World Academy of Science, Engineering and Technology
International Journal of Mathematical and Computational Sciences
Vol:4, No:8, 2010

Coupled dynamics in host-guest complex systems duplicates emergent behavior in the brain

Sergio Pissanetzky, *Member, IEEE, AAAI, APS*

Abstract – The ability of the brain to organize information and generate the functional structures we use to act, think and communicate, is a common and easily observable natural phenomenon. In object-oriented analysis, these structures are represented by *objects*. Objects have been extensively studied and documented, but the process that creates them is not understood. In this work, a new class of discrete, deterministic, dissipative, host-guest dynamical systems is introduced. The new systems have extraordinary self-organizing properties. They can host information representing other physical systems and generate the same functional structures as the brain does. A simple mathematical model is proposed. The new systems are easy to simulate by computer, and measurements needed to confirm the assumptions are abundant and readily available. Experimental results presented here confirm the findings. Applications are many, but among the most immediate are object-oriented engineering, image and voice recognition, search engines, and Neuroscience.

Keywords – AI, artificial intelligence, complex system, object-oriented, OO, refactoring.

I. Introduction

In this work, the brain is considered as an implementation of a complex dynamical system that receives some input, processes it, and generates some output. Only the input/output relationship is of concern. The internal process is described by a simple, discrete mathematical model with only minimal assumptions about the nature of the dynamics. The details of the implementation, no matter how complicated they may be, are irrelevant.

The dynamical system being studied is called the *host*. The input information describes the behavior of another, given physical system, known as the *guest*. The output generated by the host consists of hierarchies of objects that are *behaviorally equivalent* to the input information, in the sense that both the input information and the output structures describe the same behavior, the only difference being that the output is organized even if the input is not. Accordingly, one can say that the behavior of the guest is *invariant* under the transformations induced by the host's dynamics. The new systems are properly called *host-guest* systems, and the problem is one of *coupled dynamics*. This work is concerned with the dynamics of the host. Restrictions placed on the guest are minimal and are discussed below.

If *feedback* is present, in such a way that the structures obtained by the host are fed back to the guest, then the net effect is that the guest system gets better organized, its behavior has not changed, the phase space is now much smaller, and a phenomenon of *adaptation* has occurred.

Sergio Pissanetzky, Ph.D., Research Scientist
Sergio@SciControls.com

The assumptions made about the dynamics of the host must be confirmed by comparing theory with measurement. In this case the brain is the "experiment", observations of the input/output relationship of a person's brain are the "measurements", and a computer simulation of the mathematical model represents the theory. Figure 1 shows a schematic representation of a brain experiment. These experiments are carried out only on humans, involve higher brain functions, and directly detect the *behavior* of the guest system. Luckily, a very large number of measurements of this type have been performed, and the results have been carefully documented in multiple sources.

One of such sources, is object-oriented (OO) code. In the course of development of OO code, a human analyst receives a *problem statement* that describes some physical system, such as a business, or a machine, or perhaps a problem of Physics that needs to be solved, and is asked to create an object-oriented model of that system. The model is subsequently used to develop software and documentation. This process amounts to a carefully controlled experiment where the given problem is the guest system, the analyst's brain is the host, the problem statement serves as input, the analyst's brain provides the transformation dynamics, and the resulting OO code and documentation are the output. If the code is used to run the business, design the machine, or solve the problem, then feedback and adaptation have occurred.

The second source of carefully documented experiments is provided by the theories of natural science, that is, by the theories themselves, not the measurements on which the theories are based. A theory of science also consists of

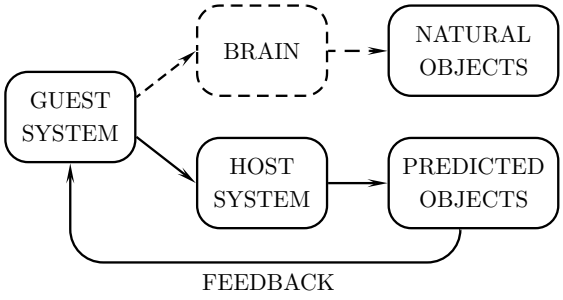

Fig. 1. A brain experiment. The elements indicated with solid lines are externally observable and discussed in this paper. Brain-generated natural objects represent experimental observations. Predicted objects calculated by the simulation are compared with the natural objects and found to confirm the theory. Adaptation occurs when the behavior-invariant objects are fed back to the guest system.

objects. In this case, the guest is the physical system being observed, the host is the scientist's brain, input consists of the experimental measurements that support the theory, the scientist's brain provides the dynamics, and the output is the theory. In this case, feedback happens when we learn the theory, the theory serves us to better our own understanding of the natural world, and we ourselves have better adapted to our environment.

The third source of interesting brain experiments, is image recognition. A television picture that consists of illuminated pixels can be the input. The eye and brain of the person who is watching the image provide the dynamics, and the output is the recognition of the objects present in the scene. Again, by interpreting a set of illuminated pixels as a meaningful image, we have better adapted to our environment.

A different type of brain experiments involves the use of *in-vitro* cultures of live neurons obtained from animal brains and placed on multi-electrode arrays. The neurons spontaneously organize in the culture, create a network, and become active. The electrodes detect the action potentials of individual neurons. These experiments seek to detect and measure the *chaotic dynamics* of the host system, and they can be carried out only with animals. They are discussed in Section VIII.

Host-guest systems should not be confused with applications where objects are predictably created, such as 2nd and higher degree mathematical logic, OO programming, including "object factories" where rules are used to create objects of different classes, image or speech recognition methods, game-playing machines, etc. In all cases, the objects or rules come from a human, the human's brain provides the dynamics, and the rest of the process is entirely predictable. The new systems, instead, use a stochastic process that prevents any predictable connection between guest and host other than the constraints, and the source of the objects is the ensuing emergent behavior. The new systems should neither be confused with host-guest composites and molecular assemblies, which have long been studied in organic and inorganic chemistry, but have not been extended to systems in general.

