Educational Administration: Theory and Practice
2023, 29(4), 2360-2373
ISSN: 2148-2403
https://kuey.net/

Research Article

A Survey On Gene Regulatory Network With Optimization Techniques

Stuti Das[1*], Soujanyaa Ghosh[2], Tanusri Roy[3], Prianka Dey[4], Chetana Bhattacharyya[5], Dwaipayan Ghosh[6]

[1*,2,34.5]Narula Institute of Technology, Kolkata, India. *Email: dasstuti2001@gmail.com. Email: aiswariya12yam@gmail.com, Email: shreetanu0812@gmail.com, Email: chetanachakravarty1@gmail.com
[4]University of Calcutta, Kolkata, India. Email: priankadey2011@gmail.com, prianka.dey@nit.ac.in
[6]Kalyani Government Engineering College, Kalyani, West Bengal, India. Email: dwaipghosh@gmail.com

Citation: Stuti Das, et al. (2023), A Survey On Gene Regulatory Network With Optimization Techniques, *Educational Administration: Theory and Practice*, *29*(4), 2360-2373
Doi: 10.53555/kuey.v29i4.6701

ARTICLE INFO

ABSTRACT

Recognizing the connections between genes is essential to comprehending biological processes in all living things with cells. Gene regulatory network is the blueprint of the connections between the genes. Understanding fundamental cellular processes and the dynamic behavior of biological systems is the primary goal of research with gene regulatory networks. In conventional biology, it is difficult to reconstruct such regulatory networks from time-series gene expression data, and it has not yet been possible to perfectly reconstruct a network that is biologically accurate. The behavior of a genome can be expressed by a biological system, but computational biology sheds light on its underlying causes. To infer the genetic relationships from the biological network dynamics obtained from the experimental time series gene expression data set, researchers have used various methods from decades. Many researchers prefer the power law-based methods like s-systems, half-systems and recurrent neural networks whereas some of them consider the probabilistic approaches like Bayesian networks or Boolean. Nowadays a new approach to graph signal processing also plays an important role in terms of reconstruction of the gene regulatory network. The objective of this paper is to give a proper overview for the recreation of the gene regulatory network from the time series datasets or from the micro array data sequences. The approaches that the researchers have employed to recreate the gene regulatory network are thoroughly surveyed in this publication. Additionally, it provides future researchers with an understanding of the advantages and disadvantages of each approach, encouraging them to think creatively and beyond the box to increase the network's prediction accuracy. This paper also includes comparative studies regarding the inference accuracy of the regulatory network which is going to help the researchers to understand the most prominent and significant approach for this work.

Keywords: Gene Regulatory Network, Optimization, Bayesian Network, Boolean Network

1 Introduction

Every living thing is made up of cells. The interactions and functionality of the cells provides power to every biological process. Genes are the fundamental functional units of living cells. Each cell consists of several genes, but it is not necessary that each of them be activated. The genes which are activated are responsible for each of the activities of the living organism. The active genes are used to produce proteins through *Transcription* and *Translation*. The process of production of proteins from genes is called *Central Dogma* which shown in the Figure 1.

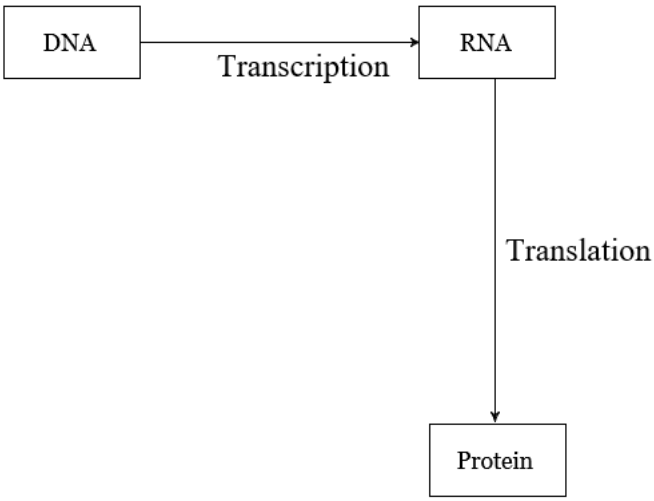

Fig. 1: Flowchart of Central Dogma

Transcription is the process of synthesis of a mRNA (messenger RNA) from a part of a DNA (Deoxyribonucleic acid) by the enzyme RNA polymerase. From this RNA copy the proteins can synthesize through the process of *Translation*. In *Translation*, mRNA is decoded into a specific ribosome from which the required amino acid chain or polypeptide is produced. A transcription factor, a protein controls the production of mRNA (messenger RNA) from DNA (Deoxyribonucleic acid) via attaching to a particular DNA sequence. For bacteria or prokaryotic cells, the *Translation* and *Transcription* process are coupled whereas for eukaryotic cells, it is decoupled. Gene interactions regulate each other to initiate every biological function; conversely, incorrect gene interactions can lead to disease in the organism. Hence, the main motivation is to develop a model to find the true interaction between the genes which helps to find the root cause of many diseases and the way a biological system works. This makes the researchers tempted to find the exact regulations between the genes which can be helpful for society to fight against some severe diseases. The expression level of gene serves as a gauge for its regulation. A typically directed graph G (V, E), where V represents a set of nodes that signify the genes and the set of edges E, can be used to depict the interactions between the genes. This graphical representation of genes is known as the Gene Regulatory Network (GRN). An example of real-life GRN is given in Figure 2. This interaction is represented by a weighted directed graph, where 0 indicates no interactions, +1 activation, and −1 inhibition. However, a GRN has two main issues: the Curse of Dimensionality and Noise. The noise present in the data sets is the impurities which can be caused by internal or external disturbances as well as environmental dangers. The "curse of dimensionality," occurs when a data set has more genes than time points. This phenomenon is referred to as effective overload in data set theory. When there are many genes present in a big network, this issue occurs. This problem does not occur in small networks where the number of genes (4–20 genes) is substantially fewer than the number of time points. A representation of time series data set is given in Table 1. To find the regulatory relationship between the genes is quite a challenge for the researchers to predict it.

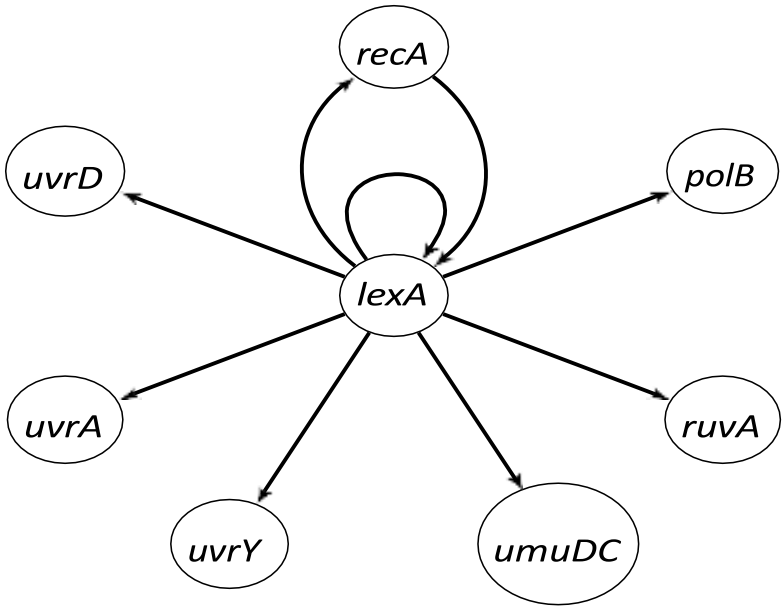

Fig. 2: The original network of the *E. coli* SOS DNA Repair network.

