Письма в ВАВИЛОВСКИЙ ЖУРНАЛ ГЕНЕТИКИ И СЕЛЕКЦИИ
Letters to Vavilov Journal of Genetics and Breeding
2026 • 12 • 3 • 145-152

pismavavilov.ru
doi 10.18699/letvjgb-2026-12-18

Оригинальное исследование

Программная архитектура модуля для цифровой платформы «Моделирование генных сетей»

Ф.В. Казанцев [1] ✉, А.М. Мухин [1], А.А. Долматова [1, 2], Ю.Г. Матушкин [1], С.А. Лашин [1]

Аннотация. В работе представлено описание архитектуры универсального вычислительного модуля, предназначенного для решения широкого спектра задач системной биологии, включая массовые вычисления, сканирование параметров и анализ чувствительности моделей. Анализ образовательной и исследовательской практики показывает, что растущая доля информационных технологий в биологических исследованиях требует от специалистов высокой технической культуры, что нередко вызывает методологические затруднения. В статье рассмотрены механизмы организации запуска и взаимодействия с вычислительными модулями на уровне операционной системы, что позволяет глубже понять принципы работы современных конвейерных инструментов обработки биологических данных. Обсуждаются типичные проблемы, возникающие вследствие технических ограничений, и предлагаются рекомендации по организации вычислительного процесса на персональных компьютерах для минимизации риска сбоев. В приложении к статье приведен программный код базовых вычислительных модулей и пример их интеграции в автоматизированный конвейер, включающий подготовку параметров, массовые вычисления с сохранением метаданных, постобработку и визуализацию результатов применительно к задаче математического моделирования генных сетей.

Ключевые слова: системная биология; математическое моделирование; шаблонная модель; унифицированный вычислительный модуль; массовые вычисления; варьирование параметров; вычислительный анализ; автоматизация исследований

Для цитирования: Казанцев Ф.В. , Мухин А.М., Долматова А.А., Матушкин Ю.Г., Лашин С.А. Программная архитектура модуля для цифровой платформы «Моделирование генных сете. *Письма в Вавиловский журнал генетики и селекции.* 2026;12(3):145-152. doi 10.18699/letvjgb-2026-12-18

Финансирование: Работа выполнена при поддержке бюджетного проекта № FWNR-2026-0023.

Original article

Software architecture of the module for the digital platform Gene Network Modeling

F.V. Kazantsev [1] ✉, A.M. Mukhin [1], A.A. Dolmatova [1, 2], Yu.G. Matushkin [1], S.A. Lashin [1]

Abstract. An analysis of educational and research practices shows that the increasing role of information technologies in biological research demands a high level of technical proficiency from specialists, which often leads to methodological challenges. Here we present the architecture of a universal computational module designed to address a wide range of systems biology tasks, including batch computing, parameter scanning, and model sensitivity analysis. The paper examines the mechanisms for launching and interacting with computational modules at the operating system level, providing a deeper understanding of the principles underlying modern pipeline tools for biological data processing. Typical problems arising from technical limitations are discussed, and recommendations are provided for organizing computational workflows on personal computers to minimize the risk of system failures. The appendix provides the source code for basic computational modules and an example of their integration into an automated pipeline, which includes parameter preparation, batch computing with metadata preservation, post-processing, and result visualization, applied to the mathematical modeling of gene networks.

Key words: systems biology; mathematical modeling; template model; unified computational module; batch computing; parameter variation; computational analysis; research automation

For citation: Kazantsev F.V., Mukhin A.M., Dolmatova A.A., Matushkin Yu.G., Lashin S.A. Software architecture of the module for the digital platform Gene Network Modeling. *Pisma v Vavilovskii Zhurnal Genetiki i Selektsii = Lett Vavilov J Genet Breed.* 2026;12(3):145-152. doi 10.18699/letvjgb-2026-12-18

Funding: This paper was supported by the budget project No. FWNR-2026-0023.

[1] Федеральный исследовательский центр Институт цитологии и генетики Сибирского отделения Российской академии наук, Новосибирск, Россия
Institute of Cytology and Genetics of the Siberian Branch of the Russian Academy of Sciences, Novosibirsk, Russia
[2] Новосибирский национальный исследовательский государственный университет, Новосибирск, Россия
Novosibirsk State University, Novosibirsk, Russia
✉ KazantcevFV@icg.sbras.ru, Kazfdr@bionet.nsc.ru

© Казанцев Ф.В. , Мухин А.М., Долматова А.А., Матушкин Ю.Г., Лашин С.А., 2026

Контент доступен под лицензией Creative Commons Attribution 4.0

Введение

Исследовательские задачи в области биологии претерпевают значительные изменения под влиянием современных технологий и развития материально-технической базы. Массовое использование высокопроизводительного оборудования приводит к накоплению больших массивов биологических данных, что требует от исследователя уверенного владения вычислительными методами для их обработки. Ключевые компетенции в этой области включают: подготовку данных в формате, пригодном для автоматизированного анализа; управление множественными запусками программ обработки данных; навыки работы с высокопроизводительными вычислительными кластерами (англ. high performance cluster – HPC).