A mathematical introduction and a motivational example are presented in Section II. The mathematical model of computation for the host-guest system is discussed in Section III. The dynamics is defined in Section IV, and the physical characterization is covered in Section V. Just as the brain is considered as a black box with an input and an output, any functional part of the brain can as well be considered as a black box with an input and output. This idea gives rise to the notion of *brain analysis*, discussed in Section VI. Algorithms and a computer implementation are discussed in Section VII. Experiments from each one of the sources of brain measurements mentioned above have been published, and are reviewed and discussed in Section VIII. The agreement between theory and experiment is very good in all cases. Two areas where host-guest simulations can outperform humans are also discussed in Section VIII.

Terms such as self-organizing, emergent, deterministic, and others, have precise meanings [1], and are used here with those precise meanings. A method based on rule abstraction has been proposed to study the brain's emergent properties [2]. However, complex systems are indivisible (see §V). Brain analysis methods (§VI) should be used instead.

II. Mathematical Introduction and Motivational Example

Two pillars support the work that is being presented. The first pillar is a computationally observed mathematical property. Let F be a finite set with a partial order ω, and let K be a total order on F compatible with ω. Order K is called a *configuration*. A subset $\varphi \subseteq F$ is said to be a *segment* in K if its elements are consecutive in K. Set F itself is always a segment in any K.

Now let $\mathcal{K} = \{K\}$ be the set of all configurations. Since F is finite, $\mathcal{K}$ is also finite. Set $\mathcal{K}$ is required to be *measurable*, meaning that a systematic procedure must be given to assign a number, say $L(K)$, to each configuration $K \in \mathcal{K}$. Let $L_{\min}$ be the minimum value of $L(K)$ over $\mathcal{K}$, and define:

$$\widetilde{\mathcal{K}} = \{K \mid L(K) = L_{\min}\} \tag{1}$$

Set $\widetilde{\mathcal{K}}$ is the set of all configurations with minimum measure. Then, the following statement can be made: *There exists a non-trivial partition Π_F of set F, such that, for each $P \in \Pi_F$, and each $K \in \widetilde{\mathcal{K}}$, P is a segment under K*. This statement is the *set partition conjecture*. It has been computationally confirmed in many cases and with several different sets and types of measures, but it has not yet been determined if it is new or established mathematics. The segments in the partition give rise to *objects*.

A motivational example is given next. Let set F be given by $F = \{f_\alpha, f_\beta, f_\gamma, f_\delta, f_\epsilon, f_\varphi\}$, or, to simplify notation and improve readability:

$$F = \{\alpha, \beta, \gamma, \delta, \epsilon, \varphi\} \tag{2}$$

and let the partial order on F be the following:

$$\omega = \{\alpha < \delta,\ \gamma < \delta,\ \delta < \epsilon,\ \beta < \varphi\}. \tag{3}$$

An order is simply a set of precedence relations among pairs of elements of F. An easy and very visual way to represent the propositions leading to the set partition conjecture, is to cast them in matrix form. The result is a *canonical* matrix [3]. To create the canonical matrix, a configuration is first chosen, and the elements of F are assigned to the rows in the order of that configuration. Then, each relation in the partial order, say $i < j$, is indicated with a C on the diagonal in position (i, i) and an A in position (j, i). Symbols A and C are explained in the next section. The result is a square, sparse [4], lower triangular matrix of order $|F|$, with a full diagonal. An advantage of this representation is that configurations in set $\mathcal{K}$ correspond to symmetric permutations of the matrix. The C's always remain on the diagonal, and the partial order is satisfied as long as all the A's remain in the lower triangle. The canonical matrix for this example is shown in Fig. 2, and it corresponds to the following configuration:

$$K = (\alpha, \beta, \gamma, \delta, \epsilon, \varphi). \tag{4}$$

Since a configuration is a total order, it can be written as an ordered tuple.

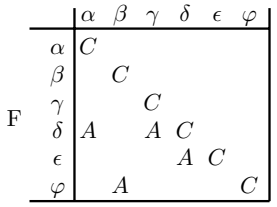

Fig. 2. A 6×6 canonical matrix corresponding to the partial order ω for set F given in Eq. (3) and the configuration K given in Eq. (4).

The next step is to define a measure that applies to all configurations. The definition is arbitrary, but it must be given. One choice that has consistently led to good results is:

$$L(K) = 2\sum(j - i) \tag{5}$$

where the sum extends to all A's in the canonical matrix, and the reason for selecting the coefficient 2 is explained in the next section. For the configuration K of Eq. (4) and shown in Fig. 2, $L(K) = 18$.

The final step is to find the set of least-measure configurations $\widetilde{\mathcal{K}}$, and determine the partition Π_F of set F. Since $\mathcal{K}$ is finite, the search can be done by enumeration. This is easy for such a small example. Set $\widetilde{\mathcal{K}}$ consists of the four configurations

$$\begin{aligned}&(\alpha, \gamma, \delta, \epsilon, \beta, \varphi)\\ &(\beta, \varphi, \alpha, \gamma, \delta, \epsilon)\\ &(\beta, \varphi, \gamma, \alpha, \delta, \epsilon)\\ &(\gamma, \alpha, \delta, \epsilon, \beta, \varphi)\end{aligned} \tag{6}$$

The measure for each of these configurations is $L_{\min} = 10$. The partition consists of subsets $\{\beta, \varphi\}, \{\alpha, \gamma\}$, and $\{\delta, \epsilon\}$, each of which is a segment in each one of the four configurations.

But the process is not finished yet. The set whose elements are the three subsets $\{\beta, \varphi\}$, $\{\alpha, \gamma\}$, and $\{\delta, \epsilon\}$ meets all three conditions required by the set partition conjecture. It is a set, it has a partial order, consisting in this case of the single relation $\{\alpha, \gamma\} < \{\delta, \epsilon\}$ obtained directly from Eq. (3), and a measure can be defined for it in the same way as in Eq. (5). The reader can easily verify that the resulting partition contains the two subsets $\{\beta, \varphi\}$ and $\{\{\alpha, \gamma\}, \{\delta, \epsilon\}\}$.

