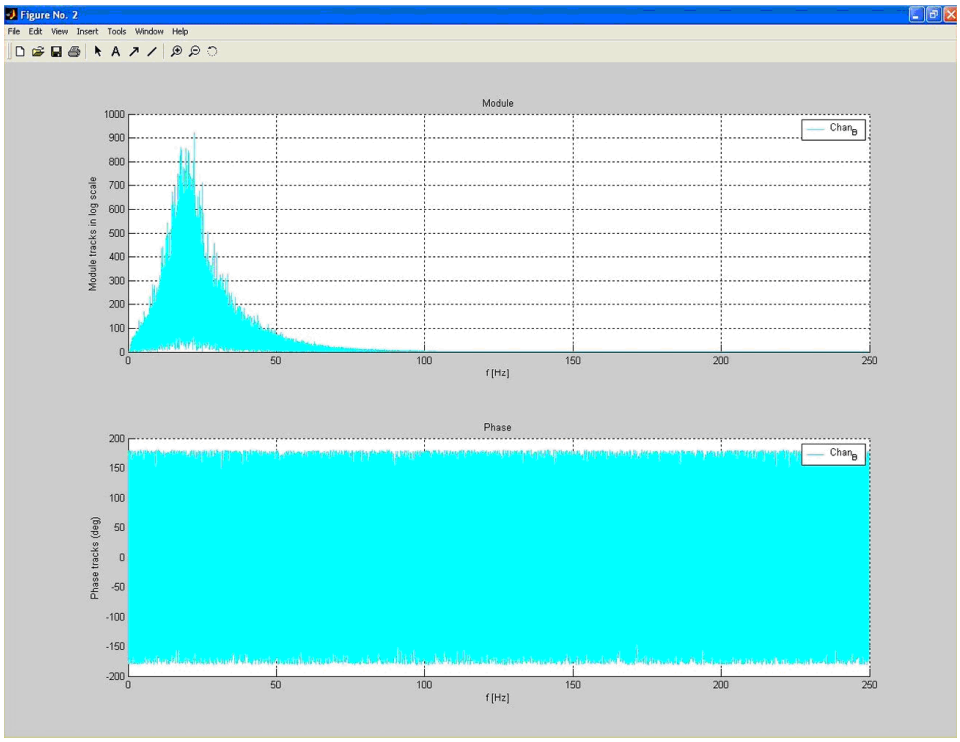


Figure 9.Example of input signal in frequency domain.

Some other adjustments are then possible for the input signals, before the transfer function elaboration; (see the figure 10). They are the following:

- Phase change of channel A.
- Idem for channel B.
- Necessary adjustment for simultaneous calibration of triaxial sensor's horizontal components (45^{o} adjustment).

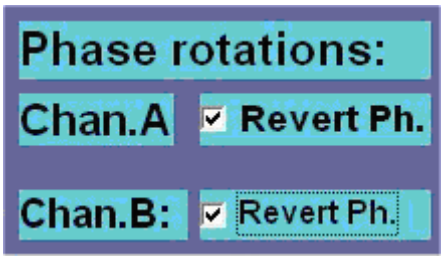


Figure 10.Phase change check box

The transfer function calculation is carried out using 6 different mathematical methods, each of then one has specific parameters. It is possible to select among these methods as well as to change the calculation parameters, (see figures 11 and 12).

The methods used are (for more details, see annex):

- (TFE), Transfer Function Estimate.
- (TFE modified).
- (SPA), Spectral Analysis.
- (EFTE), Empirical Estimate.
- (N4SID), State-space Model Estimate.
- (PEM), Linear Model Estimate.

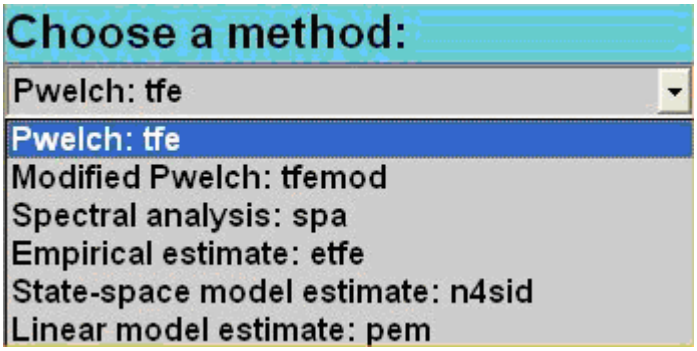


Figure 11.Mathematical method selection popup.

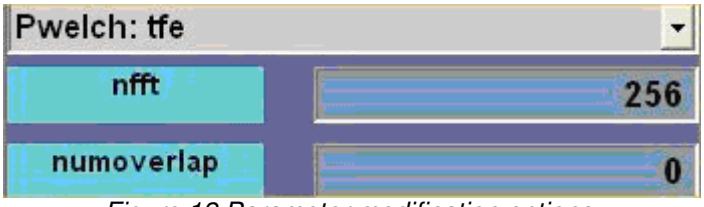


Figure 12.Parameter modification options.

