IPython Notebook の紹介と cERLでの利用計画

2015/09/15 T. Obina
KEKB制御打ち合わせ

導入を検討した動機：High Level Application

- リアルタイム表示や、基本的な操作パネルは CSS でOK
- 高レベルアプリケーションをどうするか？？？

現在のcERL/PF/PF-ARでは

- SAD, root, Matlab などが使用可能
  - お好きなモノを、お好きなように使って頂く方針
  - CSS からの Launcher もあります（GUI が必要なときは X 端末エミュレータが必要）。ちょっと使いにくい。

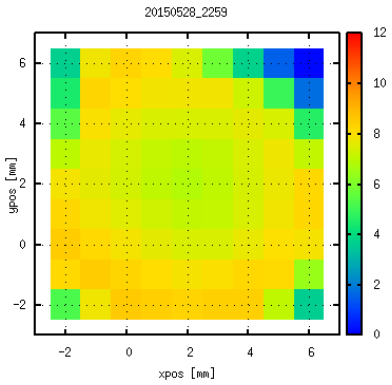
例：

- QE（量子効率）測定は python script + gnuplot でやっていた
- 入射部のマッチングは宮島さん作成の Matlab
- LCS実験では本田さん、赤木さん、小菅さん作成のroot
- LCSバンプ制御は原田さん作成の SAD Script
- これらの既存ソフトウェアを CSS に移行する必要性は高くない。
  - Launcher と操作マニュアルは必要
  - 実際に移植するのはかなり大変
- 現実的にはSADだが、軌道計算が不要な場合にはちょっと…

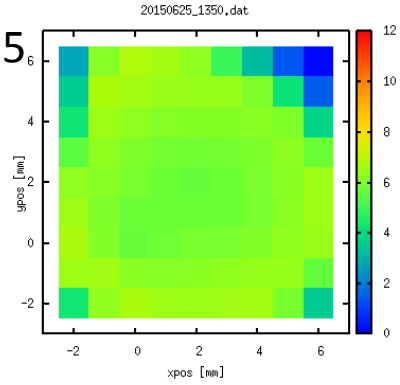
QE測定、オプティクスマッチング例

- QE(量子効率）測定例：他のスクリプトで測定を実施して、データをファイルに保存、gnuplot（ X 端末使用）

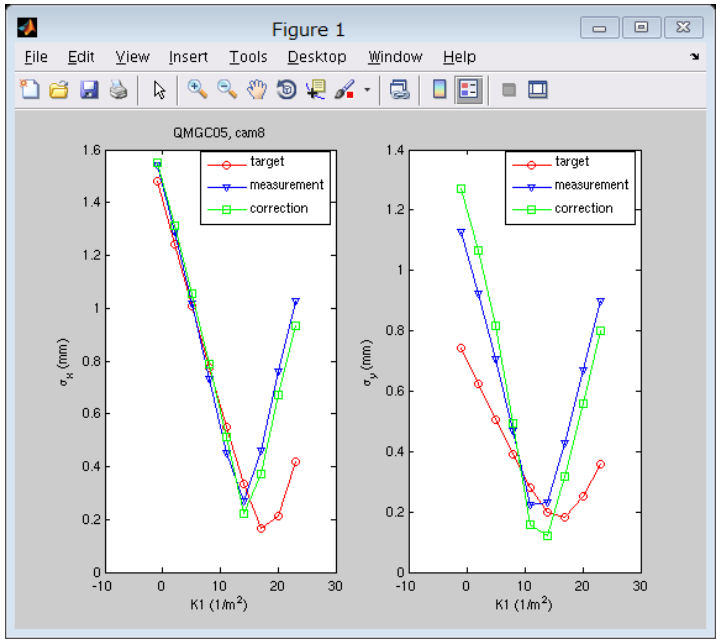
5/28



6/25

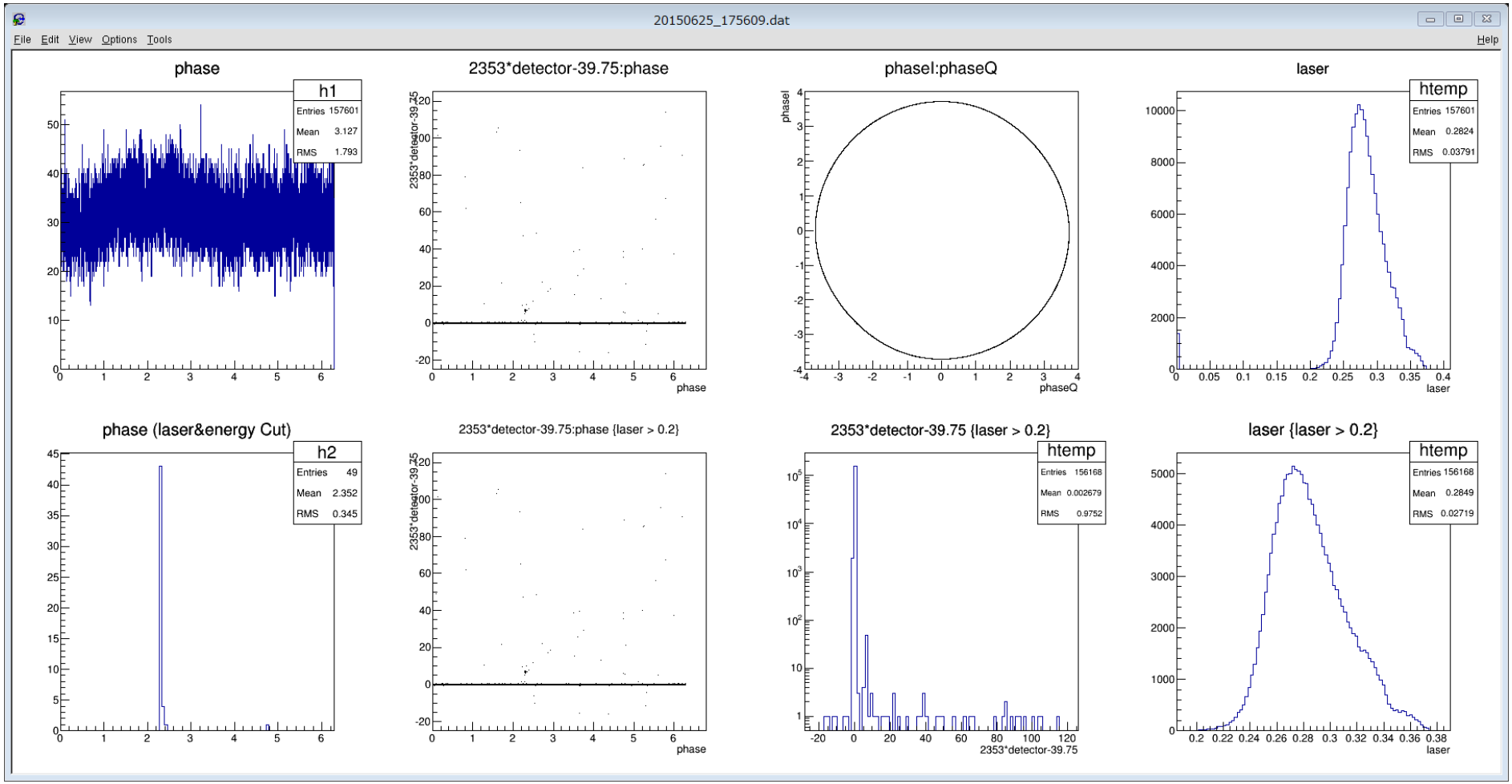


- マッチング例：Matlab script
  - やはりX端末
  - 測定スクリプトと表示は一体
  - caput/get必須
  - SVDや nonlinear fit など



