

IBM

Search

Profiles

Communities

Apps

Blogs

This Blog

Search

My Blogs


Public Blogs

My Updates

IT Best Kept Secret Is Optimization

Log in

to participate

 The developerWorks Connections Platform is now in read-only mode and content is only available for viewing. No new wiki pages, posts, or messages may be added. Please see our FAQ for more information. The developerWorks Connections platform will officially shut down on March 31, 2020 and content will no longer be available. [More details available on our FAQ.](#) ([Read in Japanese.](#))

Start your free trial

Green dice are loaded (welcome to p-hacking)

JeanFrancoisPuget | Mar 22 2016 | Visits (22719) 1



Did you know green dice are loaded? I am not making it, I designed a rigorous scientific study that proves it.

Let me repeat slowly: *painting dice in green make them loaded.*

You don't believe me? Let me describe you the study and you'll be able to reproduce my scientific results.

The study

I made a hypothesis that green dice are loaded and that they will yield a six outcome more often than usual dice would. I then took 100 green dice, and rolled them 10 times each, then I counted how many times a six appears.

A six appeared 188 times, which is rather high. Indeed, one would expect that the number of a six outcome to be closer to $1,000/6 = 167$. It seems that my hypothesis about green dice may have some truth in it.

Scientists have devised ways to quantify what I called 'some truth' in the previous sentence. The most common one is called *p-value*. It is the probability to get a result at least as deviant from normal that the one we got if we make no hypothesis. There are other approaches than p-values for assessing experimental results, but discussing them is beyond the scope of this post.

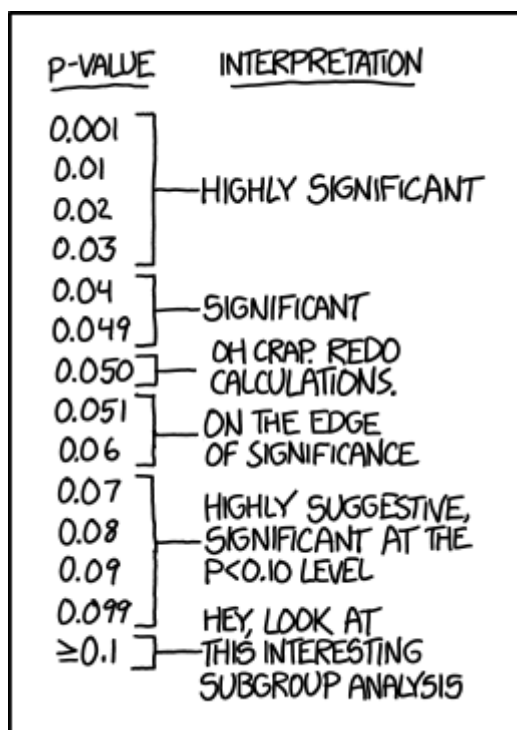
For the sake of clarity, let's compute the p-value for my experiment. The p-value is the probability to get at least 188 times a six out of 1,000 dice rolls if the dice do not favor any of the possible outcomes. This probability is about 4% (see the math section below for details), i.e. 1 odd out of 25.

A low p-value such as 4% is telling us something interesting: if we make no hypothesis about our dice other than they do not favor any outcome, then the result is rather unlikely. This is deemed sufficient by many scientific communities to conclude that the hypothesis (green dice are loaded) is true. I should therefore publish my result.

Is it that simple?

Are low p-values good enough?

A p-value of less than 5% is indeed strong enough to get published in many scientific journals. It seems however that this led some researchers to tweak their data and their results to get a p-value small enough whatever data they have. This is called p-hacking. It is illustrated by the xkcd comic below.



<u>P-VALUE</u>	<u>INTERPRETATION</u>
0.001	HIGHLY SIGNIFICANT
0.01	
0.02	
0.03	
0.04	SIGNIFICANT
0.049	
0.050	OH CRAP. REDO CALCULATIONS.
0.051	ON THE EDGE OF SIGNIFICANCE
0.06	
0.07	HIGHLY SUGGESTIVE, SIGNIFICANT AT THE P<0.10 LEVEL
0.08	
0.09	
0.099	HEY, LOOK AT THIS INTERESTING SUBGROUP ANALYSIS
≥0.1	

credit: [xkcd](#)

Examples of p-hacking are not that hard to find. For instance, a science journalist called John Bohannon managed to publish [a study showing that eating chocolate during a diet led to higher weight loss](#) using a similar approach. Of course this was a fake as the measured effect on weight loss was less than 0.1%, i.e. there was no effect at all. A growing number of scientific papers based on low p-values are being retracted because they contain results that could not be replicated, as explained in this [Nature Magazine paper by Regina Nuzzo](#). Christie Aschwanden further writes in [Science Isn't Broken](#) that: *"Scientists' over reliance on p-values has led at least one journal to decide it has had enough of them. In February, Basic and Applied Social Psychology announced that it will no longer publish p-values. 'We believe that the $p < .05$ bar is too easy to pass and sometimes serves as an excuse for lower quality research,' the editors wrote in their announcement. Instead of p-values, the journal will require 'strong descriptive statistics, including effect sizes.'"*

When are p-values good enough, and when are they misleading?

