Generating complex patterns of gene expression without regulatory circuits

Sahil B. Shah[1], Alexis M. Hill[1], Claus O. Wilke[1], and Adam J. Hockenberry[1]

[1]*Department of Integrative Biology, The University of Texas at Austin, Austin TX, 78712*

November 25, 2020

Abstract

Synthetic biology has successfully advanced our ability to design complex, time-varying genetic circuits executing precisely specified gene expression patterns. However, such circuits usually require regulatory genes whose only purpose is to regulate the expression of other genes. When designing very small genetic constructs, such as viral genomes, we may want to avoid introducing such auxiliary gene products. To this end, here we demonstrate that varying only the placement and strengths of promoters, terminators, and RNase cleavage sites in a computational model of a bacteriophage genome is sufficient to achieve solutions to a variety of basic expression patterns. We discover these solutions by computationally evolving genomes to reproduce desired target expression patterns. Our approach shows non-trivial patterns can be evolved, including patterns in which the relative ordering of genes by abundance changes over time. We find that some patterns are easier to evolve than others, and different genomes that express comparable expression patterns may differ in their genetic architecture. Our work opens up a novel avenue to genome engineering via fine-tuning the balance of gene expression and gene degradation rates.

1 Introduction

In all species, expressed RNAs or proteins are required in different amounts relative to one-another, and this demand will further vary over time [1, 2, 3]. Thus, organisms must have the ability to modulate gene expression levels to meet lifestyle needs and adapt to changing environmental conditions [4, 5, 6]. Many gene regulatory elements are well-characterized and cells have been directly engineered to produce individual protein products for decades; the strength of promoters, terminators, ribosome binding sites, *etc.* can all vary over several orders of magnitude to produce different amounts of recombinant gene products [7, 8, 9].

More recently, the field of synthetic biology has developed and achieved success in engineering large, time-varying genetic circuits that are characterized by complex interactions and often require producing gene products whose sole job it is to regulate the expression of other genes [10, 11, 12, 13, 14, 15]. While the scale and capabilities of these applications are continually expanding [16, 17, 18], the ability to design entire genomes from scratch presents tremendous challenges. One particular source of challenges is genome complexity—even the smallest free living organisms encode hundreds of gene products that interact with one-another and alter gene expression patterns in unpredictable ways [19].

Even smaller model systems are viruses and bacteriophages. Bacteriophages are useful from an engineering and genome-design standpoint because of their comparatively small size, genetic tractability, and potential uses in medical and biotechnological applications [20, 21, 22, 23, 24, 25]. Despite the limited set of genes encoded by their genomes, phages are nevertheless capable of producing complex, time-varying patterns of gene expression [26]. While some of these dynamics are governed by networks of sequence-specific promoters and transcription factors, some phages—such as the *Escherichia coli* phage T7—appear to regulate a portion of their transcriptional demand via a simpler model of variable production and degradation rates [27, 28]. The full range of expression dynamics that can be achieved over the time-course of a typical phage infection by varying only these mechanisms across different genes is currently unknown.

Here, we use an *in silico* model of phage infection to consider the range of gene expression complexity that can be achieved using only basic regulatory mechanisms, including transcription, transcript termination, transcript cleavage, and transcript degradation. Our approach relies on computation-

ally evolving genomes with variable-strength promoters, terminators, and RNase cleavage sites to reproduce predefined gene expression time-course patterns. We show that this molecular-level simulation framework can discover solutions to numerous non-trivial gene expression patterns. Evolved genomes display a variety of distinct genome architectures even when displaying comparable gene expression patterns. Taken together, these findings lay the groundwork for future efforts to use *in silico* evolutionary design methods to engineer novel phage genomes.

2 Results

2.1 Evolutionary simulation of phage gene expression

Computational simulations of phage infections have become increasingly complex and realistic over the years [29, 30, 31, 32]. Here, we use a recently developed stochastic gene expression platform (Pinetree) that simulates the process of phage infection in molecular-level detail [32]. We built upon this software to enable evolutionary simulations, where discrete generations consist of individual simulations of phage infection and the resulting gene expression time-course is our phenotype of interest. The genome-specific input varies from generation-to-generation as the location and strengths of promoters, terminators, and RNase cleavage sites are subject to mutation. As a proof-of-principle, we first attempted to evolve a genome to match the gene expression time course pattern produced by an arbitrarily chosen genome that we first simulated in Pinetree.

The positive control genome architecture and target gene expression pattern produced from a single simulation is shown in Fig. 1A (left). Our evolutionary simulation starts with a three-gene genome containing only a single promoter, whose time-course of gene expression initially bears little resemblance to the target. As generations progress, single mutations are proposed and conditionally accepted based on whether they alter the resulting gene expression pattern to more closely resemble the target (see Materials and Methods). More specifically, we define the fitness of a mutation as the inverse of the normalized-Root Mean Squared Error (RMSE) relative to the target. Individual mutations that result in a better match to the target expression pattern (*i.e.* have lower normalized-RMSE values) are accepted, with some probability of accepting slightly deleterious mutations to allow for

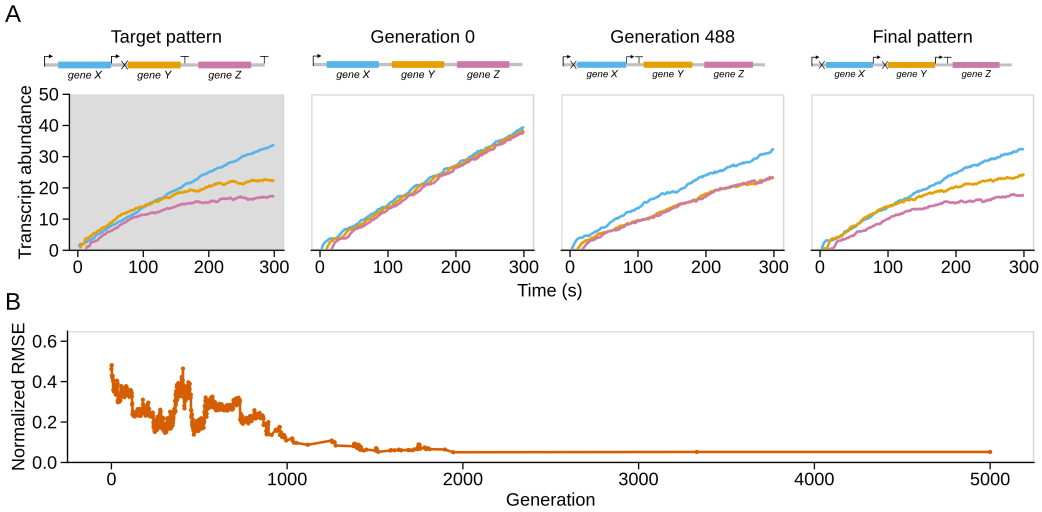

Figure 1: Evolutionary simulation of a positive control gene expression pattern. (A) The 'Target pattern' (left) is a genome architecture that was simulated in Pinetree, producing the corresponding gene expression time course (line colors correspond to the genes labeled in the genome architecture). The remaining panels illustrate the evolutionary process: the starting genome architecture and corresponding gene expression time-course (Generation 0), an example from the middle of the evolutionary simulation (Generation 488), and the final evolved architecture (Final pattern). (B) The corresponding normalized-RMSE metric showing a better match to the target over the 5,000 generation simulation.

efficient exploration of the potentially rugged fitness landscape.

