Лекция 3. Визуализация данных с помощью Python

А. Н. Лата

МГИМО(У) МИД России

20 марта 2024 г.

Содержание

Библиотека Matplotlib

Matplotlib — это комплексная библиотека для создания статических, анимированных и интерактивных визуализаций на Python. Matplotlib делает простые вещи простыми, а сложные — возможными.

Дополнительные материалы

- Документация по библиотеке Matplotlib
- Шабанов П.А. Научная графика в Python

Библиотека Matplotlib

После установки библиотеки Matplotlib вызывается в консоли или в скрипте как модуль:

```
1 import matplotlib as mpl
2
3 # Вывод на экран текущей версии библиотеки matplotlib
4 print ('Текущая версия matplotlib:', mpl.__version__)
```

Поскольку наиболее простыми для понимания человеком являются самые высокоуровневые функции, то знакомство с Matplotlib начинают с самого высокоуровневого интерфейса matplotlib.pyplot.

Иерархическая структура рисунка в Matplotlib

Главной единицей (объектом самого высокого уровня) при работе с Matplotlib является рисунок (Figure).
Любой рисунок в Matplotlib имеет вложенную структуру и чем-то напоминает матрёшку. Пользовательская работа подразумевает операции с разными уровнями этой матрёшки:

Figure(Рисунок) -> Axes(Область рисования) -> Axis(Координатная ось)

Иерархическая структура рисунка в Matplotlib: Figure

Рисунок (Figure)

Рисунок является объектом самого верхнего уровня, на котором располагаются одна или несколько областей рисования (Axes), элементы рисунка Artisits (заголовки, легенда и т.д.) и основа-холст (Canvas). На рисунке может быть несколько областей рисования Axes, но данная область рисования Axes может принадлежать только одному рисунку Figure.

Иерархическая структура рисунка в Matplotlib: Axes

Область рисования (Axes)

Область рисования является объектом среднего уровня, который является, наверное, главным объектом работы с графикой Matplotlib в объектно-ориентированом стиле. Это то, что ассоциируется со словом «plot», это часть изображения с пространством данных. Каждая область рисования Axes содержит две (или три в случае трёхмерных данных) координатных оси (Axis объектов), которые упорядочивают отображение данных.

Иерархическая структура рисунка в Matplotlib: Axis

Координатная ось (Axis)

Координатная ось являются объектом среднего уровня, которые определяют область изменения данных, на них наносятся деления ticks и подписи к делениям ticklabels. Расположение делений определяется объектом Locator, а подписи делений обрабатывает объект Formatter. Конфигурация координатных осей заключается в комбинировании различных свойств объектов Locator и Formatter.

Иерархическая структура рисунка в Matplotlib: Artists

Элементы рисунка (Artists)

Элементы рисунка Artists являются как бы красной линией для всех иерархических уровней. Практически всё, что отображается на рисунке является элементом рисунка (Artist), даже объекты Figure, Axes и Axis. Элементы рисунка Artists включают в себя такие простые объекты как текст (Text), плоская линия (Line2D), фигура (Patch) и другие.

Когда происходит отображение рисунка (figure rendering), все элементы рисунка Artists наносятся на основу-холст (Canvas). Большая часть из них связывается с областью рисования Axes. Также элемент рисунка не может совместно использоваться несколькими областями Axes или быть перемещён с одной на другую.

Обычный пользователь будет тратить 95% своего времени, работая с Artists!

Интерфейс Pyplot I

```
1 # де-факто стандарт вызова pyplot в python
2 import matplotlib.pyplot as plt
```

Рисунки в Matplotlib создаются путём последовательного вызова команд.

Графические элементы (точки, линии, фигуры и т.д.) наслаиваются одна на другую последовательно.

При этом последующие перекрывают предыдущие, если они занимают общее участки на рисунке (регулируется параметром zorder).

Интерфейс Pyplot II

В Matplotlib работает правило «текущей области» («current axes»), которое означает, что все графические элементы наносятся на текущую область рисования. Несмотря на то, что областей рисования может быть несколько, одна из них всегда является текущей.

Как было сказано выше самым главным объектом в Matplotlib является рисунок — Figure. Поэтому создание графики нужно начинать именно с создания рисунка. Создать рисунок в Matplotlib означает задать форму, размеры и свойства основы-холста (canvas), на котором будет создаваться будущий график.

Интерфейс Pyplot III

Создать рисунок figure позволяет метод plt.figure(). После вызова любой графической команды, то есть функции, которая создаёт какой-либо графический объект, например, plt.scatter() или plt.plot(), всегда существует хотя бы одна область для рисования (по умолчанию прямоугольной формы).

Чтобы результат рисования, то есть текущее состояние рисунка, отразилось на экране, можно воспользоваться командой plt.show(). Будут показаны все рисунки (figures), которые были созданы.

