УДК 004.43 + 519.62 DOI: 10.24028/gzh.0203-3100.v41i2.2019.164458

Jupyter Notebook: система интерактивных научных вычислений

А. И. Якимчик, 2019

Институт геофизики им. С. И. Субботина НАН Украины, Киев, Украина
Поступила 17 декабря 2018 г.

Jupyter Notebook — веб-додаток, що дає змогу писати і постачати коментарями код на мові Python в інтерактивному режимі, а також спосіб експериментувати, досліджувати і ділитися своїми результатами з іншими. Все частіше багато дослідників використовує у своїх роботах зазначене обчислювальне середовище. У короткій формі висвітлено основні причини зростаючої популярності мови програмування Python і проекту Jupyter. Головні серед них: висока швидкість розробки і якість програмного забезпечення; стандартна бібліотека, а також бібліотеки з відкритим вихідним кодом NumPy, SciPy, Matplotlib та ін.; простота інтеграції з кодом на C, C++ і FORTRAN; вільне поширення; підтримка і величезне співтовариство розробників і користувачів. За даними компанії TIOBE, яка щомісяця збирає статистику пошукових запитів і на підставі отриманих даних складає власні візуалізовані рейтинги мов програмування, Python займає 3-тє місце за популярністю серед мов програмування. Мову Python було обрано мовою року в 2007, 2010 і 2018 рр. Розглянуто аспекти установки програм, бібліотек і пакетів в операційній системі Windows. Рекомендується завантажувати та встановлювати бібліотеки зі сховища whl-файлів на вебсторінці Крістофа Голка з лабораторії динаміки флуоресценції Каліфорнійського університету. Формат WHL підтримується всіма основними платформами (Mac OS X, Linux, Windows). Детально описано процес запуску сервера блокнотів Jupyter з командного рядка. Продемонстровано простоту та ефективність наукових обчислень в Jupyter Notebook. Наведено тестові розрахунки розв'язання задач лінійної алгебри. Показано, що код обчислення матриці розміром 5000×5000 займає всього кілька рядків.

Ключові слова: програмне забезпечення з відкритим вихідним кодом, мови програмування, лінійна алгебра, матриця, тестові розрахунки.

Введение. За последние несколько десятилетий язык программирования Python превратился в первоклассный инструмент для научных вычислений, включая анализ и визуализацию больших наборов данных. Это может удивить давних поклонников Python: сам по себе этот язык не был создан в расчете на анализ данных или научные вычисления. В 2014 г. Фернандо Перес (Fernando Perez) анонсировал дополнительный уникальный проект от IPython под названием Project Jupyter (*https://speakerdeck.com/fperez/project-jupyter*). Он похож на интерфейс блокнота других программ, таких как Maple, Mathematica и SageMath, стиль вычислительного интерфейса которых появился в Mathematica в 1980-х годах. Jupyter Notebook представляет собой своеобразный блокнот (текстовый редактор) для браузера (рис. 1), удобный для разработки, совместной работы и использования ресурсов, а также для публикации научных результатов. Этот инструмент может использоваться не только с Python, но и с другими языками программирования: Julia, R, Haskell и Ruby. Он часто используется для работы с данными, статистическим моделированием и машинным обучением. Все чаще многие исследователи используют в своих работах вычислительную среду Jupyter [Tauxe et al., 2016; Chudnov, 2016; Braun et al., 2017; Yang-

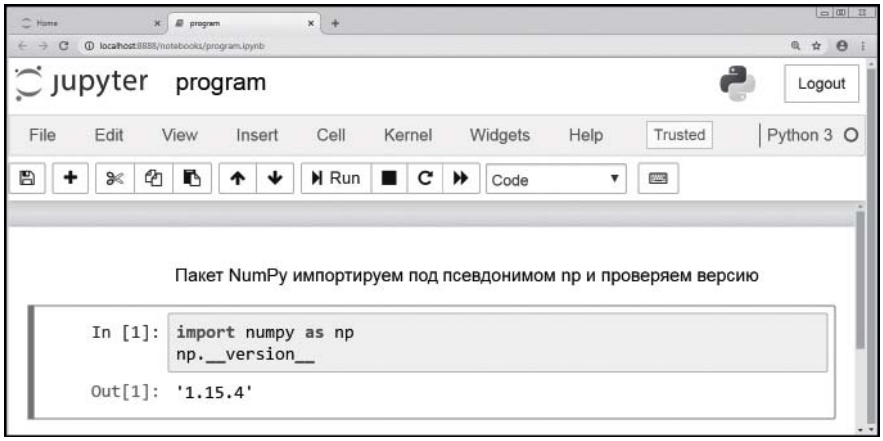

Рис. 1. Страница блокнота Jupyter.

