Relationship between p-values and confidence intervals

Xavier Dupré
http://www.xavierdupre.fr/

7 mai 2013

Résumé

This document explains the relationship between p-value and confidence intervals. It goes on with the specific case of a binamial law. Assuming we want to determine whether or not two binomial laws are significantly different, how many observations we need to get the p-value under a given threshold.

Table des matières

1 Confidence interval

The term *p-value* is very popular in the world of search engines. I usually prefer confidence interval 95%, I think it is easier to understand. Plus, because p-Value are real values, we could be tempted to compare them and it is usually wrong. On the other hand, it is more difficult to compare confidence intervals, especially if they are related to complete different variables. Their nature prevents you from doing that. Howerver p-Values and confidence interval are similar : they tell you whether or not a metric difference is significant.

Usually, it starts from a set of identically distributed random variables $(X_i)_{1 \leqslant i \leqslant N}$. We then estimate the average $\widehat{\theta}_N = \frac{1}{N} \sum_{i=1}^{N} X_i$ and we ask the question : is $\widehat{\theta}_N$ null ? In others terms, we want to know if the average is significantly different from zero. If the random variable X follows a random law which has a standard deviation [1], we can use the central limit theorem [2] which tells us :

1. Not all of them have a standard deviation. For example, if X follows a Cauchy law, $\mathbb{E}\left(X^2\right) \sim \int \frac{x^2}{1+x^2} dx$ which does not exist. This remark also concerns every distribution known as heavy tail distribution.
2. See http://en.wikipedia.org/wiki/Central_limit_theorem.

$$\sqrt{N}\widehat{\theta}_N \underset{N\to\infty}{\longrightarrow} \mathcal{N}(0,\sigma) \tag{1}$$

If $Y \sim \mathcal{N}(0,\sigma)$, then we have $\mathbb{P}(|Y| \leqslant 1.96) = 0.95$. That is why we can say :

$$\widehat{\theta}_N \text{ is not null with 95\% confidence if } \sqrt{N}\frac{|\widehat{\theta}_N|}{\sigma} > 1.96 \tag{2}$$

And the confidence intervalle at 95% would be :

$$\left[-\frac{1.96\sigma}{\sqrt{N}}, \frac{1.96\sigma}{\sqrt{N}}\right] \tag{3}$$

When $\mathbb{E}\left(\widehat{\theta}_N\right) = \theta_0 \neq 0$, it becomes :

$$\sqrt{N}\left[\widehat{\theta}_N - \theta_0\right] \underset{N\to\infty}{\longrightarrow} \mathcal{N}(0,\sigma) \tag{4}$$

We usually want to check if the mean is equal to a specific value using a statistical test :

$$\begin{aligned} H0 :&\ \widehat{\theta}_N = \theta_0 \\ H1 :&\ \widehat{\theta}_N \neq \theta_0 \end{aligned}$$

We validate $H0$ if :

$$\widehat{\theta}_N \in \left[\theta_0 - \frac{1.96\sigma}{\sqrt{N}}, \theta_0 + \frac{1.96\sigma}{\sqrt{N}}\right] \tag{5}$$

2 p-value

With confidence intervals, you first choose a confidence level and then you get an interval. You then check if your value is inside or outside your interval. Inside, the gain is not significant, outside, it is. With a p-value, we consider the problem the other way : given $\widehat{\theta}_N$, what is the probability that the difference $\left|\widehat{\theta}_N - \theta_0\right|$ is significant ? Let's consider Y following a normal law $\mathcal{N}(0,1)$. We are looking for :

$$\mathbb{P}\left(|Y| > \sqrt{N}\frac{|\widehat{\theta}_N|}{\sigma}\right) = \alpha \tag{6}$$

α is the p-value.

$$\alpha = 1 - \int_{-\beta_N}^{\beta_N} \frac{1}{\sqrt{2\pi}} e^{\frac{-x^2}{2}} dx = 2 \int_{\beta_N}^{\infty} \frac{1}{\sqrt{2\pi}} e^{\frac{-x^2}{2}} dx \tag{7}$$

$$\text{where } \beta_N = \sqrt{N} \frac{|\widehat{\theta}_N|}{\sigma}$$

At this point, we should not forget that we use a theorem which tells us that $\sqrt{N}\frac{|\widehat{\theta}_N|}{\sigma} \sim \mathcal{N}(0,1)$ when $N \to \infty$, which means everything we said is true when N is great enough.

