This question was originally posed on [Statalist](http://www.statalist.org/) and answered by several users and StataCorp’s Bill Sribney. Bill’s answer, with slight editing, appears below.

## Does Stata provide a test for trend?

|  |  |  |
| --- | --- | --- |
| Title |  | A comparison of different tests for trend |
| Author | William Sribney, StataCorp |

Let me make a bunch of comments comparing SAS PROC FREQ, Pearson’s correlation, Patrick Royston’s **ptrend** command, linear regression, logit/probit regression, Stata’s [vwls](/manuals/rvwls.pdf) command, and Stata’s [nptrend](/manuals/rnptrend.pdf) command.

### Tests for trend in 2 x r tables

Let me use Les Kalish’s example:

Outcome

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  | | | | | |
|  | group |  | Good |  | Better |  | Best |
|  | y\_i |  | a\_1=1 |  | a\_2=2 |  | a\_3=3 |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  | y\_1=0 |  | 19 |  | 31 |  | 67 |
|  |  |  | n\_11 |  | n\_12 |  | n\_13 |
|  |  |  |  |  |  |  |  |
|  | y\_2=1 |  | 1 |  | 5 |  | 21 |
|  |  |  | n\_21 |  | n\_22 |  | n\_23 |
|  |  |  |  |  |  |  |  |
|  |  | | | | | | |

20 36 88

n\_+1 n\_+2 n\_+3

### PROC FREQ

When I used SAS in grad school to analyze these data, we used

**PR0C FREQ DATA=...**

**TABLES GROUP\*OUTCOME / CHISQ CMH SCORES=TABLE**

The test statistic was shown on the output as

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | DF |  | Value |  | Prob |
|  |  |  |  |  |  |
| Mantel-Haenszel Chi-Square | 1 |  | 4.515 |  | 0.034 |

The test statistic is not a Mantel–Haenszel—at least not according to what I learned a Mantel–Haenszel statistic is (from Gary Koch at UNC—note that any errors here, I should add, are those of this student, not of this great researcher/teacher).

Dr. Koch called this chi-squared statistic Qs, where s stands for score.

### Chi-squared statistic for trend Qs

Let me express Qs in terms of a simpler statistic, T:

T = (sum over group i)(sum over outcome j) nij \* aj \* yi

The aj are scores; here 1, 2, 3, but there can be other choices for the scores (I’ll get to this later).

Under the null hypothesis there is no association between group and outcome, so we can consider the permutation (i.e., randomization) distribution of T. That is, we fix the margins of the table, just as we do for Fisher’s exact test, and then consider all the possible permutations that give these same marginal counts.

Under this null hypothesis permutation distribution, it is easy to see that the mean of T is

E(T) = N \* a\_bar \* y\_bar

where a\_bar is the weighted average of aj (using the marginal counts n+j):

a\_bar = (sum over j) n+j \* aj / N

Similarly, y\_bar is a weighted average of yi.

The variance of T, under the permutation distribution, is (exactly)

V(T) = (N - 1) \* Sa2 \* Sy2

where Sa2 is the standard deviation squared for aj:

Sa2 = (1/(N-1)) \* (sum over j) n+j \* (aj - a\_bar)2

We can compute a chi-squared statistic:

Qs = (T - E(T))2 / V(T)

If you look at the formula for Qs, you see something interesting. It is simply

Qs = (N - 1) \* ray2

where ray is Pearson’s correlation coefficient for a and y.

### Just Pearson’s correlation

This “test of trend” is nothing more than Pearson’s correlation coefficient.

Let’s try this.

**. list**

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | | |
|  |  | y a weight |  |
|  |  |  |  |
| 1. |  | 0 1 19 |  |
| 2. |  | 0 2 31 |  |
| 3. |  | 0 3 67 |  |
| 4. |  | 1 1 1 |  |
| 5. |  | 1 2 5 |  |
|  |  |  |  |
| 6. |  | 1 3 21 |  |
|  |  | | |

**. corr y a [fw=weight]**

(obs=144)

|  |  |  |
| --- | --- | --- |
|  |  | y a |
|  |  |  |
| y |  | 1.0000 |
| a |  | 0.1777 1.0000 |

**. return list**

scalars:

r(N) = 144

r(rho) = .1776868721791401

matrices:

r(C) : 2 x 2

**. display (r(N)-1)\*r(rho)^2**

4.5148853

PROC FREQ gave chi-squared = 4.515.