Min et al., 2018]. Узнать больше о Jupyter Notebook можно на сайте jupyter (*https://jupyter.org*).

Язык программирования Python интересен для студентов, разработчиков или исследователей, в основном благодаря большой и активно развивающейся экосистеме пакетов, созданных сторонними разработчиками:

- библиотеки NumPy — для работы с однородными данными в виде массивов (*http://www.numpy.org*);
- библиотеки Pandas — для работы с неоднородными и поименованными данными (*http://pandas.pydata.org*);
- SciPy — для общих научных вычислительных задач (*https://docs.scipy.org/doc*);
- библиотеки Matplotlib — для визуализаций типографского качества (*https://matplotlib.org*);
- оболочки Jupyter — для интерактивного выполнения и совместного использования кода (*https://jupyter.org*);
- библиотеки Scikit-Learn — для машинного обучения и множества других инструментов (*https://scikit-learn.org/stable*).

Python — стабильный и распространенный язык. Он используется во многих проектах и в различных качествах: как основной язык программирования или для создания расширений и интеграции приложений. На Python реализовано большое количество проектов, он активно используется для создания прототипов будущих программ, а также во многих крупных компаниях для научных вычислений: Dropbox, Google (например, некоторые части Youtube и Youtube API написаны на Python), Facebook, Instagram, NASA, Los Alamos, Fermilab, JPL. Грег Стейн в марте 2005 г., выступая на собрании SDForum, сообщил, что в компании Google язык Python является одним из трех «официальных» языков наряду с C++ и Java. Это говорит о многом.

Рейтинг языков программирования TIOBE. Очевидно, что необходимо изучать языки программирования, которые востребованы. В связи с этим сошлюсь на индекс[1] такой компании, как TIOBE (*https://www.tiobe.com*). Она ежемесячно собирает статистику поисковых запросов и на основе полученных данных составляет собственные визуализированные рей-

[1] Индекс TIOBE (*TIOBE programming community index*) – индекс, оценивающий популярность языков программирования на основе подсчета результатов поисковых запросов, содержащих название языка. Для формирования индекса используется поиск в нескольких наиболее посещаемых порталах: Google, Blogger, Wikipedia, YouTube, Baidu, Yahoo!, Bing, Amazon. Текущая информация предоставляется бесплатно, но статистика за длительные периоды доступна только за плату. Авторы индекса считают, что он может быть полезен при принятии стратегических решений. По заявлениям создателей, индекс не ранжирует языки по качеству или количеству написанного кода. Проект подразумевает, что может существовать корреляция между количеством найденных страниц и количеством инженеров, курсов и вакансий.

Dec 2018	Dec 2017	Change	Programming Language	Ratings	Change
1	1		Java	15.932%	+2.66%
2	2		C	14.282%	+4.12%
3	**4**	^	**Python**	**8.376%**	**+4.60%**
4	3	v	C++	7.562%	+2.84%
5	7	^	Visual Basic .NET	7.127%	+4.66%
6	5	v	C#	3.455%	+0.63%
7	6	v	JavaScript	3.063%	+0.59%
8	9	^	PHP	2.442%	+0.85%
9	-	^^	SQL	2.184%	+2.18%
10	12	^	Objective-C	1.477%	-0.02%
11	16	^^	Delphi/Object Pascal	1.396%	+0.00%
12	13	^	Assembly language	1.371%	-0.10%
13	10	v	MATLAB	1.283%	-0.29%
14	11	v	Swift	1.220%	-0.35%
15	17	^	Go	1.189%	-0.20%
16	8	vv	R	1.111%	-0.80%
17	15	v	Ruby	1.109%	-0.32%
18	14	vv	Perl	1.013%	-0.42%
19	20	^	Visual Basic	0.979%	-0.37%
20	19	v	PL/SQL	0.844%	-0.52%

Рис. 2. Топ 20 языков программирования.

тинги языков программирования в виде следующей таблицы (рис. 2).

Как видно из рисунка, Python сейчас занимает третье место в рейтинге языков программирования TIOBE. Отметим, что авторами TIOBE Python был выбран языком года (*Programming Language of the Year*) в 2007, 2010 и 2018 гг.

Интересно посмотреть рейтинг языка Python по годам (рис. 3). Первый пик популярности приходится на начало 2004 г., что связано, скорее всего, с выходом второй версии. Далее рейтинг — нестабилен. Он то уменьшается, то увеличивается, но глобальный тренд в целом восходящий, и на декабрь 2018 г. рейтинг составляет 8,376 %.

