The IPython Notebook: open, reproducible scientific computing

Matthias Bussonnier[1], Jonathan Frederic[2], Bradley M. Froehle[3], Brian E. Granger[2], Paul Ivanov[3], Thomas Kluyver[3], Fernando Perez[3], Benjamin Ragan-Kelley[3] and Zachary Sailer[2]

[1]Affiliation of Matthias
[2]Cal Poly State University
[3]University of CA, Berkeley

June 30, 2013

Abstract

While computing has become a foundation of all it is challenging for researchers .

1 Introduction

2 The lifecycle of research

3 The IPython Notebook

3.1 Web application

3.2 Notebook document format

3.3 Installation

4 Collaboration

5 Broader ecosystem

6 Future directions

Dexy Snippets

The notebook `notebooks/Chapter1_Introduction.ipynb` uses format `3.0`.

Here are the worksheets in the notebook:

- notebooks/Chapter1_Introduction.ipynb--ws-0

  This worksheet has 47 cells and 59 documents based on these cells.

  - notebooks/Chapter1_Introduction.ipynb--ws-0-cell-0.md

    Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-0.md:

```
Probabilistic Programming
=====
and Bayesian Methods for Hackers
========

#####Version 0.1
Welcome to *Bayesian Methods for Hackers*. The full Github repository, and addi
```

  - notebooks/Chapter1_Introduction.ipynb--ws-0-cell-1.md

    Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-1.md:

```
Chapter 1
======
***
```

  - notebooks/Chapter1_Introduction.ipynb--ws-0-cell-2.md

    Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-2.md:

```
The Philosophy of Bayesian Inference
------

> You are a skilled programmer, but bugs still slip into your code. After a par

If you think this way, then congratulations, you already are a Bayesian practit
```

  - notebooks/Chapter1_Introduction.ipynb--ws-0-cell-3.md

    Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-3.md:

```
###The Bayesian state of mind


Bayesian inference differs from more traditional statistical inference by prese

The Bayesian world-view interprets probability as measure of *believability in 

For this to be clearer, we consider an alternative interpretation of probabilit

Bayesians, on the other hand, have a more intuitive approach. Bayesians interpr

Notice in the paragraph above, I assigned the belief (probability) measure to a
```

- I flip a coin, and we both guess the result. We would both agree, assuming th

- Your code either has a bug in it or not, but we do not know for certain whic

- A medical patient is exhibiting symptoms x, y and z. There are a numbe

This philosophy of treating beliefs as probability is natural to humans. We emp

To align ourselves with traditional probability notation, we denote our belief

John Maynard Keynes, a great economist and thinker, said "When the facts change

1\. $P(A): \;\;$ the coin has a 50 percent chance of being heads. $P(A | X):\;\

2\. $P(A): \;\;$ This big, complex code likely has a bug in it. $P(A | X): \

3\. $P(A):\;\;$ The patient could have any number of diseases. $P(A | X):\;\;$

It's clear that in each example we did not completely discard the prior belief

By introducing prior uncertainty about events, we are already admitting that an

– notebooks/Chapter1_Introduction.ipynb--ws-0-cell-4.md

Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-4.md:

###Bayesian Inference in Practice

 If frequentist and Bayesian inference were programming functions, with inputs

For example, in our debugging problem above, calling the frequentist function w

>   *YES*, with probability 0.8; *NO*, with probability 0.2

This is very different from the answer the frequentist function returned. Notic

####Incorporating evidence

As we acquire more and more instances of evidence, our prior belief is *washed

```
Denote $N$ as the number of instances of evidence we possess. As we gather an *

One may think that for large $N$, one can be indifferent between the two techni

> Sample sizes are never large. If $N$ is too small to get a sufficiently-preci

### Are frequentist methods incorrect then?

**No.**

Frequentist methods are still useful or state-of-the-art in many areas. Tools l


#### A note on *Big Data*
Paradoxically, big data's predictive analytic problems are actually solved by r

The much more difficult analytic problems involve *medium data* and, especially
```

