（一社）日本官能評価学会ワークショップ

ベイズ統計学と官能評価

第1回
ベイズ分析・Python・３点識別試験

（日本女子大学）

岡本 安晴

ベイズ分析とは

データ分析のレベル

例：　PCR検査

Python入門

感覚の測定とベイズ分析

例：　３点識別試験

データ分析法のレベル

データ：　理論への依存性（cf. Godfrey-Smith, 2021）

データ分布の表現

統計量：　平均値、標準偏差、・・・
表：　度数分布表、連関表、・・・
グラフ：　ヒストグラム、散布図、・・・

母集団と確率モデル

母集団$P(D|\theta)$ ⇄ データD（生成／検定・推測）

モーメント法：　$\hat{\mu} = \frac{1}{N}\sum X_i$　　$X_i \sim N(\mu, \sigma^2), \theta = (\mu, \sigma)$

検定：　帰無仮説　「$\theta = \theta_0$」

最尤法：　最尤推定値 $\hat{\theta}_{\mathrm{ML}} = arg\max_{\theta} P(D|\theta)$

ベイズ分析：　$P(\theta, D) = P(\theta)P(D|\theta)$、$P(\theta|D) = P(\theta, D)/P(D)$

ベイズ分析の基礎式

$P(D;\,\theta)$：　パラメータθをもつデータDの生成確率モデル

$P_0(\theta)$：　パラメータθの確率分布（事前分布）

$P(\theta, D) = P_0(\theta)P(D;\,\theta)$：　同時確率分布

$P(\theta) = \int P(\theta, D)dD$ ：　周辺確率分布

$$= P_0(\theta)\int P(D;\theta)dD = P_0(\theta)$$

$$P(D|\theta) = \frac{P(\theta, D)}{P(\theta)} = \frac{P_0(\theta)P(D;\,\theta)}{P_0(\theta)} = P(D;\,\theta)$$

PCR検査の場合

	陽性	陰性
感染者	P(陽性|感染者) 感度	P(陰性|感染者)
非感染者	P(陽性|非感染者)	P(陰性|非感染者) 特異度

	陽性	陰性
感染者	0.7	0.3
非感染者	0.01	0.99

東京大学保健センターウェブサイト「検査」で用いられていた仮想値
（https://www.hc.u-tokyo.ac.jp/covid-19/tests/, 2021.09）

記述（データをそのまま）

検査結果が陽性 → 陽性です

検査結果が陰性 → 陰性です

モーメント法もどき

パラメータ ⟷ 統計量

パラメータ 感染者／非感染者	⟷	モード 陽性／陰性

	陽性	陰性
パラメータ 感染者	0.7	0.3
非感染者	0.01	0.99

感染者のとき、対応する検査結果は「陽性」
陽性であれば、「感染者である」と推定

非感染者のとき、対応する検査結果は「陰性」
陰性であれば、「非感染者である」と推定

検 定

	陽性	陰性
感染者	0.7	0.3
非感染者	0.01	0.99

帰無仮説「非感染者である」

検査結果が「陽性」のとき

$P(\text{陽性}|\text{帰無仮説}) = 0.01 < 0.05 = 5\%$

有意水準$\alpha = 5\%$で
「帰無仮説（非感染者）」を棄却

p値$= 0.01 = 1\%$

検査結果が「陰性」のとき

$P(\text{陰性}|\text{帰無仮説}) = 0.99 > 0.05 = 5\%$

有意水準$\alpha = 5\%$で
「帰無仮説（非感染者）」を棄却できない

p値$= 0.99 = 99\%$

対立仮説「感染者である」

検定力$P(\text{陽性}|\text{感染者}) = 0.7 = 70\%$

最尤法

	陽性	陰性
感染者	0.7	0.3
非感染者	0.01	0.99

尤度関数$L(\theta|D) = P(D|\theta)$　　θ = (感染者、非感染者)

D = (陽性、陰性)

最尤推定値

陽性のとき　　感染者= $arg\max_{\theta} P(陽性|\theta)$

陰性のとき　　非感染者= $arg\max_{\theta} P(陰性|\theta)$

ベイズ法

	陽性	陰性
感染者 (0.05)	0.7	0.3
非感染者 (0.95)	0.01	0.99

Cf. ステージ３の指標：PCR陽性率5%以上

同時確率$P(\theta, D) = P(\theta)P(D|\theta)$

	陽性	陰性	行和
感染者	$0.7 \times 0.05 = 0.035$	$0.3 \times 0.05 = 0.015$	0.05
非感染者	$0.01 \times 0.95 = 0.0095$	$0.99 \times 0.95 = 0.9405$	0.95
列和$P(D)$	0.0445	0.9555	1

事後確率　　$P(\theta|D) = \frac{P(\theta, D)}{P(D)}$

	感染者	非感染者	行和
陽性	0.7865	0.2135	1.0000
陰性	0.0157	0.9843	1.0000

ベイズ分析はuncertaintyを扱う

A.Gelman, J. Hill, & A. Vehtari (2021). “Regression and Other Stories”

ベイズ分析法の評価

If, in fact, the sample size at the highest level is small, statistical inferences may not be trustworthy, … Bayesian methods provide a sensible alternative approach in these cases.
S. Raudenbush & A. S. Bryk (2002). “Hierarchical Linear Models, 2nd. Ed.” p. 14.

Bayesian statistics suddenly became fashionable, opening new highways for statistical research. Using MCMC, we can now set up and estimate complicated models that describe and solve problems that could not be with traditional methods.
I. Ntzoufras (2009). “Bayesian Modeling Using WinBUGS” p. 2.

Bayesian modeling has become standard and MCMC is well-understood and trusted.
J. Gill (2015). “Bayesian Methods, 3rd ed.” p. xxv.