Correlation isn't causation

There is a trap that many fall into when it comes to p-values. Let me illustrate it with the dice experiment. The p-value does not say anything about the hypothesis (that green dice are loaded). It just says that a result as far from the mean of 167 six per 1,000 roll is unlikely if the dice do not favor any of possible outcomes. That's it. Don't read more than that. Said differently, nothing in the experiment says anything about our hypothesis:

The 4% p-value does not validate the hypothesis that the color of dice has an influence on dice roll outcome.

Right, there is an unlikely result, but there may be plenty of reasons to explain this result. It can still be due to chance, even if unlikely. Maybe the assembly line had some issues when producing this dice lot. Maybe they were all stored the wrong way, and got tweaked. Maybe we did not interpret the experiment correctly. Who knows?

It is a bit disturbing to not be able to imagine how dice color would explain that six is a more likely outcome. That should be the warning signal: if we cannot figure out a plausible reason why the color of dice influences the way it rolls, then we don't have much to publish.

The need to have evidence of a causal relationship between the hypothesis and the observed result should be mandatory. This is gaining traction: for instance, as written above, some journals like *Basic and Applied Social Psychology* won't publish findings based solely on p-value anymore.

Still, I did get an unlikely result in my experiment. If this can't be explained by dice color, then what explains it?

Let's investigate further. For that we need to look under the hood and check all the experiment settings.

The catch

When looking at what I did, I must confess I actually forgot to tell you about a seemingly minor detail. I tested 20 different group of dice with various colors, not just green. For each color, I gathered 100 dice, and rolled each of them 10 times.

Here are the results I got for each dice color:



	Number of Six
Purple	151
Brown	167
Pink	158
Blue	167
Teal	181
Salmon	162
Red	170
Turquoise	161
Magenta	165
Yellow	180
Grey	172
Tan	164
Cyan	181
Green	188
Mauve	165
Beige	172
Lilac	178
Black	176
Peach	173
Orange	157

We see that only green dice had so many six. The probability to get that high a result for green dice is low, isn't it? Remember, I computed it to be 4%.

As we are about to see, my computation of p-value is plain wrong unfortunately. Indeed, running 20 dice experiments is different from running one. What I have done is this:

1. I started with 100 dice of each color among 20 colors.
2. I rolled each dice 10 times.
3. I collected the number of times a six appeared for each color.
4. I found that green had the highest number of six.
5. I computed the p-value of the green dice result.
6. I then claimed that the p-value for this experiment was 4%.

The last step is wrong. Indeed, a p-value of 4% corresponds to this other experiment:

1. I started with 100 dice of each color.
2. I decided to watch green dice in particular.
3. I rolled each dice 10 times.

Start your free trial

4. I collected the number of times a six appeared for each color.
5. I computed the p-value of the green dice result.
6. The p-value in this experiment is 4%, hence I found something interesting.

Isn't the second experiment the same as the first one?

No, it isn't.

There is a major difference. In the first experiment, I selected the dice color *after* the experiment was run. In the second experiment, I selected the dice color *before* running the experiment.

As a matter of fact, the first experiment should be described this way:

1. I started with 100 dice of each color among 20 colors.
2. I rolled each dice 10 times.
3. I collected the number of times a six appeared for each color.
4. I selected the color with the highest number of six.
5. I computed the p-value for this result.

