PLEXUS—An On-line System for Modeling Neural Networks

J. C. Dill, D. L. Randall, and I. Richer*
California Institute of Technology, Pasadena, California

A description is presented of PLEXUS, a system which enables a user to construct and specify a neural network, to analyze the output data produced by the network, and to store and retrieve networks and data from a library. The system, operated entirely from a digital display unit, interacts directly with the user and permits easy and rapid transitions between the various phases of the modeling process. PLEXUS is designed to complement neurophysiological research so that the systematic development of neural models can be coordinated with experimental work.

PLEXUS networks are built up from components representing individual neurons, external stimuli, and interconnecting fibers, each component being of a relatively detailed nature. Provision is also made for the use of experimental data as input to a network. Convenient means for specification and modification of a network and extensive error-checking capabilities are provided. Data resulting from the simulation of a network may be analyzed by a variety of techniques ranging from examinations of the gross characteristics of the data to the determination of detailed statistical properties.

KEY WORDS AND PHRASES: biological modeling, data analysis, discrete system simulation, library systems, modeling, network simulation, neural networks, neurophysiological models, on-line simulation, simulation
CR CATEGORIES: 3.12, 3.19, 3.29, 3.65

1. Introduction

A valuable aid in the study of information processing in living nervous systems is the ability to develop neural network models based on experimental results. In recent years the digital computer has proved useful in performing this type of modeling. This paper describes PLEXUS—a system which not only provides a convenient method for modeling neural networks, but also coordinates this modeling with experimental research. PLEXUS operates on-line with all man-machine communication via a graphic display unit. A user is able to generate neural network models, to stimulate their operation, and to analyze the data produced. In addition, a library within PLEXUS provides for storage of networks and data.

PLEXUS was designed in an attempt to overcome some of the shortcomings inherent in previous neural modeling programs. For example, the capabilities of graphical input-output equipment are clearly well suited for the specification of the topology of a network. Consequently, PLEXUS allows the user to construct a schematic two-dimensional layout of network components and their interconnections. Also, since the development of a satisfactory model generally requires trial and error, the time between the specification of a network and the observation of its output should be short: in a typical session, operations should proceed cyclically from specification and modification, to simulation, to analysis, and if necessary back to modification. Therefore, PLEXUS operates in a conversational mode, permitting easy and rapid transitions between these different phases of the modeling process. Another design objective was to make PLEXUS easily approachable by those unacquainted with computers: the major concern of the user should be problem-solving, with only a short period required for familiarization with the system. For this reason, extensive error-checking capabilities are incorporated, a specific error message being displayed whenever an invalid parameter value is entered or whenever an illegal operation sequence is requested.

Figure 1 shows a simplified block diagram of PLEXUS and the framework within which it operates. Arrows indicate possible paths of data flow. The lower third of the diagram represents the experiment and the data-collection apparatus. In a typical experiment, a visual stimulus is presented to an insect, and evoked responses (picked up by microprobes in its nervous system), together with a signal corresponding to the stimulus, are converted to digital form and transmitted to the computer. Shown in the center of the figure is a simplified representation of the analysis environment. Of primary interest here are a data library and a set of routines for mathematical analysis of data and display of results, these routines being under the control of a multiprogramming monitor [1]. The functions of network generation and simulation, and of storage and retrieval of data are symbolized in the upper portion of Figure 1. These functions, together with some of the analysis procedures, constitute the PLEXUS system. PLEXUS may be used either in coordination with experimental work or independently, as a theoretical research tool.

Currently, PLEXUS operates either on the IBM 360/50 or 360/44 at the California Institute of Technology.[1] Each of these machines has several disk storage devices (IBM 2311) and a buffered graphic display unit (IBM 2250). The system is operated from one of these display units, with communication to the central processor via a function keyboard, a lightpen, and a typewriter console. On the 360/50, execution takes place under the control of a time-sharing system [2]. PLEXUS was written in assembly language and is supported by other software [2–4] which simplifies access to the data storage devices and the display unit.

The development of PLEXUS was supported by the National Institutes of Health (Grant GM01335) and by the Carnegie Corporation.
* Present address: Massachusetts Institute of Technology, Lincoln Laboratory, Lexington, MA 02173.

[1] Although the 360/44 has a smaller instruction set than the 360/50, program compatibility is obtained by means of an interpreter for those instructions which are invalid on the 360/44.

At the begining of a session, a display—the Session Options Page—is presented, which allows the user to select one of five basic options.

1. Begin specification of a new network.
2. Continue specification of a previous network.
3. Perform library functions.
4. Analyze data.
5. Terminate session.

