Editing with Credit:
A Pictorial Tutorial

Duncan Carr Agnew

1. Introduction

This document is meant to be a short introduction to the program *credit*, the part of the PIASD package that is used for editing data. How *credit* fits into the overall editing scheme is described in the PIASD document, and all the program commands are listed and described on the program man page. Here, the aim is to walk the novice user through some of the commands, in particular the ones used for editing. This is best done through direct teaching; I have tried to write this document to approximate this, by going through the steps in some detail. Each page (after this one) has a picture that is about what should be on the screen (though for technical reasons not a screen dump), with commentary and instructions. When these are carried out they will produce the picture on the next page—at least approximately. The approximation is that the actions often involve positioning the cursor, and if this is done slightly differently, slightly different outcomes will result.

2. Getting Started

The first step is to create a data file for editing. You should first `cd` into the subdirectory `working/credemo/` in the `PIASD` area. In this there is an ASCII file of data (kept in this form for platform-independence) called `straindata`. (This is laser strainmeter data from Durmid Hill, 10000 terms with 300-second sampling, from 2003:277 through 2003:311). Run the command

```
%cat straindata | datput 100
```

to put these in the *niolib* file 100 (actually called `I100.D`). (In this example and others **`this font`** is what is typed by the user, `this font` is what is printed by the program.

Before going further, you will need to set the Xterm display to Tektronix mode. On some machines this is done by typing **`xterm -t`**, which opens a new window. Under Mac OS-X it is done by using the mouse button to open a menu of options, and choosing ‘‘Open Tek Window’’.

Now run the program *credit* and answer the preliminary questions as follows:

```
%credit
File number                                  100
Start term                                   1
Plot limits                                  0 0
Mode of operation (1 for editing, 0 for display)                    0
Decimation factor (typically 1 for editing)                         1
Type of decimation (0 for straight, 1 for min/max)                            0
Number for previous edits file:                        0
Predictor weights (type 1 if wanted)                   0
File number for writing out edits:                           101
```

and the screen will blank and then appear as:

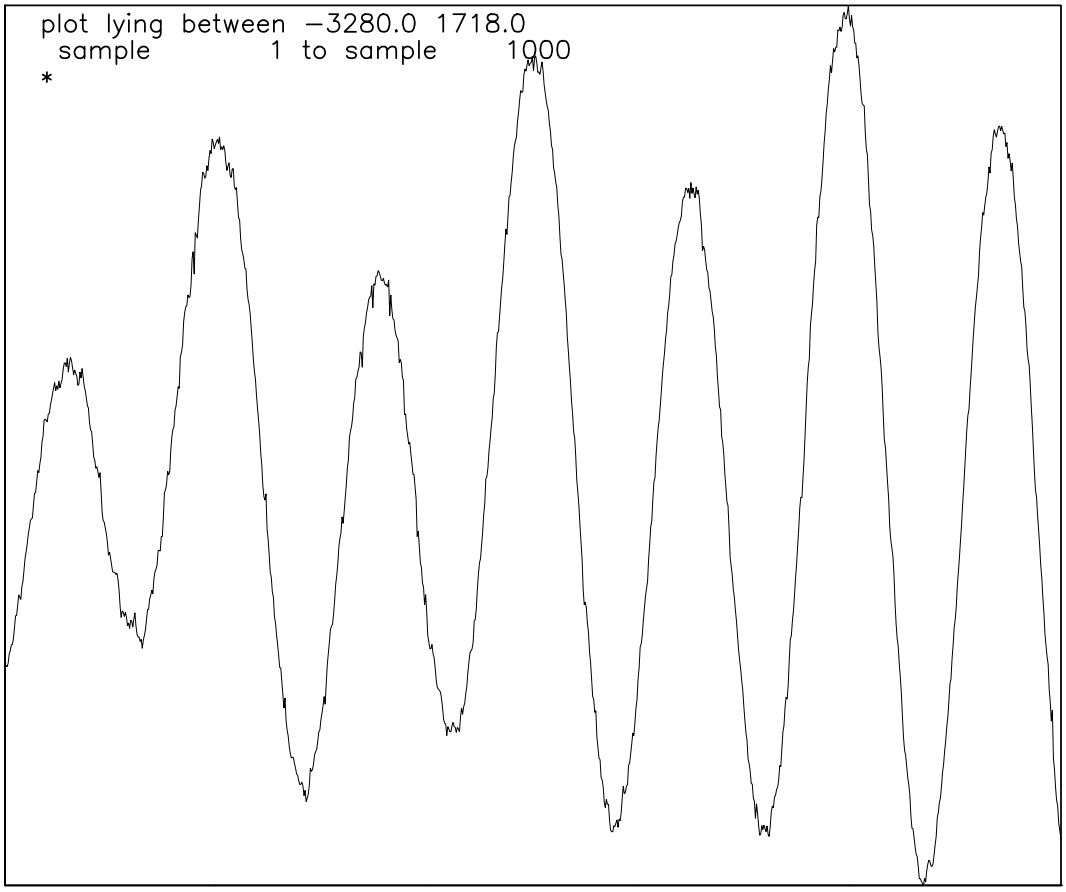

Figure 1

which shows the first 1000 points, self-scaled. To move to the next 1000 points, type **ENTER** and you will get

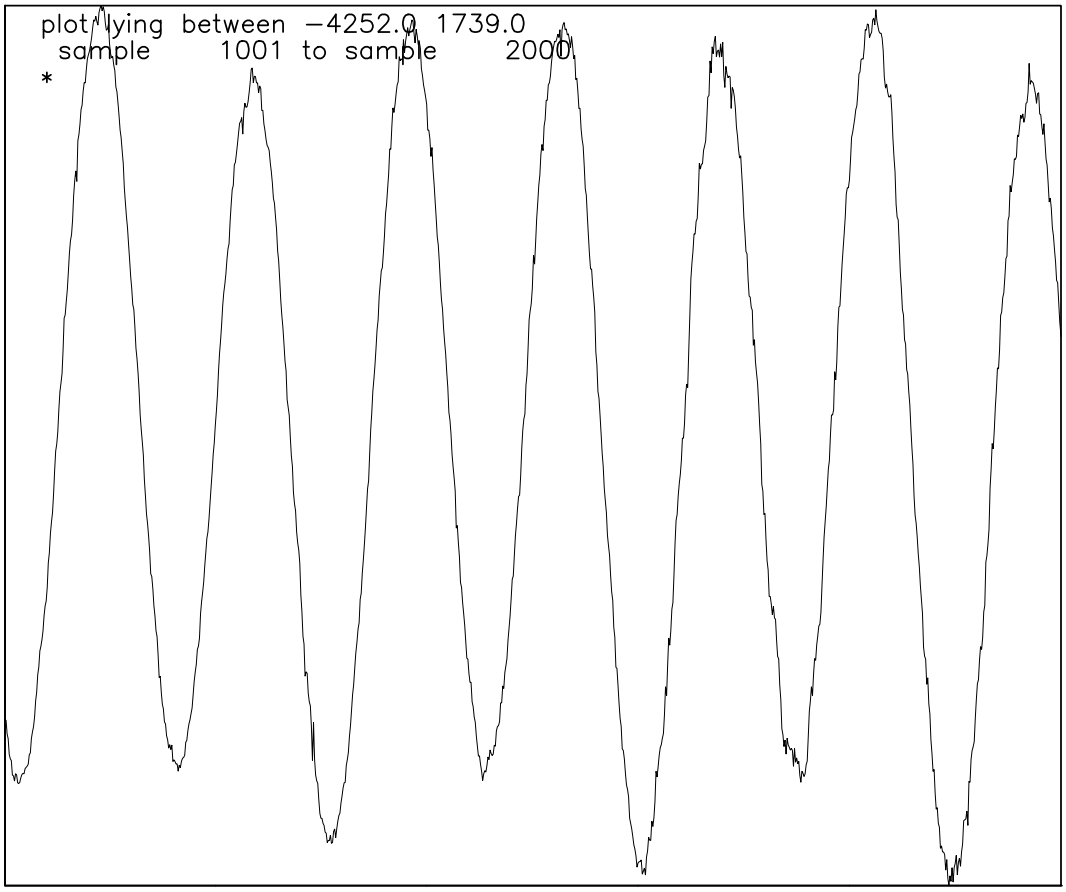

Figure 2

Type **ENTER** again to get terms 2001-3000, and you will get

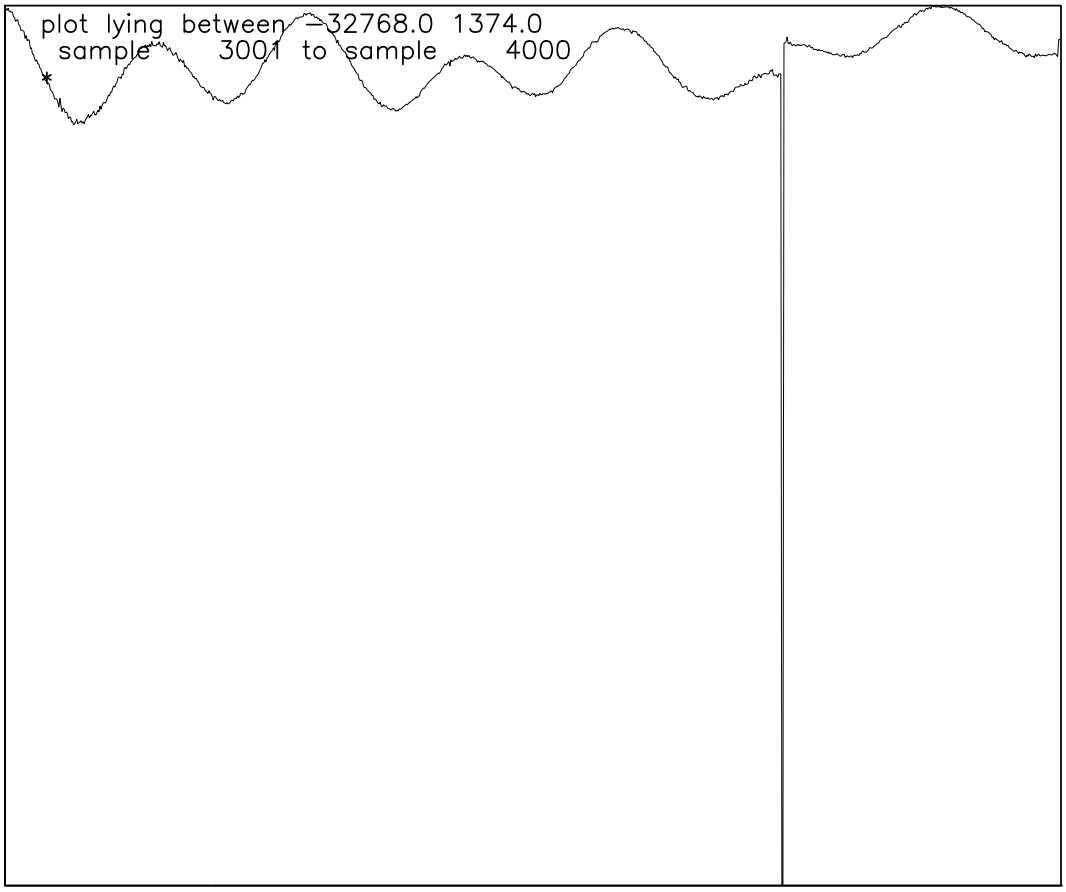

Figure 3

which shows a large spike and a small offset. Now type **c ENTER** to turn the cursor on; this will appear as a small cross, which can be moved with the mouse. When the cursor is turned on, all commands are entered by just typing a key; you do **not** need to type **ENTER**.

