An information processing approach to theory formation in biomedical research*

by H. POPLE and G. WERNER

University of Pittsburgh
Pittsburgh, Pennsylvania

INTRODUCTION

The extensive literature on modeling of biological systems published in the past decade reflects the growing expectation that theories of biological functions can be subject to more exacting tests of consistency with the natural system, and yield more powerful predictions if embodied in the formal structure of computer programs.

One principal aspect of the usefulness of such models of biological systems is attributable to their capability to generate predictions under conditions which transcend the human mind's limited capacity to trace implications through long chains of causal connections, notably when there exist possibilities for multiple interactions between components of a system designed for insuring some degree of homeostasis. Moreover, once sufficiently elaborate, refined and acceptable to a community of investigators, a model could be expected to generate upon request observable facts at various levels of resolution and generality, without having all of these facts explicitly encoded and stored. In this form, the model could be regarded as an information storage and retrieval device from which data can be generated by suitable programs and algorithms.

Irrespective of the particular purpose a model may subserve, it can be considered as an aggregate of components whose initial properties are given in the form of a set of definitions, and whose interactions are determined by suitable transfer functions. The modeler's task is, therefore, twofold: in the first place, suitable algorithms need to be selected to represent the input-output transfer functions of the individual model components. Second, the task consists in designing an explicit structural arrangement of the components, and a pattern of information flow between them, which is suggested or implied by the information in the data base, and which enables the model to mimic the performance of the natural system.

In some sense, one may anticipate that the potential usefulness of computer based models grows with the complexity of the natural domain they represent. However, as the latter increases, model building itself turns into an increasingly more formidable task, frequently complicated by either incompleteness or ambiguity of some of the available observational data. The implication of this is that the relation between a theory and its data base on the one hand, and a corresponding model on the other hand, need not be unique; instead, each given set of observational data of a certain complexity can generate a class of models. Consequently, in order to work toward a parsimonious model compatible with the available evidence, the modeler must engage in the iterative process of identifying alternative interpretations of the data, selecting between them, and evaluating their consequences. In those natural domains whose complexity makes modeling most desirable, this information processing task tends to assume a degree of complexity that impedes model development. To explore ways of assisting the modeler in this task, we have inquired into the nature of the information transactions underlying the modeling of complex biological systems. In the following sections, we discuss the nature of this information processing task. We, then, focus on the problem of structure optimization of neural networks, that is: the adaptive process of fitting together a neural network which, when simulated, gives rise to behavior analogous to that observed in the natural domain. Finally, we discuss computational procedures designed to aid in this process.

SOME ASPECTS OF THE MODELING TASK

In our work with models of complex neural control systems, we have come to recognize the need for drawing...
a sharp distinction between the concept of "hypothesis model" and that of "simulation model." The hypothesis model is what the investigator carries in his head; it consists of a collection of facts and a set of interpretive rules that allows the researcher to perceive relationships in the data, make predictions concerning consequences of untried experiments, etc. A simulation model, on the other hand, is a computer program in which the researcher can express the essential features of his internal hypothesis model, enabling rigorous tests of validity. As such, it requires a degree of explicitness which goes beyond that commonly available in the hypothesis model. We emphasize this dual aspect of the modeling process in order to bring into focus the interplay that takes place between data, hypothesis, and simulation.

In many cases, the implications of biological data, in terms of the scientist's hypothesis models, may give rise to a multitude of possible interpretations. For example, the finding that "electrical stimulation of Deiters Nucleus elicits IPSFS (inhibitory postsynaptic potentials) in gamma-motorneurons" could suggest a variety of structural connectivities; some of those are:

(a) an inhibitory pathway between the cells of Deiter's nucleus and the flexor gamma-1-motorneurons;
(b) an excitatory pathway between the cells of Deiter's nucleus and some spinal-interneurons which, in turn, inhibit gamma-motorneurons on the extensor side;
(c) orthodromic and antidromic conduction along collaterals of axons which originate in some cell group with inhibitory connections to, both, Deiter's nucleus and certain gamma-motorneurons.

While it may be possible to rule out certain of these alternatives on the basis of other data, the researcher is invariably forced—in the course of constructing a simulation model—to make essentially arbitrary choices for which there is no conclusive evidence, either in the theory or in the data.

If a hypothesis model is subjected to a simulation test that is successful in the sense that the latter exhibits behavior which is consistent with the predictions by the former, then the researcher is reinforced in the choices that were made. On the other hand, if this correspondence is not attained, a reexamination of data, of hypothesis model, and of simulation model performance may direct the investigator to more rational choices. This is one of the principal benefits of using simulation: it can reduce the degrees of freedom available to the model builder, and can force a restructuring of his hypothesis model which, in the absence of a simulation experiment, might not take place.

However, in order to exploit the full value of negative simulation results, the researcher needs to know much more than simply that the results were unsatisfactory. Of course if that is all he knows, he can still proceed with revision of the model and additional experimentation. He might, for example, go to some point in his hypothesis model where an arbitrary choice from among alternative interpretations of the data had been made, and simply try another choice. In following such an exhaustive search procedure, he may be forced to go through a large number of variations of the simulation model until one is found that supports his theoretical model. Provided he has the patience to persevere in this endeavor, it could lead ultimately to an enhancement of his theory.

As is the case in most such search processes, it is reasonable to expect that some heuristic procedure which makes use of the detailed information developed in the course of a simulation, would provide substantial improvement in terms of the time and effort required to obtain a fit with the data. The important questions are:

(a) what kind of information does the researcher need; and
(b) in what form will this information be of most use to him.

In a later part of this paper, we will describe the kinds of information support systems that we currently have under development to provide relevant inputs for the restructuring of hypothesis models. Their development was engendered by the experience gained in the design of a simulation model of the central neural control of skeletal muscle tone and reflexes. As a prelude to the discussion that follows, it will be useful to describe at this point the way in which one observes the macro- and micro-behavior in this motor control simulator, further details of which are described in the appendix.