### Royston’s ptrend and the Cochran–Armitage test

Let’s now use Patrick Royston’s **ptrend** command. Patrick posted his **ptrend** command on Statalist. The data must look like the following for this command:

**. list**

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | | |
|  |  | a n1 n2 |  |
|  |  |  |  |
| 1. |  | 1 19 1 |  |
| 2. |  | 2 31 5 |  |
| 3. |  | 3 67 21 |  |
|  |  | | |

**. ptrend n1 n2 a**

n1 n2 \_prop a

1. 19 1 0.950 1

2. 31 5 0.861 2

3. 67 21 0.761 3

Trend analysis for proportions

Regression of p = n1/(n1+n2) on a:

Slope = -.09553, std. error = .0448, Z = 2.132

Overall chi2(2) = 4.551, pr>chi2 = 0.1027

Chi2(1) for trend = 4.546, pr>chi2 = 0.0330

Chi2(1) for departure = 0.004, pr>chi2 = 0.9467

The “Chi2(1) for trend” is slightly different. It’s 4.546 rather than 4.515.

Well, **ptrend** is just using N rather than N − 1 in the formula:

Qtrend = Chi2(1) for trend = N \* ray2

Let’s go back to data arranged for the **corr** computation and show this.

**. quietly corr y a [fw=weight]**

**. display r(N)\*r(rho)^2**

4.5464579

Qtrend is just Pearson’s correlation again. A regression is performed here to compute the slope, and the test of slope = 0 is given by the Qtrend statistic. This is just the relationship between Pearson’s correlation and regression.

Qdeparture (="Chi2(1) for departure" as Royston’s output nicely labels it) is the statistic for the Cochran–Armitage test. But Qtrend and Qdeparture are usually performed at the same time, so lumping them together under the name “Cochran–Armitage” is sometimes loosely done.

The null hypothesis for the Cochran–Armitage test is that the trend is linear, and the test is for “departures” from linearity; i.e., it’s simply a goodness-of-fit test for the linear model.

Qs (or equivalently Qtrend) tests the null hypothesis of no association. Since it’s just a Pearson’s correlation, we know that it’s powerful against alternative hypotheses of monotonic trend, but it’s not at all powerful against curvilinear (or other) associations with a 0 linear component.

### Model it

Rich Goldstein recommended logistic regression. Regression is certainly a better context to understand what you are doing—rather than all these chi-squared tests that are simply Pearson’s correlations or goodness-of-fit tests under another name. Since Pearson’s correlation is equivalent to a regression of y on “a”, why not just do the regression

**. regress y a [fw=weight]**

|  |  |  |  |
| --- | --- | --- | --- |
| Source |  | SS df MS | Number of obs = 144 |
|  |  |  | F( 1, 142) = 4.63 |
| Model |  | .692624451 1 .692624451 | Prob > F = 0.0331 |
| Residual |  | 21.2448755 142 .1496118 | R-squared = 0.0316 |
|  |  |  | Adj R-squared = 0.0248 |
| Total |  | 21.9375 143 .153409091 | Root MSE = .3868 |

|  |  |  |
| --- | --- | --- |
|  | | |
| y |  | Coef. Std. Err. t P>|t| [95% Conf. Interval] |
|  |  |  |
| a |  | .0955344 .0444011 2.15 0.033 .0077618 .183307 |
| \_cons |  | -.0486823 .1144041 -0.43 0.671 -.2748375 .177473 |
|  | | |

But recall that y is a 0/1 variable. Heck, wouldn’t you be laughed at by your colleagues if you presented this result? They’d say, “Don’t ya know anything, you dummy, you should be using logit/probit for a 0/1 dependent variable!” But call these same results a “chi-squared test for linear trend” and, oh wow, instant respectability. Your colleagues walk away thinking how smart you are and jealous about all those special statistical tests you know.

I guess it sounds as if I’m agreeing fully with Rich Goldstein’s recommendation for logit (or probit, which Rich didn’t mention). I’ll say some redeeming things about Pearson’s correlation (hey, let’s call it what it really is) below, but for now, let me give you another modeling alternative.