– notebooks/Chapter1_Introduction.ipynb--ws-0-cell-5.md

Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-5.md:

```
### Our Bayesian framework

We are interested in beliefs, which can be interpreted as probabilities by thin

Secondly, we observe our evidence. To continue our buggy-code example: if our c

\begin{align}
 P( A | X ) = & \frac{ P(X | A) P(A) } {P(X) } \\\\[5pt]
& \propto P(X | A) P(A)\;\; (\propto \text{is proportional to } )
\end{align}

The above formula is not unique to Bayesian inference: it is a mathematical fac
```

– notebooks/Chapter1_Introduction.ipynb--ws-0-cell-6.md

Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-6.md:

```
##### Example: Mandatory coin-flip example

Every statistics text must contain a coin-flipping example, I'll use it here to

We begin to flip a coin, and record the observations: either $H$ or $T$. This i

Below we plot a sequence of updating posterior probabilities as we observe incr
```

– notebooks/Chapter1_Introduction.ipynb--ws-0-cell-7-input.py

Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-7-input.py:

```
"""
The book uses a custom matplotlibrc file, which provides the unique styles for
If executing this book, and you wish to use the book's styling, provided are tw
    1. Overwrite your own matplotlibrc file with the rc-file provided in the bo
       See http://matplotlib.org/users/customizing.html
    2. Also in the styles is  bmh_matplotlibrc.json file. This can be used to u
       in only this notebook. Try running the following code:

        import json
        s = json.load( open("../styles/bmh_matplotlibrc.json") )
        matplotlib.rcParams.update(s)

"""

#the code below can be passed over, as it is currently not important.
%pylab inline
figsize( 11, 9)

import scipy.stats as stats

dist = stats.beta
n_trials = [0,1,2,3,4,5,8,15, 50, 500]
data = stats.bernoulli.rvs(0.5, size = n_trials[-1] )
x = np.linspace(0,1,100)

for k, N in enumerate(n_trials):
    sx = subplot( len(n_trials)/2, 2, k+1)
    plt.xlabel("$p$, probability of heads") if k in [0,len(n_trials)-1] else No
    plt.setp(sx.get_yticklabels(), visible=False)
    heads = data[:N].sum()
    y = dist.pdf(x, 1 + heads, 1 + N - heads )
    plt.plot( x, y, label= "observe %d tosses,\n %d heads"%(N,heads) )
    plt.fill_between( x, 0, y, color="#348ABD", alpha = 0.4 )
    plt.vlines( 0.5, 0, 4, color = "k", linestyles = "--", lw=1 )

    leg = plt.legend()
    leg.get_frame().set_alpha(0.4)
    plt.autoscale(tight = True)


plt.suptitle( "Bayesian updating of posterior probabilities",
              y = 1.02,
              fontsize = 14);

plt.tight_layout()
```

– notebooks/Chapter1_Introduction.ipynb--ws-0-cell-7-output-0.txt

Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-7-output-0

– notebooks/Chapter1_Introduction.ipynb--ws-0-cell-7-output-1.png

Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-7-output-1

– notebooks/Chapter1_Introduction.ipynb--ws-0-cell-8.md

Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-8.md:

```
The posterior probabilities are represented by the curves, and our confidence i

Notice that the plots are not always *peaked* at 0.5. There is no reason it sho

The next example is a simple demonstration of the mathematics of Bayesian infer
```

– notebooks/Chapter1_Introduction.ipynb--ws-0-cell-9.md

Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-9.md:

```
#####Example: Bug, or just sweet, unintended feature?


Let $A$ denote the event that our code has **no bugs** in it. Let $X$ denote th

We are interested in $P(A|X)$, i.e. the probability of no bugs, given our debug

What is $P(X | A)$, i.e., the probability that the code passes $X$ tests *given

$P(X)$ is a little bit trickier: The event $X$ can be divided into two possibil
```

