Практикум 2. Построение графиков функций одной переменной

Цель работы – обучение построению графика функций одной переменной в декартовой системе координат, построению нескольких графиков в одной системе координат, созданию нескольких рисунков в одном графическом окне.

Продолжительность работы - 2 часа.

Оборудование, приборы, инструментарий – работа выполняется в компьютерном классе с использованием пакета Anaconda.

Порядок выполнения

1. Упражнения выполняются параллельно с изучением теоретического материала.
2. После выполнения каждого упражнения результаты заносятся в отчёт.
3. При выполнении упражнений в случае появления сообщения об ошибке рекомендуется сначала самостоятельно выяснить, чем оно вызвано, и исправить команду; если многократные попытки устранить ошибку не привели к успеху, то проконсультироваться с преподавателем.
4. Дома доделать упражнения из раздела «Краткие теоретические сведения и практические упражнения», которые Вы не успели выполнить во время аудиторного занятия.
5. После выполнения упражнений выполнить дополнительные упражнения для самостоятельной работы и ответить на контрольные вопросы и (см. ниже).
6. Подготовить **отчёт,** в который включить упражнения из раздела «Краткие теоретические сведения и практические упражнения» и упражнения для самостоятельной работы. Отчёт представить либо в формате интерактивного питон-скрипта (ipynb), либо в виде документа Microsoft Word. Файл следует назвать по следующей схеме **pin_10_Ivanov_P_01_s1** (группа, фамилия, инициалы, номер лабораторной, семестр). Отчет должен содержать по каждому выполненному упражнению: № упражнения, текст упражнения; команды, скопированные из командного окна, с комментариями к ним и результаты их выполнения, включая построенные графики; тексты python фукнций; выводы.

Краткие теоретические сведения

и практические упражнения

1. Построение графика в декартовых координатах

Для того чтобы построить график функции $y = f(x)$, достаточно тем или иным способом сформировать два вектора одинаковой размерности – вектор значений аргумента x и вектор соответствующих значений функции y (например с помощью средств пакета numpy) и обратится к функции `plt.plot` из пакета `matplotlib.pyplot` (подробное описание функции **>> help (plt.plot)**).

Пример 1.

```
import matplotlib.pyplot as plt
import numpy as np
x = np.linspace(-2,2,1000)
y = np.exp(x)
plt.plot(x,y)
plt.show()
```

В этом примере для создания массива x была использована функция **linspace**, которая в нашем случае сделает массив из значений от -2 до 2, разбив этот отрезок на 1000 равных частей. Собственно график функции получен путем соединения смежных точек таблицы x, y отрезками прямых. Чем меньше точек было создано в таблице отображаемой функции, тем заметнее была бы кусочно-линейная структура графика (попробуйте взять 10 точек).

Будет лучше, если мы снабдим график заголовком, подпишем оси, нанесем координатную сетку (текст в примере 1, начинающийся с #, является комментарием, его можно не набивать).

Пример 1 (продолжение)

```
import matplotlib.pyplot as plt
import numpy as np
x = np.linspace(-2,2,1000)
y = np.exp(x)
plt.plot(x,y)
plt.xlabel('x')
plt.ylabel('y')
plt.title('function $y = e^{x}$') # название для графика
plt.show()
```

Если обратиться к функции **plt.plot** повторно, то новый график будет отображен в текущем графическом окне вместе со старым графиком после подачи команды **plt.show**. Отметим, что добавить название для графика с помощью **plt.title** можно только в той же ячейке, где мы его построили, и только перед подачей команды на его отображение **plt.show**.

Для того чтобы создать новое окно и разместить в нем следующий график необходимо перед обращением к функции **plt.plot** вызвать функцию **plt.figure**. Затем ввести несколько графических подобластей с помощью `add_subplot`.

```
import matplotlib.pyplot as plt
import numpy as np
fig1 = plt.figure()                 # графическое окно figure 1
ax1 = fig1.add_subplot(1, 1, 1)     # добавляем график 1 на figure 1
plt.title('function $y = e^{x}$')
ax1.plot(x,y)                       # строим функцию на графике
fig1.show()

fig2 = plt.figure()                 # графическое окно figure 2
ax2 = fig2.add_subplot(1, 1, 1)
plt.title('function $y = ln(x)$')
ax2.plot(y,x) # поменяем местами x и y, построим обратную функцию x(y)
fig2.show()
```

В этом примере мы воспользовались функцией `add_subplot(n,k,cur)` для добавления новых графиков на графическое окно. Первые два аргумента задают количество рядов (n) и колонок (k) для разбиения области графического окна на отдельные подобласти, а третий параметр (cur) объявляет порядковый номер подобласти, в котором очередная функция **plot** будет стоить свой график.