At any time during a session the user may return to the Session Options Page and again choose one of the options. In Section 2 the first two options are described. If the second option is chosen, the network referred to is either one currently undergoing development or one retrieved from the library. The functions which allow manipulation of the data on the PLEXUS library and the structure of the library itself are explained in Section 3. The means by which data are analyzed are presented in Section 4. In Section 5 some of the conclusions reached as a result of experience using PLEXUS are discussed.

2. Modeling Concepts and Network Representation

A PLEXUS network may be built up from three types of components: sources, cells, and fibers. Sources represent stimuli or external inputs; cells represent neuronal cell bodies; fibers, which act as interconnections, represent axons and dendritic structures. Items of interest in a network are impulses (spikes) which are emitted by sources and cells and which represent the action potentials in an actual nerve network. If a source or cell emits a spike, then a spike is initiated on all fibers emanating from that component. Each spike traveling along a fiber reaches a fiber terminus after a specified conduction time and impinges upon a cell, thereby altering the internal state of the cell. Of primary concern in a network are the emission times of sources and cells; it is these times that are the output data of the network. In the first part of this section the manner in which a network is set up is explained; in the second part a description of the model is given; and in the final part the format in which the data representing a network is structured is outlined.

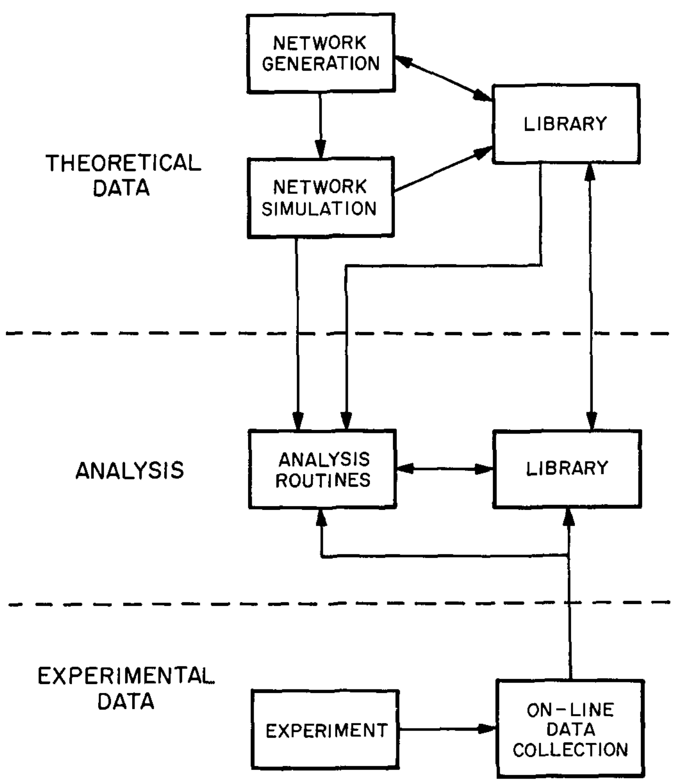

Fig. 1. Simplified block diagram of the PLEXUS system and its relation to the experimental environment

Construction and Specification of a Network. A PLEXUS network is generated by means of two basic types of operations, those having to do with the construction of a topology, and those having to do with the specification of parameters. All these operations take place within six "modes," each mode being entered by means of a particular key on the function keyboard. Two of the modes—DRAW and DELETE—pertain to topology construction, and the remainder—DEFINE, CREATE STANDARD, RETRIEVE STANDARD, and OUTPUT—pertain to parameter specification. Network generation can proceed in any sequence and may, in fact, extend over several sessions. The user has complete freedom over the order in which he constructs, specifies, and modifies a network.

DRAW—In this mode a grid of points is presented on the face of the cathode-ray tube (CRT) display. In order to "draw" a source, the user flags a grid point with the lightpen and depresses the appropriate function key; a source, represented by a square, then appears at the flagged point. A cell, represented by a triangle, is drawn in a similar manner. The user creates a fiber, represented by up to three straight-line segments, by flagging a source or a cell (the origin), up to two grid points, and a cell (the terminus), and then depressing a function key. Figure 2 shows a simple network with the drawing grid erased. A source or cell is identified by the prefix "S" or "C" followed by an unique two-digit number. A fiber is identified by its origin and terminus; if more than one fiber joins two elements, an extra identifying number is displayed, as indicated on the bent fiber between S01 and C01 in Figure 2. The user is not restricted in the layout of his network: any type of feedback and interconnection is permitted. The internal structure of the data limits the number of sources and cells to 63 of each (and at most 64 fibers between any two elements). In practice, however, these restrictions are not severe because PLEXUS is intended to be used to model small or moderate sized networks of a detailed nature. However, very large networks can often be split up and simulated in stages, as will be explained below.

