Mutation Rules and the Evolution of Sparseness and Modularity in Biological Systems

Tamar Friedlander[1], Avraham E. Mayo[1], Tsvi Tlusty[2,3], Uri Alon[1]*

[1] Department of Molecular Cell Biology, Weizmann Institute of Science, Rehovot, Israel, [2] Department of Physics of Complex Systems, Weizmann Institute of Science, Rehovot, Israel, [3] Simons Center for Systems Biology, Institute for Advanced Study, Princeton, New Jersey, United States of America

Abstract

Biological systems exhibit two structural features on many levels of organization: sparseness, in which only a small fraction of possible interactions between components actually occur; and modularity – the near decomposability of the system into modules with distinct functionality. Recent work suggests that modularity can evolve in a variety of circumstances, including goals that vary in time such that they share the same subgoals (modularly varying goals), or when connections are costly. Here, we studied the origin of modularity and sparseness focusing on the nature of the mutation process, rather than on connection cost or variations in the goal. We use simulations of evolution with different mutation rules. We found that commonly used sum-rule mutations, in which interactions are mutated by adding random numbers, do not lead to modularity or sparseness except for in special situations. In contrast, product-rule mutations in which interactions are mutated by multiplying by random numbers – a better model for the effects of biological mutations – led to sparseness naturally. When the goals of evolution are modular, in the sense that specific groups of inputs affect specific groups of outputs, product-rule mutations also lead to modular structure; sum-rule mutations do not. Product-rule mutations generate sparseness and modularity because they tend to reduce interactions, and to keep small interaction terms small.

Citation: Friedlander T, Mayo AE, Tlusty T, Alon U (2013) Mutation Rules and the Evolution of Sparseness and Modularity in Biological Systems. PLoS ONE 8(8): e70444. doi:10.1371/journal.pone.0070444

Editor: John Parkinson, Hospital for Sick Children, Canada

Received March 22, 2013; **Accepted** June 18, 2013; **Published** August 6, 2013

Funding: The research leading to these results has received funding from the Israel Science Foundation (http://www.isf.org.il/english/) and the European Research Council under the European Union's Seventh Framework Programme (FP7/2007–2013)/ERC Grant agreement n° 249919 (http://erc.europa.eu/funding-and-grants). The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Competing Interests: The authors have declared that no competing interests exist.

* E-mail: uri.alon@weizmann.ac.il

Introduction

Biological systems show certain structural features on many levels of organization. Two such features are sparseness and modularity [1–10]. Sparseness means that most possible interactions between pairs of components are not found. For example, less than 1% of the possible interactions are found in gene regulation networks of bacteria and yeast [11]. The second feature, modularity, is the near-decomposability of a system into modules - sets of components with many strong interactions within the set, and few significant interactions with other sets. Each module typically performs a specific biological function. Modularity is found for example in protein structure (functional domains) [12], in regulatory networks (gene modules, network motifs), and in body plans (organs, systems) - for reviews see [2,8,10,13]. While modular networks are essentially sparse – sparse networks are not necessarily modular. Even if interactions are few, they could be evenly distributed and therefore not form modules.

Computer simulations of evolution are used to understand the origin of these structural features. The simulations begin with a set of structures, the elements of the structures are mutated, the fitness of each structure is evaluated according to a given goal, and then the structures with the highest fitness are selected. The most commonly used form of mutation in these simulations is the sum-rule mutation: adding a random number to the value of each element. Such simulations typically find optimal structures which satisfy the goal. However, they generally do not yield modular or sparse structures. Even when starting with a modular solution the simulations typically drift towards non-modular solutions, which are usually much more prevalent and are sometimes better at performing given the goal [14]. This leaves open the question of how and why sparseness and modularity evolve in biology.

Several studies have addressed this question by employing different approaches. For example, neutral models suggest that duplicating parts of a network can increase its modularity ("duplication-differentiation" model [15]) or similarly that mutation, duplication and genetic drift [16] can lead to modularity. Modularity in metabolic networks was suggested to arise from a neutral growth process [17,18]. On the other hand, other studies suggest that modularity can be selected for, either indirectly or directly. Modularity has been suggested to be beneficial because it provides dynamical stability or robustness to recombination [19], improves the ability to accommodate beneficial foreign DNA [20], breaks developmental constraints [21], evolves due to selection for environmental robustness [22,23] or because the same network supports multiple expression patterns [24]. Horizontal gene transfer, together with selection for novelty can lead to modularity in the polyketide synthase system [25]. It was recently suggested by Clune, Mouret and Lipson that network sparseness and modularity can evolve due to selection to minimize connection costs, as is thought to occur for example in neuron networks [26]. Kashtan *et al.* [27–29] found that when goals change with time, such that goals are made of the same set of subgoals in different combinations - a situation termed modularly varying goals (MVG) - the system can evolve modular structure. Each module in the evolved structure solves one of the subgoals, and modules are quickly rewired when the goal changes. Modularly varying goals tested in several model systems, with sum-rule mutations used when applicable [14], showed modularity under a range of parameters. Modularly varying goals also speed up evolution relative to unchanging goals [30], a phenomenon evaluated using analytically solvable models [14]. Due to the importance of sparseness and modularity in biology, it is of interest to see if additional mechanisms for their evolution exist. In particular, though attention has been given to the goals and cost functions, little attention has been given to the type of mutation rule used.

Here, we address the role of the mutation rule on the evolution of modularity and sparseness. Most studies that use simulations to study evolution employ a simple rule to specify how mutations change the parameters in the structure that is evolved - namely the 'sum rule', in which a parameter is mutated by adding a random number drawn from a specified distribution. Here, we note that this sum rule is usually not a good description of the effect of cumulative genetic mutations on a given biological parameter. Instead, the effects of mutations are better approximated by product-rule processes. For example, the effect of cumulative mutations on an enzyme's activity is found to be multiplicative [31]. Similarly, the effect of mutation on binding of proteins to DNA [32,33] and proteins to proteins [34–36] is thought to be multiplicative to a first approximation, such that the change in affinity caused by several genetic mutations is approximately the product of the effects of each mutation.

One fundamental reason for the use of product rule to describe the effect of genetic mutations is that mutations affect molecular interactions such as hydrogen bonds. This affects the free energy in an approximately additive way, assuming that the different molecular interactions are independent to a first approximation. Since affinity and reaction rate are exponential in free energy, the effects of cumulative genetic mutations on these parameters are approximately multiplicative. Note that in population genetics, there are different meanings to 'additive' and 'multiplicative' mutations [37], and thus we chose the terms 'sum-rule' and 'product-rule' to avoid confusion.

A related feature of mutations is that they more often reduce the absolute strength of the interaction or activity parameter than increase it [38–40]. This asymmetry can be captured using product-rule mutations: for example, multiplying by a random number normally distributed $N(1,\sigma)$ gives equal probability to multiply by 0.5 or 1.5, which tends to reduce the absolute size of the element; in order to revert a 0.5-mutation, one needs to multiply by a 2-mutation, which is less likely to occur.

To study the role of product-rule mutations, we compare evolution of simple and widely used model structures under sum-rule and product-rule mutations in computer evolution simulations. This is of interest because most simulations of evolution use sum-rules for mutations. We found that product-rule mutations lead to evolution of sparseness without compromising fitness. This relates to the study of Burda *et al.* which used a mutation rule that is approximately product-rule [41]. In contrast, we found that sum-rule mutations only lead to sparseness under special conditions, such as when the model parameters are constrained to be non-negative. Furthermore, when the goal is modular, we found that product-rule mutations led to modular structures, whereas sum-rule mutations generally do not. Unlike Kashtan *et al.*, [14,27,28] here modularity arises from modular goals without need to change goals over time, and when there is no cost for connections. We study the speed and scaling laws of this process. The basic reason that product-rule mutations lead to sparseness and modularity is that they tend to reduce interaction terms and to keep small interaction terms small and thus cause the evolutionary dynamics to approach structures that have optimal fitness with minimal number of interactions. When goals are modular, this effect, in turn, leads to modular structure.