基本式と手順

$$P(\theta, D) = P(\theta)P(D|\theta)$$

同時確率　　事前分布　　データ生成モデル
尤度

事後分布

$$P(\theta|D) = \frac{P(\theta, D)}{P(D)} = \frac{P(\theta)P(D|\theta)}{P(D)}$$

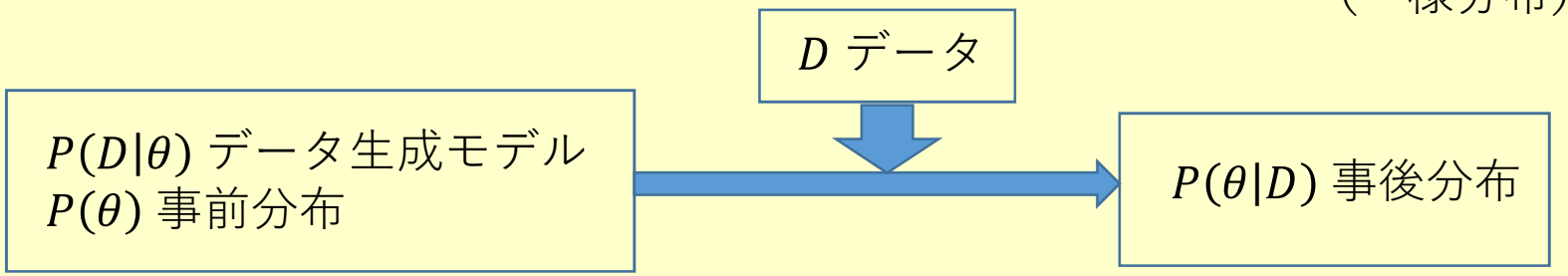
Bayes' rule

$$\propto P(\theta)P(D|\theta)$$

$P(D)$　定数

$$\approx P(D|\theta)$$

$P(\theta) \approx$ 定数のとき
（一様分布）



ベイズ分析の方法

数学的に解析：
　第１回

共役事前分布など、数学的解析が可能であるという制約がある。

グリッド法：
　第１回

パラメータ数が１～２個と少ない場合。
事後分布が領域の端に偏っている場合、その様子がよく分かる。

MCMC (Markov chain Monte Carlo)
サンプリング：
　第２，３回

Stan（２回目で説明）などソフトウェアにより、幅広い範囲のモデルが扱える。

数学的に解析：２項分布のベイズ分析

成功確率がθ、失敗確率が$1-\theta$である N回の独立な試行において、
成功試行数がk試行、失敗試行数が$N-k$試行

2項分布 $$P(k|N,\theta) = {}_NC_k\theta^k(1-\theta)^{N-k}$$

$$P(\theta|D) \propto P(\theta)P(D|\theta)$$

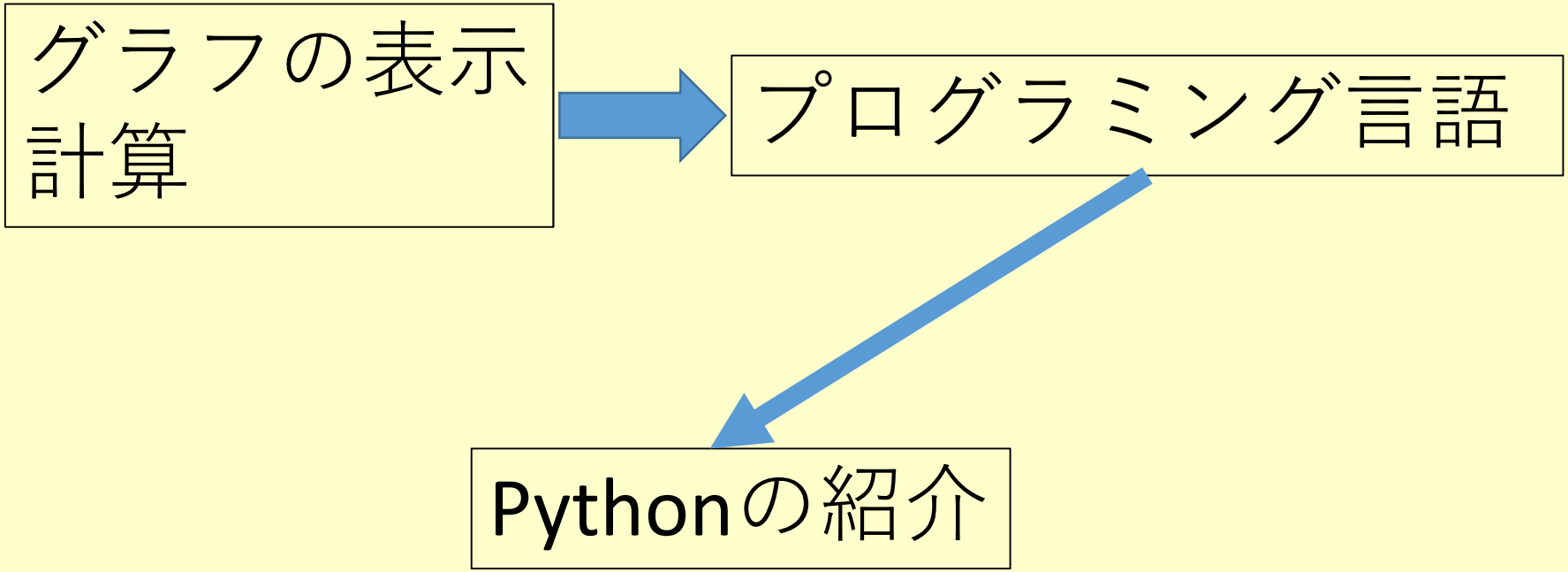
$$P(\theta) = \frac{1}{B(\alpha,\beta)}\theta^{\alpha-1}(1-\theta)^{\beta-1} = Beta(\alpha,\beta)$$ ベータ分布

$$P(\theta|D) \propto P(\theta)P(k|N,\theta) \propto \theta^{\alpha+k-1}(1-\theta)^{\beta+N-k-1}$$

$$\propto Beta(\alpha+k,\beta+N-k)$$

同じ関数族

$P(\theta)$は、$P(D|\theta)$に対して共役（conjugate）である


グラフの表示
計算
プログラミング言語
Pythonの紹介

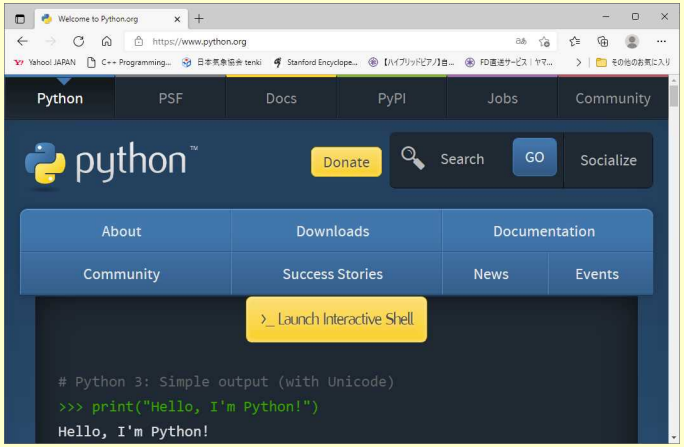
インストールの方法についてのウェブサイト：

http://y-okamoto-psy1949.la.coocan.jp/Python/tips/UsingPython/

本講習会用に簡単にインストールする場合：

http://y-okamoto-psy1949.la.coocan.jp/booksetc/Lec2022BysnPM/SimpleInstall/

Pythonのインストール

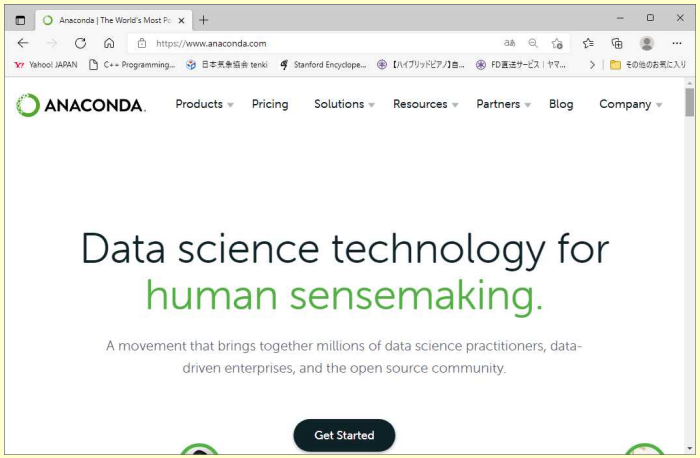


https://www.python.org/

最新版がインストールできる

ライブラリのバージョンの
選択などに注意

無料



https://www.anaconda.com/

複数のPython環境が用意できる

Jupyter notebookが使える

個人無料

その他いろいろ

Anaconda

https://www.anaconda.com/products/individual
(2021.12)

Pythonのバージョンの選択

https://www.python.org/　からダウンロードする場合

現在（2021.12）の最新版Python 3.10より１つ前Python 3.9が対応しているライブラリも多く、安定している

https://www.python.org/downloads/　　(2021.12)

Looking for a specific release?