DELETE—The user may delete any component from the network by flagging that component with the lightpen. If a source or a cell is deleted, then any associated fibers are deleted simultaneously.

DEFINE—The user may lightpen any component that he has drawn. The display of the network topology is then erased, and he is presented with a "page" indicating the parameters which must be specified for that component. The page for a fiber is shown in Figure 3. For this particular component, five values are required for complete specification. The conduction time had been specified earlier, and therefore its value is shown. The user may enter (or alter) some or all of the parameters for a component whenever the specification page representing that component is displayed. Parameter values may be assigned by four methods. First, numbers may be typed in by means of the typewriter keyboard associated with the CRT. Second, if the name of another element is entered on the line beneath the message IDENTICAL TO, then the parameter values of that element are reproduced for the current element (provided, of course, that the element types match). A group of two or more such identical elements share the same parameter set. This feature not only conserves storage space but is especially convenient in the development of a model with a group of identical components, since only one set of parameters need then be altered in order to modify the properties of all members of the group. The user may remove a component from a group of identical components by typing blanks in the IDENTICAL TO area; the component is then "independent" and has its own parameter set. Third, if the name of another element is entered in the COPY FROM area, data from the named component are assigned to the current component, but the parameter sets remain independent, and a change made in one component will not affect the other. The final means of specification utilizes information stored on the PLEXUS library and is described below under RETRIEVE STANDARD.

After performing any desired specification, the user may return to the display of the topology by depressing a function key. The parameter values from the CRT are then entered into the network data structure and error checking takes place. If an invalid value (e.g. a negative conduction time) is encountered, an error indication is displayed next to the parameter. The user is not permitted to exit from a specification page until all the parameters that he has specified are valid. It is not necessary to enter values for all parameters at one time, but those that are entered must be valid. If all the parameters for a component have been specified, then the component name is underlined in the topology (as for C01 in Figure 2). Completely specified fibers are displayed as double lines.

CREATE STANDARD—In this mode the parameter set of a component may be stored in the PLEXUS library. The user types a name to identify the parameter set, and he flags the desired component with the lightpen.

RETRIEVE STANDARD—Parameter values for an element may be specified by retrieving data that was stored in the library at some previous time by means of CREATE STANDARD. The user enters the name identifying the parameter set and flags the desired component, and the parameter values are assigned (provided that the parameter set is of the proper type).

OUTPUT—Sources and cells flagged in this mode will have their output recorded in temporary storage during simulation.

After specifying all the parameters of each component in the network, the user may indicate when simulation should terminate. Thus he specifies a maximum simulated time, a maximum execution time, and if he desires, a maximum number of spikes to be emitted from any source or cell. He can then initiate simulation, and when any of these criteria are satisfied, simulation will terminate, the Session

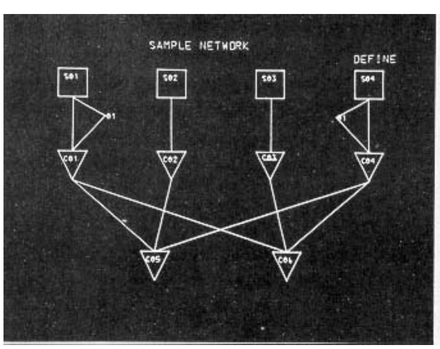

FIG. 2. A sample network containing the three types of components: sources, cells, and fibers. The line under the name "C01" indicates that all the parameter values for this component have been properly specified.

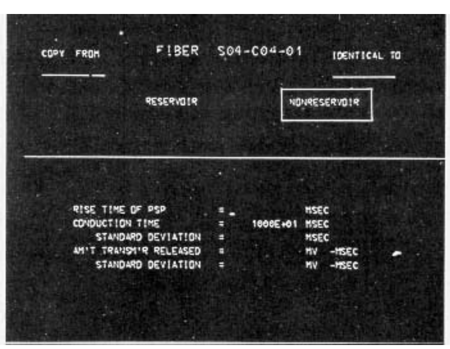

FIG. 3. The specification page for a nonreservoir type of fiber with one parameter value entered.

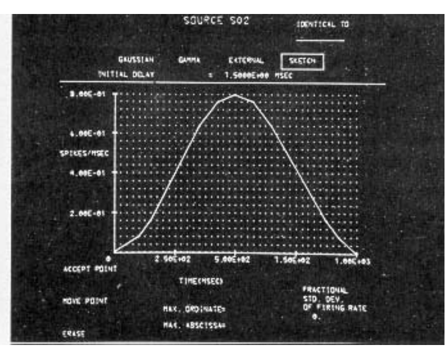

FIG. 4. The specification page for a "sketch" source. The curve, which may contain up to 20 straight-line segments, is considered to be periodic.

