Evaluating the Contribution of Model Complexity in Predicting Robustness in Synthetic Genetic Circuits

Lukas Buecherl,[*,†] Chris J. Myers,[*,‡] and Pedro Fontanarrosa[*,‡]

†*Department of Biomedical Engineering, University of Colorado, Boulder, 80309, Colorado, United States*

‡*Department of Electrical, Computer, and Energy Engineering, University of Colorado, Boulder, 80309, Colorado, United States*

E-mail: lukas.buecherl@colorado.edu; chris.myers@colorado.edu; pfontanarrosa@gmail.com

Abstract

The design-build-test-learn workflow is pivotal in synthetic biology, as it seeks to broaden access to diverse levels of expertise and enhance circuit complexity through recent advancements in automation. The design of complex circuits depends on developing precise models and parameter values for predicting circuit performance and noise resilience. However, obtaining characterized parameters under diverse experimental conditions is a significant challenge, often requiring substantial time, funding, and expertise. This paper compares five computational models of three different genetic circuit implementations of the same logic function to evaluate their relative predictive capabilities. The primary focus is on determining whether simpler models can yield similar conclusions to more complex ones and whether certain models offer greater analytical benefits. These models explore the influence of noise, parameterization, and model complexity on predictions of synthetic circuit performance through simulation.

The findings suggest that when developing a new circuit without characterized parts or an existing design, any model can effectively predict the optimal implementation by facilitating qualitative comparison of designs' failure probabilities (e.g., higher or lower). However, when characterized parts are available and accurate quantitative differences in failure probabilities are desired, employing a more precise model with characterized parts becomes necessary, albeit requiring additional effort.

Keywords: Genetic Regulatory Networks, Synthetic Genetic Circuits, Extrinsic and Intrinsic Noise in Gene Expression, Synthetic Biology, Genetic Circuit Simulation and Modeling, Comparative Analysis of Genetic Models

Synthetic biology aims to integrate the engineering *design-build-test-learn* (DBTL) workflow to create new systems with defined functions and predictable behavior.[1] In recent years, focus has been on automating the DBTL workflow using software to accelerate and improve the implementation of genetic circuits, aiming for faster commercialization cycles.[2] However, classical synthetic biology has primarily focused on testing genetic systems in ideal laboratory conditions, neglecting the impact of variable and noisy environments. As we transition from proof-of-concept designs to real-life applications, such as therapeutics production, understanding and accounting for environmental and noise effects on circuit performance (referred to as *robustness*) becomes crucial for ensuring correct and safe behavior of these genetic circuits.[3,4] Therefore, it is essential to study and accurately predict the likelihood of faulty behavior, in order to prevent irreversible harmful effects. The "learn" step is conventionally associated with the development of models aimed at achieving predictions with improved accuracy and precision. However, this stage often relies heavily on intuition-driven exploration rather than an automated and reproducible procedure. The integration of mathematical models, which can operate independently of the "learn" step, enables reliable analysis, streamlining the DBTL cycle, and ultimately conserving time and resources.

With mathematical descriptions of genetic networks, genetic design is moving towards a model-driven approach.[5] Re-parameterizing genetic parts with experiments produces more

accurate and precise behavior predictions. However, extensive training in investigating multi-dimensional design spaces and mapping simulations to experiments is required for model development and exploration as well as the parameterization of re-characterized gates.[6] Therefore, the current state of modeling gene network dynamics is characterized by a trade-off between the model's ability to quantitatively match the experimental data and the need for a large number of kinetic parameters to parameterize the model.[7–10] Properly parameterized *ordinary differential equation* (ODE) models can provide a good quantitative match and are easily generalized.[11–14] However, more detailed models also require a more challenging characterization effort. Furthermore, there is a simulation-time cost associated with these models: the more complex a model is, the longer it takes for simulations to run and the higher the memory requirements for it to be model-checked.[15–17]

While the deterministic structure of ODE analysis serves well in depicting the average behavior of a system, it lacks the element of randomness or stochasticity, consistently yielding identical results for the same initial conditions.[18] Nevertheless, the unpredictable nature of biochemical reactions, even at the level of a single gene,[19] combined with the variability in reaction rates due to environmental differences,[20,21] infuses a degree of uncertainty into a genetic system.[20,22–24] These fluctuations can profoundly impact the robustness and predictability of a system, making the stochastic analysis of *gene regulatory networks* (GRNs) an essential consideration in modeling GRNs.

Moreover, for systems where transcription factors, enzymes, and DNA copies might exist in low concentrations, such as a single molecule per cell, a realistic evaluation of these systems should include cell-to-cell variability in transcriptional outputs, as captured by the chemical master equation (i.e., *intrinsic noise*).[25] However, for systems with robust genetic expression, stochastic variations in gene expression, like those in transcription factor-producing genes, may stem from fluctuations in the quantity or states of other cellular components (i.e., *extrinsic noise*).[21,23] These are best modeled as probability density functions for different reaction rates.[20] Yet, the challenges of parameterizing these factors (either intrinsic or

extrinsic noise) may deter many scientists.

The question then arises: In terms of robustness and predictability, how much additional information can computational models and simulations involving intrinsic or extrinsic noise provide? Certainly, precise and predictive models are invaluable assets, but how much additional insight do detailed models, coupled with meticulously characterized components, genuinely provide regarding the robustness of a genetic circuit? When the primary concern is the resilience and stability of a circuit, is it justifiable to allocate substantial resources for exhaustive characterization experiments and computationally demanding simulations? This paper delves into these critical inquiries, striving to ascertain whether such investment yields a commensurate return in enhancing our understanding of genetic circuit robustness.

This study evaluates how various parameter characterizations and models influence the predictability of robustness and, consequently, circuit design decisions. The aim is to determine the effort required for parameter determination and model development to qualitatively understand relative robustness across different design choices. This research compares different interaction models, customized with experimentally-obtained parameters or utilizing standard parameters derived from literature averages. It also considers two noise sources—intrinsic and extrinsic noise—to anticipate the robustness of three distinct circuit implementations of genetic circuit 0x8E, as published by Nielsen et al.[26] This study investigates whether there are variations in predicted circuit behavior among three different circuit layouts that share identical expected functions. Additionally, it aims to determine if the various models agree on identifying the most robust design choice. The results illuminate differences in model predictions, determining the feasibility of analyzing less complex models to reliably evaluate a circuit's behavior *in silico*.

Results and Discussion

Genetic Circuit Failure Modes. Genetic circuits function within the cell by updating their internal states and output in response to changes in their inputs. These transitions from one input state to another are known as input transitions. However, during the operation of genetic circuits, various failures can occur.

One possible failure arises when circuits manifest undesirable behavior during transitions between states. When the circuit transitions from one state to another, the internal signals must adapt to the new state. However, during this transition, the circuit may exhibit an unexpected output before reaching the intended final state. Since this failure is observable only during the transition, it is categorized as *transient behavior*. The transient behaviors illustrated in Figure 1(a) and (b) manifest as undesired switching events known as *glitches*, arising from *hazards* inherent in the circuit function or implementation. A hazard signifies the potential for an undesirable effect to occur, stemming from either the circuit design or external influences. While a hazard merely denotes the possibility of such a failure, a glitch signifies the actual occurrence of the failure event.

For example, during an input transition from a state with a high output to another state with a high output (e.g., *static* $1 \rightarrow 1$), the output briefly turns low before reverting to a high state, as depicted in the red shaded area in Figure 1(a) resulting in a *static* $1 \rightarrow 1$ hazard. In a correct circuit behavior, the output should consistently maintain the high state throughout the transition, as indicated by the dashed line in the figure. Similarly, in a *static* $0 \rightarrow 0$ transition, the circuit's output is expected to stay low. However, during the transition, the output briefly turns high before settling to the anticipated low output, as illustrated in Figure 1(b) resulting in a *static* $0 \rightarrow 0$ *hazard*.

