



## Mathematical Modeling of Synthetic Genetic Circuits

Haimei Wang 

Hainan Institute of Biotechnology, Haikou, 570206, Hainan, China

 Corresponding author: [haimei.wang@hibio.org](mailto:haimei.wang@hibio.org)Computational Molecular Biology, 2025, Vol.15, No.4 doi: [10.5376/cmb.2025.15.0019](https://doi.org/10.5376/cmb.2025.15.0019)

Received: 03 Jun., 2025

Accepted: 14 Jul., 2025

Published: 02 Aug., 2025

**Copyright** © 2025 Wang. This is an open access article published under the terms of the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.<sup>6</sup>

### Preferred citation for this article:

Wang H.M., 2025, Mathematical modeling of synthetic genetic circuits, Computational Molecular Biology, 15(4): 193-207 (doi: [10.5376/cmb.2025.15.0019](https://doi.org/10.5376/cmb.2025.15.0019))

**Abstract** Synthetic genetic circuits are the core research objects in synthetic biology, and the programming of cell behavior is achieved through the combination of engineered gene elements. Mathematical modeling provides crucial support for understanding and designing synthetic genetic circuits, enabling researchers to predict the dynamic behavior of the circuits and guide experimental optimization. This study reviews the categories of synthetic genetic circuits (such as gene switches, oscillators, feedback circuits, etc.) and their biological mechanisms, with a focus on the application of ordinary differential equation (ODE) models, stochastic modeling, and network topology dynamics models in circuit modeling. We expounded on the estimation of model parameters, sensitivity analysis, and the integration methods of experimental data and models, and compared the characteristics of numerical simulation algorithms and commonly used software tools (such as MATLAB, COPASI, BioNetGen, etc.). Through the discussion of the steady-state, oscillation behavior, multiple steady-state and bifurcation analysis of system dynamics, the understanding of the influence of positive and negative feedback mechanisms on system stability is deepened. In addition, we took the classic synthetic gene oscillator Repressilator as a case to conduct modeling and simulation analysis, and compared the model predictions with the experimental data. Finally, the application prospects of synthetic genetic circuits in the fields of bioengineering and medicine were summarized, and the future directions of promoting the design of synthetic circuits with the help of model optimization and artificial intelligence-assisted design were prospected. Research shows that mathematical modeling and computational simulation have become key tools for the study and design of synthetic genetic circuits, providing a theoretical basis and practical guidance for the engineering transformation of complex biological systems.

**Keywords** Synthetic genetic circuit; Mathematical modeling; Ordinary differential equation; Gene oscillator; Feedback regulation

## 1 Introduction

Synthetic biology is all about "assembly" - assembling new life systems with standardized biological components (Vazquez-Vilar et al., 2023) - to put it simply, it is about making cells act according to human designs. In this process, the synthetic genetic circuit is like the "circuit board" of the cell, determining whether it can follow instructions. A typical circuit usually consists of promoters, regulatory proteins and various regulatory elements. When these parts are combined, they can control the switching logic of the target gene. In other words, as long as these components are reasonably selected, cells can perform logical operations like computers, thereby precisely controlling their behavior.

Modeling is actually a kind of "rehearsal" in synthetic biology; Toward predictive engineering of gene circuits. It enables people to see in advance the possible manifestations of genetic circuits, avoiding detours and reducing the waste of experimental costs. Because gene regulatory networks are often nonlinear, have feedback, and change over time, it is almost impossible to judge their dynamic behavior with the naked eye. So The mathematical model comes into play. It enables us to conduct "fake experiments" on the computer to infer the consequences of the design and see if the system will be stable or oscillate under different parameters. For instance, if you want to create a gene oscillator, the model can help you figure out how strong the feedback is needed to make it actually vibrate.

It can be said that mathematical models provide researchers with a "testing ground" for understanding complex biological processes and are key tools for designing bistable switches, sensors, and even more complex circuits. However, there is a dilemma in this matter: the model is either too simple to grasp the key points or too complex

for anyone to calculate. Researchers have to strike a balance between "sufficiency" and "understanding", which is known as "moderate simplification". Overall, the value of modeling lies not only in predicting experimental results, but also in revealing why the system operates in that way and helping to improve the robustness of the entire design (Synthetic biology).

## 2 The Basic Principles of Synthetic Genetic Circuits

### 2.1 Definition and classification of synthetic genetic circuits

Synthetic genetic circuits are actually a kind of artificially designed gene regulatory network, pieced together by some biological components, allowing them to "act" in cells according to the pre-designed logic. There are quite a few types of these circuits, mainly depending on their functions and dynamic performance (Gao et al., 2023). The most frequently mentioned one is probably the gene switch. This type of circuit is somewhat like a "memory button", capable of switching between two stable states, either on or off. The classic bistable design is achieved through two mutually inhibitory genes. As early as 2000, researchers made the first such artificial genetic switch, successfully switching the expression of two genes back and forth in *Escherichia coli*, which can be regarded as the beginning of synthetic gene switches. Later, similar switches were used as cell memory units or state transition modules (Rombouts and Gelens, 2021; Xu et al., 2022). Unlike a switch, a gene oscillator is more like a "biological metronome". Researchers constructed the renowned "three-gene loop oscillator" Repressilator, which enables the expression of proteins to fluctuate periodically within cells. Later, people improved many versions, such as adding positive feedback to make the oscillation stronger and more stable. Such oscillation circuits are very practical when studying issues such as biological rhythms and cell clocks.

### 2.2 Composition and function of circuit components (promoter, suppressor, activator)

Synthetic genetic circuits can be understood as a set of carefully arranged "biological parts" that are connected to each other and play different roles respectively. One of the most crucial components is the Promoter, a DNA sequence that attracts RNA polymerase to initiate transcription. Its "strength" directly affects the expression level of the subsequent genes. Some starters remain on all the time and are called constitutive starters. Some require signals or regulatory proteins to be triggered and belong to the controllable type. Near the promoter, there are often operons or other regulatory sequences, which are the "landing points" for regulatory protein binding. When the repressor (repressor protein) adheres, it blocks the pathway of RNA polymerase, causing transcription to stop. The most typical example is the lactose manipulation subsystem. Once the LacI protein binds to the operation sequence, the gene transcription is blocked. However, if there are exogenous inducers, such as isopropyl thiogalactoside (IPTG), it can inactivate LacI, thereby "unlocking" the circuit. The opposite role is the Activator. It can help RNA polymerase bind promoters more easily, making transcription smoother. Like the AraC in the arabinose operon, it can be transformed into an activator when there is an inducer, thereby enhancing the expression of downstream genes. In addition to these protein factors, small molecule inducers themselves are also commonly used "signal switches" in design, controlling gene activity by altering the conformation of repressors or activators. Designers usually mix and match these components.