Пользователь может повлиять на цвет графика, указав в качестве третьего параметра функции **plot** один из приведенных в табл. 1 символов (символ надо заключить в апостроф).

Таблица 1. Обозначение цвета графика

Символ цвета	Цвет графика	Символ цвета	Цвет графика
y	желтый	g	зеленый
m	малиновый	b	синий
c	голубой	w	белый

r	красный	k	черный

Также можно задать стиль линии и форму маркера, которым ставятся табличные точки. Некоторые из управляющих символов, определяющих стиль линии и форму маркера, приведены в табл. 2 и 3 (см. также Л. 1 стр. 111). Они задаются в строке третьего параметра функции **plot** вместе с символом цвета. Порядок следования символов – любой (без пробелов).

Таблица 2.Обозначение формы маркера

Символ	Тип маркера
.	жирная точка
o	круг
x	крестик
+	плюс
*	снежинка
s	квадрат
d	ромб
p, h	звезды (5-,6-ти конечные)
^, <, > v	треугольники

Таблица 3 Обозначение стиля линии

Символ	Форма
-	сплошная
:	пунктирная
-.	штрих-пунктирная
--	штриховая

Пример 2.

```
import matplotlib.pyplot as plt
import numpy as np
x = np.linspace(-3,3,10)
y = np.exp(x)
plt.plot(x,y,':ro')
plt.show()
```

Задания к упражнениям находятся в файле

Упражнение 1.

Построить графики функций, дать каждому из графиков заголовок, подписать оси, нанести координатную сетку, задать цвет графика, тип линии и форму маркера.

2. Построение нескольких графиков в одной системе координат

Упражнение 2.

В одной системе координат построить графики функций, подписать оси, нанести координатную сетку, для каждого графика задать цвет, тип линии и форму маркера:

$y = \sin x, y = \sin x - 2, y = \sin x + 1$.

Функция `plt.legend` позволяет нам вывести в графическом окне область с легендой-справкой о названиях графиков. Чтобы мы смогли воспользоваться этой функцией нам вначале нужно будет передать эти названия всем функциям plot по следующей схеме `plt.plot(...,label='название')`. Затем нам потребуется сохранить ссылку на графический объект, которую возвращает функция `plt.plot`, и передать эти ссылки `plt.legend`.

Пример 3.

```
import matplotlib.pyplot as plt
import numpy as np
x=np.linspace(-2.0*np.pi,2.0*np.pi,100)
handle1, = plt.plot(x,np.cos(x),label='y=cos(x)')
handle2, = plt.plot(x,np.cos(2.0*x),label='y=cos(2x)')
handle3, = plt.plot(x,np.cos(0.5*x),label='y=cos(x/2)')
plt.legend(handles=[handle1,handle2,handle3])
plt.axhline(y=0, color='k') # отобразим ось x
plt.axvline(x=0, color='k') # отобразим ось y
plt.show()
```

Упражнение 3.

Используя команду `plt.legend`, в одной системе координат построить графики функций, подписать оси, для каждого графика задать цвет, тип линии и форму маркера и сделать для них общую легенду-справку:

$y = \cos x, y = 2\cos x, y = 0,3\cos x, y = -\cos x$ на промежутке $[-2\pi; 2\pi]$.

3. Несколько рисунков в одном окне

Чтобы в одном графическом окне создать несколько отдельных рисунков, необходимо прибегнуть к функции `subplot`, которая позволяет разбить графическое окно на несколько прямоугольных областей равного размера, расположенных подобно элементам матрицы: `subplot(n,k,cur)`. Данная функция

полностью аналогична рассмотренной ранее `add_subplot(n,k,cur)`, и фактически является её упрощенным вариантом (работает с одним графическим окном).

Пример 4.

```
import matplotlib.pyplot as plt
import numpy as np
x=np.linspace(-2.0*np.pi,2.0*np.pi,100)
plt.subplot(2,1,1)
plt.plot(x,np.sin(x))
plt.grid(True) # включаем отображение координатной сетки
plt.subplot(2,1,2)
plt.plot(x,np.cos(x),'r')
plt.grid(True) # включаем отображение координатной сетки
```

В рассмотренном примере мы также воспользовались функцией `plt.grid`, которая позволяет управлять отображением координатной сетки на графиках.