It is important to note that glitches are characteristic of transient failures, implying that the failure self-corrects over time. However, if the output is irreversible, the circuit's functionality may be compromised. For example, in cases where an output change triggers events such as apoptosis or the release of a therapeutic drug, even a brief incorrect state can

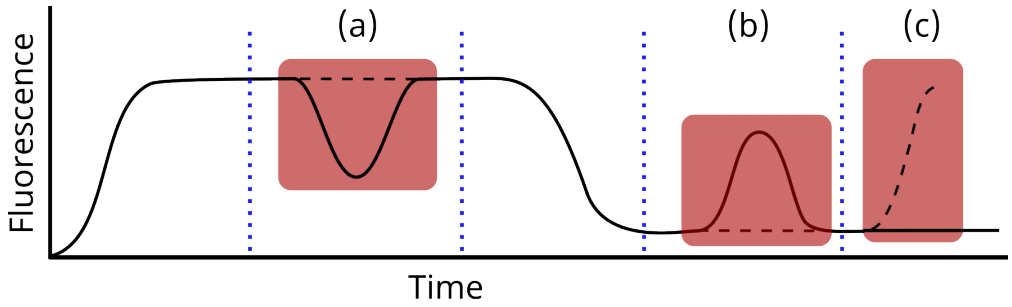

Figure 1: This graph shows the example output of a genetic circuit in fluorescence over time. The dashed blue lines indicate an input transition. The dashed black line is the expected output for that transition and the solid black line is the one observed. (a) Shows an unexpected switching behavior after an input transition. Both before and after the first input change, the output is supposed to remain high. However, the signal briefly turns low before recovering and turning high again (static $1 \rightarrow 1$ hazard). (b) Shows an unwanted switching behavior after another input transition. This time, the circuit is supposed to stay low. During the transition, the circuit briefly turns high again before finally settling in low the signal (static $0 \rightarrow 0$ hazard). The behavior in both (a) and (b) correct themselves over time and are therefore considered transient. Finally, (c) shows a steady state failure. While the circuit is supposed to turn high again after the final input change, it remains low, settling in the wrong steady state.

lead to irreversible harmful effects.

Another notable failure occurs when a circuit fails to produce the expected output, displaying unexpected and sustained behavior that does not self-correct. These failures relate to the circuit's *steady-state behavior*. The state of a circuit is determined by the arrangement of its internal molecules, representing the status of signals governing the circuit as either high (on) or low (off). The term "steady state" refers to a condition in which the circuit has reached equilibrium and remains unchanged over time. A steady-state failure indicates a scenario in which the circuit's output deviates from the expected output for a given state. In other words, the output remains off when it should be on, or vice versa. Such failures can occur when the circuit lacks the ability to effectively differentiate between on and off signals. The third behavior mode, steady-state failure, is depicted in Figure 1(c). Following the last input transition in this example, the output is expected to transition to a high state. However, it remains in an incorrect steady state, staying low instead. It is important to note that in cases where the circuit is intended to remain low but instead settles in a high state,

it is also considered a steady-state failure.

Based on the presented failures, the predictability of five computational models is assessed by comparing the predicted likelihood of failure returned by the analysis of each model. Specifically, the three failure modes analysed are:

1. Probability of glitches caused by noise
2. Probability of glitches caused by hazards
3. Probability of steady-state failures

The first investigated behavior mode involves the occurrence of glitches attributed to function hazards. Nearly all circuits have transitions that produce function hazards. The only exception would be very simple circuits, such as an inverter or buffer. Function hazard glitches arise when multiple input changes transpire simultaneously, and the propagation delay of the input changes through the circuit lead the circuit to resolve temporarily to an incorrect state. They are inherent to the logic function of the circuit and thus unavoidable.

The second investigated behavior mode involves glitches arising from the circuit's inherent noisy behavior. These transitions do not have function hazards and are not inherently faulty, thus anticipated to behave as expected. However, due to noise, there remains a small probability of temporary deviation from the expected behavior. In this paper, we have examined the probability of failure for both types of transitions. All selected transitions are *static*, meaning the circuit moves either from a state with a low output to another state with a low output or from a high state to another high state. Both the failures of transitions with and without function hazards are observed as the glitches depicted in Figure 1(a) and (b). However, transitions with known function hazards are expected to fail with a higher probability compared to their function-hazard-free counterparts, as they are inherently faulty.

The third failure mode involves analyzing a genetic circuit's steady state. It follows the failure mode shown in Figure 1(c), and aims to determine whether the circuit settles in its expected output state.

It is important to acknowledge the existence of other potential failure modes, one of which involves *logic hazards*. Unlike function hazards, which result in glitches due to the implemented function, logic hazards stem from the layout of the circuit's logic rather than its function. Another type of failure mode is dynamic hazards, where the circuit transitions from a high state to a low state or vice versa in a non-monotonic manner. For example, the output may shift from high to low, then back to high before ultimately settling in the low state.

The analysis presented here, however, focuses on static hazard analysis rather than dynamic hazards in simulations for two main reasons. First, detecting dynamic hazards can be challenging, and assessing their impact on the overall circuit behavior may be difficult. In contrast, static hazards are more straightforward to identify and can be reliably predicted through simulations. Second, in biological circuits, dynamic hazards are generally of lesser concern because biological systems often tolerate early activation or deactivation of circuit components. For instance, if the circuit output transitions from low to high earlier than expected, it may not significantly impact the biological system. Consequently, this work prioritizes static hazard analysis in simulations, providing valuable insights into a genetic circuit's overall behavior.

Circuit 0x8E. The work presented here investigates the behavior of a combinational genetic circuit first published by Nielsen et al.[26] The genetic circuit was labeled 0x8E and its *original design* can be seen in Figure 3(a). This circuit can detect three input molecules: *Arabinose* (Ara), *anhydrotetracycline* (aTc), and *Isopropyl β-D-1-thiogalactopyranoside* (IPTG) with the three input sensor promoters pBAD, pTet, and pTac, respectively. The output is reported through the production of *yellow fluorescent proteins* (YFP). The circuit's logic function is illustrated in the truth table provided in Table 2. Out of the eight states, four result in a high output, while the other four generate a low output. When these outputs are translated into binary, they form the sequence *10001110*. Converting this binary sequence to hexadecimal results in *8E*, which serves as the circuit's name.

Inputs			Output
pBAD	pTet	pTac	YFP
0	0	0	1
1	0	0	0
0	1	0	0
1	1	0	0
0	0	1	1
1	0	1	1
0	1	1	1
1	1	1	0

Figure 2: The truth table depicts the behavior of circuit 0x8E. The circuit comprises three input promoters—pBAD, pTet, and pTac— and YFP serves as the output. The binary representation of the output, *10001110*, when converted to hexadecimal, yields *8E*, which defines the circuit's name.

Fontanarrosa et al.[27] identified the different input transitions of circuit 0x8E that resulted in a glitching behavior. Based on this information, Fontanarrosa et al. designed two additional implementations of the circuit with the same logic function, but different logic gate combinations to improve the circuit's glitching behavior. The two modified layouts can be seen in Figure 3(b) and (c). The layout shown in Figure 3(b), the *two-inverter* layout, has redundant logic as two NOT-gates, were added to delay the IPTG pathway. The layout shown in Figure 3(c), the *non-logic-hazard* layout, was created by using hazard-preventing optimization methods to avoid introducing logic hazards.

Buecherl et al.[28] built on those results using stochastic simulation and stochastic model checking to determine the probability of the glitches discovered by Fontanarrosa et al.. This research leverages the identified glitching transitions alongside fault-free transitions to assess the predictive power of various mathematical models through forecasting the robustness of the three distinct designs. Since all three circuit implementations used in this work perform the same function, they share identical function hazards, making them suitable for comparison. However, the three implementations do not share the same logic hazards.

Robustness and Predictability. Engineered systems are expected to correctly function in different environments. *Robustness* is a system's ability to withstand and operate under the effects of external disturbances. Like in other engineering disciplines, synthetic biologists have to keep robustness in mind when designing a genetic circuit, especially for out-of-the-lab applications. Influences like noise can impact a circuit's behavior and thus