Move the cursor horizontally until it is vertically centered over the spike (its vertical location does not matter at this point) and type **=** (which is the **+** key with no shift), and program will zoom, redrawing the screen to look like

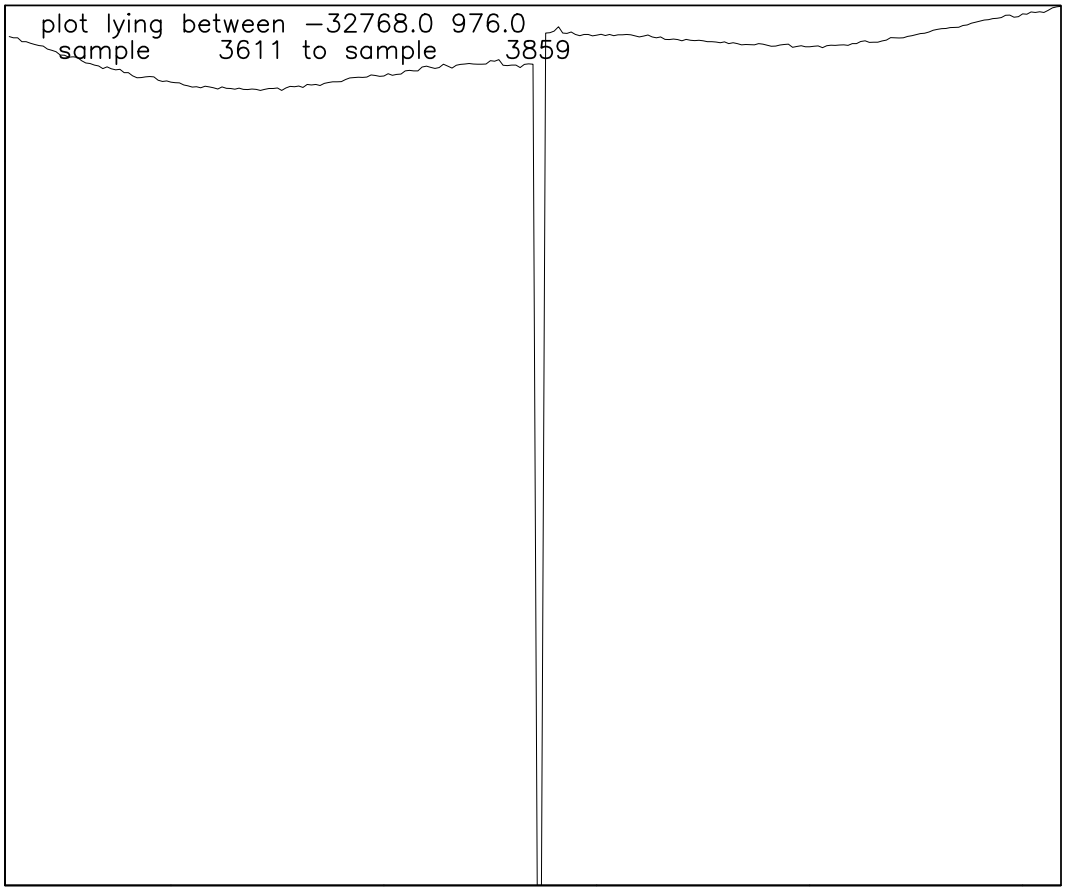

Figure 4

Center the cursor over the (enlarged) spike, and again type =, to get

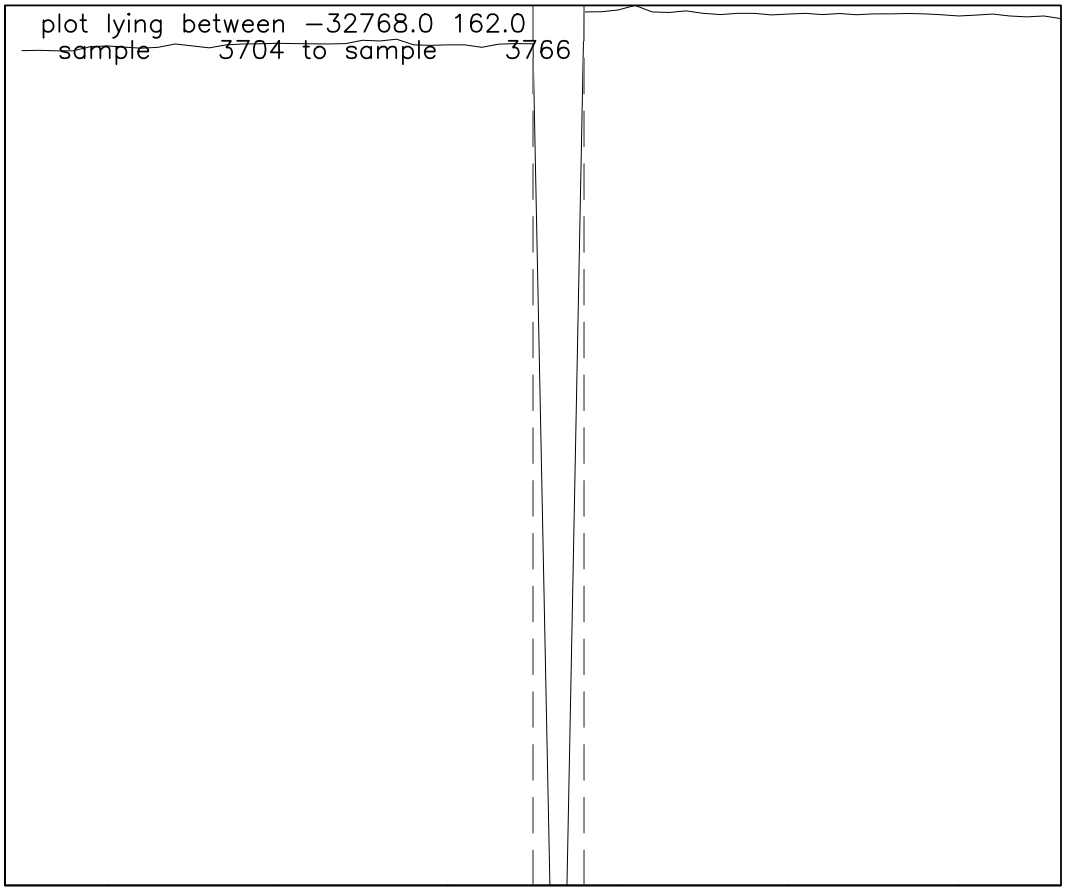

Figure 5

which is now enlarged to the point that you can distinguish individual points. In *credit* terms, this span of data needs to be edited as an offset, not a glitch (a span of bad data with no offset), but we will start by editing it as a glitch, to show how this is done. To start, type **g**; the plot will not change but you will be told

```
Set cursor on last good point and type 1,
   then on first good point and type 2
```

where the ‘‘last good point’’ and ‘‘first good point’’ refer to the points immediately before and after the bad data. These are indicated in Figure 5 with dashed grey lines. So, the next step is to:

- Move the cursor until it would be on the left-hand line and type **1**;
- Move the cursor until it would be on the right-hand line and type **2**.

The screen will be redrawn as:

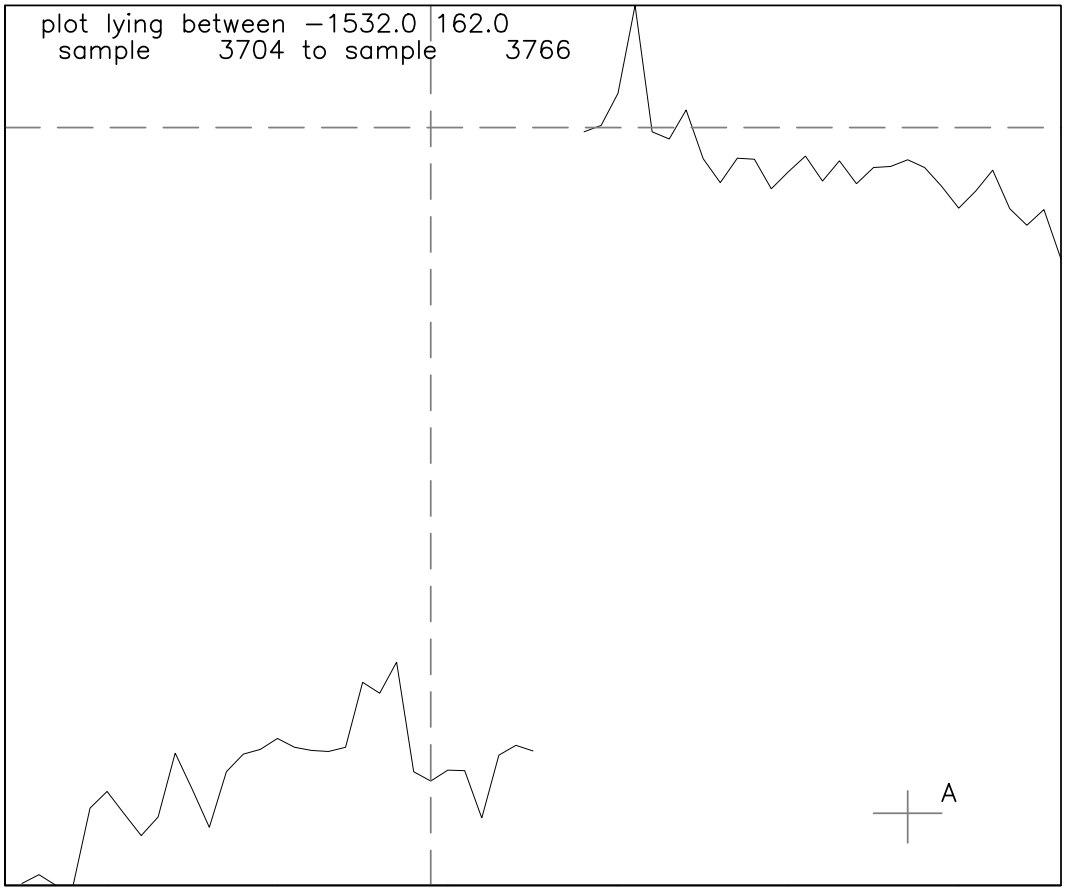

Figure 6

which shows the effect of removing the spike and rescaling the plot; note that the same terms are shown as before, and that the ‘‘bad data’’ inside the gap is not shown at all.

The next step is to deal with the offset. Since this edit has been done initially as a glitch-edit, to invoke the machinery for offsetting you need to:

- Put the cursor inside the gap, to indicate that this is the edit you will want to change;
- Type **`i`** (for ‘‘increment an offset’’—a glitch-edit being an offset-edit with the offset set to zero).

The program will respond with

```
Altering offset for edit 1; options 3,4,5,6 (! for help)
```

There are a variety of options; we will begin with option 3. To use this option, move the cursor horizontally to select a point (this will be the point vertically above or below it). Also move the cursor vertically to select the level you would like this point to move to. The dashed grey lines show one possibility, which is term 3728, y-value −73 (you can use the **`p`** and **`y`** commands to see where the cursor is). Though this selection is before the gap, the program will apply the offset to the data after it; the cursor could just as well be moved to the point marked A (this is at term 3756, y-value −1393). Once the cursor is where you want it, typing **`3`** will introduce the offset and replot the screen to produce

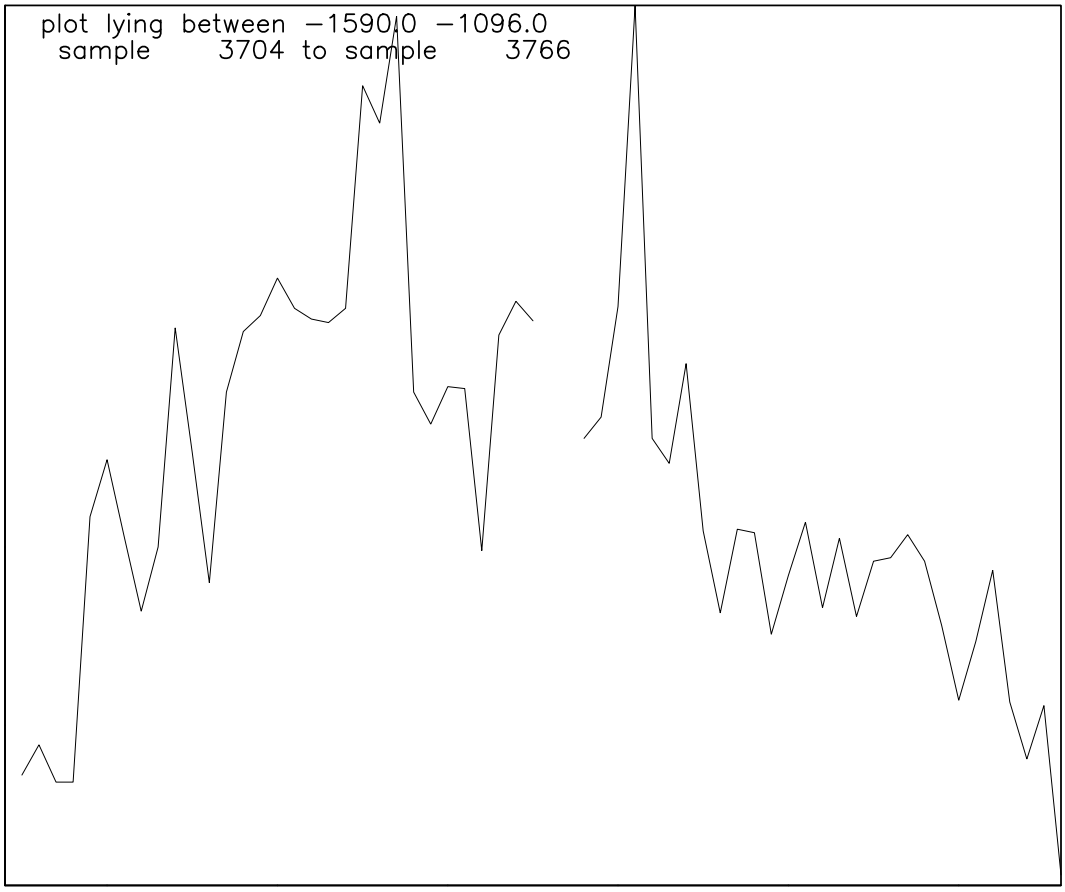

Figure 7

in which the offset has clearly been removed. You can now zoom out with the **–** command, or else type **h** to return to the page-at-a-time mode. If you type **h**, the cursor will vanish and a * will be printed; type **ENTER** and the page displayed in Figure 3 will be replotted with the edit included, as

