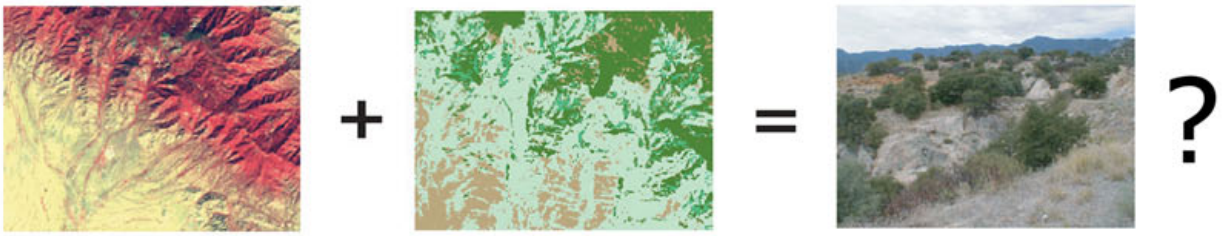
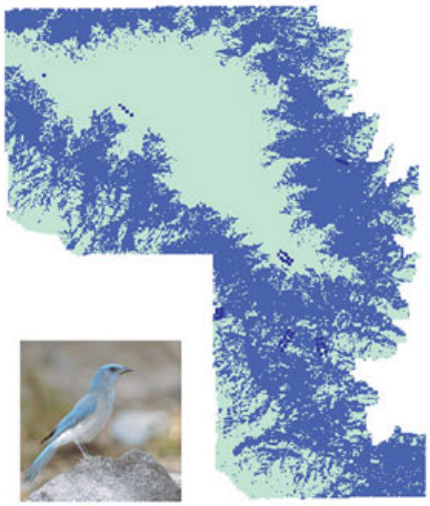
Cohen's Kappa and Classification Table Metrics 2.1:

An ArcView 3x Extension for Accuracy Assessment of Spatially-Explicit Models





Confusion Matrix

		Actual Values	
		+	-
Predicted Values	+	*a*	*b*
	-	*c*	*d*

j = Columns

Reference Data: Actual or Field-checked

i = Rows (Classification Data: Predicted from Model)	j_1	j_2	j_k	n_{i+} (Row Totals)
i_1	n_{11}	n_{12}	n_{1k}	n_{1+}
i_2	n_{21}	n_{22}	n_{2k}	n_{2+}
i_k	n_{k1}	n_{k2}	n_{kk}	n_{k+}
n_{+j} (Column Totals)	n_{+1}	n_{+2}	n_{+k}	$n_{++} = n$

(Adapted from Congalton and Green 1999, p. 47)

http://www.jennessent.com/arcview/kappa_stats.htm

Jeff Jenness

J. Judson Wynne

Abstract

In the field of spatially explicit modeling, well-developed accuracy assessment methodologies are often poorly applied. Deriving model accuracy metrics have been possible for decades, but these calculations were made by hand or with the use of a spreadsheet application. Accuracy assessments may be useful for: (1) ascertaining the quality of a model; (2) improving model quality by identifying and correcting sources of error; (3) facilitating a comparison of various algorithms, techniques, model developers and interpreters; and, (4) determining the utility of the data product in a decision-making context. When decisions are made with models of unknown or poorly-assessed accuracy, resource managers run the risk of making wrong decisions or drawing erroneous conclusions. Untested predictive surface maps should be viewed as untested hypotheses and, by extension, poorly tested predictive models are poorly tested hypotheses. Often, if any accuracy measure is provided at all, only the overall model accuracy is reported. However, numerous accuracy metrics are available which can describe model accuracy and performance. Because issues concerning data quality and model accuracy in landscape analyses have received little attention in the management literature, we found it useful to develop a systematic and robust procedure for assessing the accuracy of spatially explicit models. We created an ArcView 3.x extension that provides end users with a packaged approach for accuracy assessment, using Cohen's Kappa statistic as well as several other metrics including overall accuracy, overall misclassification rate, model specificity and sensitivity, omission and commission errors, and positive and negative predictive power. Collectively, these metrics may be used for gauging model performance. When multiple models are available, these metrics offer end users the ability to quantitatively compare and identify the "best" model within a multi-criteria model selection process.

Researchers have already cast much darkness on the subject, and if they continue their investigations, we shall soon know nothing at all about it.
Mark Twain

May we, through our research and toils, aspire to prove Sam's mark less than true and one day walk away knowing a little something about spatial model accuracy.