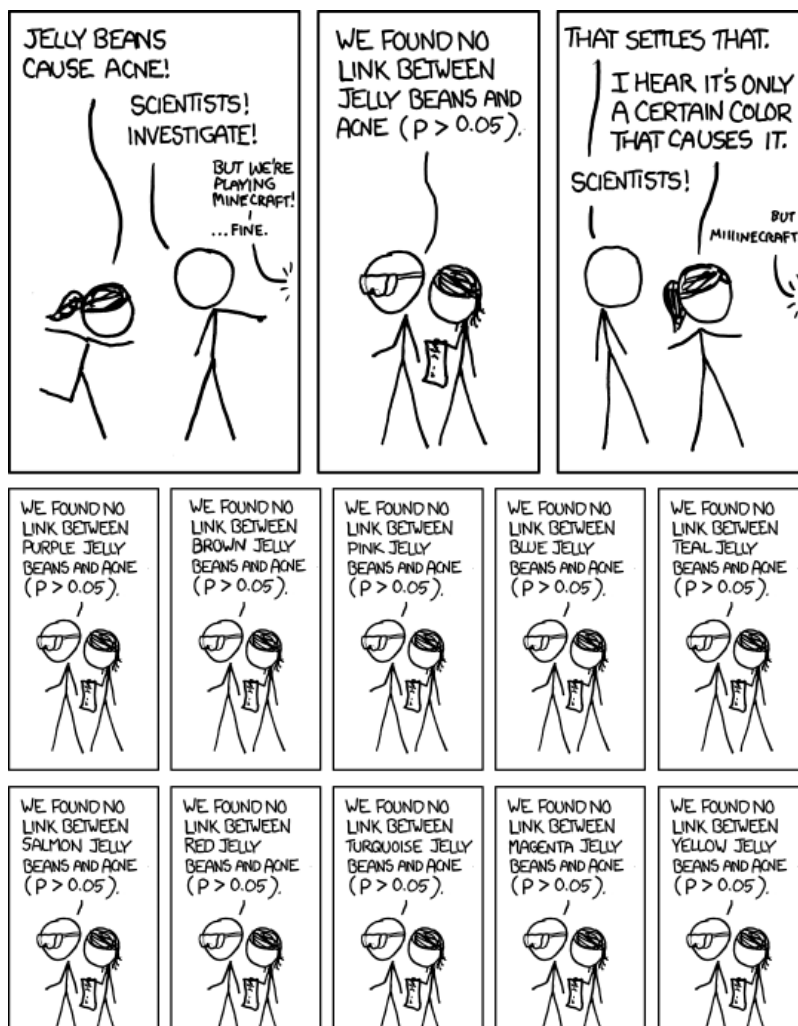
It so happens that the p-value is about 56% in this case. This is 11 times higher than the 5% threshold used for p-values in general.

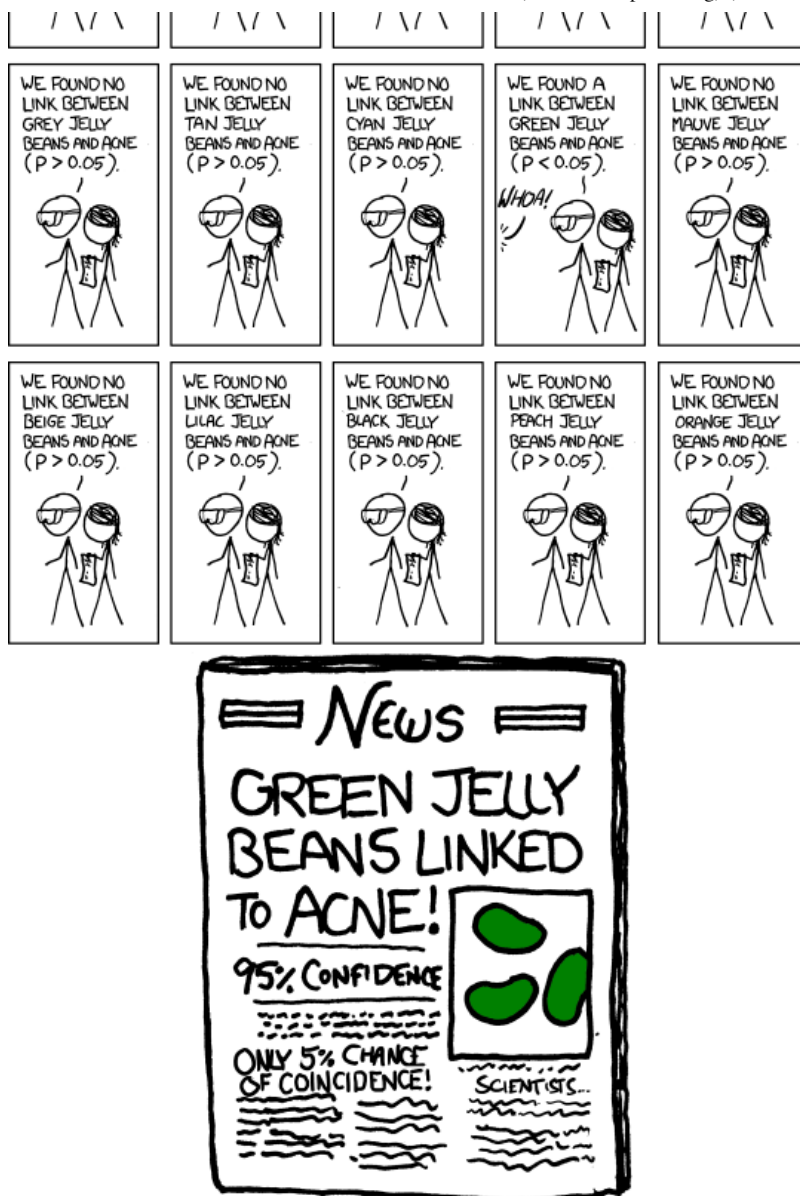
Nothing much surprising therefore as the result was quite likely after all.

