Aggregability is NP-Hard

Vladik Kreinovich[1] and Max Shpak[2*]

Departments of [1]Computer Science and [2]Biological Sciences
University of Texas at El Paso, El Paso, TX 79968, USA
emails vladik@utep.edu, mshpak@utep.edu

Abstract

Many dynamical systems are *aggregable* in the sense that we can divide their variables $x_1, \ldots, x_n$ into several (k) non-intersecting groups and find combinations $y_1, \ldots, y_k$ of variables from these groups (*macrovariables*) whose dynamics depend only on the initial values of the macrovariables. For very large systems, finding such an aggregation is often the only way to perform a meaningful analysis of such systems. Since aggregation is important, researchers have been trying to find a general efficient algorithm for detecting aggregability. In this paper, we show that in general, detecting aggregability is NP-hard even for linear systems, and thus (unless P=NP), we can only hope to find efficient detection algorithms for specific classes of systems.

We also show that in the linear case, once the groups are known, it is possible to efficiently find appropriate combinations y_a.

What is aggregability. Many systems in nature can be described as *dynamical systems*, in which the state of a system at each moment of time is characterized by the values of (finitely many) variables $x_1, \ldots, x_n$, and the change of the state over time is described by an equation $x_i' = f_i(x_1, \ldots, x_n)$, where

- for continuous-time systems, in which the time t can take any real value, x_i' is the first time derivative of x_i:
$$\frac{dx_i}{dt} = f_i(x_1, \ldots, x_n); \tag{1a}$$
- for discrete-time systems, in which the time t can only take integer values, x_i' is the value of x_i at the next moment of time:
$$x_i(t+1) = f_i(x_1(t), \ldots, x_n(t)). \tag{1b}$$

For example, the state of a biological population can be described by listing the frequencies x_i of different genotypes i; in this example, the corresponding functions $f_i(x_1, \ldots, x_n)$ describe the effects of mutation, recombination, and natural selection.

For natural systems, the number of variables is often very large. For example, for a system with g loci on a chromosome in which each of these genes can have two possible allelic states, there are $n = 2^g$ possible genotypes. For large g, dues to the large number of state variables, the corresponding dynamics is extremely difficult to analyze.

Many biological systems (and in many systems from other fields such as economics [11] and queuing theory [1] etc.) are *aggregable* in the following sense: variables $x_1, \ldots, x_n$ can be divided into groups $I_1, \ldots, I_k$ ($\cup I_a = \{1, \ldots, n\}$ and $I_a \cap I_b = \emptyset$ for $a \neq b$) so that for appropriate combinations $y_1, \ldots, y_k$ of variables within each groups, equations (1a) or (1b) lead to simpler equations

$$\frac{dy_i}{dt} = h_i(y_1, \ldots, y_k) \tag{2a}$$

or, correspondingly,

$$y_i(t+1) = h_i(y_1(t), \ldots, y_k(t)) \tag{2b}$$

for appropriate functions $h_1, \ldots, h_k$. The corresponding combinations $y_a = c_a(x_{i_1}, \ldots, x_{i_p})$, where $i_1, \ldots, i_p \in I_a$ (and c_a are functions from real numbers to real numbers) are called *macrovariables*.

For example, if we have g gene loci such that allelic substitutions at each locus have identical effects, then instead of $n = 2^g$ original variables $x_{0\ldots0}, x_{0\ldots1}, \ldots, x_{1\ldots1}$, we can consider $k = g + 1$ ($\ll 2^g$) macrovariables $y_1 = x_{0\ldots0}$ (the frequency of the original state), $y_2 = x_{0\ldots01} + x_{0\ldots010} + \ldots + x_{1\ldots010\cdot0} + \ldots + x_{10\ldots0}$ (the frequency of states with a single mutation), y_3 – overall frequency of states with 2 mutations, $\ldots$, $y_{g+1} = x_{1\ldots1}$ [8, 9, 12].

Detecting aggregability is important. In many actual problems, we know how to subdivide the variables into groups. For example, if we know the functions of all the genes, it is natural to group together all the genes with similar functions.

