NAME

evalresp − evaluate response information and output to ASCII files using *rdseed V4.16* and above RESP files.

SYNOPSIS

evalresp STA_LIST CHA_LIST YYYY DAY MIN_FREQ MAX_FREQ NFREQS [**−f** file] [**−u** units] [**−t** time−of−day] [**−s** type−of−spacing] [**−r** resp−type] [**−n** network−id] [**−l** location−id] [**−stage** start [stop]] [**−stdio**] [**−use−estimated−delay**] [**−unwrap**] [**-ts**] [**−il**] [**−ii**] [**−it** tension] [**−b62_x** x] [**−x**] [**−v**]

DESCRIPTION

Evalresp will calculate the complex response of a specified station or set of stations, channel or channels, and network, for a specified date, time and frequency, all by using the SEED ASCII response files produced by the program *'rdseed'*. *Evalresp* by itself returns the usage or 'help' lines useful for how to correctly input the options.

COMMAND-LINE PARAMETERS

−f file directory−name|filename
−u units 'dis'|'vel'|'acc'|'def'
−t time−of−day HH:MM:SS
−s type−of−spacing log|lin
−n netid 'II'|'IU'|'G'|'*'...
−l locid '01'|'AA,AB,AC'|'A?'|'*'...
−r resp_type 'ap'=amplitude/phase|'cs'=complex spectra|
'fap'=frequency/amplitude/phase
−stage start [stop] integer stage numbers
−stdio take input from stdin, output to stdout
−use−estimated−delay use estimated delay in computation of response
−il interpolate List blockette output
−ii interpolate List blockette input
−it tension tension for List blockette interpolation
−unwrap unwrap phase if the output is ap
(amplitude/phase)
−ts use total sensitivity from stage 0 instead
of computed
−b62_x value sample value/volts when we compute response
for B62
−v verbose; list parameters on stdout
−x xml; expect station.xml format

WHAT IS NEW

See ChangeLog file from evalresp distribution.

STATION.XML FORMAT INPUT

From version 4.0.0, input files in the station.xml for are converted to RESP format before processing if the -x flag is given.

The conversion is intended to be identical to that produced with the IRIS-WS package, and can be checked using the *xml2resp* program (installed alongside evalresp).

SUPPORT OF PRESSURE, MAGNETIC DATA AND TEMPERATURE

Evalresp is able to process responses of pressure instruments (in Pascals), temperature (in C) and magnetic flux density (in Tesla). Note that the default units are assumed to be Pascals, Centigrades, or Tesla respectively, therefore the "−u" option is ignored while processing pressure, temperature, and magnetic flux density responses.

SEED FILTER COEFFICIENTS AND STAGE DELAY CORRECTION

As of July 2007 the SEED standard was clarified to require that digital filter coefficients are to be listed in forward order. For a mathematical description see the SEED manual. As a reference, minimum−phase

filters (which are asymmetric) should be written with the largest values near the beginning of the coefficient list. Additionally the sign convention for the estimated delay and correction applied specified in Blockette 57 for a given stage were clarified. In almost all cases both the estimated delay and correction applied should be positive values and very nearly the same, if not exactly, the same value.

Proper evalresp operation requires the conventions described above to be followed. Unfortunately there are some SEED response descriptions which do not follow the convention. Using such responses with evalresp will likely result in improper response calculation. Furthermore, applying improper evalresp results to time series data can result in either obvious, or more problematic, subtle data modifications.

FIR FILTERS

All FIR filters are considered as having a zero phase−shift, even if they are not symmetrical and the delay correction is null. If there are 2 FIR filters in the same stage, the program assumes that both filters have the same input sample interval (in other words, the first filter has a decimation factor of 1). Typically if two FIR filters appear in the same stage, the second FIR filter is a continuation of the first. This often results when the FIR filter in question has more than 415 coefficients (which causes the length of the blockette containing the response information to exceed the "%4.4d" format of the blockette length specifier (defined by SEED). When this occurs, a second (continuation) blockette is written that contains the coefficients that would not fit in the first blockette. *evalresp* will handle such continuation filters by joining all FIR filters in the same stage into one large FIR filter in the order that they were scanned.

