|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ *Робототехники и комплексной автоматизации*

КАФЕДРА *Системы автоматизированного проектирования (РК-6)*

**ОТЧЕТ О ВЫПОЛНЕНИИ ЛАБОРАТОРНОЙ РАБОТЫ**

по дисциплине: «Вычислительная математика»

|  |  |  |
| --- | --- | --- |
| Студент |  | Журавлев Николай Вадимович |
| Группа |  | РК6-52Б |
| Тип задания |  | Лабораторная работа №3 |
| Тема лабораторной работы |  | Модель биологического нейрона |

Студент **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_Журавлев Н. В.\_\_**

*подпись, дата фамилия, и.о.*

Преподаватель **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_Першин А. Ю\_\_**

*подпись, дата фамилия, и.о.*

Москва, 2021 г.

# Оглавление

[Оглавление 2](#_Toc88401886)

[Введение 3](#_Toc88401887)

[Цель выполнения лабораторной работы 4](#_Toc88401888)

[Задачи на лабораторную работу 4](#_Toc88401889)

[Базовая часть 4](#_Toc88401890)

[Выполненные задачи 5](#_Toc88401891)

[Базовая часть 5](#_Toc88401892)

[1. Разработка функций для возврата дискретной траектории 5](#_Toc88401893)

[2. Нахождение траектории заданной динамической системы шагом h = 0.5 7](#_Toc88401894)

[3. Вывод траекторий на графиках 7](#_Toc88401895)

[4. Особенности режимов нейрона 9](#_Toc88401896)

[Заключение 9](#_Toc88401897)

[Список использованных источников 9](#_Toc88401898)

# Введение

Численные методы решения задачи Коши для систем обыкновенных дифференциальных уравнений (ОДУ) 1-го порядка активно используются далеко за пределами стандартных инженерных задач. Примером области, где подобные численные методы крайне востребованы, является нейробиология, где открытые в XX веке модели биологических нейронов выражаются через дифференциальные уравнения 1-го порядка. Математическая формализация моделей биологических нейронов также привела к появлению наиболее реалистичных архитектур нейронных сетей, известных как, спайковые нейронные сети (Spiking Neural Networks). В данной лабораторной работе мы исследуем одну из простейших моделей подобного типа: модель Ижикевича.

Дана система из двух ОДУ 1-го порядка:

и дополнительно условия, определяющего возникновение импульса в нейроне:

если (2)

где *v* - потенциал мембраны (мВ), *u* – переменная восстановления мембраны (мВ), *t* – время (мс), *I* – внешний ток, проходящий через синапс в нейрон от всех нейронов, с которыми он связан.

Описания параметров представленной системы:

*a* – задаёт временной масштаб для восстановления мембраны *(чем больше a, тем быстрее происходит восстановление после импульса)*;

*b* – чувствительность переменной восстановления к флуктуациям разности потенциалов;

с – значение потенциала мембраны сразу после импульса;

d – значение переменной восстановления мембраны сразу после импульса;

*Таблица 1: Характерные режимы заданной динамической системы и соответствующие значение её параметров.*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Режим** | *a* | *b* | *c* | *d* |
| Tonic spiking (TS) | 0.02 | 0.2 | -65 | 6 |
| Phasic spiking (PS) | 0.02 | 0.25 | -65 | 6 |
| Chattering (C) | 0.02 | 0.2 | -50 | 2 |
| Fast spiking (FS) | 0.1 | 0.2 | -65 | 2 |

# Цель выполнения лабораторной работы

Изучить численные методы задачи Коши для систем обыкновенных дифференциальных уравнений (ОДУ) 1-го порядка на примере модели *Ижикевича* и реализовать с использованием языка программирования *Python*. Проанализировать результат, полученный при применении разных режимов работы заданной динамической системы.

