Gene expression
for simulation of biological tissue

Sadyk Sayfullin, Manuel Mazzara, Ruslan Mustafin and Victor Rivera
Innopolis University
Innopolis, Russia

Abstract—BioDynaMo is a biological processes simulator developed by an international community of researchers and software engineers working closely with neuroscientists. The authors have taken part in the development of the physical engine, and they are currently working on gene expression, i.e. the process by which the heritable information in a gene - the sequence of DNA base pairs - is made into a functional gene product, such as protein or RNA. Typically, gene regulatory models employ either statistical or analytical approaches, being the former already well understood and broadly used. In this paper, we utilize analytical approaches representing the regulatory networks by means of differential equations, such as Euler and Runge-Kutta methods. The two solutions are implemented in the BioDynaMo project and are compared for accuracy and performance.

I. INTRODUCTION

The mechanisms behind cells functioning have been progressively unveiled by the scientific community discovering the physical, biological and chemical principles on which they are operating. The problem that scientists are facing today is how to generate working hypotheses starting from the data collected over the years. There is still a limited number of software tools that can be effectively used for this kind of applications. Simulators have been proven to be a viable approach to interpret the results collected by experiments. BioDynaMo [1] is one of these simulators developed by an international community of researchers and software engineers. This paper specifically focuses on the implementation of gene expression in the simulator developed by the researchers of this project on top of the existing whole system.

A. BioDynaMo

The BioDynaMo project [1] is a simulator of biological processes designed to support the work of researchers in the biological field. Nowadays, a number of world-wide labs create their own software for running simulations. However, these solutions can often be utilized by specific labs and on specific tasks, without scaling and requiring significant resources.

BioDynaMo has been designed to address this problem in order to offer provide researchers with software support. The project is developing a new general platform for computer simulations of biological tissue dynamics, with a brain development as a primary target. The platform should be executable on hybrid cloud computing systems, allowing for the efficient use of state-of-the-art computing technology [2].

A set of different cellular behaviors is covered in this simulation such as cell division, cell growth, gene expression, chemical gradient and mechanical forces. Spatial locality of interaction is the main feature that makes this project run efficiently on highly parallelized cloud systems. This principle states that simulation objects reference to each other in the case when they are close [3]. It allows simulation space to be split up into fragments that do not require a large amount of communication between each other. Scale of simulation close to Cx3D [4] upon which BioDynaMo was initially created [5]. However, with additional features, the project reaches simulation level of [6].

The aim of the BioDynaMo project is to push the limits of this simulation type with both the highly efficient code and the extensive parallelization on relatively cheap cloud-based hardware [5].

B. Gene expression

Gene expression is the process by which gene product, such as RNA and proteins, is produced. Sequences of DNA store heritable information that is used to produce gene product. Figure [1] depicts the main idea of this process: at first DNA is transcribed into RNA, which is subsequently translated into proteins [7]. Proteins make many of the structures and all the enzymes in a cell or organism. Several steps in the gene expression process might be modulated. This includes both the transcription and translation stages. Several of biological processes controlled by gene expression and slight changes of specific proteins’ concentration or links can underlie human diseases, population differences and the evolution of morphological novelties [8]. In addition, type of cancer can be classified by tracking of gene expression [9].
II. GENE EXPRESSION IN SIMULATIONS

There are two main approaches in gene regulatory models: statistical and analytical. Statistical approaches have a high level of accuracy due to the amount of studies made on gene array data and exhaustively reviewed in the literature [10], [11]. In contrast, the value of analytical approaches applied for modeling gene regulation is generally less appreciated, in particular in the field of DNA-sequence-based modeling. This paper shall examine the analytical approaches in order to investigate their potentiality and limitations. In general, analytical approaches concentrate on simulation of expression of a few genes and use a mixture of different mathematical models in the implementation. In order to develop this approach, deep understanding of how parts of the system work individually is required, and hypotheses on how the composition of these parts behaves together. Analytical approaches can simulate terms relating to the binding of transcription factors and RNA polymerase to the DNA, cooperative and inhibitory interactions between transcription factors, mRNA and protein degradation, and mRNA translation rate [12].

A. Thermodynamical model

In thermodynamical models, gene production regulated by bound activators and bound repressors. For a variety of mixture of these binding factors on regulatory regions, thermodynamical model predicts the concentration of gene product. The main assumption in this model is that the level of production is proportional to bound activators and inversely proportional to bound repressors [12].