The result of the elaboration, after the previous adjustments and settings, is stored in some MATLAB temporary variables and added to a list which can contain different results coming from calibrations previously made. The program the enables the user to introduce some info that will be inserted in the graphics and final reports, they are:

- Name of the sensor (identification).
- Serial Number.
- Limits of frequency for which the report is desired.

From the list, it is possible to carry out a lot of functions as following:

- To plot the frequency response curve in module and phase, with options to rescale the plot (e.g. maximum and minimum values), figures 13a and 13b. **Plot results**
- Save the results in a recognizable, by the program, file for further analysis, figure 14. **Save results**
- To load saved calibration files, figure 15. **Load results**
- To erase files from calibration list in case of error or disagreement with results obtained. **Delete Sel.**
- To generate an ASCII report in file. TXT. **Make Report**

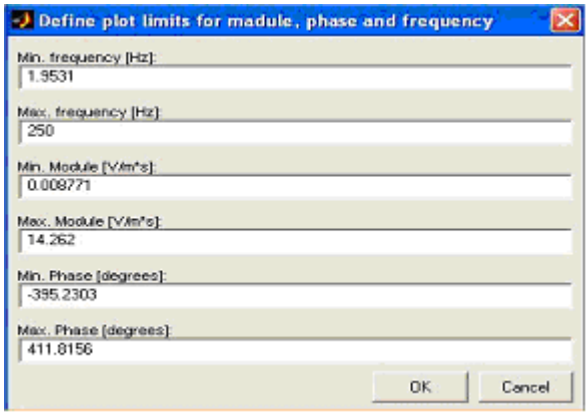


Figure 13a.Plot limits definition.

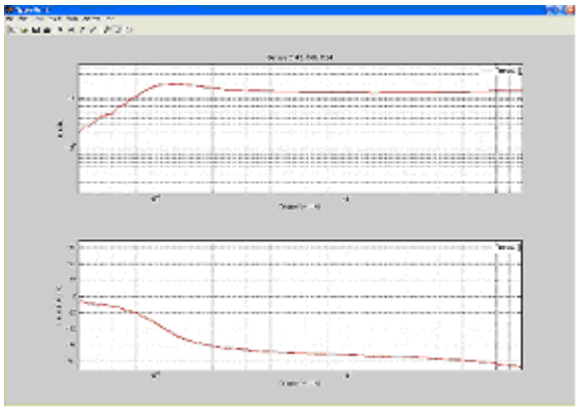


Figure 13b.Plot window.

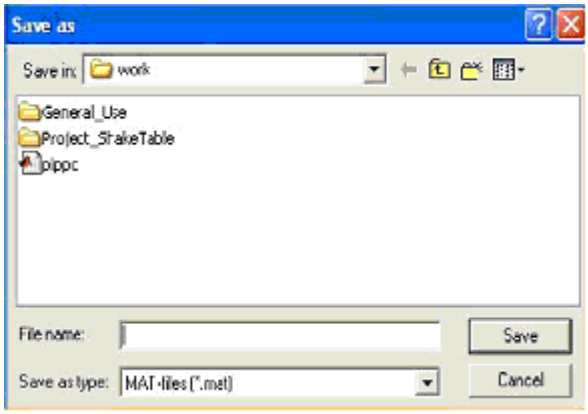


Figure 14.Save window.

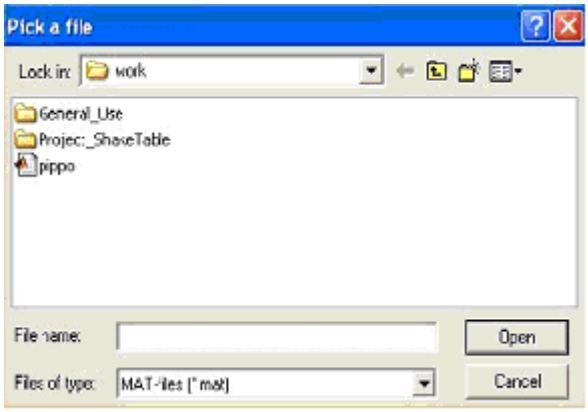


Figure 15.Load window.

Finally, VIBROCALC has a help file for detailed explanation of all the mathematical functions and the algorithms used.

How to use VIBROCAL??

This charter includes the information you need to install and operate the VIBROCALC software. It includes example of sensor calibration using short period seismometer.

Before you start using VIBROCALC, you have to install it in your computer and be sure that the configuration Matlab file allows the program to download data from the shake table management LINUX computer. It's also necessary to execute the procedure explained in the OGS-11/2005/CRS-3 (Paolo Di Bartolomeo, Fausto Ponton, Cristiano Urban, David Zuliani) report, for seismic sensors calibration.

When you have recorded from digitizer the data corresponding to laser and sensor coil, you can go ahead with the program.

In channel A window, select the signal corresponding to the laser sensor output, which corresponds to the displacement of the shake table (and sensor). In the channel B window, select the coil output of the seismometer.