Акцент в научных работах смещается от методов первичного получения данных к их высокоуровневой обработке, которая может включать несколько шагов анализа с применением математических и статистических методов. Однако у специалистов с фундаментальным биологическим образованием интеграция современных вычислительных инструментов может вызывать затруднения. Типичный сценарий работы исследователя зачастую сводится к поиску готового программного обеспечения (нередко с графическим интерфейсом), что при решении задач, даже незначительно отличающихся от стандартных, приводит к фрагментации рабочего процесса: многократному запуску программ в ручном режиме, ручной передаче файлов между ними и последующей обработке результатов в программах поддержки электронных таблиц широкого профиля, например Microsoft Excel.

Значительно более эффективным подходом к организации рабочего процесса является полная автоматизация последовательности отдельных шагов анализа в виде воспроизводимого вычислительного сценария (конвейера). В таком сценарии каждый этап документирован, а процесс обработки данных является однозначным и детерминированным. Роль исследователя в этом сценарии включает в себя подготовку входных данных, контроль промежуточных результатов и модификацию параметров сценария при изменении условий задачи. Отметим, что такая автоматизация – лишь один из больших этапов научного исследования, который также можно разбить на несколько стадий.

В данной работе рассматриваются принципы построения базового вычислительного модуля, правила его создания и контроля корректности исполнения. Вычислительный модуль определяется как универсальный, автономно выполняемый компонент сценария. В статье описаны правила интеграции модуля в конвейер обработки данных, а также масштабирование сценария на этапах от загрузки входных данных до визуализации результатов. В рамках цифровой платформы «Моделирование генных сетей», разрабатываемой в ИЦиГ СО РАН, реализованы сценарии, оформленные по этим рекомендациям.

Программные технологии и методы

Под вычислительным модулем понимается автономная вычислительная единица, запуск которой не требует наличия интегрированной среды разработки (IDE) или иной специализированной программы. Таким образом, запуск осуществляется непосредственно средствами операционной системы. Построение конвейеров обработки данных может быть реализовано как с использованием командных оболочек, например BASH, так и скриптовых языков высокого уровня, например Python, R, Julia. В работе описаны стандартные механизмы обмена данными и состояниями, поддерживаемые в языках программирования C++, Python, Octave, а также в других специализированных инженерных средах. Функционирование подобных программных модулей и сценариев кроссплатформенно и совместимо с вычислительными инфраструктурами операционных систем (Барретт, 2026).

Результаты

Базовый модуль расчета

Под *базовым модулем* в данной работе понимается единица обработки информационного блока, т. е. программная реализация, выполняющая обработку определенного набора данных. Это могут быть: скомпилированная программа, которая обрабатывает блок информации; сценарий, интерпретируемый средствами одного из языков программирования и обрабатывающий блок информации; сценарий обработки блока информации средствами операционной системы.

При проектировании модуля, ориентированного на встраивание в автоматизированные сценарии, необходимо обеспечить унификацию точек его взаимодействия с внешней средой. Эти точки служат механизмом управления работой программы и мониторинга корректности ее исполнения. Внешнее управление должно обеспечивать единообразие процесса запуска и получения результатов, гарантируя *воспроизводимость* и *повторяемость* вычислений (Tiwari et al., 2021). При выполнении модуля не должно быть никаких прерываний для взаимодействия с пользователем. Задача может иметь статус «успешно выполнена», или завершаться с информированием об ошибке исполнения.

Воспроизводимость определяется как способность системы выдавать идентичные результаты при передаче одинаковых входных параметров на различных вычислительных узлах (при условии, что в рамках вычислений не используются стохастические методы). Это достигается тем, что все параметры вычислительной процедуры, влияющие на ход выполнения, задаются в виде входных параметров модуля (параметры запуска и/или переменные окружения). В случае использования стохастических алгоритмов для обеспечения консистентности необходимо фиксировать начальное значение генератора псевдослучайных чисел (англ. seed), так же, через переменные окружения или параметры запуска (Dutta et al., 2022).

Повторяемость подразумевает получение согласованных результатов при использовании альтернативной реализации модуля (например, написанной на другом языке программирования или с использованием иной библиотеки), что обеспечивает возможность взаимозаменяемости компонентов конвейера без нарушения логики передачи данных.

Запуск модуля

Стандартный запуск вычислительного модуля должен осуществляться через интерфейс командной строки (англ. command line interface – CLI). Для минимизации усилий по конфигурации регулярных запусков необходимо решить следующие задачи:

1. Передача параметров: все управляющие воздействия должны быть вынесены во входные аргументы командной строки. Параметры должны иметь значения по умолчанию, а программа должна поддерживать их вывод (например, через флаги `--help` или `--info`).
2. Управление путями сохранения: критически важен контроль текущей рабочей директории (параметр `cwd` – current working directory). Результаты работы должны сохраняться строго в директорию, указанную в параметрах запуска.
3. Контроль исполнения: статус завершения работы модуля должен однозначно интерпретироваться внешней системой через стандартные коды возврата (англ. exit codes).

