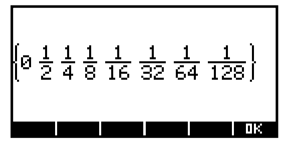
StatPack 39/40

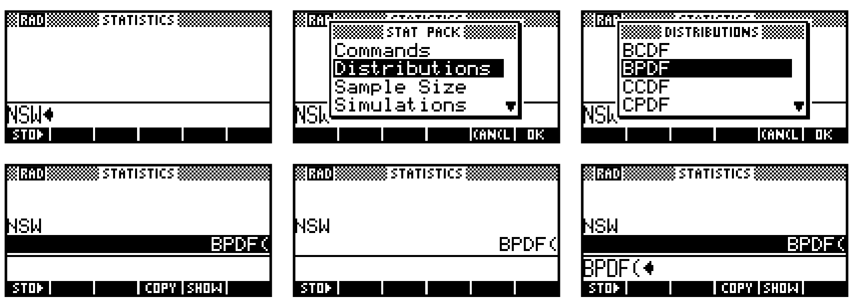
This package is designed to add many statistics functions that should have been built into the 39G. It includes everything in the TI-83, as well as a slew of other valuable and powerful commands. Most are faster than the 83's commands as well. Once installed, they work like any other built in function (even in programs).

First I'd like to give a word of warning to anyone who uses this. If you go about trying to get my commands to mess something up, you'll probably be able to find something that will. If something does mess up, it most likely won't wipe your memory. I've tried to prevent you from messing up your calculator, but some people do some pretty stupid things. =)



Changing the mode display works with these commands. So if you change the calculator to fraction mode or fixed mode for example, it will display in that way. This can produce some nice results. Look at this result from the GPDF command. It really the sequence quite well. Built in variables, such as A to Z or L0 to L9, function correctly in all the commands.

To access the commands, I've made a command called NSW. This allows quick access to all the commands from anywhere, even while not running this aplet. This is quick to type in as N,S, and W are all right in a line above the + key. On the home screen, hold down ALPHA and hit /,*,- and push ENTER. This pops up a menu with the different sub-menus. It looks like the screenshot shown. After you choose a sub menu (using either OK or just ENTER), the command menu will pop up. After selecting a command, push ENTER or OK and the command will appear on level one of the history. Push the up arrow and COPY, and the command is in the entry line. While slightly awkward at first, this is very quick after only a few times using it. These steps are shown in the pictures below.



Some of the commands will work with incorrect syntax. It is easy to tell when something has gone wrong. This is because you'll get back an answer that just can't be right (like if you run BPDF(.6,10,-3) you'll get back a string of all ones). That is wrong. Another example is if you get back a ρ value greater than 1, there is probably an error somewhere. ;-)

The rest of this document uses the syntax shown below. Look at the NOTE attached to the aplet to view built-in help on the calculator. It has syntax listing for all commands.

SYNTAX:

x :	test value
μ :	mean
σ :	standard deviation (σ^2 is also called variance)
ρ :	probability value
df :	degrees of freedom
κ :	parameters
n :	number of trials

l :	lowerbound
u :	upperbound
ndf :	numerator degrees of freedom
ddf :	denominator degrees of freedom
list :	a list, can also be L0 through L9, or C0 through C9
E :	the maximum allowable error. This is ONE SIDE of the overall error width.

DISTRIBUTION COMMANDS:

BPDF(n,ρ,x)

computes a probability at x for the discrete binomial distribution with the specified number of trials (n) and probability of success (ρ) on each trial. The parameter x can be a real number or a list of real numbers. If you specify x as a negative then a list variable is returned giving all the probabilities from x=0 to x=n. I usually use –3 because it is closer to the ENTER key so it keys in faster.

eg. BPDF(4,0.5,2) returns 0.375
BPDF(4,0.5,{0,2,3}) returns {0.0625, 0.375, 0.25}
BPDF(4,0.5,-1) returns {0.0625, 0.25, 0.375, 0.25, 0.0625}

BCDF(n,ρ,x)