NAME: Cohen's Kappa and Classification Table Metrics 2.1a:
An ArcView 3x Extension for Accuracy Assessment of Spatially-Explicit Models

Aka: kappa_stats.avx

Last modified: December 10, 2007

TOPICS: Kappa, classification, accuracy, sensitivity, specificity, omission, commission, user accuracy, producer accuracy, P-value, sample size, model, statistics, distributions, t, F, logistic, normal, skewness, kurtosis, binomial, probability, critical, Poisson, chi-square

AUTHORS

Jeff Jenness
Wildlife Biologist, GIS Analyst
Jenness Enterprises
3020 N. Schevene Blvd.
Flagstaff, AZ 86004 USA
Tel (928) 607-4638
jeffj@jennessent.com

J. Judson Wynne
Wildlife Ecologist
USGS - Southwest Biological Science Center
Colorado Plateau Research Station
2255 N. Gemini Drive
Flagstaff, AZ 86011 USA
jwynne@usgs.gov
Fax (928) 556-7092
Tel (928) 556-7172

DESCRIPTION:

 *Kappa Analysis:* The Kappa statistic is used to measure the agreement between predicted and observed categorizations of a dataset while correcting for agreement that occurs by chance. This statistic is especially useful in landscape ecology and wildlife habitat relationship (WHR) modeling for measuring the predictive accuracy of classification grids.

 *Compare Kappa Analyses:* This tool allows you to compare the kappa statistics between different analyses, perhaps comparing different observers, predictive algorithms or dates of remote sensing imagery.

 *Sample Size:* This tool provides a means to estimate the sample size required to achieve a confidence level and precision for statistical analysis.

 *Summary Statistics:* From any numeric field in a table, this function will calculate the Mean, Standard Error of the Mean, Confidence Intervals, Minimum, 1st Quartile, Median, 3rd Quartile, Maximum, Variance, Standard Deviation, Average Absolute Deviation, Skewness (normal and Fisher's G1), Kurtosis (normal and Fisher's G2), Number of Records, Number of Null Values, and Total Sum.

 *Probability Calculators:* This function will allow you to calculate the probability, cumulative probability and inverse probability (i.e. given a cumulative probability, calculate the corresponding critical value) of a wide range of statistical distributions, including the Beta, Binomial, Cauchy, Chi-Square, Exponential, F, Logistic, LogNormal, Normal, Poisson, Student's T and Weibull distributions. This function is available as a general calculator that remains open until you are finished with it, or as a Table tool that performs the calculations on all selected records in a table.

Acknowledgments: The Kappa Analysis tools in this extension are based primarily on functions described in "Assessing the Accuracy of Remotely Sensed Data: Principles and Practices" by **Russell G. Congalton** and **Kass Green** (Congalton and Green 1999). The authors recommend this source for a detailed discussion of the use of the Kappa statistic in landscape analysis.

Special thanks to **Dr. William Block**, USDA FS, Rocky Mountain Research Station for use of the retrospective dataset used in the case study. Wynne (2003) used these data as the basis of WHR model development for his M.S. thesis research. Also, special thanks to **Mr. Rudy King** and **Mrs. Kristen Covert-Bratland**, also of the Rocky Mountain Research Station, for assistance in statistical questions.

Certain tools (esp. the Field Statistics and the histogram) were originally developed by the author for the University of Arizona's Saguaro project (see http://saguaro.geo.arizona.edu/) and are included with their permission. The authors thank **Mr. Larry Kendall** of the University of Arizona, and **Mr. Scott Walker** of Northern Arizona University, for their help in developing those tools and their willingness to share them.

Special thanks also to **Dr. John Prather** and **Dr. Russell Congalton** for reviewing early drafts of this manual, and to Dr. Congalton for suggesting the correction for locational uncertainty.

The Statistical Probability tools are almost identical to those in Jenness' *Statistics and Probability Tools* extension (see http://www.jennessent.com/arcview/stats_dist.htm) and are included because they enhance and complement the statistical functions. The manual for that extension has also been incorporated into this manual.

REQUIRES: ArcView 3.x, Spatial Analyst

This extension also requires the file "avdlog.dll" be present in the ArcView/BIN32 directory (or $AVBIN/avdlog.dll) and the Dialog Designer extension be located in your ArcView/ext32 directory, which they usually are if you're running AV 3.1 or better. The Dialog Designer doesn't have to be loaded; it just has to be available. If you are running AV 3.0a, you can download the appropriate files for free from ESRI at:

http://www.esri.com/software/arcview/extensions/dialog/index.html

REVISIONS: Version 2.0, September 23, 2005: Incorporated new functions to adjust for locational uncertainty of sample points, generating statistics on multiple subsets of data, and extensive revisions to manual.