OBSERVATION OF MODEL BEHAVIOR

The main tool for observing macro-behavior in the motor-control system models is the ANALYZE program (see appendix), which is used to develop behavioral comparisons of two states of a model (one of which
typically represents the normal condition, the other some pathology). This program provides a verbal output that describes deviations in behavior in one state of the model, relative to another, along certain key dimensions. Changes in such attributes as amplitude and rate of muscle contraction, force required for smooth passive stretch, frequency of oscillatory behavior, and others are perceived and reported to the user by this processor.

While in the model-building/data-fitting phase of his work, the researcher can use ANALYZE to test his developing hypotheses concerning representation of pathology in the model. For example, he might know from a review of the data that a lesion affecting the motor-cortex (area 4) would be expected to yield predominance of extensor muscle tone in the organism. Moreover, he might already have established in his model a number of connections emanating from the motor-cortex that could explain such a change in behavior. There might, for instance, be an excitatory pathway from this cortex to alpha motoneurons supplying flexor muscles. In approaching a simulation of this pathology, the researcher would therefore start with a number of pre-conceptions and expectations. In particular, since he intends that the structures which appear in the model should have a functional correspondence with their counterparts in the real system, he would expect (and indeed would insist for the model to be considered acceptable) that destruction of the motor-cortex of his model would be comparable, in its behavior manifestations, to ablation of the real structure in the living organism.

Unfortunately, it often happens in the process of model building that such pre-conceptions are not supported by the simulation results, and it becomes then necessary for the researcher to engage in a kind of post-mortem where he reviews his assumptions and the simulation results in an attempt to assess what went wrong. Any number of things could have happened. For example, the researcher might have expected some pathway to be 'turned off' in the pathology because its source of excitation would be all or partially removed. He may find on closer examination that this particular pathway is not sufficiently active in the normal model; this would of course explain the failure to induce the anticipated pathology. There may also be other, more complicated reasons for the model to deviate from the anticipated behavior: for instance the simulated lesion may give rise to widely dispersed changes in the activity pattern that are difficult to trace in the hypothesis model, and which are therefore less likely to have been anticipated during the theoretical analysis.

The ANALYZE program is of limited value in identifying such failures in the reasoning process. While it provides an answer to the question: "How did the model fail?" it does not address the equally important question "Why?" In order to gain insight into why a model fails, we must examine the micro-structure of model behavior and the information flow at key structures along various pathways. The analysis of this information flow in the model typically requires an extensive series of simulation tests, involving various diagnostic procedures in both the normal state of the model and in the intended pathology. Once this analysis has been completed, the researcher must then determine what these new insights concerning information flow imply with regard to his hypothesis models, make the appropriate adjustments, and try again.

The basic tasks involved in this modeling process are illustrated by the flowchart of Figure 1. Here, the nodes H1 and H2 represent hypothesis models corresponding to the two experimental conditions being compared. The arcs connecting these nodes to the box labeled 'Simulation Analysis' correspond to sequences of commands of the form: 'Cut,' 'Tie,' 'Block,' 'Facilitate'—operators which have been provided as part of the motor-control simulator package for use in constructing various versions of the simulation model (see appendix).

Other aspects of the flow chart are less well defined, being part of the informal analysis engaged in by the investigator. The box labeled 'Theoretical Analysis,' for example, represents the researcher's attempts to rationalize the expected change in macro-behavior on the basis of probable changes in micro-behavior; that is, it reflects his attempts at predicting the cause and effect relationships that are operative in the pathology under consideration.

These predictions feed into a comparator labeled 'Post-Mortem Analysis,' which also receives the actual observations that result from running the simulation model. In the event that prediction and observation fail to match, an 'error signal' is generated and is fed back to the box labeled 'Theory Builder' causing a restructuring of the hypothesis models and a reiteration of the verification cycle.
Whenever this procedure requires an alteration of the hypothesis model, the experimenter must examine whether all experimental conditions accounted for in the previous version of the hypothesis model, are still satisfied. Thus, the flowchart of Figure 1 describes just one phase of the total information processing task associated with the development of a general hypothesis model that accounts for a number of different experimental results.

INFORMATION SUPPORT SYSTEMS

The modeling task described in the preceding sections comprises an extremely complex problem in information processing. Although much of the work seems to be of a routine and mechanical nature—tedious is perhaps the right word—there has been no methodology available to relieve the inordinate burden this procedure places on the researcher. To provide an effective research tool, capable of supporting a wider spectrum of the investigator’s activity rather than merely the mechanics of simulation, there is need for computational procedures that will aid in the theory building phases of the total information process, and also facilitate theoretical and post-mortem analysis.

In order to state these objectives more precisely, let us consider again the dilemma of a researcher who is attempting to develop theoretical models to explain data pertaining to the motor-control system. He has access to a number of different kinds of experimental results that can be grouped according to the following classifications:

(a) anatomical
(b) physiological
(c) pathological
(d) pharmacological

Assuming fixed algorithms for the neural elements and the peripheral motor and sensory components of the system, the researcher’s task is that of developing a theoretical model—in the form of a neural connectivity network—which is consistent with the observational data of each of these four domains, and the implications thereof.

What we have set out to do is reduce this task to a computational procedure. Toward this end, we have designed and implemented an inferential processor that can be used to analyze data and propose hypothesis models. Because of its central role in the total information system concept, we discuss next, in some detail, the conceptual basis of this theory building program.

THE THEORY BUILDER

The first step in developing computational procedures for dealing with theory is a formalization of that theory. There are, of course, a number of computational procedures extant—e.g., resolution, model elimination—for dealing with the deductive aspects of theory. However, our problem is concerned not so much with what the theory implies as with what theory is implied. This converse problem, referred to variously in the literature as ‘apagoge’ or ‘abduction,’ is put succinctly by McCulloch:

“... abduction starts from the rule and guesses that the fact is a case under that rule: All people with tuberculosis have bumps; Mr. Jones has bumps; perhaps Mr. Jones has tuberculosis. This, sometimes mistakenly called an ‘inverse probability,’ is never certain but is, in medicine, called a diagnosis or, when many rules are considered a differential diagnosis, but it is usually fixed, not by a statistic, but by finding some other observable sign to clinch the answer.”