### 2.3 Biological mechanisms of typical genetic circuits

In fact, different types of synthetic genetic circuits each have their own unique "operating logic". Take the genetic bistable switch as an example. Its core is not complicated - the products of two genes suppress each other, forming a double negative feedback structure. As long as one party's expression slightly gains the upper hand, it will further suppress the other party, pushing the system into a unilateral stable state. Either A is strong and B is weak, or the other way around. The middle state is hard to maintain, just like a seesaw, it will soon tip to one side. This design enables the system to switch clearly between "on" and "off". The  $\lambda$  switch of the bacteriophage is a classic example. The two proteins, Cro and CI, inhibit each other, maintaining the two different fates of lysogenation or dissolution. However, the logic of the oscillator is completely different. The three-gene circuit represented by Repressilator - A inhibits B, B inhibits C, and C then comes back to inhibit A - is like an endless "chase game". Because there is a time delay in each step of the reaction, the system does not stop at a fixed point but keeps cycling. A rises, B falls, C then rises, and then it's A's turn to be suppressed. As long as the delay is long enough and the inhibitory effect is "steep" enough, this negative feedback loop can continuously generate periodic

oscillations. Otherwise, the system will soon "lose steam" and fall into a steady state. To make oscillations more stable, researchers often need to artificially increase delays or enhance nonlinearity during the design process. The behavior of the feedforward loop is more like a "brief flash". Take the incoherent feedforward loop (I1-FFL) as an example. The upstream regulatory factor X, on the one hand, directly activates the target gene Z, and on the other hand, it activates an intermediate regulatory factor Y, which in turn inhibits Z. So, the expression of Z is first rapidly pushed up by X, but as Y gradually accumulates, it will then push Z down. The result is a brief "pulse-like" expression. Both experiments and models show that this structure will only trigger a one-time response under continuous signals and then automatically shut down, making it very suitable for cells to recognize transient stimuli.

### 3 Theoretical Basis of Mathematical Modeling

#### 3.1 Application of ordinary differential equations (Odes) in genetic circuit modeling

Ordinary differential equations (Odes) are almost the "old tools" for studying the dynamics of genetic circuits. Its idea is very straightforward. It regards gene expression as a series of reactions and uses equations to describe the changes in mRNA or protein concentrations over time (Turpin et al., 2023; Spartalis et al., 2024). The advantage of the ODE model lies in its ability to clearly demonstrate which parameters in the system are the most critical, such as synthesis rate, degradation rate, and regulation intensity, and how they jointly determine behavior. However, the initial assumptions of such models were actually quite idealized. It regards molecular concentration as a continuous quantity, ignoring the random fluctuations when the number of molecules is small, nor does it take into account the dilution effect brought about by cell growth or division. However, despite this, ODE remains the most commonly used modeling method, featuring fast calculation, clear form, and easy analysis of results.

#### 3.2 Stochastic models and noise analysis

In the real cellular world, gene expression does not strictly follow the average value. When the number of molecules is small, even tiny thermal noise can be amplified, resulting in significant expression differences among different cells (Goetz et al., 2025). To describe this randomness clearly, researchers usually do not rely solely on deterministic models but introduce stochastic modeling methods. The most commonly used theoretical framework is the master equation of chemistry (CME), which describes the time evolution of a system in a certain state (such as the number of molecules) in a probabilistic way. However, this equation is too complex to be solved directly. Therefore, people more often use various approximate or numerical methods, such as Monte Carlo simulation, Gillespie algorithm, and fluctuation decomposition approximation, etc. The Gillespie algorithm is the most classic one. It simulates molecular events one by one according to the probability of each reaction occurring, allowing the evolution trajectory of the system to naturally "emerge". With it, one can observe the random fluctuations of gene circuits on a time scale. For instance, in a negative feedback loop, feedback can be used to suppress the fluctuation range of the target protein expression, thereby verifying the robustness of the system (Müller et al., 2025). To theoretically analyze these fluctuations, researchers have also developed analytical methods such as linear noise approximation (LNA), breaking down the system into average trends and adding small random disturbances, and then linearizing to solve for noise intensity. This method can calculate indicators such as the Fano factor and correlation function to measure the noise level of the system. Through these analyses, it was found that negative feedback often effectively reduces noise, while positive feedback tends to amplify noise and even cause differentiation in cell populations. But noise is not always a bad thing. For instance, some cell switching phenomena, where gene expression randomly jumps between high and low states, can only be explained by random models.

#### 3.3 System dynamics model based on network topology

In addition to describing molecular changes using reaction rates, researchers often look at the problem from a structural perspective - no longer focusing on the details of each reaction, but studying the network shape of "who regulates whom" among genes. Such a model pays more attention to the overall organization of the system rather than the numerical value of each parameter. Several common methods include Boolean networks, qualitative network models, and topology analysis based on graph theory. The idea of Boolean networks is very straightforward: simplify each gene state to "on (1)" or "off (0)", and use logical rules to determine the state at the

next moment. It is not precise, but it is very practical when dealing with large-scale regulatory networks, such as simulating the gene regulatory relationships during development to see which stable states the system will eventually converge to - these "attractors" often correspond to different cell types. Although the Boolean model does not care about temporal continuity, it can help designers verify at an early stage whether the logic of the loop works and whether the output meets expectations. There are still some "compromise" solutions between the Boolean model and the ordinary differential equation model. Just as multi-valued logic models allow genes to have more than two expression levels, Petri net models describe system changes in the form of state transition diagrams. This type of method is particularly useful when the experimental data is insufficient or the parameters are difficult to measure, allowing people to make a qualitative judgment on the system behavior. On the other hand, there is another type of research that focuses on the characteristics of the network topology itself. For instance, motif analysis - within the vast gene regulatory network, certain small structures (motifs) tend to recur.

## 4 Model Parameters and System Identification Methods

### 4.1 Parameter estimation and sensitivity analysis

After the model is built, the real troubles often just begin - where do the parameters come from and how are they determined? These parameters include the transcription rate, translation rate, protein degradation rate of genes, as well as the binding constant and Hill coefficient of regulatory effects and the like. Usually, they are not set out of thin air but are "dug out" from experimental data. This process is actually like doing optimization: adjusting a set of parameters to minimize the error between the model output and the experimental observation. The most commonly used methods include least squares fitting, maximum likelihood estimation, and the more complex Bayesian inference. For example, when studying gene oscillators, the protein concentration time series measured experimentally can be used to fit the transcription rate and inhibition constant through the nonlinear least squares algorithm, so that the simulated oscillation curve coincides with the real curve as much as possible (Liu and Niranjan, 2017). But the problem is that parameters are not always "precisely identified".

Sometimes different combinations of parameters can yield almost the same result, which is called poor identifiability. In such cases, researchers usually conduct sensitivity analyses to see which parameters truly determine the behavior of the model (Cao and Grima, 2019). Local sensitivity analysis is to observe the changes within a small range, calculate the partial derivatives of the output with respect to the slight perturbations of each parameter, and identify those key parameters that "get chaotic at the slightest movement". Global sensitivity analysis is even more "crude". It measures the impact of parameter uncertainty on model output fluctuations by sampling over a wide range in the parameter space, such as using Sobol indices to calculate the contribution of each parameter to the output variance. In this way, important parameters can be screened out, and subsequent experiments can focus on measuring or optimizing them, while those parameters with little impact can be simply processed to improve estimation efficiency.

### 4.2 Integration methods of experimental data and modeling data

Getting experimental data truly "integrated" into the model is a crucial step in making the model more reliable and closer to real biological systems. But this matter is not that straightforward. The first step usually involves processing the data, as the units and scales of the things measured in different experiments vary. For example, values such as fluorescence intensity, protein concentration, and transcription rate often vary greatly. Without normalization, it is simply impossible to match the model output (Wang et al., 2014). Therefore, researchers often first convert the fluorescence signal into a relative expression level and then map it to the concentration variable in the model.

However, having only one type of data is often not enough. The experimental results of synthetic genetic circuits usually come from several sources: time series data, steady-state point data, and sometimes distribution information at the single-cell level. The usage of different data types also varies. Time series can be directly used to fit dynamic processes. The steady-state value can be regarded as the constraint condition of the equation; Distributed data, such as the differences in expression between cells, are often used to adjust the noise terms or parameter distribution assumptions in the model. A more complex integration approach is to use the Bayesian

framework, viewing experimental observations as "correction signals" for the model, and updating the posterior distribution through the prior distribution and likelihood function. In this way, different types of data can all be put into the same logical system to achieve multi-dimensional fusion in a probabilistic manner (Meyer et al., 2014).