As shown in Fig. 1A (right), the best gene expression time-course that evolved over a 5,000 generation evolutionary simulation qualitatively matches the target. To quantitatively make this determination, we consider an evolutionary simulation successful if the normalized-RMSE value of the final evolved gene expression time-course is less than or equal to 0.1, which is achieved in this case (Fig. 1B). Thus, this positive control shows that our approach is capable of evolving genome architectures that match complex patterns from simple starting points. Interestingly, we note that while the evolved genome produced a gene expression pattern that looks similar to the target (Fig. 1A), the evolved genome architecture is distinct from the input architecture that was initially used to produce the target expression data.

2.2 Evolving genomes to match a range of gene expression patterns

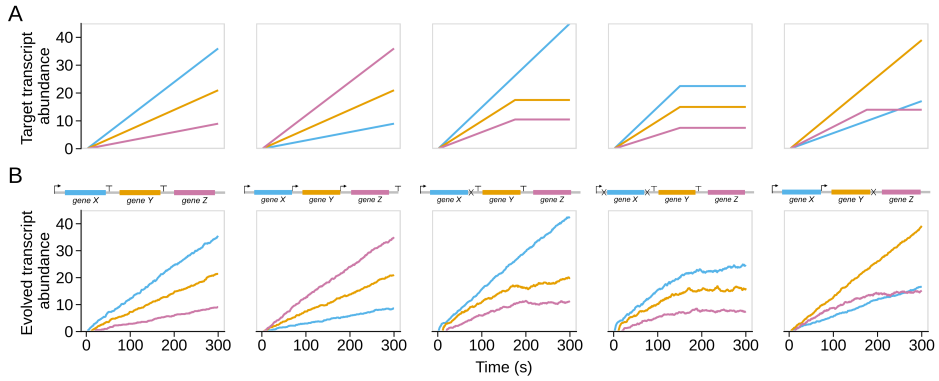

Figure 2: Examples of successfully evolved gene expression patterns. (A) Each panel contains a target gene expression time-course pattern that collectively span a diverse range of possibilities. (B) The corresponding representative gene expression patterns from successful simulations for each respective target. Shown above each time-course of gene expression is the evolved genome architecture that produced it.

After successfully conducting an evolutionary simulation with a positive control, we applied the same method to a range of gene expression patterns that have no *a priori* known genetic solution. We began with a simple pattern

where all three genes linearly increase in abundance over time but do so at different rates. A rational solution to this pattern might be to have a strong upstream promoter with weak terminators between each successive gene. An alternative solution would be to to start with a weak upstream promoter followed by subsequent promoters between each gene so that each successive gene will be expressed at a higher level. While in the first case that we outlined the *first* gene in the genome would be expressed at the highest rate, in this latter scenario the *last* gene would be expressed at the highest rate.

As we expected, the left two panels of Fig. 2B show that we were able to successfully evolve this linearly increasing pattern and that the evolved genome architectures match our rational expectation. The final three examples presented in Fig. 2 further show successful simulations (the lowest normalized-RMSE values found across 50 independent replicates) for more complex patterns that are difficult to rationalize solutions to. While this demonstrates that genomes in our system can evolve to match several distinct patterns with qualitatively successful solutions, we wanted to further increase the complexity of our targets and evaluate replicate simulations more quantitatively.

We conducted a total of 300 independent evolutionary simulations for each of 10 general patterns (Fig. 3A), split into 50 simulations for each of the 6 possible gene arrangements per pattern (Supplementary Fig. S1, see Materials and Methods). For the particular gene arrangement that yielded the best results for each pattern, Fig. 3B illustrates the distribution of normalized-RMSE values from the 50 replicate simulations (Supplementary Fig. S2 shows corresponding gene expression time-courses for the best replicate). Each pattern had at least two successful replicate simulations (normalized-RMSE $\leq$ 0.1), but there was nevertheless a clear difference between how easily solutions to particular patterns were able to evolve—compare and contrast patterns #5 and #6 in Fig 3B. This result suggests certain gene expression time-course patterns may have a more restrictive set of genome architectures that can produce them, but we also note that the evolutionary parameters could perhaps be altered to better optimize solutions to particular patterns.

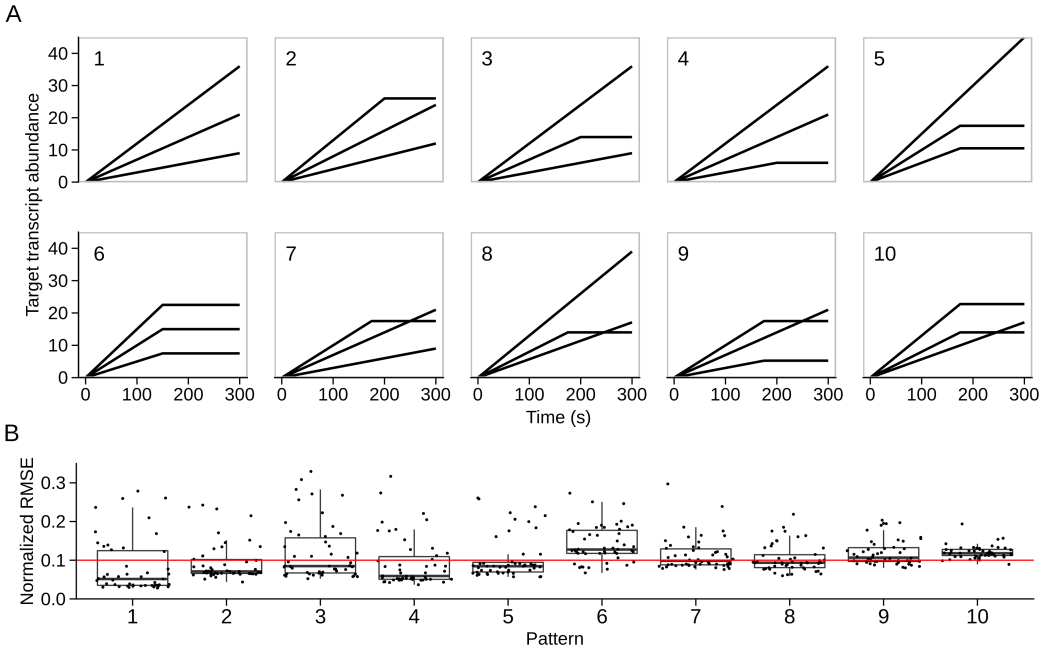

Figure 3: Quantitative assessment of pattern achievability. (A) The 10 target gene expression time-course patterns that we simulated are displayed as black lines representing general patterns (each of which has 6 possible gene arrangements). (B) For 50 replicate simulations of each pattern, shown are the lowest achieved normalized-RMSE values. All patterns had at least two successful independent evolutionary simulations, as determined by the red line highlighting a normalized-RMSE value of 0.1.

