

# **IRCAM Signal Editor Manual**

**Institut de Recherche et Coordination Acoustique/Musique  
31, rue S. Merri  
F-75004 Paris France**

**Version of 28 January 1992  
Corresponds to Release 0.9 of the software  
Copyright 1992 IRCAM  
All Rights Reserved**

## **About SE**

### **Release Notes for Release 0.9**

**Known Bugs  
Caveat Emptor**

## **Installation**

**System Requirements  
Installation Notes**

## **Signal Document Reading and Writing**

**Creating a New Document  
    Setting the Default Directory  
    Setting the Default Sampling Rate**

**Opening a Document**

**Writing a Document**

**How SE Editing Works**

**Foreign Samples in Save Operations**

**Foreign Samples in Close Operations**

## **Making Copies of Signal Documents**

## **View Windows**

**Resizing and Hiding the View Window**

**Moving Forward, Backward, Up, and Down**

	Scrolling Via Scroll Bars
	Scrolling Via Scroll Arrow Buttons
	Zooming In and Out
	Continuous Zooming
	Partitioning a View Window
	Printing a View Window
<b>Selecting and Playing</b>	
<b>Editing</b>	
	Editing Time Bounds of a Selection
<b>The Change Stack</b>	
	Erasing the Change Stack
<b>Multiple Views of One Signal Document</b>	
<b>Using Named Segments in the Segment Browser</b>	
	Editing Segment Boundaries
	Multiple Segment Browsers
<b>The Document Inspector</b>	
<b>More About Signal Documents</b>	
<b>The Scheme Interpreter</b>	
<b>Credits</b>	
<b>References</b>	

## **About SE**

The IRCAM Signal Editor (SE) is a general-purpose digital sound editor that runs on NeXT computers. SE is particularly optimized for rapid display and editing of large sound files, which SE calls *signal documents*. Unlike other sound editors, opening and editing large documents takes no more time than small documents. Signal document navigation is flexible in SE. The program supports an unusually wide range of scrolling and zooming behaviors, including continuous scrolling and continuous zooming, as well as precise and repeatable incremental scrolling and zooming.

Another unique feature of SE is its capacity for "unlimited undo." SE keeps a history of all editing operations so you can go back to an earlier point in the editing at any time during a session, and even across sessions.

SE is supplied with a Scheme interpreter called Elk as its extension language (Laumann 1990). This is to allow developers and users to extend the functionality of the editor, particularly with respect to signal processing operations and the undo facility.

SE is distributed with the IRCAM Musical Workstation (IMW) software package. However, it is not necessary to have the IMW hardware in order to

run SE, since Release 0.9 SE runs on the NeXT's 680X0 microprocessor with no assistance from the Ariel IMW signal processor board.

## **Release Notes for Release 0.9**

Please note that Release 0.9 has several limitations:

1. Only mono (one-channel) signal documents are supported.
2. SE currently has no crossfading or mixing facilities.
3. The Scheme interpreter interface to SE is undocumented.

The SE user interface follows the NeXT conventions. In order to operate SE you should be familiar with these protocols, as they are not all documented in this manual. See *NeXT User's Reference*.

## **Known Bugs**

Sometimes when you are greatly zoomed in-near to the level of individual samples--vertical bars might appear. These benign bars are not in the data; they represent an error in the display code.

## **Caveat Emptor**

Release 0.9 is an experimental Beta release of SE and is not guaranteed. We recommend that you make a read-only copy of any signal document you intend to edit before invoking SE, so that if there is a problem the original signal document is intact. You can do this by invoking the Duplicate command in the NeXT Workspace Manager and using the Inspector within the Workspace Manager to make the document read-only.

We welcome comments and criticisms of SE. Please contact: [wyngaard@ircam.fr](mailto:wyngaard@ircam.fr). Or write to: Peter Wyngaard, IRCAM, 31, rue S. Merri, F-75004 Paris, France.