If we express the syllogism contained in this argument symbolically:

\[
\begin{align*}
&T_x \supset B_x \\
&B_a \\
&\text{Perhaps } T_a
\end{align*}
\]

it is clear that a stronger conclusion ‘Ta’ would be fallacious, and the qualifier ‘perhaps’ is a necessary part of the abduction. The uncertainty that attaches to such a result can be lessened, however—as McCulloch points out—by finding additional evidence to clinch the case. Even another abduction argument can lend support; for instance, if we also have:

\[
\begin{align*}
&T_x \supset C_x \quad \text{and} \quad P_x \supset C_x \\
&C_a \\
&\text{Perhaps } T_a \quad \text{Perhaps } P_a
\end{align*}
\]

the occurrence of two distinct data supporting ‘Ta,’ while only one supports ‘Pa,’ would cause most of us to hypothesize ‘Ta.’

Such support can also come indirectly through a somewhat lengthier chain of reasoning. Consider for
example the set of rules:

\[
\begin{align*}
T_x \rightarrow & E_x \quad T_x \rightarrow B_x \quad B_x \rightarrow C_x \quad P_x \rightarrow G_x \quad B_x \rightarrow A_x \\
D_x \rightarrow & C_x \quad G_x \rightarrow E_x \quad P_x \rightarrow D_x \quad D_x \rightarrow A_x
\end{align*}
\]

We can illustrate the network of implications contained in these premises graphically as follows, where the upward branches at any node denote 'reverse implication':

![Diagram of implications]

If we observe the data: \{Ca, Ea\}—then we might reasonably hypothesize: \{Ta, Ba, Aa\}—since this collection of nodes defines a sub-graph of the net that is mutually supportive:

![Diagram of hypothesis]

Since only 2 of these 5 nodes have actually been observed, however, any assumption concerning the others is merely theory. An alternative theory that might be put forth to account for these same data would be:

![Alternative diagram]

where again, only 'Ca' and 'Ea' correspond to the actual observations.

If we have the opportunity to ask questions, we often are able to discriminate between alternative theories. For example, in this case if we ask "Aa?" then one or the other of the proposed theories will necessarily be set aside. Note that although the pair \{A \rightarrow B, B\} does not yield a definite conclusion, the pair \{A \rightarrow B, \bar{B}\} does.

Provided we have data that can be structured in the form of an implicational or causal net—as above—we can clearly write heuristic computer programs that engage in the abductive process of theory building. The objective of such a program would be to find a single connected subgraph that 'covers' all of the observations; or, failing that, a pair of such subgraphs; or a triple, etc. Since each additional subgraph included in a theory contains an additional basic assumption (top node), the goal of parsimony in theory development would provide a rule for selection from among alternative hypotheses even in the absence of other criteria. If, in addition, we have the ability to ask questions of the user, of a simulation model, and of the structured and unstructured data files, the power of our program to develop sound theory is clearly enhanced.

We have encoded and successfully demonstrated a heuristic processor, using a combination of LISP\textsuperscript{10,11} and GOL\textsuperscript{4} routines, that constructs hypothesis models on the basis of the heuristic procedure outlined above. We are in the process of interfacing this Theory Building program with a natural language query system that also possesses some inferential capability, subsequently referred to as ENGOLISH.\textsuperscript{9} This makes it possible that:

(a) an uninterpreted literature data base, in the
form of ENGLISH data structures, can be accessed by the theory builder program, and inconsistencies, ambiguities, and incompleteness in the data base recorded.

(b) the theory builder program can be invoked to develop hypothesis models while engaged in an ENGLISH dialog with the user; thus avoiding

(GOLDE SUGGESTS
(S0000
(EXT ((P Q R S U V W X Y Z))
 ((PROJECTS P Q) ((POLARITY X) (CELL-TYPE P Y) (CELL-TYPE Q Z))
 ((PATHWAY MONOSYNAPTIC X Y Z))
 ((ELICITS (P Q) (R S))
 ((PATHTYPE P R W X))
 ((PATHWAY W X Q S))
 ((PATHWAY POLYSYNAPTIC P Q R)
 ((MATCHUP P Y W) (DISTINCT Q S)
 (DISTINCT S R)
 (ORTHODROMIC Z))
 ((PATHWAY MONOSYNAPTIC X Y Z))
 ((PATHWAY POLYSYNAPTIC P Q R)
 ((ORTHODROMIC X) (MATCHA P X Y W)
 (DISTINCT Q S)
 (DISTINCT R S))
 ((PATHWAY POLYSYNAPTIC Y Q S) (PATHWAY Z W S R))
 ((PATHWAY ANTIDROMIC P Q R)
 ((ORTHODROMIC X) (MATCHA P X Y W)
 (DISTINCT Q S)
 (DISTINCT R S))
 ((PATHWAY MONOSYNAPTIC Y S Q) (PATHWAY X W S R))
 ((INTERRUPTED (PATHWAY MONOSYNAPTIC P Q R) U)
 ((GOLDIG ((LAMBDA NIL (OR (DESTROYED R) (DESTROYED Q))))) U)
 ((PATHWAY MONOSYNAPTIC P Q R))
 ((INTERRUPTED (PATHWAY POLYSYNAPTIC P Q R) U)
 ((MATCHUP P Y W)
 (DISTINCT Q S)
 (DISTINCT S R)
 (ORTHODROMIC X)
 (GOLDIG ((LAMBDA NIL (OR (DESTROYED Q) DESTROYED S))) U)
 ((PATHWAY MONOSYNAPTIC Y Q S) (PATHWAY X W S R))
 ((INTERRUPTED (PATHWAY POLYSYNAPTIC P Q R) U)
 ((MATCHUP P Y W) (DISTINCT Q S)
 (DISTINCT S R)
 (ORTHODROMIC X))
 ((PATHWAY MONOSYNAPTIC Y Q S)
 (INTERRUPTED (PATHWAY X W S R) U))
 ((INTERRUPTED (PATHWAY ANTIDROMIC P Q R) U)
 (ORTHODROMIC X)
 ((MATCHA P X Y W)
 (DISTINCT Q S)
 (DISTINCT R S)
 (GOLDIG ((LAMBDA NIL (OR (DESTROYED Q) DESTROYED S))) U)
 ((PATHWAY MONOSYNAPTIC Y S Q) (PATHWAY X W S R)))
 ((INTERRUPTED (PATHWAY POLYSYNAPTIC P Q R) U)
 (ORTHODROMIC X) (MATCHA P X Y W)
 (DISTINCT Q S)
 (DISTINCT R S))
 ((PATHWAY MONOSYNAPTIC Y S Q)
 (INTERRUPTED (PATHWAY X W S R) U))))
This table is encoded as a GOL extensional data structure in which the symbols: (P, Q, R, S, U, V, W, X, Y, Z) stand for universally quantified variables. Each entry of the table consists of three parts, (A B C), which can be read:

"C implies A under conditions B"

For example, the first entry would be interpreted:

A(P, Q, X, Y, Z) pathway (monosynaptic, X, Y, Z) ⊆ projects (P, Q); where (polarity (X) ∧ cell-type (P, Y) ∧ cell-type (Q, Z)).

In this expression,

'pathway' predicates a monosynaptic connection, of polarity X between cells of type Y and type Z.

'projects' predicates a fiber projection between neural structures P and Q.

'polarity' is the property: {excitatory, inhibitory}; and

'cell-type' is a relation that associates the names of neural structures with the various neural populations subsumed thereunder.

The operation of abduction, using the SUGGESTS data structure entails an associative access whereby some observational datum (e.g., 'projects motor-cortex-4 flexor-alphas') is matched against the first position (A) of appropriate entries of the table, with the result a hypothesis (C) qualified by the expression (B). The reason for employing quantifiers and qualifiers in the SUGGEST-Table is that this permits large numbers of nodes to be represented compactly by a single entry; furthermore, it enables the use of much more powerful search techniques than would otherwise be possible. This refined principle of operation would appear to have the same relation to simple abduction that resolution holds to Herbrand expansion.²

While these concepts and their relationship to other paradigms of artificial intelligence will be the subject of subsequent publications, we present at this juncture an illustration of their operational aspects. For this purpose, we consider an application of the theory builder in the context of a collection of typical neurophysiological data giving rise to alternative hypothesis models.

In a discussion of the diversity of the effects of cerebellar stimulation, Eccles et al.⁴ describe the various pathways which mediate the inhibitory influence of the cerebellar cortex on a cell group in the vestibular nuclei. Some aspects of this discussion can be stated in terms of the following propositions:

(1) the inferior olive projects monosynaptically via the climbing fibers to the cerebellar cortex;
(2) activity engendered by electrical stimulation in the main inferior olivary nucleus (neo-olive) elicits EPSP's in Purkinje cells;
(3) activity engendered by electrical stimulation in the main inferior olivary nucleus elicits IPSP's in Deiters Nucleus.
(4) electrical stimulation of Purkinje cells elicits IPSP's in Deiters Nucleus.

To deal with these propositions meaningfully, some general background knowledge is required which finds its representation in the data encoded in the SUGGEST-table, given previously:

(a) fiber projections identified by neuro-anatomical procedures are monosynaptic pathways;
(b) electrical stimulation in the typical experimental procedure elicits activity not only in the target cells, but also in fibers and their termination in the vicinity of these cells.
(c) impulse traffic in fiber tracts under condition of artificial electrical stimulation can be either ortho- or antidromic, or both.
(d) chronic lesion of a structure interrupts impulse transmission (both, ortho- and antidromic) in all pathways of which this structure is a part.

As illustrated by the following dialogue and in Figure 2a, when supplied with just these propositions and facts, the theory builder proposes a partial model, consisting of one excitatory and one inhibitory mono-
synaptic pathway:

*(STRUCTURE INFERIOR-OLIVE NEO-INF-OLIVE PALEO-INF-OLIVE)
NIL
*(STRUCTURE CEREBELLAR-CORTEX PURKINJE-CELLS)
NIL
*(STRUCTURE GOLGI-CELLS BASKET-CELLS)
NIL
*(STRUCTURE VESTIBULAR-NUCLEI MEDIAL-VN DEITERS-NUCLEUS SUPERIOR-VN DESCENDING-VN)
NIL
*(OBSERVATION (PROJECTS INFERIOR-OLIVE CEREBELLAR-CORTEX))
NIL
*(OBSERVATION (ELICITS (STIM NEO-INF-OLIVE) (EPSP PURKINJE-CELLS))))
NIL
*(OBSERVATION (ELICITS (STIM NEO-INF-OLIVE) (IPSP DEITERS-NUCLEUS))))
NIL
*(OBSERVATION (ELICITS (E-STIM PURKINJE-CELLS) (IPSP DEITERS-NUCLEUS))))
NIL
*(PONDER)
(PROPOSE MODEL:
  ((PATHWAY MONOSYNAPTIC INHIBITORY)
   NEO-INF-OLIVE DEITERS-NUCLEUS))
  ((PATHWAY EXCITATORY)
   NEO-INF-OLIVE PURKINJE-CELLS))

If now provided with the additional observational datum that a chronic lesion of the inferior olive abolishes the inhibitory response in Deiters nucleus that follows electrical stimulation of Purkinje cells, the program recognizes the inconsistency between the implications of this finding and its internal model. This gives rise to a restructured hypothesis model, as follows (see Figure 2 B):

