Taming Statistics with Defined Operators

Stephen M. Mansour
University of Scranton and The Carlisle Group, Inc.

Abstract

One problem with statistics is the proliferation of functions. In Excel there are at least 87 such functions. And that doesn't include some mathematical functions used in statistics such as SQRT, SUM and SUMSQ. The use of operators in APL reduces the number of primitive functions needed in the system. There are also some functions in APL that are not particularly useful in and of themselves (such as left tack and right tack), but they can be extremely useful as operands. Operators are a natural way of thinking about applying functions, and we would like to extend these concepts to statistics.

Statistical functions are numerous, but they can be categorized into groups. We can reduce the number of statistical functions by creating a few functions and designing operators to modify them. For instance most statistical distributions come in three general forms, the density function, the distribution function and the inverse or critical value function. The density function is not terribly useful in and of itself, but functions derived from it are extremely useful. In addition to this we can generate random variables and perform goodness of fit tests based on a particular distribution. In order to do this all distributions must have the same general form. The left argument of the function will contain the parameters (if any), while the right argument of the function should be pervasive and contain particular values of the random variable. For discrete distributions, we use the probability mass function; for continuous distributions the density function. The shape and structure of the result should be identical to the shape and structure of the right argument.

Although I have implemented most of the functions and operators in APL; this paper is primarily a design document. Many of these functions could be replaced with calls to R for example. They could also be used as a front end to R or some other statistical package.

Discrete Distributions

Suppose we write an APL function `binomial` which generates the probability mass function for the Binomial distribution. The syntax is:

```
      n p binomial x
```

where n and p are the number of trials and the probability of success respectively, and x is the number of successes. In the following example we show the probability of 2 successes given 3 trials and a ¼ probability of success:

```
      3 0.25 binomial 2
0.0140625
```

All distributions are scalar functions with respect to the right argument. The parameters on the left apply to the function as a whole; thus the following expression shows the probabilities of 0, 1 and 2 successes respectively:

```
     3 0.25 binomial 0 1 2
0.421875 0.421875 0.140625
```

Now let us create a dyadic operator called `probability.` Its left operand is a statistical distribution function, while its right operand is a relational function. The left argument consists of the parameters of the statistical distribution, while the right argument consist of the boundary value(s).

```
     3 0.25 (binomial probability =) 2
0.0140625
```

Note that the parenthesis above are redundant; we simply used them explain the syntax, separating the left and right operands from the left and right arguments. From the user's point of view, the below expression is easy to understand:

```
     3 0.25 binomial probability = 2
0.0140625
```

What is the probability that a baseball player bats safely in a 9-inning game? Suppose his batting average is .206. The average player usually gets 4 at bats during a game and to bat safely he must get at least one hit. This will happen about 60% of the time:

```
     4 0.206 binomial probability ≥ 1
0.60255
```

We can express cumulative probability by using the operand `≤` also without the parentheses:

```
     3 0.25 binomial   ≤ 2
0.984375
```

Note that this is equivalent to:

```
     +/3 0.25 binomial 0 1 2
0.984375
```

We can also calculate the complements by using the following syntax:

```
     3 0.25 binomial probability ≠ 2
0.859375
```

```
     3 0.25 binomial probability > 2
0.015625
```

In addition to the six standard relational functions, we can also define "between" which excludes the endpoints and "include" which does not. We also may use the membership function. Thus:

```
      between←{¯1=×/×⍺∘.-⍵}
      10 0.25 binomial probability between 3 6
0.2043972015
      include←{1≠×/×⍺∘.-⍵}
      10 0.25 binomial probability include 3 6
0.4709014893
      10 0.25 binomial probability ∈ 4 5
0.2043972015
```

For a multinomial or custom distributions, we first define a probability distribution. A probability distribution consists of two columns, distinct values and their associated probabilities. The values can be either numeric or text. The probabilities are listed in the same order as the values and must be between 0 and 1 and must also sum to 1. If they sum to a value less than one, a category of 'other' is implicitly added.

```
       COLORS←⊃'Brown' 'Yellow' 'Red' 'Blue' 'Orange' 'Green'
       PROPORTION
0.13 0.14 0.13 0.24 0.2 0.16
       COLORS PROPORTION multinomial probability = 'Red'
0.13
       COLORS PROPORTION multinomial probability ≠ 'Green'
0.84
```