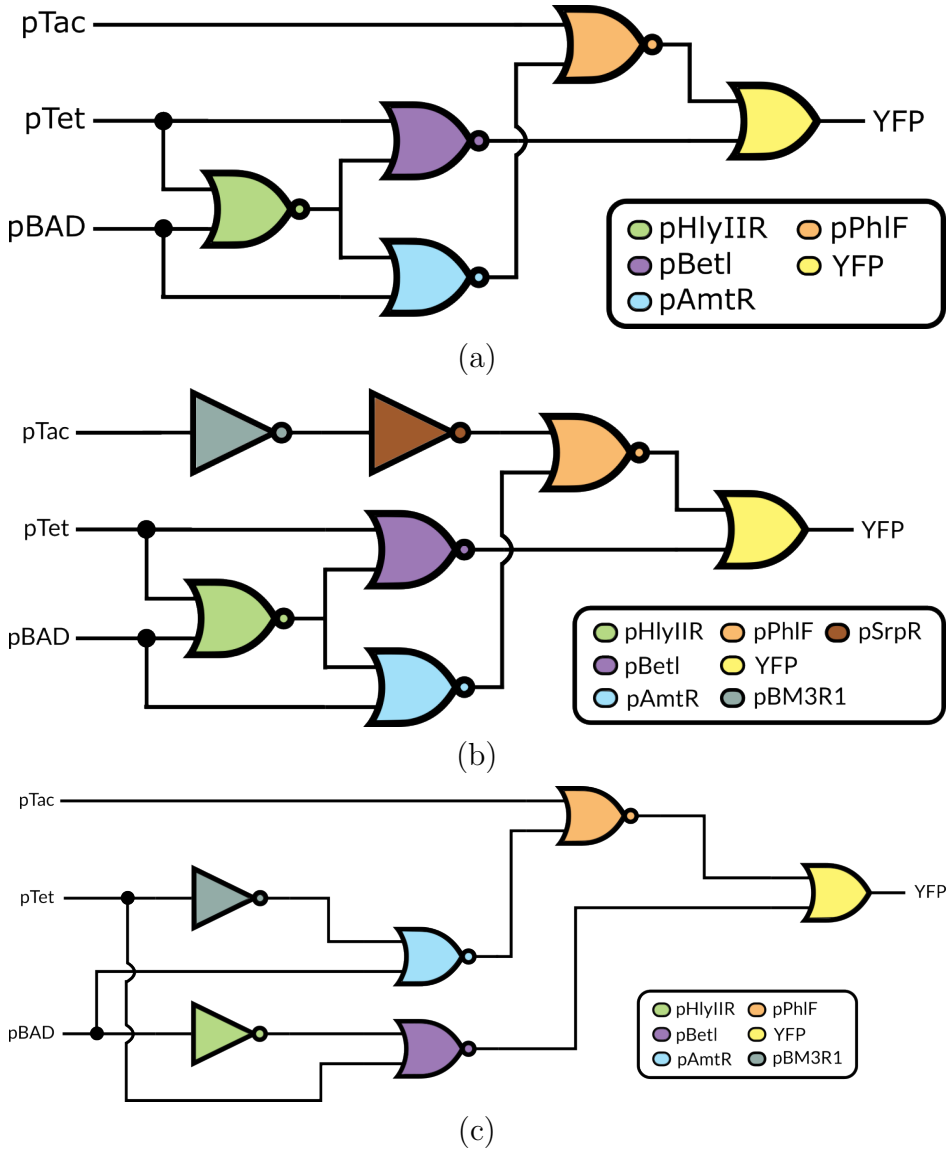

Figure 3: Three different logic layouts for the circuit 0x8E. The three input sensor promoters are *pTac*, *pTet* and *pBAD*, and the output is *YFP*. The OR gate is represented by  and the NOR gate by  . (a) Original design (O) as published by Nielsen et al.[26] (b) The two-inverter implementation (T) with added redundant logic as two NOT gates, which adds an extra delay to the IPTG pathway (starting from the input sensor promoter pTac). The NOT gate is represented by  . (c) The No-Logic-Hazard design (N) with logic-hazard-free optimizations. Reproduced from.[27] Copyright 2020 American Chemical Society.

jeopardizing the circuit's function.

Predictability is the computational analysis of the likelihood of erroneous behavior occurring. However, like in all engineering principles, there is a trade-off between the computational and mathematical complexity of a model and its accurate predictability. Usually, the more detailed the model is, the more accurate the results. Nonetheless, the enhanced complexity leads to more intricate analyses, necessitating more powerful computing capabilities, and requiring extended periods of analysis time. Moreover, procuring more refined, characterization parameter values for the model intensifies the demand for time and resources needed to obtain them. Consequently, an appropriate equilibrium must be achieved to ensure satisfactory accuracy of results while preserving manageable analysis requirements.

Extrinsic and Intrinsic Noise Simulation. *Extrinsic* noise sources of transcriptional variability refer to cell-to-cell differences in the transcriptional inputs as well as the transcriptional output. Beal[21] showed that this cell-cell variation might be accounted for by the emergent properties of complex reaction networks, which drive a lognormal distribution of gene expression levels across a population. This study draws from the Cello part library's[26] measured parameter distributions and, based on Beal's work, implements a lognormal-distributed parameter value model to calculate the incidence of glitching behavior in a population.

Intrinsic noise encompasses the stochasticity inherent within the cell, influenced by various factors such as spatial considerations, resource distribution, and stress. These factors significantly affect the likelihood or ability of a cell to execute its reactions. Given that the chemical reactions involved are discrete events with probabilities contingent upon the number of molecules, they reflect this inherent stress. Thus, to accurately capture these fluctuations, Gillespie's *stochastic simulation algorithm* (SSA)[29,30] was used for the analysis of the models.

Computational Models. While other model techniques are published,[31,32] this work utilizes the Cello model, not to advocate for this model, but because it is a model that has

parameters *and* experimental results of glitching behavior. The various models, parameterizations, and analysis methods employed in this study are denoted as follows: The terms "extrinsic" or "intrinsic" highlight whether the model was examined with extrinsic or intrinsic noise. Subsequently, the models are classified as either the "Cello model", as developed by Nielsen et al.,[26,32] the "default" model, as generated in iBioSim,[33] or the "abstracted" model based on stoichiometry amplification. The Cello model employs kinetic abstractions, yielding a model comparable to Hill equations. The default iBioSim model[33,34]models each protein's transcription initiation, production, dimerization, transcription factor binding, and degradation. The abstracted model modifies the default model by adjusting its degradation reactions. Termed stoichiometry amplification, this model increases the stoichiometry of degradation reactions while reducing reaction rates, thereby reducing the frequency of reaction occurrences but increasing their impact. Finally, it is indicated whether the model uses parameters characterized in experiments or default parameters obtained from the literature. In the Methods section, the models and their differences are described in more detail. The five different model types are as follows:

- *Extrinsic/Cello model/Characterized parameters* (E/C/C)
- *Extrinsic/Cello model/Default parameters* (E/C/D)
- *Extrinsic/Default model/Default parameters* (E/D/D)
- *Intrinsic/Default model/Default parameters* (I/D/D)
- *Intrinsic/Abstracted model/Default parameters* (I/A/D)

Results. The results are organized into four main stages of comparison. First, a negative control, a positive control, and a circuit made hazard-free using a register are presented. Second, the quantitative performance of the five different models is compared using the original circuit design. Third, within each model, the performance of the three different

implementations is evaluated qualitatively. Finally, all five models are compared to each other by determining which circuit layout they deem to be the best.

Figure 4 shows the controls for the analysis. Panel (a) displays an inverter along with its truth table. The analysis was conducted to determine if the circuit could fail to reach its intended state. Since an inverter cannot have a function hazard, the failure probability is expected to be low. The analysis confirms this, returning a failure probability of 0.2 percent, likely due to stochasticity in the system.

Figure 4 (b) shows the analysis of an OR gate, along with its Karnaugh map. In the analysis, the OR gate transitions from state (0, 1) to (1, 0). During this transition, the circuit can either pass through state (0, 0), following the red arrows in the figure, or state (1, 1), following the green arrows. Since state (1, 1) is also a high state, the output remains on. However, if the gate transitions through (0, 0), the output briefly turns off. This indicates that the OR gate has a function hazard, leading to a higher probability of failure. The results confirm this hypothesis, returning a failure probability of 37.3 percent.

Finally, Figure 4 (c) presents a third example circuit, with its output rendered hazard-free using a register to filter the circuit's output. The circuit consists of an inverter and a NOR gate, where the inverter sets a toggle switch and the NOR gate resets it. The circuit transitions its two inputs, A and B, from (0, 1) to (1, 0). The analysis reveals a failure probability of 40.8 percent, indicating that the attempt to make the circuit hazard-free using a register was unsuccessful. While a register could filter the hazard if it was clocked, as in synchronous designs, the genetic circuits analyzed in this work are asynchronous and lack a clock. As a result, the glitch is stored in the toggle switch instead of being transitory explaining the high failure probability. The same analysis presented on the three circuit examples was repeated across all models, transitions, and implementations of circuit 0x8E.