*(EXPLAIN (ABOLISHES (CHRONIC-LESION INFERIOR-OLIVE) (ELICITS (E-STIM PURKINJE-CELLS) (IPSP DEITERS-NUCLEUS)))))

(THIS FINDING INCONSISTENT WITH PREVIOUS MODEL)

(PROPOSE PARTIAL MODEL):

  ((PATHWAY MONOSYNAPTIC EXCITATORY)
   NEO-INF-OLIVE PURKINJE-CELLS))
  ((PATHWAY POLYSYNAPTIC INHIBITORY)
   NEO-INF-OLIVE DEITERS-NUCLEUS)))

The operation of the theory building program starts with the definition of the domain of discourse: this is accomplished by use of the STRUCTURE—command. The first argument of this command designates the name of a neural structure; subsequent arguments identify its components. The command OBSERVA-
TION is used to perform a preliminary encoding of an observational datum which may be either a statement concerning automical connections, or a description of functional relations. Supplied with these facts,
PONDER is called to invoke the abduction process which generates a parsimonious hypothesis model that accounts for the data supplied thus far.

The continuation of the model building process consists of the addition of new data and their evaluation relative to the existing model. The macro command EXPLAIN is available for this purpose: in the first place, it enables the user to supply a new observational datum; the program, then, ascertains whether this new fact is consistent with the existing hypothesis model: if this is the case, it renders an account of that part of the existing model which has been enriched, either by way of confirmation of an existing aspect of the model, or by expanding its domain. Alternatively, the program may discover an inconsistency between the model it contains and the new datum, in which case it reverts to PONDER to attempt the generation of a new model which accommodates all of the facts it has been given up to that time.

ON MECHANIZED MODEL BUILDING—
AN OUTLOOK

Having presented and illustrated the mode of operation and capability of the theory building program, we are now in a position to assess its role within the total information support system for model building. With reference to the flow chart of Figure 1 which dissected the process of model building into identifiable steps and the interrelations between them, we have demonstrated that the theory building program can generate hypotheses on the connectivity of model components which correspond to the boxes labeled $H_1$ and $H_2$ in that Figure: that is, a connectivity that may obtain under normal, and also satisfies some abnormal, conditions; the latter consists in the example given of an artificial, experimental lesion. The hypotheses concerning connectivity are of a form which permits them directly to be represented in the simulation model by means of the CUT and TIE commands (see appendix).

The example illustrates that the formation of hypothesis models can be mechanized, which was the principal objective we have set out to accomplish. Beyond this, the theory building program contains features which enable it to contribute also in other ways to the development of theoretical models; namely, by posing questions, and by proposing experiments to be conducted in the “simulation laboratory.” To this end, the EXPLAIN command can be used to generate predictions and expectations for meaningful post-mortem analysis (see Figure 1): it directs the investigator’s attention to those components and pathways of the simulation model which are expected to be most prominently involved in the anticipated behavioral differences. The outcomes of the theoretical and the simulation analysis can now be compared; if significant differences obtain—which would signify the negation of some implication of the hypothesis model—this new datum can be fed back and employed in the restructuring of the hypothesis.

Operating in this manner, the theory-building program can now be envisaged to perform the structure optimization of the neural network, relieving the investigator from the tedium of the iterative process of fitting observational data into a useful simulation model. Instead, once he has defined the first principles which will serve as the basis of theory formation—in the form of rules in the SUGGESTS data structure—he is able to assess continuously the sufficiency and necessity of his concept of the natural domain under study, and the consistency of this concept with newly acquired data.

APPENDIX

Modeling tools of the motor control system simulator

The motor control system simulator is a processor that can be used to construct and evaluate simulation models of neural control mechanisms which play a role in the regulation of certain skeletal muscle reflexes and of muscle tone. This requires, in the first place, that the investigator have available a versatile set of commands to define neural structures and to interconnect them by pathways of suitably chosen length (i.e., conduction time). These commands are intended to enable the investigator to formalize his conception of the system under study in the form of a model that can be altered conveniently in accord with new evidence; that can reflect some experimentally induced condition; and that can mimic disease states as well as effects due to administration of pharmacological agents. The second requirement was to provide programming tools which would exercise the model in a manner analogous to the tests used by the experimenter to ascertain the functional capabilities of the natural system. Thirdly, it seemed to us necessary to provide the ability for examining the performance of components in the model, and of the model as a whole, at different levels of resolution, comparable to the procedures used by the experimenter: at one time he may be interested in the fine temporal pattern of activity in one single node of the model; and another time, in a time average of the activity, or in the trend; and at some other occasion,
in the gross overt performance of an effector organ (in the case of the prototype model: muscle length and tone), or merely in the deviation of this performance from a normal baseline; finally, the investigator may wish to recognize the relation between performance of an effector organ and the pattern or activity flow between the interacting components of the (neural) control system. Accordingly, programs to extract and perceive any one of these aspects of the model’s behavior are available for selection by the user.

**Algorithms of system components**

The basic element of the simulated neural system is a stylized neuron whose activity state is described by a quadruple of numbers, each ranging over the numerical values from zero to four. These numbers designate, in this sequence, output, threshold, input, and a term representing a “driving force.” The latter is a constant which sums algebraically with the value of the threshold and enables the user to set a positive or negative bias value on the neuron computation.