In many other situations, however, we only know the equations (1a) or (1b) (we may know these equation from the the analysis of the empirical data), but we do not yet know how to properly divide and combine the variables. For example, one may not know the functional role or epistatic interactions of genes on a chromosome a priori, only the phenotypes or Darwinian fitnesses of different genotypes. In such situations, it is important to be able to detect whether an aggregation is possible – and, if possible, to find such an aggregation. The aggregation itself may be instructive as to the function and interaction of genes, and may inform one as to which system components are relevant. For a detailed discussion see, e.g., [1, 3, 4, 5, 6, 7, 9, 10, 11].

Usually, we have some *partial* information about the variables – e.g., we may know that a certain variable x_i should affect one of the combinations y_a. In such situations, it is desirable to find the groups which are consistent with this partial information.

What we do in this paper. For some systems, there exist efficient techniques that detect aggregability and find the corresponding aggregations. Since it is important to detect aggregability, researchers have been trying to find a *general* efficient method for its detection.

In this paper, we show that even in the simplest case when the system is *linear* (i.e., all the dependencies f_i are linear), the number of classes is $k = 2$, and the additional information consists of a single variable that has to be involved in one of the combinations y_a, the problem of detecting aggregability is NP-hard. This means that even for linear systems (unless P=NP), there is no hope of finding a *general* method for detecting aggregability; we should therefore concentrate our efforts on detecting aggregability for *specific* classes of dynamic systems.

Let us formulate our result in precise terms.

Definition 1 *Let n be a given integer.*

- *By a* linear system*, we mean an $n \times n$ rational-valued matrix $c_{i,j}$, $1 \le i \le n$, $1 \le j \le n$.*
- *By the* additional information*, we mean an integer i_0 such that $1 \le i_0 \le n$.*
- *We say that a linear system with the additional information is* 2-aggregable *if there exist non-empty sets I_1 and I_2 such that $i_0 \in I_1$, $I_1 \cup I_2 = \{1, \ldots, n\}$, $I_1 \cap I_2 = \emptyset$, and the values $w_1, \ldots, w_n$ and $h_{a,b}$ ($a = 1, 2$, $b = 1, 2$) such that $w_{i_0} \ne 0$, and for every $x_1, \ldots, x_n$, $x'_i = \sum\limits_{j=1}^{n} c_{i,j} \cdot x_j$ implies that $y'_a = \sum\limits_{b=1}^{2} h_{a,b} \cdot y_b$, where*

$$y_a \stackrel{\text{def}}{=} \sum_{i \in I_a} w_i \cdot x_i, \quad y'_a \stackrel{\text{def}}{=} \sum_{i \in I_a} w_i \cdot x'_i.$$

Theorem 1 *Detecting 2-aggregability is NP-hard.*

Proof. To prove NP-hardness of 2-aggregability, we will reduce, to this new problem, a known NP-hard *subset problem* (see, e.g., [2]). The subset problem is as follows: given n positive integers $s_1, \ldots, s_m$ and an integer s_0, whether there exists a subset $I \subseteq \{1, \ldots, m\}$ such that $\sum\limits_{i \in I} s_i = s_0$.

For every instance of the subset problem, we take $s_{m+1} \stackrel{\text{def}}{=} -s_0$, $n = m + 2$, $i_0 = m + 2$, and we form the following linear system:

- for $1 \le i \le m + 1$, we take $c_{i,i} = 1$, $c_{i,n+1} = s_i$, and $c_{i,j} = 0$ for all other j;
- for $i = m + 2$, we take $c_{m+2,m+2} = 1 + \beta$, where $\beta \stackrel{\text{def}}{=} 1 + s_0 - \sum\limits_{i=1}^{m} s_i$, and $c_{m+2,i} = 1$ for all other i.

Let us prove that this system is 2-aggregable if and only if the original instance of the subset problem has a solution.

"If" part. If the original instance has a solution I, with $CI \stackrel{\text{def}}{=} \{1, \ldots, m\} - I$, then we take $I_1 = CI \cup \{m+2\}$, $I_2 = I \cup \{m+1\}$, $w_i = 1$ for all i, $h_{1,1} = 2$, $h_{1,2} = h_{2,2} = 1$, and $h_{2,1} = 0$. Then, we should have $y'_1 = 2y_1 + y_2$ and $y'_2 = y_2$.

Indeed, the fact that I is a solution means that $\sum\limits_{i \in I} s_i = s_0$. For our choice of weights w_i, we get $y_1 = \sum\limits_{i \in CI} x_i + x_{m+2}$ and $y_2 = \sum\limits_{i \in I} x_i + x_{m+1}$. For y'_2, we get

$$y'_2 = \sum_{i \in I} x'_i + x'_{m+1} = \sum_{i \in CI} x_i + x_{m+1} + \left(\sum_{i \in I} -s_0 \right) \cdot x_{m+2}.$$

Since $\sum\limits_{i \in I} s_i = s_0$, we conclude that $y'_2 = y_2$.

