WAVE User's Guide

Third Edition
(revised and with corrections for WAVE version 6.1)
7 May 1997

George B. Moody
Harvard-MIT Division of Health Sciences and Technology
Biomedical Engineering Center

For information on obtaining the most recent version of WAVE, visit our web site (`http://ecg.mit.edu`), or write to:

> MIT-BIH Database Distribution
> Massachusetts Institute of Technology 77 Massachusetts Avenue,
> Room 20A-113
> Cambridge, MA 02139
> USA

An HTML version of this guide is available; point your web browser to `http://ecg.mit.edu/wug/` to view it. Additional printed copies of this guide are available for US$10 each (including shipping by surface mail) by writing to the address above. Please make checks payable to "Beth Israel Hospital Biomedical Engineering Division."

Contents

List of Figures

Preface

Many areas of clinical practice and research share a common need for visualization and analysis of physiologic signals. Typically, signals such as electrocardiograms (ECG), respiration, blood pressure, electroencephalograms (EEG), electrooculograms (EOG), and electromyograms (EMG) may be acquired during a clinical procedure or experiment, for durations ranging from a few seconds to many hours. Often these signals must be monitored and analyzed in real time (i.e., while they are being acquired). In other cases, the signals can be recorded for later analysis.

There are two common reasons for recording signals for later analysis. First, the analysis may be too complex to perform in real time, or it may require observation of long time periods (for example, in Fourier spectral analysis of very low frequencies). Second, the analytic technique itself, as well as the signals, may be a subject of investigation. Digitally recorded signals are ideally suited as test material in such cases, since they can be used to provide strictly reproducible inputs to a variety of analysis methods.

WAVE is a computer program that helps you, the clinician or researcher, to analyze digitally recorded signals. Using WAVE, you can view any desired portion of your signals as if you were browsing through a chart recording. (WAVE can print a paper 'chart recording' of any portion of the signals, if you wish.) You can annotate (label) any features of the signals you choose. You can select any subset of the signals, and any time interval, to be analyzed by an external program under the control of WAVE. If the analysis program generates signal annotations, you can view and correct them. Starting with example programs and library functions provided in the DB Software Package, you can write your own programs for signal analysis, and add their capabilities to WAVE's repertoire, without recompiling (or even restarting) WAVE.

This guide describes how to use WAVE, how to extend its capabilities, and what hardware and software you will need in order to do so. It does not presume extensive familiarity with computers or signal processing, but it would be helpful to have a friend with some computer experience available when you begin.

Many friends have suggested improvements in WAVE. Thanks to Penny Ford-Carleton, Ted Clancy, Kevin Clark, Leon Glass, Scott Greenwald, Farzin Guilak, Jeff Hausdorff, Yuhei Ichimaru, David Israel, Franc Jager, Joe Mietus, Sheila Ryan, Kambiz Soroushian, Alessandro Taddei, and Andy Wieckiewicz, and to all of the early users of WAVE. I would especially like to thank Roger Mark for

his continuous support and encouragement of this project.

Your comments on this guide, and on WAVE, are welcome. Please send them to:

George B. Moody
MIT Room 20A-113
Cambridge, MA 02139
USA

(e-mail: `george@hstbme.mit.edu`)

Notes on the third edition

Long time users of WAVE will note that this edition has been greatly expanded over previous editions. In part, this reflects more complete coverage of the original material, but WAVE itself has also grown. If you are still using WAVE 5.0, pay attention to the marginal notes here and there indicating features of WAVE 6.0 that are not available in earlier versions. WAVE 6.0 is available for downloading from the Internet (see section C.1, page 87).

I have received many requests for a version of WAVE that runs on a PC. This is possible at last! Under the remarkably complete, and completely free, Linux operating system, WAVE now runs exactly as it does on SPARCstations. If you have wanted to use WAVE but have been limited to using a PC, please try out Linux (see section B.6, page 85, for details).

This guide is now available in HTML form as on-line help for WAVE (choose (User's Guide) in WAVE's Help Topics window), and it may also be read with any Web browser independently of WAVE (point your browser to `file:/usr-/local/help/wave/wug.htm` if you have installed WAVE on your system, or to `http://ecg.mit.edu/software/wave-doc/guide/wug/wug.htm` otherwise).

Once again, my sincere thanks to all of the users of WAVE, whose questions, comments, and suggestions have helped to make WAVE and this guide more useful for all of us.

GBM
Cambridge, Massachusetts
August, 1996

Chapter 1

Introducing WAVE

The rest of this guide will be much easier to follow if you read it while running WAVE, so that you can try out the features described. This chapter contains a five-minute exercise in which you will start WAVE, take a quick look around, and exit from it. The later chapters assume that you have been able to complete this exercise successfully – if you encounter problems here, get help before continuing.

WAVE is an X Window System client application that runs on Sun SPARCstations (under Solaris or SunOS), and on PCs running Linux. X clients communicate with an X server and a window manager, programs that handle display output, and keyboard and mouse input, on behalf of the X clients, in much the same way that the Macintosh and Microsoft Windows operating systems do for applications running on Macs and PCs. Unlike Macintosh and Windows applications, however, X clients need not necessarily run on the same computer as the X server. If your computer can run an X server, and it is connected to a network, you can use it to interact with X clients running on any other computers ("hosts") connected to the same network. Thus you do not need to have your own SPARCstation or Linux PC in order to use WAVE, since X server software is available for almost all currently manufactured computers (including Macintoshes and Windows PCs) and many older computers as well.

You may use X server software running on your own computer to interact with a copy of WAVE running on another networked computer (the WAVE host). (Since the window manager is also an X client, it can run on any computer in the network, although it is usually run on the same computer as the X server.) If your own computer is a SPARCstation or a PC running Linux, it can act as the WAVE host, and no network connection is required.

If you don't yet have a WAVE host, see Appendix B, page 81 for hardware recommendations. For information about installing WAVE, see Appendix C, page 87. If WAVE has already been installed on the WAVE host, proceed to the next section after filling in the worksheet on the next page (get help from an expert such as your system administrator if necessary).

1.1 Start-up worksheet for WAVE

In order to use WAVE, you must know how to log onto and off of your computer (and the WAVE host, if they are not the same computer). Answers to the questions on this worksheet may also be needed. If you need help, see the notes keyed to each question below, or consult your system administrator. Record your answers in the spaces provided for future reference.

1. What command (if any) must be run *on your computer* to start the X server and the window manager, and open a terminal window?	
2. What command (if any) must be run *on the* WAVE *host* to initialize the database environment (the `DB` and `DBCAL` environment variables)?	
3. If you have a one- or two-button mouse, how do you simulate a middle mouse button click?	
4. If you have a one-button mouse, how do you simulate a right mouse button click?	

Skip the remaining questions if your computer is also the WAVE *host.*

5. What is your computer's name?	
6. What is the name of the WAVE host?	
7. What command (if any) must be run *on your computer* to permit the WAVE host to create a window on your computer's display?	
8. What command must be run *on your computer* to log onto the WAVE host?	
9. What command (if any) must be run *on the* WAVE *host* to redirect its output to your computer's display?	

Notes:

1. Starting the X server. On some computers, the X server and the window manager are started automatically whenever you log in, and no command is needed. Otherwise, you must run a command to start the X server before you can run WAVE. This command may be '`startx`' or '`openwin`'; for details, check your X server manual (on a UNIX system, type '`man X`'). If you have a choice of window managers, use `olwm` or `olvwm` if possible (see Appendix B, page 81). If no terminal window is available after starting the X server, you can usually open one by moving the mouse pointer over the background (the *root window*) of the display, then clicking the left or right mouse button to open a menu. Depending on your system, the menu may include 'xterm', 'shell', 'cmdtool', 'terminal emulator', or something similar; choose any of these to open a terminal window.

2. **Initializing the environment.** The form of this command depends on what shell (command interpreter) you use on the WAVE host. To identify your shell, log onto the WAVE host and type '`echo $SHELL`'. If the response contains the characters '`csh`', you are using the C-shell (or a variant of it); in this case, use '`source /usr/local/bin/cshsetdb`' to initialize the environment. Otherwise, use '`. setdb`' to do so (don't omit the '`.`' in this case). Usually, the appropriate command is included in your '`.profile`' or '`.login`' script on the WAVE host, so that it need not be entered each time you log onto the WAVE host. See `setdb(1)` (type '`man setdb`' on the WAVE host) for further information.

3. **Simulating a middle button click.** On a two-button mouse, this is often done by clicking both buttons simultaneously; on a one-button mouse, this is usually performed by pressing and holding a keyboard key such as Shift while clicking the mouse button. Refer to your X server documentation.

4. **Simulating a right button click.** This is usually done by pressing and holding a keyboard key while clicking the mouse button. Refer to your X server documentation.

5. **Your computer's name.** If you don't know your computer's name, you may be able to discover it by typing the command '`hostname`' (on a UNIX system), or by logging in to the WAVE host and typing the command '`who am I`' (your computer's name should appear at the end of the output). If the WAVE host doesn't recognize your computer by name, use your computer's IP address (in the form `a.b.c.d`, where `a`, `b`, `c`, and `d` are decimal numbers between 0 and 255).

6. **The WAVE host's name.** If your computer doesn't recognize the WAVE host by name, use the WAVE host's IP address.

7. **Permitting access to your display.** This command is usually needed if your computer is running UNIX. For example, if the name of the WAVE host is `atlantic`, this command would be '`xhost +atlantic`'. If your computer is not running UNIX, there may not be any command required; if in doubt, see your X server manual.

8. **Logging onto the WAVE host.** This is probably a command of the form '`telnet atlantic`'. You will probably be prompted to enter your password when you execute this command.

9. **Redirecting WAVE output to your display.** Assume that the name of your computer is `arctic`. If you use the C-shell on the WAVE host (see note 2 above), the command would be '`setenv DISPLAY arctic:0`'. Otherwise, it would be '`DISPLAY=arctic:0; export DISPLAY`'. It is possible to set up your `.profile` or `.login` script so that this command can be executed automatically each time you log onto the WAVE host. Consult your system administrator for details.

1.2 A quick look at WAVE

To begin this exercise, log onto your computer and do whatever you usually do to start the X server, the window manager, and a terminal window (item 1 from the worksheet).

About window managers

Some window managers require you to place and size windows manually, so the terminal window may not appear immediately. Usually such window managers show an outline of the window in place of the mouse pointer; typically, you click the left mouse button after dragging the window outline to the desired location on your screen in order to make the window appear.

Before going any further, find out how to move windows on your screen if you don't already know how to do so. (You will need to do this occasionally while using WAVE, since its windows may sometimes overlap.) Different window managers have different methods for moving windows; a method that usually works is to press either the left or the center mouse button while pointing to the center section of the window's title bar (at the top edge of the window), and then to move the pointer (which may have been replaced by the window outline) to the desired location before releasing the mouse button. (This common action is called *dragging*, as in the expression 'drag the window using the left button'.) Try moving the terminal window now.

For this exercise, all commands should be typed into the same terminal window. Once again, however, differences in window managers may affect how you do this. Some window managers use a *focus-follows-mouse* policy: you can type into a window whenever the mouse pointer is in the window. Others use a *click-to-type* policy: you must click the left mouse button within a window before you can type into it, but the mouse pointer need not remain in the window while you type (this policy will be familiar to Macintosh and Microsoft Windows users). If you use `olwm` or `olvwm` as your window manager, as recommended, you may choose the policy you prefer using the Properties menu (click right on the background or root window, and select Properties from the pop-up menu that appears).

If you have previous experience with X Window System and Open Look user interface applications, much of what you see will be familiar. This exercise does not assume that you have any such experience, but you may find it helpful to read through an introductory book on the Open Look user interface such as Sun's *OpenWindows User's Guide*, or a more general introduction such as O'Reilly's *X Window System in a Nutshell.* Please note that, although WAVE uses many Open Look controls, it is not fully Open Look compliant; in particular, actions within its signal window (see chapter 2) do not comply with Open Look style guidelines.

If you need to run WAVE remotely

If your computer is also the WAVE host, skip ahead to the next paragraph. Otherwise, type the command(s) from items 7, 8, and 9 of the worksheet, in that order (entering your password if prompted to do so by the WAVE host). These commands are the only differences in the procedures for running WAVE remotely (with different desktop and WAVE host systems) and locally (with your own computer serving as the WAVE host).

Checking the environment and the sample record

In common with all programs built using the DB library, WAVE finds its input files in any of the directories specified by the `DB` environment variable. (See the discussion of the database path in the *ECG Database Programmer's Guide* for details.) Type "`echo $DB`" to check the value of the `DB` variable. If the response is

```
DB: Undefined variable.
```

type the command needed to initialize the database environment (item 2 from the worksheet).

In this exercise, you will use WAVE to examine record `100s`, a sample record containing one minute of annotated two-channel ECG. Record `100s` is distributed with all versions of WAVE and should be available on any WAVE host. Check that it is available by typing

```
rdsamp -r 100s -f 59.99; rdann -r 100s -a atr -f 59
```

The output should be

```
   21596       2       1
   21597      -2      -5
   21598      -5      -6
   21599      -5      -5
      0:59.508    21423     N     0     0     0
```

If you do not obtain this output, see your system administrator.

Starting WAVE

WAVE is usually started by typing a command in a terminal window. Type:

```
wave
```

(Be sure to type this using lower-case letters.) If you see a response similar to '`wave:  Command not found.`', you will need to find the executable copy of WAVE on the WAVE host system. (It may be in a directory that is not part of your `PATH`; if so, you may wish to add that directory to your `PATH` by editing `.cshrc` or `.profile` in your home directory. If the previous sentence makes no sense to you, refer to any introductory book on UNIX for guidance.) If the response is '`wave: permission denied`', see your system administrator about obtaining permission to run WAVE.

If all goes as it should, WAVE prints a concise summary of its command format, which should appear (approximately) as:

```
WAVE version XXX; DB library version XXXXX  Copyright (C) MIT 1980-1997.
usage: wave -r record-name [ options ]

Options are:
 -a annotator-name  Open an annotation file
 -dpi XX[xYY]       Calibrate for XX [by YY] dots/inch
 -f TIME            Open the record beginning at TIME
 -g                 Use shades of grey only
 -H                 Use high-resolution mode
 -m                 Use black and white only
 -O                 Use overlay graphics
 -s SIGNAL [SIGNAL ...]  Initialize the signal list
 -S                 Use a shared colormap
 -Vx                Set initial display option x

wave is an X11 client.  You must specify the X server
connection for it in the DISPLAY environment variable.

Be sure to set the DB environment variable to
indicate a list of directories that contain
input files for wave.

For more information, type 'more /usr/local/help/wave/wave.hlp',
or open 'http://ecg.mit.edu/wug/wug.htm' using your web browser.
```

The comments about setting the `DISPLAY` and `DB` variables will not appear if these variables have been set (as they should have been earlier).

Now try using WAVE to view record `100s`, together with its `atr` (reference) annotations. Type:

```
wave -r 100s -a atr &
```

The '`&`' is optional but recommended; it allows you to type additional commands in the terminal emulator window without interrupting or exiting from WAVE. After a few seconds, WAVE's *main window* (figure 1.1) opens.

A Five-Minute Tour of WAVE

At the top of the window is the *title bar*, which reads 'Record 100s atr'. Below the title bar is WAVE's *main control panel*, which contains three groups of buttons. Several of these (File ∇, Edit ∇, and Properties ∇) are *menu buttons* (distinguished by the '∇' at the right end of the button), and are selectable using the right mouse button. The other buttons are selected using the left mouse button.

Below the main control panel is the *signal window*. This window shows a portion of the sample record `100s`, which contains two signals. Between the two signals, *annotations* are shown. The first annotation ('(N', at the left edge of the

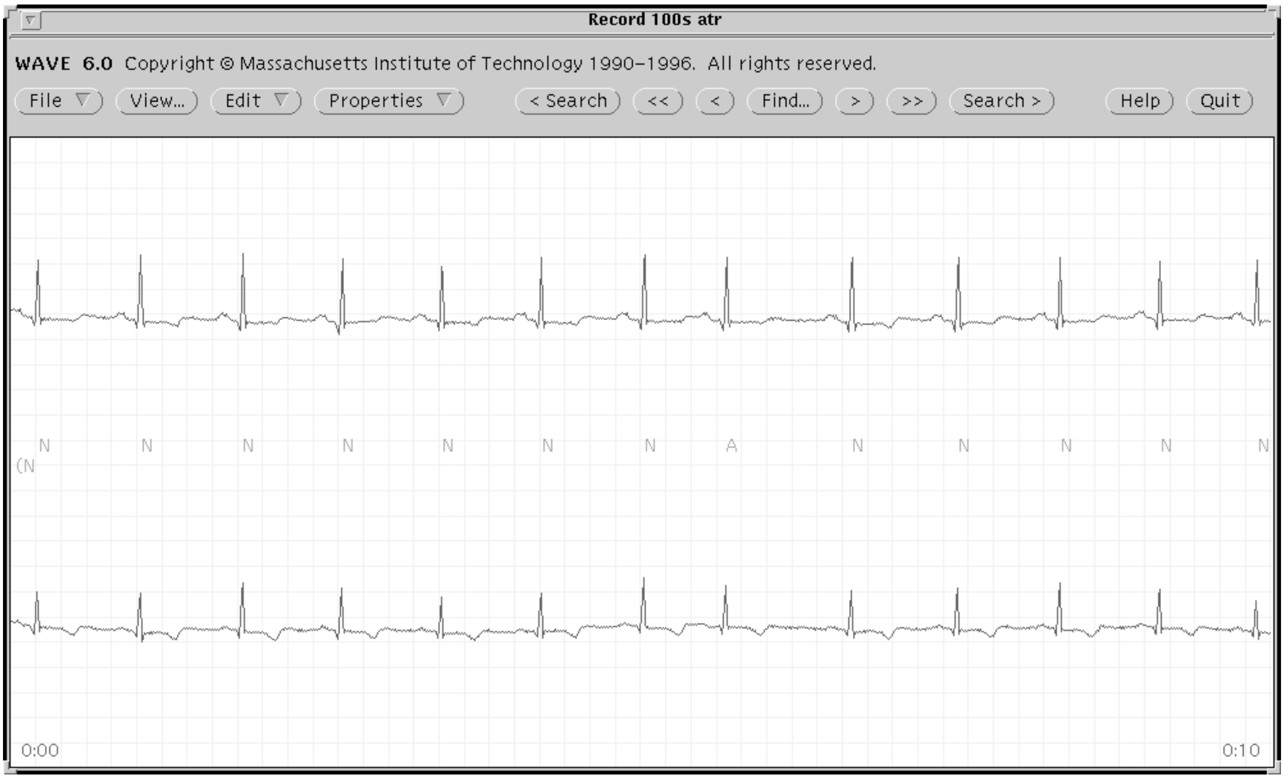


Figure 1.1: WAVE's main window.

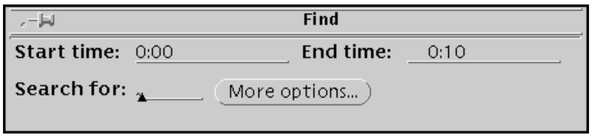


Figure 1.2: The Find window.

window) is shown below the level of the others, to indicate that it is a rhythm label ('(N' means normal sinus rhythm). Most of the other annotations are 'N', and indicate normal beats; the 'A' indicates an atrial premature beat. In the lower corners of the signal window, the *time indicators* ('0:00' and '0:10') show the elapsed time in minutes and seconds from the beginning of the record to the samples at the left and right edges of the window respectively. (If your display is less than about 260 mm wide, the signal window will not show a full 10 seconds of the record.) When you move the pointer into the signal window, it changes shape. Annotation editing and other operations are possible while the pointer is in the signal window; these operations are described in later exercises.

The buttons in the middle group ((< Search), (<<), (<), (Find...), (>), (>>), and (Search >)) are the controls that you use to navigate through the record. The (>>) button advances the signal window display by the width of the window (10 seconds in this case). Move the mouse pointer over (>>) and press the left button. (This common action is referred to below as 'clicking left on (>>)'.) The signal window is redrawn, and now shows the interval from 0:10 to 0:20. The (>) button also advances the window, but by only half the window width. Try it now. Similarly, (<<) moves the window back by one screenful, and (<) moves it back by half a screenful. Use these buttons to move to a window that begins at 0:55. Since the record is only 1 minute long, the right half of the window is empty. Notice that you can continue to advance past the end of the record if you wish. (WAVE will not allow you to back up past the beginning of the record, however.)

Buttons such as (Find...), with labels that end with '...', make WAVE open other (so-called 'pop-up') windows. Click left on (Find...) now. The Find window (figure 1.2) will appear. Its location on your screen is controlled by your window manager; typically it will appear either at the upper-left corner of the screen, or directly beneath the pointer (obscuring part of the signal window). If you wish, move the Find window to another location on your screen.

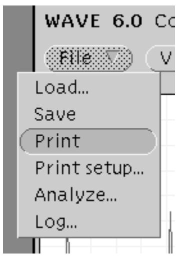
Within the Find window are three text fields in which you may type. Move the pointer into the the part of the window below the title bar. A black, upward-pointing triangle (the text cursor or *insertion point*) should appear in one of the three fields. (Once again, your window manager may influence what you see. If you see a grey diamond rather than a black triangle, you must perform whatever action your window manager requires to give the *keyboard focus* to the Find window. Usually, clicking left is sufficient.)

The Start time and End time fields match the times shown in the lower corners of the signal window. If you change either of these fields, the signal window moves accordingly. To change a text field, move the pointer to the right of the existing text and click left, press Backspace or DEL to erase any characters you wish to change, and type in the desired value, finishing by pressing Enter (or Return). (Always remember to press Enter after changing a text field; your changes are not registered until you press Enter.) Try changing the Start time and End time fields now, and watch how the contents of the signal window change. You can type any desired time into these fields; specifically, it need not be a multiple of 5 seconds.

Using any of the controls you have seen so far, set the signal window so that it shows the segment of the record beginning at 0:22. Now select the Search for field, and enter 'A' (followed, as always, by Enter). This action asks WAVE to find the next occurrence of an 'A' (atrial premature beat) label, and to redraw the signal window roughly centered on that label. As you may have noticed, the only 'A' annotation in record 100s occurs at about 0:06, so WAVE is unable to satisfy this request. Whenever WAVE cannot do what you have asked, it displays a *notice box*, with a brief message describing what has happened. (In this case, the message is 'No match found!', in other words, there are no 'A' labels between 0:32 and the end of the record.) Notice boxes are usually accompanied by a beep (although your window manager may allow you to disable the beep). Notice boxes always contain at least one button (in this case, labelled Continue). When a notice box is displayed, you *must* click left on one of the buttons in the notice box before you can do anything else in WAVE. Click left on Continue now to dismiss the notice box.

In WAVE *5.0, notice boxes perform a so-called 'full screen grab', which prevents mouse or keyboard events from being delivered to any other window, including those that belong to other applications.*

The < Search and Search > buttons on WAVE's main control panel are used to search for annotations that match the Search for field in the Find window. Searching forward using Search > will be unsuccessful (we know this, because WAVE has already tried to do so after we changed the contents of the Search for field). Use < Search button to search backwards. The signal window now shows the segment from 0:01 to 0:11, with the 'A' annotation at 0:06 roughly centered in the window. If you use either < Search or Search > now, the search fails, because the only 'A' annotation in the record is already on-screen. For more information, see section 2.12, page 21.



If a PostScript printer is available, try printing the contents of the signal window. To do this, press and hold the right mouse button while the pointer is above File ∇; then drag the pointer downwards until the Print selection is highlighted, and release the right mouse button. Printing will require between 20 seconds (if you have a SPARCprinter) and several minutes (if you have a typical PostScript printer); it is not necessary to wait for the output to appear before continuing. The output will appear as in figure 1.3.

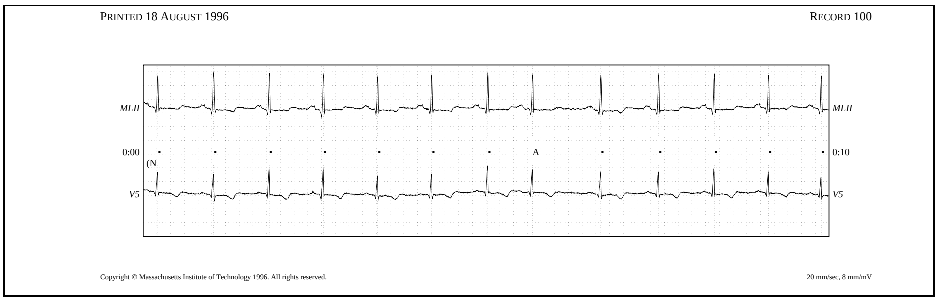


Figure 1.3: Printed “chart recording” made using Print from the File menu.

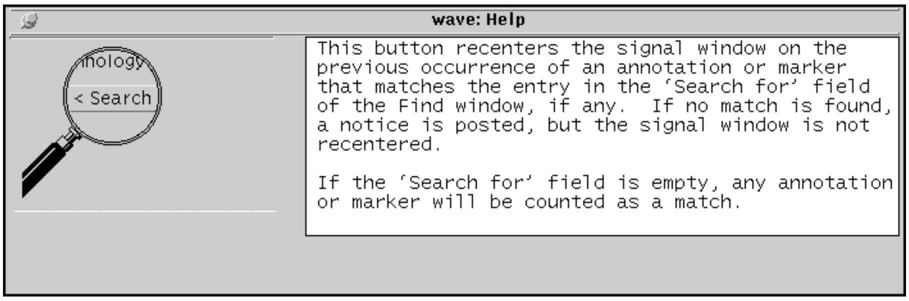


Figure 1.4: A spot help window.

WAVE has three types of on-line help. The first type, called *spot help*, works properly only if you are using an Open Look window manager (`olwm` or `olvwm`), although a local expert may be able to show you how to use it with another window manager. To use spot help with an Open Look window manager, move the pointer to any control or display area in a WAVE window and press the HELP key (or the F1 key if your keyboard does not have a HELP key). A *spot help window* (see figure 1.4) appears; it contains a magnifying glass icon showing the area you have selected, and a description of the control or display.

To get an overview of how to use WAVE, or if you cannot use spot help, click left on (Help) in WAVE's main control panel. The Help Topics window (see figure 1.5) appears, from which you may select any of the available topics by

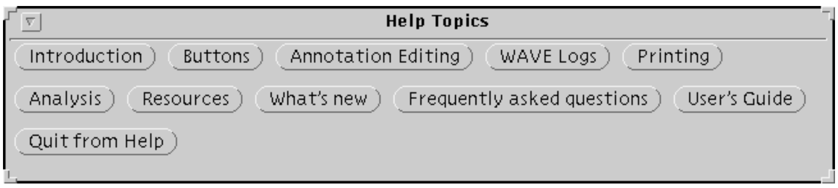


Figure 1.5: The Help Topics window.

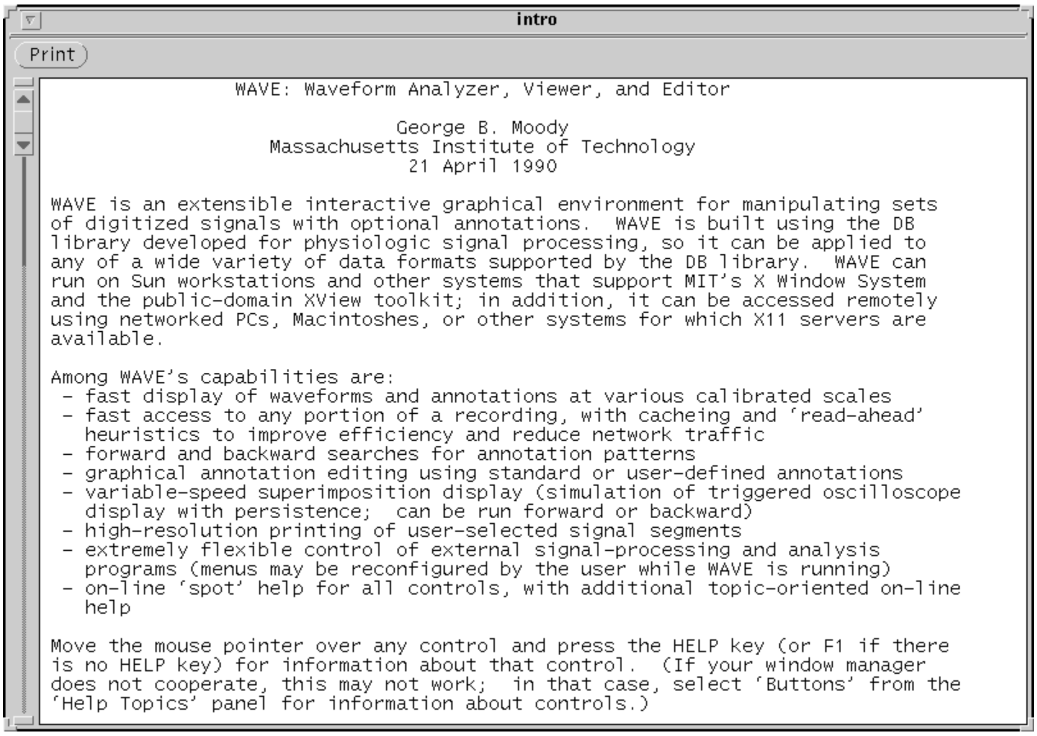


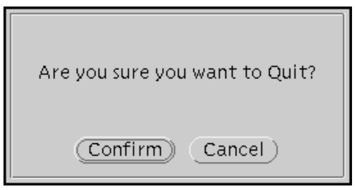
Figure 1.6: A help text window.

clicking left on the corresponding button. If you do so, a scrollable *help text window* (see figure 1.6) appears, containing information on the selected topic. Scroll up and down through the window by clicking left on the black triangles in the scroll bar (or use any other Open Look methods if you are familiar with them). If you wish, you can click left on the (Print) button at the top of the window to obtain a paper copy. Dismiss the window by clicking right on the *window menu button* (the square button containing a triangle at the upper left corner of the window), and then clicking left on the Quit item in the window menu.

These (Print) *buttons are not available in* WAVE *5.0.*

The third (and most comprehensive) form of on-line help can be used if a suitable web browser is available on the WAVE host system. (WAVE uses Netscape 1.1 or any later version by default. See section 3.10, page 43, for details on obtaining Netscape, or on configuring WAVE to use a different web browser.) Click left on the (WAVE User's Guide) button in the Help Topics window to open this manual using your web browser. If your browser is not running already, this may take a few moments while WAVE starts it up.

To complete this exercise, exit from WAVE by clicking left on (Quit) in WAVE's main control panel. A notice box appears as shown



at right; click left on (Confirm) to exit, or on (Cancel) if you're having so much fun that you don't want to stop yet. Once you have exited from WAVE, log off of the WAVE host (and your own computer, if you have been running a remote copy of WAVE).