Python releases by version number:

Release version	Release date
Python 3.10.1	Dec. 6, 2021
Python 3.9.9	Nov. 15, 2021
Python 3.9.8	Nov. 5, 2021
Python 3.10.0	Oct. 4, 2021

Pythonのバージョン
Anacondaの場合

現在（2021.12）、Python 3.9で一応、大丈夫なようである

conda create –n py39 python=3.9

多くのライブラリの使用を期待するなら、Python 3.8が安全である

conda create –n py38 python=3.8

Pythonの使い方

インタラクティブモード

文単位で実行

プロンプト「>>>」の右側に入力した文が
Enterキーを押すと実行される。

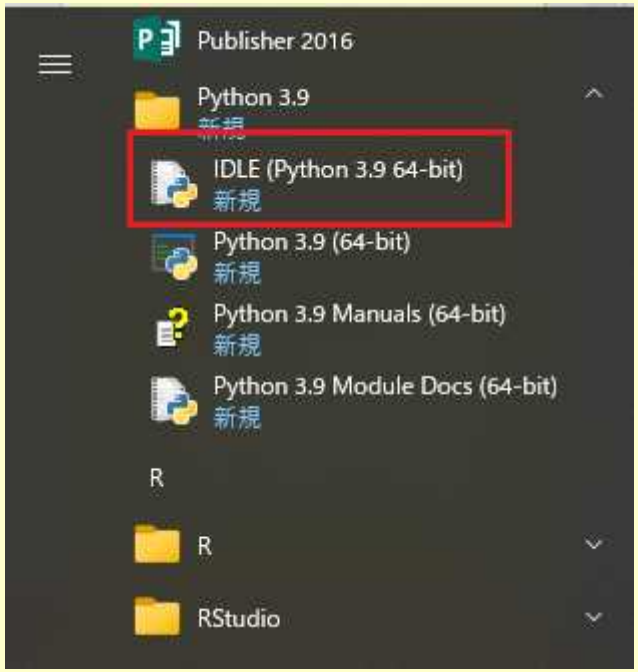
スクリプトモード

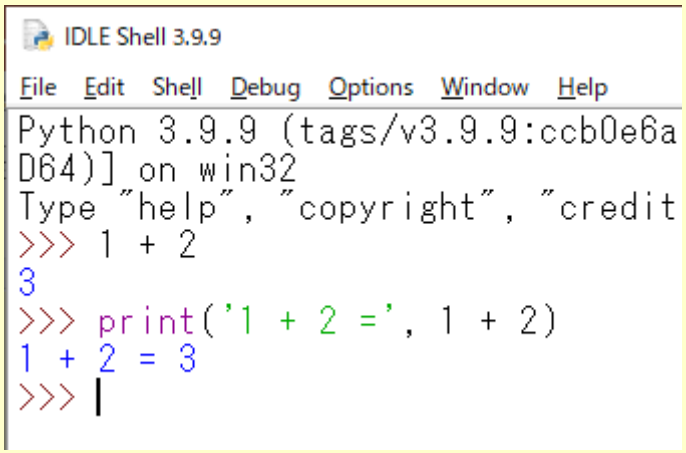
スクリプトファイルの実行

ファイル内のスクリプトが実行される。

Jupyter Notebook

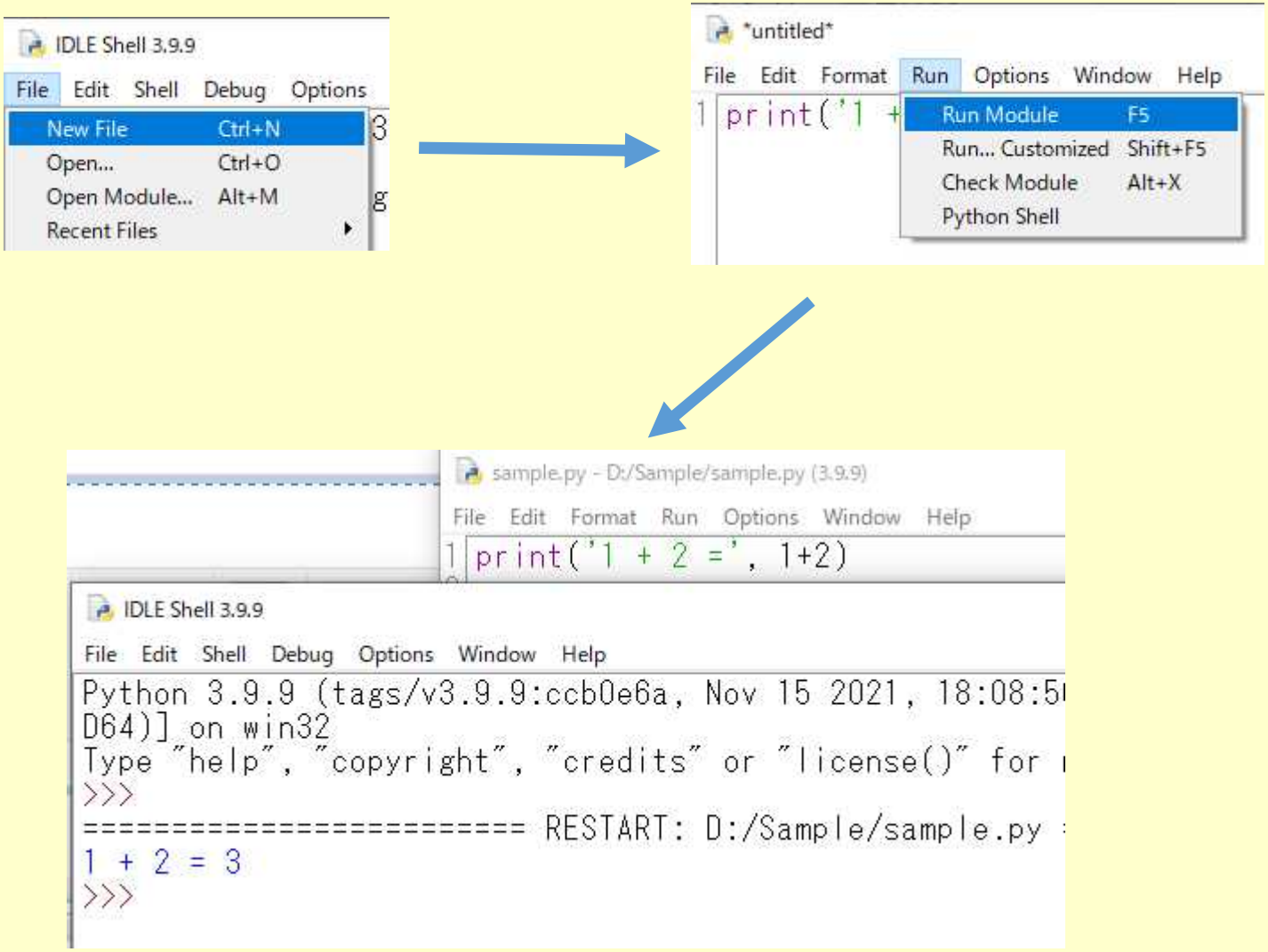
スクリプトを分割したセル単位で実行





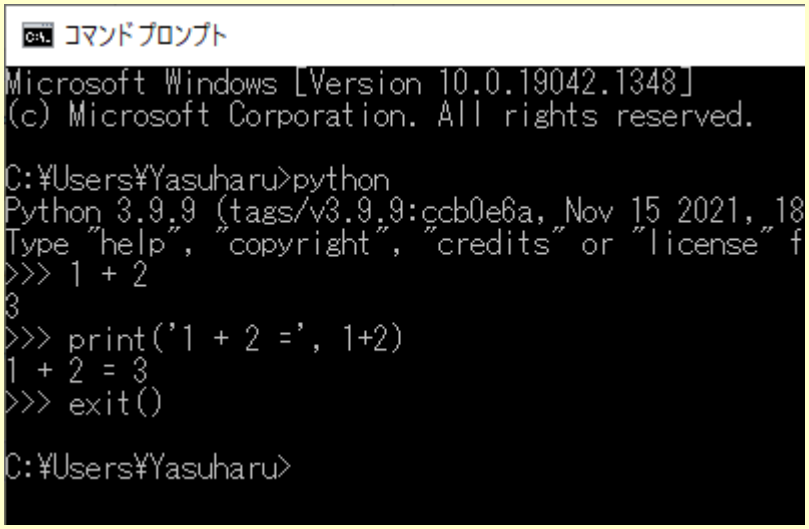
インタラクティブモード

スクリプトモード


IDLE Shell 3.9.9
File Edit Shell Debug Options
New File Ctrl+N
Open... Ctrl+O
Open Module... Alt+M
Recent Files
untitled
File Edit Format Run Options Window Help
print('1 +
Run Module F5
Run... Customized Shift+F5
Check Module Alt+X
Python Shell
sample.py - D:/Sample/sample.py (3.9.9)
File Edit Format Run Options Window Help
print('1 + 2 =', 1+2)
IDLE Shell 3.9.9
File Edit Shell Debug Options Window Help
Python 3.9.9 (tags/v3.9.9:ccb0e6a, Nov 15 2021, 18:08:5
D64)] on win32
Type "help", "copyright", "credits" or "license()" for
>>>
========== RESTART: D:/Sample/sample.py
1 + 2 = 3
>>>