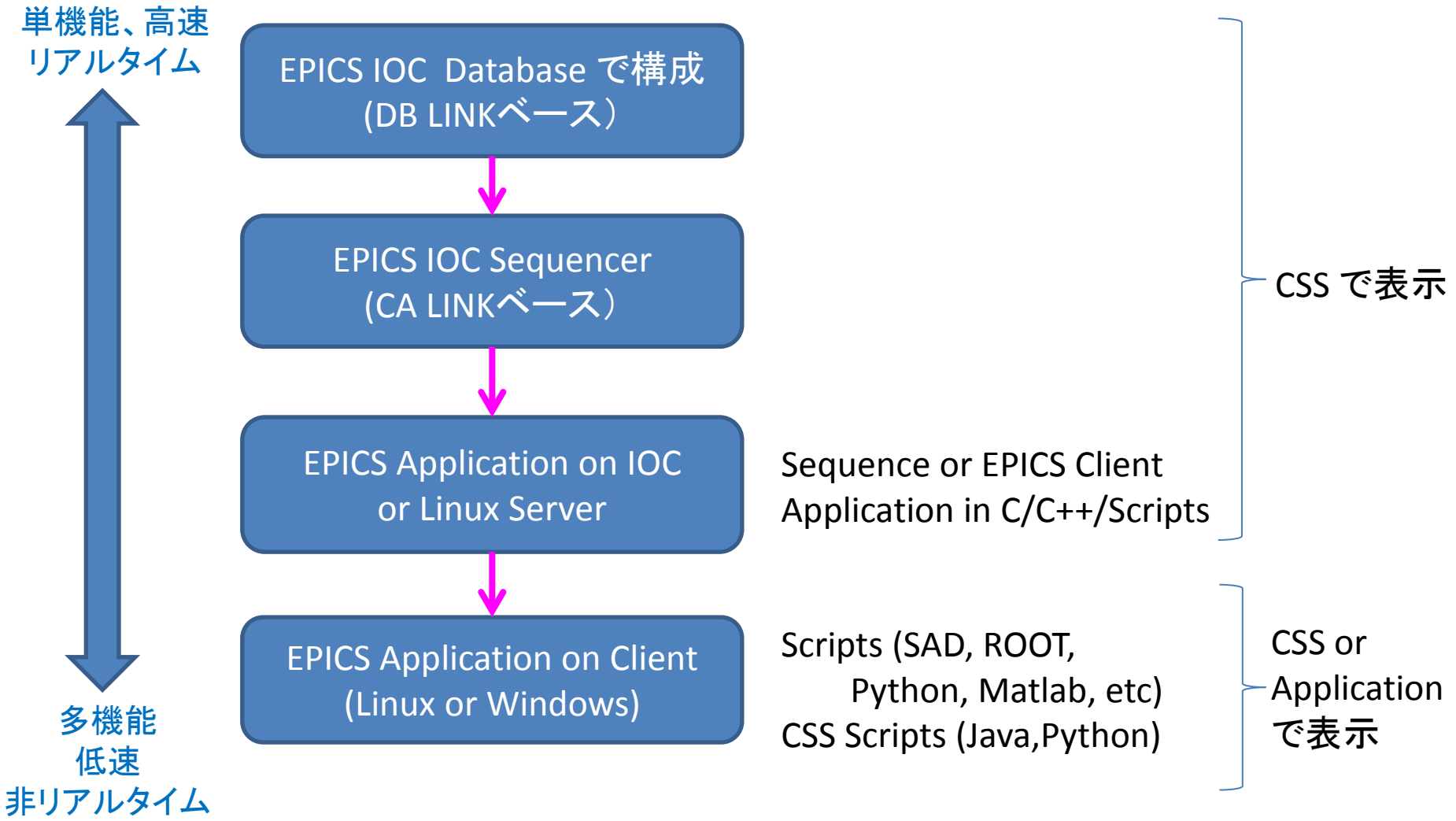
LCS実験測定例

- 本田さん、赤木さん、小菅さん作成の root
  - ファイル保存は別スクリプトで、rootで解析・表示
  - 表示は X 端末使用



ツール選択の指針:上から順に検討するべき

本来なら開発前に関係者全員に周知すべき図のはずだが…



やりたいことは何か

- Matlabのように、対話的な解析がしたい
  - オフラインのデータ解析あるいはスクリプト内から caget/put してデータを取得しながら表示したい
  - （可能であれば）ＤＢと接続したい
  - データを確認しながら不要データを除外したりフィット区間を決めたりしたい
- その他
  - SVDなど、各種計算ルーチンのデバッグ作業は最小限にしたい
  - ネット上に情報が多くあると嬉しい
  - できれば x端末は避けたい

これらの要望をSADで実現するのは難しい（Expertなら不可能ではない？）

- しかし、Matlab は有償！！しかも高価。
  - Academic discount 適用ならば安価なのに…
- これらの要望解決手段として IPython Notebook を評価してみた。
  - IPython Notebook の環境をcERLおよびPFに整備した
  - 高速応答やリアルタイム性（１～５秒程度）が必要なモノはCSSに任せる方針

IPython Notebook とは

- もちろん、Python です。
- IPython：Interactive Python の略で、通常はコマンドラインで使う対話的シェル環境のこと。
  - タブ補完
  - 便利なコマンド類(magic command)
    - %who, %whos, %who_ls
    - %reset
    - %magic
  - 容易なヘルプ表示
    - ?command or command? でのヘルプ表示
  - lsとかcd 等も使える：標準pythonでは無理

```
% ipython
Python 2.7.3 (default, Dec 10 2012, 21:24:23)
Type "copyright", "credits" or "license" for more information.

IPython 4.0.0 -- An enhanced Interactive Python.
?         -> Introduction and overview of IPython's features.
%quickref -> Quick reference.
help      -> Python's own help system.
object?   -> Details about 'object', use 'object??' for extra details.

In [1]:
```