B. Boolean model

Each gene product in this method has the property that represents its state "ON" and "OFF". To define the relationships between entities, logic functions "and", "or" and "not" are used [13]. For example, if an expression of a gene is controlled by two products, the gene produces mRNA only if both products are ON in case of the function AND, the gene is transcribed if one of two products is on in case of OR function, and NOT means that gene is not transcribed if both are ON.

C. Differential equation model

Differential equations can be used to model gene regulation network. In this model, interactions between entities produced by gene expression and concentration of them are defined by a set of differential equations. These equations depend on the variety of parameters, such as time, space, the concentration of other products such as mRNA and proteins and production and degradation rate of the particular entity [14].

III. IMPLEMENTATION

In BioDynaMo, a differential equation model has been implemented using Ordinary Differential Equations (ODE) which depend on a single variable i.e. time. The given task is

\[ f(p(t), t) = \frac{dp(t)}{dt} \]  

(1)

where \( f(p(t), t) \) the given function of changing of protein concentration over time, \( p(t) \) is the concentration of protein at time \( t \). The idea is to track \( p \) through time. Two methods have been implemented in order to solve this task:

- Euler method
- Runge-Kutta method

These methods solve the Cauchy boundary value problem [15] for differential equation \( f(p(t), t) \) and given initial values \( t_0, p(t_0) = p_0 \). Both methods calculate \( p_n = p(t_n) \) step by step at \( t_0, t_1, t_2 \ldots \ldots \).

A. Runge-Kutta method

In general, Runge-Kutta method for order \( s \) is

\[ p_n = p_{n-1} + h \sum_{i=1}^{s} b_i F_i, \quad n = 1, 2, \ldots \]  

(2)

where \( F_i \) is defined as

\[ F_i = f(x_{n-1} + h c_i, p_{n-1} + h \sum_{j=1}^{s} a_{ij} F_j) \]  

(3)

where \( a_{ij}, c_i, b_i \) defined by tableau [16] see table I.

| \( a_{ij} \) | \( c_i \) | \( b_i \) |
|---|---|---|
| 0 | 0 | 0 |
| \( a_{21} \) | 0 | 0 |
| \( a_{22} \) | 0 | 0 |
| \vdots | \vdots | \vdots |
| \( a_{s1} \) | \( a_{s2} \) | \( a_{s3} \) |

Table I: Tableau for Runge-Kutta method of order \( s \)

and \( h = (t_n - t_{n-1}) \) is time step of one iteration. In this work, we use Runge-Kutta of order 4. Thus, it has an error rate \( O(h^4) \) [17]. For Runge-Kutta of order 4 tableau

---

2 https://en.wikipedia.org/wiki/Gene_expression#/media/File:Genetic_code.svg see table II.
Thus, Runge-Kutta method of order 4 defined by equations [3] and [5]

\[
p_n = p_{n-1} + \frac{h}{6}(F_1 + 2F_2 + 2F_3 + F_4) \quad (4)
\]

\[
F_1 = f(p_{n-1}, t_{n-1})
\]

\[
F_2 = f(p_{n-1} + \frac{h}{2}F_1, t_{n-1} + \frac{h}{2})
\]

\[
F_3 = f(p_{n-1} + \frac{h}{2}F_2, t_{n-1} + \frac{h}{2})
\]

\[
F_4 = f(p_{n-1} + hF_4, t_{n-1} + h)
\]

\[
(5)
\]

| Algorithm 1 Runge-Kutta method |
|--------------------------------|
| 1: procedure COMPUTE(current_time, time_step, protein) |
| 2: F1 = Equation(c_f, p) |
| 3: F2 = Equation(c_f + t_s/2, p + t_s * F1/2) |
| 4: F3 = Equation(c_f + t_s/2, p + t_s * F2/2) |
| 5: F4 = Equation(c_f + t_s, p + t_s * F3) |
| 6: p = p + t_s * (F1 + 2 * F2 + 2 * F3 + F4)/6 |
| 7: c_f = c_f + t_s |

Where c_f is current time t_n, t_s time step h, p is protein concentration p. Equation is the given differential equation f(p(t), t) and F1 - F4 parameters from equation 5.

Runge-Kutta method has high accuracy but the performance of this method is lower in comparison with Euler method due to the amount of operations it carries out: 4 calls to Equation against 1 call in Euler method.