Options Page will reappear, and any of the basic options may again be chosen.

Description of Neurophysiological Model. Before the PLEXUS system was developed, a neural modeling program written by Perkel [5] operated in batch processing mode on the IBM 7094. The neural network model used in that program provided a compromise between the complexity of individual components and the complexity of the networks that could be handled. Consequently, this model, with significant additions and modifications, was incorporated into PLEXUS. The components in a PLEXUS network can account for such physiological phenomena as threshold, relative and absolute refractory periods, tiring, adaptation, and smoothly rising postsynaptic potentials. On the other hand, a network of considerable size and complexity can be specified and simulated in a reasonable length of time. The types of components used in PLEXUS and the algorithm employed for simulation are briefly described below.

Four types of sources are available in PLEXUS: gaussian, gamma, external, and sketch. Gaussian and (up to 9th order) gamma sources emit spikes with interspike intervals distributed according to the appropriate probability distribution. An "external" source generates a spike train with interspike intervals specified by data stored in the PLEXUS library. A "sketch" source enables the user to specify a curve representing output spike frequency versus time as shown in Figure 4. The user scales the axes and, with the lightpen, "draws" a curve made up of straight-line segments. The curve is considered to be periodic with period equal to the full x-axis scale. Thus, this type of source is well adapted to the representation of the more important types of experimental stimuli.

A cell is described by an internal potential and an internal threshold, both of which decay exponentially toward resting values. If the potential exceeds the threshold, the cell emits a spike (fires). After firing, the cell enters an absolute refractory period during which it cannot fire; after the refractory period the potential and threshold are reset to specified values (which may depend upon the values before firing) which again decay toward the resting values. The internal cell potential is affected by spikes which arrive along incoming fibers. Each input spike may cause either a positive (excitatory) or a negative (inhibitory) change in the cell potential.

A fiber is described by a conduction time and by one of two models for the cell-fiber interface (synapse). In the simpler nonreservoir model each spike which reaches the fiber terminus causes a postsynaptic potential (PSP) of fixed amplitude.[2] In the reservoir model the amplitude of each PSP is proportional to a specified fraction of a reservoir associated with the synapse. Each PSP thus partially depletes the reservoir, but the level then begins to return exponentially to its resting value. In both models for the synapse the rise time of the PSP may be specified.

[2] The PSP is the amount by which the cell potential is altered.

One of the most powerful features of the PLEXUS model is the external source. As mentioned above, this type of source can make use of any data in the PLEXUS library. Thus experimental data may be used as input, making possible a direct link between theory and experiment. Similarly, input data may be the output of some other network; therefore, if a very large network can be decomposed, it can be simulated sequentially.

Many of the parameters for the network components are coupled with standard deviations. If the user specifies a nonzero standard deviation on a parameter, then each time that parameter is used in a computation (i.e. each time that parameter is sampled) its current value is obtained from a gaussain distribution by means of a pseudorandom number generator. The user specifies a starting value for this generator, thereby in effect specifying the string of pseudorandom numbers used in the simulation. Thus the statistical properties of a network's behavior may be investigated, and the effects of the statistics may be isolated from the effects of changes in parameter values.

Associated with each component in a network is a floating point number which represents the time of occurrence of the next "event" for that component. (This time could be infinite.) An event in this context is the firing of a source or cell, the emergence of a cell from its refractory state, or the arrival of a spike at the terminus of a fiber. Although the computation of the time of occurrence of the next event for a source or a fiber is relatively straightforward, the determination of the firing time of a cell involves the solution of a transcendental equation. The algorithm for network simulation searches for the component having the next scheduled event, adjusts the times associated with components affected by this event, determines the time of occurrence of the next event for the component just processed, and then repeats the search until one of the previously mentioned termination criteria is satisfied. If the

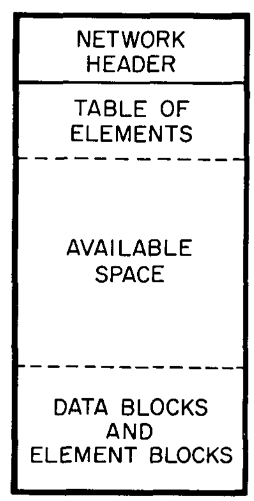

FIG. 5. Partitioning of storage for the data structure of a network

event that occurs is the firing of a source or cell from which output was requested, then the time of this event is saved in a buffer. As necessary, the contents of the buffer are transferred to disk storage, thus permitting the recording of large amounts of output data. For maximum accuracy, all times-of-events are measured with respect to past events rather than on an absolute scale. Thus the output is a sequence of intervals, each interval being associated with a particular component, and each interval being measured from the preceding interval in the sequence. In this manner the accuracy of an item of data is independent of the position of the item in the output record.