Основные причины высокой популярности рассматриваемого языка следующие:

- качество программного обеспечения;
- простота и высокая скорость разработки;
- переносимость программ;
- библиотеки поддержки;
- интеграция компонентов.

Из недостатков выделим один. Это скорость выполнения программ, которая не всегда может быть такой же высокой, как у программ, написанных на компилирующих языках программирования, таких как C или C++.

Существенной сильной стороной языка является то, что Python может использоваться и распространяться совершенно бесплатно. Как и в случае с другими от-

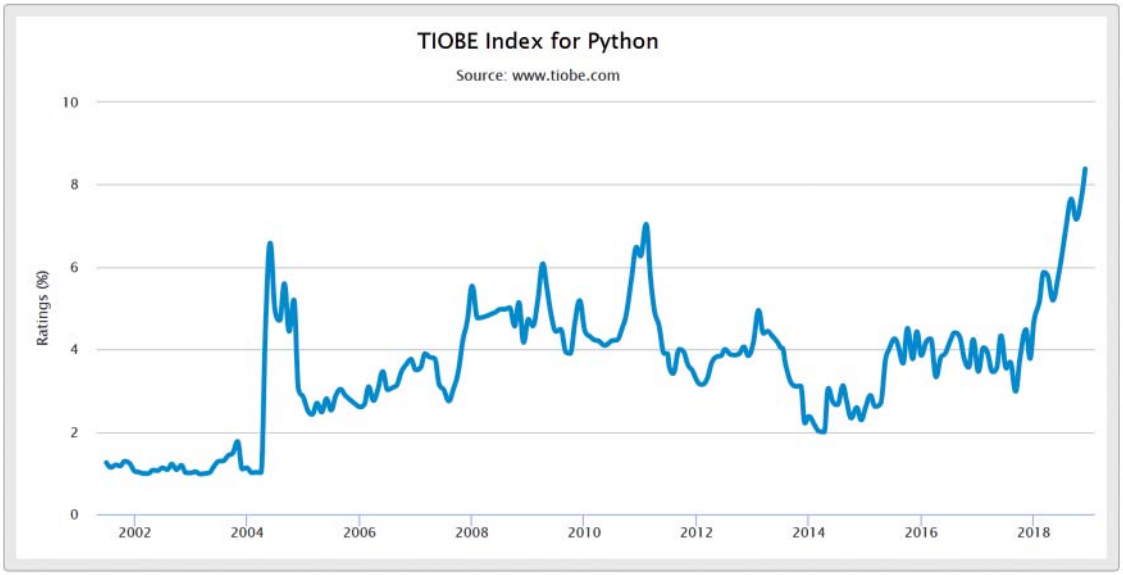

Рис. 3. Индекс TIOBE для Python.

крытыми программными продуктами, можно получить в Интернете полные исходные тексты реализации Python. Нет никаких ограничений на его копирование, встраивание в свои системы или распространение в составе ваших продуктов. Но «свободный» не означает «не поддерживается». Напротив, сообщество сторонников Python в Интернете отвечает на вопросы пользователей со скоростью, которой могло бы позавидовать большинство разработчиков коммерческих продуктов. Кроме того, свободное распространение исходных текстов Python способствует расширению команды экспертов по реализации. И хотя предоставляемая возможность изучать или изменять реализацию языка программирования не у всех вызывает восторг, тем не менее наличие последней инстанции в виде исходных текстов придает уверенность. Вы уже не зависите от прихотей коммерческого производителя — в вашем распоряжении находится полный комплект исчерпывающей документации.

Важным аспектом является то, что Python прост в изучении. Дефицит пособий по языку Python в мире отсутствует. Одними из лучших, по моему мнению, являются книги Марка Саммерфилда, Билла Любановича и Марка Лутца [Саммерфилд, 2009; Любанович, 2016; Лутц, 2011]. И не секрет, что у Python найдется библиотека практически для всего, что бы вы не захотели реализовать.

Вопросы установки. Весь материал настоящей статьи протестирован в среде Windows 7. При тестировании исходного кода за основу взят Python версии 3.7.1 (дата выхода — 20.10.2018 г.). Дистрибутивный комплект программной среды Python можно найти на официальном сайте (*https://www.python.org*). Этот комплект в версии для Windows представляет собой обычный пакет формата Microsoft Installer (MSI) и доступен как в 32-, так и в 64-разрядной редакциях. Установка Python не представляет особой сложности и выполняется точно так же, как и установка любого другого Windows-приложения. Есть лишь пара существенных моментов, на которые необходимо обратить внимание. Они касаются базового набора библиотек для разработчика, необходимых для дальнейшей работы.