Version 2.1 (May 23, 2006) includes minor code revisions to avoid problems with Chinese versions of Windows, as well as adds menu items to search the Kappa scripts for errors.

Version 2.1a (December 10, 2007): Minor update to allow extension to work with PointZ, PointM, PolygonZ and PolyonM shapefiles.

Recommended Citation Format: For those who wish to cite this extension, the authors recommend something similar to:

Jenness, J. and J. J. Wynne. 2007. Kappa analysis (kappa_stats.avx) extension for ArcView 3.x. Jenness Enterprises. Available at: http://www.jennessent.com/arcview/kappa_stats.htm.

Please let us know if you cite this extension in a publication (jeffj@jennessent.com), so we may update the citation list accordingly.

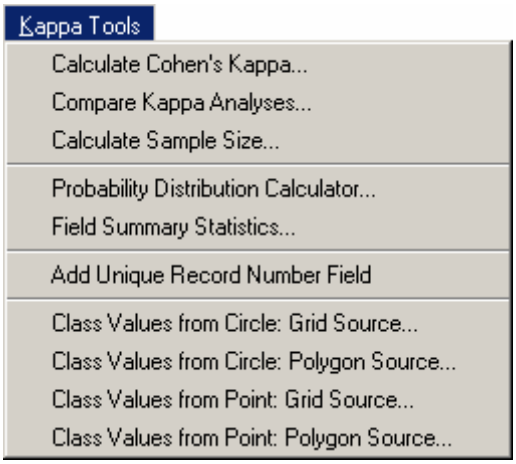
Table of Contents

General Instructions:

1) Begin by placing the "**kappa_stats.avx**" file into the ArcView extensions directory (../../Av_gis30/Arcview/ext32/).

2) After starting ArcView, load the extension by clicking on **File --> Extensions… ,** scroll down through the list of available extensions, and then click on the checkbox next to "**Kappa Analysis.**"

3) When active, this extension will add a new menu to your View menu bar:



4) This extension will also add five buttons to your View button bar: 

5) This extension will also add two buttons to your Table button bar: 

All models are wrong but some are useful.
George E.P. Box

Kappa Analysis:

Introduction

Spatially explicit models have various applications in resource management, including the development of vegetation and wildlife habitat relationship (WHR) predictive surface maps. Appropriate applications of these models are impossible without informed approaches to model development and accuracy assessment of resultant data products. In the absence of incisive model development and error analysis, spatially explicit models may be applied in ways which confound, rather than illuminate, our understanding of vegetation land cover and wildlife habitat.

Accuracy assessment provides a means of gauging model performance and thus may serve to elucidate our understanding of predictive models. End users who conduct accuracy assessment are also provided with important information regarding model reliability and suitability of the modeling process (Csuti and Crist 1998; Drost et al. 1999). Accuracy assessments are useful for: (1) ascertaining the quality of a predictive surface; (2) improving map quality by identifying and correcting sources of error; (3) facilitating a comparison of various algorithms, techniques, model developers and interpreters; and, (4) determining the relevance of the data product in the decision making process (Congalton and Green 1999).

Well-developed accuracy assessment methodologies are lagging far behind predictive modeling. In the wildlife habitat-modeling field, methods used for modeling species occurrence and abundance are more developed than methods used for assessing model predictions (Boone and Krohn 2002). Perhaps more perplexing, issues concerning data quality and model accuracy in landscape analyses have received little attention in the management literature (Hess 1994; Hess and Bay 1997; Luck and Wu 2002). When accuracy assessments are conducted, generally only the overall accuracy metric is provided (Congalton and Green 1999). However, there are numerous accuracy metrics available which are highly useful in determining overall model performance.