2.3 Disparate genome architectures can recapitulate the same target expression pattern

To investigate possible differences in achievability for different gene expression patterns, we interrogated the diversity of the successfully evolved genome architectures for each pattern. We did not discover a particular genome configuration that any individual pattern converged on. As illustrated in Fig. 4A, even the simplest pattern that we assessed had numerous distinct (albeit qualitatively similar) genome architectures that evolved, each of which were capable of producing gene expression time-courses that successfully matched the target. However, some patterns were more heterogeneous than others in terms of the number of possible genome architectures found (Fig. 4B).

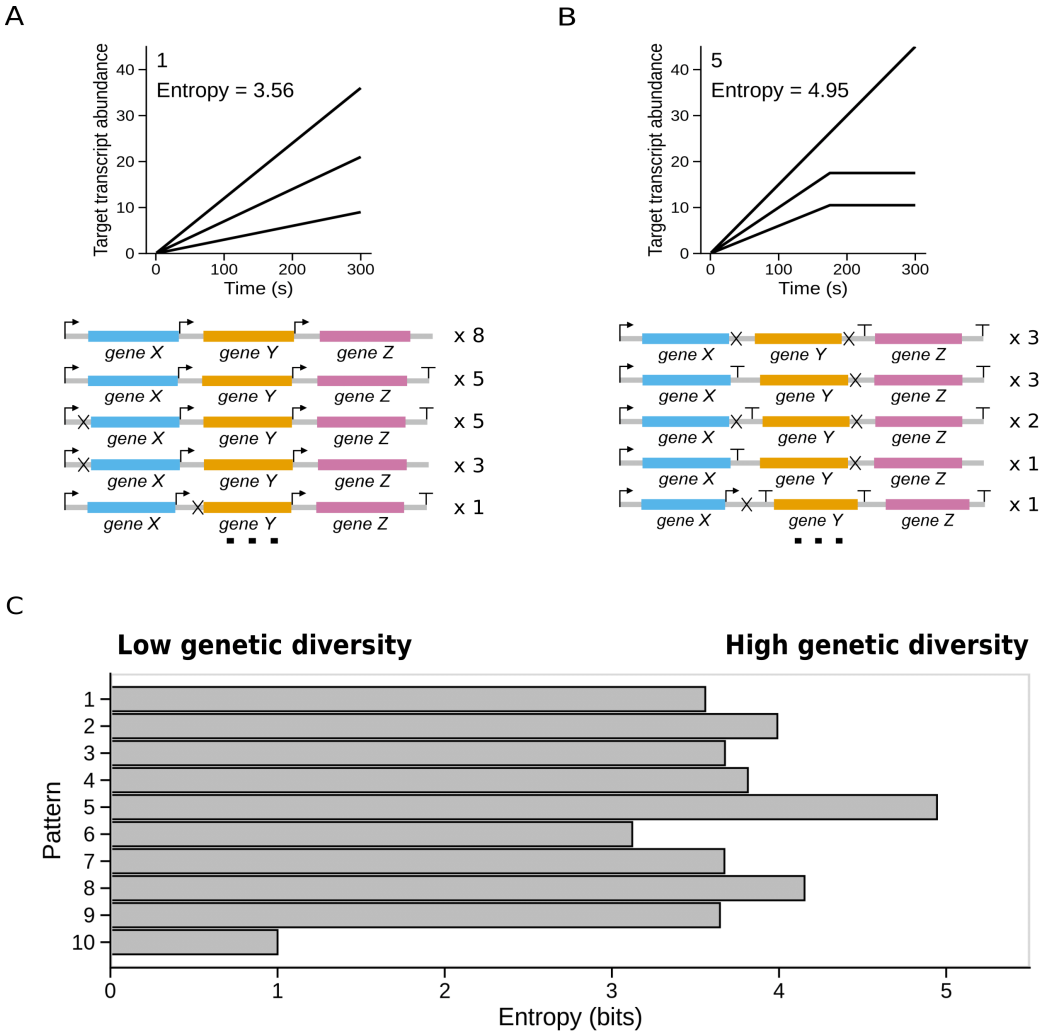

Figure 4: Diversity of successfully evolved genome architectures. (A) A sample of the successful genome architectures for pattern #1 showing that multiple potential architectures can reproduce the target expression pattern. (B) Similar to (A), a sample of successful architectures for pattern #5. (C) Entropy values for the set of successfully evolved genome architectures for each target pattern. Lower values indicate that successful replicate simulations converged upon one or a small number of genome architectures.

We quantified the diversity of evolved genome architectures for successful replicate simulations for each pattern using information entropy (choosing the best of the 6 possible gene arrangements for each pattern). Lower entropy values imply that only a single or a small number of distinct genome architectures evolved for a particular pattern. By contrast, high entropy values imply that the genome architecture solutions for a particular pattern were largely distinct from one another. Fig. 4C illustrates the entropy scores for each of the 10 patterns showing that pattern 5 had the most diverse set of genome solutions whereas pattern 10 (which had only two successful simulations) was the most constrained. Taken together, these results show that particular patterns may be more flexible than others in terms of the number of possible solutions but the cause of this variability is unknown at this stage and a possible avenue for future research.

2.4 Evolutionary simulation of a ten-gene model

Although the three-gene model is ideal for studying the basic principles behind genetic regulation, the size of this genome is impractically small compared to most phage genomes. Thus, we wanted to test whether our simulation approach was capable of evolving larger genomes containing up to 10 genes. As shown in Fig. 5, we achieved a qualitatively suitable result when attempting to simulate a simple linear pattern with variable rates for each gene within the genome. However, we note that increasing the number of genes within the genome also increases the run-time of the simulation substantially since more generations are required to adequately explore the fitness landscape given that there are many more possible mutations (*i.e.* regions between genes that can potentially contain regulatory elements). This result, nevertheless, shows that our approach is generalizable and can, in principle, be extended to any number of genes with complex predefined target patterns barring computational limitations.

3 Discussion

Genomes of all species employ a myriad of regulatory strategies to ensure that individual genes are expressed at particular levels that vary over time. Here, we have used computational modeling to demonstrate that bacteriophages can generate a range of complex and time-varying expression patterns with-