## **Installation**

This section specifies the system requirements and the installation procedure for Release 0.9 of SE.

### **System Requirements:**

To run SE we recommend that you have 16 Mbytes of random-access-memory on your NeXT machine. For disk storage, remember that 16-bit

mono signal documents sampled at 44.1 KHz take up approximately 5.2 Mbytes per minute. If you intend to save an edited document, you need to have enough disk space for any samples you inserted from another document. See the section on "How SE Editing Works" later.

## Installation Notes

This section duplicates the information found in the online **INSTALL** note supplied with the Signal Editor distribution. To obtain the Signal Editor distribution run the Unix shell program `ftp` to access the appropriate directory at the distribution site. The distribution is named **SignalEditor.tar.z**.

Once you have transferred the compressed file archive **SignalEditor.tar.Z** to your machine, the first step in installing SE is to unpack the file package. You do this by typing the shell command line:

```
zcat SignalEditor.tar.Z | tar xvf -
```

This command line uses the Unix `zcat` program to uncompress the file package and write it to its standard output. The `|` sign pipes this output to the `tar` program. The `tar` program extracts the files from the tar archive. (See the Unix documentation for more on `zcat` and `tar`, if necessary.) This creates a directory called **SE**. In this directory you will find a subdirectory for the manual, the Signal Editor application directory, and two text files: a **README**, and an **INSTALL** note.

## Where to Install the SE Application

In order for the NeXT Workspace Manager to properly handle signal documents, that is, to open them and launch SE when you double-click on them, you must install SE in one of the directories where the Workspace Manager looks for applications. The **Apps** directory in your home directory is one such place.

## Quitting SE

When you quit from SE (Command-q) all signal document documents are closed and the screen positions of the Scheme, Document Inspector, Help, and Font panels are saved. The next time you launch SE these windows will appear in the same position. This lets you customize your screen layout.

## Signal Document Reading and Writing

SE supports two file formats: NeXT's standard **snd** format, and its own native **sig** format. You can tell what format a file is in by looking at it with the NeXT Workspace Manager and noticing its file extension, for example, **flute.snd** or

**sitar.sig**.

SE editing is performed exclusively on **sig** documents. The structure of a signal document is what makes SE editing fast. See the section "More About Signal Documents" for details on their format.

### **Creating a New Document**

You can create a new document in four ways:

1. By selecting New Document in the Document menu (or Command-n)
2. By selection New From NeXT File in the Document menu
3. By creating a new view window, or empty viewer, by selecting New Viewer Window from the Windows menu (command-N), and pasting a selection from another document into it
4. By dragging a NeXT **snd** file icon from the Workspace Manager into an empty viewer

When creating a new signal document from a NeXT **snd** file, recall that SE supports monophonic, 16-bit linear uncompressed samples with either a 22.05 or 44.1kHz sampling rates. The new signal document will have the same sample format as the **snd** file. Sample format conversion is not supported in release 0.9.

SE treats all new documents as temporary documents. That is, by default SE labels the new document as **untitled $n$ .sig** (where  $n$  is an integer) in the **/tmp** directory. (In a moment we explain how to change this default location.) When you close the documents, SE will pop up an alert box to ask you if you want to delete the temporary untitled document or save it.

In Release 0.9 it is not possible to rename an open document or save it under another name. The workaround for this, which applies only to new documents, is as follows:

1. Close the unsaved document. At this point SE asks you if you would like to save it.
2. Choose the Save operation. Now SE gives you the opportunity to rename the file and save it in the directory of your choice. Note that if you click on Cancel in the Save panel, this cancels the save operation but not the previously issued close operation.

So in Release 0.9 a good strategy is to close and save under a new name immediately after you create a new document. Then simply open the document you just named and saved. You can, of course, rename non-open

documents with the Workspace Manager at any time.