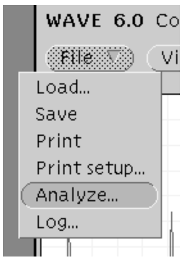
Chapter 2

Annotation Editing

There are five editing operations that you can perform using WAVE: changing, moving, copying, inserting, and deleting annotations. In this chapter, you will create and edit an annotation file using WAVE and an external QRS detection program. Start as in the previous chapter, but use the command

```
wave -r 100s &
```

omitting the '`-a atr`' option this time. Note that the title bar contains only the record name.



Click right on (File ∇), and click left on the Analyze... selection in the File menu. After a short delay, the Analyze window (see figure 2.1) and the Analysis Commands window (see figure 2.2) appear. At any time, you can click on (Show command window) in the Analyze window to make the Analysis Commands window visible if it becomes covered by another window.

Below the title bar of the Analyze window, text fields allow you to specify the beginning and end of a segment of the record, and to choose signals. The Analyze window also contains buttons specifying various actions that you can apply to the signals. The separate Analysis Commands

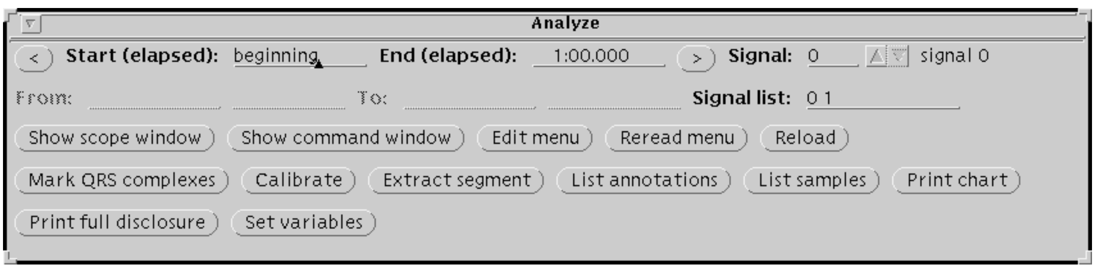


Figure 2.1: The Analyze window.

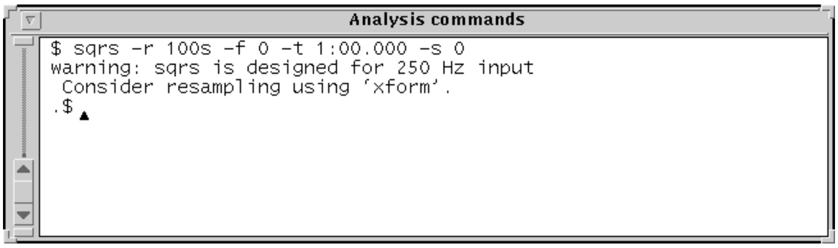


Figure 2.2: The Analysis Commands window.

window behaves like an OpenWindows `cmdtool` window (a scrollable terminal emulator). Commands may be run in the Analysis Commands window either by clicking left on action buttons in the Analyze window, or by typing directly into the window (depending on your window manager's policy for moving the keyboard focus, you may need to click left within the window before you can type into it, however). The Analysis Commands window may be resized if desired, and standard Open Look user interface methods may be used to cut and paste text between it and other windows, including those in other applications.

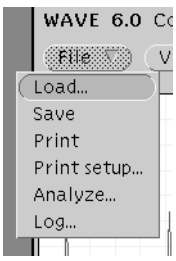
To create an annotation file, click left on (Mark QRS complexes). The command

```
sqrs -r 100s -f 0 -t 1:00.000 -s 0
```

appears in the Analysis Commands window, as shown in figure 2.2. Below the command, its error output appears. When the command has run to completion (which will take only a few seconds unless the WAVE host is heavily loaded), an annotation file named '`qrs.100s`' is written into the current directory.

If you wish, the Analyze window or the Analysis Commands window, or both, may be dismissed (by selecting Quit from the window menu to the left of the title bar); this can be done at any time, even while analysis is still in progress. If you recall the Analysis Commands window later (using (Show command window) in the Analyze window), it will show any output sent to it while it was dismissed. Note that the Analysis Commands window is dismissed whenever you exit from WAVE, and its contents are not retained after that point.

2.1 Loading annotations to be edited



To load annotations from '`qrs.100s`' into WAVE, click right on (File ▽), and click left on the Load... selection in the File menu. The Load window (see figure 2.3) appears. It contains text items that specify a record and and an annotator name. Click left next to the Annotator field, type '`qrs`', and, as always, press [Enter]. The contents of '`qrs.100s`' are loaded into WAVE, and annotations appear in the signal window. The title bar changes to indicate the annotator name. The Load window

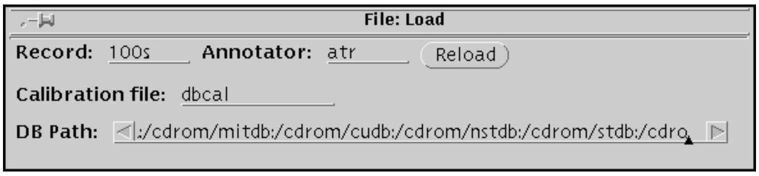


Figure 2.3: The Load window.

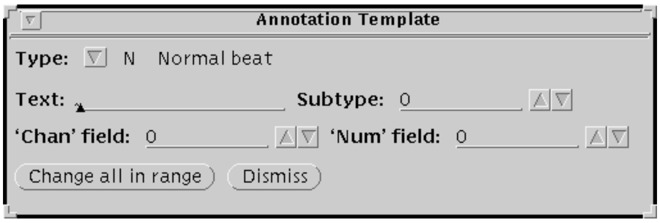


Figure 2.4: The Annotation Template window.

also contains items that specify the name of the calibration file and the database path (see section 1.2, page 5); you will not need to change these items for this exercise. You may dismiss the Load window or leave it up as you wish.

If you have looked through the record using the navigation buttons, use them now to return to the beginning of the record. The time indicator in the lower left corner of the signal window should read '0:00'. Since the (Mark QRS complexes) button only produces 'N' (normal) beat labels, the eighth beat in the window (an atrial premature beat) is mislabelled. In the next part of this exercise, we will correct this label.

2.2 The Annotation Template

Move the pointer into the signal window and click left. WAVE's Annotation Template window appears (see figure 2.4). The contents of the Annotation Template window indicate what type of annotation will be inserted if you decide to insert an annotation. Click right on the Type: $\boxed{\nabla}$ abbreviated menu button. The Type menu appears (see figure 2.5). Select 'A Atrial premature beat' from the Type menu by clicking left on it. This action dismisses the Type menu, and your selection now appears in the Annotation Template window to the right of the Type: $\boxed{\nabla}$ menu button. The other fields of the Annotation Template can be ignored for this exercise.

Single-character annotation mnemonics that appear in the Annotation Template's Type menu can be used as shortcuts during annotation editing. (These mnemonics are the same as those used by WAVE to indicate annotations in the signal window.) Instead of calling up the Type menu, you may simply type the mnemonic at any time while the pointer is in the signal window, provided that

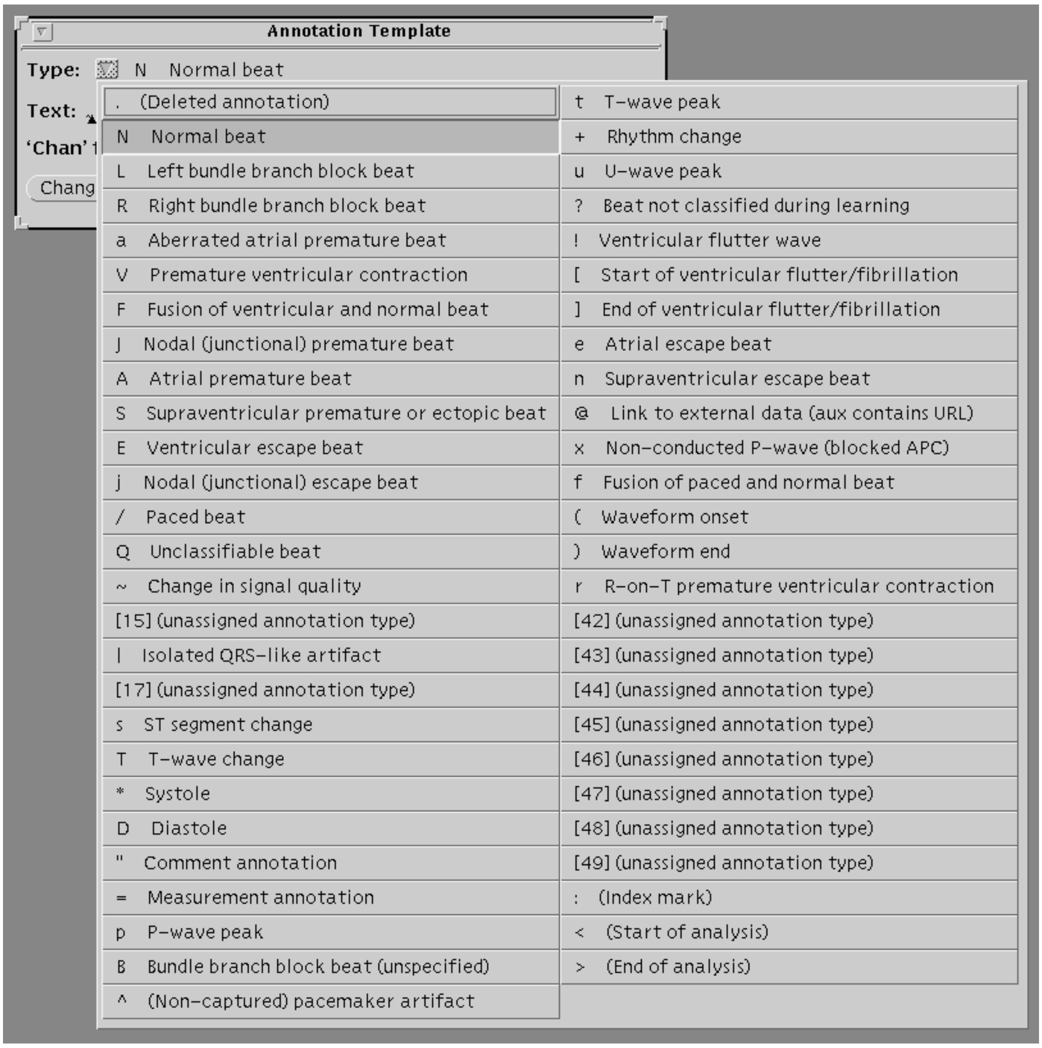


Figure 2.5: The Type menu.

the Annotation Template window is open. If you type a recognized mnemonic, the Annotation Template is updated immediately to reflect your choice.

Annotation editing is disabled by default, to avoid inadvertent modification of annotation files. If you now move the pointer into the signal window and click the middle button, this action is understood to mean that you wish to insert an annotation of the type shown in the Annotation Template at the location indicated by the pointer. Since WAVE cannot satisfy this request until annotation editing has been enabled, a notice box (shown above) appears to warn you that no action has been taken. Click left on (Continue) to dismiss the notice box.

To enable annotation editing, click right on (Edit ▽), and click left on the Allow editing selection in the Edit menu. (Your selection from the Edit menu persists for the duration of your WAVE session, unless you return to the Edit menu to change it. Thus it is sufficient to enable annotation editing once if you plan to edit several records in a single session.)

2.3 Selecting an annotation

In this case, we need to change an existing 'N' annotation into an 'A' annotation. To do so, we must select the annotation to be changed. Annotations are selected by clicking left or right in the signal window. Whenever the pointer is in the signal window, clicking left selects the annotation to the left of the pointer, and clicking right selects the annotation to the right of the pointer. (If 'Num Lock' is off, and the pointer is within the signal window, you can use the [←] or [→] keys instead of the mouse buttons if you prefer.) The annotation is highlighted by a *selection rectangle* drawn around it, and the pointer jumps to the center of the rectangle (see figure 2.6). If clicking left or right would otherwise move the pointer out of the signal window, WAVE recenters the signal window on the selected annotation. While any of the mouse buttons is down, *marker bars* appear above and below the pointer to allow you to identify the pointer location with respect to the signals.

The arrow keys cannot be used in WAVE *5.0.*

It is often useful, especially while editing annotations, to display marker bars above and below *each* annotation. If you wish to do so, click left on (View...), then click left on the [markers] selection in the View window that appears (see figure 2.7), and finally click left on (Redraw) to dismiss the View window and redraw the signal window with annotation marker bars (see figure 2.6). The marker bars can be turned off by repeating this procedure.

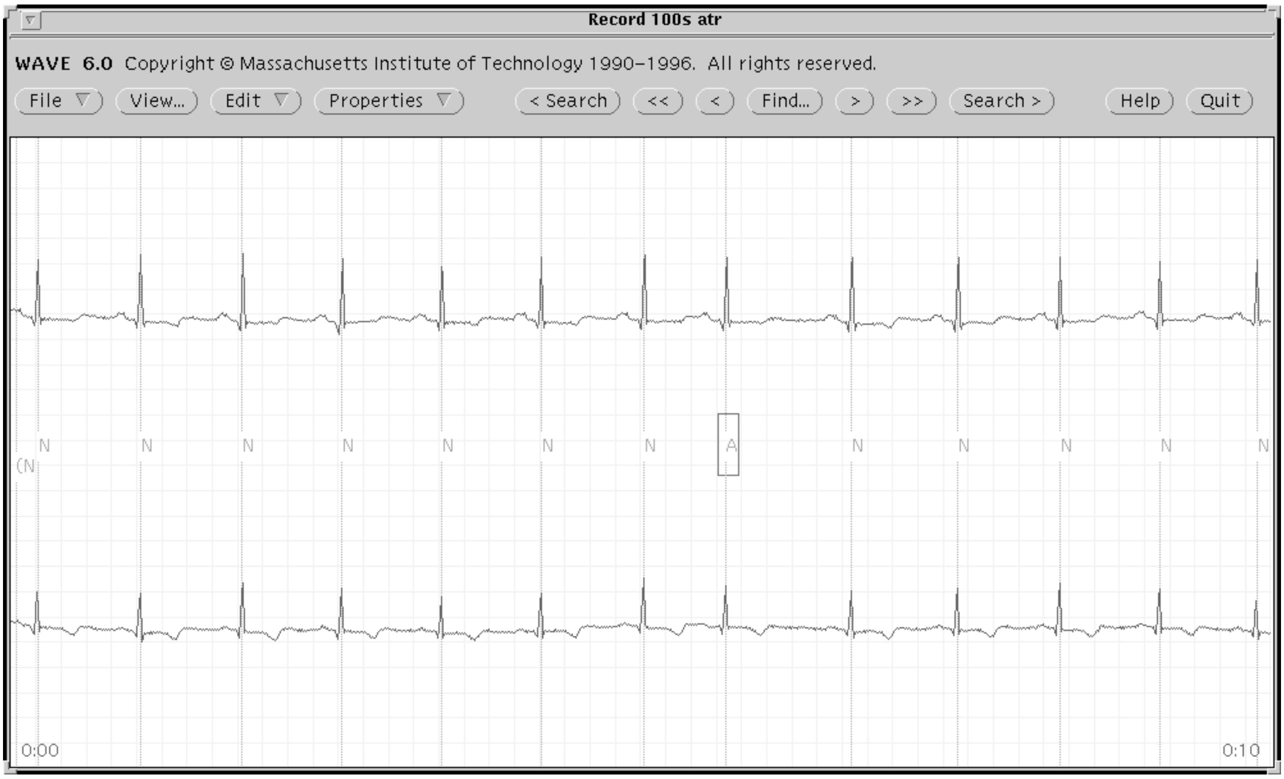


Figure 2.6: The main window, showing the selection rectangle and marker bars.

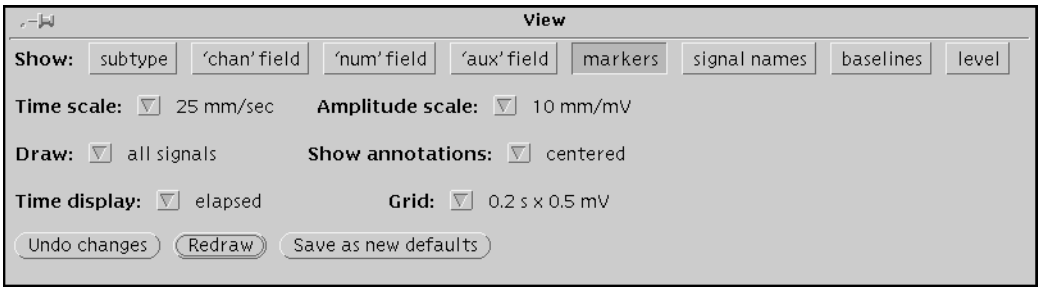


Figure 2.7: The View window.

2.4 Changing an annotation without moving it

Select the eighth beat label by clicking left or right, or using the arrow keys, until the selection rectangle surrounds it. Change the 'N' to an 'A' by clicking the middle button or pressing F2. (On some keyboards, if 'Num Lock' is off, you can use the 5 key on the numeric keypad instead of F2 if you prefer.) The 'N' is replaced by an 'A' and the selection rectangle disappears. To indicate that there are unsaved edits in the annotation file, the annotator name in WAVE's main window title bar is marked by parentheses.

F2 and 5 can't be used in WAVE *5.0.*

2.5 Moving an annotation

Although no further editing should be necessary, try out the other editing operations now. To move an annotation, select it with the left or right button, drag its marker bars (with the left or right button still depressed) to the desired location, then release the mouse button to drop the annotation. The selection rectangle stays in the original location until the button is released, but the marker bars move with the pointer once the pointer moves outside of the rectangle. To move an annotation by less than the width of the rectangle, simply drag the pointer above or below the rectangle.

If you prefer, you may use the keyboard rather than the mouse for this operation. To do so, drag the marker bars left or right using F3 or F4, and then drop the annotation using F2.

These keyboard commands are not available in WAVE *5.0.*

If you change your mind about moving an annotation, drag the pointer back into the selection rectangle, and then drop the annotation.

2.6 Inserting an annotation

To insert an annotation, move the pointer into the signal window (but outside of the selection rectangle if it is visible), press the middle button, drag the marker bars to the desired location, and release the middle button to complete the insertion.

2.7 Copying an annotation

This feature is not available in WAVE *5.0.*

To copy an annotation, select it, then press the Copy key (or F6 if your keyboard does not have a Copy key). This action copies the selected annotation into the Annotation Template. Now insert the copy at the desired location. (This is a particularly useful shortcut if you are working with complex annotations for which fields other than the Type must be set.)

2.8 Deleting and restoring annotations

To delete an annotation, first click right on the Annotation Template window's Type: [▽] menu button, and select '. (deleted annotation)', the first entry. Now select the annotation to be deleted by clicking left or right in the signal window, and click the middle button to delete it. The annotation is replaced by a '.' marker, which remains in WAVE's annotation buffer until a new annotation file is read or until you exit from WAVE. This feature allows you to locate deleted annotations if necessary (you may search for them by entering '.' in the Search for field of the Find window; see section 2.12, page 21). To restore a deleted annotation, simply repeat the operations that you used to delete it.

2.9 Markers

The '.' that appears when you delete an annotation is an example of a *marker*.

The important difference between *markers* and *annotations* is that markers are never written to annotation files; they remain in WAVE's annotation buffer only until WAVE reads another annotation file, or until you exit from WAVE, whichever comes first. Other types of markers are ':' (index marks, which you may insert anywhere in a record, and which you can return to by searching for them), '<' (beginning of region), and '>' (end of region). The '<' and '>' markers are special in that only one of each may be defined at any time; inserting a new one deletes the old one, if any. If the Analyze window is visible, you will notice that inserting or moving the '<' marker has the effect of changing the contents of the Start (elapsed) field (and the From field, if it is enabled); another way to insert a '<' marker is to enter the desired time directly into the Start (elapsed) field. The '>' marker is similarly coupled with the End (elapsed) field (and the To field) of the Analyze window.

2.10 Changing the Annotation Template

This feature is not available in WAVE *5.0.*

WAVE maintains a record of several of the most recently used settings in the Annotation Template (this information is not saved when you exit WAVE, however). You can cycle through these settings if you have a three-button mouse; to do so, press and hold the middle button as if to insert an annotation, then click left or right to change the contents of the Annotation Template. (If you don't wish to insert an annotation, release the middle button before releasing the left or right button.) This feature is particularly useful if you are manually annotating a record and you find it necessary to use a small number of different annotations to do so.

2.11 Changing many annotations at once

The usual method for changing annotations (clicking right or left to select the annotation to be changed, then clicking the middle button to perform the change) can become tedious if you have many annotations to change. WAVE offers a shortcut if you wish to change all annotations in a specified region in the same way. First, specify the region by inserting a '<' marker before the first annotation to be changed, and a '>' marker after the last annotation to be changed. Next, fill in the Annotation Template to indicate how you wish to change the annotations in the region. Finally, click left on (Change all in range). (If the region includes any annotations that are not visible on-screen, WAVE prompts you to confirm that you wish to change them.) Try out this technique now. Insert the '<' marker at 0:15, the '>' marker at 0:30, and change all of the annotations in the region to type 'V' (premature ventricular contraction). Inspect the results, then change them back to 'N' annotations.

2.12 Searching for annotations

WAVE can search in either direction for the next annotation that meets criteria you specify using the Search for field in the Find window. Any string that WAVE displays as an annotation label can be entered into the Search for field as a search criterion; these strings include annotation and marker mnemonics (usually single characters; see the Type menu for a list), signal quality strings (these contain one or more characters from the set "c", "n", and "u"; "U" is also a legal signal quality annotation), and *any substring* within a note, rhythm change, ST or T change, or link annotation. In addition to these, there are several Search for strings that WAVE interprets in special ways:

`*v`	matches any ventricular ectopic beat
`*s`	matches any supraventricular ectopic beat
`*n`	matches any other beat type
`*`	matches any annotation or marker
`.`	matches a deletion made during this WAVE session

More complex search criteria can be defined by clicking on (More Options...) in the Find window. This opens the Search Template window (figure 2.8).

The Search Template *is not available in* WAVE *5.0.*

In the Search Template, you may select any combination of annotation fields as search criteria using the abbreviated menu buttons to the left of each field. Enter the values to be matched next to each active match control. Any fields marked by 'Ignore' are not used as match criteria. In figure 2.8, the Type, Subtype, and 'Chan' field criteria must be satisfied by an annotation in order for WAVE to consider that annotation as a match; the Text and 'Num' field criteria are ignored by WAVE.

If the Text criterion is active, WAVE requires that the specified text string appear in the aux field of any matching annotation. Note that, as for search criteria specified in the Search for field of the Find window, the aux field of a matching annotation is required only to *contain* a matching text string, and

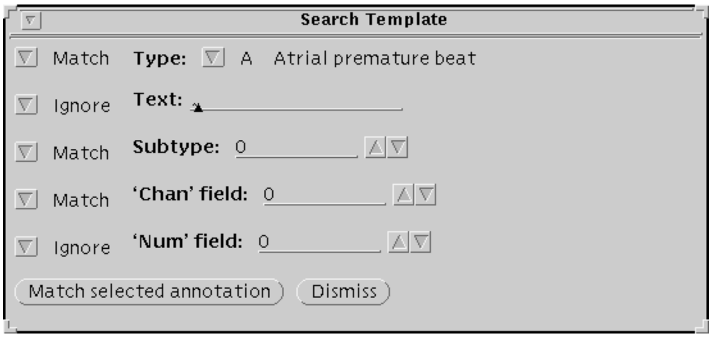


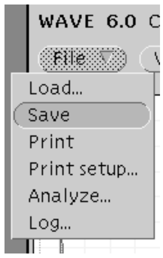
Figure 2.8: The Search Template window.

additional text may appear in the aux field before or after the matching string.

As a shortcut, click on (Match selected annotation) to copy all fields of the selected annotation into the Search Template. This action also sets all of the criteria menu buttons to Match.

2.13 Saving your work

In a lengthy editing session, WAVE saves your annotation file periodically. You can force WAVE to save your edits at any time by selecting the Save entry from the File menu. Once all of the changes have been saved, WAVE removes the parentheses from the title bar.



When you are finished, exit from WAVE as in the first exercise. It is not necessary to take any explicit action to save your edits, since WAVE does so automatically on any normal exit (via (Quit) on the main control panel, or from the window menu). You will find two new files in the current directory. The file called 'qrs.100s' is the edited version of the annotation file that you have just created, and '_qrs.100s' is the previous version (in this case, containing the unedited annotations as produced by sqrs). At most one previous version is saved as a backup; if the original annotation file is not located in the current directory (i.e., if it was found elsewhere in the database path), it is left unchanged and WAVE does not create a separate backup file.

Do not use Quit on the window menu to exit from WAVE 5.x and earlier; these versions will not save your edits before exiting this way.

2.14 Creating an annotation file manually

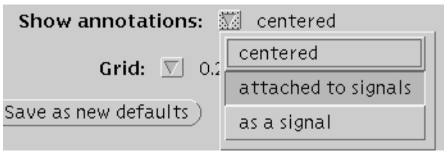
In this exercise, we created the annotation file using sqrs, and then edited it. It is also possible to create an annotation file manually; to do so, simply fill in the Annotator field in the Load window with the annotator name of your choice. If there is no existing annotation file associated with the specified annotator for the current record, WAVE creates an initially empty annotation file, to which you can add and edit annotations just as we did in this exercise.

2.15 Multi-edit mode

WAVE 5.0 does not support the features described in this section.

If you plan to use WAVE with records containing signals that must be annotated independently, read this section.

Using the Show annotations ▽ menu button in WAVE's View window, you may choose to display annotations attached to signals, rather than in the center of the signal window as is the usual default. In this case, the chan field of each annotation specifies the signal number of the signal to which it is attached. If annotation editing is enabled when annotations are displayed in this way, WAVE is in *multi-edit mode.*



WAVE draws the signals in order of signal number from the top to the bottom of the signal window, beginning with signal number 0 at the top of the signal window. When annotations are attached to signals, WAVE draws them about 2 mm above the center of the range of the attached signal (except for special annotations that are displaced above or below the usual level). Although this presentation usually helps to avoid the visual confusion that might result from drawing annotations directly on the associated signals, it may contribute to confusion if the spacing between signals is too small (as may happen if the signal window is reduced in height, or if many signals are displayed). If this becomes a problem, try increasing the height of the signal window (by dragging on the resize handles on the window frame), or displaying only a subset of the signals (by specifying which signals are to be shown in the Signal list field in WAVE's Analyze window, and selecting 'listed signals only' from the Draw: ▽ menu in WAVE's View window).

In multi-edit mode, annotation editing operations are slightly different from those described above. The most important difference is that you must always point to the desired signal when inserting or moving annotations. In this mode, the chan field in the Annotation Template window does not determine the chan field of an inserted annotation; rather, the signal to which you point determines the chan field, and the chan field in the Annotation Template window is updated accordingly after each insertion. To move an annotation to a different signal, simply select it and drag the pointer to the desired time and signal. If you wish to change the chan field of an annotation without changing its time, select the annotation and use the ↑ and ↓ keys to move it to the desired signal. Hold down Ctrl during these operations to copy the annotation, rather than to move it (simultaneous annotations are permitted if their chan fields differ).

2.16 About link annotations

The features described in this section are not supported by WAVE *5.0.*

A link (‘@’) annotation permits external data to be associated with a specific time (and, optionally, a specific signal) in a record. To make link annotations visually distinctive, WAVE displays them in the color used for signals rather than that used for other annotations, and it underlines them. The `aux` field of a link annotation contains a URL (uniform resource locator) that points to the external data, and an optional description displayed by WAVE at the location of the annotation. (If the description is missing, WAVE displays the URL name.)

To view the external data associated with a link annotation, select it and then press Return (or Enter). WAVE then directs your web browser to display the data specified by the URL. (If your web browser is not running, WAVE starts it.) By default, WAVE uses Netscape 1.1 or any later version available on the WAVE host. To configure WAVE to use a different browser, see section 3.10, page 43.

If the URL string does not contain a *protocol*`://` prefix (where *protocol* is typically `http` or `ftp`), WAVE assumes that the URL refers to a file located somewhere in your DB path. In this case, WAVE finds the file if possible, and attaches the necessary path information to the beginning of the URL before directing your web browser to display the associated data.

WAVE does not include built-in facilities for editing external data. You may insert and edit link annotations themselves, however, using the normal annotation editing facilities of WAVE. To create a link annotation, select ‘@’ from the Type menu of the Annotation Template, and enter the URL in the `aux` field. If you wish to supply a description to be displayed by WAVE in place of the URL, enter it after the URL, separating it from the URL with a space character. The current URL (i.e., the one associated with the most recently selected link annotation) can be passed by WAVE to an external program such as a web browser or an HTML editor via the `$URL` menu variable (see the comments in WAVE’s menu file for details).

2.17 Creating and using new annotation types

WAVE was originally developed for use with ECGs, and the standard set of annotation types reflects this history. If you work with other types of signals, or if the standard set of annotations does not meet your needs, you may define your own annotation types and add them to the Type: ▽ menu in the Annotation Template. To do so, first create a text file containing one-line entries defining each new type, as described in section 6.4, page 61, or see the file ‘`anntab`’ included in the WAVE distribution for an example. Next, set the environment variable `ANNTAB`, or the X11 resource `Wave.Anntab`, to the name of this file. The next time you run WAVE, the new types from your `ANNTAB` file will appear in the Type: ▽ menu. You may use them in the same way as the standard annotation types. Any annotation files you create or edit while your `ANNTAB` file is loaded will contain embedded copies of your type definitions, so that these files can be used on other systems and by other users without the need to reference your `ANNTAB` file each time.

In most cases, you should use the unassigned type codes between 42 and 49 for custom annotation types, rather than redefining the standard types. If you are not working with ECGs, however, you may redefine any type codes you wish (the legal range is defined in '`<ecg/ecgcodes.h>`').

Chapter 3

Analyzing Data with WAVE

For many users, WAVE's data analysis capabilities will be of greatest interest. WAVE itself does not perform analysis of the signals or annotations it displays. What WAVE offers is an easy-to-use means of controlling external analysis programs and viewing their outputs. The buttons in the Analyze window activate these external programs. These buttons are easy to create – in fact, you can add your own buttons to those in the Analyze window while WAVE is running.

In the previous chapter, we used the (Mark QRS complexes) button to run the standard DB application program `sqrs`. In this chapter, we will see how WAVE controls `sqrs`, and we will create and test a new button to generate a heart rate signal using `sqrs` and another standard DB application, `tach`. Discussions of WAVE's "logbook" and "oscilloscope" facilities follow, and then we study an extended example of how to develop an analysis program controlled by WAVE, using the C programming language. At the end of this chapter, we explore how this relationship can be inverted, so that WAVE can be driven by an external program.