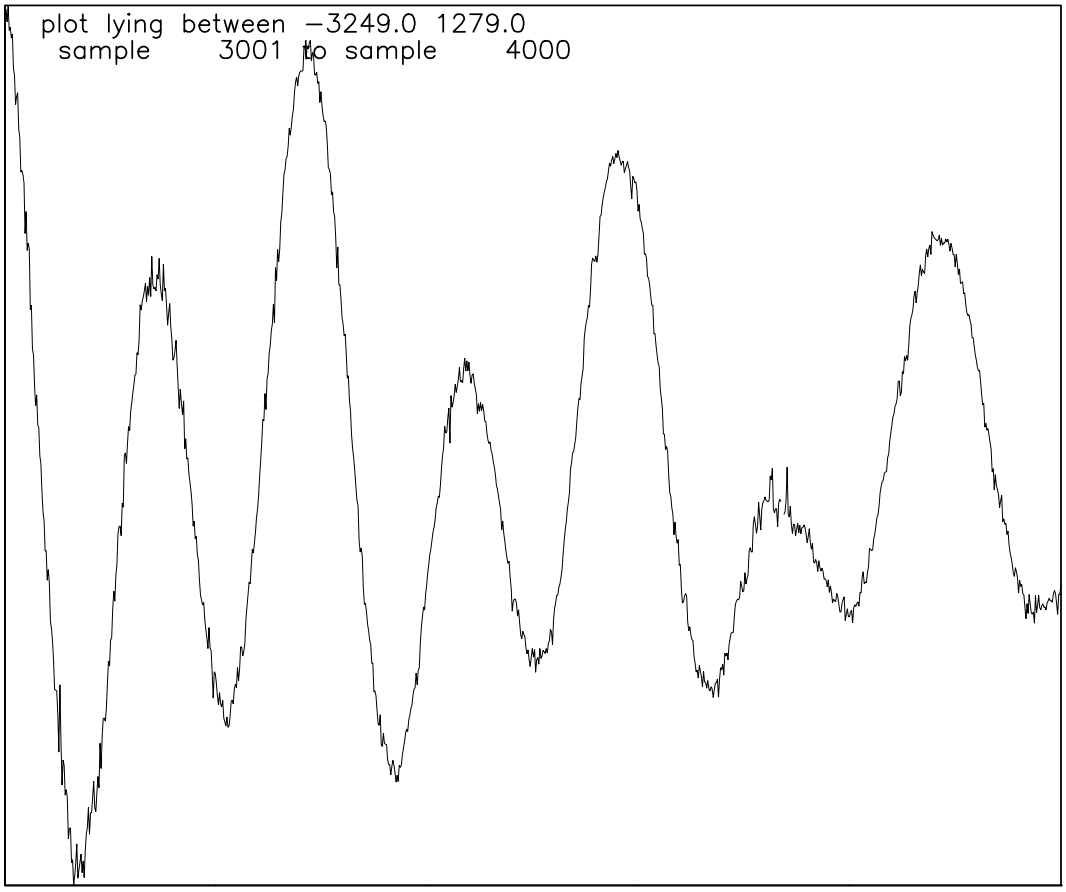

Figure 8

Paging to the next chunk of data gets

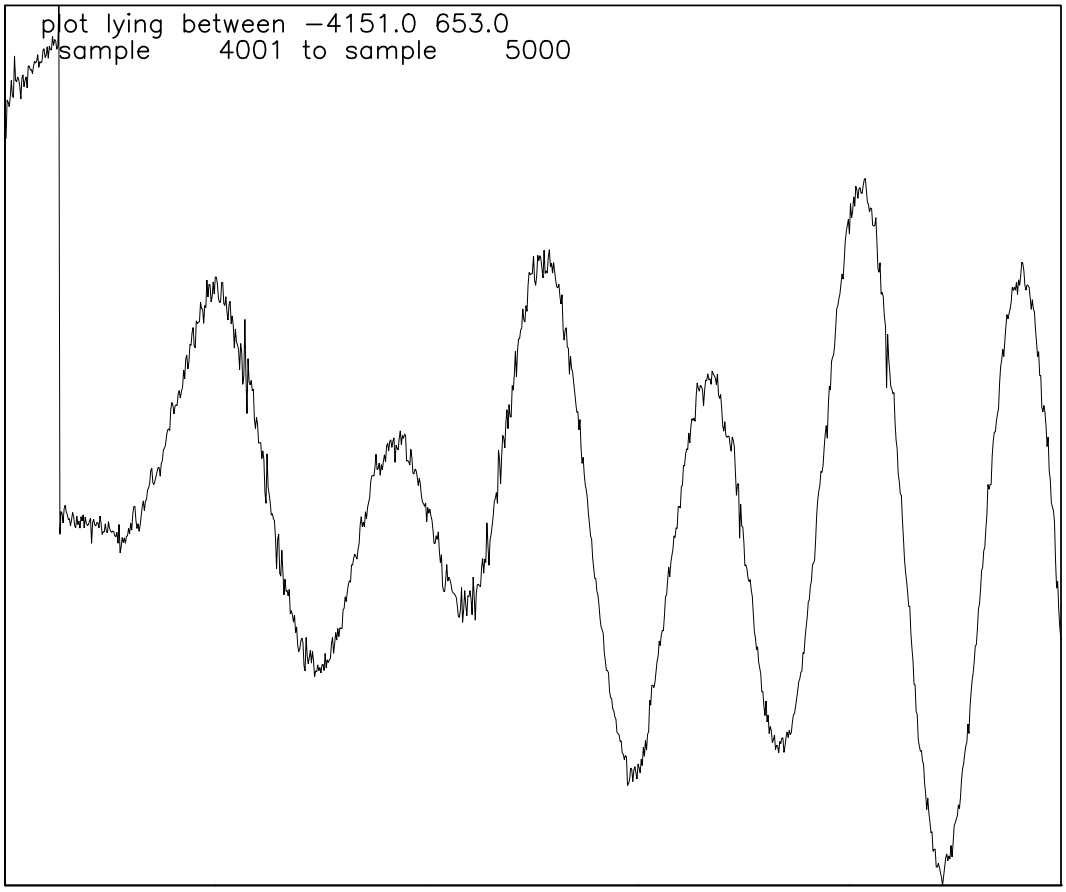

Figure 9

which evidently has another offset. So, again type **c ENTER** to get back into cursor mode, place the cursor over the offset, and type **=** to zoom in. This gets us

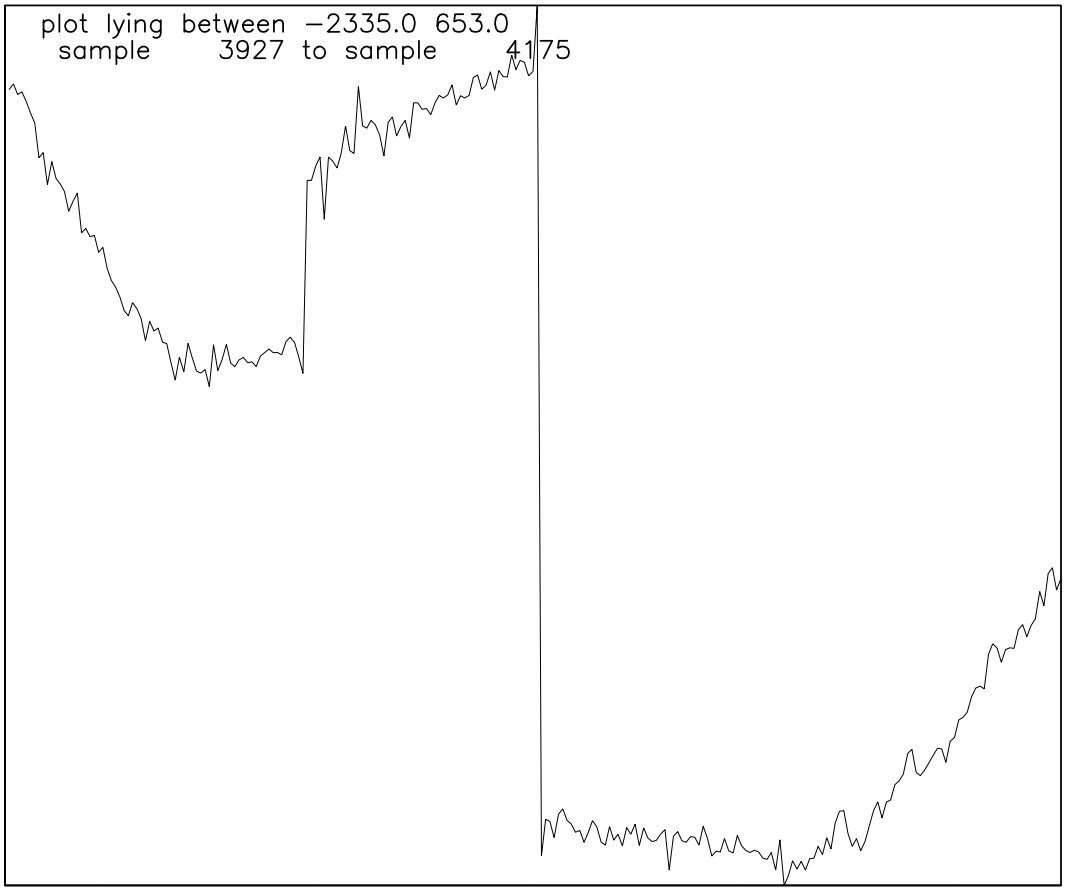

Figure 10

which shows two offsets. The reason that the earlier one didn't show up before is that it was actually on the previous "page", but was missed because it was so close to the page boundary. Place the cursor over this first offset, type **=** to zoom again, and get

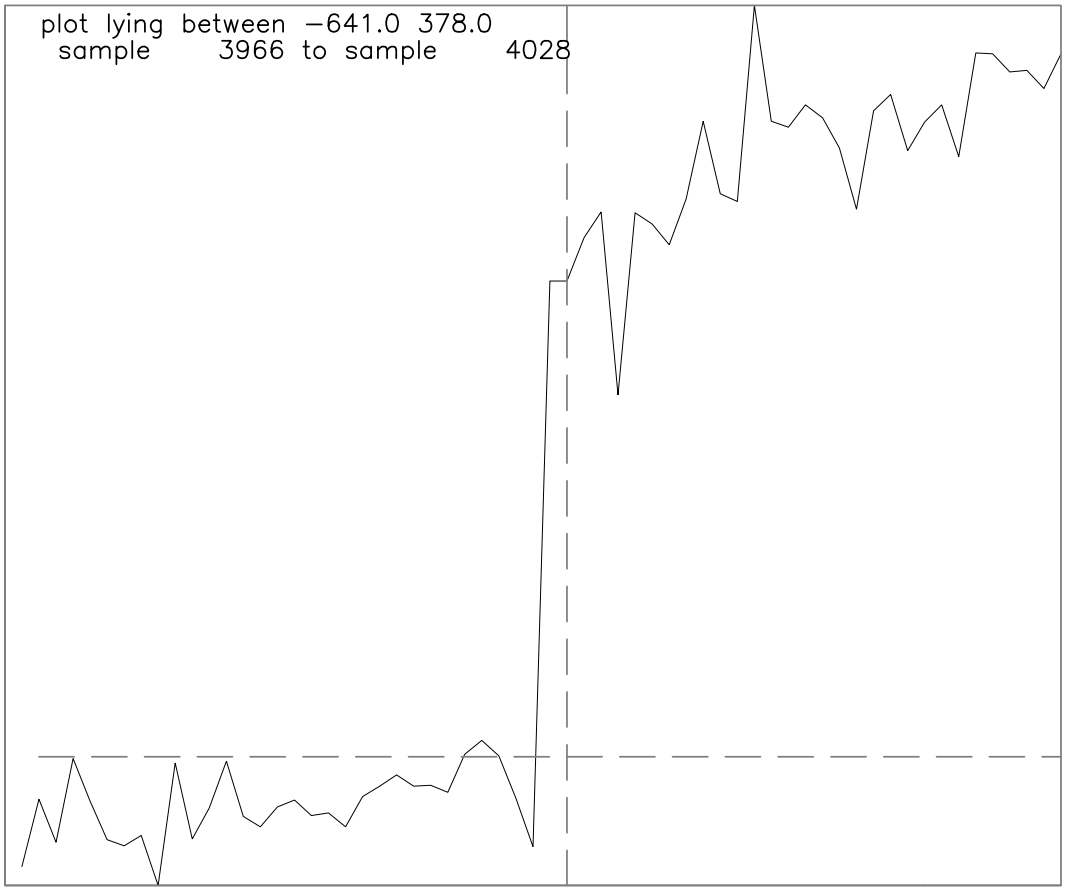

Figure 11

Now, edit this offset by typing **o**. You will again need to move the cursor to the last good point, type **1**, move the cursor to the first good point, type **2**, and move the cursor to indicate an offset (as suggested by the gray lines) and type **3**. (The lines correspond to term 3998, y-value -493) The screen will be redrawn to produce