### Setting the Default Directory for New Documents

In release 0.9, you can specify the default new document directory, or *untitled directory*, with a Unix shell command as follows:

```
dwrite SignalEditor UntitledDirectory "new untitled directory"
```

(See the Unix documentation for more on the dwrite command.) Do this in a terminal window while SE is running. The next time that SE creates a new untitled document it will use the directory that you specify. By default, SE uses **/tmp** as the default new document directory.

### Setting the Default Sampling Rate for New Documents

You can set the default sampling rate for new documents by means of the Unix shell command dwrite. Just type the following after you have opened SE:

```
dwrite DefaultSamplingRate rate
```

where *rate* is either 22050 or 44100. (Note: don't add punctuation to these numbers, like 22.050 or 44,100.) The default sampling rate is only used when you create a new document with Command-n. If you create a document from a NeXT **snd** file, the **snd** file's sampling rate is used.

### Opening a Signal Document

You can open an existing signal document by choosing Open from the Document menu (Command-o). SE presents you with an open panel. You may either directly type in the full file name, or use the panel's browser to find and double-click on the document. The open panel first looks in the directory where the *current document* resides. The current document is the document in the *current document viewer window*, indicated by a black or dark grey bar at the top of the window. If there is not a current document, SE looks in the user's home directory (if this is the beginning of a new session), or the last directory accessed in this session.

\*Note: If you try to open a document name that is a *symbolic link* to a signal document, SE warns you that it is going to use the real name of the document instead.

### Writing a Document

When you are done editing, SE can either save the signal document or write out a NeXT's **snd** file. Having the file in **snd** format means you can apply other NeXT sound tools to the file, or transfer it in the digital domain to another digital audio device.

To save a signal document select Save in the Document menu or type Command-s.

To write a signal document to an **snd** file select Write to NeXT File, in the Document menu.

To close a signal document, select Close in the Document menu.

\*Note: There is no file-locking mechanism in SE. You should be careful when you have multiple SE applications running that edit the same document. The last save of the file may overwrite previous work done by someone else.

### **How SE Editing Works**

Opening and editing signal documents is fast in SE because of the structure of the **sig** file format. It is important to realize that SE editing manipulates pointers only; it does not rearrange the order of the samples in their original signal documents on the disk. Imagine, for example, that you have opened five source documents that you intend to cut sections from them to create a new master document. Every time you cut and paste from a source into the master, the display of the master changes to reflect its new contents. SE plays the master document seamlessly, giving you the illusion that all of the samples have already been written to the master document. In actuality, SE is following pointers in the master and reading from the different source documents in order to play the master.

### **Foreign Samples in Save Operations**

When you decide to save an edited master document, SE needs to make sure that any samples that have been inserted from a source document, or *foreign document*, are copied on disk into the master document. This is done because after you quit SE, you might delete, move, or change the name of the foreign document making it impossible for SE to access the foreign samples. When you save a master document containing foreign samples, SE guides you through a sequence of alert panels. You always have the choice to cancel the save.

A special situation can arise when the foreign samples are references in undone paste operations stored in the change stack (see the section on the change stack for more information). In this case, SE gives you the option to

remove all undone operations from the change stack, leaving only the modifications which influence the current state of the document. Choosing to remove the undone change stack entries can greatly reduce the time and disk space necessary for saving a document.

### **Foreign Samples in Close Operations**

Imagine that you have two documents open, document A and document B, and that Document B includes samples from document A. If you try to close document A, SE will alert you and give you a choice between cancelling the close or copying the samples from document A included in document B, into document B. If any samples from document A are references in undone change stack entries in document B, you can also opt to remove the undone entries from document B's change stack.

When you save a document (Command-s) that includes foreign samples, SE appends the foreign samples in the order in which they were pasted into the document. This allows SE to remove samples from a document if a paste is undone, keeping a document's disk usage to a minimum. In the above example, samples from document A had to be copied into document B because document A was closing. These samples might not have been appended onto document B in the optimum order, and SE might not be able to remove the samples from the document if the paste is undone, thus wasting disk space. It would have been a better idea to cancel the close, save document B, then close document A.