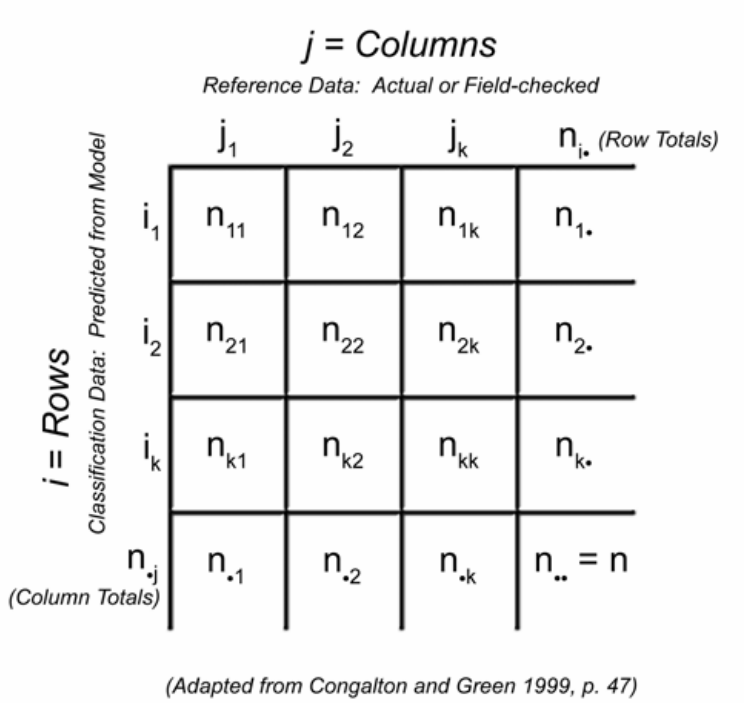
The Kappa Analysis extension will provide end users with a packaged approach for accuracy assessment, using the Kappa statistic as well as several additional model performance metrics. Additional metrics include overall accuracy, overall misclassification rate, model specificity and sensitivity, omission and commission errors, and positive and negative predictive power. When multiple competing models are available, these metrics can be used to quantitatively compare and identify the “better” model within a multi-criteria model selection process.

Kappa Analysis strives to raise the bar for accuracy assessment and provide a quantitative approach to making model comparisons. We hope end users will agree.

The Kappa Statistic

The Kappa statistic is used to measure the agreement between two sets of categorizations of a dataset while correcting for chance agreements between the categories. In terms of landscape ecology and wildlife habitat analysis, this statistic is especially useful for estimating the accuracy of predictive models by measuring the agreement between the predictive model and a set of field-surveyed sample points. The Kappa statistic makes use of both the overall accuracy of the model and the accuracies within each category, both in terms of the predictive model and the field-surveyed sample points, to correct for chance agreement between categories.

This concept can best be understood by viewing the predictive model values and the field-surveyed values in an error matrix:



(Adapted from Congalton and Green 1999, p. 47)

In this matrix, the rows represent the predicted values while the columns represent the observed values. Each cell represents the number of sample points that were classified as *i* and observed to be *j*. The diagonal (where *i* = *j*) represents cases where the predicted value agreed with the observed value. The off-diagonal cells contain misclassified values, and the row and column describe exactly how those values were misclassified.

The row totals are the number of sample points classified into category *i* by the producer's classification model, and are calculated as:

$$n_{i\bullet} = \sum_{j=1}^{k} n_{ij}$$

The column totals are the number of sample points classified into category *j* by the user's field tests, and are calculated as:

$$n_{\bullet j} = \sum_{i=1}^{k} n_{ij}$$

The Kappa statistic provides a measure of agreement between the predicted values and the observed values, and is calculated as (*sensu* Congalton and Green 1999:50):

$$\text{Estimated Kappa } \hat{K} = \frac{n\sum_{i=1}^{k} n_{ii} - \sum_{i=1}^{k} n_{i\cdot} n_{\cdot i}}{n^2 - \sum_{i=1}^{k} n_{i\cdot} n_{\cdot i}}$$

$$\text{with Variance } \hat{\text{var}}\left(\hat{K}\right) = \frac{1}{n}\left(\frac{\theta_1(1-\theta_1)}{(1-\theta_2)^2} + \frac{2(1-\theta_1)(2\theta_1\theta_2 - \theta_3)}{(1-\theta_2)^3} + \frac{(1-\theta_1)^2(\theta_4 - 4\theta_2^2)}{(1-\theta_2)^4} \right)$$

where:

$$\theta_1 = \frac{1}{n}\sum_{i=1}^{k} n_{ii}$$

$$\theta_2 = \frac{1}{n^2}\sum_{i=1}^{k} n_{i\cdot} n_{\cdot i}$$

$$\theta_3 = \frac{1}{n^2}\sum_{i=1}^{k} n_{ii}\left(n_{i\cdot} + n_{\cdot i}\right)$$

$$\theta_4 = \frac{1}{n^3}\sum_{i=1}^{k}\sum_{j=1}^{k} n_{ij}\left(n_{i\cdot} + n_{\cdot j}\right)$$

The advantage of the Kappa statistic is that it corrects for chance agreements between the observed and predicted values. The logic behind this can be summarized as follows (from Agresti, 1990:366-367):

1) If π_{ij} denotes the probability that a point on the landscape will be predicted to be *i* and observed to be *j* (i.e. the probability of being in cell *ij* in the error matrix), then

$$\Pi_o = \sum_{i=1}^{k} \pi_{ii}$$

is the overall probability of correctly classifying a point, and is equal to the overall accuracy described below.