Figure 5 presents a scatter plot illustrating the predicted failure probability of each analyzed transition in the original design for every model. Color coding in the plot corresponds to the models, as indicated in the legend. The plot is divided into five sections: the first two

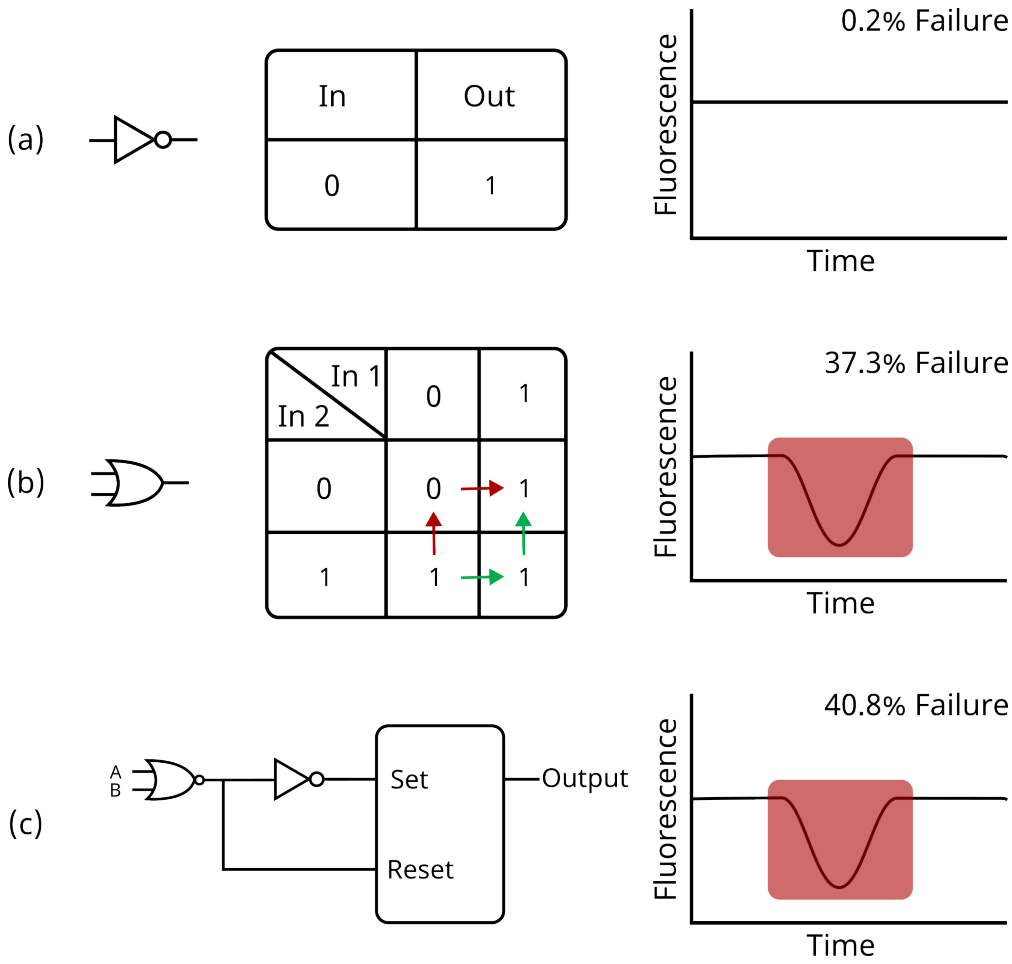

Figure 4: Analysis of an inverter and an OR gate from the Cello library. (a) Results of the inverter analysis, showing that the inverter is on when the input is off and fails to reach its intended state in only 0.2 percent of cases. Since an inverter is a single-input gate, it does not have any function hazards. (b) Analysis of an OR gate, a simple gate with a known function hazard. The Karnaugh map visualizes this hazard, where a transition from (0, 1) to (1, 0) can pass through either state (0, 0) or state (1, 1), as indicated by the colored arrows. If the transition follows the red arrows, a glitch occurs, briefly turning off the output. The analysis indicates a failure probability of 37.3 percent for this scenario. (c) Shows the analysis of a circuit including a register. The circuit consists of an inverter and a NOR gate, with the inverter setting a toggle switch and the NOR gate resetting it. The circuit transitions its two inputs, A and B, from (0, 1) to (1, 0). Similar to the OR gate transition, this circuit is designed to stay on throughout the transition, but it still has a hazard. The analysis shows a failure probability of 40.8 percent, indicating that the attempt to eliminate the hazard using a register was unsuccessful.

sections represent transitions with function hazards, the middle section shows steady-state failures, and the last two sections depict transitions without function hazards.

The plot highlights variations in the quantitative predictions made by the models. Specif-

ically, it compares the failure probability predicted by each model for the original circuit. Similar analyses for the other two designs can be found in the supplemental material. For example, while some models predict similar failure likelihoods for certain transitions across models (e.g., transition (1,1,1) to (0,1,0)), discrepancies are observed in others, such as transition (0,1,1) to (0,0,0).

Notably, the scatter plot reveals a distinction between transitions and function hazards. Function hazards, located at the top, display greater dispersion and skew towards the right, whereas transitions, found in the lower sections, tend to cluster towards the left, indicating lower probabilities. This observation aligns with expectations, as function hazard transitions are known to exhibit faults unlike regular transitions. However, substantial differences exist in the predicted absolute failure probabilities among the five different models. For example, in the first transition shown in the figure, (0, 1, 0) to (1, 1, 1), the (E/D/D) model predicts a failure probability of 2.4 percent, whereas the (I/A/D) model predicts a failure probability of 36.9 percent. This discrepancy raises questions about the models' quantitative predictive power. While the absolute failure probabilities are likely not accurate and do not permit direct comparison between models, different circuit implementations can still be qualitatively compared when evaluated using the same model.

Transitioning from quantitative to qualitative comparison, instead of comparing individual failure percentages for each transition and steady state, the analysis involved comparing the number of preferred choices for each circuit within a specified model. For each of the five models, the three circuit designs were compared, and the design that outperformed the other two for the transition or steady state within the model was considered the preferred choice. The results are illustrated in Figure 6.

Examining the results of the (E/C/C) model in Figure 6 as an example, it is evident that the non-logic-hazard design outperforms the other two designs for 13 choices, while the original design performs better for ten choices, and the two-inverter design for three choices. These numbers sum to $10+3+13 = 26$, rather than the total number of analyzed transitions

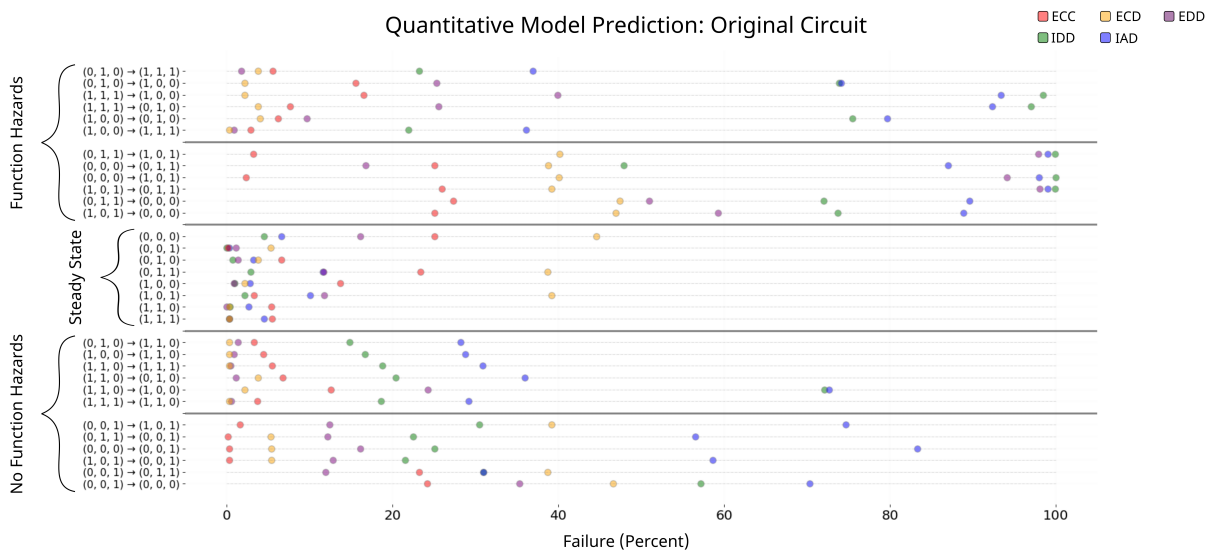

Figure 5: Model predictions for the original circuit 0x8E design are depicted in the scatter plot. The x-axis represents the predicted likelihoods of failures for each transition. The plot highlights disparities in failure probabilities calculated by the various models, presenting a challenge in quantitative comparison between them.

and steady states, which is 32. This discrepancy arises because the failure probabilities for the remaining choices were too similar.

The determination of preferred choices was based on the percent difference from the median. Let x_i represent the sample failure percentage and $\bar{x}$ denote the median. The percent difference was calculated using the following equation:

$$\text{Percent Difference} = \frac{x_i - \bar{x}}{\frac{|x_i + \bar{x}|}{2}} \times 100$$

A choice's percent difference was classified as preferred if it fell below -10 percent. Further details regarding this methodology can be found in the Methods section.

Figure 6 also illustrates the same analysis for the other four models. Two main take-away messages emerge from these results. First, the models unanimously agree that the non-logic-hazard design outperforms the others. Therefore, regardless of the model chosen by the designer, the results consistently support the construction of this design. This outcome is expected, as all three circuit implementations share the same function hazards.