You will also be warned if the pasteboard contains any parts of a document being closed. You can either clear the pasteboard or cancel the close operation.

### **Making Copies of Signal Documents**

You can make copies of documents, either using the Unix shell (`cp -r` command) or using the Workspace Manager.

\*Note: Either method of copying replicates the version of the file that is saved on disk. As with other editors, we recommend that you do not copy a document that is open for editing in SE.

### **View Windows**

A *view window* displays a part of or an entire signal document. The sole *view type* available in Release 0.9 is a *time-domain* display, which shows the waveform contained in the file as a graph of amplitude (vertical axis) versus



time (horizontal axis). The units on the amplitude line show the sample value, between + and - 32767 for the 16-bit integer signal documents handled by SE. All units on the time line are in seconds or decimal fractions of seconds.

## **Resizing and Hiding the View Window**

You can resize and hide a view window using the standard NeXT window resizing protocols. See *NeXT User's Reference*.

## **Moving Forward, Backward, Up, and Down**

View Window scrolling is very flexible in SE. The main tools are the *scroll bars* at the bottom and left side of the screen, and the *scroll arrow buttons* (horizontal and vertical) in the left bottom corner.

### **Scrolling Via Scroll Bars**

Scroll bar scrolling works in two modes of resolution: *regular* and *fine*. In regular resolution, you can move forward or backward in a view window by dragging the horizontal scroll bar at the bottom of the view window. To move up or down the display, use the vertical scroll bar at the left of the view window.

Pressing the Alternate key as you move the scroll bar makes the scroll bar work in fine resolution mode. The speed at which you move the scroll bar determines how far and how fast the scroll occurs. Hence this is a *velocity-based* scrolling mode.

### **Scrolling Via Scroll Arrow Buttons**

The scroll arrow buttons at the left bottom of the view window work in a precise manner. Every time you click on one of them, the screen scrolls by a tenth of a screenful. If you press Alternate and click on a scroll arrow button, the screen scrolls 90%, leaving 10% of the previous screen displayed.

## **Zooming In and Out**

SE view windows let you look at waveforms at various levels of resolution by means of *zooming* in and out. SE supports two types of zooming ranges:

1. Zoom in/out by a fixed factor of two
2. Zoom to fit a selection to the size of the view window

Both *horizontal zooms* (zooming in/out on the time axis) and *vertical zooms* (zooming in/out on the amplitude axis) are supported. Vertical zooms let you look a piece of a waveform, such as a low-amplitude signal, in more detail

than you would normally see.

The zoom applies to the center of the display. That is, whatever was in the middle of the screen remains in the middle as you zoom in and out.

Here is the complete list of zoom commands:

Horizontal in (fixed factor)	Command-[
Horizontal out (fixed factor)	Command-]
Vertical in (fixed factor)	Command-{
Vertical out (fixed factor)	Command-}
Vertical center the center)	(scrolls display back to Command-
Vertical all (full zoom out)	Command-_
To selection (zooms to fit selection to window)	Command-=
Show all (full zoom out horizontally and vertically)	Command--

### **Continuous Zooming**

A convenient feature in SE is continuous zooming. Shift-mouse click on either the horizontal or vertical scroll bar causes the scroll bar to grow or shrink in a continuous manner as you move the mouse. The window zooms to follow your mouse movement.

\*Note: this feature is not mature in Release 0.9, meaning that you should not be surprised by anomalies in its behavior.

### **Partitioning a View Window**

viewer.534133.tiff ↵

A single view window can be partitioned into a multiple viewer. You can either look at several different parts of a single document or look at multiple documents.