Similarly,

$$y'_1 = \sum_{i \in CI} x'_i + x'_{m+2}.$$

Describing the sum $\sum\limits_{i=1}^{m+2} x_i$ in the expression for x'_{m+2} as the sum of the values from I, CI, $m+1$, and $m+2$, we conclude that

$$y'_1 = \sum_{i \in CI} x_i + \left(\sum_{i \in CI} s_i \right) \cdot x_{m+2} + \sum_{i \in CI} x_i + \sum_{i \in I} x_i + x_{m+1} + x_{m+2} + \beta \cdot x_{m+2} =$$

$$\left(\sum_{i \in I} x_i + x_{m+1} \right) + 2 \cdot \sum_{i \in CI} x_i + \left(\sum_{i \in CI} s_i + 1 + \beta \right) \cdot x_{m+2}.$$

Since $\sum\limits_{i \in I} s_i = s_0$, we have $\sum\limits_{i \in CI} s_i = \sum\limits_{i=1}^{m} s_i - \sum\limits_{i \in I} s_i = \sum\limits_{i=1}^{m} s_i - s_0$. Due to our choice of β, we thus have $\sum\limits_{i \in CI} s_i + 1 + \beta = 2$, hence $y'_1 = 2y_1 + y_2$.

"Only if" part. Conversely, let us assume that the system is 2-aggregable, i.e., that there exist sets $I_1 \ni m+2$ and I_2, and the values w_i and $h_{a,b}$ for which all the above conditions are satisfied. In other words, $w_{m+2} \neq 0$, and from the equations

$$x'_i = x_i + s_i \cdot x_{m+2}, \quad 1 \leq i \leq m+1, \tag{3}$$

$$x'_{m+2} = \sum_{i=1}^{m+2} x_i + \beta \cdot x_{m+2}, \tag{4}$$

we should be able to conclude that for

$$y_1 = \sum_{i \in I_1} w_i \cdot x_i, \quad y_2 = \sum_{i \in I_2} w_i \cdot x_i, \tag{5}$$

$$y'_1 = \sum_{i \in I_1} w_i \cdot x'_i, \quad y'_2 = \sum_{i \in I_2} w_i \cdot x'_i, \tag{6}$$

we have

$$y'_1 = h_{1,1} \cdot y_1 + h_{1,2} \cdot y_2 \tag{7}$$

and

$$y_2' = h_{2,1} \cdot y_1 + h_{2,2} \cdot y_2. \tag{8}$$

Let us denote $I_1' \stackrel{\text{def}}{=} I_1 - \{m+2\}$. From (3), (4), and (6), we conclude that

$$y_1' = \sum_{i \in I_1'} w_i \cdot x_i + \left(\sum_{i \in I_1'} w_i \cdot s_i \right) \cdot x_{m+2} + w_{m+2} \cdot \sum_{i=1}^{m+2} x_i + w_{m+2} \cdot \beta \cdot x_{m+2}.$$

Thus, the equation (7) takes the form

$$\sum_{i \in I_1'} w_i \cdot x_i + \left(\sum_{i \in I_1'} w_i \cdot s_i \right) \cdot x_{m+2} + w_{m+2} \cdot \sum_{i=1}^{m+2} x_i + w_{m+2} \cdot \beta \cdot x_{m+2} =$$

$$h_{1,1} \cdot \left(\sum_{i \in I_1'} w_i \cdot x_i + w_{m+2} \cdot x_{m+2} \right) + h_{1,2} \cdot \sum_{i \in I_2} w_i \cdot x_i. \tag{9}$$

Since this equality must hold for all possible values of x_i, for each i, the coefficient at x_i in the left-hand side of (9) must be equal to the coefficient at x_i in the right-hand side of (9).

In particular, for $i \in I_1'$, we conclude that $w_i + w_{m+2} = h_{1,1} \cdot w_i$, i.e., that $(h_{1,1} - 1) \cdot w_i = w_{m+2}$. Since $w_{m+2} \neq 0$, we conclude that $w_i \neq 0$ and $h_{1,1} - 1 \neq 0$, hence $w_i = w_{m+2}/(h_{1,1} - 1)$ for all such i – i.e., all the values w_i, $i \in I_1'$ are equal to each other. Let us denote the common value of these w_i by $w^{(1)} \neq 0$.