Even for such a very small example, a nested hierarchy of partitions with several levels has been obtained. It is called an *inheritance hierarchy*. Inheritance hierarchies are well known in object-oriented analysis. Here, they have been mathematically explained.

The mathematical property reported in this Section has an extraordinary significance. However, sets alone can not represent physical systems. Unless they are sets of functions. A model of computation based on sets of functions is presented next.

III. The Matrix Model of Computation

The second pillar is the mathematical model for the new host-guest dynamical systems. The model is the *Matrix Model*

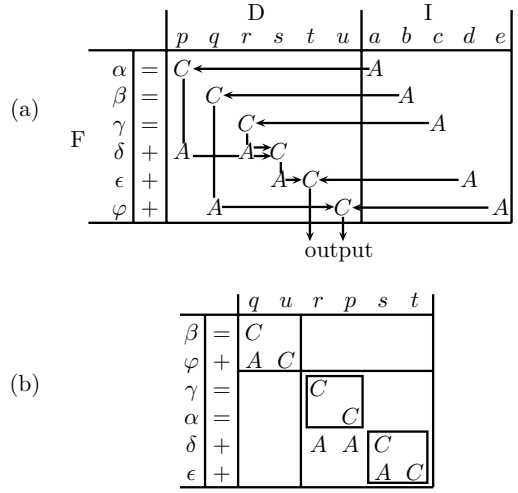

Fig. 3. (a) A 6×6 canonical matrix and a 6×5 input matrix. The edges of the corresponding directed graph are superposed. (b) The permuted canonical matrix partitioned into 2 objects, one of which is in turn partitioned into 2 smaller, coupled objects.

of Computation in its *canonical* form (cMMC). The model was first introduced in its *imperative* form (iMMC) [5], and later in the canonical form [6]. The motivation for developing the cMMC was the occasional appearance in the iMMC of certain submatrices with extraordinary self-organizing properties. The iMMC is Turing-complete, in the sense that it can host a universal Turing machine [5]. General transformations are available to convert an iMMC into an equivalent cMMC ([6], §III). However, Turing completeness has not yet been proved for the cMMC. Rigorously speaking, and at least for now, the present work applies only to systems that can be represented by a cMMC model. Experience indicates that many types of systems can be written directly in cMMC format. For one thing, computer programs, which have been used to model virtually every known physical system, are easy to convert directly to cMMC.

A formal definition of the cMMC is available [6]. For our purposes here, it is better to explain the cMMC with the help of the motivational example of Section II and the 6×6 canonical matrix shown in Fig. 2, and repeated in Fig. 3(a). Let C be the canonical matrix. In the cMMC, set F is a set of functions. Let the functions be the following:

$$\begin{aligned}\alpha &: a \to p\\ \beta &: b \to q\\ \gamma &: c \to r\\ \delta &: p \times r \to s\\ \epsilon &: s \times d \to t\\ \varphi &: q \times e \to u,\end{aligned} \tag{7}$$

where all symbols on the right represent sets, and the functions are defined over the union $D \cup I$ of the set of domains $D = \{p, q, r, s, t, u)$ and the input set $I = \{a, b, c, d, e\}$. Symbols A and C can now be explained: for each function, the A's in matrix C represent arguments, the C's, codomains.

Matrix C only represents the functional dependencies among the functions of set F, but not the maps themselves.

This limited representation is sufficient for the set partition conjecture. There are cases, however, where an explicit representation of the maps is required. In such cases, an *expanded representation* should be used ([6], §II.A.2). The expanded representation is also sufficient for the set partition conjecture. In an expanded representation, the matrix remains canonical, but all functions are boolean and the matrix resembles a digital circuit. There are also cases where the functions are standard and have names, and it then becomes possible to keep track of them by simply assigning that name in correspondence with the rows of C. This is the case for the present example. Suppose the 6 functions are:

$$\begin{array}{rcl} \alpha & : & p = a \\ \beta & : & q = b \\ \gamma & : & r = c \\ \delta & : & s = p + r \\ \epsilon & : & t = d + s \\ \varphi & : & u = q + e, \end{array} \tag{8}$$

where the symbols now represent variables, and the function names are "=" and "+". These names are shown in Fig. 3(a) and (b) in correspondence with the rows of C.

Matrix C represents the host system, and the functions in set F represent the behavior of the guest system. Let $\widehat{D} = p \times \ldots \times u$ and $\widehat{I} = a \times \ldots \times e$. Then, the composite map $B : \widehat{I} \rightarrow \widehat{D}$, if it exists, is the *behavior* of the guest. But configuration K is not unique. A large matrix can have a very large set of configurations $\mathcal{K} = \{K\}$. Since the partial order ω of Eq. (3) is satisfied for every $K \in \mathcal{K}$, behavior B is the same for all K. Consequently, set $\mathcal{K}$ is a behavior-invariant search space for configurations that minimize the measure of Eq. (5). The process that searches $\mathcal{K}$ and finds the set of minimum-measure configurations $\widetilde{\mathcal{K}}$, or a statistically significant sample of configurations in $\widetilde{\mathcal{K}}$, is the dynamics of the host.

IV. Dynamics

Superposed on the matrices of Fig. 3(a) are the edges E of the directed graph $G = (V, E)$ ([4], §5.2). The vertices of G are the diagonal elements of C. To each vertex there corresponds a pair (function, codomain). Edges in E correspond to the off-diagonal elements in C, that is, to the A's. Graph G is a good description of the guest's behavior B because G is invariant under the host's dynamics, just like B itself is. A *search* in G is an operation where the vertices and edges are visited in a certain order. The rules for the search are as follows: (1) all *input* and *output* edges are excluded from the search, and all vertices and included edges are considered unvisited; (2) a vertex can be visited only when all its incoming edges are visited, and thus, the A's act as AND gates; (3) an edge can be visited when its source vertex is visited; and (4) the search ends when all vertices and included edges have been visited. G is assumed to be *connected*, meaning that at least one search exists. If vertices are visited one at a time, then each search finds one configuration, defined by the order in which the vertices have been visited.