Results

A simple Matrix-multiplication Model of Transcription Networks

To study the effect of the mutation rule on evolved structures, we use a standard evolutionary simulation framework [42,43]. Briefly, the evolutionary simulation starts with a population of N structures, duplicates them, and mutates each structure with some probability according to a mutation rule (the mutation rules described below will be our main focus). Fitness is evaluated for each structure in comparison to a goal. The fittest individuals are selected by a selection criterion, and the process is repeated, until high fitness evolves (Fig. 1A).

We consider, for simplicity, structures described by continuous-valued matrices. These serve as simple models for biological interactions, where the elements of the matrix A_{ij} are the interaction strengths between components i and j in the system. Evolution entails varying the matrix elements to reach defined goals. Linear matrix models have a long history in modeling of biological systems [41,44–49]. Use of a matrix to describe gene expression is a standard approach. Several studies use matrices to reverse-engineer the underlying network [50]. Matrix models have also been used to understand developmental gene regulation, as in the pioneering work of Reinitz in Drosophila [51–53]; matrix models were recently used by De-Pace *et al.* to relate the strengths of regulation to the level of gene expression across fruit fly species using detailed gene expression measurements [54].

In the field of modularity, matrix models have been extensively used. Matrix models were used in the pioneering work of Lipson *et al.* [55] and also Wagner *et al.* [24]. We previously used a matrix model to analytically study a different route to modularity [14]. We evolved the matrix A to satisfy the goal $Au=v$, where u and v are vectors. The fitness is the distance to the goal, $F=-\|Au-v\|$ where $\|.\|$ denotes sum of squares of elements (related to Fisher's geometric model [56]).

Often, biological systems have multiple layers [57] where components in one level – e.g. receptors, send signals to components in the next level, e.g. transcription factors. We model this situation using a matrix multiplication model in which we evolve two matrices A and B towards the goal $AB=G$, where G is a specified matrix that represents an evolutionary goal (Fig. 1B). The fitness in this case is $F=-\|AB-G\|$. Note that there is an infinite number of matrix pairs A and B that satisfy a given goal G.

As one concrete biological case, which may be kept in mind to guide the reader, the model can be interpreted in the context of a transcription network: if A is the matrix connecting transcription factor (TF) activities to gene expression, the relationship $Au=v$ means that a vector of TF activities u leads to a vector of gene expression v. The matrix element A_{ij} thus represents the regulatory strength of gene i by TF j. Similarly, if B is a matrix of interactions between external signals s and TF activities, one finds that the TF activities are $u=Bs$. The matrix element B_{ij} represents the effect of signal j on TF i. In total, the output gene expression vector that results from an input vector of signals s is ABs. The goal $AB=G$ means that for every set of signals s, the gene expression at the output of the system is $ABs=Gs$, where Gs

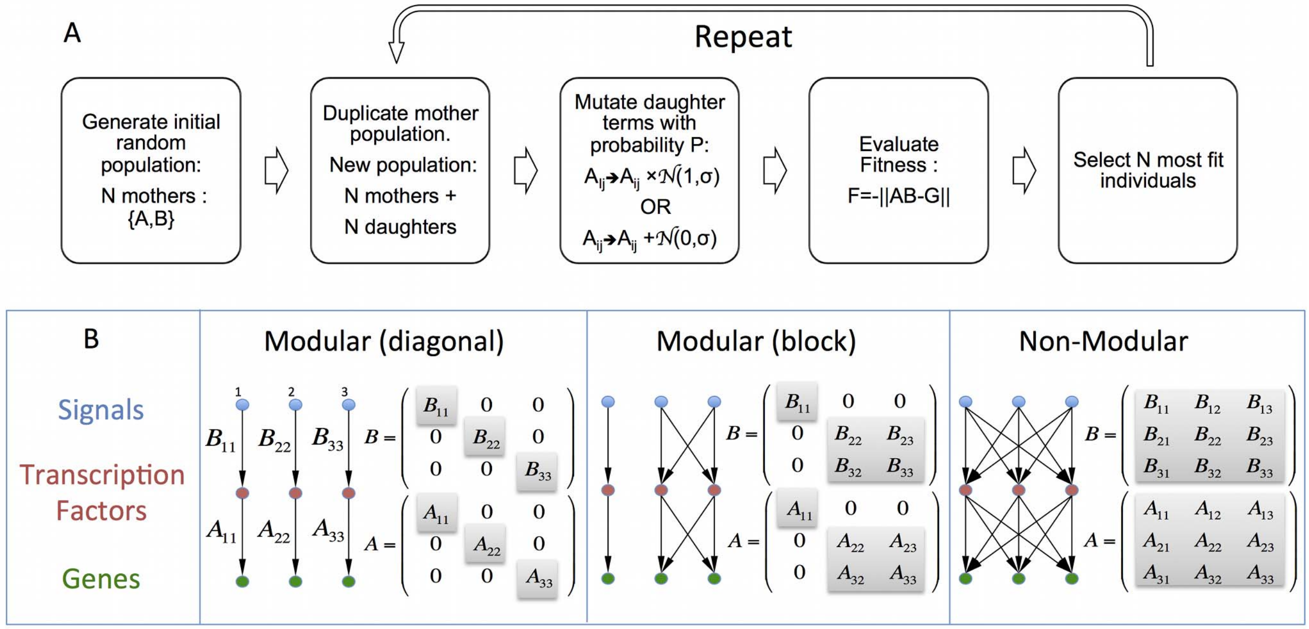

A
Repeat
Generate initial random population:
N mothers : {A,B}
Duplicate mother population.
New population:
N mothers +
N daughters
Mutate daughter terms with probability P:
$A_{ij}\rightarrow A_{ij}\times\mathcal{N}(1,\sigma)$
OR
$A_{ij}\rightarrow A_{ij}+\mathcal{N}(0,\sigma)$
Evaluate Fitness :
F=-||AB-G||
Select N most fit individuals
B
Modular (diagonal)
Modular (block)
Non-Modular
Signals
Transcription Factors
Genes
1
2
3
B_{11}
B_{22}
B_{33}
A_{11}
A_{22}
A_{33}
$B = \begin{pmatrix} B_{11} & 0 & 0 \\ 0 & B_{22} & 0 \\ 0 & 0 & B_{33} \end{pmatrix}$
$A = \begin{pmatrix} A_{11} & 0 & 0 \\ 0 & A_{22} & 0 \\ 0 & 0 & A_{33} \end{pmatrix}$
$B = \begin{pmatrix} B_{11} & 0 & 0 \\ 0 & B_{22} & B_{23} \\ 0 & B_{32} & B_{33} \end{pmatrix}$
$A = \begin{pmatrix} A_{11} & 0 & 0 \\ 0 & A_{22} & A_{23} \\ 0 & A_{32} & A_{33} \end{pmatrix}$
$B = \begin{pmatrix} B_{11} & B_{12} & B_{13} \\ B_{21} & B_{22} & B_{23} \\ B_{31} & B_{32} & B_{33} \end{pmatrix}$
$A = \begin{pmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{pmatrix}$

Figure 1. Evolutionary simulation scheme, and definition of model. (**A**) Simulation was initiated by randomly choosing N population members each consisting of 2 matrices A and B. The next steps were repeated at each generation until the stopping condition was satisfied: the population was duplicated, one copy was kept unchanged and the other was mutated with probability p. Mutation could be either sum-rule or product-rule. Fitness of all $2N$ members (original and mutated) was evaluated by the distance of the product AB from a desired goal matrix G, $F=-\|AB-G\|$, where $\|\cdot\|$ denotes the sum of squares of terms which is the square of L_2 (Frobenius) norm. N individuals were selected according to their fitness. Several selection methods were employed (see Text S1 for details). The simulation was stopped when the mean population fitness reached a value which was within a preset difference from the optimal fitness (usually 0.01). (**B**) Model represents a three layer network with a linear transformation function. Input signals s are transformed to intermediate layer activities (transcription factors) u with $u=Bs$. The intermediate layer is then transformed to output layer (gene expression) $v=Au=ABs$. Modularity means block or diagonal structure of the matrices, corresponding to signals that affect only subsets of intermediate and output nodes.
doi:10.1371/journal.pone.0070444.g001

is the desired gene expression profile for input signals s (see Fig. 1B).