2) If the predicted and observed classifications are statistically independent (which they probably are not; hence this correction factor), then any agreement would occur purely by chance. In this case

$$\pi_{ii} = \pi_{i\cdot}\pi_{\cdot i}$$

and the overall probability of correctly classifying a point purely by chance is

$$\Pi_e = \sum_{i=1}^{k} \pi_{i\cdot}\pi_{\cdot i}$$

3) Therefore, $\Pi_o - \Pi_e$ equals the excess classification accuracy occurring beyond the accuracy expected purely by chance.

4) Kappa adjusts for chance accuracy by dividing this excess accuracy $(\Pi_o - \Pi_e)$ by the excess accuracy that would have occurred if the observed accuracy was perfect $(1 - \Pi_e)$. Note that the "1" in the denominator replaces Π_o with a "perfect" observed accuracy value. Therefore (when expressed in terms of probabilities rather than cell counts), Kappa can be stated as

$$\text{Estimated Kappa } \hat{K} = \frac{\sum_{i=1}^{k} \pi_{ii} - \sum_{i=1}^{k} \pi_{i\cdot}\pi_{\cdot i}}{1 - \sum_{i=1}^{k} \pi_{i\cdot}\pi_{\cdot i}} = \frac{\Pi_o - \Pi_e}{1 - \Pi_e}$$

5) The $\hat{K}$ statistic typically ranges between 0 and 1, with values closest to 1 reflecting highest agreement. Negative values are possible but rare.

A CAUTION: Congalton and Green (1999:58-59) point out that some researchers object to Kappa when used to assess remote sensing accuracy because it underestimates the actual classification accuracy, in that the chance agreement term $\Pi_e = \sum_{i=1}^{k} \pi_{i\cdot}\pi_{\cdot i}$ includes some agreement that is not purely due to chance. This is especially the case when the marginals (row and column totals) are not fixed *a priori*, which is normally the case in remote sensing. We suspect that it is rare that a researcher would decide *a priori* that *X*% of their landscape will be classified as *Y*.

However, Congalton and Greene also point out that, even given this potential problem, Kappa comes with some powerful statistical properties that make it extremely useful for assessing accuracy.

For example, because $\hat{K}$ is asymptotically normally distributed, a basic Z-score can be used for significance testing:

$$Z = \frac{\hat{K}_1}{\sqrt{\hat{\text{var}}\left(\hat{K}_1\right)}}$$

Thereafter, the significance of the Z-score may be evaluated based on the associated *P*-value. If hypothesis testing is employed, the null hypothesis H_0: $K_1 = 0$, means the classification accuracy is no different than a purely random classification. The alternative hypothesis H_1: $K_1 \neq 0$, means the accuracy is significantly different (hopefully better) than a random classification, and H_0 would be rejected at some critical Z-score.

Also, the Kappa statistic and variance can be used to calculate a confidence interval around Kappa, where:

$$CI = \hat{K}_1 \pm Z_{\alpha/2}\sqrt{V\left(\hat{K}_1\right)}$$

Furthermore, because the variance can be estimated, 2 $\hat{K}$ values may be compared to see if they are significantly different. This is useful when you are interested in whether different models, methodologies or interpreters produce significantly different results, or if a landscape has changed over time (Congalton and Mead 1983). In this case the *Z*-score is calculated as

$$Z = \frac{\left|\hat{K}_1 - \hat{K}_2\right|}{\sqrt{\hat{\text{var}}\left(\hat{K}_1\right) + \hat{\text{var}}\left(\hat{K}_2\right)}}$$

This comparison can be extended to a general test for multiple equal $\hat{K}$ values by estimating the supposed 'common' $\hat{K}$ value (as described by Fleiss [1981:222]):

$$\text{'Common' } \hat{K} = \frac{\sum_{m=1}^{g} \frac{\hat{K}_m}{Var\left(\hat{K}_m\right)}}{\sum_{m=1}^{g} \frac{1}{Var\left(\hat{K}_m\right)}}$$

This 'common' $\hat{K}$ value can then be used to test for equal $\hat{K}$ values on the Chi-Square distribution with *g - 1* degrees of freedom:

$$\chi^2_{equal\ \hat{K}} = \sum_{m=1}^{g} \frac{\left(\hat{K}_m - \hat{K}_{'Common'}\right)^2}{Var\left(\hat{K}_m\right)}$$

