If you use this flag and LoadWave can not determine the source text encoding for a file, it will return an error. If you want your procedure to continue with other files you must check for and handle the error using GetRTError.

Loading Very Large Files

The number of waves (columns) or points (rows) that LoadWave can handle when loading a text file is limited only by available memory.

You can improve the speed and efficiency of loading very large files (i.e., more than 50,000 lines of data) using the *numLines* parameter of the /L flag. Normally this parameter is used to load a section of the file instead of the whole file. However, in delimited, general text and fixed field text loads, the *numLines* parameter also specifies how many rows the waves should initially have. Thus all of the required memory is allocated at the start of the load, rather than increasing the number of wave rows over and over as more lines of data are loaded. When loading very large files, if you know the exact number of lines of data in the file, use the *numLines* parameter of the /L flag. If you don't know the exact number of lines, you can provide a number that is guaranteed to be larger.

If you omit the /L flag or if the *numLines* parameter is zero, and if you are loading a file greater than 500,000 bytes, LoadWave automatically counts the lines of data in the file so that the entire wave can be allocated before data loading starts. This acts as if you used /L and set *numLines* to the exact correct value which normally speeds the loading process considerably. You can disable this feature by using the /V flag and setting bit 2 of the *loadFlags* parameter to 1.

Escape Sequences

An escape sequence is a two-character sequence used to represent special characters in plain text. Escape sequences are introduced by a backslash character.

By default, in a text column, LoadWave interprets the following escape sequences: \t (tab), \n (linefeed), \r (carriage-return), \\ (backslash), \" (double-quote) and \' (single-quote). This works well with Igor's Save operation which uses escape sequences to encode the first four of these characters.

If you are loading a file that does not use escape sequences but which does contain backslashes, you can disable interpretation of these escape sequences by setting bit 3 of the loadFlags parameter of the /V flag. This is mainly of use for loading a text file that contains unescaped Windows file system paths.

Loading Igor Text Files

An Igor Text file consists of keywords, data and Igor commands. The data can be numeric, text or both and can be of dimension 1 to 4. Many Igor users have found this to be an easy and powerful format for exporting data from their own custom programs into Igor.

The file name extension for an Igor Text file is ".itx". Old versions of Igor used ".awav" and this is still accepted.

Examples of Igor Text

Here are some examples of text that you might find in an Igor Text file.

Simple Igor Text

```
IGOR
WAVES/D unit1, unit2
BEGIN
      19.7  23.9
      19.8  23.7
      20.1  22.9
END
X SetScale x 0,1, "V", unit1; SetScale d 0,0, "A", unit1
X SetScale x 0,1, "V", unit2; SetScale d 0,0, "A", unit2
```