For $i \in I_2$, we similarly conclude that $w_{m+2} = h_{1,2} \cdot w_i$. Since $w_{m+2} \neq 0$, we similarly conclude that $w_i \neq 0$, and that $w_i = w_{m+2}/h_{1,2}$ is the same for all $i \in I_2$. Let us denote the common value of these w_i by $w^{(2)} \neq 0$. Thus, the formulas (5)–(6) take the following form:

$$y_1 = w^{(1)} \cdot \sum_{i \in I_1'} x_i + w_{m+2} \cdot x_{m+2}, \quad y_2 = w^{(2)} \cdot \sum_{i \in I_2} x_i, \tag{5a}$$

$$y_1' = w^{(1)} \cdot \sum_{i \in I_1'} x_i' + w_{m+2} \cdot x_{m+2}', \quad y_2' = w^{(2)} \cdot \sum_{i \in I_2} x_i'. \tag{6a}$$

By using the expressions (3)–(5) and (6a), we conclude that y_2' takes the form

$$y_2' = w^{(2)} \cdot \sum_{i \in I_2} x_i + w^{(2)} \cdot \left(\sum_{i \in I_2} s_i \right) \cdot x_{m+2}.$$

Thus, the equation (8) takes the form

$$w^{(2)} \cdot \sum_{i \in I_2} x_i + w^{(2)} \cdot \left(\sum_{i \in I_2} s_i \right) \cdot x_{m+2} =$$

$$h_{2,1} \cdot \left(w^{(1)} \cdot \sum_{i \in I_1'} x_i + w_{m+2} \cdot x_{m+2} \right) + h_{2,2} \cdot w^{(2)} \cdot \sum_{i \in I_2} x_i. \tag{10}$$

For $i \in I_1'$, by equating coefficients at x_i in both sides of (10), we conclude that $h_{2,1} \cdot w^{(1)} = 0$. Since $w^{(1)} \neq 0$, we thus conclude that $h_{2,1} = 0$. By comparing coefficients at x_{m+2}, we now conclude that $w^{(2)} \cdot \sum\limits_{i \in I_2} s_i = 0$. Since $w^{(2)} \neq 0$, we thus conclude that $\sum\limits_{i \in I_2} s_i = 0$. Since all the values s_i are positive except for $s_{m+1} = -s_0$, the only possibility to have $\sum\limits_{i \in I_2} = 0$ is when $m+1 \in I_2$. In this case, for $I \stackrel{\text{def}}{=} I_2 - \{m+1\}$, we get $\sum\limits_{i \in I} s_i + (-s_0) = 0$, i.e., $\sum\limits_{i \in I} s_i = s_0$. So, the original instance of the subset problem has a solution.

This completes the proof of the theorem.

Once we find the partition, finding the combinations is feasible. We have shown that finding the partition is NP-hard. Our original problem was not only to find a partition, but also to find the appropriate combinations y_a. Let us show that once the partition is found, finding the weights of the corresponding combinations y_a is easy.

Indeed, in matrix terms, the original dynamic equation has the form $x' = cx$. Once the partition $I_1, \ldots, I_k$ is fixed, we can represent each n-dimensional state vector x as a sum $x = \sum\limits_a x^{(a)}$ of vectors $x^{(a)}$ formed by the components x_i, $i \in I_a$. In these terms, the equation $x' = cx$ can be represented as $x'^{(a)} = \sum\limits_b c^{(a),(b)} x^{(b)}$, where $c^{(a),(b)}$ denotes the corresponding block of the matrix c (formed by elements $c_{i,j}$ with $i \in I_a$ and $j \in I_b$). For the corresponding linear combinations $y_a = w^{(a)\,T} x^{(a)}$, the dynamics takes the form $y_a' = \sum\limits_b w^{(a)\,T} c^{(a),(b)} x^{(b)}$. The only possibility for this expression to only depend on the combinations $y_b = w^{(b)\,T} x^{(b)}$ is when for each b, the coefficients of the dependence of y_a' on x_i, $i \in I_b$, are proportional to the corresponding weights w_i, i.e., when for every a and b, we have $w^{(a)\,T} c^{(a),(b)} = \lambda_{a,b} w^{(b)\,T}$ for some number $\lambda_{a,b}$. By transposing this relation, we conclude that

$$c^{(a),(b)\,T} w^{(a)} = \lambda_{a,b} w^{(b)}. \tag{11}$$

In particular, for $a = b$, we conclude that $w^{(a)}$ is an eigenvector of the matrix $c^{(a),(a)\,T}$. Since the weight vectors $w^{(a)}$ are defined modulo a scalar factor, we can thus select one of the (easy-to-compute) eigenvectors of $c^{(a),(a)\,T}$ as $w^{(a)}$.