Foregone is my scientific discovery!

P-hacking

I have been sin of p-value hacking (p-hacking) when I first reported our experiment results. Indeed, I reported results for one color without mentioning I tried 20 colors all in all. I did exactly what is depicted in the xkcd comics below.



Credit: [xkcd](#)

The fact is, the more colors we try, the more likely a result deviant from the norm. In general, the more features you measure when performing an experiment, the more likely you'll get some values far apart from the norm. This is now a well-established fact, see John D Cook's in [The empty middle: why no one is average](#) for more details.

I described one way of hacking p-values, but there are many more. Should we avoid using p-values given they can be hacked in many ways?

The cure

Banning the use of p-values may be a bit extreme. After all, low p-values tell us something interesting. It is just that a low p-value alone isn't strong enough to warrant a scientific discovery. And we must make sure p-values are computed correctly, i.e. using all the data that was used, and using all the experiment results that were produced.

Issues with p-values have led the American Statistical Association (ASA) to recently issue [six principles about p-values](#):

1. *P-values can indicate how incompatible the data are with a specified statistical model.*
2. *P-values do not measure the probability that the studied hypothesis is true, or the probability that the data were produced by random chance alone.*

3. *Scientific conclusions and business or policy decisions should not be based only on whether a p-value passes a specific threshold.*
4. *Proper inference requires full reporting and transparency.*
5. *A p-value, or statistical significance, does not measure the size of an effect or the importance of a result.*
6. *By itself, a p-value does not provide a good measure of evidence regarding a model or hypothesis.*

I violated several of them in my account of the dice experiment. I jumped to the conclusion that dice color was relevant solely based on a p-value, violating principle 3. I did not report the exact experiment I conducted, violating principle 4. I did not find any evidence that color was relevant, yet I jumped to conclusion based on p-value, violating principle 6.

Understanding and following these six principles should help use p-values for what they're worth, and not more.

My take is that p-values are useful if they are computed correctly, and if one does not read more into them than what they are. For more on these principles, see this [interview](#) of Ron Wasserstein, ASA's executive director. I also recommend Arthur Charpentier's [p-hacking, or cheating on a p-value](#). In particular, he writes: *In Economics, there is an old saying: “when a measure become a target, it is no longer a measure”. That’s Goodhart’s law.* This provides another definition of what not to do with p-values.

In the remainder of this post I provide Python code for running the experiment, then I provide details on how to compute the various p-values we used so far.

Experiment details

Here is the Python code that we used to run the dice experiment. The code used in this post is available on [github](#) and [nbviewer](#).

```
import numpy as np
import pandas as pd
from numpy.random import random_integers, seed

dice = ['Purple', 'Brown', 'Pink', 'Blue',
        'Teal',
        'Salmon', 'Red', 'Turquoise',
        'Magenta', 'Yellow',
        'Grey', 'Tan', 'Cyan', 'Green',
        'Mauve',
        'Beige', 'Lilac', 'Black', 'Peach',
        'Orange']

def dice_experiments(dice, n):
    df = pd.DataFrame(index=dice)
    for die in dice:
        result = random_integers(1,6,n)
        df.loc[die, 'Number of Six'] = np.sum(result[result==6])/6
    return df

np.random.seed(75000)
df = dice_experiments(dice, 1000)
df
```

We fixed the random seed for reproducibility, but we could have used other seeds. Remember that we have more than 50% chances to get at least 188 six out of 1,000 rolls for at least one color in this experiment. It means that if we do not set the random seed, and run the above code, we are likely to get what we look for. Let's confirm this by running the experiment 1,000 times:

```
def get_fraction_any_color(dice, k, n,
    repeat):
    success = 0.0
    for experiment in range(repeat):
        df = dice_experiments(dice, n)
        if df['Number of Six'].max() >= k:
            success += 1
    return success/repeat

get_fraction_any_color(dice, 188, 1000, 1000)
```

Running the code yields 0.556, i.e. 55.6%. It means that we get the result we want in 556 of the 1,000 experiments.