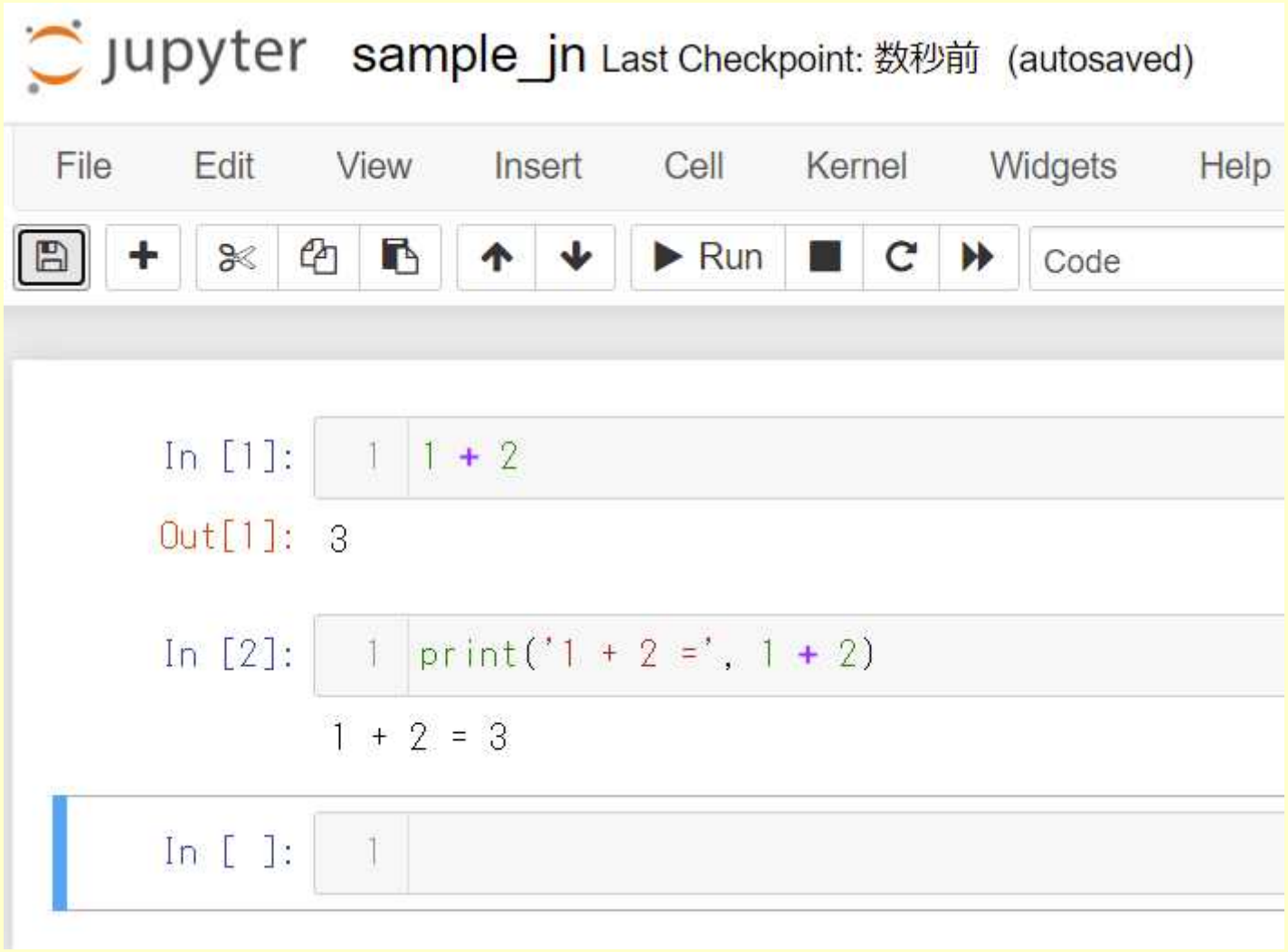
```
PS D:¥Sample> cat sample.py
print('1 + 2 =', 1+2)
PS D:¥Sample> python sample.py
1 + 2 = 3
PS D:¥Sample>
```

スクリプトモード



インタラクティブモード

Jupyter notebook



名前／識別子

文字（a～z、A～Z）、下線（_）で始まり、
0 個以上の文字、下線、数字（0～9）が続く並び。
大文字と小文字は区別される。

データの基本型

整数	ピリオドを含まない形式で表される。
実数	ピリオドを含む形式で表される。
文字列	「’」、「”」で挟む。 1 行以内。
	「”””」で挟む。複数行に亘ることができる。
	間に何も文字がない「””」などは空の文字列である。

整数型と実数型の例

```
>>> a = 1
>>> b = 2
>>> a+b
3
>>> a = 1.0
>>> b = 2.
>>> c = .3
>>> a+b+c
3.3
>>>
```

```
>>> 1 + 2
3
>>> 1 - 2
-1
>>> 2 * 3
6
>>> 5 // 2
2
>>> 5 / 2
2.5
>>> 1.0 + 3.0
4.0
>>> 5.0 - 2.0
3.0
>>> 2.0 * 3.0
6.0
>>> 5.0 / 3.0
1.6666666666666667
>>> 5.0 // 3.0
1.0
>>> 5.0 % 3.0
2.0
>>> 5 % 3
2
>>>
```

文字列型の例

```
>>> x = 'A'
>>> y = "BCD"
>>> z = """123"""
>>> x+y+z
'ABCD123'
>>>
```

```
>>> s = 'abc'
>>> s[0]
'a'
>>> s[2]
'c'
>>>
```

入力input
数字の入力は文字列として扱われる

```
>>> a = input('a = ')
a = 1
>>> b = input('b = ')
b = 2
>>> a+b
'12'
>>> int(a) + int(b)
3
>>>
```

出力print
指定した出力の後の出力は
endパラメータで指定
endパラメータが無いときは、
end = ‘¥n’と同じ

```
>>> print('abc'); print('xyz')
abc
xyz
>>> print('abc', end = ''); print('xyz')
abcxyz
>>> print('abc', end = '/'); print('xyz')
abc/xyz
>>>
```

タプル
( ) 内に値をコンマ,で区切って
並べる
( ) は無くてもよい
タプル内の値は変えられない

```
>>> a = (1, 2, 3)
>>> a
(1, 2, 3)
>>> b = 1, 2, 3
>>> b
(1, 2, 3)
>>> a[0]
1
>>> b[2]
3
>>> a[0] = 10
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: 'tuple' object does not
support item assignment
>>>
```

リスト
[ ]内に値をコンマ,で区切って
並べる。
リスト内の値は変更できる

```
>>> x = [1, 2, 3]
>>> x[0]
1
>>> x[1] = 10
>>> x
[1, 10, 3]
>>>
```

