Wojciech Myszka

Jupyter

10 listopada 2022

Spis treści

1. Jupyter

1.1. Wstęp

Jupyter jest projektem Open Source, który zdobywa obecnie sporą popularność. Jego historia sięga roku 2011. Wystepuje w trzech smakach:

— Jupyter Notebook — klasyczne rozwiązanie bazujące na IPython Notebook, pozwalające łatwo edytować kod źródłowy.
— Jupyter Lab — najnowszy (2018) interfejs ma nieco większe możliwości.
— Jupyter Hub — serwer przystosowany do równoczesnej pracy wielu użytkowników.

Na pierwszy rzut oka jest to coś podobnego do Mathematici: interfejs pozwalający tworzyć notatnik oraz prowadzić w nim obliczenia czy wizualizować wyniki. O ile jądrem obliczeniowym Mathematici jest jej jądro o sporych możliwościach obliczeń symbolicznych i numerycznych, to jądrem obliczeniowym Jupytera może być jeden z wielu języków programowania[1]:

— głównie Python wraz z bibliotekami NumPy i SciPy,
— R,
— Julia,

ale również inne:

— JavaScript,
— Java,
— Scala,
— Go,

[1] Nazwa Jupyter pochodzi od Ju(lia) + Py(thon) + (e)R.

— C#,
— Ruby,
— Kotlin,
— Haskell.
Ale to nie koniec. Jąder obliczeniowych jest w sumie około stu, a wśród nich zarówno Matlab jak i Mathematica.

1.2. Instalacja

Sposób instalacji nie jest przedmiotem niniejszego krótkiego tutoriala. Znaleźć można go na stronach z dokumentacją jupytera[2] lub tu. Ale nie tylko tam. Program jest tak popularny, że praktycznie wszędzie tam gdzie sugerują korzystanie z niego podają też sposób instalacji.

Oprogramowanie bez problemu powinno pracować w każdym popularnym środowisku[3]

Wydaje mi się, że instalacja anacondy może być najlepszym rozwiazaniem niż instalacja pojedynczych pakietów za pomocą programu pip. W jednym (sporawym) pakiecie dostaje się to co może być przydatne w trakcie zajęć.

W szczególności w Windows (nie ma tam środowiska Python) dostajemy ten język programowania z wielu dodatkowymi pakietami. SW systemie Linux (tam python zazwyczaj już jest) Anaconda duplikuje wiele pakietów, ale ułatwia instalowanie nowych. W szczególności do różnego rodzaju testów. Wraz z usunięciem Anacondy otrzymujemy nienaruszone orginalne środowisko.

Deinstalacja anacondy usuwa wszystko i można o niej zapomnieć.

[2] https://jupyter.org/install
[3] Linux, Window, IOS.

1.3. Uruchomienie w laboratorium 604 B1

W przypadku chęci skorzystania z jupytera (i innych narzędzi związanych z Anacondą) należy **jednorazowo** wykonać następujące czynności:

1. Otworzyć terminet (najprościej, na klawiaturze nacisnąć równocześnie trzy klawisze: Ctrl-Alt-T).
2. W terminalu napisać:
   `source /opt/anaconda3/bin/activate`
3. A następnie:
   `conda init`
   i zamknąć terminal (na przykład naciskając klawisze Ctrl-D).

Od tej chwili będzie można już korzystać z anacondy i/lub jupytera. Kolejne otworzenie terminala pokaże zmodyfikowany napis zachęty (*prompt*), który będzie wyglądał jakoś tak:

```
(base) 123456@piwko012:~$
```

Dobrym sosobem na rozpoczęcie może być uruchomienie nawigatora, czyli napisanie w terminalu:

```
anaconda-navigator
```

(W chwili obecnej nie można instalować dodatkowych pakietów, ani tworzyć środowisk!)