Table 1: Representation of time series dataset

Time point	G1	G2	G3	G4
6	0.23	0.65	0.98	0.00
12	0.89	0.45	0.67	1.00
18	1.00	0.00	0.45	0.00
24	-	-	-	-
30	-	-	-	-
36	-	-	-	-

The micro array data experiments assist researchers in obtaining time series expression [14] values at different time points with different genes to understand gene interactions in a living organism. Reverse Engineering is a popular methodology to find the interactions between the genes. Researchers used a variety of techniques for reverse engineering for many years. Some of them became successful in inferring the regulatory interactions whereas some of them failed drastically. The literature review shows that differential equations and the probabilistic technique can accurately simulate biological behavior. Consequently, a lot of scientists use the well-liked Boolean Network [12] technique to anticipate how genes will interact. Some of them prefer Bayesian networks [23] for the job. But most promising method is the power law-based approach S-systems [10]. But the computation time is very much high for S-systems, so another approach Half-system has been introduced by [15]. Recurrent Neural Network (RNN) [34] is also plays an important role in the reconstruction process of GRN. Subsequently, RNN is essential in both replicating the structure of the GRN and maintaining the biological characteristics of the living organism in a variety of techniques, including Long Short-Term Memory (LSTM) Networks, Gated RNN Networks, Time Delay Neural Networks (TDNN), and RNN with Graph Neural Networks. The rest of the paper is arranged as follows. In section 2 we have discussed the literature reviews regarding the above-mentioned approaches in detail. In section 3 an overview of optimization techniques is given and in section 4 a discussion of the methods of reconstruction process is given. The paper concluded with section 5.

2 Literature survey

Several distinct models or architectures have been used in the application of Gene Regulatory Network (GRN) analysis employing Recurrent Neural Networks (RNNs) to capture the temporal dynamics of gene expression data and infer regulatory links. We have surveyed a few well-known models here.

2.1 Boolean Network

The root of Boolean Networks lies in the automation theory. This network serves as a dynamic model for the synchronization of its nodes. It is the simplest network model which is capable of explaining the properties of biological systems and other networks. The Boolean system of *n* interconnected binary elements, or nodes, makes up a Boolean network. Every node has the capacity to receive input from other nodes. The input may differ from node to node or be the same for every node if the number of inputs is *K*. The representation of a genetic network through a Boolean network was first proposed by Kaufmann in 1969 [12]. In his article, he described a directed graph as a *NK* Boolean network, where each node has a degree of at most *K* and *N* is the number of nodes present in the graph. An example of GRN using the Boolean Network is given in the Figure 3. In [12] the author showed that the binary genes (on or off) had stability comparable with that of the living organism. In the year of 2002, Kubiket *et al.* [20] showed two ways to model the gene network using a Boolean network and also Artificial Neural Network (ANN). These two models are based on the gene expression measurement using the gene expression data set.

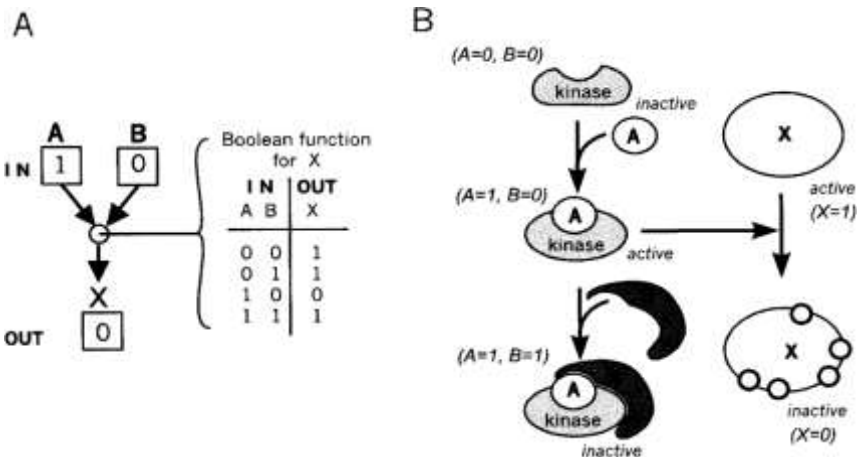

Fig. 3: An example of Boolean Network

Another work also helps in the path of gene regulation through the Boolean network, proposed by Xiao *et al.* [40]. Here, the author has used probabilistic Boolean network, a dynamic approach for analyzing of gene regulatory network.

2.2 Bayesian Network

Bayesian Network is another probabilistic visual aid which is capable of combining two main areas of mathematics, i.e. graph theory and probability which is very useful for inductive learning. This supervised learning process is complex but is also capable of solving many learning tasks. A directed acyclic graph, *G(X,E)*, which represents the Bayesian network, has two sets of edges: *E*, which represents the dependency among the gene expressions, and *X*, which represents the set of all nodes, or the gene expressions for the GRN. This network implicitly represents the Markova's assumptions. An example of Bayesian network for the inferring the GRN is shown in the Figure 4. In 2003, Perrin *et. al* [33] used a statistical machine learning approach in identification of gene regulatory network and they used Bayesian network to deal with the stochastic nature of the gene regulatory network. In 2004, Zou *et al.* [48] introduced a dynamic Bayesian network approach in gene regulation which increased the accuracy of the network and reduced the computational time. An example of the dynamic Bayesian Network is given in the Figure 5. In 2011, Campos *et al.* [5] used the Bayesian network approach in three ways for gene expression data. First, they induced Bayesian classifier from micro array data, then they have proposed a preprocessing scheme to induce the Bayesian classifier for gene expression data and lastly, they evaluated the different types of Bayesian classifiers to evaluate this kind of data. They took *nine* sets of various cancer data sets for their proposed model strategy.