In recent years, machine learning has also begun to be involved in the integration of models and experiments. Some studies use machine learning to first predict the approximate range of model parameters and then guide experiments to focus on measuring key parameters, saving a considerable amount of work. Some people use neural networks as "proxy models" to first learn the complex patterns in experimental data and then combine them with mechanism models to improve the accuracy of predictions. Another more systematic approach is the so-called "DBTL loop" (Design-Build-test-learn), which means first using the model to predict Design experiments, then feeding the data back to the model for updates after the experiments are completed, followed by the next round of improvements. Researchers worked in this way. They designed a model-experimental closed-loop to ensure that the synthetic loop maintained stable functionality across different hosts.

However, there are also many pitfalls in the integration of data and models. The experiment itself is noisy and the model is approximate, so the uncertainty of the results is almost inevitable. Therefore, when integrating, researchers usually conduct confidence interval evaluations or posterior predictive tests to see if the model can truly explain all the data. If there is any type of data that never matches, it is necessary to consider whether the model is too simplified or a key parameter is missing. Sometimes, new mechanisms also need to be added to the model, such as introducing host growth effects or resource competition modules.

#### 4.3 Model validation and error evaluation criteria

The model has been built and the parameters have been determined, but this does not mean it is reliable. The next step is to verify - to see if the model can actually live up to its promises. Usually, people will first compare the model's prediction results with independent experimental data to see if it can reproduce the real trend or numerical characteristics. For instance, is the steady-state expression level calculated by the model close to that measured experimentally? Can the period of the oscillation model match the rhythm observed experimentally? If none of these match, there is an 80% chance that there is a problem with the model structure or parameter Settings, and it needs to be corrected back.

Just looking at the picture is not enough; there must be quantitative standards for the gap. Several commonly used indicators by researchers include mean square error (MSE), mean absolute error (MAE), and coefficient of decision ( $R^2$ ), etc., which are used to measure how far the prediction is from the experiment. For dynamic systems, methods such as Dynamic Time Warping (DTW) are also used to compare the matching degree between the simulated curve and the experimental curve (Dahlquist et al., 2015). Sometimes people do not only look at the overall error but also separately examine certain key features, such as the deviation of peak values, amplitudes or steady-state values. Furthermore, statistical tests (such as chi-square test, F-test, etc.) can be used to determine whether the error of the model is within a reasonable range.

However, model validation is not merely about "matching the data". A good model should remain accurate even under unfitted conditions, such as when the concentration of the inducer, temperature, or environmental background is changed, and the prediction remains reasonable. If a model can only perform well on the training data but crashes once the conditions change, it is mostly overfitting and requires the deletion of redundant parameters or the addition of regularization. Sensitivity analysis can also come in handy here - if the model is overly sensitive to a certain parameter and the parameter itself is not accurately estimated, the credibility of the entire prediction will be compromised. At this point, either increase the experimental accuracy or reconstruct the model to make it less affected by such parameters.

## 5 Simulation and Calculation Methods

### 5.1 Numerical simulation algorithms (euler, runge-kutta, etc.)

When studying genetic circuits, numerical simulation is almost an unavoidable step. Most models are composed of a bunch of nonlinear differential equations. It is basically impossible to write analytical solutions for them, so



they can only be calculated step by step through "integration" by a computer. The most commonly used ones are the time-domain integration algorithms for ordinary differential equations, such as the simplest Euler method or the more refined Runge-Kutta method. The Euler method is quite straightforward in its approach, using the current derivative as a linear approximation to move forward one step. However, it is not very accurate, and once the equation becomes too "rigid", the result is prone to divergence. In contrast, the Runge-Kutta series, especially the fourth-order RK4, is more stable and accurate. It calculates the slope four times at each step and takes the weighted average. In this way, the error is much smaller, and even if the step size is slightly larger, a stable result can be achieved (Manninen et al., 2006).

However, not all systems are honest. Some genetic circuits are very "stubborn", featuring both fast-changing processes, such as protein degradation which is completed within a few minutes, and slow-changing processes, such as cell growth dilution which takes several hours. When encountering rigid systems with such huge time scale differences, explicit algorithms often fail and implicit methods have to be adopted, such as the implicit Euler or Gear method (BDF method). They are more troublesome to calculate, but in this case, they can keep the values stable and won't accidentally crash. For non-rigid cases, Runge-Kutta is sufficient. However, if efficiency is to be pursued, some people also use adaptive step size algorithms, such as Runge-Kutta-Fehlberg. It can automatically adjust the step size according to the curvature of the solution. Take small steps where the changes are drastic and large steps when it is stable. This is fast and accurate (Pájaro et al., 2017).

In addition to the integration of the ODE, there is another type of stochastic simulation. For instance, the well-known Gillespie algorithm simulates each event one by one with great precision, but at the cost of an explosive amount of computation. This led to the emergence of approximate algorithms such as the  $\tau$ -transition method and the virtual response method, which package and process many small events, making long-term simulations feasible. If the model takes spatial distribution into account, it becomes even more complex and requires the use of partial differential equations (Pdes) or spatial stochastic models for calculation. The finite difference method and the finite element method can all be put into use. Or use grid random walks to simulate molecular diffusion. If one wants to study spatial patterns within a group, such as how bacteria self-organize to form fringes, cellular automata or Agent-based models must be used. Such methods can make complex spatial behaviors clear at a glance.

## 5.2 Software platforms and computing tools (MATLAB, COPASI, BioNetGen, etc.)

To simulate and analyze genetic circuits more conveniently, researchers have long developed a complete set of software "toolboxes". Different platforms have their own preferences. Some pursue flexibility, some focus on visualization, and others are specifically designed to serve automation. For instance, MATLAB is almost one of the most common numerical computing environments. The Simulink and SimBiology toolboxes that come with it enable users to build models directly using reaction diagrams. The system automatically generates differential equations, making it very convenient to run simulations, perform parameter scans, and conduct sensitivity analyses (Figure 1) (Sequeiros et al., 2023). What many researchers like about MATLAB is that it can not only perform theoretical calculations but also be used to verify experimental data. A complete process does not require a change of environment.

However, MATLAB is, after all, commercial software, and there are quite a few free options available. COPASI (Complex Pathway Simulator) is a fully functional open-source option. It supports ordinary differential equations and stochastic simulation, and can perform parameter fitting, steady-state analysis, and even bifurcation analysis. The greatest advantage is the user-friendly graphical interface. Modeling, importing SBML files, and viewing the results directly without writing code (Welsh et al., 2018). Due to its low operational threshold, COPASI is particularly widely used in the modeling of metabolic networks and gene regulatory networks. Another situation is when the model is too complex, involving molecular binding or multi-site modification, and it is almost impossible to write equations manually. At this time, BioNetGen comes in handy. It enables users to define interactions in a regular way, automatically generate reaction networks and equations, and is highly suitable for the design of signal pathways or molecular interaction classes.