Product-rule Mutations Lead to Sparse Structures, Sum-rule Mutations do not

We compared sum and product mutation rules in evolving the model systems using an evolutionary simulation. The sum-rule is the commonly used addition of a normally distributed random number to a randomly chosen element of the matrices, which represents a mutation in the intensity of a single interaction between network components,

$$\text{Sum-rule} \quad A_{ij}\rightarrow A_{ij}+N(0,\sigma) \text{ or } B_{ij}\rightarrow B_{ij}+N(0,\sigma).$$

We also tested product-rules, in which an element of the matrix is multiplied by a random number. We tested

$$\text{Product-rule} \quad A_{ij}\rightarrow A_{ij}\cdot N(\mu,\sigma) \text{ or } B_{ij}\rightarrow B_{ij}\cdot N(\mu,\sigma).$$

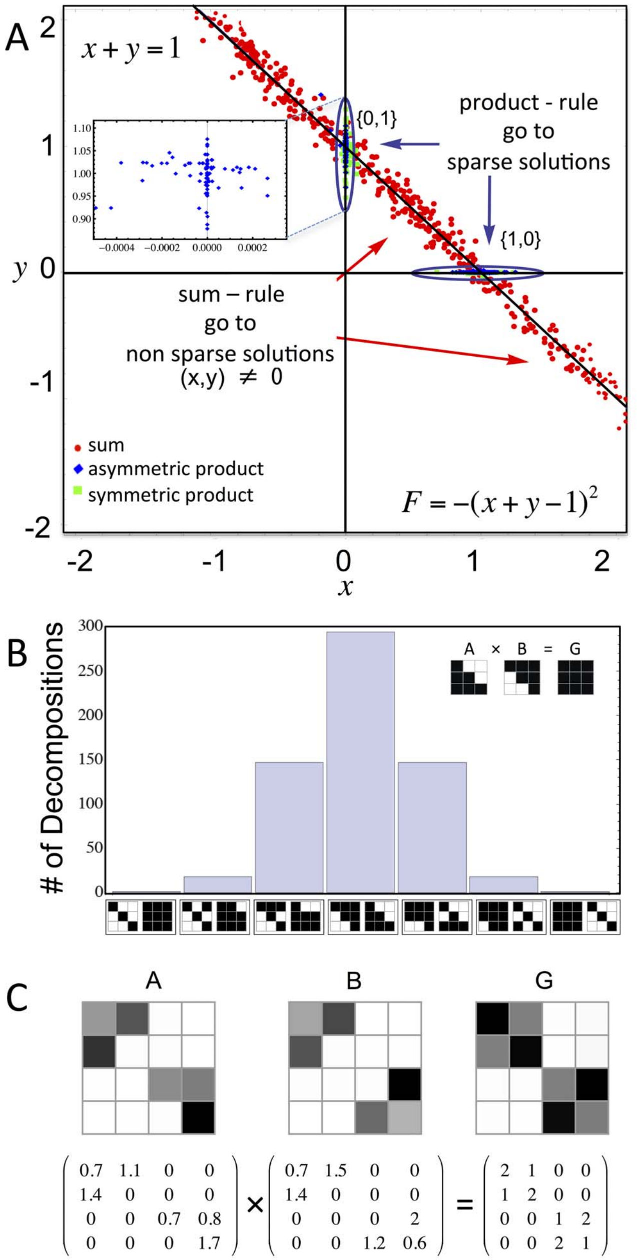
We study the case of $\mu=1$, and also cases in which $\mu<1$ and $\mu>1$. We also tested symmetric multiplication rules where the random number is log-normally distributed with $\mu=0$ (see Text S1 for details), and thus has equal chance to increase or decrease the absolute strength of the interaction:

$$\text{Symmetric product-rule} \quad A_{ij}\rightarrow A_{ij}\cdot 1LN(0,\sigma) \text{ or }$$
$$B_{ij}\rightarrow B_{ij}\cdot 1LN(0,\sigma).$$

.

All cases gave qualitatively similar results, and most of the data below is for multiplying by $N(1,\sigma)$. We also tested other forms of mutation distributions, including long tailed distributions that describe experimental data on sizes of mutation effects [Gamma distributions [39], see also [40,58,59] and references there], and found that the results are insensitive to the type of distribution used (see Movies S1, S2). Similarly, we tested the effect of mutation size, that is the parameter σ, which we varied between 0.01 and 3, and found that the results are insensitive to this parameter. The evolutionary simulation and parameters are described in the Methods Section below.

To demonstrate the effect of the mutation rule, we begin with a very simple model, namely a structure with two elements, x and y, with fitness $F=-(x+y-1)^2$. The optimal solutions lie on a line in the (x,y) plane, namely $x+y=1$ (Fig. 2A). Evolutionary simulations reach this line regardless of the mutation rule. Populations under sum-rule mutations evolve and spread out over the line. In contrast, product-rule mutations lead to solutions near the axes, either $(x=0,y=1)$, or $(x=1,y=0)$. In other words, they lead to solutions in which one of the elements is close to zero – these are the sparsest solutions that satisfy the goal (see Fig. 2A, Movies S1, S2 and Figs. S10–S11 in Text S1).



$$\begin{pmatrix} 0.7 & 1.1 & 0 & 0 \\ 1.4 & 0 & 0 & 0 \\ 0 & 0 & 0.7 & 0.8 \\ 0 & 0 & 0 & 1.7 \end{pmatrix} \times \begin{pmatrix} 0.7 & 1.5 & 0 & 0 \\ 1.4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 \\ 0 & 0 & 1.2 & 0.6 \end{pmatrix} = \begin{pmatrix} 2 & 1 & 0 & 0 \\ 1 & 2 & 0 & 0 \\ 0 & 0 & 1 & 2 \\ 0 & 0 & 2 & 1 \end{pmatrix}$$

Figure 2. Product-rule mutations reach sparse solutions, sum-rule ones do not. (**A**) We demonstrate the difference between sum-rule and product-rule mutations in a simple 2–variable system (x,y), where the goal is that $x+y=1$. The optimal solutions lie on the line $y=1-x$. We compare solutions to this problem achieved by 3 different mutational schemes. Sum-rule mutations ($+\mathrm{N}(0,0.5)$ red circles) provide solutions that are spread along the line. In contrast, solutions achieved with both Gaussian product-rule ($\times\mathrm{N}(1,0.5)$ blue diamonds) and log-normal product-rule ($\times\mathrm{LN}(1,0.5)$ green squares) are concentrated near the intersection with the axes, i.e. near either (0,1) or (1,0). Since one coordinate is near zero, these are sparse solutions. Inset illustrates the solutions obtained with Gaussian product-rule mutations, demonstrating that matrix values can be negative as well as positive. Evolutionary simulation parameters were $p=0.5$, $N=500$, selection scheme was Boltzmann-like selection with $\beta=10$. Simulations initiated utilizing random matrices with elements U(0,0.05). (**B**) Sparse solutions evolve in the matrix-multiplication model under product-rule mutations in response to a full-rank non-zero goal matrix G. The solutions have the maximal number of zeros while still satisfying the goal. Zeros are distributed between the two matrices A and B. Shown are the possible configurations of A and B for matrices of dimension $D=3$, in which 6 zeros are distributed between the two matrices A and B. (**C**) In general, if the goal is block diagonal and full-rank, each of its blocks can be decomposed separately into blocks of A and B, such that each block has the maximal number of zeros possible. Here we show an example in $D=4$, where G has 2 blocks of 2×2. The evolved A and B are such that each of their blocks is either an upper or a lower triangular matrix. Color represents numerical value (white = zero).
doi:10.1371/journal.pone.0070444.g002

The intuitive reason for the sparseness achieved by product-rule mutations is that once they are near a zero element, the size of the next mutation will be small (since it is a product of the element with a random number). Thus, the effective diffusion rate decreases (see Text S1). Strictly zero terms are fixed-points and near-zero terms remain small under mutations - so that the population becomes concentrated near zero elements. Sum-rule mutations, in contrast, show a constant drift rate regardless of the value of the elements. A full analytical solution of the dynamics of this simple model can be obtained by means of Fokker-Planck equations (see Text S1, Section 1), in excellent agreement with the simulations.

