TDD Demo -- Implementing a t-distribution

Specifications

Background: A t-distribution is a probability density function that approximates a normal distribution for a small population. The t-curve is described by an equation that is characterized by the numbers of samples in the population, referred to as the degrees of freedom. How closely the curve matches a normal curve depends on the degrees of freedom. The larger the number of degrees of freedom, the closer the t-curve comes to a normal curve. The t-distribution is used frequently in planning to calculate the likelihood of a forecasted software effort estimate. Likelihood, calculated as a probability, is determined by determining the area under a t-curve.

class Sample

Class *Sample* is an abstraction that represents a body of data elements upon which statistical characteristics are determined.

Constructor

Sample(int n)

Creates an instance of *Sample* that represents a population of size *n.*

Parameters:	"n" is an integer .GE. 2 and .LT. 30. Arrives unvalidated. Mandatory.	
Returns:	An instance of Sample with n degrees of freedom.	
State change:	Retains the number of degrees of freedom.	
Exceptions:	Throws:	ValueError (“invalid n")
	Rasied when:	"n" violates its specifications
	Exit conditions:	No instance is created.

Instance Methods

double p(float t, int tails)

Returns the probability of a t-distribution, given a value of t and the number of tails.

Parameters:	"t" is a numeric value .GT. 0. Arrives unvalidated. Mandatory	
	"tails" is a numeric having the value 1 or 2. Arrives unvalidated. Optional, defaults to 1.	
Returns:	A floating point value .GE. 0 and .LE. 1.0	
State change:	No state change	
Exceptions:	Throws:	ValueError(“Invalid t")
	Rasied when:	t violates its specifications
	Exit conditions:	The instance undergoes no state change.
	Throws:	ValueError("Invalid tails")
	Rasied when:	"tails" is not 1 or 2
	Exit conditions:	The instance undergoes no state change.

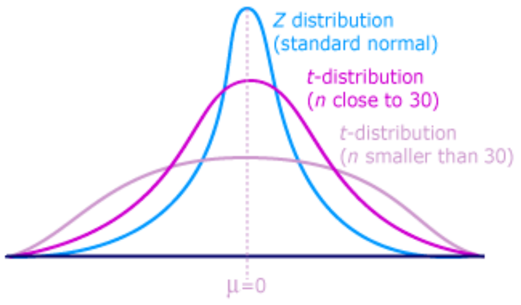
Architecture

The function which describes the t curve is

$$f(u) = \left(\frac{\Gamma\left(\frac{n+1}{2}\right)}{\left(\frac{n}{2}\right)\sqrt{n\pi}}\right)\left(1+\frac{u^2}{n}\right)^{-\frac{n+1}{2}}$$

where *n* is the degrees of freedom and Γ is a function defined recursively as Γ(x) = (x-1)Γ(x-1), with termination conditions Γ(1)=1 and Γ(1/2)=√π.

Graphing this function from -∞ to ∞ results in the following:



source: http://ci.columbia.edu/ci/premba_test/c0331/s7/s7_4.html .

Because we are interested in probability, we want to calculate the area under the curve as follows:

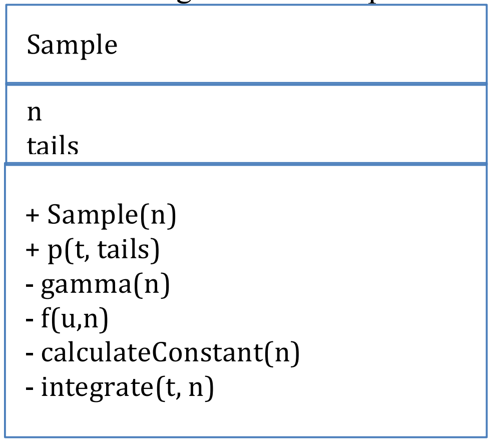
$$prob = \begin{cases} if\ tails == 1: \left(\frac{\Gamma\left(\frac{n+1}{2}\right)}{\Gamma\left(\frac{n}{2}\right)\sqrt{n\pi}}\right)\int_{-\infty}^{t}\left(1+\frac{u^2}{n}\right)^{-\left(\frac{n+1}{2}\right)} du \\ if\ tails == 2: \left(\frac{\Gamma\left(\frac{n+1}{2}\right)}{\Gamma\left(\frac{n}{2}\right)\sqrt{n\pi}}\right)\int_{-t}^{t}\left(1+\frac{u^2}{n}\right)^{-\left(\frac{n+1}{2}\right)} du \end{cases} \quad (1)$$

where *tails* is {1,2}, *t* is the value along the t-axis, and *n* is the degrees of freedom.