– notebooks/Chapter1_Introduction.ipynb--ws-0-cell-10.md

Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-10.md:

```
\begin{align}
P(X ) & = P(X \text{ and } A) + P(X \text{ and } \sim A) \\\\[5pt]
 & = P(X|A)P(A) + P(X | \sim A)P(\sim A)\\\\[5pt]
& = P(X|A)p + P(X | \sim A)(1-p)
\end{align}
```

– notebooks/Chapter1_Introduction.ipynb--ws-0-cell-11.md

Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-11.md:

```
We have already computed $P(X|A)$ above. On the other hand, $P(X | \sim A)$ is

\begin{align}
P(A | X) & = \frac{1\cdot p}{ 1\cdot p +0.5 (1-p) } \\\\
& = \frac{ 2 p}{1+p}
\end{align}
This is the posterior probability. What does it look like as a function of our
```

– notebooks/Chapter1_Introduction.ipynb--ws-0-cell-12-input.py

Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-12-input.py

```
figsize(12.5,4)
p = np.linspace( 0,1, 50)
plt.plot( p, 2*p/(1+p), color = "#348ABD", lw = 3 )
#plt.fill_between( p, 2*p/(1+p), alpha = .5, facecolor = ["#A60628"])
plt.scatter( 0.2, 2*(0.2)/1.2, s = 140, c ="#348ABD"  )
plt.xlim( 0, 1)
plt.ylim( 0, 1)
plt.xlabel( "Prior, $P(A) = p$")
plt.ylabel("Posterior, $P(A|X)$, with $P(A) = p$")
plt.title( "Are there bugs in my code?");
```

– notebooks/Chapter1_Introduction.ipynb--ws-0-cell-12-output-0.png

Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-12-output-(

– notebooks/Chapter1_Introduction.ipynb--ws-0-cell-13.md

Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-13.md:

```
We can see the biggest gains if we observe the $X$ tests passed when the prior

Recall that the prior is a probability: $p$ is the prior probability that there

Similarly, our posterior is also a probability, with $P(A | X)$ the probability
```

– notebooks/Chapter1_Introduction.ipynb--ws-0-cell-14-input.py

Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-14-input.py

```
figsize( 12.5, 4 )
colours = ["#348ABD", "#A60628"]

prior = [0.20, 0.80]
posterior =  [1./3, 2./3]
plt.bar( [0,.7], prior ,alpha = 0.70, width = 0.25, \
             color = colours[0], label = "prior distribution",
              lw = "3", edgecolor = colours[0])


plt.bar( [0+0.25,.7+0.25], posterior ,alpha = 0.7, \
          width = 0.25, color =  colours[1],
          label = "posterior distribution",
          lw = "3", edgecolor = colours[1])

plt.xticks( [0.20,.95], ["Bugs Absent", "Bugs Present"] )
plt.title("Prior and Posterior probability of bugs present, prior = 0.2")
plt.ylabel("Probability")
plt.legend(loc="upper left");
```

– notebooks/Chapter1_Introduction.ipynb--ws-0-cell-14-output-0.png

Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-14-output-(

– notebooks/Chapter1_Introduction.ipynb--ws-0-cell-15.md

Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-15.md:

```
Notice that after we observed $X$ occur, the probability of bugs being absent i

This was a very simple example of Bayesian inference and Bayes rule. Unfortunat
```

– notebooks/Chapter1_Introduction.ipynb--ws-0-cell-16.md

Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-16.md:

```
-------

##Probability Distributions


**Let's quickly recall what a probability distribution is:** Let $Z$ be some ra

We can divide random variables into three classifications:

-   **$Z$ is discrete**: Discrete random variables may only assume values on a

-   **$Z$ is continuous**: Continuous random variable can take on arbitrarily e

- **$Z$ is mixed**: Mixed random variables assign probabilities to both discret

###Discrete Case
If $Z$ is discrete, then its distribution is called a *probability mass functio

$$P(Z = k) =\frac{ \lambda^k e^{-\lambda} }{k!}, \; \; k=0,1,2, \dots $$

What is $\lambda$? It is called the parameter, and it describes the shape of th

Unlike $\lambda$, which can be any positive number, the value $k$ in the above

If a random variable $Z$ has a Poisson mass distribution, we denote this by wri

$$Z \sim \text{Poi}(\lambda) $$

One very useful property of the Poisson random variable, given we know $\lambda

$$E\large[ \;Z\; | \; \lambda \;\large] = \lambda $$

We will use this property often, so it's something useful to remember. Below we
```