As before, use WAVE to open record `100s`:

```
wave -r 100s
```

Click right on (File ∇), click left on Analyze..., and once again, click left on (Mark QRS complexes). Where does the command that appears in the Analysis Commands window,

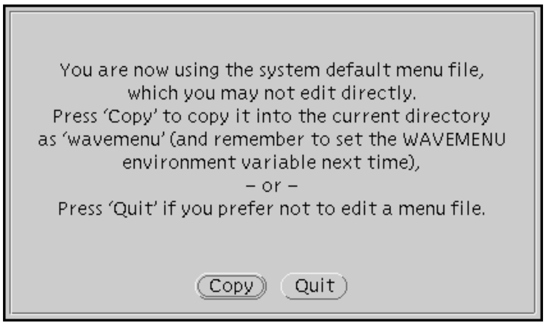
```
sqrs -r 100s -f 0 -t 1:00.000 -s 0
```

come from, and what does it mean?

If you refer to the `man` page for `sqrs`, `sqrs(1)` (either by typing '`man sqrs`' in a terminal emulator window, or by looking it up in the *ECG Database Applications Guide*), you will see that the second command-line argument (`100s`) is (not surprisingly) the record name, '`-f 0`' instructs `sqrs` to start at the beginning of the record, '`-t 1:00.000`' tells it to stop 1 minute later, and '`-s 0`' specifies that signal 0 is to be used for QRS detection.

3.1 WAVE's menu file

To understand where the command comes from, we need to inspect WAVE's *menu file*. Click left on the (Edit menu) button in the Analyze window. Unless your WAVE menu has already been customized, WAVE pops up a notice as shown at right. Click left on (Copy) to continue with this exercise. After a few seconds, the WAVE menu file appears (in an OpenWindows `textedit` window as in figure 3.1, unless you have specified a different editor by setting the `EDITOR` environment variable).

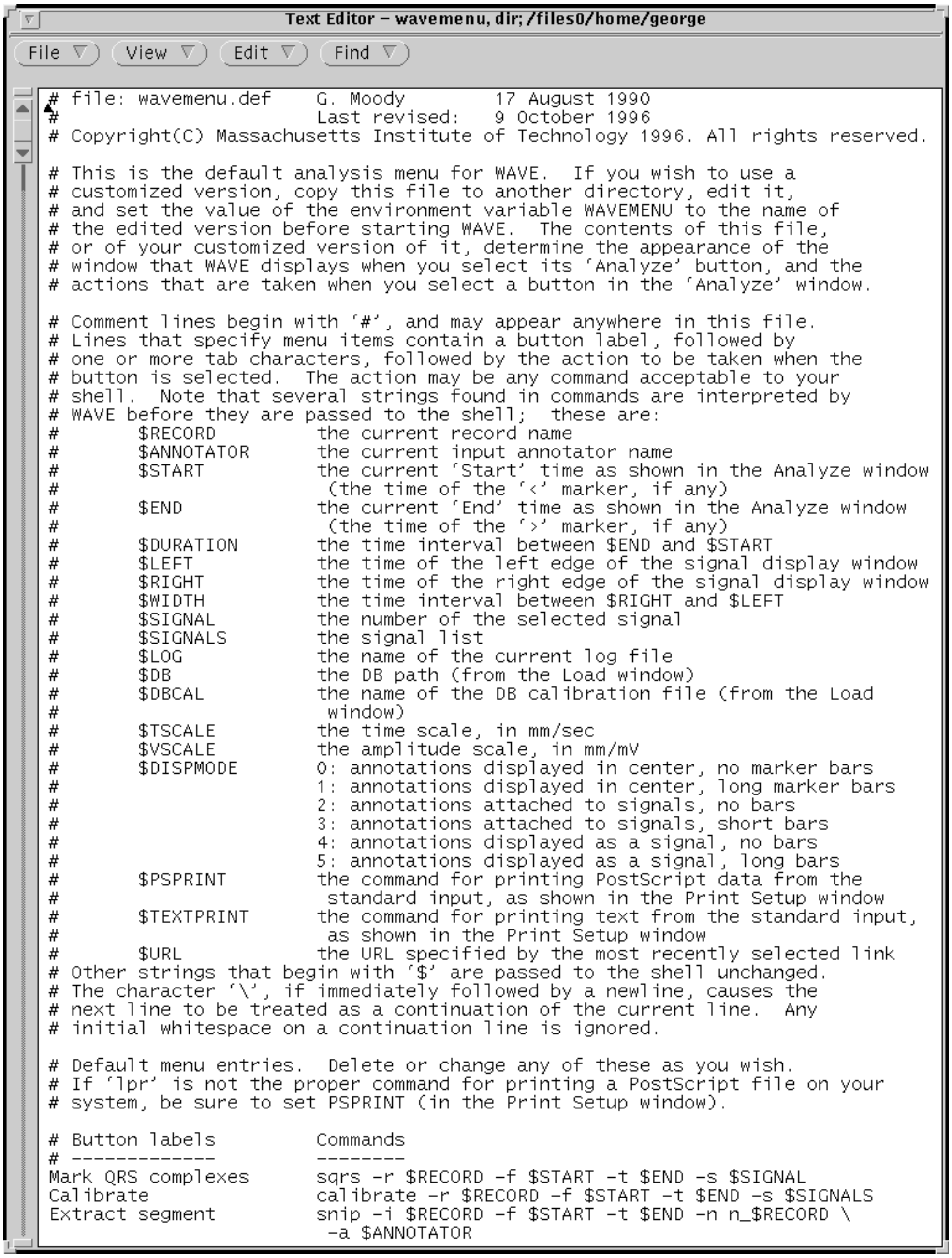


The comments at the beginning of the WAVE menu file describe how it works. Each of the buttons in the Analyze window (except for those in the first three rows) corresponds to an entry in the menu file. The first part of the entry – up to the first tab character – specifies the label that appears on the button; the remainder of the entry specifies the command that is inserted into the Analysis Commands window if you click left on the button. As noted in the comments, certain strings that begin with '`$`' are recognized as *menu variables* and are interpreted by WAVE before being passed to the shell in the Analysis Commands window. In the case of the `sqrs` command that appears in the Mark QRS complexes entry, these strings are '`$RECORD`', '`$START`', '`$END`', and '`$SIGNAL`'. WAVE replaces '`$RECORD`' with the record name, as it appears in the `Record` field of the Load window. '`$START`' is the time of the '`<`' marker, shown in the Start field of the Analyze window; if no '`<`' marker has been defined, the Start field contains the string 'beginning of record', and '`$START`' becomes `0`. Similarly, '`$END`' is the time of the '`>`' marker, shown in the End field; if no '`>`' marker has been defined, the End field is 'end of record', and '`$END`' is determined by the record length as encoded in the header file for the record (see `header(5)`, in the *ECG Database Applications Guide*). Finally, '`$SIGNAL`' is taken from the Signal field of the Analyze window; it specifies which signal is to be analyzed. (For applications that can analyze more than one signal, the string '`$SIGNALS`' is taken from the Signal list field.) Examine the commands defined in the default WAVE menu; they should give you an idea of what is possible.

The signal list does not exist in WAVE *5.0.*

3.2 Defining a region of interest

A common problem in many cases is simply selecting a segment to be analyzed, from a digitized signal file that may include extraneous material at the beginning or at the end. As you can see from the `sqrs` example, this can be done easily




```
# file: wavemenu.def    G. Moody        17 August 1990
#                       Last revised:   9 October 1996
# Copyright(C) Massachusetts Institute of Technology 1996. All rights reserved.

# This is the default analysis menu for WAVE.  If you wish to use a
# customized version, copy this file to another directory, edit it,
# and set the value of the environment variable WAVEMENU to the name of
# the edited version before starting WAVE.  The contents of this file,
# or of your customized version of it, determine the appearance of the
# window that WAVE displays when you select its 'Analyze' button, and the
# actions that are taken when you select a button in the 'Analyze' window.

# Comment lines begin with '#', and may appear anywhere in this file.
# Lines that specify menu items contain a button label, followed by
# one or more tab characters, followed by the action to be taken when the
# button is selected.  The action may be any command acceptable to your
# shell.  Note that several strings found in commands are interpreted by
# WAVE before they are passed to the shell;  these are:
#       $RECORD         the current record name
#       $ANNOTATOR      the current input annotator name
#       $START          the current 'Start' time as shown in the Analyze window
#                        (the time of the '<' marker, if any)
#       $END            the current 'End' time as shown in the Analyze window
#                        (the time of the '>' marker, if any)
#       $DURATION       the time interval between $END and $START
#       $LEFT           the time of the left edge of the signal display window
#       $RIGHT          the time of the right edge of the signal display window
#       $WIDTH          the time interval between $RIGHT and $LEFT
#       $SIGNAL         the number of the selected signal
#       $SIGNALS        the signal list
#       $LOG            the name of the current log file
#       $DB             the DB path (from the Load window)
#       $DBCAL          the name of the DB calibration file (from the Load
#                        window)
#       $TSCALE         the time scale, in mm/sec
#       $VSCALE         the amplitude scale, in mm/mV
#       $DISPMODE       0: annotations displayed in center, no marker bars
#                       1: annotations displayed in center, long marker bars
#                       2: annotations attached to signals, no bars
#                       3: annotations attached to signals, short bars
#                       4: annotations displayed as a signal, no bars
#                       5: annotations displayed as a signal, long bars
#       $PSPRINT        the command for printing PostScript data from the
#                        standard input, as shown in the Print Setup window
#       $TEXTPRINT      the command for printing text from the standard input,
#                        as shown in the Print Setup window
#       $URL            the URL specified by the most recently selected link
# Other strings that begin with '$' are passed to the shell unchanged.
# The character '\', if immediately followed by a newline, causes the
# next line to be treated as a continuation of the current line.  Any
# initial whitespace on a continuation line is ignored.

# Default menu entries.  Delete or change any of these as you wish.
# If 'lpr' is not the proper command for printing a PostScript file on your
# system, be sure to set PSPRINT (in the Print Setup window).

# Button labels         Commands
# -------------         --------
Mark QRS complexes      sqrs -r $RECORD -f $START -t $END -s $SIGNAL
Calibrate               calibrate -r $RECORD -f $START -t $END -s $SIGNALS
Extract segment         snip -i $RECORD -f $START -t $END -n n_$RECORD \
                         -a $ANNOTATOR
```

Figure 3.1: The beginning of WAVE's menu file, in a `textedit` window.

by inspecting the signals with WAVE and placing '<' and '>' markers at the beginning and end of the region of interest. If the analysis program is written to accept starting and ending times (more on this subject below), these times can be passed as command-line arguments by WAVE. Similarly, it's simple to choose one or more of a number of signals to analyze after viewing them with WAVE (either type into the Signal field directly, use the increment/decrement buttons, or point to a signal in the signal window, press and hold the Shift key, and click left). Close the `textedit` window and experiment with changing the Start, End, and Signal fields of the Analyze window, and running `sqrs`, until you are familiar with the results.

WAVE *5.0 does not support 'point-and-click' signal selection.*

3.3 Analysis example: Generating a heart rate signal

In the rest of this exercise, we'll develop a command for generating a heart rate signal, add it to the WAVE menu, and test it. First, set the Start and End fields to `0` and `1:0` respectively, and click on (Mark QRS complexes) to regenerate the annotation file. Open the annotation file using the Load window.

Generating a heart rate signal is simple enough, given a beat annotation file and `tach`, a program supplied as part of the DB Software Package. `tach` reads an annotation file, calculates an instantaneous heart rate for each cardiac cycle from the reciprocal of the interval between successive beat annotations, and then uniformily resamples this signal (by default, at 2 Hz). Referring to the *ECG Database Applications Guide*, we find the command we need to generate a heart rate signal from '`qrs.100s`':

```
tach -r 100s -a qrs -f 0 -t 1:0
```

If this command is typed into the Analysis Commands window, it prints:

```
73.9726
73.9726
73.7238
    .
    .
    .
75.7788
74.5703
73.8015
```

These numbers are the calculated samples of the heart rate signal (the units are beats per minute).

We can generalize this command as:

```
tach -r $RECORD -a $ANNOTATOR -f $START -t $END
```

where the parameters $RECORD, $ANNOTATOR, etc., are replaced by the appropriate values when the command is executed. This is the form of the command we should add to WAVE's menu; by parameterizing the arguments in this way, we can use WAVE to select a record, annotator, and segment to be analyzed as for the other commands in the menu. To add this command to the menu, click on (Edit menu) again. Position the cursor at the end of the menu file and add the line:

```
Heart rate<TAB>tach -r $RECORD -a $ANNOTATOR -f $START -t $END
```

where <TAB> represents one or more TAB characters; at least one TAB must be present (spaces don't count). Be sure to end the line with RETURN. Save the menu (choose Save from the `textedit` window's (File ▽) menu), and quit from `textedit`. Now click on (Reread menu) in the Analyze window. The Analyze window will disappear briefly, then reappear (possibly in a different location, depending on your window manager) with a new (Heart rate) button. If you click on this button, WAVE executes the `tach` command as above.

You might prefer to look at heart rate as a signal rather than as a column of numbers. `tach` has the capability of generating an output signal file instead of text. To try this out, edit the menu file again, and add '`-o hr_$RECORD`' to the end of the entry. (You may split the command across two lines using a '\' at the end of the first line if you wish.) Once again, save the menu file, quit from `textedit`, click on (Reread menu), and on (Heart rate). This time, there is no text output from `tach`; if you enter '`hr_100s`' in the Record field of the Load window, however, WAVE displays the calculated heart rate signal. (You may wish to change the time and amplitude scales using the View window, in order to see more of the signal at once. If you set the time scale to 250 mm/min and the amplitude scale to 100 mm/mV, you will be able to see the heart rate signal for the entire record as in figure 3.2; the periodic modulation is due to respiratory sinus arrhythmia.)

3.4 The signal list

Although this exercise used only one signal of the input record, many applications will use two or more signals simultaneously. The Signal list initially contains the signal numbers of all available signals; as noted above, the signal list can be passed to an application listed in the menu file as a set of command-line arguments in place of the string '`$SIGNALS`'. Although you can type into the Signal list field directly to change it, you can also do so using the mouse. Pointing to a signal, hold down the [Control] key while clicking left to add it to the signal list, or hold down the Meta key while clicking left to delete its first occurrence from the signal list. (Most keyboards do not have a key labelled Meta. On most Sun keyboards, the Meta keys look like this: [◇]; they flank the space bar. On most other keyboards, the [Alt] key is equivalent to Meta.)

These features are not available in WAVE *5.0.*

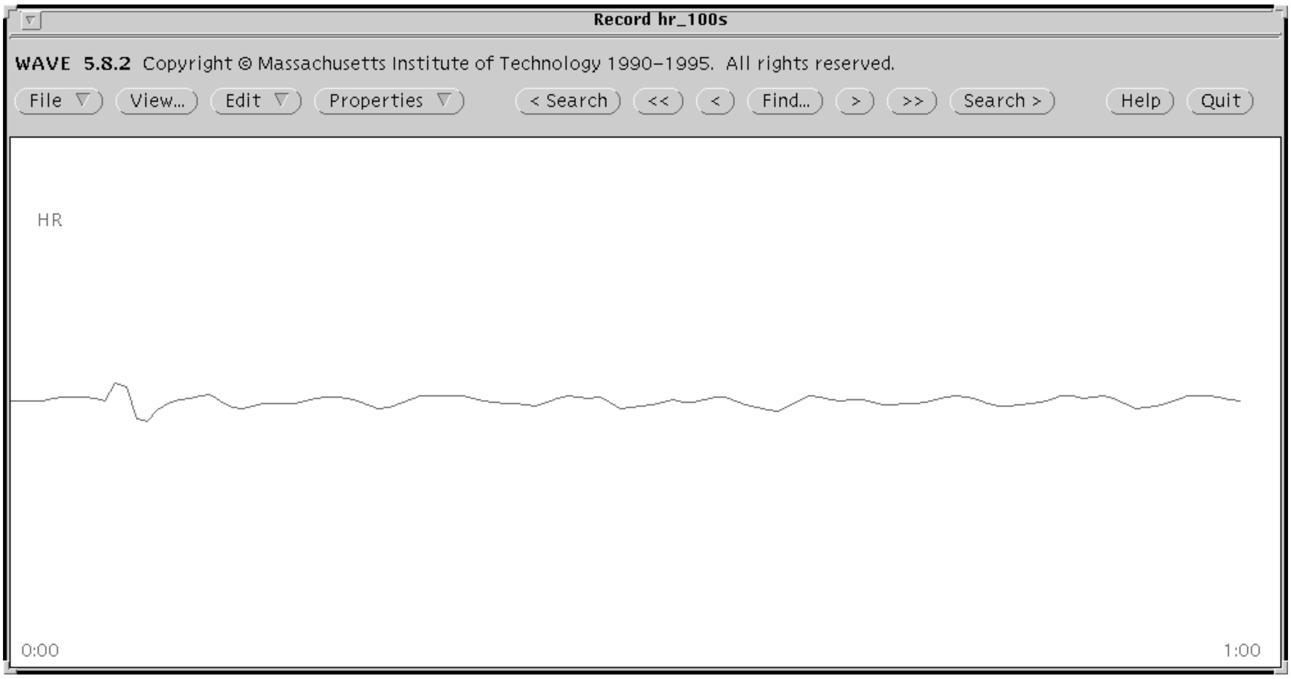


Figure 3.2: Heart rate signal.

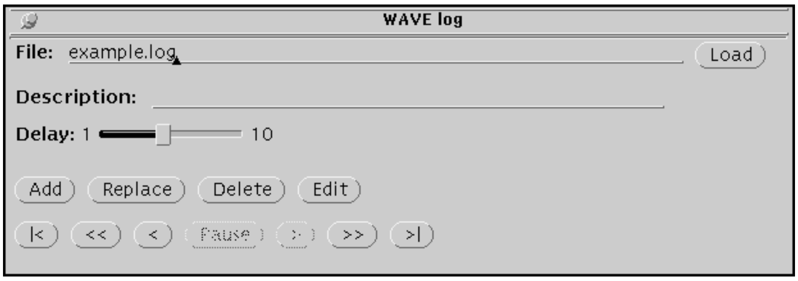


Figure 3.3: The Log window.

3.5 Using a customized menu file

Recall that when you first examined the menu file, WAVE warned you to set the `WAVEMENU` variable the next time you run WAVE. `WAVEMENU` should be set to the name of your customized menu file (you can give the menu file any name you choose; `wavemenu` is the default name). For example, if you use the C-shell and `wavemenu` is in your home directory, use the command

```
setenv WAVEMENU ~/wavemenu
```

before running WAVE again. You may wish to add this command to your `.login` script to set the variable automatically each time you log in. (If you forget to set `WAVEMENU`, WAVE will find your customized menu file only if it is in the current directory and is named `wavemenu`; otherwise, WAVE will use the system default menu again.)

WAVE *5.0 always uses the system default if* `WAVEMENU` *is not set.*

3.6 Using the Log window

When examining a recording, you may find features that you wish to document for further study. In addition to its capabilities for interactive annotation editing, WAVE also offers an easy-to-use "logbook" for collecting brief notes about segments to which you may wish to return. Each entry in a WAVE log file contains the record name, time, and an optional comment – so it is possible to put together a collection of examples taken from any number of records in a single log. Once the log has been created, the entries in it can be reviewed very easily, or the log can be used as a script for `pschart`, which will print the signals, annotations, and your comments for each entry, changing records automatically.

Open the Log window (see figure 3.3) by choosing Log... from WAVE's (File ∇) menu. When you first open the Log window, many of its controls are dimmed until you specify a log file and begin adding entries to it. To create a log file, or to review or add to an existing one, enter its name in the File: field of the Log window.

To add an entry, use WAVE's navigation controls so that the segment upon which you wish to comment appears in the signal window. Enter your comments in the Description: field, then click on (Add).

The log navigation buttons become visible once there are log entries recorded. You can use (<) and (>) to step through the log one entry at a time, or use (<<) or (>>) to present a "slide show" (control the length of time each entry appears using the Delay: slider, and use (Pause) to interrupt the show). (|<) and (>|) can be used to jump directly to the beginning or the end of the log.

To replace a log entry, first use the log navigation buttons to display that entry. Make any adjustments necessary (use the navigation buttons in WAVE's main window to reposition the signal window as needed, and correct the log entry text in the Description: field), then click on (Replace).

To delete a log entry, select it first as above, then click on (Delete). For more complex editing (such as rearranging entries), click on (Edit) to open the log file in a `textedit` window (or in the editor of your choice, if you have set the `EDITOR` environment variable). After completing your edits, save them and exit from the editor, then click on (Load) to reload the edited log into WAVE. (You can also use (Load) if another process, such as one started from the Analyze window, has modified the log file.)

This function does not exist in WAVE *5.0.*

WAVE's default menu file contains a commented-out entry with the tag "Print log with charts". If a button with this text does not appear in the Analyze window, click on (Edit menu), remove the comment characters from this entry, save the menu and exit the editor, and click on (Reread menu). You can now print the log together with the signals and annotations by clicking on (Print log with charts).

3.7 Using the Scope window

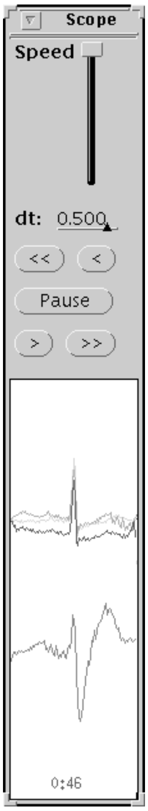


Figure 3.4: Scope

Some types of signals are characterized by changes in waveform shape (morphology), which may occur gradually or suddenly. For example, ischemic ST changes in the ECG are usually visible as quasi-continuous changes over a period of at least 30 seconds (often much longer). It can be helpful to view such signals on an oscilloscope, where the trigger can be adjusted so that the (approximately) periodic waveforms are superimposed. Such displays, and their digital counterparts, are very widely used in analysis of long-term ECG recordings. WAVE's Scope window offers some of the features of this type of display, with a few unique capabilities and limitations of its own.

To open the Scope window, first open the Analyze window, then click left on (Show scope window). The Scope window simulates an oscilloscope by using color map animation (on hardware with an X server that supports this technique) or stipple masks (otherwise). The display is "triggered" by beat (QRS) annotations (other annotations, such as rhythm changes, are ignored). As each waveform is drawn, the previously drawn waveform is made to fade progressively into the background, disappearing completely after four new waveforms have been drawn. The appearance is similar to that of an analog scope with moderately long-persistence phosphor. The x-position in the window of the trigger point (the sample that the annotation points to for each waveform) is adjustable using the dt: setting (the value is expressed in seconds, and may be negative). It is also possible to resize the window, so that several waveforms can be viewed side-by-side as well as superimposed. As shown in the figure at left, WAVE shifts waveforms to the bottom of the window if they have been annotated as ventricular ectopic beats. (This behavior was chosen to be most useful for studying ECGs, but other signals can be annotated in the same way if desired.)

The navigation controls are similar to those in the Log window. (<) and (>) may be used to move backward and forward through the record one beat at a time. (<<) and (>>) provide a continuous "movie" (at a speed controlled by the Speed slider) that can be interrupted by (Pause), which also recenters the signal window on the current beat. Whenever the current beat is visible in the signal window, it is marked with the selection rectangle.

Only one signal at a time can be displayed in the Scope window. Select that signal using the Signal field in the Analyze window, or press and hold the [Shift] key while clicking left in the signal window near the desired signal.

3.8 Analysis example: Measuring ST deviations

We now turn to an extended example showing how an application can be written in C to be used with WAVE. The application is an example only; it is not intended for any serious application, clinical or otherwise. Its structure illustrates, however, how a real application can be built.

As noted in the previous section, changes in the ST segment of the ECG are of interest because of their relationship to ischemia (which occurs when the oxygen supplied by the coronary arteries is insufficient to meet the demands of the myocardium). Characteristically, ischemia affects the shape of the ST segment, which corresponds to the first phase of ventricular repolarization following the QRS complex; typically, the ECG signal may not return to its baseline, or isoelectric, level (defined in the interval between the P-wave and the QRS complex) until after the T-wave. The level of the ST segment relative to the isoelectric level is the ST deviation. Conventionally, the ST deviation is measured at a single point 80 milliseconds after the J point (the end of the QRS complex). An ST deviation of 100 μV is considered to be clinically significant, and consistent with ischemia.

We will design a program for measuring ST deviations using WAVE. We assume that the QRS complexes have been marked, and that, in addition, two special annotations (`(`, signifying the beginning of the QRS complex, and `)`, signifying its end) have also been marked for the first beat that we wish to measure. (To use our program on an unannotated record, we can use (Mark QRS complexes), and then insert the special annotations manually around the first beat.)

Let us begin by deciding how the program should be invoked from WAVE. At a minimum, we should be able to specify which signal should be measured, and the region of interest. The program also needs to know the record and annotator names (the latter so that it can read the annotations it needs to get started). Following the conventions used by many other DB applications, we add the following entry to WAVE's menu file:

```
Measure ST deviation<TAB>stdev -r $RECORD -a $ANNOTATOR \
                              -f $START -t $END -s $SIGNAL
```

Here is the C source for the program. If this looks unfamiliar, read about how to write DB applications in the *ECG Database Programmer's Guide.*

```
/* file: stdev.c        G. Moody        19 August 1996

Sample application for use with WAVE

This program measures ST deviations given QRS annotations and a pair of
'(' and ')' annotations around the first QRS annotation.  Its output
contains two columns of numbers:  the elapsed time in minutes, and the
measured ST deviation in microvolts.
*/
```

```
#include <stdio.h>
#include <ecg/db.h>
#include <ecg/ecgmap.h>

main(int argc, char **argv)
{
    char *from_string = "0", *record = NULL, *to_string = "e";
    static DB_Siginfo si[DB_MAXSIG];
    DB_Anninfo ai;
    DB_Annotation annot;
    int i, s = 0;        /* analyze signal 0 unless otherwise specified */
    double spm;
    DB_Sample dv, v0[DB_MAXSIG], v1[DB_MAXSIG];
    DB_Time dt = 0L, dt0 = 0L, dt1 = 0L, epoch, ms80, tfrom = 0L, tpq = 0L,
        tqrs = 0L, tto = 0L;

    ai.name = NULL; ai.stat = READ;
    /* Read all of the command-line arguments. */
    for (i = 1; i < argc-1; i++) {
        if (argv[i][0] == '-') switch (argv[i][1]) {
        case 'a':        /* annotator name follows */
            ai.name = argv[++i];
            break;
        case 'f':        /* start time follows */
            from_string = argv[++i];
            break;
        case 'r':        /* record name follows */
            record = argv[++i];
            break;
        case 's':        /* signal number follows */
            s = atoi(argv[++i]);
            break;
        case 't':        /* end time follows */
            to_string = argv[++i];
            break;
        default:
            fprintf(stderr, "%s: unrecognized option %s\n",
                    argv[0], argv[i]);
        }
        else {
            fprintf(stderr, "%s: unrecognized argument %s\n",
                    argv[0], argv[i]);
        }
    }
    if (record == NULL || ai.name == NULL) {
        /* complain unless both RECORD and ANNOTATOR were specified */
        fprintf(stderr, "usage: %s -r RECORD -a ANNOTATOR [ OPTIONS ]\n",
                argv[0]);
        fprintf(stderr, "OPTIONS may include:\n");
```

```
    fprintf(stderr, "-f START\n");
    fprintf(stderr, "-s SIGNAL\n");
    fprintf(stderr, "-t END\n");
    exit(1);
}
if (s < 0 || isigopen(record, si, DB_MAXSIG) <= s) {
    fprintf(stderr, "%s: invalid signal number %d\n",
            argv[0], s);
    exit(2);
}
if (annopen(record, &ai, 1) < 0) {
    fprintf(stderr, "%s: can't read annotations\n",
            argv[0]);
    exit(3);
}

ms80 = strtim("0.08");        /* 80 milliseconds, in sample intervals */
spm  = strtim("1:0");        /* 1 minute, in sample intervals */
if ((tfrom = strtim(from_string)) < 0L)        /* start time */
    tfrom = -tfrom;                /* convert absolute to elapsed time */
epoch = tfrom + spm;
if ((tto = strtim(to_string)) < 0L)                /* end time */
    tto = -tto;
while (getann(0, &annot) == 0) {        /* read an annotation */
    if (annot.time > epoch) {
        fprintf(stderr, ".");        /* show progress by minutes read */
        epoch += spm;
    }
    if (annot.anntyp == WFON)        /* found a `(', save its TOC */
        tpq = annot.time;
    else if (annot.anntyp == WFOFF && tqrs > 0L) {
                            /* found a valid `)', update dt1 and dt */
        dt1 = ms80 + annot.time - tqrs;     /* samples from QRS to J+80 */
        if (dt0 > 0L) dt = dt0 + dt1;        /* samples from PQ to J+80 */
    }
    else if (isqrs(annot.anntyp)) {        /* found a beat label */
        tqrs = annot.time;
        if (tpq > 0L) { /* previous `(' went with this beat, update dt0 */
            dt0 = tqrs - tpq;                /* samples from PQ to QRS */
            tpq = 0L;
            if (dt1 > 0L) dt = dt0 + dt1;
        }
        if (tqrs >= tfrom && dt0 > 0L && dt1 > 0L) {
            if (tto > 0L && /* if tto is zero, continue to end of record */
                tqrs > tto)
                break;        /* nothing further to be done */
            /* tqrs, dt0, and dt1 are all valid: make a measurement */
            if (isigsettime(tqrs - dt0) < 0 ||
                getvec(v0) <= s) {
                fprintf(stderr,
```

```
                    "%s: can't read signal %d at time %s\n",
                    argv[0], s,
                    mstimstr(tqrs - dt0));
                exit(4);
            }
            for (i = 1; i <= dt; i++)
                if (getvec(v1) <= s) {
                    fprintf(stderr,
                        "%s: can't read signal %d at time %s\n",
                        argv[0], s,
                        mstimstr(tqrs - dt0 + i));
                exit(4);
                }
            dv = v1[s] - v0[s];         /* ST deviation, in ADC units */
            printf("%g\t%d\n",         /* print a line of output */
                   tqrs/spm,          /* elapsed time in minutes */
                   adumuv(s, dv));    /* ST deviation in microvolts */
        }
    }
    else        /* found some other annotation: ignore it */
        continue;
  }
  if (dt == 0) {        /* we didn't make any measurements -- complain */
      fprintf(stderr, "\n%s: no measurements made\n", argv[0]);
      exit(5);
  }
  exit(0);
}
```

The source for this program is included in the WAVE distribution as `stdev.c`. Copy this file to the WAVE host if necessary, and compile it in a terminal window using a command such as

```
cc -o stdev -O stdev.c -ldb
```

Depending on how the DB Software Package has been installed on the WAVE host, you may need to use `-I/usr/local/include` and `-L/usr/local/lib` options so that the compiler can locate the `*.h` files and the DB library. On some systems, the C compiler may have a different name, such as `gcc`. Consult an expert such as your system administrator if in doubt about compiling a C program with the DB library.

