*Complex Systems* 4 (1990) 151–200

# A Mathematical Theory of Generalization: Part I

David H. Wolpert
*Theoretical Division, Center for Nonlinear Studies,*
*Los Alamos National Laboratory, Los Alamos, NM, 87545, USA*

**Abstract.** The problem of how best to generalize from a given learning set of input–output examples is central to the fields of neural nets, statistics, approximation theory, and artificial intelligence. This series of papers investigates this problem from within an abstract and model-independent framework and then tests some of the resulting concepts in real-world situations. In this abstract framework a generalizer is completely specified by a certain countably infinite set of functions, so the mathematics of generalization becomes an investigation into candidate sets of criteria governing the behavior of that infinite set of functions. In the first paper of this series, the foundations of this mathematics are spelled out and some relatively simple generalization criteria are investigated. Elsewhere the real-world generalizing of systems constructed with these generalization criteria in mind have been favorably compared to neural nets for several real generalization problems, including Sejnowski's problem of reading aloud. This leads to the conclusion that (current) neural nets in fact constitute a poor means of generalizing. In the second of this pair of papers, other sets of criteria, more sophisticated than those criteria embodied in this first series of papers, are investigated. Generalizers meeting these more sophisticated criteria can readily be approximated on computers. Some of these approximations employ network structures built via an evolutionary process. A preliminary and favorable investigation into the generalization behavior of these approximations finishes the second paper of this series.

## Outline of these papers

In section 1 of this paper the topic of generalization is discussed from a very broad perspective. It is argued that it is their ability to generalize that constitutes the primary reason for current interest in neural nets (even though such neural nets in fact generalize poorly on average, as is demonstrated in [1–3]). This section goes on to discuss the benefits that would come from having a particularly good generalizing algorithm. Section 1 then ends with a

detailed outline of the rest of these papers, presented in terms of the preceding discussion of generalization.

Here and throughout these papers, generalization is assumed to be taking place without any knowledge of what the variables involved "really mean." An abstract, model-independent formalism is the most rigorous way to deal with this kind of generalizing.

Section 2 of this paper begins with a mathematically precise definition of generalizers. It then goes on to explore some of the more basic properties that can be required of generalizers and elucidates some of the more straightforward mathematical consequences of such requirements. Some of these consequences (e.g., no linear mathematical model should be used to generalize when the underlying system being modelled is known to be geometric in nature) are not intuitively obvious.

The paradigm here and throughout these papers is to make general, broad requirements for generalizing behavior, and then see what (if any) mathematical solutions there are for such requirements. This contrasts to the usual (extremely ad hoc) way of dealing with generalizers, which is to take a concrete generalizer and investigate its behavior for assorted test problems. In these papers, the behavior defines the architecture, not the other way around. The other way of trying to build an engine which exhibits "good" generalization is, ultimately, dependent to a degree on sheer luck.

Sections 1 and 2 of the second paper present and explore some sets of restrictions on generalization behavior which are more sophisticated than those found in section 2 of the first paper. The first of these new restrictions, explored at length in section 1 of the second paper, is the restriction of "self-guessing" in its various formulations. Intuitively, self-guessing refers to the requirement that if taught with a subset of the full training set, the generalizer should correctly guess the rest of the training set. One of the more interesting results concerning self-guessing is that it is impossible to construct generalization criteria which, along with self-guessing, specify unique generalization of a learning set. (Any particular set of criteria will always be either under-restrictive or over-restrictive.)

Section 2 of the second paper then discusses the restriction of information compactification, which can be viewed as a mathematically precise statement of Occam's razor. Particular attention is drawn to the fact (and its consequences) that at present there is no known way of making an *a priori* most reasonable definition of information measure in a model-independent way.

Finally, section 3 of the second paper is a partial investigation of the real-world efficacy of the tools elaborated in the first paper and in the first two sections of the second paper. References [1–3] consist of other such investigations and show that these techniques far outperform backpropagation [4, 5] and in particular easily beat NETtalk [6]. The tests and investigations presented in section 3 of this second paper are intended to be an extension and elaboration of these tests presented in [1–3].

> "This then is the measure of a man — that from the particulars he can discern the pattern in the greater whole." [U. Merre, from *Studies on the Nature of Man*]

## 1. Background

Neural nets is a field that has recently started to attract interest from many separate disciplines, including physics (especially condensed matter physics), mathematics, computer science, neurobiology, and cognitive science. In addition to the intrinsic mathematical interest in investigating networks of nonlinear mappers, there are several other reasons why neural nets are proving so popular. In this section some of these reasons are discussed, with particular attention paid to the reason of ability to generalize. The usefulness of this ability is discussed, and then an outline of this series of papers, which constitute an investigation into the problem of generalization, is presented from the perspective of the discussion of the importance of the ability to generalize.

### 1.1 Reasons for interest in neural nets

Historically neural nets were first investigated as a means of modeling networks of neurons (hence the name). By integrating models of neurons with models of the connectivities in the human brain, it was hoped that something of how brains function would be uncovered [7–10]. Although this approach is so compelling that it is hard to imagine a mature neurobiology not making use of it, to date neural nets have not made any novel and substantive predictions concerning brain function which were later corroborated in the laboratory [11] (although it can be argued that some aspects of neuromorphology, especially in the hippocampus, are easiest to understand in terms of simple Hebbian neural nets [12]). This lack of major results is probably due to the current extreme lack of knowledge of just exactly how neurons operate and are connected [12, 13]. Given how easy it is to render a human brain inoperative, for example by slightly varying the amounts of various neurotransmitters [14], perhaps it should not be surprising that our inability to model a human brain has accompanied our lacking a detailed understanding of its operation all the way down to the molecular level.

After investigating neurobiology, perhaps the next most obvious use of neural nets is in the field of artificial intelligence (AI). Simply put, since brains are intelligent, it is hoped that by modeling them one could build an artificial intelligence. Unfortunately, insofar as neural nets are unable to substantially aid the field of neurobiology, they also are unable to substantially aid the field of AI. Unmotivated hopes of "collective emergent intelligence," often heard in connection with the field of neural nets [4], seem to be wishful thinking, at least at present. No novel principles of how to design intelligent systems (whatever that means) have yet been discovered through investigations of neural nets. Yet, as pointed out by Clark [15], it is precisely such novel

principles which must be discovered if neural nets are to make any significant addition to the field of AI.

Clark's search for such principles, based on statistical mechanics, touches on the third usual motivation for investigating neural nets, their similarity to spin glasses. This is the traditional entry point into the field for physicists [16, 17]. Unfortunately, although much has been learned of how to model neural nets as spin glasses, no new principles concerning intelligence have emerged from this work to date.

In addition to these three broad reasons for interest in neural nets, there are several features of neural nets which, although of more minor breadth, do constitute concrete results and not simply motivational generalities. The first and perhaps most famous of these grew out of the spin-glass approach to the field. This is Hopfield's [18] (and subsequent researchers' [19, 20]) idea of using neural nets as a means of implementing associative memories in parallel. (See appendix A for a review of how Hopfield's scheme works.) Although Hopfield's associative memory scheme does indeed work, it should be noted that it does not provide any novel insights into how to construct an artificial intelligence. What it does provide is a possible means of making particularly fast associative memories. It constitutes a potential speed-up of something we can already do (build associative memories), but not an entirely novel skill.

Another contribution by Hopfield is the realization that his associative memory neural nets can be used as a means of finding approximate solutions to optimization problems quadratic in their arguments [21]. Recently, the reported efficacy of this technique has come under attack [22]. However, even if the attacks are mistaken and the technique ends up proving useful, as with neural net associative memories it will only constitute a potential speed-up of something we can already do (approximate solutions to optimization problems), not an entirely novel skill.

There is one last application of neural nets which, in addition to being more than just a speeding up of something we can already implement through other means, is actually exhibited by neural nets that are up and running right now. This application is the ability to generalize exhibited by Boltzmann machines [4, 23], perceptrons [24], and in general all feedforward neural nets made via backpropagation [4, 5], simulated annealing [25], or any other technique which does not try to restrict the net's output to the set of attractors typical of Hopfield-type neural nets (and, indeed, typical of all associative memories). (See appendix B for a review of Boltzmann machines and the technique of back-propagation.) That is, it is the ability of such neural nets to be taught with a learning set of input–output pairs, be fed a novel input as a question, and then to form a (hopefully) reasonable response to that question, a response which might differ from all of the outputs in the learning set of input–output pairs [26].[1] It seems that in this ability to gen-

[1]These nets have the advantage over Hopfield-type associative memories in that they can respond to a query with an output not contained in those which make up the learning set. Without such an ability you are not fully generalizing. You are only classifying. In a

eralize neural nets can in fact do something that few other current systems can do, or at least which few other current systems can do as well with as broad a range of applicability. Since it is the most widely studied neural net generalizer, it is the technique of backpropagation that is taken to be the archetypical neural net generalizer for the purposes of this series of papers. It is the hope of the neural net community that this (or some other) neural net scheme somehow picks out the important information from the learning set when generalizing and therefore generalizes well (see reference [4]).

### 1.2 Generalization

A generalizing system has very many potential uses, the most obvious of which lie in the field of statistics. Indeed, as has also been pointed out by Wolpert [1, 2], Lapedes and Farber [27], and by Smith [28], generalizing neural nets can be viewed as simply a novel kind of statistical curve fitting. Clearly, a solution to the problem of how "best" to generalize from a learning set when no *a priori* assumptions are made about the type of function doing the generalizing would have many ramifications for the (nonparametric) statistics problem of how best to generalize (sic) from experimental data to the best model to fit the data.

In addition to this potential application to statistics (and therefore all experimental science), a solution to the problem of how best to generalize from a learning set might be helpful at the task of image reconstruction (the "learning set" here being incomplete information about the full image). It might also be helpful for approximation theory and therefore for numerical analysis. Indeed, at the risk of being overspeculative, a successful solution to the problem of generalization could even be helpful in the broadest scientific task of all, the task of inferring the "best" theory to explain a set of observations.[2]

In addition to these applications, the ability to generalize well also has many potential uses in the field of AI. To understand one such use, first note (as is pointed out in [1] and [2]) that in an abstract sense the goal of AI research is to create a system which meets certain sets of "human-like" criteria. In general, the broader the criteria a system satisfies, the better. When these criteria are specified very precisely — "given $a$, respond with $b$" — then the system created to meet them constitutes a database. As a next step it is straightforward to incorporate into the system an algorithm for making logical inferences from the provided criteria, resulting in what is essentially an expert system. Insofar as databases, being a set of responses to queries, can be viewed as input–output functions over an appropriate space, such expert systems are just extensions of the domain of the original

certain sense, a Hopfield-type associative memory is nothing more than a crude, parallel, Bayesian classifier, since its "basins of attraction" are nothing more than the set of vectors strongly correlated with one another.

[2]See reference [29] for an interesting attempt at solving this last task which might have some applications to the problem of generalization.

database. The logical inference algorithm used has "broadened" the (human-like) criteria defining the properties of the system.

Unfortunately, such logical inference algorithms cannot extend the domain of the system to include the whole of the input space. A purely deductive algorithm can not solve the problem of inductive inference [30]. This is not a trivial shortcoming. Insofar as it is impossible to explicitly list all of the behavioral attributes of human intelligence, it seems that any artificial system that will be able to fully mimic human behavior will have to extrapolate that full behavior from a core subset of (externally provided) behavioral attributes. It would therefore seem that any AI system that can successfully pass the Turing test will have to be able to form an extension of an initial set of behavior criteria. It will have to be able to *generalize* from the original criteria.[3]

To understand the second reason that generalization is crucial to AI, first define a system's general intelligence as a measure of how well it performs at an arbitrary cognitive task for which it is provided limited information, where either the task and/or the information is novel. The system is explicitly told beforehand how its performance is being measured. In an academic setting, the limited information might be a chapter in a math book, the novel task might be a problem set based on that chapter, and the performance measure might be the percentage of problems from the problem set done correctly. At a more primitive level, the limited information might be a novel visual image, the task might be to pick out any creatures which might be predators from that image, and the performance measure might be whether the system gets eaten or not. Alternatively, the limited information might contain a set of images with all predators in them pointed out, as well as the novel visual image. The more generally intelligent a system, the larger the number of (task, information) pairs at which it performs well and the better it performs at them (see appendix C for a more mathematically precise formulation of this definition of intelligence).

Now define *generalization* as the ability of a system to take a learning set of input–output vector pairs from a particular pair of input and output spaces, and, based only that learning set, make a "good" guess as to the output vector corresponding to an input vector not contained in the learning set. The guessing is explicitly independent of any information not contained in the learning set, such as what the input and output spaces "really mean." Loosely speaking, generalization is the ability to perceive and extrapolate patterns in a learning set. It is clear then that in addition to traits like facility with logic and the ability to reason by analogy with other, successfully completed tasks, the ability to generalize well from limited information (i.e.,

[3]Note that as opposed to a conventional "inside-out" system in which the internal structure is externally (and overtly) provided, such a successful mimic of human behavior would be "outside-in" in the sense of only having behavioral criteria externally provided, with the internal structure determined implicitly as a means of meeting those criteria (along with their extrapolation).

from a limited learning set) is a prerequisite for any system to be intelligent over a broad range of tasks.[4]

As a final example of the wide-ranging utility of being able to generalize well, note that if you could construct a generalizer which, for whatever reason, you thought was optimal for a particular collection of learning sets, then by using that generalizer you could define a measure of randomness (and therefore you would be able to decide which of your learning sets was least "random"). The way to measure this generalizer-based randomness of a learning set is to teach the generalizer using various subsets of the learning set and then measure the generalization error on the rest of the learning set. The more random a learning set is with respect to the given generalizer, the more slowly the average generalization error rate should fall off as the size of the subset being used to teach the generalizer increases. If the generalizer can discern very little new about the learning set as a whole as it is given larger and larger samples of that learning set, then as far as that generalizer is concerned the learning set is effectively "random."

The objection has sometimes been made that however you define "good" generalization you can never be assured that a "good" generalizer will produce the correct generalization for an arbitrary real-world guessing situation [31]. However, note that all real intelligences (i.e., people) generalize all the time, in the broadest sense possible, and that such generalizing seems to constitute an essential part of their intelligence. Although in fact people perform such "intelligent guessing" surprisingly well, such guessing carries no assurance of being correct. Rather, it is used as a strategy for dealing with an external environment. Trying to find rules for generalizing well in a series of generalizing situations is similar to using a particular game theory strategy (like minimax say) for a series of different games. No such strategy is guaranteed to give you best results in all games, so you try to find a strategy which will give as good results as possible over a wide variety of games. Similarly, the problem of finding a good generalization scheme is to find a scheme which gives good results over a wide variety of generalizing situations.

While a lot of work has been done that relates to the problem of generalization, none has been done that directly addresses the issue itself:

1. Approximation theory and regularization theory [32, 33] can be viewed as attempts to deal with the problem of generalization when one knows what the input and output spaces "really mean."

2. The field of machine learning [34–36] consists to a large degree of various schemes for how to generalize in certain strictly limited contexts (e.g., having Boolean valued variables).

3. Bayesian classification [37], and in general all of the work on classification in image processing [38], is similar to the problem of generalization,

[4]Note that the problem of "making a best guess for $I_{\text{miss}}$, given $I_{\text{prov}}$ and $f$," discussed in appendix C, is a generalization problem.

the major distinction between the two being that in classification the set of possible answers to questions is strictly controlled and finite (indeed, often having only two elements).

4. Time-series analysis and prediction theory, especially as extended to chaotic systems by Farmer [39] (and consequently, using backpropagated neural nets, by Lapedes and Farber [27, 40]), can be viewed as an *ad hoc* approach to a limited version of the full problem of generalization.

5. Information complexity theory [41] can be viewed as dealing with a generalization problem modified by adding to the condition that they have to reproduce the learning set some extra, rather stringent problem-dependent conditions on the allowable generalizations.

6. The work with local languages [42], with grammatical inference [43], with mathematical criteria for associaters [44], and with finding the minimal high-level language set of statements to reproduce a learning set [45] can also all be viewed as dealing with problems related to that of generalization, but not with the full problem itself.

Not dealing directly with the full problem of generalization, these approaches do not provide any set of tractable mathematical criteria for good generalization. Accordingly, the criterion in common use for judging how well an algorithm generalizes is to simply test the performance of the algorithm on lots of learning sets created with a "correct" generalization in mind. The accuracy of the algorithm in guessing this "correct" generalization from the learning set is taken to be a measure of how well it generalizes. (Although the field of machine learning can make use of more objective criteria in measuring generalization efficacy, since it is restricted to situations in which the number of possible input–output mappings is finite, its applicability to real-world problems is limited.) Clearly, this is an unsatisfactory state of affairs, having little if any potential to meet the promise of a full theory of generalization.

### 1.3 Synopsis of the rest of these papers

It is due to these inadequacies in conventional approaches to the problem of generalization that using neural nets to generalize has excited so much interest. Unfortunately, there does not currently exist any understanding of which features of neural nets are helpful for generalizing and which are not. Nonetheless, of all the motivating causes for interest in neural nets listed in section 1.1, it is the ability of such nets to generalize which seems to be the most substantive and which seems to have the most potential.

It was with the idea of exploring generalization as a more abstract, mathematical problem and (hopefully) thereby improving upon the performance of neural nets that the generalization theory of this series of papers was developed. Generalization theory is an abstract mathematics of generalizers, various reasonable criteria that can be required of them, and the consequences

of those criteria. Some examples of such criteria are invariance under various kinds of coordinate transformations of the input and/or output spaces, having the generalization of part of the learning set guess the rest of it (self-guessing), and information compactification (i.e., Occam's razor). All the generalizers commonly used at present can be categorized in terms of which of these and other similar criteria they meet. Section 2 of this first paper and sections 1 and 2 of the second paper in this series of papers describe generalization theory in more detail, spelling out the categories created by various sets of criteria and investigating the problem of finding a set of criteria which specify a unique generalization of any learning set.