To partition a view window, click on the thin borderline around the document viewer and drag to position while also pressing the specified modifier keys:

Create a new partition	Alternate-mouse drag
Move an existing partition	Mouse and drag
Delete a partition	Alternate-mouse and

end	Create a partition along a complete side	dragbeyond partition
mouse and		Alternate-shift-
partitions along a complete side		drag Delete
and		Alternate-shift-mouse
end		drag beyond partition

Clicking on a partition with the mouse selects it as the current view.

## Printing a View Window

The Print menu (Command-p) prints the current viewer in high-resolution PostScript format.

## Selecting and Playing

To select a portion of a signal document to edit or play, simply press down on the mouse and drag in a view window. You can select in the vertical dimension by pressing the Alternate key as you drag the mouse. SE scrolls the screen if your selection extends beyond the current screenful.

If you press the Shift key and click on the mouse, the selection grows or shrinks to the point at which you click the mouse, depending on if you click outside or inside of the existing selection. Shift-Alternate-mouse-click causes the same selection behavior as Shift-mouse-click, except that the vertical dimension is also affected.

If you select a segment of length zero (by clicking without dragging), the selection flashes (provided that the window is the current view).

After you've made your selection, these selection-oriented commands become operational:

Play	Command-,
Stop play	Command-.
Zoom to selection	Command=

When a signal document is playing, a vertical *play cursor* moves through the selected region in increments of 0.01 seconds.

## Editing

SE executes a range of editing operations on selections within signal

documents. Some of these editing operations are *registered* (i.e., recorded) on the *change stack* (CS), which means that they can be undone. See the next section for an explanation of the change stack.

You can reach SE editing operations through the Edit menu:

equivalent)	Cut	(selection to pasteboard and register on the CS)	Command-x
	Delete	(cut selection without saving to pasteboard and register on the CS)	(No key)
	Copy	(selection to pasteboard)	Command-c
	Paste	(selection insert at cursor point and register on the CS)	Command-v
	Undo	(previous editing operation and register on the CS)	Command-z
	Redo	(previous editing operation and remove the previous Undo from the CS)	Command-Z
		(*Note: uppercase Z)	
	Select all	(selects the contents of the entire view window)	Command-a

### Editing Time Bounds of a Selection

SE lets you manipulate not just the samples within a selected region, but the *time bounds* of that region. For example, if a given selection starts at 10 seconds and ends at 30 seconds, you can cut and paste these time points "Beginning 10 seconds" and "Ending 30 seconds" from one window to another. In other words, a copy time bounds command causes the Signal Editor to store the begin time and end time of the current selection in the pasteboard, not the data that is between the two times. This feature lets you align editing points in several view windows. The time bounds commands are found under the Selection menu:

Copy Selection	Command-C
	(*Note: uppercase C)
Paste Selection	Command-V
	(*Note: uppercase V)

Here is an example of their use. We have two viewer windows open for document **DrumSound**, viewer *A* and viewer *B*. In viewer *A*, we select a region that we want to view more closely, that is, zoom to, but we don't want to zoom in viewer *A* because we already have it zoomed into an interesting portion of the signal. So we want to use viewer *B* to see the selected data in viewer *A*.

1. In viewer *A*, we copy the time bounds using the keystroke Command-C (Copy in the Selection menu)
2. Then we select viewer *B* by clicking on it
3. Then we paste the time bounds from viewer *A* to viewer *B* by typing Command-V (Paste from the Selection menu). This makes the bounds in viewer *B* the same as the bounds in viewer *A*.
4. Then we zoom to the time bounds by typing Command-=. We could have also typed command-'(Paste and Zoom in the Selection menu) which does the paste and the zoom in one step.

## **The Change Stack**

SE implements an unlimited undo facility. This means you can go back to a previous state of a document by undoing a series of editing operations. SE keeps a history of editing operations inside the signal document itself. This history is retained even across editing sessions.

SE's unlimited undo facility is made possible by an internal data structure called the change stack. In order to understand the undo facility, it is important to realize what operations are registered (i.e., written or saved) on the change stack and what are not. Specifically, the change stack keeps track of cut, paste, and delete operations and all undos and redos of operations.