We tested product-rule mutations also in the matrix-multiplication model, using as goals full rank matrices G. In numerical simulations, we refer to terms that are relatively small (<0.1% of the average element in G) as ''zero terms'', because strictly zero terms are not reached in finite time. We find that product-rule mutations lead to sparseness: matrices A and B with the highest number of zeros possible while still satisfying the goal. In contrast, sum-rule mutations result in non-sparse solutions A and B with non-zero elements (Fig. 3).

The sparse solutions found with product-rule mutations have many zero terms, whose number can be computed by means of the LU decomposition theorem of linear algebra. The LU decomposition expresses a nonsingular matrix as a product of an upper triangular matrix and a lower triangular matrix [60]). The total number of zeros in A and B is the number of zero elements in the LU decomposition of G. This number can be calculated exactly: for a given full rank matrix G of dimension D with no zero elements, the maximal number of zeros in A and B together is D^2-D (for proof see Text S1). This result is found in our simulations.

The zeros are distributed between A and B in various ways in the different simulations: Sometimes A and B are both (upper and lower) triangular, each with $(D^2-D)/2$ zero elements. Other runs show one full matrix with no zeros and the other a diagonal matrix with D^2-D zeros. All other distributions of zeros are also found (Fig. 2B–C; Fig. S16 in Text S1 for comparison with sum-mutations). When G is full rank and has k zeros, the total number of zeros in the evolved matrices A and B is D^2-D+k, again the maximal possible number of zeros in matrices that show optimal fitness (for proof see Text S1).

We note that there is a special situation in which sum-rule mutations can also lead to sparseness in the present models. This occurs when the models are constrained to have only non-negative terms $A_{ij},B_{ij}\geq 0$. In this case, the sum rule, constrained to keep terms non-negative – for example, by using $A_{ij}\rightarrow\max(0,A_{ij}+\mathrm{N}(0,\sigma))$, can also lead to sparseness. This relates to known results from non-negative matrix factorization [61]. However, in general biological models, structural terms are expected to be both negative and positive, representing, for example, inhibition and activation interactions between components. Our mechanism for the evolution of sparseness and modularity is different from non-negative matrix factorization and works regardless of the sign of the interaction terms (see for examples Fig. S15 in Text S1 and Movies S1, S2).

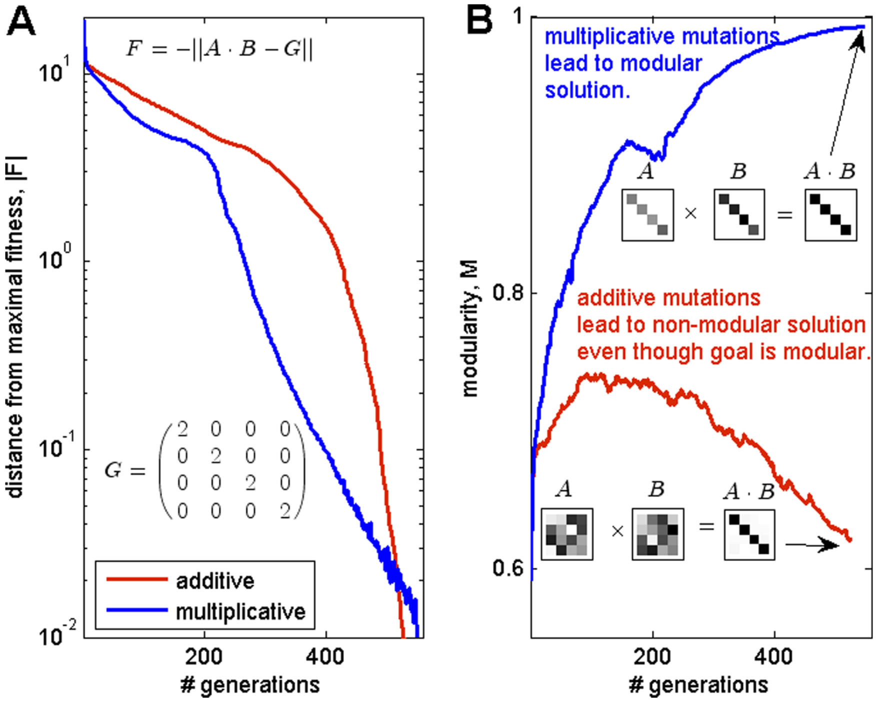


Figure 3. Product-rule mutations lead to modular structure under modular goal, sum-rule mutations do not. (**A**) Both sum-rule and product-rule mutations reach high fitness towards the goal $G=2I$. (**B**) Product-rule mutations reach high modularity, but sum-rule mutations do not. Simulations are in the matrix-multiplication model, matrix dimension $D=4$. Examples of matrices drawn from the simulations are shown, with gray scale corresponding to element absolute value (white = zero). Fitness reaches a value of 0.01 due to constantly occurring mutations. Evolutionary simulation parameters are: sum-rule mutation size N(0,0.05), product-rule mutation size N(1,0.27), $p=$ 0.0031, tournament selection $s=4$.
doi:10.1371/journal.pone.0070444.g003

When the Goal is Modular, Product-rule Mutations Lead to Modular Structure; Sum-rule Mutations do not

Up to now, we considered goals G which are described by general matrices. We next limit ourselves to the case where the goals G are described by matrices which are modular, for example, diagonal or block diagonal matrices. The main result is that when the goals are modular, the evolved structures A and B are also modular if mutations are product-rule; in contrast, sum-rule mutations lead to A and B that are not modular despite the fact that the goal is modular.

We first define modular structures and modular goals in the context of the present study. Modular structures are structures that can be decomposed into sets of components, where each set shows strong interactions within the set and weak interactions with other sets [1,2,10,62] (Fig. 1B). Here, modular structure means block-diagonal matrices. For ease of presentation, we first consider the most modular of structures – namely diagonal matrices. We define modularity by $M = 1 - \langle|n|\rangle/\langle|d|\rangle$ where $\langle|n|\rangle$ and $\langle|d|\rangle$ are the mean absolute value of the non-diagonal and diagonal terms respectively, and where we permute rows/columns to maximize modularity M (same permutation for rows of A and columns of B, see Text S1). Thus, a diagonal matrix has $M=1$, and a matrix with diagonal and non-diagonal terms of similar size has M close to zero.

Modular goals are goals which can be satisfied by a modular structure. Modular goals in the present models are represented by diagonal or block-diagonal goal matrices G. These goals, in the biological interpretation of transcription networks (Fig. 1B), are goals in which each small set of signals affects a distinct set of genes, and not the rest of the genes. For example, the signal lactose affects the *lac* genes in *E. coli*, whereas a DNA damage signal affects the *SOS* DNA-repair genes, with little crosstalk between these sets. Other examples for biological goals that are modular are sugar metabolism [63] and the tasks of chemotaxis and organism development (see detailed discussion in [27]). All are composed of several sub-tasks that are associated with different sets of genes.

We note that a modular goal does not necessarily lead to modular structures. For example the goal $G=I$ is modular since I is the diagonal identity matrix. This modular goal can be satisfied by a product of infinitely many pairs of non-modular matrices AB. In fact, for every invertible A, the inverse $B=A^{-1}$ satisfies the goal. As a result, the vast majority of the possible solutions are non-modular (modular solutions have measure zero among possible solutions to $AB=G$). In line with this observation, we find that simulations with sum-rule mutations lead to solutions with optimal fitness ($AB=G$), but with non-modular structure A and B (Fig. 3, Fig. S16 in Text S1).

In contrast, we find that product-rule mutations lead to modular structures A and B, for a wide range of parameters. For the goal $G=I$, the evolved A and B are both diagonal matrices, with elements on the diagonal of A that are the inverse of the corresponding elements on the diagonal of B. Thus $AB=G$. Similar results are found if the goal is nearly modular (e.g. diagonal with small but nonzero off-diagonal terms): in this case, the evolved A and B are both nearly diagonal (Fig. S14 in Text S1).