Network Data Structure. As the user specifies the topology and the parameters describing a network, a data structure is built which contains all information necessary for use by the simulation algorithm and for regeneration of the relevant displays. Several considerations were involved in the design of the data structure. Apart from the usual need to minimize storage requirements, it was essential to arrange the data representing the network components so that the simulator could reference them efficiently, and so that modifications and additions to the network could be accomplished in a simple manner. Finally, in order to permit storage and reuse of a network over several sessions, and possibly on different computers, it was advantageous to make the structure easily relocatable.

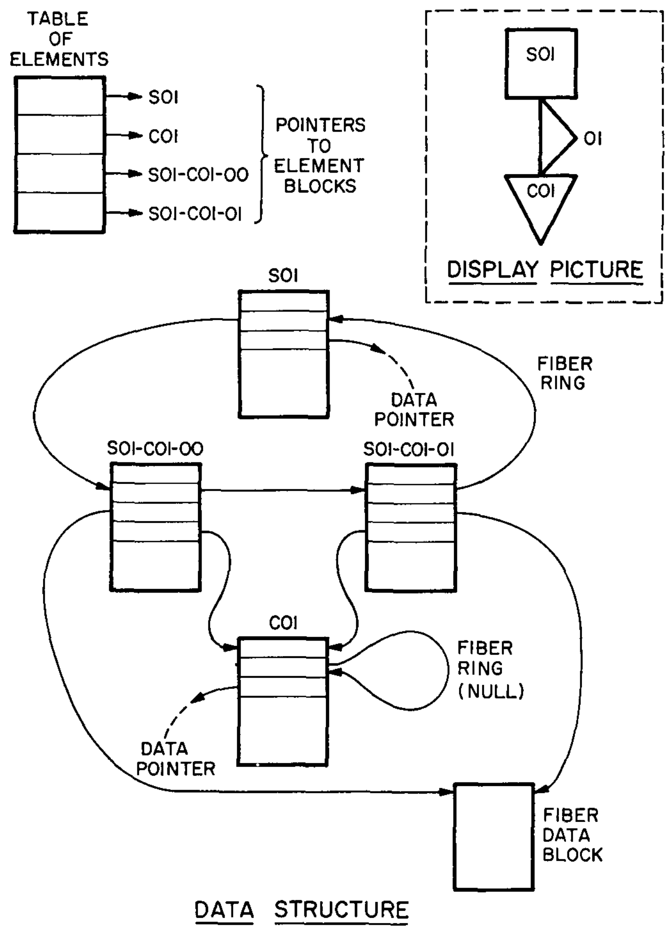

FIG. 6. Schematic representation of the data structure for the small network shown in the upper right portion of the figure. The second and third words of each element block contain a fiber ring pointer and a data-block pointer, respectively. For this network, the two fibers, S01-C01-00 and S01-C01-01, were chosen to be identical, and therefore they share a common data block.

The allocation of space for the data structure is shown in Figure 5. Initially, a large section (approximately 2000 full-words) of contiguous core is provided. A fixed-length header containing information on the current status of the network occupies the first 18 words. Following the header is a table consisting of one 2-word entry for each network component—source, cell, or fiber. Each entry contains the encoded element name, a space for the time of the next event, and a pointer to an "element block" for the component. The element blocks are in unique correspondence with the network components and are linked to reflect the topology of the network. For example, all fibers emanating from a component are put in a ring, thereby facilitating addition or deletion of fibers and permitting convenient processing of output spikes from the given component. Included in each element block is a pointer to a data block which contains the parameter values for that element.

Element blocks and data blocks are assigned to storage locations which are as near as possible to the end of the section of core provided for the network. As the specification of a network progresses, both the table of elements and the area occupied by the data and element blocks increase in size, thereby absorbing the available space which lies between them. The blocks representing different types of components vary in size from 2 words to 134 words and are allocated dynamically by a storage manager. An attempt is made to recover space occupied by any blocks which are returned to free storage as, for example, after the deletion of a network component. Since the blocks vary so widely in size, it is generally not possible to efficiently use all returned space. However, if all contiguous core has been allocated, requests for additional blocks are satisfied either by reusing returned blocks of the proper length, or, if this is not possible, by fragmenting larger returned blocks.

Figure 6 illustrates a small network and its associated data structure. In addition to the pointers shown, each element block contains the coordinates necessary to regenerate the display of that component, as well as space for certain quantities which are used to describe the changing internal state of the component during simulation. These quantities, which are unique to an element, must be kept in the element block rather than in the data block since distinct components may share the same data block (as shown for the two identical fibers in Figure 6). Kept in the data block are the parameter values entered by the user during specification of the component. These values are used in the redisplay of the component's specification page. Also, they are referenced by—but not modified by—the simulation algorithm.