– notebooks/Chapter1_Introduction.ipynb--ws-0-cell-17-input.py

Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-17-input.py

```
figsize( 12.5, 4)

import scipy.stats as stats
a = np.arange( 16 )
poi = stats.poisson
lambda_ = [1.5, 4.25 ]

plt.bar( a, poi.pmf( a, lambda_[0]), color=colours[0],
        label = "$\lambda = %.1f$"%lambda_[0], alpha = 0.60,
        edgecolor = colours[0], lw = "3")

plt.bar( a, poi.pmf( a, lambda_[1]), color=colours[1],
         label = "$\lambda = %.1f$"%lambda_[1], alpha = 0.60,
          edgecolor = colours[1], lw = "3")

plt.xticks( a + 0.4, a )
plt.legend()
plt.ylabel("probability of $k$")
plt.xlabel("$k$")
plt.title("Probability mass function of a Poisson random variable; differing \
$\lambda$ values");
```

– notebooks/Chapter1_Introduction.ipynb--ws-0-cell-17-output-0.png

Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-17-output-(

– notebooks/Chapter1_Introduction.ipynb--ws-0-cell-18.md

Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-18.md:

```
###Continuous Case
Instead of a probability mass function, a continuous random variable has a *pro

$$f_Z(z | \lambda) = \lambda e^{-\lambda z }, \;\; z\ge 0$$

Like the Poisson random variable, an exponential random variable can only take

When a random variable $Z$ has an exponential distribution with parameter $\lam

$$Z \sim \text{Exp}(\lambda)$$

Given a specific $\lambda$, the expected value of an exponential random variabl

$$E[\; Z \;|\; \lambda \;] = \frac{1}{\lambda}$$
```

– notebooks/Chapter1_Introduction.ipynb--ws-0-cell-19-input.py

Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-19-input.py

```
a = np.linspace(0,4, 100)
expo = stats.expon
```

```python
lambda_ = [0.5, 1]

for l,c in zip(lambda_,colours):
    plt.plot( a, expo.pdf( a, scale=1./l), lw=3,
                color=c, label = "$\lambda = %.1f$"%l)
    plt.fill_between( a, expo.pdf( a, scale=1./l), color=c, alpha = .33)

plt.legend()
plt.ylabel("PDF at $z$")
plt.xlabel("$z$")
plt.title("Probability density function of an Exponential random variable;\
 differing $\lambda$");
```

– notebooks/Chapter1_Introduction.ipynb--ws-0-cell-19-output-0.png

Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-19-output-(

– notebooks/Chapter1_Introduction.ipynb--ws-0-cell-20.md

Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-20.md:

```
###But what is $\lambda \;$?


**This question is what motivates statistics**. In the real world, $\lambda$ is

Bayesian inference is concerned with *beliefs* about what $\lambda$ is. Rather

This might seem odd at first: after all, $\lambda$ is fixed, it is not (necessa
```

– notebooks/Chapter1_Introduction.ipynb--ws-0-cell-21.md

Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-21.md:

```
##### Example: Inferring behaviour from text-message data

Let's try to model a more interesting example, concerning text-message rates:

>  You are given a series of text-message counts from a user of your system. Th
```

– notebooks/Chapter1_Introduction.ipynb--ws-0-cell-22-input.py

Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-22-input.py

```python
figsize( 12.5, 3.5 )
count_data = np.loadtxt("data/txtdata.csv")
n_count_data = len(count_data)
plt.bar( np.arange( n_count_data ), count_data, color ="#348ABD"  )
plt.xlabel( "Time (days)")
plt.ylabel("count of text-msgs received")
plt.title("Did the user's texting habits change over time?")
plt.xlim( 0, n_count_data );
```