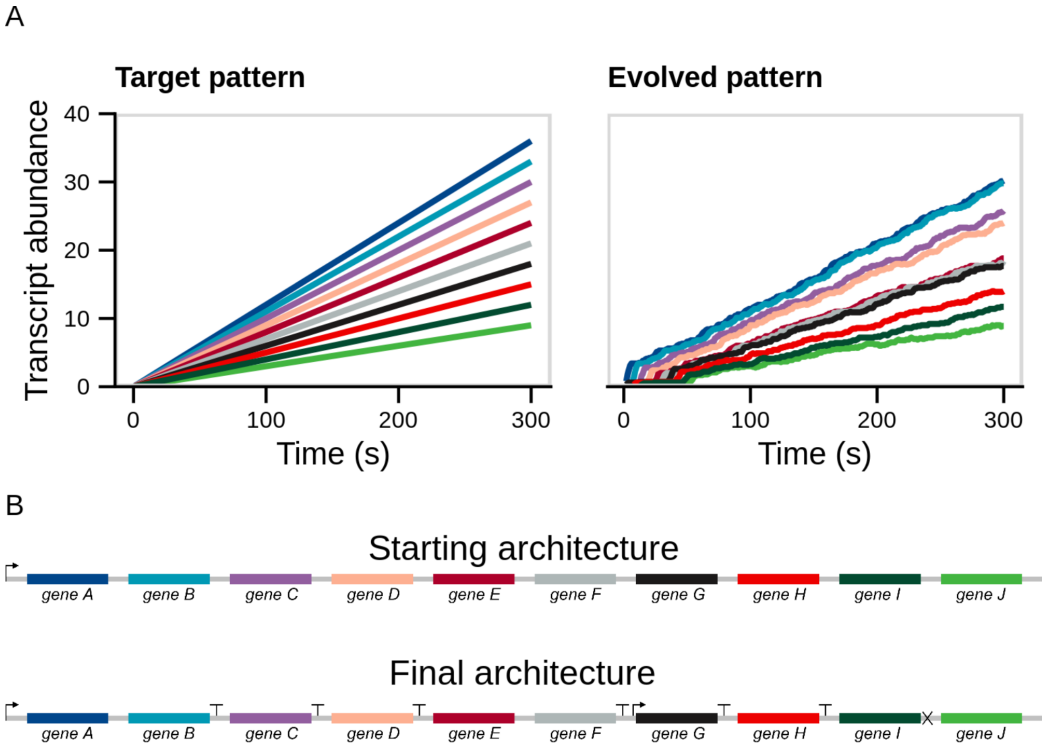

Figure 5: Evolutionary simulation of a ten-gene model. (*A*) Shown are the target (left) and evolved (right) patterns. The evolved pattern is the best of 5 independent replicates, each of which was simulated for 100,000 generations. (*B*) The starting (top) and best fitting evolved (bottom) genome architectures.

out the need for specific regulatory molecules or complex genetic circuitry. By varying only the strength of promoters, terminators, and RNase binding sites, our *in silico* evolution platform was able to generate genomes capable of matching numerous and distinct gene expression time-course patterns. These findings show that networks of interacting promoters and transcription factors are not explicitly necessary to achieve certain gene expression designs, and provide a proof-of-principle for the *de novo* design and engineering of phage genomes using molecular-level evolutionary simulation.

Phage genomes are typically small and highly gene dense, which is likely to limit the overall complexity of their native gene expression programs [33, 34]. The phage PhiX, for instance, encodes only 11 essential genes on

a genome that spans ≈5,000 base-pairs [35, 36]. Even with relatively small genomes, phages are nevertheless capable of producing complex gene expression dynamics over the course of an infection cycle. In phage T7, for instance, genes that are expressed early in the infection cycle are typically involved in shutting down synthesis of host-cell macromolecules, whereas the genes that are expressed later have roles in viral assembly and lysis [37, 38, 39]. While the T7 genome encodes its own polymerase that is critical for the delayed expression of these late genes, phage T7 gene expression dynamics are partially accomplished through variable strength regulatory features that are scattered throughout the genome [29, 40, 27, 41, 42, 28].

Our findings here highlight some of the strengths and the limitations of regulatory approaches built solely off of these simple elements. In addition to accumulating at different rates over time, gene products can be designed to plateau at different levels by tuning degradation rates. In the most complex cases that we investigated, a combination of variable production and degradation rates is sufficient to produce expression dynamics whereby the relative ordering of genes by abundance can switch over time (Fig. 3, patterns 7–10). However, the regulatory elements that we assessed do not provide a means to shut off the production of certain genes, which would enable decreases in absolute transcript abundances at particular times. Similarly, gene expression delays within this system are limited to short time-scales when the phage genome is still being injected; there is no apparent mechanism for inducing expression at a particular time without considering more complex genetic regulation.

Despite these constraints, our approach may provide important insight into the overall organization and regulatory principles governing phage genomes. For instance, RNase cleavage sites and promoter sequences frequently co-occur in phage T7 [28]. In our simulations, we observed numerous instances of this co-occurrence, but the generality and significance of this finding is difficult to assess based on the limited set of patterns that we investigated. Future work, particularly focused on larger genomes with more complex expression patterns might uncover interesting design principles that have heretofore gone unnoticed.

While our results provide insight into the limits and capabilities of phage regulatory strategies, our approach may also prove useful for synthetic biology applications in phages [43, 44]. Synthetic genetic circuits are typically designed for free-living species where they must operate in variable cellular contexts over comparatively long time-scales [45, 46]. By contrast, phage

infections are typically brief, with relevant time-scales on the order of tens of minutes [47, 48]. We performed our simulations with realistic cellular parameters over an initial 5 minute infection period. Producing more complicated dynamics from much larger genomes over longer timescales could, in principle, be accomplished using minimal genetic circuitry to divide the genome into early and late gene sets; within each of these two sets our results show that it is possible to encode diverse dynamics without any further regulatory mechanisms. Indeed, the phage T7 genome is partitioned in precisely this manner, having distinct classes of early, middle, and late expressed genes that perform distinct functions [37, 38, 39].

In principle, the approach that we presented here could be adapted to a design oriented-framework but doing so presents several challenges. While Pinetree relies on genome-level information, it does not make use of *sequence*-level information. Elements such as promoters are encoded solely by their location and strength—not by an explicit sequence. Translating a particular evolutionary design into an actual genome sequence would thus require using standardized parts with predefined element strengths, which could be calibrated against the values encoded in Pinetree [49]. In the case of RNase sites, this may be particularly challenging because while degradation rates are known to vary considerably across genes, the overall rules concerning RNase binding and cleavage are less well understood than the actions of promoters and terminators [50, 51].

We envision that the results presented here can be extended in a number of different directions in future research. First, we only considered non-overlapping gene arrangements in our study but many phage genomes are highly compact and successive genes frequently overlap; this feature might impact gene expression in ways that are difficult to predict but could be assessed using our approach [33, 23]. Second, an obvious way to create more complex gene expression dynamics would be to encode a regulatory protein on the phage genome rather than the arbitrary proteins that we have thus far investigated. Extensions of our framework to consider more genes would likely benefit from such an approach, which could allow for expression delays or abundance decreases. Third, we previously noted that gene order can play an important role in achieving particular targets. With only three genes we were able to enumerate all possible gene arrangements, but this approach is intractable for larger genomes and will require an additional mutational step that swaps the placement of genes for larger applications. Finally, we focused our attention on transcript abundances but differences in translation

initiation and elongation rates offer further ways to tune gene expression at the level of protein products to produce more complex dynamics [8, 7].

An extraordinary number of phages have been discovered in recent years, but only a small number of these have been interrogated experimentally [52, 53, 54]. Computational approaches can help to better characterize the general constraints and principles that affect genome organization and function, and can additionally provide a way to engineer phages based on predetermined design constraints. The design and engineering of whole phage genomes using principles and approaches that we developed here may have a number of future medical and biotechnological applications, including combating antibiotic resistant bacteria [55].