These formal neurons can be connected by excitatory or inhibitory links to form a finite state machine in which each neuron can receive an arbitrary number of inputs. These are summed algebraically to determine the net input. Computation within the neural network is deterministic and proceeds in discrete time steps, each corresponding to the synaptic delay of approximately 1 millisecond duration in real time. At each time slice, the input and output values of each neuron, representing a neurophysiologically characterized neuron population, are updated on the basis of the following algorithms:

\[
\begin{align*}
    \text{if } ((\text{input}(t-1) > \text{thresh}) \text{ and } (\text{output}(t-1) < \text{input}(t-1)) \text{ and } (\text{thresh} \neq 4)) &: \\
    \quad \text{output}(t) &= \text{output}(t-1) + 1 \\
    \text{if } ((\text{thresh} = 4) \text{ or } (\text{input}(t-1) < \text{thresh}) \text{ or } (\text{output}(t-1) > \text{input}(t-1)) &: \\
    \quad \text{output}(t) &= \text{output}(t-1) - 1 \\
    \text{else}: \text{output}(t) &= \sum_{k \in K} (\text{output}_{k}(t-d_{k})) 
\end{align*}
\]

(where \( A \) is the set of structures \( K \) that are connected, \( via \) delays of lengths \( d_{k} \), to the \( j_{th} \) neuron.

The neural network controls the length of reciprocally connected flexor and extensor muscles through the algorithm of table 1 which also computes for each value of muscle length and gamma motorneuron output the feedback signals generated by muscle spindles (nuclear bag and chain organs) and Golgi tendon stretch receptors as schematically illustrated in Figure 3 (for review, see Reference 9). The connecting links between the neural control network and the muscle effector system are provided by three motor neurons, each, on flexor and extensor side; and by afferent pathways originating from the length and tension transducers of muscles and tendons. Items 1 through 6 of Table 1 designate the relationship between the respective

<table>
<thead>
<tr>
<th>Table I—Periphery Algorithms</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. motor-output(t) := flexor alpha-motorneuron</td>
</tr>
<tr>
<td>2. ext-motor-output(t) := extensor alpha-motorneuron</td>
</tr>
<tr>
<td>3. gamma1-output(t) := flexor-gamma1-motorneuron</td>
</tr>
<tr>
<td>4. gamma2-output(t) := flexor-gamma2-motorneuron</td>
</tr>
<tr>
<td>5. ext-gamma1-output(t) := extensor-gamma1-motorneuron</td>
</tr>
<tr>
<td>6. ext-gamma2-output(t) := extensor-gamma2-motorneuron</td>
</tr>
<tr>
<td>7. flex-length(t+1) := flex-length(t) + flex-mult * (golgi-output(t) - motor-output(t))</td>
</tr>
<tr>
<td>8. ext-length(t+1) := ext-length(t) + ext-mult * (ext-golgi-output(t) - ext-motor-output(t))</td>
</tr>
<tr>
<td>9. golgi-output(t+1) := (ext-length(t+1) + flex-length(t+1)) / (ext-length(t+1) + flex-length(t+1))</td>
</tr>
<tr>
<td>10. ext-golgi-output(t+1) := (flex-length(t+1) + ext-length(t+1) + flex-length(t+1)) / (ext-length(t+1) + motor-output(t))</td>
</tr>
<tr>
<td>11. chain-output(t+1) := scale (flex-length(t+1) - chain-length(t, kchn))</td>
</tr>
<tr>
<td>12. chain-length(t+1) := chain-length(t) + cmult * (chain-output(t+1) - gamma2-output(t)); (clip at zero)</td>
</tr>
<tr>
<td>13. ext-chain-output(t+1) := scale (ext-length(t+1) - ext-chain-length(t, kchn))</td>
</tr>
<tr>
<td>14. ext-chain-length(t+1) := ext-chain-length(t) + emult * (chain-output(t+1) - gamma2-output(t)); (clip at zero)</td>
</tr>
<tr>
<td>15. ext-bag-output(t+1) := scale (ext-length(t+1) - ext-bag-length(t, kbag))</td>
</tr>
<tr>
<td>16. ext-bag-length(t+1) := ext-bag-length(t) + ext-chain-length(t, kbag)</td>
</tr>
<tr>
<td>17. ext-bag-rate := ext-mult * (ext-golgi-output(t) - ext-motor-output(t)) + emult * (ext-bag-output(t+1) - gamma1-output(t)) + ext-gamma1-output(t) + ext-bag-output(t+1)</td>
</tr>
<tr>
<td>18. bag-output(t+1) := scale (flex-length(t+1) - bag-length(t, kbag))</td>
</tr>
<tr>
<td>19. bag-length(t+1) := bag-length(t) + bmult * (bag-output(t+1) - gamma1-output(t)); (clip at zero)</td>
</tr>
<tr>
<td>20. bag-rate := flex-mult * (golgi-output(t) - motor-output(t)) + bmult * (bag-output(t+1) - gamma1-output(t)) + bag-output(t+1)</td>
</tr>
<tr>
<td>21. golgi-tendon(t+1) := golgi-output(t+1)</td>
</tr>
<tr>
<td>22. ext-golgi-tendon(t+1) := ext-golgi-output(t+1)</td>
</tr>
<tr>
<td>23. nuclear-chain(t+1) := chain-output(t+1)</td>
</tr>
<tr>
<td>24. nuclear-bag(t+1) := bag-rate(t+1)</td>
</tr>
<tr>
<td>25. ext-nuclear-bag(t+1) := ext-chain-output(t+1)</td>
</tr>
<tr>
<td>26. ext-nuclear-chain(t+1) := ext-bag-rate(t+1)</td>
</tr>
</tbody>
</table>
motor neurons and the variables appearing in the periphery algorithms; items 21 through 26 designate the sources of feedback signals. The parameters appearing in these algorithms have the following meanings:

- **flex-mult [ext-mult]**: scale factor used to translate the tension changes of flexor [extensor] extrafusal muscle fibers into length changes;
- **emult [eemult]**: a scale factor to accomplish the same for intrafusal muscle fibers of the nuclear chain receptors;
- **bmult [ebmult]**: a scale factor for intrafusal muscle fibers of the nuclear bag receptors;
- **scale**: a scaling function that uses the tabular functions KBAG and KCRN (see later) to translate length differences between extra- and intrafusal muscle fibers of nuclear bag and nuclear chain receptors, respectively, into the static component of the transducer outputs.