Once you're checked, on the information boxes, for the channel data loaded and, you have verified that they correspond to the wanted signals, it you can choose to carry out any phase change or carrying out the elaboration for the simultaneous calibrations of the horizontal components of the seismometer under test (in case it is a triaxial type).To perform this last method mark the 45° checking box for the automatic magnitude adjustments, (see figure 16).

Figure 16.For horizontal components calibration.

In all cases, you can plot one or both signals in the time or in the frequency domain and you can also print them from MATLAB (see figures 7, 8 and 9).

After this, we can select the mathematical method which will be used for the calculation of the frequency response by means of a set of definite MATLAB algorithms (mathematical tools library, see appendix 2).

According to the specific interests of each analysis, the program allows you to select the more powerful method for the elaboration. For each of then, the parameters can be introduced manually, otherwise, they are assumed, by the program, from default values established in the algorithm.
If you are not convinced about parameter values, leave empty all boxes.

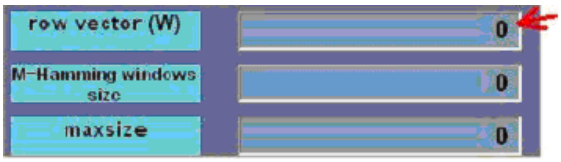


Figure 17.Parameter definition.

After the parameter setup, you have to push the button 

to start the calculus procedure, in few seconds it will appear the following dialog box:

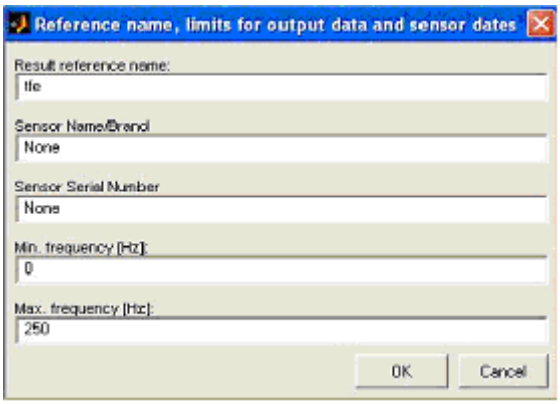


Figure 18.Sensor data and frequency limits.

The program take into account, as a reference, the name of the used method, but you may change it as you want, the dialog box also contains input for other info's, for example: sensor name or manufacturer name, serial number as well as the lower and the upper limits used to record the elaborated data.

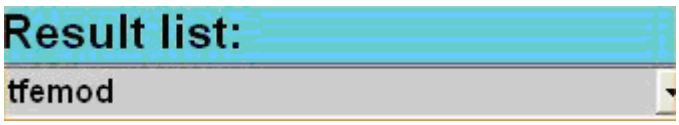


Figure 19.Result list of performed calculus.

When the elaboration is done, the result will be present in the result list (Figure 13) with the reference name you're choose in the previous box (Figure 18).

With the obtained result, push down the plot button

The following dialog box:

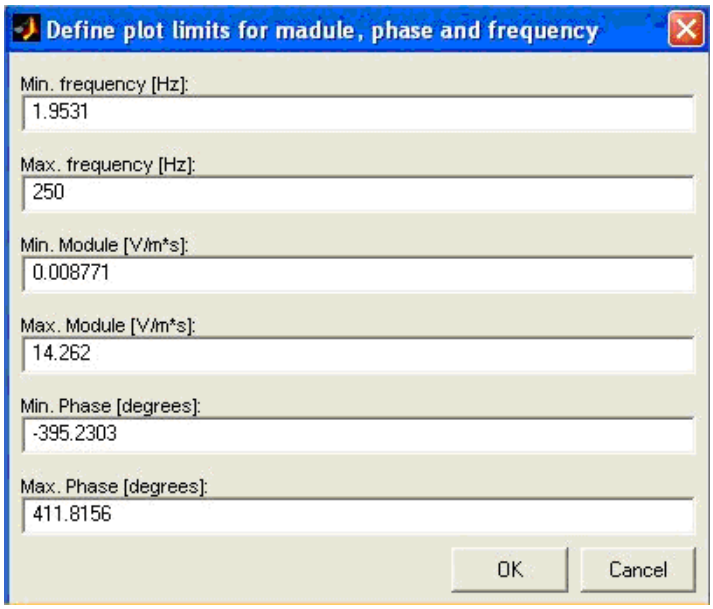


Figure 20.Plot limits dialog box.

This box is designed to allow you to set up the diagram limits; it means minimum and maximum frequency, module and phase.
Note that you need to be sure about these limits, especially in case of short period seismometers, which has a limited band pass and in which, outside this bandwidth, you can expect unusual or wrong draws.

Pressing the OK button of the box in figure 20, the program will draws the corresponding wave diagram, limited to selected frequency, module and phase values.

If you need to save the result file, then push down the following button:

The window of the figure 14 will be opened; in this dialog box you can choose the path for recording your results into a file (binary).

Furthermore, you can generate TXT file report including input channel data and final calibration values; this ASCII file report is useful when you need by push down the button:

This action generates automatically, a new dialog box in which it is possible to choose the variables that it is going to be registered in this report, and it can be use as selected as follow:

- To include or not, in the final report, the data coming out from input signals, by default (N) the program does not include them.
- To define a set of spans where the frequency response is going to be extrapolate. In each span you can choose different frequency steps defined in the next point. If you leave the value to NONE, the program will not perform any extrapolation.
- To choose the extrapolation frequency step as mentioned before.