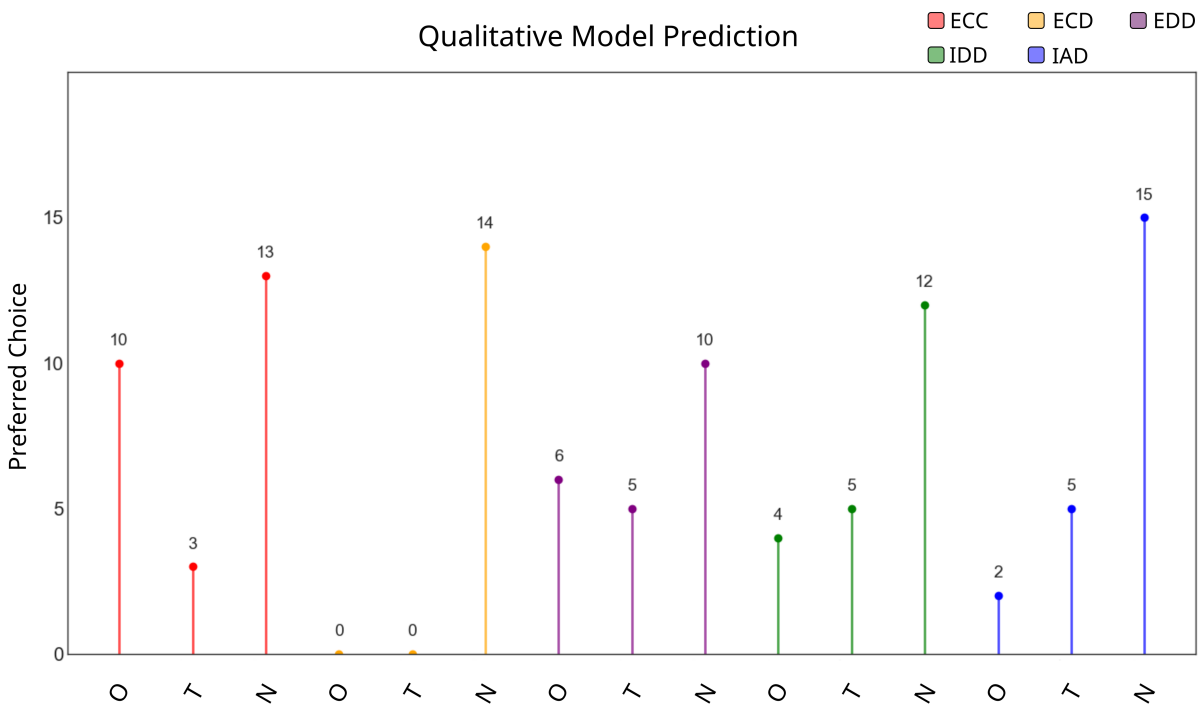

Figure 6: The graph illustrates the number of preferred choice for each circuit and model, with the y-axis representing the count of preferred choices. Across all five models, consensus is reached that the non-logic-hazard (N) circuit design exhibits the highest number of preferred choices, thereby outperforming the other designs. Specifically, according to the (E/C/C) model, the non-logic-hazard design outperforms the two-inverter (T) or original (O) design with 13 preferred choices, compared to three and ten preferred choices, respectively.

However, unlike the two-inverter and original design, the non-logic-hazard design also lacks logic hazards. Second, models that incorporate extrinsic noise predict the original design to outperform the two-inverter design, while models considering intrinsic noise suggest the opposite, favoring the two-inverter design over the original.

Figure 6 focused on transitions and steady states where a design outperformed the others, omitting cases where the design performs substantially worse than the others. To identify designs with notably worse choices, the same equation was utilized, this time considering choices worse if they exceeded a percent difference of 10 percent from the median.

Figure 7 depicts the five models on the y-axis, with the circuit design represented as data points and their scores on the x-axis. The score was calculated by summing up all preferred

and worse failures for each design, where a preferred choice equaled 1 and a worse choice equaled -1.

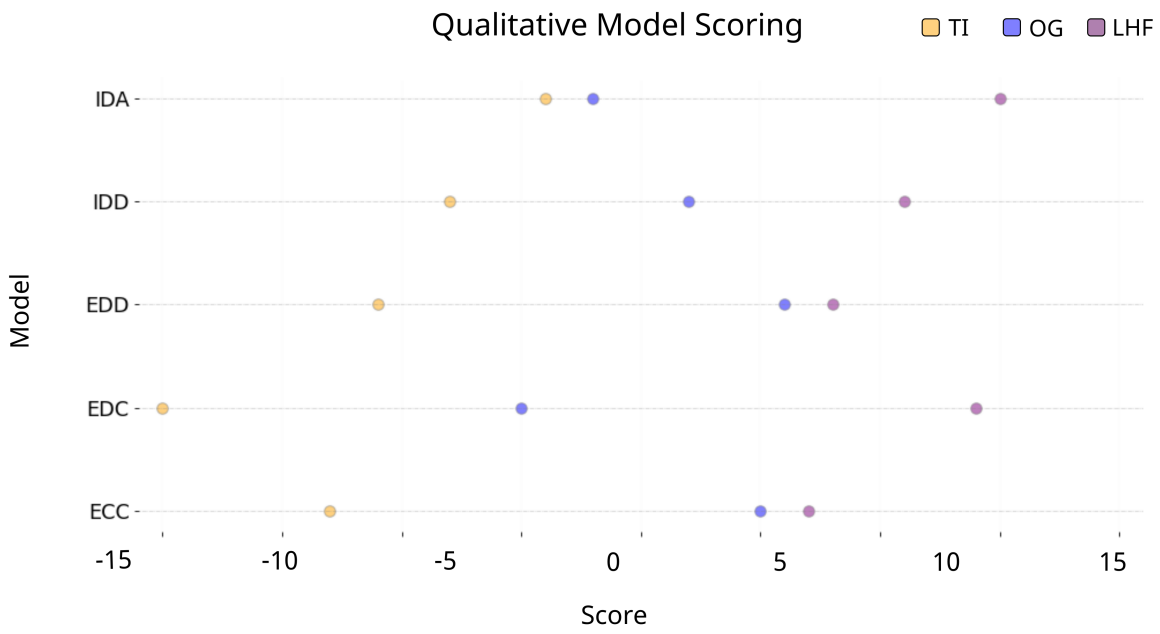

Figure 7: Qualitative model scoring. The figure illustrates the score of each circuit implementation within a specific modeling method. The model is depicted on the y-axis, while the x-axis represents the score. Each colored marker corresponds to one of the three circuit designs. The score is calculated as the difference between the number of preferred and worse choices.

Figure 7 indicates that when considering both preferred and worse choices, all models converge in their assessments. They unanimously conclude that the non-logic-hazard design exhibits the best performance, followed by the original, and then the two-inverter design. Upon comparing the score with the number of preferred choices, it becomes evident that although the two-inverter design performs better in more cases than the original design, it also fares worse in more cases, resulting in a lower score across all models.

The predictions for the percentage failure of the models, environmental conditions, and investigated variables are displayed in the supplementary material. Table S1 illustrates the failure percentages in the original design circuit transitions, Table S2 provides the same information for the two-inverter design, and Table S3 presents the corresponding data for

Table 1: Comparison of runtimes for analysis simulations conducted in iBioSim.[33] The first column displays the selected model, while the second column indicates the chosen failure scenario. Runtimes are presented in the MM:SS.MS format. All simulations were executed on a computer equipped with a 2.3 GHz Quad-Core Intel Core i7 processor and 32 GB 3733 MHz LPDDR4X memory.

Circuit	Inputs	(E/C/C)	(E/C/D)	(E/D/D)	(I/D/D)	(I/D/A)
O	$(0,1,0) \rightarrow (1,1,1)$	00:37.61	00:38.68	04:09.79	00:31.81	00:16.78
O	$(0,1,1) \rightarrow (1,0,1)$	00:41.86	00:39.16	02:28.77	00:05.65	00:03.68
T	$(0,1,0) \rightarrow (1,1,0)$	00:49.57	00:52.77	04:46.02	00:21.52	00:12.36
T	$(0,0,1) \rightarrow (1,0,1)$	00:55.65	00:48.12	04:31.78	00:39.09	00:15.98
N	$(0,0,0)$	00:27.53	00:26.77	03:55.39	00:18.07	00:11.58

the logic-hazard-free design.

To assess the complexity of each model, the runtimes for the simulations were collected. Table 1 presents five selected examples for runtime comparison. Each example corresponds to a specific circuit failure scenario. These include static $0 \rightarrow 0$ and static $1 \rightarrow 1$ transitions, with and without function hazards, along with an example of steady-state failure. The table comprises seven columns: the first column indicates the selected circuit design, the second column specifies the chosen failure scenario, and the remaining five columns display the runtime for each of the five models.