It might be useful at this point to comment on issues of implementation. We have had access for this development to a medium scale PDP-10 system, on which a superb LISP interpreter is available. To provide the user a convenient means for interacting with his simulation models, we have encoded a set of supervisor and exerciser routines—to be described presently—that operate in the context of this LISP interpreter. Since LISP is singularly inefficient in its provision for numerical computation, those algorithms described above which define computations at the neural nodes as well as in the peripheral effector and sensor organs have been encoded in FORTRAN, and the resulting compiled modules have been incorporated in the LISP system as callable subroutines. Thus in the discussion that follows, although the notation used to illustrate features of the simulation system if that of LISP, the reader should be aware that the actual simulation results are produced by execution of the appropriate FORTRAN subroutines, upon direction by the LISP supervisor programs.

**Command structure for model building**

The devices needed to build a model and to alter connections, add or delete structures, specify delays (i.e., conduction times between neural structures, in milliseconds real time) and to pass numerical parameters to the computational algorithm are available in the form of a set of LISP functions. Examples of some steps in the building of a model are illustrated in the following interactive dialogue with the model building supervisor:

```
*(ADD NEURON @N-RUBER 1$)
N-RUBER
(49 0 0 1 0 0) (IMPINGE+) (IMPINGE-)
NIL
*(ADD DELAY DEEP-CEREBELLAR-NUCLEUS 15$)
DELAY40
(INPUT: 3)
*(ADD DELAY N-RUBER 20$)
DELAY36
(INPUT: 49)
*(TIE EXCITATORY DELAY40 N-RUBER$)
N-RUBER
(49 0 0 1 0 0) (IMPINGE+ DELAY40)
(IMPINGE-)
NIL
*(TIE EXCITATORY DELAY36 FLEXOR-ALPHA-MOTORNEURON$)
```

Figure 3—Schematic display of an antagonistic muscle pair (F = flexor, E = extensor muscle), and their efferent and afferent innervation. MN = motorneurons supplying the extrafusal muscle fibers; γ-m and γ-MN's, gamma-motorneurons supplying the muscle fibers of nuclear bag and nuclear chain transducers, respectively, from which the afferent nerve impulses signaling muscle length originate. Golgi = receptors situated in muscle tendons and signaling muscle tension.

From the collection of the Computer History Museum (www.computerhistory.org)
In this example, the user wished to add a new structure to the model, namely nucleus ruber (N-ruber), and to connect it via appropriate delays so that it would get excitatory input from the deep-cerebellar-Nucleus (already in the model), and deliver excitatory output to the alpha- and gamma-1 motorneurons on the flexor side. The instruction (ADD N-RUBER) requires specification of the new structure's threshold which was chosen to be 1.

Upon requesting the addition of delays (ADD DELAY with the statement of point of origin and duration), the program returns the call by assigning a number to the new delay (the numbers 40 and 36 were assigned, respectively).

The counterpart to ADD is the instruction DELETE (not shown here).

Once all new structures are added and their thresholds and delay length specified, the user applies (TIE) which calls for three arguments: excitatory (or inhibitory)

name of the structure of origin
name of the structure of endpoint of new link.

The input to the newly added delays is already furnished by the instruction ADD. After the new connection is established, the program lists all excitatory and inhibitory inputs to the recipient structure of the new connection in the form if (IMPINGE+ ...) and (IMPINGE- ...), respectively.

The counterpart to TIE is CUT which severs existing connections and requires the same format of arguments as does TIE.

The user then decides to save the new model under the name of RUBER.

Two more changes are then made: a driving force of +2 is added to the Deiters-Nucleus by means of the instruction (TRY DRIVER... value). This means that deiters nucleus will now have an output of two in the absence of any excitatory input, and this value of two will be added to any output level computed (with the ceiling value, of course, remaining 4). Furthermore, the sensitivity of the nuclear bag end organs is changed: first, the user examines their old value with KBAG; there are 4 criterion levels set (1 through 4); any length difference between extra- and intrafusal muscle fibers up to 25 generates an output of 1; similarly, the values 2, 3 and 4 are generated by the specified length differences. These criterion values are changed by STORE (KBAG etc.).

Neurological lesions can be simulated by the command (KILL—name of structure—) which sets the threshold of this structure to the value of 4 and, thus, effectively eliminates the structure from the flow of activity in the network.

The command BLOCK, decreases the activity level of a set of one or more structures by a specified value: for example, (BLOCK T 1 @(RAS RAS-INTERNEURON)). The neural structures within the parentheses are affected. The converse effect is generated by the command (FACILITATE T +1 @(RAS)) which mimics an excitatory drug effect.
OBSERVATION OF MODEL PERFORMANCE

To enable testing the performance of the model with the types of maneuvers applied by neurophysiologists or neurologists to evaluate the functional state of the motor system, two functions are defined:

**PULSE**, designed to mimic a brief muscle pull, and consisting in the application of input pulses to the muscle spindle afferents of a specified muscle for a brief, specified period of time. This function is intended to test the motor system for its ability to support a phasic stretch reflex of the kind commonly employed in the form of the "knee jerk."

**STRETCH**, on the other hand, is designed to test for active resistance to muscle stretch, developed in the course of slow extension of the muscle under study, with rate and duration of passive muscle stretch to be specified by the user. The muscle tension developed during passive stretch is computed as the incremental activity required at the Golgi stretch receptors to sustain the specified rate of muscle elongation during the stretch interval.

For the assessment of the model’s response to these kinds of stimuli, the important variable to be observed is that of muscle length. In addition, the user may wish to observe activity levels of selected neural structures. He can accomplish this defining a “WINDOW” that lists these structures by name.

Even if only a few structures of the model are being monitored, the output generated as a result of several hundred state transitions in discrete time becomes bulky and cumbersome to interpret; details that should stand out to attract the user’s immediate attention, tend to become obscured. Thus, some compression of output is needed to enable quick and reliable “perception” of the performance.