Для реализации вышеописанных задач модуль должен:

1. Принимать и обрабатывать аргументы командной строки, используя общепринятые соглашения об именовании (например, `--input`, `--output`, `--prm` для путей к входным данным, выходной директории и файлу параметров соответственно (Барретт, 2026)).
2. Поддерживать передачу наборов параметров через структурированные, машино- и человеко-читаемые файлы (форматы CSV, TSV, JSON). Операции чтения и записи таких форматов поддерживаются библиотеками во многих популярных языках программирования и инженерных средах моделирования.
3. Осуществлять запись результатов исключительно в предварительно созданную выходную директорию.

а) При разрешении создания вложенных директорий существует риск появления трудноуловимых ошибок, связанных с возможным случайным использованием неоднозначно интерпретируемых символов в именах файлов и директорий (при наличии в пути до директорий пробелов, спецсимволов, символов в разных регистрах, символов языков, отличных от английского и т. п.).

б) При запрете создания директорий необходимо проконтролировать существование нужной директории до запуска модуля. В этой директории, и только в ней, модуль должен иметь права на создание файлов.

4. Возвращать системе целочисленный код завершения: `0` свидетельствует об успешном выполнении; любое ненулевое значение указывает на ошибку. Значения 0 и 1 зарезервированы за условиями успеха/неуспеха в общем смысле работы модуля. Вы можете специфицировать собственные коды для идентификации условий сбоя программы: `1` – сбой при выполнении; `2` – переданы неверные параметры или файл не найден; `3` – ошибка на этапе сохранения результатов вычислений и т. д. (Барретт, 2026).

Дополнительно следует предусмотреть параметр, управляющий очисткой выходной директории перед запуском. По умолчанию очистка не рекомендуется, что позволяет сохранять файлы параметров метода вместе с результатами расчета в одном месте для обеспечения полной воспроизводимости вычислительного эксперимента.

Пример базового модуля

В качестве демонстрационного примера рассматривается модуль, реализующий численный анализ математической модели гипотетической генной сети (ГГС4), состоящей из четырех белков, взаимно ингибирующих синтез друг друга (Лихошвай и др., 2003; Фадеев, Лихошвай, 2003; Kazantsev et al., 2025). Данная модель является модификацией классического репрессилятора (Elowitz, Leibler, 2000), (рис. 1, ребра генной сети, помеченные черным цветом), в которую добавлен четвертый белок, ингибирующий биосинтез остальных трех, при этом исходные белки также ингибируют его синтез (см. рис. 1, ребра генной сети, помеченные красным цветом). Математическая модель представляет собой систему из четырех обыкновенных дифференциальных уравнений (ОДУ) с соответствующими параметрами (см. рис. 1, правая часть). Исходная модель репрессилятора при выбранных значениях параметров демонстрирует неустойчивое стационарное состояние и осциллирующее поведение. Включение белка P4 добавляет новый режим функционирования – триггерный, позволяющий переключаться между устойчивым стационаром и осцилляциями исключительно за счет варьирования начальной концентрации белка P4 (Kazantsev et al., 2025).

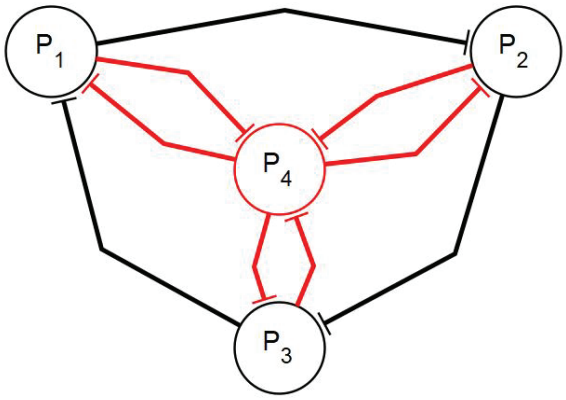
Представленная модель иллюстрирует возможности управления вычислительными сценариями для поиска границ параметров, разделяющих различные режимы функционирования системы. В приложении[1] приведены файлы модели и запуска вычислительного модуля, реализованные на языке программирования Python. Модель описана восемью параметрами: 4 начальные концентрации белков и 4 параметра уравнений модели: α или *kalpha* – базальный уровень экспрессии белка; γ или *kgamma* – коэффициент Хилла; *k* – константа ингибирования биосинтеза белка; *kd* – скорость диссипации белка. Дополнительные параметры управляют вычислительным процессом: время моделирования (`tEnd`) и интервал вывода данных (`printStep`). Все значения вынесены в конфигурационный табличный файл `default.csv` (см. приложение). Этот файл содержит параметры, которые будут использоваться на дальнейших этапах изложения как значения по умолчанию.