# Задачи на лабораторную работу

## Базовая часть

1. Реализовать следующие функции, каждая из которых возвращает дискретную траекторию системы ОДУ с правой частью, заданной функцией **f**, начальным условием **x\_0**, шагом по времени **h** и конечным временем **t\_n**:
2. ***euler(x\_0, t\_n, f, h)***, где дискретная траектория строится с помощью метода Эйлера;
3. ***implicit\_euler(x\_0, t\_n, f, h)***, где дискретная траектория строится с помощью неявного метода Эйлера;
4. ***runge\_kutta(x\_0, t\_n, f, h)***, где дискретная траектория строится с помощью метода Рунге-Кутта 4-ого порядка;
5. Для каждого из реализованных методов численно найти траектории заданной динамической система, используя шаг *h* = 0.5 и характерные режимы, указанные в *таблице 1*. В качестве начальных условий можно использовать *v*(0) = c и *u*(0) *= bv*(0). Внешний ток принимается равным *I =* 5.
6. Вынести полученные траектории на четырёх отдельных графиках как зависимость потенциала мембраны *v* от времени *t*, где каждый график должен соответствовать своему характерному режиму работы нейрона.
7. По полученным графикам кратко описать особенности указанных режимов.

# Выполненные задачи

## Базовая часть

1. Разработка функций для возврата дискретной траектории

Разработаны функции, которые возвращают дискретную траекторию систему ОДУ с правой частью, заданной функцией **f**, начальным условием **x\_0**, шагом по времени **h**, конечным временем **t\_n**:

1. ***euler(x\_0, t\_n, f, h)***, где потенциал и переменная восстановления мембраны рассчитываются с помощью метода Эйлера (*Листинг 1*).

*Листинг 1: реализация метода Эйлера для нахождения потенциала мембраны и переменной восстановления мембраны*

|  |
| --- |
| def euler(t0, tn, f, h):  m = int((tn - t0) / h)  v = np.zeros((m + 1,))  u = np.zeros((m + 1,))  v[0] = c  u[0] = b \* v[0]  for i in range(m):  v[i + 1] = v[i] + h \* f[0](u[i], v[i])  u[i + 1] = u[i] + h \* f[1](u[i], v[i])  if v[i + 1] >= 30:  v[i + 1] = c  u[i + 1] = u[i + 1] + d  return u, v |

1. ***implicit\_eluer(x\_0, t\_n, f, h)***, где потенциал и переменная восстановления мембраны находятся с помощью с помощью неявного метода Эйлера (*Листинг 2*);

*Листинг 2: реализация неявного метода Эйлера для нахождения потенциала мембраны и переменной восстановления мембраны*

|  |
| --- |
| def implicit\_euler(t0, tn, f, h):  def phi\_v(vi\_1, ui, vi):  return vi\_1 - vi - h \* f[0](u[i], v[i])  def phi\_u(ui\_1, ui, vi):  return ui\_1 - ui - h \* f[1](u[i], v[i])  m = int((tn - t0) / h)  v = np.zeros((m + 1,))  u = np.zeros((m + 1,))  v[0] = c  u[0] = b \* v[0]  for i in range(m):  v[i + 1] = optimize.fsolve(phi\_v, v[i], args=(u[i], v[i]))  u[i + 1] = optimize.fsolve(phi\_u, v[i], args=(u[i], v[i]))  if v[i + 1] >= 30:  v[i + 1] = c  u[i + 1] = u[i + 1] + d  return u, v |

1. ***runge\_kutta(x\_0, t\_n, f, h)***, где потенциал и переменная восстановления мембраны находятся с помощью с помощью метода Рунге-Кутта 4-ого порядка (*Листинг 3*);

*Листниг 3: реализация метода Рунге-Кутта* *для нахождения потенциала мембраны и переменной восстановления мембраны*