To quantify the advantages of neural nets as generalizers, it is important to develop a benchmark for measuring the efficacy of a system's generalizing from a learning set. Without use of such a benchmark, any claims for neural nets having "emergent and adaptive intelligence" (beyond that which follows simply from their ability to form associative memories) are vague at best, and misleading at worst. Such a benchmark should be very easy to construct, flexible, quick, and possess simple to analyze generalization behavior. (It is important to realize that the unanalyzability of neural nets is *not* reason to believe they generalize well, but rather is just an impediment to seeing precisely how they go about trying to generalize.) The generalization theory criteria discussed in the first paper which most naturally lead to such a benchmark are those defining a HERBIE (HEuRistic BInary Engine). In essence, a HERBIE benchmark generalizer is a simple surface-fitter which creates an input–output surface which goes through all the points of the learning set and has the whole of the inputs space as its domain. A precise definition of a certain kind of benchmark HERBIE (a hyperplanar HERBIE), exploration of its behavior, and use of it to benchmark the generalization of neural nets makes up reference [1]. As is discussed in [1–3], the hyperplanar and other kinds of HERBIEs have been found to be very helpful in benchmarking generalization, and actually beat backpropagated neural nets rather decisively in the tests performed so far.[5]

Despite its usefulness as a benchmark for measuring generalization, however, there is no reason to believe this HERBIE and the criteria defining it will be the best generalizer for an arbitrary learning set. It is not the definitive answer to the question, "If I have a given learning set and several different algorithms which can generalize from it, which one do I choose?" Some of the other generalization criteria which constitute a possible answer to this question are those making up self-guessing and information compactification, which are presented in the second paper. Discussion of some systems approximately obeying self-guessing and information compactification, along with examples of some tests of such systems make up section 3 of the second paper. In addition to utilizing evolutionary (as opposed to neuro-) biology, such systems tend to make use of parallel distributed network structures. In

[5] "Beats" here meaning what it does in the present-day literature: HERBIE does better at guessing the "correct" input–output mapping the researcher had in mind when making up the learning set.

this sense, this series of papers comes full circle, concluding with an exploration of what can be thought of as very sophisticated, feedback, variable number of iteration, neural nets.

## 2. Introduction to generalization theory

As mentioned in section 1, a generalizer is any program which tries to "broaden the criteria" defining a system's behavior by extrapolating from them and making guesses for appropriate behavior when faced with any situation, whether or not it is contained in these original criteria fed to the system. These extrapolations are often "human-like," in that they are not necessarily made according to the rules of logic. Their being nonlogical extrapolators of the database of the defining criteria allows such generalizers to make a guess in situations where such a guess is not decidable by logic alone — the whole of the input space is in their domain. In this section, the foundations of a mathematical framework for exploring such generalizers will be presented and explored. In particular, various sets of criteria which can be required of generalizers will be investigated from within this framework.

For the purposes of this investigation, all broadenings of criteria can be assumed to first transform the space of the criteria into an $m+n$ dimensional (usually) Euclidean vector space, where $m$ of the dimensions span the input space, and $n$ of the dimensions span the output space. Such a transformation can always by done — if need be, every distinct variable in the database can be assigned its own dimension, and every distinct state of the variable can be assigned an arbitrary number. More rigorously, by Church's hypothesis any criteria computable by humans is computable by a Turing machine [46], and therefore can be digitally encoded — this encoding can serve as the desired transformation. In this way, every piece of information in the data base is mapped to a *data vector* in the vector space, producing a numerical learning set. In general, since the vector space is supposed to be an input–output space, it's required that every $m$-dimensional image of an input component of an element of the data base uniquely specifies the associated $n$-dimensional image of the output component of that data base element. All of these considerations also apply to encoding criteria for use by neural nets.

Define the $n$ *separated systems* as the $n$ vector spaces formed by taking the Cartesian products of the $m$-dimensional input space with each of the $n$ distinct output dimensions. The set of all the resultant data vectors for one of the separated systems is called the *learning set* or sometimes the *training set*.

The essence of any generalizer, even a neural net generalizer, is to take each separated system and fit a surface with an infinite domain (i.e., extending over the whole of the $m$-dimensional input space) to the data vectors of that system. More precisely, given an output value for each of $m$-dimensional input vectors, i.e., given $k$ $m+1$-dimensional vectors, a generalizer creates a surface in the $m+1$-dimensional space which passes through all $k$ of the $m+1$-dimensional vectors, and which has an output value for any and all

other input vectors — the domain of the mapping delineated by the surface is the entire input space. To ask a generalizer a question, one takes the appropriate input vector and reads off the output value on the surface generated by the surface-fitting algorithm. The surface's passing through the $n$ original vectors makes the generalizer a superset of the database (for any one of the original input vectors it returns the correct output value), and the surface's having an $m + 1$th dimensional value for any input coordinates constitutes the broadening of the original criteria. Different generalizers use different surface-fitting algorithms. In addition to choosing between different generalizers, in general there will be an infinite number of transformations from the data base to $n$ $m + 1$ dimensional separated systems. Usually it is best to pick such a transformation with as small an $m$ as possible, to channel the information contained in the data to the detailed shape of the surface, which is where we want it.

From an abstract point of view, the generalization problem then is to first create broadly applicable mathematical criteria for whether or not a given generalizer performs well the generalization of a given learning set living in a particular separated system (these criteria for good generalization should not be confused with the criteria making up the learning set). Once this is done, a procedure must be created to take as input any learning set and from it build a generalizer which generalizes in at least approximate accordance with the generalization criteria, for that learning set. To date, several sets of seemingly reasonable generalization criteria have been explored. Although some such sets have turned out to be over-restrictive (certain learning sets cannot be generalized at all), most are under-restrictive (the generalization of a learning set is not unique) and therefore, instead of specifying a unique generalization for any learning set, serve to categorize equivalence sets of such generalizations. In addition to this theoretical work, computer programs designed to approximate some of these (under-restrictive) criteria sets have been tested. As explained in section 3 of the second paper of this series, these programs have usually proven to be good generalizers in such tests, in the sense that they do well at the task of guessing the parent function the researcher has in mind and from which the elements of the learning set were constructed. Such results serve as (partial) real-world corroboration of the reasonableness of these criteria sets.

All of the more sophisticated sets of generalization criteria investigated so far have made use of the criteria of self-guessing and/or information compactification. Information compactification, which is discussed in section 2 of the second paper, has proven to be under-restrictive. Depending on the criteria that go with it, self-guessing, discussed in section 1 of the second paper, has turned out to be either over- or under-restrictive. For the future, it is hoped that some combination of the two will be discovered which is a fully satisfactory generalizer (i.e., which specifies a unique generalization of any learning set). Before discussing these sophisticated kinds of criteria, though, it is first necessary to formulate a rigorous definition of the generalization from a learning set (of arbitrary cardinality) to a mapping from questions

to guesses, and then to investigate some of the simpler sets of generalization criteria. The rest of the current section constitutes such an investigation.

### 2.1 Definition of a generalizer

A generalizer is a set of continuous mappings from learning sets of arbitrary cardinality along with a single question, to an output. More precisely,

**(2.1)** An *m-dimensional generalizer* is a countably infinite set of continuous functions from a subset of $(\mathbf{R}^m \times \mathbf{R} \times \mathbf{R}^m)$ to $\mathbf{R}$, from a subset of $(\mathbf{R}^m \times \mathbf{R} \times \mathbf{R}^m \times \mathbf{R} \times \mathbf{R}^m)$ to $\mathbf{R}$, etc., where $\mathbf{R}$ denotes the space of real numbers and $\mathbf{R}^m$ denotes the Cartesian product of m such spaces. Notationally, an $m$-dimensional generalizer is a set of continuous functions $g\{i\}$ along with associated domains of definition, $i$ being a nonzero natural number, and $g\{i\}$ being from $\mathbf{R}^{i(m+1)+m}$ to $\mathbf{R}$.

The $g\{i\}$ are defined for all of $\mathbf{R}^{i(m+1)+m}$, up to explicit holes in the space discussed below. The last $\mathbf{R}^m$ component of the input to any $g\{i\}$ is referred to as the *question*. The rest of the components of the input constitute the *learning set*. (Sometimes, as above in the discussion of separated systems and as below in the discussion of expansions of a learning set, the term "learning set" will be taken to mean simply the set of all input–output pairs provided to the researcher, as opposed to the set provided to a particular $g\{i\}$. The context should make it clear which definition is being used.) Each $\mathbf{R}^m \times \mathbf{R}$ input–output pair of the learning set is called a *datum space*. The number of datum spaces is the *order* of the learning set. As explained above, any question in generalization theory can be cast such that the image of the guessing function is one dimensional. This is why the mapping is taken to be to $\mathbf{R}^1$.

If desired, decomposition and expansion of generalizers can be performed quite easily. One natural way of doing this is to use a pseudo inner product, mapping to a countably infinite set of real numbers, rather than the usual inner product mapping to a single real number. The pseudo inner product between any two generalizers $G$ and $H$, $\langle G|H\rangle$, is defined as the set of all of the $\langle g\{i\}|h\{i\}\rangle$, where the $g\{i\}$ are the set of functions making up $G$ and the $h\{i\}$ are the set of functions making up $H$. The inner product between the $g\{i\}$ and the $h\{i\}$ can be defined as any of the usual inner products amongst ($L^2$) real functions (see appendix D).

Any $g\{i\}$ can be approximated by a Turing Machine (TM), of course. What is more, any TM can be uniquely specified by a pair of two-dimensional $g\{i\}$. Each of the two $g\{i\}$ takes a real number and an integer as its question: the current state of the tape and the head's position on it is given by the real number, and the internal state of the TM is given by the integer. One of the $g\{i\}$'s has its output interpreted as the new tape and head position and the other one has its output interpreted as the new internal state of the machine. To run a TM, you just feed the outputs of this pair of $g\{i\}$'s back into themselves as inputs, terminating when the integer output corresponds

to the halting internal state of the TM.[6] The number of possible internal states of a given TM sets i, and the rule table for the TM sets both the learning set and how the $g\{i\}$ generalize from the learning set.

In addition to this relation with TMs, any scheme for building neural nets from learning sets can be viewed as a generalizer, though for some such schemes the requirement of the continuity of the $g\{i\}$ has to be relaxed. Since any $g\{i\}$ can be approximated by an appropriate generalizer, it is also true that any $g\{i\}$ can be approximated by a neural net. Similar comments hold for genetic algorithms and classifier systems [47]. A final, and perhaps most important example of a generalizer is any statistical algorithm for finding a curve which has in some sense the least variation from a set of data points. In this case, the learning set simply consists of the data points.

## 2.2 Basic generalization criteria

Beyond those implicit in the simple definition (2.1), there are several other restrictions which immediately come to mind when trying to formulate criteria for generalization. The first of these is that the order in which the datum spaces are presented should be irrelevant:

**(2.2)** Every $g\{i\}$ is invariant under permutation of the datum spaces.

This invariance, being a simple relabeling, is on surer ground than many of those which will be described later. A second restriction is that the $g\{i\}$ must form a database:

**(2.3)** If, for any $g\{i\}$, the value of the question is the same as the value of an $\mathbf{R}^m$ entry in one of the datum spaces, the output of the function is the corresponding $\mathbf{R}$ entry from that datum space.

For example, $g\{2\}(x, y, x', y', x) = y$ for all $x, x', y$, and $y'$. One can investigate generalization schemes in which the data in the learning set is suspect and therefore should not necessarily be followed, but they will not be considered here. For single-valuedness, (2.3) necessitates the following restriction on the domains of the $g\{i\}$:

**(2.4)** For any $g\{i\}$, if any two datum spaces have the same value for their $\mathbf{R}^m$ entries, then they must have the same values for their $\mathbf{R}$ entries.

[6] From this perspective a TM is simply a function taking a two-dimensional input vector to a two-dimensional output vector which is then fed back into the input. One component of the output determines when the process stops, and the other determines the output value of the TM as a whole. Insofar as it is just such an iterated map, it is interesting to note that a TM has the potential to exhibit what is in some sense the richest possible behavior, namely chaos. In the same spirit, it is interesting to note that the Halting Problem in computer science seems to parallel the nonlinear science problem of finding when a chaotic orbit alights within a certain region of the appropriate state space.

For example, $g\{2\}(x, y, x, z, w)$ where $z$ does not equal $y$ is outside the domain of definition for the generalizer. This induces an odd structure on the domain of definition of any $g\{i\}$, turning it into $\mathbb{R}^{i(m+1)+m}$ minus an uncountably infinite number of holes. With the usual definition in $\mathbb{R}^n$ of an open set being a union of open hyperspheres, the domain of definition is part open, part closed. For example, if one datum space of a one-dimensional generalizer has the values $(x', y')$, then another datum space has the following domain of definition: $\{x \in (-\infty, x'),\ y \in (-\infty, +\infty)\} \cup \{x = x', y = y'\} \cup \{x \in (x', +\infty),\ y \in (-\infty, +\infty)\}$. No union of open circles can give this structure around the line $x = x'$. In general, unless otherwise indicated, the domains of definition of the $g\{i\}$ will always be considered to be the full Euclidean space minus the holes caused by (2.4).

In addition to the restrictions listed above, it is usually helpful to impose the following restriction on the domain of the $g\{i\}$:

**(2.5)** Unless $i$, the order of the learning set, exceeds $m$, the dimension of the generalizer, $g\{i\}$ is not defined. Even if $i > m$, $g\{i\}$ is not defined if the values of the $\mathbb{R}^m$ entries of the datum spaces all lie on the same $(m-1)$-dimensional hyperplane.

The rationale for (2.5) can be seen by taking $m = 2$. If the input components of the elements of the learning set are colinear (i.e., there are only two elements in the learning set), then the learning set provides no information for guessing the outputs of points not on the line of the elements of the learning set. It provides no information to fix generalization in any direction perpendicular to the line containing the learning set. Since in any real-world scenario the data making up the learning set will not be infinitely precise, we can never be sure that a given question is exactly on that line, and therefore can never even be sure that our learning set provides any pertinent information for guessing the output to a question which we think is colinear with the learning set.

From now on, unless explicitly stated otherwise, the term "generalizer" will be assumed to imply adherence to restrictions (2.2) through (2.5). Beyond these simple restrictions, there are several others which help distinguish between different types of generalizers and which suggest avenues through which to attack the problem of finding a reasonable set of generalization criteria which fixes unique generalization of any learning set. The rest of this first paper consists of an exploration of these restrictions, their mathematical formulations, and their relations to and interdependencies with one another.

### 2.3 HERBIEs

The first such additional restriction which comes to mind is to assume invariance of the $g\{i\}$ under certain coordinate transformations. A number of the techniques used in classification, image processing, and the like obey such an invariance, under certain conditions. For example, for certain kinds of constraints on the allowed form of a differential probability distribution

over a Euclidean space $\Sigma, P(\sigma \in \Sigma)$, choosing that probability distribution which meets those constraints and which also maximizes the entropy [48], $-\int P(\sigma) ln[P(\sigma)] d\sigma$, will pick out a distribution in a manner which is invariant under all differentiable transformations of the space $\Sigma$. If the space $\Sigma$ is transformed to a new Euclidean space $\Sigma'$ via a transformation operator $T$, and if the constraint is re-expressed in terms of the space $\Sigma'$, then the technique of maximizing entropy over $\Sigma'$ subject to the transformed constraint will produce a distribution $P'(\sigma' \in \Sigma')$ which is simply the image under $T$ of the distribution produced by maximizing entropy over the original space $\Sigma$ subject to the original constraint: $P'(\sigma' = T\sigma) d(T(\sigma)) = P(\sigma) d(\sigma)$. For example, this invariance will be obeyed if the constraint on the probability distributions is fixing the expectation value of the position in the spaces to be some constant.

Of course, it is not true that the technique of maximizing entropy will give a probability distribution invariant under all coordinate transformations for *arbitrary* constraints on that distribution. Perhaps the most well-known example of such an invariance-breaking occurs when the constraint is the microcanonical distribution of classical statistical mechanics [49]. In this case the constraint is to fix the energy of the system to be some constant, and maximizing entropy over phase space subject to this constraint gives a uniform distribution over a surface in phase space. In general, however, the re-expression of that distribution in other spaces will not be uniform, as maximizing entropy over those spaces would require. With a constraint of this nature the invariance under arbitrary coordinate transformations is broken. You will make different predictions using probability distributions constructed by maximizing entropy over different spaces.

Similarly, no nonconstant $g\{i\}$ is invariant under all coordinate transformations. No $g\{i\}$ exists for which

$$g\{i\}(x_1, y_1, \ldots, q) = T[g\{i\}(T'[x_1], T[y_1], \ldots, T'[q])]$$

for arbitrary coordinate transformations $T$ and $T'$. However it can be required (for example) that the $g\{i\}$ obey a scale invariance. More precisely, if, for any $g\{i\}$, in each datum space the $\mathbf{R}$ coordinate is scaled by a real, nonzero factor $k$, the output $\mathbf{R}$ of that $g\{i\}$ is scaled by $k$. Notationally, if $g\{i\}$ takes $(\mathbf{R}^m \times \mathbf{R} \times \mathbf{R}^m \times \mathbf{R} \ldots) \rightarrow \mathbb{R}$, then it also takes $(\mathbf{R}^m \times k\mathbf{R} \times \mathbf{R}^m \times k\mathbf{R} \ldots) \rightarrow k\mathbf{R}$ in the same way. Intuitively, this restriction of scale invariance says that generalization should not be dependent on the output dimension's units.

A similar restriction is to require that if for any $g\{i\}$, for every $\mathbf{R}^m$ entry (including the question) there is the same scaling transformation of the all the axes (by a nonzero real number), then again the output value is unchanged. In the same spirit it can also be required that if every $\mathbf{R}^m$ entry undergoes the same rotation, parity, or translation transformation, then the output value is unchanged. If we make the same requirement of the $\mathbf{R}$

entries, we can summarize this restriction of invariance under these coordinate transformations by saying

**(2.6)** Every $g\{i\}$ is invariant under any rotation, parity, translation, or nonzero scaling transformation, applied simultaneously to the all the $\mathbb{R}^m$, including the question, or to all the $\mathbb{R}$, including the output.

For example, for the $g\{2\}$ in a one-dimensional generalizer obeying the input space invariances,

$$g\{2\}(x, y, x', y', z) = g\{2\}(a + kx, y, a + kx', y', a + kz),$$

for all $x, x', y, y', z, a$ and nonzero $k$. Technically speaking, for multidimensional input spaces translation invariance refers to translation by any constant vector (not simply to translation by a vector proportional to the unit vector), and scaling invariance refers to scaling of any set of components of the input vectors (not simply to scaling of all components simultaneously). In practice this extra freedom is not important, however, since almost all of the cases considered in these papers involves one-dimensional input spaces.

We are now in a position to define the first category of generalizers with which we will work. A HERBIE generalizer is defined to be a generalizer which obeys the criteria (2.2) through (2.6). (In addition, in practice the term "HERBIE" will often be taken to mean a generalizer for which the $g\{i\}$ are straightforward and simple functions of their arguments.) As in referring to neural net as generalizers, algorithms taking learning sets to question-guess mappings will often be referred to as HERBIEs even if their $g\{i\}$ are not everywhere continuous functions of their arguments, so that they are not even generalizers, technically speaking. The context should make it clear when this relaxation of the continuity requirement is taking place. In addition, sometimes the term "HERBIE" will be taken to mean a particular kind of HERBIE generalizer. Sometimes the term will just mean the concept of using HERBIEs to generalize. Again, as always, the context should make the precise meaning clear.