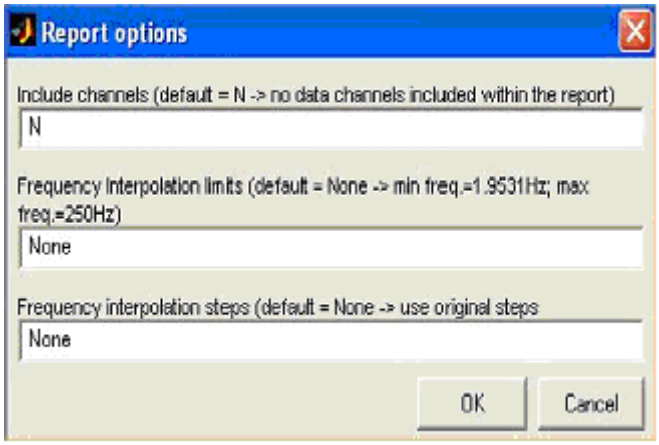


Figure 21.Report option dialog box

The reports are recorded like ASCII files (.TXT) in the path: MATLAB\work\. TXT and their structure are shown in the following figures:

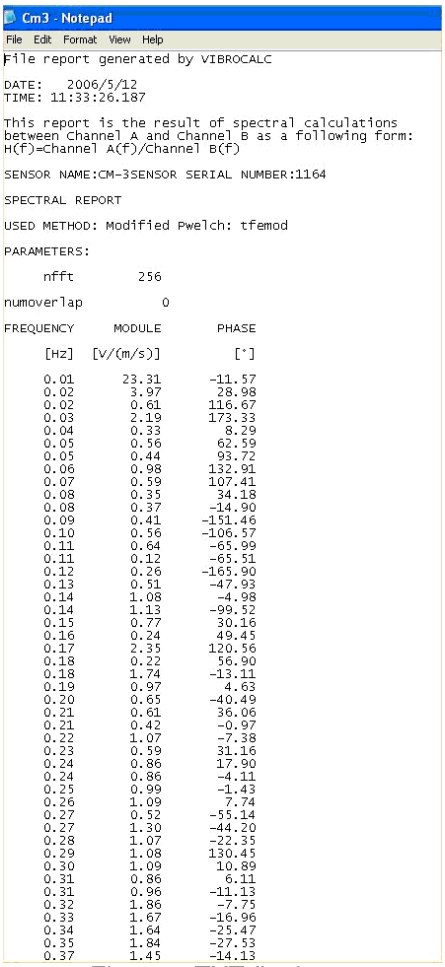


Figure 2.TXT final report.

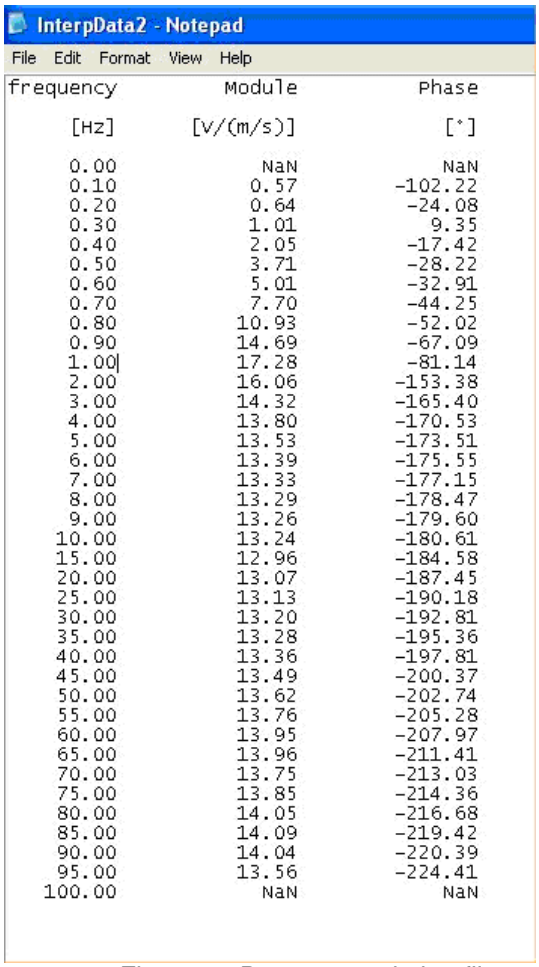


Figure 23.Data extrapolation file.