IIR FILTERS

Versions 3.2.17 and above support IIR digital filters in coefficients format with a non−zero phase shift. IIR coefficients for a single stage must fit in a single blockette.

DECIMATION BLOCKETTE

Without additional command−line option **−use−estimate−delay evalresp** computes a delay introduced by an asymmetrical FIR filter as correction_applied (Field 8 of B57) minus calculated delay.

A user can choose to run **evalresp** with **−use−estimate−delay** command line option.In this case estimated delay (field 7 of B57) will be used as a value of the delay for asymmetrical FIR filter phase computation.

A user is advised to select **−use−estimate−delay** option only if it is known that correction_applied (Field 8 of B57) is not properly set in the RESP file.

For all other filter types **evalresp** uses delay of 0 while processing decimation blockettes.

NON-LINEAR BLOCKETTE 62

In general, we cannot represent Blockette 62 as a Fourier response because it is intrinsically non−linear. See, for example, p28 of Bendat & Piersol, where they explicitly say that Fourier analysis is done on the assumption of a linear system (as defined p 26).

But there are two exceptions.

First, the polynomial may be linear (have only two terms, y = a x + b). In that case, it is (by definition) linear and can be treated as an offset ("a", ignored by the Fourier response) and gain ("b", the slope, independent of frequency). Note that the amplitude response is always positive (abs(b)); the phase will be zero if the slope of the curve is positive (b>0) and 180 degrees if negative (b<0).

Second, the user may have a signal that uses some part of the (non−linear) response curve that is sufficiently small (for the accuracy they require) that it can be treated as linear in that region. In this case we use a tangent (first order Taylor expansion) of the polynomial at the given point (and then treat that as the linear case above).

For a polynomial f(x) = a0 + a1 x + a2 x^2 + ... an x^n the slope at X is f'(X) where f' = a1 + 2 a2 x + 3 a3 x^2 ... n an x^(n−1). So the recipe to calculate the gain (b) at a given X is:

sum(i=1 to n) i * ai * x^(i−1)

In **evalresp−3.3.4** we require a user to input a particular value x as in the equation above using command line option **−b62_x**. This value is assumed to be in digital counts if B62 is the only blockette of the response or in volts if B62 is the part of the cascade.

The results of processing (response amplitude) depends on the valued selected by a user.

For responses without B62 command line option **−b62_x** is not needed

GENERIC RESPONSE BLOCKETTES

Versions 3.2.17 and above support generic response blockette (SEED blockettes 55). Generic response blockette is a list of phases and amplitudes computed for the preselected set of frequencies. This filter type is supported only if the response input file contains blockette(s) 55 as a stage 1 and possibly channel sensitivity blockette as a stage 0. If a generic response blockette is recognized in the input, *evalresp* ignores the user−defined frequency sampling from the command line. The ouput, therefore, contains responses for only those frequencies which have been defined in the generic response blockette.

FILTER SEQUENCE

The program assumes that the response information consists of a series of filter stages arranged in a cascade. It is assumed that the first filter in a given stage is one of the following: (1) A Laplace−Transform or Analog pole− zero filter, (2) an IIR pole−zero filter, (3) a FIR filter (either symmetric or asymmetric), or (4) a stand−alone gain blockette that indicates the overall sensitivity of the filter sequence (a stage zero filter). Versions of *evalresp 3.2.17* and higher also support (5) IIR digital coefficients filters and (6) provide limited support for Generic Response Blockette. It is further assumed that the filters will be followed by a gain blockette (except Generic Response Blockettes). If the stage is a decimation stage, then a decimation blockette will be included. This decimation blockette typically precedes the gain blockette for the stage in a SEED response file, although the order of the blockettes within a stage does not matter. If the blockettes within a stage are not in the order that evalresp expects to find them in, evalresp will rearrange them so that they appear in the "correct" order. If the response is a single stage response, *evalresp* will allow the user to specify an overall (stage 0) gain, rather than requiring the user to specify a stage 1 and stage 0 gain blockette (since, in this case, the stage 0 and stage 1 gains are identical).