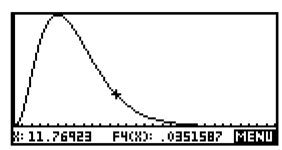
computes a cumulative probability at x for the discrete binomial distribution with the specified number of trials (n) and probability of success (ρ) on each trial. The parameter *x* can be a real number or a list of real numbers. If you specify x as a negative then a list variable is returned giving all the probabilities from x=0 to x=n. I usually use –3 because it is closer to the ENTER key so it keys in faster.

eg. BCDF(4,0.5,2) returns 0.6875
BCDF(4,0.5,{0,2,3}) returns {0.0625, 0.6875, 0.9375}
BCDF(4,0.5,-1) returns {0.0625, 0.3125, 0.6875, 0.9375, 1}

CPDF(x,df)

computes the probability density function (pdf) for the χ^2 distribution at a specified x value for specified df (degrees of freedom) which must be >0. To plot the χ^2 distribution, put CPDF(x,df) into the functions aplet. (I would like to note that this graphs faster than the TI-83 =)

In this example,Y1 is CPDF(x,7). The screen is Xmin=0, Xmax=30 and then autoscaled.



CCDF(l,u,df)

computes the χ^2 distribution probability between lower and upper boundaries for the specified df (degrees of freedom), which must be > 0.

eg. To find the ρ value of a χ^2 dstribution between 1 and 5, with 5 degrees of freedom, one can either use CCDF once or UTPC twice.

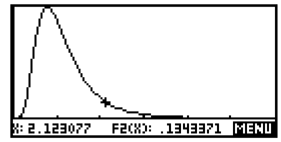
CCDF(1,5,9) returns 0.165129

This is equivalent to UTPC(9,1)-UTPC(9,5).

FPDF(x,ndf,ddf)

computes the probability density function (pdf) for the F distribution at a specified x value. Both numerator and denominator df (degrees of freedom) must be >0. To plot the F distribution, put FPDF(X,df) into the Function aplet. (I would like to note that this graphs faster than the TI-83 =)

In this example, Y1 is FPDF(x,20,12). The window is Xmin=0, Xmax=6, and then autoscaled.



FCDF(l,u,ndf,ddf)

computes the F probability distribution between lower and upper bounds for the specified ndf and ddf, which must be > 0.

eg. To find the ρ value between 1 and 3 on a distribution with a ndf of 8 and ddf of 6, you can use FCDF once or UNTF twice.

FCDF(1,3,8,6) returns 0.165129

This is equivalent to UTPF(8,6,1)-UTPC(8,6,5)

GPDF(ρ,x)

computes a probability at x, the number of the trial on which the first success occurs, for the discrete geometric distribution with the specified probability of success. X can be a real number or a list of real numbers in the same fashion as BPDF, but cannot be negative since the set of all values is infinite rather than of finite length as in BPDF.

eg. GPDF(0.5,4) returns 0.0625
GPDF(0.5,{0,2,3}) returns {0, .25, .125}

GCDF(ρ,x)

computes a cumulative probability at x, the number of the trial on which the first success occurs, for the discrete geometric distribution with the specified probability of success. X can be a real number or a list of real numbers in the same fashion as BCDF, but cannot be negative since the set of all values is infinite rather than of finite length as in BCDF.

eg. GPDF(0.5,4) returns 0.9375
GPDF(0.5,{0,2,3}) returns {0, .75, .875}

INVN(ρ,σ^2,μ)

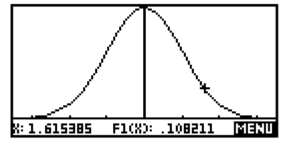
computes the inverse normal for given probability (ρ), variance (σ^2), and mean (μ). It is important to remember that this goes on LOWER TAIL probability. So a ρ value less than .5 means it is to the left of the mean (μ).

eg. INVN(0.76,1^2,0) returns .7063
INVN(.23,5^2,10) returns 6.3057

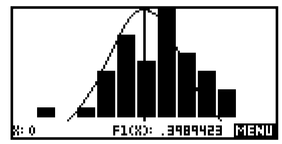
NPDF(x,σ^2,μ)

computes the probability density function (pdf) for the normal distribution at a specified x value. To plot the normal distribution, put NPDF(x,σ^2,μ) into the Function aplet. (I would like to note that this graphs faster than the TI-83 =)

In this example, Y1 is NPDF(x,1,0). The window is Xmin=-3.5, Xmax=3.5, and then autoscaled.