Intuitively speaking, a HERBIE is a generalizer which, owing to restriction (2.6), makes its guesses based purely on the geometric relationship between the question and the data vectors making up the learning set. Whenever you are in a situation in which you thinks the units used should not matter, or where your zeroes should not matter, then you had better use a HERBIE (or an approximation to one). If you do not, then whether you are aware of it or not, you are actually introducing preferred scales, preferred origins, and the like, when you know there are none such in the problem at hand.

Note that the restriction of a generalizer that it be a HERBIE is not powerful enough to force unique generalization of any learning set. For example, many of the standard algorithms for surface-fitting used in statistical data analysis are HERBIEs, with fitting the lowest order polynomial surface possible to the data vectors and using that surface to determine the

question-guess mapping being perhaps the most commonly used example. Another, completely different kind of HERBIE is having the output guessed by the generalizer be the output component of the data vector closest to the question.

Although the individual restrictions making up being a (continuous) HERBIE are all quite weak, taken together they are enough to force the following:

**(2.7)** For any $i$, if the values of the **R** entry of every datum space of a $g\{i\}$ of a continuous HERBIE are equal, the output is this value, regardless of the question.

**Proof.** Let $g\{i\}(x, y, x', y, \ldots, u) = v$ for an $m$-dimensional generalizer ($x, x'$ and $u$ are $m$-dimensional vectors). The **R** value of each datum space is $y$. It is assumed that $u$ does not equal any of the $x, x'$, etc. If it did, then we would have $v = y$ immediately by (2.3) and would be done. By (2.6), $g\{i\}(x, ay + b, x', ay + b, \ldots, u) = av + b$ for all nonzero real $a$ and $b$. In particular, the equality holds for $b = y$. Therefore $g\{i\}(x, y(a+1), x', y(a+1), \ldots, u) = av + y$. Now take the limit of both sides as $a$ goes to 0. Since $g\{i\}$ is continuous, we get $g\{i\}(x, y, x', y, \ldots, u) = y$. ∎

There exist a number of other somewhat trivial properties that follow from a generalizer's being a HERBIE. An example of such a property applies to a one-dimensional generalizer's $g\{2\}$ when the question is halfway inbetween the two elements of the learning set. Let the resulting guessed output be $z$; $g\{2\}(x, y, x', y', (x + x')/2) = z$. Then, using the coordinate transformation invariances we get in order:

$$g\{2\}(x, -y, x', -y', (x + x')/2) = -z,$$

$$g\{2\}((x - x')/2, -y, (x' - x)/2, -y', 0) = -z,$$

$$g\{2\}((x' - x)/2, -y, (x - x')/2, -y', 0) = -z,$$

$$g\{2\}(x', -y, x, -y', (x + x')/2) = -z,$$

$$g\{2\}(x', y', x, y, (x + x')/2) = -z + y + y'.$$

Comparing this with our starting point, $z = y + y' - z$, and therefore $z$ must equal $(y + y')/2$. This kind of reasoning can be extended in an obvious manner to show that any $g\{2\}$ of a one-dimensional HERBIE must have odd symmetry about the point $\{\frac{x+x'}{2}, \frac{y+y'}{2}\}$.

We can also make some comments about the derivatives of any one of the functions making up any HERBIE generalizer if we assume that that function, in addition to being continuous throughout its domain of definition and meeting the restrictions of requirement (2.6), is also analytic. For example, if our generalizer is one-dimensional and we expand $g\{i\}$ about a certain point (i.e., about a certain set of values $x_1, y_1, \ldots, q$) and then evaluate that expansion at points whose difference from the expansion point is

a vector $(a, 0, a, 0, \ldots, a)$, then by translation invariance we know that that expansion's value must be the same for all values $a$. Writing out the Taylor series expansion for $g\{i\}$, this means that $\sum_{i=1}^{\infty} k_i a^i = 0$ for all $a$, where the $k_i$ are linear combinations of derivatives of $g\{i\}$ with respect to the variables $x_1, x_2, \ldots$ all the way up to $q$. Taking derivatives with respect to $a$ of both sides of this equality and then setting $a$ equal to 0, we get that all the $k_i$ must equal 0. For example,

$$k_1 = \frac{\partial g\{2\}(x_1, y_1, x_2, y_2, q)}{\partial x_1} + \frac{\partial g\{2\}(\cdots)}{\partial x_2} + \frac{\partial g\{2\}(\cdots)}{\partial q} = \frac{\partial 0}{\partial a}|_{a=0} = 0.$$

(Interestingly enough, it turns out that we can gain no additional information from the equations involving $k$'s of order higher than $k_1$. This is because the equation $k_n = 0$ follows directly from allowing the expansion point at which the derivatives are evaluated to vary, taking various derivatives of the equation $k_1 = 0$ (now, with the expansion point varying, viewed as an equality of functions rather than as an equality of real numbers), and then forming linear combinations of these resultant equations.)

We can similarly make use of the the requirement of scaling invariance along with analyticity of the $g\{i\}$ to derive restrictions on the derivatives of the $g\{i\}$. If we expand all the $\{x_i\}$ along with $q$ by a factor $b \equiv 1 + a$, where $a \neq -1$, then we can again collect powers of $a$ and get, for example,

$$x_1 \frac{\partial g\{2\}(x_1, y_1, x_2, y_2, q)}{\partial x_1} + x_2 \frac{\partial g\{2\}(\cdots)}{\partial x_2} + q \frac{\partial g\{2\}(\cdots)}{\partial q} = 0.$$

(Again, we gain no information from the equations involving higher-order derivatives since these equations follow directly from this equation involving first order derivatives.) Combining this equation with the one above for translation invariance, we get three equations of the nature

$$(q - x_1) \frac{\partial g\{2\}(\cdots)}{\partial x_1} + (q - x_2) \frac{\partial g\{2\}(\cdots)}{\partial x_2} = 0.$$

Only two of these three equations are independent, so we cannot derive the values of $\partial g\{2\}(\cdots)/\partial x_1$, $\partial g\{2\}(\cdots)/\partial x_2$, and $\partial g\{2\}(\cdots)/\partial q$. However, given $q, x_1$, and $x_2$, it is sufficient to know one of the three derivatives $\partial g\{2\}(\cdots)/\partial x_1$, $\partial g\{2\}(\cdots)/\partial x_2$, or $\partial g\{2\}(\cdots)/\partial q$ to get the other two. So knowing just one of these three derivatives gives all derivatives of all powers. Since we have assumed analyticity, this means that in its dependence on $x_1$, $x_2$ and $q$, the function $g\{2\}$ is in fact fully specified by any one of these three first-order derivatives. In other words, assuming analyticity, the two requirements of translation invariance and scaling invariance remove 2 degrees of freedom from $g\{2\}$ and, in fact, from all the functions $g\{i\}$, and therefore, in a sense, from the generalizer as a whole.

If our generalizer is multidimensional, then the requirement of rotation invariance in the input space will also lead to restrictions on the derivatives. Results similar to all these also follow from the requirements of translation and scaling invariance in the output space.

As it turns out, these and all the other ramifications of the requirements of (2.6) can be derived without recourse to this derivative-based framework. This is done by using the following geometric argument to enumerate all possible HERBIEs.

First examine the case where there are two elements in the learning set and the input space is one-dimensional. Assume $y_1$ and $y_2$, the output components of the two datum spaces, are fixed. We want to see how $g\{2\}(x_1, y_1, x_2, y_2, q)$ varies with the input components, $x_1, x_2$, and $q$. Our restrictions on this varying are those of (2.6), namely that if any vector $(a, a, a)$ is added to the vector $(x_1, x_2, q)$, then the output of the generalizer, $g\{2\}$, is unchanged, and similarly, if the vector $(x_1, x_2, q)$ is multiplied by any nonzero constant then again $g\{2\}$ is unchanged. Whatever the $g\{2\}$ value at a point $(x_1, x_2, q)$, the $g\{2\}$ value is the same at all points on the plane containing the two lines formed by all translations and by all scalings of this point $(x_1, x_2, q)$. (We are implicitly assuming that we do not have $x_1 = x_2 = q$, which is the only case where these two plane-delineating lines are identical and therefore do not specify a unique plane. This assumption does not make us lose any generality in the argument, since we already know what the output must be in the case $x_1 = x_2 = q$.) In other words, the function $g\{2\}$ taking points in the $\mathbf{R}^3$ space of the input components $(x_1, x_2, q)$ to the $\mathbf{R}$ space of the generalizer's output is made up of invariant planes.

To calculate the equation of the plane of invariance containing any particular point $(x_1, x_2, q)$, we solve for the coefficients $a$, $b$, and $c$ in the plane-defining equation $aX_1 + bX_2 + Q = c$ ($X_1, X_2$, and $Q$ being the three coordinates constituting our $\mathbf{R}^3$ space). Since the plane has to be invariant under identical scalings of all three of the coordinates, we know that $c$ must equal 0. Since the plane has to be invariant under identical translations of all three of the coordinates, we know that $a + b = -1$. Therefore, our plane is uniquely defined by the single parameter $a$, which is set by the values $x_1, x_2$, and $q$. Any triplet $x_1 = x_2 = q$ is contained in all of the planes, regardless of the value of $a$, so every plane contains the line going through the origin and the point $(1, 1, 1)$. In addition, translation invariance means that every plane contains lines parallel to the vector $(1, 1, 1)$ but which lie off to the side of the line going from the origin through $(1, 1, 1)$. Therefore, looking down the vector $(1, 1, 1)$ toward the origin, we see all of our planes of invariance edge-on, and any one of these planes can be generated from any other one simply by rotating it about this line going from the origin out to $(1, 1, 1)$. We could parameterize the planes by this angle of rotation instead of by the value $a$ if we wished.

Define $G(a)$ to be the function mapping any value of $a$ (i.e., any set of values $x_1$, $x_2$, and $q$) to the output of the function $g\{2\}(x_1, 1, x_2, 0, q)$. By translation and scaling invariance in the output components of the datum spaces we know that $g\{2\}(x_1, y_1, x_2, y_2, q)$ for arbitrary $y_1$ and $y_2 = g\{2\}(x_1, 1, x_2, 0, q)[y_1 - y_2] + y_2$. In other words, now allowing all arguments of $g\{2\}$ to take on arbitrary values, $g\{2\}(x_1, y_1, x_2, y_2, q) = G(a)y_1 + [1 - G(a)]y_2$, where $a$ is set by $x_1$, $x_2$, and $q$, and $G(a)$ is the dependence of $g\{2\}$ on $a$

for the case where $y_1 = 1$ and $y_2 = 0$. This is a complete enumeration of all possible $g\{2\}$'s for one-dimensional HERBIEs; $g\{2\}$ for one-dimensional HERBIEs is fully specified by a one-dimensional function, $G(a)$, in agreement with our discussion of the derivatives of $g\{i\}$'s of HERBIEs.

A similar analysis can be carried out for arbitrary $g\{i\}$ for HERBIE generalizers of arbitrary dimension. If the dimension of the generalizer is greater than 1, then rotation invariance and the like in the input space can serve to restrict the possible form of the $g\{i\}$'s more than would otherwise be the case. Note that this kind of geometric reasoning does not require any assumptions about analyticity, or even about continuity. Such considerations come in, for example, as restrictions to be made of $G(a)$; they do not come in in our concluding that $g\{2\}(x_1, y_1, x_2, y_2, q) = G(a)y_1 + [1 - G(a)]y_2$ for some $G(a)$. Similarly, the restriction of datum-space interchange symmetry is also simply a restriction on the possible forms of $G(a)$.

### 2.4 LMMGs and upward compatibility

Note that the HERBIE of guessing the closest element of the learning set is not continuous, even when the learning set is fixed so that only the question is allowed to vary. Such a lack of being everywhere continuous is a problem with many real-world HERBIEs. One interesting situation in which such loss of continuity is common is when two points in the learning set approach one another. As an example, consider the "fit the lowest possible order polynomial to the learning set" generalizer. This generalizer is a HERBIE (with the proviso that the definition of a generalizer is here being extended to allow noncontinuity). Yet as we take the limit of one point in the learning set approaching another, the shape of the question-output function will, in general, depend on the direction in input–output space along which that second point in the learning set is approaching the first. Unfortunately, this property violates the requirement of continuity together with single-valuedness of the $g\{i\}$. The same problem occurs for generalizers which try to fit a Fourier series to the learning set, and for the hyperplanar HERBIE generalizer described in [1], even when the surface such a hyperplanar HERBIE creates from the learning set is "smoothed out" to be continuous with finite derivative everywhere in the domain of definition.

Indeed, any so-called "linear mathematical model" generalizer (LMMG) which generalizes by fitting the (lowest order possible) elements of a set of basis functions to the elements of the learning set will have this problem with discontinuities, whether or not it happens to be a HERBIE. The proof of this is appendices E and F (Appendix E being a detailed definition of LMMGs). On the other hand, LMMGs have the nice property that if two of the points in the learning set are identical, the generalizer makes the same guess as it would if it had simply been fed the same learning set but without one of the duplicated datum spaces. (This property is formally defined in restriction (2.8) below). Finally, while on the subject of LMMGs, it is interesting to note that there is only one LMMG which obeys the invariances required of

HERBIEs (requirement (2.6)). This is the LMMG which works by fitting the learning set with polynomials. No other basis set of functions gives an LMMG which obeys scaling and translation invariance in both input and output. And even polynomial LMMGs, strictly speaking, do not meet input space rotation invariance. (The proof of all of this is in appendix G.) What this means is that whenever someone uses a non-polynomial LMMG to generalize, whether they mean to or not they have an implicit preferred origin, preferred scaling dimension, preferred orientation, or perhaps even a combination of all three. Conversely, if you think the system you are modeling with your generalizer *does not* have such preferred spatial characteristics, then under no circumstances should you use an LMMG as the generalizer.

There are other restrictions besides the various coordinate transformation invariances of (2.6) and besides the restrictions implicit in the definition of LMMGs which can be added to the core definition of restrictions (2.1) through (2.5). For example, along the same lines as requirement (2.3), we can require *upward compatibility* of the $g\{i\}$:

**(2.8)** For all $g\{i\}, i > m+1$, if the values of the entries of 2 datum spaces are identical, then the output is equal to the output of $g\{i-1\}$ working on the rest of the learning set and on one of the two identical datum spaces.

For example, $g\{3\}(x,y,x,y,x',y',z) = g\{2\}(x,y,x',y',z)$. Intuitively, (2.8) is expressing the idea that if we add a new point to the learning set which tells us nothing new, our guessing should be unchanged. As an example, the generalizer which works by fitting the lowest-possible-order polynomial surface to the points of the learning set is upwardly compatible, as are all LMMGs, for that matter. Sometimes a generalizer meeting restriction (2.8) (along with (2.1) through (2.5) but not necessarily (2.6)) will be called "semi-proper," whereas a generalizer meeting all of (2.1) through (2.8), including restriction (2.6), will be referred to as "proper." The reason for such definitions will become clear below when we discuss self-guessing. Any proper generalizer is necessarily a HERBIE, though of course not vice versa.

Although LMMGs (for example), while upwardly compatible are not everywhere continuous, with care it is possible to create fully proper generalizers; that is, there are HERBIEs which are continuous throughout the whole domain and which meet restriction (2.8). One example of such a proper generalizer is the nearest-point-averaged (npa) generalizer, described in appendix H. In any case, due to their simple and overt generalizing behavior, HERBIEs are, in general, well-suited to serving as benchmarks for generalization, irrespective of the question of upward compatibility or continuity when elements of the learning set approach one another. Indeed, the hyperplanar HERBIE discussed in [1], although overtly nonupwardly compatible and noncontinuous, was explicitly created to serve as a benchmark for generalization.

## 2.5 Output linearity

In addition to restrictions (2.6) and (2.8), yet another possible restriction on generalization which can be added to requirements (2.2) through (2.5) is to require that the generalizer be *output linear*.

**(2.9)** A generalizer is said to be output linear iff for all of the $g\{i\}$, if $g\{i\}(x, y, x', y', ...u) = v$ and $g\{i\}(x, z, x', z', ...u) = w$, then $g\{i\}(x, ay + bz, x', ay' + bz', \ldots u) = av + bw$, for all $a, b, x, y, z, x', y', z', \ldots u$.

**Examples:**

1. $g\{2\}$ for one-dimensional HERBIEs, shown above to be of the form $g\{2\}(x_1, y_1, x_2, y_2, q) = G(a)y_1 + [1 - G(a)]y_2$, is output linear by inspection. All LMMGs (that is, all generalizers which work by summing functions taken from a basis set in such a way as to reproduce the learning set) are output linear for all cardinalities of the learning set, as is shown in appendix I. For example, the HERBIE generalizer "fit the lowest-possible-order polynomial to the learning set" is output linear. Of course, as is discussed in appendix F, such generalizers are not everywhere continuous in the domain of definition delineated in (2.4), and are not even HERBIEs, strictly speaking, if the dimension of the input space is $> 1$ (see appendix G).

2. The analytic continuation of the generalizer defined by

$$g\{n\}(x_1, y_1, \ldots, q) = \{\sum_{i=1}^{n} \frac{y_i}{|q - x_i|}\} / \{\sum_{i=1}^{n} \frac{1}{|q - x_i|}\}$$

a variation of the surface-fitter used in [2], is an everywhere (in the domain of definition) continuous generalizer meeting all the restrictions of being a HERBIE in full, which is also output linear. This HERBIE, along with its variations, like

$$g\{n\}(x_1, y_1, \ldots, q) = \{\sum_{i=1}^{n} \frac{y_i}{d(q, x_i)}\} / \{\sum_{i=1}^{n} \frac{1}{d(q, x_i)}\},$$

(where $d(.,.)$ is a metric such that for any real constant $k$, $d(kx, ky)$ is proportional to $d(x, y)$), all of which are output linear, continuous throughout the domain delineated in (2.4), and intimately related to radial basis functions and kernel-density estimators, will generically be referred to as "metric-based HERBIEs." Note that metric-based HERBIEs, like npa generalizers, can never guess an answer which either exceeds the maximum of the learning set or is less than the minimum of that learning set. This shortcoming is the primary reason for caution in making use of such generalizers. (To see why this shortcoming holds, simply rewrite the output of the generalizer as $\{\sum_{i=1}^{n} a_i y_i\} / \{\sum_{i=1}^{n} a_i\}$, where all the $a_i$ are positive. It is clear that if you take any but the

largest of the $y_i$ and increase its value, the value of the output must increase, going to the value of that maximal $y_i$ when all the other $y_i$ have been increased to the value of that maximal $y_i$. In other words, the output is bounded above by $\max(y_i)$. For similar reasons, the output is also bounded below by $\min(y_i)$, which concludes the proof.)