1.4. Uruchomienie

Notatnik jupyter pracuje w układzie serwer (jądro) — klient (klientem jest przeglądarka www). Uruchamiamy go wpisując w terminalu[4]:

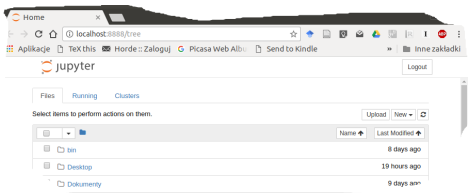
```
jupyter notebook
```

[4] Aby otworzyć terminal najprościej nacisnąć równocześnie trzy klawisze Ctrl-Alt-T albo znaleźć terminal na liście aplikacji.

Po uruchomieniu jądra, uruchamiana jest przeglądarka (rys. 1.1) i automatycznie kierowana pod adres http://localhost:8888. W przypadku gdy zamkniemy przeglądarkę, wystarczy w tym oknie, w którym uruchomione jest polecenie `jupyter notebook` nacisnąć równocześnie klawisze Ctrl-C. Próba przerwania jupytera zaowocuje zadaniem pytania czy na pewno przerwać, ale dodatkowo wyświetli również informację pod jakim portem szukać jupuytera:

```
The Jupyter Notebook is running at:
 http://localhost:8888/?token=5ac2de4791858399254a74de588cd4016a3816
 Shutdown this notebook server (y/[n])?
```

(Niestety) ten token jest niezbędny!

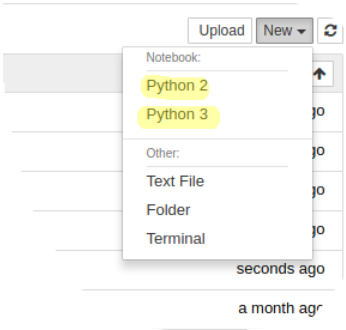


Rysunek 1.1. Okno przeglądarki po uruchomieniu notatnika Jupyter

1.5. Praca z notatnikiem

Nowy notatnik tworzony jest wybierając opcję „New” (rys. 1.2). W chwili obecnej (w laboratorium) mamy dwie możliwości (dwa języki) do wyboru: Python 2 i Python 3. Na pierwszy rzut oka różnice między tymi dialektami nie są wielkie, ale nie każdy, nawet najprostszy kod napisany w Pythonie 2 uruchomi się gdy używamy interpretera Python 3[5].

[5] Najważniejsze różnice posumowane są na stronie: http://sebastianraschka.com/Articles/2014_python_2_3_key_diff.html. Standardowo dostępny jest program 2to3, który pokazuje różnice ale może też



Rysunek 1.2. Tworzenie nowego notatnika

Korzystanie z notatnika jupyter jest bardzo podobne jak w Mathematici. Wpisujemy polecenia w języku, który rozumie jądro, a następnie naciskamy klawisze Shift+Enter. Jądro wykonuje natychmiast polecenie.

W każdej komórce można oprócz polecenia umieścić albo kod albo tekst. Tekst może być formatowany zgodnie z pewnym podzbiorem standardu markdown[6].

Kod musi spełniać wszystkie wymagania używanego języka programowania. Po przełączeniu komórki w tryb *Markdown* można wprowadzać tekst. Będzie on wprowadzany zgodnie z następującymi zasadami:

— Aby zacząć nowy ustęp (paragraf) tekstu należy zostawić co najmniej jedną linię pustą.
— Listy tworzymy w sposób naturalny.

---

automatycznie dokonać odpowiednich korekt (gdy użyć go z parametrem **-w**): **2to3 -w a.py**.

[6] Pod tym odsyłaczem można znaleźć podstawowe informacje na temat języka Markdown. Swoją drogą Markdown to bardzo wygodny sposób do przygotowywania nawet dosyć rozbudowanych tekstów. Mimo, że wprowadzanie tekstu jest całkowicie tekstowe, można sobie pozwolić na wiele.