4 Materials and Methods

4.1 Pinetree simulations

In order to simulate a phage infection, Pinetree (v0.3.0) requires several pieces of information: i) the length of the genome, ii) the start and stop locations of each gene, and iii) the location, size, and strength of promoters, terminators, ribosome binding sites, and RNase cleavage sites. Additionally, Pinetree also requires a variety of cellular-level information about the infected bacteria: i) cell volume, ii) the speed, concentration, and footprint sizes of polymerases and ribosomes within a cell, and iii) the processing speed and footprint of RNase molecules. The overall amount of time that the infection process will be tracked for is a further free parameter, we simulated a 5 minute infection period throughout this manuscript but note that longer infection cycles and larger/more complex genomes will require longer run-times.

Some of the aforementioned parameters have either physiological or computational restrictions. To determine the effective ranges of individual elements, we simulated simplified genomes and measured the expression response of downstream genes (Supplementary Fig. S3) using a genome model containing only three coding sequences. Each coding sequence was 150 nucleotides long corresponding to 50 amino acid long proteins (none of which have a function within the simulation). The sizes of promoter binding sites (and thus polymerase footprints), terminators, RNase binding sites, and ribosome footprints were set to physiologically realistic values of 35, 30, 10, and 30 nucleotides long, respectively. This same genome configuration was

also used as the starting genome for evolutionary simulations described in this manuscript.

We note that RNase binding strengths were previously hard-coded in prior releases of Pinetree, effectively giving each transcript the same average degradation rate. In the latest release, we altered this behavior and now allow for site-specific rates to be specified for each RNase cleavage site. This rate can be thought of as a composite parameter that includes both the binding of the RNase molecule and enyzmatic cleavage rates (either of which may exhibit sequence specificity).

4.2 Evolutionary simulations

Our evolutionary simulations consist of a series of repeated calls to Pinetree using mutated phage genomes. In addition to all of the typical Pinetree parameters described above, evolutionary simulations also require: i) target gene expression data, ii) a starting genome information file, iii) the number of generations to run the simulation for, and iv) the number of replicate Pinetree simulations to average at each generation. Examples of all input files and simulation parameters are included within the code repository.

In our study, we began evolutionary simulations with a starting genome that consists of a single, moderate-strength promoter upstream of three equally sized coding sequences as described above. Mutations are proposed by selecting either i) addition, ii) removal, or iii) modification of existing regulatory element strengths from a uniform distribution of all possible choices. The addition and removal functions act as insertion and deletion mutations, respectively, whereas variable strength mutations are analogous to point mutations that alter specific rate parameters. Once a mutation is proposed, the new genome is simulated using Pinetree to obtain a gene expression timecourse. For each proposed mutation, we run 10 independent Pinetree simulations and average the results to guard against selecting for stochastic noise in individual Pinetree simulations. Results from these averaged simulations then determine whether to accept the given mutation.

4.3 Fitness calculation

We used a simplified origin-fixation model of evolution to calculate acceptance probabilities for individual mutations at discrete, non-overlapping generations, which relies on an accelerated algorithm to limit the number of

computationally expensive calls to the Pinetree platform [56]. Any mutation that increases fitness at generation i will be accepted whereas mutations that result in decreased fitness are exponentially unlikely to be accepted according to:

$$P_{\text{accept}} = e^{-2N_{\text{eff}}(x_{i-1}-x_i)} \tag{1}$$

where N_{eff} is the effective population size and x_i is the log-transformed fitness value:

$$x_i = \log(f_i) = -\log(e^{\beta(\text{nRMSE}_i)} + 1). \tag{2}$$

Here, β is analogous to a temperature value that allows us to tune the strength of selection and we calculate fitness directly from the normalized-root mean squared error (RMSE) at a given generation:

$$\text{RMSE} = \sqrt{\frac{\sum_{t=1}^{T}(y_t - \hat{y}_t)^2}{T}} \tag{3}$$

where T is the number of time steps in the gene expression time course, y_t is the gene abundance in the simulated data and $\hat{y}_t$ is the gene abundance in the target data at each time step. We calculate the RMSE separately for each gene, normalize these values by the mean target abundance (nRMSE):

$$\text{nRMSE} = \frac{\text{RMSE}}{\bar{\hat{y}}} \tag{4}$$

and finally average the nRMSE values across genes to arrive at the overall nRMSE for the given genome architecture used in Eq. (2). For more details on this evolutionary approach, see Ref. [56].

In the implemented fitness calculation, we fixed the effective population size at 1000 and varied β between 10e-3 and 1.3 across the 5,000 simulated generations. We did so in a piece-wise fashion where β begins at a value of 10e-3 for the first 10% of the generations to noisily explore the fitness landscape (Supplementary Fig. S4). The β value then linearly increases to a value of 1.1 until 90% of the generations are completed and finally increases to a value of 1.3 for the remaining generations of the simulation. This simulated annealing approach allows ensures that deleterious mutations will be less frequently accepted towards the end of the simulation compared to the beginning, and that individual simulations will largely settle on one genome architecture. Large changes to the aforementioned parameters may result in simulations that are either incapable of evolving towards higher

fitness (too many deleterious mutations are selected and fitness varies wildly from generation to generation to generation) or become stuck in local fitness optima (too few deleterious mutations are selected). Our goal in this design was not to replicate a true evolutionary process but rather simply to permit efficient exploration of a potentially rugged fitness landscape and maximize our chance of finding suitable genome architectures.

4.4 Creating target patterns of gene expression

We started our exploration of gene expression patterns by allowing for variable rates of production over an infection cycle lasting 300 seconds (5 minutes). We then introduced the concept of "plateaus" in the expression data for individual genes that occurs when the abundance of a particular transcript remains steady as the infection cycle progresses (*i.e.* production and degradation are balanced). We created patterns that had transcript abundances that plateaued comparatively early (150 seconds) and late (200 seconds) in the 300 second infection cycle.

4.5 Accounting for possible gene re-arrangements

For a simple case of a general pattern where one gene is expressed at a high rate, one gene at an intermediate rate, and one gene at a low rate the *identities* of the genes expressed at these different rates could vary. Which is to say, the first gene in genome could be selected to be expressed at either high, intermediate, or low levels. Similarly, given a particular choice for the first gene, the second gene can be expressed at one of two possible levels. All told, there are 6 possible combinations for matching the 3 genes in the genome to the 3 genes described in any general pattern. Rather than include gene rearrangement steps we simply simulated all 6 possibilities for every general pattern. We selected the gene arrangement that had the lowest average normalized-RMSE as the representative for each pattern depicted in Fig. 3.

4.6 Removing elements with insignificant functions

When the evolutionary simulation settles on a particular genome architecture, we wanted to verify that each element contained within that architecture had a significant effect on the overall expression pattern. If the strength of any individual element was modified to a value where it had essentially

no effect, the element should be removed to permit comparison between architectures (as in Fig. 4). To evaluate each element's effect, we first removed that element from the genome, simulated the mutated genome using Pinetree, and compared the simulated expression pattern to the expression pattern produced with the element in place. If the normalized-RMSE value increased by an insubstantial value (using a threshold of 0.1) or if the value decreased, then we would add it to a list of elements to potentially be removed. Once a list of *removable* elements were compiled, we removed the element whose removal had the least impact. We then repeated this process, greedily removing elements one by one until removal of an element had a substantial effect and thus halted the process.