The odds of getting the result for the green dice is way lower. Let's confirm this experimentally as well, by running again the experiment 1,000 times.

```
def get_fraction_one_color(color, dice, k, n,
    repeat):
    success = 0.0
    for experiment in range(repeat):
        df = dice_experiments(dice, n)
        if df.loc['Green', 'Number of Six'] >=
k:
            success += 1
    return success/repeat

get_fraction_one_color('Green', dice, 188,
    1000, 10000)
```

This yields 0.041, i.e. 4.1%. We only got the desired result 41 times among the 1,000 experiments.

For the fun, I wanted to use the exact same colors as in the xkcd comic, hence I looked for random seeds that led to it. The code is provided below.

```
def find_seed(dice, k, n, repeat, rounding):
    nseed = 0.0
    for seed in range(0, repeat, rounding):
        np.random.seed(seed)
        df = dice_experiments(dice, n)
        m = df['Number of Six'].max()
        if m == k and m == df.loc['Green', 'N
umber of Six']:
            return seed

res = find_seed(dice, 188, 1000, 1000000,
    1000)
res
```


Note that this is yet another p-hacking instance: I selected the random seed based on the result we wanted to show. This is really bad p-hacking, as I define first the p-value, then find experiment settings that produce it.

This dice experiment design is really not a model to follow!

Some math

Let's now compute exact p-values for our two experiments.

In order to get exactly k times a six among n , we must first decide where those six appear in the sequence of 1,000 rolls. There are exactly n choose $k = n!/(k!(n-k)!)$ ways to do it. Then for each of these sequence, the probability that all of the k occurrences are a six is $1/6^k$ and the probability that the remaining $n-k$ occurrences are not a six is $(5/6)^{n-k}$. The probability to get exactly k times a six out of n dice rolls is therefore equal to $1/6^k (5/6)^{n-k} n!/(k!(n-k)!)$. It can be computed by this Python function, where `comb(n,k)` computes $n!/(k!(n-k)!)$

```
from scipy.misc import comb

def proba_k_among_n(k, n):
    p = 1/6
    q = 1-p
    result = p**k * q**(n-k) * comb(n, k)
    return result
```

From it we can easily compute the probability that at least k out of n occurrences are a six, by summing the probabilities for each of the possible result.

```
def proba_one_color(k, n):
    proba = 0.0
    for i in range(k, n+1):
        proba += proba_k_among_n(i, n)
    return proba

print("%0.3f" % proba_at_least_k_among_n(18, 1000))
```

Running it yields 0.040, i.e. 4%. This is close to the experimental result we got.

The probability that at least one of the colors gets 188 times a six is the complement of the probability that no color gets at least 188 times a six.

```
def proba_at_least_one_color(ncolors, k, n):
    p = proba_one_color(k, n)
    proba_no_color = (1-p)**ncolors
    return 1 - proba_no_color

print("%0.3f" % proba_at_least_one_color(20, 188, 1000))
```

Running it yields 0.559, i.e. 55.9%, which is close to what we got experimentally.

The code used in this post is available on [github](#) and [nbviewer](#).

Update on march 23, 2016. A [reddit reader suggested](#) the following way to compute exact probabilities:

```
from scipy import stats
```

Probability that a given color has at least 188 six:

```
stats.binom(1000, 1/6.0).sf(187)
```

Probability that one color among 20 has at least 188 six:

```
1 - (stats.binom(1000, 1/6.0).cdf(187)**20)
```

This uses the built-in binomial distribution from the scipy package. It provides the built-in probability mass function:

```
binom.pmf(k) = choose(n, k) * p**k * (1-p)**  
(n-k)
```

It also provides additional methods. The call to `sf(187)` gives the probability that one given color gets more than 187 six. The call to `cdf(187)` gives the probability to get less than 188 six.

[Add a Comment](#) | [More Actions](#)

Comments (0)

[Add a Comment](#) | [More Actions](#)

There are no comments to display

[Previous Entry](#) | [Main](#) | [Next Entry](#)

[Contact](#)

[Privacy](#)

[Terms of use](#)

[Accessibility](#)

[Cookie
Preferences](#)

Start your free trial