If the values are specifically the set (0,1,…N), we do not need to specify them explicitly:

```
      .1 .5 .4 multinomial probability = 1
.5
      .1 .5 .4 multinomial probability < 2
.6
```

Continuous Distributions

The normal density function is ambivalent and based on the standard normal distribution unless the mean and standard deviation are the explicitly specified as the left argument. Thus

```
      normal ¯2 ¯1 0 1 2          ⍝ Standard Normal density
0.053991 0.24197 0.39894 0.24197 0.053991
      8 3 normal 4 6 8 10 12      ⍝ mean=8, std deviation=3
0.16401 0.31945 0.39894 0.31945 0.16401
```

Applying the `probability` operator, notice that strict inequalities make no difference and that the probability that a continuous random variable is a particular value is zero:

```
       normal probability ≤ 2 ⍝ Cumulative probability
0.84134
       normal probability < 2 ⍝ Strict inequality doesn't matter
0.84134
```

```
      normal probability = 2 ⍝ Probability of exact value is zero.
0
```

We can also use the operand **between** to determine the probability between two values (note that include makes no difference for continuous distributions):

```
       normal probability between ¯1.96 1.96 ⍝ Interval probability
0.95
       8 2 normal probability > 5 ⍝ Non-standard upper-tail normal
probability
0.93319
```

Critical Value Operator

We now introduce the **criticalValue** operator to calculate cumulative probabilities from critical values. Sometimes we want to use the upper-tail probability as input; other times we may want to use the lower-tail probability. The dyadic operator **criticalValue** takes the density function on the left and a relational function on the right. To indicate the critical value which cuts off the upper tail we use < ; the reasoning behind this is that if the critical value is less than exactly 5% of all values of the normal distribution for example, we use the following expression:

```
       normal criticalValue < 0.05 ⍝ Critical value less than 5% of all
values
1.645
```

For a lower-tail critical value, we would use > for the same reason:

```
       normal criticalValue > 0.05 ⍝ Critical value greater than 5%
(lower-tail)
¯1.645
```

For two-tail critical value we can use ≠ or **between** ; however this becomes problematic for asymmetrical distributions:

```
      normal criticalValue ≠ 0.05 ⍝ Two-tail critical value (2.5% in
each tail)
1.96
```

randomVariable Operator

The randomVariable operator generates independent random variables based on its operand. When the operand is the uniform distribution it is similar to the roll (?) function in APL. The right argument is an integer indicating the sample size.

```
      normal randomVariable 5      ⍝ Generate 5 standard normal r.v.'s
0.58949 0.35082 0.1357 0.88211 ¯1.7619

      3 poisson randomVariable 10 ⍝ Generate 10 poisson r.v.'s (mean=3)
5 4 5 2 3 4 1 2 2 0
```

Comparison to Excel and R

To illustrate the power of using operators on distribution functions can be illustrated in the following table showing the equivalent Excel functions for the t-distribution:

Desc	Excel	R	APL Operator
Density	T.DIST(DF,X,0)	dt(X,df=DF)	DF tDist X
Cum Prob	T.DIST(DF,X,1)	pt(X,df=DF)	DF tDist prob ≤ X
2-Tail Pr	T.DIST.2T(DF,X)	2*pt(X,df=DF)	
Upper Tail	T.DIST.RT(DF,X)	qt(X,df=df, lowertail=FALSE)	DF tDist prob > X
Crit. Value	T.INV(DF,P)	qt(P, df=DF, lower,tail=FALSE)	DF tDist criticalValue< P
2-tail c.v.	T.INV.2T(DF,P)	qt(P/2,df=DF, lower.tail=FALSE)	DF tDist criticalValue≠ P
Hyp test	T.TEST(X1,X2)	t.Test(X1,X1, paired=FALSE,mu=0)	mean hypothesis X1=X2