Once the source has been compiled successfully, install the executable binary file `stdev` in a directory in your `PATH`. If you are using the C-shell, you may need to run the command `rehash` to make your shell aware that the program has been installed. Test the program in the terminal window by typing

```
stdev
```

which should produce a brief summary of its options. *If this test fails, figure out why and correct the problem before continuing.*

Once the program has been compiled and installed successfully, try running it from within WAVE. Remember that the annotation file must include the '`(`' and

')' annotations to mark the beginning and end of the first beat to be measured. Using it on signal 0 of record 100s, with a suitably augmented set of reference annotations, I obtained output beginning with:

```
0.0171296 -84
0.0306481 -24
0.0437963 -9
0.0569907 -34
0.0701389 -64
0.08375 -59
0.0946296 -49
0.111204 -29
0.125278 -4
0.138796 -59
0.151944 -9
0.164815 -34
```

(Your output will vary, depending on the exact locations of the '(' and ')' annotations that you insert.)

You can plot this output by piping it to the `plot2d` command (see `plot2d(1)`, in the *ECG Database Applications Guide*, for details on `plot2d`, a command-line preprocessor for `gnuplot`). To do so, modify the entry in the menu file, so that it becomes:

`plot2d` *will not work unless* `gnuplot` *has been installed.* `gnuplot` *is included in most Linux distributions, and is also available by anonymous FTP from* `prep.ai.mit.edu` *and numerous other sites. If* `plt` *is available on your system, you may use it instead of* `plot2d` *(with the same options).*

```
Plot ST deviation<TAB>stdev -r $RECORD -a $ANNOTATOR \
                        -f $START -t $END -s $SIGNAL | \
                       plot2d 0 1 \
                        -t "ST deviations, Record $RECORD" \
                        -x "Elapsed time (minutes)" \
                        -y "ST deviation (microvolts)"
```

If you now reread the menu and use the new (Plot ST deviation) button, `plot2d` opens a window with a plot in it, similar to figure 3.5. (To make this plot, I used the first ten minutes of record 100; the sample record 100s contains only the first minute of data shown in the plot.) This window stays open until you dismiss it by pressing [Enter] ([Return]) in the Analysis Commands window; if you forget to do so, the next command you run from the Analyze window will serve this function (but the command itself won't run, since `plot2d`, rather than the command interpreter, will have read the command string).

Further work

If you would like to make this program work better, here are a few ideas:

- Extending it to make measurements on multiple signals is an easy project. Allow the user to set measurement reference points on each signal separately.

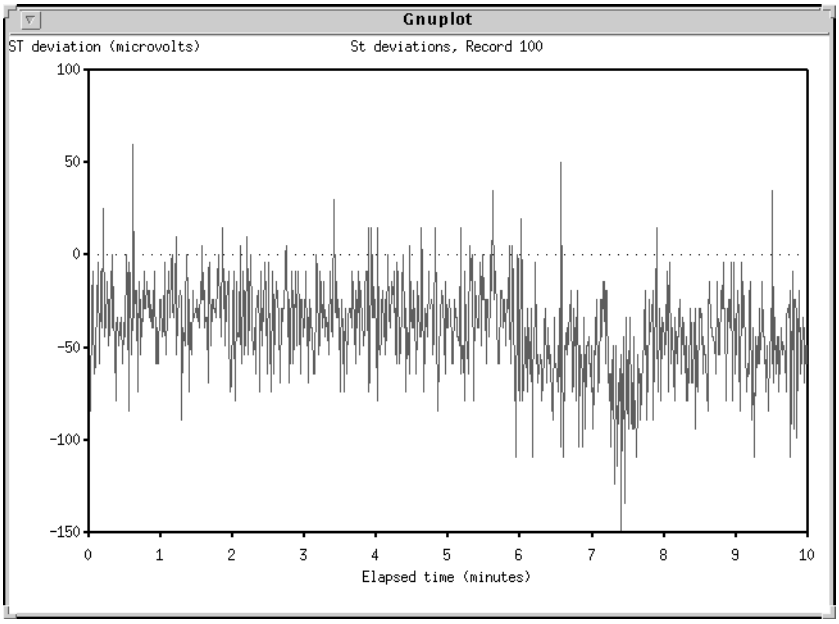


Figure 3.5: ST deviations measured by the example program.

- It's also not too difficult to copy the input annotation file, writing the measurements into the `num` fields of the annotations, so that WAVE can display them as a signal. If you try this, keep in mind that `num` values must be integers in the -128 to 127 range, so don't record the deviations in μV. If you want to write the measurements into a *signal* file (as we did with `tach` earlier in this chapter), this can get complicated—see the source for `tach` for hints about doing this.

- The measurement point should move closer to the J point at high heart rates; see any reference on the ECG for a discussion of this issue.

- The measurements as made above are very easily contaminated by small amounts of noise in the signals. You should be able to improve them significantly by averaging the amplitudes of several samples in the neighborhoods of the isoelectric point and the ST level measurement point.

- The program should not produce measurements for ventricular ectopic beats, or for beats that are extremely noisy. Noise detection is an interesting research problem in itself.

- You may wish to examine using beat averaging (or median filtering) to reduce noise in the measurements further; the *ECG Database Programmer's Guide* describes a beat averager that may be a useful starting point.

- Another source of measurement error is baseline wander in the ECG, visible as slopes in the intervals between beats. You can reduce this error by *modelling* the baseline drift and subtracting it from the signal. Simple linear models, or quadratic or cubic splines, can be used.

3.9 Controlling WAVE from an external program

Although the facilities discussed in the previous sections provide highly flexible control of external analysis programs, in some cases it is desirable for another program to control WAVE rather than for that program to run under WAVE's control.

`wave-remote` *is not provided with* WAVE *5.0, which does not allow for external control.*

The program `wave-remote`, provided with the WAVE distribution, can drive WAVE's display. Using `wave-remote`, you can cause WAVE to go to any specified location in the current record, to load another set of annotations for the current record, or to open another record. By providing a command-line interface for driving WAVE, `wave-remote` simplifies the task of driving WAVE from other programs or shell scripts; in general, it is sufficient merely to fork a new process and invoke `wave-remote` with the appropriate arguments (for example, using the `system()` function provided in the standard C library). If it is not acceptable to start a new process for this purpose, `wave-remote` is also provided in C source form, and the functions it uses to drive WAVE can be invoked directly from your program.

Options for `wave-remote` are:

`-pid` ***processid*** Control the WAVE process with the specified *processid*. (If this option is omitted, and more than one instance of WAVE is running, `wave-remote` controls the one with the highest *processid*.)

`-r` ***record*** (Re)open the specified *record*.

`-a` ***annotator*** (Re)open the specified *annotator* for the current record.

`-f` ***time*** Go to the specified *time* in the current record.

For example, the command

```
wave-remote -r 100s -a atr -f 0:22
```

causes WAVE to open record `100s`, with annotator `atr`, and to show data starting at 22 seconds after the beginning of the record.

If WAVE is not running when `wave-remote` is invoked, `wave-remote` launches WAVE provided that the record to be opened has been specified. Once `wave-remote` has delivered its instructions to WAVE, it exits immediately.

As an alternative to the command-line interface offered by `wave-remote`, the `wavescript` application provides the same services, but is controlled by a script file (named on the `wavescript` command line). Scripts for `wavescript` should contain on each non-comment line a single option/argument pair as described above for `wave-remote`. Any line that does not begin with a '`-`' is treated as a comment line. For example, if the file `example.xws` contains

```
# Here is a comment
-r 100s
-a atr
-f 0:22
```

then the command '`wavescript example.xws`' has the same effect as the `wave-remote` example above.

Using these interfaces, an analysis program (or any program, such as a web browser) can drive WAVE's display automatically. In a typical application, a semi-automated analysis program might require the user to make or correct annotations for portions of a record that are selected as part of the program's analysis (and that cannot be known *a priori*).

An analysis program that runs `wave-remote` or `wavescript` may itself be running within WAVE's own Analysis Commands window. This might be useful to permit the user to set the region of interest interactively, before the analysis begins and takes over control of WAVE to show its progress or to obtain additional input.

3.10 WAVE and the Web

In this section we explore how World Wide Web documents and DB records (i.e., signals and annotations viewable using WAVE) can be linked using WAVE together with a web browser. Beginning with WAVE 6.0, it is possible while running a web browser to click on a link from a web document to a specified location in a DB record and have WAVE open the record at that location. It is similarly possible while running WAVE to click on a link annotation (see section 2.16, page 24) and have your web browser open the data specified by the link.

Controlling WAVE from a web browser

Most web browsers, such as Netscape, can use so-called *helper* or *viewer* applications. These are external applications that are invoked by the browser to present data in formats that (usually) are not supported by the browser directly. For example, browsers often use `ghostscript` to display PostScript data. It is possible to configure Netscape and other web browsers so that WAVE can be used indirectly as a helper application.

By using `wavescript` (see section 3.9, page 42) as the helper application for viewing DB records, an already-running WAVE process can be made to open a record and an annotation set and to move to a specified location in the record. With this approach, it is not necessary to start a new WAVE process each time.

You can set up your browser to view files with the MIME type `application/x-wavescript` or suffix `.xws` using `wavescript`. To do so, add the line

```
type=application/x-wavescript  exts=xws
```

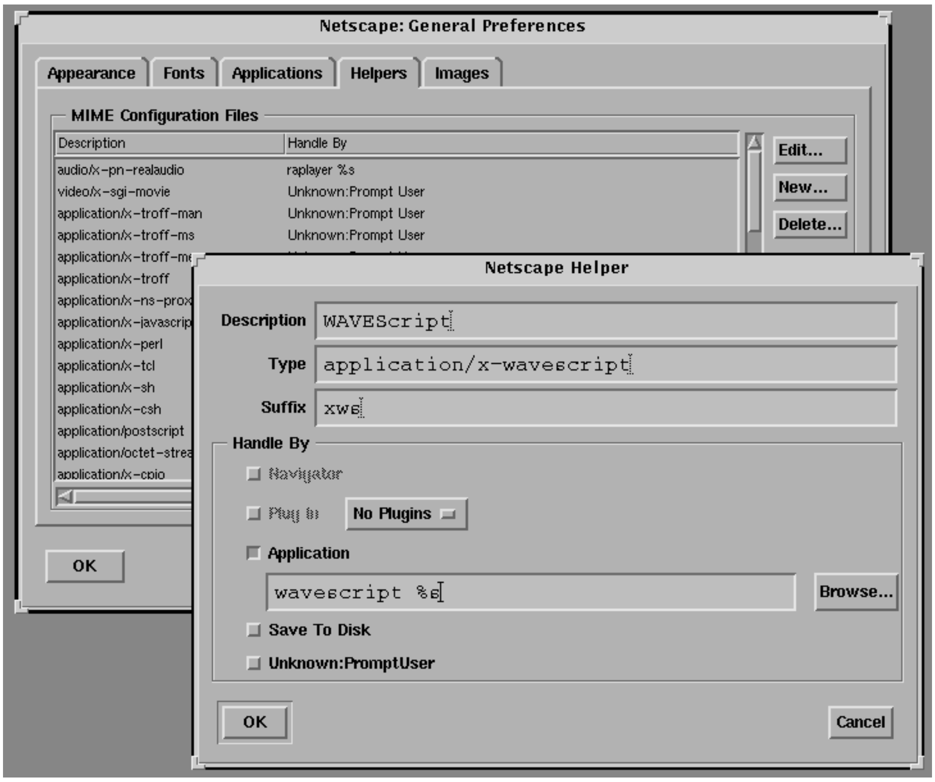
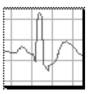


Figure 3.6: Adding `wavescript` as a Netscape helper.

to the file named `.mime.types` in your home directory, and add the line

```
application/x-wavescript; wavescript %s
```

to the file named `.mailcap` (also in your home directory). If your browser is Netscape, you can make these additions to `.mime.types` and `.mailcap` by choosing General Preferences from the Options menu, clicking on the Helpers tab, then clicking on the New button. Fill in the dialog box as shown in figure 3.6, then save your settings.



Once `wavescript` has been installed as a helper application, your web browser will launch `wavescript` whenever you click on a link to a `.xws` script file. As an example, the on-line version of this guide contains a link to the `example.xws` script shown in the previous section. To provide such a link in an HTML document, use a statement such as

```
<a href="example.xws">
<img src="wave.gif" alt="click here to view waveforms"></a>
```

The `wave.gif` icon (shown above) is provided in the WAVE distribution.

Serving .xws files

If you plan to make .xws files available from a Web server, you will also need to configure your Web server so that it sends the appropriate MIME type encoding (application/x-wavescript) whenever a browser requests a .xws file. (When you use a browser to read a local file, this is not required, since no Web server is involved in this case; rather, the local .mime.types file is used by the browser to recognize the type based on the file suffix.)

To configure the Apache httpd (at this time, the most widely-used Web server), find the file srm.conf (typically in /etc/httpd/conf) and add the line

```
AddType application/x-wavescript .xws
```

in the section that contains other AddType directives. Restart the server to force it to reread srm.conf. The NCSA httpd and variants of it can be configured in the same way; if you use the CERN httpd, the configuration file is httpd.conf, and the syntax of the AddType directive is different:

```
AddType .xws application/x-wavescript text
```

If you use some other Web server, consult its documentation to see how to proceed.

Note that readers of Web pages with a link to a .xws file will need to have WAVE and wavescript, as well as the DB record indicated in the .xws file, available to them *on the system on which their Web browser is running.*

Note that if you attempt to read a .xws file before completing the necessary configuration steps, your browser's cache may interfere with proper operation after the configuration is completed. If this happens, simply delete any .xws files from your browser's cache.

Controlling a web browser from WAVE

WAVE uses a web browser to display external data associated with link annotations, and to display the on-line version of this guide. A suitable web browser must be installed on the WAVE host in order for this to work. By default, WAVE uses Netscape version 1.1 (or any later version; free copies of Netscape for academic use or for evaluation purposes may be obtained by anonymous FTP from ftp.netscape.com).

WAVE controls the web browser in the same way it controls other external programs: by running commands in the Analysis Commands window. WAVE's interface to the web browser is defined by the line beginning with the <Open URL> tag in WAVE's menu file. In the standard version of the menu file, this line reads:

```
<Open URL>     urlview $URL
```

The program urlview is a shell script (normally installed at the same time as WAVE, and in the same directory) that handles starting the browser if necessary and instructing it to display the specified URL. The standard version of urlview does so using the command:

```
( netscape -remote 'openURL($URL)' || netscape $URL ) &
```

As this example illustrates, the menu variable `$URL` can be used to pass the selected URL from WAVE to your browser. (WAVE expands any incomplete URLs from annotation files as needed before evaluating `$URL`.) In this case, WAVE first uses Netscape's `-remote` option to instruct an already-running copy of Netscape to open the desired URL. This is a very fast operation if Netscape is already running (since Netscape is already in RAM, it is not necessary to load another in order to run the `netscape -remote ...` process, and the remote process efficiently delivers its message and exits immediately). If Netscape was not running, the `netscape -remote` command fails, and (in this case only) WAVE starts a new Netscape browser process.

To configure WAVE to use a different web browser, edit the `urlview` script appropriately. (You may also do this by editing the action associated with `<Open URL>`, but it's better to modify `urlview`, since WAVE also uses `urlview` to display some of its on-line help. If you decide to modify the action in the WAVE menu file, be careful not to change the `<Open URL>` tag at the beginning of the line, since WAVE uses this tag to identify the browser interface command within the menu file.) Consult the documentation for your browser to see what commands will be needed. (The `-remote` option is unique to Netscape; other browsers will probably require different methods.) Note that early versions of Mosaic (prior to 2.5) used a different method for remote control than current versions do. Browsers that do not support any means of remote control are best avoided (since the only way to use them is to start a new browser for each new URL, a very inefficient solution).

Chapter 4

Printing from WAVE

4.1 Standard methods for printing

There are three standard ways to print annotated signals from WAVE. The first is the easiest:

- From the File menu, select Print. This produces a "chart recorder" format copy of the contents of the signal window, as shown in figure 1.3 on page 10.

The other two methods require that you first mark the region to be printed using the '<' and '>' markers, or by entering the time interval in the Start and End fields of the Analyze window. You may also arrange the Signal list so that it contains the signals you wish to print, in the desired top-to-bottom order. After this preparation, you are ready to print:

- Click on (Print chart) in the Analyze window to produce a "chart recording" as shown in figure 4.1.

- Click on (Print full disclosure) in the Analyze window to produce the highly compact format shown in figure 4.2.

In addition to these three methods, you may also want to try the method described in section 3.6, page 34, for printing a WAVE log together with charts of the signals and annotations for each entry in it.

In each case, printing is performed by spawning a `pschart` or `psfd` process in the Analysis Commands window (after saving any edits). Since these commands reread the signals, the output generally has much higher resolution than is possible on screen.

If you are unable to print from WAVE, your printer may not be set up. See section C.2, page 88 for instructions.

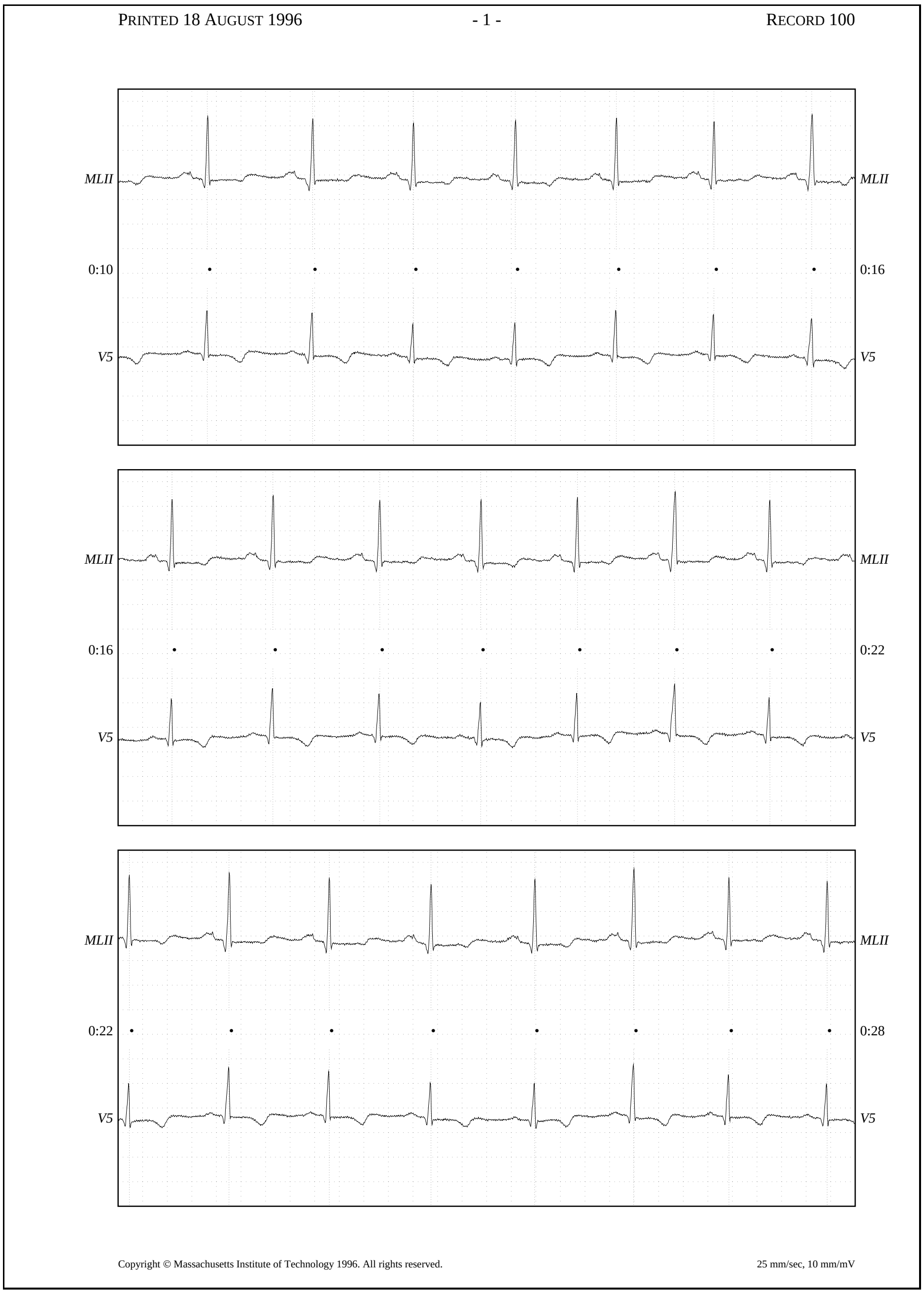


Figure 4.1: "Chart recording" made using Print chart from the Analyze window.

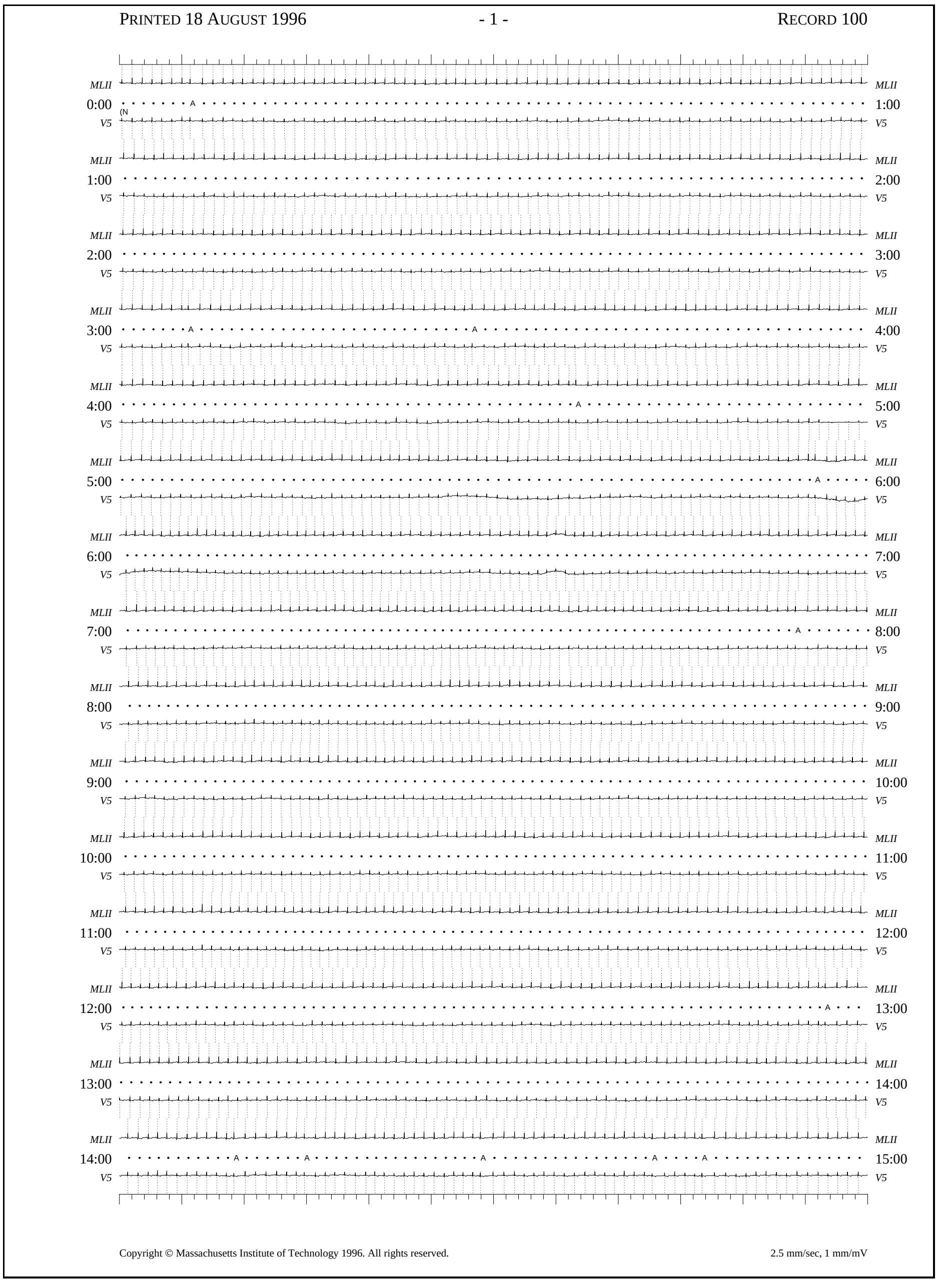


Figure 4.2: Plot made using Print full disclosure from the Analyze window.

4.2 Creating custom formats for printing

Since WAVE's printing facilities are implemented through its menu file, you have complete control over the format of the output, simply by editing the menu file as in the previous chapter. Both `pschart` (used to produce "chart recorder" output) and `psfd` (used to produce "full disclosure" output) are highly flexible and can be configured to produce almost any reasonable format by inserting the appropriate options in the commands given in WAVE's menu file. You may change the predefined commands or add new ones as you wish.

In the menu file, the special tag "`<Print>`" identifies the command that runs when you select Print from the File menu. You can change this command as you wish, but don't change the "`<Print>`" tag itself.

A few of the most commonly-used options (usable with either `pschart` or `psfd`) are listed below. For further details on these and other options, see `pschart(1)` and `psfd(1)`, in the *ECG Database Applications Guide*. (You can also get a brief summary of the many available options by typing `pschart -h` and `psfd -h`.)

`-d` ***n*** Optimize for use with a printer with a resolution of *n* dots per inch (default: 300 dpi, the typical resolution for low-end laser printers). This option has no effect on the output scale, and need not be correct for your printer; it does, however, affect the amount of detail rendered and the thickness of the lines that are drawn.

`-E` Generate EPSF format, suitable for inclusion in another PostScript file. (I used this option when preparing the `pschart` and `psfd` plots in this guide). The default format is PostScript, but *not* EPSF, for two reasons. First, a legal EPSF plot must fit entirely on one page, but `pschart` and `psfd` are frequently used to produce multi-page plots. Second, an EPSF plot may be rescaled if it is included in another document, and the scales printed at the bottom of the plot will be incorrect if this happens (as is the case in the plots reproduced in this guide).

`-g` Print a grid (using `pschart`) or a set of time axes (using `psfd`).

`-l` Label the signals in the plot margins.

`-L` Print in "landscape" orientation (default: portrait orientation). The charts made by selecting Print from the File menu are printed using this option.

`-M`***n*** The numeric argument *n* attached to this option (there should be no whitespace between `-M` and *n*) controls where annotations appear relative to the signals, and the appearance of the marker bars. Unless you display annotations as a signal in WAVE, you may wish to use the option `-M$DISPMODE` in the menu file – if you do so, then your printed output will show annotations and marker bars the same way that WAVE does.

`-P` ***pagesize*** Use this option to print on non-standard paper, or to make a plot that fits in a smaller space. The default *pagesize* is set when `psfd` and `pschart` are compiled. For the precompiled binaries distributed with WAVE, the default is `letter` (US letter size paper, 8.5 x 11 inches, or 216 x 279 mm). In most of the rest of the world, the standard is `A4` (210 x 297 mm). You can specify pagesize as "`letter`", "`A4`", or a number of other popular sizes, or as "*width*x*height*", where *width* and *height* are the dimensions in millimeters. I used this option to prepare figure 1.3 (on page 10) in a relatively compact form; if you print a figure like it with the default page size, there will be considerable empty space between the plot itself and the page headers and footers.

`-t` ***n*** This option sets the time scale, in mm/sec. Defaults are 12.5 mm/sec for `pschart`, and 2.5 mm/sec for `psfd`. For `pschart`, you might wish to use the option `-t $TSCALE` in the menu file, so that your charts will be printed at the same time scales you choose for WAVE's signal window.

`-v` ***n*** This option sets the amplitude scale, in mm/mV. Defaults are 5 mm/mV for `pschart`, and 1 mm/mV for `psfd`. For `pschart`, you might wish to use the option `-v $VSCALE` in the menu file, so that your charts will be printed at the same amplitude scales you choose for WAVE's signal window.

This list of options only hints at what is possible using `pschart` and `psfd`. There are many other options described in the `man` pages, and it is also possible to redefine the PostScript primitives used by these programs (for example, to change the appearance of the grid, or to plot in color on a color-capable printer) by modifying their PostScript prolog files. It is also possible to replace these programs entirely (for example, with your own applications for printing on a non-PostScript printer) if desired, simply by editing WAVE's menu file.

Chapter 5

Calibration

By *calibration*, we refer to the information needed to establish the true amplitudes of signals.

There are two distinct aspects of calibration. First is *signal calibration*, the process of establishing the relationship between the sample values in analog-to-digital converter units (*adus*) and the physical units of the signals (millivolts, millimeters of mercury, etc.). Second is *display calibration*, the process of determining standard scales for displaying various types of signals (e.g., 10 mm/mV, 1 mm/10 mmHg).

5.1 Signal calibration

If your signals come from standard databases (see Appendix B, page 81), signal calibration is usually not a concern. Briefly, the *header file* for each database record contains calibration information for each signal. This information includes (for each signal) the gain (the number of adus per physical unit), the baseline (the sample value that corresponds to zero physical units), the type of physical unit, and the type of signal. (Not all header files contain all of this information, but WAVE can make reasonable guesses about any of it that may be missing in most cases.)