B. Euler method

Euler method is the Runge-Kutta method of order s = 1. For Euler method tableau see table III

| Table III: Tableau for Euler method(Runge-Kutta method of order 1) |
|------------------------|
| 0 | 0 | 0 | 0 | 0 |
| 1/2 | 0 | 0 | 0 |
| 1/2 | 0 | 1/2 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 |
| 1/6 | 1/3 | 1/3 | 1/6 |

Euler method [16] defined as [6]

\[
p_n = p_{n-1} + h \cdot f(p_{n-1}, t_{n-1}), \quad n = 1, 2, \ldots \quad (6)
\]

Euler method at each iteration makes only one operation and once calls for f(p(t), t). This allows the calculations to be faster than the previous method. However, the error rate of this method is bigger \( O(h) \) [17].

Algorithm 2 Euler method

1: procedure COMPUTE(current_time, time_step, protein) |
2: \( p = p + t_s \cdot f(p(t), t) \) |
3: \( c_f = c_f + t_s \) |

C. Comparison

Both methods are used to solve the same task. However, they differ in accuracy and performance. The error rate of Runge-Kutta method \( O(h^4) \) and \( O(h) \) for Euler method. At the same time Runge-Kutta has lower performance due to the amount of operations on each iteration.

Both Iglorithms were implemented in C++ and experiments were run on Intel® CoreTM i5-6200U CPU @ 2.30GHz x 4.

IV. RESULTS

Figures 2b, 3 and table IV present the results of the solution to three Cauchy problems for equations 2, 9 and 11. Solutions are presented using the Euler and Runge-Kutta numerical methods with different parameter h. Furthermore, they present the analytical solution for the same equations in order to evaluate the accuracy of the methods. Each cell in table IV displays the difference between analytical and numerical solutions.

Figure 2b depicts plots for the Cauchy problem 7

\[
\begin{aligned}
& f(p(t), t) = \frac{1}{t} \\
& p_0 = p(0) = 0
\end{aligned} \quad (7)
\]

Analytical solution for this task is:

\[
p(t) = \sqrt{2t} \quad (8)
\]

Figure 2a shows the solution by Euler method. It is observed that for \( h = 0.1 \), \( h = 0.01 \) and the analytical solution the lines overlap. Plot with \( h = 1 \) does not give correct values from the beginning.

Figure 2c shows the solution by Runge-Kutta method. It is observed that lines for all h and the analytical solution overlap.
Table IV: Accuracy of Euler and Runge-Kutta methods

| Equation 7 | Euler | Runge-Kutta |
|------------|-------|-------------|
| time       | h = 0.01 | h = 0.1 | h = 1 | h = 0.01 | h = 0.1 | h = 1 | analytical |
| 1          | 0.004081 | 0.045621 | 1.20004 | 3.026e-7 | 4.03e-5 | 0.07955 | 1.48324 |
| 10         | 0.002578 | 0.027378 | 0.591468 | -1.011e-6 | 8.989e-6 | 0.02693 | 4.49444 |
| 100        | 0.001195 | 0.012795 | 0.235295 | -4.925e-6 | -4.925e-6 | 0.00859 | 14.14920 |
| 500        | 0.000661 | 0.006961 | 0.118461 | -3.872e-5 | -3.872e-5 | 0.00386 | 31.62594 |
| 1000       | 0.000504 | 0.005304 | 0.087704 | -3.872e-5 | -3.872e-5 | 0.00386 | 44.72235 |

| Equation 9 | Euler | Runge-Kutta |
|------------|-------|-------------|
| time       | h = 0.01 | h = 0.1 | h = 1 | h = 0.01 | h = 0.1 | h = 1 | analytical |
| 1          | 0.00088 | 0.010849 | 0.303724 | 1.342e-7 | 1.342e-7 | 0.002486 | 0.196275 |
| 10         | -0.01806 | -0.17896 | -1.63646 | 3.948e-5 | 3.948e-5 | 0.003139 | 29.09346 |
| 100        | -0.04063 | -0.40863 | -3.94863 | 0.000369 | 0.000369 | 0.003369 | 724.16563 |
| 500        | -0.05769 | -0.56769 | -5.55769 | 0.002309 | 0.002309 | 0.002309 | 5217.74769 |
| 1000       | -0.05115 | 0.65115 | -6.25115 | 0.00270 | 0.00270 | 0.00270 | 11818.651 |

| Equation 11 | Euler | Runge-Kutta |
|-------------|-------|-------------|
| time        | h = 0.01 | h = 0.1 | h = 1 | h = 0.01 | h = 0.1 | h = 1 | analytical |
| 1           | 0.000789 | 0.007675 | 0.053916 | -7.037e-8 | -7.037e-8 | 6.293e-5 | 0.005272 |
| 10          | 2.406e-6 | 2.406e-6 | 2.241e-5 | 0.005272 | 0.005272 | 0.520572 | 1.168347 |
| 100         | 4.533e-6 | 4.533e-6 | -5.467e-6 | 0.002309 | 0.002309 | 0.002309 | 724.16563 |
| 500         | 0.005004 | 0.050044 | 0.500314 | -3.122e-6 | -3.122e-6 | -3.122e-6 | 5217.74769 |
| 1000        | 0.004997 | 0.049957 | 0.499637 | -3.122e-6 | -3.122e-6 | -3.122e-6 | 11818.651 |