3. All HERBIEs are close to being output linear, in that if

$$g\{i\}(x, z, x', z', \ldots u) = w,$$

then $g\{i\}(x, az, x', az', q) = aw$. It is the restriction concerning addition of output components of the learning set that HERBIEs are capable of violating. (One currently open question is whether an *analytic* HERBIE can avoid being output linear.)

4. The npa generalizer of appendix H, which is both a HERBIE and upwardly compatible (i.e., is proper), is not output linear.

The restriction of output linearity can be viewed in a number of ways. On one hand, it can be viewed as a combination of output scaling invariance together with an output-summing property. Viewed another way, in a loose sense, every $g\{i\}$ of an output linear generalizer is like a tensor field defined over the $(m \times i)$-dimensional manifold of all $(x, x', \ldots u)$, with the tensor at any point on the manifold being a linear mapping from the $i$-fold Cartesian product $(\mathbf{R} \times \mathbf{R} \times \ldots)$ to $\mathbf{R}$.[7] Alternatively, if the Cartesian product of all the output components of the datum spaces is taken to itself be a vector space $V$, then any $g\{i\}$ is a dual vector field $V^*$. See [50] and [51] for a general discussion of linear operators.

Note that restriction (2.9) is completely compatible with restriction (2.6). If $b$ and $z$ are constants, (2.6) says that $g\{i\}(x, ay + bz, x', ay' + bz, \ldots, u) = a(g\{i\}(x, y, x', y', \ldots, u)) + bz$ for all $x, x', y, y', \ldots, u$. Plugging into (2.9) implies that $g\{i\}(x, z, x', z, .., u) = z$. This, of course, is exactly (2.7), the restriction we derived earlier from (2.6).

There are a number of properties which must be obeyed by any generalizer which is output linear. The first of these is that

**(2.10)** Any output linear $g\{i\}$ can be written as

$$\text{guess} = h_0(x, x', x'', \ldots, q)y + h_1(x, x', x'', \ldots, q)y' + \ldots$$

To prove (2.10) it suffices to note that by (2.9) any output linear generalizer can be written as

$$\begin{aligned} g\{i\}(x, y, x', y', x'', y'', \ldots, q) \;=\; & g\{i\}(x, 1, x', 0, x'', 0, \ldots, q)y \\ & + g\{i\}(x, 0, x', 1, x'', 0, \ldots, q)y' + \ldots \end{aligned}$$

[7] Rigorously speaking, though, $g\{i\}$ is not really a tensor field unless $m \times i$, the dimension of the manifold, equals 1, the dimension of the spaces whose Cartesian product makes up the space on which the tensors work. There is also the problem, of course, that the mapping is only linear from $(\mathbf{R} \times \mathbf{R} \times \mathbf{R} \times \ldots)$ to $\mathbf{R}$, and not multilinear (see reference [51]).

assuming the individual $g\{i\}$ terms on the right hand side exist. Therefore $h_0(x, x', x'', \ldots, q) = g\{i\}(x, 1, x', 0, x'', 0, \ldots, q)$, for example. Unless explicitly noted otherwise, it will always be assumed that the cardinality of the learning set exceeds the dimension of the input space and that all the input space components of the datum spaces, $x, x', x''$, etc., are different from one another. These two conditions ensure that those $g\{i\}$ terms all exist, so that we can indeed set $h_0(x, x', x'', \ldots, q) = g\{i\}(x, 1, x', 0, x'', 0, \ldots, q)$, etc. (see restriction 2.5).

If we assume, as we often do, that the $g\{i\}$ are everywhere differentiable, then there is a weaker condition than (2.9) which results in (2.10) and therefore in (2.9). Namely, if all of the $g\{i\}$ of a generalizer are differentiable and obey the property that $g\{i\}(x, y, x', y', \ldots u) + g\{i\}(x, z, x', z', \ldots u) = g\{i\}(x, y+z, x', y'+z', \ldots u)$ for all $x, y, z, x', y', z', \ldots u$, then in fact the generalizer is output linear. To see this simply note that this property implies that

$$\lim_{z \to 0} \frac{g\{i\}(x, y+z, x', 0, x'', 0, \ldots u) - g\{i\}(x, y, x', 0, x'', 0, \ldots u)}{z}$$

$$= \lim_{z \to 0} \frac{g\{i\}(x, z, x', 0, x'', 0, \ldots u)}{z}$$

The right-hand side is simply some constant, independent of $y$, so we can write $g\{i\}(x, y, x', 0, x'', 0, \ldots u) = yk + k'$, where $k$ and $k'$ are independent of $y$. Since this property also implies that

$$2g\{i\}(x, y, x', 0, x'', 0, \ldots u) = g\{i\}(x, 2y, x', 0, x'', 0, \ldots u),$$

we know that $k'$ must equal 0. This allows us to write

$$k = g\{i\}(x, 1, x', 0, x'', 0, \ldots u).$$

In a similar manner we see that (for example) $g\{i\}(x, 0, x', y', x'', 0, \ldots u) = y' g\{i\}(x, 0, x', 1, x'', 0, \ldots u)$. Therefore, pulling this all together,

$$\begin{aligned} g\{i\}(x, y, x', y', x'', y'', \ldots u) \;=\; & y g\{i\}(x, 1, x', 0, x'', 0, \ldots u) \\ & + y' g\{i\}(x, 0, x', 1, x'', 0 \ldots u) + \ldots \end{aligned}$$

which is exactly of the form given in (2.10). ∎

As a result of (2.10), we see overtly that the guess to any question is 0 if the output values of all the datum spaces are 0. Requirement (2.2) of invariance under permutation of the datum spaces means that $h_0(x, x', \ldots)y + h_1(x, x', \ldots)y' + \ldots = h_0(x', x, \ldots)y' + h_1(x', x, \ldots)y + \ldots$. Since this must be true for all $y, y', y''$, etc. (up to the restrictions of 2.5), we can conclude that $h_0(x, x', x'', \ldots) = h_1(x', x, x'', \ldots)$, and that $h_j(x, x', x'', \ldots) = h_j(x', x, x'', \ldots)$ for all $j > 1$. In general then, if we define $T_{ij}(h_n)(x, x', x'', \ldots)$ as the function created by interchanging the variables in the $i$th and $j$th slots of the argument list before acting on it with $h_n$ (the first slot being defined to be slot 0),

**(2.11)** $T_{ij} = T_{ji}, (T_{ij})^2 = 1, T_{ii} = 1, T_{ij}(h_k) = h_k$ if $i \neq k$ and $j \neq k$, and $T_{ik}(h_k) = h_i$.

A number of equalities follow directly from (2.11). For example,

$$h_0(x, x', x'') + h_0(x'', x', x) = h_0(x'', x, x') + h_0(x, x'', x')$$

results from (2.11). Also (2.11) means that $h_j = T_{j0}(h_0)$ and therefore the output of any output linear generalizer can be written as $\sum_{i=0}^{n-1} T_{i0}(h_0)y_i$, or $\sum_{i=0}^{n-1} T_i(h)y_i$ for short ($n$ is the order of the learning set). So the output of the generalizer is

$$yh(x, x', x'', \ldots, q) + y'h(x', x, x'', \ldots, q) + y''h(x'', x', x, \ldots, q) + \ldots$$

To ensure that the equalities of (2.11) cannot place us in a situation where we have two expressions which appear to be different but in fact are identical, we must fix a protocol for standardizing the order of the argument list of $h$. The protocol used here is to rearrange the variables occurring in the slots beyond the first in increasing order. So, for example, instead of writing $h(x''', x'', x, x', q)$, we would write the equivalent expression $h(x''', x, x', x'', q)$. Therefore, we standardize how we write the output of an output-linear generalizer as

$$\begin{aligned} yh(x, x', x'', x''', \ldots, q) \;&+\; y'h(x', x, x'', x''', \ldots, q) \\ &+\; y''h(x'', x, x', x''', \ldots, q) + \ldots \end{aligned}$$

As was mentioned, there is a slight caveat to the equality

$$h(x, x', x'', \ldots, q) = g\{i\}(x, 1, x', 0, x'', 0, \ldots, q)$$

presented just below equation (2.10), which comes from the fact that the $g\{i\}$ on the right-hand side of the equality must exist: the equality holds only so long as $x$ does not equal any of the variables $x', x'', \ldots$ etc., since $g\{i\}$ is not defined for such a set of argument values, given that $y$ does not equal any of the variables $y', y'', \ldots$ etc. This does not mean that $h(x, x, x'', \ldots, q)$ is necessarily undefined, simply that it does not equal $g\{i\}(x, 1, x, 0, x'', 0 \ldots, q)$ (which *is* undefined). Indeed, $h(x, x, x'', \ldots, q)$ can be consistently defined without any difficulty, assuming that

$$g\{i\}(x, 1, x, 1, x'', 0, \ldots, q)$$

is defined (see 2.5). To do this, write the output-linear generalizer

$$g\{i\}(x, y, x, y, x'', y'', x''', y''', \ldots, q)$$

as

$$\begin{aligned} yg\{i\}(x, 1, x, 1, x'', 0, x''', 0, \ldots, q) \;&+\; y''g\{i\}(x, 0, x, 0, x'', 1, x''', 0, \ldots, q) \\ &+\; y'''g\{i\}(x, 0, x, 0, x'', 0, x''', 1, \ldots, q) \\ &+\; \ldots \end{aligned}$$

Exploiting datum space interchange symmetry between the unprimed and doubly primed variables, we can also write this expression as

$$\begin{aligned} y''g\{i\}(x'',1,x'',1,x,0,x''',0,\ldots,q) \; &+ \; yg\{i\}(x'',0,x'',0,x,1,x''',0,\ldots,q) \\ &+ \; y'''g\{i\}(x'',0,x'',0,x,0,x''',1,\ldots,q) \\ &+ \; \ldots \end{aligned}$$

Therefore, equating the coefficients of $y$, $g\{i\}(x,1,x,1,x'',0,x''',0,\ldots,q)$ must $= g\{i\}(x'',0,x'',0,x,1,x''',0,\ldots,q)$. Similarly, equating the coefficients of $y'''$,

$$g\{i\}(x,0,x,0,x'',0,x''',1,\ldots,q) = g\{i\}(x'',0,x'',0,x,0,x''',1,\ldots,q).$$

If we define $H(x,x'',x''',\ldots,q) \equiv g\{i\}(x,1,x,1,x'',0,x''',0,\ldots,q)$, the equality arising from equating the coefficients of $y$ means that we can write the output of the generalizer as

$$\begin{aligned} yH(x,x'',x''',\ldots,q) \; &+ \; y''H(x'',x,x''',\ldots,q) \\ &+ \; y'''g\{i\}(x,0,x,0,x'',0,x''',1,\ldots,q) + \ldots \end{aligned}$$

On the other hand, we want be able to extend (2.10) to our case where input-space arguments are equal and have the output of the generalizer equal

$$\begin{aligned} 2yh(x,x,x'',x''',\ldots,q) \; &+ \; y''h(x'',x,x,x''',\ldots,q) \\ &+ \; y'''h(x''',x,x,x'',\ldots,q) + \ldots \end{aligned}$$

This can done if we equate $h(x,x,x'',x''',\ldots,q)$ with

$$\frac{1}{2}g\{i\}(x,1,x,1,x'',0,x''',0,\ldots,q).$$

Since we also evidently have $h(x,x'',x'',x''',\ldots,q) = H(x,x'',x''',\ldots,q)$, we also see that $h(x,x'',x'',x''',\ldots,q)$ must equal $1/2h(x,x,x'',x''',\ldots,q)$. When investigating upwardly compatible output-linear generalizers it is necessary to have at hand all such restrictions on $h$ in the situation where it has two (or more) of its arguments equal in value.

For output linear HERBIEs $y$ translation symmetry combined with (2.10) immediately gives $\sum_{i=0}^{n-1} T_i(h) = 1$. For all the input components of the learning set different from one another, requirement (2.3) immediately gives $h(q,x',x'',\ldots,q) = 1$, and in general $h_n(x,x',\ldots,q,\ldots,q) = \delta(n,p)$, where $p$ is the number of the slot having the same value ($q$) as the question. This is in complete agreement with the other restrictions on $h$ and the $h_n$.

It is important to understand that although $g\{i\}$ completely specifies $h$, and vice versa, all in the rather simple manner described above, $h$ does not have to satisfy the same restrictions that apply to the $g\{i\}$. For example, for $n > 2$, we can take $x = x' = q$, $y = y'$, and reproduction of the learning set forces $h$ to equal $1/2$. However, as was just shown, for $x = q = x' + \varepsilon$, $h$ must $= 1$, even if $\varepsilon$ is infinitesimally small. Since the point $x = q = x'$ is within the domain of definition even for generalizers satisfying requirement

(2.4), this means that $h$ cannot be a continuous function of its arguments if the associated $g\{i\}$ is to also reproduce the learning set and be output linear. Note however that this does not mean that the full $g\{i\}$ cannot both be a continuous function of its arguments throughout the whole domain delineated in (2.4), reproduce the learning set, and be output linear. As has been mentioned, metric-based HERBIEs are examples of generalizers which meet all of these requirements.

Given that such continuous output linear generalizers exist, since (as was shown before) all LMMGs, although output linear, are somewhere discontinuous, we can conclude that LMMGs form a proper subset of the set of all output linear generalizers. Even ignoring such questions of continuity, we can still see that not all output linear generalizers are LMMGs since metric based HERBIEs are full HERBIEs, even being invariant under rotation in the input space, whereas we know there are no LMMGs which can satisfy (2.6) in its entirety (see closing arguments in appendix G).

As an aside, it is interesting to note that the kinds of thresholding functions usually used to model neurons in neural nets satisfy all the restrictions on $h$ for $n = 2$, output-linear, one-dimensional HERBIEs. In proving this by enumerating all output linear HERBIEs (see appendix J), it is also shown how the restriction of output linearity is not enough to force unique generalization of HERBIEs. For purely heuristic reasons, it is therefore interesting to explore an additional criterion which does force unique generalization of output linear one-dimensional HERBIEs, at least for the $n = 2$ case. This criterion is to require that there exists an analytic function $f(x)$ going through the two points of the learning set such that if the two points of the learning set had lain anywhere along $f(x)$, then the output guessed for any question $q$ would be $f(q)$. This amounts to a sort of self-consistency restriction on the guessing[8] and is automatically satisfied by LMMGs. (As was mentioned previously, such generalizers usually are not HERBIEs, however.) More precisely, we require that for any learning set $((a, b), (a', b'))$, there exists a function $f(x)$ such that $f(a) = b, f(a') = b'$, and in general obeying $h(x, x', q)f(x) + (1 - h(x, x', q))f(x') = f(q)$, which is equivalent to

$$\textbf{(2.12)} \quad h(x, x', q) = \frac{f(q) - f(x')}{f(x) - f(x')}.$$

Note that an $f(x)$ obeying equation (2.12) is only fixed up to an overall scaling and/or translation (i.e., if $f(x)$ is a solution to (2.12), so is $af(x) + b$). It is shown in appendix K that for $n = 2$ any $f(x)$ obeying (2.12) is a straight line or an exponential, with the freedom in fixing the scaling and translation of $f(x)$ allowing it to be made to go through any two points making up a learning set. If we require that the generalizer be a full HERBIE, input space scaling invariance says that $h(ax, ax', aq)$ must equal $h(x, x', q)$ for all nonzero real numbers $a$, and as a result the exponential solution must be discarded.

[8]This restriction is somewhat similar to the restriction of strong self-guessing discussed in the first section of the second paper. The primary difference is that here we are not making any assumptions about upward compatibility.

(The fact that $f(x)$ must be a straight line shouldn't be too surprising, since we know that in addition to being expressible as it is in (2.12), as is discussed in appendix J $h$ must be expressible as a function of $(q - x')/(x - x')$, and it is hard to think of any way of doing this without having $f$ be linear in its argument.) Therefore, for any $n = 2$ learning set, there exists a unique function $f(x)$ which reproduces the learning set and which obeys (2.12), and therefore this restriction sets unique generalization of any $n = 2$ learning set. Note that this is all markedly similar to the situation encountered when enumerating all HERBIE LMMGs (see appendix G). In solving for all such LMMGs we are allowed both exponentials and polynomials as members of our basis set of functions, unless we require input-space scaling invariance, in which case the exponential solution must be discarded.

As it turns out, the extension of the restriction (2.12) to $n = 3$ can't be met by any generalizer. Ironically, this is because such an extension would necessitate upward compatibility between the $n = 2$ and the $n = 3$ cases. For $n = 3$, we write $f(q) = h(x, x', x'', q)f(x) + h(x', x, x'', q)f(x') + (1 - h(x, x', x'', q) - h(x', x, x'', q))f(x'')$. For $x = x'$, this becomes $2h(x, x, x'', q) = (f(q) - f(x'')/(f(x) - f(x''))$, and the exact same analysis as in the $n = 2$ case means that $f(x)$ must be linear. Therefore we have upward compatibility between $n = 2$ and $n = 3$. Now, however, let $x \neq x'$ and the three points of the learning set not be co-linear. By hypothesis, there is some smooth curve $f(x)$ going through those three points such that $g\{3\}(x, f(x), x', f(x'), x'', f(x''), q) = f(q)$ for all $x, x', x''$, and $q$. Now letting this $x'$ become infinitesimally close to $x$, continuity of $g\{3\}$ and upward compatibility mean that our $f(q)$ must be a straight line. However, we already assumed that $f(q)$ goes through three points which are not co-linear, resulting in a contradiction. Therefore we cannot extend the restriction embodied in equation (2.12) to the $n = 3$ case.

It is interesting to note that away from these points where datum spaces have identical input values, fitting a parabola to three points constitutes the $g\{3\}$ of an output linear continuous HERBIE obeying (the $n = 3$ extension of) equation (2.12). Indeed, fitting with a polynomial constitutes an output linear, upwardly compatible, (2.12) obeying continuous HERBIE of arbitrary dimension and order, so long as one stays away from regions where datum spaces have identical input values. (An interesting and as yet unsolved problem is to determine all solutions to this partial set of restrictions.) The Achilles' heel of polynomial fitting, of course, just all other LMMGs, is its inability to satisfy continuity in the situation where upward compatibility applies (i.e., when elements of the learning set approach one another). Note, however, that if we are willing to relax the requirement that generalizers be continuous, we can easily have fully proper output linear generalizers. A trivial example of such a generalizer is a HERBIE that guesses as the output to any given question the output component of the datum vector nearest to that question.