The stage sequence number is checked by *evalresp* during parsing and any break in the sequence is considered to be an error. The result is that filter sequences with out of order stages are rejected as invalid responses. In addition, the output units of a stage and the input units of the next stage are compared by *evalresp*. If the output units of a stage do not match the input units of the next stage, the filter sequence is considered to be invalid and the response is rejected as an invalid response. The only exception to this rule are so called "gain−only" stages. Since these stages have no units associated with them, the *evalresp* program will skip them in determining the input units of the next stage. If a gain−only filter is found in the sequence, *evalresp* will scan to the next non−gain−only stage and compare the output units of the current stage with the input units of that stage. Again, a difference in the units will be considered to be an error in the filter sequence and cause that response to be rejected as invalid.

UNEXPECTED CASES:

- Stand alone FIR filters (i.e. those with no sample rate and gain specified) are discarded. (Only that stage is discarded, the rest of the filter sequence is kept and used to calculate a response).
- FIR filters which are not normalized to 1 at frequency 0 are normalized.
- IIR coefficients filter with a stage containing more than a single blockette 54.
- Mixing generic response stage with the other responses in a single file.

HOW THE PROGRAM SEARCHES FOR RESPONSES

If the '**−f**' option is specified, a determination is made as to whether the filename that follows the '**−f**' flag is a directory.

(1) If it is a directory, then that directory, and only that directory, is searched for files with names like RESP.NET.STA.LOC.CHA (or RESP.NET.STA.CHA), where the NET, STA, and CHA match the user supplied (or default) network−code, station names (from the STA_LIST), location−code, and channel names (from the CHA_LIST).

(2) If it is not a directory, then a file with that name is used as input to the program. That file, and only that file, will be searched for response information that matches the user's request.

(3) If the **−f** option is not specified, then both the current working directory and the directory pointed to by the SEEDRESP environment variable (if it exists) are searched for response information that matches the user's request. As in the directory search (above), the filenames are constructed automatically. The files are searched starting with the local directory, so if a match is found in both the local and SEEDRESP directories, the information from the local file will be used.

(4) Because it is possible to use wildcards to specify the network−code, stations and channels that are of interest, when the **−f** flag is used to pass the name of a directory to search or when the **−f** option is not given and the local and SEEDRESP directories are searched for matching files, all files whose names match the user's requested station, channel, and network code are searched for responses that have an effective time that includes the requested date (and time, if specified). This is necessary because there may be multiple, unique station−channel−network's that match a single input station−channel−network tuple from the user if wildcards are used. A list of all of the files that match is constructed and each is searched in turn. However, only the first matching response in each file is calculated.

If the **−stdio** option is given, the SEED response information is scanned from standard input and the resulting response is returned to standard output. In this case, the program will continue to search standard input for matching responses as long as it remains open (i.e. until an EOF is signaled). This allows the user to place evalresp into a pipeline of commands, or to use I/O redirection to read SEED responses from a file containing the response information.

NOTES ABOUT USAGE

(1) First, you must create an ASCII file containing the response information for the SEED volume. For *evalresp V3.0* (and later), *rdseed V4.16* or later must be used to create these files. To create the files, the R option to rdseed can be specified (either on the command line or interactively). This places the response information in the SEED volume into ASCII files with names like RESP.NET.STA.LOC.CHA. Alternatively, the **−d** option can be specified and, by responding "yes" to the query of whether you want response files written, these same files will be extracted only for the station−channel−network tuples for which data is extracted from the SEED volume.

(2) If the file argument is a directory, that directory will be searched for RESP files of the form RESP.NET.STA.LOC.CHA (or RESP.NET.STA.CHA).

(3) If the file argument is a file, that file is assumed to be a concatenated version of the output from a call to rdseed with the **−R** option. If this is the case, then only this file will be searched for matching response information

(4) If the file argument is missing, the current directory will be searched for RESP files of the form RESP.NET.STA.LOC.CHA or RESP.NET.STA.CHA (see *"How the Program Searches for Responses"*, above).