For the purpose of perceiving the characteristic features in muscle behavior following PULSE or STRETCH perturbations, we view the resulting transient departure from resting muscle length as comprising four states (Figure 4):

- **State 1**: commencing with the application of the stimulus and extending until deviation of muscle length exceeds criterion level;
- **State 2**: follows state 1 and extends until resting muscle length is recovered; value and time of occurrence of the extremum, and time till recovery are recorded.
- **State 3**: follows state 2 and encompasses the period during which the primary overshoot occurs; its magnitude is recorded.
- **State 4**: encompasses the remainder of the observation period during which mean muscle length, maximal excursions and frequency of oscillation are recorded.

In the case of what would be considered a normal reflex, the observation terminates in state 4. Abnormal or abortive reflex behavior can cause the perception algorithm to terminate in any of the earlier states, in which case appropriate messages are generated.

For the neural structures named in a WINDOW, the compressed output consists of a mean activity level over the observation period and the number of oscillations that may have occurred. Moreover, as shown below, the algorithm records duration and time of occurrence of the longest sustained periods of high and low levels of activity. Although not illustrated, the program also computes time of occurrence and intensity of the first burst of activity triggered by the test procedure; this reflects the temporal dispersion of transient activity in the neural network.

*(SETQ WINDOW @(N-VENTRALIS-ANTERIOR GAMMA1-MOTORNEURON))
(N-VENTRALIS-ANTERIOR GAMMA1-MOTORNEURON)*

**PULSE**
FLEXOR-MUSCLE
STARTING-LENGTH 1600
EXTREME-VALUE 800, OCCURS-AT 75
STARTING-VALUE-REGAINED-AT 240,
OVERSHOOT 32

Figure 4
STEADY-STATE-VALUES: MEAN 999, STANDARD-DEVIAITION 51 HIGHEST-VALUE 1088, LOWEST-VALUE 908, #-OSCILLATIONS 28 FINAL-VALUE 1036
DO-YOU-WISH-TO-EXAMINE-WINDOW *YES

N-VENTRALIS-ANTERIOR: MEAN 17, #-OSC 56
LONGEST-RISE 38, OCCURS-AT 112
LONGEST-FALL 20, OCCURS-AT 188

GAMMA1-MOTORNEURON: MEAN 11, #-OSC 70
LONGEST-RISE 20, OCCURS-AT 126
LONGEST-FALL 26, OCCURS-AT 963

While these perception algorithms describe model behavior in absolute and numerical terms, observational records in the natural domain are typically stated in terms of qualitative descriptors, most commonly of a comparative nature; that is: in terms of deviations from a reference state. To enable the modeler to perceive his simulation results in these same terms, we devised the program ANALYZE. This program performs a sequence of tests on two specified states of a model, compares the results and generates a verbal output that describes differences in behavior along certain key dimensions. An application of this program is illustrated below: at first, a lesion of substantia-nigra is produced by applying the KILL-instruction; this causes the behavior of the model to depart from the normal condition in a manner resembling that of Parkinson’s disease in that the response to a PULSE stimulus of the flexor muscle is augmented and abnormally prolonged while the STRETCH of the flexor muscle induces an oscillatory response (i.e., tremor).

*(KILL SUBSTANTIA-NIGRA)
DONE

*(ANALYZE @NORMAL)

((PULSE-FLEX NEVER-RECOVERS LATE-PEAK AUGMENTED-PEAK) (PULSE-EXT ABSENT) (STRETCH-FLEX OSCILLATIONS EARLY-RECOVERY) (STRETCH-EXT NEVER-RECOVERS))

The tremor component of this pathology is in the next simulation run alleviated by combining the lesion of substantia nigra with a second lesion, intended to mimic one form of neuro surgical management of Parkinson’s disease, namely a lesion of the nucleus ventralis anterior of the thalamus:

*(KILL SUBSTANTIA-NIGRA)

DONE

*(KILL N-VENTRALIS-ANTERIOR)
DONE

*(ANALYZE @NORMAL)

((PULSE-FLEX NEVER-RECOVERS LATE-PEAK AUGMENTED-PEAK) (PULSE-EXT ABSENT) (STRETCH-FLEX EARLY-RECOVERY) (STRETCH-EXT NEVER-RECOVERS))

REFERENCES

1 D W LOVELAND
   Theorem provers combining model elimination and resolution
   Machine Intelligence Vol 4 ed B Meltzer and D Michie
   American Elsevier 1970

2 J A ROBINSON
   A review of automatic theorem proving
   Ann Symposium in Applied Mathematics 19:1 1967

3 C S PEIRCE
   Collected papers of Charles Sanders Peirce
   C Hartshorne P Weiss and A Burks eds 8 vols Cambridge

4 W S McCULLOCH
   What’s in the brain that ink may characte
   Presented at the International Congress for Logic,
   Methodology and Philosophy of Science 1964—Reprinted in
   W S McCulloch Embodiments of Mind MIT Press 1962

5 H E POPLE JR
   A goal oriented language for the computer
   In Representation and Meaning—Experiments with
   Information Processing Systems—H Simon and N Siklos
   eds Prentice Hall 1972

6 D ISNER
   An inferential processor for interacting with biomedical data
   using restricted natural language
   AFIPS Conference Proceedings Spring Joint Computer
   Conference 1972

7 H E POPLE JR
   On the mechanization of abductive logic
   In preparation

8 J C M ECCLES ITO SZENTAGOTHAI J SZENTAGOTHAI
   The cerebellum as a neuronal machine
   Springer Publications 1971

9 R GRANIT
   The basis of motor control
   Academic Press 1970

10 J McCARTHY P W ABRAHAMS D J EDWARDS T P HART M I LEVIN
    LISP 1.5 programmer manual
    The MIT Press 1982

11 L H QUAM
    LISP 1.6 manual
    Stanford Artificial Intelligence Project