It is easy to overlay a normal curve onto another graph as well. For example, to overlay a normal graph on top of a histogram, simply graph the histogram in the Statistics aplet first, adjusting the axes in PLOT SETUP so that they are not strange decimals. Then change to the Function aplet and specify F1(X) to be NPDF(x,σ^2,μ) where μ and σ are the mean and standard deviation of the stats data. Now set the axes to be exactly the same as those of the histogram and then, rather than pressing PLOT, use the VIEWS – Overlay Plot command to draw the normal curve on top of the existing histogram. This also works for all other graph types and graphing commands.



NCDF(l,u,σ^2,μ)

computes the normal distribution probability between lower and upper bounds for the specified variance (which must be $\sigma^2 > 0$), and mean (μ).

eg. To find the ρ value for a normal distribution with a mean of 50 and a standard deviation of 15 between 40 and 55, one can either use NCDF once, or UTPN twice.

NCDF(40,55,15^2,50) gives .378066

This is equivalent to UTPN(50,15^2,40)-UTPN(50,15^2,55)

PPDF(μ,x)

computes a probability at x for the discrete Poisson distribution with the specified mean (μ), which must be > 0. X can be an integer or a list of integers (if you use decimals, it rounds to the nearest integer internally).

eg. PPDF(10,6) returns 6.3055E-2
PPDF(10,{6,10,13)}) returns {6.3055E-2, .12511, 7.2907E-2 }

PCDF(μ,x)

computes a cumulative probability at x for the discrete Poisson distribution with the specified mean (μ), which must be > 0. X can be an integer number or a list of integers (if you use decimals, it rounds to the nearest integer internally).

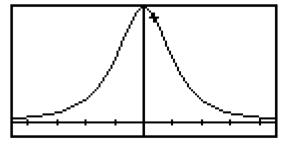
Eg. PCDF(6,3) returns .1512

To find the probability of 2<=x<=5 for a Poisson with a mean of 6, you would enter PCDF(6,5)-PCDF(6,1) which results in .428328. The change from 2 to 1 is due to the discrete nature of the pdf.

TPDF(x,df)

computes the probability density function (pdf) for the student-T distribution at a specified x value for specified df (degrees of freedom) which must be >0. To plot the student-T distribution, put TPDF(x,df) into the functions aplet. (I would like to note that this graphs faster than the TI-83 =)

In this example, FX(1) is TPDF(x,2). The window is Xmin=-4.5, Xmax=4.5, and then autoscaled.



TCDF(l,u,df)

computes the student-T distribution probability between lower and upped boundaries for the specified df (degrees of freedom), which must be > 0.

eg. To find the ρ value for a student-T distribution with 2 degrees of freedom, between –1.5 and 1, you can either use TCDF once, or UTPT twice.

TCDF(40,55,15^2,50) gives .65247

This is equivalent to UTPT(2,-1.5)-UTPT(2,1)

SIMULATIONS:

It is sometimes useful for teaching purposes or for the writing of assessment items to be able to produce a set of simulated observations on a specified PDF. For example, to be able to produce 100 simulated observations on a Normal distribution with a mean of 50 and a standard deviation of 15. These commands will allow you to do this.

This also makes many types of test question that involve the distributions very easy!

Each one is used in the HOME view (or in a program) and returns a list of values {…, …n} which can then be stored into a list variable L0 through L9 or into one of the columns of the Statistics aplet C0 through C9.

Eg. BINSIM(0.6,20,100)▶C1 returns a list of 100 simulated observations on a binomial pdf(20,0.6) and stores the results into C1 of the Statistics aplet ready for use. Note: ▶ is obtained using the STO▶ screen button.

BERNSIM(ρ,n)	Returns a list containing n values of 1 or 0 which are a Bernoulli simulation using a probability of ρ. This is a simulation of either success, or failure using the given ρ.
BINSIM(ρ,N,n)	Returns a list containing N values which are a Binomial simulation for n trials and ρ. This command can be a little slow because of the massive number of numbers that must be generated for large values of N or n.
EXPSIM(κ,n)	Returns a list containing n values which are a simulation on an exponential pdf for specified parameter κ (where κ=1/μ).
NORMSIM(μ,σ^2,n)	Returns a list containing n values which are a simulation on a normal pdf with specified mean and variance. Not an exact simulation due to use of a SIN approximation, it is still fairly accurate however.