|  |
| --- |
| def runge\_kutta(t0, tn, f, h):  m = int((tn - t0) / h)  v = np.zeros((m + 1,))  u = np.zeros((m + 1,))  v[0] = c  u[0] = b \* v[0]  for i in range(m):  vk1 = h \* f[0](u[i], v[i])  vk2 = h \* f[0](u[i] + h / 2, v[i] + vk1 / 2)  vk3 = h \* f[0](u[i] + h / 2, v[i] + vk2 / 2)  vk4 = h \* f[0](u[i] + h, v[i] + vk3)  uk1 = h \* f[1](u[i], v[i])  uk2 = h \* f[1](u[i] + uk1 / 2, v[i] + h / 2)  uk3 = h \* f[1](u[i] + uk2 / 2, v[i] + h / 2)  uk4 = h \* f[1](u[i] + uk3, v[i] + h)  v[i + 1] = v[i] + (vk1 + 2 \* vk2 + 2 \* vk3 + vk4) / 6  u[i + 1] = u[i] + (uk1 + 2 \* uk2 + 2 \* uk3 + uk4) / 6  if v[i + 1] >= 30:  v[i + 1] = c  u[i + 1] = u[i + 1] + d  return u, v |

2. Нахождение траектории заданной динамической системы шагом h = 0.5

Для каждого из реализованных методов были численно найдены траектории заданной динамической системы, используя шаг ℎ = 0.5 и характерные режимы, указанные в таблице 1. В качестве начальных условий были использованы 𝑣(0) = 𝑐 и 𝑢(0) = 𝑏𝑣(0). Внешний ток равен 𝐼 = 5. Подставим эти значения в формулу (1) для каждого режима (*Листинг 4*).

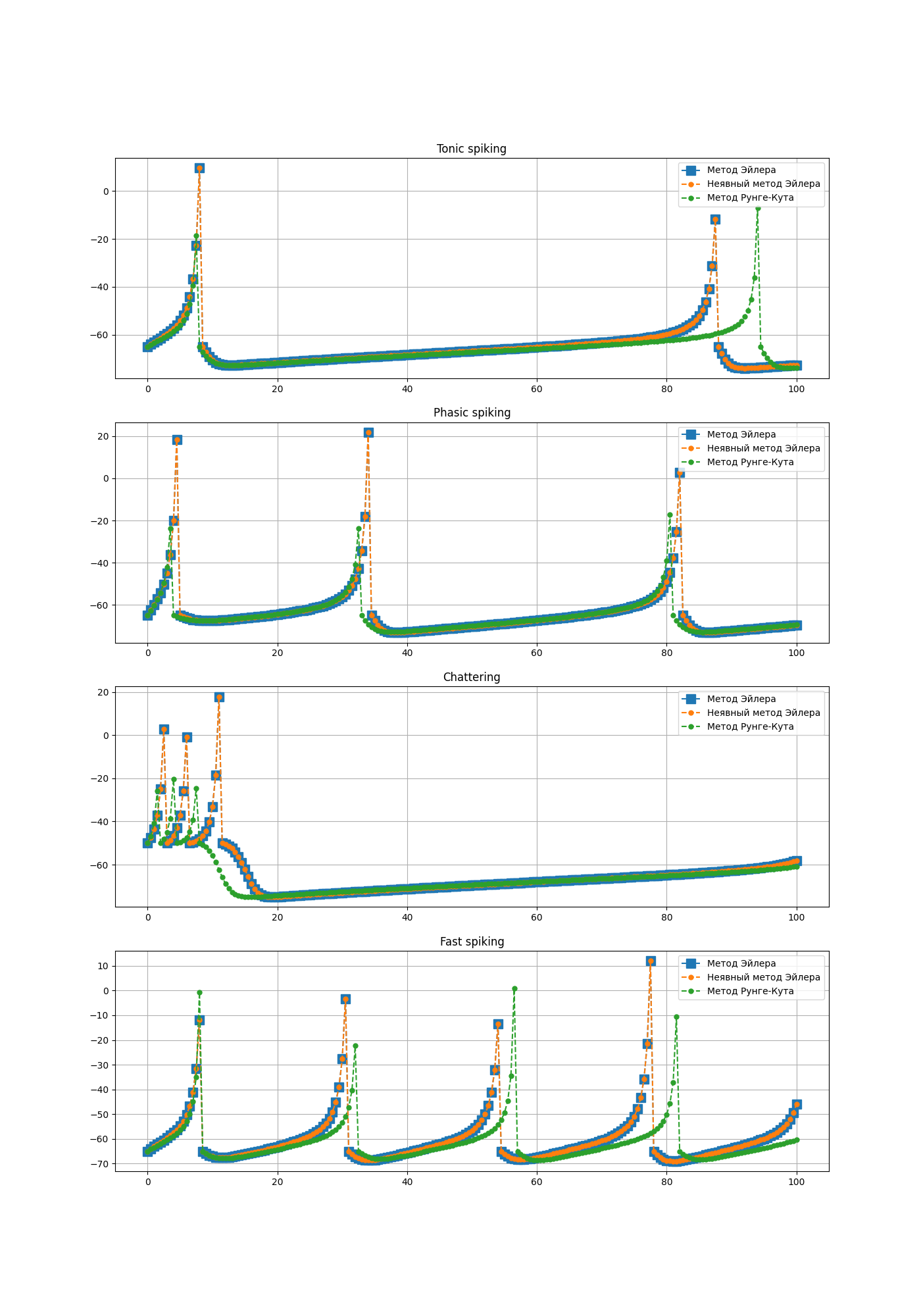
*Листинг 4: реализация численного нахождения дискретной траектории.*