The results indicate that the (I/D/A) model boasts the fastest analysis time, followed by the (I/D/D) model, while the (E/D/D) model exhibits the slowest performance. The remaining two models, (E/C/C) and (E/C/D), show similar analysis runtimes. However, it is important to note that the (E/C/C) and (E/C/D) models rely on characterized parameters, necessitating extensive and time-intensive laboratory work. Therefore, considering both the analysis results and runtime, one could argue that the faster and less complex intrinsic modeling method can offer valuable and reliable insights into genetic circuit behavior. The runtime measurements were conducted on a computer equipped with a 2.3 GHz Quad-Core Intel Core i7 processor and 32 GB 3733 MHz LPDDR4X memory.

This paper evaluates the predictive quality of five different computational models by

examining the robustness of three circuit implementations, all representing the same logic function. These modeling techniques employ distinct noise sources, abstractions, and parameters. Overall, when comparing the five groups of modeling techniques for the three circuit designs, it becomes apparent that more abstract models are inadequate for quantitatively evaluating the precise probability of failure. However, they can be utilized to qualitatively assess which design choice is optimal for a given logic function with multiple available designs. Specifically, as shown in Figure 7, the models consistently predict which circuit implementation is the best in aggregate. The actual predicted values may not be accurate for any model, as there is no absolute ground truth beyond experimentation.

In the case presented here, the non-logic-hazard-free design exhibited lower failure rates and greater overall robustness compared to the other two designs.Therefore, a designer can initially use a higher-level abstract model, which is less complex, to develop an overall robust design. Throughout the process, if critical transitions and steady states are identified as highly failure-prone, the user can switch to a lower-level abstraction model to refine the design for the specific use case.

These findings, along with further research, contribute to advancing the DBTL pipeline. This advancement facilitates the learning and design stages by filtering out circuit layouts with a higher likelihood of glitches for input transitions that are deemed critical by the designer. Additionally, the implementation of a model generator in genetic design automation tools, which automatically incorporates intrinsic or extrinsic noise sources, would assist genetic circuit designers in applying and testing different noise levels to obtain failure predictions and assess circuit robustness. This work demonstrates that more abstracted models yield similar results to expanded or characterized models. Thus, computationally less complex models can still provide meaningful insights for design space exploration. It would be beneficial to have a "knob" feature in the future that allows easy adjustment of model abstraction.

Methods

Both models and simulations were employed using iBioSim.[33] Models describing the GRN's behavior are automatically created using iBioSim's automatic model generator.[27,35,36] The different environments (noise simulations, input concentration changes, and circuit failure constraints) where generated using iBioSim's GUI. All of the projects use the *Synthetic Biology Open Language* (SBOL)[37,38] for the representation of genetic designs and their function; the *Systems Biology Markup Language* (SBML)[39,40] for the mathematical model representation of the different genetic circuits and their interactions; and the *Simulation Experiment Description Language* (SED-ML)[41] for the simulation description of the mathematical models and are available in the Supplementary Information.

Automatic modeler in iBioSim. To generate models automatically in iBioSim, SBOLDesigner[42] is initially utilized for creating circuit designs for each layout. Subsequently, the designs are enriched using the *Virtual Parts Repository*[43,44] (VPR) model generator, which adds interactions between components specified in the SBOL file and incorporates non-DNA elements. Finally, the SBOL file is converted into an SBML model using the built-in SBOL to SBML converter in iBioSim. A full workflow of this process is described in Watanabe et al.[33]

Models describing general behavior and their parameter values. This study employs two distinct models to represent the functioning of the GRNs in this work: the "Cello" model, developed by Nielsen et al.,[26,32] and the "default" model, generated in iBioSim.[33] The default model encompasses reactions related to transcription, translation, protein-binding, protein degradation, transcriptor-function protein interactions, and binding, repression, and activation processes. The Cello model has been a benchmark in Synthetic Biology due to: 1) its provision of transparent and reproducible characterization experiments for transcriptional regulatory gates, 2) the availability of gate characterization results for researchers to utilize in their models, and 3) the model offering simpler parameters to characterize by consolidating various regulatory kinetics into experimentally-observable variables. In particular, for

this work, we chose the Cello model because there are public and accessible gate parameter values[1] for the gates used in the circuit designs being studied (Figure 3).

In addition, a comparison between default (obtained from literature) model parameter values and characterized gate parameter values was used to determine the effect on predicted circuit failure percentages. The default parameter values were used for both the default model in iBioSim,[33] and the Cello model published in;[26,32] and part-characterized model parameter values for each component obtained from experimentation[26] [2]. he parameter values for both the Cello and default models can be found in Tables S7 and S9 in the supplementary material. The characterized parameters are specific to the Cello model since they are tailored to an ODE model that employs concentrations rather than molecule counts. In contrast, the default model employs different parameters that consistently operate within the model but do not align with the characterized parameters. This discrepancy also contributes to the low molecular counts within the default model, which remain consistent with each other. A threshold of 60 molecules was selected for input, as any value exceeding this results in an identical output molecular count. The thresholds of ten and 30 were identified in.[15] The method employs ODE analysis to characterize the sensitivity of a NOT gate with iBioSim's default parameters, as all gates utilized in building the circuit models utilize these default parameters. This analysis facilitates the determination of the input signal required to repress the output (30 molecules), or the minimum input (ten molecules) required to trigger a high output.

In future work, it would be beneficial to consider mapping the Cello parameters to allow them to be utilized with SSA analysis. This could be done by making assumptions about the volumes of the cells. However, since the point of this paper is to demonstrate that even with default parameters design choices can be effectively evaluated, this was not done for this paper. A major observation is that design choices can be evaluated without the time-consuming experimental part characterization work done in the Cello project.

[1] `https://synbiohub.programmingbiology.org/public/Eco1C1G1T1/Eco1C1G1T1_collection/1`
[2] `https://synbiohub.programmingbiology.org/public/Eco1C1G1T1/Eco1C1G1T1_collection/1`

Intrinsic noise model. In this study, we followed iBioSim's default settings for the default intrinsic model, where production and degradation occur in steps of ten and one, respectively. During translation, mRNA is translated ten times, producing an average of ten proteins per mRNA overall. Each protein is degraded by itself, and hence, modeled in steps of one. Therefore, when a production reaction is fired, ten molecules are produced, while only one protein is degraded when a degradation reaction is fired. In the abstracted model, we aimed to simplify the complexity of the intrinsic model by adjusting protein production and degradation rates to operate both in increments of ten molecules per reaction step. This adjustment means that each time a production or degradation reaction occurs, ten molecules are either produced or degraded. While the production reaction remains unchanged from the default model, modifications are necessary for the degradation reaction. With ten molecules now being degraded per reaction event instead of one, the rate of degradation needs to be reduced by a factor of ten. This adjustment does not alter the underlying reaction mechanism; rather, it affects how frequently the reaction occurs, ultimately saving time during model analysis.

Furthermore, to reduce the number of species in both models, only the internal molecules and complexes are modeled, and the input molecules such as $IPTG$, aTc, and Ara are not modeled. For instance, $IPTG$ binds to $LacI$ which regulates the circuit. Instead of modeling both species, only $LacI$ is modeled.

Extrinsic noise model. In this study, the extrinsic noise model employs a basic instance of *static* external disturbances, characterized as a random selection from a *lognormal* distribution for each parameter value used in the model at the start of each simulation run. The distribution's mean is the default parameter value in iBioSim (derived from the literature), and the standard deviation is forty percent of the mean's absolute value mimicking the "extrinsic noise". This noise value was determined during calibration with various noise values but should be replaced with a more accurate estimate derived from experiments. However, since the focus of this research is on qualitative rather than quantitative comparisons,

the exact extrinsic noise values are not essential.

Simulation Methodology for Models. The mathematical models employed in this study, depending on their character, are deterministic for the extrinsic noise models and stochastic for the intrinsic noise models. Consequently, distinct simulation methods were utilized for each. The deterministic models, representing extrinsic noise, are *Ordinary Differential Equation* (ODE) models. The Runge–Kutta–Fehlberg (4, 5) method (rkf45),[45] a widely accepted numerical method, was applied for their simulation. In contrast, Gillespie's Stochastic Simulation Algorithm (SSA)[29,30] was implemented for the stochastic models, which represent intrinsic noise. The extrinsic noise models underwent analysis through 1000 rkf45 runs, while the intrinsic noise models were subjected to analysis using 1000 SSA runs.