Библиотеки для Python можно разрабатывать не только на чистом Python. Довольно часто библиотеки пишутся на C (динамические библиотеки), и для них пишется обертка на Python. Или же биб-

лиотека пишется на Python, а для оптимизации узких мест часть кода пишется на C. Такие библиотеки получаются очень быстрыми, однако с вкраплениями кода на C их программисту на Python тяжелее установить ввиду банального отсутствия соответствующих знаний либо необходимых компонентов и настроек в рабочей среде (в особенности в Windows). Для решения описанных проблем разработан специальный формат (файлы с расширением whl) для распространения библиотек, который содержит заранее скомпилированную версию библиотеки со всеми ее зависимостями. Формат WHL поддерживается всеми основными платформами (Mac OS X, Linux, Windows).

В обычных условиях библиотеки Python можно скачать и установить из каталога библиотек Python PyPi (*https://pypi.org*) с помощью менеджера пакетов pip. Однако следует учесть, что в операционной системе Windows для работы некоторых библиотек, в частности SciPy, Scikit-learn и Scikit-image, требуется, чтобы в системе была установлена библиотека Numpy+MKL. Библиотека Numpy+MKL привязана к библиотеке Intel® Math Kernel Library и включает в свой состав необходимые динамические библиотеки (DLL) в каталоге numpy.core. Библиотеку Numpy+MKL следует скачать из хранилища whl-файлов на веб-странице Кристофа Голка из лаборатории динамики флуоресценции Калифорнийского университета в г. Ирвайн (*https://www.lfd.uci.edu/~gohlke/pythonlibs*) и установить с помощью менеджера пакетов pip как whl (соответствующая процедура установки пакетов в формате whl описана ниже). Например, для 64-разрядной операционной системы Windows и среды Python 3.7.1 команда будет такой:

pip install numpy-1.15.4+mkl-cp37-cp37m-win_amd64.whl.

Такой режим установки также касается библиотек scipy, scikit-image и scikit-learn. Стоит отметить, что эти особенности установки не относятся к операционным системам Linux и Mac.

Запуск сервера записных книжек Jupyter. Для установки[2] Jupyter в операционной системе Windows необходимо набрать и выполнить в командной строке

pip install jupyter,

а для обновления —

pip install -- upgrade jupyter.

Далее создаем папку для работы. Назовем ее, например, projects. Создадим в ней текстовый файл с расширением cmd и назовем его start.cmd (рис. 4). Содержимое этого файла состоит из одной строчки: jupyter-notebook. Дважды щелкнув по файлу, запустим программу Jupyter Notebook: создастся локальный веб-сервер, прослушивающий сетевой порт с номером 8888, далее автоматически на странице *http://localhost:8888/tree* откроется веб-браузер. Заметим, что окно командной строки закрывать нельзя (просто сворачиваем). По умолчанию браузер показывает ту папку, в которой мы находились, т. е. projects. На уровень выше подняться не получится, но можно создавать подпапки: New→Folder.

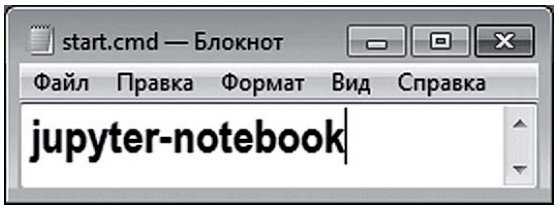

Рис. 4. Файл запуска.

Создадим новый блокнот для дальнейшего запуска программ на языке Python 3: New→Python 3. Откроется окно, похожее на то, что показано на рис. 1. Описывать здесь все, что можно делать в этой программе, нет необходимости. Все интуитивно понятно и имеется встроенная документация. Дело в том, что объем интересной информации, приемов, советов и инструкций по работе с Jupyter Notebook соответ-

[2] Альтернатива — установить дистрибутив Anaconda, в состав которого входит Jupyter Notebook. Anaconda содержит более 400 самых популярных библиотек Python для вычислений в области естественных наук, математики, инженерии и анализа данных.

ствует почти безграничным возможностям языка Python, помноженным на коллективный разум огромного сообщества разработчиков и пользователей Jupyter. Это делает нереальным и неэффективным описание их в какой-либо одной статье или справочнике.