(5) If the environment variable SEEDRESP exists and is the name of a directory, that directory will also be searched for the requested files (if the **−f** option is not used, see *"How the Program Searches for Responses"*, above).

i.e. if typed setenv SEEDRESP /foo/resp_dir and no file or directory is specified to search on the command line, then the current directory and the directory /foo/resp_dir will be searched for matching RESP files from which to calculate responses.

(6) The units argument is one of the following: DIS (displacement), VEL (velocity), ACC (acceleration), DEF (default units), and represents the units for which the output response should be calculated (regardless of the units that are used to represent the response in the RESP file). If Default Units are chosen, the response is calculated in output units/input units, where these units are exactly the input units

of the first stage of the response and the output units of the last stage of the response. This is a useful alternative if the units for a particular type of sensor (e.g. a pressure sensor) are not in units that can be converted to displacement, velocity, or acceleration. The default value for this argument is VEL.

(7) The time−of−day argument is in HH:MM:SS format. This is used only in the case where there is more than one response in a given SEED volume for a given day. In that case, this argument can be used to choose one response over another according to the effective time of each. If this argument is not specified, then the first response that is found in the file that matches the requested year and day will be used. The default value for this argument is 00:00:00.0.

(8) The type−of−spacing argument is either logarithmic or linear ("log" or "lin" respectively). This governs whether the frequencies chosen are spaced evenly between the minimum frequency and the maximum frequency in a linear or logarithmic sense. This argument defaults to a value of "log".

(9) The **−v** argument indicates that the user would like to receive the verbose ouput from the *evalresp* program. When this flag is included on the command line, diagnostic information will be sent to standard output showing summary information of the calculated response for each station−channel−network tuple that matches the user's request. If this option is not specified, only error output will occur in the program.

(10) The **−r** argument indicates the response type the user desires. Available values are "cs" for complex−spectra output, "ap" for amplitude−phase output, and "fap" for frequency−amplitude−phase output. If the "cs" option is chosen, then the result is a set of files like SPECTRA.NET.STA..CHA (SPECTRA.NET.STA..CHA if location ID is present in the input file) that contain the frequency, real response and imaginary response (in that order). If the "ap" option is chosen, then a set of files like AMP.NET.STA..CHA (or AMP.NET.STA.LOC.CHA) and PHASE.NET.STA..CHA (PHASE.NET.STA.LOC.CHA) are created, containing the amplitude and phase response, respectively. If the "fap" option is selected, the program writes out frequency−amplitude−phase triplets. The resulting file names are in the form : "FAP.Net.Sta.Loc.Chan". The phase is always unwrapped in this output. Essentially this is just a re−packaging of the amplitude−phase output into a single, three−column file with unwrapped phase. This argument defaults to a value of "ap".

(11) The use of wildcards is allowed in the specification of stations, channels, and networks to search for. The first response of each station−channel−network that matches the wildcard pattern will be calculated and saved. For example, if the user requested response information from PFO 'BH?' with a network flag of **−n** '*', then the first response that matches the specified date for each of the broadband, high−gain channels will be returned for all of the networks that report a response for PFO. The wildcarding scheme used here is a "glob" style rather than "regular expression" style of pattern matching. The total length of the patterns used for the stations, channels, or networks is restricted to 64 characters by the program, although multiple examples can be combined in a comma separated list for the station and channel lists.

(12) The **−stage** argument can be used to specify a stage number or a range of stage numbers, if both a starting and stopping stage number are included, for which to evaluate responses. For example, if this argument is included on the command line as **−stage** 3, then only the response of stage 3 will be calculated (ignoring all other stages). If the user wishes to calculate a response for stages 1 through 3, then the appropriate usage would be **−stage** 1 3. Setting the starting stage to a number less than zero will cause the default behavior to occur; evaluation of responses for all stages in a RESP file. If the number specified for a "single stage" response is higher than the number of stages in the response, no output will occur and an error message will be printed indicating why no output occurred. If a range of responses is specified that is outside of the range that is given in the RESP file, then no output will occur. Otherwise, the stages with numbers within the interval from the starting to the stopping stage will be used to calculate the response.

(13) The **−unwrap** argument is used to unwrap the output phase if used in combination with **−r ap** option (see note 10 above).