Запуск вычислительного эксперимента

Для запуска модуля рекомендуется создать отдельную рабочую директорию с минимальной глубиной вложенности, расположенную ближе к корневой директории файловой системы. Путь к директории должен содержать только символы латинского алфавита, а также допустимые специальные символы подчеркивания (`_`) или точки (`.`). Наличие пробелов или символов национальных алфавитов в путях недопустимо, так как это может привести к ошибкам чтения и записи файлов.

Следующим этапом необходимо перенести в созданную рабочую директорию файлы модели и сценариев вычисли-

[1] Приложение см. по адресу:
https://pismavavilov.ru/wp-content/uploads/2026/09/appx2.zip



$$\frac{dP1}{dt} = \frac{a}{1+\left(\frac{P3}{k}\right)^{\gamma}+\left(\frac{P4}{k}\right)^{\gamma}} - kd \times P1$$

$$\frac{dP2}{dt} = \frac{a}{1+\left(\frac{P1}{k}\right)^{\gamma}+\left(\frac{P4}{k}\right)^{\gamma}} - kd \times P2$$

$$\frac{dP3}{dt} = \frac{a}{1+\left(\frac{P2}{k}\right)^{\gamma}+\left(\frac{P4}{k}\right)^{\gamma}} - kd \times P3$$

$$\frac{dP4}{dt} = \frac{a}{1+\left(\frac{P1}{k}\right)^{\gamma}+\left(\frac{P2}{k}\right)^{\gamma}+\left(\frac{P3}{k}\right)^{\gamma}} - kd \times P4$$

Рис. 1. Математическая модель гипотетической генной сети ГГС4. Слева – структурная схема с отображением процесса биосинтеза четырех белков и направление ингибиторных воздействий. Справа – соответствующая структуре система дифференциальных уравнений, учитывающая процессы деградации синтезируемых белков. Черным цветом отмечена структура и уравнения, относящиеся к модели репрессилятора (Elowitz, Leibler, 2000). Красным цветом отмечены дополнения к структуре модели и соответствующие им изменения в уравнениях модели по отношению к модели репрессилятора.

Fig. 1. Mathematical model of the hypothetical gene network HGN4. Left: a structural diagram showing the biosynthesis process of four proteins and the directions of inhibitory interactions. Right: the corresponding system of ordinary differential equations, incorporating the degradation processes of the synthesized proteins. The structure and equations pertaining to the repressilator model (Elowitz, Leibler, 2000) are highlighted in black. Additions to the model structure and the corresponding changes in the equations, relative to the repressilator model, are highlighted in red.

тельных экспериментов из приложения. Команда запуска из терминала имеет следующий вид:

```
>> python modelHGN4.py --prm default.csv --out outputFolder,
```

где «modelHGN4.py» – вычислительный модуль для модели ГГС4, с настроенными параметрами запуска. Через параметр «--prm» задается файл с параметрами расчета, а параметр «--out» задает название директории, куда будут сохраняться результаты. В рамках предложенных условий, выходная директория «outputFolder» должна быть создана до запуска программы.

Результатом выполнения является генерация серии файлов: а) копии файла стартовых параметров; б) табличного файла с временными рядами переменных модели; в) файл со словарем (в данном примере в формате JSON), сопоставляющим идентификаторы переменных (X1, X2...) их смысловой нагрузке (например,

```
«{‘X1’:{ ‘type’ : ‘protein’, ‘name’ : ‘P1’ },
‘X2’:{ ‘type’ : ‘protein’, ‘name’ : ‘P2’},
‘X3’:{ ‘type’ : ‘protein’, ‘name’ : ‘P3’ } }»).
```

Наличие файла-словаря критически важно для автоматизированной выборки целевых данных из результатов моделирования, особенно в моделях с большим числом переменных. Сохранение полного набора параметров и метаданных вместе с результатами является ключевым условием для задач массового анализа.

Сценарий расчетов

Вычислительный сценарий определяет способ организации запуска одного или нескольких базовых модулей с полным контролем над потоком данных между этапами. Создание сценариев возможно как с использованием системных утилит (bash или powershell), так и в специализированных средах (Jupyter Notebook на языке Python, специализированная система управления процессами Snakemake) (Köster, Rahmann, 2012; Kluyver et al., 2016). В данной работе используются примеры на bash, поскольку этот интерпретатор является штатным инструментом на большинстве высокопроизводительных вычислительных кластеров. Кроме этого, системы управления заданиями, например SLURM, требуют написания сценариев запуска именно на bash.

Стандартный сценарий запуска модуля подразумевает определение следующих пунктов:

1. Параметры запуска. Текстовый структурированный файл.
2. Путь до директории сохранения данных.
3. Статус выполнения задачи.