Интерфейс Pyplot IV

Чтобы сохранить получившийся рисунок нужно воспользоваться методом plt.savefig(). Он сохраняет текущую конфигурацию текущего рисунка в графический файл с некоторым расширением (png, jpeg, pdf и др.), который можно задать через параметр fmt. Поэтому её нужно вызывать в конце исходного кода, после всех вызова всех других команд. Если в python-скрипте создать несколько рисунков figure и попытаться сохранить их одной командой plt.savefig(), то будет сохранён последний рисунок figure.

Содержание

Графические команды I

Графические команды — это функции, которые, принимая некоторые параметры, возвращают какой-то графический результат. Это может быть текст, линия, график, диаграмма и др.

Подробнее о конкретных типах графиков смотри в электронных ресурсах

- Галерея примеров различной графики в Matplotlib
- Полный список команд для pyplot
- Matplotlib. Уроки
- Книга «Библиотека Matplotlib»

Графические команды II

В Matplotlib заложены как простые графические команды, так и достаточно сложные. Доступ к ним через pyplot означает использование синтаксиса вида «plt.название_команды()».

Самые простые графические команды

- plt.scatter() - маркер или точечное рисование;
- plt.plot() - ломаная линия;
- plt.text() - нанесение текста;

Графические команды III

Диаграммы

- plt.bar() - столбчатая диаграмма;
- plt.hist() - гистограмма;
- plt.pie() - круговая диаграмма;
- plt.boxplot() - «ящик с усами» (boxwhisker)
- plt.errorbar() - оценка погрешности, «усы».

Изображения в изолиниях (линиях уровня)

- plt.contour() - изолинии (линии уровня);
- plt.contourf() - изолинии с послойной окраской.

Графические команды IV

Отображения

- plt.imshow() - вставка графики (пиксели + сглаживание);
- plt.matshow() - отображение данных в виде квадратов.

Заливка

- plt.fill() - заливка многоугольника;
- plt.fill_between(), plt.fill_betweenx() - заливка между двумя линиями;

matplotlib.patches

Создание геометрически сложных фигур является достаточно специальной задачей, поэтому matplotlib.patches рассматриваться здесь не будут.

Содержание

Работа с текстом и шрифтами

Текст является одним из базовых графических элементов рисунка в Matplotlib. Подписи координатных осей и их делений, заголовки, пояснительные подписи на графиках и диаграммах - это всё текст. В Matplotlib возможна поддержка кириллицы для создания научной графики с подписями на русском языке. Одним из весомых преимуществ matplotlib при работе с текстом является простая поддержка математических формул с помощью LaTeX.

Электронные ресурсы

- Инструкция по конфигурации файла настройки matplotlibrc
- Об особенностях использования LaTeXв matplotlib
- О работе с текстом в matplotlib

Файл настройки matplotlibrc I

В matplotlib существует файл настройки в котором хранятся значения по умолчанию для разных свойств графических элементов. Он называется matplotlibrc. Он инициализируется при каждой загрузке модуля matplotlib. Изменив содержание файла matplotlibrc можно сохранить пользовательские настройки для работы при следующих загрузках модуля matplotlib. Чтобы узнать какой именно файл настройки используется при инициализации, можно воспользоваться специальной функцией matplotlib.matplotlib_fname().

Файл настройки matplotlibrc II

Для работы с настройками в интерактивном режиме, то есть из консоли или в скрипте, существует matplotlib.rcParams. С помощью matplotlib.rcParams можно, во-первых, установить все параметры рисования (шрифты, толщина/тип линий, оформление рамок) на необходимые значения, а во-вторых, увидеть, собственно, список настроек, то есть список того, что можно изменить.

Файл настройки matplotlibrc III

```
1  import matplotlib as mpl
2
3  # показывает откуда был загружен текуший файл matplotlibrc
4  print(mpl.matplotlib_fname())
5
6  # показывает текущие настройки
7  print(mpl.rcParams)
8
9  # показывает значения параметров, которые хранятся в файле
       matplotlibrc
10 #mpl.rcParamsDefault
11 # сбрасывает значения на те, которые хранятся в файле
       matplotlibrc
12 #mpl.rcdefaults()
```

Текст на рисунке I

Самой простой командой, отображающие текст, не привязаной к какому-либо объекту вроде координатной оси или делений координатной оси, является команда plt.text(). В качестве входящих данных она принимает координаты положения будующей строки и сам текст в виде строки.

Текст на рисунке II

Методы отображения текста в pyplot

- plt.xlabel() - добавляет подпись оси абсции OX
- plt.ylabel() - добавляет подпись оси ординат OY
- plt.title() - добавляет заголовок для области рисования Axes
- plt.figtext() - добавляет текст на рисунок Figure
- plt.suptitle() - добавляет заголовок для рисунка Figure
- plt.annotate() - добавляет примечание, которое состоит из текста и необязательной стрелки в указанную область на рисунке