辞書
{ } 内に、「キー: 値」の組を並べる
キーの値を指定することにより、対応する値が読み出せる
キーに対応する位置の値を変更することもできる

```
>>> d = {1: 'A', 2: 'B', 3: 'C'}
>>> d
{1: 'A', 2: 'B', 3: 'C'}
>>> d[2]
'B'
>>> d[3] = 100
>>> d
{1: 'A', 2: 'B', 3: 100}
>>>
```

for文
for 変数 in イタラブル オブジェクト:
　　スィート（suite）

スィートが複数の文からなるときは
字下げによりまとめる

while文の例、参照

Sample1.py

```
for i in [1, 2, 3]:
    print(i)
```

```
PS V:¥xxxxx¥Lecture_1> python sample1.py
1
2
3
```

while文
while 条件:
　　スィート

Sample2.py

```
i = 1
while i <= 3:
    print(i)
    i += 1

print('OK!')
```

```
PS V:¥xxxxx¥Lecture_1> python sample2.py
1
2
3
OK!
PS V:¥xxxxx¥Lecture_1>
```

break　スィートの外に飛び出す

continue　スィートの最後まで処理を飛ばす

Sample3.py

```
i = 0
while True:
    i += 1
    if i > 15:
        break
    if 3 < i and i < 9:
        continue
    print(i)
print('Good-bye!')
```

```
PS V:¥xxxxx¥Lecture_1> python sample3.py
1
2
3
9
10
11
12
13
14
15
Good-bye!
PS V:¥xxxxx¥Lecture_1>
```

ライブラリのインストール

pipコマンドの場合

pip install ライブラリパッケージ名の並び

PS V:¥xxxxx¥Lecture_1> pip install numpy matplotlib scipy seaborn

condaコマンドの場合

conda install ライブライパッケージ名の並び

(base) PS V:¥xxxxx¥Lecture_1> conda install numpy matplotlib scipy seaborn

リストとnumpyの配列ndarray型（n次元配列）

```
>>> a = [1,2,3]
>>> a * 3
[1, 2, 3, 1, 2, 3, 1, 2, 3]
>>> b = [5,6,7]
>>> a + b
[1, 2, 3, 5, 6, 7]
>>> import numpy
>>> x = numpy.array([1,2,3])
>>> x * 3
array([3, 6, 9])
>>> y = numpy.array([5,6,7])
>>> x + y
array([ 6,  8, 10])
>>> import numpy as np
>>> np.array([10,20,30]) * 2
array([20, 40, 60])
```

リスト

Ndarray型配列

講習者のサンプルスクリプト

http://y-okamoto-psy1949.la.coocan.jp/Python/sampleprgs/SmplStat/

http://y-okamoto-psy1949.la.coocan.jp/Python/sampleprgs/DrawGraphs/

http://y-okamoto-psy1949.la.coocan.jp/Python/sampleprgs/KdeBoxViolin/

岡本安晴「いまさら聞けないPythonでデータ分析」丸善出版

計量心理学、心理物理学関係のPythonスクリプトは、拙著
岡本安晴「心理学データ分析と測定」勁草書房
のウェブサイト
http://y-okamoto-psy1949.la.coocan.jp/booksetc/pm2010/
に用意している

Pythonのドキュメント

英語

https://docs.python.org/3/

日本語

https://docs.python.org/ja/3/

数学的に解析：２項分布のベイズ分析

成功確率がθ、失敗確率が$1-\theta$である N回の独立な試行において、
成功試行数がk試行、失敗試行数が$N-k$試行

2項分布 $$P(k|N,\theta) = {}_NC_k\theta^k(1-\theta)^{N-k}$$

$$P(\theta|D) \propto P(\theta)P(D|\theta)$$

$$P(\theta) = \frac{1}{B(\alpha,\beta)}\theta^{\alpha-1}(1-\theta)^{\beta-1} = Beta(\alpha,\beta)$$ ベータ分布

$$P(\theta|D) \propto P(\theta)P(k|N,\theta) \propto \theta^{\alpha+k-1}(1-\theta)^{\beta+N-k-1}$$

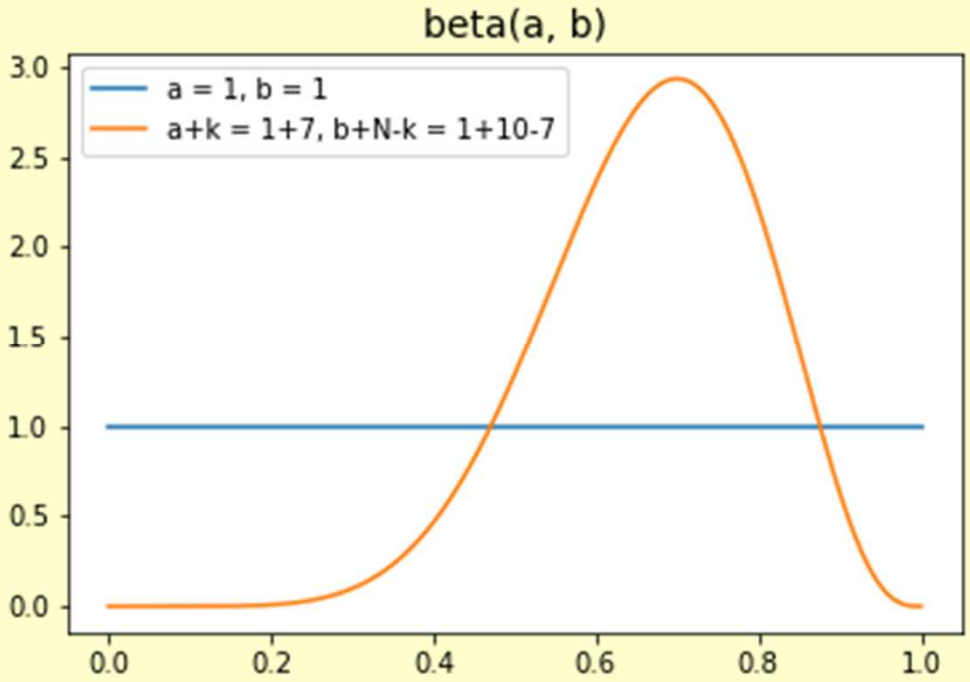
$$\propto Beta(\alpha+k,\beta+N-k)$$

同じ関数族

$P(\theta)$は、$P(D|\theta)$に対して共役（conjugate）である

$$P(\theta) = \frac{1}{B(\alpha, \beta)} \theta^{\alpha-1}(1-\theta)^{\beta-1} = Beta(\alpha, \beta)$$

```
import numpy as np
import matplotlib.pyplot as plt
import scipy.stats as ss

x_thetas = np.linspace(0.0, 1.0, 1001)
print(x_thetas[0], x_thetas[-1])
a = 1.0; b = 1.0
y_prior = ss.beta.pdf(x_thetas, a, b)
plt.plot(x_thetas, y_prior, label = 'a = 1, b = 1')
k = 7; N = 10
y_post = ss.beta.pdf(x_thetas, a + k, b + N - k)
plt.plot(x_thetas, y_post, label ='a+k = 1+7, b+N-k = 1+10-7')
plt.legend()
plt.title('beta(a, b)', fontsize = 14)
plt.savefig('Betas.png')
plt.show()
```


beta(a, b)
a = 1, b = 1
a+k = 1+7, b+N-k = 1+10-7
3.0
2.5
2.0
1.5
1.0
0.5
0.0
0.0
0.2
0.4
0.6
0.8
1.0