### Try the vwls command

One can do a little better using the command [vwls](/manuals/rvwls.pdf) (variance-weighted least squares) in Stata rather than [regress](/manuals/rregress.pdf)

**. vwls y a [fw=weight]**

Variance-weighted least-squares regression Number of obs = 144

Goodness-of-fit chi2(1) = 0.01 Model chi2(1) = 7.79

Prob > chi2 = 0.9359 Prob > chi2 = 0.0053

|  |  |  |
| --- | --- | --- |
|  | | |
| y |  | Coef. Std. Err. z P>|z| [95% Conf. Interval] |
|  |  |  |
| a |  | .0944406 .0338345 2.79 0.005 .0281261 .160755 |
| \_cons |  | -.0459275 .0758028 -0.61 0.545 -.1944984 .1026433 |
|  | | |

The “test for linear trend” is again the test of the coefficient of a = 0. It also gives you a goodness-of-fit test. The linear model is the same as **regress**, but the weighting is a little different. The weights are 1/Var(y|a). Now

Var(y|aj) = pj \* (1 - pj) / n+j

where pj = n2j / n+j (recall n2j = #(y=1) for outcome j).

Essentially, this means you are downweighting (relative to **regress**) points aj that have pj close to 0 or 1. **regress** with weights nij puts too much weight on these points.

If you are determined to fit a linear model for y vs. aj, I believe **vwls** is a better way to do it.

**vwls** can do more. y can be a continuous variable, too. The regressors x1, x2, x3, ... can be any variables as long as for each unique value of the combination of x1, x2, x3, ... there are enough points to reasonably estimate the variance of y. For example, Gary Koch had us using this to model y = # of colds versus x = 1 (drug) or 0 (placebo).

You can do the equivalent of **vwls** in SAS using PROC CATMOD with a RESPONSE MEAN option.

### Using different scores aj: aj = average ranks

As I said above, the scores aj are simply a regressor variable, and we can use anything we want for them.

Let’s revisit Les’s little dataset:

Outcome

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | | | | | |
| group |  | Good |  | Better |  | Best |
| y\_i |  | a\_1=1 |  | a\_2=2 |  | a\_3=3 |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
| y\_1=0 |  | 19 |  | 31 |  | 67 |
|  |  | n\_11 |  | n\_12 |  | n\_13 |
|  |  |  |  |  |  |  |
| y\_2=1 |  | 1 |  | 5 |  | 21 |
|  |  | n\_21 |  | n\_22 |  | n\_23 |
|  |  |  |  |  |  |  |
|  | | | | | | |

20 36 88

n\_+1 n\_+2 n\_+3

ranks 1-20 21-56 57-144

sum of

ranks 210 1386 8844

average

rank 10.5 38.5 100.5

Let’s now use the average ranks instead of aj.

Call these scores rj. Let’s compute Pearson’s coefficient for y and rj.

**. list**

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | | |
|  |  | y a weight r |  |
|  |  |  |  |
| 1. |  | 0 1 19 10.5 |  |
| 2. |  | 0 2 31 38.5 |  |
| 3. |  | 0 3 67 100.5 |  |
| 4. |  | 1 1 1 10.5 |  |
| 5. |  | 1 2 5 38.5 |  |
|  |  |  |  |
| 6. |  | 1 3 21 100.5 |  |
|  |  | | |

**. corr y r [fw=w]**

(obs=144)

|  |  |  |
| --- | --- | --- |
|  |  | y r |
|  |  |  |
| y |  | 1.0000 |
| r |  | 0.1755 1.0000 |

**. display (r(N)-1)\*r(rho)^2**

4.4063138

The above is our chi-squared test statistic for Pearson’s correlation coefficient, which Gary Koch called Qs.

This is exactly what [nptrend](/manuals/rnptrend.pdf) is doing. **nptrend** is again Pearson’s correlation coefficient by another name.

**nptrend**, unfortunately, does not allow weights, so we must expand the data:

**. expand weight**

(138 observations created)

**. nptrend a, by(y)**

y score obs sum of ranks

0 0 117 8126.5

1 1 27 2313.5

z = 2.10

Prob > |z| = 0.036

**. ret list**

scalars:

r(p) = .0358061342131948

r(z) = 2.099122151413003

r(T) = 2313.5

r(N) = 144

**. display r(z)^2**

4.4063138

Voila, the same answer!