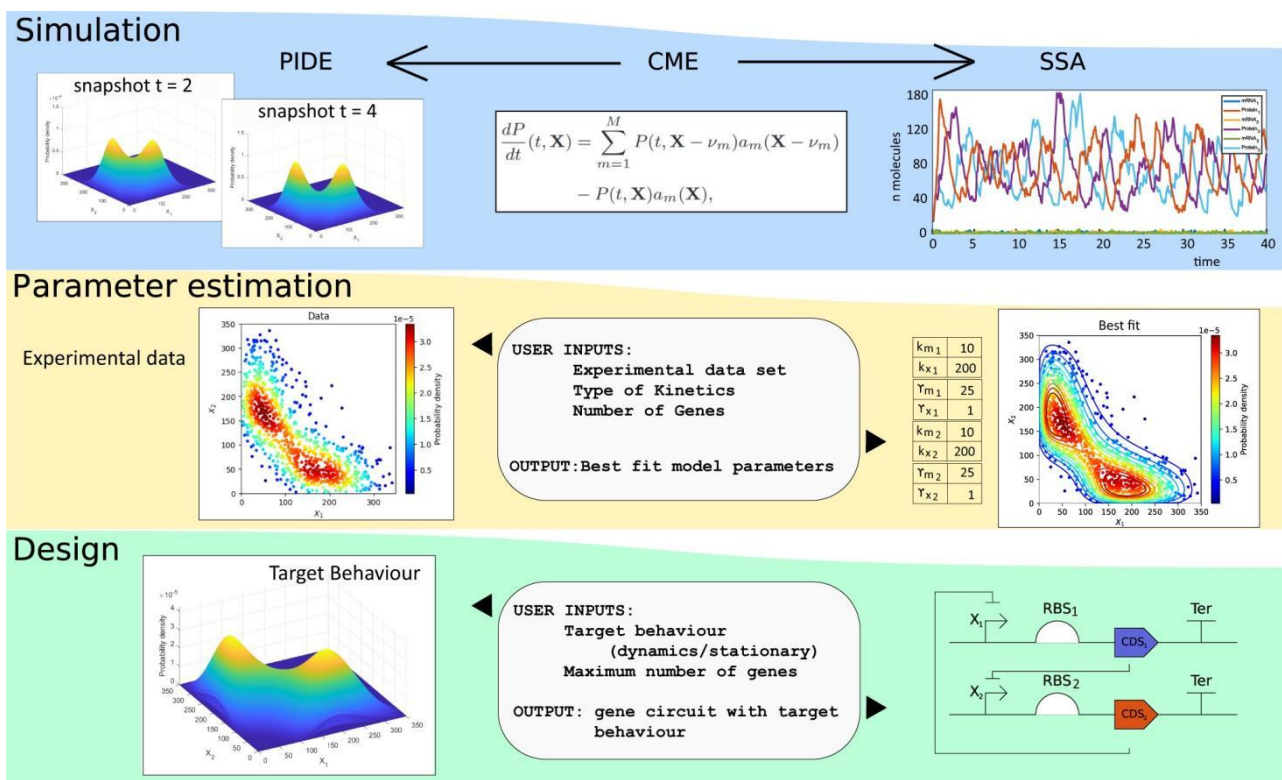


Figure 1 Main features of IDESS: (i) simulation of biocircuit dynamics by Semi-Lagrangian (PIDE model) or SSA Algorithms, (ii) parameter estimation from experimental data for model identification, and (iii) automated design of biocircuits, delivering synthetic gene circuits (topology and parameters) with predefined target behaviors. IDESS applies CPU and GPU parallel implementations of stochastic simulation and global optimization to accelerate computing (Adopted from Sequeiros et al., 2023)

If the goal is to develop genetic logic circuits or leans more towards synthetic design, iBioSim is a good open-source tool. It integrates functions such as logic circuit modeling, ordinary differential equations and stochastic simulation, and is suitable for researchers who want to draw circuit diagrams and run models (Myers et al., 2009). In contrast, Tellurium is more popular among users who prefer the Python environment. It integrates Antimony language, roadrunner solver, etc., and can directly load models, run simulations, optimize parameters and plot using scripts (Choi et al., 2018). Unlike COPASI, Tellurium does not rely on a graphical interface but is more suitable for scenarios such as batch analysis, automation, and flexible computing.

### 5.3 Visualization and model optimization strategies

After building the model and running the simulation, the diagrams drawn are often the ones that can best help people "understand" the system's behavior. Complex multi-variable dynamic processes are hard to understand the patterns just by numbers, but good visualization can immediately bring the problem to light. For instance, a curve of protein concentration varying over time can visually reveal the period and amplitude of the oscillation. Phase diagrams can display the switching trajectories of the system between different states. The heat map can immediately reveal which parameters have the greatest impact on the output (Madsen et al., 2019). Time series diagrams, phase diagrams and heat maps are the three most commonly used forms, each with its own application - the former is used to observe dynamics, while the latter two are used to assess the stability and sensitivity of the system. Especially during the parameter adjustment stage, researchers often rely on graphs to determine whether the system's direction is reasonable.

However, visualization is not only for presenting results but can also, in turn, guide model optimization. For instance, in a high-dimensional parameter space, we can draw the distribution of the objective function to see if there are regions similar to "basins", which can help select the initial point or algorithm. If there are too many parameters, principal component analysis can also be used to project the high-dimensional information into a two-dimensional graph to see which parameters dominate the system behavior. These graphs are often more

persuasive than a bunch of tables. Sometimes, static diagrams are not intuitive enough, so interactive visualization becomes the trend. Adjust the parameters in real time with the slider and immediately see the output changes. This "what you see is what you get" approach is particularly suitable for teaching or prototype testing. Software like COPASI and SimBiology of MATLAB all offer such functions, allowing users to conduct what-if analysis at will and experience the impact of parameter changes on the system. In terms of model optimization, the task has shifted from "understanding" to "improving". Parameter optimization is the most common form. Researchers often use methods such as evolutionary algorithms and simulated annealing to automatically search for parameter combinations (Merzbacher et al., 2023). For instance, if one wants to keep the oscillator period stable within a specific range, the period deviation can be defined as an objective function, and the ideal value can be gradually approximated using a genetic algorithm. What is more complex is structural optimization, adjusting the topology, adding or deleting regulatory components, and even allowing the computer to "evolve" itself into a network that meets the target. Genetic programming and other intelligent algorithms have begun to show their prowess in this regard.

## 6 System Dynamics and Steady-State Analysis

### 6.1 Dynamic characteristics of steady-state and oscillation behaviors

The behavior of synthetic genetic circuits is not always "stable"; many times, they dance as if they have their own rhythm. The two most common dynamic characteristics are steady state and oscillation. The so-called steady state refers to the situation where, after a certain point in the system's operation, the concentrations of each component no longer change, just like the water surface calming down. At this point, the right side of the dynamic equation of the gene circuit equals zero, and the solution obtained is the steady-state point of the system. Many engineered circuits pursue this monostable characteristic because in this way, the system can return to its original level after being disturbed, and the output is as stable as a straight line. A typical example is the steady-state loop with negative feedback, whose output is almost unaffected by input fluctuations and can achieve a robust constant output (Likhoshvai et al., 2020).

However, a stable state not only needs to "exist", but also "stand firm". To determine whether a steady state is stable, one needs to linearize the equation to calculate the Jacobian matrix and then look at the real part of the eigenvalues. If all are negative, it indicates that small disturbances will not overthrow the system, and the steady state is locally attractive - such points are the "operating points" where we hope the system will settle. But sometimes, stability is not the goal at all. Some loop designers, on the contrary, hope that the system will "move", such as oscillating. Oscillation means that the system cannot find a point of eternal rest but repeats itself in a closed orbit, which is known as a limit loop. Such behavior is very common in biological clocks or periodic signal generators (Zhu and Shen, 2021). Its two core features are the oscillation period and amplitude. Model analysis often tells us what controls these features - factors such as negative feedback delay, synthesis rate, and degradation rate can all affect the cycle. For instance, the NF- $\kappa$ B oscillation circuit can switch from rapid small oscillation to slow large oscillation by changing the expression intensity of the inhibitory gene, achieving adjustable period.