Организация файла параметров в формате CSV (Comma-Separated Values, см. таблицу) – простое и надежное решение, удовлетворяющее критериям чтения и модификации штатными средствами операционной системы.

Преимущества такого подхода:

1. Один файл однозначно задает стартовую точку расчетов.
2. Файлы параметров могут быть сгенерированы программой, что помогает в решении задач сканирования параметров и оценки чувствительности модели.
3. Каждый результат работы модуля сопровождается полным списком использованных параметров, что предотвращает потерю метаданных при обработке больших массивов результатов.

Пример сценария в виде bash файла приведен в листинге ниже. В скрипте переменной «*code*» задается файл вычислительного модуля, который будет запускаться (см. приложение, файл run_1.sh).

Варианты оформления параметров в виде CSV файлов с разными разделителями
Options for formatting parameters in CSV files with different delimiters

Разделитель ‘,’	Разделитель ‘:’	Разделитель ‘=’
T_start,0.0	T_start:0.0	T_start = 0.0
T_end,200.0	T_end:200.0	T_end = 200.0
T_steps,10000	T_steps:10000	T_steps = 10000
X1,1.1	X1:1.1	X1 = 1.0
X2,2.0	X2:2.0	X2 = 2.0
X3,3.0	X3:3.0	X3 = 3.0
X4,0.0	X4:0.0	X4 = 0.0
Kalpha,1.0	Kalpha:1.0	Kalpha = 1.0
Kgamma,3.0	Kgamma:3.0	Kgamma = 3.0
Kd,0.2	Kd:0.2	Kd = 0.2
K,0.5	K:0.5	K = 0.5

```
#!/usr/bin/bash
code=modelHGN4.py
if [ -z $1 ] || [ -z $2 ] # проверка существования параметров командной строки
then
    echo «There is no 1st or 2th parameter to the script»
    exit 1
fi
out=$2
mkdir -p $out
python3 $code --prm $1 --out $out
ret=$?
if [ $ret -ne 0 ]
then
    echo $code with params $1 and $2 have faulted >&2
fi
```

Сначала проверяется наличие параметров запуска. В случае их отсутствия выводится сообщение об этом и сценарий завершается с кодом выхода 1 – ошибка. Если аргументы есть, то на следующем этапе переменной *out* присваивается второй параметр – путь до директории для сохранения результатов. Далее создается эта директория в системе, так как мы условились, что она должна существовать до запуска вычислительного модуля. После чего запускается программа, указанная в переменной *code*, и ей передаются аргументы. Благодаря команде ‘$?’ в переменную *ret* сохраняется значение статуса выполнения последней выполненной программы. При условии неуспеха, выводится сообщение об этом в консоль.

Этот сценарий может быть легко настроен на запуск другого вычислительного модуля через смену значения переменной *code* или расширен дополнительными этапами анализа.

При сравнении bash с Python для запуска реализованного алгоритма следует учитывать следующие моменты:

1. Интерпретатор bash установлен на подавляющем большинстве unix-подобных операционных систем, т. е. имеется практически на всех высокопроизводительных кластерах.
2. Системы запуска задач на кластерах, такие как SLURM, требуют реализовывать сценарии запуска именно в виде скриптов bash.
3. Разработка сценария на языке Python незначительно сложнее варианта на bash, но для работы требует загрузки и установки необходимых сторонних библиотек.
4. При этом Python имеет значительно более богатый функционал как для работы с файловой системой, так и для более тонкой настройки сценария задачи.
5. Допустимо объединять сценарии, написанные на bash и Python. Сценарий bash может запускать дополнительные сценарии на Python как независимые программы, опираясь на озвученные практики управления запуском вычислительного модуля.

Завершение работы модуля, как обсудили выше, сопровождается возвращением статуса выполнения. Это штатный способ информирования в операционной системе. По стандарту (Барретт, 2026), целое число со значением ‘0’ возвращается в случае успешного завершения. Отличное от ‘0’ целое число считается завершением с ошибками. Таким образом, для контроля исполнения вычислительного модуля в рамках этапа сценария достаточно представленных трех пунктов: параметров расчета, задаваемых извне, контроля за местом вывода результатов и обратной связи о корректном завершении. Эти пути взаимодействия позволяют выстраивать сложные сценарии обработки данных.