PASCSIM(ρ,κ,n) Returns a list containing n values which are a simulation on a Pascal pdf with specified probability and required number of successes. This simulates the number of trials it takes for a given number of successes (κ) to be reached. This can be a little slow due to the massive number of numbers that must be calculated for large κ or n.

RINT(l,u,n) Returns a list containing n random integer values between lower and upper bounds inclusive (lower and upper can be reversed if so desired) which are simulated observations on a discrete uniform pdf.

Eg. RINT(1,6,100) will simulate the tossing of a normal die 100 times, while RINT(1,6,100)+RINT(1,6,100) will simulate throwing two dice and adding the face values together.

RNUM(l,u,n) Returns a list containing n random real values between lower and upper bounds inclusive (which can be reversed if so desired) which are simulated observations on a continuous uniform pdf.

COMMANDS:

These are additional commands that do various things that assist in calculating and analyzing statistics.

AANDB(ρ1,ρ2,ρ1&ρ2) calculates A and B (A∪B) for given probabilities.

AIFB(ρ1,ρ2,ρ1&ρ2) calculates A if B (A|B) for given probabilities.

AORB(ρ1,ρ2,ρ1&ρ2) calculates A or B (A∩B) for given probabilities.

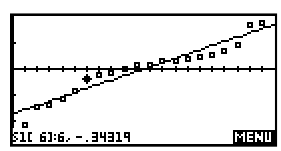
CLIST(list) does a cumulative summation of a list.

eg. CLIST({0,1,2,3,4}) returns {0,1,3,6,10}

LEQ(list,real) tests a list for equal x value. The built in testing done on lists will not work with the equal test as it tests to see if the lists are identical. This tests the interior of the list versus the given value.

eg. LEQ({1,-1,1,4,5,7,1},1) returns {1,0,1,0,0,0,1}

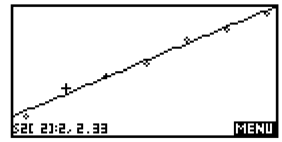
NPP(list) generates a list of normalized data. Used to do normal probability plots.

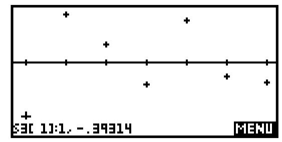


eg. Using the NORMSIM(0,1,20) command, we save it into C2. Then using MAKELIST(X,X,1,20,1), we store that to C1. Now go into the statistics aplet and sort C2. Graph a 2 variable scatter plot with C1 and C2. The result should look something like this. In the picture the fit line is turned on.

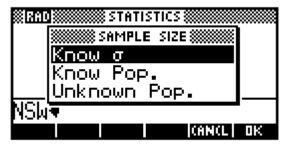
RESID(xlist,ylist) — Calculates the residuals from the last calculated regression. This means you MUST have first calculated a regression line using the statistics aplet. Xlist is the list on the X axis used in the regression, while Y list is the list on the Y axis.

eg. Using {1,2,3,4,5,6,7} as C1 and {.5,2.33,3.19,3.987,5.53,6.2,7.24} as C2, and linear regression set up as S1fit, the scatter plot will look like the picture seen on the right. Now we return to the home screen, and use RESID(C1,C2) and store the result into C3. Now go to the STAT SYMB setup screen and select C1 and C3 to be graphed. Autoscale it, and the result will look like the one on the left. There appears to be no trend in the residuals, so it is fairly safe to say the linear fit is a good approximation.





Sample Size:



These commands are used to calculate sample size. They are hard to remember, so they are set up in a menu that describes them by what is known.

NMEAN(σ,E,confidence) — this is the command that appears when “know σ” is selected. To use it, input the σ (NOT THE VARIANCE as with other commands), the maximum error, and the confidence level.

eg. It has been determined that a population has a standard deviation of .5. The maximum error allowable for our survey is .05% at a 95% confidence interval. Enter .025 in as the error because the TOTAL size is to be .05 (it is .025 to the left and right of the μ.). Confidence is entered as .95 for 95%. This results in NMEAN(.5,.025,.95) which returns 1536.6. This means a sample of 1567 should be sufficient for a 95% confidence interval.