3. Library Description

The library system was designed to complement the neural modeling capabilities of PLEXUS. Since a neural model which attempts to describe the properties of an actual structure within a living nervous system may be quite complex, the user may wish to construct and modify his model gradually, perhaps during several sessions. Thus a completed problem or segments of it should be able to be filed in the library. Moreover, the ability to maintain continuity between sessions allows the user to develop his model in parallel with the collection and analysis of real data.

The basic units operated on by the library system are data sets. These may be divided into the following three categories:

1. Networks—these are the data structures of partially or completely specified models.
2. Output data—as mentioned above, these are sets of ordered interspike intervals produced by individual network components, either sources or cells.
3. Element data—these are the parameter values of individual cells, sources, or fibers which have been entered into the library via the CREATE STANDARD mode.

A data set is identified by an arbitrary 12-character name and a 3-character prefix consisting of the user's initials. The data stored in the library are subdivided according to prefixes. Thus each user of the system maintains his own private library and is free from naming conflicts with other users. Two special prefixes are reserved for system use. The sequence *** denotes a data set which is associated with the current problem. For example, ***NET refers to the current network and ***C03 to the output data from cell 3. The prefix PLX is used to refer to a privileged library of networks, output data, and elements. Users have read-only access to this data.

The library data structure is shown schematically in Figure 7. The master directory contains a file of all prefixes which are currently active in the system, i.e. one entry for each user having stored data. At the present time the master directory contains a minimum of information and serves only to identify existing user directories. In the future, as the amount of stored data becomes large, other attributes (e.g. date of last data access) will be added to facilitate editing. As indicated in Figure 7, the user's directory contains one entry for each data set in his private library. Each entry contains the full alphanumeric name of the data set, its type, its date of creation, and an internal link to the actual data. This separate directory allows rapid determination of the existence of a particular data set, facilitates display of a list of the user's data sets and their attributes, and obviates the need for the data sets to contain any information other than the data itself.

There are five basic functions which may be performed on data sets in the PLEXUS library.

RETRIEVE—This operation locates and verifies the validity of a data set saved in any of the private libraries. The data set may be a network that the user desires to study or data that he wishes to analyze. The user may assume that the retrieved data are in core even though they are not physically moved from disk storage until needed. Considerable time is thereby saved, and difficulties in allocating temporary core storage are avoided since many data sets might be retrieved in sequence, and since data sets can vary in size from just a few machine words to many thousand.

CREATE and REPLACE—These are the basic operations by which new data may be entered in the library. Each operation has two data-set names as operands. In both cases, the second operand must specify a currently existing data set. This operand often has the prefix "***" indicating data pertaining to the current problem. For CREATE, the first operand specifies a new name under which the data referenced by the second operand is to be

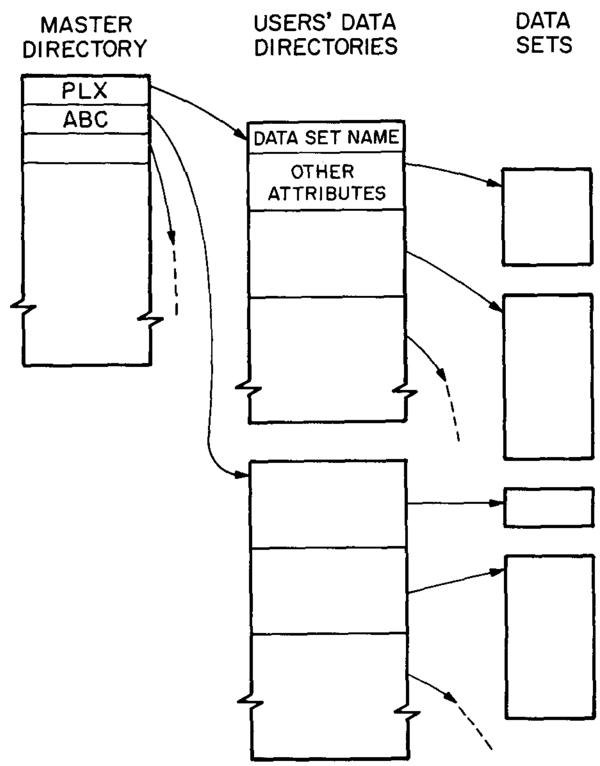

FIG. 7. Internal organization of the PLEXUS library. Examples of users directories and associated data sets are shown for the system prefix PLX and for a hypothetical user ABC. All directories and data sets reside on disk storage.

filed. If the first operand name has not been previously defined, and if the second operand is a valid data-set name, the operation proceeds by entering the name and attributes of the new data set into the user's directory and by making a copy of the data from the existing data set. For REPLACE, on the other hand, the first operand must be the name of an existing data set belonging to the current user of the system. Therefore, a user may only replace a data set stored in his own private library. When the operation is completed, the data being replaced (first operand) no longer exists, but the name is now associated with the data specified by the second operand.