IPython：magic command例

- who, whos: 変数の表示

```
In [1]: a = [1,2,3]
In [2]: b = 'sample text string'
In [3]: import os
In [4]: who
a       b       os
In [5]: whos
Variable   Type      Data/Info
-----------------------------
a          list      n=3
b          str       sample text string
os         module    <module 'os' from '/cerl/<...>64/lib/python2.7/os.pyc'>
In [6]:
```

whoで変数の一覧表示

whosでタイプまで含めて表示

- タブ補完、ヘルプ表示

```
In [6]: os.mk
os.mkdir   os.mkfifo  os.mknod
In [6]: os.mknod?
Docstring:
mknod(filename [, mode=0600, device])

Create a filesystem node (file, device special file or named pipe)
named filename. mode specifies both the permissions to use and the
type of node to be created, being combined (bitwise OR) with one of
S_IFREG, S_IFCHR, S_IFBLK, and S_IFIFO. For S_IFCHR and S_IFBLK,
device defines the newly created device special file (probably using
os.makedev()), otherwise it is ignored.
Type:      builtin_function_or_method
In [7]:
```

ここで tab 入力すると、補完候補が出る

command? でヘルプ(docstring)表示

reset、その他

- reset：ユーザが定義した namespace をクリア

```
In [10]: whos
Variable   Type      Data/Info
-----------------------------
a          list      n=3
b          str       sample text string
ca         module    <module 'ca' from '/cerl/<...>.7/site-packages/ca.pyc'>
os         module    <module 'os' from '/cerl/<...>64/lib/python2.7/os.pyc'>
In [11]: reset
Once deleted, variables cannot be recovered. Proceed (y/[n])? y
In [12]: whos
Interactive namespace is empty.
In [13]:
```

- その他にも色々便利な機能はありますが、ここで紹介していると時間が足りなくなるのでここまで。
  - Google先生に聞いて下さい。山のように出てきます。

余談：デフォルトでの行間

- デフォルトでは、下のように出力が１行おきになります。

```
In [1]: a = [1,2,3]

In [2]: b = 'some sample string'

In [3]: import os

In [4]: who
a       b       os

In [5]: whos
Variable   Type      Data/Info
-----------------------------
a          list      n=3
b          str       some sample string
os         module    <module 'os' from '/cerl/<...>10/lib/python2.7/os.pyc'>

In [6]:
```

ずいぶん前に使ったとき、個人的には「便利ではあるが、ヒストリ番号も要らないし、デフォルトでは行間が空くのも気に食わない…」と思っていました。プロットは matplotlib を使っていましたが X端末が必須だし、遅いし、表示はあまり美しくないしーーーーーーー後述のNotebookで考えが変わります。

行間については、ipython --nosep で起動すれば解決です。

IPython Notebook

IPy The IPython Notebook ×

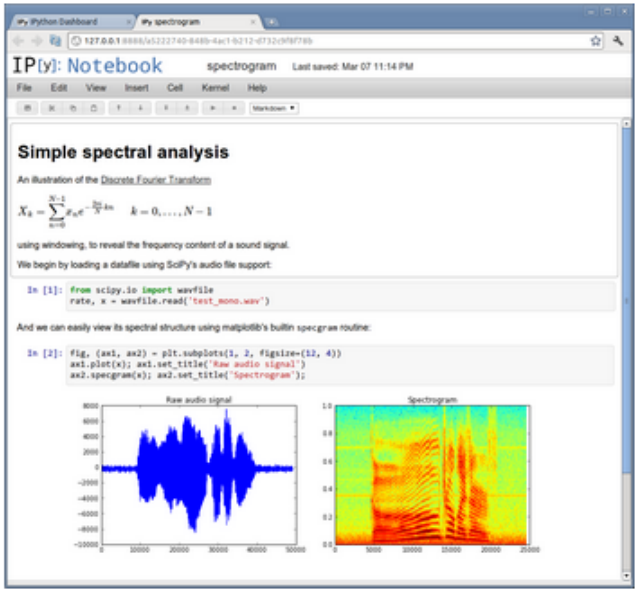
ipython.org/notebook.html

IP[y]: IPython
Interactive Computing

Install · Documentation · Project · Jupyter · News · Cite · Donate

The IPython Notebook

The IPython Notebook is an interactive computational environment, in which you can combine code execution, rich text, mathematics, plots and rich media, as shown in this example session:



It aims to be an agile tool for both exploratory computation and data analysis, and provides a platform to support **reproducible research**, since all inputs and outputs may be stored in a one-to-one way in notebook documents.

Google™ Custom Search Search

NOTEBOOK VIEWER

Share your notebooks

COMMUNITY

Stack Overflow
Mailing list
File a bug
Reddit

Tweet 647

FOR DEVELOPERS

Mailing list
Development Chat Room

IPython Notebookの機能

- Webブラウザ上での対話的実行環境
- コメント記述可能
  - Texフォーマットでの数式入力可
  - html, reStructuredText, Markdown記法なども使用可能
  - 日本語もOK
- デフォルトでの保存は .ipynb フォーマット(XML)
- PDF, html, python script など、各種フォーマットに変換可能
  - 公開・共有が容易
  - cERLでの例は後ほど

- プロットした画像は別ウィンドウで表示することも可能だが、インラインでの表示することもできる：こちらがお勧め
- nbviewer 公開・共有サイト　http://nbviewer.ipython.org/
  - Github/Gist に上げたipynbファイルを共有するサイト

仕組み

- IPython : Decoupled two-process model で構築されている
  - 通常の(古典的な) REPL(Read-Evaluate-Print Loop) の仕組みを拡張し、Evaluation を独立したプロセス(kernel)として実装
  - kernel は client からの要求を処理し、結果を返す
  - これによりIPython はTerminal,Qt console, Notebookなど様々なクライアント環境で実行可能
- IPython Notebook の場合
  - IPython エンジン(kernel) + JSON によるデータ通信 (Web Browser)
- ベースとなるソフトウェア
  - IPython
  - zero-MQ
  - Tornado (web server)
  - jQuery
  - Bootstrap (front-end framework)
  - MathJax (数式表示)
  - Jinja2 (template engine)
- 並列化も可能だが、今回はパス

起動と終了

- コマンドラインから ipython notebook 実行。
- カーネルを起動し、デフォルトで localhost:8888 で listen する
- 自動的にブラウザを起動。
  - --no-browser オプションをつけるとカーネルの起動のみ。
  - ポート指定オプションあり
  - ブラウザを終了しても、カーネルを実行し続けることが可能