### 6.2 The impact of positive and negative feedback mechanisms on system stability

In genetic circuits, feedback regulation is almost the "soul" mechanism. For a system to be stable, controllable, or capable of generating complex dynamic behaviors, it is indispensable. Negative feedback is the most common form. It is somewhat like a thermostat: when the output is too high, the system automatically "brakes". When the output is too low, it will "step on the accelerator" again. This self-regulation can keep the output near a balance point and also make the system less sensitive to external disturbances. For example, in a negative feedback loop, the gene product will in turn inhibit its own generation. The inhibition strengthens when the concentration increases and weakens when the concentration decreases, eventually achieving a dynamic equilibrium (Kelly et al., 2017). Such a design can also enable the system to respond more quickly and have smaller steady-state errors. Researchers constructed a feedback control system using dCas9. When the expression burden is too heavy, dCas9 will inhibit the transcription of some genes, allowing cells to automatically reduce stress and making growth and product output more stable.



However, negative feedback is not omnipotent. Theoretically, it accelerates the system's return to a steady state by changing the roots of the system's characteristic equation, pulling the real part to a negative value. However, if the feedback is too strong and there is a time delay, it may cause the system to over-correct and even start oscillating - at this point, the characteristic roots may pass through the virtual axis, resulting in Hopf bifurcation. That is to say, feedback that is "too frequent" may instead make the system unstable (Hu and Murray, 2019). Therefore, when designing, it is necessary to strike a proper balance between feedback gain and delay.

On the contrary, positive feedback is like "adding fuel to the fire", which amplifies the deviation of the system. When a certain gene product can promote its own expression, even a small fluctuation may be rapidly amplified, and the system is pushed in a certain direction. Weak positive feedback can increase the signal output strength, but if it is too strong, it often leads to bistability or multistability, that is, there are two or more stable points in the system - high and low expression states coexist (Sun et al., 2022). This type of mechanism is precisely suitable for constructing gene switches, but from a control perspective, it is very "dangerous" because the system can be easily pushed to another steady state by noise or initial conditions. For this reason, in engineering, a small negative feedback is sometimes added to the strong positive feedback to form compound regulation, so that the system is not too extreme (Loman et al., 2025).

### 6.3 Bifurcation analysis and multistable phenomena

When studying genetic circuits, people often find that the behavior of the system will suddenly "change", as if a certain switch has been activated. This phenomenon is not accidental but rather a result of parameter changes triggering what is called a "bifurcation". The purpose of bifurcation analysis is to figure out when and how a system jumps from one dynamic state to another when its parameters change slowly. For synthetic genetic circuits, there are mainly two common types of bifurcations: saddle bifurcation and Hopf bifurcation. The former usually implies the emergence or disappearance of a new steady state, while the latter marks the birth or extinction of oscillations.

Take bistable switches as an example. This system will exhibit different stable structures under different concentrations of inducers. At the beginning, the system had only one steady state. When the concentration of the inducer gradually increases, two stable equilibrium points and one unstable equilibrium point will suddenly appear after a certain critical point - a typical saddle junction bifurcation (Leon et al., 2016). Experimentally, this corresponds to the different states that cells exhibit when the concentration of the inducer is increased or decreased, which is known as the "lag" phenomenon. The reaction trajectories of cells are different when they rise and fall. Through bifurcation analysis, the width of this lag interval can be precisely identified. When designing the gene switch, engineers hope that this area is wide enough so that the system switches more cleanly and is more noise-resistant (Dey and Barik, 2021).

## 7 Case Study: Repressilator Model for Synthetic Genes

### 7.1 The biological design and modeling framework of repressilator

Repressilator can almost be regarded as the "pioneering work" of synthetic biology. It was the first artificially designed gene oscillator, successfully constructed by Elowitz and Leibler in 2000. The structure of this system is actually very simple - the three genes are connected end to end and inhibit each other. The product of gene *A* inhibits *B*, the product of *B* inhibits *C*, and *C* in turn inhibits *A*, forming a closed loop. Each promoter is controlled by the repressor protein of the upstream gene, and thus the entire circuit forms a delayed negative feedback. It is precisely this delay that makes the system never "catch up with itself", thus causing continuous oscillation (Tyler et al., 2019).

The process is roughly as follows: when *A* begins to be expressed in large quantities, its protein gradually accumulates, and after a period of delay, it inhibits *B*. After *B* was suppressed, *C* took the opportunity to rise. Then the product of *C*, in turn, inhibits *A*, causing *A* to decline. This cycle repeats itself. The system can never stay stable at a certain point but keeps circling along a periodic trajectory, with the concentrations of the three proteins fluctuating one after another. Researchers selected three different sources of repressor proteins - TetR,  $\lambda$ CI and LacI - and their respective exclusive promoters back then, so as to avoid cross-interference (Henningsson et al.,

2020). Meanwhile, they also carefully controlled the degradation rate of the protein, keeping the oscillation period on a scale of several hours, which was convenient for experimental observation.

However, the original Repressilator was not perfect. In *Escherichia coli*, although periodic oscillations have indeed been observed, the phases between cells are not synchronized, and the amplitude of the oscillations also decays over time. Later, many improved versions emerged. For instance, some people have attached degradation labels to the repressor proteins to make them break down more quickly, thereby reducing the cycle fluctuations. Some people have joined the positive feedback pathway or used group communication signals (such as acyl high-serine lactone) to achieve oscillation synchronization between cells (Zhang et al., 2022). These adjustments make the system more stable, neater and closer to the ideal periodic behavior.

## 7.2 Mathematical equations and parameter settings

Based on the design topology of Repressilator, we can write the corresponding dynamic equation for each gene. Let  $[A]$ ,  $[B]$ , and  $[C]$  represent the concentrations of repressor proteins encoded by genes  $A$ ,  $B$ , and  $C$ , respectively. Then their generation and degradation can be described by the following ordinary differential equation:

$$\begin{aligned}\frac{d[A]}{dt} &= \alpha_A \cdot f_A([C]) - \beta[A] \\ \frac{d[B]}{dt} &= \alpha_B \cdot f_B([A]) - \beta[B] \\ \frac{d[C]}{dt} &= \alpha_C \cdot f_C([B]) - \beta[C]\end{aligned}$$

Among them,  $\alpha_A, \alpha_B, \alpha_C$  represent the maximum synthesis rate of each gene in the absence of repression (determined by promoter strength and translation efficiency), and  $\beta$  is the assumed constant of the same protein degradation (dilution) rate. The function  $f_X(\text{repressor})$  describes the inhibitory effect of repressor proteins on promoter activity. Common Hill function forms:

$$\begin{aligned}f_A([C]) &= \frac{1}{1 + ([C]/K_C)^n} \\ f_B([A]) &= \frac{1}{1 + ([A]/K_A)^n} \\ f_C([B]) &= \frac{1}{1 + ([B]/K_B)^n}\end{aligned}$$

Here,  $K_C, K_A$ , and  $K_B$  represent the concentrations at which the repressor protein reduces the promoter activity to half (a parameter for measuring repressor affinity), and  $n$  is the Hill coefficient, reflecting the degree of the synergistic effect. For dimer repression, generally,  $n=2$  is taken. The above system of equations constitutes the basic model of Repressilator. In terms of parameter selection, to achieve oscillation, it is necessary to make the closed-loop gain large enough and introduce an effective delay. The general experience for adjusting parameters is that high expression intensity (large  $\alpha$ ) and high repression efficiency (small  $K$  and large  $n$ ) help to meet the Hopf bifurcation conditions of oscillations (Verdugo, 2018), while a moderate degradation rate determines that the oscillation period is within an appropriate range. If the degradation is too slow and the system lag is too strong, the cycle will be too long and even chaos may occur. The degradation is too rapid and reduces the delay, which is not conducive to continuous oscillation (Sun et al., 2023).