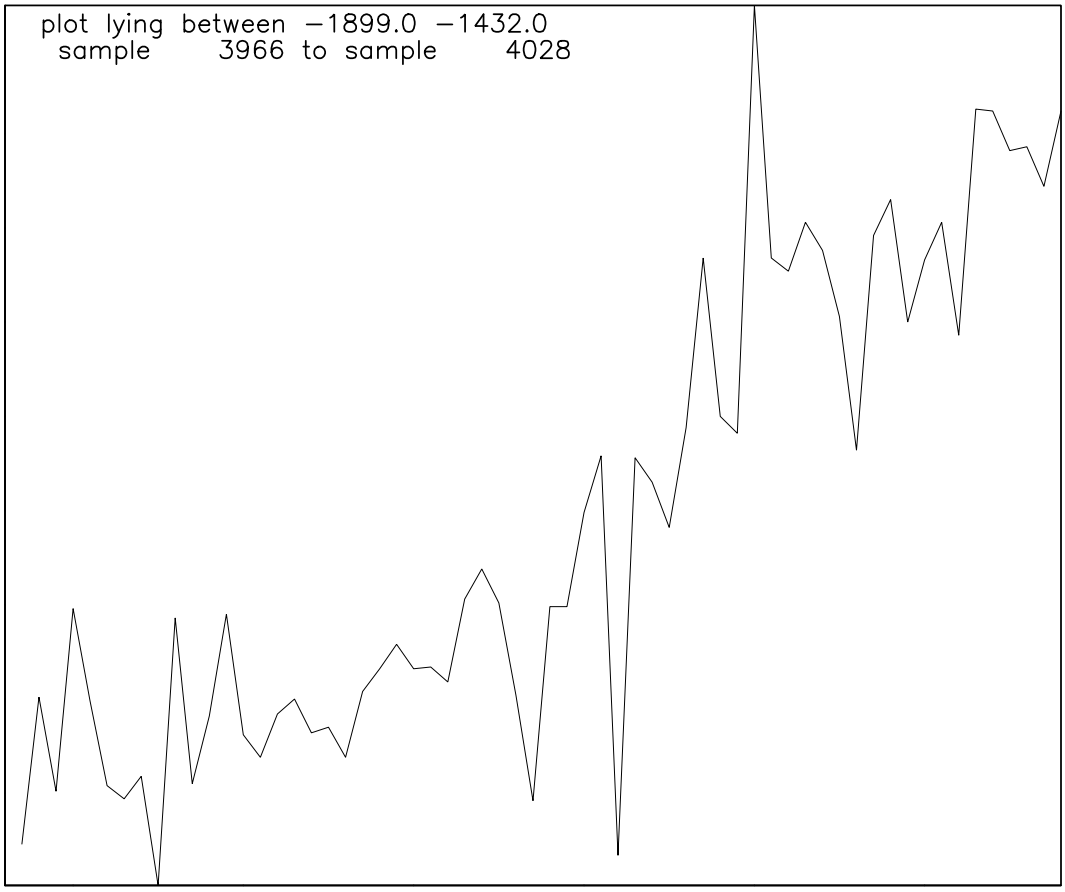

Figure 12

in which the offset is removed. (Nothing is blanked out because there were no bad points: this was a step betweeh terms 3996 and 3997). Move the cursor over the (former) bad data and type **−** to zoom out to get

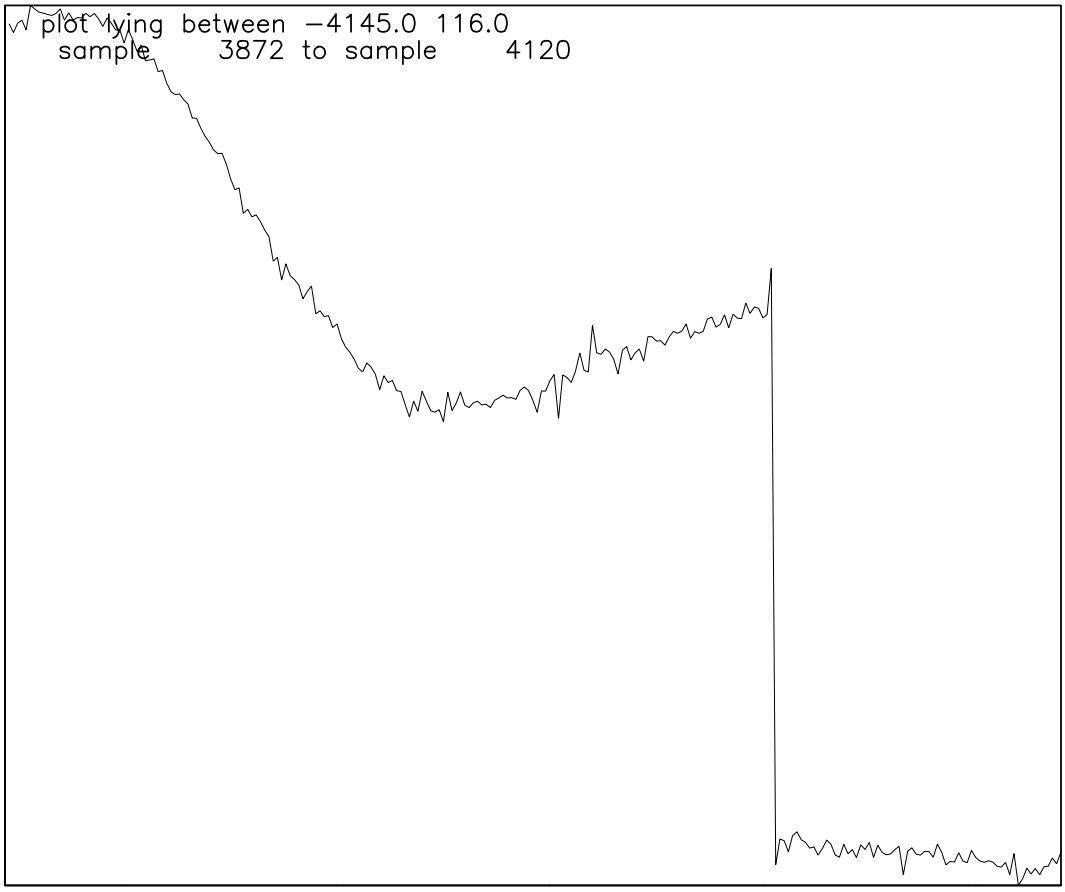

Figure 13

Now place the cursor over the remaining offset, and type **=** to zoom in again, giving

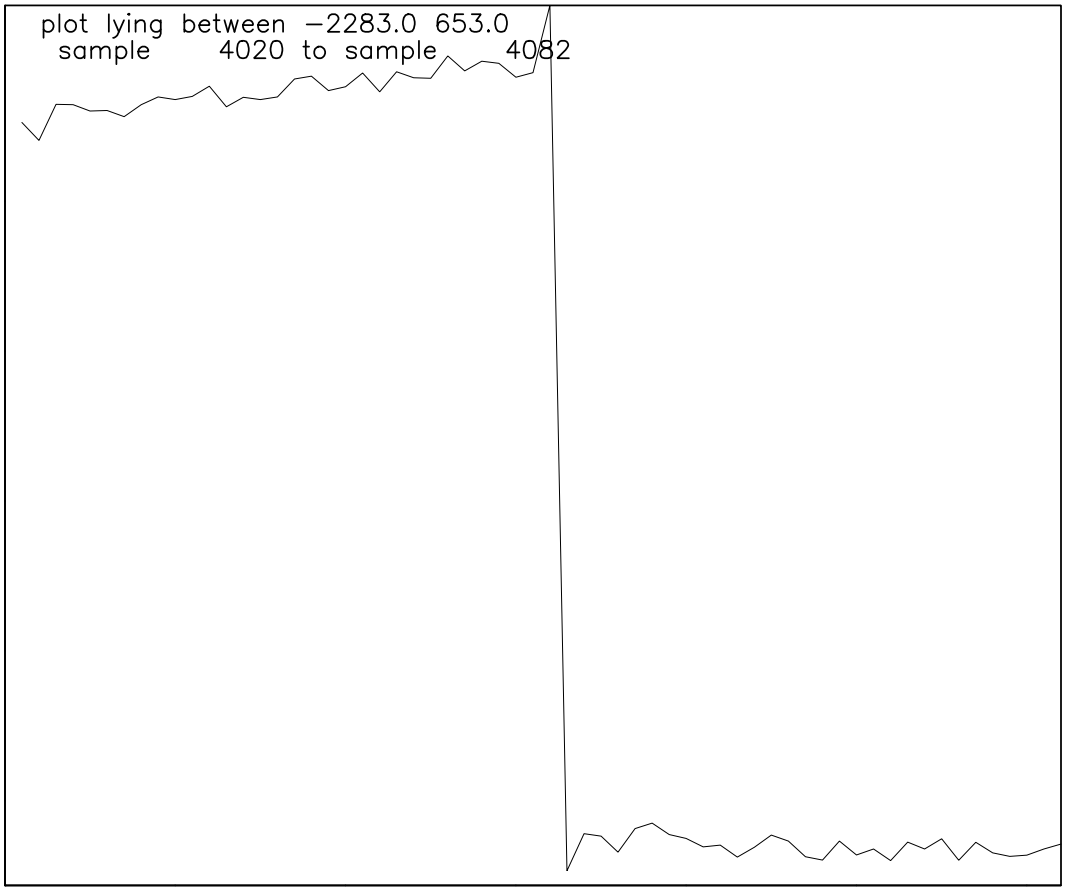

Figure 14

Again, type **o** to edit the offset, and go through the steps of indicating the edges of bad data with **1** and **2**. Then type **−** to zoom out, giving

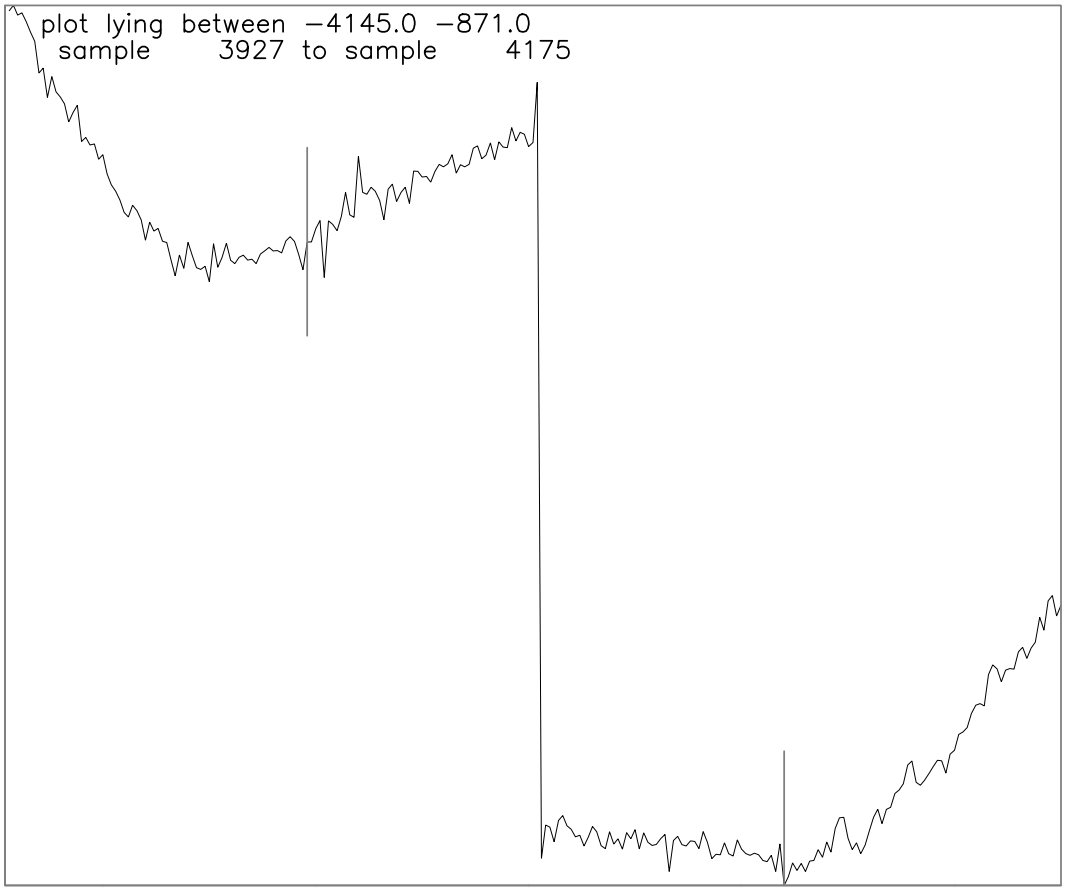

Figure 15

For the next step, finding the offset, type **6**. The program will type:

```
Set cursor to left of gap, then right of gap, to
 define areas to fit lines to; type a grave accent for each one
```