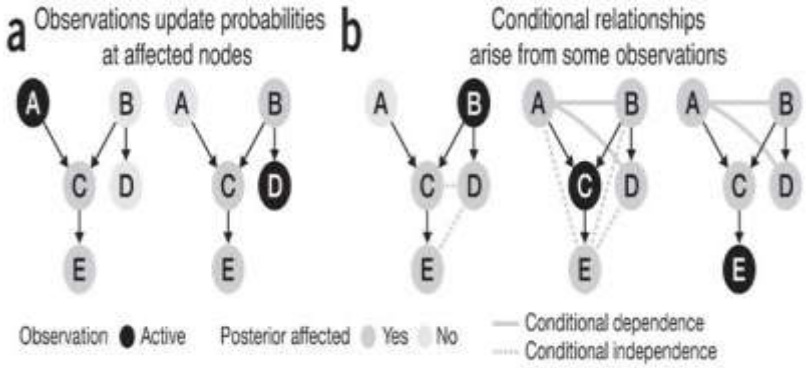

Fig. 4: An illustration of GRN using Bayesian Network

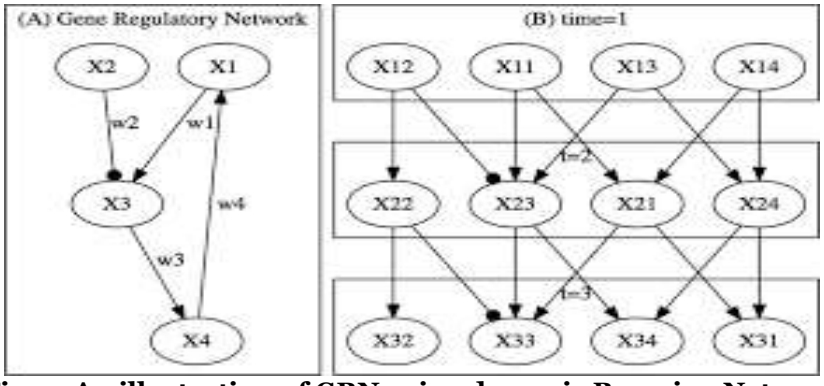

Fig. 5: An illustration of GRN using dynamic Bayesian Network

2.3 S-systems

The formalization of the complex biological system started at the very beginning of 1960 but in the year of 1988, Savageau [37] first proposed the S-system. This S-system can be said to be the canonical form of many differential equations which are nonlinear in nature. Later, the S-system was employed by numerous researchers to deduce gene regulatory networks as well as complex biological traits. In 2010, Wang *et. al* [39] proposed S-system to interfere the gene regulation. Though the S-system is applied to a very small network their unified approach makes the application of the S-system to explore comparatively large-scale networks by fast parameter estimation for finding the interaction. In August 2013, Palafox, Member, IEEE, *et. al* [32] proposed S-systems and Dissipative Particle Swarm Optimization (DPSO) based model. They used this model to analyse a small *five*-gene network and also *two in silico* actual data sets of yeast and SOS DNA Repair network of *E. coli*. Juang *et. al* [46] presented a hybrid parameter estimation algorithm-based S-systems model in 2015 to study the gene regulatory network. In the year of 2016, Mandal *et. al* [27] proposed a model which can infer GRNs. The Bat algorithm and the S-system are the foundation of this model. They used the model in

a 5-gene artificial noise-free data set and as well as in vivo noisy data set of 8-gene *E.coli* DNA SOS Repair network and a real-world 20-gene network extracted from GNW.

2.4 Half-systems

To reconstruct the gene regulatory network using S-system 2*N* (*N* +1) parameter is required to train. As the model is non-linear it is time-consuming for large networks. A value greater than zero means activation and less than zero means regression but at one time both regression and activation are not possible. In computational biology, Khan *et. al* [16] firstly derived a very promising method to reconstruct the GRN with fewer training parameters from the S-systems, *i.e.* half systems.

Half-system is the first part of the S-system model. Here only *N* (*N* +1) number of parameters is required to train, and so time requirement is less. The discrete mathematical equation of half-system is as follows:

$$\frac{dX_i}{dt} = \alpha_i \cdot \prod_{j=1}^{N} [X_j]^{g_{i,j}} \tag{1}$$

where α_i and $g_{i,j}$ are the training parameters and X_i is the expression level of the i-th gene at the time t, X_j is the expression level of the j-th gene at the time point $t + 1$. The discrete mathematical equation of the Half-systems is as follows:

$$\frac{X_i(t+\Delta t) - X_i(t)}{\Delta t} = \alpha_i \cdot \prod_{j=1}^{N} [X_j]^{g_{i,j}}(t) \tag{2}$$

The main advantage of this model is that it does not support activation and regression simultaneously. But on the other hand, as previously said half system itself is very much unstable. In order to stabilize the model during the reconstruction of the gene regulatory network, we consequently added a negative feedback term. Consequently, the equation of the half system changes to the following with the feedback term:

$$X_i(t+\Delta t) = \Delta t \cdot \alpha_i \cdot \prod_{j=1}^{N} X_j^{g_{i,j}}(t) + (1-\mu_i) \cdot X_i(t) \tag{3}$$

here μ_i is the feedback constant, a random value within [-1, 1]. As the second exponent term is removed in the half system thus the problem of S-System has overcome. For the fitness value calculation, we used MSE based fitness evaluation function [41]. The equation is as follows:

$$MSE = \frac{1}{NT} \cdot \sum_{1}^{N} \sum_{1}^{T} [\tilde{X}_i(t) - X_i(t)]^2 \tag{4}$$

where the expected value of gene expression for the subsequent time point is denoted by $\tilde{X}_i(t)$. In [7] also used this half system using a new meta-heuristic approach combined with Artificial Bee Colony algorithm with the integration of Dragon Fly algorithm. Here author also gets a satisfactory result in terms of reverse engineering.

2.5 Recurrent Neural Network

Recurrent Neural Network is another power law-based approach which provides a significant success rate in the reduction of the false interaction prediction in the GRN. The equation is provided as the following [38]:

$$\tau_i \frac{dx_i}{dt} = f \cdot \left(\sum_{j=1}^{N} \omega_{i,j} x_j(t) + \beta_i \right) - x_i(t) \tag{5}$$

where $f(.)$ is defined as,

$$f(x) = \frac{1}{1+e^{-x}} \tag{6}$$

Here, τ_i, β_i and $\omega_{i,j}$ are the training parameters of RNN and x_i is the gene expression level at the time t.