Тестовые расчеты. Эффективность использования оболочки Jupyter продемонстрируем на решении нескольких задач линейной алгебры. Это связано исключительно с тем, что автору данной статьи близка эта проблематика [Черная, Якимчик, 2005а,б; Якимчик, Чорна, 2006; Якимчик, Черная, 2007].

Каждый, кому приходится иметь дело с реализацией численных методов на компьютере, хорошо знает, насколько сложно отладить составленную программу. Труд, затрачиваемый на поиск ошибок и обоснование уверенности в правильности работы программы, нередко намного превосходит усилия, связанные с написанием самой программы. Решающее и окончательное слово в процессе отладки всегда остается за проведением серии тестовых расчетов. Конечно, публикуя программы, авторы, как правило, проводят ту или иную работу по оценке их качества, но чужие программы все-таки вызывают естественное недоверие пользователя. Рассеять это недоверие также помогают тестовые расчеты.

В работе [Фаддеева, Колотилина, 1982а, б,в] содержится значительное количество тестовых примеров, предназначенных для тестирования алгоритмов решения основных задач линейной алгебры. Тесты имеют довольно четкую систематизацию не только по задачам, но и по особенностям строения матриц.

Всего приведем три компактных примера.

Тест 1. Вычисление обратной матрицы A^{-1}, если задана матрица A [Caffrey, 1963]. Формулы, записанные ниже, в точности соответствуют их написанию в наборе матриц для тестирования [Фаддеева, Колотилина, 1982а,б,в]:

$$A=\begin{bmatrix}1 & 1 & 1 & 1\\ 1 & 2 & 3 & 4\\ 1 & 3 & 6 & 10\\ 1 & 4 & 10 & 20\end{bmatrix},$$

$$A^{-1}=\begin{bmatrix}4 & -6 & 4 & -1\\ -6 & 14 & -11 & 3\\ 4 & -11 & 10 & -3\\ -1 & 3 & -3 & 1\end{bmatrix}.$$

Тест 2. Решение системы линейных уравнений $Ax=b$ [Chan, Klassen, 1975]:

$$A=\begin{bmatrix}1 & 10^2 & 10^4 & 10^6\\ 10^2 & 1 & 10^2 & 10^4\\ 10^4 & 10^2 & 1 & 10^2\\ 10^6 & 10^4 & 10^2 & 1\end{bmatrix},$$

$$b=\begin{bmatrix}10101101\\ 10201\\ 10201\\ 10101101\end{bmatrix},\quad x=\begin{bmatrix}1\\ 1\\ 1\\ 1\end{bmatrix}.$$

Тест 3. Вычисление матрицы большой размерности. В публикации [Страхов, Страхов, 2002] с целью проверки технической корректности разработанных программ и оценки их эффективности были выполнены расчеты на модельных и практических примерах. В частности использовалась матрица размера 5000×5000 с элементами

$$a_{ij}=\frac{h}{(i-j)^2+h^2},\quad h=\frac{1}{2}.$$

На рис. 5 приведен код для всех трех примеров. Подробную информацию о инструкциях, используемых в коде, можно найти в документации [Numpy..., 2018; SciPy..., 2018]. Нетрудно заметить, что синтаксис языка Python более чем интересен. Он прост, понятен и нагляден. В некотором смысле его можно даже назвать поспартански лаконичным. Одновременно с этим программные коды, написанные на Python, обычно легко читаются и анализируются, а объем программного кода

Реализация первого теста выглядит так:

```
In [1]: import numpy as np

In [2]: A = np.array([[1, 1, 1, 1], [1, 2, 3, 4],
                      [1, 3, 6, 10], [1, 4, 10, 20]], dtype=np.float64)

In [3]: A # Вывод массива

Out[3]: array([[ 1.,  1.,  1.,  1.],
               [ 1.,  2.,  3.,  4.],
               [ 1.,  3.,  6., 10.],
               [ 1.,  4., 10., 20.]])

In [4]: from numpy.linalg import inv

In [5]: ainv = inv(A)

In [6]: ainv # Результат

Out[6]: array([[  4.,  -6.,   4.,  -1.],
               [ -6.,  14., -11.,   3.],
               [  4., -11.,  10.,  -3.],
               [ -1.,   3.,  -3.,   1.]])
```