Researchers accelerated degradation by adding *ssrA* tags at the end of the protein sequence in their experiments, which was equivalent to increasing  $\beta$ , and ultimately achieved an oscillation period of approximately 2 to 3 hours. In the model, the corresponding  $\beta$  is set to be of the same order as the cell division rate (approximately  $0.5 \text{ h}^{-1}$ ). The maximum transcription rate  $\alpha$  can be estimated based on the strength of the promoter and the available amount of RNA polymerase, such as dozens of protein molecules per minute. To reflect qualitative behavior, parameters are often dimensionless. In Elowitz's original paper,

dimensionless parameters such as  $T = \alpha/\beta$  (steady-state yield multiple without repression) and  $\mu = \text{Protein\_decay}/\text{mRNA\_decay}$  were used. Dimensionless models can more clearly analyze the influence of parameter combinations on dynamics. By numerically simulating the above model, the phase diagrams of the system under different parameters can be obtained. For instance, when  $K$  is fixed and  $\alpha$  is added, the simulation shows that the system converges to a steady state when  $\alpha$  is less than a certain threshold. Self-oscillation occurs when the threshold is exceeded, which is consistent with the results of Hopf bifurcation analysis. If  $\alpha$  continues to increase too much, the oscillation may become irregular or even disappear, corresponding to the occurrence of high-order bifurcations in the model. To improve the consistency between the model and the experiment, some model improvements were proposed in subsequent studies. For instance, by incorporating dissipative processes or RNA dynamics: expand the equation to six (three mRNAs + three proteins) to make the phase relationship of oscillations more accurate. However, adding mRNA variables also introduces new parameters such as the degradation rate of mRNA. Before obtaining the corresponding experimental data, typical values in the literature (such as the half-life of mRNA in a few minutes) are generally adopted. Overall, a reasonable set of parameter Settings for the Repressilator model might be:  $n = 2$ ,  $100 \text{ nM/h}$ ,  $0.5 \text{ h}^{-1}$ ,  $\beta \sim 1 \text{ h}^{-1}$ ,  $K \sim 40 \text{ nM}$ , etc. By numerically integrating the model, the oscillation curve of protein concentration over time can be obtained. Adjusting parameters can change the period and amplitude of oscillation. Model equations and parameters can not only reproduce qualitative behavior but also be used to guide loop improvement.

### 7.3 Comparison and analysis of simulation results with experimental data

When conducting numerical simulations with Repressilator's model, the first thing that can be observed is that familiar sense of rhythm. Under appropriate parameter conditions, the system will automatically enter a nearly sinusoidal periodic oscillation state. The three repressor proteins will reach their peaks in sequence, with a temporal difference of approximately  $120^\circ$ . For instance, after A rises, it quickly reaches its peak, then B starts to rise and A drops. Then C comes in to drive the next cycle. The simulated curve seems to be constantly cycling around a restrictive loop. Based on Fourier analysis, the period  $T$  is approximately 2.5 hours, which is roughly the same as the 2-3 hours in the single-cell experiments with *Escherichia coli*. In terms of amplitude, the model can also show similar differences in the experiment - when the parameters change, the oscillation intensity changes (Figure 2) (Park et al., 2024).

When approaching the Hopf bifurcation boundary, the oscillation is easily worn away by the noise, and after several rounds, the amplitude decays to a steady state. In the deeper oscillation region, the model provides continuous fixed-amplitude oscillation, corresponding to the kind of "durable" oscillation observed under the improved experimental conditions. To see if the model is reliable or not, we compared the simulation results with the experimental data point by point. Potvin-Trottier et al. (2016) once tracked the fluorescence variation trajectory of single cells. We superimposed the A protein concentration curve predicted by the model on their data. The periods and waveforms of both were quite consistent - both were quasi-sine waves with a period of approximately 200 minutes. It's just that the amplitude of the model is slightly smaller, possibly because there are inherent differences among different cells in the experiment, and some cells oscillate even more vigorously. Later, we added a noise term to the model and ran random simulations using the Gillespie algorithm. As a result, the period and amplitude showed intercellular variations, which were closer to the degree of dispersion observed in the experiment.

## 8 Application and Future Outlook

Synthetic genetic circuits are no longer just conceptual devices in laboratories; they are now being regarded as "molecular logic units" capable of performing specific tasks. In the fields of medicine and bioengineering, its potential is regarded as extremely huge. The most typical example is cell therapy, especially CAR-T therapy. In the past, once T cells were activated, it was like a floodgate that had been opened; it was extremely difficult to close. Later, researchers attempted to incorporate synthetic gene circuits into it and use logical control to make it work "with a brain". The circuit can identify tumor-specific miRNA signals. Only when the correct "tumor

combination" is detected will T cells release cytotoxic molecules. The result was that in mice, these cells precisely cleared tumors but hardly touched normal tissues. Such examples show for the first time that genetic circuits can become "programmed drugs".