This ends the first paper of the series. The next paper continues the discussion of generalization, using generalization criteria of a qualitatively different nature than those presented here.

## Appendix A.

### Hopfield's scheme for using neural nets as associative memories

In Hopfield's scheme each neuron, having a value of $+1$ or $-1$, is connected to every other neuron. Evolution of the net proceeds according to the rule $\sigma_i(t) = g[\sum_j J_{ij}\sigma_j(t-1)]$, $\sigma_i(t)$ being the state of the $i$th neuron at cycle number $t$, and $J_{ij}$ constituting the connections between the neurons and usually taken to be symmetric. The function $g$ is usually taken to be the sgn function. In analogy to simple statistical mechanics we can now define a Hamiltonian $H$ as $-\sum_{ij} J_{ij}\sigma_i(t-1)\sigma_j(t-1)$. Let $\Delta\sigma_i$ represent the change in $\sigma_i$ over one cycle. If it is nonzero, then by the definition of the evolution operator $g$ the sign of $\Delta\sigma_i$ is the same as the sign of $\sum_j J_{ij}\sigma_j$ evaluated at the previous cycle. Therefore $\Delta\sigma_i \sum_j J_{ij}\sigma_j$ is positive definite, $\Delta H$ is negative definite under evolution of the system, and $H$ descends monotonically to a local minimum. To see how this can serve as an associative memory, view the state of all the neurons in the net as a (potential) memory. We then want to pick the $J_{ij}$ in such a way that if we start with the system in a state close to a memory we want stored in the system, then the evolution takes the neurons to that stored memory. This behavior can be achieved by using the Hebbian rule to set the $J_{ij}$ as $\sum_m x_i{}^m x_j{}^m$, where $x_i{}^m$ is the $i$th component of the $m$th memory to be "stored." With this $J_{ij}$, $H = -\sum_m (\vec{x}^m \cdot \vec{\sigma})^2$, $\vec{\sigma}$ being the state of all the neurons at a given time. For rather reasonable assumptions on the distribution of the $\vec{x}^m$ $H$ will be minimized for $\vec{\sigma}$ parallel to $\vec{x}^n$ for some $n$. In other words, the $\vec{x}^m$ serve as stable points of the evolution of the system, as desired.

## Appendix B.

### Boltzmann machines and backpropagation

To understand how neural nets generalizers behave, it is helpful to examine the details of their operation. As a first example, consider Boltzmann machines, which constitute a sort of stochastic generalization of Hopfield nets. Instead of viewing the state of the whole net as being the input and output, in Boltzmann machines certain neurons are declared to be input and certain other neurons are declared to be output. Input neurons have their state set by the outside world exclusively and output neurons have their state fed to the outside world exclusively. Therefore connectivities are, in a certain sense, explicitly nonsymmetric. Take the possible states of any given neuron to be $\{0, 1\}$. Evolution of the net proceeds according to the probabilistic rule that $\sigma_i(t) = 1$ with probability $f(\Delta E_i)$, $f$ being the fermi function and $\Delta E_i$, a function solely of the other spins in the net, being the difference between

the energy the net would have at cycle $t-1$ if neuron $i$ were fixed as a 0 and the energy it would have if neuron $i$ were fixed as a 1. (The Hopfield net's evolution is akin to the temperature = 0 case.) This, of course, is just conventional statistical mechanics. The probability of a particular state $\vec{\sigma}$ for the whole net is $[\prod_{i \ni \sigma_i=1} P(\sigma_i(t))][\prod_{j \ni \sigma_j=0}(1-P(\sigma_j(t)))]$, $P(\sigma_i(t))$ being the probability of neuron $i$ of the state $\vec{\sigma}$ being a 1 (i.e. $f(\Delta E_i)$). Taking a particular neuron $k$ which is off in state $\vec{\alpha}$ and turning it on to make state $\vec{\beta}$, we see that $P_{\vec{\alpha}}/P_{\vec{\beta}} = e^{-(E_\alpha - E_\beta)/kT}$, just as in Boltzmann statistical mechanics (hence the name "Boltzmann machine"). Therefore, as in a Hopfield net, letting a Boltzmann machine evolve will (likely) cause it to end up in a local minimum of its energy function. Simulated annealing [25] is often used in practice to try to "freeze in" that minimum by gradually lowering the temperature in the fermi function.

Again, just as in the Hopfield net, "teaching" the net in a Boltzmann machine consists of choosing the energy function to assure certain dynamic behavior in certain situations. Unlike the Hopfield net, however, here the behavior to be taught only entails fixing the values of a subset of the neurons, those which serve as the inputs and the outputs. Along with the fact that the input neurons do not evolve but remain constant throughout the running of the machine, this is what allows the Boltzmann machine to be viewed as a generalizer and respond with outputs completely different from any with which it was taught. As with Hopfield nets, a magnetic energy function is usually chosen: $E_{\vec{\alpha}} = -\sum_{i<j} J_{ij}\sigma_i^{\vec{\alpha}}\sigma_j^{\vec{\alpha}} + \sum_i \theta_i \sigma_i^{\vec{\alpha}}$. $J_{ij}$ is usually taken to be symmetric. For the Boltzmann machine the determination of the $J_{ij}$ from the learning set is more complicated than in the Hopfield case. Define a cross-entropy error function $G = \sum_{\vec{\alpha}} P_{\vec{\alpha}}^+ ln[P_{\vec{\alpha}}^+/P_{\vec{\alpha}}^-]$, where $\vec{\alpha}$ runs over the set of input and output states with a given input vector fixed on the input units, $P_{\vec{\alpha}}^+$ is the desired probability of state $\alpha$, and $P_{\vec{\alpha}}^-$ is the actual (equilibrium) probability of state $\alpha$ when the machine is running. By inspection, when $G$ is minimized (to 0), $P_{\vec{\alpha}}^+ = P_{\vec{\alpha}}^-$ and the system will definitely evolve to a state which has the desired output units when the input units are held fixed at the corresponding values. A number of techniques can be used to minimize $G$, gradient descent being perhaps the most common. When it is desired to have the learning set taught to the net consist of more than just one input vector/output vector pair, the $G$ to be minimized is modified by summing over all elements of the learning set.

Another neural net generalizer which is perhaps the simplest (and certainly the most widely researched) is the technique of backpropagation. Like the Boltzmann machine, this net has a subset of the neurons designated input neurons and another subset designated output neurons. Also like the Boltzmann machine, a fermi function (or, in general, any sigmoidal function) is used to do the evolving. However, here the neurons do not only take on discrete values and the evolving is not stochastic: $\sigma_i(t) = f(\sum_j J_{ij}\sigma_j(t-1))$ is the updating rule. Usually the $J_{ij}$ are chosen so that the connectivity is that of layers of neurons feeding forward into one another, starting from the input layer and proceeding all the way to the output layer. As in the previ-

ously described neural nets, output is when the system reaches a steady state, which in this case means after $n$ cycles, $n$ being the number of neuron layers. "Backpropagation" refers to the teaching technique of this scheme. The $J_{ij}$ are found by minimizing an energy (i.e., error) function via gradient descent. The energy function is usually chosen to be the sum of the squares of the differences between desired and actual output states for a given input state. Any $J_{ij}$ which constitute a zero of the energy function will give the desired output for any input chosen from the set of input/output pairs making up the provided learning set. Since the net is feedforward, it is a simple matter to write down the function relating the $J_{ij}$ and the output of the net, given any input. Therefore, although such a function is not directly invertible, it is trivially easy to run an iterative procedure to approximate the $J_{ij}$ which give zeros of the energy function by analytically calculating the functional form of the gradient of the function energy($J_{ij}$) and then using a steepest descent procedure. Again as with the Boltzmann machines, since a backpropagated net can respond in a novel way to a novel input, it constitutes a generalizer.

## Appendix C.

## A precise definition of intelligence

If desired, the definition of general intelligence in section 1.2 can be made more precise (and, accordingly, more narrow). An *intelligence task* can be defined in terms of the quintuple of a set of possible decisions, $\Delta$, a set of possible results, $\Omega$, one of which is delineated as being the *desired* result, a set of provided information, $I_{\text{prov}}$, a set of nonprovided information, $I_{\text{miss}}$, and a function $f$ taking $I_{\text{miss}}$, $I_{\text{prov}}$, and a single element $\delta \in \Delta$ to an element $\omega \in \Omega$. The intelligence task for the system is to pick the $\delta$ which, when fed to $f$ along with $I_{\text{prov}}$ and $I_{\text{miss}}$, produces the desired element of $\Omega$. The system's decision is required to be based only on $f$ and $I_{\text{prov}}$. The system is not provided with $I_{\text{miss}}$. (In the language of section 2 of the second paper, the mapping taking $I_{\text{miss}}$ and $I_{\text{prov}}$ to the (induced) projection of f taking $\Delta$ to $\Omega$ is a "method." This method is found from $f$, with $I_{\text{miss}}$ and $I_{\text{prov}}$ forming the defining set.) On the one extreme, $f$ could be relatively independent of the contents of $I_{\text{miss}}$, in which case the intelligence task essentially reduces to the problem of inverting $f$. On the other extreme, $f$ could be a fairly straightforward function, easy to invert, but highly dependent on the contents of $I_{\text{miss}}$. In this case, the intelligence task is essentially the problem of making a best guess for $I_{\text{miss}}$ based on $I_{\text{prov}}$ and $f$. Given the concept of an intelligence task, a system's general intelligence can be defined as a measure of how well the system performs at arbitrary intelligence tasks. Many different measures of that performance can be used, of course. Perhaps the simplest is to tally the percentage of intelligence tasks at which the system succeeds. Another possible measure requires a metric giving how close any element in the result set is to the desired result. Then a measure of a system's general intelligence could be the average distance of the system's guess from the desired result.

## Appendix D.

### Discussion of inner products among generalizers

Some interesting mathematics result from trying to define a vector space in which generalizers live and then defining a mapping meeting all of the criteria usually required of an inner product and which takes pairs of vectors from that vector space to elements of an appropriate field (see reference [50]). One way to go about this is to have the infinite set of numbers making up the "pseudo" inner product serve as the inner product in the vector space of generalizers. To do this, define the field $F$ to consist of all countably infinite sets of real numbers: if $a \in F$, then $a = \{a_1, a_2, \ldots\}$ where all $a_i \in \mathbf{R}$. For $a$ and $b$ both $\in F$, we say that $a = b$ iff $a_i = b_i$ for all $i$. Similarly, we say that $a >$ or $< b$ iff $a_i >$ or $< b_i$ for all $i$. (Note that with these definitions, it is no longer true that either $a > b$, $a = b$, or $a < b$ for all $a$ and $b$ — the field is only partially, not totally, ordered.) Addition in $F$ in performed component by component: $a + b = \{a_1 + b_1, a_2 + b_2, \ldots\}$. The additive identity is simply $\{0, 0, \ldots\}$. $F$ differs from the standard $L^2$ Hilbert space in that, being a field, multiplication of elements in $F$ is defined. As with addition, elements in $F$ multiply by component: $a \times b = \{a_1 b_1, a_2 b_2, \ldots\}$. The multiplicative identity therefore is $\{1, 1, \ldots\}$. (Note that if $F$ were to be viewed as an infinite vector space over $\mathbf{R}$, this definition of multiplication would not be a proper tensor operation, since after appropriate rotations $a \times b$ could always be made to be either parallel to $a$ or parallel to $b$.) Closure, the existence of multiplicative inverses (for elements of $F$ with no components equal to 0), commutativity, and the distributive law all follow, confirming that $F$ is indeed a field. Intuitively, $F$ is simply an infinite set of $\mathbf{R}$ fields all acting in parallel — there is no interaction between components of an element of $F$.

Now that we have an appropriate field, we can define the vector space in which generalizers live. Simply put, generalizers live in the same space as the elements of $F$, except that their components are functions from $\mathbf{R}^n$ to $\mathbf{R}$, rather than just elements of $\mathbf{R}$. This space is a vector space over $F$, just as the set of all well-behaved (i.e., $L^p$) functions from $\mathbf{R}^n$ to $\mathbf{R}$ is a vector space over $\mathbf{R}$. $a \in F$ multiplied by the generalizer $\{g\{1\}, g\{2\}, \ldots\} = \{a_1 g\{1\}, a_2 g\{2\}, \ldots\}$. Addition of generalizers goes exactly as addition of elements of $F$. All the usual restrictions on legal vector spaces follow from these definitions (see reference [50] again).

Using Dirac notation, we can now define the inner product between two generalizers $|G>$ and $|H>$ to be the element of $F$ $\{\langle g\{1\}|h\{1\}\rangle, \langle g\{2\}|h\{2\}\rangle, \ldots\}$. Except for the fact that the inner product as defined this way does not live in $\mathbf{R}$, it obeys all the usual restrictions on inner products. As a result, both the triangle inequality and Schwartz's inequality hold. An orthogonal basis for the space of generalizers consists of the generalizers $\{B_{i1}, 0, 0, \ldots\}$ for all $i$, $\{0, B_{i2}, 0, \ldots\}$ for all $i$, etc., where "0" denotes the 0 function, and $B_{ij}$ denotes a complete orthonormal set of basis functions for the mappings from $\mathbf{R}^{j(m+1)+m}$ to $\mathbf{R}$. The elements of this basis will be generically denoted as $|B_{ij}>$. If the $i$ are discrete, we can expand any generalizer $|H>$ as

$|H>=\sum_{ij}|B_{ij}><\langle B_{ij}|H\rangle$. A similar expression holds for continuous $i$. Although easy to work with, it should be noted that these $|B_{ij}>$ are not normalized. Nor can they be normalized simply by multiplying them by the appropriate constant, since the multiplicative identity is $\{1,1,\ldots\}$, and therefore the 0 components in the $|B_{ij}>$ make their multiplicative inverses undefined. To construct an orthonormal basis, we would have to take the (nonorthogonal) basis $\{B_{i1}, B_{i2}, \ldots\}$ and apply Graham–Schmidt.

As opposed to this formalism using an infinite number of components, it is possible to view generalizers as a vector space over the reals rather than over $F$. $|G>+|H>$ still $=\{g\{1\}+h\{1\}, g\{2\}+h\{2\}, \ldots\}$, but now $a|G>=\{ag\{1\}, ag\{2\}, \ldots\}$. The difficulty with this new formalism is in trying to define an inner product mapping two generalizers to an element of the reals in such a way that the magnitude of a generalizer $|G>$ is finite even if all of its component functions $g\{i\}$ have nonzero magnitude. One way to do this is to define

$$\langle G|H\rangle = \lim_{n\to\infty} \frac{\sum_{i=1}^{n}\langle g(i)|h(i)\rangle}{n}.$$

The usual required properties of inner products follow, assuming that the limit defining the inner product exists for the generalizers in question. Unfortunately, the magnitude of the inner product of any one of the basis vectors $|B_{ij}>$ with another generalizer is 0. We can no longer write $|H>=\sum_{ij}|B_{ij}><\langle B_{ij}|H\rangle$. Graham–Schmidt (for example run over the set of elements from $\{B_{i1}, B_{i2}, \ldots\}$ which have well-defined inner products with one another) has to be used just to form a basis allowing us to write this kind of expansion, regardless of any questions of overall normalization.

## Appendix E.

### Basic properties of LMMGs

In this appendix a rigorous definition of LMMGs is given and a property of them which is used in appendices G and F is proven.

Make the rigorous definition that an LMMG is a generalizer which works by taking an ordered basis set of infinitely differentiable nonzero functions, $\phi_1(\vec{r}), \phi_2(\vec{r}), \ldots \in \Phi$, and for the given learning set $\theta$ forms the linear combination of basis functions, $\sum_{i=1}^{N}\alpha_i\phi_i(\vec{r})$, $\alpha_N \neq 0$, which has as small $N$ as possible and yet still goes through every element of the $\theta$. The guessed output of the generalizer for a question $\vec{q}$ is given by $\sum_{i=1}^{N}\alpha_i\phi_i(\vec{q})$. Any generalizer like "fit the learning set with a power series" or "fit the learning set with a sum of Bessel functions" or the like is an LMMG.

First note that $\alpha_N$ must be unique. To see this, first note that if $\sum_{i=1}^{N}\alpha_i\phi_i(\vec{r})$ reproduces a learning set of pairs $\{(\vec{r}_j, y_j)\}$, then the matrix equation $\phi_j^i\alpha_i = y_j$ must hold, where implied sum notation is being used and where the matrix $\phi_j^i$ is defined to have as elements the values $\phi_i(\vec{r}_j)$. Now if the set of $\{\alpha_i\}$ which reproduce the learning set is not unique, we have that $\phi_j^i(\alpha_i - \alpha_i') = 0$ where at least one of the $\alpha_i$ differs from the corresponding $\alpha_i'$.

Therefore, indexed by $i$, the vectors $\phi_j^i$ are linearly dependent. This means that there exists some value $\gamma$ such that $\phi_j^\gamma$ can be written as $\beta_i \phi_j^i$ where $\beta_\gamma = 0$. Therefore we can write

$$\phi_j^i \alpha_i = \phi_j^{i \neq \gamma} \alpha_{i \neq \gamma} + \beta_{i \neq \gamma} \phi_j^{i \neq \gamma} \alpha_\gamma = \phi_j^{i \neq \gamma} (\alpha_{i \neq \gamma} + \alpha_\gamma \beta_{i \neq \gamma}).$$

If in particular $\alpha_N \neq \alpha'_N$, we can take $\gamma = N$, and therefore have just derived a formula for a linear combination of the first $N-1$ $\phi_i(\vec{r})$ which reproduces the learning set. Since by hypothesis $N$ is as small as possible, this is impossible, so $\alpha_N$ indeed must equal $\alpha'_N$. If the remaining coefficients are not unique, then we can choose between any two candidate sets $\{\alpha_i\}$ and $\{\alpha' i\}$ according to any arbitrary method, so long as we are consistent in applying it. Here we will require that we choose the set which has the lowest absolute value for the highest $i$ in which they differ. For example, choose $\{6, 6, -3, 7, 2\}$ over $\{8, 9, 4, 7, 2\}$, since $|-3| < |4|$. (Rigorously speaking, it is only once we have set such a rule for choosing between two candidate sets that we have completely defined the LMMG.)