In January 2007, Xu *et al.* proposed an RNN based model where Particle Swarm Optimization (PSO) was used as training algorithm [42]. A 4-gene artificial network and an 8-gene DNA SOS Repair network of *E. coli* were

used to generate an experimental data set in real life using this model. Using the same previously mentioned data sets, the authors used Differential Evolution (DE) and PSO optimization algorithms to create a second RNN-based model in July 2007 [41].
The author said that, though RNN is quite difficult to train parameters, still it can provide the insight of the dynamic nature of the network and capable to find the interaction between the genes in a network. In 2012, Kentzoglanakis *et. al* [13] proposed a model based on RNN, Ant Colony Optimization (ACO), and PSO. The model parameters were trained using PSO, and biologically plausible candidate architectures were produced by the authors using ACO. They used a 10-gene real-world network, from which the 4-gene synthetic data set was created artificially using Gene Net Weaver (GNW), and a 10-gene synthetic data set. All that GNW is is a simple Java tool.
Mandal *et. al* [27] proposed an RNN, Bat Algorithm, and PSO based model in 2016. Here, the writers examined a synthetic dataset of a small-scale artificial network, a synthetic dataset created from a real-world yeast network, and a synthetic and real-world dataset of the DNA SOS Repair network of *E. coli*. Here, the authors improved the method for shrinking the search space by maintaining a constant number of regulators for every gene.
In 2015, Razaet *et. al* [35] proposed an RNN, Back propagation through Time and Kalman Filter based model. This suggested model was applied to real data from the DREAM challenge 50-gene network, the *E. coli* DNA SOS Repair network, the *in vivo* reverse-engineering and modeling assessment yeast network (IRMA), and the *in silico* 10-gene network. In 2017, Mandal *et. al* [28] proposed a model based on RNNs and the Bat Algorithm. First, six sets of noise-free data, each involving four genes, were analyzed using this model. Then, this methodology was used on a 4-gene noisy dataset. This dataset has 5% added Gaussian noise.

2.6 Long Short-Term Memory (LSTM) Networks

Another popular approach that helps to mitigate the vanishing gradient problem in RNNs and preserve long-term dependencies is LSTMs [25]. They are very good at capturing sequential patterns and have been successfully applied to GRN modeling. While theoretically intriguing, current techniques offer no discernible practical benefits as compared to the backdrop in feed- forward nets with constrained time frames. An LSTM is made up of three gates: an output gate, a forget gate, and an input gate. Gates in Long Short-Term Memory (LSTM) provide sigmoid activation functions, which output a value between 0 and 1, usually one of the two. ”0” means that the gates are preventing anything from passing through, and ”1” means that everything is permitted to do so. The equations for LSTM can be given by Equation 7.

$$\begin{aligned} i_t &= \sigma\left(\omega_i\left(h_{t-1}, x_t\right) + b_i\right) \\ f_t &= \sigma\left(\omega_i\left(h_{t-1}, x_t\right) + b_f\right) \\ o_t &= \sigma\left(\omega_i\left(h_{t-1}, x_t\right) + b_o\right) \end{aligned} \tag{7}$$

where h_{t-1} denotes the output of the preceding LSTM block at timestamp $(t - 1)$, and σ de- notes the sigmoid function. The weight of the i-th input gate is b, ω_i, and the biases for each gate are b. The input gate (i_t) is represented by the first equation, which indicates the new data that will be stored in the cell state (as we shall see below). The forget gate (f_t), which instructs what data to discard from the cell state, is represented by the second equation. The output gate (o_t), which is utilized to activate the last block's final output at timestamp t, is represented by the third one. To get the memory vector for the current timestamp (c_t) the candidate is calculated as

$$\begin{aligned} \tilde{c}_t &= \tanh \omega_c\left[h_{t-1}, x_t\right] + b_c \\ c_t &= f_t \cdot c_{t-1} + i_t \cdot \tilde{c}_t \\ h_t &= o_t \cdot \tanh c_t \end{aligned} \tag{8}$$

where $c\tilde{}_t$ represents a candidate for the cell state at timestamp t and c_t represents the cell state or memory at timestamp t. The above equation shows that the cell state is aware of what it needs to remember from the previous state, $f_t \cdot c_{t-1}$, and what it needs to take into account from the current timestamp, $i_t \cdot c_t$. Once the authors have filtered the cell state one last time, they apply the activation function, which determines what should be displayed as the output of the current LSTM unit at timestamp t. We can pass this h_t the output from the current LSTM block through the softmax layer to obtain the expected output (y_t) from the current block.

2.7 Gated Recurrent Unit (GRU) Networks

A kind of recurrent neural network (RNN) architecture called Gated Recurrent Networks (GRUs) [4] is intended to more effectively capture and handle long-range dependencies in sequential data. To mitigate vanishing gradient problems and better retain important infor- mation over longer sequences, they incorporate gating mechanisms to selectively control the flow of information within the network. GRUs consist of reset and

update gates that regulate the information flow, enabling them to remember or forget information at different time steps during sequence processing. There are various variations of the full gated unit, where gating is implemented by varying how the bias and the previous hidden state are combined. There is also a more basic version called the minimal gated unit. The Hadamard product is indicated by the operator ⊙ in the following Equation 9.

– Fully gated unit

Initially, for t = 0, the output vector is h_0 = 0.

$$\begin{aligned} z_t &= \sigma\left(W_z \cdot x_t + U_z \cdot h_{t-1} + b_z\right) \\ r_t &= \sigma\left(W_r \cdot x_t + U_r \cdot h_{t-1} + b_r\right) \\ \hat{h_t} &= \phi\left(W_h \cdot x_t + U_h\left(r_t \odot h_{t-1}\right) + b_h\right) \\ h_t &= (1 - z_t) \odot h_{t-1} + z_t \odot \hat{h_t} \end{aligned} \tag{9}$$

where the input vector is denoted by x_t, the output vector by h_t, the updated gate vector by z_t, and the reset gate vector by r_t. The parameter metric for learning is W, and the vector metric is b. The original hyperbolic tangent is ϕ, and the original logistic function is σ.

– Minimal gated unit

With the exception of merging the update and reset gate vectors into a forget gate, the minimal gated unit (MGU) and fully gated unit are comparable. This suggests that Equation 8, which represents the output vector equation, needs to be modified.

$$\begin{aligned} f_t &= \sigma\left(W_f \cdot x_t + U_f \cdot h_{t-1} + b_r\right) \\ \hat{h_t} &= \phi\left(W_h \cdot x_t + U_h\left(r_t \odot h_{t-1}\right) + b_h\right) \\ h_t &= (1 - f_t) \odot h_{t-1} + f_t \odot \hat{h_t} \end{aligned} \tag{10}$$

where the training metrics are W, U and b, and the input and output vectors are x_t and h_t, respectively.

– Light gated recurrent unit The light gated recurrent unit (LiGRU) [2] applies batch normalization (BN), substitutes the ReLU activation for tanh, and completely eliminates the reset gate.

$$\begin{aligned} z_t &= \sigma\left(BN\left(W_z \cdot x_t\right) + U_z \cdot h_{t-1}\right) \\ \tilde{h}_t &= ReLU\left(BN\left(W_z \cdot x_t\right) + U_z \cdot h_{t-1}\right) \\ h_t &= z_t \odot h_{t-1} + (1 - z_t) \odot \tilde{h}_t \end{aligned} \tag{11}$$

A Bayesian approach has been used to study LiGRU [2]. A variant known as the light Bayesian recurrent unit (LiBRU) was discovered through this analysis, and it performed marginally better on speech recognition tasks than the LiGRU.