|  |
| --- |
| fig, axs = plt.subplots(4, 1, figsize=(14, 20))  names = ['Tonic spiking', 'Phasic spiking', 'Chattering', 'Fast spiking']  for i, j, ax in zip(['TS', 'PS', 'C', 'FS'], names, axs):  a = mode[i][0]  b = mode[i][1]  c = mode[i][2]  d = mode[i][3]  u\_euler, v\_euler = euler(t0, tn, [f1, f2], h)  u\_implicit\_euler, v\_implicit\_euler = implicit\_euler(t0, tn, [f1, f2], h)  u\_runge\_kutta, v\_runge\_kutta = runge\_kutta(t0, tn, [f1, f2], h)  ax.plot(t, v\_euler, 's--', label='Метод Эйлера', markersize=10)  ax.plot(t, v\_implicit\_euler, 'o--', label='Неявный метод Эйлера', markersize=5)  ax.plot(t, v\_runge\_kutta, 'o--', label='Метод Рунге-Кута', markersize=5) |

3. Вывод траекторий на графиках

Были выведены полученные траектории на четырёх отдельных графиках как зависимость потенциала мембраны *v* от времени *t*, где каждый график соответствует своему характерному режиму работы нейрона. Полученные графики изображены на *Рисунке 1*.

*Рисунок 1: зависимость потенциала от времени. Для разных режимов был выведен свой график.*



4. Особенности режимов нейрона

Восстановление мембраны, зависит от расстояния между двумя скачками графика, чем раньше они появляются, тем лучше восстановление мембраны. Используя это, по полученным графикам можно описать особенности указанных режимов:

1. Режим *Tonic Spiking* имеет среднее восстановление мембраны, относительно остальных.
2. Режим *Phasic Spiking* значительно улучшенное восстановление мембраны по сравнению с предыдущем режимом.
3. Режим *Chattering* имеет самое долгое восстановление мембраны.
4. Режим *Fast Spiking* имеет самое быстрое восстановление мембраны.

# Заключение

Во время выполнения лабораторной работы были выполнены следующие пункты:

1. Изучены и применены численные методы уравнения Коши: *Эйлера, Рунге-Кутта 4-ого порядка и неявный метод Эйлера.*
2. Изучены и применены режимы заданной динамической системы *(Tonic Spiking, Phasic Spiking, Chattering, Fast Spiking)*, и, основываясь на них, были построены графики зависимости потенциала мембраны от времени.
3. По графикам изучены особенности режимов динамической системы.

# Список использованных источников

1. Першин А.Ю. Лекции по курсу «*Вычислительная математика*». Москва, 2018-2021. URL: https://archrk6.bmstu.ru/index.php/f/810046. (облачный сервис кафедры РК6).
2. Соколов, А.П., Першин, А.Ю «*Инструкция по выполнению лабораторных работ (общая)».* Москва: Соколов, А.П., Першин, А.Ю., 2018-2021. URL: *https://arch.rk6.bmstu.ru. (облачный сервис кафедры РК6).*