3 Significant difference between samples mean

Usually, we do not want to test if an average is null but if the difference between two averages is null. We consider two random samples having the same size, each of them described by (X_i) and (Y_i). All variables are independant. (X_i) are distributed according the same law, we assume the same for (Y_i). We expect the following difference to be null.

$$\widehat{\eta}_N = \frac{1}{N} \sum_{i=1}^{N} X_i - \frac{1}{N} \sum_{i=1}^{N} Y_i = \frac{1}{N} \left[\sum_{i=1}^{N} X_i - Y_i \right] \tag{8}$$

Considering expression (8), we can applying the central limit theorem on variable $Z = X - Y$, we get $(eta_0 = 0)$:

$$\sqrt{N}\widehat{\eta}_N \underset{N \to \infty}{\longrightarrow} \mathcal{N}\left(\eta_0, \sqrt{\frac{\mathbb{V}(Z)}{N}}\right) \tag{9}$$

If both samples do not have the same number of observations, this expression becomes :

$$\sqrt{N}\widehat{\eta}_N \underset{\substack{N_1 \to \infty \\ N_2 \to \infty \\ \frac{N_1}{N_2} \to x}}{\longrightarrow} \mathcal{N}\left(\eta_0, \sqrt{\frac{\mathbb{V}(X)}{N_1} + \frac{\mathbb{V}(Y)}{N_2}}\right) \tag{10}$$

4 Application on binomial variables

A binomial variable $X \sim \mathcal{B}(p)$ is defined by :

$$\mathbb{P}(X = 0) = 1 - p$$
$$\mathbb{P}(X = 1) = p$$

Let's consider two series of observations $(X_i) \sim \mathcal{B}(p)$ and $(Y_i) \sim \mathcal{B}(q)$. We assume $p \neq q$ and we want to determine how many observations we need to get a p-value below 5%. We know that

$\mathbb{V}(X_i) = p(1-p)$ and $\mathbb{V}(Y_i) = q(1-q)$. Table 1 shows the values. First column contains values for p, first row contains values for $q - p$. We also assume we have the same number N of random observations for each variable. The statistical test cen be defined like following :

$$\begin{aligned} H0: &\ p = q = p_0 \\ H1: &\ p \neq q \end{aligned}$$

If H0 is true, then :

$$\sqrt{N}\widehat{\theta}_N \underset{N\to\infty}{\longrightarrow} \mathcal{N}\left(0, \sqrt{p_0(1-p_0)}\sqrt{\frac{1}{N_1} + \frac{1}{N_2}}\right) \tag{11}$$

p/d	**-0.200**	**-0.100**	**-0.020**	**-0.010**	**-0.002**	**-0.001**	**0.001**	**0.002**	**0.010**	**0.020**	**0.100**	**0.200**
0.05			913	3650	91235	364939	364939	91235	3650	913	37	10
0.10		70	1729	6915	172866	691463	691463	172866	6915	1729	70	18
0.15		98	2449	9796	244893	979572	979572	244893	9796	2449	98	25
0.20	31	123	3074	12293	307317	1229267	1229267	307317	12293	3074	123	31
0.25	37	145	3602	14406	360137	1440548	1440548	360137	14406	3602	145	37
0.30	41	162	4034	16135	403354	1613413	1613413	403354	16135	4034	162	41
0.35	44	175	4370	17479	436966	1747864	1747864	436966	17479	4370	175	44
0.40	47	185	4610	18440	460976	1843901	1843901	460976	18440	4610	185	47
0.45	48	191	4754	19016	475381	1901523	1901523	475381	19016	4754	191	48
0.50	49	193	4802	19208	480183	1920730	1920730	480183	19208	4802	193	49
0.55	48	191	4754	19016	475381	1901523	1901523	475381	19016	4754	191	48
0.60	47	185	4610	18440	460976	1843901	1843901	460976	18440	4610	185	47
0.65	44	175	4370	17479	436966	1747864	1747864	436966	17479	4370	175	44
0.70	41	162	4034	16135	403354	1613413	1613413	403354	16135	4034	162	41
0.75	37	145	3602	14406	360137	1440548	1440548	360137	14406	3602	145	37
0.80	31	123	3074	12293	307317	1229267	1229267	307317	12293	3074	123	31
0.85	25	98	2449	9796	244893	979572	979572	244893	9796	2449	98	
0.90	18	70	1729	6915	172866	691463	691463	172866	6915	1729	70	
0.95	10	37	913	3650	91235	364939	364939	91235	3650	913		