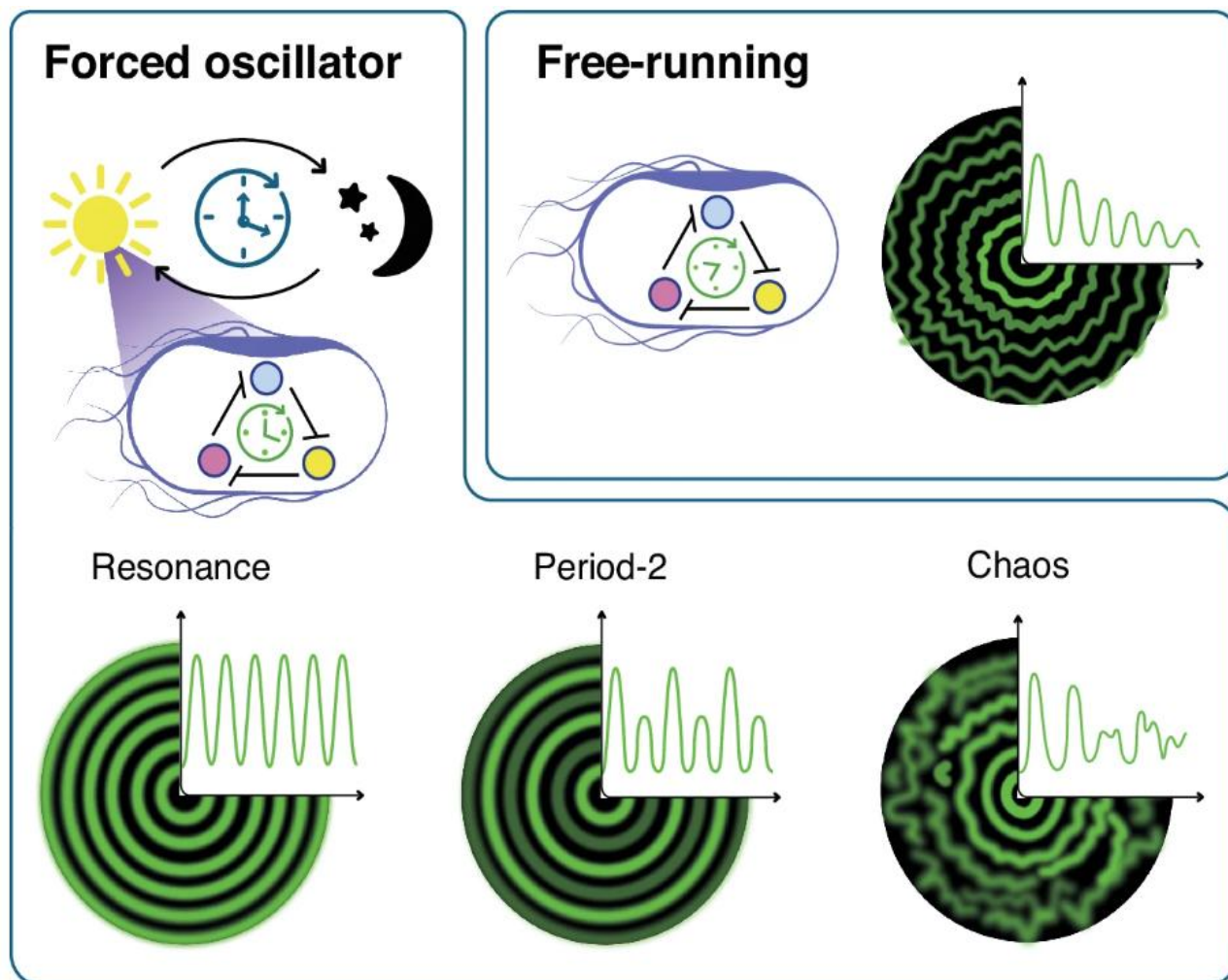


Figure 2 Complex dynamics of a forced oscillator are transformed to intricate spatial patterns (Adopted from Park et al., 2024)

However, putting such "intelligent systems" into living organisms or natural environments is no easy task. Safety is always a prerequisite. Engineered bacteria may mutate, escape, or have unexpected interactions with the host system. Therefore, many designs have added suicide modules or external control switches, allowing the system to "shut down" at any time in case of any problem. The interference of complex physiological environments is more difficult to predict, which is why mathematical models and multiple rounds of experimental verification are so important. They can tell us in advance whether the circuit will still work as expected under different conditions. For future "intelligent cells" to truly enter clinical practice, they must first learn to maintain stability in complex environments.

Looking ahead to the next stage, the status of models will only become more important, especially when they are combined with AI, the design approach may undergo a complete transformation. In the past, building a synthetic loop often relied on repeated trial and error through experiments. Now, model simulation can help us filter out a large number of inappropriate schemes, leaving only the most promising ones for experimentation. This is equivalent to moving half of the "design - build - test - learn (DBTL)" cycle into the computer. For instance, through the model, the response curves under different promoters or RBS intensities can be predicted in advance, thereby selecting a combination that is both fast and stable. This method significantly reduces the number of experiments. For complex circuits, it is a scenario where AI can truly shine. Machine learning can identify

nonlinear patterns, and deep models can reverse-engineer DNA sequences that satisfy specific dynamic characteristics. Nowadays, there are even studies that allow reinforcement learning algorithms to "evolve" their loop structures in virtual environments, enabling them to learn to resist noise or metabolic burdens on their own.

The study of synthetic genetic circuits is no longer a game for a single discipline; it requires biologists, engineers, mathematicians, and even computer scientists to collaborate on the same table. Future researchers must be proficient in both experiments and modeling. Only in this way can we handle increasingly complex systems and transform models from auxiliary tools into design engines. It can be foreseen that in the future, we will witness the emergence of more intelligent and stable artificial gene networks in the medical, industrial and ecological fields. At that time, synthetic biology will truly move from "being able to create" to "knowing how to create".

### Acknowledgments

The author extends sincere thanks to two anonymous peer reviewers for their invaluable feedback on the manuscript.

### Conflict of Interest Disclosure

The author affirms that this research was conducted without any commercial or financial relationships that could be construed as a potential conflict of interest.

### References

- Turpin B., Bijman E.Y., Kaltenbach H.M., and Stelling J., 2023, Efficient design of synthetic gene circuits under cell-to-cell variability, BMC Bioinformatics, 24(Suppl 1): 460.  
<https://doi.org/10.1186/s12859-023-05538-z>
- Cao Z., and Grima R., 2019, Accuracy of parameter estimation for auto-regulatory transcriptional feedback loops from noisy data, Journal of the Royal Society Interface, 16(153): 20180967.
- Choi K., Medley J., König M., Stocking K., Smith L., Gu S., and Sauro H., 2018, Tellurium: an extensible python-based modeling environment for systems and synthetic biology, BioSystems, 171: 74-79.  
<https://doi.org/10.1016/j.biosystems.2018.07.006>
- Dahlquist K., Fitzpatrick B., Camacho E., Entzminger S.D., and Wanner N.C., 2015, Parameter estimation for gene regulatory networks from microarray data: cold shock response in *Saccharomyces cerevisiae*, Bulletin of Mathematical Biology, 77(8): 1457-1492.  
<https://doi.org/10.1007/s11538-015-0092-6>
- Dey A., and Barik D., 2021, Potential landscapes, bifurcations, and robustness of tristable networks, ACS Synthetic Biology, 10(2): 391-401.  
<https://doi.org/10.1021/acssynbio.0c00570>
- Gao G., Bian Q., and Wang B., 2023, Synthetic genetic circuit engineering: principles, advances and prospects, Synthetic Biology Journal, 6(1): 45-64.  
<https://synbioj.cip.com.cn/EN/10.12211/2096-8280.2023-096>
- Goetz H., Zhang R., Wang X., and Tian X.J., 2025, Resource competition-driven bistability and stochastic switching amplify gene expression noise, PLoS Computational Biology, 21(4): e1012931.  
<https://doi.org/10.1371/journal.pcbi.1012931>
- Henningesen J., Schwarz-Schilling M., Leibl A., Gutiérrez J., Sagredo S., and Simmel F., 2020, Single cell characterization of a synthetic bacterial clock with a hybrid feedback loop containing dCas9-sgRNA, ACS Synthetic Biology, 9(12): 3377-3387.  
<https://doi.org/10.1021/acssynbio.0c00438>
- Hu C.Y., and Murray R., 2019, Design of a genetic layered feedback controller in synthetic biological circuitry, bioRxiv, 1101: 647057.  
<https://doi.org/10.1101/647057>
- Kelly C.L., Harris A.W.K., Steel H., Hancock E., Heap J., and Papachristodoulou A., 2017, Synthetic negative feedback circuits using engineered small RNAs, Nucleic Acids Research, 46(18): 9875-9889.  
<https://doi.org/10.1101/184473>
- Leon M., Woods M., Fedorec A.J.H., and Barnes C., 2016, A computational method for the investigation of multistable systems and its application to genetic switches, BMC Systems Biology, 10(1): 130.  
<https://doi.org/10.1186/s12918-016-0375-z>
- Likhoshvai V.A., Golubyatnikov V., and Khlebodarova T., 2020, Limit cycles in models of circular gene networks regulated by negative feedback loops, BMC Bioinformatics, 21(Suppl 11): 255.  
<https://doi.org/10.1186/s12859-020-03598-z>
- Liu X., and Niranjani M., 2017, Parameter estimation in computational biology by approximate Bayesian computation coupled with sensitivity analysis, arXiv Preprint, 1704: 9021.
- Loman T.E., Schwall C.P., Saez T., Liu Y., and Locke J.C., 2025, Mixed positive and negative feedback loops drive diverse single-cell gene expression dynamics, bioRxiv, 14: 632931.  
<https://doi.org/10.1101/2025.01.14.632931>