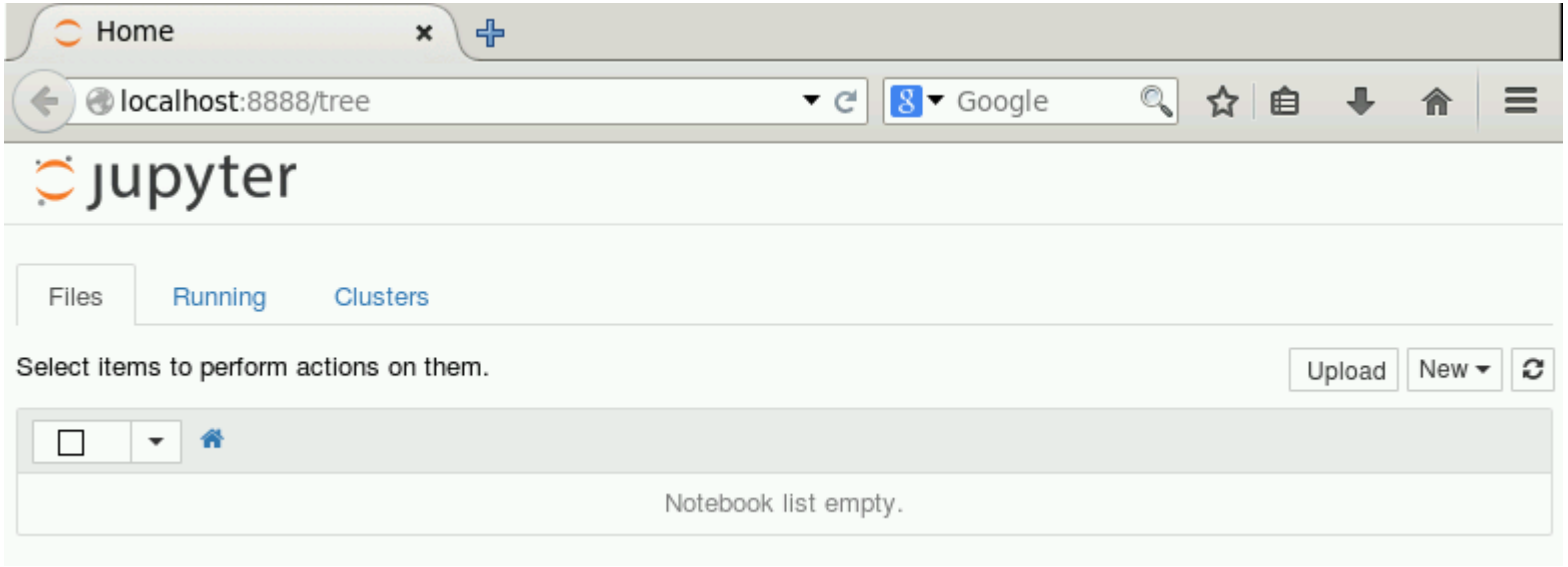
```
File  Edit  Setup  Control  Window  KanjiCode  Help
% ipython notebook
[I 10:35:48.263 NotebookApp] Serving notebooks from local directory: /mnt/users/obina/python/ipython
[I 10:35:48.263 NotebookApp] 0 active kernels
[I 10:35:48.263 NotebookApp] The IPython Notebook is running at: http://localhost:8888/
[I 10:35:48.263 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
/usr/bin/xdg-open: line 402: htmlview: command not found



^C[I 10:35:53.930 NotebookApp] interrupted
Serving notebooks from local directory: /mnt/users/obina/python/ipython
0 active kernels
The IPython Notebook is running at: http://localhost:8888/
Shutdown this notebook server (y/[n])? y
[C 10:35:54.794 NotebookApp] Shutdown confirmed
[I 10:35:54.794 NotebookApp] Shutting down kernels
%
```

  - 終了は Ctrl+C ２回

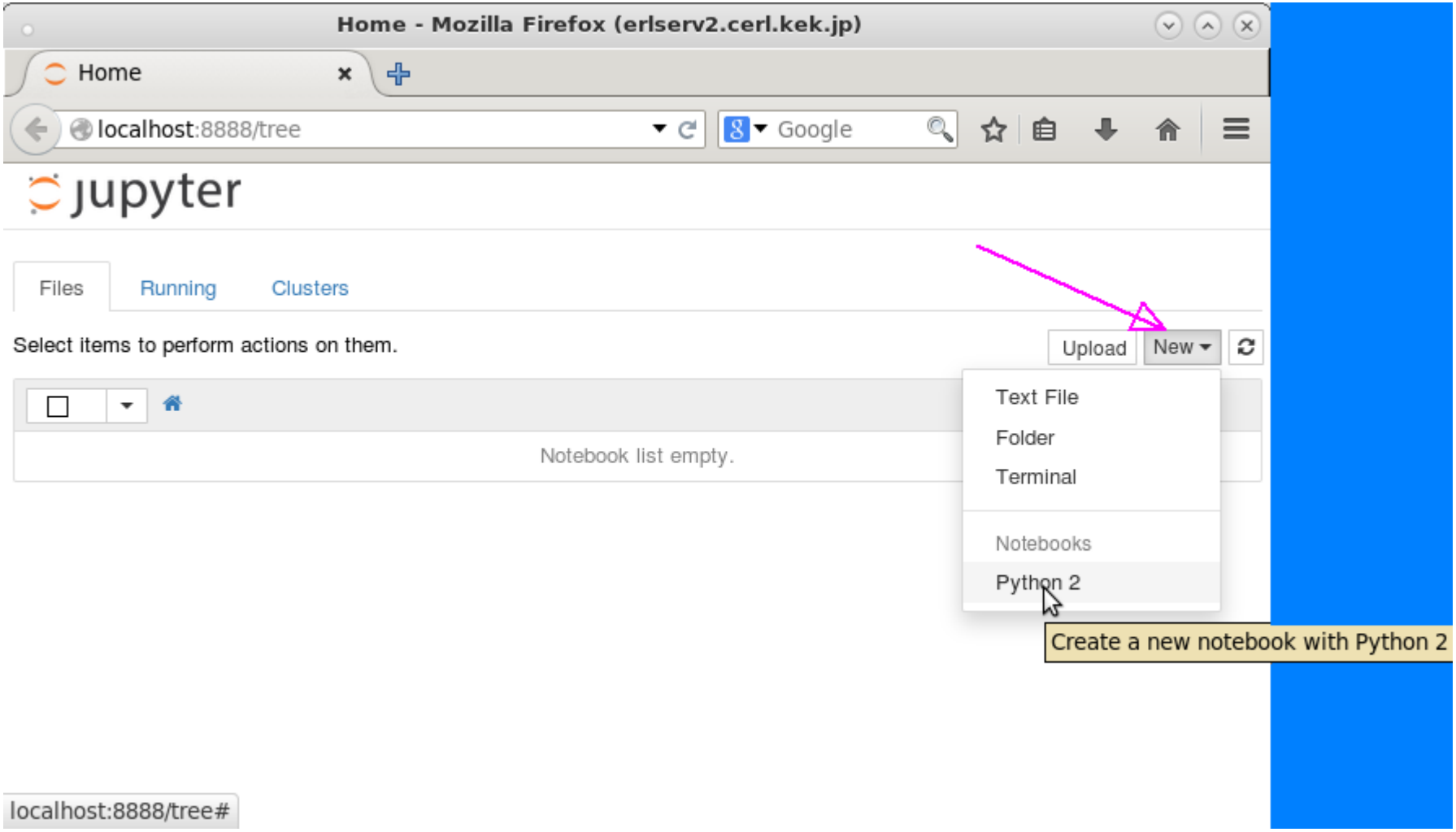
まずは簡単な例を紹介

- cERL wiki に記載（KEK内部のみアクセス可能）
  - http://pfconrg07.kek.jp:8082/trac/cerl/wiki/control/Soft/IPython