Table 1 : *Given a binomial law with parameter p and a difference d, this table gives the number of observations needed on both sides to get a significant difference assuming p is the expected pourcentage (see program section 7, page 5).*

5 Estimate a p-value by using the distribution function

Expresion (7) gives a way to estimate the p-value. Computing the integral is not always possible but there is a way to do it using Monte Carlo method. Let's assume $X \sim \mathcal{N}(0,1)$. We denote f_X as the density function of X. We also consider an interval $I = [-a, a]$. Then we have $f(a) = f(-a)$ and :

$$\mathbb{P}(X \in I) = \mathbb{P}(|X| \leqslant a) = \mathbb{P}(f(X) \geqslant f(a)) \tag{12}$$

This is true because f is decreasing for $x > 0$. The p-value α for a estimator β using Monte Carlo method is :

$$\frac{1}{N}\sum_{i=1}^{N} \mathbb{1}_{\{f(X_i) < f(\beta)\}} \longrightarrow \alpha \tag{13}$$

Assuming every $(X_i)_i$ follows a normal law $\mathcal{N}(0,1)$.

6 Correlated variables

Let's assume we now have a vector a correlated variables $X = (X_1, ...X_d)$ drawn following a law $\mathcal{N}(\theta_0, \Sigma)$.

The central limit theorem is still valid :

$$\sqrt{N}\widehat{\theta}_N \underset{N\to\infty}{\longrightarrow} \mathcal{N}(\theta_0, \Sigma) \tag{14}$$

We know estimators for the average and the covariance matrix defined as follows :

$$\widehat{\theta_N} = \frac{1}{n}\sum_{i=1}^{N} X_i \tag{15}$$

$$\widehat{\Sigma_N} = \frac{1}{n}\sum_{i=1}^{N}(X_i - \widehat{\theta_N})(X_i - \widehat{\theta_N})' \tag{16}$$

We usually want to check if :

$$H0 : \ \widehat{\theta}_N = \theta_0$$
$$H1 : \ \widehat{\theta}_N \neq \theta_0$$

If Λ is diagonal matrix of Σ (diagonal matrix with eigen values). All eigne values are real and positive, we then define :

$$\Sigma = P\Lambda P' \text{ and } \Sigma^{\frac{1}{2}} = P\Lambda^{\frac{1}{2}}P' \tag{17}$$

We consider $Z_i = (X_i - \widehat{\theta_N})\Sigma^{-\frac{1}{2}}$. We then have : $\mathbb{E}(Z_i) = 0$ and $\mathbb{V}(Z_i) = I_2$ where I_2 is the identity matrix. We could now consider each dimension of Z_i independently as illustrated in Figure 1 : it shows the difference on an example if we consider the correlation of two variables correlated such as $\Sigma = \begin{pmatrix} 0.1 & 0.05 \\ 0.05 & 0.2 \end{pmatrix}$.

But that would not be the best way to do it. The confidence interval for a couple of indenpendant gaussian (N_1, N_2) variables is an ellipse. Two independent normal $N_1^2 + N_2^2$ with a null mean and standard deviation equal to one follows a χ_2 law. Based on that, we can deduce a boundary for the confidence zone at 95%. Figure 2 shows this zone for a non-correlated couple and a correlated couple ($\Sigma = \begin{pmatrix} 0.1 & 0.05 \\ 0.05 & 0.2 \end{pmatrix}$).