Реализация второго теста:

```
In [7]: A = np.array([[1, 10**2, 10**4, 10**6], [10**2, 1, 10**2, 10**4],
                      [10**4, 10**2, 1, 10**2], [10**6, 10**4, 10**2, 1]])

In [8]: A

Out[8]: array([[      1,     100,   10000, 1000000],
               [    100,       1,     100,   10000],
               [  10000,     100,       1,     100],
               [1000000,   10000,     100,       1]])

In [9]: b = np.array([1010101, 10201, 10201, 1010101], dtype=np.float64)

In [10]: b # Выводим массив b

Out[10]: array([1010101.,   10201.,   10201., 1010101.])

In [11]: x = np.linalg.solve(A, b)

In [12]: print(x) # Печатаем решение

         [1. 1. 1. 1.]
```

Третий тест:

```
In [13]: n = 4 # n - порядок квадратной матрицы
         h = 0.5
         np.fromfunction(lambda i, j: h / ((i - j) ** 2 + h * h), (n, n), dtype=float)

Out[13]: array([[2.        , 0.4       , 0.11764706, 0.05405405],
                [0.4       , 2.        , 0.4       , 0.11764706],
                [0.11764706, 0.4       , 2.        , 0.4       ],
                [0.05405405, 0.11764706, 0.4       , 2.        ]])
```

Замечание. Для вычисления матрицы размера 5000x5000, нужно заменить n = 4 в предыдущей ячейке ввода на n = 5000 и нажать Shift+Enter

Рис. 5. Код рассматриваемых примеров.

намного меньше, если сравнивать с аналогичными программами, написанными на других языках программирования.

Заключение. Jupyter Notebook можно охарактеризовать как мощь ipython, заключенную в интерактивную веб-страницу.

Самое примечательное в Jupyter Notebook — возможность править и запускать код, не покидая страницу, и, если есть необходимость, также можно добавлять текст и графические элементы. Ученые начинают применять блокноты Jupyter для того, чтобы публиковать свои исследования, включая весь код и данные, использованные в них. Jupyter Notebook — потрясающее приложение на Python.

Благодарность. Автор выражает признательность канд. физ.-мат. наук А. С. Савченко (Институт геофизики НАН Украины, Киев) за квалифицированную помощь, оказанную при подготовке рукописи к печати.

Список литературы

Лутц М. Изучаем Python. 4-е издание. Санкт-Петербург: Символ-Плюс, 2011. 1280 с.

Любанович Б. Простой Python. Современный стиль программирования. Санкт-Петербург: Питер, 2016. 480 с.

Саммерфилд М. Программирование на Python 3. Подробное руководство. Санкт-Петербург: Символ-Плюс, 2009. 608 с.

Страхов В. Н., Страхов А. В. Компьютерные технологии нахождения устойчивых приближенных решений систем линейных алгебраических уравнений с приближенно заданной правой частью. *Вопросы теории и практики геологической интерпретации гравитационных, магнитных и электрических полей: Материалы 29-й сессии Междунар. семинара им. Д. Г. Успенского (Екатеринбург, 28 января—2 февраля 2002 г.).* Ч. 2. Москва: ОИФЗ РАН, 2002. С. 48—62.

Фаддеева В. Н., Колотилина Л. Ю. Вычислительные методы линейной алгебры. Набор матриц для тестирования. Ч. 1. Ленинград: ЛОМИ АН СССР, 1982а. 131 с.

Фаддеева В. Н., Колотилина Л. Ю. Вычислительные методы линейной алгебры. Набор матриц для тестирования. Ч. 2. Ленинград: ЛОМИ АН СССР, 1982б. 111 с.

Фаддеева В. Н., Колотилина Л. Ю. Вычислительные методы линейной алгебры. Набор матриц для тестирования. Ч. 3. Ленинград: ЛОМИ АН СССР, 1982в. 144 с.

Черная О. А., Якимчик А. И. О процессах доортогонализации некоторых семейств векторов, возникающих при построении характеристических полиномов матриц и используемых при решении систем линейных алгебраических уравнений. 1. *Геофиз. журн.* 2005а. Т. 27. № 3. С. 503—511.

Черная О. А., Якимчик А. И. О процессах доортогонализации некоторых семейств векторов, возникающих при построении характеристических полиномов матриц и используемых при решении систем линейных алгебраических уравнений. 2. *Геофиз. журн.* 2005б. Т. 27. № 5. С. 790—805.

Якимчик А. И., Черная О. А. О нахождении устойчивых приближенных решений систем линейных алгебраических уравнений с плохо обусловленными матрицами. *Вопросы теории и практики геологической интерпретации гравитационных, магнитных и электрических полей: Материалы 34-й сессии Междунар. семинара им. Д. Г. Успенского (Москва, 29 января—3 февраля 2007 г.).* Москва: Изд. ИФЗ РАН, 2007. С. 300—301.