- 最初に Web Browser 起動したときの画面
  - 起動したディレクトリに日本語ファイルがあるとエラー出るので注意



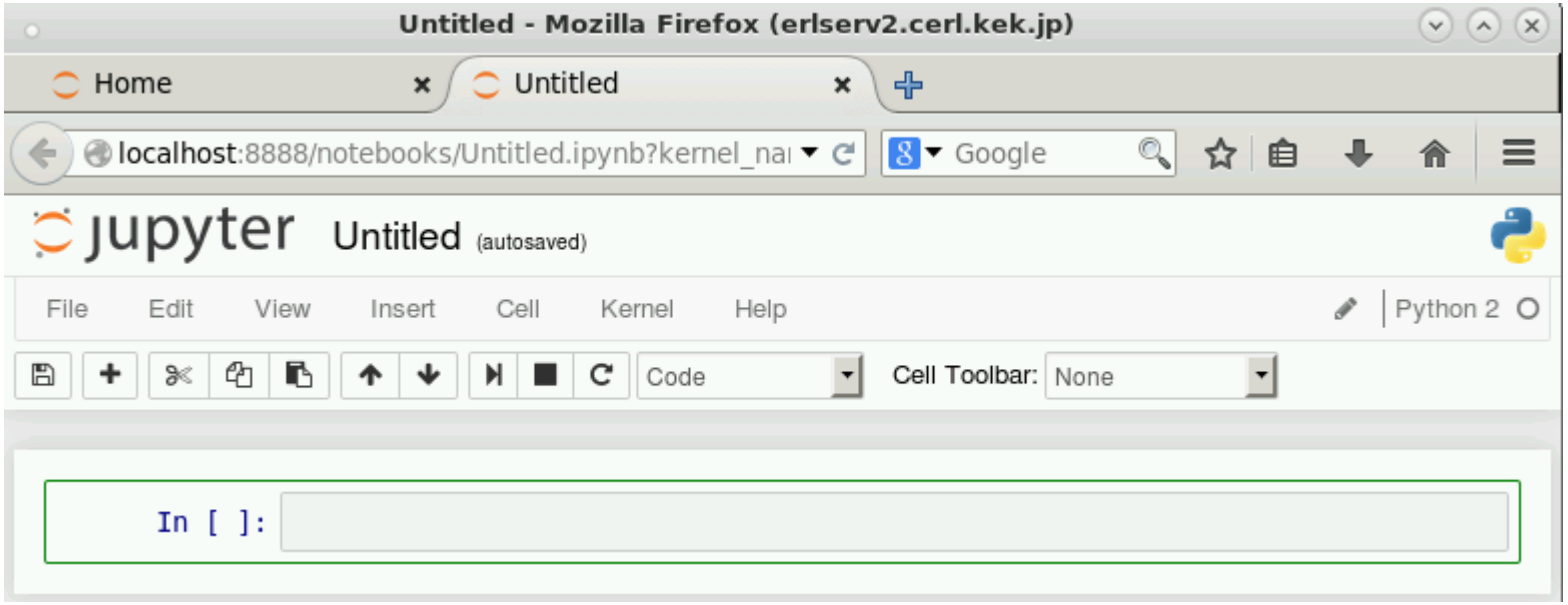
新しい Notebook の作成：python2



新 notebook 入力画面

- デフォルトは Code Cell　（Cell → Cell Type から変更可能）
  - Code　　：コード入力。Enterで改行、Shift+Enterで実行
  - Markdown：コメントなど。LaTex, html 使用可能
  　　：Level1 ～　6 まで。Markdown や reST で記述可能

- 切り貼りや移動（順序変更）可能



コード入力

- 最初の magic command はプロットした図を inline 表示するため
- 入力した後、Shift+Enter で実行される。
- 実行中は左のコマンド番号が [*] になる
  - 長時間になって途中で止めたいときは kernel メニューから停止可能