The *length* $\ell_e(K)$ of an edge $e \in E$ is defined as the number of cells in C that e traverses. The length depends on the configuration K because the positions of the A's depend on K. The total length is the sum of the lengths of all edges visited by the search, that is, excluding input and output edges. Since edges correspond to the A's in C, the total length is, precisely, $L(K)$, as given by Eq. (5). The reason for the coefficient 2 in that equation has now been explained.

The *dynamics* of the host consists of multiple searches of G with random path selection. The distribution of probability slightly favors shorter paths. If edge selection during each search depends on the current length of each edge, then successive searches will find progressively shorter and shorter paths, and the searches will never return to configurations with long paths. This mechanism represents *dissipation*, where measure L is the quantity being dissipated. The dissipation mechanism is entirely controlled by the constraints, that is, by the A's, because ℓ_e is determined by the position of the corresponding A. The effect of selecting configurations with shorter paths is to bring the A's closer to the diagonal.

After a long time, the dynamics will find only configurations with the least value of L, that is, members of set $\widetilde{\mathcal{K}}$. These configurations are the *attractors* of the host's dynamics, and contain the segments that represent the hierarchy of objects in the guest system. By definition, an object is any submatrix of matrix C that remains invariant under the dynamics except for symmetric permutations of its internal rows and columns. Since any submatrix of C *encapsulates* a set of functions and their corresponding codomains, it can be figuratively said that an object *knows* how to calculate the codomains. It can also be said that the definition of object closely corresponds to the traditional definition used in OO analysis, where an object is defined as encapsulated behavior and attributes.

Since nothing has been done to cause the objects to form in the attractors, and their formation can not be justified by the guest's physics alone, the formation of objects should be considered as *emergent behavior*. The dissipative dynamics decodes the objects encoded in the constraints.

If edges represent actual connections in a physical system where information flows, then graph G describes the *flow* of information, the edges describe *flux lines*, and $L(K)$ is an indicator of the *resources* used by the system when it is in configuration K. This is strongly reminiscent of the neurons in the brain that are known to migrate and reconnect while trying to save resources by shortening their synaptic connections (*Hebbian learning*), and helps to explain why and how the brain creates objects. It also suggests that mathematical objects correspond to *physical* objects in the brain, the kind observed by functional MRI techniques, perhaps neural clique assemblies [7]. This suggestion may explain memory.

As mentioned above, the selection of measure $L(K)$ is arbitrary, provided it depends on K and has a minimum. The one-dimensional measure defined in Eq. (5) is the simplest choice, but if applied to the neurons, it would result in a one-dimensional brain. For a more realistic approach, a three-dimensional measure should be used in this case, such as the distance in Euclidean space. All the remaining considerations remain the same, except that the search in space $\mathcal{K}$ becomes more difficult.

Dynamics and behavior are two independent processes, and

they can run simultaneously without interfering with each other. This is clearly the case with the brain, and is also the case with host-guest dynamical systems.

The configuration shown in Fig. 3(a) is the same one given in Eq. (4), and it has $L = 18$. The dynamical process finds the four attractors listed in Eq. (6), all with $L = 10$, and the same objects and 3-level inheritance hierarchy as before. The hierarchy is shown in Fig. 3(b) for one of the configurations. The reader can verify that all sets define invariant submatrices, and that each submatrix is present in each one of the four configurations.

The strong dependence on initial conditions can be easily observed, even in such a small example. If equation $t = d + s$ is changed to $t = u + s$, then only two configurations $(\alpha, \gamma, \delta, \beta, \epsilon, \varphi)$ and $(\gamma, \alpha, \delta, \beta, \epsilon, \varphi)$ are found, both with $L_{\min} = 16$, there are only 2 levels in the hierarchy, and the objects are $\{\alpha, \gamma\}$, $\{\beta, \delta\}$, and $\{\epsilon, \varphi\}$ in the top level and the 6 functions in the bottom level. A small change in input has resulted in a completely different output. This effect is very well known to chaos researchers, as well as to code developers working in *refactoring*.

V. Physical characterization

It is now necessary to characterize the physical systems that exhibit or are likely to exhibit host-guest emergent behavior of the type described in the preceding Section. The characterization is obtained by close examination of the properties of the processes involved.

As explained above, the new host-guest systems are discrete, finite, self-organizing, dissipative, and deterministic. They also exhibit a very strong sensitivity to initial conditions. Since the systems are finite, the least-cost configurations can always be found by simple enumeration in a finite number of steps. However, when the matrix is very large, it is more convenient to use random techniques to find the attractor configurations. The measure function is very irregular in the space of configurations. It has many local minima that can trap any search algorithm. Random path-finding methods that can escape the traps have been tried and have performed consistently better than predictable techniques, such as, for example, steepest descent, even in problems of an order as small as 20 or 30. For all practical cases of interest, it can be ascertained that predictable techniques are not appropriate, and that random, unpredictable techniques are the best choice. Therefore, host-guest systems of interest should be considered as deterministic, but unpredictable. Since, in addition, host-guest systems are very sensitive to initial conditions, and even though they are not governed by a differential equation, they are also chaotic.

Another possibility, would be to treat very large systems as continuous, in the same way that a continuous model is used to treat a discrete fluid made of molecules, but this is not considered in the present paper.