In examining this from an architecture (i.e., high-level design or component) perspective, we see that we can decompose the calculation in the distinct parts:

$$\begin{tikzcd}
 & \left(\frac{\Gamma\left(\frac{n+1}{2}\right)}{\Gamma\left(\frac{n}{2}\right)\sqrt{n\pi}}\right) \arrow[r, dash] & \Gamma \\
prob \arrow[ur, dash] \arrow[dr, dash] & & \\
 & \int \arrow[r, dash] & \left(1+\frac{u^2}{n}\right)^{-\left(\frac{n+1}{2}\right)}
\end{tikzcd}$$

This leads us to the following notional component:



Sample
n tails
+ Sample(n) + p(t, tails) - gamma(n) - f(u,n) - calculateConstant(n) - integrate(t, n)

Implementation

There are two major parts to the formula, the segment to the left of the integral and the integral itself. The left-hand segment is fairly straightforward to calculate.
Example calculations of Γ are

$\Gamma(5) = 4*\Gamma(4) = 4*3*\Gamma(3) = 4*3*2*\Gamma(2) = 4*3*2*1*\Gamma(1) = 4*3*2*1*1 = 24$

and

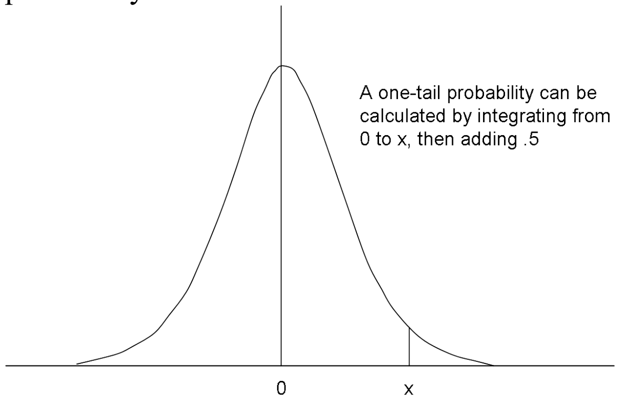
$\Gamma(5/2) = 3/2*\Gamma(3/2) = 3/2*1/2*\Gamma(1/2) = 3/2*1/2*\sqrt{\pi} = 1.329$

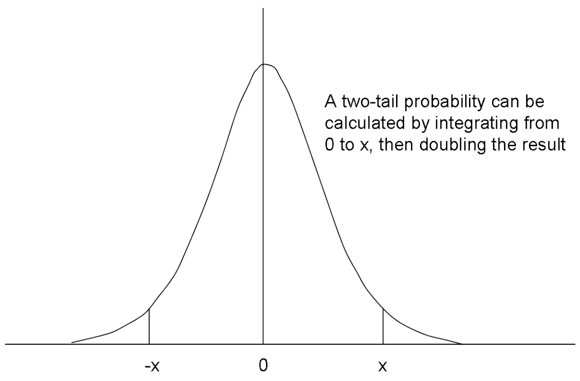
We will calculate the right-hand segment of the formula – the area under the curve – using a numerical integration technique known as Simpson's Rule. The concept of Simpson's Rule is to divide the curve into slices, then to calculate the total area of the slices. The general formula is

$$\int_{low}^{high} F(u)du = \frac{W}{3}\begin{bmatrix} f(low) + 4f(low+W) + 2f(low+2W) + 4f(low+3W) + \\ 2f(low+4W) + \cdots + 2f(high-2W) + 4f(high-W) + f(high) \end{bmatrix} \tag{2}$$

where W is the width of each slice.

The area under the curve is calculated by multiplying the result of the integration by the value obtained by calculating the equation to the left of the integral. The problem with this is that the integration is not feasible when the lower bound is minus infinity. We can get around this by taking advantage of the characteristics of the t-curve itself. First, the area under the entire t-curve, stretching from minus infinity to plus infinity is 1.0. The t-curve equation yields a value of 0.5 when the low bound is minus infinity and the high bound is 0. This means that running the integral from 0 to t, then adding 0.5 will calculate a one-tailed probability. Similarly, the curve is symmetrically reflected around the vertical axis at 0. This means integrating from 0 to t and then doubling the result will yield a two-tailed probability.