Correction

7 Program which produces the table in Section 4

```
from scipy.stats import norm
```

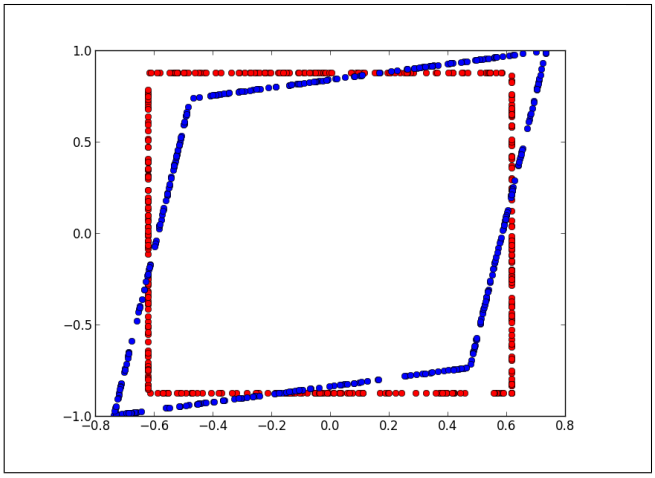

Figure 1 : *We assume we observe two Bernouilli variables correlated. Red points represents the area for which we would accept hypothesis H0 in case both variables are independant. Blue area represents the same but with the correlation. See Section 8.*

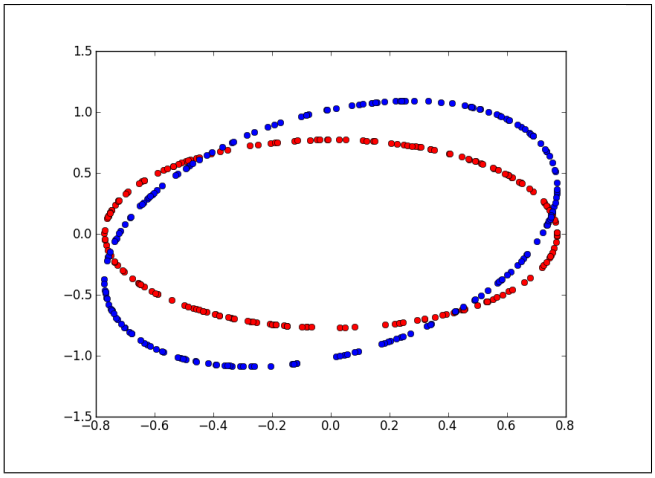

Figure 2 : *We assume we observe two Bernouilli variables correlated. Red points represents the area for which we would accept hypothesis H0 in case both variables are independant. Blue area represents the same but with the correlation. See Section 8.*

```
def pvalue (p, q, N) :
    theta   = abs(p-q)
    var     = p*(1-p)
    bn      = (2*N)**0.5 * theta / var**0.5
    ret     = (1 - norm.cdf(bn))*2
    return ret

def pvalue_N (p, q, alpha) :
    theta   = abs(p-q)
    var     = p*(1-p)
    rev     = abs(norm.ppf (alpha/2))
    N       = 2 * (rev * var**0.5 / theta)** 2
    return int(N+1)

def alphatable (ps, dps, alpha) :
    values = []
    for p in ps :
        row=[]
        for dp in dps :
            q = p+dp
            r = pvalue_N (p,q,alpha) if 1 >= q >= 0 else -1
            row.append (r)
```

```
        values.append (row)
    return values

def pprint(ps,dps,table, format, fileFormat = "latex") :
    text = []
    if fileFormat == "latex" :
        cc = "r" + ("|r" * len(dps))
        text.append("\\begin{tabular}{" + cc + "}")
        row  = [""] + [ r"\textbf{%1.3f}" % _ for _ in dps ]
        text.append("&".join(row) + r"\\\hline")
        for p,line in zip(ps,table) :
            row  = ["\\textbf{%1.2f}" % p] + \
                   [ format % _ if _ != -1 else ""  for _ in line ]
            text.append("&".join(row) + r"\\\hline")
        text.append("\\end{tabular}")
        return "\n".join(text)
    else :
        # TSV
        row  = [""] + [ r"%1.3f" % _ for _ in dps ]
        text.append("\t".join(row) )
        for p,line in zip(ps,table) :
            row  = ["%1.2f" % p] + \
                   [ format % _ if _ != -1 else ""  for _ in line ]
            text.append("\t".join(row) )
        return "\n".join(text)


if __name__ == "__main__" :
    print ("norm.ppf(0.025)",norm.ppf (0.025)) # -1.9599639845400545
    ps  = [0.001, 0.002] + [ 0.05*i for i in range (1,20) ]
    dps = [ -0.2, -0.1,  -0.02, -0.01, -0.002, -0.001,
             0.2,  0.1,   0.02,  0.01,  0.002,  0.001, ]
    dps.sort()
    t = alphatable (ps, dps, 0.05)
    print (pprint (ps, dps, t, "%d", "latex"))
    print (pprint (ps, dps, t, "%d", "html"))
```