4.7 Calculating the entropy of genome solutions from replicate simulations

Each target pattern that we simulated resulted in at least two, and sometimes many, independently evolved genomes that were capable of matching it. To quantify the diversity of the resulting genome architecture solutions, we used a measure of information entropy. We first converted each genome architecture into a string representing the presence / absence of each element along the genome (regardless of strength) and counted the number of times each representation evolved (using only the successful simulations). Then, for each of the n observed patterns, we converted these frequencies into probabilities (p_i) and calculated the entropy of the architectures for a particular pattern as:

$$S = -\sum_{i=1}^{n}(p_i * \log_2(p_i)). \tag{5}$$

The resulting units of entropy (when calculated with base 2) are expressed in "bits".

4.8 Data and code availability

All data and code required to recreate the findings of this manuscript can be found at: `https://github.com/SahilBShah/pinetree-evolution`.

5 Acknowledgments

This work was supported by National Institutes of Health grants F32 GM130113 to A.J.H. and R01 GM088344 to C.O.W. The authors additionally acknowledge support from the Texas Advanced Computing Center (TACC) at The University of Texas at Austin for providing high-performance computing resources and the TIDES research fellowship for providing support to S.B.S.

References

[1] Dekel, E. & Alon, U. Optimality and evolutionary tuning of the expression level of a protein. *Nature* **436**, 588–592 (2005).

[2] Lalanne, J.-B. *et al.* Evolutionary convergence of pathway-specific enzyme expression stoichiometry. *Cell* **173**, 749–761.e38 (2018).

[3] Buccitelli, C. & Selbach, M. mRNAs, proteins and the emerging principles of gene expression control. *Nature Reviews Genetics* **21**, 630–644 (2020).

[4] López-Maury, L., Marguerat, S. & Bähler, J. Tuning gene expression to changing environments: from rapid responses to evolutionary adaptation. *Nature Reviews Genetics* **9**, 583–593 (2008).

[5] Bartl, M., Kötzing, M., Schuster, S., Li, P. & Kaleta, C. Dynamic optimization identifies optimal programmes for pathway regulation in prokaryotes. *Nature Communications* **4**, 2243 (2013).

[6] Liu, Y., Beyer, A. & Aebersold, R. On the dependency of cellular protein levels on mRNA abundance. *Cell* **165**, 535–550 (2016).

[7] Mutalik, V. K. *et al.* Precise and reliable gene expression via standard transcription and translation initiation elements. *Nature Methods* **10**, 354–360 (2013).

[8] Kosuri, S. *et al.* Composability of regulatory sequences controlling transcription and translation in Escherichia coli. *Proceedings of the National Academy of Sciences* **110**, 14024–14029 (2013).

[9] Hausser, J., Mayo, A., Keren, L. & Alon, U. Central dogma rates and the trade-off between precision and economy in gene expression. *Nature Communications* **10**, 68 (2019).

[10] Temme, K., Zhao, D. & Voigt, C. A. Refactoring the nitrogen fixation gene cluster from Klebsiella oxytoca. *Proceedings of the National Academy of Sciences* **109**, 7085–7090 (2012).

[11] Eames, M. & Kortemme, T. Cost-benefit tradeoffs in engineered lac operons. *Science* **336**, 911–915 (2012).

[12] Brophy, J. A. N. & Voigt, C. A. Principles of genetic circuit design. *Nature Methods* **11**, 508–520 (2014).

[13] Nielsen, A. A. K. *et al.* Genetic circuit design automation. *Science* **352** (2016).

[14] Schwander, T., Borzyskowski, L. S. v., Burgener, S., Cortina, N. S. & Erb, T. J. A synthetic pathway for the fixation of carbon dioxide in vitro. *Science* **354**, 900–904 (2016).

[15] Espah Borujeni, A., Zhang, J., Doosthosseini, H., Nielsen, A. A. K. & Voigt, C. A. Genetic circuit characterization by inferring RNA polymerase movement and ribosome usage. *Nature Communications* **11**, 5001 (2020).

[16] Hutchison, C. A. *et al.* Design and synthesis of a minimal bacterial genome. *Science* **351** (2016).

[17] Ostrov, N. *et al.* Design, synthesis, and testing toward a 57-codon genome. *Science* **353**, 819–822 (2016).

[18] Richardson, S. M. *et al.* Design of a synthetic yeast genome. *Science* **355**, 1040–1044 (2017).

[19] Breuer, M. *et al.* Essential metabolism for a minimal cell. *eLife* **8**, e36842 (2019).

[20] Chan, L. Y., Kosuri, S. & Endy, D. Refactoring bacteriophage T7. *Molecular Systems Biology* **1**, 2005.0018 (2005).

[21] Lu, T. K. & Collins, J. J. Engineered bacteriophage targeting gene networks as adjuvants for antibiotic therapy. *Proceedings of the National Academy of Sciences* **106**, 4629–4634 (2009).

[22] Ghosh, D., Kohli, A. G., Moser, F., Endy, D. & Belcher, A. M. Refactored M13 bacteriophage as a platform for tumor cell imaging and drug delivery. *ACS Synthetic Biology* **1**, 576–582 (2012).

[23] Jaschke, P. R., Lieberman, E. K., Rodriguez, J., Sierra, A. & Endy, D. A fully decompressed synthetic bacteriophage øX174 genome assembled and archived in yeast. *Virology* **434**, 278–284 (2012).

[24] Pires, D. P., Cleto, S., Sillankorva, S., Azeredo, J. & Lu, T. K. Genetically engineered phages: a review of advances over the last decade. *Microbiology and Molecular Biology Reviews* **80**, 523–543 (2016).

[25] Leuven, J. T. V. *et al.* phiX174 attenuation by whole genome codon deoptimization. *bioRxiv* 2020.02.10.942847 (2020).

[26] Amir, A., Kobiler, O., Rokney, A., Oppenheim, A. B. & Stavans, J. Noise in timing and precision of gene activities in a genetic cascade. *Molecular Systems Biology* **3**, 71 (2007).

[27] Jack, B. R. *et al.* Reduced protein expression in a virus attenuated by codon deoptimization. *G3: Genes, Genomes, Genetics* **7**, 2957–2968 (2017).

[28] Jack, B. R., Boutz, D. R., Paff, M. L., Smith, B. L. & Wilke, C. O. Transcript degradation and codon usage regulate gene expression in a lytic phage. *Virus Evolution* **5** (2019).

[29] Endy, D., Kong, D. & Yin, J. Intracellular kinetics of a growing virus: A genetically structured simulation for bacteriophage T7. *Biotechnology and Bioengineering* **55**, 375–389 (1997).

[30] Endy, D., You, L., Yin, J. & Molineux, I. J. Computation, prediction, and experimental tests of fitness for bacteriophage T7 mutants with permuted genomes. *Proceedings of the National Academy of Sciences* **97**, 5375–5380 (2000).