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

x = np.linspace(-1.2, 1.2)
y = x**2

plt.plot(x,y,'bo-')
```

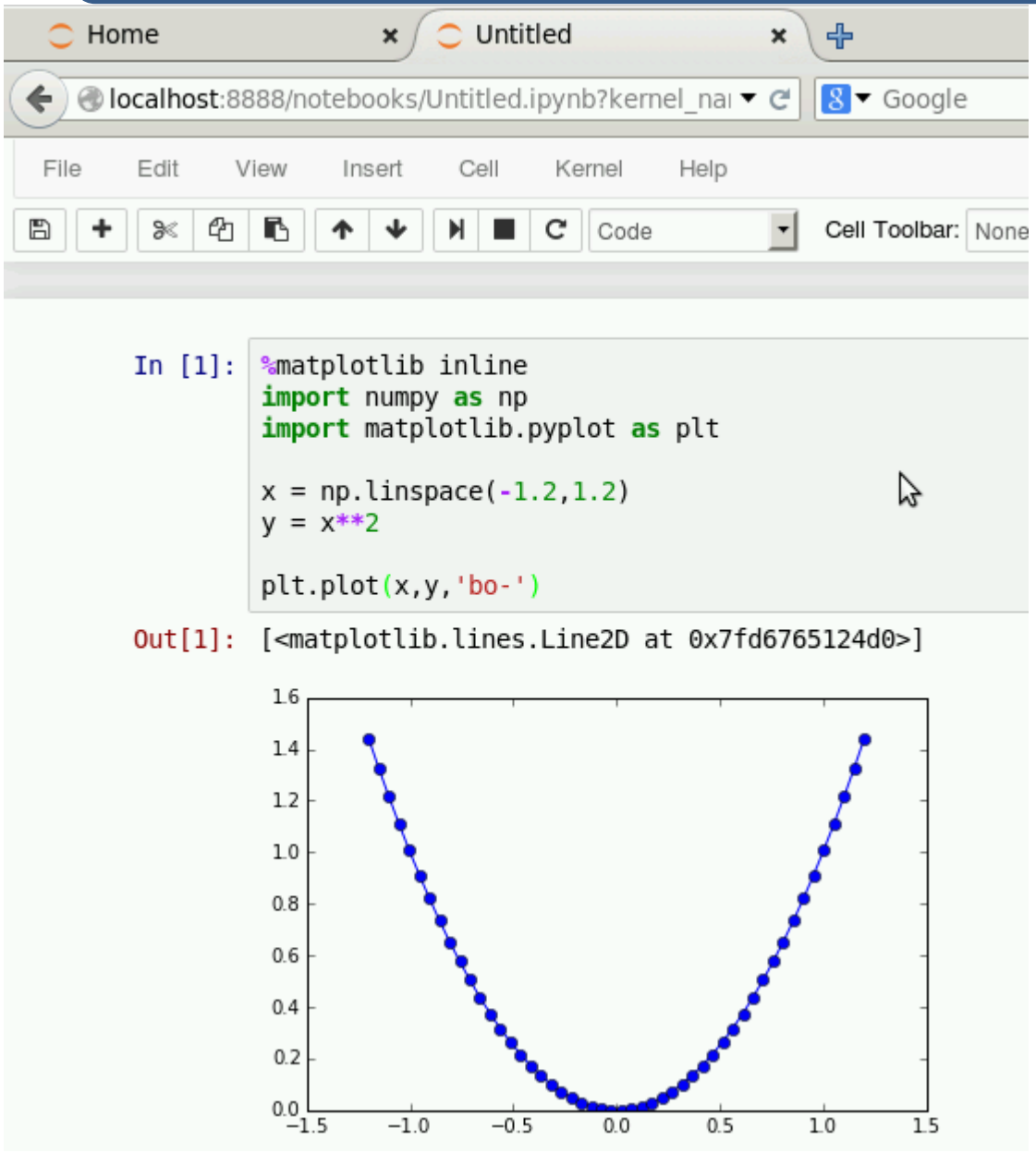
図をインライン表示

定番のモジュールimport

linspace は linear spacing で配列を生成
(デフォルトで50点）

pyplot.plt ではMatlab like なプロット指定が可能
この例では青色で、シンボルは〇、データ間は線で繋ぐ、の意

出力



- ブラウザ内に、結果が inline 表示される。
- 次のセルで変数プロットや、前のセルに戻って実行も可能

他のサンプルに行く前に NumPy について説明

- NumPy : Numerical Python
  - http://www.numpy.org/
- 特徴
  - ndarray （n-dimentional array) を基本とした多次元配列
    - 高速演算（ベクトル、スカラー）
  - Broadcast機能（サイズの異なる配列同士の演算）
  - ループを記載しない配列計算
  - 行列演算、乱数、フーリエ変換など、基礎的な数学演算
  - C/C++, Fortran へのインターフェース
- IPython Notebook や scipy, pylab, pandas など、各種データ解析をする際にはこのnumpyが基本となっている。
- 典型的には import numpy as np としてから使用する。
- Matlab/Octave ユーザであれば
  - どうしても Matlab like に使いたいなら from pylab import *
  - Numpy for matlab users というページもあるので関数を対応させながら見ると楽。

ndarray