Finally, if you need some help, push down the correspondent button:

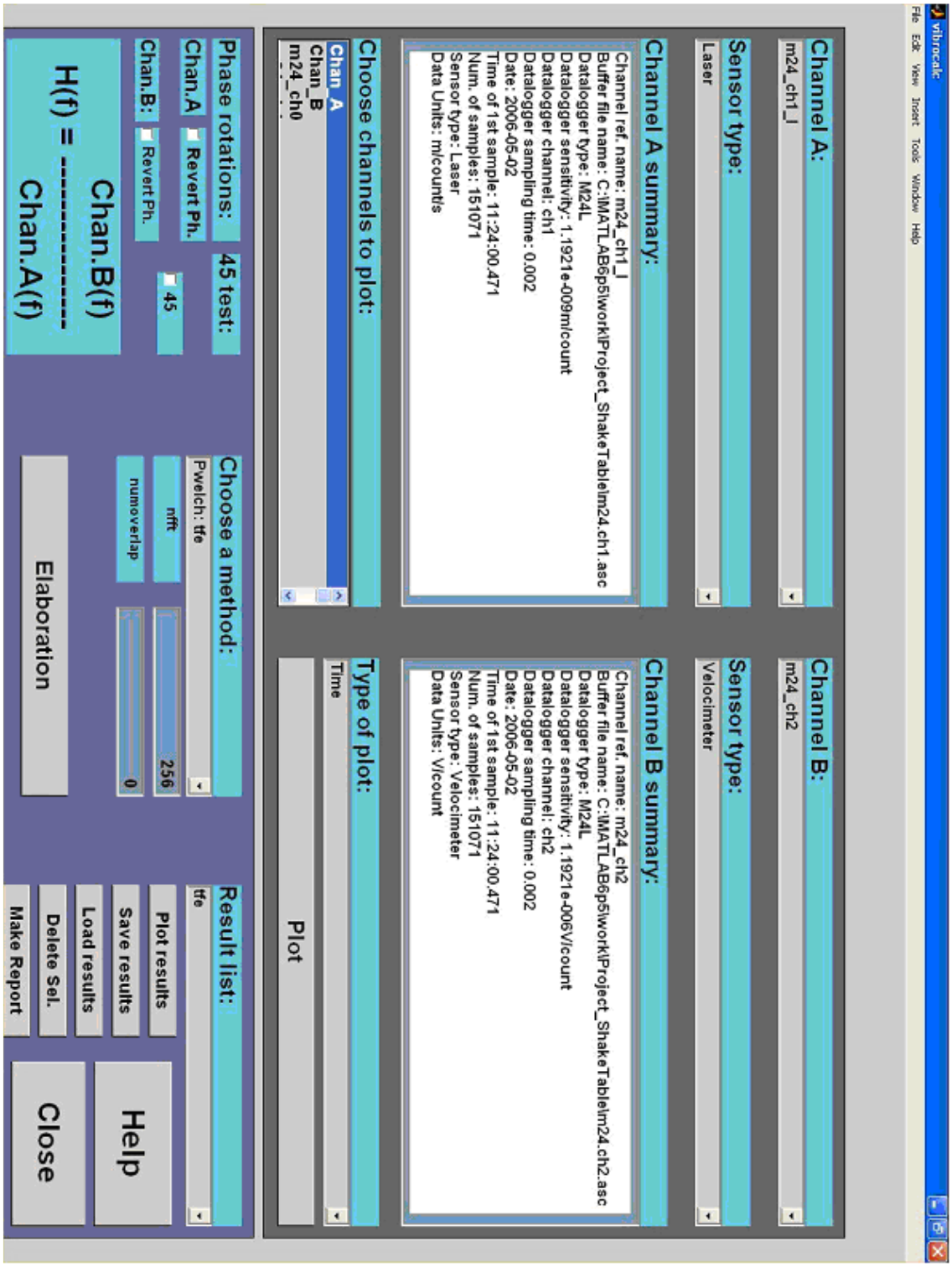
And for exit from VIBROCALC, press

References

- *Bulfon, A., Ponton, F., Duri, G. (1992). Calibraziones dei Geofoni della rete sismometrica del Friuli, procedimiento e risultati. Rapporto interno Rel. I-92-21/CRS-8, OGS Trieste.*
- *MATLAB 6.5 Manual.*
- *Paolo Di Bartolomeo, Fausto Ponton, Cristiano Urban, David Zuliani (2005). OGS-11/2005/CRS-3 Technical Report.*
- *Ponton, F., Di Bartolomeo, P., Govoni, A., Zuliani, D. (2003). Seismometer calibration at OGS, presented at EGS, Génova.*
- *Scherbaum, F. (1996). Of poles and zeros, fundamentals of digital seismology, Kluwer academic publishers, Netherlands.*

APPENDIX

Appendix 1. VIBROCALC main window.



Appendix 2. Main mathematical methods used in VIBROCALC, (extracted from MATLAB 6.5 help).