which describes what to do, though not why. What is done in this case is to fit a straight line to the data before and after the edit, and choose an offset that will make the lines themselves match up. This is usually the best choice, as it almost always gives excellent results and is easier than option **3**; you do not have to choose offset amounts, but only which terms the lines will be fit to. Obviously, you want to make this some range over which the data approximate a straight line; in this case, the region between the two dashed lines. That in this case there is a change in slope does not matter, since it coincides with the bad data, and the lines are fit separately on either side. So, first move the cursor to a point before the edit (left-hand dashed line) and type `; the pre-edit fit then runs from that point to the last good point (before the edit). Next, move to a point after the edit (right-hand dashed line) and again type `; the post-edit line is fit from the first good point (after the edit) to the point chosen. (The ` is used because on most keyboards it is easily reached with the left hand, leaving the right free to move the mouse). When you type the second ` the screen will be redrawn as

Figure 16

which shows that the offset has been removed. Now type **h** to return to paging mode, and then **ENTER**, and you will get

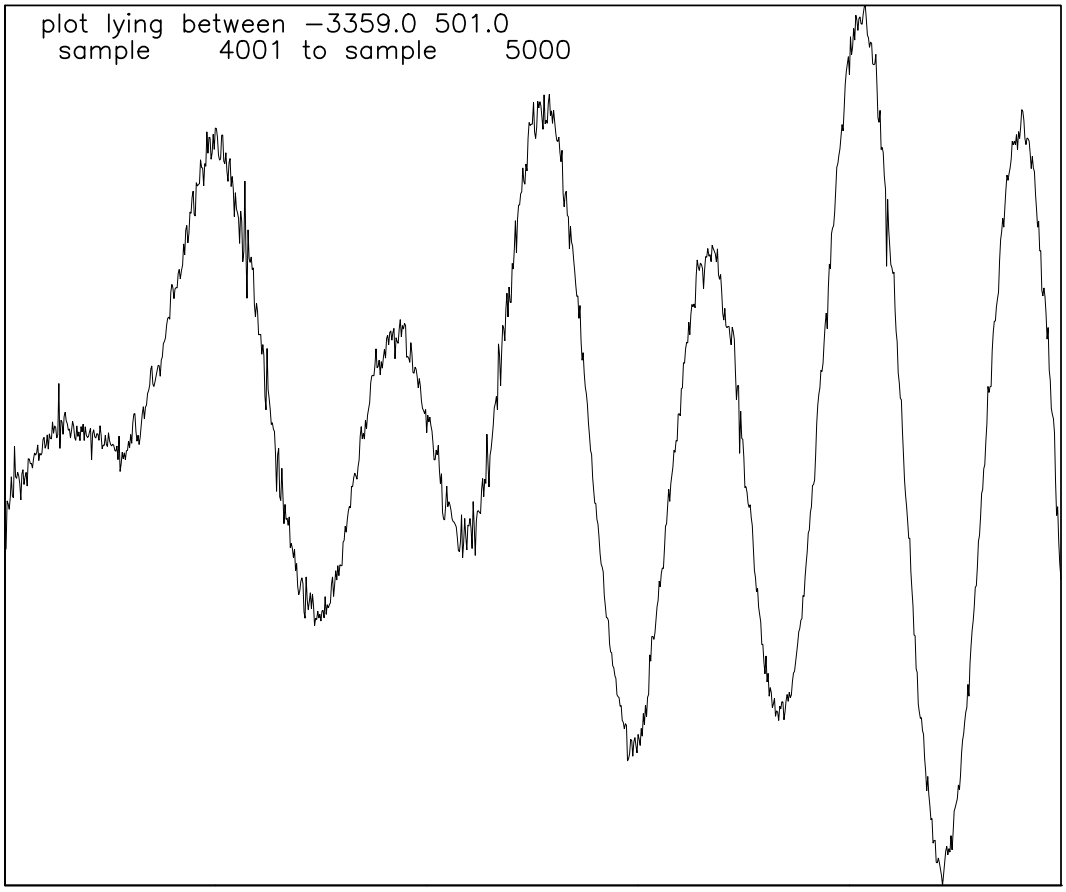

Figure 17

which is the edited version of Figure 9. Now resume paging by typing **ENTER** to get

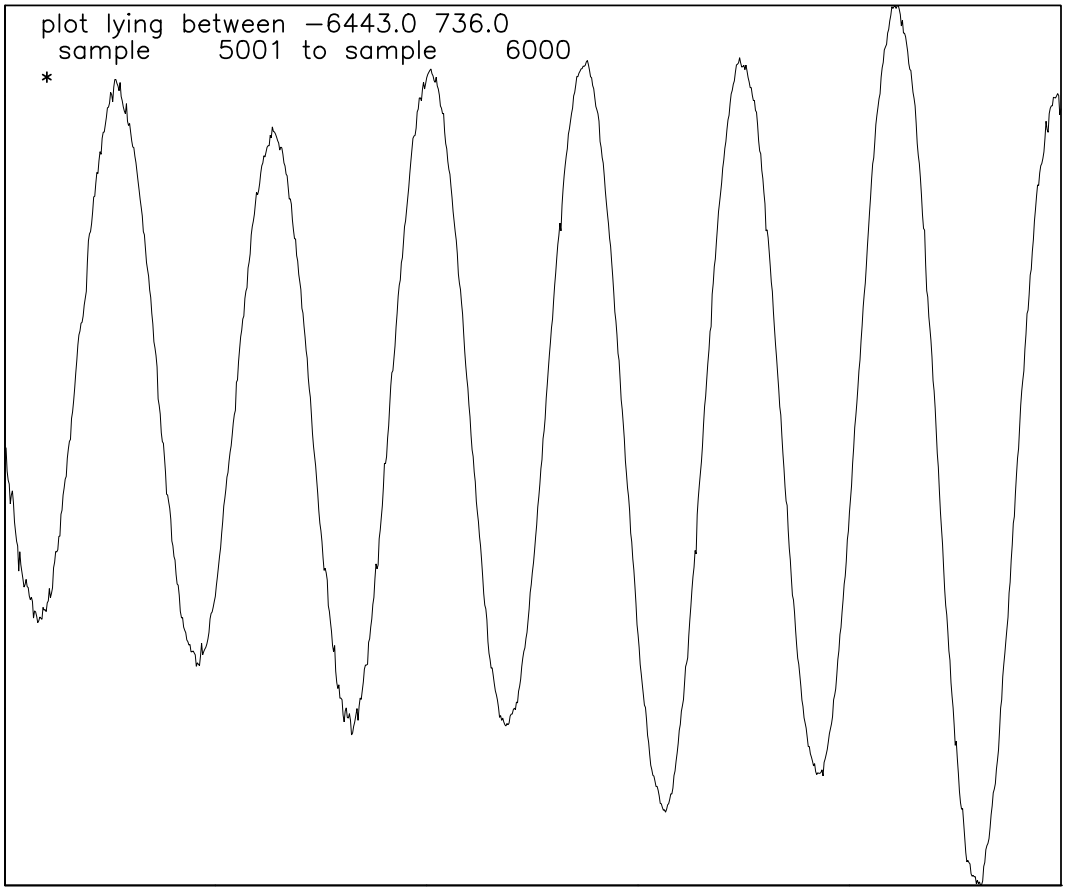

Figure 18

and again to get

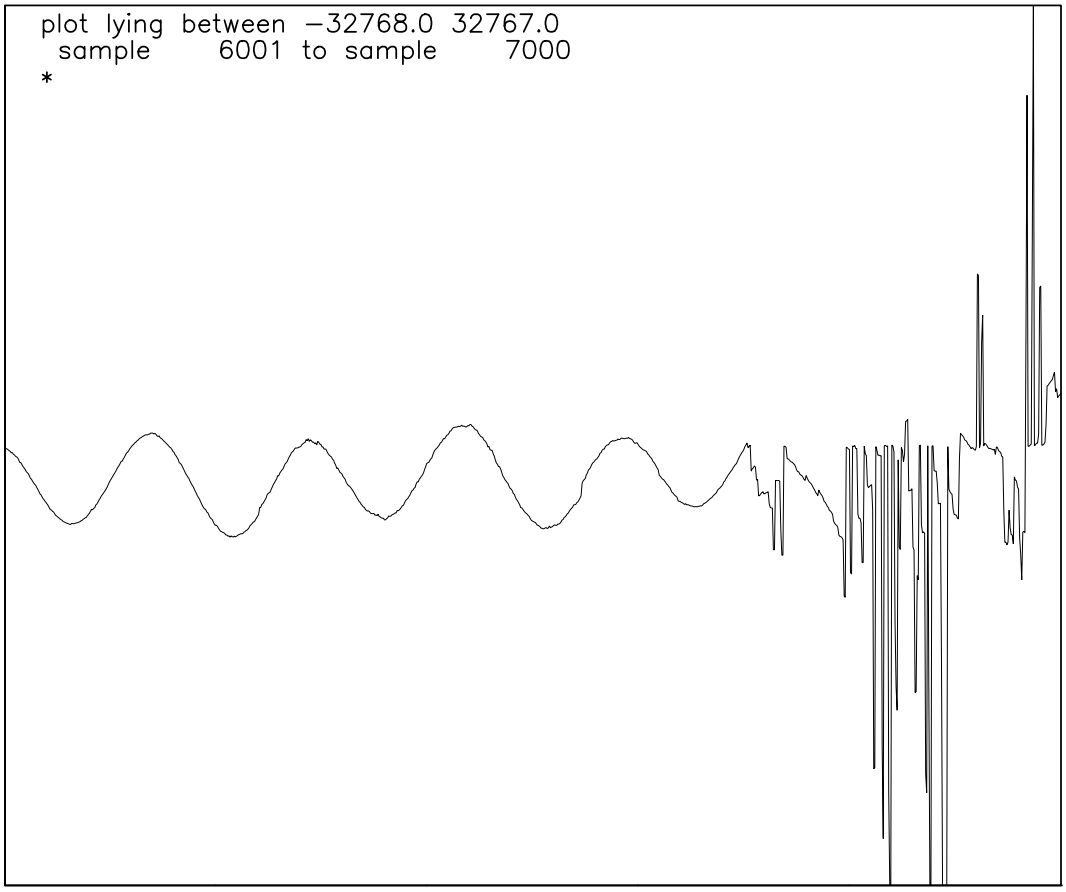

Figure 19

which shows the kind of problem that an automated editor would have difficulty dealing with: a longer span of bad data (in this case, caused by gradual misalignment of the strainmeter beam-steering). Given that this extends off the right edge, the best strategy is to look further ahead, which is best done by typing **`c ENTER`** to get back into cursor mode, and **`-`** to zoom out, giving