If you have digitized your own signals, however, you should generally calibrate them before processing them further. WAVE makes it easy to do so, provided that you have recorded signals with known amplitudes. The procedure is:

- Use WAVE to display the uncalibrated signals.
- Find a calibration pulse (or a segment with known amplitudes). Mark samples of the low and high amplitude phases using the ‘<’ and ‘>’ markers respectively.
- Set the Signal list field in the Analyze window to match the signal(s) you have marked. You can calibrate more than one signal at a time if suit-

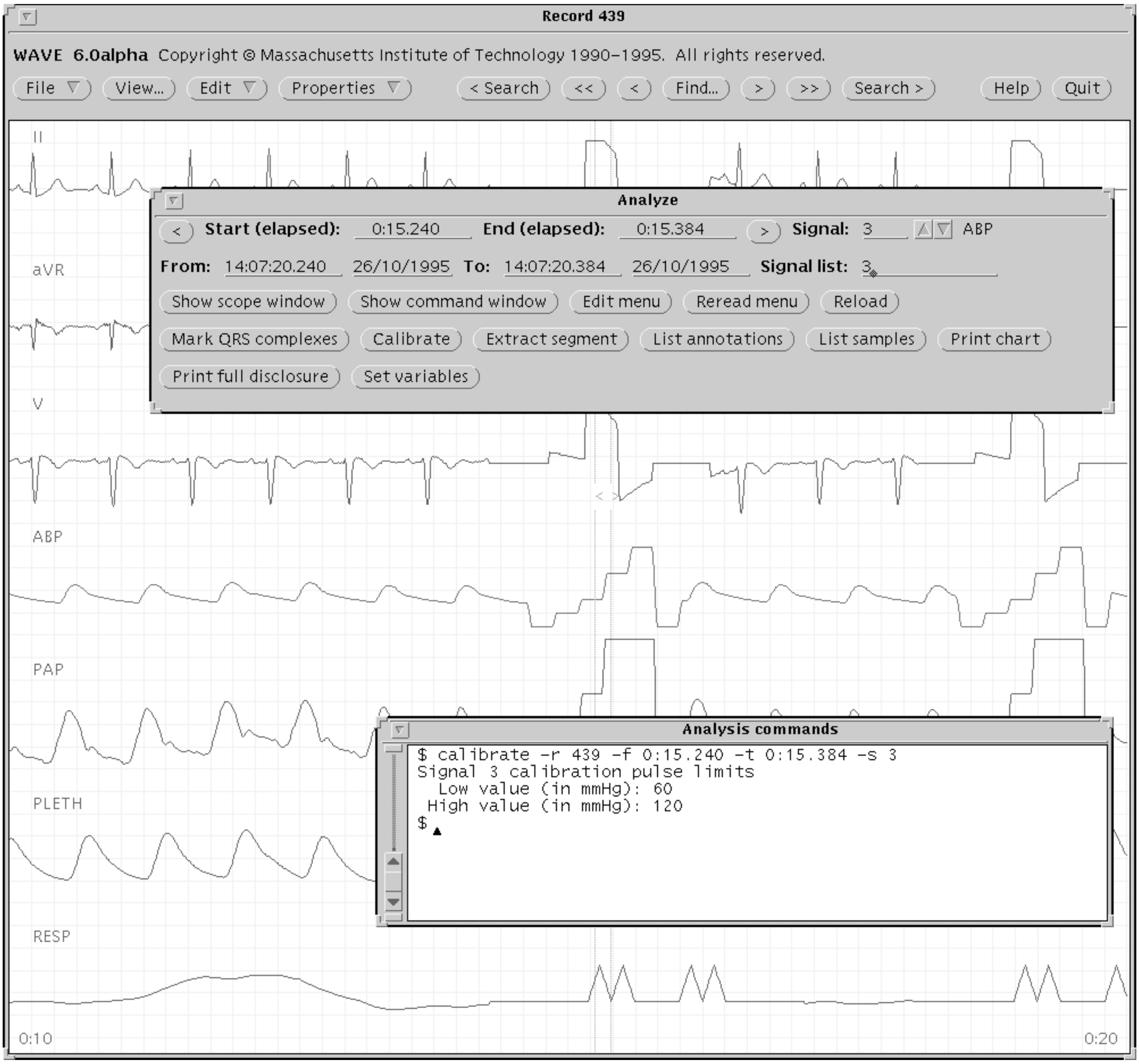


Figure 5.1: Signal calibration using WAVE. Signal 3 (ABP) has been selected (note that all of the other signal numbers have been removed from the Signal list). The '<' and '>' markers bracket a step in the recorded calibration pulse near the center of the signal window. In the Analyze window, we have clicked on (Calibrate), and the result of doing so appears in the Analysis Commands window.

able calibration pulses for each signal are present between the '<' and '>' markers. Be sure to remove any other signals from the Signal list.

- Click left on (Calibrate) in the Analyze window, and answer any questions that appear in the Analysis Commands window. These questions will always appear if the header file for the record does not contain signal descriptions or physical unit specifications; some or all of the questions may not appear if the signal type and calibration pulse type are already known.

- Repeat the previous steps for any other signals that require calibration.

- Close and reopen the record. (WAVE does not adjust the display scales until this has been done.)

Figure 5.1 illustrates signal calibration using a record from the MIMIC Database. In this case, the type of signal (ABP) is known from an entry in the `DBCAL` file (not shown here), so that `calibrate` is able to determine the physical units of the signal (mmHg). Since a variety of calibration pulses are used for ABP signals, the `DBCAL` file does not specify the pulse levels, which `calibrate` has asked us to enter (in this case, we have entered `60` and `120`). Based on this information, `calibrate` determines the offset and gain needed to convert raw sample values for signal 3 into ABP measurements in mmHg. `calibrate` then makes the appropriate changes to the header file for the current record. When WAVE or another DB application next opens this record, the ABP signal will be properly calibrated.

By default, `calibrate` generates an amplitude histogram of the samples between the '<' and '>' markers. It then identifies the low and high amplitude portions of the calibration pulse by searching for the two largest distinct modes in this amplitude histogram. For this reason, `calibrate` works best if the segment bounded by the '<' and '>' markers includes at least a few samples of both the high and low amplitude phases. Avoid placing either marker immediately next to the transition point between the phases if possible. If `calibrate` fails to find two distinct peaks in the amplitude histogram, it will produce an error message; if this happens, adjust the positions of the markers and try again.

In some cases (for example, if the calibration pulse is a sawtooth, as in the RESP signal at the bottom of figure 5.1), this strategy may fail, no matter where the markers are placed. In such cases, try again with `calibrate`'s `-q` or `-Q` options to use one of its alternate algorithms. (These are less robust since they depend on differential rather than integrative measurements, but they can be used in a pinch.)

For further information on signal calibration, see `calibrate(1)`, in the *ECG Database Applications Guide.*

5.2 Display calibration

Display calibration is performed by WAVE in several steps:

- First, WAVE determines the display resolution (either from the `-dpi` command-line option if possible, otherwise from the `Wave.Dpi` resource if possible, or by querying the X server if neither the command-line option nor the resource was specified).
- Next, WAVE determines what scales to use for time and amplitude. You can set these using the Time scale: [▽] and Amplitude scale: [▽] menu buttons in the View window. The amplitude scale refers to standard ECG signals; the default scales (10 mm/mV and 25 mm/sec) are those used for standard ECG chart recordings. If you choose a non-standard amplitude scale, WAVE sets its magnification factor to the ratio between that scale and 10 mm/mV.
- Finally, if any of the signals to be displayed are not ECGs, WAVE checks the DB calibration file (named by the `DBCAL` environment variable) for entries that specify the standard display scales for those signals. WAVE multiplies these scales by its magnification factor (see above) in order to determine the final display scales.

WAVE *5.0 must be restarted in this case.*

In general, it is not necessary to do anything to make WAVE display common signals at reasonable scales. If you have a previously undefined signal type, or you need to change the size of a particular type of signal relative to that of an ECG, however, simply edit a copy of the DB calibration file, reset `DBCAL`, and restart WAVE. (If you wish, you need not exit and restart WAVE; instead, give the copy a different name, then change the Calibration file item in the Load window to match. Any changes are effective only when the signal window is redrawn.) See `dbcal(5)`, in the *ECG Database Applications Guide*, for details.

If the display calibration appears incorrect for all signals, the X server may be supplying incorrect information about the display size to WAVE. This problem can be diagnosed by setting the time and amplitude scales to their default values (nominally 25 mm/sec and 10 mm/mV), and displaying the default grid (0.5 mV x 0.2 s). If the grid intervals are not 5 mm in each direction, follow the procedure in section D.3.4, page 97.

Chapter 6

Customizing WAVE

In the previous chapters, we saw how WAVE's Analyze window can be modified by editing the WAVE menu file. This chapter describes several other ways in which WAVE can be customized.

6.1 Using the View window

In section 2.3, we briefly opened the View window (see figure 2.7, page 18) in order to turn on the marker bar display. Open the View window again, and examine the other controls in it.

Three important controls are located along the bottom edge of the window. (Redraw) makes WAVE refresh the signal window, and also closes the View window (unless you have "pinned" it by clicking on the pushpin at the upper left corner). You should click on (Redraw) in order to see the effects of any changes you make in the View window. The other two buttons in this group are described below.

You will not see the pushpin unless your window manager supports "pinnable popups", as do `olwm` *and* `olvwm`.

Along the top of the window, just below the title bar, is a row of on/off controls used to turn on various optional display elements. These controls appear depressed (darker than the background) when they are 'on'. The first four of these (subtype, 'chan' field, 'num' field, and 'aux' field) control the display of secondary fields in the annotations. If any of these controls are 'on', the corresponding fields are displayed below each annotation mnemonic, in the order shown in the View window.

The Time scale: [▽] and Amplitude scale: [▽] menu buttons allow you to choose any of a wide range of standard scales for the signal (and Scope) windows. Click right on the (▽) to open the menu.

The Draw: [▽] menu offers the choice of displaying all signals (the default) or listed signals only (i.e., those named in the Signal list in the Analyze window). By choosing to display listed signals only, you may rearrange the signals within in the signal window. By listing a signal in two or more entries in the signal list, you can arrange to have that signal drawn in two or more locations; this can be useful for making side-by-side comparisons of a signal against several others. You may also find it useful to remove one or more signals from the signal list in order to reduce crowding in the signal window; this technique is nearly essential if the record has more than about 12 signals. (You can also change the spacing between signals uniformly by resizing WAVE's main window.)

The Show annotations: ▽ menu offers three ways to display annotations: centered (the default), attached to signals, and as a signal. If you choose attached to signals, the chan field of each annotation specifies the signal number to which the annotation is attached, and the annotation is displayed slightly above this signal. (Any annotations that have chan fields that are not valid signal numbers for the current record are displayed at the center of the signal window.) If you choose as a signal, the num field of each annotation is taken as the amplitude of a signal at the time of the annotation, and WAVE draws this signal in the center of the signal window by connecting the amplitude/time pairs specified by the annotations with straight line segments.

The Time display: ▽ menu provides three alternatives for display of times in the lower corners of the signal window: elapsed (the default), absolute, and in sample intervals. Elapsed time measures the interval in hours, minutes, and seconds from the beginning of the record. Absolute time (i.e., time of day and date) is not defined for all records. If it is available, choosing absolute from this menu will cause WAVE to show absolute times in the signal window, using the color WAVE uses for annotations; otherwise, WAVE switches to elapsed time display. Times may also be displayed in sample intervals (counting from the beginning of the record) for any record; these may be recognized in the signal window by the 's' prefix.

The choice you make from the Grid display: ▽ menu determines how WAVE draws the background grid in the signal window. The grid intervals are fixed in time and amplitude units. At the default scales (25 mm/sec, 10 mm/mV) the default grid (0.2 s x 0.5 mV) has 5 mm intervals if the display is properly calibrated. (If this is not the case, see section D.3.4, page 97.) If you change the time or amplitude scales, the grid intervals change size to match, so that you always have a visual cue about the display scale if you leave the grid display on. If you choose a very small time scale (i.e., one that permits WAVE to display a large amount of data in the signal window), the grid may appear solid grey; in this case, you may wish to choose "1 m x 0.5 mV" from this menu (so that the vertical grid lines appear at 1-minute intervals), or '0.5 mV" (thereby suppressing the vertical grid lines), or even "no grid". At relatively large scales, where it may be useful to have a finer grid, choose "0.04 s x 0.1 mV". To display highly time-compressed data with fine horizontal grid lines, choose "1 m x 0.1 mV".

If you make changes in the View window and wish to discard them before clicking on (Redraw), you may do so by clicking on (Undo changes). This button does not restore the initial values if you have registered earlier changes by using (Redraw); to do this, you must restart WAVE.

If, on the other hand, you have made changes in the View window and wish to have WAVE start up with these settings, click on (Save as new defaults). Note that WAVE saves the current state of the signal window *as it appears when you click on* (Save as new defaults); if you haven't made your changes effective using (Redraw), they won't be saved.

6.1.1 Highly time-compressed displays

At highly compressed time scales (defined as any scale of 125 mm per minute or less), WAVE normally suppresses the grid display, even if you have specified a default grid display for use at other time scales. To change this behavior, select the desired grid display mode, click on (Redraw), and then on (Save as new defaults) *while viewing signals at 125 mm/min or less.*

6.1.2 Low-rate records

Low-rate records (defined as those that are sampled at rates of 10 Hz or less) are displayed by default at a time scale of 5 mm per minute. To change this default, simply change to the desired scale (all of the standard scales are available in any case) click on (Redraw), and then on (Save as new defaults) *while a low-rate record is open.*

6.2 Resizing the signal and Scope windows

WAVE's main window, which contains the signal window, may be resized using the resize "handles" provided by your window manager. If you are using an Open Look window manager such as `olwm` or `olvwm`, as recommended, the resize handles are at the corners of the window. To use them, move the pointer over a resize handle; when the pointer changes shape, click left and drag it to the desired location. (WAVE adjusts the window width to the nearest multiple of one second.) If you use (Save as new defaults) in the View window, as described above, the new window size will be the default for future sessions.

The Scope window may be resized in the same way, but its default (initial) size may not be changed.

6.3 WAVE environment variables

WAVE uses many environment variables; they are listed in this section roughly in order of importance. Many of them need not be set at all, since WAVE uses reasonable default values in most cases. Those that are set must be set on the WAVE host. For general information on how to set environment variables, see the discussion of the `DISPLAY` variable in section 1.1, page 3.

`DISPLAY` The name of the X server and display you are using (see section 1.1, page 2). If not set, WAVE will not be able to open windows on your display unless your display is attached to the WAVE host directly (not via network connection).

DB The database path (see `setdb(1)`, in the *ECG Database Applications Guide*. If not set, WAVE can find database files only in the current directory (on the WAVE host).

DBCAL The DB calibration file (see `setdb(1)` and `dbcal(5)`, in the *ECG Database Applications Guide*, and chapter 5, page 53). If not set, WAVE may not scale signals other than ECGs correctly.

WAVEMENU The name of WAVE's menu file (see section 3.1, page 28). If not set, WAVE uses `wavemenu` in the current directory if possible; otherwise, it uses WAVE's default menu (`$MENUDIR/wavemenu.def`).

SHELL The command interpreter used within the Analysis Commands window. If not set, WAVE uses `/bin/sh` (the Bourne shell). Other shell-related variables, such as `PATH`, are also significant when WAVE is running commands within the Analysis Commands window.

EDITOR The name of the text editor to be used for modifying WAVE's menu file. If not set, WAVE uses `textedit`.

PRINTER The name of a PostScript printer accessible to the WAVE host, to be used for paper output. If not set, WAVE uses the default printer.

PSPRINT The command needed to print PostScript from the standard input. If not set, WAVE uses the command `lpr` (or `lpr -P$PRINTER`, if `PRINTER` is set).

TEXTPRINT The command needed to print ordinary text from the standard input. If not set, WAVE uses the command `lpr` (or `lpr -P$PRINTER`, if `PRINTER` is set).

ANNTAB The name of a file that contains custom annotation definitions (see the discussion of `Wave.Anntab` in section 6.4, page 61 for details). If not set, WAVE uses standard annotation definitions only.

The environment variables below are not needed unless the WAVE binary distribution, or XView, has been installed in non-standard directories:

HELPPATH The path for XView spot help; if not set, WAVE initializes it to `/usr/lib/help`. WAVE's own spot help is in `$HELPDIR/wave`, which is appended to the end of `HELPPATH` by WAVE.

HELPDIR The directory in which WAVE's help directory is located; if not set, WAVE uses `/usr/local/help`.

MENUDIR The name of the directory that contains WAVE's default menu file; if not set, WAVE uses `/usr/local/lib`.

RESDIR The name of the directory in which system-wide default X11 resource files are kept; if not set, WAVE uses `/usr/lib/X11/app-defaults`. See the `man` page for WAVE for information about other directories that are searched for resource files.

6.4 X11 resources for WAVE

You can control many aspects of WAVE's appearance and behavior by setting its resources. If you are not familiar with this concept, refer to an introductory book on using the X Window System, such as Quercia and O'Reilly's *X Window System User's Guide*. Since WAVE is built using the XView toolkit, all of the resources listed in the `man` page for `xview` can be used with WAVE (type '`man xview`' for details). If your system doesn't have an XView `man` page, refer to the copy provided with the WAVE distribution (`xview.7`). In addition, the WAVE-specific resources listed below may also be set.

The standard way to change default values of X11 resources is to define them in a file named `.Xdefaults` (note the initial '`.`') in your home directory. WAVE does this when you use (Save as new defaults) in the View panel.

If you use WAVE on workstations with different display capabilities, you can create custom resource settings for each one by moving these resource definitions from `.Xdefaults` to `.Xdefaults-`*hostname* (where *hostname* specifies the system to which the display is attached).

`Wave.AllowDottedLines` This resource specifies if WAVE is allowed to render dotted lines. WAVE normally draws annotation marker bars as dotted lines, and may use dotted lines for other display elements on black-and-white displays for clarity. Some X servers do not properly render dotted lines, however; if you observe irregular or missing annotation marker bars, change the value of this resource from '`True`' to '`False`' (by editing `.Xdefaults`).

`Wave.Anntab` This resource specifies the name of a file that contains a table of annotation definitions. The environment variable `ANNTAB` can also be used to specify this filename; the resource overrides the environment variable if both are set. The file contains one-line entries of the form

```
15 % Funny looking beat
```

in which the first field specifies the (numeric) annotation code in the range between 1 and `ACMAX` inclusive (see `/usr/include/ecg/ecgcodes.h` for a list of predefined codes and for the definition of `ACMAX`); the second field ('`%`' in the example) is a mnemonic (used in annotation display and entry), and the remainder of the entry is a description of the intended use of the annotation code (which appears next to the mnemonic in the Type field and menu of the Annotation Template window). Lines in the annotation table that begin with '`#`' are treated as comments and ignored. It is not necessary to specify an annotation table when editing an existing annotation file unless previously undefined annotation types are to be added to it during the editing process, although it is generally harmless to do so.

`Wave.Dpi` This resource specifies the display resolution in dots per inch in the form *mm*x*nn*, where *mm* is the horizontal resolution and *nn* is the vertical resolution. Normally, the resolution is known to the X server, and it is unnecessary to specify this resource. If your X server is misinformed, WAVE's calibrated display scales will be incorrect; the best solution is to specify the resolution using a server option such as the `-dpi` option supported by MIT's X11 servers, since this will solve problems common to any other applications that require calibrated scales as well. Not all X11 servers support such an option, however, so this option is available as a work-around. The command-line option `-dpi` overrides the resource if both are specified.

`Wave.GraphicsMode` This resource specifies the graphics mode used by WAVE; it can be overridden using the `-g`, `-m`, `-O`, or `-S` options. The legal values are 1 (monochrome mode), 2 (overlay greyscale mode), 4 (shared color mode), 6 (shared grey mode), and 8 (overlay color mode).

`Wave.SignalWindow.`{`Grey|Color`}`.`*element* These resources specify the colors to be used on greyscale or color displays. The `Color.*` resources are used only if the display is color-capable and neither greyscale nor monochrome mode has been specified. *element* is one of `Background`, `Grid`, `Cursor`, `Annotation`, or `Signal`. The defaults are:

	`Grey`	`Color`
`Background`	white	white
`Grid`	grey75	grey90
`Cursor`	grey50	orange red
`Annotation`	grey25	yellow green
`Signal`	black	blue

`Wave.SignalWindow.Mono.Background` In monochrome mode, the background is normally white, and all other elements are normally black. The reverse can be obtained by setting this resource to '`black`'. (There is at least one X server for which this fails.)

`Wave.Scope.`{`Grey|Color`}`.`{`Foreground|Background`} These resources specify the colors to be used in the Scope window on greyscale or color displays. The `Foreground` color is used for the waveform and the time display; by default, it matches the color used for signals in the signal window (see the previous item). Some X servers do not allow the background color of the Scope window to be set, because of the color map animation and stippled erasing techniques used.

`Wave.Scope.Mono.Background` This resource can be used to invert the foreground and background of the Scope window when WAVE is running in monochrome mode. This does not work for all X servers.

`Wave.SignalWindow.`{`Height_mm|Width_mm`} These resources specify the preferred dimensions (in millimeters) for the signal window. The defaults are 120 and 250 respectively.

`Wave.SignalWindow.Font` This resource specifies the font used to display annotations and time marks in the signal window. The default is '`fixed`'.

`Wave.TextEditor` This resource specifies the name of the text editor invoked by WAVE to permit you to edit WAVE's log and analysis menu files. The default is `textedit` (the OpenLook visual editor). You may override this resource by using the environment variable `EDITOR`, which is also used by many other UNIX applications that invoke editors.

`Wave.View.`*showoption* These resources specify which of the Show: options at the top of the View window are enabled by default. *showoption* is one of `Subtype`, `Chan`, `Num`, `Aux`, `Markers`, `Signal Names`, `Baselines`, or `Level`. The values of these resources are '`True`' or '`False`'.

`Wave.View.`*menuname* These resources specify the positions of the initial choices in the corresponding View menus, where the top item on each menu is in position 0, the one below it is in position 1, etc. *menuname* is one of `TimeScale`, `AmplitudeScale`, `SignalMode`, `AnnotationMode`, `TimeMode`, or `GridMode`. For example, to set the initial time scale to 50 mm/sec (the item at position 6 in the Time Scale: $\boxed{\nabla}$ menu), set `Wave.View.TimeScale` to 6.

`Wave.View.CoarseGridMode` This resource specifies the initial grid display mode to be used at highly compressed time scales (125 mm/minute or less). The possible values are the same as those for `Wave.View.GridMode`. The default is 0 (no grid); recommended alternatives are 5 (1 m x 0.5 mV) or 6 (1 m x 0.1 mV).

`Wave.View.CoarseTimeScale` This resource specifies the initial time scale to be used for low-rate records (those sampled at 10 Hz per signal or less). The possible values are the same as those for `Wave.View.TimeScale`. The default is 2 (corresponding to 5 mm per minute).

In addition to the usual ways of setting X resources, it is possible to set any of those listed above, as well as any of the generic XView resources, by using the `-xrm` or `-default` options on the command line when starting WAVE. For example, you can set the background color of the signal window using a command such as

```
wave -r 100s -xrm Wave.SignalWindow.Color.Background:lightblue
```

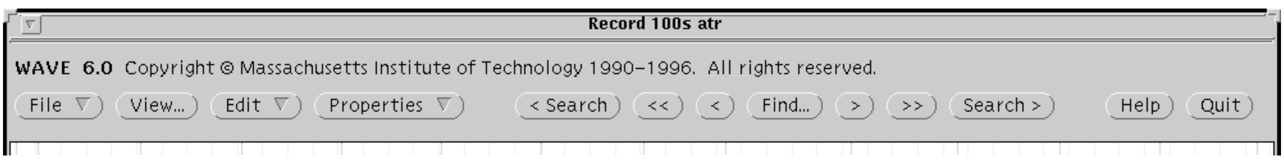
Appendix A

Summary of WAVE controls

This appendix contains an entry for each of WAVE's controls. The entries appear according to the position of the controls in left-to-right order within each window or menu. The Annotation Template, Search Template, and Scope window controls are listed separately at the end.

This information is also available as spot help for each control (point to the control and press the HELP key, or the F1 key if your keyboard does not have a HELP key).

A.1 WAVE's main window



File ▽ This button opens the File menu, containing selections for loading, saving, printing, analyzing, and logging database files. (See section A.2, page 66.)

View... This button pops up the View window, which allows you to choose (or merely examine) display scales, grid styles, and annotation, signal, and time display styles. Changes are not effective until the signal window is redrawn. (See section A.7, page 72.)

Edit ▽ The Edit menu allows you to specify if annotation editing is to be allowed or forbidden. By default, editing is forbidden when WAVE starts up. (See section A.8, page 73.)

Properties ▽ This button brings up the Properties menu, with selections for obtaining information about the current signal and annotation files and about this version of WAVE. (See section A.9, page 73.)

< Search This button recenters the signal window on the previous occurrence of an annotation or marker that matches the entry in the Search for field of the Find window, if any. If no match is found, a notice is posted, but the signal window is not recentered. If the Search for field is empty, any annotation or marker will be counted as a match.

<< This button scrolls the signal window towards the beginning of the record, by an amount equal to the width of the signal window (i.e., a full screen).

< This button scrolls the signal window towards the beginning of the record, by an amount equal to half of the width of the signal window.

Find... This button opens the Find window (see section A.10, page 74), which allows you to specify what portion of the current record should be displayed next. You may set a specific start or end time, or you may specify an annotation to be searched for by < Search and Search > buttons.

> This button scrolls the signal window towards the end of the record, by an amount equal to half of the width of the signal window.

>> This button scrolls the signal window towards the end of the record, by an amount equal to the width of the signal window (i.e., a full screen).

Search > This button recenters the signal window on the next occurrence of an annotation or marker that matches the entry in the Search for field of the Find window, if any. If no match is found, a notice is posted, but the signal window is not recentered. If the Search for field is empty, any annotation or marker will be counted as a match.

Help This button pops up the Help Topics window (see section A.11, page 75), containing buttons that name several subjects for which extensive on-line help is available. Choosing a topic allows you to browse through or print the associated help file in a scrollable text window.

Quit WAVE exits when you press this button, after saving your edits if any. If your input file would be overwritten as a result of saving your edits, it is first renamed by prefixing its annotator name with an underscore ('_'). If you have entered annotations but have not specified an annotator, the name by which you invoked this program (normally, '`wave`') is used for the annotator name.

A.2 The File menu

Open this menu using File ∇ in WAVE's main window.

Load This selection pops up the Load window (see section A.3, page 67), in which you can enter a new record or annotator name, or change the name of the calibration file or the value of the database path within this window.

Save If there are unsaved edits, this selection saves them. The annotator name in the title bar is marked with parentheses if there are unsaved edits. Only one level of backup is preserved, so you will overwrite the original annotation file if it is in the current directory and you open the same annotator more than once.

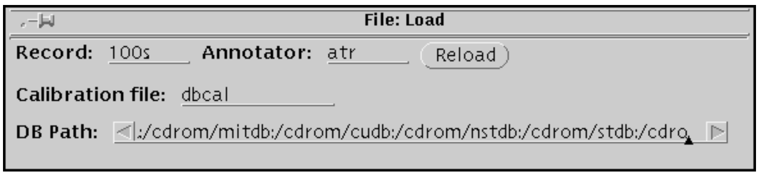
Print This selection prints the contents of the signal window on paper. The output is made from the original signal files, and therefore is of better quality than a screen dump would be. Your edits, if any, are saved before printing, so that the output reflects any changes you have made.

Print setup This selection pops up the Print setup window (see section A.4, page 68), showing the commands WAVE uses to print PostScript and text data from the standard input.

Analyze This selection pops up the Analyze window (see section A.5, page 69) containing a customizable set of action buttons, and the Analysis Commands window (a `cmdtool`-style terminal emulator). The names of the action buttons and their assigned actions are read from WAVE's menu file (by default, this is `wavemenu`, if it exists in the current directory, otherwise `/usr/local/lib/wavemenu.def`; the default may be overridden by setting the environment variable `WAVEMENU` to the name of a different file). The buttons are usually configured to perform various analysis functions on the current record; read the default menu file for details.

Log This selection pops up the Log window (see section A.6, page 70), which allows you to name a log file, and to record in that file the current record name, the time of the samples at the center of the signal window, and (optionally) a one-line comment. Log files may be used as scripts for `pschart`. You may write to as many log files in a single session as you choose, and you may accumulate entries from multiple sessions in a single log file.

A.3 The Load window



Open this window by selecting Load from the (File ∇) menu.

Record Specifies the name of the record to be viewed. The initial value of this field is the name of the record that you specified on the command line. To view another record, select this field and enter another record name.

Annotator Specifies the name of the annotator whose annotations are shown. If you specified an annotator on the command line, the annotator name is the initial value of this field; otherwise, the field is initially empty. Fill in or change this field to view or edit a different set of annotations.

(Reload) Press this button to reload the current record. This is intended to be used if an external process has modified the record (e.g., by writing annotations, or by recalibrating a signal) since it was loaded into WAVE.

Calibration file Specifies the name of the DB calibration file (a text file containing information on the relative scales of many types of signals). Initially, this field contains the value of the `DBCAL` environment variable. You should include path information in this field only if the DB calibration file is not found in the DB path.

DB path Specifies the search path for WAVE's input files. Initially, this field contains the value of the `DB` environment variable. Components are directory names, separated by colons (':'). An empty component (either an initial or final colon, or two consecutive colons) specifies the current directory.

A.4 The Print setup window

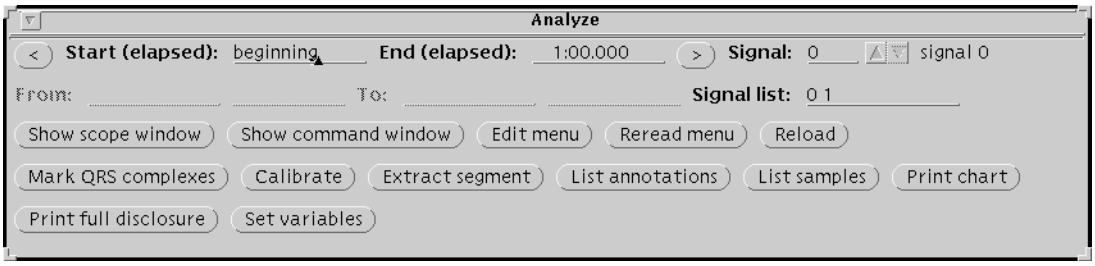
Open this window by selecting Print setup from the (File ∇) menu.

PostScript print command WAVE supplies any PostScript data to be printed (such as the chart generated when you choose Print from the (File ∇) menu) to the standard input of this command.

Text print command WAVE supplies any text to be printed (such as on-line help) to the standard input of this command.

The initial settings in the Print setup window are determined by the environment variables `PSPRINT` and `TEXTPRINT`; if either of these variables is not set, the corresponding command is determined by the `PRINTER` environment variable (see section C.2, page 88). You may change these commands (for example, to specify use of a different printer). When executing commands from its menu file, WAVE replaces the strings `$PSPRINT` and `$TEXTPRINT` with the corresponding commands from the Print setup window (see appendix F, page 115).

A.5 The Analyze window



Open this window by selecting Analyze from the (File ∇) menu.

(<) This button shifts the segment to be analyzed toward the beginning of the record, by an amount equal to the length of the segment if possible.

Start (elapsed) This field specifies (as elapsed time from the beginning of the record) the beginning of the segment to be analyzed. You may enter a time directly in this field, or you may insert a ‘<’ marker using the standard procedure for inserting annotations.

End (elapsed) This field specifies the end of the segment to be analyzed. You may enter a time directly in this field, or you may insert a ‘>’ marker using the standard procedure for inserting annotations.

(>) This button shifts the segment to be analyzed toward the end of the record, by an amount equal to the length of the segment.

Signal This field specifies the selected signal. You may enter a signal number directly in this field, or you may point to a signal, depress the [Shift] key, and click left to select the signal. In the WAVE menu file, the symbol ‘`$SIGNAL`’ refers to the signal number of the selected signal; this symbol usually specifies a signal to be analyzed. The name of the selected signal appears to the right of its signal number. The uppermost signal displayed by WAVE is signal 0.

From This field specifies the absolute time (and date, if defined) of the beginning of the segment to be analyzed. The value in this field is updated automatically whenever the value in the Start (elapsed) field changes, and vice versa. The From field is disabled if the current record’s header does not define the absolute time of the beginning of the record.

To This field specifies the absolute time (and date, if defined) of the end of the segment to be analyzed. The value in this field is updated automatically whenever the value in the End (elapsed) field changes, and vice versa. The To field is disabled if the current record’s header does not define the absolute time of the beginning of the record.