TFE

Estimate the transfer function from input and output

Syntax:
Txy = tfe(x,y)
Txy = tfe(x,y,nfft)
[Txy,f] = tfe(x,y,nfft,fs)
Txy = tfe(x,y,nfft,fs,window)
Txy = tfe(x,y,nfft,fs,window,numoverlap)
Txy = tfe(x,y,...,'dflag')
tfe(x,y)

Description:

Txy = tfe(x,y) finds a transfer function estimate Txy given input signal vector x and output signal vector y. The transfer function is the quotient of the cross spectrum of x and y and the power spectrum of x.

$$T_{xy}(f) = \frac{P_{xy}(f)}{P_{xx}(f)}$$

The relationship between the input x and output y is modeled by the linear, time-invariant transfer function Txy. Vectors x and y must be the same length. Txy = tfe(x,y) uses the following default values:

- *nfft = min(256,(length(x))*
- *fs = 2*
- *window is a periodic Hann (Hanning) window of length nfft*
- *numoverlap = 0*

nfft specifies the FFT length that tfe uses. This value determines the frequencies at which the power spectrum is estimated.
fs is a scalar that specifies the sampling frequency.
window specifies a windowing function and the number of samples tfe uses in its sectioning of the x and y vectors.
numoverlap is the number of samples by which the sections overlap. Any arguments that are omitted from the end of the parameter list use the default values shown above. If x is real, tfe estimates the transfer function at positive frequencies only; in this case, the output Txy is a column vector of length nfft/2+1 for nfft even and (nfft+1)/2 for nfft odd.
If x or y is complex, tfe estimates the transfer function for both positive and negative frequencies and Txy has length nfft. Txy = tfe(x,y,nfft) uses the specified FFT length nfft in estimating the transfer function.
[Txy,f] = tfe(x,y,nfft,fs) returns a vector f of frequencies at which tfe estimates the transfer function. fs is the sampling frequency. f is the same size as Txy, so plot(f,Txy) plots the transfer function estimate versus properly scaled frequency. fs has no effect on the output Txy; it is a frequency scaling multiplier.
Txy = tfe(x,y,nfft,fs,window) specifies a windowing function and the number of samples per section of the x vector. If you supply a scalar for window, Txy uses a Hann window of that length. The length of the window must be less than or equal to nfft; tfe zero pads the sections if the length of the window exceeds nfft.
Txy = tfe(x,y,nfft,fs,window,numoverlap) overlaps the sections of x by numoverlap samples.
You can use the empty matrix [ ] to specify the default value for any input argument except x or y.
For example, Txy = tfe(x,y,[ ],[ ],kaiser(128,5))
uses 256 as the value for nfft and 2 as the value for fs.
Txy = tfe(x,y,...,'dflag') specifies a detrend option, where 'dflag' is

- *'linear', to remove the best straight-line fit from the prewindowed sections of x and y*
- *'mean', to remove the mean from the prewindowed sections of x and y*
- *'none', for no detrending (default)*

The 'dflag' parameter must appear last in the list of input arguments. tfe recognizes a 'dflag' string no matter how many intermediate arguments are omitted. tfe(...) with no output arguments plots the magnitude of the transfer function estimate as decibels versus frequency in the current figure window.

SPA

Estimate frequency response and spectrum by spectral analysis.

Syntax:
g = spa(data)
g = spa(data,M,w,maxsize)
[g,phi,spe] = spa(data)

Description:

spa estimates the transfer function g and the noise spectrum Φ_v *of the general linear model*

$$y(t) = G(q)u(t) + v(t)$$

where $\Phi_v(\omega)$ *is the spectrum of* $v(t)$ *.*

Data contains the output-input data as an iddata object. The data may be complex-valued. g is returned as an idfrd object with the estimate of $G(e^{i\omega})$ *at the frequencies specified by row vector w.*
The default value of w is
*w = [1:128]/128*pi/Ts*
Here Ts is the sampling interval of data. g also includes information about the spectrum estimate of $\Phi_v(\omega)$ *at the same frequencies.*
Both outputs are returned with estimated covariances, included in g. See idfrd.
M is the length of the lag window used in the calculations. The default value is
M = min(30,length(data)/10)
Changing the value of M exchanges bias for variance in the spectral estimate. As M is increased, the estimated functions show more detail, but are more corrupted by noise.
The sharper peaks a true frequency function has, the higher M it needs. See etfe as an alternative for narrowband signals and systems.
For time series, where data contains no input channels, g is returned with the estimated output spectrum $\Phi_v(\omega)$ *and its estimated standard deviation. When spa is called with two or three output arguments:*

- *g is returned as an idfrd model with just the estimated frequency response from input to output and its uncertainty.*
- *phi is returned as an idfrd model, containing just the spectrum data for the output spectrum and its uncertainty.*
- *spe is returned as an idfrd model containing spectrum data for all output-input channels in data. That is if z = [data.OutputData, data.InputData], spe contains as spectrum data the matrix-valued power spectrum of z.*

$$S = \sum_{m=-M}^{M} Ez(t+m)z(t)'\exp(-iWmT)win(m)$$

Here win(m) is weight at lag m of an M-size Hamming window and W is the frequency value i rad/s. Note that ' denotes complex-conjugate transpose. The normalization of the spectrum differs from the one used by spectrum in the Signal Processing Toolbox. See Spectrum Normalization and the Sampling Interval in the "Tutorial" for a more precise definition.

ETFE

Estimate empirical transfer functions and periodograms.