We also studied block-modular goals. In this case, product-rule mutations led to block-modular matrices A and B, with the same block structure as the goal matrix G (Fig. 2C). Each of the blocks in the matrices A and B had the maximal number of zeros possible so that the product of the two blocks is equal to the corresponding block in the goal matrix G (the total number of zeros is equal to that in the LU decomposition of each block) – compared to Fig. S16 in Text S1 (block-diagonal goal with sum-rule mutations).

It is important to note that in order to observe the evolution of modularity in the present setting, the selection criteria should not be too strict, otherwise non-modular solutions cannot be escaped effectively (Text S1). In other words, overly strict selection does not allow the search in parameter space needed for product-rule mutations to reach near-zero elements. In the present simulations, we find evolution of modularity using standard selection methods including tournament, elite (truncation) and continuous Boltzmann-like selection (see Methods, and Text S1 for analysis of sensitivity to parameters).

Time to Evolve Modular Structure Increases Polynomially with Matrix Dimension

We studied the dynamics of the evolutionary process in our simulations with product-rule mutations. We found that over time, fitness and modularity both generally increase, until a solution with optimal fitness and maximal modularity is achieved. We found that the matrix multiplication model often shows plateaus where fitness is nearly constant, followed by a series of events in which fitness improves sharply (Fig. 4) [22,64]. In these events, modularity often drops momentarily. Analysis showed that the plateaus represent non-modular and sub-optimal structures. A mutation occurs which reduces modularity but allows the system to readjust towards higher fitness, and then to regenerate modularity.

We also tested the time to reach high fitness solutions, and its dependence on the dimension of the matrices D. The time to high fitness solutions depends on the settings of the simulations: initial conditions, selection criteria and mutation rates and size, and the stopping criteria of the simulations. Here we present results in which time to high fitness was measured as the median time over repeat simulations to reach within 0.01 of optimal fitness, with product mutation rule ·N(1,0.1) and probability of mutation per element that is dimension-independent ($p = 5\times10^{-4}$). Initial conditions were matrices with small random elements (U(0,0.05)). The time to high fitness increased approximately as D^{θ_1} with $\theta_1 = 1.40+/-0.01$ and the time to modularity (see Methods for definition) increased as D^{θ_2} with $\theta_2 = 1.21+/-0.04$ (Fig. 5).

Discussion

We found that product-rule mutations lead evolution towards structures with the minimal number of interaction terms that still satisfy the fitness objective. Thus, product-rule mutations lead to sparseness. When the goal is modular, product-rule mutations lead to modular structure. This is in contrast to sum-rule mutations, which lead, under the same conditions, to non-sparse and non-modular solutions.

The mechanism by which product-rule mutations lead to sparseness and modularity is that near-zero interaction terms are kept small by product-rule (but not sum-rule) mutations. A second effect is mutation asymmetry, where it is more likely to reduce an interaction than increase it. However, using a symmetric product-rule (multiplying by a number drawn from a symmetric log-normal distribution) combined with selection still leads to sparseness and

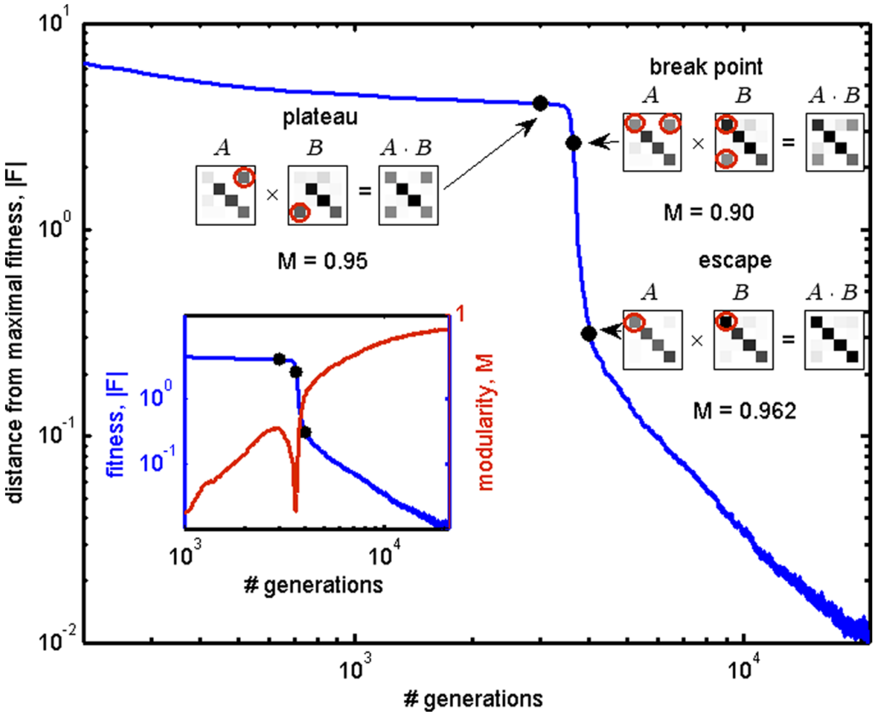


Figure 4. Escape from a fitness plateau entails a temporary decrease in modularity. Mean distance from maximal fitness as a function of time in the matrix-multiplication model with product-rule mutations, towards a diagonal goal. Note the plateau in the dynamics. Matrices and their modularity drawn from the simulations at different time-points (designated by black points) are shown, with gray scale corresponding to element absolute value (white = zero). Inset: mean modularity of population (red curve), showing a sharp decrease at the time of escape from the plateau (same time points are shown). In order to escape the plateau ("break point"), the circled terms in A and B are changed. This occurs through a simultaneous increase of the new term and decrease of the old one, such that temporarily modularity is decreased (see inset). Finally, the correct arrangement of terms is attained ("escape") and modularity increases again. Fitness reaches a value of 0.01 due to constantly occurring mutations.
doi:10.1371/journal.pone.0070444.g004

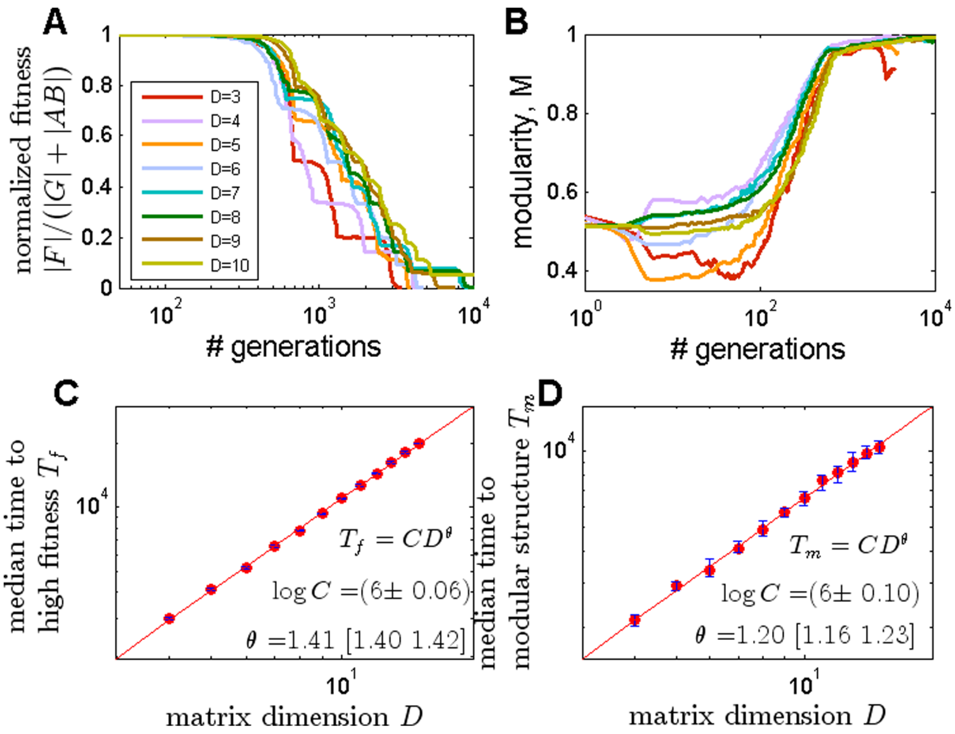


Figure 5. Time to high fitness and modularity grows polynomially with dimension. (**A**) Normalized distance to maximal fitness $|F|/(|G|+|AB|)$ as a function of generations in the matrix-multiplication model evolved towards $G=2I$, for matrix dimensions $D=3$ to 10. Each color represents a different value of D. Curves typically have D "steps", where each step corresponds to the build-up of an additional significant term. (**B**) Modularity in the same simulations. (**C**) Median time (generations) to high fitness (distance from maximal fitness <0.01) as a function of the dimensions of the matrices D goes as $T \sim D^{\theta_1}$ with $\theta_1 = 1.41$ [1.40, 1.42] [CI 5%, 95%]. (**D**) Median time (generations) to modular structure (see methods for dimension dependent criterion for high modularity) goes as D^{θ_2} with $\theta_2 = 1.20$ [1.16, 1.23]. Initial conditions are random matrices with small elements drawn from U(0,0.1). Element-wise mutation rate p at all simulations was 0.0005; product-rule mutations normally distributed N(1,0.1). See Text S1 for details on error calculation in **C**–**D**.
doi:10.1371/journal.pone.0070444.g005