Host-guest systems are also integrated, and therefore *indivisible*. Dividing a host-guest system into parts contradicts the fundamental assumption that configurations are selected by a random process, constrained by, and *only* by, the required invariance of the guest's behavior. Dividing the system, for any reason, would introduce an additional, artificial constraint, the effect of which would be to split the search space $\mathcal{K}$ into disconnected subspaces, leading to incorrect results. The integration of host-guest systems is confirmed by functional MRI, where it is well known that processes in the brain are not localized, but require the activation of many areas distributed over the entire brain. It is, however, permissible, to *unify* two or more host-guest systems, trained on different subjects, and obtain a larger integrated system, by allowing each of the original systems to train the large system, followed by a step of random integration for the large system to create its own objects. *Training* has been discussed in detail ([6], §IV).

There is one more important feature of the dynamics being examined that must be clearly identified. The A's in the lower triangle of matrix C play a triple role. For the guest system, they represent the arguments in the functions that describe its behavior. For the host system, they represent the constraints imposed on the host's dynamics by the required invariance of the guest's behavior. For the host-guest relationship, they represent the only communication signals. But the functions are the ones that represent the behavior, not the A's. The A's represent functional dependencies among the functions, that is, the *structure* of the behavior. The function of the A's is to *encode* the structure, transmit it to the host, and serve as constraints for the host's dynamics. When the host's otherwise random dynamics is constrained by the A's, that is, by the *existing* though still invisible structure, the structure is decoded and revealed as emergent behavior. The complete process can be viewed as one where *structure encoded in invariance constraints is decoded and revealed by the random dynamics of the host*.

The connection between constraints and structure has been repeatedly observed by many authors in the area of Complex Systems, such as [8]. But this is the first time that the connection has been put so precisely in perspective and discussed in such detail. The notion that structure is encoded in constraints and that a random dynamical process decodes it, and therefore the resulting structures depend on, and are determined by, the constraints, appears to be a very general principle in nature.

With the principle of structural encoding just identified providing an axiomatic foundation, the cMMC as a mathematical model (§III), the set partition conjecture (§II), the introduction and physical characterization of the new class of host-guest dynamical systems, the application of the new systems to the mathematical analysis of information processing in the brain (§VI), the prediction that chaos exists in all brains and is essential for all brain processes (§IX), the core implementation that allows for practical calculations of objects to be made (§VII), and the experimental verification (§VIII), this work can be proposed as a theory to explain a natural phenomenon that we constantly observe, the emergence of objects in the brain. The theory is the *Matrix Theory of Objects*, which was anticipated in a previous publication ([6], §II.F.5), and discussed at a workshop level [9, 10].

The physical characterization just made in this Section provides a powerful tool to search and recognize other natural systems that may have a similar dynamics. One example of such systems is provided by biological systems, where

chemical and physiological behavior must remain invariant, as objects, in this case organs or individuals of a species, are formed. The discipline of *host-guest chemistry* is concerned with the study of natural phenomena such as molecular recognition and molecular self-assembly. There may also be other natural systems that respond to the host-guest paradigm but where the distinction between host and guest is not as clearly defined as in the brain. Such research is beyond my scope, but authors who have observed the connection between constraints and structure are invited to examine the issue.

VI. Brain analysis

As explained at the beginning of the Introduction, the brain can be considered as a "black box" with an input and an output. In the same way, any functional part of the brain can also be considered as a black box with an input and output, where only the relationship between the input and output is of interest. This notion establishes the techniques discussed in this paper as an analytical tool for the mathematical analysis of information processing in the brain.

The notion of analysis implies that *synthesis* should also be possible. As discussed in Section V, host-guest systems are indivisible. If a system is analyzed into parts, it would not be correct to simply merge the parts together in order to synthesize the original system. Instead, the process of unification also mentioned in the same Section should be applied. Together, the two processes, analysis and synthesis, define an application of the "divide and conquer" technique to the study of host-guest systems.

Questions begin to arise right away. How small can those functional units be? An individual neuron? A neural clique [7]? An entire brain region? How can mathematical brain analysis be combined with experimental techniques, such as functional MRI? Can a sensory organ, such as the eye, which is not traditionally considered as part of the brain, still be analyzed in the same manner? Answers to these and many other questions are not discussed in this paper. However, some insight can be gained from consideration of the experiments of Section VIII.

All three experiments of Section VIII indicate that the notion of brain analysis is not just convenient, but necessary. The brain receives input in the form of bursts of synchronized *action potentials* transmitted to it by sensory nerves or afferent neurons. The observable output consists of objects that we can use to make decisions and take actions. However, the input for experiment 1 consists of a set of equations, the input for experiment 2 is a table representing points on a plane, and the input for experiment 3 is a computer program. If that kind of input is presented to a human analyst, the analyst's senses would convert the equations, the table, or the program into action potentials, and a great deal of processing would have to happen in the analyst's brain before the action potentials are again interpreted as equations, a table, or a program. This processing is excluded from the experiments.

VII. Implementation

Two core algorithms are needed to implement a computer simulation of a host-guest dynamical system: the Scope Constriction Algorithm (SCA) [11], and the Object Recognition Algorithm (ORA), which has not been published yet.

The purpose of SCA is to search the space $\mathcal{K}$ of configurations and to find a statistically significant sample of configurations with the least value of the length L, as discussed in Section IV. SCA does not actually search graph G. Instead, it operates *locally* by minimizing each ℓ_e individually. SCA applies the notion of commutativity to the functions in set F. Two functions, say $f, f' \in F$, are said to be *commutative* if they do not participate in the partial order π, or *non-commutative* if they do. If f and f' are commutative, they may appear in a configuration in any relative order. If a configuration, $K = (f_1, \ldots, f, f', \ldots, f_n)$ is available, where f and f' are adjacent, then an operation of *commutation* can be applied to f and f', resulting in the configuration $K' = (f_1, \ldots, f', f, \ldots, f_n)$. Now, since L depends on the configuration, it may well be that $L(K') < L(K)$.