**(E.1)** For all $N$ and $\{\alpha_i\}$ where $\alpha_N \neq 0$ there exists an $N$-point learning set such that the LMMG chooses those $\{\alpha_i\}$ to fit the learning set.

**Proof.** Since the $\phi_i(\vec{r})$ are linearly independent, for any $N$ of the $\phi_i(\vec{r})$ there does not exist a nonzero $N$-component vector $\vec{\beta}$ such that

$$\vec{\beta} \cdot [\phi_1(\vec{r}), \phi_2(\vec{r}), \ldots, \phi_N(\vec{r})] = 0$$

for all vectors $[\phi_1(\vec{r}), \phi_2(\vec{r}), \ldots, \phi_N(\vec{r})]$ (i.e., for all $\vec{r}$). In other words, the vectors $[\phi_1(\vec{r}), \phi_2(\vec{r}), \ldots, \phi_N(\vec{r})]$ span $\mathbf{R}^N$. Therefore, there exist $N$ vectors $[\phi_1(\vec{r}), \phi_2(\vec{r}), \ldots, \phi_N(\vec{r})]$ which are linearly independent, i.e. for any $N$ there exists a set of $N$ input values $\{\vec{r}_j | 1 \leq j \leq N\}$ such that the $N$ vectors

$$\{\phi_i(\vec{r}_1) | 1 \leq i \leq N\}, \{\phi_i(\vec{r}_2) | 1 \leq i \leq N\}, \ldots, \{\phi_i(\vec{r}_N) | 1 \leq i \leq N\}$$

are linearly independent. Now for any $N$ and any associated set $\{\alpha_i\}$ the $N$-element learning set

$$\{(\vec{r}_1, \sum_{i=1}^N \alpha_i \phi_i(\vec{r}_1)), \ldots, (\vec{r}_N, \sum_{i=1}^N \alpha_i \phi_i(\vec{r}_N))$$

is such that it lies on the surface $\sum_{i=1}^N \alpha_i \phi(\vec{r})$. If $\alpha_N \neq 0$, then it is possible that the LMMG will fit the learning set using those $\{\alpha_i\}$ and therefore this surface $\sum_{i=1}^N \alpha_i \phi(\vec{r})$. To ensure that this surface is the actual one which the LMMG will use to fit the learning set, it suffices to note that since all the rows of our matrix $\phi_j^i$ are linearly independent, there does not exist any set of values $\{\alpha'_i\}$ different from the set $\{\alpha_i\}$ such that the learning set can be reproduced by the surface $\sum_{i=1}^N \alpha'_i \phi_i(\vec{r})$. This completes the proof of the lemma. ∎

## Appendix F.

### Proof that no LMMG is continuous

See appendix E for a rigorous definition of LMMGs and a proof of the fact that for any $N$ and any set $\{\alpha_i\}$ where $\alpha_N \neq 0$, we can always build an $N$-element learning set $\theta$ such that the LMMG fits it with a surface of the form $\sum_{i=1}^{N} \alpha_i \phi_i(\vec{r})$, where the $\phi_i(\vec{r})$ are taken from the basis set of functions of the LMMG. Build such a learning set, and indicate its elements by $(\vec{r}_j, y_j)$, where $j$ ranges from 1 to $N$. Indicate the surface the LMMG fits to $\theta$ by $f(\vec{r})$.

To proceed in the proof, it is first necessary to establish the following lemma:

**(F.1)** Let $\phi_j^i$ indicate $N-1$ linearly independent vectors of dimension $N$, $1 \leq i \leq N-1$, $1 \leq j \leq N$. Then there exist two indices $a$ and $b$ such that if $\phi_j^i$ is modified by having all $N-1$ elements $\phi_b^i$ set equal to the corresponding $N-1$ elements $\phi_a^i$, then the new set of vectors $\phi_j^i$ is still linearly independent.

To prove (F.1) make an $N \times N$ matrix $\Phi$ whose first $N-1$ rows are the vectors $\phi_j^i$ and whose last row is a vector linearly independent with all $N-1$ of the $\phi_j^i$. We know that $\det(\Phi)$ can not equal 0. However, we can evaluate that determinant by cofactors of the entries in the $N$th row. At least one of these cofactors cannot equal 0, since the full determinant does not equal 0. This means that there is a column index $c$ such that if we make an $(N-1)\times(N-1)$ matrix $\Phi'$ from the first $N-1$ rows of $\Phi$ by removing the $c$th column from those rows, then $\det(\Phi') \neq 0$. This means that the vectors making up the rows of $\Phi'$ are linearly independent. Now make a new matrix of $N-1$ rows and $N$ columns by taking $\Phi'$, duplicating one of its columns, and appending this column to the right-hand side of $\Phi'$. The vectors formed by the rows of this new matrix are still linearly independent. However, this matrix is exactly the one made by modifying $\phi_j^i$, as referred to in (F.1). ∎

Take our original learning set $\theta$ and modify it continuously by sliding point $a$ of the learning set (i.e., the point $(\vec{r}_{j=a}, y_{j=a})$) toward point $b$ (i.e., toward the point $(\vec{r}_{j=b}, y_{j=b})$) along the surface $f(\vec{r})$. At any point during this sliding the LMMG can fit the (modified) learning set with the first $N$ $\phi_i(\vec{r}_j)$, just by fitting it with $f(\vec{r})$. Therefore we know that at any point in the sliding the LMMG will fit the learning set with the first $m \leq N$ of the $\phi_i(\vec{r}_j)$.

Now by (F.1), we can choose $a$ and $b$ so that when the two points have been slid on top of one another, the $N-1$ vectors $\phi_i(\vec{r}_j)$ formed by taking the first $N-1$ $\phi_i$'s of the LMMG and evaluating them at the $N$ points of the learning set are still linearly independent. However, we know that when the two points are on top of one another the first $N$ vectors $\phi_i(\vec{r}_j)$ must be linearly dependent, since now the determinant of $\phi_i(\vec{r}_j) = 0$. The only way this is possible is if the vector $\phi_N(\vec{r}_j)$ is expressible as a linear

combination of the first $N-1$ $\phi_i(\vec{r}_j)$. Since we know that the LMMG *can* fit the modified learning set with (at most) $N$ of the first $\phi_i(\vec{r}_j)$, this linear dependence tells us that the LMMG will in fact fit the modified learning set with only $m \leq (N-1)$ of the first $\phi_i(\vec{r}_j)$, when point $b$ has been moved on top of point $a$.

So we know that there must be a point during this sliding at which the LMMG suddenly stops fitting the learning set with the first $N$ of the $\phi_i(\vec{r}_j)$ and instead fits it with the first $N-1$ of them. However, all of the $\phi_i(\vec{r})$ are linearly independent of one another. This means that any linear combination of the first $N-1$ of the $\phi_i(\vec{r}_j)$ must differ from all linear combinations of the first $N$ of the $\phi_i(\vec{r}_j)$ by a nonzero amount for some value of $\vec{r}$. This means that the LMMG's $g\{N\}$ must be a discontinuous function of its arguments when the arguments are at this point of transition from $N$ function fitting to $N-1$ function fitting. This concludes the proof of the proposition.

## Appendix G.

### Proof that only polynomial LMMGs obey (2.6)

See appendix E for a precise definition of LMMGs. To begin our proof that only polynomial LMMGs obey (2.6), it is easy to show that output translation invariance forces $\phi_1(\vec{r})$ to $= 1$. Let $D$ indicate the support of $\phi_1(\vec{r})$. By definition of LMMGs, $D$ is nonempty. Now consider any point $\vec{r} \in D$. Let our learning set be $\{\vec{r}, \phi_1(\vec{r}) \neq 0\}$. Output-translating our learning set by $\lambda$ we get the new learning set $\{\vec{r}, \phi_1(\vec{r}) + \lambda\}$. By output-translation invariance, $g\{1\}(\vec{r}, \phi_1(\vec{r}), \vec{q}) = g\{1\}(\vec{r}, \phi_1(\vec{r}) + \lambda, \vec{q}) - \lambda$ for all $\vec{q}$. However, since $g\{1\}$ is part of an LMMG, $g\{1\}(\vec{r}, \phi_1(\vec{r}), \vec{q}) = \phi_1(\vec{q})$. Moreover, output space scaling invariance requires that

$$\begin{aligned} g\{1\}(\vec{r}, \phi_1(\vec{r}) + \lambda, \vec{q}) &= \phi_1(\vec{q})[\frac{\phi_1(\vec{r}) + \lambda}{\phi_1(\vec{r})}], \text{ so} \\ \phi_1(\vec{q}) &= \phi_1(\vec{q})[\frac{\phi_1(\vec{r}) + \lambda}{\phi_1(\vec{r})}] - \lambda; \\ \phi_1(\vec{q}) &= \phi_1(\vec{r}) \end{aligned}$$

In other words, $\phi_1(\vec{q})$ is a nonzero constant, $k$, independent of $\vec{q}$, and without loss of generality we can set $k = 1$. ∎

Now let $T_\lambda$ indicate any one of the (parameterized) transformations of requirement (2.6). For example, we indicate the operation of output space translation invariance by the (nonlinear) operator $T_\lambda$ where $T_\lambda[f(\vec{r})] = \lambda + f(\vec{r})$. Then it follows that

**(G.1)** For a given basis set of functions $\phi_i(\vec{r}) \in \Phi$, if the LMMG based on $\Phi$ obeys invariance under $T_\lambda$ then for all sets of real numbers $\{\alpha_i\}$ and all $N$ (where $\alpha_N \neq 0$) there exists a set $\{\alpha'_i\}$ and an $N'$ (where $\alpha'_{N'} \neq 0$) such that $T_\lambda[\sum_{i=1}^{N} \alpha_i \phi_i(\vec{r})] = \sum_{i=1}^{N'} \alpha'_i \phi_i(\vec{r})$.

To see (G.1) first recall from appendix E that since the $\phi_i(\vec{r})$ are linearly independent of one another, for all $N$ and $\{\alpha_i\}$ there exists an $N$-point learning set such that the LMMG chooses those $\{\alpha_i\}$ to fit the learning set. Next note that by the hypothesis of meeting (2.6), the curve $T_\lambda[\sum_{i=1}^{N} \alpha_i \phi_i(\vec{r})]$ reproduces the transformation of this learning set and also makes the transformed question-guess mapping. It is unique in doing this, and therefore invariance of the LMMG under $T_\lambda$ means that it must be the same as the curve generated by running the LMMG off of the transformed learning set. This LMMG-generated curve can always be written as $\sum_{i=1}^{N'} \alpha'_i \phi_i(\vec{r})$, however, which proves the supposition.

Note that $\phi_1(\vec{r}) = 1$ is completely in accord with (G.1). Also note that if a learning set $\{\vec{r_j}, y_j\}$ is fit by an LMMG with the curve $\sum_{i=1}^{N} \alpha_i \phi_i(\vec{r})$, then the learning set $\{\vec{r_j}, \lambda y_j\}$ will be fit with $\sum_{i=1}^{N} \lambda \alpha_i \phi_i(\vec{r}) = \lambda \sum_{i=1}^{N} \alpha_i \phi_i(\vec{r})$. Therefore LMMGs automatically obey output scaling invariance.

The proof that there is only one set $\Phi$ whose LMMG obey the invariances of (2.6) and that that set is the set of all polynomials works by showing that there is only one set $\Phi$ which meet (G.1) for input space invariances. Note, however, that (G.1) is only a necessary condition for the invariance of an LMMG under $T_\lambda$. Fortunately, in what follows the sufficient condition (namely, that the given basis set $\Phi$, which obeys $T_\lambda[\sum_{i=1}^{N} \alpha_i \phi_i(\vec{r})] = \sum_{i=1}^{N'} \alpha'_i \phi_i(\vec{r})$ for all transformations $T_\lambda$, is such that the associated LMMG obeys invariance under all the $T_\lambda$) will be obvious. Therefore, in the exposition below sufficiency will not be explicitly illustrated.

Without loss of generality we can take $N' \leq N$ in (G.1) when we are dealing with a single $\lambda$ – if $N' > N$, then $T_\lambda^{-1}[\sum_{i=1}^{N'} \alpha'_i \phi_i(\vec{r})] = \sum_{i=1}^{N} \alpha_i \phi_i(\vec{r})$, and since our (parameterized) transformations are groups, $T_\lambda^{-1} = T_{\lambda'}$ for some $\lambda'$ (in practice we usually parameterize the group so that $\lambda' = -\lambda$ so that our groups are explicitly Lie groups). Therefore by simply flipping which we call the transformed space and which we call the untransformed space we can have $N' \leq N$.

In the analysis below, however, we are going to need to make use of an even stronger condition on the $N'$. Instead of just choosing between $T_\lambda$ and $T_{-\lambda}$ to ensure $N' \leq N$ for a single $\lambda$, we will require $N' \leq N$ for all $\lambda \in [-\Delta\lambda, \Delta\lambda]$, with $\Delta\lambda > 0$. In other words, as we move through $[-\Delta\lambda, \Delta\lambda]$ our choice of $T_\lambda$ or $T_{-\lambda}$ (to ensure $N' \leq N$) must not change.

This stronger condition is not difficult to meet, fortunately. Since the $\phi_i(\vec{r})$ are all linearly independent, as before there exists a learning set $\theta$ with input coordinates $\vec{r_1}, \vec{r_2}, \ldots, \vec{r_N}$ such that the LMMG working from this $\theta$ will create a curve $\sum_{i=1}^{N} \alpha_i \phi_i(\vec{r})$ where (in particular) $\alpha_N$ is nonzero. Now assume there exits an open ball of nonzero radius living in $\mathbb{R}^{mN}$ (the vector space delineating all datum space inputs components for the $m$-dimensional LMMG's $g\{N\}$) and centered on the input components of $\theta$, such that the matrix $\phi_i(\vec{r_j})(1 \leq i, j \leq N)$ has nonzero determinant for all sets $\{\vec{r_j}\}$ contained within that ball. Then any $T_\lambda$ moving $\theta$ to any point within that ball will have $N' \leq N$. ($\alpha'_N$ might $= 0$, for example, which is why $N'$ is allowed to be $< N$. The important point is that there is no linear dependence

among the transformed $\phi_i(\vec{r}_j)$ which might force $N'$ to be $> N$.) Therefore our "stronger condition" will be met if this assumption holds. To prove this assumption it is sufficient to disprove its negation. To do this, note that if there does not exist any such ball of nonzero $\det(\phi_i(\vec{r}_j))$, then since we know that $\det(\phi_i(\vec{r}_j))$ is nonzero at $\theta$, there must exist an axis in $\mathbb{R}^{mN}$ such that the set of points along that axis with 0 $\det(\phi_i(\vec{r}_j))$ contains the intervals $[x, \theta)$ and $(\theta, y]$, where $x$ is $\theta$ shifted a nonzero amount in one direction and $y$ is $\theta$ shifted a nonzero amount in the opposite direction. (At first it might be thought that the statement that no such open ball of nonzero radius encloses $\theta$ could be satisfied if, for example, the set of points with 0 $\det(\phi_i(\vec{r}_j))$ in the vicinity of $\theta$ were $[x, \theta)$ and either $(z, y]$ or $[z, y]$, where $z$ is $\theta$ shifted by a nonzero amount in the $y$ direction. Note, however, that if this were the case then any learning set lying between $\theta$ and $z$ would lie wholly within a ball of nonzero $\det(\phi_i(\vec{r}_j))$. (I am assuming here that it is only this one axis in $\mathbb{R}^{mn}$ which has the property that $\det(\phi_i(\vec{r}_j))$ flips from being 0 to nonzero as you move along it at the point $\theta$. The argument if there is more than one such axis goes similarly.) So for our purposes of finding any such ball we can safely remove all such situations from consideration.) Now note however that $\det(\phi_i(\vec{r}_j))$ is a continuous function of the $\vec{r}_j$. Therefore it is impossible for it to be 0 in $[x, \theta)$ and $(\theta, y]$ but nonzero at $\theta$. Therefore our assumption of the existence of such a ball of nonzero $\det(\phi_i(\vec{r}_j))$ is justified, and our "stronger condition" can be satisfied. In what follows we will always take our learning set to be in the center of such a ball of nonzero $\det(\phi_i(\vec{r}_j))$, and all $T_\lambda$ will keep us within that ball. (Note that we make no restrictions on the value of $q$, however.) Therefore we will always be able to take $N' \leq N$.

Now examine the invariance of scaling and translation in the input space. Rearranging (G.1), and assuming our learning set is in the center of a ball of nonzero $\det(\phi_i(\vec{r}_j))$, we immediately get

**(G.2)** For a given basis set of functions $\Phi$, if the LMMG based on $\Phi$ obeys (2.6) then there exists a function $K(a, \vec{b}, n, i)$ such that $\phi_n(a\vec{r} + \vec{b}) = \sum_{i=1}^{n} K(a, \vec{b}, n, i)\phi_i(\vec{r})$ for all $\vec{r}$. $a$ and $\vec{b}$ are assumed close enough to 1 and $\vec{0}$ respectively to ensure that we are within the ball mentioned in the previous paragraph.

Note that although the transformation of the learning sets is restricted to being within the ball, the restrictions on the $\phi_i(\vec{r})$ given by (G.2) apply for all $\vec{r}$.

$K(a, \vec{b}, n, i)$ is a differentiable function of $a$ and $\vec{b}$ everywhere that (G.2) holds. For example, to see differentiability of $K$ with respect to the components of $\vec{b}$, re-express (G.2) in an abbreviated format (for the particular case where $a$ is fixed and close to 1) as $\phi_n(\vec{r} + \vec{b}) = K^i(\vec{b})\phi_i(\vec{r})$, where implied sum notation is being used. Then

$$\frac{\partial \phi_n(\vec{r}^* + \vec{b})}{\partial b_m}|_{\vec{b}=\vec{0}} = \phi_i(\vec{r}^*)\frac{\partial K^i(\vec{b})}{\partial b_m}|_{\vec{b}=\vec{0}}$$

for any vector $\vec{r}^*$. In particular, since the $\phi_i(\vec{r})$ are all linearly independent, as before we can choose $N\vec{r}^*$'s so that $\det(\phi_i(\vec{r}^*)) \neq 0$. This means that there exists a matrix $A_k^l$ such that $A_k^l \phi_i(\vec{r}^*) = \delta_i^l$. Therefore

$$\frac{\partial K^l(\vec{b})}{\partial b_m}|_{\vec{b}=\vec{0}} = \frac{\partial [A_k^l \phi_n(\vec{r}^* + \vec{b})]}{\partial b_m}|_{\vec{b}=\vec{0}}.$$

Since the $\phi_i(\vec{r})$ are all differentiable functions, the right-hand side of this vector equation is well defined. This means that for any $l$, $K(a, \vec{b}, n, l)$ is a differentiable function of all the components of $\vec{b}$, assuming $a$ and $\vec{b}$ meet the conditions given in (G.2). A similar proof holds for showing that $K(a, \vec{b}, n, i)$ is a differentiable function of $a$.