3 点識別試験（Triangular Test）

3AFC Oddity Task

心理物理学

物
社会、文化、行為、主張

→

感覚

Mechanism
Process

計量心理学

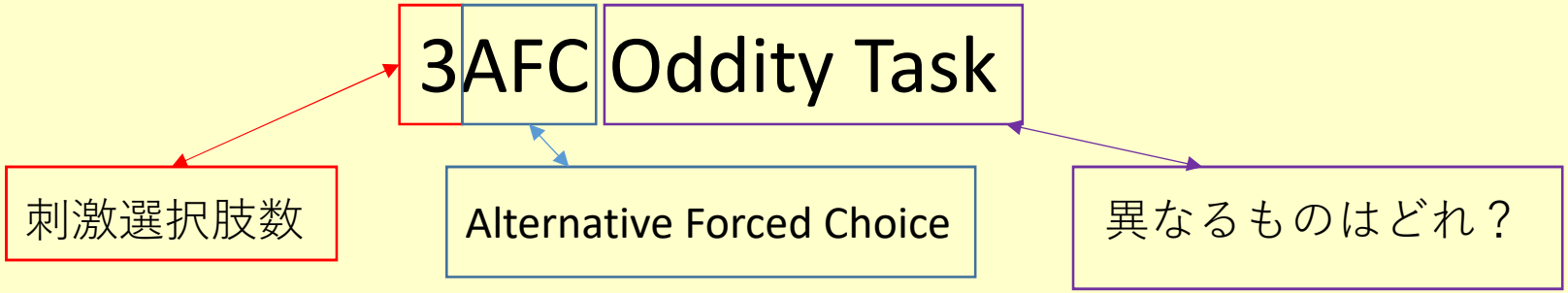
こころ
感覚、心理学的特性

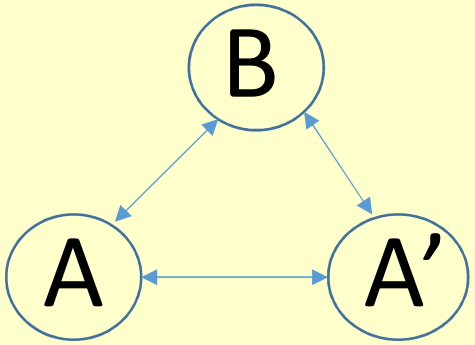
→

測定

Measurement
Representation by mathematics

3 点識別試験（Triangular test/method）





異なるものはどれ？

物理的には　A=A’≠B

3 点識別試験（Triangular Test）

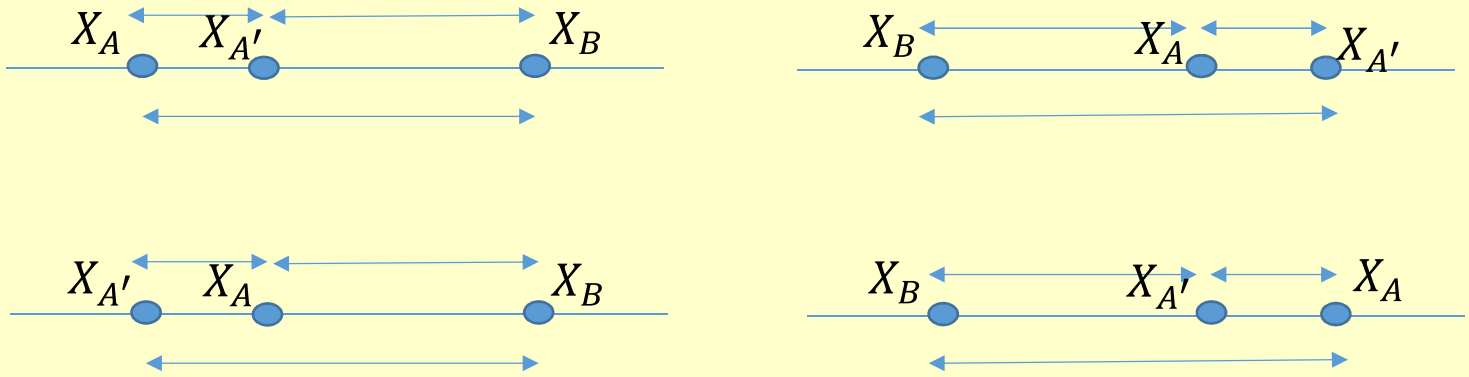
3AFC Oddity Task

２つの同じ刺激AとA'、および少し異なる刺激Bの３つの刺激を提示
他の２つと異なる１つを選ぶ

A: $X_A \sim N(\mu, \sigma^2)$　　A': $X_{A'} \sim N(\mu, \sigma^2)$　　B: $X_B \sim N(\mu + d', \sigma^2)$

Bが選ばれるとき

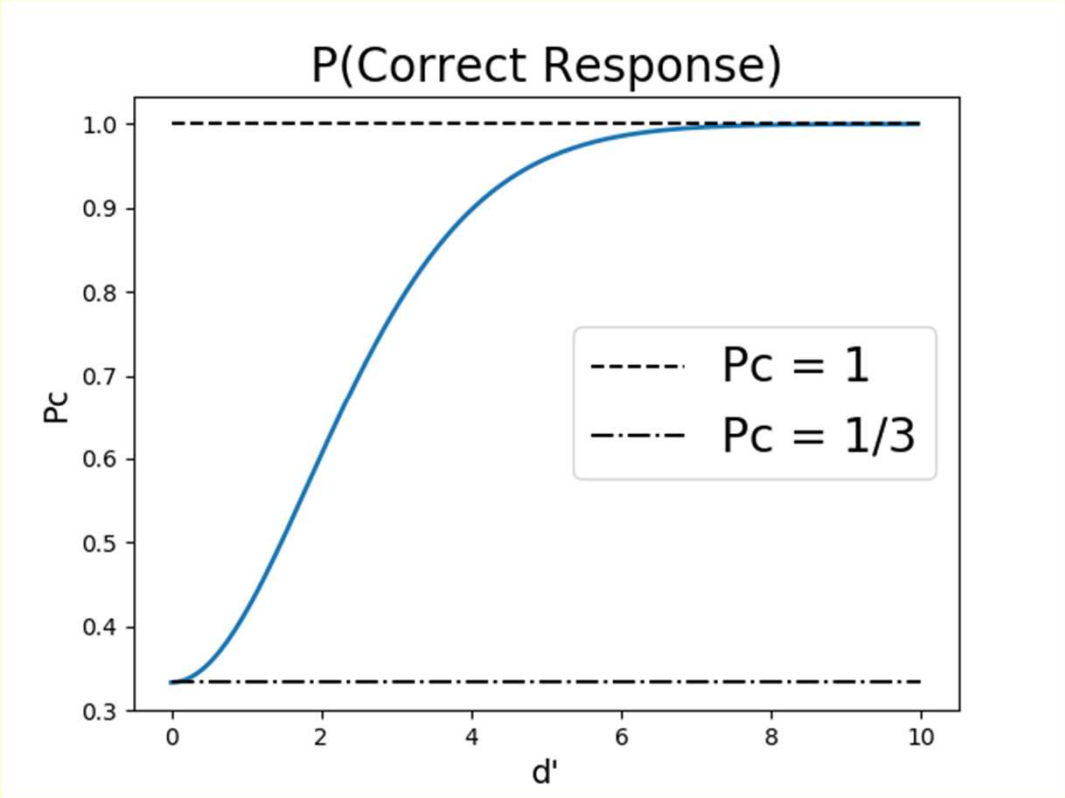
$$|X_A - X_{A'}| < |X_A - X_B| \quad \text{かつ} \quad |X_A - X_{A'}| < |X_{A'} - X_B|$$



正しくBが選ばれる確率P_C

Frijters et al. (1980). Perception & Psychophysics, 27, 176-178.

$$Pc = 2\int_0^{\infty}\left[\Phi\left(-u\sqrt{3}+d'\sqrt{2/3}\right)+\Phi\left(-u\sqrt{3}-d'\sqrt{2/3}\right)\right]e^{-1/2\cdot u^2}/\sqrt{2\pi}\,du$$



```
import numpy as np; import scipy.stats as ss; import scipy.integrate as si
import scipy.optimize as so; import matplotlib.pyplot as plt
CumPhi = ss.norm.cdf

class k_Pc:
    def __init__(self, d):
        self.d = d
    def f(self,u):
        v = (CumPhi(-u*(3**0.5) + self.d * ((2/3)**0.5))
             + CumPhi(-u*(3**0.5) - self.d * ((2/3)**0.5))) ¥
            * np.exp(-0.5 * (u**2)) / ((2.0*np.pi)**0.5)
        return v

def f_Pc(d):
    int_k_Pc = k_Pc(d);  v = si.quad(int_k_Pc.f, 0.0, +np.inf)
    return v[0] * 2.0

plt.title('P(Correct Response)', fontsize = 20)
dvalues = np.arange(0.0, 10.0, 0.05)
PcValues = [f_Pc(d) for d in dvalues]
plt.plot(dvalues, PcValues, linewidth = 2)
plt.plot([0.0, 10.0], [1.0, 1.0], 'k--', label = 'Pc = 1')
plt.plot([0.0, 10.0,], [1/3, 1/3], 'k-.', label = 'Pc = 1/3')
plt.xlabel("d'", fontsize = 14)
plt.ylabel('Pc', fontsize = 14)
plt.legend(fontsize = 20); plt.savefig('FigTriPF.png'); plt.show()
```