2.8 Time-delayed Neural Networks (TDNNs)

TDNNs [9] process sequential data using fixed-size windows, segmenting gene expression time series into windows to predict gene interactions within these segments. There is no one person who is credited with creating Time Delayed Neural Networks (TDNNs); rather, the idea stems from the larger fields of neural networks and time-series analysis, though for the reconstruction process this methodology still now not in use. The hidden layer and output layer activations of a time delay neural network (TDNN) [31] are calculated using the following equations. Equation 14 provides the following for a single-layer TDNN:

– Hidden layer activation

$$z_h(t) = \sum_{i=0}^{N} \omega_i \cdot x(t - \tau_i) + b_h \tag{12}$$

where

– Hidden Layer Output (after activation function f_h)

$$h(t) = f_h(z_h(t)) \tag{13}$$

Where: $h(t)$ is the output of the hidden layer at time t.
– Output Layer Activation

$$z_o(t) = \sum_{j=0}^{M} v_j \cdot h(t - \gamma_j) + b_o \tag{14}$$

where $z_o(t)$ represents the activation of the output layer at time t, and v_j is the weighted matrix joining the hidden layer to the output layer. The delayed outputs of the hidden layer are indicated by $h(t - \gamma_j)$, while the time delays for the output layer's connections are indicated by γj.

– Output (after activation function f_o)

$$y_t = f_o(z_o(t)) \tag{15}$$

Where $y(t)$ is the output of the TDNN at time t.

2.9 Hybrid Models - RNNs with Graph Neural Networks (GNNs)

Integration of RNNs with Graph Neural Networks allows the incorporation of graph structures representing gene regulatory interactions. This hybrid approach captures both temporal dependencies and network topology for a more comprehensive GRN model. Modelling the complex relationships and interactions between genes is made possible by the combi- nation of Recurrent Neural Networks (RNNs) and Graph Neural Networks (GNNs) within the framework of Gene Regulatory Networks (GRNs). RNNs are well-suited for capturing temporal dependencies in sequential data, such as gene expression profiles over time. They can learn patterns and relationships within temporal sequences, aiding in understanding the dynamics of gene expression. However, constructing and training such hybrid models require careful design considerations, effective integration of RNN and GNN components, as well as optimization to handle the inherent complexity and scale of gene regulatory networks. The equation can be written as:

$$z_t = Concat(h_t, h_{GNN}) \tag{16}$$

Where h_{GNN} represents the output of the GNN on the graph structure at time t. Concat denotes the operation to concatenate the RNN hidden state and GNN output.

– Final Output:

$$\hat{y}_t = OutputLayer(z_t) \tag{17}$$

It is the predicted output at time t and the output layer, denoted as OutputLayer, is responsible for producing predictions by combining the representations.

3 Optimization Techniques

Till now there are some approaches for this reconstruction of the regulatory network using the micro-array time series data set. According to the "No Free Lunch" (NFL) theory [21]," any elevated performance over one class of problems is offset by performance over another class for any algorithm." Thus, considering this NFL theory there are several techniques to solve this meta heuristic problem but none of them are efficient to correctly detect the regulation of a GRN.

3.1 Particle Swarm Optimization

Eberhart and Kennedy developed Particle Swarm Optimization, a stochastic optimization technique based on population, in 1995. It is modeled after the way fish schools or flocks of birds behave. PSO is an evolutionary optimization algorithm which improves the candidate solutions in an iterative process. In the year of 2007, Rui *et al.* [32] has proposed a hybrid of PSO and differential evolution (DEPSO) to optimize the problem of gene regulatory network. Their study showed that DEPSO has performed better than RNN.
In the year of 2009, Zhang *et. al* proposed a noble hybrid model of particle swarm optimization and recurring neural network (PSO-RNN) [46] for gene inference method. Chien-Pang Lee *et. al* in the year 2011 tried to reconstruct a gene regulatory network using the microarray data set using GA/PSO [22]. PSO is effective for local optimization, but it is poor for exploration. Thus, in the year 2017 Liu, *et. al* proposed [24] Multi leader

PSO (MPSO) which is capable of enhancing the exploration search of PSO by changing the memory structure of the canonical PSO. Here the particles choose their leaders using the game theory instead of random selection. Many particles in a specially bounded search space are said to be unaware of the food's location, according to the theoretical idea of PSO. There will be one particle among the others whose location is closest to the food source. The particle is said to be in the best position globally among all of the particles at that particular location. That particle will lead the other particles. The best position a particle has found thus far during a local search is referred to as each particle's personal best position. Up until the stopping condition is met, this process keeps going.

For solving a problem using PSO we first initialize the position vector and the velocity vector of each particle randomly consisting of same number of parameters and same range. At each instant of set all the randomly initialized parameters produce candidate solution. Each parameter should be within a specified range and that will be the same for all particles. The range of every parameter may vary. There will be an objective function which will calculate the fitness value for each particle using the corresponding parameters. We will calculate the fitness value using the objective function and then find the local best which is the position of the particle up to which it has minimum fitness so far. This position is stored as the personal best *(pbest)* position for each particle. Up to this PSO will do the local search. After this, we will step to the global version of PSO. Here we will find the global best position of the full search space i.e the particle's location which has reached most nearly to the position of the food. The minimum of the pbest, or *gbest*, is the symbol for this global best. The particle's position and velocity are then updated in the subsequent generation using the following equations:

$$v_i(t+1) = v_i(t) \cdot w_i + c_1 \cdot r_1 \cdot (pbest_i - x_i(t)) + c_2 \cdot r_2 \cdot (gbest - x_i(t)) \quad (18)$$

$$x_i(t+1) = v_i(t+1) + x_i(t) \quad (19)$$

where w_i is the inertia weight, c_1 and c_2 are called the learning factor, initialized as $c_1 = c_2 = 2$, r_1 and r_2 are randomly generated two numbers within [0,1], and $v_i(t)$ and $x_i(t)$ are the initial velocity vector and initial position vector of the *ith* particle at the *t* instance.

3.2 Bat Algorithm Inspired PSO (BAPSO)

Bat inspired PSO (BAPSO) is another novel approach for the optimization problem which is the hybridization of the BAT algorithm [43] and PSO. Yang *et al.* used the echolocation behavior of micro bat for searching operation. Microbats employ a unique form of sonar known as echolocation to find food and avoid obstacles. The microbats are able to determine the objects' direction and distance by receiving the waves. The bats in this BAT algorithm can be assumed to fly at random, and some bats are sent out for local exploration and search after each iteration.