Based on this property, SCA applies traditional cost minimization techniques. Starting from an arbitrary configuration, SCA randomly selects a function, and examines its near-neighbors in K. When it finds a commutation that would result in a lower value of L, it effects that commutation. After effecting the commutation, it either continues looking for other commutations of the same function, or selects another one, and continues until it can find no more L-reducing commutations. At that point, SCA tests for a local minimum by increasing L slightly, or may even try to find a maximum-L configuration, and repeats the procedure until some statistical requirement is met. Then, SCA randomly selects another arbitrary configuration, and starts all over again and tries to find other least-L configurations, and continues until it runs out of time. All these techniques are traditional and well known. There is no guarantee that SCA will find any least-L configurations, but it did in all cases within my limited experience. As usual for this type of work, a "sufficiently random" number generator is used to implement all random procedures.

Once a sample of configurations has been found, it is necessary to find the objects. That task is done by ORA. An object is a submatrix represented by a subset of functions that remains invariant under the statistics and is therefore *independent of initial conditions*. There are many such subsets and they define a *partition* of F, and therefore also of C. In the past, heuristic methods had been proposed [6] to find the objects. They work, but are difficult to program because of the many adjustable parameters involved. The only valid procedure to find objects is one that rigorously applies the definition of object, and ORA implements that procedure.

For experimental purposes only, an implementation of SCA and ORA in C++ was created, which includes several other support algorithms, input and output, and verification procedures. It served well for many experiments, but it is not production code.

VIII. Experimental verification

In this Section, several different sets of brain measurements are discussed and compared with theoretical predictions. The sets have been obtained from each one of the sources described

in the Introduction. All experiments have been previously published, and reference is made to the corresponding publications. In the present publication, the experiments serve as verification of the properties of the new host-guest systems.

In all cases, the same pattern is followed. The brain provides the measurements, the simulation, the theoretical results. To obtain the simulation, an initially untrained cMMC is trained by supervised learning ([6], §IV) with the problem description. The SCA algorithm, which simulates the random dissipative dynamics and was discussed in Section VII, is applied to find a statistically significant sample of attractors. The ORA algorithm, which recognizes objects based only on their invariance under the statistics and was also discussed in Section VII, is applied to find the objects and hierarchies in the attractors. Finally, the resulting objects are compared with those reported by a human analyst who uses the same input information.

A. Experiment 1

The example of Fig. 3(a) is actually one component of a previously published example [12], describing the motion of a particle with mass m under the action of a constant force $\mathbf{F}$ in 3D. Today, we use Newton's laws to describe the particle's motion. For example, for the x-component:

$$\begin{aligned} x &= x_0 + v_{x0}\tau + \frac{F_x}{2m}\tau^2 \\ v_x &= v_{x0} + \frac{F_x}{m}\tau. \end{aligned} \tag{9}$$

These equations, and the corresponding equations for the other two components, explicitly represent the following properties: (1) the motion can be described in terms of separate components, (2) the number of components is three, and (3) two independent variables must be used to describe each component, for example x and v_x for the first component. These three properties were not explicitly present in Kepler's laws and experimental measurements that Newton may have used to obtain his laws. Therefore, they must be considered as emergent behavior of Newton's brain.

The purpose of Experiment 1, is to simulate the same emergent behavior using an artificial host-guest system. To prove the simulation, the system is trained with information that describes the motion but does not contain any of the three properties. Since the motivational example involves only one component, that case is discussed first. In this case, property (3) is the only property that applies. To obtain the training information, Eq. (9) is re-written with the following substitutions: $a = v_{x0}\tau$, $b = v_{x0}$, $c = F_x\tau^2/2m$, $d = x_0$, $e = F_x\tau/m$, $t = x$, and $u = v_x$. where τ is the time. With these substitution, Eqs. (9) become Eqs. (8), where property (3) is not explicitly present. When the host-guest system is trained with Eqs. (8), the random dynamics generates and recognizes the objects shown in Fig. 3(b), where object (β, φ) knows how to calculate u, and object $(\alpha, \gamma, \delta, \epsilon)$ knows how to calculate t, that is, x and v_x, respectively. The two objects are clearly separated, indicating the presence of property (3). Within the calculation of x, object (α, γ) knows how to calculate the last 2 terms on the right of the x equation, and object (δ, ϵ) knows how to calculate their sum.

When all 3 components are included, as in the published example [12], Program P1 is the training information for the host-guest system. Program P1 does not contain any of the three properties. However, after application of the random dynamics, and as Fig. 3(b) in that publication indicates, all three properties are explicitly present. The system has been separated into 6 objects of 2 different classes. Two objects, one of each class, correspond to each one of the 3 components of motion. The two classes correspond to the two independent variables. There is full agreement between the theoretical predictions obtained by the host-guest simulation, and the experiment, that is, Newton's brain.

Today, anyone with a knowledge of elementary algebra can perform the same transformation in a few minutes. But that's not the point. The point is that the transformations can be performed by a purely random process *without* any knowledge of algebra or classical mechanics.

Following publication of [12], but without a citation, an algorithm appeared that discovers conservation laws from motion data by randomly searching a space of functions and minimizing some error metrics [13]. This work is heuristic, because no attempt is made to explain the underlining theory. It is also hybrid, because no clear distinction is made between the authors' brain emergent behavior and that of the simulator. It would be very interesting to know, for example, how the proposed key insight into identifying conservation laws computationally, can be identified computationally. It is a good example of the challenges that await those who want to automate science, but it should not be confused with the present work.

B. Experiment 2

Experiment 2 is based on a published example [11]. A set of points is given in a space. The number of points or the number of dimensions of the space are irrelevant. The points differ only by their positions, but the example would work the same if they differed by their color, shade, texture, or any combination thereof. In this case there are 167 points in a 2D space. If a plot is made, the eye can immediately recognize 3 well differentiated areas, and some more detailed structure within the areas. The number 3, the differentiation between the areas, and the more detailed structure, are emergent behavior in the observer's brain.