Model Accuracy

The overall accuracy of the model is simply defined as the total number of correct classifications divided by the total number of sample points, and is calculated as:

$$\text{Overall Accuracy} = \frac{\sum_{i=1}^{k} n_{ii}}{n}$$

The overall accuracy is often the only accuracy statistic reported with predictive landscape models (Congalton and Green 1999:46), but the error matrix provides a means to calculate numerous additional metrics describing model performance. In particular, accuracies of each category from both the model's (or producer's) perspective and the observer's (or user's) perspective may be useful in determining model performance. The producer's accuracy reflects the proportion of sample points correctly classified as *X* over the number of points observed to be *X*, and is calculated as:

$$\text{Producer's Accuracy} = \frac{n_{jj}}{n_{.j}}$$

The user's accuracy reflects the proportion of sample points correctly classified as *X* over the number of points predicted to be *X*, and is calculated as:

$$\text{User's Accuracy} = \frac{n_{ii}}{n_{i.}}$$

The difference between producer's and user's accuracy is the difference between defining accuracy in terms of how well the landscape can be mapped (producer's accuracy) versus how reliable the classification map is to the user (user's accuracy; Story & Congalton 1986). For example, please review the sample error matrix below.

		Reference Data			
		Deciduous Forest	**Coniferous Forest**	**Grassland**	**Total**
Classification Data	**Deciduous Forest**	60	22	4	86
	Coniferous Forest	2	30	3	35
	Grassland	1	4	10	15
	Total	63	56	17	136

Overall Accuracy: $\frac{100}{136} = 73.5\%$

Category	**Producer's Accuracy**	**User's Accuracy**
Deciduous Forest	$\frac{60}{63} = 95.2\%$	$\frac{60}{86} = 69.8\%$
Coniferous Forest	$\frac{30}{56} = 53.6\%$	$\frac{30}{35} = 85.7\%$
Grassland	$\frac{10}{17} = 58.8\%$	$\frac{10}{15} = 66.7\%$

Suppose we are interested in the accuracy of deciduous forest classification. The overall accuracy of this example is 73.5%, but the overall accuracy explains little regarding how well the classification process

captures deciduous forest. Fortunately, the error matrix provides us with the means to calculate deciduous forest classification in two different ways.

Notice that the reference data (in columns) includes 63 cases identified as deciduous forest. Of these 63 cases, 60 were correctly classified as deciduous forest, suggesting that deciduous forest was accurately classified 95.2% of the time. However, notice also that 86 cases were actually classified as deciduous forest, and therefore a classification of deciduous forest is only correct 69.8% of the time. Although deciduous forest may be classified with a particular accuracy (producer's accuracy = 95.2%), the actual reliability of the classification to the user may be much different (user's accuracy = 69.8%).

Classification Considerations

This extension calculates the accuracy of a predictive model developed using a particular classification system and a set of reference sample points. Not surprisingly, the choice of classification system can greatly influence the classification accuracy. Congalton (1991) discusses several guidelines that should be followed when developing an effective classification system.

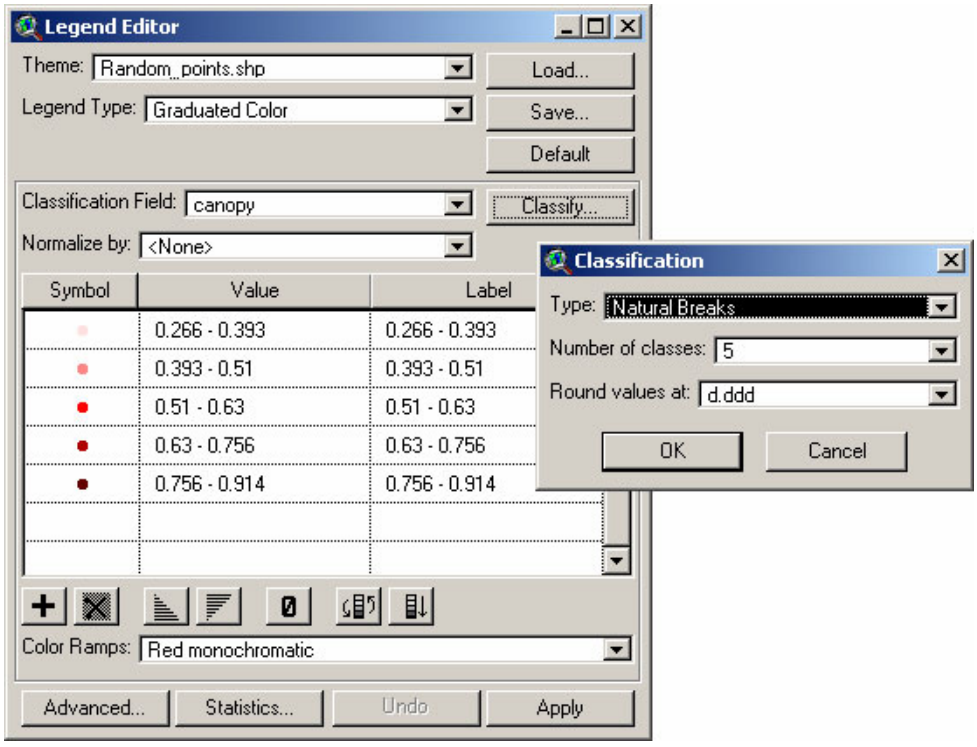
1) All areas to be classified should fall into one and only one category.

