NAME

isoprep, isoparms − prepare captured data for isopotential mapping

SYNOPSIS

isoprep [**−n**] [**−s** latency] prmfile runfile ...

isoparms

DESCRIPTION

Isoprep generates a matrix of voltage measurements, for use by *isopot*(1), from runs of data captured by one of the capture programs (*dsepr*(1)). Each run it uses must correspond to a single *track*. The *depth* information is taken from one of the captured channels.

The first argument, *prmfile*, is the name of an *analysis*(1) parameter file, in which all the necessary parameters have been set. The **.prm** suffix is added to the name, if not already present. If the **-s** option is given, the specified *latency* will override the parameter file's trace level sample latency. The remaining arguments, the *runfile*s, are the names of frame files produced by the capture program, which can be specified with or without the **.frm** suffix. The generated matrix is sent to the standard output, which you should redirect to a file.

Isoprep calls up the *analysis* program, to calculate the trace amplitudes at each depth, in each track, using the file of parameters you provide. (It executes the analysis program in a way which doesn't make use of the graphics terminal, so it can be run from any terminal.) It then goes through the results generated by *analysis*, checking for consistency across runs. Finally, if all goes well, it produces the final measurement matrix, of the format required by *isopot*(1).

If the depth information is not consistent across all runs given, you are asked whether or not the intermediate results are to be kept anyway. If you answer **Y**, these results are sent to the standard output, in place of the measurement matrix. Note that the intermediate results are of a different format from the measurement matrix, and must be processed by *fixdepths*(1) before you can use them.

If the **-n** option is given, *isoprep* will use a different analysis method, which doesn't depend on the presence of a depth signal. Instead, it depends on the timing of the capture of triggered sweeps. The range of data in each run is divided into the number of bins specified in the parameter file, with each bin representing a different depth. As long as the timing is consistent across all tracks, with a consistent pause at each depth and consistent spacing between depths, this technique will allow the generation of a measurement matrix without the availability of a depth signal.

Isoparms greatly simplifies the process of initially setting up the *analysis*(1) parameter file required by *isoprep*. It will prompt you for most of the relevant parameters, and calculate the rest of them. The parameters are stored in a parameter file, with the **.prm** suffix appended to the run file name you specify. One parameter which you are not asked for is the *latency*; you will either have to specify this when running *isoprep*, or change the "Trace amplitude point" in *analysis*. It assumes the presence of a depth signal, and will ask you for the W.F. number for it, so it may not be that helpful when you plan to use the **−n** option to *isoprep*.

Run File Requirements

Each *track* of measurements (i.e. each row in the final matrix) must be captured as a separate run. The signal for which you want to map the isopotentials must be captured as a *triggered* channel. The *depth* information (i.e. the levels at which the measurements are taken) must be captured as an *untriggered* channel. It is important that the depths be consistent across all tracks.

By default, it is assumed that the spacing between tracks is kept constant. If this is not the case, then the "run description" associated with each run must contain a string of the form

x=*n*

where *n* is the distance of this track from the origin. The units in which *n* is given are arbitrary, but the same units must be used for all tracks. If the track spacing is constant, these strings can be omitted, but then they must be omitted from *all* tracks, and the *runfile* arguments must be given in the correct order, from closest to farthest from the origin.

Parameter File Requirements

You must use *analysis*(1) to set up the parameter file for *isoprep*. In this file, you must set all of the parameters which are required for the "Averaged trace amplitude vs W.F. level" analysis. You can safely ignore the display options, however, since these are not really needed.

You can use *isoparms* to initially set all of these parameters. You should then use *analysis* to verify them; especially the min. and max. waveform levels, and the trace amplitude point and reference.

Note that this single parameter file is used for several runs of data. This should not be a problem, since the data are supposed to be consistent across all runs anyway. You simply have to set the parameters for one of the runs, normally the first one. *Isoprep* automatically applies these parameters to the other runs.

There is one instance where you may want to set parameters independently for each run. If some "garbage" was captured at the start or end of some of the tracks, you may want to set the "Start of run" and "End of run" parameters for these runs. If you do this, then you must set these two parameters for every track in the map. I.e. Every run file will have to have an associated parameter file of the same name. You will still have to give *isoprep* the name of one parameter file, and almost all parameters will still be taken from this one file. Only the "Start of run" and "End of run" parameters will be taken from the separate parameter files.

