The beautiful student’s t test probability
algorithm

G Rob Burgess

September 27, 2016

Years back I made a program that let you perform the student’s t test in a
web browser, back when java was young and java web applets were the thing.
Most of the program was easy to write, but there was no built in function for
getting the probability that two samples came from different populations for a
given t score and degrees of freedom.

We used to look these probabilities up on tables, and although the student’s
t distribution is pretty straightforward, integrating the probability density func-
tion to get the probability did not look like fun. Although it would disappoint
my physics advisor to say so, gamma functions scare me. I’m working on getting
over it.

I couldn’t and I still can’t find much on the web on a nice algorithm for
getting the probability, but I found a free to use library that did that so I used
that. It was closed source, so I couldn’t really see how it was done.

After a bit of searching a found a book “Statistical Computing in Pascal”
(D. Cooke, A.H. Craven, G.M. Clarke; 1985, London) which had a very nice
algorithm involving finite series and was exact. I still can’t find much on this
so I am reproducing it here.

Given a t-score of $t$ and degrees of freedom $k$, we define

$$\theta = \tan^{-1}\left(\frac{t}{\sqrt{k}}\right)$$

then probability $p$ is

$$p = \frac{1}{2} (1 + A)$$

where if $k = 1$

$$A = \frac{2\theta}{\pi}$$

and if $k > 1$ and odd

$$A = \frac{2}{\pi} \left[ \theta + \sin\theta \left( \frac{2}{3} \cos^3\theta + \ldots + \frac{2 \cdot 4 \ldots (k-3)}{3 \cdot 5 \ldots (k-2)} \cos^{k-2}\theta \right) \right]$$

and if $k$ is even
$A = \sin \theta \left[ 1 + \frac{1}{2} \cos^2 \theta + \frac{1 \cdot 3}{2 \cdot 4} \cos^4 \theta + \ldots + \frac{1 \cdot 3 \cdot \ldots (k-3)}{2 \cdot 4 \cdot \ldots (k-2)} \cos^{k-2} \theta \right]$ 

The nice thing is if you look at the above series for $A$ you see that each term for both series differs by a factor of $\frac{n-1}{n} \cos^2 \theta$ up to $k-2$ so for the algorithm you just carry a term through a loop, multiplying it by a factor and then summing it.

Here is the actual algorithm, in some sort of baloney pseudocode:

**pseudocode for a function to return a probability $p$ given a t-score $t$ and degrees of freedom $k$**

**variables:**
- `term` (float): for carrying running term
- `sum` (float): sum of terms for final $p$
- `theta` (float): a mishmash of $t$ and $k$
- `i` (integer): index for the while loop

**function stprob(t,k)**

```python
    term = k
    theta = arctan(t/sqrt(k))
    sum = 0.0
    if k > 1 then
        if (k is odd) then
            i = 3
            term = cos(theta)
        else
            i = 2
            term = 1.0
        endif
        sum = sum + term
    while (i < k) do
        term = term * [cos(theta)]^2 * (i-1)/i
        sum = sum + term
        i = i + 2
    endwhile
    sum = sum * sin(theta)
    if (k is odd) then
        sum = (2/pi) * (sum + theta)
    p = 0.5 * (1 + sum)
```

return $p$

So that’s it, enjoy!