Furthermore, the Cello model and the default model exhibited other disparities, particularly in their handling of timesteps. The Cello model operates using arbitrary timesteps to ensure consistency with the parameters of the iBioSim default model, which are derived from literature sources. As a result, simulations for the default model spanned 2000 time units, with input changes introduced after 1000 seconds. Conversely, simulations of Cello models lasted for 24-time points (or hours), with input modifications occurring at the twelfth hour.

To assess the effectiveness of the circuit, the output values from all circuits were examined and contrasted with a predetermined constraint value at a specific time point (1000 for the default model, 12 for the Cello model). This allowed the determination of circuit failure for all input change transitions and incorrect steady states. A circuit was deemed to have failed during a high to high ($1 \rightarrow 1$) transition if its output dipped below the critical constraint value. Since the default model predicts molecule counts and the Cello model forecasts *relative promoter units* (RPUs),[46] this critical value differed between the Cello and default models.

For low to low ($0 \rightarrow 0$) transitions, the critical value was set at ten molecules for the default model and 0.1875 RPUs for the Cello model, representing 20% of the maximum predicted output. If a circuit's production exceeded this threshold during a $0 \rightarrow 0$ transition, the circuit was classified as having failed. Thus, if a circuit's output is expected to remain

low (or 0) throughout the simulation, exceeding the constraint value is considered a glitch. Conversely, if the circuit's output is projected to remain high throughout the simulation, falling below the constraint value is also deemed a glitch. Simulations were halted upon the observation of a glitch (only static glitches were examined).

For steady-state failures, if the output surpassed the constraint value for expected low output states, or fell below the constraint value for expected high output states, the circuit was considered to have failed the steady-state simulation, and the simulation was terminated. The frequencies of these simulation terminations contribute to the percent failure rates exhibited in Tables S1, S2, S3, and S4.

Choice Evaluation. A circuit design was classified as a preferred choice if its likelihood of failure has a percent difference that falls below -10 percent, while it was deemed worse if it exceeded 10 percent compared to the median. For instance, consider the transition from (0, 1, 0) to (1, 1, 1) in the (E/C/C) model. According to the supplemental material, the failure percentages for each model are as follows: original design 5.6 percent, two-inverter design 4.5 percent, and non-logic-hazard design 3.1 percent, with the median being 4.5 percent for the non-logic-hazard model.

1. Percent difference of **non-logic-hazard**:

$$\frac{3.1 - 4.5}{\frac{|3.1+4.5|}{2}} \times 100 \approx -36.84\%$$

2. Percent difference of **original**:

$$\frac{5.6 - 4.5}{\frac{|5.6+4.5|}{2}} \times 100 \approx 21.78\%$$

In this example, the non-logic-hazard implementation shows a percent difference of -36.84 percent, which falls below the -10 percent threshold, thus qualifying the non-logic-hazard as the preferred design choice. Similarly, the original design exhibits a percent difference of

21.78 percent, which exceeds the 10 percent threshold, thus indicating a worse choice.

Considerations/Assumptions. This work compares the predictive power of five different model techniques by comparing the prediction of the most robust circuit layout for certain input transitions or steady states. However, it is out of this work's scope to determine what noise source is a more accurate representation of the true GRNs behavior, or what is the magnitude of each noise source's influence on the predicted output. The main objective is to determine if there *are* any differences in robustness predictions if we use abstracted models or not if we consider noise being *extrinsic* or *intrinsic*, and/or if we use characterized parameter values or literature-obtained values. Therefore, certain assumptions were made which are listed as follows:

1. We assume the probability distribution for *extrinsic* noise follows a truncated-normal distribution. However, Beal[21] argues that this noise is better described using a log-normal distribution. For the purpose of this work, we chose the normal distribution for parameter values to simulate extrinsic noise for simplicity and clarity of simulations, though further work could include log-normal distributions for these.

2. The magnitude of noise level for the truncated-normal distribution was chosen to be = 0.4, which came from initial testing. However, the absolute value (magnitude) of intrinsic or extrinsic noise could be different for GRNs, and if results show that there are differences in robustness predictions, should be measured for more accurate results.

3. The level of extrinsic noise could be different for each reaction (therefore have a different effect on a parameter's value distribution). However, for this work, we are assuming that the magnitude of the noise is the same for each reaction parameter.

4. For all simulation runs, we assume that the change in input molecule concentrations is instantaneous, instead of being a gradual process.

5. The Cello model described in Nielsen et al, and Shin et al.[26,32] can use τ_{ON} and τ_{OFF} parameters to describe how quickly a gate turns ON or OFF. However, for this work,

we are assuming all the different parts have the same values of τ_{ON} and τ_{OFF} given that there are no experimental parameter values for them.

Establishing the accuracy of these assumptions necessitates further experimental efforts that are currently expensive and unavailable. Nevertheless, as the findings of this study revealed differences in robustness predictions among various models, it warrants deeper examination to ascertain the legitimacy of these assumptions, considering that they may influence a designer's decision-making process.

Associated Content

Additional Information

The latest version of iBioSim with the dynamic model generator, including source code, instructions, and related files, can be found online at `https://github.com/MyersResearchGroup/iBioSim`. All models created in this work are accessible on Github at `https://github.com/MyersResearchGroup/RobustnessPrediction/tree/main`.

Supplemental Information

- Detailed simulation results for all circuits (Tables S1, S2, S3, S4, S5, and S6)
- Parameters values used in the models (Tables S7, S8, and S9)
- Additional quantitative Figures mentioned in this paper (Supplemental Figures 1 and 2)

Special Issue Paper

Invited contribution from the 15th International Workshop on Bio-Design Automation.

Author Information

Corresponding Author

Chris J. Myers - *Department of Electrical, Computer, and Energy Engineering, University of Colorado Boulder, Boulder, 80309, Colorado, USA*; Email:chris.myers@colorado.edu

Authors

Pedro Fontanarrosa - *Department of Electrical, Computer, and Energy Engineering, University of Colorado Boulder, Boulder, 80309, Colorado, USA*

Lukas Buecherl - *Department of Biomedical Engineering, University of Colorado Boulder, Boulder, 80309, Colorado, USA*

Author Contribution

P.F. and L.B. created the models and ran the simulations. P.F. and L.B. created all figures in their entirety and all images used in the TOC graphic. C.M. supervised P.F. and L.B.'s work that contributed to this paper. All authors contributed to the writing of the paper.

Notes

The authors declare no competing financial interest.

Acknowledgement

The authors of this work are supported by DARPA FA8750-17-C-0229 and by National Science Foundation Grant No. 1856740. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the funding agencies.

References

(1) Xiang, Y.; Dalchau, N.; Wang, B. Scaling up Genetic Circuit Design for Cellular Computing: Advances and Prospects. *Natural Computing* **2018**, *17*, 833–853.

(2) Myers, C. J.; Beal, J.; Gorochowski, T. E.; Kuwahara, H.; Madsen, C.; McLaughlin, J. A.; Mısırlı, G.; Nguyen, T.; Oberortner, E.; Samineni, M.; Wipat, A.; Zhang, M.; Zundel, Z. A Standard-Enabled Workflow for Synthetic Biology. *Biochemical Society Transactions* **2017**, *45*, 793–803.

(3) Schladt, T.; Engelmann, N.; Kubaczka, E.; Hochberger, C.; Koeppl, H. Automated Design of Robust Genetic Circuits: Structural Variants and Parameter Uncertainty. 00000 tex.ids= AutomatedDesignRobust_schladt_2021a publisher: American Chemical Society.

(4) Brooks, S. M.; Alper, H. S. Applications, Challenges, and Needs for Employing Synthetic Biology beyond the Lab. *Nature Communications* **2021**, *12*, 1390.

(5) Peccoud, J. Synthetic Biology: Fostering the Cyber-Biological Revolution. *Synthetic Biology* **2016**, *1*.

(6) Dray, K. E.; Muldoon, J. J.; Mangan, N. M.; Bagheri, N.; Leonard, J. N. GAMES: A Dynamic Model Development Workflow for Rigorous Characterization of Synthetic Genetic Systems. 00000 Publisher: American Chemical Society.

(7) Karlebach, G.; Shamir, R. Modelling and Analysis of Gene Regulatory Networks. *Nature Reviews Molecular Cell Biology* **2008**, *9*, 770–780.

(8) Heath, A. P.; Kavraki, L. E. Computational challenges in systems biology. *Computer Science Review* **2009**, *3*, 1–17.

(9) Machado, D.; Costa, R. S.; Rocha, M.; Ferreira, E. C.; Tidor, B.; Rocha, I. Modeling formalisms in systems biology. *AMB express* **2011**, *1*, 1–14.

(10) Gonçalves, E.; Bucher, J.; Ryll, A.; Niklas, J.; Mauch, K.; Klamt, S.; Rocha, M.; Saez-Rodriguez, J. Bridging the layers: towards integration of signal transduction, regulation and metabolism into mathematical models. *Molecular BioSystems* **2013**, *9*, 1576–1583.