modularity, because selection also breaks the symmetry. Once a parameter becomes small, product rule mutations keep it small (as opposed to sum-rule mutations). This creates a dynamic 'trap' in which the steady state distribution of phenotypes is highly enriched with near zero parameters. Thus, the mutational asymmetry effect is not essential for the present conclusions (see Figs. S10–S11 in Text S1 and Section 2 in Text S1). Furthermore, in special situations a sum-rule can also lead to sparseness, namely if the structural terms in the model are constrained to be non-negative. We note that sparseness can also be enhanced in some networks due to physical constraints such as spatial/geometric limitations in networks that describe protein structure [12] or neuron wiring networks [65].

We used a simple but general model of biological systems, namely linear matrix models, and matrix multiplication models. These models have been widely used to describe gene regulation, neuronal networks, signal transduction and other systems [41,44–49,66,67]. The matrix multiplication model is a commonly used model for three layer systems, such as signals → transcription factors → genes. As in many biological models, many combinations of parameters can achieve the same goal.

We believe that the present mechanism has generality beyond the particular model used here. Consider a general map H between a coarse-grained genotype G (described as a set of biochemical parameters and interaction parameters) and phenotype P, $P=H(G)$. The optimal phenotype P^* is obtained by a manifold of different genotypes G^*. Given reasonably strong selection relative to genetic drift and mutation, evolutionary dynamics will reach close to this manifold. One can then ask how the mutation rule affects evolutionary dynamics along this manifold. Sum-rule mutations lead to a random walk on the manifold that does not prefer regions with small parameters, whereas product-rule mutations lead to solutions with the maximal number of zero (very small) parameters: once evolution comes close to a zero parameter, product-rule mutations keep that parameter small.

Product-rule is a more realistic description of the effect of cumulative genetic mutations on a biochemical parameter than sum-rule mutations, because of the nature of biological interactions. The effect of genetic mutations was also shown in several experimental studies to be asymmetric (for example [38–40]), with bias to decrease interactions, enzymatic activity [38] or body size [40]. The discussion of symmetric product rule mutations (that is - multiplying by log-normally distributed random numbers) is given here for completeness, and not because of biological relevance. Further studies can use other microscopic models for mutations (such as Ising-like models for bonds between macromolecules [41,68]), and explore the effect of mutations that set interactions to near-zero with large probability. Due to the inherent product-rule nature of biological mutations, we could not think of experimental tests that can compare sum-rule to product-rule mutations, beyond computer simulations or experiments in the realm of electronics [69,70] or mechanics [71].

The present mechanism does not exclude previous mechanisms for the evolution of modularity. In fact, it can work together with other mechanisms and enhance them. For example, in Kashtan *et al.* [14,27,29,30], modularity evolved when the modular goal changed over time (MVG mechanism). In the present study, no change of the goal over time is required. Using product-rule mutations in the models of Kashtan *et al.* (instead of the original sum-rule mutations) is expected to enhance the range of parameters over which modularity evolves. Supporting evidence was recently provided by Clune *et al.* [26] that demonstrated how a different mechanism for the evolution of sparseness significantly enlarges the range of parameters over which the MVG mechanism produces modular structures. Another difference from some previous studies is that modularity evolves here with no need for an explicit cost for interaction terms in the fitness function [14,27,29,30,72]. Adding such a cost, as in Clune *et al.*, [26] would likely enhance the evolution of sparseness and modularity. It would be intriguing to search for additional classes of mechanisms to understand the evolution of sparseness, modularity, and other generic features of biological organization [73].

Materials and Methods

Evolutionary Simulation

Simulation was written in Matlab using standard framework [42,43]. All source codes, data and analysis scripts are freely available in a permanent online archive at http://dx.doi.org/doi:10.5061/dryad.75180. We initialized the population of matrix pairs by drawing their $N\cdot 2D^2$ terms from a uniform distribution. Population size was set as $\mathcal{N}=500$. In each generation the population was duplicated. One of the copies was kept unchanged, and elements of the other copy had a probability p to be mutated – as we explain below. Fitness of all $2\mathcal{N}$ individuals was evaluated by $F=-\|AB-G\|$, where $\|\cdot\|$ denotes the sum of squares of elements [56]. The best possible fitness is zero, achieved if $AB=G$ exactly. Otherwise, fitness values are negative. In the figures we show the absolute value of mean population fitness, which is the distance from maximal fitness (Fig. 3–4, Fig. S12 in Text S1), or the normalized fitness $|F|/(|G|+|AB|)$ (Fig. 5). The goal matrix was either diagonal , nearly-diagonal (diagonal matrix with small non-diagonal terms), block-diagonal or full rank with no zero elements. N individuals are selected out of the $2N$ population of original and mutated ones, based on their fitness (see below). This mutation–selection process was repeated until the simulation stopping condition was satisfied (usually when mean population fitness was within 0.01 of the optimum).

Mutation. We mutated individual elements in the matrix. We set mutation rate such that on average 10% of the population members were mutated at each generation, so the element-wise mutation rate was This relatively low mutation rate enables beneficial mutants to reproduce on average at least 10 generations before they are mutated again. In simulations where we compared dependence on matrix dimension (Fig. 5) we used the same mutation rate at all dimensions, generally the one that pertains to the highest dimension used in the simulation.

We randomly picked the matrix elements (in both A and B) to be mutated. Mutation values were drawn from a Gaussian distribution (unless otherwise stated). For sum-rule mutations, this random number was added to the mutated matrix value: $A_{ij}\to A_{ij}+\mathrm{N}(0,\sigma)$ or $B_{ij}\to B_{ij}+\mathrm{N}(0,\sigma)$, and for product-rule mutation, the mutated matrix element was multiplied by the random number: or $B_{ij}\to B_{ij}\cdot\mathrm{N}(\mu,\sigma)$. Mean mutation value μ was usually taken as 1, however we also tested other values of μ (both larger and smaller than 1) and other mutation distributions (Gamma and log-normal) and results remained qualitatively similar, although the time-scales changed. In most simulations shown here we used $\sigma=0.1$ (unless stated otherwise). Fitness convergence and its time scale depend on the mutation frequency and size, as demonstrated in our sensitivity test (Text S1).

Selection methods. We tested 3 different selection methods and all gave qualitatively very similar results with only a difference in time scales. Most results presented here were obtained with tournament selection with group size S = 4 (see [43] chap. 9). We also tested truncation-selection (elite) [42] and proportionate reproduction with Boltzmann-like scaling [41,55,74]. For a detailed description see Text S1.

Definition of modularity. If the goal is diagonal, we define modularity as $M=1-\langle|n|\rangle/\langle|d|\rangle$ where $\langle|n|\rangle$ and $\langle|d|\rangle$ are the mean absolute value of the non-diagonal and diagonal terms respectively. At each generation, the D largest elements of each matrix (both A and B), were considered as the diagonal $\langle|d|\rangle$ and the rest D^2-D terms as the non-diagonal ones $\langle|n|\rangle$. Averages were taken over matrix elements and over the population. This technique copes with the unknown location of the dominant terms in the matrices, which could form any permutation of a diagonal matrix. Thus, $0\le M\le 1$: a diagonal matrix has $M=1$, and a matrix whose terms are all equal has $M=0$. Since we choose the largest elements to form the diagonal, negative values of M do not occur. When the goal is non-diagonal, one can use standard measures for modularity such as (49) [not used in the present study].