Four superposed grids of cells are used in order to simulate the continuity of the space. The original canonical matrix is of order 1433 and has $L = 2,803,702$. The attractors have $L = 51,517$. The 3 distinct areas are readily found in the attractors, at the top level, while deeper levels discern the details of the image. Again, there is full agreement, in this case with the human eye and brain.

This experiment is about image recognition. The points may represent the retina, the simulation may represent the first step of a recognition process in the brain, and the same domain-independent procedure used for all experiments applies. By contrast, predictable procedures used for artificial image recognition are highly domain-specific and have serious limitations, even after years of considerable efforts.

It is important to note that a person can not interpret the image if provided with the same information that was provided to the artificial host-guest system, that is, with a table of (x, y) coordinates of the 167 points. It would be necessary for that person to first plot the points on a graph, and then look at the plot. This is one area where artificial host-guest systems are clearly superior to the brain-eye combination.

C. Experiment 3

This experiment is in the area of OO software. The source code is a publicly available program in Java used for teaching at many European universities [14]. The analysis of the problem was also published ([6], §V). The analysis follows approximately the same steps as in the two previous examples. The classes and objects in the original Java code are considered as a measurement of the emergent behavior in the brains of the developers that developed the code. But the host-guest simulator has no knowledge of the Java code. Instead, the code is subject to a series of transformations, the purpose of which is to completely eliminate the emergent behavior, while keeping the basic functions that describe the behavior of the given system. First, the code was manually converted from Java to C in order to destroy all the original classes and objects. Then, the resulting C code was automatically converted to teacher's instructions, resulting in a set of 33 instructions. Finally, the instructions were randomly scrambled in order to destroy any remnants of the original partial order or of the original emergent behavior. These final instructions were used to train the simulator.

The resulting canonical matrix is of order 33 and has $L = 426$. The SCA algorithm found a sample of 596 attractors, all with $L_{\min} = 314$, and the ORA algorithm partitioned the sample into 19 objects at the top level, all of them belonging to just 4 classes. The classes and objects were finally compared with the classes and objects in the original Java code. The agreement is excellent, but they are not identical. The reason for them not being identical, is that the rules for writing OO code are empirical, and the solution is not unique. Different developers presented with the same problem will usually write somewhat different code.

Experiment 3 is about *refactoring* [15], the single most frequently used transformation in code development. It is also the riskiest, because of its empirical rules and high potential for human error, and has proved to be very difficult to automate. The present work offers a theoretical framework that can alleviate all these problems and allow the automation of refactoring.

A team of developers and an analyst working from the C code can create an object model and write code in Java probably in a few days of work, plus some more time for testing. The host-guest simulator can create the object model in minutes. Automatic conversion from cMMC object-oriented models to Java has not been developed yet, but once it is, the whole process from original C code to Java will take minutes, and testing will not be required because code generation is automatic. This is clearly another area where host-guest simulators can outperform humans.

D. In-vitro experiments

The dynamical process in all host-guest systems, including the brain, was theoretically predicted ([6] §II.F.5, and [10]) to be deterministic, dissipative, convergent to attractors with objects, and unpredictable. Chaos in the brain has long been considered by neuroscientists as critical for consciousness and memory [16], but has only recently been experimentally detected in neuronal networks [17] by means of experiments with *in vitro* cultures of live neurons and multi-electrode arrays designed to observe and measure spontaneous chaos in the networks. This experimental result is an important confirmation of the theoretical prediction.

While the existence of chaos itself is now confirmed, the classical nonlinear techniques used by nearly all researchers to measure it and interpret the results, are inadequate. These techniques were drawn from the theory of deterministic chaos, and apply to complex systems that obey nonlinear differential equations, but can not represent in sufficient detail the constraints created by the interactions among the individuals in the complex system. Since structure is encoded in the constraints, as proposed in this paper, the techniques can not adequately describe the structures, tell apart the various orthogonal processes that occur simultaneously in the brain, or answer the fundamental question why or how stable structures and a smooth behavior can emerge from the chaos. The application of differential equations to networks also requires the use of global information to decide local issues, something very unlikely to happen in real biological systems. An example is the application of network stability considerations based on the calculation of a Jacobian [8] to decide the admission or expulsion of an individual from the network. The extended use of nonlinear techniques may have led to the popular belief that brain function is "complicated", while in fact, it is actually quite simple.

By contrast, host-guest systems represent the constraints in full detail, clearly explain the decoding mechanism, identify the communications between host and guest, propose an answer to the fundamental question of smoothness, explain adaptation, and use only local information associated with each individual to make local decisions.

E. The physical characterization as an experiment

As a result of a conference on "Understanding Complex Systems", held in May 2007 at the University of Illinois at Urbana-Champaign, a general understanding of complex system dynamics was developed. The source for the understanding are multiple studies conducted in disciplines as diverse as physics, biology, sociology, ecology, economics, and others. The results, reported in [1], are of a general nature and not detailed, but represent an important experiment carried out by many researchers over many years of study, observation, and discussion.

The Urbana-Champaign Understanding can be directly compared with the physical characterization of host-guest systems discussed in Section V, which resulted from years of theoretical work by the author and considerable input from the field. Striking similarities arise when the comparison is

made. Complex systems cannot be modeled with traditional mathematical techniques. They are *open*, as reported in ([6], §IV). *Interactions* are non-trivial, and result in *constraints*. Constraints give rise to *coordinated* global behavior. Since there are no explicit rules or central director, the coordination is *self-organization*. *Structures* appear and new behaviors *emerge* at a larger *scale* than originally. The systems display *adaptation* and *evolve*. A *hierarchy* arises because the cycle repeats itself at progressively larger and larger scales. The agreement is excellent, but they missed indivisibility.

IX. Conclusions and Outlook

This work has introduced a new class of host-guest complex dynamical systems with extraordinary self-organizing properties. The new systems are physically characterized as being discrete, dissipative, deterministic, indivisible, very sensitive to initial conditions, and unpredictable.