8 Program which generates Figure 1

```
import numpy, matplotlib, random, pylab, math

def matrix_square_root(sigma) :
    eigen, vect = numpy.linalg.eig(sigma)
    dim = len(sigma)
    res = numpy.identity(dim)
    for i in range(0,dim) :
        res[i,i] = eigen[i]**0.5
    return vect * res * vect.transpose()

def chi2_level (alpha = 0.95) :
    N = 1000
    x = [ random.gauss(0,1) for _ in range(0,N) ]
    y = [ random.gauss(0,1) for _ in range(0,N) ]
    r = map ( lambda c : (c[0]**2+c[1]**2)**0.5, zip(x,y))
    r = list(r)
    r.sort()
    res = r [ int (alpha * N) ]
```

```
    return res

def square_figure(mat, a) :
    x = [ ]
    y = [ ]
    for i in range (0,100) :
        x.append ( a * mat[0][0]**0.5 )
        y.append ( (random.random ()-0.5) * a * mat[1][1]**0.5*2 )
        x.append ( -a * mat[0][0]**0.5 )
        y.append ( (random.random ()-0.5) * a * mat[1][1]**0.5*2 )

        y.append ( a * mat[1][1]**0.5 )
        x.append ( (random.random ()-0.5) * a * mat[0][0]**0.5*2 )
        y.append ( -a * mat[1][1]**0.5 )
        x.append ( (random.random ()-0.5) * a * mat[0][0]**0.5*2 )

    pylab.plot(x,y, 'ro')

    x = [ ]
    y = [ ]
    for i in range (0,100) :
        x.append ( a )
        y.append ( (random.random ()-0.5) * a*2 )
        x.append ( -a )
        y.append ( (random.random ()-0.5) * a*2  )

        y.append ( a )
        x.append ( (random.random ()-0.5) * a*2  )
        y.append ( -a )
        x.append ( (random.random ()-0.5) * a*2  )

    xs,ys = [],[]
    for a,b in zip (x,y) :
        ar = numpy.matrix( [ [a], [b] ] ).transpose()
        we = ar * root
        xs.append ( we [0,0] )
        ys.append ( we [0,1] )

    pylab.plot(xs,ys, 'bo')
    pylab.show()

def circle_figure (mat, a) :
    x = [ ]
    y = [ ]
    for i in range (0,200) :
        z = random.random() * math.pi * 2
        i = a * mat[0][0]**0.5 * math.cos(z)
        j = a * mat[0][0]**0.5 * math.sin(z)
        x.append ( i )
        y.append ( j )
    pylab.plot(x,y, 'ro')

    x = [ ]
    y = [ ]
    for i in range (0,200) :
        z = random.random() * math.pi * 2
        i = a * math.cos(z)
        j = a * math.sin(z)
        x.append ( i )
```

```
        y.append ( j )

    xs,ys = [],[]
    for a,b in zip (x,y) :
        ar = numpy.matrix( [ [a], [b] ] ).transpose()
        we = ar * root
        xs.append ( we [0,0] )
        ys.append ( we [0,1] )

    pylab.plot(xs,ys, 'bo')
    pylab.show()

if __name__ == "__main__" :
    level = chi2_level ()

    mat   = [ [0.1, 0.05], [0.05, 0.2] ]
    npmat = numpy.matrix(mat)
    root  = matrix_square_root (npmat)

    square_figure (mat, 1.96)
    circle_figure (mat, level)
```

9 Program which estimates the p-value

```
import random, math

def densite_gauss (mu, sigma, x) :
    e = -(x - mu)**2 / (sigma**2 * 2)
    d = 1. / ((2*math.pi)**0.5 * sigma)
    return d * math.exp (e)

def simulation_vector (N, mu, sigma) :
    return [ random.gauss(mu,sigma) for n in range(N) ]

def ratio (vector, x, fdensite) :
    under = 0
    above = 0
    fx    = fdensite(x)
    for u in vector :
        f = fdensite (u)
        if f >= fx : above += 1
        else : under += 1
    return float(above) / float (above + under)

if __name__ == "__main__" :
    x     = 1.96
    N     = 10000
    mu    = 0
    sigma = 1

    v = simulation_vector (N, mu, sigma)
    g = ratio (v, x, lambda y : densite_gauss (mu, sigma, y) )
    print (g)
```