Вестник Сыктывкарского университета.
Серия 1: Математика. Механика. Информатика. 2024.
Выпуск 2 (51)
Bulletin of Syktyvkar University.
Series 1: Mathematics. Mechanics. Informatics. 2024; 2 (51)

Научная статья

УДК 539.3
https://doi.org/10.34130/1992-2752_2024_2_14

ВИЗУАЛИЗАЦИЯ ЧИСЛЕННЫХ РАСЧЕТОВ СРЕДСТВАМИ PYTHON

Анатолий Альбертович Дуркин,
Андрей Васильевич Ермоленко,
Надежда Олеговна Котелина,
Оксана Игоревна Туркова
Сыктывкарский государственный университет
имени Питирима Сорокина, ea74@list.ru

Аннотация.

Простота синтаксиса и большое количество библиотек языка Python делают его незаменимым инструментом при проведении лабораторных работ по ряду предметов, упрощая рутинные действия при визуализации численных расчетов.

В статье авторы показывают примеры использования библиотек Python для построения поверхностей и интерполяционных кривых, анимации численных методов.

Ключевые слова: Python, визуализация, численный эксперимент, анимация

Для цитирования: Дуркин А. А., Ермоленко А. В., Котелина Н. О., Туркова О. И. Визуализация численных расчетов средствами Python // *Вестник Сыктывкарского университета. Сер. 1: Математика. Механика. Информатика.* 2024. Вып. 2 (51). С. 14–26. https://doi.org/10.34130/1992-2752_2024_2_14

Article

Visualization of Numerical Calculations with Python

Anatolij A. Durkin, Andrey. V. Yermolenko,
Nadezhda O. Kotelina, Oksana. I. Turkova
Pitirim Sorokin Syktyvkar State University, ea74@list.ru

Abstract. The simplicity of the syntax and the large number of Python libraries make it an indispensable tool for conducting laboratory work on a number of subjects, simplifying routine operations when visualizing numerical calculations.

In the article, the authors show examples of using Python libraries for plotting surfaces and interpolation curves, animating numerical methods.

Keywords: Python, visualization, numerical experiment, animation

For citation: Durkin A. A., Yermolenko A. V., Kotelina N. O., Turkova O. I. Visualization of Numerical Calculations with Python. *Vestnik Syktyvkarskogo universiteta. Seriya 1: Matematika. Mekhanika. Informatika* [Bulletin of Syktyvkar University, Series 1: Mathematics. Mechanics. Informatics], 2024, no 2 (51), pp. 14–26. (In Russ.) https://doi.org/10.34130/1992-2752_2024_2_14

1. Введение

Под визуализацией данных понимают их графическое представление для более эффективной работы. Визуализация широко используется при проведении научных и статистических исследований. Это обусловлено, например, тем, что визуальная информация является основной для человека. Так, исследования показывают, что 90 % информации человек воспринимает через зрение [1]. Кроме того, визуализация данных в значительной степени способствует выявлению скрытых в них закономерностей.

В статье [2] отмечается, что основополагающим фактором развития визуализации в представлении результатов расчетов является интенсивное развитие высокопроизводительных и параллельных вычислений. Также авторы статьи утверждают, что «основополагающую роль для исследования, обработки, трактовки и верификации численных результатов будет играть визуализация данных».

Кроме высокопроизводительных вычислений в настоящее время большой упор делается на развитие интерфейсов; создать график, диаграмму, анимацию, чарт–бар — это нередко дело нескольких кликов. При этом создаются специальные языки, например, R, F# [1; 3; 4], ориентированные на визуализацию статистической информации. Данные языки удобны для манипуляции с большими данными, но они являются достаточно узкоспециализированными и обладают достаточно высоким порогом вхождения.

В последнее время широкую популярность получил язык программирования Python [5], который достаточно прост в изучении, но является мощным в реализации визуализации как анализа данных, так и проведения численных расчетов.

Цель данной статьи — обосновать выбор языка программирования Python в качестве первого при обучении студентов кафедры прикладной математики и компьютерных наук основам программирования, поскольку одним из важнейших навыков студентов-математиков является умение графически представить результаты вычислений. Авторы приводят примеры программ, которые показывают соответствующие возможности языка Python.

2. Материалы и методы

Данное исследование построено на базе теоретического анализа научно-технической литературы, посвященной современным инструментам визуализации численных расчетов, возможностям визуализации библиотек языка Python, а также методической литературы по вопросам применения средств визуализации в процессе преподавания математики. Авторами разработаны задания для лабораторных работ, которые апробированы в процессе преподавания дисциплин «Численные методы», «Технологии программирования», «Алгоритмы и алгоритмические языки», «Визуализация численных расчетов», «Анализ данных», «Компьютерная геометрия» студентам направлений подготовки «Прикладная математика и информатика», «Математика и компьютерные науки».