A mathematical model has been proposed to explain the new systems. Careful analysis of the mathematics confirms the accepted point of view that emergent properties are not properties of individuals in a system, or sums of properties of individuals. Instead, when individuals come in proximity, they interact. The interactions determine their collective behavior and give rise to the emergence of structure as a *response* to interactions. Structure originates in, and is encoded in, the relationships among individuals, not in the individuals themselves, and can be decoded from there by a random dissipative process that leaves the collective behavior invariant. This structural encoding has been observed by many authors in many different systems, and is proposed in this paper as a general principle of nature, and also as the mechanism for neural encoding. Based on the principle, the prediction has been made that chaos exists in all brains, and is essential for all brain functions, including intelligence.

Theories of Physics originate from mathematical properties that remain invariant under certain transformations. In this work, the guest system's collective behavior is the property that remains invariant, the transformations are induced by the host dynamics, and the theory is the Matrix Theory of Objects. The foundation for the theory is the principle of structural encoding, and the model is the mathematical model proposed here. Objects calculated by the theory duplicate those created by the brain. Experimental verification has been discussed, and several large repositories of carefully documented brain experiments have been identified and proposed for further experimentation.

This work proposes a technique for brain analysis, where the brain, or any functional part of it, is considered as a black box with input and output. The black box is simulated by an artificial host-guest dynamical system, and the input/output function of the simulator is compared with that of the brain or the functional part.

Applications are many, but tools need to be developed before undertaking any. The property of indivisibility possessed by all host-guest systems and discussed in Section V, establishes them as a tool for the *unification* of systems in general. The cMMC model was previously proposed as a unifying tool for systems ([11], [6]). Software engineering and maintenance are areas that could benefit greatly from unification. Wikipedia could also benefit from unification [12]. Applications to Internet search engines are easy to imagine. Host-guest-based Internet search engines will be able to match objects, not just words. Image and voice recognition algorithms will be able to match natural objects, just as the brain does. Two areas where artificial host-guest simulators can outperform humans have been identified in this paper. An influence of the present work on several disciplines such as Physics, Neuroscience, Computer Science, and Artificial Intelligence is also to be expected.

Acknowledgement

To Dr. P. Thieberger of Brookhaven National Laboratory, for his constant interest and many lively discussions.

References

[1] M. Prokopenko, F. Boschetti, and A. J. Ryan, "An Information-Theoretic Primer on Complexity, Self-Organization, and Emergence." Complexity, vol. 15, 9. 11-28, (2009).

[2] D. Miner, M. Pickett, and M. desJardins, "Understanding the Brains Emergent Properties." Proc. Second Conference on Artificial General Intelligence, Arlington, VA (2009).

[3] S. Pissanetzky, "The matrix model of computation," *Proc. 12th SCI Conference*, Orlando, FL, 2008, vol. IV, pp. 184-189.

[4] S. Pissanetzky, *Sparse Matrix Technology*. London: Academic Press, 1984. Russian translation: Moscow: MIR, 1988. Electronic Edition (in English), 2008.

[5] S. Pissanetzky, "A relational virtual machine for program evolution," in *Proc. 2007 Int. Conf. on Software Engineering Research and Practice*, Las Vegas, vol. I, pp. 144-150. In this publication, the model was introduced with the name Relational Model of Computation, but was later renamed as the Matrix Model of Computation because of a name conflict.

[6] S. Pissanetzky, "A new universal model of computation and its contribution to learning, intelligence, parallelism, ontologies, refactoring, and the sharing of resources." Int. J. Computational Intelligence, vol. 5, nbr.2, pp.143-173, August 22, 2009. Available on-line: http://www.waset.org/journals/ijci/v5.php

[7] L. Lin, R. Osan, and J. Z. Tsien. "Organizing principles of real-time memory encoding: neural clique assemblies and universal neural codes." Trends in Neurosciences, Vol. 29, No. 1, pp. 48-57 (2006).

[8] J. Perotti *et al.* "Emergent Self-Organized Complex Network Topology out of Stability Constraints." Phys. Rev. Letters, **103**, 108701 (2009).

[9] S. Pissanetzky. "The New Theory of Objects and the Automatic Generation of Intelligent Agents." Workshop on Automation and Robotics. NASA Gilruth Center, Johnson Space Center, Houston, Texas. Sept. 25, 2009.

[10] S. Pissanetzky. "The Matrix Theory of Objects. An Update." AIAA Houston Section. Annual Technical Symposium 2010. NASA/JSC Gilruth Center. Houston, Texas. April 30, 2010.

[11] S. Pissanetzky, "A new type of structured artificial neural networks based on the matrix model of computation," *Proc. 2008 Int. Conf. on Artificial Intelligence*, Las Vegas, vol. I, pp. 251-257.

[12] S. Pissanetzky, "Applications of the matrix model of computation." *Proc. 12th SCI Conference*, Orlando, FL, 2008, vol. IV, pp. 190-195.

[13] M. Schmidt and H. Lipson, "Distilling free-form natural laws from experimental data." Science, **324**, pp. 81-85 (April 2009).

[14] S. Demeyer *et al.*, "The LAN-simulation: a refactoring teaching example," *Proc. 8th. Int. Workshop on Principles of Software. Evolution*, Lisbon, 2005, pp. 123-134.

[15] W. F. Opdyke, "Refactoring object-oriented frameworks," Ph.D. thesis, Dep. Comp. Sc., Univ. of Illinois, Urbana-Champaign, 1992.

[16] H. Korn and P. Faure, "Is there chaos in the brain? II. Experimental evidence and related models." Comptes Rendus Bilogies, **326**, pp.787-840 (2003).

[17] W. Chen, X. Li, J. Pu, and Q. Luo, "Spatial-temporal dynamics of chaotic behavior in cultured hippocampal networks." Phys. Rev. E, **81**, 061903 (2010).