This differentiability of $K$ is what allows us to delineate the set of all $\Phi$ whose associated LMMG obeys (2.6). To see this first examine the case where the input space is one-dimensional. $\phi_2(x + b) = K(1, b, 2, 2)\phi_2(x) + K(1, b, 2, 1)$. Now since the $\phi_i(x)$ are linearly independent, $K(1, 0, m, n) = \delta(m, n)$, the kronecker delta of $m$ and $n$. This allows us to subtract terms like $K(1, 0, 2, 1)$ at will, so we can write

$$\begin{aligned} \frac{d\phi_2(x)}{dx} &= \lim_{b \to 0}[\frac{\phi_2(x + b) - \phi_2(x)}{b}] \\ &= \phi_2(x)\frac{dK(1, b, 2, 2)}{db}|_{b=0} + \frac{dK(1, b, 2, 1)}{db}|_{b=0}. \end{aligned}$$

The two derivatives with respect to $b$ in this equation are arbitrary constants. If the first of them $= 0$, we have the solution $\phi_2(x) = Ax + B$ ($A$ and $B$ being arbitrary constants). Otherwise $\phi_2(x) = Ae^{Bx} + C$ ($A$, $B$, and $C$ being arbitrary constants). This second solution can be discarded, however, since scaling invariance would require that for any real number $a$, $\phi_2(ax) = Ae^{Bax} + C$ be expressible as a linear combination of $Ae^{Bx} + C$ and 1, which is impossible. Therefore any one-dimensional LMMG obeying (2.6) taking a two-element learning set fits that learning set with a straight line. This result is in accord with (2.7).

These kinds of arguments can be iterated to get all the other $\phi_i(x)$:

$$\frac{d\phi_3(x)}{dx} = a\phi_3(x) + bx + c.$$

$a = 0$ gives $\phi_3(x) = Ax^2 + Bx + C$. $a \neq 0$ has the solution $\phi_3(x) = e^{ax}[\int e^{-ax}(bx + c)dx + D]$. This solution cannot $\in \Phi$ if $D \neq 0$, due to the requirement of input space scaling invariance. However, the remaining terms in this solution are linearly dependent on $\phi_2(x)$ and $\phi_1(x)$, and therefore must be discarded. Therefore $\phi_3(x) = Ax^2 + Bx + C$. Similarly, via induction $\phi_n(x) = \sum_{i=0}^n a_i x^i$, or, without loss of generality, $\phi_n(x) = x^n$. Note that if we were not requiring scaling invariance we would also have as legal $\Phi$'s those whose individual member functions are polynomials + exponentials.

Now let us consider the case where the input space dimension $m$ is $> 1$. $\phi_1(\vec{r})$ still (can be taken to) $= 1$. Now, however, we get p.d.e.'s rather than

o.d.e.'s for the other $\phi_i(\vec{r})$. For example, we get $\frac{\partial \phi_2(\vec{r})}{\partial \vec{r}_i} = A_i \phi_2(\vec{r}) + B_i$, where $i$ ranges over the coordinates of the input space. If, for $i = \alpha$, $A_i = 0$, then

$$\phi_2(\vec{r}) = B_\alpha r_\alpha + C(r_1, r_2 \ldots, r_{\alpha-1}, r_{\alpha+1}, \ldots)$$

(note there is no implied sum over the repeated $\alpha$'s in this equation). If all the $A_i = 0$, $\phi_2(\vec{r}) = B^i r_i + C$ (implied sum over the repeated $i$). Let $A_i \neq 0$ for $i = \alpha$. Then

$$\phi_2(\vec{r}) = D(r_1, r_2, \ldots, r_{\alpha-1}, r_{\alpha+1}, \ldots) e^{[A_\alpha r_\alpha]} + B_\alpha,$$

and as before we cannot meet scaling invariance. So all the $A_i$ must $= 0$, and $\phi_2(\vec{r}) = \vec{B} \cdot \vec{r} + C$.

Next up we get

$$\frac{\partial \phi_3(\vec{r})}{\partial r_i} = E_i \phi_3(\vec{r}) + F_i \vec{B} \cdot \vec{r} + G_i,$$

where this $\vec{B}$ is the same as the one in the solution for $\phi_2(\vec{r})$. Without loss of generality we can set all $E_i$ to 0. This is because if $E_\alpha$ did not equal 0 for a particular $\alpha$, then

$$\begin{aligned} \phi_3(\vec{r}) &= e^{[E_\alpha r_\alpha]} [H(r_1, r_2, \ldots, r_{\alpha-1}, r_{\alpha+1}, \ldots) \\ &\quad + \int e^{[-E_\alpha r_\alpha]} (F_\alpha \vec{B} \cdot \vec{r} + G_\alpha) dr_\alpha]. \end{aligned}$$

Scaling invariance forces $H(r_1, r_2, \ldots, r_{\alpha-1}, r_{\alpha+1}, \ldots)$ to $= 0$, as usual, so we would have

$$\phi_3(\vec{r}) = (\frac{G_\alpha}{-E_\alpha}) + F_\alpha (\frac{\vec{B} \cdot \vec{r}}{-E_\alpha}) - F_\alpha (\frac{B_\alpha}{E_\alpha^2}),$$

which is linearly dependent on $\phi_1(\vec{r})$ and $\phi_2(\vec{r})$, contrary to the requirement that $\Phi$ be a basis.

If in addition to the $E_i$ all the $F_i = 0$, then we get $\phi_3(\vec{r}) = \vec{G} \cdot \vec{r} + H$. Since $\phi_3(\vec{r})$ is linearly independent of $\phi_2(\vec{r})$ and $\phi_1(\vec{r})$, $\vec{G}$ and $\vec{B}$ cannot be parallel. We can similarly set all but the last constant to 0 in the next $m - 2$ sets of p.d.e.'s (for $\phi_4(\vec{r})$ through $\phi_{m+1}(\vec{r})$), and then without loss of generality choose $\vec{B}$, $\vec{G}$ and all the other constants so that $\phi_2(\vec{r}) = r_1$, $\phi_3(\vec{r}) = r_2, \ldots,$ all the way up to $\phi_{m+1}(\vec{r}) = r_m$.

For the next set of p.d.e.'s (those concerning $\phi_{m+2}(\vec{r})$) we cannot continue this procedure of setting all but the last constant to 0, since $\phi_{m+2}(\vec{r})$ must be linearly independent of $\phi_1(\vec{r})$ through $\phi_{m+1}(\vec{r})$. Therefore

$$\frac{\partial \phi_{m+2}(\vec{r})}{\partial r_i} = \vec{I}_i \cdot \vec{r} + J_i$$

where for all $i$, $|\vec{I}_i| \neq 0$. This results in $\phi_{m+2}$ being a quadratic polynomial of the components of $\vec{r}$.

*A priori*, we are not justified in setting all those constants to 0 in the p.d.e.'s for $\phi_3(\vec{r})$ through $\phi_{m+1}(\vec{r})$. It is entirely possible, for example, that $\phi_3(\vec{r})$ could have one or more of its $F_i \neq 0$ (which would result in $\phi_3(\vec{r})$ being a quadratic polynomial) and yet all but the last constant in the p.d.e.'s for $\phi_4(\vec{r})$ could be 0, so that $\phi_4(\vec{r})$ would be one of the remaining linear functions. In other words, the ordering of the elements of $\Phi$ is not necessarily according to the power of the individual polynomials. It is only in the one-dimensional case where the requirements of scaling and translation invariance force us to order the basis set of polynomials according to the power of the polynomials. Indeed, for the multidimensional case it is even possible that a given linear function appears nowhere in $\Phi$, but can be constructed by forming a linear combination of polynomials from $\Phi$ each of which has power $> 1$.

However, note that there is an important invariance from (2.6) which we have not yet taken into account: invariance of the mapping under rotation of the input space. Indeed, no basis set $\Phi$ for an input space having dimension $> 1$ can obey this invariance. This is because as was shown above, if we have a two-element learning set, in general it gets fitted with a linear combination of $\phi_1(\vec{r}) = 1$ and $\phi_2(\vec{r}) = \vec{B} \cdot \vec{r}$, where $\vec{B}$ is an arbitrary constant. For any such fixed $\vec{B}$ though, it is impossible to express the rotation of the surface fitted to the two-element learning set with a new linear combination of $\phi_1(\vec{r}) = 1$ and $\phi_2(\vec{r}) = \vec{B} \cdot \vec{r}$. Rotation invariance is violated by LMMGs. However, note that, for example, if the $\phi_3(\vec{r})$ through $\phi_{m+1}(\vec{r})$ are indeed the linear functions, then for learning sets of cardinality $m + 1$, we do get rotation invariance, since such a learning set will be fit with a hyperplane $\alpha_1 + \sum_{i=2}^{m+1} \alpha_i r_i$, and the rotation of such a hyperplane is just another hyperplane, $\alpha_1' + \sum_{i=2}^{m+1} \alpha_i' r_i$. We can therefore make a watered-down version of the requirement of rotation invariance: if the $\gamma(i)$ are the successive values of $N$ such that the rotation of any linear combination of the first $N$ elements of $\Phi$ is expressible as another linear combination of those first $N$ elements of $\Phi$, then we require first that $\gamma(2) - \gamma(1)$ be as small as possible, then that $\gamma(3) - \gamma(2)$ be as small as possible, and so on through all $\gamma(j) - \gamma(j-1)$. Trivially, $\gamma(1) = 1$. Since $\phi_2(\vec{r})$ is necessarily linear, to meet this "modified requirement of rotation invariance" for $\gamma(2) - \gamma(1)$ we do indeed have to have $\phi_3(\vec{r})$ through $\phi_{m+1}(\vec{r})$ be linear functions of the $r_i$. As above this then forces $\phi_{m+2}(\vec{r})$ to be a quadratic, and so we have to have all the other linearly independent quadratics (there are $m(m-1)/2$ of them) coming next in $\Phi$. Iterating this argument, it is clear that

**(G.3)** The only basis sets of functions $\Phi$ whose associated LMMGs obey the invariances of (2.6) (with the requirement of rotation invariance in the input space modified as above) are the sets of functions

$$\{1, (r_1, r_2, \ldots, r_m), (r_1^2, \ldots, r_m^2), \ldots\},$$

where notationally $(x, y, \ldots, z)$ means all possible bases which span the same space as the functions $x, y, \ldots, z$.

This concludes the proof of the proposition. Again, note that if we do not make any requirements concerning rotation invariance at all, then all we can say is that $\Phi$ must be a set of polynomials, arranged in any order, which spans the space. Similarly, if we do not require scaling invariance in the input space, the individual elements of $\Phi$ can have exponentials added to them.

## Appendix H.

### The npa generalizer

The npa generalizer begins its calculation of the output corresponding to any given question by computing the input space vector $\vec{r}_0$ from the question to the point in the learning set nearest the question. (In general, all vectors are here assumed to be relative to the question.) Next it forms an $m$-dimensional open hypersphere centered on the question with radius $2|\vec{r}_0|$. Label the position vectors of the points from the learning set which fall within the hypersphere as $\vec{r}_i$. Assume there are $n$ of them, so $0 \leq i \leq n-1$, where it is assumed that $|\vec{r}_i| \geq |\vec{r}_{i-1}|$ for all $i$. Label the associated outputs by $m_i$. Let $\vec{r}_n$ be any point $2|\vec{r}_0|$ away from the question. Now define the value of the scalar field $\rho(\vec{r})$ to be the median of the $m_j$ whose associated positions obey $|\vec{r}_j| \leq |\vec{r}|$. In other words, at any point $\vec{r}$, $\rho(\vec{r})$ is the average of the smallest and the largest $m_i$ which lie inward of $\vec{r}$. The output value the generalizer guesses for the question is given by $[\int_{|\vec{r}_0|}^{2|\vec{r}_0|} \rho(\vec{r}) d^m r] / [\int_{|\vec{r}_0|}^{2|\vec{r}_0|} d^m r]$. If one of the $m_i$ changes by an infinitesimal amount $\delta m$, then its greatest possible effect on the generalizer's guess would be if it's at $\vec{r}_0$, and is (say) the largest $m_i$ throughout the whole hypersphere. In this case, $\rho(\vec{r})$ everywhere increases by at most $\delta m/2$ ($\delta m$ if this point happens to be the only one in the hypersphere), and therefore the generalizer's output increase by $\delta m/2$. Consequently, this generalizer is continuous in the $m_i$. Furthermore, although $\rho(\vec{r})$ is in general a discontinuous function of $\vec{r}$ and the $\vec{r}_i$, $\int_{|\vec{r}_0|}^{2|\vec{r}_0|} d^m r$ is a continuous function of the $\vec{r}_i$, as is $\int_{|\vec{r}_0|}^{2|\vec{r}_0|} \rho(\vec{r}) d^m r$, since any infinitesimal variation in $\vec{r}_i$ can make a noninfinitesimal change in $\rho(\vec{r})$ only over an infinitesimally thin (hyper)shell. As a result, the output of the generalizer is a continuous function of the $\vec{r}_i$. Similarly, an infinitesimal variation in the question, being equivalent to an infinitesimal variation in all the $\vec{r}_i$, can only result in an infinitesimal variation in the output. This verifies requirement (2.1) of properness. Now note that if we translate all the $m_i$ up or down by some constant $k$, $\rho(\vec{r})$ is translated by the same $k$, and therefore the output as a whole is translated by $k$. Similarly, if we multiply all the $m_i$ by a scaling factor $z$, the output gets multiplied by $z$. Since the output is purely a function of metric distances, which are tensor scalars, the generalizer is automatically invariant under rotation, parity and translation. Invariance under nonzero scaling of all the inputs and the question follows from how the

output's normalized. This completes the verification of requirement (2.6). Requirement (2.2) is immediate, since no *a priori* ordering is assumed of the datum spaces. Requirement (2.4) is a restriction on the learning set, not the generalizer. Given a learning set, as the question approaches any point in the learning set $|\vec{r}_0|$ and therefore $2|\vec{r}_0|$ shrink to 0, by requirement (2.4) and continuity all points within the hypersphere approach the value $m_0$, $\rho(\vec{r})$ approaches $m_0$ everywhere within the hypersphere, and the output of the generalizer becomes $m_0$. Therefore reproduction of the learning set is assured, and requirement (2.3) is met. As one of the $\vec{r}_i$ becomes arbitrarily close to $\vec{r}_{i-1}$, $m_i$ approaches $m_{i-1}$, and $\rho(\vec{r})$ gets arbitrarily close to what it would be if the point at $\vec{r}_i$ had never been an element of the learning set. This verifies requirement (2.8), and therefore properness in general.

## Appendix I.

### Proof that all LMMGs are output linear

Assume we have one learning set, $\theta_1$, consisting of the $n$ pairs

$$(x_1, y_1), \ldots (x_n, y_n),$$

and another learning set $\theta_2$ consisting of the $n$ pairs $(x_1, y'_1), \ldots (x_n, y'_n)$, where the vectors $x_j$ might be multidimensional, in general. Assume further that our LMMG fits $\theta_1$ with the surface $\sum_{i=1}^{N} \alpha_i \phi_i(\vec{r})$ and fits $\theta_2$ with $\sum_{i=1}^{N'} \alpha'_i \phi_i(\vec{r})$. So $\sum_{i=1}^{N} \alpha_i \phi_i(x_j) = y_j$ and $\sum_{i=1}^{N'} \alpha'_i \phi_i(x_j) = y'_j$ for all $n$ $x_j$. Without loss of generality we can take $N \geq N'$. We can then extend the set of $\alpha'_i$ to those $i$ between $N$ and $N'$ (including $N$) by simply setting $\alpha'_i = 0$ for $N \geq i > N'$. So the LMMG fits $\theta_1$ with $\sum_{i=1}^{N} \alpha_i \phi_i(\vec{r})$ and fits $\theta_2$ with $\sum_{i=1}^{N} \alpha'_i \phi_i(\vec{r})$. Now it is obvious that

$$A \sum_{i=1}^{N} \alpha_i \phi_i(\vec{r}) + B \sum_{i=1}^{N} \alpha'_i \phi_i(\vec{r}) = \sum_{i=1}^{N} (A\alpha_i + B\alpha'_i) \phi_i(\vec{r})$$

goes through the points $\{x_j, Ay_j + By'_j\}$ for all $j \leq N$ and $\geq 1$. If we can prove that the LMMG will actually pick these coefficients $A\alpha_i + B\alpha'_i$ to fit the learning set $\theta'$ consisting of the points $\{x_j, Ay_j + By'_j\}$, then we will have proven that the LMMG is output linear. Since no particular attributes were assumed of the LMMG, we will have proven in fact that all LMMGs are output linear.

So first we must prove that it is not possible for the LMMG to fit the new learning set without using the function $\phi_N(\vec{r})$. (If the LMMG could fit the learning set without using $\phi_N(\vec{r})$ it would do so, by definition of LMMGs (see appendix E), and therefore would not generalize from $\theta'$ with the surface $\sum_{i=1}^{N} (A\alpha_i + B\alpha'_i) \phi_i(\vec{r})$.) In other words, we need to prove that if $\theta_1$ needs to be fit with $\phi_N(\vec{r})$ and $\theta_2$ needs to be fit with $\phi_{N' \leq N}(\vec{r})$, then $\theta' = \theta_1 + \theta_2$ needs to be fit with $\phi_N(\vec{r})$. To prove this assertion it suffices to disprove its negation. If $\theta'$ could be fit using only functions with ordinality $< N$, after

rearranging terms we would see that there must exist a set of coefficients $\beta_i$, $1 \leq i < N$, such that

$$\sum_{i=1}^{N} \alpha_i \phi_i(x_j) = \sum_{i=1}^{N-1} \beta_i \phi_i(x_j)$$

for all $x_j$ in $\theta'$. However if this were true, then we could have fit $\theta_1$ with only the first $N-1$ of the $\phi_i(\vec{r})$. This is contrary to our initial assumption that the LMMG actually fit $\theta_1$ using the first $N$ of the $\phi_i(\vec{r})$, so it is true that the LMMG cannot fit $\theta'$ without using $\phi_N(\vec{r})$.