- notebooks/Chapter1_Introduction.ipynb--ws-0-cell-22-output-0.png

  Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-22-output-(

- notebooks/Chapter1_Introduction.ipynb--ws-0-cell-23.md

  Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-23.md:

```
Before we begin, with respect to the plot above, would you say there was a chan
during the time period?

How can we start to model this? Well, as I conveniently already introduced, a P

$$ C_i \sim \text{Poisson}(\lambda)  $$

We are not sure about what the $\lambda$ parameter is though. Looking at the ch

How can we mathematically represent this? We can think, that at some later date

$$
\lambda =
\begin{cases}
\lambda_1  & \text{if } t \lt \tau \cr
\lambda_2 & \text{if } t \ge \tau
\end{cases}
$$


 If, in reality, no sudden change occurred and indeed $\lambda_1 = \lambda_2$,

We are interested in inferring the unknown $\lambda$s. To use Bayesian inferenc

\begin{align}
&\lambda_1 \sim \text{Exp}( \alpha ) \\\
&\lambda_2 \sim \text{Exp}( \alpha )
\end{align}

$\alpha$ is called a *hyper-parameter*, or a *parent-variable*, literally a par

$$\frac{1}{N}\sum_{i=0}^N \;C_i \approx E[\; \lambda \; |\; \alpha ] = \frac{1}

Alternatively, and something I encourage the reader to try, is to have two prio

What about $\tau$? Well, due to the randomness, it is too difficult to pick out

\begin{align}
& \tau \sim \text{DiscreteUniform(1,70) }\\\\
& \Rightarrow P( \tau = k ) = \frac{1}{70}
```

\end{align}

So after all this, what does our overall prior for the unknown variables look l

Introducing our first hammer: PyMC

PyMC is a Python library for programming Bayesian analysis [3]. It is a fast, w

We will model the above problem using the PyMC library. This type of programmin

B. Cronin [5] has a very motivating description of probabilistic programming:

>   Another way of thinking about this: unlike a traditional program, which onl

Due to its poorly understood title, I'll refrain from using the name *probabili

The PyMC code is easy to follow along: the only novel thing should be the synta

- notebooks/Chapter1_Introduction.ipynb--ws-0-cell-24-input.py

  Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-24-input.py

```
import pymc as mc



alpha = 1.0/count_data.mean() #recall count_data is
                               #the variable that holds our txt counts

lambda_1 = mc.Exponential( "lambda_1",  alpha )
lambda_2 = mc.Exponential( "lambda_2", alpha )

tau = mc.DiscreteUniform( "tau", lower = 0, upper = n_count_data )
```

- notebooks/Chapter1_Introduction.ipynb--ws-0-cell-25.md

  Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-25.md:

  In the above code, we create the PyMC variables corresponding to $\lambda_1, \;

- notebooks/Chapter1_Introduction.ipynb--ws-0-cell-26-input.py

  Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-26-input.py

```
print "Random output:", tau.random(),tau.random(), tau.random()
```

- notebooks/Chapter1_Introduction.ipynb--ws-0-cell-26-output-0.txt

  Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-26-output-0

- notebooks/Chapter1_Introduction.ipynb--ws-0-cell-27-input.py

  Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-27-input.py

```
@mc.deterministic
def lambda_( tau = tau, lambda_1 = lambda_1, lambda_2 = lambda_2 ):
    out = np.zeros( n_count_data )
    out[:tau] = lambda_1 #lambda before tau is lambda1
    out[tau:] = lambda_2 #lambda after tau is lambda2
    return out
```

- notebooks/Chapter1_Introduction.ipynb--ws-0-cell-28.md

  Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-28.md:

  ```
  This code is creating a new function `lambda_`, but really we think of it as a
  ```

- notebooks/Chapter1_Introduction.ipynb--ws-0-cell-29-input.py

  Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-29-input.py

  ```
  observation = mc.Poisson( "obs", lambda_, value = count_data, observed = True)

  model = mc.Model( [observation, lambda_1, lambda_2, tau] )
  ```