Характерные сценарии расчетов

Единичный запуск вычислительного модуля интересен в рамках демонстрации работы математической модели для конкретного набора параметров. Типичный вариант орга-

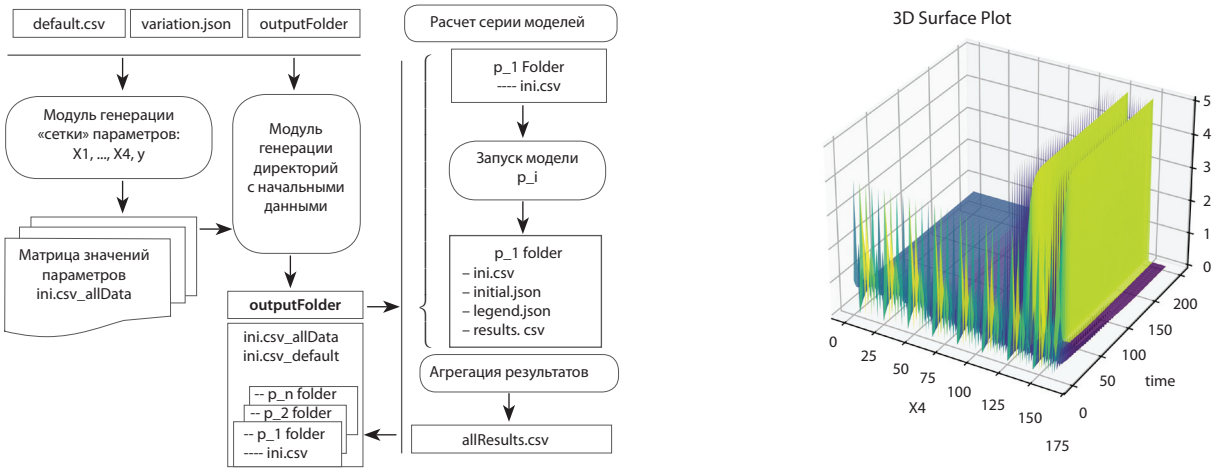


Рис. 2. Слева схема работы сценария, демонстрирующая движение данных между этапами. Сценарий направлен на массовый расчет модели ГГС4 в рамках поиска режимов функционирования модели на заданной сетке параметров. Справа представлены сводные значения временных рядов, соответствующих переменной X4 (белок P4) из всего набора вычислительных экспериментов в условиях варьирования стартовых значений X2, X4 и kgamma в интервалах [2, 3], [0, 4] и [1, 3] соответственно. Остальные параметры представлены в таблице.
Fig. 2. Left: a workflow diagram demonstrating data flow between stages. The scenario is designed for the batch computation of the HGN4 model to identify its operating regimes across a specified parameter grid. Right: summary time series values for variable X4 (protein P4) from the entire set of computational experiments, with the initial values of X2, X4, and kgamma varied within the intervals [2, 3], [0, 4], and [1, 3], respectively. The remaining parameters are listed in the Table.

низации вычислительного эксперимента в математической биологии – это поиск или сканирование параметров для выявления интересных режимов функционирования модели. Часто требуется получить не просто динамику модели в виде временных рядов, описывающих изменения переменных модели, а набор стационарных состояний при варьировании серии параметров. Именно в этих ситуациях возможность управлять параметрами вычислительного сценария через внешние файлы (без модификации кода самого сценария) играют ключевую роль.

Типичный вариант массового анализа включает следующие этапы:

1. Формирование набора стартовых данных (генерация «сетки» параметров).
2. Расчет целевых характеристик (параллельный или последовательный запуск модулей).
3. Агрегация и обработка результирующих данных.
4. Построение сводных диаграмм и визуализация.

Эти этапы могут быть оформлены либо как единый модуль, например, в рамках блокнота Jupyter, либо как четыре независимых модуля. Обмен данными между модулями задает управляющая программа/сценарий. В приложении приведен bash-сценарий, который автоматизирует этот процесс, генерируя набор параметров, в котором варьируются начальные концентрации белков P1-P4 и параметр `kgamma` (коэффициент Хилла). Принципиальная схема сценария представлена на рис. 2, где показано движение данных в рамках сценария и отображены результирующие временные ряды для переменной X4 (белок P4) из модели ГГС4 (см. рис. 1). Файл сценария см. в приложении.

В рамках представленного сценария для каждого расчета создается уникальная поддиректория в «outputFolder». Для контроля исполнения генерируется сводный файл, сопоставляющий варьируемые параметры с именами директорий, где они размещаются. Следующим этапом происходит обход всех вложенных директорий и запуск модуля вычислительного эксперимента с каждым из наборов стартовых параметров, которые находятся там. Соответственным образом меняется директория для выходных данных модуля.

После завершения вычислений запускается финальный этап – обход всех директорий для извлечения значений временных рядов переменных X1 и X4 в единую матрицу, пригодную для дальнейшего анализа. Это отдельный вычислительный модуль, оформленный на языке Python (см. приложение). Получившийся файл с матрицей достаточен для дальнейшей работы по построению сводных диаграмм и анализа данных.

Интеграция с инструментом TSOCED

Получившиеся файлы с характеристиками каждого расчета и результирующие временные ряды были использованы для тестирования метода параметрического анализа расчетов TSOCED (Klimenko et al., 2023). Данный метод, основанный на идее ординации временных рядов, позволяет исследовать влияние параметров модели на форму динамического поведения системы.