NPOP(pop. ratio,E,confidence) — this command appears when “Know Pop.” is selected. Enter the ratio of the sample to the overall population, the error/2, and the confidence interval.

NNOPOP(E, confidence) — This command appears when “Unknown Pop.” is selected. Enter the error (total allowable error/2), and the confidence. A sample size of .05 maximum error, at a 95% confidence interval is calculated with NNOPOP(.025,.95) which results in 1536.6, so a sample of at least 1537 is required.

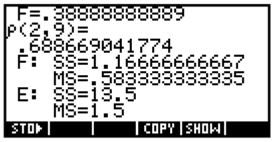
Tests:

These commands are used to do statistical tests upon data.

A list of lists looks like { {…},{…},{…}}. They are most easily created by making a matrix, and then storing it into a list on the home screen. An example would be M3▶L2.

ANOVA(list of lists)

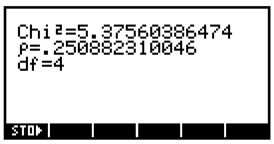
Calculates a one way analysis of variance for comparing the means of sample data. The null hypothesis $H_0 : \mu1=\mu2\ldots=\mu n$ is tested against the alternate hypothesis H_a that not all the means are equal. Works only with equal length samples.



eg. Using {{4,6,6,5},{6,5,5,8}, {4,7,6,6}} saved into L3, running ANOVA(L3) returns the screen shown on the left. Ndf and ddf is shown as ρ(ndf,ddf). F: is the factor data. E: is the error data. MS is mean of the squares, and SS is sum of the squares.

CHI2(list of lists)

Calculates a χ^2 test on the given data. The list of lists entered is the observed values, and the expected values are returned after the test. To easily view the expect counts, store the returned list into a matrix.



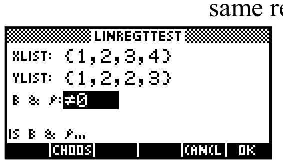
eg. Using {{5,23,21},{8,18,23},{11,13,25}} saved into L5, running CHI2(L5) returns the screen shown on the left.

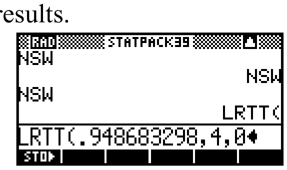
GOF(observed list,expected list)

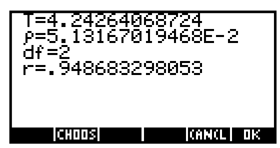
Input a list of observed scores, such as {1,2,3}, and a list of expected scores, such as {1.3,2.5,2.9} and it returns a χ^2 score of .21833333.

LRTT(r,n,0/>0/<0)

First asks whether the input is data or stats. If data, a graphical interface is activated like the one shown. If stats is selected, the correlation, number of data points, and either 0 for the not equal test, >0 for the >0 test, and <0 for the <0 test. Either option returns the same results.

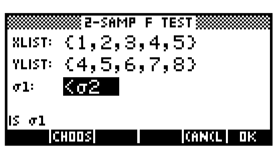


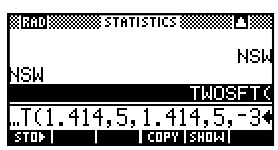


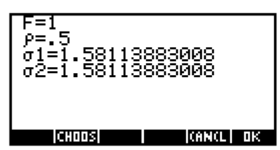


2SFT(σ1,n1,σ2,n2,0/<0/>0)

First asks whether the input is data or stats. If data, a graphical interface is activated like the one shown. If stats is selected, both σ's and number of points are entered, and either 0 for the not equal test, >0 for the >σ2 test, and <0 for the <σ2 test. Either option returns the same results.







How do I contact you?

Well it will be pretty hard until December of 2003. I will be in Honduras until then serving a religious mission for my church (www.lds.org), and will not be doing anything on the computer until I get back. So if you send an e-mail to me at timwessman@yahoo.com, I won't be able to answer it until December 2003, and probably won't even get it because my mailbox will be full.

If you'd like to write me a snail mail letter however (talking about things other then calculators), you can send an email with the subject of "Tim's mail address in honduras" or something like that. My parents will check occasionally for these types of messages. Then they'll send it to you.