パラメータの推定

データ数 N　　正答数 k　　正答確率 P_C

二項分布　　$P(k|P_C) = \binom{N}{k} {P_C}^k (1-P_C)^{N-k}$

信号検出理論：パラメータd'（感覚差）

$$Pc(d') = 2\int_0^\infty \left[\Phi\left(-u\sqrt{3} + d'\sqrt{2/3}\right) + \Phi\left(-u\sqrt{3} - d'\sqrt{2/3}\right)\right] e^{-1/2\cdot u^2}/\sqrt{2\pi}\, du$$

$$d' \Rightarrow Pc \Rightarrow P(k|Pc) = P(k|d')$$

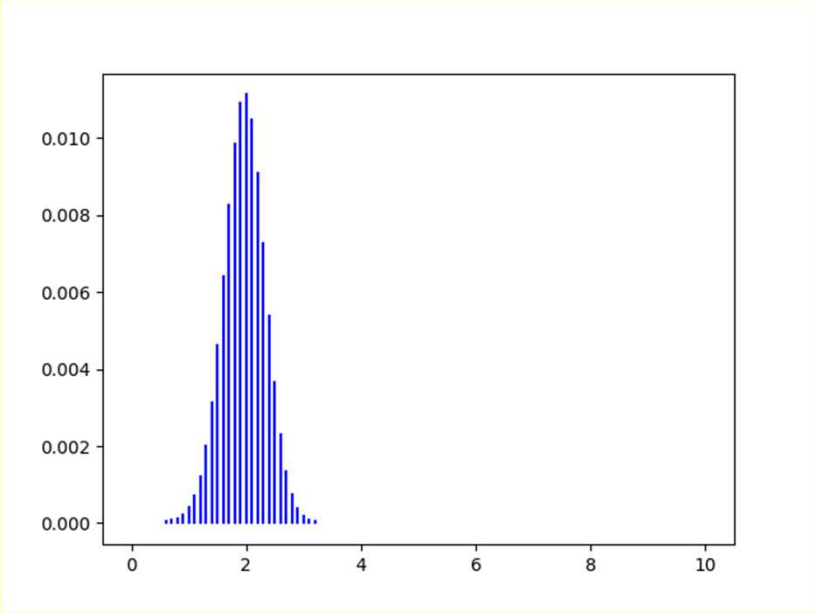
事後分布　　$P(d'|k) \propto P_0(d')P(k|d')$

```
import numpy as np;    import scipy.integrate as si
import scipy.stats as ss;    import matplotlib.pyplot as plt
N = 50; k = 30
class Triangular:
    def __init__(self, d_prime):
        self.d_prime = d_prime
    def func(self, u):
        v = ss.norm.cdf(-u * (3.0**0.5) + self.d_prime * ((2/3)**0.5)) + ¥
              ss.norm.cdf(-u * (3.0**0.5) - self.d_prime * ((2/3)**0.5))
        return v * np.exp(-0.5 * (u**2)) / ((2.0 * np.pi)**0.5)

def Pc_d_prime(d_prime):
    d_func = Triangular(d_prime).func
    return 2 * si.quad(d_func, 0.0, +np.inf)[0]

dprimes = np.linspace(0, 10, 1001)
postPdprime = np.empty(len(dprimes))
for i in range(len(dprimes)):
    if i % 100 == 0:
        print('{0:}/{1:}     '.format(i, len(dprimes)))  #, end = '¥r')
    Pc = Pc_d_prime(dprimes[i])
    postPdprime[i] = (Pc ** k) * ((1 - Pc) ** (N - k))
postPdprime = (postPdprime / np.sum(postPdprime))

for i in range(len(dprimes)):
    postPdprime[i] = (dprimes[i] ** k) * ((1 - dprimes[i]) ** (N - k))
postPdprime = (postPdprime / np.sum(postPdprime))
```