TSOCED реализован в виде вычислительного конвейера и включает в себя нормализацию временных рядов, вычисление матрицы попарных расстояний на основе Dynamic Time Warping (DTW) и снижение размерности методом главных координат (PCoA). Конвейер полностью автоматизирован и имеет графический интерфейс: для проведения анализа пользователю необходимо загрузить файл с временными рядами и файл со значениями параметров, предварительно присвоив траекториям и наборам параметров

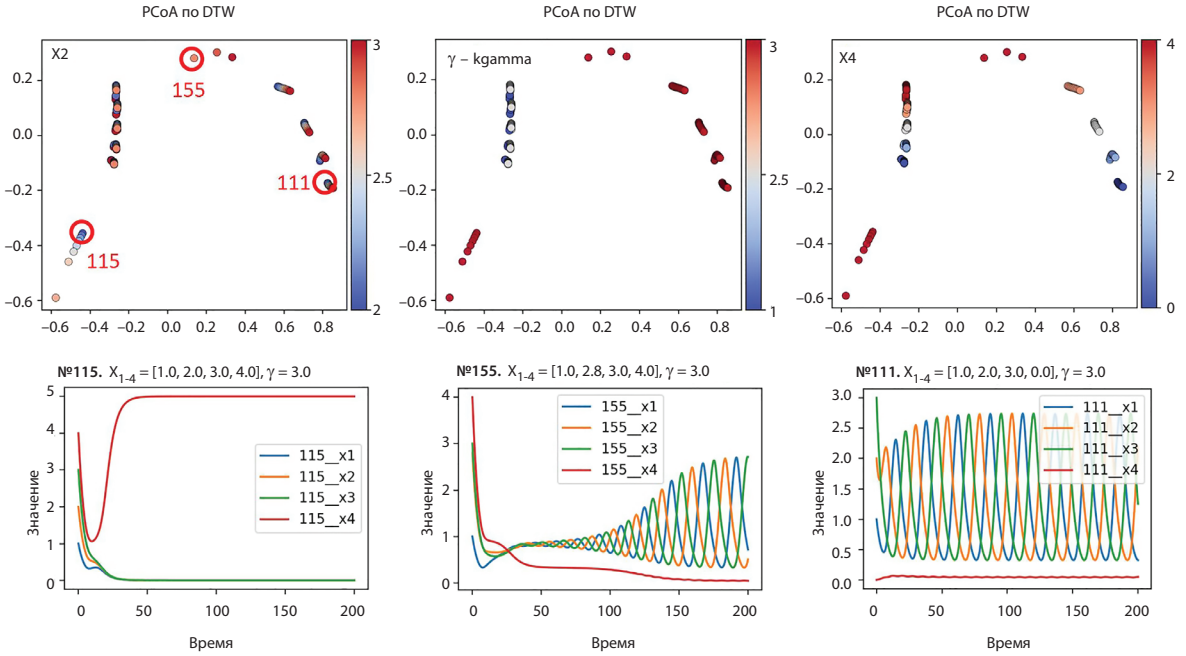


Рис. 3. Сверху: отображение точек решений модели ГГС4 в пространстве главных координат, рассчитанное методом TSOCED. Цветом точек обозначены значения стартовых параметров расчета (слева направо: X2, kgamma и X4 соответственно). Снизу приведены конкретные временные ряды решений модели для выбранных точек из пространства главных координат.

Fig. 3. Top: visualization of the HGN4 model solution points in the principal coordinate space, calculated using the TSOCED method. Point colors represent the values of the initial parameters (from left to right: X2, kgamma, and X4, respectively). Bottom: specific time series of the model solutions for selected points in the principal coordinate space.

согласованные индексы. Именно для этого и готовились сводные файлы данных в предыдущем сценарии.

В полученном низкоразмерном пространстве ординации каждая комбинация параметров представлена отдельной точкой, а их взаимное расположение отражает сходство динамики решений, что позволяет визуально выявлять кластеры траекторий, переходные области между ними и связь между параметрами модели и характером ее решений. Результаты применения метода к полученным сводным временным рядам представлены на рис. 3. На диаграмме отчетливо разделяются режимы функционирования модели: выход на стационар, осциллирующие режимы и переходный процесс, что демонстрирует эффективность предложенного конвейера для параметрического анализа.

Обсуждение

В настоящей работе сформулирован набор правил и рекомендаций по проектированию вычислительных модулей широкого профиля. Данные рекомендации направлены на унификацию механизмов запуска и взаимодействия, что снижает временные затраты на разработку конвейеров обработки данных. Демонстрационный пример иллюстрирует как единичный запуск, так и множественное сканирование параметров модели ГГС4 (Kazantsev et al., 2025), показывающей различные режимы функционирования.

Перенос управляющих воздействий на уровень операционной системы позволяет запускать стандартные сценарии анализа без привлечения ресурсоемких сред разработки, используя легковесные системные инструменты. Это также значительно упрощает масштабирование расчетов на высокопроизводительные вычислительные системы.