**nptrend** takes the variable given after the command—here “a”—and computes the average ranks for it just as we did. It then correlates the average ranks with the values in y.

In SAS, if you specify SCORES=MODRIDIT, the scores used are the average ranks. PROC FREQ with SCORES=MODRIDIT should give exactly what **nptrend** produces.

### More on nptrend: r x c tables

**nptrend** allows y to be anything. If y is 0, 1, 2, then we are doing Pearson’s correlation in 3 x 3 tables.

**nptrend a, by(y)** allows you to substitute different values (i.e., scores) for y: when you specify the **score(***scorevar***)** option on **nptrend**, it uses the values of *scorevar* in place of the values of y in computing the correlation coefficient.

**nptrend** always uses averaged ranks for the scores for the “a” variable.

This is a little confusing. Remember, we are simply computing **corr y x**. y can be anything, and x can be anything in theory. But **nptrend** is restrictive; it allows any values for y but only allows x to be averaged ranks.

There are other ways to do r x c tables. We do not have to assume an ordering on r. If we don’t, guess what we get. ANCOVA! The scores for the outcome “a” are the “continuous” variable. We can model different slopes per block r. Test slopes = 0 with a chi-squared stat, give it a fancy name, and no one will know we are just doing ANCOVA!

### Stratified 2 x c tables

Generalizing the above analysis to stratified 2 x c tables gives what I learned as the “Extended Mantel–Haenszel” test. Rothman calls it the “Mantel extension test” and then says it can be done with or without strata. Yes, it can be done with or without strata, but in the terminology I know, it’s only called the Extended Mantel–Haenszel test only when there are strata. Thanks, Ken, for adding yet another name for unstratified 2 x c tables!

Say, the test for stratified tables is NOT just Pearson’s correlation coefficient in disguise. First, define, as we did before, the statistic

Th = (sum over group i)(sum over outcome j) nhij \* ahj \* yhi

for each stratum h. Then form the statistic T by summing over strata. Sometimes the sum is weighted by stratum totals, sometimes strata are equally weighted. (I believe that SAS does the former; Rothman gives the formula for the latter.) Since the strata are independent, the mean and variance of T are easy to compute from Th.

### Distribution of Pearson’s correlation coefficient

I said that Qs = (N − 1) \* ray2 (where ray is Pearson’s correlation coefficient) had a chi-squared distribution. And one can use this to get a *p*-value for Pearson’s correlation coefficient. But this isn’t what [**pwcorr**](/manuals/rcorrelate.pdf) does. **pwcorr** is using a t distribution to get a *p*-value (which assumes normality of "a" and y). These are asymptotically equivalent procedures (under the null hypothesis).

Qs is a second-moment approximation for the permutation distribution for ray. The permutation distribution for ray makes no assumptions about "a" and y.

If you want to get fussy about getting *p*-values, then one should compute the permutation distribution or compute higher-moment terms for the permutation distribution for ray.

For small N, Pearson’s correlation coefficient has an advantage over logistic regression. One can compute the permutation distribution of Pearson’s correlation coefficient exactly. For the exact distribution, true type I error is equal to nominal type I error. Such won’t be the case for logistic. Also, I bet it’s more powerful than logistic for small N. One could do some simulations and look at this—it’s a good master’s paper project, perhaps. This is the case where Pearson’s correlation coefficient is a better choice than logistic regression or other regression modeling.

### Tests for trend in Stata

Clearly, we need a command to do r x c tables, stratified and unstratified, with various choices of scores.

We plan to implement something in the future. But, in the meantime, for moderate to large N, there is logit/probit regression (and **vwls**). If you do want Qs for linear scores, it’s easy to use [**corr**](/manuals/rcorrelate.pdf) as I’ve done here. One can also use [**stmh**](/manuals/ststrate.pdf), [**stmc**](/manuals/ststrate.pdf), and [**tabodds**](/manuals/repitab.pdf).

For average-rank scores, it’s not too difficult to compute the average ranks yourself and then use **corr**.

For stratified tables, there’s nothing easy available I know of.