Syntax:
g = etfe(data)
g = etfe(data,M,N)

Description:

etfe estimates the transfer function g of the general linear model

$$y(t) = G(q)u(t) + v(t)$$

data contains the output-input data and is an iddata object.

g is given as an idfrd object with the estimate of $G(e^{i\omega})$ *at the frequencies*

*w = [1:N]/N*pi/T*

The default value of N is 128.
In case data contains a time series (no input channels), g is returned as the periodogram of y.
When M is specified other than the default value M = [ ], a smoothing operation is performed on the raw spectral estimates.
The effect of M is then similar to the effect of M in spa.
This can be a useful alternative to spa for narrowband spectra and systems, which require large values of M.
*When etfe is applied to time series, the corresponding spectral estimate is normalized in the way that is defined in the section Spectrum Normalization and the Sampling Interval in the Tutorial. Note that this normalization may differ from the one used by spectrum in the Signal Processing Toolbox. If the (input) data is marked as periodic (data.Period = integer) and contains an even number of periods, the response is computed at the frequencies k*2*pi/period for k=0 up to the Nyquist frequency.*

Algorithm:

The empirical transfer function estimate is computed as the ratio of the output Fourier transform to the input Fourier transform, using fft. The periodogram is computed as the normalized absolute square of the Fourier transform of the time series. The smoothed versions (M less than the length of z) are obtained by applying a Hamming window to the output fast Fourier transform (FFT) times the conjugate of the input FFT, and to the absolute square of the input FFT, respectively, and subsequently forming the ratio of the results. The length of this Hamming window is equal to the number of data points in z divided by M, plus one.

N4SID

Estimate state-space models using a subspace method.

Syntax:
m = n4sid(data)
m = n4sid(data,order,'Property1',Value1,...,'PropertyN',ValueN)

Description:

The function n4sid estimates models in state-space form, and returns them as an idss object m. It handles an arbitrary number of input and outputs, including the time-series case (no input). The state-space model is in the innovations form

$$x(t+Ts) = Ax(t) + Bu(t) + Ke(t)$$
$$y(t) = Cx(t) + Du(t) + e(t)$$

m: The resulting model as an idss object.
data: An iddata object containing the output-input data.
order: The desired order of the state-space model. If order is entered as a row vector (like order = [1:10]), preliminary calculations for all the indicated orders are carried out. A plot will then be given that shows the relative importance of the dimension of the state vector. More precisely, the singular values of the Hankel matrices of the impulse response for different orders are graphed. You will be prompted to select the order, based on this plot. The idea is to choose an order such that the singular values for higher orders are comparatively small. If order = 'best', a model of "best" (default choice) order is computed, among the orders 1:10. This is the default choice of order.
idss properties that are of particular interest for n4sid are:
nk: The delays from the inputs to the outputs, a row vector with the same number of entries as the number of input channels. Default is nk = [1 1 ... 1]. Note that delays being 0 or 1 show up as zeros or estimated parameters in the D matrix. Delays larger than 1 means that a special structure of the A, B and C matrices are used to accommodate the delays. This also means that the actual order of the state-space model will be larger than order.

- *CovarianceMatrix (can be abbreviated to 'co'): Setting CovarianceMatrix to 'None' will block all calculations of uncertainty measures. These may take the major part of the computation time. Note that, for a 'Free' parameterization, the individual matrix elements cannot be associated with any variance (these parameters are not identifiable). Instead, the resulting model m stores a hidden state-space model in canonical form, that contains covariance information. This is used when the uncertainty of various input-output properties are calculated. It can also be retrieved by m.ss = 'can'. The actual covariance properties of n4sid estimates are not known today. Instead the Cramer-Rao bound is computed and stored as an indication of the uncertainty. DisturbanceModel: Setting DisturbanceModel to `None' will deliver a model with K = 0. This will have no direct effect on the dynamics model, other that that the default choice of N4Horizon will not involve past outputs.*
- *InitialState: The initial state is always estimated for better accuracy. However. it is returned with m only if InitialState = `Estimate'.*

Algorithm properties that are special interest are:

- *Focus: Assumes the values 'Prediction' (default), 'Simulation', ` Stability', or any SISO linear filter.Setting 'Focus' to 'Simulation' chooses weights that should give a better simulation performance for the model. In particular, a stable model is guaranteed. Selecting a linear filter will focus the fit to the frequency ranges that are emphasized by this filter.*
- *N4Weight: This property determines some weighting matrices used in the singular-value decomposition that is a central step in the algorithm. Two choices are offered: 'MOESP' that corresponds to the MOESP algorithm by Verhaegen, and 'CVA' which is the canonical variable algorithm by Larimore. The default value is 'N4Weight' = 'Auto', which gives an automatic choice between the two options. m.EstimationInfo.N4Weight tells you what the actual choice turned out to be. N4Horizon: Determines the prediction horizons forward and backward, used by the algorithm. This is a row vector with three elements:*
- *N4Horizon =[r sy su], where r is the maximum forward prediction horizon, i.e., the algorithms uses up to r-step ahead predictors. sy is the number of past outputs, and su is the number of past inputs that are used for the predictions. See pages 209-210 in Ljung(1999) for the exact meaning of this. These numbers may have a substantial influence of the quality of the resulting model, and there are no simple rules for choosing them. Making 'N4Horizon' a k-by-3*

matrix, means that each row of 'N4Horizon' will be tried out, and the value that gives the best (prediction) fit to data will be selected. (This option cannot be combined with selection of model order.) If the property 'Trace' is 'On', information about the results will be given in the MATLAB command window. If you specify only one column in 'N4Horizon', the interpretation is r=sy=su. The default choice is 'N4Horizon' = 'Auto', which uses an Akaike Information Criterion (AIC) for the selection of sy and su. If 'DisturbanceModel' = 'None', sy is set to 0. Typing m.EstimationInfor.N4Horizon will tell you what the final choice of horizons were.

