Different effect sizes for different ttests

November 14, 2025

1 Cohen's d for paired t-tests and one-sample t-tests

Mathematically, there should be no difference between a paired t-test, and a one-sample t-test comparing manually-calculated difference scores to zero. Indeed, if we generate some data X and some data Y and run both of these tests, we find that the t-statistic, p-value, and confidence intervals are all the same. However, the measure of effect size, Cohen's d, is slightly different. The reason for this is that although the t-test itself if mathematically equivalent, `pingouin` uses a different formula to calculate Cohen's d in these two cases. First, let's generate some data…

[9]:
```
import pandas as pd
import numpy as np
import pingouin as pg
import warnings
import numpy as np

warnings.filterwarnings("ignore")

np.random.seed(42)  # I'm a nerd
df = pd.DataFrame({
    'X': np.random.randn(20),  # 20 samples from standard normal
    'Y': np.random.rand(20)    # 20 samples from uniform[0,1)
})
df['D'] = df['X'] - df['Y']
df.head()
```

[9]:
```
          X         Y         D
0  0.496714  0.456070  0.040644
1 -0.138264  0.785176 -0.923440
2  0.647689  0.199674  0.448015
3  1.523030  0.514234  1.008795
4 -0.234153  0.592415 -0.826568
```

Now, let's run a paired t-test and a one-sample t-test:

[10]:
```
modpaired = pg.ttest(df['X'], df['Y'], paired=True)
modsingle = pg.ttest(df['D'], 0)
```

Here are the paired t-test results…

[11]:
```
modpaired
```

[11]:
```
                T  dof alternative     p-val          CI95%   cohen-d   BF10  \
T-test -2.858382   19   two-sided  0.010056  [-1.1, -0.17]  0.885538  5.099

           power
T-test  0.963531
```

and here are the one-sample t-test results...

[12]:
```
modsingle
```

[12]:
```
                T  dof alternative     p-val          CI95%   cohen-d   BF10  \
T-test -2.858382   19   two-sided  0.010056  [-1.1, -0.17]  0.639154  5.099

           power
T-test  0.773837
```

Sure enough, the t-value, p-value, and CI's are all the same, but the effect size is greater when the test is run as a paired t-test.

For the paired samples t-test, **pingouin** uses the following forumula for d_{avg} to calculate Cohen's d:

$$d_{avg} = \frac{\overline{X} - \overline{Y}}{\sqrt{\frac{(\sigma_1^2 + \sigma_2^2)}{2}}}$$

using the mean of the original standard deviations $\frac{(\sigma_1^2+\sigma_2^2)}{2}$ in the denominator. This allows us to interpret the effect size in the context of the original scale in which X and Y were measured.

We can calculate this manually like so:

[13]:
```
sigmax = np.std(df['X'], ddof=1)
sigmay = np.std(df['Y'], ddof=1)
var = (sigmax**2 + sigmay**2)/2
d_avg = abs(round((np.mean(df['X']) - np.mean(df['Y']))/np.sqrt(var),6))
print('Cohen\'s d (average):', d_avg)
```

```
Cohen's d (average): 0.885538
```

For the one-sample t-test, on the other hand, **pingouin** uses a different formula to calculate the effect size:

$$d_{diff} = \frac{\overline{D}}{\sigma_D}$$

This version of Cohen's d uses the standard deviation of the difference score in the denominator, rather than the mean of the standard deviations of the original measures. Using this formula, we shift the focus to the size of the mean difference relative to the variation in the difference scores. We can calculate this manually like this:

```
[14]: d_diff = abs(round(np.mean(df['D'])/np.std(df['D'], ddof=1),6))
      print('Cohen\'s d (difference):', d_diff)
```

```
Cohen's d (difference): 0.639154
```

I imagine this is a corner case that we are unlikely to run into often, since most people will use a paired t-test to compare the same participants in two conditions, but it is nice to see that there *is* a reason for this unsettling difference! Now, just don't ask me why the power is also different for the two tests..

I thought I said don't ask me!

Ok, it's because power is derived in part from the effect size, so… different effect sizes, different power :-)