Якимчик А. І., Чорна О. А. Про узагальнення одного методу розв'язання систем лінійних алгебраїчних рівнянь на випадок несиметричної матриці, що мають місце в задачах геофізики. *Доп. НАН України*. 2006. № 7. С. 139—143.

Braun, N., Hauth, T., Pulvermacher, C., & Ritter, M. (2017). An Interactive and Comprehensive Working Environment for High-Energy Physics Software with Python and Jupyter Notebooks. *Journal of Physics: Conference Series*, 898, 072020. doi: 10.1088/1742-6596/898/7/072020.

Caffrey, J. (1963). Another test matrix for determinants and inverses. *Communications of the ACM,* 6(6), 310. https://doi.org/10.1145/366604.366618.

Chan, C., & Klassen, M. (1975). A performance comparison study between subroutine packages LINSYS, IBMSSP and IMSL for solving systems of linear equations. *Journal of Computational and Applied Mathematics,*

1(2), 111—113. https://doi.org/10.1016/0771-050X(75)90028-5.

Chudnov, D. (2016). The Intentional DATA SCIENTIST (Part II): JUPYTER — A New Kind of Notebook. *Computers in Libraries, 36*(6), 26—28.

Numpy Reference Release 1.15.4. Written by the NumPy community. (2018). [Электронный ресурс]. Режим доступа: https://docs.scipy.org/doc/numpy-1.15.4/numpy-ref-1.15.4.pdf.

SciPy Reference Guide. Release 1.1.0. Written by the SciPy community. (2018). [Электронный ресурс]. Режим доступа: https://docs.scipy.org/doc/scipy-1.1.0/scipy-ref-1.1.0.pdf.

Tauxe, L., Shaar, R., Jonestrask, L., Swanson-Hysell, N. L., Minnett, R., Koppers, A. A. P., Constable, C. G., Jarboe, N., Gaastra, K., & Fairchild, L. (2016). PmagPy: software package for paleomagnetic data analysis and a bridge to the Magnetics Information Consortium (MagIC) Database. *Geochemistry, Geophysics, Geosystems*, 17(6), 2450—63. https://doi.org/10.1002/2016GC006307.

Yang-Min, K., Jean-Baptiste, P., & Guillaume, D. (2018). Experimenting with reproducibility: a case study of robustness in bioinformatics. *GigaScience*, 7(7), 1—8. https://doi.org/10.1093/gigascience/giy077.

Jupyter Notebook: a system for interactive scientific computing

A. I. Yakimchik, 2019

Jupyter Notebook — is a web-appendix which allows writing and supplying comments a code to Python in interactive regime. It is an exclusive method to make experiments and studies and intercommunicate with others. Many research people use this calculative medium in their works more often. The main factors of growing popularity of programming language Python and project Jupyter are characterized in brief. The basic of them are: high velocity of development and merit of software; standard library and libraries with open initial code NumPy, SciPy, Matplotlib et al.; simplicity of integration with code to C, C++ and FORTRAN; free distribution; support and numerous assemblage of designers and users. According to the data of TIOBE company, collecting monthly statistics of search inquiries and on the base of data obtained compiles its own visualized rates of programming languages Python ranks the third place in popularity among programming languages. It was chosen as a language of a year in 2007, 2010 and 2018. Aspects of installation of programs, libraries and packets in operational system Windows have been considered. It is recommended to download and install the libraries from the storage of whl-files on the web-page by Christoph Gohlke from the laboratory of fluorescence dynamics of California University. WHL format is supported by all basic platforms (Mac OS X, Linux, Windows). The process of starting the server of Jupyter notebooks from command line has been described in details. The simplicity and effectiveness of scientific calculations in Jupyter Notebook have been demonstrated. Test calculations have been given for solving the problems of linear algebra. It has been shown in particular that the code of calculation of the matrix of 5000×5000 size occupies only several lines.

Key words: open source software, languages of programming, linear algebra, matrix, test calculations.

References

Lutz, M. (2011). *Learning Python. 4th edition*. St. Petersburg: Simvol-Plyus, 1280 p. (in Russian).

Lubanovic, B. (2016). *Simple Python. Modern programming style*. St. Petersburg: Piter, 480 p. (in Russian).

Summerfield, M. (2009). *Programming in Python 3. A detailed guide*. St. Petersburg: Simvol-Plyus, 608 p. (in Russian).