Table 1: Statistical Distributions with Excel, R and APL equivalents

Statistical Functions and Confidence Intervals

A summary function takes a vector and produces a scalar result. Thus the functions sum, average proportion, variance, max and min fall into this category. Many of these summary functions produce sample statistics. When the operator "estimate" is applied, the result will be the 95% confidence interval for the unknown parameter which the statistic represents. The left argument will override the default 95% confidence level.

```
      mean DATA←25 30 10 22 23 29 29 28 ⍝ sample mean
24.5
     mean confInterval DATA      ⍝ confidence interval (95%)
19.009 29.991
      99 mean confInterval DATA  ⍝ 99% confidence interval
16.373 32.627
      proportion B←?1000⍴2  ⍝ sample proportion (p0=0.5)
0.497
      proportion confInterval B ⍝ 95% confidence interval contains 0.5
0.46601 0.52799
      V←variance DATA          ⍝ sample variance
```

```
      V*÷2                        ⍝ sample standard deviation
6.5683
      variance confInterval DATA  ⍝99% confidence int for variance
      VV*÷2                         ⍝ confidence interval for std dev
4.3428 13.368
```

Hypothesis Tests

We can also perform hypothesis tests using defined operators. Hypothesis tests typically involve comparing a sample statistic to a population parameter. Thus we require a summary function to calculate the statistic and a relational function to do the comparison. This suggests that we use a dyadic operator .

We define the functions mean, variance, standardDeviation and proportion.

The relational functions in APL return a 1 if the relationship is true and a 0 if the relationship is false. In hypothesis testing we don't know for sure if the relationship is true or false, but we can say with a degree of certainty. The p-value is defined as the probability that you obtain the result you did given the null hypothesis. The larger the p-value, the more certain you are that the null hypothesis is true. Thus instead of producing a 1 or a 0, the hypothesis operator produces a value between those two extremes and leaves it up to the researcher to make a decision. This is known as fuzzy logic.

Suppose we have a pool of 10,000 mortgages and we wish to test the hypothesis that the average loan balance is $72,000. We set up the null and alternative hypotheses:

$$H_0 : \mu = 72000 \qquad H_1 : \mu \neq 72000$$

The variable `BALANCE` is a vector containing the balances of 10000 mortgages.

```
       BALANCE←72000 20000 normal randomVariable 30
       mean BALANCE
76053
```

The expression to perform the test and calculate the p-value is:

```
       BALANCE (mean hypothesis =) 70000
0.12005
```

Again, the parentheses are redundant. We show them to illustrate the syntax of the hypothesis operator. Suppose we take the same 10000 mortgages and test the hypothesis that the average interest rate is greater than 7.25%. This is an upper-tail test. We would then set up the null and alternative hypopthesis as follows:

$$H_0 : \mu \leq 7.25 \qquad H_1 : \mu > 7.25$$

The expression to perform this test is:

```
      RATE mean hypothesis ≤ 7.25
```

```
0.0001
```

Since the p-value is very small we would reject the null hypothesis and conclude that the average interest rate is greater than 7.25%.

Suppose we wanted to test that the proportion of high-interest mortgages less than 40%. We would state the null and alternative hypothesis as follows:

$$H_0 : \mathrm{p} \geq 0.4 \qquad H_1 : \mathrm{p} < 0.4$$

Let us define a high-interest mortgage as one whose interest rate is greater than 8.5%. The APL expression to perform this test includes a Boolean vector as the left argument:

```
      (RATE>8.5) proportion hypothesis ≥ 0.15
0.0236
```

Two-Sample Tests

Suppose we have two samples and we wish to test if the means of the two groups are the same. The null and alternative hypotheses are:

$$H_0 : \mu_A = \mu_B \qquad H_1 : \mu_A \neq \mu_B$$

Let us define two sample groups and perform a hypothesis test:

```
     GROUPA←2 5 3 8 1
     GROUPB←4 3 2 2
     GROUPA mean hypothesis = GROUPB
0.46564
```

To obtain a confidence interval for the difference, we use the estimate operator:

```
     mean confInterval GROUPA GROUPB
¯2.3691 4.4691
```