It was shown in appendix E that the coefficient the LMMG fits to $\phi_N(\vec{r})$ is unique, so that coefficient must indeed equal $A\alpha_N + B\alpha'_N$. So now we must prove that the LMMG will also choose the values $A\alpha_{i<N} + B\alpha'_{i<N}$ (as the coefficients of the basis functions $\phi_{i<N}(\vec{r})$) to construct a surface which fits $\theta'$. To carry out such a proof we need to assume some means for the LMMG to choose between two (or more) possible sets of coefficients which both fit a given learning set and which both have the same $N$. Here I will assume the choosing scheme outlined in appendix E.

Examine the $(N-1)$th coefficient which the LMMG uses to fit $A\theta + B\theta' : \beta_{N-1}$. (We already know that $\beta_N = A\alpha_N + B\alpha'_N$.) In general either the vector $\{\phi_{N-1}(\vec{r}_1), \phi_{N-1}(\vec{r}_2), \ldots, \phi_{N-1}(\vec{r}_n)\}$ is linearly independent of the vectors $\{\phi_i(\vec{r}_1), \phi_i(\vec{r}_2), \ldots, \phi_i(\vec{r}_n)\}$, $1 \leq i < N-1$, or it is linearly dependent on them. If it is linearly dependent on them, then we can set $\beta_{N-1}$ equal to 0. In light of our choosing method, in point of fact we have no choice but to choose the $\beta_i$ such that $\beta_{N-1} = 0$ if this linear dependence holds. Similarly, if we have this linear dependence then both $\alpha_{N-1}$ and $\alpha'_{N-1}$ must $= 0$. Therefore, in this situation, $\beta_{N-1}$ does indeed $= A\alpha_{N-1} + B\alpha'_{N-1}$.

Now examine the case where the vector

$$\{\phi_{N-1}(\vec{r}_1), \phi_{N-1}(\vec{r}^2), \ldots, \phi_{N-1}(\vec{r}_n)\}$$

is linearly independent of all the vectors

$$\{\phi_i(\vec{r}_1), \phi_i(\vec{r}_2), \ldots, \phi_i(\vec{r}_n)\}, \ 1 \leq i < N-1.$$

Hypothesize that the $\beta_{N-1}$ which reproduces the learning set $\theta'$ is not unique, and represent the alternative set of coefficients as $\beta'_i$, $1 \leq i \leq N-1$. Since by hypothesis of nonuniqueness $\beta_{N-1} \neq \beta'_{N-1}$ and yet $\sum_{i=1}^{N-1} \beta_i \phi_i(\vec{r}_j) = \sum_{i=1}^{N-1} \beta'_i \phi_i(\vec{r}_j)$ for all $\vec{r}_j$ in the learning set, we can write

$$\{\phi_{N-1}(\vec{r}_1), \phi_{N-1}(\vec{r}_2), \ldots, \phi_{N-1}(\vec{r}_n)\}$$

as a linear combination of the vectors

$$\{\phi_i(\vec{r}_1), \phi_i(\vec{r}_2), \ldots, \phi_i(\vec{r}_n)\}, \ 1 \leq i < N-1.$$

Such an equality, however, is in violation of our assumption of linear independence. Therefore our hypothesis of nonuniqueness must be wrong, and if we do indeed have linear independence then we can conclude that $\beta_{N-1}$ must be

unique. Since we know that the coefficient $A\alpha_{N-1} + B\alpha'_{N-1}$ is a legal coefficient of the $\phi_{N-1}\vec{r}$ term, we can in fact conclude that $\beta_{N-1} = A\alpha_{N-1} + B\alpha'_{N-1}$ for this case of linear independence as well as for the previously explored case of linear dependence.

These same arguments can be used for all the remaining coefficients the LMMG fits to $A\theta_1 + B\theta_2$. The result is clearly that $\beta_i = A\alpha_i + B\alpha'_i$ for all $i$. This concludes the proof that LMMGs are output linear.

## Appendix J.

### Output linear HERBIEs and neural nets

To see that neural net neurons satisfy the restrictions on $h$ for $n = 2$, output linear, one-dimensional HERBIEs, note that for any output linear generalizers,

$$\begin{aligned} h(x+b, x'+b, x''+b, \ldots, q+b) &= h(ax, ax', ax'', \ldots, aq) \\ &= h(x, x', x'', \ldots q), \end{aligned}$$

due to freedom in picking the $y, y', \ldots$ and invariance under scaling and translating the input space. Since the equality holds for arbitrary $a$ and $b$, if the $x, x', x'', \ldots$ are not all the same as one another then the value of $h$ is the same everywhere on the plane in $(x, x', x'', \ldots q)$ space formed by allowing $a$ and $b$ to take on all possible values. (For $x = x' = \ldots = q$, varying $a$ and $b$ only traces out a line, not a plane.) For an $n = 2$, one-dimensional HERBIE, restriction (2.5) in fact requires that $x \neq x'$, so the generalization is indeed determined by such iso-$h$ planes. So given any triplet of values $X$, $X' \neq X$, $Q$ in $(x, x', q)$ space, we have a unique iso-$h$ plane. Now any such plane has to intersect the line $x = -1/2$, $x' = 1/2$, as can be seen by solving for $a$ and $b$ in the pair of equations $aX + b = -1/2$, $aX' + b = 1/2$ (since $X \neq X'$, we always get such a solution for $a$ and $b$). Since $a$ and $b$ are therefore uniquely specified by $X$ and $X'$, $\mathcal{Q} = aQ + b$, the q value of the intersection of the plane with the line $x = -1/2$, $x' = 1/2$, is specified uniquely. Because any point on an iso-$h$ plane completely specifies that plane, we can therefore use $(-1/2, 1/2, \mathcal{Q})$ to specify the plane instead of $(X, X', Q)$ — the plane specified by $(X, X', Q)$ is the same as the locus of points $(-a'/2 + b', a'/2 + b', a'\mathcal{Q}/2 + b')$ for all $a'$ and $b'$. As a result, the iso-$h$ planes form a one-parameter set, with $\mathcal{Q}$ being the parameter. Now all points $(x, x', q)$ on any given iso-$h$ plane will agree on the value for $w = 1/2 - (q - x)/(x' - x)$, and this function specifies a unique $\mathcal{Q}$ value for the intersection of the plane containing the points with the line $x = -1/2$, $x' = 1/2$. Therefore $w$ uniquely specifies the iso-$h$ plane. Under interchange of $x$ and $x'$, $w$ becomes $-w$. Since from before $h(x', x, q) = 1 - h(x, x', q)$, for all $x$, $x' \neq x$, and $q$, $h(-w) = 1 - h(w)$. In particular since $h(q, x', q) = 1$, we can write $h(w = 1/2) = 1$, and $h(-1/2) = 0$. Also $h(0) = 1/2$. This of course is exactly the behavior exhibited by fermi functions (for appropriate rescaling of the $w$ so that $w = 1/2$ becomes $w = \infty$ and $w = -1/2$ becomes $w = -\infty$), step functions and most of the other

thresholding functions used to model neurons in neural nets. Note that since there exist an infinite number of functions $h(w)$ obeying $h(w) = 1 - h(-w)$ and $h(1/2) = 1$, the restriction of output linearity is not enough to force unique generalization of HERBIEs. Also note that most of this analysis applies equally well to the $g\{i\}$ of HERBIEs seen as functions only over the $x_1, x_2, \ldots, q$.

To place the iso-$h$ concept in a larger context, view the direct product of the input components of the datum spaces for a given order $g\{i\}$, $(x, x', x'', \ldots, q)$, as defining a vector space $V$. Then the invariances of restriction (2.6), in addition to being viewed as referring to each component of a vector in $V$ undergoing coordinate transformations within its own private $\mathbb{R}^m$ space, can also be viewed as referring to coordinate transformations in the whole space $V$. The iso-$h$ planes are planes of invariance in $V$. If we now construct a space $V'$ with elements $(y, y', y'', \ldots, \text{guess})$ in exactly the same way we constructed $V$, then datum space interchange symmetry is an invariance of $g\{i\}$ under certain simultaneous reflections (those which flip around the axes) in both $V$ and $V'$. Although it has not yet been done (mainly because it's hard to see what its physical significance would be) it might be mathematically interesting to explore the consequences of requiring that the $g\{i\}$ be invariant under more general sets of simultaneous reflections in $V$ and $V'$. For output linearity, where the $g\{i\}$ can be viewed as a sort of dual vector field, such an invariance amounts to a symmetry under simultaneous reflections of the underlying manifold across which the field is defined and the accompanying transformation of the individual vector spaces living on that manifold.

## Appendix K.

### Proof that for $n = 2$, the solution to (2.12) is a straight line or an exponential

Clearly both a linear $f(x)$ and an exponential one are solutions to this new restriction. To prove uniqueness of such these solutions, we want to solve for the set of all $f(x)$ which obey (2.12) and give an $h$ obeying the coordinate transformation invariances required by the generalizer's being a HERBIE. To do this, first exploit the symmetries allowed all HERBIEs to translate and scale the input space and then translate the output space so that any original triple of points $(x, f(x))$, $(x', f(x'))$, $(q, f(q))$ becomes $(0, 0)$, $(a, b)$, $(c, d)$, respectively, where the original $x$, $x'$, and $q$ are assumed chosen such that $c > a > 0$. Due to the fact that the value of $h$ is unchanged if its three arguments are all translated by the same amount, if $q$ now refers to a point a little to the right of $c$,

$$\frac{f(q) - f(q - (c - a))}{f(q - (c - a)) - f(q - c)} = \frac{d - b}{b} = \frac{f(c)}{f(a)} - 1. \tag{K.1}$$

We can ignore the case where both $d$ and $b = 0$, since by (2.12) this means $f(q)$ is the straight line $f(q) = 0$. By continuity, $f(x)$ is nonzero everywhere

in a noninfinitesimal neighborhood of whichever point $a$ or $c$ obeys $f(x) \neq 0$. Without loss of generality, we will from now on assume that both $a$ and $c$ have been chosen to be from this region of $f(x) \neq 0$. Defining $\delta$ to be $q - c$, the amount by which the input space was translated in making (K.1), we can solve (K.1) for $f(q)$, getting

$$f(q) = [f(\delta + a) - f(\delta)]f(c)/f(a) + f(\delta).$$

Now change $a \rightarrow a + \epsilon$, with $\epsilon$ small enough so that $f(a + \epsilon)$ still does not $= 0$. This results in

$$f(q) = [f(\delta + a + \epsilon) - f(\delta)]\frac{f(c)}{f(a + \epsilon)} + f(\delta).$$

Equating the two formulas for $f(q)$,

$$f(a)[f(\delta + a + \epsilon) - f(\delta)] = f(a + \epsilon)[f(\delta + a) - f(\delta)]. \qquad \text{(K.2)}$$

This must be true for all $\delta$ and (small enough) $\epsilon$. Let $\delta$ and $\epsilon$ both be small and expand both sides of (K.2) to second order in $\delta$ and $\epsilon$. We recover $f(0) = 0$ (which we knew already) and for the $\delta\epsilon$ term get $f(a)f''(a) = f'(a)[f'(a) - f'(0)]$. This must be true for a whole range of $a$, so setting $f'(0)$ equal to a constant, $\alpha$, and letting $a$ vary, we get a second-order nonlinear differential equation:

$$f(x)f''(x) - (f'(x))^2 + \alpha f'(x) = 0.$$

Now change variables to $u(y) \equiv y'(x)/y(x)$. This transforms our differential equation to $\frac{du}{dy} = -\frac{\alpha}{y^2}$. This has the solution $u \equiv \frac{y'}{y} = \frac{\alpha}{y} + \beta$, $\beta$ being an arbitrary real constant. If $\beta = 0$, the solution for $y(x)$ is a straight line. If $\beta$ doesn't equal 0, the solution is $\gamma e^{\beta x} - \alpha/\beta$, where plugging in to equation (K.2) yields $\gamma = \alpha/\beta$. ∎

## References

[1] D. Wolpert, "A benchmark for how well neural nets generalize," *Biological Cybernetics*, **61** (1989) 303–315.

[2] D. Wolpert, "Using a mathematical theory of generalization to build a generalizer superior to NETtalk," *Neural Networks*, in press.

[3] D. Wolpert, "Generalization, surface-fitting, and network structures," Ph.D. thesis, University of California, 1989.

[4] D.E. Rumelhart, et al., *Explorations in the Microstructure of Cognition*, Vol. I and II (MIT Press, Cambridge, MA, 1986).

[5] D.E. Rumelhart, et al., "Learning representations by back-propagating errors," *Nature*, **323** (1986) 533–536.

[6] T.J. Sejnowski and C.R. Rosenberg, *NETtalk: A Parallel Network that Learns to Read Aloud*, Johns Hopkins University Electrical Engineering and Computer Science Technical Report JHU/EECS-86/01, 1986.

[7] R. Linsker, "Self-organization in a perceptual network," *Computer*, **3** (1988) 105–117.

[8] T.J. Sejnowski, "Neural net models of visual processing," *Short Course on Computational Neuroscience* (Society for Neuroscience, Department of Biophysics, Johns Hopkins University, 1987).

[9] D.O. Hebb, *The Organization of Behavior* (Wiley, New York, 1949).

[10] O.S. McCullough and WQ. Pitts, "A logical calculus immanent in nervous activity," *Bulletin of Mathematical Biophysics*, **5** (1943) 115–133.

[11] Andrew Moore, Personal communication, 1988.

[12] B. McNaughton, et al., "Hippocampal synaptic enhancement and information storage within a distributed memory system," *Trends in Neurosciences*, **10**, 408–415 (Elsevier Publications, Cambridge, 1987).

[13] *The Brain, A Scientific American Book* (W.H. Freeman, San Francisco, 1979).

[14] P.L. Carlton, *A Primer of Behavioral Pharmacology* (W.H. Freeman, San Francisco, 1983).

[15] J.W. Clark, "Statistical mechanics of neural networks," *Physics Reports*, **158** (1988) 91–157.

[16] D. Amit, et al., "Spin glass models of neural networks," *Physical Review A*, **32** (1985) 1007–1018.

[17] B. Derrida, et al., "An exactly soluble asymmetric neural network model," *Europhysics Letters*, **4** (1987) 167–173.

[18] J.J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities," *Proceedings of the National Academy of Science*, **79** (1982) 2554–2558.

[19] H. Gutfreund, "Neural networks with hierarchically correlated patterns," UCSB ITP preprint NSF-ITP-86-151 (1986).

[20] A. Dembo, et al., "General potential surfaces and neural networks," *Physical Review A*, **37** (1988) 2134–2143.

[21] J.J. Hopfield and D.W. Tank, "Neural computation of decisions in optimization problems," *Biological Cybernetics*, **52** (1985) 141–152.

[22] G.V. Wilson and G.S. Pawley, "On the stability of the traveling salesman problem algorithm of Hopfield and Tank," *Biological Cybernetics*, **58** (1988) 63–70.

[23] I. Parberry, et al., *Relating Boltzmann machines to conventional models of computation*, Pennsylvania State University Department of Computer Science Technical Report CS-87-07, March, 1987.

[24] M. Minsky and S. Papert, *Perceptrons* (MIT Press, Cambridge, MA, 1969).

[25] S. Kirkpatrick, et al., "Optimization by simulated annealing," *Science*, **229** (1983) 671–680.

[26] A. Bergmann, et al., "Breeding intelligent automata," *Proceedings of the IEEE Conference on Neural Nets* (San Diego, 1987).

[27] A. Lapedes and R. Farber, "Non-linear signal processing using neural nets: Prediction and system modeling," Los Alamos preprint LA-UR-87-2662 (1987).

[28] T. Smith, Personal communication, 1987.

[29] R. Solomonoff, "A formal theory of inductive inference: I and II," *Information and Control*, **7** (1964) 1, 224.

[30] S. Watanabe, "Inductive ambiguity and the limit of artificial intelligence," *Computational Intelligence*, **3** (1987) 304–309.

[31] N. Littlestone, A. Bergmann, and P. Caianiello, Personal communication, 1988.

[32] T. Poggio and staff, MIT AI Lab, "MIT progress in understandings," to be published in *Proceedings of the Image Understanding Workshop*, L. Bauman, ed. (SAI Corporation, McLean, VA, 1988).

[33] T. Poggio, et al., "Computer vision and regularization theory," *Nature*, **317** (1985) 314–319.

[34] *Computational Intelligence*, **3** (1987) special issue on machine learning.

[35] L.G. Valiant, "A theory of the learnable," *Communications of the ACM*, **27** (1984) 1134–1142.

[36] N. Littlestone, "Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm," *Machine Learning*, **2** (1988) 285–318.

[37] R. Duda and P. Hart, *Pattern classification and scene analysis* (Wiley, New York, 1973).

[38] V. Cappellini and R. Marconi, eds., *Advances in Image Processing and Pattern Recognition: Proceedings of the International Conference Pisa, Italy, December, 1985* (Elsevier, 1986).

[39] J.D. Farmer and J.J. Sidorowich, "Exploiting chaos to predict the future and reduce noise," Los Alamos preprint LA-UR-88-901 (1988).

[40] A. Lapedes and R. Farber, "How neural nets work," *Proceedings of the 1987 IEEE Denver Conference on Neural Networks.*

[41] E.W. Packel, et al., "Information-based complexity," *Nature*, **328** (1987) 29–33.

[42] P. Garcia, et al., "Local languages, the successor method, and a step towards a general methodology for the inference of regular grammars," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **pami-9** (1987) 841–845.

[43] K.S. Fu, et al., "Grammatical inference: Introduction and survey parts 1 and 2," *IEEE Transactions on Systems, Man, and Cybernetics*, **5** (1975) 95–111, 409–423.

[44] C. Thompson, et al., "On limits to perception and pattern recognition," Institute for theoretical physics at Santa Barbara report, UCSB ITP preprint NSF-ITP-87-127, 1987.

[45] T.M. Mitchell, "Generalization as search," *Artificial Intelligence*, **18** (1982) 203–226.

[46] J.E. Hopcroft and J.D. Ullman, *Introduction to Automata Theory, Languages, and Computation* (Addison-Wesley, 1979).

[47] Holland, John, *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence* (University of Michigan Press, Ann Arbor, 1975).

[48] E.T. Jaynes, "On the rationale of maximum-entropy methods," *Proceedings of the IEEE*, **70** (1982) 939–952.

[49] F. Reif, *Fundamentals of Statistical Physics and Thermal Physics* (McGraw-Hill, 1965).

[50] G. Birkhoff and S. Maclane, *A Survey of Modern Algebra*, 4th ed. (Macmillan Publishing Co., New York, 1977).

[51] R. Wald, *General Relativity*, (University of Chicago Press, Chicago, IL, 1984) chapter 2.