```
In [1]: import numpy as np
In [2]: arr = np.array([[1,2,3],[4,5,6]])
In [3]: arr
Out[3]:
array([[1, 2, 3],
       [4, 5, 6]])
In [4]: whos
Variable   Type        Data/Info
-------------------------------
arr        ndarray     2x3: 6 elems, type `int64`, 48 bytes
np         module      <module 'numpy' from '/ce<...>ages/numpy/__init__.pyc'>
In [5]: arr*3   ← 各要素の定数倍
Out[5]:
array([[ 3,  6,  9],
       [12, 15, 18]])
In [6]: arr[0]   ← 配列要素へのアクセス。
Out[6]: array([1, 2, 3])
In [7]: arr[0][1:] ← 配列要素へのアクセス。スライス表記
Out[7]: array([2, 3])
In [10]: arr2 = np.arange(12) ← arangeで連続する数値配列を生成。
In [11]: arr2
Out[11]: array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11])
In [12]: arr2.reshape(3,4)   ← 配列形状の変換。
Out[12]:
array([[ 0,  1,  2,  3],
       [ 4,  5,  6,  7],
       [ 8,  9, 10, 11]])
```

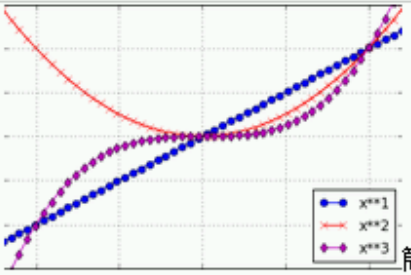
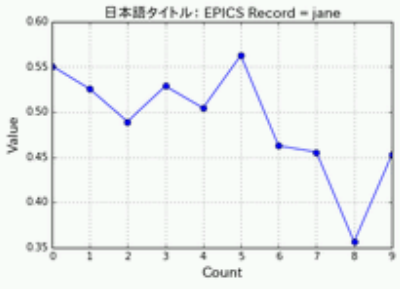
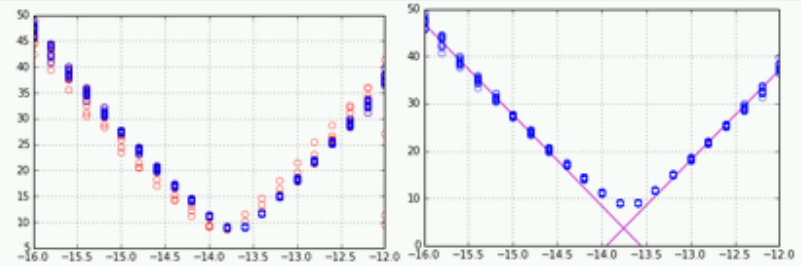
型推定

その他のサンプル（KEK内のみ）

- http://pfconrg07.kek.jp:8082/trac/cerl/wiki/control/Soft/IPython_Example

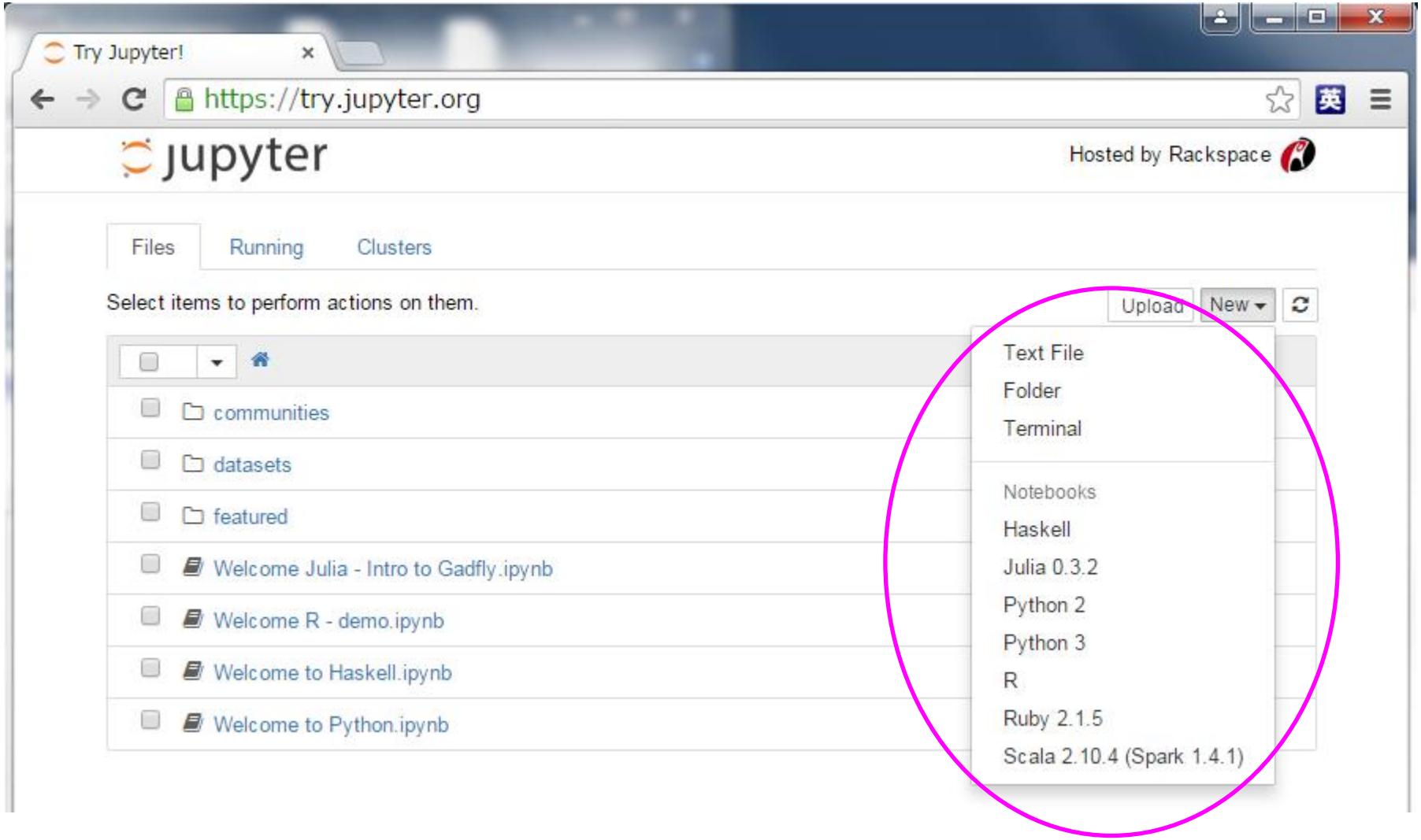
IPython Notebook Example

notebookファイルは右クリック→「対象のファイルを保存」してください
IPython Notebook の起動方法や設定については IPython Notebook について を参照してください。

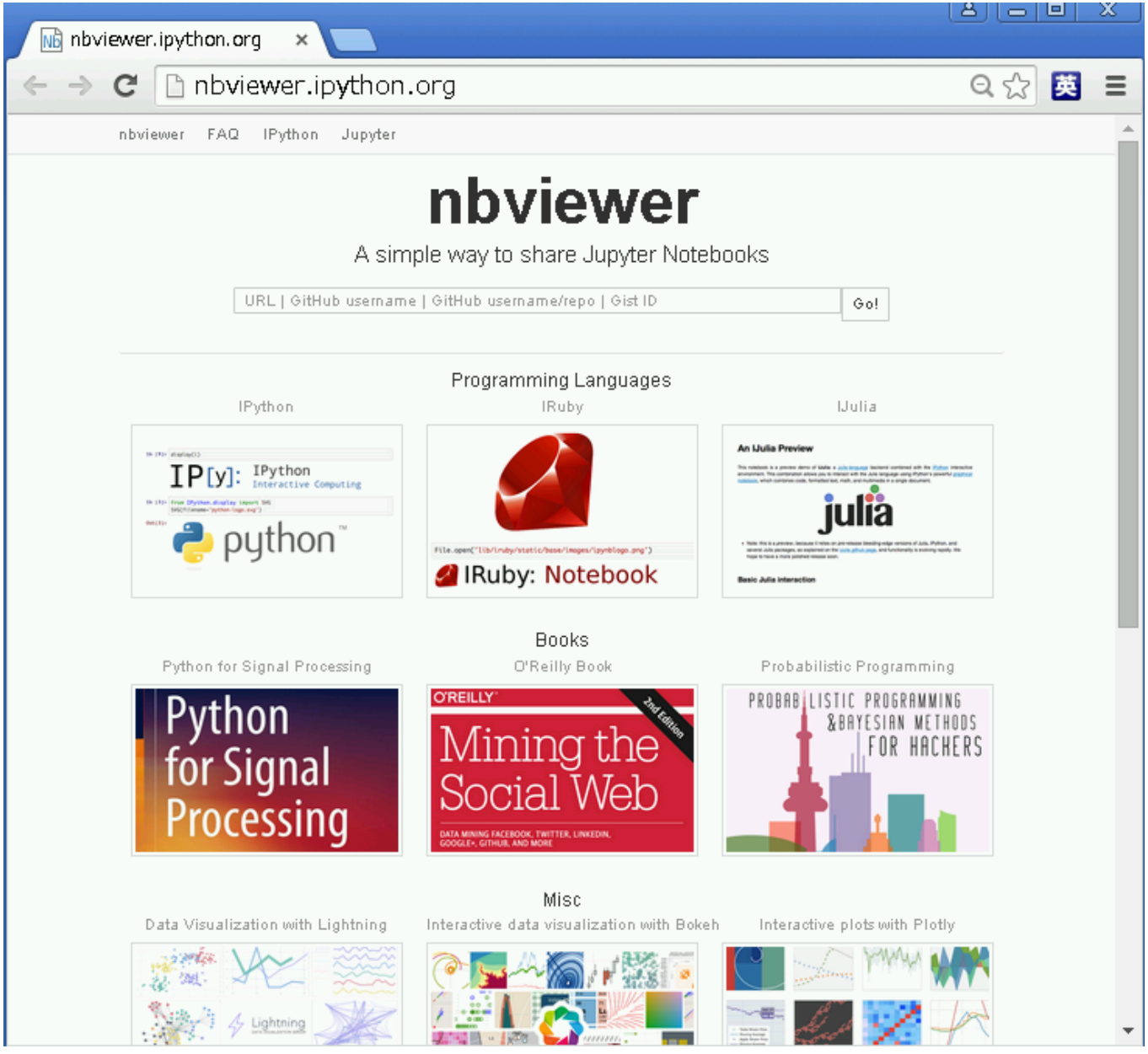
web	notebookファイル	コメント
example1.html	00example.ipynb	 簡単なプロット例：y=x, y=x^2, y=x^3
ca1.html	ca1.ipynb	 EPICS Channel Access 例、日本語のラベル例
Qscan.html	Qscan.ipynb	 データ除外、QScan測定例(線形フィット例)

Notebook環境 Jupyter は多言語対応

- http://jupyter.org/
- 試してみるなら： try.jupyter.org



膨大なサンプル


nbviewer.ipython.org
nbviewer.ipython.org
nbviewer FAQ IPython Jupyter
nbviewer
A simple way to share Jupyter Notebooks
URL | GitHub username | GitHub username/repo | Gist ID
Go!
Programming Languages
IPython
IRuby
IJulia
IPython Interactive Computing
python
IRuby: Notebook
An IJulia Preview
julia
Basic Julia interaction
Books
Python for Signal Processing
O'Reilly Book
Probabilistic Programming
Python for Signal Processing
O'REILLY
2nd Edition
Mining the Social Web
DATA MINING FACEBOOK, TWITTER, LINKEDIN, GOOGLE+, GITHUB, AND MORE
PROBABILISTIC PROGRAMMING & BAYESIAN METHODS FOR HACKERS
Misc
Data Visualization with Lightning
Interactive data visualization with Bokeh
Interactive plots with Plotly
Lightning

cERLでのセットアップ記録

- http://pfconrg07.kek.jp:8082/trac/cerl/wiki/control/Soft/IPython_setup
  - ＫＥＫ内からのみアクセス可

- Python自体のコンパイルも含めて実施した
  - Tcl/Tk
  - Python
  - pip
  - ATLAS/LAPACK
  - Numpy
  - Scilab, matplotlib
  - Pandas, Pillow(PIL後継), Sympy
  - EPICS
  - PyVISA

cERL環境での使用計画

- まだ試験段階。みなさんへの広報をし始めたところ

- ipywidgetsは(いまのところ）そんなに魅力は感じていない
  - GUIが使いたいなら CSS 使っとけ
    （いずれ手のひらを返すかもしれませんが...）

- 使用形態：セキュリティ問題もあり、悩ましいところ
  - サーバー計算機へログイン + X端末
  - Client上のwebブラウザ + SSH 転送
  - Client上のwebブラウザ + パスワード認証

  - 常時エンジンを起動しておくべきか
  - リモートマシンにログインしている状態と同じなので、何でもできる
    - 利点でもありセキュリティ上は欠点でもあり

おまけ

Pandas

- Python Data Analysis Library
  - http://pandas.pydata.org/
- “Data Frame”という考え方をメインにしている
  - どちらかというと統計処理向け（Rの代替で使っている人多数）
  - Time Series データは扱いやすい
    - resample も出来る
  - stdev, mean, sum などの統計処理はもちろん可能
- データが無い列の処理
  - NaN 処理が楽
  - データを drop / fill 可能

Pandas Example from '10 minutes to pandas'

http://pandas.pydata.org/pandas-docs/stable/10min.html