A one-tail probability can be calculated by integrating from 0 to x, then adding .5
0
x


A two-tail probability can be calculated by integrating from 0 to x, then doubling the result
-x
0
x

Below is a description of the numerical integration itself. In the algorithm, f(u) is $\left(1+\frac{u^2}{n}\right)^{-\left(\frac{n+1}{2}\right)}$, n is the number of degrees of freedom, ε is the termination condition, S is the number of slices in the curve, W is the width of each slice, highBound is the upper bound on the integration, and lowBound is the lower bound (which, as indicated above, is 0).

```
{
    let ε = 0.01
    let Simpson_old = 0
    let Simpson_new = ε
    set S to a value of your choice (a good starting value is 4)
    while abs((Simpson_new - Simpson_old ) / Simpson_new) > ε
    {
        Simpson_old = Simpson_new
        W = (highBound - lowBound) / S
        Simpson_new = (W/3) * (f(0) + 4f(0+W) + 2f(0+2W) + …
            + 4f(highBound-W) + f(highBound))
        S = S * 2
    }
    return Simpson_new  as the answer
}

Note:  ε may have to be adjusted to provide the desired amount of accuracy.
```

For example, with lowBound = 0, highBound=16, and S=4,

W = (16-0)/4 = 4

and

$\text{Simpson}_{\text{new}}$ = (4/3) * (f(0) + 4f(4) + 2f(8) + 4f(12) + f(16))

With lowBound = 0, highBound = 16, and S=8

W = (16-0)/8 = 2

and

$\text{Simpson}_{\text{new}}$ = (2/3) * (f(0) + 4f(2) + 2f(4) + 4f(6) + 2f(8) + 4f(10) + 2f(12) + 4f(14) + f(16))

The characteristics of the curve guarantee that $\text{Simpson}_{\text{new}}$ will converge with $\text{Simpson}_{\text{old}}$ given a high enough value of S.

Selected values of the t-distribution are shown in the table below:

	p(a)=	0.6	0.7	0.85	0.9	0.95	0.975	0.99	0.995
	p(a/2)=	0.2	0.4	0.7	0.8	0.9	0.95	0.98	0.99
degrees of freedom	1	0.3249	0.7265	1.9626	3.0777	6.3138	12.7062	31.8205	63.6567
	2	0.2887	0.6172	1.3862	1.8856	2.9200	4.3027	6.9646	9.9248
	3	0.2767	0.5844	1.2498	1.6377	2.3534	3.1824	4.5407	5.8409
	4	0.2707	0.5686	1.1896	1.5332	2.1318	2.7764	3.7469	4.6041
	5	0.2672	0.5594	1.1558	1.4759	2.0150	2.5706	3.3649	4.0321
	6	0.2648	0.5534	1.1342	1.4398	1.9432	2.4469	3.1427	3.7074
	7	0.2632	0.5491	1.1192	1.4149	1.8946	2.3646	2.9980	3.4995
	8	0.2619	0.5459	1.1081	1.3968	1.8595	2.3060	2.8965	3.3554
	9	0.2610	0.5435	1.0997	1.3830	1.8331	2.2622	2.8214	3.2498
	10	0.2602	0.5415	1.0931	1.3722	1.8125	2.2281	2.7638	3.1693
	15	0.2579	0.5357	1.0735	1.3406	1.7531	2.1314	2.6025	2.9467
	20	0.2567	0.5329	1.0640	1.3253	1.7247	2.0860	2.5280	2.8453
	30	0.2556	0.5300	1.0547	1.3104	1.6973	2.0423	2.4573	2.7500

The table gives the one- and two-tailed probabilities for value of t at various degrees of freedom. For example, suppose we want to find the probability where t=1.1342 and degrees of freedom=6. Go along the row designed as "6" and find 1.1342. The top of column where 1.1342 was found gives the probabilities. "p(a)" indicates the probability for a one-tailed test and "p(a/2)" indicates the probability for a two-tailed test. Given this, the one-tailed probability is 0.85 and the two-tailed probability is 0.7.

Cast in terms of the requirements of the software,

```
Sample mySample(6);    ← This instantiates the class with 6 degrees of freedom
mySample.p(1.1342,1);  ← This should result in 0.85
mySample.p(1.1342,2);  ← This should result in 0.7
```