EXAMPLES

Assuming we have captured five tracks of measurements in the files **map01.frm**, **map02.frm**, **map03.frm**, **map04.frm** and **map05.frm**, with the depth signal captured in the corresponding **map***nn***.w00** files, then we could proceed as follows. First, we execute

```
analysis map01
```

to set up the parameters. We select *Analysis/Graphs/Trace-amplitude/Trace-vs-W.F.-amp/Averaged* as our analysis method, then set the parameter "Fixed W.F. level bins" to **Y**, and set "# bins− graph" to the number of depths at which measurements were taken. If the depth signal was recorded on something other than waveform number **0**, we should also set "Amplitude W.F. #" to the appropriate number.

If we now select *Set/Levels/Waveform/Range/Visually* we can use the cursor to select the minimum and maximum W.F. amplitudes, and we will see how this range is divided into bins. We set these bounds to be slightly below and above the lowest and highest depth readings displayed, in such a way that the dotted lines (bin separators) fall halfway between the steps in the displayed signal.

Of course, we must also set the parameters "Trace amplitude point" and "Trace amplitude ref" to indicate which part of the triggered signal (i.e. at which latency) we want to measure. (The "Trace amplitude point" parameter doesn't have to be set if it will be given to *isoprep* using the **−s** option.) We can then select *Go* to perform a trial run of this analysis. This will give us an indication of whether some parameters were improperly set. After that, we can save our parameters and quit the analysis program.

We can finally run *isoprep*, giving it the parameters we just set, and all of the run files. We will save the measurement matrix in **map.t**.

```
isoprep map01.prm map0[1−5].frm > map.t
```

After all that, we finally get to see the results. We execute *isopot* and read in the matrix by selecting *Measurement/Read-matrix* then typing in the file name. We must also set, at the very least, the coordinates for the origin, depth and width of the contour map area. We can then select *Go*, and the map will begin to be drawn.

FILES

*.prm	analysis parameter files
*.frm	frame files (for voltage readings)
*.w[0-9][0-9]	waveform files (for depth readings)
/tmp/iso*	temporary files

SEE ALSO

isopot(1), analysis(1), fixdepths(1), cap(1), dsepr(1)

DIAGNOSTICS

If *isoprep* runs without error, the only message given is "Analysing...", which appears before the analysis program is called.

If the specified parameter file does not exist, or you don't have read-permission on it, you get the message "Can't open *name*."

The message "analysis failed!" appears if a fatal error occurred while the analysis program was running. It should be preceded by a message from the analysis program itself.

If the message "Error in output file − line count wrong" appears, it means the analysis of one or more of the tracks has failed. Check to make sure all of the required parameters are properly set in the specified parameter file. If some *runfile*s have associated parameter files, make sure they all do, and check that the "Start of run" and "End of run" are correct for all of them.

If the message "Error in output file − depths don't match" appears, it means the depths at which readings were captured are not consistent across all tracks. Check to make sure all of the required parameters are properly set, as for the "line count wrong" error above. Also, check to see whether any track is missing measurements at any depth. When this error occurs, you still have an opportunity to use the results generated. You are asked whether you want to keep the results, and if you answer **Y**, then these intermediate results are sent to the standard output, instead of the final measurement matrix. These intermediate values are not usable directly by *isopot*, but can be converted to the proper form with the *fixdepths*(1) program. All you have to do is tell it, in another file, which depths you want measurements from, and it will interpolate values at these points. This solution should be used only when all else fails. If you can get around this error by changing your parameters, that is probably the better approach.

BUGS

Specifying a mix of positive and negative track positions will not cause any error or warning. It will simply cause the tracks to be ordered from highest (most positive) to lowest (most negative) position. The origin is assumed to be the highest position.

If some, but not all of the runs in a map have their own parameter files (with the same file name, but with the **.prm** suffix), then *isoprep* will get confused about which "Start of run" and "End of run" parameters apply to runs with no parameters. It will use those from the last parameter file read, rather than those from the parameter file specified as the first argument.

The handling of errors in the analysis is quite poor, and does not give you much indication of what the cause of the error actually is.