3. Результаты и обсуждение

3.1. Визуализация в численных методах

Интерес к языку Python на кафедре прикладной математики и компьютерных наук появился при необходимости визуализации числен-

ных расчетов. Так до использования языка Python на первом занятии по численным методам разбирали эффективный вывод графика средствами языка C или Pascal, а уже потом переходили непосредственно к изучению предмета. Еще одним способом визуализации было использование системы компьютерной математики Maple [6] путем загрузки готовых расчетов через файл, что замедляло проведение численного эксперимента.

Python же предлагает мощные интегрированные инструменты для визуализации данных, такие как Matplotlib, Seaborn и Plotly. Эти библиотеки позволяют студентам легко создавать различные виды графиков и диаграмм, которые помогают лучше понять и проанализировать полученные результаты. Достаточно определить списки x, y, содержащие значения узлов, а затем ввести код, показанный на листинге 1, чтобы построить график.

Листинг 1

Вывод графика функции по узлам

```
import numpy as np
import matplotlib.pyplot as plt

# Реализуем численный метод

plt.plot(x, y)
plt.grid()
plt.xlabel("x")
plt.ylabel("y")
plt.show()
```

Также легко обучающиеся могут построить графики функций, поверхности решений дифференциальных уравнений, а также визуализировать динамику систем. В листинге 2 показан способ рисования поверхности в Python с использованием библиотеки Matplotlib.

Листинг 2

Построение поверхностей

```
import numpy as np
import matplotlib.pyplot as plt
```

Окончание листинга 2

```
# Реализуем численный метод
...

# Создаем данные для поверхности
X, Y = np.meshgrid(x, y)  # x, y - списки значений по осям x
                          # и y
u = np.array(u)  # u - двумерный массив, результат расчетов

# Создаем фигуру и оси
fig = plt.figure()
ax = fig.add_subplot(1, 1, 1, projection="3d")

# Рисуем поверхность
surf = ax.plot_surface(X, Y, u, cmap="viridis")

# Показываем график
plt.show()
```

В рамках проведения занятий по дисциплине «Технологии программирования» используется анимация графиков библиотеки Matplotlib языка программирования Python. Студентам демонстрируется готовый график, объясняется, как вычислить точки графика при помощи цикла, наглядно показывается построение полученных точек на плоскости при помощи анимации. Существует несколько методов реализации анимации в библиотеке Matplotlib. Рассмотрим примеры построения спирали Архимеда.

Полная перерисовка графика

В этом случае необходимо включить интерактивный режим plt.ion(), чтобы программа могла перерисовывать график, не закрывая окно. Внутри цикла for для получения точек x и y нам необходимо сначала очищать данные графика plt.clf(), чтобы добавлялись данные каждого последующего шага. Для вывода на экран полученных точек используется функция рисования plt.draw() и функция обработки данных каждого шага plt.gcf().canvas.flush_events(). Для того чтобы график отображался на плоскости в динамике, необходима задержка по времени, которую можно установить функцией time.sleep() из библиотеки time.

После этого мы выходим из цикла for. Теперь нам нужно закрыть интерактивный режим и вывести на экран получившийся анимированный график. Стоит отметить, что, если не определить интервалы координатных осей, то до тех пор, пока график не будет построен, оси самостоятельно будут подстраиваться под график на каждом шаге (рис. 1).