    - Lista numerowana zaczyna się od liczby. Lista nienumerowana zaczyna się od punktora (może być nim znak + albo −.
    - Pod-lista powinna być „wcięta” w stosunku do pozycji macierzystej.
- Można korzystać z tabelek, ale ich tworzenie może być (na pierwszy rzut oka) niezbyt proste. Ale można się posiłkować generatorem tabelek.
- Wzory matematyczne — wprowadzane zgodnie ze standardami LaTeXa. Nie jest on potrzebny (nie musi być zainstalowany) — do wyświetlania używana jest znakomita biblioteka MathJax. Żeby wzory umieszczać — potrzebna jest dosyć elementarna znajomość LaTeXa. Można skorzystać z jednego z wielu dostępnych w sieci serwisów on-line pozwalających wzór „wyklikać”. Może to być, na przykład, Sciweavers.
- Obrazki wstawia się zgodnie z zasadami Markdown:

```
![tekst altenatywny](jup_ekran.png "Tytuł obrazka")
```

Tak zwany *rich content* może być bardzo łatwo włączany do notatników Jupyter. Opisuje to dokumentacja.

Najważniejsze skróty klawiaturowe zawiera ściąga.

1.6. Dokumentacja

Ponieważ Jupyter jest dziś produktem bardzo modnym, wszędzie można znaleźć (czasami różnej jakości) przykłady tłumaczące zasady pracy z tym programem. W związku z tym tu, dokumentacja nie zostanie umieszczona.

W pewnym sensie najlepszym źródłem informacji jest serwis bazowy.

1.7. Przykłady

1. Najlepszym źródłem przykładów będzie strona umieszczona w serwisie GitHub jupytera.
2. Warto sięgnąć do książki Python Data Science Handbook której źródła dostępne są jako szereg natatników jupyter umieszczonych (oczywiście) w serwisie GitHub.
3. Wspomnieć też muszę o serwisie Binder pozwalającym na udostępnianie notatników Jupytera on-line. Jest tam też podstawowy zestaw dokumentacji.
4. Na potrzeby zajęć przygotowałem kilka przykładowych noteboków. Na razie nie bardzo wiem jak je udostępnić najsensowniej. Zapewne skończy się utworzeniem własnego kąta w GitHubie. Na razie dostępne są (wraz z niezbędnymi plikami) jako jeden plik ZIP[7].

   Pokazują one rozwiązanie (lub może raczej dają jakiś rodzaj wędki) kilku problemów związanych z tematyką zajęć.

   a) Aproksymacja:
      — Aproksymacja wielomianowa — bardzo podstawowe informacje na temat aproksymacji.
      — Przykład realizacji aproksymacji liniowej: Aproksymacja liniowa.

   b) Interpolacja:
      — Bardzo podstawowe dane na temat interpolacji: Interpolacja.
      — Coś jeszcze o interpolacji nieudolna próba pokazania w jaki sposób można walczyć z niewłaściwym zachowaniem funkcji interpolacyjnych, które „wariują” na końcach przedziału.

   c) Różne:
      — Czytanie danych zawierających ciągi znaków NaN. Zawiera

[7] Ściągnięty plik należy gdzieś rozpakować, a następnie w tej kartotece uruchomić notatnik jupytera.

również informacje na temat odrzucania różnych rodzajów błędnych danych.
   - Informacje na temat wczytania do notatnik danych z „drugiego zestawu”[8]. Są tam również informacje na temat proponowanego sposobu „redukcji” danych (resampling).

d) FFT:
   - Szybka transformata Fouriera. Zawiera podstawowe informacje na temat szybkiej transformaty Fouriera.
   - Rozdzielczość transformaty Fouriera. Zadanie do rozwiązania związane z Laboratorium 2a.
   - Zagadka. Drobiazg do przemyśleń na temat tego co, tak na prawdę, pokazuje FFT.
   - Piki Przykład starający się pokazać jak losowość sygnały potrafi „wychodzić” spod zaburzeń.

e) Obliczenia dużej precyzji:
   - Obliczenia dużej precyzji w Pythonie. Przykład do laboratorium nr 4.

f) Optymalizacja
   - Przykład optymalizacji funkcji jednej i dwu zmiennych w Pythonie.

•

1.8. Instrukcja w postaci jednego pliku...

...jest również dostępna.

[8] Sugeruję zapoznanie się z odpowiednią instrukcją laboratoryjną. Same dane są tutaj.