[31] Kosuri, S., Kelly, J. R. & Endy, D. TABASCO: A single molecule, base-pair resolved gene expression simulator. *BMC Bioinformatics* **8**, 480 (2007).

[32] Jack, B. R. & Wilke, C. O. Pinetree: a step-wise gene expression simulator with codon-specific translation rates. *Bioinformatics* **35**, 4176–4178 (2019).

[33] Brandes, N. & Linial, M. Gene overlapping and size constraints in the viral world. *Biology Direct* **11**, 26 (2016).

[34] Wright, B. W., Ruan, J., Molloy, M. P. & Jaschke, P. R. Genome modularization reveals overlapped gene topology is necessary for efficient viral reproduction. *bioRxiv* 2020.06.10.143693 (2020).

[35] Jaschke, P. R., Dotson, G. A., Hung, K. S., Liu, D. & Endy, D. Definitive demonstration by synthesis of genome annotation completeness. *Proceedings of the National Academy of Sciences* **116**, 24206–24213 (2019).

[36] Logel, D. Y. & Jaschke, P. R. A high-resolution map of bacteriophage phiX174 transcription. *Virology* **547**, 47–56 (2020).

[37] McAllister, W. T. & Barrett, C. L. Roles of the Early Genes of Bacteriophage T7 in Shutoff of Host Macromolecular Synthesis. *Journal of Virology* **23**, 543–553 (1977).

[38] McAllister, W. T. & Wu, H. L. Regulation of transcription of the late genes of bacteriophage T7. *Proceedings of the National Academy of Sciences* **75**, 804–808 (1978).

[39] Springman, R., Badgett, M. R., Molineux, I. J. & Bull, J. J. Gene order constrains adaptation in bacteriophage T7. *Virology* **341**, 141–152 (2005).

[40] Cecchini, N., Schmerer, M., Molineux, I. J., Springman, R. & Bull, J. J. Evolutionarily stable attenuation by genome rearrangement in a virus. *G3: Genes, Genomes, Genetics* **3**, 1389–1397 (2013).

[41] Garry, D. J., Ellington, A. D., Molineux, I. J. & Bull, J. J. Viral attenuation by engineered protein fragmentation. *Virus Evolution* **4** (2018).

[42] Paff, M. L., Jack, B. R., Smith, B. L., Bull, J. J. & Wilke, C. O. Combinatorial approaches to viral attenuation. *mSystems* **3** (2018).

[43] Lemire, S., Yehl, K. M. & Lu, T. K. Phage-Based Applications in Synthetic Biology. *Annual Review of Virology* **5**, 453–476 (2018).

[44] Lammens, E.-M., Nikel, P. I. & Lavigne, R. Exploring the synthetic biology potential of bacteriophages for engineering non-model bacteria. *Nature Communications* **11**, 5294 (2020).

[45] Elowitz, M. B. & Leibler, S. A synthetic oscillatory network of transcriptional regulators. *Nature* **403**, 335–338 (2000).

[46] Gardner, T. S., Cantor, C. R. & Collins, J. J. Construction of a genetic toggle switch in Escherichia coli. *Nature* **403**, 339–342 (2000).

[47] Wang, I.-N. Lysis Timing and Bacteriophage Fitness. *Genetics* **172**, 17–26 (2006).

[48] Shao, Y. & Wang, I.-N. Bacteriophage Adsorption Rate and Optimal Lysis Time. *Genetics* **180**, 471–482 (2008).

[49] Hossain, A. *et al.* Automated design of thousands of nonrepetitive parts for engineering stable genetic systems. *Nature Biotechnology* 1–10 (2020).

[50] Venturelli, O. S. *et al.* Programming mRNA decay to modulate synthetic circuit resource allocation. *Nature Communications* **8**, 15128 (2017).

[51] Rousseau, B. A., Hou, Z., Gramelspacher, M. J. & Zhang, Y. Programmable RNA Cleavage and Recognition by a Natural CRISPR-Cas9 System from Neisseria meningitidis. *Molecular Cell* **69**, 906–914.e4 (2018).

[52] Roux, S., Hallam, S. J., Woyke, T. & Sullivan, M. B. Viral dark matter and virus–host interactions resolved from publicly available microbial genomes. *eLife* **4**, e08490 (2015).

[53] Paez-Espino, D. *et al.* Uncovering Earth's virome. *Nature* **536**, 425–430 (2016).

[54] Nayfach, S. *et al.* A genomic catalog of Earth's microbiomes. *Nature Biotechnology* 1–11 (2020).

[55] Dedrick, R. M. *et al.* Engineered bacteriophages for treatment of a patient with a disseminated drug-resistant Mycobacterium abscessus. *Nature Medicine* **25**, 730–733 (2019).

[56] Teufel, A. I. & Wilke, C. O. Accelerated simulation of evolutionary trajectories in origin-fixation models. *Journal of The Royal Society Interface* **14**, 20160906 (2017).

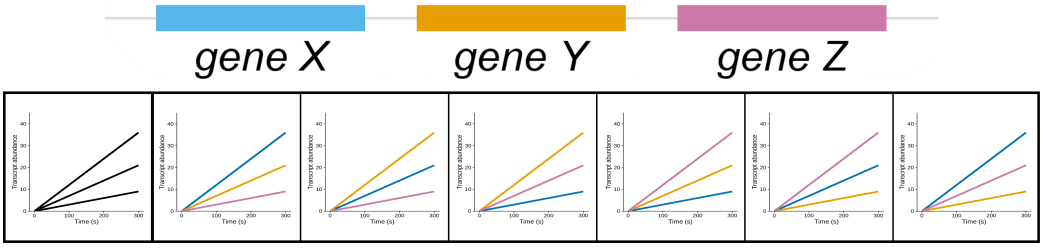

Figure S1: The 6 possible gene arrangements are shown for general pattern. Each gene is denoted by the colors shown in the genome above: blue representing gene X, orange for gene Y, and purple for gene Z. The need for simulating 6 gene arrangements per general pattern arises because genomic ordering of the elements may be important and our simulation does not currently include a mutational step to swap the identity of individual elements on the genome (as might happen when recombination occurs).