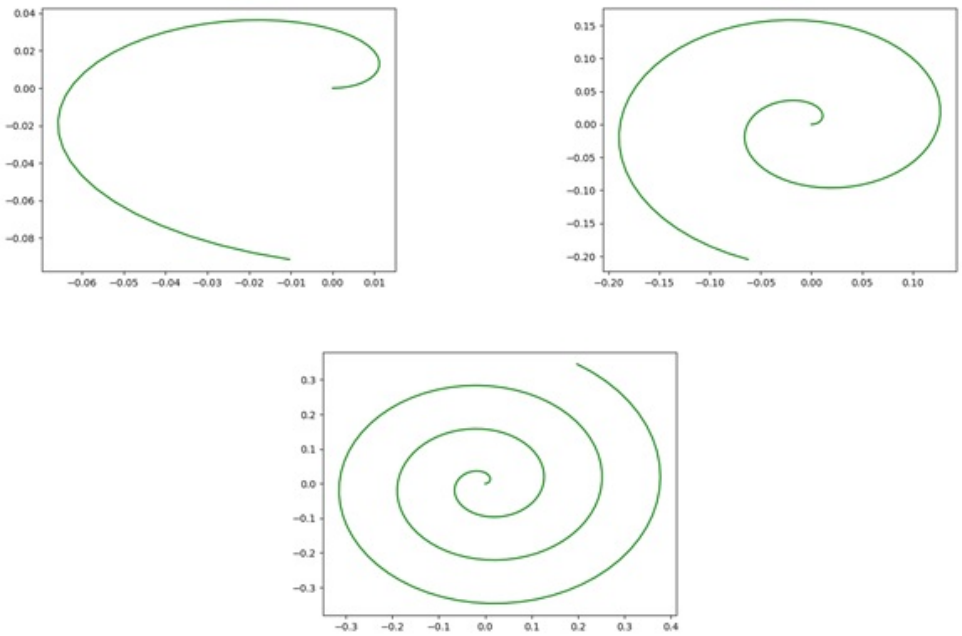

Рис. 1. Спираль Архимеда в разные моменты времени

Данный способ достаточно медленно обрабатывает данные, поэтому чаще графики анимируют при помощи специального класса FuncAnimation.

Анимация графика с помощью вызова класса FuncAnimation библиотеки Matplotlib

Здесь используется функция обновления данных графика, которая передается в класс. Первое, что необходимо сделать, — определить дорисовывающую график функцию, которая зависит от количества обновляемых кадров:

```
def spiral(i):
    # архимедова спираль
    th = i * 0.1
```

```
    r = k * th  # формула спирали в полярных координатах
    x = r * math.cos(th)  # перевод в декартовы координаты - x
    y = r * math.sin(th)  # перевод в декартовы координаты - y

    plt.plot(x, y, "o", color="green")  # рисуем новую точку
```

Далее необходимо определить отображаемую фигуру fig = plt.figure() и вызывать класс анимации из библиотеки, где fig — это фигура, которую строим, spiral — функция обновления данных, frames — изменяющийся параметр от кадра к кадру, interval — интервал, с которым обновляется кадр, repeat — повторение анимации после прорисовки:

```
anim = animation.FuncAnimation(fig, animate, frames=360,
                               interval=10, repeat=False)    ,
```

и показываем получившееся на экран с помощью вызова plt.show(). Также в этом классе можно сохранять полученные анимированные графики в файл функцией anim.save("sp_py10.gif").

В численных методах класс FuncAnimation также оказывается полезен для анимирования расчетов, например, уравнения колебаний струны (см. листинг 3).

Листинг 3

Реализация анимации при помощи модуля animation

```
import matplotlib.pyplot as plt
import matplotlib.animation as animation

# Реализуем численный метод
...

# Функция для создания анимации
def animate(i):
    plt.cla()  # Очищаем график перед каждым обновлением
    plt.ylim(-1, 1)
    plt.grid()
    plt.plot(u[i])  # Рисуем часть графика
```

Окончание листинга 3

```
# Данные для графика
fig = plt.figure()
ax = fig.add_subplot(1, 1, 1)

# Запуск анимации
anim = animation.FuncAnimation(fig, animate, frames=len(u), \
    interval=100)
plt.show()
```

3.2. Решение задач интерполяции при помощи библиотек Python

Во многих практических задачах требуется построить гладкую кривую линию, проходящую через заданные точки [7]. В рамках курсов численных методов и компьютерной геометрии студентам направления «Прикладная математика и информатика» обычно предлагают решать задачу интерполяции и строить графики интерполяционных полиномов Лагранжа, Ньютона («глобальная интерполяция»), или сплайны («локальная интерполяция») [8]. Для проверки корректности построенных кривых можно использовать библиотеку SciPy и модуль scipy.interpolate.

Приведем пример построения кубического сплайна Эрмита, проходящего через заданную последовательность точек и имеющего в этих точках заданные производные. Пусть радиус-векторы этих точек равны p_i, векторы производных радиус-вектора кривой в этих точках равны q_i, а значения параметра в этих точках равны t_i, где $i = 0, 1, 2, ..., n$ — номера точек. На участке между точками p_i и p_{i+1} составной сплайн Эрмита является полиномом третьей степени местного параметра w:

$$r_i(w) = a_0 + a_1 w + a_2 w^2 + a_3 w^3,$$

где $w \in [0, 1]$.