DELETE—A data set may be removed from the library with this operation. Both the data itself and the entry in the user's directory are eliminated. The user's directory is then compacted so that no empty entries exist. If a user deletes his last data set, his entire directory and his prefix in the master directory are removed. As in the case of REPLACE, a user may delete a data set only from his own library.

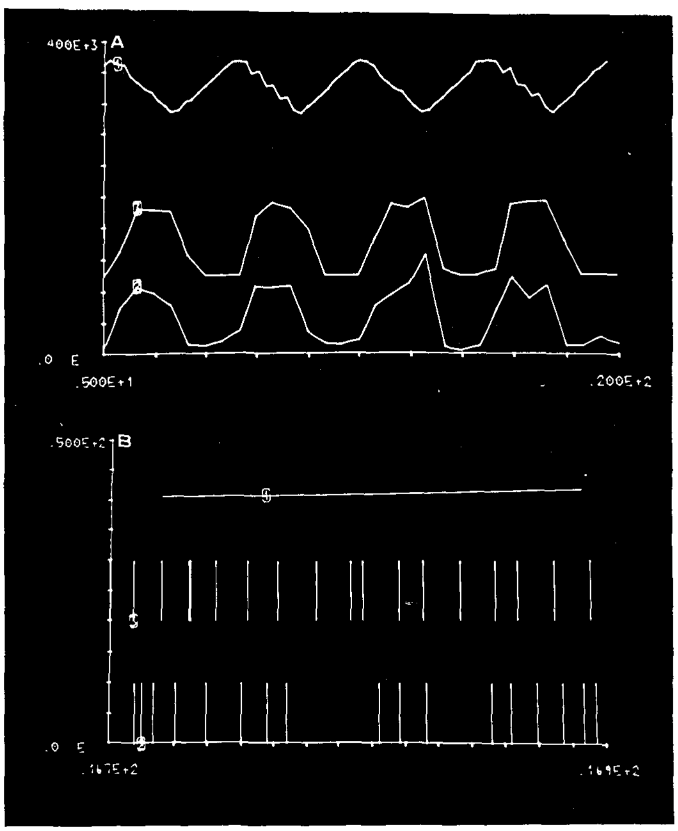

FIG. 8. Graphical display of some typical data. In both (a) and (b) the upper curve represents an experimental stimulus, the middle curve, the output from a PLEXUS network, and the lower curve, the response to the stimulus. For clarity, the upper and middle curves have been shifted. (a) Average frequency versus time for a 15-sec interval. (b) Spike trains corresponding to a small portion of the data shown in (a)

DISPLAY CATALOG—This operation displays a list of the names and attributes of all data sets currently stored under a given prefix. Two function keys allow scanning forward and backward in this list.

4. Data Analysis

After the simulation of a PLEXUS network, the user must be able to examine and perform computations on the data produced in order to determine the behavior of the network. A set of analysis and display programs provides this capability. Since the direct comparison of theoretical and experimental results is of importance, these routines also accept data from experiments.

As previously mentioned, the analysis routines operate under the control of a multiprogramming monitor. The user selects one or more routines by pressing appropriate keys on the function keyboard and then specifying the required input and/or output data-set names.[3] The set of programs that is chosen may include several copies of a particular routine and may also contain routines which depend upon others in the set for their input. The monitor passes control from one routine to another, a given routine losing control when input data are not available or perhaps during normal data access. Since most results are shown graphically, an analysis routine may have one or more display "pages" associated with it. Therefore, when the set of analysis programs in execution needs more than one display page, a queue is formed so that the user may sequence through the different pages.

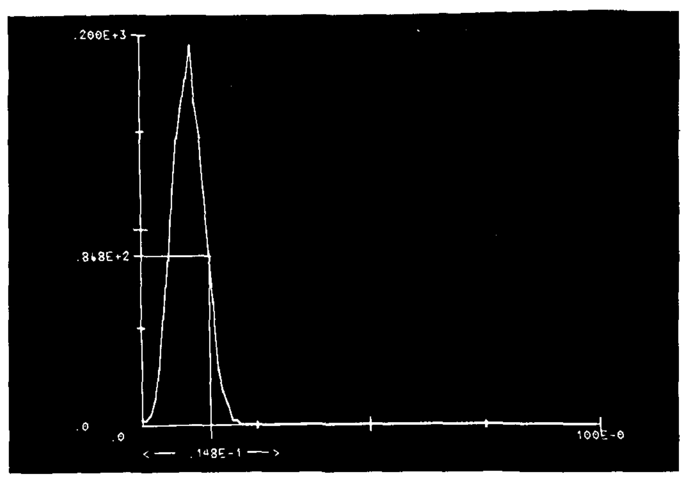

FIG. 9. Histogram of interspike intervals for the PLEXUS data shown in Figure 8a

[3] These data sets reside in the analysis library, as indicated in Figure 1.