(14) Note that there is also a configuration option −−enable−phase−unwrap (which can be enabled at as ./configure −enable−phase−unwrap before making stage of evalresp. This option not only unwraps the

phase but also shifts it to keep the values of phase in the range −180:180 degree.

(15) The **−ts** argument is forcing useage of the stage 0 total sensitivity instead of product of the stage gains. The idea is that this can be utilized in combination with the −stage option to provide a full scale response, i.e. just stage 1 (the sensor) with a correct system gain (which is exactly what SAC Poles and Zeros do)

(16) The **−stdio** argument can be used to specify that input should be taken from standard input and output should be sent to standard output. In the case where both **−stdio** and **−v** are specified, the response can be separated from the "verbose" output by splitting the standard output (which will contain the response) from the standard error (which will contain the verbose output). When this flag is defined, standard input is parsed for input responses until an EOF is found, indicating the end of the input stream of response information.

LIST BLOCKETTE INTERPOLATION

The following command−line parameters are used to enable List−blockette interploation:

−il : Specifies that the amplitude/phase values generated from responses containing List blockettes (55) are to be interpolated to correspond to the set of frequencies requested by the user. A cubic−spline interpolation algorithm is used, with a "tension" value specified via the **−it** parameter (see below). If any of the user−requested frequency values fall outside of the range of frequencies defined in the List blockette then the out−of−range frequencies will be "clipped" (ignored), the output will be generated for the in−range frequencies, and a warning message will be sent to the console. If a response does not contain a List blockette or if the complex−spectra response output type is selected ("−r cs") then this parameter will have no effect. If this parameter and the **−ii** parameter are not specified then the output for a response containing a List blockette will be generated only for the frequencies defined in the List blockette.

−ii : Specifies that the amplitude/phase values input from a response containing a List blockette (55) are to be interpolated to correspond to the set of frequencies requested by the user. The interpolated values are then processed by the program. A cubic−spline interpolation algorithm is used, with a "tension" value specified via the **−it** parameter (see below). If any of the user−requested frequency values fall outside of the range of frequencies defined in the List blockette then the out−of−range frequencies will be "clipped" (ignored), the values will be generated for the in−range frequencies, and a warning message will be sent to the console. If a response does not contain a List blockette then this parameter will have no effect. This parameter (rather than **−il**) can be useful when the complex−spectra response output type is selected ("−r cs"). If this parameter and the **−il** parameter are not specified then the output for a response containing a List blockette will be generated only for the frequencies defined in the List blockette.

−it : The "tension" value used by the cubic−spline interpolation algorithm (see the **−il** and **−ii** parameters). A relatively high "tension" value is desirable because it makes the interpolated values "track" closely to the original values. This parameter may be specified as a floating−point value, and its default value is 1000.0.

Note: The **−il** ("interpolate List−blockette output") parameter differs from the **−ii** ("interpolate List−blockette input") parameter in that when **−il** ("output") is specified the interpolation happens after the response data values have been processed by the program. When **−ii** ("input") is specified the List−blockette data values are interpolated before they are processed by the program. The two types of interpolation should generate results that are basically identical.

EXAMPLE

evalresp HRV,ANMO 'BHN,BHE,LH?' 1992 231 0.001 10 100 −f /home/RESP/NEW −t 12:31:04 −v

The quotes in this command are required to prevent the shell from expanding the '?' character before passing it into *evalresp*. If the RESP files for HRV and ANMO are contained in the directory '/home/RESP/NEW', then this example will output eight files, called AMP.I U.HRV..B HE, PHASE.I U.HRV..B HE, AMP.I U.HRV..B HN, PHASE.I U.HRV..B HN and AMP.I U.ANMO..B HE, PHASE.I U.ANMO..B HE, AMP.I U.ANMO..B HN, PHASE.I U.ANMO..B HN for the HRV and ANMO BHE and BHN channels.

A corresponding set of files would be output for the ANMO broadband channels and for all the HRV and ANMO long−period high−gain channels in the directory '/home/RESP/NEW'. These files contain the amplitude and phase information, respectively.

These can be used as input for any graphing programs capable of reading simple columns of data.

SEE ALSO

rdseed(dmc), *xml2resp*.