Signal list This field specifies the signal list (a list of signal numbers, separated by spaces). In the WAVE menu file, the symbol '`$SIGNALS`' refers to the signal list; this symbol usually appears where a list of signals to be analyzed is required. To change the signal list, either type into this field, or point to a signal, press and hold the [Control] key (to add the signal to the list) or the [◇] (or [Alt]) key (to delete the first occurrence of the signal from the list), and click left.

(Show scope window) This button pops up WAVE's Scope window (see section A.15, page 78, which can be used to display a signal in 'oscilloscope' mode.

(Show command window) This button pops up the Analysis Commands window, a terminal emulator that receives commands generated by selecting most of the other buttons in this window, and that displays any text output of those commands. You may type commands directly into the window.

(Edit menu) This button allows you to edit the WAVE menu file, which configures the analysis buttons in the Analyze window, using the text editor named in the `EDITOR` environment variable (or `textedit` if `EDITOR` is not set). After you have saved your changes, select (Reread menu) to reconfigure this window.

(Reread menu) Select this button to reconfigure the Analyze window after you have made changes to the menu file (most easily done by using (Edit menu)) or after changing records.

(Reload) This button causes WAVE to reload the current annotation file. If a process is in progress in the Analysis Commands window, WAVE defers reloading until the process is finished.

The remaining buttons in the Analyze window perform actions determined by the WAVE menu file. If the environment variable `WAVEMENU` is set, it names that file; otherwise, the WAVE menu file is `wavemenu` (if it exists in the current directory) or `/usr/local/lib/wavemenu.def`.

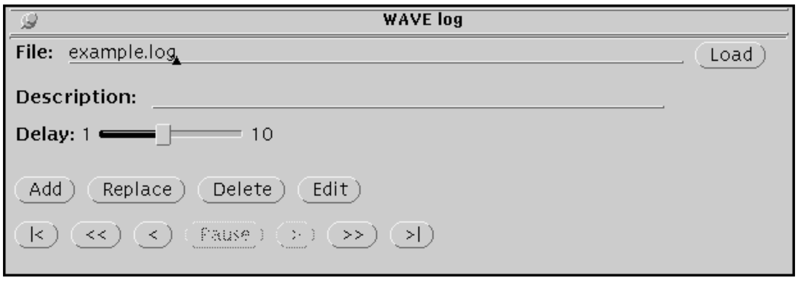
A.6 The Log window

Open this window by selecting Log from the (File ▽) menu.

File This field names the current log file.

(Load) Press this button to load (or reload) the log file if an external process (such as one started from the Analyze window, or an editor started using the (Edit) button in the Log window) creates or modifies the log file.

Description This field contains the description associated with the current log entry.



Delay This slider controls the interval between display updates when a log review is in progress.

Add This button adds the current record name, the time corresponding to the center of the signal window, and the contents of the description field, as an entry in the log file; it also inserts an index mark (':') in the center of the signal window.

Replace This button replaces the description attached to the current log entry with the current contents of the description field. It does not create a new entry (use the Add button for that purpose).

Delete This button deletes the current entry from the log. This button also causes WAVE to display the next log entry if it exists.

Edit This button opens the current log file using the text editor named in the `EDITOR` environment variable (or `textedit` if `EDITOR` is not set). Save your edits, exit from the editor, and click on the Load button in the Log window before attempting to make changes to the log using Add, Replace, or Delete.

|< This button causes WAVE to "rewind" the log (i.e., to show the first log entry).

<< This button causes WAVE to begin reviewing each entry in the log file in reverse order, pausing 5 seconds between entries. While a review is in progress, only the Pause button is enabled.

< This button causes WAVE to show the previous log entry.

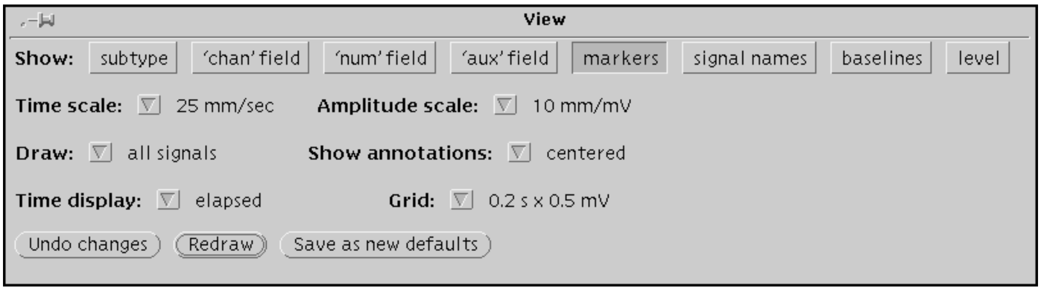
Pause This button causes WAVE to stop the review of the log file that was begun by << or >>. The Pause button is disabled unless a review is in progress.

> This button causes WAVE to show the next log entry.

>> This button causes WAVE to begin reviewing each entry in the log file, pausing 5 seconds between entries. While a review is in progress, only the Pause button is enabled.

(>|) This button causes WAVE to “fast forward” the log (i.e., to show the last log entry).

A.7 The View window



Open this window using (View...) in WAVE’s main window.

Show Toggle these options by selecting them. Multiple annotation fields are shown in the following arrangement:

> *annotation mnemonic*
> *subtype*
> `chan` *field*
> `num` *field*
> `aux` *field*

Signal names and baselines are defined in the header file for the current record. Markers show the precise locations of all annotations. Levels show the amplitudes of each signal at the time indicated by the pointer, whenever a mouse button is depressed. The Level window appears at this time.

Time scale: [▽] Set the horizontal scale for the signal and Scope windows by selecting one of the choices.

Amplitude scale: [▽] Set the vertical scale for the signal and Scope windows by selecting one of the choices.

You can check the calibration of the display by enabling the 0.2 s x 0.5 mV grid, and then by measuring the spacing of the grid lines. If the spacing is incorrect, your X server does not know the actual display resolution. See your X server documentation if this is the case, or use the `-dpi` option when starting WAVE (start WAVE with no arguments for instructions).

Draw: [▽] Use this control to toggle WAVE’s signal display mode. By default, WAVE displays all signals in order of signal number, with signal 0 at the top of the signal window. If you select listed signals only, WAVE displays only those signals that appear in the signal list (in the Analyze window, from top to bottom in the order in which they appear in the signal list).

Show annotations: ▽ Use this menu button to choose how annotations are to be displayed. By default, WAVE displays annotations in the center of the signal window. If you select attached to signals, each annotation appears near the signal specified by its `chan` field. If you select as a signal, WAVE draws a signal derived from the `num` fields of any annotations in the window, in place of the standard annotation display.

Time display: ▽ By default, WAVE displays elapsed time from the beginning of the record in hh:mm:ss format. Use this control to select display of absolute time (if defined for the record), or to display time in sample intervals. If you select absolute time display, you may enter absolute times in the Find window.

Grid: ▽ Use this menu button to select a grid style, or to suppress the grid display.

(Undo changes) This button cancels any changes you have made to the View window settings and restores the indicators to reflect the current settings.

(Redraw) This button causes WAVE to accept any changes you have made in the View window. Pressing (Redraw) dismisses the View window and refreshes the signal window.

(Save as new defaults) This button preserves the current settings within the View window in your home `.Xdefaults` file.

A.8 The Edit menu

Open this menu using (Edit ▽) in WAVE's main window.

Allow editing This selection allows you to make changes to the annotation buffer.

View only This selection disallows annotation editing. You may still edit '<', ':', and '>' markers.

A.9 The Properties menu

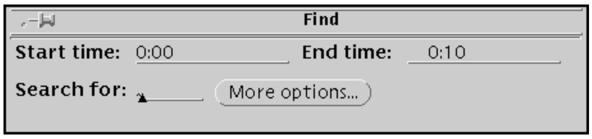
Open this menu using (Properties ▽) in WAVE's main window.

Signals... This selection pops up a window containing information about the signals of the current record, obtained by running `dbdesc`.

Annotations... This selection pops up a window containing a summary of the contents of the annotation buffer, obtained by running `sumann` (after saving any edits).

About Wave... This selection pops up a window containing the version number and date of this copy of WAVE. This window may also contain news about recent changes in WAVE.

A.10 The Find window



Open this window using (Find...) in WAVE's main window.

Start time Specifies the time of the sample shown at the left edge of the signal window, in the format specified by the Time display item in the View window. Go to any other part of the record by entering the time in this field.

End time Specifies the time of the sample shown at the right edge of the signal window, in the format specified by the Time display item in the View window. Go to any other part of the record by entering the time in this field.

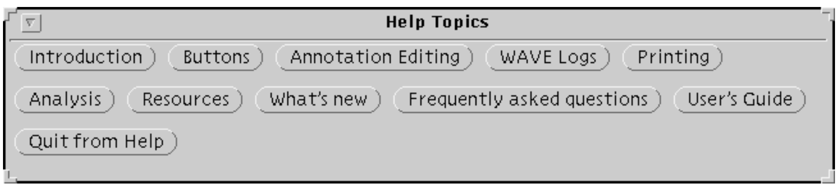
Times can be entered in `h:m:s` format, with hours or hours and minutes omitted, or in `s`*nnnnn* format, in which *nnnnn* is a number of sample intervals from the beginning of the record.

Search for Specifies a target for (< Search) and (Search >). Changing this field causes an immediate forward search. The contents of this field should match an annotation or marker mnemonic, signal quality code, rhythm, comment, or other text string, or one of the following:

`*v`	matches any ventricular ectopic beat
`*s`	matches any supraventricular ectopic beat
`*n`	matches any other beat type
`*`	matches any annotation or marker
`.`	matches a deletion made during this WAVE session

(More options...) This button clears the contents of the Search for field and opens the Search Template window (see section A.13, page 76).

A.11 The Help Topics window



Open this window using (Help) in WAVE's main window.

Each of the buttons in this window, except for the last two described below, opens a help file in a scrollable text window. Use the (Print) button in each window to get a paper copy of the help file if you wish. The on-line manual for WAVE includes many of these files verbatim.

(Introduction) A quick introduction to WAVE's features.

(Buttons) A summary of the functions of the buttons in WAVE's main control panel and pop-up windows.

(Annotation Editing) A summary of annotation editing procedures.

(WAVE Logs) How to create and review WAVE log files.

(Printing) Methods for printing annotated signals from WAVE.

(Analysis) Using and customizing the Analyze window.

(Resources) WAVE-specific X11 resources.

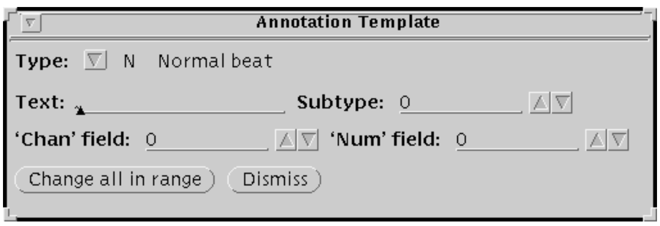
(What's new) Recent changes in WAVE.

(Frequently asked questions) Common questions about WAVE (answers, too!).

(User's Guide) Click on this button to begin reading this guide in a web browser window. By default, WAVE uses Netscape 1.1 or any later version available on the WAVE host. To configure WAVE to use a different browser, see section 3.10, page 43.

(Quit from Help) Press this button to dismiss the Help Topics window.

A.12 The Annotation Template window



Open this window at any time by clicking left anywhere within the signal window. Use the data fields in the Annotation Template before inserting an annotation, to specify the characteristics of the annotation you wish to insert.

Type: [▽] This field specifies the type of annotation to be inserted. It may be changed by selecting a new value from the pull-down menu, by typing the mnemonic while the pointer is within the signal window, or by selecting an existing annotation and pressing the [Copy] or [F6] keys (these key commands also copy the other fields of the selected annotation into the corresponding fields of the Annotation Template).

Text This field specifies the contents of the optional annotation `aux` field. It may be changed by typing into it directly or by selecting an existing annotation and pressing the [Copy] or [F6] keys. In most cases, it should be empty, but it must be filled in for rhythm and certain other non-beat annotations. When the annotation is written, WAVE prefixes the required byte count to this field before transferring it to the `aux` field.

Subtype This field specifies the contents of the annotation `subtyp` field. In most cases, it should be 0; legal values range from -128 to +127.

'Chan' field This field specifies the contents of the annotation `chan` field. In most cases, it should be 0; legal values range from -128 to +127. In multi-edit mode, the `chan` field of the annotation indicates the signal number of the attached signal. When inserting annotations in multi-edit mode, the value of the `chan` field in the annotation is determined by which signal is nearest to the pointer when the insertion is performed, and the 'Chan' field in the Annotation Template is updated accordingly.

'Num' field This field specifies the contents of the annotation `num` field. In most cases, it should be 0; legal values range from -128 to +127. If you have chosen to Show annotations: as a signal in the View window, the `num` fields of the annotations determine the signal amplitudes.

(Change all in range) This button changes all annotations between the '<' and '>' markers (the Start and End times in the Analyze window) to match the Annotation Template.

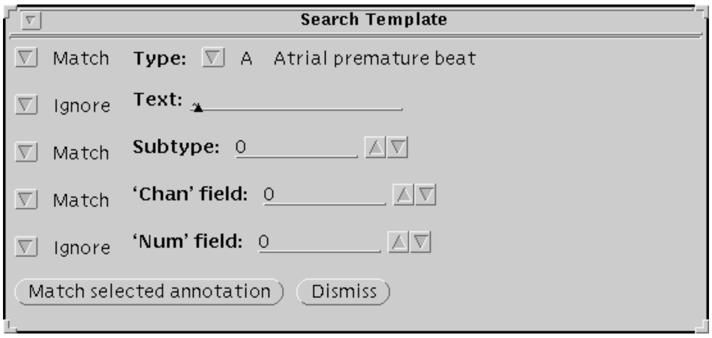
(Dismiss) This button makes the Annotation Template window disappear (until it is recalled by clicking left while the pointer is within the signal window).

A.13 The Search Template window

The Search Template *is not available in* WAVE *5.0.*

Open this window by clicking left on (More Options...) in the Find window. This window specifies criteria for searches performed using (< Search) and (Search >). Each of the five menu buttons at the left edge can be set to Ignore (the default) or Match in order to exclude or include, respectively, the associated annotation field in the search criteria.

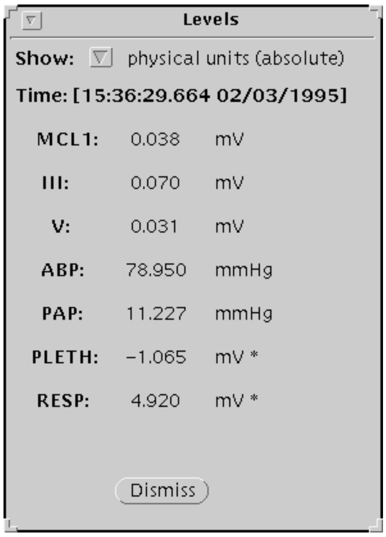
The Type, Text, Subtype, 'Chan' field, and 'Num' field controls work in the same way as their counterparts in the Annotation Template window (see section A.12, page 75).



Match selected annotation This button copies the fields of the selected annotation (if any) into the Search Template, and sets all five of the menu buttons at the left edge of the window to Match. This is a convenient shortcut if you wish to find another annotation identical to a given one.

Dismiss This button makes the Search Template window disappear (until it is recalled by clicking left on More Options... in the Find window).

A.14 The Level window



Open the Level window by choosing level in the View window, then by clicking a mouse button anywhere in the signal window.

The Level *window is not available in* WAVE *5.0*

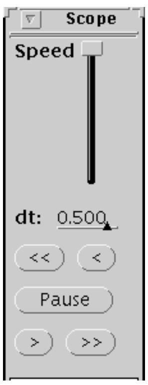
Show ▽ Choose how time and signal levels are displayed in the Level window using this menu. The four menu choices are physical units (absolute), physical units (relative), DB units (absolute), and DB units (relative). Physical units are seconds for time, and as specified in the record's header file for each of the signals. DB units are sample intervals and analog-to-digital units (adu). In relative mode, all measurements are shown as differences beween the current location and the reference location (which is determined by the placement of the '`;`' reference marker).

Time This item shows the time corresponding to the most recent cursor position while a button was depressed within the signal window (and to the signal levels shown in the remainder of the Level window). Depending on the choice made using the Show menu, this item may be named Interval or Sample number.

Levels Most of the Level window contains an informational display indicating the names and levels (amplitudes) of the signals. If an asterisk (*) appears at the end of any line, it indicates that the corresponding signal is uncalibrated.

(Dismiss) This button makes the Level window disappear (until it is recalled by clicking within the signal window).

A.15 The Scope window



Open the Scope window from the Analyze window, using (Show scope window). The Scope window provides a simulated oscilloscope display of one signal, triggered by QRS (beat) annotations. (Annotations that are not considered QRS annotations, such as rhythm changes and comments, are those that appear displaced vertically with respect to the QRS annotations in the signal window; these non-QRS annotations do not trigger the scope display.)

Select the signal to be drawn in the Scope window using the Signal control in the Analyze window. The Time scale: [▽] and Amplitude scale [▽] controls in the View window define the scales used in the Scope window.

Speed This slider controls the playback speed. Click left on the slider and drag it to the desired position – lower to reduce speed, higher to increase speed.

dt This field specifies the interval from the left edge of the Scope window to the annotated sample, in seconds. Negative values may be used to shift the left edge of the scope window past the annotated sample. Drag the resize corners to change the interval between the left and right edges of the scope window.

(<<) This button displays frames continuously in reverse order. Use (Pause) to interrupt the display. The display stops when it reaches the beginning of the record, an index mark ('`:`'), or the '<' marker.

< This button displays the previous frame.

Pause This button freezes the scope display. It also forces the signal window to be redrawn, roughly centered on the time shown at the bottom of the scope window, with the current annotation selected.

> This button displays the next frame.

>> This button displays frames continuously in normal order. Use Pause to interrupt the display. The display stops when it reaches the end of the record, an index mark (‘:’), or the ‘>’ marker.

Appendix B

System Requirements for WAVE

This section includes a shopping list of hardware and software that you will need in order to run WAVE, or that you may find useful in order to use WAVE most effectively.

B.1 Necessities

At a minimum, you will require:

WAVE software This is available in binary form from MIT for SPARC-based systems running Solaris 2.x or SunOS 4.1.x, and for PCs running Linux. (See section C.1, page 87 for details.)

A computer capable of acting as a WAVE host (one for which a binary version of WAVE is available). Virtually any PC with a 386 or better CPU can run Linux, and such systems are likely to be the least expensive choice. Ideally, a Linux PC to be used as a WAVE host should have at least 8 Mb of RAM, at least 200 Mb of available disk space, a three-button mouse (or trackball), and a graphics card and monitor (17-inch or larger, with a dot pitch of .26 mm or less) capable of non-interlaced display at 65 Hz or faster with a resolution of at least 1024x768 with 256 colors. In most cases, you will also want the system to be equipped with a CD-ROM drive (for loading software and digitized signals) and an Ethernet adapter or a modem. (Check the Linux Hardware-HOWTO to be sure that your chosen hardware is supported. Most graphics cards, including the popular accelerated cards, are fully supported by the standard XFree86 X server, but a few high-end models are supported only in SVGA compatibility mode. Commercially available X servers for Linux support the accelerated modes of many high-end models.) In early 1997, it was possible to assemble a suitable Linux PC for about US$1000 (not including the monitor). It is not unreasonable to budget an equal amount for a good monitor, since WAVE's usability depends to a significant extent on being able to see its output clearly. If your budget permits you to spend more, buy extra RAM rather than a faster CPU. Although Linux does not require large amounts of RAM, it can use additional RAM very effectively, and you are likely to find that purchasing (say) 16 or 32 Mb of additional RAM results in a bigger performance improvement than spending the same amount on a faster CPU. Fully configured and supported Linux PCs are available from many vendors if you prefer not to assemble your own; see the Linux Commercial-HOWTO for further information.

SPARC-based systems available from Sun and other vendors are slightly more expensive than comparable Linux PCs.

A source of digitized signals Desktop SPARC systems are equipped with a single-channel analog-to-digital converter that may be adequate for some applications (newer versions have high-speed two-channel 16-bit ADCs). Several vendors supply data acquisition subsystems for SPARC systems, which may be useful if you require more speed, resolution, or inputs. PC-based hardware and software for digitizing signals is available at significantly lower cost from many vendors. The DB Software Package includes MS-DOS software for converting signals from analog to digital form and back again, using ISA-bus ADC boards from Microstar Laboratories (see section B.5, page 83). Digitized reference recordings of ECGs and other signals are available on CD-ROMs from MIT and others; several CD-ROMs are currently available for use in basic research in physiologic signal processing as well as for evaluation of instrumentation. Single CD-ROM drives are widely available for prices ranging from US$50 to $300, depending on speed; CD-ROM changers and jukeboxes are also available at higher prices.

B.2 Printers

All currently available WAVE-compatible applications for printing require the use of a PostScript (or compatible) printer accessible to the WAVE host system. If you need paper output, therefore, you should obtain a PostScript printer, or a PostScript-compatible interpreter (such as the freely available GNU Ghostscript) for your printer, or you should be prepared to write your own printing applications. For users of SBus-based SPARC systems, the Sun SPARCprinter is an excellent choice (less expensive and much faster than standard PostScript printers, with better quality output).

B.3 Remote access requirements

In order to use WAVE, you may log onto a WAVE host directly, or you may do so remotely using virtually any local system (computer or X terminal) connected by TCP/IP-based Ethernet to a WAVE host. (PPP, SLIP, or TERM serial-line connections also work, and may be acceptable for viewing data, but even 28.8 Kbaud connections are likely to be intolerably slow for extensive annotation editing.) Thus it is possible for several users to access WAVE simultaneously using a single WAVE host, provided only that each user's computer or terminal runs an X server. X servers are available for virtually any computer in current use, including those running any version of UNIX, as well as for MS-DOS and MS-Windows PCs, Macintoshes, Amigas, and DEC VAXen running VMS.

For many users seeking to add an additional point of access to WAVE, a networked PC may be attractive. With current CPUs and graphics adapters, rendering speed is much faster than was available on far more expensive workstations only a few years ago. X server software for MS Windows 3.1/95/NT is available from many vendors, and may be a good choice if you are heavily invested in Windows-based software. (For less than the cost of a Windows X server, however, you can buy a 1 Gb disk drive and install Linux, XFree86, and WAVE on it. Consider this before choosing a commercial X server for Windows.) If costs of maintenance and system administration are a concern, an X terminal is probably a better choice than a networked PC.

B.4 X11 window managers

Unlike other windowing systems, X does not impose a particular 'look and feel' on the user; rather, the user may select one of a number of 'window managers' that provide different user interfaces for common operations such as moving and resizing windows. Window managers are X client applications that have a special status allowing them to manipulate the windows of other X clients. Any window manager that conforms to the ICCCM (a standard published by the X Consortium) may be used with WAVE; among these are `fvwm`, `olwm`, and `olvwm`, included in source form with most Linux distributions and widely available elsewhere (`olwm` is also standard on Sun workstations); and `mwm` (the Motif window manager, included with most current commercial UNIX systems). The Open Look window managers (`olwm` and `olvwm`) are particularly recommended, since the others do not support certain Open Look features of WAVE (notably spot help and 'pinnable' popups). If you are using an MS-DOS PC or an X terminal to access WAVE, the window manager must usually be run remotely, often on the same computer that runs WAVE.

B.5 Data acquisition and digitization

PC-based data acquisition is a good choice given the large number of ADC boards available; note, however, that Linux drivers for these boards are rare. The DB Software Package available separately from MIT includes an MS-DOS program (named `sample`) for digitization and replay of analog signals using a DAP 1200- or 2400-series data acquisition board (available from Microstar

Laboratories, 2863 152nd Avenue NE, Redmond, WA 98052 USA; telephone +1 206 881 4286).

You may also find that an existing database of digitally recorded signals may be useful for your studies. Four such databases are currently available on CD-ROM (the MIT-BIH Arrhythmia Database, the European ST-T Database, the MIT-BIH Polysomnographic Database, and the MGH/MF Waveform Database), and several more databases in this format are in preparation. These disks are compatible with any CD-ROM drive.

Note that the WAVE host must be able to read the signal files. If you use an MS-DOS data acquisition program, this can be accomplished in several ways:

- The data acquisition PC can be set up to boot either Linux or MS-DOS, and when running WAVE under Linux, signal files written to MS-DOS file systems can be read directly. (Newer versions of Linux can also read files in Windows NT, Windows 95, and OS/2 file systems.)

- Alternatively, you can transfer the digitized signals to another system acting as the WAVE host, either during digitization (by writing to a network drive) or afterwards. Since signal files are usually very large, Ethernet transmission is recommended. To set up a minimal Ethernet including a Sun workstation or Linux PC and an MS-DOS PC, you will need an Ethernet card for each PC (usually around US$100; Sun workstations have built-in Ethernet adapters but may require external transceivers, usually around $50), TCP/IP software for the PC (such as Sun's PC-NFS, about $300), and miscellaneous cabling and terminators ($20 or less). For a truly minimal network of only two systems, see the Linux Ethernet-HOWTO for details on how to connect two UTP interfaces without using a hub. You will need a hub (US$150 or more) to connect more than two systems, unless you are able to use the older 'thin' (10 Base 2) Ethernet. If you use `sample` for digitization, signal files can be written directly on the host system's drive at rates up to 80,000 samples per second using PC-NFS (or up to 100,000 samples per second on local disks). This configuration allows you to examine the signal files using WAVE as the data are being acquired; although WAVE does not currently support real-time, continuously updated display of incoming data, it is not difficult to create a log file for WAVE that can be used to drive a continuously updated time-delayed display (see section D.6.9, page 108 for details).

- You can write digitized data to a removable disk, then read it on the WAVE host. Devices such as IOmega's 100 Mb Zip or 1 Gb Jaz drives are suitable for this purpose; the SCSI versions of these devices are usable under Linux, Solaris, and SunOS. For high-speed or high-volume digitization, the Jaz drive is a good choice. An advantage of this approach is that the removable disks are inexpensive enough to be used for archival storage of the data.

B.6 About Linux

As mentioned earlier, an excellent choice for a WAVE host is a PC running Linux. Linux is a (very) complete, and completely free, robust, modern reimplementation of the UNIX operating system, written by Linus Torvalds and a cast of thousands. It is available in source and ready-to-run form by anonymous FTP from many sources, including `tsx-11.mit.edu`, `sunsite.unc.edu`, and `ftp.funet.fi`. You can also obtain Linux on CD-ROMs from many commercial sources, generally at very low prices (typically US$10 to $40, depending mainly on the amount of printed documentation and technical support offered). Current Linux distributions include X11R6, XView 3.2, `olwm` and `olvwm`, TCP/IP networking including NFS support, the complete collection of GNU software including the GNU C/C++ compiler, Ghostscript, TeX, and much more. For further information, visit the home page of the Linux Documentation Project (`http://sunsite.unc.edu/mdw/`), or get the Linux INFO-SHEET by anonymous FTP from `tsx-11.mit.edu` (`pub/linux/docs/INFO-SHEET`) or from `sunsite.unc.edu` (`pub/Linux/docs/INFO-SHEET`). The other Linux HOWTO documents mentioned earlier can be obtained from these sources as well.

Since mid-1995, Linux has supported ELF binaries, and current versions of WAVE for Linux are made available in this format. Older versions of Linux used the `a.out` binary format. If you have been running one of these older versions of Linux, updating your system to one with ELF support is highly recommended. Until you do so, you can use the Linux `a.out` WAVE version 5.4 binary available from `ecg.mit.edu` (see section C.1, page 87) .

B.7 Other useful software

WAVE makes extensive use of the *DB Software Package*, available from the same sources as WAVE itself (`http://ecg.mit.edu`). The DB Software Package includes `calibrate`, `dbcollate`, `dbdesc`, `dbwhich`, `mrgann`, `plot2d`, `pschart`, `psfd`, `rdann`, `rdsamp`, `sample`, `snip`, `sqrs`, `tach`, `wrann`, `wrsamp`, and `xform` (among many other applications). The package also includes the DB library of interface functions for user-written applications that read and write signal and annotation files in the formats supported by WAVE.

A web browser, though not a necessity for everyone who uses WAVE, should be part of your software toolbox. If you use link annotations, you will need a browser in order to follow the links to the external data. Even if you don't anticipate using link annotations, you can still use a web browser to view the on-line version of this guide. Netscape is an attractive choice because of its simple remote-control interface, among other reasons. As noted earlier, free copies of Netscape for academic use or for evaluation purposes may be obtained by anonymous FTP from `ftp.netscape.com`).

An X-Y plotting program capable of being run from the command line is another nearly essential tool. A suitable program should be able to accept multicolumn text input, allowing specification of which columns to plot from the command line (so that it can be driven by WAVE). Avoid commercial software designed for business graphics with arbitrary (often very low) limits on the number of points that can be plotted. If you expect to prepare plots for publication, avoid packages that only provide screen dumps; any decent plotting program should be able to generate plots at the resolution of your printer. `plot2d` (included in the DB Software Package) is a bare-bones command-line front end to `gnuplot`, a fairly capable interactive plotting program that is included in most Linux distributions, and is also available by anonymous FTP from `prep.ai.mit.edu` and numerous other sites. An interesting alternative is `xvgr`, available from `ftp.teleport.com/pub/users/pturner/acegr` and in most Linux distributions. WAVE and `xvgr` use the same Open Look/XView user interface.

Appendix C

Setting up a WAVE host

C.1 Obtaining and installing the current version of WAVE

Current (demo) versions of WAVE for SPARCstations running SunOS and Solaris, and for PCs running Linux, may be downloaded via the Internet. Point your Web browser to `http://ecg.mit.edu` for details, or use anonymous FTP to get `pub/software/README.DOC` from `ecg.mit.edu` and follow the instructions there for downloading and installing a demo version of WAVE, and the DB Software Package. (Demo versions of WAVE do not permit you to save the results of annotation editing operations, but they are otherwise fully functional.) Fully functional WAVE binaries are also included on some of our CD-ROMs, in the `software` directory. Follow the instructions in `software/README.DOC` to install WAVE from any of these CD-ROMs.

The second edition of the MIT-BIH Arrhythmia Database CD-ROM contains WAVE *version 5.0 binaries for SPARC SunOS 4.1.x, within the '*`src/db/wave`*' directory. The CD-ROM also contains the second edition of this guide, which documents version 5.0, in both LaTeX source and PostScript formats. Follow the instructions in* `readme.doc` *for installing the DB Software Package on your system. Doing so will install* WAVE *5.0 as well.*

Note that WAVE cannot be run at all until the DB library (contained within the DB Software Package) and the XView libraries (`libxview.so.3` and `libolgx.so.3`) have been installed. The standard installation script installs the DB library in all cases, and (on Linux PCs) the XView libraries if they are not already installed. (The XView libraries are preinstalled by Sun on its current workstations.) It is strongly recommended that you install the complete DB Software Package, since WAVE uses many of the applications included in this package.