2) All areas should be classified. No areas should be left unclassified.

3) If possible, use a hierarchical classification system. If necessary, two or more categories may be collapsed into a single, more general category. This may be necessary if you are unable to achieve your minimum accuracy requirement using the original set of categories.

4) Recognize that some standard classifications may be fairly arbitrary and you may do better to use natural breaks in the data. For example, continuous values like canopy closure, tree density or basal area are often separated into categories based on artificial breakpoints (e.g., Class A = 25% - 50% Canopy Closure, while Class B = 51% - 75% Canopy Closure, etc.). If your data cluster around a breakpoint, then your classification system will not capture a possibly important phenomenon.

   Slocum et al. (2005) provides some insights regarding how to implement the Fisher-Jenks algorithm for determining natural breakpoints within a dataset. As a shortcut, both ArcView and ArcGIS implement a version of the Fisher-Jenks algorithm in their legend-generation tools and therefore you can easily identify breakpoints using the legend editor.

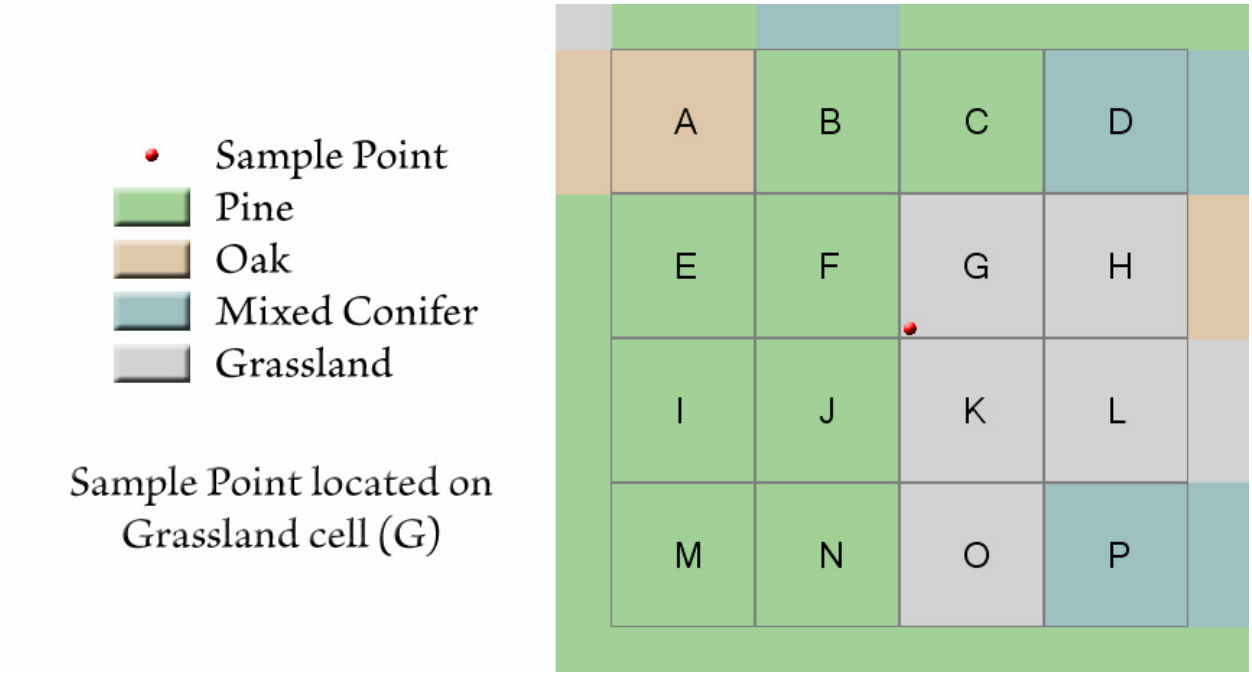


Sample Point Considerations

Congalton and Green (1999:11-25) review sample size and sample design, and this extension includes tools to calculate sample size necessary to meet a desired accuracy level according to their guidelines (see p. 35 of this manual). As a general rule of thumb, Congalton and Green recommend a minimum of 50 sample points per category, increasing to 75-100 sample points per category if you have large numbers of categories (> 12). This may be an obvious point, but under no circumstances should you completely fail to sample any of the classification categories. It is difficult to estimate the classification accuracy of a particular category if you don't ever check whether that category was correctly classified.