To perform a test on matched samples we do the following:

$$H_0 : \mu_d = 0 \qquad H_1 : \mu_d \neq 0$$

```
     BEFORE←2 5 3 8 1
     AFTER←4 3 2 5 2

     (AFTER-BEFORE) mean hypothesis = 0
0.55289
     mean confInt AFTER-BEFORE
¯3.1748 1.9748
```

We can also test for equality of two variances:

```
        GROUPA variance hypothesis = GROUPB
```

```
0.055783
```

GoodnessOfFit Operator

The goodness of fit operator takes a statistical distribution as its left operand and a vector as its right argument. The result is the p-Value of the Chi-Square Distribution. For the uniform distribution, the data can be text or numeric. If the numerical data are non-integer data, the test will be for a continuous uniform distribution; otherwise we will test for a discrete uniform distribution.

```
    DATA← 2 1 2 7 4 1⌿COLORS
    uniform goodnessOfFit DATA
0.091703
```

For non-uniform data we use the multinomial function with its probability distribution as the left argument. The following example shows that the color distribution of the sample fits the proportions of the multinomial distribution:

```
     COLORS PROPORTIONS multinomial goodnessOfFit DATA
0.53179
```

To test if data are bell-shaped, we can use the normal distribution. Notice that the APL function roll produces a uniform (not-bell shaped) distribution:

```
      normal goodnessOfFit ?100⍴100
0.047736
```

Data Representation

Sometimes raw data are not available. In many cases one must use summary information. There are generally two ways to represent sample data. The first is with a frequency distribution which can be represented by a two column matrix to distinguish it from a data set. The second is with summary statistics; i.e. sample size, mean and variance. This can be represented with a namespace consisting of named variables. Both of these representations can be substituted for raw data as arguments to any statistical function or operator. Since they are structurally different, the functions/operators know that they represent summary data.

```
VALUES←⍳7


COUNTS ← 1 3 4 9 8 4 3
FR←VALUES ,⍤0⊢COUNTS

PS←⎕NS ''
PS.count← 30
PS.mean ← 3.4
PS.variance ← 2.3172
```

Conclusion

Many statistical packages have large libraries of functions to handle the endless variety of probability calculations, parameter estimates and statistical tests. Operators are a natural way express many of these statistical concepts and allow us to reduce the number of functions—for example we need only one function for each statistical distribution. From this we can handle the probability density, cumulative distribution, inverse function, simulation and goodness-of-fit. Probabilities and statistical tests also have upper-tail, lower-tail and two-tail varieties.

	probability	criticalValue	randomVariable	goodnessOfFit
hyperGeometric	<,<=,=,>=,>,ne,in,between		x	
binomial	<,<=,=,>=,>,ne,in,between		x	
poisson	<,<=,=,>=,>,ne,in,between		x	
negativeBinom	<,<=,=,>=,>,ne,in,between		x	
uniform	<,<=,=,>=,>,ne,in,between		x	x
multinomial	<,<=,=,>=,>,ne,in,between		x	x
rectangular	<,>,between	<,=,>,ne	x	x
triangular	<,>,between	<,=,>,ne	x	
normal	<,>,between	<,=,>,ne	x	x
chiSquare	<,>,between	<,=,>,ne	x	
tDist	<,>,between	<,=,>,ne	x	
fDist	<,>,between	<,=,>,ne	x	
gamma	<,>,between	<,=,>,ne	x	
beta	<,>,between	<,=,>,ne	x	
weibull	<,>,between	<,=,>,ne	x	
lognormal	<,>,between	<,=,>,ne	x	

Table showing discrete and continuous distributions and corresponding operators. 16 distribution functions, 8 relational functions and 4 operators produce 136 statistical functions.

	confInt	hypothesis
mean	1-Sample,2-sample	<,ne,> ;1Sample,2Sample
proportion	1-Sample,2-sample	<,ne,> ;1Sample,2Sample
variance	1-Sample,2-sample	<,ne,> ;1Sample,2Sample
standardDeviation	1-Sample,2-sample	<,ne,> ;1Sample,2Sample

Table showing 4 summary functions, 3 relational functions and 2 operators produce 32 inferential statistical functions when we include both one and two-sample tests.