```
In [1]: import pandas as pd
        import numpy as np
        import matplotlib.pyplot as plt

In [2]: s = pd.Series([1,3,5,np.nan,6,8])

In [3]: s

Out[3]: 0     1
        1     3
        2     5
        3   NaN
        4     6
        5     8
        dtype: float64

In [4]: dates = pd.date_range('20130101', periods=6)
```

Pandas DataFrame

```
In [4]: dates = pd.date_range('20130101', periods=6)

In [5]: dates
Out[5]: DatetimeIndex(['2013-01-01', '2013-01-02', '2013-01-03', '2013-01-04',
                       '2013-01-05', '2013-01-06'],
                      dtype='datetime64[ns]', freq='D', tz=None)

In [6]: df = pd.DataFrame(np.random.randn(6,4), index=dates, columns=list('ABCD'))

In [8]: df
```

Out[8]:

	A	B	C	D
2013-01-01	-1.089780	-1.146133	-0.441667	0.714819
2013-01-02	-0.159744	0.535293	0.413670	0.034012
2013-01-03	0.358351	-0.229831	0.544658	0.510868
2013-01-04	0.154386	1.640050	-1.999353	0.937615
2013-01-05	0.168758	-0.847718	0.609653	0.722332
2013-01-06	-0.507637	-2.573859	0.676223	1.015939

```
In [9]: df[df>0]
```

Out[9]:

	A	B	C	D
2013-01-01	NaN	NaN	NaN	0.714819
2013-01-02	NaN	0.535293	0.413670	0.034012
2013-01-03	0.358351	NaN	0.544658	0.510868
2013-01-04	0.154386	1.640050	NaN	0.937615
2013-01-05	0.168758	NaN	0.609653	0.722332
2013-01-06	NaN	NaN	0.676223	1.015939

```
In [10]: df[df.A>0]
```

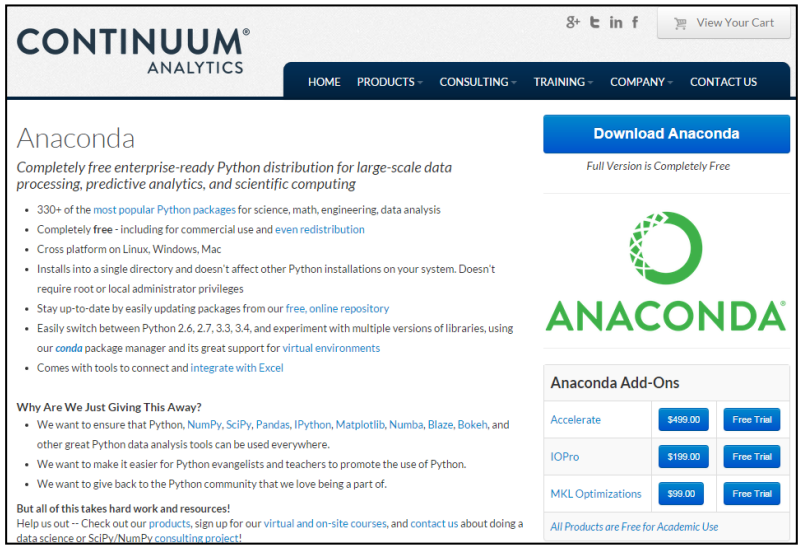
Out[10]:

	A	B	C	D
2013-01-03	0.358351	-0.229831	0.544658	0.510868
2013-01-04	0.154386	1.640050	-1.999353	0.937615
2013-01-05	0.168758	-0.847718	0.609653	0.722332

他の環境

- IDE
  - spyder
- 各種モジュールがパッケージされたもの
  - Anaconda
  - Enthought Canopy
  - pythonxy

EPICS CA が無しで良いならば、お手軽
（追加インストールは不可能ではないが…）



ipython nbconvert

- notebook converter コマンド
- --to でhtml, PDF, python script などに変換可能

- これを使ってhtmlに変換し、そのまま web に乗せています。