The analytical solution for this task is equation 10:

\[ p(t) = t \cdot \log(t^2 + 1) - 2t + 2\arctan(t) - \frac{1}{2} \exp^{-2t} \]  

\[ (10) \]
V. Discussion

The BioDynaMo project has a set of classes to deal with different types of simulation: biological, physical and diffusion. The biological simulation consist of different modules, for example: *GrowthDivide*, *GrowthModule*, *Chemotaxis*. Each cell can carry an individual set of biology modules.

As an extension to the project, a new biology module *GeneExpression* has been added. This module stores protein values for each cell. Furthermore, this module allows to define custom laws in the form of ODEs that define protein concentration. To solve sets of ODEs, a user may select from two available options depending on their current needs: Euler method for higher performance and Runge-Kutta method for accuracy. However, in the case when time step of simulation is small in order to speed up simulation, Euler method can be used without a notable loss of accuracy. In addition, user specifies functions $f(p(t), t)$ by which changes in concentration of proteins is calculated and initial
values for proteins.

VI. RELATED WORKS

The project described in [18] aims at solving problems similar to those addressed by BioDynaMo: creating highly scalable biological simulations that can cover a wide range of tasks in biomedical science. The two projects also use a similar data structure in the spatial organization, i.e. spatial trees. This structure provides good performance on the most frequent operations, such as finding the nearest neighbor. However, this simulation does not include features like gene expression that have been already implemented in BioDynaMo.

The work presented in [19] uses differential equations in order to model changes in the concentration of proteins as well. Although, this work uses gene expression in a specific task, i.e. studying the influence of different protein pathways in the spreading of cancer cell

The article [11] presents the reconstruction of gene regulatory networks (GRNs) from experimental data of gene expression through computational methods. Paper describes models for reverse engineering GRNs from gene expression data. Along with Boolean and Bayesian models Differential equation model described.

The research [20] introduces new statistical gene network estimation method based on the dynamic Bayesian network and nonparametric regression model with advantages over Bayesian and Boolean networks. For example, it can detect nonlinear dependencies. This article can be taken into consideration in further work if the statistical approach will be implemented in BioDynaMo.

VII. CONCLUSION

BioDynaMo is a simulator of biological processes developed by an international community of researchers and software engineers working in close synergy in order to implement the requirements coming from neuroscientists. Bridging the gap between the neuroscience perspective and the technical software engineering perspective is one of the task of our team within the project. Collecting requirements from specialists and implementing them is a non-trivial task. We advocate the necessity of developing a requirements engineering framework offering different syntaxes to represent the same concept (text, graphical, mathematical) in order to facilitate the communication between the stakeholders without forcing them to change their current habits. For example, a comprehensive modeling framework as discussed in [21] may solve the problem.

Among the requirements to be understood and implemented, our research team focused first on the development of data structures for the physical engine [22], and then on the modeling of gene expression in the simulation, which is discussed in detail in this paper. We discussed the entire process of modeling and development presenting the problem, the possible solving techniques, providing multiple implementations, and comparing them under qualitative and quantitative aspects.

Simulation of biologic dynamics has several applications, some direct and some indirect. Computer simulations of biological tissue dynamics can serve the purpose of understanding diseases and dysfunction at the operational and sub-operational level as well as being the effective substitute for drug testing that does not involve living beings. At the same time, potential application scenarios are also foreseeable in cognitive architecture [23], where simulations may provide insight into and, in general, for the development of smart systems, including smart houses [24], [26] and smart automotive systems [27].

REFERENCES

[1] L. Breitwieser, R. Bauer, A. D. Meglio, L. Johard, M. Kaiser, M. Manca, M. Mazzara, F. Rademakers, and M. Talanov, “The biodynamo project: Creating a platform for large-scale reproducible biological simulations,” 4th Workshop on Sustainable Software for Science: Practice and Experiences (WSSSPE4), 2016.

[2] L. Johard, L. Breitwieser, A. Di Meglio, M. Manca, M. Mazzara, and M. Talanov, “The biodynamo project: a platform for computer simulations of biological dynamics,” arXiv preprint arXiv:1608.01818, 2016.