Векторы $a_j, j = 0, 1, 2, 3$ на каждом участке найдем из следующих условий на границе участка кривой:

$$r_i(0) = p_i, r_i(1) = p_{i+1},$$

$$r'_i(0) = q_i, r'_i(1) = q_{i+1}.$$

После решения последней системы уравнений получим радиус-вектор сплайна Эрмита в виде

$$r(t) = p_i(1-3w^2+2w^3)+p_{i+1}(3w^2-2w^3)+q_i(w-2w^2+w^3)+q_{i+1}(-w^2+w^3),$$

где p_i, p_{i+1} — радиус-векторы граничных точек, q_i, q_{i+1} — векторы производных радиус-вектора кривой в этих точках.

За построение сплайна Эрмита в модуле scipy.interpolate отвечает функция CubicHermiteSpline. Пример ее использования приведен в листинге 4.

Листинг 4

Построение интерполяционного сплайна Эрмита

```
import numpy as np
from scipy.interpolate import CubicHermiteSpline
import matplotlib.pyplot as plt

def f(p0, p1, q0, q1, t):
    return p0*(2*t**3 - 3*t**2 + 1 ) + p1*(-2*t**3 + 3*t**2) +\
       q0*(t**3 - 2*t**2 + t) + q1*(t**3-t**2)
def x():
    temp = []
    for i in range(n):
        temp = temp + [f(px[i], px[(i+1) % n],\
            qx[i], qx[(i+1) % n], j/10) for j in range(0, 11)]
    return temp
def y():
    temp = []
    for i in range(n):
        temp = temp + [f(py[i], py[(i+1) % n],\
            qy[i], qy[(i+1) % n], j/10) for j in range(0, 11)]
    return temp
n = 6
p = np.array([[0, 0],[0, 60], [50, 60], [50, 40], [20, 40], \
              [20, 0], [0, 0]], dtype = float)
fig, ax = plt.subplots()

t = np.arange(0, n + 1, 1)
```

Окончание листинга 4

```
q = np.array([(p[(i + 1) % n] - p[(i - 1 + n) % n])/2 \
for i in range(0, n + 1))])

px = [el[0] for el in p]
py = [el[1] for el in p]
qx = [el[0] for el in q]
qy = [el[1] for el in q]

poly = CubicHermiteSpline(t, p, q)
ax.set_aspect("equal")
xx = np.linspace(0, n, 150)
ax.plot(poly(xx)[:,0], poly(xx)[:,1], "b-", lw=1, alpha=0.7, \
        label="Функция SciPy")
ax.scatter(x(), y(),  label="Формула")
ax.axis("off")
legend = ax.legend(loc="lower right",  fontsize="medium")
plt.show()
```

В листинге 4 приведены два способа построения интерполяционного сплайна Эрмита, студенты могут непосредственно реализовать построение сплайна по формулам и использовать готовую функцию для проверки правильности формул. Результат работы показан на рис. 2.

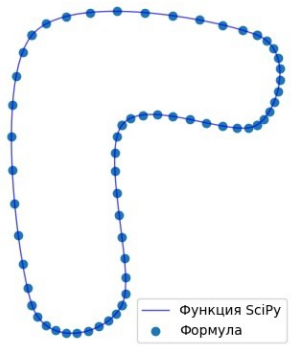

Рис. 2. Пример построения сплайна Эрмита

4. Заключение

Python является отличным выбором для визуализации результатов численных расчетов на занятиях, связанных с численными методами, благодаря своей универсальности и доступности специализированных библиотек, что позволяет студентам легко создавать профессиональные графики и диаграммы, делая процесс обучения более наглядным и интересным.

Представленная работа поможет исследователям, использующим математическое моделирование, обоснованно выбирать средства визуализации численных расчетов.

Список источников

1. **Бакунова О. М., Буркин А. В., Протько Д. Э. и др.** Визуализация данных на .NET F# // *Web of Scholar*. 2018. Т. 1. № 4 (22). С. 19–22.

2. **Бондарев А., Галактионов В.** Современные направления развития визуализации данных в вычислительной механике жидкости и газа // *Научная визуализация.* 2013. Т. 5. № 4. С. 18–30.

3. **Акберова Н. И., Козлова О. С.** Основы анализа данных и программирования в R : учебно-методическое пособие. Казань: Альянс, 2017. 33 с.

4. **Егошин В. Л., Иванов С. В., Саввина Н. В. и др.** Визуализация биомедицинских данных с использованием программной среды R // *Экология человека.* 2018. № 8. С. 52–64.

5. **Ермоленко А. В., Осипов К. С.** О применении библиотек Python для расчета пластин // *Вестник Сыктывкарского университета. Сер. 1: Математика. Механика. Информатика.* 2019. Вып. 4 (33). С. 86–95.