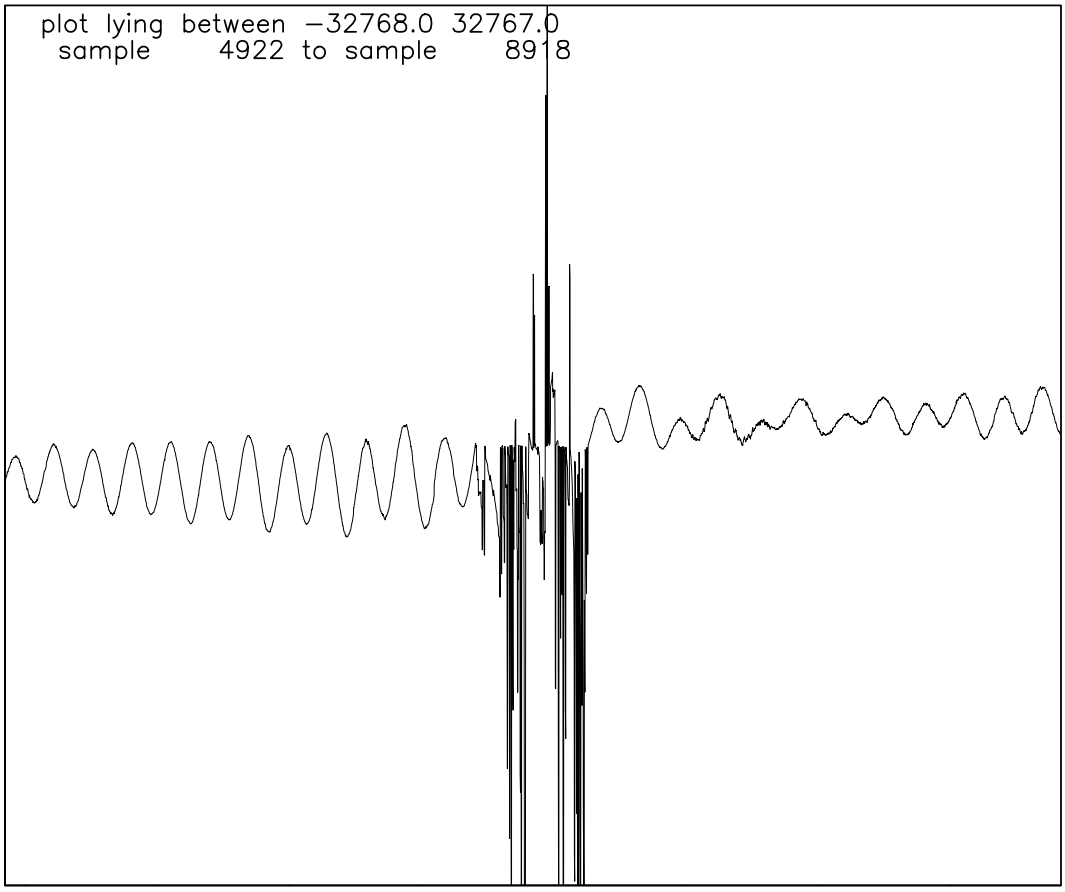

Figure 20

which shows the extent of the bad data. Place the cursor over the left-hand edge of the bad data and type **=** to zoom in, giving

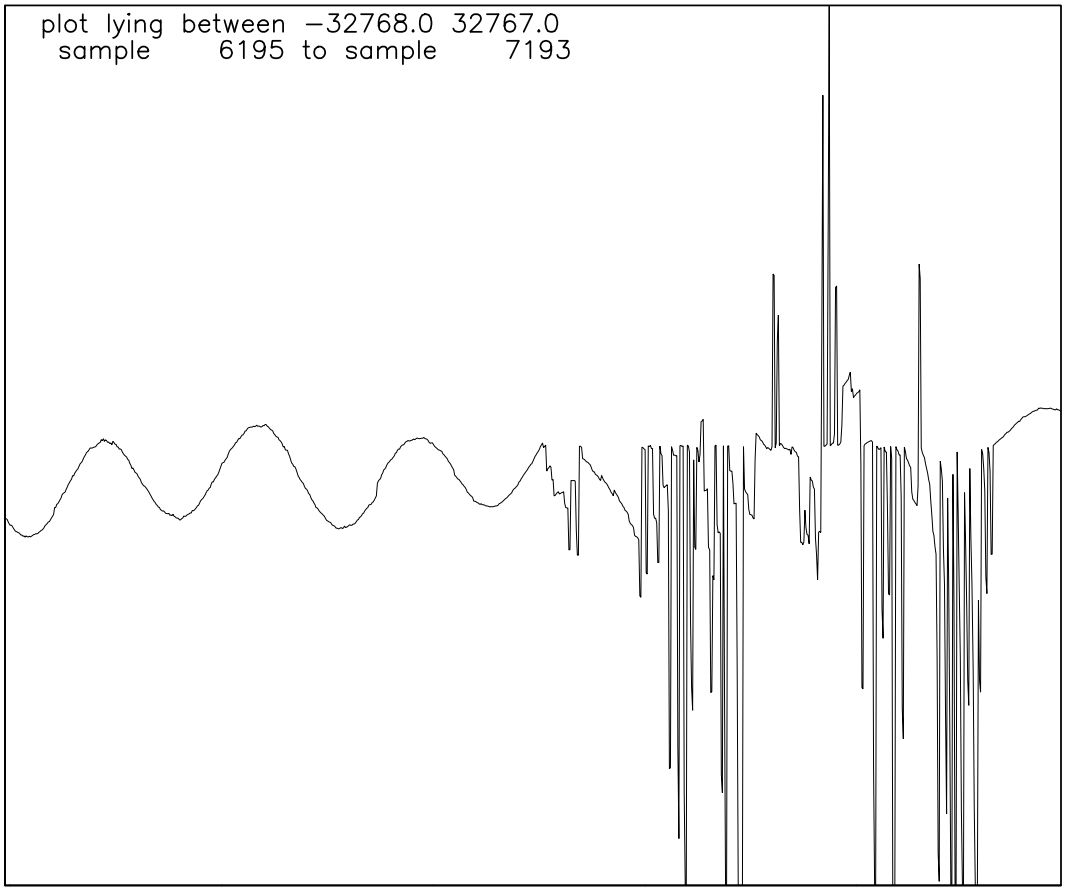

Figure 21

again place the cursor over the left-hand edge of the bad data and type **=** to zoom in further, giving

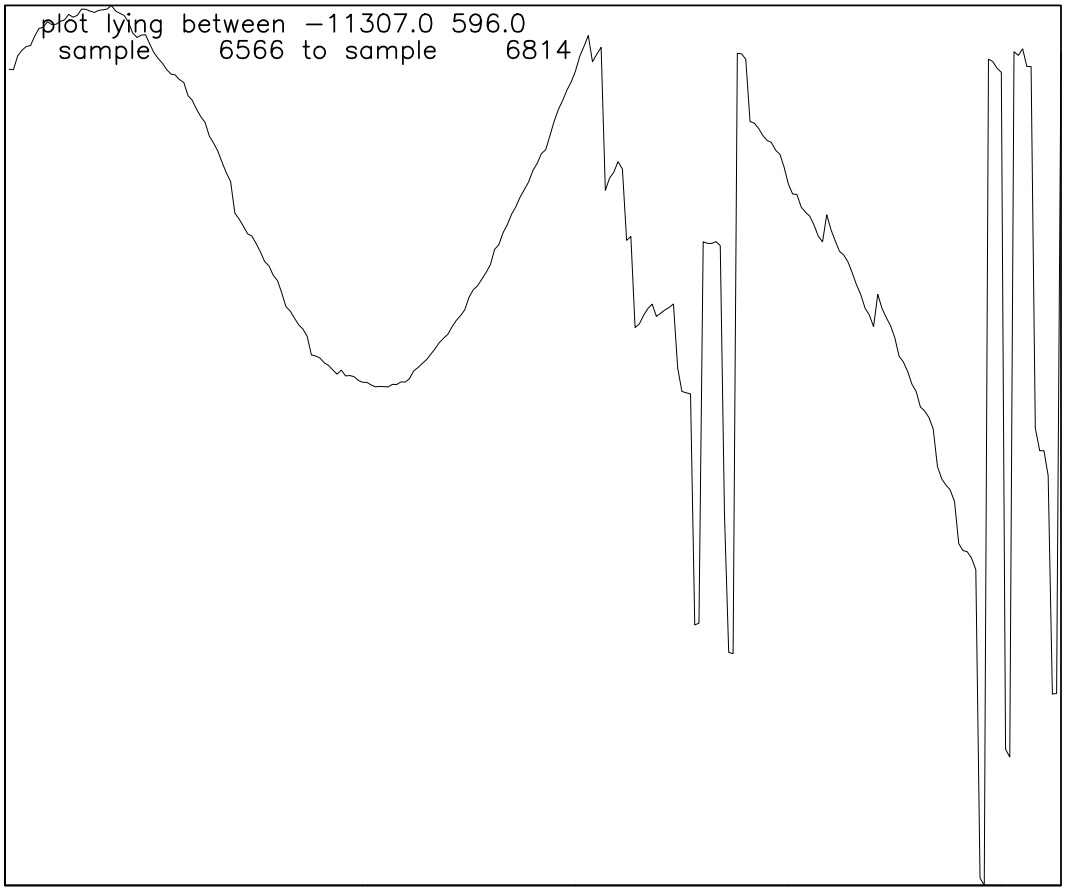

Figure 22

Having looked ahead, you know this will be an offset edit, so type **o** to edit the offset, indicate the start of bad data (center of screen) and type **1**, then **–** to zoom out, giving

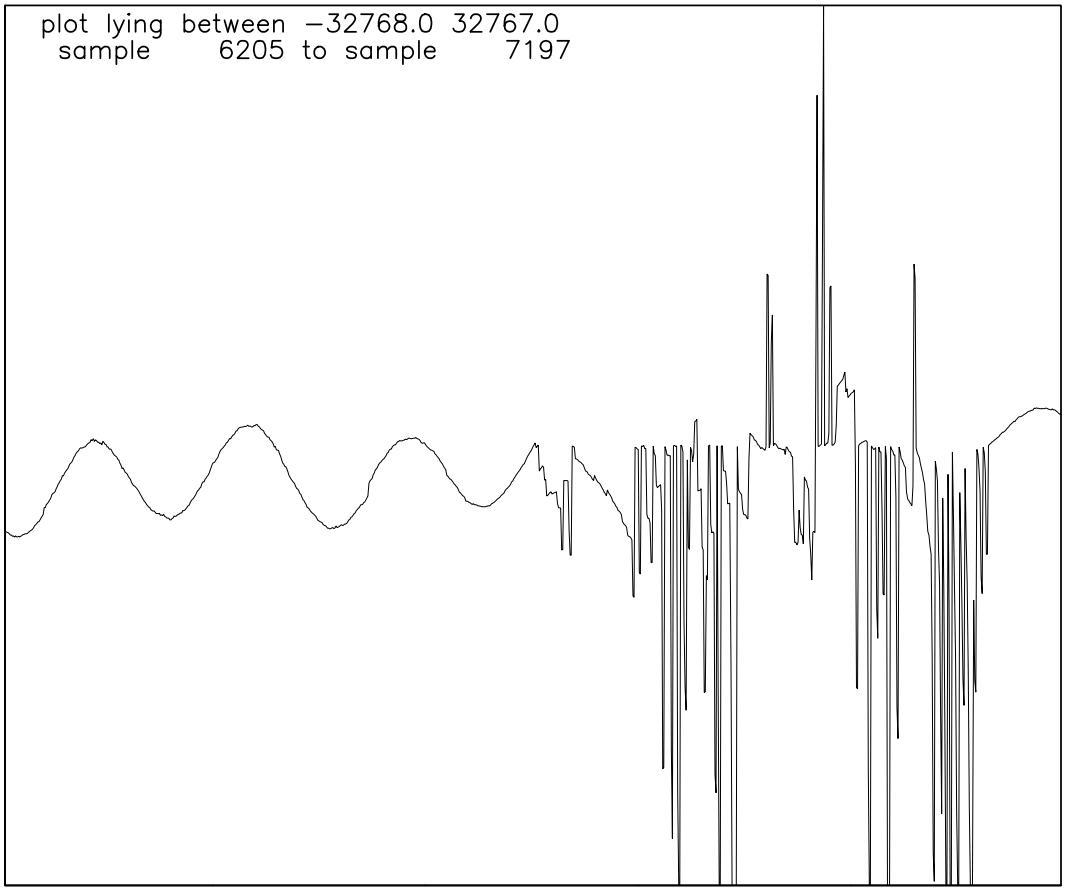

Figure 23

which, fotunately, contains the end of the bad data (if you weren't sure about this, you could zoom out and then back in). Place the cursor over the right-hand edge of the bad data and type **=** to zoom in, giving