Strakhov, V. N., & Strakhov, A. V. (2002). Computer technologies for finding stable approximate so-

lutions of systems of linear algebraic equations with an approximately given right-hand side. *Theory and practice of the geological interpretation of gravitational, magnetic and electric fields: Proc. of the 29th session of the Intern. seminar them. D. G. Uspensky (Ekaterinburg, January 28—February 2, 2002)* (pp. 48—62). Part 2. Moscow: Edition of the Institute of Physics of the Earth RAS (in Russian).

Faddeeva, V. N., & Kolotilina, L. Yu. (1982a). *Computational methods of linear algebra. A set of matrices for testing.* Part 1. Leningrad: Publ. of the Leningrad Branch of the Mathematical Institute of the Academy of Sciences of the USSR, 131 p. (in Russian).

Faddeeva, V. N., & Kolotilina, L. Yu. (1982b). *Computational methods of linear algebra. A set of matrices for testing.* Part 2. Leningrad: Publ. of the Leningrad Branch of the Mathematical Institute of the Academy of Sciences of the USSR, 111 p. (in Russian).

Faddeeva, V. N., & Kolotilina, L. Yu. (1982c). *Computational methods of linear algebra. A set of matrices for testing.* Part 3. Leningrad: Publ. of the Leningrad Branch of the Mathematical Institute of the Academy of Sciences of the USSR, 144 p. (in Russian).

Chernaya, O. A., & Yakimchik, A. I. (2005a). On the processes of pre-orthogonalization of some vectors families which appear while plotting characteristic polynomials of matrices and used while solving the systems of linear algebraic equations. 1. *Geofizicheskiy zhurnal, 27*(3), 503—511 (in Russian).

Chernaya, O. A., & Yakimchik, A. I. (2005b). On the processes of pre-orthogonalization of some vectors families which appear while plotting characteristic polynomials of matrices and used while solving the systems of linear algebraic equations. 2. *Geofizicheskiy zhurnal, 27*(5), 790—805 (in Russian).

Yakimchik, A. I., & Chernaya, O. A. (2007). On the determination of stable approximate solutions of systems of linear algebraic equations with ill-conditioned matrices. *Questions of the theory and practice of the geological interpretation of gravitational, magnetic and electric fields: Proc. of the 34th session of the Intern. seminar them. D. G. Uspensky (Moscow, January 29 — February 3, 2007).* Moscow: Edition of the Institute of Physics of the Earth RAS (in Russian).

Yakimchik, A. I., Chorna, O. A. (2006). On the generalization of a method for solving systems of linear algebraic equations in the case of an asymmetric matrix that occur in problems of geophysics. *Dopovidi NAN Ukrainy*, (7), 139—143 (in Ukrainian).

Braun, N., Hauth, T., Pulvermacher, C., & Ritter, M. (2017). An Interactive and Comprehensive Working Environment for High-Energy Physics Software with Python and Jupyter Notebooks. *Journal of Physics: Conference Series*, 898, 072020. doi: 10.1088/1742-6596/898/7/072020.

Caffrey, J. (1963). Another test matrix for determinants and inverses. *Communications of the ACM,* 6(6), 310. https://doi.org/10.1145/366604.366618.

Chan, C., & Klassen, M. (1975). A performance comparison study between subroutine packages LINSYS, IBMSSP and IMSL for solving systems of linear equations. *Journal of Computational and Applied Mathematics, 1*(2), 111—113. https://doi.org/10.1016/0771-050X(75)90028-5.

Chudnov, D. (2016). The Intentional DATA SCIENTIST (Part II): JUPYTER — A New Kind of Notebook. *Computers in Libraries, 36*(6), 26—28.

Numpy Reference. Release 1.15.4. Written by the NumPy community. (2018). Retrieved from https://docs.scipy.org/doc/numpy-1.15.4/numpy-ref-1.15.4.pdf.

SciPy Reference Guide. Release 1.1.0. Written by the SciPy community. (2018). Retrieved from https://docs.scipy.org/doc/scipy-1.1.0/scipy-ref-1.1.0.pdf.

Tauxe, L., Shaar, R., Jonestrask, L., Swanson-Hysell, N. L., Minnett, R., Koppers, A. A. P., Constable, C. G., Jarboe, N., Gaastra, K., & Fairchild, L. (2016). PmagPy: software package for paleomagnetic data analysis and a bridge to the Magnetics Information Consortium (MagIC) Database. *Geochemistry, Geophysics, Geosystems*, 17(6), 2450—63. https://doi.org/10.1002/2016GC006307.

Yang-Min, K., Jean-Baptiste, P., & Guillaume, D. (2018). Experimenting with reproducibility: a case study of robustness in bioinformatics. *GigaScience,* 7(7), 1—8. https://doi.org/10.1093/gigascience/giy077.