The creator of the BAPSO algorithm, as suggested by Khan *et. al* [17], took into consideration that each microbat is initially at rest. The two PSO parameters, the inertia weight w and r, are evenly distributed and chosen at random within the range [0,1]. Because of this, the BAT algorithm can both explore and exploit. The lowest error criterion or the maximum quantity of repetitions serves as the halting condition. The mathematical representation of the BAPSO becomes as follow:

$$v_i(t+1) = r \cdot v_i(t) + c_1 \cdot r_1 \cdot (pbest_i - x_i(t)) + c_2 \cdot r_2 \cdot (gbest - x_i(t)) \quad (20)$$

$$x_i(t+1) = v_i(t+1) + x_i(t) \quad (21)$$

3.3 Grey Wolf Optimization Inspired Particle Swarm Optimization (GWPSO)

Considering the social leadership hierarchy and hunting behavior of the grey wolf, Mirjalili *et. al* [30] presented Grey Wolf Optimisation, another algorithm inspired by nature. As the top predators, grey wolves typically dwell in packs of five to twelve. There are four different kinds of wolves in each group. The decision of what to hunt is primarily made by the alphas, the male and female pair who lead the group. The second tier of the hierarchy is made up of beta wolves, the subordinate wolves that support the alphas in decision-making and other pack activities. It is possible for these beta wolves to be male or female. After alpha, the beta wolves represent the next best option (solution) in the group. The next category of pack candidates are those classified as delta under beta. The pack's final group, known as Omega, serves as the group's scapegoat.

Khan *et al.* [17] has suggested another hybrid meta heuristic, the GWPSO, and integrated this hierarchical GW approach into the conventional PSO. The particles in this suggested method remember the second and third

best solutions in addition to the best one during each iteration. Next, the following is an improvised global best solution for Equation 18:

$$gbest_{mean} = \frac{g_1 + g_2 + g_3}{3} \tag{22}$$

where g_1 is the first best solution, g_2 is the second best solution, g_3 is the third best solution respectively. Thus, the equation 18 becomes,

$$\begin{aligned} v_i(t+1) = r \cdot v_i(t) + c_1 \cdot r_1 \cdot (pbest_i - x_i(t)) \\ + c_2 \cdot r_2 \cdot (gbest_{mean} - x_i(t)) \end{aligned} \tag{23}$$

3.4 Artificial Bee Colony Optimization (ABC)

Artificial Bee Colony optimization is another meta-heuristic nature-inspired algorithm which is based on the forging behaviour of the honeybee. It was first proposed by Karaboga *et al.* [11] for the unconstrained optimization problem. It has been proved that it is superior to any other existing heuristic algorithm for unconstrained problems. In the proposed algorithm there are three types of bees in a colony: Employed bees, Onlookers, and scouts. Half of the colony consists of employed bees and the remaining half includes the onlookers. For every employed bee, there is only one food source, i.e. the number of food sources is equal to the number of employed bees in the hive. The scouts are employed bees whose food supply has run out. The location of the food source, or the position of the working bee, is one potential solution to the optimization problem, and the amount of nectar is the best solution, or the optimization problem's fitness value.
The following equation defines the probability value by which the onlooker bee chooses the food position:

$$p_i = \frac{fit_i}{\sum_{i=1}^{P} fit_n} \tag{24}$$

where P is the number of the food source, pi is the position of the food source, fit_i is the fitness value of the *ith* position. In order to produce the candidate solution or the food position from the old one ABC used the following equation:

$$x_{i,j}^{new} = x_{i,j}^{old} + \phi_{i,j}(x_{i,j} - x_{i,k}) \tag{25}$$

where j is the randomly selected index and $k \in 1, 2, 3, \ldots, P$ and $\phi_{i,j}$ is the randomly chosen number in between [1,-1]. It manages the process of producing potential food sources. The found food source is presumed to be abandoned if it is determined that it cannot be improved, and the scout bee reinitializes it using the equation below:

$$x_i^j = x_{min}^j + rand(0,1)(x_{max}^j - x_{min}^j) \tag{26}$$

here x_i is the position of the discarded food source and $j \in 1, 2, \cdots P$. Babayigit *et. Al* [1] has modified this ABC algorithm by providing a probability function for improving the exploration mechanism of the onlooker bee. The author proposed that the onlooker bee will choose the food source according to the likelihood function which is as follows:

$$P_i = exp\left(\frac{-1}{\rho \cdot f_i}\right) \tag{27}$$

where the fitness value normalized within [0,1] is denoted by f_i, and ρ is the governing parameter of the ABC algorithm. It signifies that the higher the fitness there is the more probability for the selection of food source by the onlooker bees. The position of onlooker bee is also improved by the authors for increasing diversity. Thus, the equation for onlooker bee becomes:

$$x_{i,j} = x_{best,j} + \phi_{i,j}(x_{best,j} - x_{i,j}) \tag{28}$$

In the equation 28, x_{best} is the best solution in the current population, and $x_{i,j}$ is the present position exploited by the onlooker bee. This is known as the **ABCbest** algorithm. In the year 2015, Forghany *et. al* [8] used ABC

and modified ABC in GRN construction with S- system as objective function. They set the population size as 50 and applied their proposed work to three networks. The maximum iteration was 2,000,000 for the first two networks and 5,000,000 for the third network. They proposed that modified ABC gave a superior result in the context of the GRN rather than any other evolutionary algorithm.

4 Discussion

As per the previous literature survey Gene regulatory networks (GRNs) play a crucial role in understanding how genes interact and regulate various biological processes within organ- isms. Analyzing these networks can provide insights into cellular behavior, development, and disease mechanisms. Strategies and optimization techniques in GRN analysis are diverse, encompassing computational, experimental, and integrated approaches. Various approaches are present for this reconstruction process which is discussed below Table 2.

Table 2: Comparative studies of various approaches

Method	Observations
Computational models like Boolean network [40], Bayesian network [33]	•Help to find out the interactions between the genes. •It fails to define the dynamic nature of biology. •Sometimes these models fail to represent the temporal dynamics which results as the loss of information.
Reverse engineering algorithms [36], [29]	•Infers GRN structures from experimental data. •Data dependency is the main issue of this method •Inference of causality is another consequence of this approach
Single-cell RNA sequencing [26]	•Enable the reconstruction of GRNs at single-cell resolution. •Sparse and noisy data are the limitations of this approach. •Cellular heterogeneity complicates the GRN reconstruction process.
Network Fusion and Multi view learning [45], [44]	•A comprehensive view of network. •Computational cost is more. •Increased Sensitivity to Errors. •Overfitting
Constraint-based modeling techniques [3]	•Used for the metabolic analysis of the cellular metabolism. •Integration with metabolic networks. •Hypothesis generation. •Simplified Representation of GRN. •Limited Incorporation of Regulatory Information.
Evolutionary algorithms [19]	•Global optimization. •No derivative required. •Computational cost is very high. •Parameter Tuning is challenging.
Machine Learning and Deep Learning [47]	•Can Capture Nonlinear Relationships. •Feature Selection and Dimensionality Reduction. •Data Quality and Preprocessing is required. •Model Selection and Hy- per parameter Tuning.

Reviewing the comparative studies between various approaches now we need to compare the accuracy of the optimization techniques to find the most adoptable techniques for the reconstruction of gene regulatory network which is stated in the Table 3.