To confirm that all necessary libraries are available on your system, type

```
ldd /usr/local/bin/wave
```

after installing WAVE and the DB Software Package. If no libraries are reported as missing, the installation is complete. Test it by working through the exercise in chapter 1.

If you would like to try porting WAVE to another operating system for which X11 client support (Xlib) is available, please obtain the XView source distribution and attempt to port 'cmdtool' (a simple terminal emulator included in the XView distribution) first. If you are successful, write to the author (george@hstbme.mit.edu) for information on obtaining a current set of WAVE sources. Compared to the original sources, the Linux version of XView (freely available by anonymous FTP from ecg.mit.edu, sunsite.unc.edu, tsx-11.mit.edu, and their mirrors, and also available on many low-cost Linux CD-ROM archives) is easier to port to another operating system, since many Sun-specific dependencies were removed. Nevertheless, porting XView is likely to require significant effort.

C.2 Setting up a printer for WAVE

WAVE's printing capabilities (see chapter 4, page 47) require that the WAVE host be able to print PostScript. Your WAVE host may already be set up to do this if its default printer is a PostScript printer. (Under Solaris, make sure that the directory containing the BSD command `lpr` is in your `PATH`.)

If the printer itself is incapable of rendering PostScript, you may still be able to use it if it is supported by Ghostscript (a freely available PostScript interpreter). Ghostscript is included in executable and portable C source form in most if not all Linux distributions, and is also available by anonymous FTP from many sources, including prep.ai.mit.edu and other archives of GNU software. Ghostscript supports a wide variety of ink jet, dot matrix, and laser printers including popular models made by Canon, Epson, Hewlett Packard, IBM, and others; it can also render PostScript into files in a wide variety of graphics formats, including PBM, PGM, PPM, PCX, and TIFF (including G3 fax format – with a fax modem, you can even print PostScript output on any fax machine). Ghostscript works by rasterizing Postscript input on the host system, then transmitting the raster image to the printer in whatever format the printer accepts. For details on setting up Ghostscript for your printer, see the documentation that comes with Ghostscript, or the Linux Printing-HOWTO.

Assuming that the WAVE host is somehow able to print PostScript output (either directly to a PostScript printer, or via Ghostscript or another interpreter), setup for WAVE is straightforward:

- Set the `PRINTER` environment variable to the name of the printer you wish to use, if that printer is not the default printer on the WAVE host.
- Set the `PSPRINT` environment variable to the command needed to print PostScript from the standard input on the desired printer. WAVE assumes that this command is '`lpr`' if neither `PSPRINT` nor `PRINTER` are set; if only `PRINTER` is set, WAVE assumes that this command is '`lpr -P$PRINTER`'.
- Set the `TEXTPRINT` environment variable to the command needed to print ordinary text from the standard input on the desired printer. (This command is used, for example, to print the text of help topics.) WAVE assumes that this command is '`lpr`' if neither `TEXTPRINT` nor `PRINTER` are

set; if only `PRINTER` is set, WAVE assumes that this command is '`lpr -P$PRINTER`'.

- Finally, if your printer is capable of resolution higher than 300 dpi, you may also wish to add appropriate `-d` options to the `pschart` and `psfd` commands in WAVE's menu file (see section 4.2, page 50, for details). (This is not necessary in order to obtain properly-scaled output, but this will affect the amount of fine detail in your plots, and the thickness of the plotting lines.)

In most cases, you will want to add the commands needed to set these variables to your `.login` or `.profile` file on the WAVE host, so that they are executed automatically whenever you log in. For details on setting environment variables, see the discussion of the `DISPLAY` variable in section 1.1, page 3.

If you are using WAVE remotely, you may prefer to use a local printer for your output. WAVE has no specific support for doing so, but this can often be arranged if, for example, your own computer is able to make its printer available for printing from the WAVE host (this is can be done easily if your computer runs any version of UNIX, or with somewhat more difficulty if it runs Microsoft Windows). In this case, usually all that is needed is to set `PRINTER` to the name of your printer (as it is known to the WAVE host). Another possibility is to set `PSPRINT` and `TEXTPRINT` to commands that capture output in a file, and then transfer the file to your own computer (via FTP or any other method) for local printing. For example, you might set `PSPRINT` to '`cat >>output.ps`' and `TEXTPRINT` to '`cat >>output.txt`'. These commands accumulate all PostScript output into `output.ps`, and all text output into `output.txt`, so you should delete these files from the WAVE host after copying them to your computer, so they don't grow indefinitely.

If you need to change printers or print commands while running WAVE, this can be easily accomplished within WAVE's Print setup window (see section A.4, page 68), which may be opened by selecting Print setup from the (File ∇) menu.

Appendix D

Frequently Asked Questions

This is the WAVE FAQ (frequently asked questions) list. If your version of WAVE is newer than this guide, you may wish to review the on-line copy of this FAQ (click on (Frequently Asked Questions) in WAVE's Help window).

If your question is not on this list, and you think it should be, please send it to me (`george@hstbme.mit.edu`)!

D.1 Hardware questions

D.1.1 Can I use WAVE if I don't have a SPARCstation?

If you have a 386, 486, or Pentium-based PC with a VGA card, mouse, at least 8 Mb of RAM and 100 Mb of available disk space, you can run WAVE under Linux, a freely available UNIX clone for the PC.

If you have any networked computer that can run X11R4 or a later version (this includes all current UNIX workstations, PCs, Macintoshes, and a variety of other systems), and access via network to either a SPARCstation or a PC running Linux, you can run WAVE remotely (see the next question).

Otherwise, you're out of luck at present. WAVE requires X11R4 or later and the XView toolkit (both are freely available in source form for UNIX systems). See Appendix C, page 87, for details if you would like to try porting WAVE to another version of UNIX.

Other DB applications (WVIEW for Microsoft Windows, VIEW for MS-DOS, MacView for the Macintosh, `dbtool` for SunView, `dbplot` for a variety of graphics terminals and plotters under UNIX, and `pschart` and `psfd` for PostScript devices) offer many of the display features of WAVE in other environments; none of these currently support annotation editing or control of external programs, however.

D.1.2 How can I use WAVE from an X terminal, PC, Mac, etc.?

As with any X application, you can use an X terminal or any system equipped with an X server to interact with a copy of WAVE running on another computer connected to the same network. To do so, however, requires a few additional steps:

- Find out the host names of the two systems. On a UNIX system, the '`hostname`' command will give you this information. If your display is connected to a PC or other non-UNIX system, you can try remotely logging onto the UNIX system on which WAVE resides, and using the '`who`' command to discover the name of the system in front of you.
- Assume that your system is named `hither` and that WAVE resides on `yon`. You may need to add `yon` to the X server access control list on `hither`, in order to make it possible for X clients on `yon` to open windows on your screen. If `hither` is a UNIX system, you can accomplish this by typing (on `hither`) '`xhost +yon`'. (Note: if `hither` doesn't recognize `yon` by name, this command will have no effect. In this case, use `yon`'s IP address in place of its name.)
- Remotely log in to `yon` (for example, via `telnet`), and set the `DISPLAY` environment variable on `yon` to point back to your display. If you use the C-shell on `yon`, this will normally be done by '`setenv DISPLAY hither:0.0`'. Users of `sh`, `ksh`, or `bash` should type '`DISPLAY=hither:0.0; export DISPLAY`'. (Note: if `yon` doesn't recognize `hither` by name, this command won't work. In this case, use `hither`'s IP address in place of its name.)
- Check the connection by starting an X11 client such as `xterm` on `yon`. A new window should appear on your display within a few seconds. (See the documentation for your X server if this doesn't work).
- Once you have succeeded in the test in the previous step, start WAVE in the usual way ('`wave -r` *record* ...') on `yon`. See "Where do I find the missing fonts?" (section D.2.2, page 95) if your X server complains about missing fonts. See "I can't find the file named *record*!" (section D.4.1, page 99) if WAVE complains about missing files.

X servers are standard on all current UNIX workstations, but not on PCs or Macintoshes. There are many MS Windows-hosted X servers available commercially, as well as a smaller number of MS-DOS and Macintosh-hosted X servers.

D.1.3 Can I run WAVE remotely using a modem?

It is possible, with appropriate (SLIP, PPP, or TERM) software, to run WAVE across a serial connection (see the previous question for details), but this will be painfully slow even with a 28.8 kbps modem. Using WAVE remotely via Ethernet is usually quite tolerable, however.

D.1.4 How can I insert annotations using a two-button mouse?

Normally, the middle button is used to insert annotations, so the problem is to simulate a middle button click if you don't have a middle button. Most, if not all, X servers that support two-button mice provide a way to do this. The most common method is to press both buttons simultaneously (an operation sometimes called "chording," by analogy with playing a chord on the piano). Typically, you have an adjustable window of 50 to 100 milliseconds in which to press both buttons, so that you don't need exactly simultaneous clicks (which would be impossible to achieve reliably). Some X servers delay reporting a button-down event until a button-up event occurs; this approach makes chording less error-prone, but it means that the feedback that WAVE produces in response to button-down events gets delayed until it may no longer be useful. Another approach, more commonly used with one-button mice, is to simulate the middle button (and, if necessary, the right button) using a mouse button and keyboard key combination, such as clicking while pressing the SHIFT or CTRL keys. Check your X server's manuals for information about which of these methods is supported. With a little practice, annotation editing can be tolerable with a button-impaired mouse.

Independent of the X server, WAVE also makes it possible to simulate a middle button click, by using the F2 key (or the 5 key on the numeric keypad). If you use this technique, you might also wish to use the F3 and F4 keys to drag the pointer and marker bars left or right.

Inexpensive three-button mice, trackballs, and touchpads manufactured by Logitech and many others are widely available for PCs, and are highly recommended if you do much annotation editing. Most are fully compatible with Microsoft two-button mice and with X server software for PCs. Some users prefer trackballs for precision editing since there is no tendency for the pointer to move when clicking the buttons, as with a mouse.

D.1.5 How do I get spot help if I don't have a Help key?

If you are not using WAVE with a Sun keyboard, use the F1 key to open XView spot help for WAVE. If this doesn't work, you may need to use the '`xmodmap`' utility (a standard component of X11, usually found in the same directory as other X clients such as '`xterm`'). Try the following command:

```
xmodmap -e "keysym F1 = Help"
```

If you are now able to use F1 to open spot help, you may wish to include this command in your '`.xinitrc`' (a text file in your home directory; create it if it doesn't exist) so that spot help via F1 is enabled whenever you log in.

If you are not using `olwm` or `olvwm`, spot help may not work properly; in particular, your window manager may pass incorrect information about which window is active when you invoke spot help. There is no general solution to this problem; try using `olvwm` or `olwm`, at least until you are familiar with WAVE's controls.

D.2 Problems starting WAVE

D.2.1 Why won't WAVE run?

Here are a few things to check:

1. The command used to run WAVE must be typed in lower case, as in '`wave -r 100s -a atr`'. Note that letters in record names and annotator names are also usually in lower case.

2. Try typing `wave` with no command-line arguments. You should see a summary of options. If you don't, `wave` has not been installed properly, is not in your `PATH`, or another program named `wave` is in your `PATH`.

   Errors similar to '`wave: can't load library 'libxview.so.3''` indicate an installation problem which is probably shared by other applications that use the same libraries. Use the command '`ldd 'which wave'`' to identify which dynamically linked libraries are missing. Locate these on your system (for example, using a command such as '`find / -name libxview.so.3 -print`'; if any of these libraries cannot be found on your system, it may be downloaded from MIT.) Add the directories in which the missing libraries are found to your `LD_LIBRARY_PATH`. For example, if the missing libraries are located in `/usr/openwin/lib`, and you are using the C-shell, use the command '`setenv LD_LIBRARY_PATH /usr/openwin/lib:${LD_LIBRARY_PATH}`'. Under Linux, if you can obtain root permissions, a more permanent solution is to add `/usr/openwin/lib` to the list of directories in '`/etc/ld.so.conf`', and then to run `ldconfig`.

3. If the previous test works, try typing '`wave -r 100s -a atr`'. (All WAVE distributions come with record 100s.) One of the following should happen:

   - If you see a summary of options, rather than the WAVE main window, read the message carefully; you may not have set the `DISPLAY` or `DB` environment variables, or the X server may not be running.
   - If you see a lengthy error message referring to missing fonts, see "Where do I find the missing fonts?" (section D.2.2, page 95).
   - If you see neither a summary of options nor the WAVE main window, the `DISPLAY` environment variable may be set incorrectly, the X server may be refusing permission to open a window, or your window manager may be waiting for you to specify the location or size of the window to be opened.

- If you see only a message box reading 'Record 100s is unavailable', your DB environment variable may be set incorrectly; in this case, find the directory (probably /usr/local/database) containing a file named '100s.hea' and append the name of that directory to the value of DB.
- If WAVE's signal window appears, but is solid white, your X server has a bug. Click on any of the navigation controls (e.g., (<<), (<), (>), or (>>)), or resize the window, to make the signals appear. If the signal window is solid (or nearly solid) blue or black, or has only horizontal lines across it, see "I can't see the signals!" (section D.3.1, page 96).
- If the signals appear, but there are no annotations or time stamps in the signal window, your X server has a (different) bug. Restart WAVE, adding the '-S' option to the 'wave' command. Also see "I can't see text in the signal window!" (section D.3.2, page 97).
- If you see a correct display, your problem is specific to the record you were previously trying to view. Find the directory containing the header file for the desired record, and append the name of that directory to the value of DB. If the record is on a CD-ROM, or was copied from a CD-ROM, its header file is '*record*.hea' (where *record* is the record name); otherwise, its name is probably 'header.*record*'.

D.2.2 Where do I find the missing fonts?

WAVE is an XView application, and requires the Open Look cursor font on the server. If your server doesn't have this font ('olcursor.snf' for X11R4 and earlier servers, 'olcursor.pcf' for X11R5 and later servers), you must install it before WAVE will run. XView applications also use the 'olgl??.snf' or 'olgl??.pcf' glyph fonts; if these are missing, WAVE will run but certain standard Open Look graphical elements will not have the correct appearance. To install these fonts on your X server's system:

1. Generate copies of 'ol*.snf' or 'ol*.pcf' for your X server system's architecture.
2. Copy 'ol*.snf' or 'ol*.pcf' to a world-readable directory on the server machine.
3. Update the font database on the server machine.
4. Force the server to reread the font database.

You should be able to find 'ol*.snf' or 'ol*.pcf' files on the machine on which WAVE resides (probably in /usr/lib/X11/fonts/misc). The 'ol*.pcf' files are portable (although usually optimized for the architecture of the system on which they are installed). If you need to use 'snf' fonts, however, and the server machine is not of the same architecture, it will be necessary to find '`ol*.bdf`' in the XView distribution and use '`bdftosnf`' to generate '`.snf`' files for your X server system's architecture (see the man page for '`bdftosnf`'; this step can be done trivially on the server machine, if '`bdftosnf`' is available there, or with a little more trouble on the remote machine). If the server machine runs UNIX, read the man pages for '`mkfontdir`' and '`xset`' to see how to update and reread the font database; otherwise, consult the documentation for your X server. Note that (under UNIX) you do not need to have write permission in the standard font directory in order to perform these steps; if you add your own font directory to the font path using '`xset`', however, remember that this setting lasts only until the server exits and will need to be repeated afterwards.

D.3 Display-related questions

D.3.1 I can't see the signals!

If the signal window is solid white, your X server has a bug. Click on any of the navigation controls (e.g., (<<), (<), (>), or (>>)), or resize the window, to make the signals appear.

If the signal window is solid or nearly solid blue (or black if you have a monochrome or greyscale display), the signals are too big. WAVE may be attempting to display signals that fill the entire window. In rare cases, noise in the signals themselves can produce this effect. More common causes include incorrect signal format or gain specifications in the header file for the record you have opened, incorrect display calibration, or a choice of amplitude scale that results in excessive vertical range of one or more signals. This effect can also occur as a result of various problems related to the DB calibration file, for example, if the `DBCAL` environment variable does not name an accessible DB calibration file (see `dbcal(5)` for details), if any of the signals in the record are of types not listed in the DB calibration file, or if the DB calibration file does not contain appropriate default display scales for all of the signal types in the record. See the next several questions for suggestions on correcting these problems.

If the signal window contains horizontal lines (in blue if you have a color display) where the signals should appear, the signals are too small. In this case, the signal gains specified in the header file may be incorrect, the display calibration may be incorrect, or you may have specified an amplitude scale that reduces the vertical range of the signals to zero. Click on (View...) to check the scales, and adjust them if necessary.

If the signal window contains time indicators in the lower corners, look for the signal names along the left edge of the window. If there are no signal names visible, click on (View...), then on signal names and (Redraw) in the View window. If signal names still do not appear, either the header file or the signal file for the record you have opened may be inaccessible. Find these files (see the next question), add the directory that contains them to your DB path, and try to read a few samples using '`rdsamp -r` *`record`* `-t 1`'. If this fails, examine the header file for the record (a text file), and be sure that it specifies at least one signal in an accessible signal file.

D.3.2 I can't see text in the signal window!

This may result from an X server bug. This bug appears to be limited to X servers that support 'true color' (typically 24- or 32-bit) displays, when using overlay graphics. Restart WAVE, adding the '`-S`' option to the end of the '`wave`' command.

This problem may also occur on 8-bit 'pseudo-color' displays, if another application such as Netscape has already allocated most of the color map. To avoid this problem, start WAVE before starting Netscape.

D.3.3 WAVE doesn't draw/erase properly in the scope window!

This problem may result from several different X server bugs. You may be able to avoid it by switching from overlay graphics mode (the usual default) to shared color mode (selected using WAVE's `-S` command-line option), or vice versa, since WAVE uses different techniques for drawing in the scope window depending on the display mode.

Some X servers have bugs that interfere with both of the scope display methods used by WAVE *5.0. If changing display modes doesn't help, try using a current version of* WAVE.

D.3.4 How can I get correct display scales?

Use '`xdpyinfo`' (a standard X client that is provided as part of the X core distribution) to check several server parameters. (As with virtually all X clients, it doesn't matter which machine you find '`xdpyinfo`' on; it will query your server from wherever it is run.) '`xdpyinfo`' will report the version and release numbers of your server; you should have version 11, release 3 or later. It will also report the screen size in pixels and the resolution in pixels per inch. Take a minute to verify the screen resolution by direct measurement of the screen with a ruler. If the resolution reported by '`xdpyinfo`' is incorrect, WAVE will not be able to draw waveforms at properly calibrated scales. The sample X11 servers from MIT all support a '`-dpi`' option, which allows you to specify the correct screen resolution at the time the server is started; read the man page for X for details. (If your server does not support this feature, WAVE can be given a similar '`-dpi`' option.)

You can also double-check your display calibration using WAVE itself. Run WAVE, enable the grid display, and set the display scales to their default values (25 mm/sec, 10 mm/mV). (Click on (View...) to pop up the controls for the grid and the display scales; remember that any adjustments you make become effective only after you click on (Redraw) within the View window.) The grid intervals should measure exactly 5 mm in each direction. If they are too small, use a larger value for the '`-dpi`' specification; if the grid lines are more than 5 mm apart, decrease the '`-dpi`' values.

D.3.5 Why are signals too big or too small?

There are several reasons why the size of signals may be incorrect. WAVE determines display scales for signals based on several parameters:

- First, the `DBCAL` environment variable specifies the name of a text file (located in a directory in the DB path) containing the names of many common signals and customary display scales for each (expressed as physical units per centimeter). For example, ECG signals are customarily displayed at a scale of 1 mV per centimeter. If a signal appears at an inappropriately large or small scale, its name may be missing from the calibration file, the `DBCAL` variable may not correctly specify the name of the calibration file, or the `DBCAL` file may not reside in a directory in the DB path. On most systems, the calibration file is '`/usr/local/database/dbcal`', and the `DBCAL` environment variable is set by the same command used to set the DB path (see the discussion about the database environment in section 1.2, page 5). Instructions for adding additional signal names and scales to the calibration file are located within the file itself, and may also be found in the *ECG Database Applications Guide*.

- Second, the header file for the record specifies the gain for each signal (the number of analog-to-digital units per physical unit). If the gain has not been determined, or is incorrect, the size of the signal as drawn by WAVE may be incorrect as well. If you know the physical values that correspond to at least two levels represented in the signal (for example, the top and bottom of a calibration pulse), you can use the '`calibrate`' application to determine the gain and to rewrite the header file. See the *ECG Database Applications Guide* for details. '`calibrate`' can be most conveniently run from WAVE's Analysis window.

- Finally, the display calibration determines the number of pixels per inch (25.4 mm). WAVE usually gets its information about the display calibration from the X server, but the X server may not supply the correct information. See "How can I get correct display scales?" (section D.3.4, page 97) for details on diagnosing and correcting this problem.

D.3.6 How can I change WAVE's default scales or display colors?

The easiest way to make simple changes in WAVE's default display parameters is by using the controls in the View window to configure the display to your liking. If you then click on (Save as new defaults) in the View window, your choices are recorded in your '`.Xdefaults`' file (in your home directory) and will be used for future WAVE sessions. (Be sure that you have made your changes effective by using (Redraw) before (Save as new defaults).)

Colors are also determined by X resources that may be specified in your '`.Xdefaults`' file, but the View window does not include color controls. See section 6.4, page 62 for guidance on setting display colors.

D.3.7 Why do colors in other windows change when I use WAVE?

WAVE queries the X server to determine how many colors it can display, and what methods it can use to do so. If the X server does not answer this query correctly, WAVE may attempt to use a suboptimal or unavailable display mode. If you find that the colors of other windows change when the cursor moves into a WAVE window, you may prefer to use the '`-S`' option to avoid this behavior at a slight (often unnoticeable) cost in speed.

D.3.8 Everything in the Analysis Commands window appears twice!

This problem, which may occur when you are running WAVE under Linux, appears to result from a bug in the Linux version of the XView 'termsw' package. Type the command '`stty -echo`' in the Analysis Commands window, and the problem should go away.

D.4 File-related questions

D.4.1 I can't find the file named '*record*'!

Record names are *not* file names; they are components of file names only. For example, three files belong to record 100s: `100s.hea`, `100s.dat`, and `100s.atr`. There is no file named '`100s`' belonging to the database. See `annot(5)`, `header(5)`, and `signal(5)`, in the *ECG Database Applications Guide*, for information about file types and formats.

D.4.2 Why does WAVE tell me 'Record ... is unavailable'?

In common with other DB applications, WAVE searches for its input signals and annotations in directories named by the `DB` environment variable (see the discussion of the DB path in the *ECG Database Programmer's Guide*). If WAVE is running remotely, remember to set `DB` *on the* WAVE *host* (a default can usually be set for C-shell users by '`source /usr/local/bin/cshsetdb`', or for Bourne shell, Korn shell, and '`bash`' users by '`. setdb`'; don't omit the '`.`'). The directories named by `DB` are those on the WAVE host; thus (unless your DB files are already on the WAVE host) you must either transfer them there, or NFS- or RFS-mount the directory in which they reside onto a suitable point in the file system of the WAVE host. Furthermore, remember that output annotation, log, and print files will be written to your current directory on the WAVE host.

To get the pathname of a DB file, use '`dbwhich`' (see `dbwhich(1)` for details). For example, the output of '`dbwhich header 100s`' is the pathname of the header file for record 100s (usually '`/usr/local/database/100s.hea`'). If '`dbwhich`' can't find the file, neither can WAVE (or any other DB application). In this case, you will need to check and correct the value of the `DB` environment variable. In unusual cases, the DB path may be correct, but you may not have permission to read the file or one of the directories in the path to the file; in such cases, you will need to get the file's owner or your system administrator to give you appropriate permissions.

D.4.3 Where is my annotation file?

When you edit an annotation file, WAVE makes a copy with your edits, gives it a name of the form *annotator.record*, and saves it in the current directory (i.e., whatever directory was current when you started WAVE). To be able to read your edited copy in a later WAVE session, be sure that the directory that contains your edited copy is listed in your DB path *before* any other directory that contains an older version of the same annotation file. Usually, your DB path begins with '`.`', a synonym for the current directory. If this is the case, simply return to the directory that contains your edited annotation file before starting WAVE each time.

If you created an annotation file without specifying an annotator name, WAVE uses its own name as the annotator name, so you should look for a file named `wave.`*record* in this case.

If the annotation file was originally created under MS-DOS, or if it is intended for use under MS-DOS, the filename is constructed in the opposite order (i.e., as *record.annotator*, where *annotator* is truncated to at most three characters). This is necessary in order to accommodate MS-DOS's restrictions on file names while still permitting flexibility in the choice of record names. Since the standard CD-ROM databases are intended to be usable under MS-DOS as well as in other environments, file names on these CD-ROMs follow this "record name first" convention as well.

D.5 Questions about editing

D.5.1 Why does WAVE tell me 'You may not edit annotations ...'?

Since WAVE is often used to *view* annotated data, without necessarily editing annotations, annotation editing is disabled when WAVE starts. If you have inadvertently done something (such as clicking the middle mouse button in the signal window) that WAVE interprets as an editing command, it will warn you that the editing action is disabled.

If you did intend to edit the annotations, select Allow editing from the (Edit ∇) menu, then repeat the edit.

D.5.2 How do I edit a record on a CD-ROM?

Be sure that your current directory is writable, that both your current directory and the CD-ROM directory containing the record you wish to edit are in your

DB path, and that your current directory comes *before* the CD-ROM directory in your DB path. When WAVE writes your edits, they always go into the current directory (so the current directory cannot be on the CD-ROM, since the CD-ROM is not writable). At a later time, when WAVE (or any other DB application) opens the record, it finds your edited version first, and doesn't look for the original version on the CD-ROM, provided only that, in your DB path, the CD-ROM directory follows the one in which your edits are stored.

D.5.3 How can I undo an edit?

WAVE doesn't support a general mechanism for 'undoing' edits. Once you move an annotation or change its attributes, you must manually restore its position and attributes. Deleted annotations can always be restored, however, by selecting the 'phantom' annotation left behind (these are displayed with a '.' for the annotation mnemonic) and then 'deleting' the phantom. This action restores the attributes of the original annotation, including the `subtype`, `chan`, `num`, and `aux` fields; if the annotation was moved, however, its original position (`time` field) is not restored.

WAVE is careful not to destroy the version of the annotation file that was current at the time WAVE loaded the record. WAVE always saves edited annotation files to the current directory (even if the original annotation file was loaded from another directory). WAVE's saved annotation files are given names of the form '*annotator.record*'. If, by saving its output, WAVE would overwrite an existing file with the same name, WAVE first renames the existing file by prefixing an underscore ('_') to its name. Thus it is always possible to recover the state of the annotation file as it was before you began the most recent editing pass. To do so, exit WAVE, and:

- If there is a file with a name of the form '*_annotator.record*' in the current directory, rename it as '*annotator.record*'.
- Otherwise, delete or rename '*annotator.record*' so that the previous version (which must be in another directory in the DB path) becomes visible once again to WAVE.

D.5.4 Can I define my own annotations?

Yes. See section 2.17, page 24, or the file named '`anntab`' in the WAVE distribution for examples.

D.5.5 How can I recover work after a crash?

WAVE autosaves your edits at 1-minute intervals, or after every 20 insertions, deletions, or attribute changes (repositioning doesn't count). You don't need to do anything special to recover your work, but you may lose up to a minute's worth of edits.

D.5.6 Can I edit annotations with a text editor?

Believe it or not, sometimes users ask this question.

If you have unusual requirements (for example, if you want to change all annotations of a certain type), you may be able to edit annotations more efficiently using a text editor such as `emacs`. Since annotation files are not text files, however, you will need to convert your files from binary to text form to edit them, and then convert the edited text back to binary form in order to use them further with WAVE. You can do this using `rdann` and `wrann` (see `rdann(1)` and `wrann(1)`, in the *ECG Database Applications Guide*, for details).

D.5.7 What are 'link' annotations?

By analogy to hyperlinks in World Wide Web documents, link annotations allow external data (including documents, images, and audio annotations among other possibilities) to be associated with specific locations in a record. In fact, any URL that can be understood by a supported web browser can be used to specify the location of external data in a link annotation. Within the link annotation, the URL appears as the first part of the `aux` string (in the Text field of the Annotation Template). If the `aux` string contains any spaces, any characters following the space are treated as a label to be displayed by WAVE in place of the URL itself.

Support for link annotations is a new feature of WAVE 6.0. To view the external data associated with a link annotation, select the link and press [Enter] (or [Return]). If your web browser is not already running, this may take a few moments while WAVE starts it. If nothing happens after a minute or so, check WAVE's Analysis Commands window for errors that may have occurred when WAVE attempted to start your web browser. (WAVE uses Netscape 1.1 or any later version by default. See section 3.10, page 43, for details on obtaining Netscape, or on configuring WAVE to use a different web browser.)

Link annotations may be moved, copied, attached to specific signals, inserted, changed, and deleted just as for any other annotation. WAVE does not include built-in facilities for editing the external data, however; other software such as a text editor or a special-purpose HTML editor may be useful for this purpose.

D.5.8 Can WAVE edit signal files?

WAVE itself doesn't do this, but certain kinds of signal file editing are easy to do using `snip` from WAVE's Analyze panel. A common requirement is to remove unwanted data from the beginning or end of a record. To do this, simply mark the segment to be retained using the '<' and '>' markers, then click on (Extract segment).

If your record contains unwanted signals, remove them from the Signal list before extracting the segment. You can also rearrange the order of signals within the signal list if you wish.

Note that this operation does not modify the original record; rather, the desired portions of the record are copied to create a new record. If you wish to copy a set of annotations as well as data from signal files, be sure to load the annotation file for the original record into WAVE before extracting the segment. (See `snip(1)`, in the *ECG Database Applications Guide*, if you need to copy two or more sets of annotations.)

To join two or more records end-to-end, use `dbcollate` to create a multi-segment record. `dbcollate` writes a special header file for the new record, but does not copy or modify the signal or header files of the original records; it will, however, write new annotation files on request, since the DB library does not support multi-segment annotation sets. The records to be joined must be ordinary (not multi-segment) records. If necessary, `xform` can be used to rewrite a multi-segment record as an ordinary record. See `dbcollate(1)`, in the *ECG Database Applications Guide*, for further information.

More complex editing of signal files, such as splicing segments or modifying individual samples, can be done by converting the signals to text form using `rdsamp` (accessible from WAVE via the (List samples) button in the Analyze window), editing the text as required, and converting the edited text to a signal file using `wrsamp` (see `wrsamp(1)`, in the *ECG Database Applications Guide*, for details).

D.6 What else can WAVE do?

D.6.1 How can I make a screen dump?

If you really want a screen dump, '`xwd | xpr`' will produce one in PostScript form. (Type '`man xwd`' and '`man xpr`' for details on options.) Another way to do this, if `xpr` is not available, is '`xwd | xwdtopnm | pnmdepth 255 | pnmtops`'. If your printer is not a PostScript printer, you can still print the output of `xpr` using Ghostscript, a freely available PostScript interpreter.