Currently, the available analysis routines range from simple displays of spike trains to detailed statistical processes such as auto-correlations and cross-correlations. A user requiring a new technique may easily add an appropriate program to the library of analysis routines. An example of results obtained using one of the simpler routines is given in Figure 8, where data from PLEXUS and from an experiment are plotted as functions of time. In both parts of the figure the upper curve represents a stimulus (motion of an oscillating pattern converted from its analog form) presented to a fly, the middle curve the output of a PLEXUS network, and the lower curve the output of a neuron in the fly's optic lobe. In Figure 8*a* the ordinate is spikes/second; the curves are smoothed by averaging this rate over a user-specified time interval. The middle curve has been shifted for clarity. Figure 8*b* shows in greater detail and in different format a portion of the data from Figure 8*a*; the vertical bars represent individual spikes or neuron firings. The results of a somewhat more complex analysis of the PLEXUS data from Figure 8 is presented in Figure 9. Shown is a histogram of the distribution of interspike intervals in a particular segment of the data.

5. Conclusions

The PLEXUS system was developed and refined over a period of approximately one year, the current version having been in operation since June 1967. Some experience has been gained in application of the system to neurophysiological research. The graphical method of network construction and modification is not only convenient but has also proved to be of value in providing insight into network behavior. The use of identical components and of prototype components stored in the library greatly simplifies network specification. Also, the simulation algorithm is fast enough so that computation of network output typically requires only a small fraction of the total session time. For example, for reasonable parameter values, the network of Figure 2 would produce approximately 500 spikes from both C05 and C06 in about 15 seconds of execution. Although arithmetic computations are performed using only single-precision floating point operations, sufficient accuracy is obtained by storing the output data in the form of interspike intervals. The detailed nature of individual network components, together with the capability for rapid simulation, analysis, and modification, makes practical the investigation of structures representing realistic neural networks. In general, in order to obtain satisfactory results on these problems, more than one session is required. The necessary continuity between sessions is provided by the library. Thus, as work on the problem progresses, a user is able to maintain a complete history of the networks and data leading up to a particular stage of development. This has proved to be quite helpful, since the user may easily return to some previous stage of problem development in case a particular approach proves unsuccessful.

The library of analysis routines is somewhat different from the other parts of the PLEXUS system. As users wish to apply different analysis and display techniques, new routines are added; thus the library is dynamic, its contents continually changing. For example, the analysis routines that were first written permitted observation of only the gross characteristics of the output data. These routines proved to be of great value and are still used in the preliminary analysis of almost all data. However, as experience was gained in modeling, the need for certain more detailed types of analysis became evident. The use of these routines, in turn, has suggested other types of analysis that would be appropriate.

The interactive nature of PLEXUS, its error-checking capabilities, and the ease of transition between its different phases appear to be of help in the development and study of neural network models. Various researchers have easily learned the operations of the PLEXUS system and have found it a valuable complement to their experimental work.

Acknowledgments. We thank R. S. Deverill, D. K. Reece, and W. S. Woodward for their help with programming, and Prof. G. D. McCann for many valuable discussions.

REFERENCES

1. Lockemann, P. C., and Knutsen, W. D. A multiprogramming environment for on-line data acquisition and analysis. Tech. Rep. No. 134, Booth Computing Center, Calif. Inst. of Tech., Pasadena, Calif., June 1967.
2. Caine, S. H., and Huber, H. R. An operating environment for programming research. Programming Rep. No. 1, Booth Computing Center, Calif. Inst. of Tech., Pasadena, Calif. To be published.
3. Gwynn, J. W., and Mahon, M. J. The internal structure of the CIT real time operating system. Programming Rep. No. 3, Booth Computing Center, Calif. Inst. of Tech., Pasadena, Calif., Sept. 1967.
4. Knutsen, W. D., Lockemann, P. C., and Gwynn, J. W. Advanced programming support—user's guide. Programming Rep. No. 5, Booth Computing Center, Calif. Inst. of Tech., Pasadena, Calif. To be published.
5. Perkel, D. H. A digital computer model of nerve-cell functioning. Mem. RM-4132-NIH, The RAND Corp., Santa Monica, Calif., June 1964.