- Madsen C., Goñi Moreno Á., Palchick Z., Roehner N., Bartley B., Bhatia S., Bissell M., Clancy K., Cox R., Gorochowski T., Grunberg R., Luna A., McLaughlin J., Nguyen T., Le Novère N., Pocock M., Sauro H., Scott-Brown J., Sexton J., Stan G., Tabor J., Voigt C., Zundel Z., Myers C., Beal J., and Wipat A., 2019, Synthetic biology open language visual (SBOL Visual) version 2.1, *Journal of Integrative Bioinformatics*, 16(2): 20180101.  
<https://doi.org/10.1515/jib-2019-0025>
- Manninen T., Makiraatikka E., Ylipää A., Pettinen A., Leinonen K., and Linne M., 2006, Discrete stochastic simulation of cell signaling: comparison of computational tools, In: 2006 International Conference of the IEEE Engineering in Medicine and Biology Society, IEEE, pp.2013-2016.  
<https://doi.org/10.1109/IEMBS.2006.260023>
- Merzbacher C., Mac Aodha O., and Oyarzún D., 2023, Machine learning for optimization of multiscale biological circuits, *bioRxiv*, 2: 526848.  
<https://doi.org/10.1101/2023.02.02.526848>
- Meyer P., Cokelaer T., Chandran D., Kim K., Loh P., Tucker G., Lipson M., Berger B., Kreutz C., Raue A., Steiert B., Timmer J., Bilal E., Sauro H., Stolovitzky G., and Saez-Rodriguez J., 2014, Network topology and parameter estimation: from experimental design methods to gene regulatory network kinetics using a community based approach, *BMC Systems Biology*, 8(1): 13.  
<https://doi.org/10.1186/1752-0509-8-13>
- Müller M.M., Arndt K.M., and Hoffmann S.A., 2025, Genetic circuits in synthetic biology: broadening the toolbox of regulatory devices, *Frontiers in Synthetic Biology*, 3: 1548572.  
<https://doi.org/10.3389/fsybi.2025.1548572>
- Myers C., Barker N., Jones K., Kuwahara H., Madsen C., and Nguyen N., 2009, iBioSim: a tool for the analysis and design of genetic circuits, *Bioinformatics*, 25(21): 2848-2849.  
<https://doi.org/10.1093/bioinformatics/btp457>
- Pájaro M., Otero-Muras I., Vázquez C., and Alonso A., 2017, SELANSI: a toolbox for simulation of stochastic gene regulatory networks, *Bioinformatics*, 34(5): 893-895.  
<https://doi.org/10.1093/bioinformatics/btx645>
- Park J.H., Holló G., and Schaerli Y., 2024, From resonance to chaos by modulating spatiotemporal patterns through a synthetic optogenetic oscillator, *Nature Communications*, 15(1): 7284.  
<https://doi.org/10.1038/s41467-024-51626-w>
- Potvin-Trottier L., Lord N., Vinnicombe G., and Paulsson J., 2016, Synchronous long-term oscillations in a synthetic gene circuit, *Nature*, 538(7626): 514-517.  
<https://doi.org/10.1038/nature19841>
- Rombouts J., and Gelens L., 2021, Dynamic bistable switches enhance robustness and accuracy of cell cycle transitions, *PLOS Computational Biology*, 17(1): e1008231.  
<https://doi.org/10.1371/journal.pcbi.1008231>
- Sequeiros C., Pájaro M., Vázquez C., Banga J.R., and Otero-Muras I., 2023, IDESS: a toolbox for identification and automated design of stochastic gene circuits, *Bioinformatics*, 39(11): btad682.  
<https://doi.org/10.1093/bioinformatics/btad682>
- Spartalis T.R., Lizano A., Copeland C.E., Kwon Y.C., and Tang X., 2024, Current application of modeling and cell-free system for synthetic gene circuit design, *Synthetic Biology and Engineering*, 2(3): 10013.  
<https://doi.org/10.35534/sbe.2024.10013>
- Sun H., Comet J., Folschette M., and Magnin M., 2023, Condition for sustained oscillations in repressilator based on a hybrid modeling of gene regulatory networks, In: International Conference on Bioinformatics Models, Methods and Algorithms (BIOINFORMATICS 2023), SCITEPRESS-Science and Technology Publications, pp.29-40.  
<https://doi.org/10.5220/0011614300003414>
- Sun Z., Wei W., Zhang M., Shi W., Zong Y., Chen Y., Yang X., Yu B., Tang C., and Lou C., 2022, Synthetic robust perfect adaptation achieved by negative feedback coupling with linear weak positive feedback, *Nucleic Acids Research*, 50(4): 2377-2386.  
<https://doi.org/10.1093/nar/gkac066>
- Tyler J., Shiu A., and Walton J., 2019, Revisiting a synthetic intracellular regulatory network that exhibits oscillations, *Journal of Mathematical Biology*, 78(7): 2341-2368.  
<https://doi.org/10.1007/s00285-019-01346-3>
- Vazquez-Vilar M., Selma S., and Orzaez D., 2023, The design of synthetic gene circuits in plants: new components, old challenges, *Journal of Experimental Botany*, 74(13): 3791-3805.  
<https://doi.org/10.1093/jxb/erad167>
- Verdugo A., 2018, Hopf bifurcation analysis of the repressilator model, *American Journal of Computational Mathematics*, 8(2): 137-152.  
<https://doi.org/10.4236/ajcm.2018.82011>
- Wang Y., Li R., Ji C., Shi S., Cheng Y., Sun H., and Li Y., 2014, Quantitative dynamic modelling of the gene regulatory network controlling adipogenesis, *PLoS ONE*, 9(10): e110563.  
<https://doi.org/10.1371/journal.pone.0110563>
- Welsh C., Fullard N., Proctor C., Martinez-Guimera A., Isfort R., Bascom C., Tasseff R., Przyborski S., and Shanley D., 2018, PyCoTools: a Python toolbox for COPASI, *Bioinformatics*, 34(21): 3702-3710.  
<https://doi.org/10.1093/bioinformatics/bty409>

Xu Y., Zhou Z., and Liu H., 2022, Harnessing naturally occurring bistable switches for their application in synthetic biology, *SynBio*, 2(4): 363-377.

<https://doi.org/10.3390/synbio2040023>

Zhang F., Sun Y., Zhang Y., Shen W., Wang S., Ouyang Q., and Luo C., 2022, Independent control of amplitude and period in a synthetic oscillator circuit with modified repressilator, *Communications Biology*, 5(1): 23.

<https://doi.org/10.1038/s42003-021-02987-1>

Zhu Q., and Shen J., 2021, Bifurcation analysis and energy landscapes of a synthetic gene regulatory network, In: *Vibration Engineering for a Sustainable Future: Numerical and Analytical Methods to Study Dynamical Systems*, Vol. 3, Springer International Publishing, pp.325-330.

[https://doi.org/10.1007/978-3-030-46466-0\\_43](https://doi.org/10.1007/978-3-030-46466-0_43)



#### **Disclaimer/Publisher's Note**

The statements, opinions, and data contained in all publications are solely those of the individual authors and contributors and do not represent the views of the publishing house and/or its editors. The publisher and/or its editors disclaim all responsibility for any harm or damage to persons or property that may result from the application of ideas, methods, instructions, or products discussed in the content. Publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.