6. **Дьяконов В. П.** Maple 9.5/10 в математике, физике и образовании. М.: СОЛОН-ПРЕСС, 2006. 720 с.

7. **Голованов Н. Н.** Геометрическое моделирование. М.: Издательство Физико-математической литературы, 2002. 472 с.

8. **Piegl L., Tiller, W.** The NURBS Book. Monographs in Visual Communications. New York: Springer, 1995. 646 p.

References

1. **Bakunova O. M., Burkin A. V., Protko D. E. at al.** Data visualisation with .NET F#. *Web of Scholar*. 2018. Vol. 1. No 4 (22). Pp. 19–22. (In Russ.)

2. **Bondarev A., Galaktionov V.** Modern development directions of data visualization in computational mechanics of fluid and gase. *Nauchnaya vizualizatsiya* [Science visualisation]. 2013. Vol. 5. No 4. Pp. 18–30. (In Russ.)

3. **Akberova N. I., Kozlova O. S.** *Osnovy analiza dannyh i programmirovaniya v R : uchebno-metodicheskoe posobie* [Fundamentals of data analysis and programming in R : a textbook]. Kazan': Al'yans, 2017. 33 p. (In Russ.)

4. **Egoshin V. L., Ivanov S. V., Savvina N. V. at al.** Visualization of biomedical data using the R software environment. *Ekologiya cheloveka* [Human ecology]. 2018. No 8. Pp. 52–64. (In Russ.)

5. **Yermolenko A. V., Osipov K. S.** On using Python libraries to calculate plates. *Vestnik Syktyvkarskogo universiteta. Seriya 1: Matematika. Mekhanika. Informatika* [Bulletin of Syktyvkar University. Series 1: Mathematics. Mechanics. Computer science]. 2019. No 4 (33). Pp. 86–95. (In Russ.)

6. **D'yakonov, V. P.** *Maple 9.5/10 v matematike, fizike i obrazovanii* [Maple 9.5/10 in mathematics, physics and education]. Moscow: SOLON-PRESS, 2006. 720 p. (In Russ.)

7. **Golovanov N. N.** *Geomyetricheskoe modelirovaniye* [Geometry modeling]. Moscow: *Izdatyelstvo Phisiko-matematicheskoy litye*ratury [Publishing House of physical and mathematical literature], 2002. 472 p. (In Russ.)

8. **Piegl L., Tiller W.** *The NURBS Book. Monographs in Visual Communications*. New York: Springer, 1995. 646 p.

Сведения об авторах / Information about authors
Дуркин Анатолий Альбертович / Anatolij A. Durkin
аспирант / aspirant
Сыктывкарский государственный университет имени Питирима Сорокина / Pitirim Sorokin Syktyvkar State University
167001, Россия, г. Сыктывкар, Октябрьский пр., 55 / 55, Oktyabrsky Ave., Syktyvkar, 167001, Russia

Ермоленко Андрей Васильевич / Andrei V. Yermolenko
к.ф.-м.н., доцент, заведующий кафедрой прикладной математики и компьютерных наук / Candidate of Science in Physics and Mathematics, Associate Professor, Head of Department of Applied Mathematics and Computer Science
Сыктывкарский государственный университет имени Питирима Сорокина / Pitirim Sorokin Syktyvkar State University
167001, Россия, г. Сыктывкар, Октябрьский пр., 55 / 55, Oktyabrsky Ave., Syktyvkar, 167001, Russia

Котелина Надежда Олеговна / Nadezhda O. Kotelina
к.ф.-м.н., доцент кафеды прикладной математики и компьютерных наук / Candidate of Science in Physics and Mathematics, Associate Professor of Department of Applied Mathematics and Computer Science
Сыктывкарский государственный университет имени Питирима Сорокина / Pitirim Sorokin Syktyvkar State University
167001, Россия, г. Сыктывкар, Октябрьский пр., 55 / 55, Oktyabrsky Ave., Syktyvkar, 167001, Russia

Туркова Оксана Игоревна / Oksana I. Turkova
аспирант / aspirant
Сыктывкарский государственный университет имени Питирима Сорокина / Pitirim Sorokin Syktyvkar State University
167001, Россия, г. Сыктывкар, Октябрьский пр., 55 / 55, Oktyabrsky Ave., Syktyvkar, 167001, Russia

Статья поступила в редакцию / The article was submitted 03.09.2024
Одобрено после рецензирования / Approved after reviewing 12.09.2024
Принято к публикации / Accepted for publication 18.09.2024