You can "redo" an editing operation by "undoing" an undo, for example. Redo removes the previous undo from the change stack, so there will be no record of that undo in the future.

The undo facility does not apply to every operation that you can possibly perform while in SE. Do not expect to undo document-related operations such as close or save, or window-related operations such as create new viewer, scroll, zoom, select, etc.

## **Erasing the Change Stack**

When you save a file, its change stack is saved along with it. This history may include portions of signal documents that were cut out of the document, that is, that are no longer visible. Thus the change stack may reference many megabytes of data. If you are sure that you do not want to undo a change and you would like to compress the size of your signal document, the only way to do this in release 0.9 is to save the file as a NeXT **snd** file using the

command:

Write to NeXT File, under the Document menu

This operation writes a fresh contiguous NeXT **snd** file. If you would like to edit the file some more, you can create a new signal document from this **snd** file with the command:

New from NeXT File, under the Document menu

## Multiple Views of One Signal Document

When you open a file, SE invokes a view window and places a representation of the entire signal document in that window. In addition to this initial view window, SE lets you create any number of views of a given open signal document. Each view can be scrolled, zoomed, and resized independently of the other views.

No matter how many views that you have open, it is important to realize that all of them refer to the same underlying signal document. If you change the data in one representation by a cut, paste, or delete, this data change is reflected in every view window.

In order to create a new view window, select the New Viewer Window item under the Windows menu or type Command-N. This creates a view window called EMPTY VIEWER. There are three ways to insert a signal document into an empty viewer.

Method 1. Invoke the Document Inspector (Command-D), which shows a list of open signal documents. Select a signal document by clicking on its name. This puts the name of the selected signal document next to an icon in the top left corner of the Document Inspector. Select the icon and move the icon across the screen to the empty viewer. Release the mouse, which drops the signal document into the viewer.

Method 2. Copy a portion of an open document (Command-c) and move the mouse to the EMPTY VIEWER window. Paste the portion into the window (Command-v).

Method 3. Drag a **sig** file icon from the NeXT Workspace Manager into an EMPTY VIEWER window. The border of the viewer highlights to indicate that it accepts the signal document. You can also drag NeXT **snd** files, which creates a new untitled signal document from the **snd** file before placing it into a viewer.

\*Note: Although you can run multiple instances of SE at the same time, at present, *interapplication pasteboard editing* does not work between these instances. That is, you cannot copy signal data from one running instance of SE to another running instance of SE.

## Using Named Segments in the Segment Browser

A *named segment* (called segment in the rest of this text) is a selected portion of a signal document that you assign a unique name to in the Segment Browser. Named segments are a way of labeling parts of signal documents for easy access, as we explain in a moment.

To invoke the Segment Browser, select Segments in the Utilities menu, then select Browser. Or type Command-B.

To create a named segment, first select a portion of a signal document in a view window, then click on the Segment button in the Segment panel. At the top left of the segment panel is an icon with the name **NewSegment**. You can edit this name to rename the segment. Remember to hit the carriage return button to finish the operation.

WNGraphic.341306.tiff –

To select a named segment, simply click on a name in the list of segment. The figure above shows the named segment "Transition." Note that this does not affect the highlighted selection in the view window. To set view window's selection to the currently selected segment in the segment browser, you must click on the Select button. To play a segment, simply double-click on the segment name.

WNGraphic.753006.tiff –

## Editing Segment Boundaries

The Segment Browser shows the *offset* (starting time), length (in seconds) and end time of the current segment. You can edit any of these fields to change the boundaries of a named segment.

The Move box determines how the segment boundaries change if you edit the "Length" field in the Segment Browser. The default position of the Move button is "End," which means that the endpoint of the segment is extended or shrunk as you edit the length of the segment. The "Offset" button moves the beginning point (starting time) of the named segment, and the "Both" button

adjusts both boundaries equally.