Once we know $w^{(a)}$ for one a, we can determine all other weight vectors $w^{(b)}$ from the condition (11), i.e., as $w^{(b)} = c^{(a),(b)\,T} w^{(a)}$.

Comment. The computational complexity of the problem comes from the fact that the sets I_a should be non-overlapping. If we allow overlapping sets, then we can easily find the corresponding combinations y_k, e.g., as the coordinates $y_a = \sum x_i \cdot e_i^{(a)}$ of the vector $x = (x_1, \ldots, x_n)$ in the basis formed by the eigenvectors $e^{(a)}$ of the matrix $c_{i,j}$.

Conclusion. This result shows, crudely speaking, that no general efficient algorithm is possible for detecting aggregable systems – unless we have some additional a priori information about the system. Thus, to efficiently aggregate a system, we must, in general, specify some additional information about the grouping of the variables. Otherwise, we may not be able to find the desired aggregation faster than by testing all possible subsets – i.e., faster than the exhaustive search.

Note that this result is a consequence of the type of aggregation – with non-overlapping sets of variables; for overlapping sets, aggregations can be feasibly determined.

Acknowledgments. This work was supported in part by NASA under cooperative agreement NCC5-209, NSF grants EAR-0225670 and DMS-0532645, and by Texas Department of Transportation grant No. 0-5453.

The authors are thankful to Pierre-Jacques Courtois and Michael Vose for useful ideas, references, and suggestions.

References

[1] P. J. Courtois. *Decomposability: Queueing and Computer System Applications*. Academic Press, New York, 1977.

[2] M. R. Garey and D. S. Johnson. *Computers and Intractability, a Guide to the Theory of NP-Completeness*. W. H. Freemand and Company, San Francisco, California, 1979.

[3] Y. Iwasa, V. Andreasen, and S. A. Levin. Aggregation in model ecosystems. I. Perfect aggregation. *Ecological Modelling*, 37:287–302, 1987.

[4] Y. Iwasa, S. A. Levin, and V. Andreasen. Aggregation in model ecosystems. II. Approximate aggregation. *IMA Journal of Mathematics Applied in Medicine and Biology*, 6:1–23, 1989.

[5] C. C. J. Moey and J. E. Rowe. Population aggregation based on fitness. *Natural Computing* 3(1):5–19.

[6] J. E. Rowe, M. D. Vose, and A. H. Wright. Coarse Graining Selection and Mutation. *Proceedings of the 8th International Workshop on Foundations of Genetic Algorithms FOGA'2005, Aizu-Wakamatsu City, Japan, January 5–9, 2005*, Springer Lecture Notes in Computer Science, Vol. 3469, pages 176–191, 2005.

[7] J. E. Rowe, M. D. Vose, and A. H. Wright. State aggregation and population dynamics in linear systems. *Artificial Life* 11(4):473–492, 2005.

[8] P. Schuster and J. Swetina. Stationary mutant distributions and evolutionary optimization. *Bulletin of Mathematical Biology* 50:635–650, 1988.

[9] M. Shpak, P. F. Stadler, G. P. Wagner, and J. Hermisson. Aggregation of variables and system decomposition: application to fitness landscape analysis. *Theory in Biosciences* 123: 33–68, 2004.

[10] M. Shpak, P. F. Stadler, G. P. Wagner, and L. Altenberg. Simon-Ando decomposability and mutation-selection dynamics. *Theory in Biosciences* 123: 139–180, 2004.

[11] H. Simon and J. Ando. Aggregation of variables in dynamical systems. *Econometrica* 29:111–138, 1961.

[12] P. P. Stadler and G. Tinhofer. Equitable partitions, coherent algenras, and random walks: applications to the correlation structure of landscapes. *MATCH* 40:215–261, 2000.