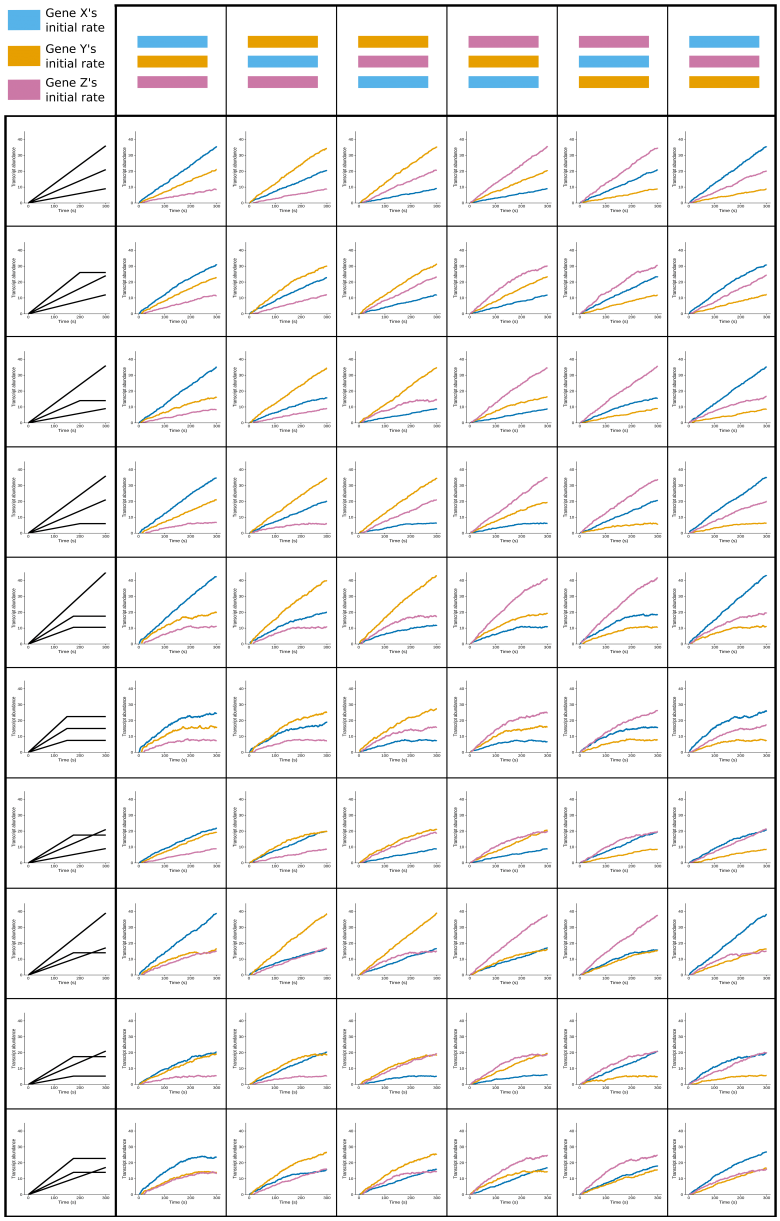

Figure S2: The best simulated expression time-courses found per target pattern. The y-axis contains each of the general patterns shown in Fig. 3 and the x-axis shows the 6 possible gene arrangements for each pattern. Each of the patterns have a normalized-RMSE value below 0.1.

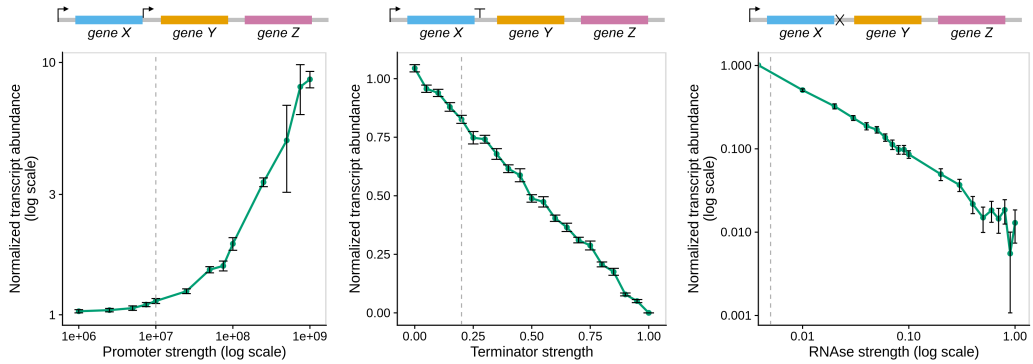

Figure S3: Analyzing the effective range of individual element strengths. Each plot corresponds to one of the three elements in which their effects at differing strengths were analyzed by comparing the ratio of Gene Y over Gene X (normalized transcript abundances). The genome architectures above the plots show the element's placement when analyzing their strengths. The grey dashed line denotes the strength at which we decided to insert each respective element in our evolutionary simulations when an "add" mutation is proposed.

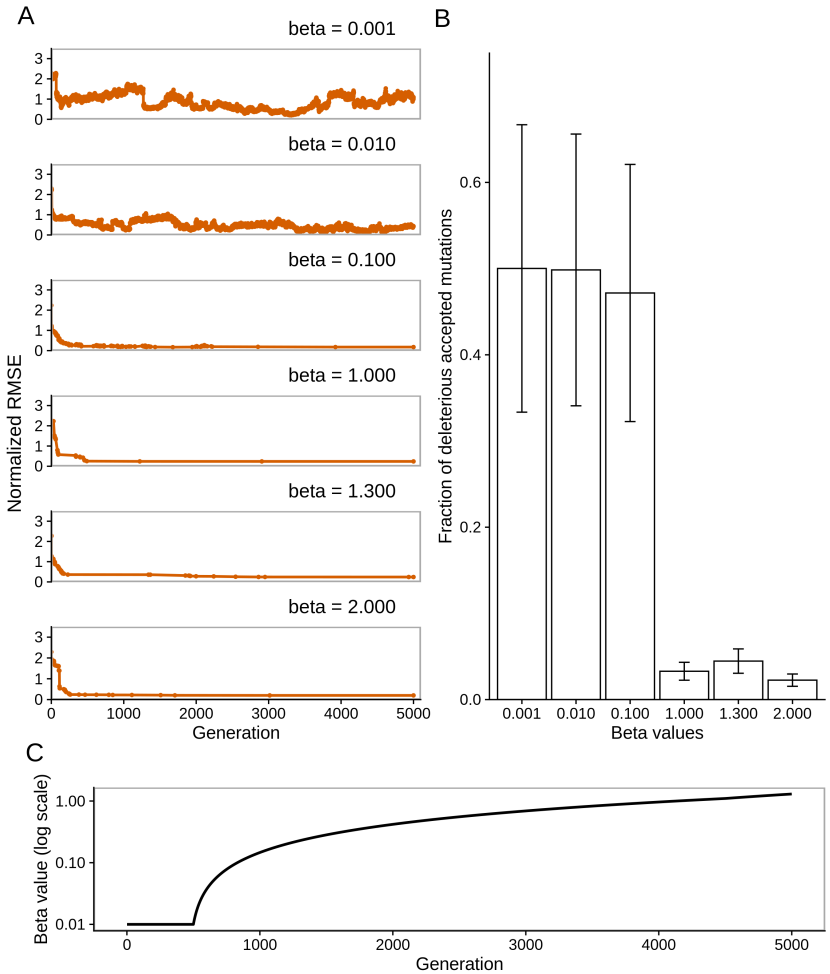

Figure S4: Influence of β parameter on the strength of selection. (A) We fixed effective population size at 1000 and ran 5,000 generation simulations with the β value varying from 0.001 to 2.000. Each plots shows representatives from 10 simulations of pattern number 1. (B) We summarized subplot (A) by plotting the fraction of accepted mutations that were deleterious for each β value. (C) We settled on varying β over the course of each simulation, with this plot showing the piecewise function where β remains constant for the first 10% of generations, linearly increases until 90% of the generations have passed, and then linearly increases at a different rate for the final 10% of generations.