Calculation of time to modular structure. To estimate the time when modular structure is first obtained, we used the following approximation for fitness value with diagonal goal. Assume that A and B are D- dimensional matrices consisting of 2 types of terms: diagonal terms all with size d and non-diagonal terms all with size n and that the goal is $G=g\times I_{D\times D}$. The fitness then equals:

$$-F=D[d^2+(D-1)n^2-g]^2+D(D-1)(2dn+(D-2)n^2)^2.$$

We collect terms by powers of n, and obtain a constant term and terms with powers $n^{2,3,4}$. Modular structure is obtained when the solution has the correct number of dominant terms at the right location and their size is approximately $d^2\cong g$. At the beginning of the temporal trajectory, when non-diagonal elements are relatively large, F is dominated by the $O(n^4)$ term. When a modular structure emerges, non-diagonal elements become relatively small, and the dominant term remaining in F is $O(n^2)$. Our criterion for determining time to modular structure was the time when the $O(n^2)$ term first became dominant, i.e. when $F-n^2(\cdots)<n^2(\cdots)$.

Supplementary movies. Movies demonstrate the simulation dynamics in the problem under product-rule mutations with various distributions of mutations. All distributions converge to either of the sparse solutions.

Supporting Information

Text S1 Contains the following additional data: 1. Analytical solution and simulations of toy model, 2; 2. Mutation properties: product vs. sum mutation, mutation symmetry, 11; 3. Evolutionary simulations – detailed dexcription, 16; 4. Evolutionary simulation parameter sensitivity test, 18; 5. Modularity: definitions and error calculation, 20; 6. LU decomposition – proofs, 21; 7. Nearly modular - supplementary figure, 23; 8. Mutation sign and distribution – supplementary figure, 24; 9. Block diagonal goal – supplementary figure, 25.
(PDF)

Movie S1 Mutations had Gamma distribution with parameters Gamma(1, 40.25). In addition, each mutation value was multiplied by −1 with probability 0.1, so that matrix values could also change their sign. - selection was used with.
(AVI)

Movie S2 Mutations had log-normal distribution with parameters LN(−0.11, 0.47). In addition, each mutation value was multiplied by −1 with probability 0.1, so that matrix values could also change their sign. - selection was used with.
(AVI)

Acknowledgments

We thank Yuval Hart, Adam Lampert, Omer Ramote, Hila Sheftel, Oren Shoval and Pablo Szekely for critical reading of the manuscript; and Amos Tanay, Dan Tawfik, Nadav Shnerb and Gheorghe Craciun for useful discussions. We thank the two anonymous referees for helpful comments and suggestions that improved our manuscript.

Author Contributions

Conceived and designed the experiments: UA. Performed the experiments: TF AEM. Analyzed the data: TF AEM. Contributed reagents/materials/analysis tools: TT. Wrote the paper: TF AEM UA.

References

1. Simon HA (1962) The architecture of complexity. Proceedings of the American philosophical society 106: 467–482.
2. Wagner GP, Pavlicev M, Cheverud JM (2007) The road to modularity. Nature Reviews Genetics 8: 921–931. doi:10.1038/nrg2267.
3. Lipson H (2007) Principles of modularity, regularity, and hierarchy for scalable systems. Journal of Biological Physics and Chemistry 7: 125.
4. Carroll SB (2001) Chance and necessity: the evolution of morphological complexity and diversity. Nature 409: 1102–1109.
5. Hintze A, Adami C (2008) Evolution of complex modular biological networks. PLoS computational biology 4: e23.
6. Mountcastle VB (1997) The columnar organization of the neocortex. Brain 120: 701–722.
7. Guimera R, Amaral LAN (2005) Functional cartography of complex metabolic networks. Nature 433: 895–900.
8. Hartwell LH, Hopfield JJ, Leibler S, Murray AW, others (1999) From molecular to modular cell biology. Nature 402: 47.
9. Alon U (2007) An Introduction to Systems Biology: Design Principles of Biological Circuits (Mathematical and Computational Biology Series vol 10). Boca Raton, FL: Chapman and Hall. Available: http://www.lavoisier.fr/livre/notice.asp?ouvrage = 1842587. Accessed 4 June 2013.
10. Lorenz DM, Jeng A, Deem MW (2011) The emergence of modularity in biological systems. Physics of Life Reviews 8: 129–160. doi:10.1016/j.plrev.2011.02.003.
11. Gama-Castro S, Salgado H, Peralta-Gil M, Santos-Zavaleta A, Muñiz-Rascado L, et al. (2011) RegulonDB version 7.0: transcriptional regulation of Escherichia coli K-12 integrated within genetic sensory response units (Gensor Units). Nucleic Acids Research 39: D98–D105.
12. Rorick M (2012) Quantifying protein modularity and evolvability: a comparison of different techniques. Biosystems.
13. Alon U (2003) Biological networks: the tinkerer as an engineer. Science 301: 1866–1867.
14. Kashtan N, Mayo AE, Kalisky T, Alon U (2009) An Analytically Solvable Model for Rapid Evolution of Modular Structure. PLoS Comput Biol 5: e1000355. doi:10.1371/journal.pcbi.1000355.
15. Solé RV, Valverde S (2008) Spontaneous Emergence of Modularity in Cellular Networks. J R Soc Interface 5: 129–133. doi:10.1098/rsif.2007.1108.
16. Force A, Cresko WA, Pickett FB, Proulx SR, Amemiya C, et al. (2005) The origin of subfunctions and modular gene regulation. Genetics 170: 433–446.
17. Takemoto K (2012) Metabolic network modularity arising from simple growth processes. Phys Rev E 86: 036107. doi:10.1103/PhysRevE.86.036107.
18. Takemoto K (2013) Does Habitat Variability Really Promote Metabolic Network Modularity? PLoS ONE 8: e61348. doi:10.1371/journal.pone.0061348.
19. Variano EA, McCoy JH, Lipson H (2004) Networks, dynamics, and modularity. Physical review letters 92: 188701.
20. Rainey PB, Cooper TF (2004) Evolution of bacterial diversity and the origins of modularity. Research in microbiology 155: 370–375.
21. Leroi AM (2000) The scale independence of evolution. Evolution & development 2: 67–77.
22. Ancel LW, Fontana W, others (2000) Plasticity, evolvability, and modularity in RNA. Journal of Experimental Zoology 288: 242–283.
23. He J, Sun J, Deem MW (2009) Spontaneous emergence of modularity in a model of evolving individuals and in real networks. Phys Rev E 79: 031907. doi:10.1103/PhysRevE.79.031907.
24. Espinosa-Soto C, Wagner A (2010) Specialization Can Drive the Evolution of Modularity. PLoS Comput Biol 6: e1000719. doi:10.1371/journal.pcbi.1000719.
25. Callahan B, Thattai M, Shraiman BI (2009) Emergent gene order in a model of modular polyketide synthases. Proceedings of the National Academy of Sciences 106: 19410–19415.
26. Clune J, Mouret J-B, Lipson H (2013) The evolutionary origins of modularity. Proc R Soc B 280: 20122863. doi:10.1098/rspb.2012.2863.
27. Kashtan N, Alon U (2005) Spontaneous Evolution of Modularity and Network Motifs. PNAS 102: 13773–13778. doi:10.1073/pnas.0503610102.
28. Parter M, Kashtan N, Alon U (2008) Facilitated variation: How evolution learns from past environments to generalize to new environments. PLoS Computational Biology 4: e1000206.
29. Kashtan N, Parter M, Dekel E, Mayo AE, Alon U (2009) Extinctions in heterogeneous environments and the evolution of modularity. Evolution 63: 1964–1975.
30. Kashtan N, Noor E, Alon U (2007) Varying environments can speed up evolution. Proceedings of the National Academy of Sciences 104: 13711.
31. Wells JA (1990) Additivity of mutational effects in proteins. Biochemistry 29: 8509–8517. doi:10.1021/bi00489a001.
32. Von Hippel PH, Berg OG (1986) On the specificity of DNA-protein interactions. Proceedings of the National Academy of Sciences 83: 1608.
33. Maerkl SJ, Quake SR (2007) A Systems Approach to Measuring the Binding Energy Landscapes of Transcription Factors. Science 315: 233–237. doi:10.1126/science.1131007.
34. Maslov S, Ispolatov I (2007) Propagation of large concentration changes in reversible protein-binding networks. Proceedings of the National Academy of Sciences 104: 13655–13660.
35. Zhang J, Maslov S, Shakhnovich EI (2008) Constraints imposed by non-functional protein–protein interactions on gene expression and proteome size. Molecular systems biology 4. Available: http://www.nature.com/msb/journal/v4/n1/synopsis/msb200848.html.
36. Heo M, Maslov S, Shakhnovich E (2011) Topology of protein interaction network shapes protein abundances and strengths of their functional and nonspecific interactions. Proceedings of the National Academy of Sciences 108: 4258–4263.
37. Wacholder S, Han SS, Weinberg CR (2011) Inference from a multiplicative model of joint genetic effects for ovarian cancer risk. Journal of the National Cancer Institute 103: 82–83.
38. Soskine M, Tawfik DS (2010) Mutational effects and the evolution of new protein functions. Nature Reviews Genetics 11: 572–582. doi:10.1038/nrg2808.
39. Silander OK, Tenaillon O, Chao L (2007) Understanding the evolutionary fate of finite populations: the dynamics of mutational effects. PLoS biology 5: e94.
40. Azevedo RB, Keightley PD, Laurén-Määttä C, Vassilieva LL, Lynch M, et al. (2002) Spontaneous mutational variation for body size in Caenorhabditis elegans. Genetics 162: 755–765.
41. Burda Z, Krzywicki A, Martin OC, Zagorski M (2010) Distribution of essential interactions in model gene regulatory networks under mutation-selection balance. Phys Rev E 82: 011908. doi:10.1103/PhysRevE.82.011908.
42. Goldberg DE (1989) Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley.
43. Spall JC (2003) Introduction to Stochastic Search and Optimization: Estimation, Simulation and Control. Wiley-Blackwell. 618 p.
44. Wagner A (1994) Evolution of gene networks by gene duplications: a mathematical model and its implications on genome organization. Proceedings of the National Academy of Sciences 91: 4387–4391.
45. Wagner A (1996) Does evolutionary plasticity evolve? Evolution: 1008–1023.
46. Siegal ML, Bergman A (2002) Waddington's canalization revisited: developmental stability and evolution. Proceedings of the National Academy of Sciences 99: 10528–10532.
47. Ciliberti S, Martin OC, Wagner A (2007) Innovation and robustness in complex regulatory gene networks. Proceedings of the National Academy of Sciences 104: 13591–13596.
48. Borenstein E, Krakauer DC (2008) An end to endless forms: epistasis, phenotype distribution bias, and nonuniform evolution. PLoS computational biology 4: e1000202.
49. Burda Z, Krzywicki A, Martin OC, Zagorski M (2011) Motifs emerge from function in model gene regulatory networks. Proceedings of the National Academy of Sciences 108: 17263–17268.
50. Yeung MKS, Tegnér J, Collins JJ (2002) Reverse engineering gene networks using singular value decomposition and robust regression. PNAS 99: 6163–6168. doi:10.1073/pnas.092576199.
51. Mjolsness E, Sharp DH, Reinitz J (1991) A connectionist model of development. Journal of theoretical Biology 152: 429–453.
52. Reinitz J, Sharp DH (1995) Mechanism of eve stripe formation. Mechanisms of Development 49: 133–158. doi:10.1016/0925-4773(94)00310-J.