- notebooks/Chapter1_Introduction.ipynb--ws-0-cell-30.md

  Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-30.md:

  ```
  The variable `observation` combines our data, `count_data`, with our proposed d

  The below code will be explained in the Chapter 3, but this is where our result
  ```

- notebooks/Chapter1_Introduction.ipynb--ws-0-cell-31-input.py

  Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-31-input.py

  ```
  ### Mysterious code to be explained in Chapter 3.
  mcmc = mc.MCMC(model)
  mcmc.sample( 40000, 10000, 1 )
  ```

- notebooks/Chapter1_Introduction.ipynb--ws-0-cell-31-output-0.txt

  Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-31-output-(

- notebooks/Chapter1_Introduction.ipynb--ws-0-cell-31-output-1.txt

  Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-31-output-

- notebooks/Chapter1_Introduction.ipynb--ws-0-cell-32-input.py

  Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-32-input.py

  ```
  lambda_1_samples = mcmc.trace( 'lambda_1' )[:]
  lambda_2_samples = mcmc.trace( 'lambda_2' )[:]
  tau_samples = mcmc.trace( 'tau' )[:]
  ```

- notebooks/Chapter1_Introduction.ipynb--ws-0-cell-33-input.py

  Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-33-input.py

  ```
  figsize(12.5, 10)
  #histogram of the samples:
  ```

```
ax = plt.subplot(311)
ax.set_autoscaley_on(False)

plt.hist( lambda_1_samples, histtype='stepfilled', bins = 30, alpha = 0.85,
        label = "posterior of $\lambda_1$", color = "#A60628",normed = True )
plt.legend(loc = "upper left")
plt.title(r"Posterior distributions of the variables $\lambda_1,\;\lambda_2,\;\
plt.xlim([15,30])
plt.xlabel("$\lambda_2$ value")
plt.ylabel("probability")

ax = plt.subplot(312)
ax.set_autoscaley_on(False)

plt.hist( lambda_2_samples,histtype='stepfilled', bins = 30, alpha = 0.85,
          label = "posterior of $\lambda_2$",color="#7A68A6", normed = True )
plt.legend(loc = "upper left")
plt.xlim([15,30])
plt.xlabel("$\lambda_2$ value")
plt.ylabel("probability")

plt.subplot(313)


w = 1.0/ tau_samples.shape[0] * np.ones_like( tau_samples )
plt.hist( tau_samples, bins = n_count_data, alpha = 1,
         label = r"posterior of $\tau$",
         color="#467821", weights=w, rwidth =2. )
plt.xticks( np.arange( n_count_data ) )

plt.legend(loc = "upper left");
plt.ylim([0,.75])
plt.xlim([35, len(count_data)-20])
plt.xlabel("$\tau$ (in days)")
plt.ylabel("probability");
```

– `notebooks/Chapter1_Introduction.ipynb--ws-0-cell-33-output-0.png`

  Here are the contents of `notebooks/Chapter1_Introduction.ipynb--ws-0-cell-33-output-(`

– `notebooks/Chapter1_Introduction.ipynb--ws-0-cell-34.md`

  Here are the contents of `notebooks/Chapter1_Introduction.ipynb--ws-0-cell-34.md`:

```
### Interpretation

Recall that the Bayesian methodology returns a *distribution*, hence we now hav

Also notice that the posterior distributions for the $\lambda$'s do not look li
```

```
Our analysis also returned a distribution for what $\tau$ might be. Its posteri
```

- notebooks/Chapter1_Introduction.ipynb--ws-0-cell-35.md

  Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-35.md:

```
###Why would I want samples from the posterior, anyways?


We will deal with this question for the remainder of the book, and it is an und

In the code below, we are calculating the following: Let $i$ index samples from
```