Why settle for a screen dump, though? In most cases, you will prefer to use '`pschart`' to produce a much nicer plot, and in much less time. Select Print from WAVE's (File ▽) menu to print the current contents of the signal window; use (Print chart) in the Analyze window if you wish to specify start and stop times. See `pschart(1)`, in the *ECG Database Applications Guide* for information about `pschart`'s numerous formatting options. If you don't like the defaults, change them by editing WAVE's menu file. Doing so also allows you to change the default format for the Print choice on the (File ▽) menu; see the comments in the menu file for details).

If you are plotting a great deal of data, you may wish to use `psfd` rather than `pschart`; to do so, select (Print full disclosure) rather than (Print chart) in WAVE's Analyze window. Most of `pschart`'s options are accepted by `psfd`.

If you need to collect a set of figures, either from a single record or from many records, use WAVE's Log window to record the times of interest, and enter captions for each figure in the Description field of the Log window. The log file generated in this way can be interpreted directly as a command file by `pschart`, which prints the captions as titles for each figure.

D.6.2 Can WAVE read digitized signals in *(fill in the blank)* format?

Probably it can. The most common formats for storage of digitized signals are fixed-length binary formats that encode integer samples, and WAVE supports dozens of such formats. WAVE does not require any embedded header information within signal files (but it can be made to ignore such information). In most cases, all you need to do is to write a header file that describes the format in terms understandable to WAVE; once you have done so, any of the other DB applications can also read your signals. Writing a header file can be most easily done by copying an existing header file and modifying it using a text editor. See `signal(5)` and `header(5)` in the *ECG Database Applications Guide*, for details on supported signal formats and how to write a header file.

There are at least three common ways of storing multiple signals: in separate files, in multiplexed (sample-interleaved) files, and in block-interleaved files. In a multiplexed file containing N signals, each set of N consecutive samples contains one sample from each signal. In a block-interleaved file containing N signals in blocks of 512 samples, for example, the first 512 samples come from the first signal, the next 512 samples come from the second signal, and so on. WAVE supports reading multiple signals from separate files or from multiplexed files (or both in the same recording), but it does not directly support block-interleaved files in general.

WAVE does support reading a special type of file containing signals sampled at different sampling frequencies. Files of this type contain "frames" of samples. Each frame contains at least one sample of each signal, but there may be two or more samples of signals sampled at multiples of the frame rate within each frame.

Block-interleaved files may be described simply as files with very large frames. If you have a block-interleaved file, therefore, you may be able to view it with WAVE via this expedient:

- First, check that the block size in samples is no greater than `DB_MAXSPF`, a symbol defined in the DB library header file `db.h` (which you will probably find in `/usr/local/include/ecg`). If necessary, increase the value of `DB_MAXSPF`, recompile and reinstall the shared DB library (`libdb.so.*`), and check to be sure that WAVE loads this version of the DB library (use the command ``ldd `which wave` ``).

- Write a header file describing the layout of the block-interleaved signal file (see `header(5)`, in the *ECG Database Applications Guide*, for details). The sampling frequency in the first line of the header should be given as the frame rate (i.e., the basic sampling rate divided by the number of samples per block), and the "samples per frame" field in each signal specification line should be given as the number of samples per block.

- Open the record in "high-resolution" mode with WAVE (see section D.6.3, page 106).

One significant limitation of this approach is that the time resolution of any annotation files created in this way is limited to the frame interval. You can avoid this limitation by reformatting the signal file (for example, by following the steps above to verify that the signal file is properly readable, then using `xform` with its own `-H` option; see `xform(1)` in the *ECG Database Applications Guide*.

A common feature of many signal file formats is *embedded header data*: bytes at the beginning of the signal file that do not contain digitized samples. By specifying the length of the embedded header data as the byte offset for the sample data in the (external) header file, such formats can be easily accommodated; WAVE and any other applications built using the DB library simply ignore any preamble. If the signal file format is documented, it is usually not difficult to write a program to generate a header file based on embedded header data. See `header(5)`, in the *ECG Database Applications Guide*, for details on byte offsets.

WAVE does not support directly reading signals stored in text files, because there is no general method for computing the position within such files of samples occuring at arbitrarily-chosen times. It is usually simple to convert such text files into WAVE-compatible binary files using `wrsamp`, an application supplied with WAVE.

For the same reason, WAVE does not support directly reading signals stored using variable-length encoding or non-uniform sampling intervals. If a decompression utility is available, convert the data to fixed-length, uniform samples.

WAVE does not support any non-integer storage formats either, mainly because such formats are not commonly used for digitized data.

If you have signals in an unsupported format, you have two choices if you wish to analyze them using WAVE:

- Write a utility to convert your data to a supported format. This is usually easy to do – use '`wrsamp.c`' (the source for `wrsamp`, provided with WAVE) as a model.

- Add support for your format to the DB library, and recompile the shared DB library. Once the new library is installed, WAVE and the other DB applications will be able to use your format without recompilation. If you have a large amount of data, this may be the only reasonable choice. It may not be much more difficult than writing a conversion utility, provided that your format permits computing file position as a function of time. See the appendix titled 'Extensions' in the *ECG Database Programmer's Guide* for hints on how to proceed.

For example, WAVE does not support directly reading EDF, the European Data Format first used for recordings of polysomnograms. (This is *not* the format used for either the European ST-T Database or the MIT-BIH Polysomnographic Database, both of which can be read directly by WAVE.) EDF uses block interleaving, and also includes an embedded header. Although the method outlined above for reading block-interleaved files can be used to read EDF files, a difficulty in doing so is that the block size is specified within the embedded header, and may vary between records. Moreover, since typical block sizes are quite large, the time resolution of annotations in this case is very coarse. A format converter for EDF files, `edf2mit`, may be used to rewrite such files in a format that WAVE can read without these limitations. This converter also constructs a suitable header file from the embedded header in the EDF file.

All currently supported signal formats use 16 bits per sample or less. In principle, formats requiring up to 32 bits per sample should pose no problem in environments that support WAVE, but formats requiring more than 16 bits may be difficult to support in other environments.

D.6.3 What is "high-resolution" mode?

Briefly, it is an alternative display mode for multi-frequency records only. See the *ECG Database Programmer's Guide* for details on multi-frequency records.

The `-H` option, introduced in WAVE 6.0, selects high-resolution mode. For ordinary records (those for which all signals are sampled at the same frequency), high-resolution mode is identical to standard mode.

In a multi-frequency record, some signals are sampled at multiples of the basic sampling rate. These are referred to as *oversampled signals.* In standard mode, WAVE *decimates* these signals (i.e., WAVE reduces their sampling rates to the basic sampling rate by averaging successive samples) before displaying them. In high-resolution mode, however, WAVE displays each sample from any oversampled signals if the screen resolution is high enough (note that at standard display scales, it may be difficult to see the difference). In this mode, WAVE replicates additional copies of each sample of any signals sampled at lower rates before displaying them; in consequence, particularly at highly magnified display scales, these signals may have a marked stairstep appearance.

Note that the time resolution for placement of annotations is the same in high-resolution mode and in standard mode. Thus, for example, it is not possible to place annotations on consecutive samples of an oversampled signal, since the second annotation will replace the first one. This is a limitation of the DB library, not of WAVE itself, and may be removed in a future release of the DB library.

D.6.4 Can WAVE open more than one record at once?

Yes. All records should be listed in one command-line argument, following the `-r` argument on the command line; separate the record names by '`+`' characters. For example, the command

```
wave -r 237+237n &
```

starts a WAVE *process group*, opening a separate WAVE signal window for each of record 237 and record `237n`. Each signal window corresponds to a separate

WAVE process; thus you may browse through and edit each record independently. The signal window assigned to the last record in the group (in this case, `237n`) is the *master signal window*; it contains a (Sync) button, which may be used to synchronize the other signal windows with the master at any time. By synchronizing, we mean that the other windows are redrawn as necessary so that the times shown in their lower left corners match that shown in the master signal window.

Remote-control applications such as `wavescript` and `wave-remote` can also start or control a WAVE process group. All signal windows in the group move synchronously in response to commands from `wavescript` or `wave-remote`. You may change the record that is open in any signal window, either via the Load window or using a remote-control application. You may quit from any signal window without affecting the others.

Note that there are a few important limitations:

- The number of signal windows (WAVE processes) in any process group cannot be *increased* once the process group has been created.
- Although you may run two or more WAVE process groups simultaneously, applications such as `wavescript` can control only a single process group at a time (by default, the last one started).
- Although you may have the same record open in more than one signal window, you should not edit the same annotator of the record in more than one window at a time (just as you should not edit the same text file in more than one editor window at once). You may edit different annotators of the same record freely.

Taken together, the first two of these points imply that if you decide after opening a record that you would like to add another record to the group, it will be necessary to exit and start again, naming all desired records in the group on the WAVE command line when you do so.

If you wish to move through two records sampled at the same sampling frequency in lockstep (for example, to view digitally filtered signals and the original unfiltered signals side-by-side), another approach is to create a single header file that names the signal files for both records. It is not necessary for all signals to be in the same signal file, or even on the same disk drive. In this case, however, the total number of open signals must be 32 or less (see the next question).

D.6.5 Can WAVE open more than 32 signals in a window at once?

Not without recompiling. The limit is `DB_MAXSIG` (defined as 32 in `<ecg/db.h>`); it is necessary to recompile WAVE itself, as well as the DB library, in order to increase this limit. In a future version, the current limit of 32 signals will be substantially increased or removed.

D.6.6 Can WAVE open more than one annotation file in a window at once?

No. In most cases, however, it's not really necessary to do so.

For example, if you want to compare two sets of annotations of the same data in order to study their differences, use bxb with the '-o' option to prepare a comparison annotation file, and then use WAVE to open that file. (See bxb(1), in the *ECG Database Applications Guide*, for details.) If your goal is to find discrepancies, this is by far the best way to do so; 'bxb' produces a statistical summary of them, and you can search for NOTE annotations to find each disagreement (enter '"' in the 'Search for' field of WAVE's Find window).

If you want to annotate multiple signals independently (for example, ECG and respiration), choose 'attached to signals' from the Show annotations: ▽ menu in WAVE's View window. See section 2.15, page 23, for details.

If you already have multiple independent annotation files for a single record (whether these contain annotations for independent signals or annotations for disjoint segments of the record), you can use mrgann to merge them into a single file, preserving information about which file each annotation came from if you wish (see mrgann, in the *ECG Database Applications Guide*, for details).

dbcollate can also be used for this purpose if you have separate annotation files for each segment of a multi-segment record.

D.6.7 Can WAVE do smooth scrolling?

No.

D.6.8 Can WAVE be used for real-time display of data being acquired?

Not easily. If you have a UNIX application that can write data in a format readable by WAVE, it should be possible in principle to run it from WAVE's Analyze window, and to use WAVE to review the data being acquired, using the (Reload) button in WAVE's Analyze window to read additional samples as they become available. (WAVE will not attempt to read beyond what it thinks is the end of the record; if it ever catches up with the digitizing process, therefore, WAVE can only be forced to read further by using (Reload), or by closing and reopening the record.)

D.6.9 Can WAVE scroll through a record without user intervention?

Yes. Create a log file that contains entries spaced at the desired intervals throughout the record. A suitable interval might be 10 seconds if that is the width of the signal window. Open the Log window, load the log file, and click on (>>) to start the scrolling. Adjust the delay between frames using the slider on the Log window.

Another way to review a record is via the scope window (accessible by clicking on (Show scope window) in WAVE's Analyze window). Note that the scope display, since it is triggered by annotations, is of no use in detecting erroneously unannotated waveforms (false negatives), since such waveforms are simply not displayed in the scope window. The scope is useful for studying morphologic variation of waveforms, and for locating erroneously annotated waveforms (false positives).

D.6.10 How can I use WAVE's menu variables in a script or other program?

See WAVE's default menu file for examples of passing WAVE's menu variables as command-line arguments to other programs.

Click on (Export variables) in WAVE's Analyze window if you need to use these variables within a shell script. (If you are not using `csh` or `tcsh`, edit the WAVE menu file first to select the alternate definition of this action.) Once the variables have been exported, they may be used in the same way as any other environment variables within your shell scripts. For example, the time of the left edge of the signal window is `$LEFT`. Within a C program, read the values of these variables using `getenv`; for example, `getenv("LEFT")` returns the (string) value of the time of the left edge of the signal window. Note that the exported values are only a snapshot of the variables at the time you click on (Export variables); the values do not track later actions performed within WAVE.

D.6.11 Is there a Motif version of WAVE?

No. (Rant begins here.) When I designed WAVE, I attempted to use Motif, but switched to XView when it became clear that much of the documented functionality of the Motif toolkit (at the time, version 0.9) was unimplemented or unusable. It also seemed unlikely that anyone would prefer Motif's buggy, ugly, proprietary, bloated, kludgy procedural interface over XView's buggy, attractive, non-proprietary, streamlined, elegant object-oriented interface (end of rant).

Motif has clearly become the standard, however, and a consequence of this is that the Open Look user interface presented by WAVE is unfamiliar to many users. There is a scarcity of introductory material for Open Look in print. Unfortunately, the Motif API is vastly different from the XView API with which WAVE is written, and a port to Motif would be decidedly non-trivial.

D.6.12 How can I find out about ...?

There are several sources of information about WAVE:

- If you have not used WAVE before, start by reading the WAVE *User's Guide* (this book) and working through the tutorial examples it contains. If you need to use existing DB applications under WAVE's control, refer to the *ECG Database Applications Guide*. If you need to develop your own applications for use with WAVE, read the *ECG Database Programmer's Guide*. Printed copies of these guides may be purchased from MIT, or you may print your own copies on a PostScript printer by following the instructions in the '`manuals.doc`' file distributed with WAVE.

- The on-line version of the WAVE *User's Guide* can be opened for browsing by clicking left on (User's Guide) in WAVE's Help Topics window (this requires WAVE 6.0 or later, and Netscape 1.1 or later on the WAVE host). This guide may also be read with any Web browser independently of WAVE (point your browser to `file:/usr/local/help/html/wug/wug.htm` if you have installed WAVE 6.0 or later on your system, or to `http://ecg.mit.edu/wug/wug.htm` otherwise).

- Use XView spot help for brief descriptions of WAVE's controls and other graphic elements. Point to any control and press the [Help] key (see "How do I get spot help ...?", section D.1.5, page 93, if you don't have a [Help] key).

- Use WAVE's task-oriented on-line help to learn how to use WAVE as a tool for interactive data analysis, control of external programs, annotation editing, and other topics. Click left on (Help) in WAVE's main control panel to open the Help window, containing buttons for each topic. Click left on any of these buttons to open a help window for the associated topic. Within the help window, use the scroll bar to move through the topic, or click on (Print) button to obtain a printed copy of the entire topic.

- Select 'About WAVE...' from the (Properties ∇) menu, or (What's new) from the Help window, for a summary of new features in WAVE, a current list of known bugs, and instructions for printing a copy of the on-line manual, which includes the text of all of the other help topics. Click on (Print) to obtain a printed copy of the 'What's new' topic.

- The `man` page for WAVE (`wave(1)`) contains a concise description of its command-line options, the environment variables and X resources it uses, and the WAVE menu file. (This information is also included in this Guide; the `man` page is intended as a compact reference. The `man` page for WAVE can be found in the *ECG Database Applications Guide*, or on-line by typing '`man wave`', or by using `xman` or `tkman`, etc.)

- For additional information on the WAVE menu file, see the comments in the default menu file distributed with WAVE and usually installed as '`/usr/local/lib/wavemenu.def`'.

- For information about the data formats supported by WAVE and the other DB applications, see `annot(5)`, `dbcal(5)`, `header(5)`, and `signal(5)`, in the *ECG Database Applications Guide*.

Appendix E

Command-line Options

To start WAVE, open a terminal window and type into it:

`wave -r` *record options ...*

In most cases, *record* is a single record name; you may, however, name several records to be opened by concatenating their names with `+` between them, as in 'tt wave -r 100+101+102 -a atr', which opens records `100`, `101`, and `102`, each in its own signal window. Note that there must be no spaces around the '+' characters (the list of records should appear as a single command-line argument without embedded spaces).

If the `-r` *record* arguments are omitted, WAVE simply prints a summary of its command-line syntax in the terminal window.

The *options* may include any of these, in any order:

`-a` *annotator* Open the specified annotation file for *record*. If this file does not exist, it will be created once you insert an annotation.

`-dpi` *xx*[x*yy*] Calibrate WAVE for use with a display having a resolution of *xx* (by *yy*) dots per inch.

`-f` *time* Open the record beginning at the specified *time*.

`-g` Use shades of grey only, even on a color monitor.

`-H` Operate in "high-resolution" mode when reading multi-frequency records.

`-m` Use monochrome (usually black and white) only, even on a color or greyscale monitor. The line styles selected by the `-m` option may be easier to distinguish on some greyscale monitors than the default shades of grey.

`-O` Use overlay graphics for maximum speed and display quality if possible. This is the usual default if the X server supports a PseudoColor or GrayScale visual. This option exists only to force use of overlay graphics if a different mode has been chosen as the default.

`-s` *signal* [*signal* ...] Initialize the signal list. By default, the signal list includes all available signals, in numerical order.

`-S` Use the standard (shared) color map, even if is possible to modify the color map. Using this option conserves color resources if you have other applications that use non-standard colors, at a small (possibly unnoticeable) expense of some speed in redrawing the display. The `-S` option may be used in combination with the `-g` option if desired.

The following group of options can be used to enable the optional display elements normallly chosen in the View window.

`-Vs` Display annotation `subtyp` fields.

`-Vc` Display annotation `chan` fields.

`-Vn` Display annotation `num` fields.

`-Va` Display annotation `aux` fields. (This is the default for annotations such as rhythm changes and comments; selecting this option causes `aux` fields to be displayed for all annotations, and also causes the annotation mnemonics to be displayed for rhythm changes, comments, etc.)

`-Vm` Display annotation marker bars.

`-VN` Display signal names along the left edge of the signal window.

`-Vb` Display baselines for any DC-coupled signals, and label the zero levels and the units along the right edge of the signal window.

`-Vl` While the pointer is in the signal window and any mouse button is depressed, track the intersections of the marker bar with the signals and draw horizontal marker bars across the signal window at the levels of these intersections.

The final group of options allows you to select the initial choices from the menu buttons in the View window. Each of these options should be followed by a numeric argument that matches the *position* of the desired choice in the menu, where the top item on each menu is in position 0, the one below it is in position 1, etc. For example, to set the initial amplitude scale to 5 mm/mV (the item at position 2 in the Amplitude scale menu), add `-Vv 2` to the command line.

`-Vt` *n* Set the time scale (0: 0.25 mm/min, 1: 1 mm/min, 2: 5 mm/min, 3: 25 mm/min, 4: 50 mm/min, 5: 125 mm/min, 6: 250 mm/min, 7: 500 mm/min, 8: 12.5 mm/sec, 9: 25 mm/sec (default), 10: 50 mm/sec, 11: 125 mm/sec, 12: 250 mm/sec).

`-Vv` *n* Set the amplitude scale (0: 1 mm/mV, 1: 2.5 mm/mV, 2: 5 mm/mV, 3: 10 mm/mV (default), 4: 20 mm/mV, 5: 40 mm/mV, 6: 100 mm/mV).

`-VS` *n* Set the choice on the Draw menu (0: all signals (default), 1: listed signals only).

`-VA` *n* Set the choice on the Show annotations menu (0: centered (default), 1: attached to signals, 2: as a signal).

`-VT` *n* Set the choice on the Time display menu (0: elapsed (default), 1: absolute, 2: in sample intervals).

`-VG` *n* Set the choice on the Grid menu (0: none, 1: 0.2 s, 2: 0.5 mV, 3: 0.2 s x 0.5 mV, 4: 0.04 s x 0.1 mV).

WAVE also accepts generic XView options, including the `-default` and `-xrm` options for setting X11 resources (see section 6.4, page 61).

E.1 Running two or more WAVE processes

By specifying two or more record names, separated by '`+`' characters, in the command-line argument that follows '`-r`' (see above), you may open separate WAVE signal windows (processes) for each record. These processes are almost completely independent: from any signal window, you may navigate within the record, change display settings, edit annotations, run external analysis programs, quit the process, etc., without affecting any other signal windows.

For example, you may open two signal windows for the same record by:

```
wave -r 100+100 -a atr
```

You can now move about the record freely in either window. This facility makes it easy to compare different segments of the record. Note that whenever two or more windows are displaying the same set of annotations, as in this case, only one should be editing the annotations at any given time.

The window associated with the *last* record named on the command line has a special status: it is designated the master signal window, and a (Sync) button appears at the top of this window. Clicking on (Sync) causes all of the other signal windows to be redrawn so that the times shown in their lower left corners match that in the master signal window. (Note, however, that if you have quit a signal window from the middle of the list, any signal windows from earlier in the list will no longer respond to sync requests.)

By default, all command-line arguments apply to all signal windows. You may specify an argument that is to apply to only one signal window, however, by prefixing the argument with '`+`*n*`/`', where *n* is the *signal window number*. (The first signal window, corresponding to the first record named on the command line, is signal window number 0; the next is number 1, etc.)

This facility has many applications. For example, you may wish to open two copies of the same record, with two different annotators:

```
wave -r 100+100 -a +0/atr +1/qrs
```

In this case, record `100` is opened in two windows, with annotator `atr` in window 0 and annotator `qrs` in window 1. (The '`-a`' option applies to both windows since it does not have a '`+`*n*`/`' prefix.)

As another example, you may wish to discuss a record with colleagues at other locations:

```
wave -r 200+200+200 -a qrs +0/-display +0/atlantic.bigu.edu:0 \
    +1/-display +1/pacific.widget.com:0
```

Here, record `200` is opened in three windows. Window 0 is opened on display 0 of `atlantic.bigu.edu`, window 1 on display 0 of `pacific.widget.com`, and window 3 (the master window) on the local display. (For this to work, your colleagues must first allow your computer to open windows on their displays, typically using `xhost`. See `xview(7)` for information about the `-display` option. Notice that the '`+`*n*`/`' prefix must be attached to both the `-display` option and to its argument in order to apply both of these arguments to the same signal window.) Your colleagues can freely move about the record, but you can direct the discussion at any time by using the (Sync) button in your signal window. In a case such as this one, anyone can enable editing; you should do so only after making sure that no one else has. Once you have saved your work (by selecting Save from the File menu), your changes become visible to your colleagues if they reload the annotations (by clicking on Reload from the Load window).

As a final example, the MIMIC Database includes both high-resolution waveform records and medium-resolution (roughly 1 sample per second) computed measurement records. You may view both of these at the same time using a command such as:

```
wave -r 237+237n -a all
```

Typically, you will wish to view the high-resolution and low-resolution data at different time scales. Although WAVE attempts to choose reasonable defaults, you can adjust the scales independently if you wish:

```
wave -r 237+237n -a all +1/-Vt +1/2
```

If you use `wavescript` or `wave-remote` to control the master signal window (this happens by default unless you use the `-pid` option of these programs to control a different signal window), the other signal windows are kept synchronized with the master window.

Note that you cannot *increase* the number of signal windows in a group once you have started a WAVE process group, although you can run more than one process group at a time if you wish.

Appendix F

Menu Variables

In WAVE's menu file, the variables listed below are interpreted by WAVE before they are passed to the shell, typically as command-line arguments in button-action commands.

`$RECORD`	the current record name
`$ANNOTATOR`	the current input annotator name
`$START`	the current Start (elapsed) time as shown in the Analyze window (the time of the '<' marker, if any)
`$END`	the current End (elapsed) time as shown in the Analyze window (the time of the '>' marker, if any)
`$DURATION`	the time interval between `$END` and `$START`
`$LEFT`	the time of the left edge of the signal window
`$RIGHT`	the time of the right edge of the signal window
`$WIDTH`	the time interval between `$RIGHT` and `$LEFT`
`$SIGNAL`	the number of the selected signal
`$SIGNALS`	the signal list
`$LOG`	the name of the current log file
`$DB`	the DB path (from the Load window)
`$DBCAL`	the DB calibration file (from the Load window)
`$TSCALE`	the time scale, in mm/sec
`$VSCALE`	the amplitude scale, in mm/mV
`$DISPMODE`	the annotation display mode (0: annotations displayed in center, no marker bars; 1: annotations displayed in center, long marker bars; 2: annotations attached to signals, no bars; 3: annotations attached to signals, short bars; 4: annotations displayed as a signal, no bars; 5: annotations displayed as a signal, long bars)

`$PSPRINT`	the command for printing PostScript data from the standard input, as specified in the Print setup window
`$TEXTPRINT`	the command for printing text from the standard input, as specified in the Print setup window
`$URL`	the URL specified by the most recently selected link

Other strings that begin with '`$`' are passed to the shell unchanged.

Appendix G

Keyboard Command and Mouse Action Summary

G.1 General rules for all Open Look applications

The basic rules for using a mouse in an Open Look application such as WAVE are:

- *To open menus (always identified by a* ∇ *at the right end of the control), click right.*
- *To use any other control, click left.*

Notice boxes and some other windows have *default buttons*, identified by a double oval outline. If the pointer is anywhere in the window, you can press [Enter] ([Return]) instead of clicking left on the default button if you wish.

To type into a text field, move the mouse pointer near the insertion point and click left. The text cursor should appear nearby; if it is a black triangle, you may begin typing. Use [BackSpace] or [Delete] to erase the character to the left of the text cursor. To move the cursor within the field use the arrow keys ([←], [→]), or use `emacs`-style commands:

[Ctrl]+[A] move to the beginning

[Ctrl]+[E] move to the end

[Ctrl]+[F] move forward one character

[Ctrl]+[B] move backward one character

Until you press [Return] (or [Enter]), your typing is not interpreted.

Unless spot help is disabled, you can get help on any control by pointing to it and pressing [Help] (or [F1] if you don't have a [Help] key). To enable spot help, see "How do I get spot help ...?", section D.1.5, page 93,.

G.2 Using the keyboard and mouse in the signal window

In this section, wherever two keyboard actions are specified, the first should work on any keyboard, but the second may be usable on Sun type 4 keyboards only.

Mouse actions in the signal window are generally related to selecting and editing annotations. If annotation editing is disabled, actions that would otherwise modify the annotation file are disallowed.

Simple mouse actions

These actions are used to select, insert, change, and move annotations.

left mouse button click If the Annotation Template is visible or iconified, select the annotation to the left of the pointer (if this annotation is not in the signal window, recenter the signal window on it). Otherwise, display the Annotation Template.

middle mouse button click Insert an annotation at the location indicated by the pointer, with fields as specified in the Annotation Template. If the pointer is within the selection rectangle, the selected annotation is replaced by the inserted one; in this case, if the Type is '`.`' (deleted annotation), the selected annotation is marked for deletion. An annotation so marked can be restored by deleting it again.

right mouse button click Select the annotation to the right of the pointer (if this annotation is not in the signal window, recenter the signal window on it).

If the pointer is moved during any of the simple mouse actions above, the selected annotation (if any) is *dragged* along with the pointer. When the button is released, the selected annotation is *dropped* at the new location.

Even if editing is disabled, you may insert, delete, or move '`:`', '<', and '>' markers using these mouse actions (or their keyboard equivalents, below) if you first set the Type field of the Annotation Template to the desired marker type.

If you prefer not to have the Annotation Template visible while editing, you can iconify it so that it will not continually pop up whenever you click left.

Keyboard equivalents for simple mouse actions

These actions can be used on any system. If you have a one- or two-button mouse, your X server should provide other methods for simulating middle and right button clicks; see the documentation for your X server for details.

[←] Simulate a left mouse button click.

[F2] **(or [5] on the numeric keypad)** Simulate a middle mouse button click.

[→] Simulate a right mouse button click.

To move the pointer left or right, use [F3] and [F4] (or [=] and [*] on the numeric keypad). To simulate a dragging action, press and hold [←], [F2], or [→], use [F3] or [F4] to move the pointer to the desired location, then drop the selected annotation by releasing all keys.

Mouse and keyboard combination actions

These actions are used to select signals. To perform these actions, press and hold the designated key, perform the mouse action, then release the key.

[Shift]+**left mouse button** Select the signal nearest the pointer (the selected signal is highlighted, and its signal number is entered into the Signal field in the Analyze window). This action can also be performed using [Shift]+[←].

[Ctrl]+**left mouse button** Select the signal nearest the pointer as above, and also enter it into the Signal list field in the Analyze window. This action can also be performed using [Shift]+[←].

[Meta]+**left mouse button** Select the signal nearest the pointer as above, but *delete* it from the Signal list. (There is no keyboard-only equivalent.)

(Most keyboards do not have a key labelled Meta. On most Sun keyboards, the Meta keys look like this: [◇]; they flank the space bar. On most other keyboards, the [Alt] key is equivalent to Meta.)

Other keyboard actions

These actions are used for annotation editing and navigation through the record.

[↑] (In multi-edit mode only, and only if an annotation is attached.) Move the selected annotation to the next emphlower signal number. If another annotation is at that location, however, select that annotation instead. If (within the View window) you have chosen to draw all signals, this corresponds to moving the selection up one signal on the display. If you have chosen to draw listed signals only, the order of signals in the signal list determines where the annotation moves on the display.

[Ctrl]+[↑] As for [↑] alone, except that the selected annotation is copied rather than moved.

[↓] As for [↑], except that the selected annotation is moved to the next *higher* signal number (i.e., down one signal on the display if all signals are drawn).

[Ctrl]+[↓] As for [↓] alone, except that the selected annotation is copied rather than moved.

[F6] **(or** [Copy]**)** Copy the selected annotation into the Annotation Template.

[F9] **(or** [Find]**)** Search forward (as for (Search >)).

[Ctrl]+[F9] **(or** [Ctrl]+[Find]**)** Search backward (as for (< Search)).

[Shift]+[F9] **(or** [End]**)** Move to the end of the record.

[Ctrl]+[Shift]+[F9] **(or** [Home]**)** Move to the beginning of the record.

[F10] **(or** [PgDn]**)** Move forward half a screen (as for (>)).

[Shift]+[F10] **(or** [PgUp]**)** Move backward half a screen (as for (<)).

[Ctrl]+[F10] **(or** [Ctrl]+[PgDn]**)** Move forward a full screen (as for (>>)).

[Ctrl]+[Shift]+[F10] **(or** [Ctrl]+[PgUp]**)** Move backward a full screen (as for (<<)).

Support for link annotations is not available in WAVE *5.0.*

[Enter] **(or** [Return]**)** (Only if a link annotation has been selected.) Show the external data specified by the link using a web browser; start the web browser first if necessary.

Typing an annotation or marker mnemonic sets the Type field of the Annotation Template (even if the Annotation Template is not visible, and even if editing is disabled). These mnemonics are shown on the Type menu (see figure 2.5, page 16).

Index