53. Reinitz J, Mjolsness E, Sharp DH (1995) Model for cooperative control of positional information in Drosophila by bicoid and maternal hunchback. Journal of Experimental Zoology 271: 47–56.
54. Wunderlich Z, Bragdon MD, Eckenrode KB, Lydiard-Martin T, Pearl-Waserman S, et al. (2012) Dissecting sources of quantitative gene expression pattern divergence between Drosophila species. Molecular Systems Biology 8. Available: http://www.nature.com/msb/journal/v8/n1/full/msb201235.html.
55. Lipson H, Pollack JB, Suh NP (2007) On the origin of modular variation. Evolution 56: 1549–1556. doi:10.1111/j.0014–3820.2002.tb01466.x.
56. Fisher RA (1930) The Genetical Theory of Natural Selection. 1st ed. Bennett JH, editor Oxford University Press, USA. 318 p.
57. Bray D (1995) Protein molecules as computational elements in living cells. Nature 376: 307–312.
58. Sanjuán R, Moya A, Elena SF (2004) The distribution of fitness effects caused by single-nucleotide substitutions in an RNA virus. Proceedings of the National Academy of Sciences of the United States of America 101: 8396–8401.
59. Good BH, Rouzine IM, Balick DJ, Hallatschek O, Desai MM (2012) Distribution of fixed beneficial mutations and the rate of adaptation in asexual populations. Proceedings of the National Academy of Sciences 109: 4950–4955.
60. Cormen TH, Leiserson CE, Rivest RL, Stein C (2001) Introduction To Algorithms. MIT Press. 1216 p.
61. Lee DD, Seung HS (1999) Learning the parts of objects by non-negative matrix factorization. Nature 401: 788–791. doi:10.1038/44565.
62. Newman MEJ (2006) Modularity and community structure in networks. Proceedings of the National Academy of Sciences 103: 8577–8582.
63. Kaplan S, Bren A, Zaslaver A, Dekel E, Alon U (2008) Diverse Two-Dimensional Input Functions Control Bacterial Sugar Genes. Molecular Cell 29: 786–792. doi:10.1016/j.molcel.2008.01.021.
64. Kauffman S, Levin S (1987) Towards a general theory of adaptive walks on rugged landscapes. Journal of Theoretical Biology 128: 11–45. doi:10.1016/S0022–5193(87)80029–2.
65. Wen Q, Stepanyants A, Elston GN, Grosberg AY, Chklovskii DB (2009) Maximization of the connectivity repertoire as a statistical principle governing the shapes of dendritic arbors. PNAS 106: 12536–12541. doi:10.1073/pnas.0901530106.
66. Hopfield JJ, Tank DW (1986) Computing with neural circuits- A model. Science 233: 625–633.
67. Haykin SS (1999) Neural networks: a comprehensive foundation. Prentice Hall. 872 p.
68. Lancet D, Sadovsky E, Seidemann E (1993) Probability Model for Molecular Recognition in Biological Receptor Repertoires: Significance to the Olfactory System. PNAS 90: 3715–3719.
69. Thompson A, Harvey I, Husbands P (1996) Unconstrained evolution and hard consequences. In: Sanchez E, Tomassini M, editors. Towards Evolvable Hardware. Lecture Notes in Computer Science. Springer Berlin/Heidelberg, Vol. 1062. 136–165. Available: http://www.springerlink.com/content/f6m8731w5706w108/abstract/.
70. Thompson A (1997) An evolved circuit, intrinsic in silicon, entwined with physics. In: Higuchi T, Iwata M, Liu W, editors. Evolvable Systems: From Biology to Hardware. Lecture Notes in Computer Science. Springer Berlin/Heidelberg, Vol. 1259. 390–405. Available: http://www.springerlink.com/content/u734gr0822r13752/abstract/.
71. Zykov V, Mytilinaios E, Adams B, Lipson H (2005) Robotics: Self-reproducing machines. Nature 435: 163–164. doi:10.1038/435163a.
72. Pan RK, Sinha S (2007) Modular networks emerge from multiconstraint optimization. Phys Rev E 76: 045103. doi:10.1103/PhysRevE.76.045103.
73. Csete M, Doyle J (2004) Bow ties, metabolism and disease. TRENDS in Biotechnology 22: 446–450.
74. Lampert A, Tlusty T (2009) Mutability as an altruistic trait in finite asexual populations. Journal of Theoretical Biology 261: 414–422. doi:10.1016/j.jtbi.2009.08.027.