Представленный сценарий поэтапной подготовки серии данных, множественных запусков вычислительного модуля и агрегации результатов моделирования является базовым инструментом в анализе поведения математических моделей биологических процессов. Показан вариант расширения этого сценария сторонним инструментом TSOCED, позволяющим наглядно отобразить расчеты в двумерной проекции пространства решений на оси главных координат. Этот вспомогательный инструмент помогает оценивать вклад параметров в поведение модели и отбирать наиболее интересные варианты для дальнейшего исследования.

Заключение

Современное исследование математико-биологической проблемы требует не только разработки самой модели, но и создания инфраструктуры для анализа ее параметрической устойчивости и визуализации множества решений. Среди инструментов этой инфраструктуры – методы анализа параметрической устойчивости, визуализации множества решений и другие вспомогательные инструменты. Предложенный набор рекомендаций по оформлению универсального вычислительного математико-биологического модуля позволяет объединять разнородные программные компоненты в единые сценарии путем стандартизации протоколов обмена данными. Это позволяет создавать управляемое движение данных без непосредственного вмешательства в

программный код самих модулей, снижая вероятность появления трудноуловимых технических ошибок. Сценарии вычислительного анализа типичны, несмотря на отличие самих математических моделей.

Вынесение моделей, методов анализа и визуализации результатов в отдельные вычислительные блоки с унифицированными стандартными протоколами передачи данных между ними позволяет реализовывать сценарии анализа в рамках единой программной платформы. Платформа позволит проводить анализ новых моделей и их вариаций в рамках единых сценариев. При этом подготовка сценариев в рамках стандартных системных инструментов позволяет снизить риски от односторонних изменений лицензий компаний-производителей программного обеспечения.

Список литературы / References

Барретт Д.Д. Linux. Командная строка. Лучшие практики. СПб.: Питер, 2026
[Barrett D.J. Efficient Linux at the Command Line: Boost Your Command-Line Skills. O'Reilly Media, Inc., 2022]

Лихошвай В.А., Матушкин Ю.Г., Фадеев С.И. Задачи теории функционирования генных сетей. *Сибирский журнал индустриальной математики*. 2003;6(2):64-80
[Likhoshvai V.A., Matushkin Y.G., Fadeev S.I. Problems in the theory of the functioning of genetic networks. *Sibirskii Zhurnal Industrial'noi Matematiki*. 2003;6(2):64-80 (in Russian)]

Фадеев С.И., Лихошвай В.А. О гипотетических генных сетях. *Сибирский журнал индустриальной математики*. 2003;6(3):134-153
[Fadeev S.I., Likhoshvai V.A. On hypothetical gene networks. *Sibirskii Zhurnal Industrial'noi Matematiki*. 2003;6(3):134-153 (in Russian)]

Dutta S., Arunachalam A., Misailovic S. To Seed or Not to Seed? An Empirical Analysis of Usage of Seeds for Testing in Machine Learning Projects. In: 2022 IEEE Conference on Software Testing, Verification and Validation (ICST). Valencia, Spain, 2022;151-161. doi 10.1109/ICST53961.2022.00026

Elowitz M.B., Leibler S. A synthetic oscillatory network of transcriptional regulators. *Nature*. 2000;403(6767):335-338. doi 10.1038/35002125

Kazantsev F.V., Lashin S.A., Matushkin Yu.G. Frame-based mathematical models – a tool for the study of molecular genetic systems. *Vavilov J Genet Breed*. 2025;29(8):1288-1294. doi 10.18699/vjgb-25-135

Klimenko A.I., Vorobeva D.A., Lashin S.A. A new visualization and analysis method for a convolved representation of mass computational experiments with biological models. *Mathematics*. 2023;11(12):2783. doi 10.3390/math11122783

Kluyver T., Ragan-Kelley B., Pérez F., Granger B., Bussonnier M., Frederic J., Kelley K., … Ivanov P., Avila D., Abdalla S., Willing C.; Jupyter Development Team. Jupyter Notebooks – publishing format for reproducible computational workflows. In: Positioning and Power in Academic Publishing: Players, Agents and Agendas. IOS Press, 2016;87-90. doi 10.3233/978-1-61499-649-1-87

Köster J., Rahmann S. Snakemake – a scalable bioinformatics workflow engine. *Bioinformatics*. 2012;28(19):2520-2522. doi 10.1093/bioinformatics/bts480

Tiwari K., Kananathan S., Roberts M.G., Meyer J.P., Sharif Shohan M.U., Xavier A., Maire M., Zyoud A., Men J., Ng S., Nguyen T.V.N., Glont M., Hermjakob H., Malik-Sheriff R.S. Reproducibility in systems biology modelling. *Mol Syst Biol*. 2021;17(2):e9982. doi 10.15252/msb.20209982

Конфликт интересов. Авторы заявляют об отсутствии конфликта интересов.
Поступила в редакцию 28.08.2026. После доработки 14.09.2026. Принята к публикации 15.09.2026.