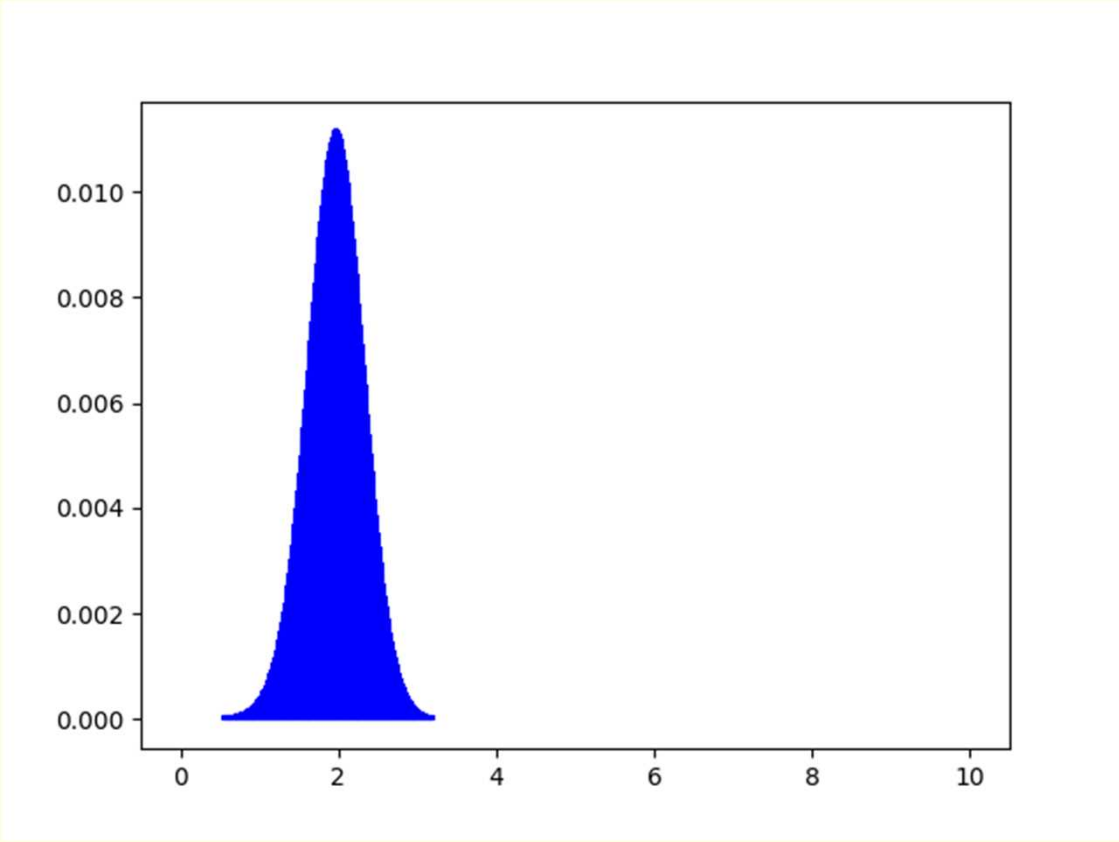
N = 50; k = 30

$$P(d'|k) \propto P_0(d')P(k|d') \qquad P_0(d') = const$$

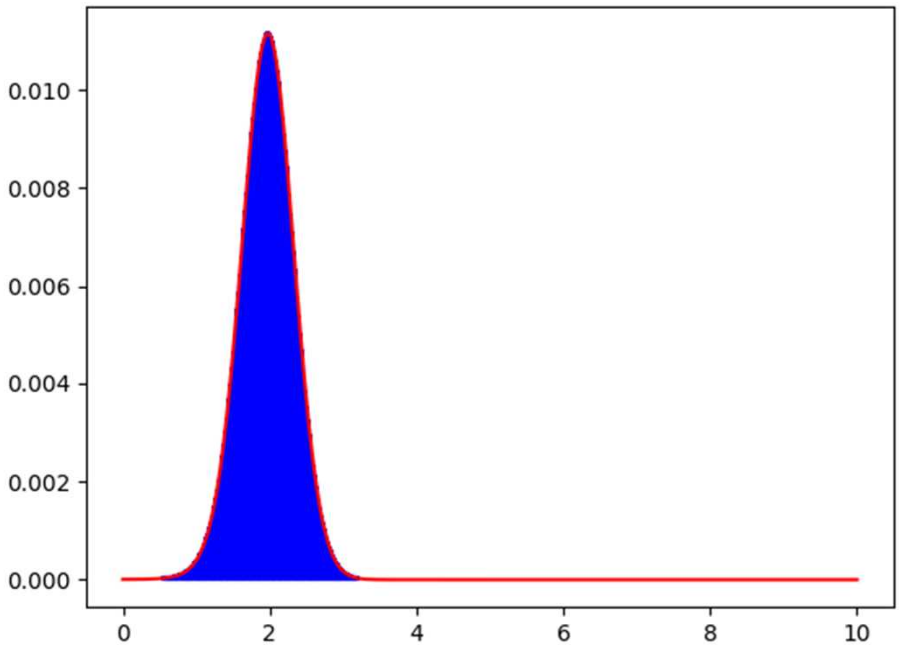
```
for i in range(len(dprimes)):
    if i % 10 == 0:
        plt.plot([dprimes[i], dprimes[i]], [0, postPdprime[i]], c = 'b')
plt.savefig('FigPostD_prime.png')
plt.show()
```



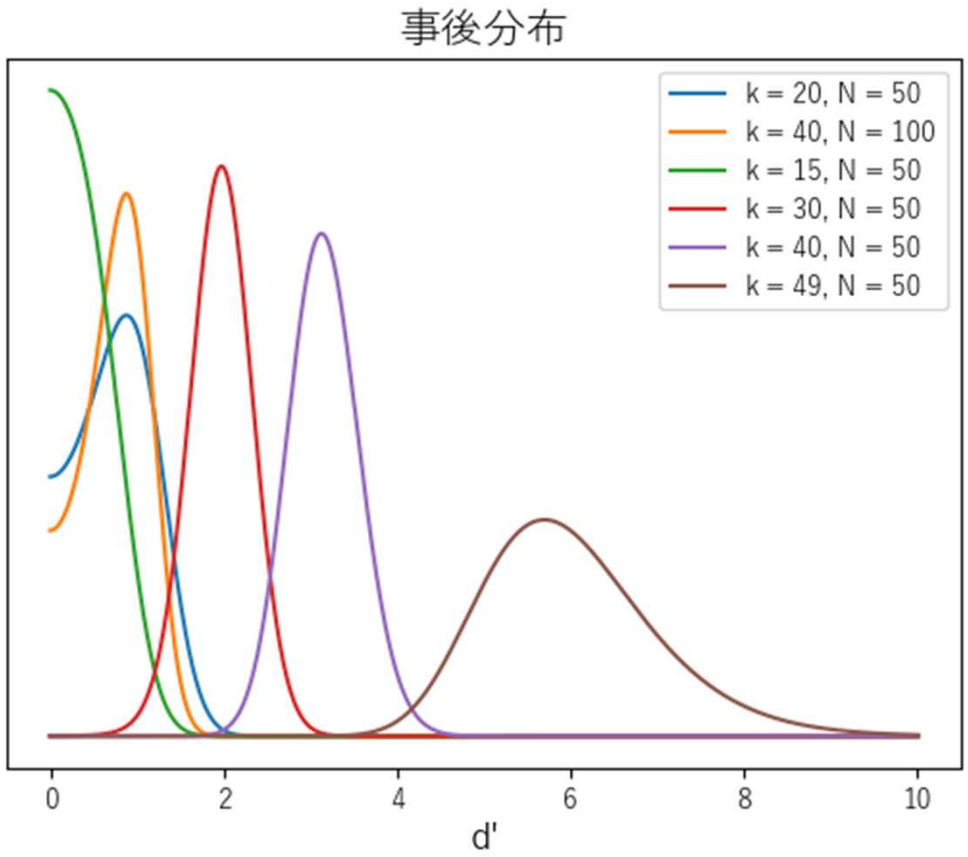
```
for i in range(len(dprimes)):
    plt.plot([dprimes[i], dprimes[i]], [0, postPdprime[i]], c = 'b')
plt.savefig('FigPostD_primeAll.png')
plt.show()
```



```
for i in range(len(dprimes)):
    plt.plot([dprimes[i], dprimes[i]], [0, postPdprime[i]], c = 'b')
plt.plot(dprimes, postPdprime, c = 'r')
plt.savefig('FigPostD_primeLine.png')
plt.show()
```



```
dprimes = np.linspace(0, 10, 1001)
Pc = np.empty(len(dprimes))
for i in range(len(dprimes)):
    if i % 100 == 0:
        print('{0:}/{1:}     '.format(i, len(dprimes)))  #, end = '¥r')
    Pc[i] = Pc_d_prime(dprimes[i])
ks = [20, 40, 15, 30, 40, 49]
Ns = [50, 100, 50, 50, 50, 50]
for k, N in zip(ks, Ns):
    postPdprime = np.empty(len(dprimes))
    for i in range(len(dprimes)):
        postPdprime[i] = (Pc[i] ** k) * ((1 - Pc[i]) ** (N - k))
    postPdprime = (postPdprime / np.sum(postPdprime))
    plt.plot(dprimes, postPdprime, label = f'k = {k}, N = {N}')
plt.legend(); plt.yticks([]); plt.xlabel("d'", fontsize = 14)
plt.title('事後分布', fontsize = 16); plt.savefig('PostMulti.png')
plt.show(); plt.yticks([]); plt.xlabel("d'", fontsize = 14)
plt.title('事後分布', fontsize = 16)
plt.savefig('PostMulti.png'); plt.show()
```


事後分布
k = 20, N = 50
k = 40, N = 100
k = 15, N = 50
k = 30, N = 50
k = 40, N = 50
k = 49, N = 50
0
2
4
6
8
10
d'

参考資料

データと理論

P. Godfrey-Smith (2021). Theory and reality, second edition. The University of Chicago Press.

Python

J. V. Guttag (2016). Introduction to computation and programming Using Python: With application to understanding data, second edition. The MIT Press

ベイズ分析

A. Gelman, J. B. Carlin, H. S. Stern, D. B. Dunson, A. Vehtari, & D. B. Rubin (2014). Bayesian data analysis, third edition. CRC Press. Pp. 6-7.
http://www.stat.columbia.edu/~gelman/book/
for non-commercial purposes

岡本安晴「いまさら聞けないPythonでデータ分析」丸善出版、pp. 172-177.

まとめ

ベイズ分析	パラメータの情報とデータの情報を まとめて分析
Python	データ分析のスクリプトが容易に書ける
スクリプト例	３点識別試験における d’の分析

サンプルスクリプト sample_1.zip のダウンロード・ウェブサイト

http://y-okamoto-psy1949.la.coocan.jp/booksetc/Lec2022BysnPM/

次回　予告

Stan入門

３点識別試験のStanによる分析
ロジスティック回帰分析

第2回までにStanをインストールしておくことが望ましい。
参考資料のウェブサイト
http://y-okamoto-psy1949.la.coocan.jp/booksetc/Lec2022BysnPM/