Table 3: Comparative studies among the optimization techniques

Name of the Optimization Algorithms	Accuracy
PSO [18]	79%
BAPSO [18]	78%
GWPSO [17]	88%
ABC [17]	91%

Computational models [6] simulate the interactions between genes and their products, such as transcription factors and regulatory elements. Techniques like Boolean networks, ordinary differential equations (ODEs), Bayesian networks, and agent-based models are commonly used to represent GRNs computationally. Every method has advantages and disadvantages when it comes to capturing various facts of the dynamics of gene regulation. Reverse engineering algorithms [36, 29] infers GRN structures from experimental data, such as

gene expression pro- files or chromatin immunoprecipitation sequencing (ChIP-seq) data. These algorithms include methods like relevance networks, Bayesian network inference, and mutual information-based approaches. They aim to identify regulatory relationships between genes and predict the underlying network topology. Multiple optimization techniques also being useful for this reconstruction of GRN which already discussed in the section 3. Single-cell RNA sequencing [26] techniques (scRNA-seq) enable the reconstruction of GRNs at single-cell resolution. However, analyzing single-cell data poses unique challenges due to noise, sparsity, and heterogeneity, requiring specialized algorithms for network inference. Network Fusion and Multi-view learning [45], [44] approach integrating multiple types of omics data, such as gene expression, DNA methylation, and protein-protein interaction data, provides a more comprehensive view of GRNs. The accuracy and robustness of inferred GRNs can be increased by combining heterogeneous data sources using integration techniques like network fusion and multi-view learning algorithms. Constraint-based modelling techniques [3], such as flux balance analysis (FBA) and metabolic control analysis (MCA), analyze GRNs within the context of cellular metabolism. These methods consider constraints on biochemical reactions and cellular re- sources to predict gene regulatory mechanisms that optimize metabolic objectives, such as growth rate or energy production. Evolutionary algorithms optimize GRN models by iteratively refining network structures to fit experimental data. Evolutionary algorithms optimize GRN models by iteratively refining network structures to fit experimental data. Techniques like genetic algorithms and simulated annealing search through the space of possible GRN configurations to identify models that best explain observed biological behaviors. In GRN models, evolutionary optimization is capable of handling big search areas and intricate fitness landscapes. The best nature-inspired algorithms for optimizing GRN parameters are PSO, BAPSO [43], and ABC [11]. The ability of machine learning and deep learning [47] techniques to identify intricate patterns in high-dimensional data has made them popular in GRN analysis. Algorithms such as neural networks, random forests, and support vector machines can be trained about regulatory links using large-scale genomic datasets, which enables the algorithms to predict gene interactions and regulatory mechanisms with accuracy. For GRNs to have biological significance, experimental validation of their computational predictions is required.

5 Conclusion

Reconstructing gene regulatory networks (GRNs) through surveying the field yields important information on problems, upcoming paths and cutting-edge techniques. The survey shows that though reconstructing GRNs is still a challenging and intricate task, it holds great promise for advancing our knowledge of how genes are controlled and how cells operate. For the purpose of deriving GRNs from high-throughput omics data, including gene expression, DNA-protein interactions, and epigenetic changes, a variety of computational and experimental methods have been devised and implemented. To sum up, the survey emphasizes how crucial it is for computational biologists, bioinformaticians, and experimental biologists to work together transdisciplinary in order to advance the field of GRN reconstruction. Researchers can continue to unravel the intricate regulatory networks governing cellular processes and diseases by addressing the issues raised in this survey and utilizing cutting-edge technologies and methodologies. This will ultimately lead to advancements in precision medicine and therapeutic interventions.

6 Compliance with Ethical Standards

Conflict of Interest: The authors have no conflicts of interest to declare that are relevant to the content of this article.

Financial Support: The authors did not receive financial support from any organization for the submitted work. This survey paper does not involve any animals or human participants.

Author's Contribution:
Every author has contributed equally in this research work.

References

1. Babayigit, B., Ozdemir, R.: A modified artificial bee colony algorithm for numerical function optimization. In: 2012 IEEE Symposium on Computers and Communications (ISCC). pp. 000245–000249. IEEE (2012).
2. Bittar, A., Garner, P.N.: A bayesian interpretation of the light gated recurrent unit. In: ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). pp. 2965–2969. IEEE (2021).
3. Bordbar, A., Monk, J.M., King, Z.A., Palsson, B.O.: Constraint-based models predict metabolic and associated cellular functions. Nature Reviews Genetics 15(2), 107–120 (2014).
4. Chung, J., Gulcehre, C., Cho, K., Bengio, Y.: Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv preprint arXiv:1412.3555 (2014).