Упражнение 4. Преобразование графиков функций

1) Используя команду subplot, в одном графическом окне создать 6 подобластей (2×3), в первой из них построить график функции $y=f(x)$ на промежутке $[-5;5]$, где $f(x)=\left|\left|x\right|-2\right|$, в остальных областях на том же промежутке построить графики функций $y=f(x-2), y=f(x+2), y=f(2x), y=f(0,5x), y=f(-x)$.

В отчет добавить комментарии о том, какими преобразованиями каждый из графиков получается из графика функции $f(x)$.

5. Построение графика функции с использованием логарифмической шкалы

При проведении технических расчетов (например, в физике или электротехнике) часто возникает необходимость отображения данных с очень большим или сильно неравномерным разбросом значений. Для этих целей пользуются так называемой *логарифмической шкалой.* В этом случае равным отрезкам на оси соответствуют равные относительные приращения показателя, а не равные абсолютные приращения как на линейной шкале.

Для построения таких графиков в python используется функция **loglog** (подробное описание функции **>> help (plt.loglog)**).

Также python позволяет отобразить график с использованием линейного масштаба по одной координатной оси и логарифмического масштаба по второй оси (для обозначения таких графиков используют термин *полулогарифмическая шкала*). Для этого используются функции **semilogx** и **semilogy**.

Упражнение 6.

Задайте массив значений переменной x = 0.001, 0.01, 0.1, 1, 10, 100, 1000.

Подсказка: Для удобства задания массива можно воспользоваться схемой:

```
import matplotlib.pyplot as plt
import numpy as np
x = np.arange(-3,4,1) # массив степеней
z = np.ones((1,7))*10 # массив вида [10 10 ... 10]
y = np.power(z,x) # поэлементное возведение в степень
```

Построим в той же ячейке график в линейной шкале и полулогарифмической:

```
plt.subplot(2,1,1)
plt.plot(x,y)
plt.subplot(2,1,2)
plt.semilogy(x,y)
plt.show()
```

Упражнение 7.

В одной системе координат построить графики функций $y_1 = \cos x$, $x \in [0;\pi]$ и график обратной функции. Первый график построить пунктирной линией зеленого цвета, второй – штрих-пунктирной линией красного цвета. Отобразить оси в виде сплошных линий черного цвета. Масштаб по осям сделать одинаковым с помощью команд `plt.axis('equal')`. Подписать оси, нанести координатную сетку. Построить прямую, относительно которой графики симметричны (в виде сплошной линии синего цвета). Вывести заголовок.

Задания для самостоятельной работы

1. Выполнить упражнения из раздела «Краткие теоретические сведения и практические упражнения», которые не успели сделать в аудитории.
2. Самостоятельно выполнить упражнения:

Упражнение С1.

Построить графики функций, дать каждому из графиков заголовок, подписать оси, нанести координатную сетку, задать цвет графика, тип линии и форму маркера:

а) $y=\sqrt{x+3}$, б) $y=\text{sign}(x)$.

Упражнение С2.

В одной системе координат построить графики функций, подписать оси, нанести координатную сетку, для каждого графика задать цвет, тип линии и форму маркера:

$y=e^x$ на промежутке $[-2;2]$ и $y=\ln x$ на промежутке $[e^{-2};e^2]$, $y=x$ на промежутке $\left[-2;e^2\right]$

Упражнение С3

В одном графическом окне создать 2 подобласти. В каждой из них постройте на одном графике функции $y=x$, $y=\frac{1}{x}$ и $y=\frac{1}{\sqrt{x}}$. При этом в первой подобласти графики должны быть отображены с использованием линейной шкалы, во второй подобласти с использованием логарифмической шкалы. Для каждой линии задайте свой цвет и форму маркера.

Упражнение С4.

В одной системе координат построить графики функций $y=\sin x$, $x\in[-\pi/2;\pi/2]$, также график обратной функции. Первый график построить сплошной линией голубого цвета, второй –пунктирной линией красного цвета. Подписать оси, нанести координатную сетку. Построить прямую, относительно которой графики симметричны (в виде сплошной линии фиолетового цвета).

3. Ответить на контрольные вопросы:

1) С помощью каких команд можно построить график функции на заданном промежутке?
2) Как построить несколько графиков в разных графических окнах?
3) Как построить несколько графиков в одном графическом окне?
4) Как в одном графическом окне построить несколько графиков в различных подобластях?
5) Как сделать заголовок и подписать оси?