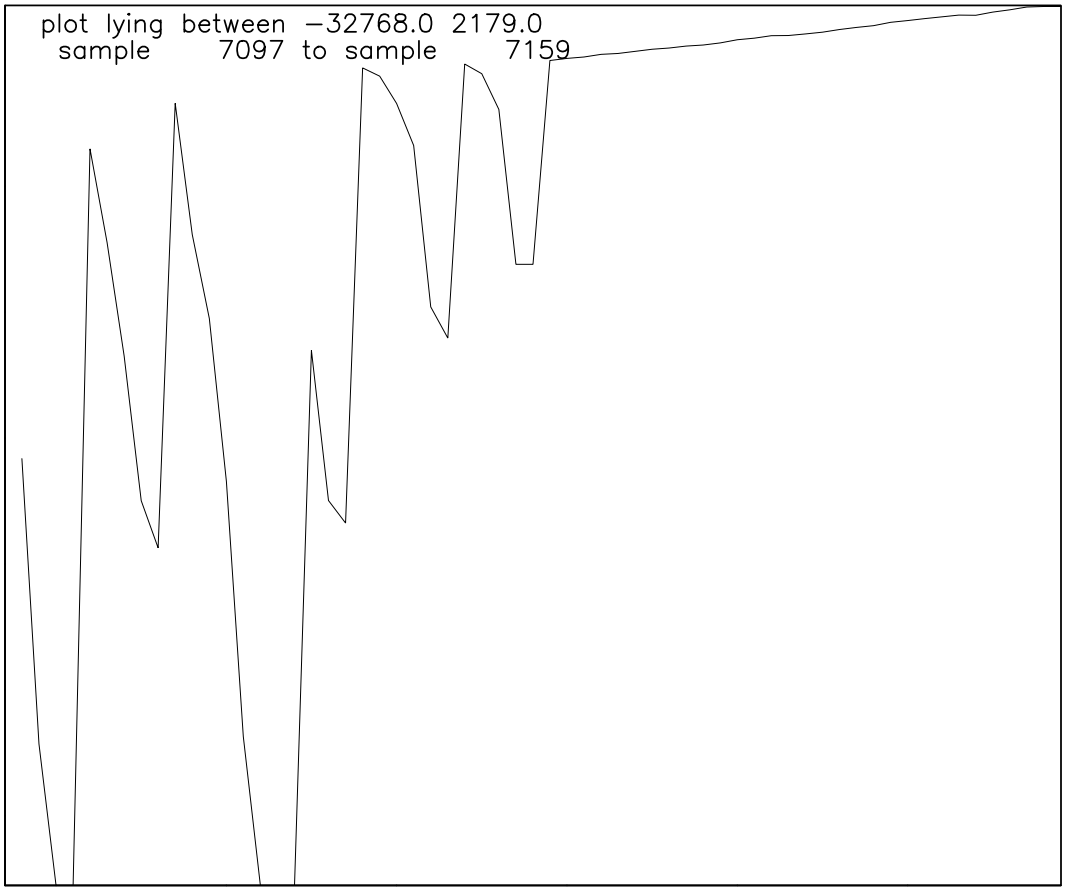

Figure 24

Indicate the end of bad data (center of screen) and type **2**, then **−** to zoom out, (do this twice) giving

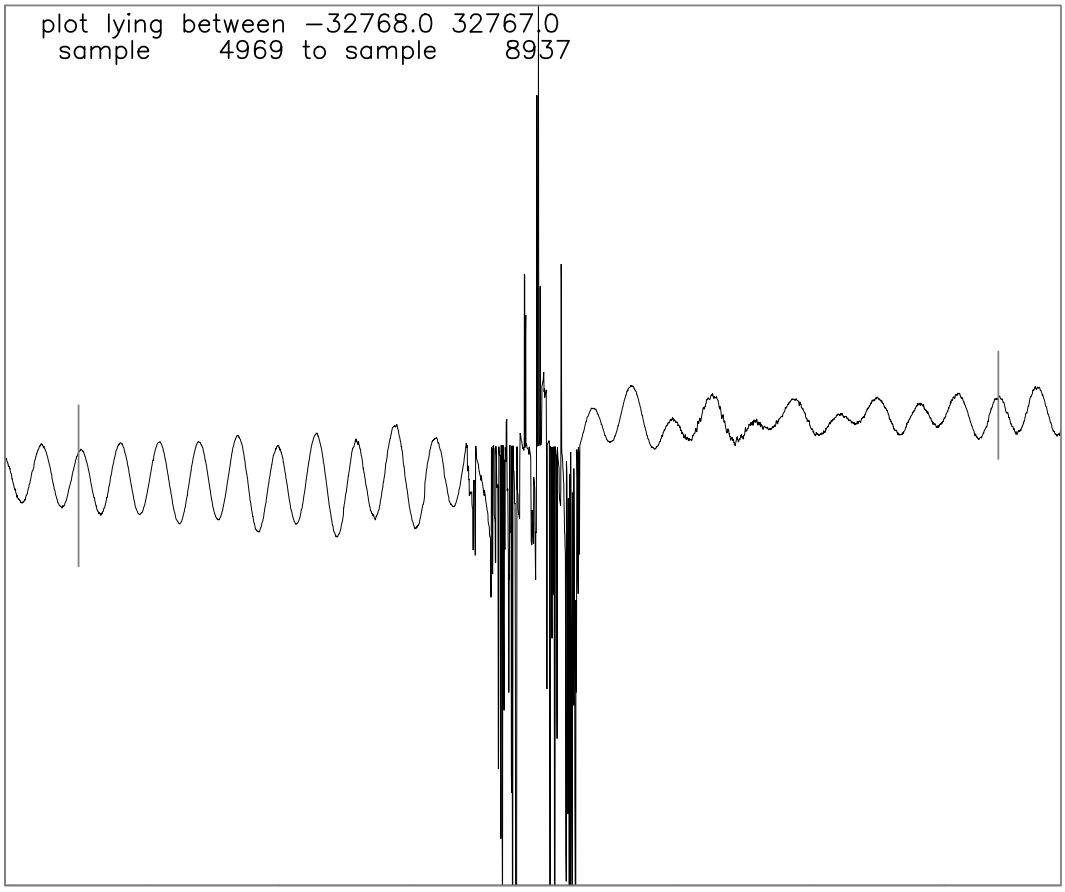

Figure 25

You should now type **6** to invoke the straight-line matching mode for finding the offset. The lines suggest where to place the edges for fitting: far enough that the fit will average over a number of tidal cycles. (In actual practice we would remove the predicted tides before editing, which means that the fit could be over a shorter span. The span of the fit should not exceed 10000 points.)

When you type the second ` the screen will be redrawn as

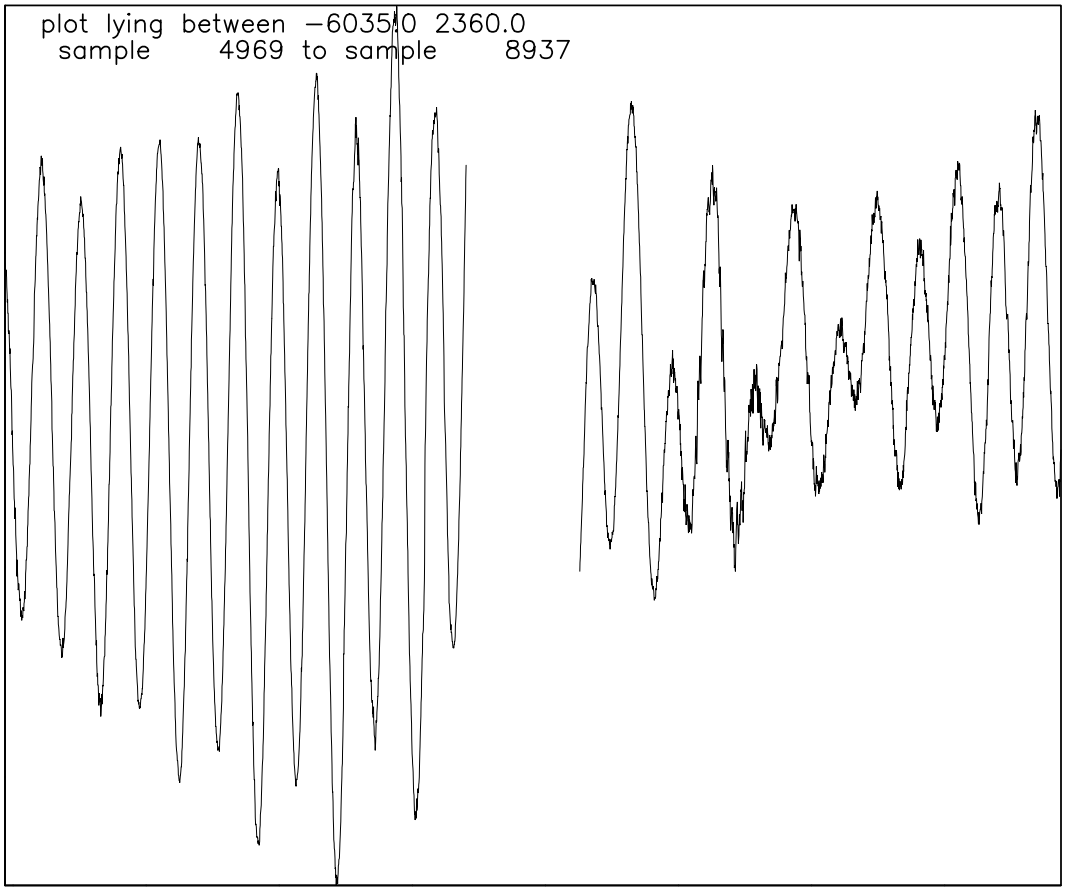

Figure 26

which looks much better. Now type **h** to return to paging mode, and then **ENTER**, and you will get

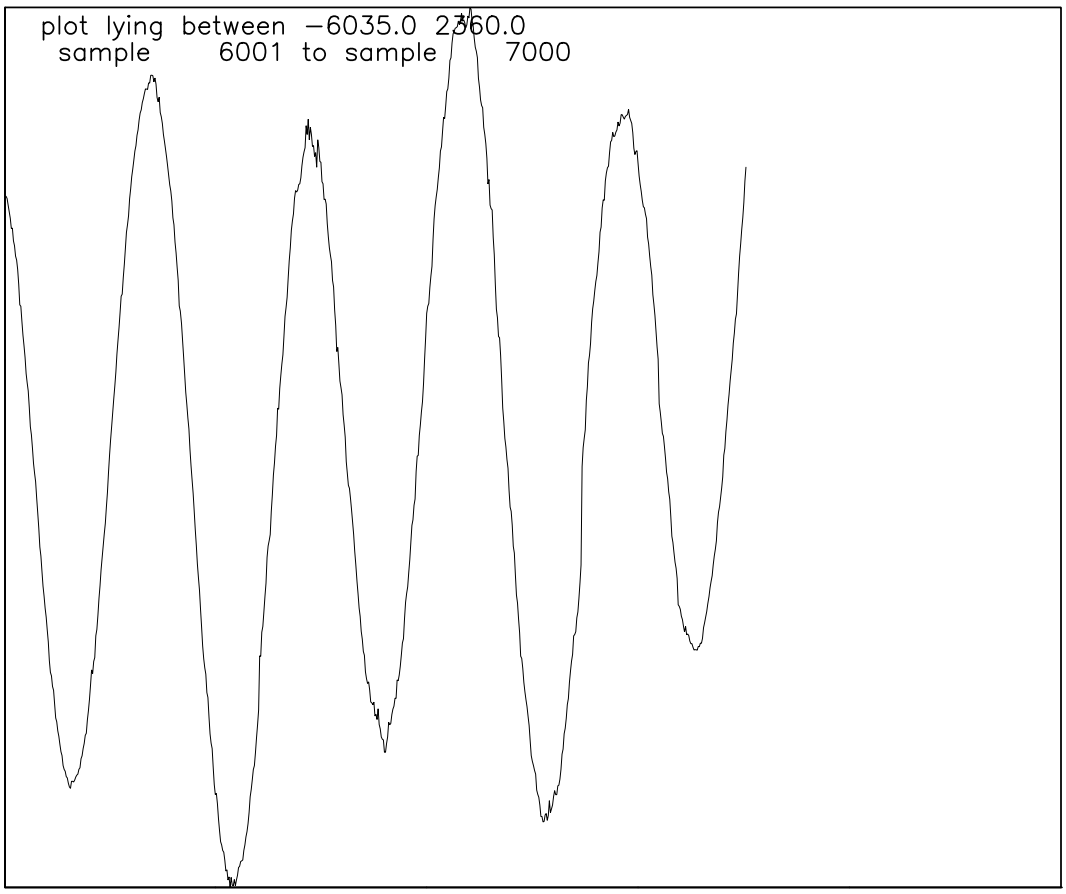

Figure 27

in which the bad data is blanked out, and the self-scaling shows the data at a much better resolution. Again type **ENTER**, and you will get the next page:

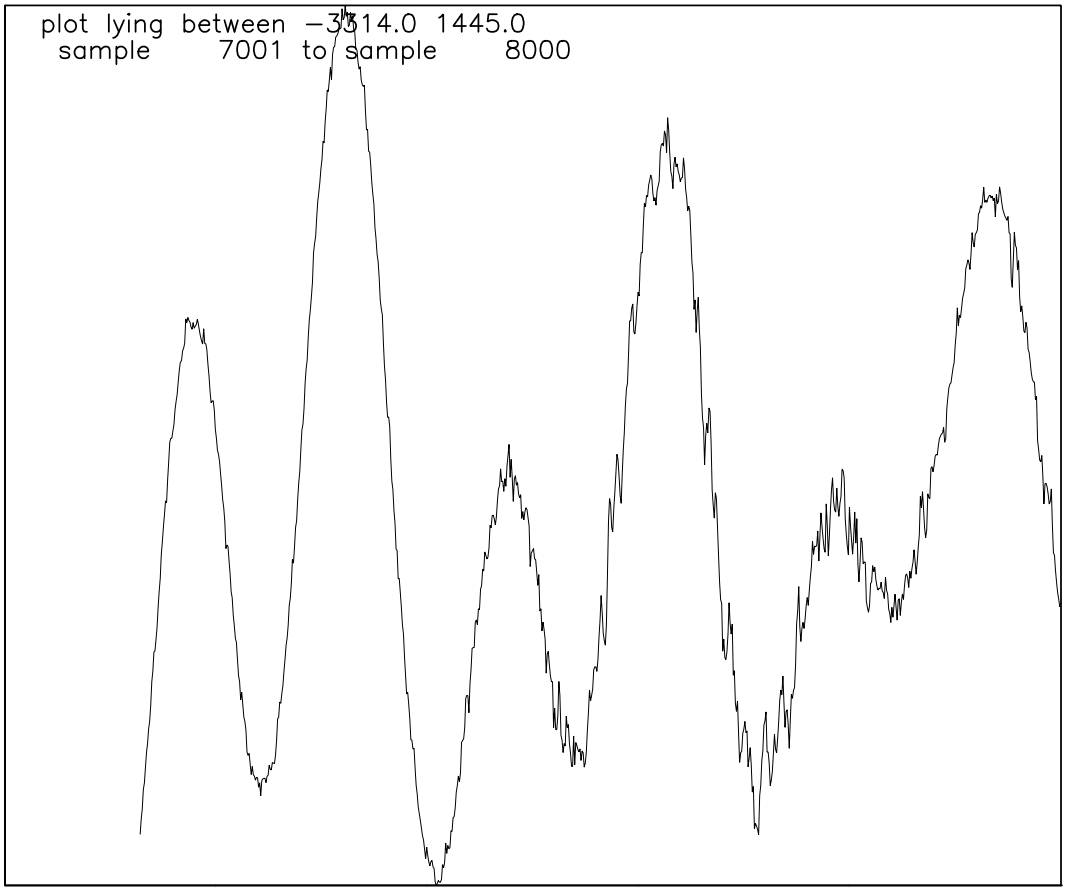

Figure 28

which shows a noisier data span on the right side of the plot. It becomes a matter of judgement what to do with this. We choose to leave it by moving to the next page with **ENTER**:

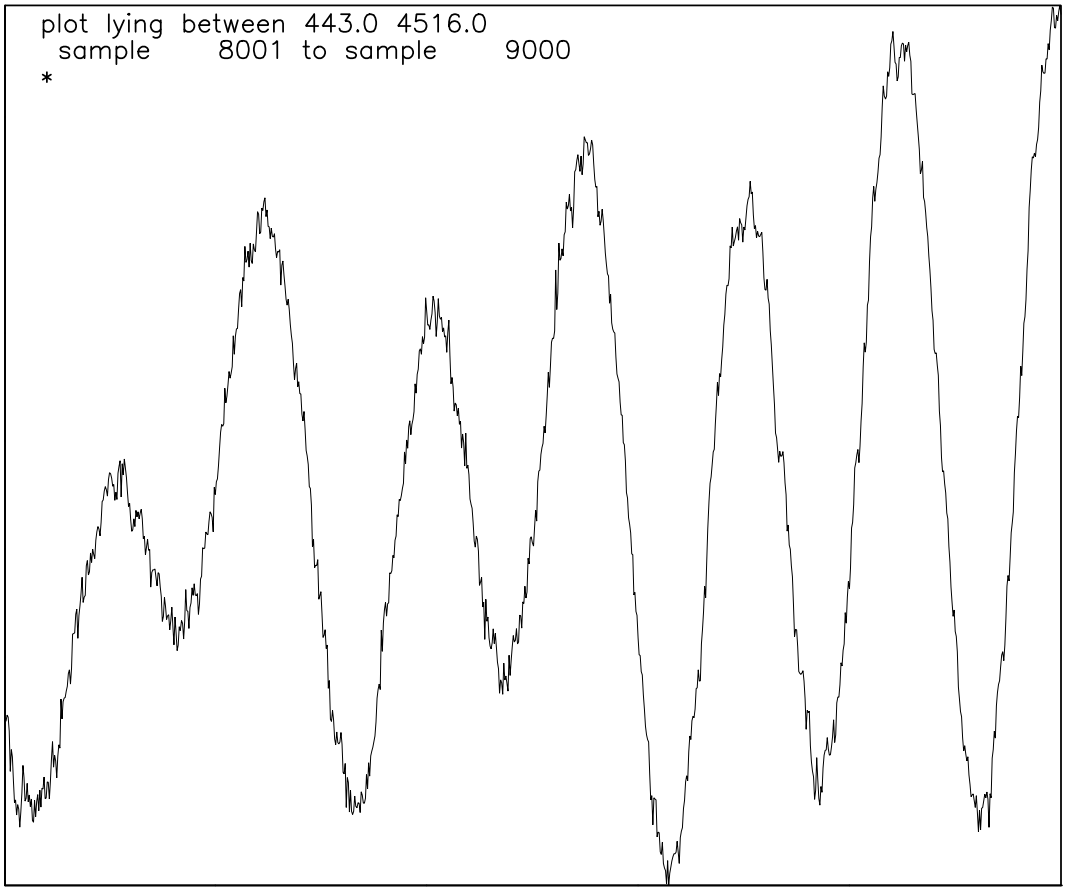

Figure 29

and again using **ENTER** we get:

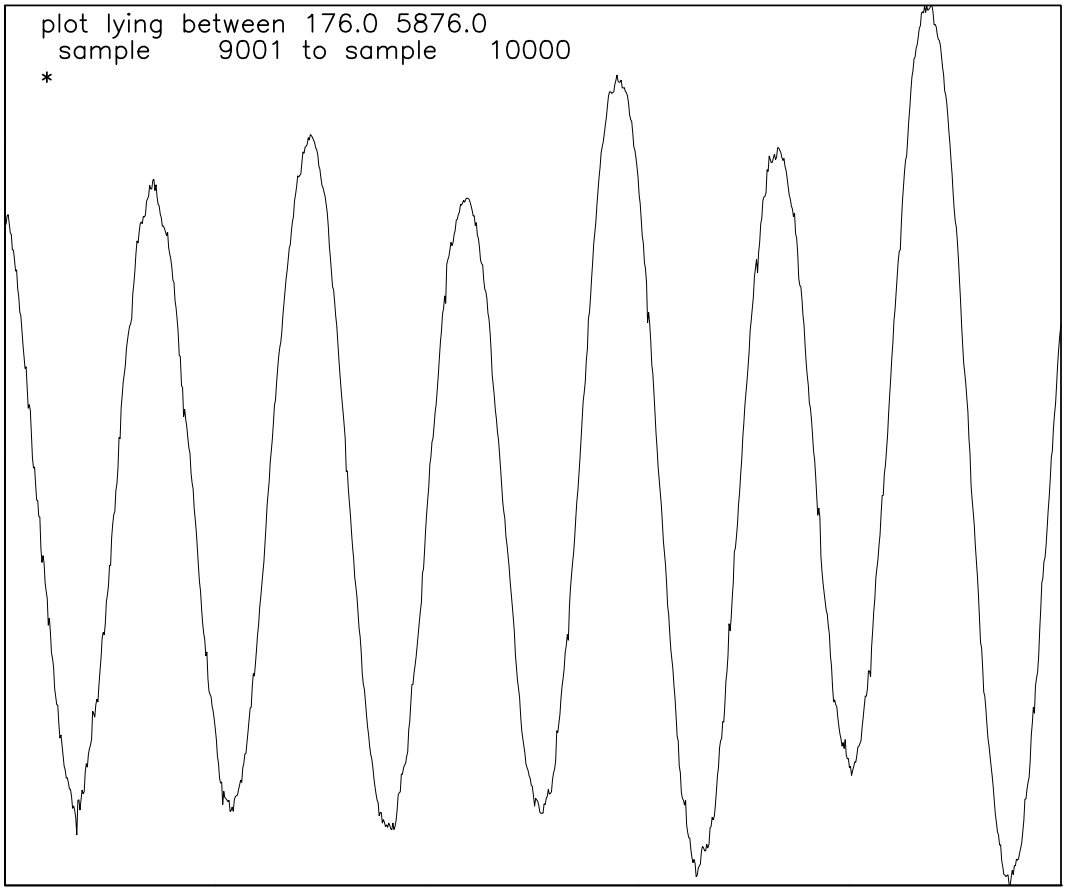

Figure 30

which is the end of the series, as the program will inform you if you type **ENTER** again. You could now exit the program by typing **x ENTER**, which would get you the message:

```
4 edits written to file 101
```

If you wanted to see what the edited file looked like before exiting, type **c ENTER** to get back into cursor mode, and **–** to zoom out, zooming until you have the whole series in view:

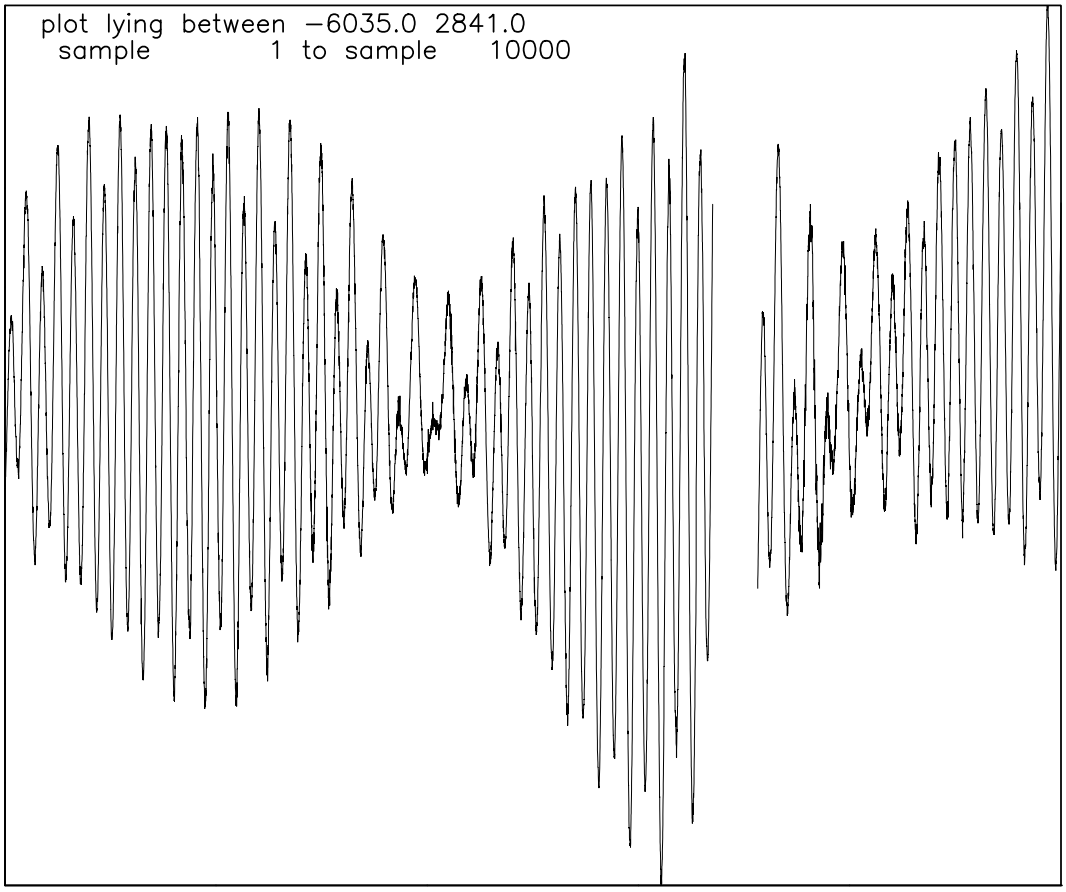

Figure 31

Since you are now in cursor mode, type **e** to leave it, then exit the program by typing **x ENTER**.