[3] P. J. Denning, “The locality principle,” in Communication Networks And Computer Systems: A Tribute to Professor Erol Gelenbe. World Scientific, 2006, pp. 43–67.

[4] F. Zubler and R. Douglas, “A framework for modeling the growth and development of neurons and networks,” Frontiers in computational neuroscience, vol. 3, p. 25, 2009.

[5] M. Mazzara, F. Rademakers, M. Talanov, and A. D. Tchitchigin, “The biodynamo project: experience report,” Advanced Research on Biologically Inspired Cognitive Architectures, p. 117, 2017.

[6] S. Hoehme and D. Drasdo, “A cell-based simulation software for multi-cellular systems,” Bioinformatics, vol. 26, no. 20, pp. 2641–2642, 2010.

[7] T. Misteli, “Protein dynamics: implications for nuclear architecture and gene expression,” Science, vol. 291, no. 5505, pp. 843–847, 2001.

[8] S. B. Carroll, J. K. Grenier, and S. D. Weatherbee, From DNA to diversity: molecular genetics and the evolution of animal design. John Wiley & Sons, 2013.

[9] T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. L. Loh, J. R. Downing, M. A. Caligiuri et al., “Molecular classification of cancer: class discovery and class prediction by gene expression monitoring,” science, vol. 266, no. 5439, pp. 531–537, 1999.

[10] F. Markowitz and R. Spang, “Inferring cellular networks--a review,” BMC bioinformatics, vol. 8, no. 6, p. S5, 2007.
[11] M. Hecker, S. Lambeck, S. Toepfer, E. Van Someren, and R. Guthke, “Gene regulatory network inference: data integration in dynamic models - a review,” *Biosystems*, vol. 96, no. 1, pp. 86–103, 2009.

[12] A. Ay and D. N. Arnosti, “Mathematical modeling of gene expression: a guide for the perplexed biologist,” *Critical reviews in biochemistry and molecular biology*, vol. 46, no. 2, pp. 137–151, 2011.

[13] S. A. Kauffman, “Metabolic stability and epigenesis in randomly constructed genetic nets,” *Journal of theoretical biology*, vol. 22, no. 3, pp. 437–467, 1969.

[14] T. Chen, H. L. He, G. M. Church et al., “Modeling gene expression with differential equations.” in *Pacific symposium on biocomputing*, vol. 4, no. 29, 1999, p. 40.

[15] J. Hadamard, *Lectures on Cauchy’s problem in linear partial differential equations*. Courier Corporation, 2014.

[16] J. Butcher, “Runge-kutta methods,” *Scholarpedia*, vol. 2, no. 9, p. 3147, 2007.

[17] K. E. Atkinson, *An introduction to numerical analysis*. John Wiley & Sons, 2008.

[18] M. Cytowski and Z. Szymanska, “Large-scale parallel simulations of 3d cell colony dynamics,” *Computing in Science & Engineering*, vol. 16, no. 5, pp. 86–95, 2014.

[19] I. Ramis-Conde, M. A. Chaplain, A. R. Anderson, and D. Drasco, “Multi-scale modelling of cancer cell intravasation: the role of cadherins in metastasis,” *Physical biology*, vol. 6, no. 1, p. 016008, 2009.

[20] S. Kim, S. Imoto, and S. Miyano, “Dynamic bayesian network and nonparametric regression for nonlinear modeling of gene networks from time series gene expression data,” *Biosystems*, vol. 75, no. 1, pp. 57–65, 2004.

[21] Z. Yan, E. Cimpian, M. Zaremba, and M. Mazzara, “BPMO: semantic business process modeling and WSMO extension,” in *2007 IEEE International Conference on Web Services (ICWS 2007), July 9-13, 2007, Salt Lake City, Utah, USA*, 2007, pp. 1185–1186.

[22] I. Dmitrenok, V. Drobnyy, L. Johard, and M. Mazzara, “Evaluation of spatial trees for the simulation of biological tissue,” in *Advanced Information Networking and Applications Workshops (WAINA), 2017 31st International Conference on*. IEEE, 2017, pp. 276–282.

[23] J. Vallverdú, M. Talanov, S. Distefano, M. Mazzara, A. Tchitchigin, and I. Nurgaliev, “A cognitive architecture for the implementation of emotions in computing systems,” *Biologically Inspired Cognitive Architectures*, vol. 15, pp. 34–40, 2016.

[24] M. Nalin, I. Baroni, and M. Mazzara, “A holistic infrastructure to support elderlies’ independent living,” *Encyclopedia of E-Health and Telemedicine, IGI Global*, 2016.