In release 0.9, the only way you can change the boundaries of a named segment is by editing the fields in the Segment Browser. Note that SE does not automatically update them when you edit a signal document. For example, suppose you have a named segment "Intro" that points to the region between 1.0 seconds and 2.0 seconds. If you cut out a half-second from 1.25 to 1.75 seconds, "Intro" still points to the region between 1.0 and 2.0 seconds--whatever happens to be there.

### **Multiple Segment Browsers**

You can have multiple Segment Browser panels, one for each open document. To save screen space, we recommend that you place all Segment Browser panels on top of one another. SE automatically pops the panel for a viewer to the top of the stack of panels.

As mentioned, each Segment Browser panel corresponds to a particular document. If, as a result of haphazard window operations, the current document does not correspond to the current Segment Browser panel, SE will warn you when you try to create or select a segment. If this happens, rearrange your windows to ensure that the viewer and the Segment Browser panel refer to the same document.

### **The Document Inspector**

The Document Inspector is a utility inside SE that shows you open signal documents. Note that the Document Inspector does not show a list of document viewers. All viewers for an open **sig** file may be closed but the document is still open, available for editing, and thus listed in the Document Inspector. (To see a list of view windows, look under the Windows menu.)

The Document Inspector highlights the name and shows an icon corresponding to the currently selected document. You can click on this icon and move it to an empty viewer window, where SE will display the corresponding document.

The "Path" field in the top half of the display shows the physical path name of the open document. That is, it resolves the actual name any symbolic links in the path name. The "Host" field tells you what host file system the document resides on. This is useful information if you are working in a networked computer environment.

### **More About Signal Documents**



SE edits signal documents. If you look at your signal document directory with a Unix shell you'll see a number of files associated with each named signal document. This is because an SE signal document is actually a NeXT *file package*, containing a number of files that, together, constitute the document. A signal document includes the following files:

**editlist**--A list of pointers that specify the current editing state of the document and the change stack

**header**--Basic information about the signal document such as the number of channels, the sampling rate, and so on

**segments**--A list of pointers to views that you can go to or select

**c1**--A directory containing the actual sound samples and the *outline files* (reduced sample rate representation) used by SE in displaying the document.

## The Scheme Interpreter

Within SE you can invoke an interpreter for the Scheme programming language--a dialect of Lisp. This particular implementation of Scheme is called Elk (Laumann 1990). The Scheme interpreter allows access to signal documents and the change stack. In this preliminary release this interface is not documented. But you can use the Scheme interpreter for miscellaneous calculations, if you wish. To learn about Scheme, see Rees and Clinger (1986), Abelson, Sussman, and Sussman (1985).

To change the appearance of the text in the Scheme window, use the operations under the Format menu.

## Credits

The IRCAM Signal Editor was designed and implemented by Peter Wyngaard and Gerhard Eckel. This manual was written by Curtis Roads and Peter Wyngaard, with assistance from Gerhard Eckel.

## References

Abelson, H., G. Sussman, and J. Sussman. 1985. *Structure and Interpretation of Computer Programs*. Cambridge, Massachusetts: The MIT Press.

Eckel, G. 1990. "A signal editor for the IRCAM musical workstation." In S. Arnold and G. Hair, eds. 1990. *Proceedings of the 1990 International Computer Music Conference*. San Francisco: International Computer Music Association.

Laumann, O. 1990. "Reference Manual for the Elk Language Interpreter." Included in the software distribution from the Technische Universität Berlin, email address [net@tub.cs.tu-berlin.de](mailto:net@tub.cs.tu-berlin.de)

Rees, J., and W. Clinger. 1986. "Revised Report on the Algorithmic Language Scheme." A. I. Memo 848a. Cambridge, Massachusetts: M.I.T. Artificial Intelligence Laboratory.

\*       \*

\*