- notebooks/Chapter1_Introduction.ipynb--ws-0-cell-36-input.py

  Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-36-input.py

```
figsize( 12.5, 5)
# tau_samples, lambda_1_samples, lambda_2_samples contain
# N samples from the corresponding posterior distribution
N = tau_samples.shape[0]
expected_texts_per_day = np.zeros(n_count_data)
for  day in range(0, n_count_data):
    # ix is a bool index of all tau samples corresponding to
    # the switchpoint occurring prior to value of 'day'
    ix = day < tau_samples
    # Each posterior sample corresponds to a value for tau.
    # for each day, that value of tau indicates whether we're "before"
    # (in the lambda1 "regime") or
    #  "after" (in the lambda2 "regime") the switchpoint.
    # by taking the posterior sample of lambda1/2 accordingly, we can average
    # over all samples to get an expected value for lambda on that day.
    # As explained, the "message count" random variable is Poisson distributed,
    # and therefore lambda (the poisson parameter) is the expected value of "me
    expected_texts_per_day[day] = (lambda_1_samples[ix].sum()
                                   + lambda_2_samples[~ix].sum() ) /N


plt.plot( range( n_count_data), expected_texts_per_day, lw =4, color = "#E24A33
         label = "expected number of text-messages recieved")
plt.xlim( 0, n_count_data )
plt.xlabel( "Day" )
plt.ylabel( "Expected # text-messages" )
plt.title( "Expected number of text-messages received")
plt.ylim( 0, 50 )
plt.bar( np.arange( len(count_data) ), count_data, color ="#348ABD", alpha = 0.
            label="observed texts per day")
```

```
plt.legend(loc="upper left");
```

– notebooks/Chapter1_Introduction.ipynb--ws-0-cell-36-output-0.png

Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-36-output-(

– notebooks/Chapter1_Introduction.ipynb--ws-0-cell-37.md

Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-37.md:

```
Our analysis shows strong support for believing the user's behavior did change
```

– notebooks/Chapter1_Introduction.ipynb--ws-0-cell-38.md

Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-38.md:

```
##### Exercises

1\.  Using `lambda_1_samples` and `lambda_2_samples`, what is the mean of the p
```

– notebooks/Chapter1_Introduction.ipynb--ws-0-cell-39-input.py

Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-39-input.py

```
#type your code here.
```

– notebooks/Chapter1_Introduction.ipynb--ws-0-cell-40.md

Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-40.md:

```
2\.  What is the expected percentage increase in text-message rates? `hint:` co
```

– notebooks/Chapter1_Introduction.ipynb--ws-0-cell-41-input.py

Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-41-input.py

```
#type your code here.
```

– notebooks/Chapter1_Introduction.ipynb--ws-0-cell-42.md

Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-42.md:

```
3\. What is the mean of $\lambda_1$ **given** we know $\tau$ is less than 45.
```

– notebooks/Chapter1_Introduction.ipynb--ws-0-cell-43-input.py

Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-43-input.py

```
#type your code here.
```

– notebooks/Chapter1_Introduction.ipynb--ws-0-cell-44.md

Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-44.md:


```
### References


-  [1] Gelman, Andrew. N.p.. Web. 22 Jan 2013. <http://andrewgelman.com/2005/07
-  [2] Norvig, Peter. 2009. [*The Unreasonable Effectiveness of Data*](http://w
- [3] Patil, A., D. Huard and C.J. Fonnesbeck. 2010.
PyMC: Bayesian Stochastic Modelling in Python. Journal of Statistical
Software, 35(4), pp. 1-81.
- [4] Jimmy Lin and Alek Kolcz. Large-Scale Machine Learning at Twitter. Procee
- [5] Cronin, Beau. "Why Probabilistic Programming Matters." 24 Mar 2013. Googl
```

- notebooks/Chapter1_Introduction.ipynb--ws-0-cell-45-input.py

  Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-45-input.py

  ```
  from IPython.core.display import HTML
  def css_styling():
      styles = open("../styles/custom.css", "r").read()
      return HTML(styles)
  css_styling()
  ```

- notebooks/Chapter1_Introduction.ipynb--ws-0-cell-46-input.py

  Here are the contents of notebooks/Chapter1_Introduction.ipynb--ws-0-cell-46-input.py