(11) Kitano, H. Systems biology: a brief overview. *science* **2002**, *295*, 1662–1664.

(12) Voit, E. O. *A first course in systems biology*; Garland Science, 2017.

(13) Alon, U. *An introduction to systems biology: design principles of biological circuits*; CRC press, 2019.

(14) Chen, K. C.; Calzone, L.; Csikasz-Nagy, A.; Cross, F. R.; Novak, B.; Tyson, J. J. Integrative analysis of cell cycle control in budding yeast. *Molecular biology of the cell* **2004**, *15*, 3841–3862.

(15) Buecherl, L.; Roberts, R.; Fontanarrosa, P.; Thomas, P. J.; Mante, J.; Zhang, Z.; Myers, C. J. Stochastic Hazard Analysis of Genetic Circuits in iBioSim and STAMINA. *ACS Synthetic Biology* **2021**, acssynbio.1c00159.

(16) Neupane, T.; Myers, C. J.; Madsen, C.; Zheng, H.; Zhang, Z. In *Computer Aided Verification*; Dillig, I., Tasiran, S., Eds.; Springer International Publishing: Cham, 2019; Vol. 11561; pp 540–549.

(17) Roberts, R.; Neupane, T.; Buecherl, L.; Myers, C. J.; Zhang, Z. In *Verification, Model Checking, and Abstract Interpretation*; Finkbeiner, B., Wies, T., Eds.; Springer International Publishing: Cham, 2022; Vol. 13182; pp 319–331.

(18) Baldwin, G., Bayer, T., Dickinson, R., Ellis, T., Freemont, P. S., Kitney, R. I., Polizzi, K. M., Stan, G.-B., Eds. *Synthetic Biology: A Primer*, revised edition ed.; Imperial College Press ; World Scientific Publishing Co. Pte. Ltd: [London] : Singapore ; Hackensack, NJ ; London, 2016.

(19) Elowitz, M. B.; Levine, A. J.; Siggia, E. D.; Swain, P. S. Stochastic Gene Expression in a Single Cell. *Science* **2002**, *297*, 1183–1186.

(20) Paulsson, J. Summing up the Noise in Gene Networks. *Nature* **2004**, *427*, 415–418.

(21) Beal, J. Biochemical Complexity Drives Log-normal Variation in Genetic Expression. *Engineering Biology* **2017**, *1*, 55–60.

(22) Purnick, P. E. M.; Weiss, R. The Second Wave of Synthetic Biology: From Modules to Systems. *Nature Reviews Molecular Cell Biology* **2009**, *10*, 410–422.

(23) Swain, P. S.; Elowitz, M. B.; Siggia, E. D. Intrinsic and Extrinsic Contributions to Stochasticity in Gene Expression. *Proceedings of the National Academy of Sciences* **2002**, *99*, 12795–12800.

(24) Lestas, I.; Paulsson, J.; Ross, N. E.; Vinnicombe, G. Noise in Gene Regulatory Networks. *IEEE Transactions on Automatic Control* **2008**, *53*, 189–200.

(25) Gillespie, D. T. Exact Stochastic Simulation of Coupled Chemical Reactions. *The Journal of Physical Chemistry* **1977**, *81*, 2340–2361.

(26) Nielsen, A. A. K.; Der, B. S.; Shin, J.; Vaidyanathan, P.; Paralanov, V.; Strychalski, E. A.; Ross, D.; Densmore, D.; Voigt, C. A. Genetic Circuit Design Automation. *Science* **2016**, *352*, aac7341.

(27) Fontanarrosa, P.; Doosthosseini, H.; Espah Borujeni, A.; Dorfan, Y.; Voigt, C. A.; Myers, C. J. Genetic Circuit Dynamics: Hazard and Glitch Analysis. *ACS Synthetic Biology* **2020**,

(28) Buecherl, L.; Fontanarrosa, P.; Thomas, P. J.; Mante, J.; Zhang, Z.; Myers, C. J. Stochastic Hazard Analysis of Genetic Circuits in iBioSim and STAMINA. *ACS Synthetic Biology (Under Revision)* **2021**,

(29) Gillespie, D. T. Stochastic Simulation of Chemical Kinetics. *Annual Review of Physical Chemistry* **2007**, *58*, 35–55.

(30) Gillespie, D. T. Exact stochastic simulation of coupled chemical reactions. *The Journal of Physical Chemistry* **1977**, *81*, 2340–2361.

(31) Gander, M. W.; Vrana, J. D.; Voje, W. E.; Carothers, J. M.; Klavins, E. Digital Logic Circuits in Yeast with CRISPR-dCas9 NOR Gates. *Nature Communications* **2017**, *8*, 15459.

(32) Shin, J.; Zhang, S.; Der, B. S.; Nielsen, A. A.; Voigt, C. A. Programming *Escherichia Coli* to Function as a Digital Display. *Molecular Systems Biology* **2020**, *16*, e9401.

(33) Watanabe, L.; Nguyen, T.; Zhang, M.; Zundel, Z.; Zhang, Z.; Madsen, C.; Roehner, N.; Myers, C. iBioSim 3: A Tool for Model-Based Genetic Circuit Design. *ACS Synthetic Biology* **2018**,

(34) Myers, C. J.; Barker, N.; Jones, K.; Kuwahara, H.; Madsen, C.; Nguyen, N.-P. D. iBioSim: A Tool for the Analysis and Design of Genetic Circuits. *Bioinformatics* **2009**, *25*, 2848–2849.

(35) Fontanarrosa, P.; Hosseini, H.; Borujeni, A.; Dorfan, Y.; Voigt, C.; Myers, C. Analyzing Genetic Circuits for Hazards and Glitches. 11th International Workshop on Bio-Design Automation. Cambridge, UK, 2019; pp 32–33.

(36) Fontanarrosa, P. Investigating Genetic Circuit Failures. Ph.D. thesis, The University of Utah, United States – Utah, 2022.

(37) Buecherl, L.; Mitchell, T.; Scott-Brown, J.; Vaidyanathan, P.; Vidal, G.; Baig, H.; Bartley, B.; Beal, J.; Crowther, M.; Fontanarrosa, P., et al. Synthetic biology open language (SBOL) version 3.1. 0. *Journal of integrative bioinformatics* **2023**, *20*, 20220058.

(38) Galdzicki, M. et al. The Synthetic Biology Open Language (SBOL) Provides a Community Standard for Communicating Designs in Synthetic Biology. *Nature Biotechnology* **2014**, *32*, 545–550.

(39) Chaouiya, C. et al. SBML Qualitative Models: A Model Representation Format and Infrastructure to Foster Interactions between Qualitative Modelling Formalisms and Tools. *BMC Systems Biology* **2013**, *7*, 135.

(40) Hucka, M. et al. The Systems Biology Markup Language (SBML): A Medium for Representation and Exchange of Biochemical Network Models. *Bioinformatics* **2003**, *19*, 524–531.

(41) Waltemath, D.; Adams, R.; Bergmann, F. T.; Hucka, M.; Kolpakov, F.; Miller, A. K.; Moraru, I. I.; Nickerson, D.; Sahle, S.; Snoep, J. L.; Le Novère, N. Reproducible Computational Biology Experiments with SED-ML - The Simulation Experiment Description Markup Language. *BMC Systems Biology* **2011**, *5*, 198.

(42) Zhang, M.; McLaughlin, J. A.; Wipat, A.; Myers, C. J. SBOLDesigner 2: an intuitive tool for structural genetic design. *ACS synthetic biology* **2017**, *6*, 1150–1160.

(43) Misirli, G.; Hallinan, J.; Wipat, A. Composable modular models for synthetic biology. *ACM Journal on Emerging Technologies in Computing Systems (JETC)* **2014**, *11*, 1–19.

(44) Cooling, M. T.; Rouilly, V.; Misirli, G.; Lawson, J.; Yu, T.; Hallinan, J.; Wipat, A. Standard virtual biological parts: a repository of modular modeling components for synthetic biology. *Bioinformatics* **2010**, *26*, 925–931.

(45) Carpenter, M. H. K. *Fourth-Order 2N-Storage Runge-Kutta Schemes*; 1994.

(46) Kelly, J. R.; Rubin, A. J.; Davis, J. H.; Ajo-Franklin, C. M.; Cumbers, J.; Czar, M. J.; de Mora, K.; Glieberman, A. L.; Monie, D. D.; Endy, D. Measuring the Activity of BioBrick

Promoters Using an in Vivo Reference Standard. *Journal of Biological Engineering* **2009**, *3*, 4.

Graphical TOC Entry