5. De Campos, L.M., Cano, A., Castellano, J.G., Moral, S.: Bayesian networks classifiers for gene-expression data. In: 2011 11th International Conference on Intelligent Systems Design and Applications. pp. 1200–1206. IEEE (2011).
6. Delgado, F.M., Gómez-Vela, F.: Computational methods for gene regulatory networks reconstruction and analysis: a review. Artificial intelligence in medicine 95, 133–145 (2019).
7. Dey, P., Khan, A., Saha, G., Pal, R.K.: Computational reconstruction of gene regulatory networks using half-systems incorporating false positive reduction techniques. In: Mathematical Modeling, Computational Intelligence Techniques and Renewable Energy: Proceedings of the Second International Conference, MMCITRE 2021. pp. 157–167. Springer (2022).
8. Forghany, Z., Davarynejad, M., Snaar-Jagalska, B.E.: Gene regulatory network model identification using artificial bee colony and swarm intelligence. In: 2012 IEEE Congress on Evolutionary Computation. pp. 1–6. IEEE (2012).
9. Hampshire, Waibel: A novel objective function for improved phoneme recognition using time delay neural networks. In: International 1989 Joint Conference on Neural Networks. pp. 235–241. IEEE (1989).
10. Irvine, D.H.: S-system analysis of organizationally complex systems: Network regulation of the immune response. University of Michigan (1988).
11. Karaboga, D., Basturk, B.: Artificial bee colony (abc) optimization algorithm for solving constrained optimization problems. In: International fuzzy systems association world congress. pp. 789–798. Springer (2007).
12. Kauffman, S.A.: Metabolic stability and epigenesis in randomly constructed genetic nets. Journal of theoretical biology 22(3), 437–467 (1969).
13. Kentzoglanakis, K., Poole, M.: A swarm intelligence framework for reconstructing gene networks: searching for biologically plausible architectures. IEEE/ACM Transactions on Computational Biology and Bioinformatics 9(2), 358–371 (2012).
14. Kerr, M.K., Martin, M., Churchill, G.A.: Analysis of variance for gene expression microarray data. Journal of computational biology 7(6), 819–837 (2000).
15. Khan, A., Mandal, S., Pal, R.K., Saha, G., et al.: Construction of gene regulatory networks using recurrent neural networks and swarm intelligence. Scientifica 2016 (2016).
16. 16. Khan, A., Saha, G., Pal, R.K.: A swarm intelligence based scheme for reduction of false positives in inferred gene regulatory networks. In: 2016 IEEE Congress on Evolutionary Computation (CEC). pp. 40–47. IEEE (2016).
17. 17. Khan, A., Saha, G., Pal, R.K.: An approach for reduction of false predictions in reverse engineering of gene regulatory networks. Journal of theoretical biology 445, 9–30 (2018).
18. 18. Khan, A., Saha, G., Pal, R.K.: Modified half-system based method for reverse engineering of gene regulatory networks. IEEE/ACM Transactions on Computational Biology and Bioinformatics 17(4), 1303–1316 (2019).
19. 19. Knabe, J.F., Wegner, K., Nehaniv, C.L., Schilstra, M.J.: Genetic algorithms and their application to in silico evolution of genetic regulatory networks. Computational Biology pp. 297–321 (2010).
20. 20. Kubik, T., Bogunia-Kubik, K., Sugisaka, M.: Gene expression modelling with the use of Boolean network and artificial neural network. In: Proceedings of the 2nd IEEE Conference on Nanotechnology. pp. 157–160. IEEE (2002).
21. Lattimore, T., Hutter, M.: No free lunch versus occam's razor in supervised learning. In: Algorithmic Probability and Friends. Bayesian Prediction and Artificial Intelligence: Papers from the Ray Solomonoff 85th Memorial Conference, Melbourne, VIC, Australia, November 30–December 2, 2011. pp. 223–235. Springer (2013).
22. Lee, C.P., Leu, Y., Yang, W.N.: Constructing gene regulatory networks from microarray data using ga/pso with dtw. Applied Soft Computing 12(3), 1115–1124 (2012).
23. Liu, F., Zhang, S.W., Guo, W.F., Wei, Z.G., Chen, L.: Inference of gene regulatory network based on local bayesian networks. PLoS computational biology 12(8), e1005024 (2016).
24. Liu, P., Liu, J.: Multi-leader pso (mlpso): A new pso variant for solving global optimization problems. Applied Soft Computing 61, 256–263 (2017).
25. Liu, S., Li, T., Ding, H., Tang, B., Wang, X., Chen, Q., Yan, J., Zhou, Y.: A hybrid method of recurrent neural network and graph neural network for next-period prescription prediction. International Journal of Machine Learning and Cybernetics 11, 2849–2856 (2020).
26. Liu, Z., Lou, H., Xie, K., Wang, H., Chen, N., Aparicio, O.M., Zhang, M.Q., Jiang, R., Chen, T.: Reconstructing cell cycle pseudo time-series via single-cell transcriptome data. Nature communications 8(1), 22 (2017).
27. Mandal, S., Khan, A., Saha, G., Pal, R.K.: Reverse engineering of gene regulatory networks based on s-systems and bat algorithm. Journal of Bioinformatics and Computational Biology 14(03), 1650010 (2016).
28. Mandal, S., Saha, G., Pal, R.K.: Recurrent neural network based modeling of gene regulatory network using bat algorithm. arXiv preprint arXiv:1509.03221 (2015).
29. Marku, M., Pancaldi, V.: From time-series transcriptomics to gene regulatory networks: A review on inference methods. PLOS Computational Biology 19(8), e1011254 (2023).

30. Mirjalili, S., Mirjalili, S.M., Lewis, A.: Grey wolf optimizer. Advances in engineering software 69, 46–61 (2014).
31. Molho, D., Ding, J., Tang, W., Li, Z., Wen, H., Wang, Y., Venegas, J., Jin, W., Liu, R., Su, R., et al.: Deep learning in single-cell analysis. ACM Transactions on Intelligent Systems and Technology (2022).
32. Palafox, L., Noman, N., Iba, H.: Reverse engineering of gene regulatory networks using dissipative particle swarm optimization. IEEE Transactions on Evolutionary Computation 17(4), 577–587 (2013).
33. Perrin, B.E., Ralaivola, L., Mazurie, A., Bottani, S., Mallet, J., d'Alche Buc, F.: Gene networks inference using dynamic bayesian networks. Bioinformatics 19(suppl_2), ii138– ii148 (2003).
34. Posner, E.: Ieee (institute of electrical and electronic engineers) conference on neural information processing systems-natural and synthetic held in denver, colorado on november 8-12, 1987. abstracts of papers. Tech. rep. (1989).
35. Raza, K., Alam, M.: Recurrent neural network based hybrid model for reconstructing gene regulatory network. Computational Biology and Chemistry 64, 322–334 (2016).
36. Ristevski, B.: A survey of models for inference of gene regulatory networks. Nonlinear Analysis: Modelling and Control 18(4), 444–465 (2013).
37. Savageau, M.A.: Introduction to s-systems and the underlying power-law formalism. Mathematical and Computer Modelling 11, 546–551 (1988).
38. Vohradsky, J.: Neural model of the genetic network. Journal of Biological Chemistry 276(39), 36168–36173 (2001).
39. Wang, H., Qian, L., Dougherty, E.: Inference of gene regulatory networks using s-system: a unified approach. IET Systems Biology 4(2), 145–156 (2010).
40. Xiao, Y.: A tutorial on analysis and simulation of boolean gene regulatory network models. Current Genomics 10(7), 511–525 (2009).
41. Xu, R., Venayagamoorthy, G.K., Wunsch II, D.C.: Modeling of gene regulatory networks with hybrid differential evolution and particle swarm optimization. Neural Networks 20(8), 917–927 (2007).
42. Xu, R., Wunsch II, D., Frank, R.: Inference of genetic regulatory networks with recurrent neural network models using particle swarm optimization. IEEE/ACM Transactions on Computational Biology and Bioinformatics 4(4), 681–692 (2007).
43. Yang, X.S.: A new metaheuristic bat-inspired algorithm. In: Nature inspired cooperative strategies for optimization (NICSO 2010), pp. 65–74. Springer (2010).
44. Yuan, L., Guo, L.H., Yuan, C.A., Zhang, Y., Han, K., Nandi, A.K., Honig, B., Huang, D.S.: Integration of multi-omics data for gene regulatory network inference and application to breast cancer. IEEE/ACM transactions on computational biology and bioinformatics 16(3), 782–791 (2018).
45. Zafari, N., Bathaei, P., Velayati, M., Khojasteh-Leylakoohi, F., Khazaei, M., Fiuji, H., Nassiri, M., Hassanian, S.M., Ferns, G.A., Nazari, E., et al.: Integrated analysis of multiomics data for the discovery of biomarkers and therapeutic targets for colorectal cancer. Computers in Biology and Medicine p. 106639 (2023).
46. Zhang, Y., Xuan, J., de los Reyes, B.G., Clarke, R., Ressom, H.W.: Reverse engineering module networks by pso-rnn hybrid modeling. BMC Genomics 10(1), S15 (2009).
47. Zhao, M., He, W., Tang, J., Zou, Q., Guo, F.: A hybrid deep learning framework for gene regulatory network inference from single-cell transcriptomic data. Briefings in bioinformatics 23(2), bbab568 (2022).
48. Zou, M., Conzen, S.D.: A new dynamic bayesian network (dbn) approach for identifying gene regulatory networks from time course microarray data. Bioinformatics 21(1), 71–79 (2004).