PEM

Estimate the parameters of general linear models.

Syntax:
m = pem(data)
m = pem(data,mi)
m = pem(data,mi,'Property1',Value1,...,'PropertyN',ValueN)
m = pem(data,orders)
m = pem(data,'nx',ssorder)
m = pem(data,'na',na,'nb',nb,'nc',nc,'nd',nd,'nf',nf,'nk',nk)
m = pem(data,orders,'Property1',Value1,...,'PropertyN',ValueN)

Description:
pem is the basic estimation command in the toolbox and covers a variety of situations.
data is always an iddata object that contains the input/output data.
With Initial Model
mi is any idmodel object, idarx, idpoly, idss, or idgrey.
It could be a result of another estimation routine, or constructed and modified by the constructors (idpoly, idss, idgrey) and set. The properties of mi can also be changed by any property name/property value pairs in pem as indicated in the syntax.
m is then returned as the best fitting model in the model structure defined by mi.The iterative search is initialized at the parameters of the initial/nominal model mi. m will be of the same class as mi.
Black-Box State-Space Models
With m = pem(data,n), where n is a positive integer, or m = pem(data,'nx',n) a state-space model of order n is estimated. The default situation is that it is estimated in a 'Free' parameterization, that can be further modified by the properties 'nk', 'DisturbanceModel', and 'InitialState' (see the reference pages for idss and n4sid). The model is initialized by n4sid, and then further adjusted by optimizing the prediction error fit.
You can choose between several different orders by
m = pem(data,'nx',[n1,n2,...nN])
and you will then be prompted for the "best" order. By
m = pem(data,'best')
an automatic choice of order among 1:10 is made.
m = pem(data)
is short for
m = pem(data,'best').
To work with other delays use, e.g. m = pem(data,'best','nk',[0,...0]). In this case m is returned as an idss model.
Black-Box Multiple-Input-Single-Output Models
The function pem also handles the general multi-input-single-output structure

$$A(q)y(t) = \frac{B_1(q)}{F_1(q)}u_1(t-nk_1) + \ldots + \frac{B_{nu}(q)}{F_{nu}(q)}u_{nu}(t-nk_{nu}) + \frac{C(q)}{D(q)}e(t)$$

The orders of this general model are given either as
orders = [na nb nc nd nf nk]

or with (...'na',na,'nb',nb,...) as shown in the syntax. Here na, nb, nc, nd, and nf are the orders of the model and nk is the delay(s). For multi-input systems, nb, nf, and nk are row vectors giving the orders and delays of each input.
When the orders are specified with separate entries, those not given are taken as zero.
In this case m is returned as an idpoly object.

Properties:
In all cases the algorithm is affected by the properties

- *Focus, with possible values 'Prediction' (Default), 'Simulation' or a SISO filter (given as an LTI or idmodel object or as filter coefficients)*
- *MaxIter and Tolerance govern the stopping criteria for the iterative search.*
- *LimitError deals with how the criterion can be made less sensitive to outliers and bad data*
- *MaxSize determines the largest matrix ever formed by the algorithm. The algorithm goes into for-loops to avoid larger matrices, which may be more efficient than using virtual memory.*
- *Trace, with possible values 'Off', 'On', 'Full', that governs the information sent to the MATLAB command window.*

For black-box state-space models, also 'N4Weight' and 'N4Horizon' will affect the result, since these models are initialized with n4sid estimate.. Typical idmodel properties to affect are (see idmodel properties for more details): Ts, the sampling interval. Set 'Ts'=0 to obtain a continuous-time state-space model. For discrete-time models, 'Ts' is automatically set to sampling interval of the data. Note that, in the black box case, it is usually better to first estimate a discrete-time model, and then convert that to continuous time by d2c. nk, the time delays from the inputs (not applicable to structured state-space models). Time delays specified by 'nk', will be included in the model. DisturbanceModes determines the parameterization of K for free and canonical state-space parameterizations, as well as for idgrey models. InitialState. The initial state may have a substantial influence on the estimation result for system with slow responses. It is most pronounced for Output-Error models (K=0 for state-space, and na=nc=nd=0 for input/output models). The default value 'Auto', estimates the influence of the initial state and sets the value to 'Estimate', 'Backcast' or 'Zero', based on this effect. Possible values of 'InitialState' are 'Auto', 'Estimate', 'Backcast', 'Zero' and 'Fixed'.