Adjusting for Locational Uncertainty

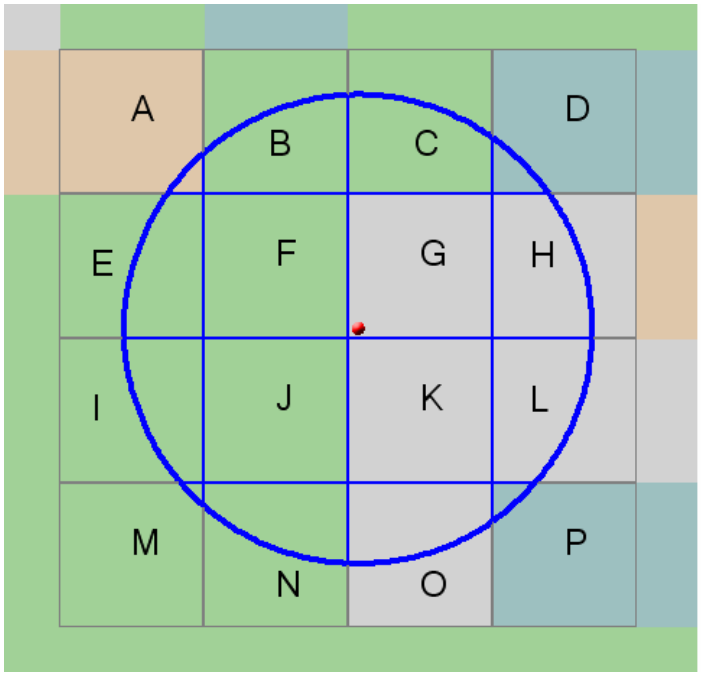
Locational uncertainty is a commonly unacknowledged source of error. A GIS assumes locations are perfectly precise and accurate, and the classification value at a particular sample point may be extracted simply by checking the cell value of the classification grid or polygon intersected by the sample point. In the example below, the sample point is located on a Grassland cell and, by default, the classification value of this point will be Grassland.



However, the coordinates of that sample point are probably not known exactly. For example, if you use a GPS receiver to determine the point location, then you should have some idea of the general accuracy possible with that receiver. Some of the higher-end receivers also allow you to collect positional accuracy metadata along with the actual locations, providing a much better sense of locational accuracy during a particular survey session.

As an aside, GPS has become an indispensable tool for collecting geospatial data. In general, however, we strongly recommend that you read past the "X-meter Accuracy" blurb on the box that your GPS came in and attempt to understand exactly how that accuracy is defined. Most receivers define accuracy in terms of a distance from the true location within which some percentage of points actually occurs. Common values include 50% (referred to as Circular Error Probable, or CEP) or 95%. Notice there may be a very large difference between saying that 50% vs. 95% of points lie within X meters of the true location. Furthermore, the stated accuracy often depends on differential correction, the codes and frequencies used (P-code and/or C/A-code, or extremely high-accuracy carrier-wave receivers), good satellite coverage and spatial arrangement, and low levels of common GPS error (ephemeris, atmospheric delays, multipath errors and receiver problems). For a thorough discussion on how GPS systems work, we recommend Tom Logsdon's "The Navstar Global Position System" (Logsdon 1992).

If you suspect that significant locational uncertainty exists in your sample points, it may be reasonable to take a more conservative approach to determine the predicted classification value at each point. The illustration above takes the simple approach of using the cell value at a point. An alternative would be to use the majority value in a circular area surrounding the point (illustrated below). Note the same point would be classified differently using the circular area instead of the cell value of where the point resides.



Cell Areas within Circle

Cell A:	Class = Oak	Area = 0.060 hectares
Cell B:	Class = Pine	Area = 0.838 hectares
Cell C:	Class = Pine	Area = 0.911 hectares
Cell D:	Class = Mixed Conifer	Area = 0.133 hectares
Cell E:	Class = Pine	Area = 0.706 hectares
Cell F:	Class = Pine	Area = 1.517 hectares
Cell G:	Class = Grassland	Area = 1.517 hectares
Cell H:	Class = Grassland	Area = 0.917 hectares
Cell I:	Class = Pine	Area = 0.639 hectares
Cell J:	Class = Pine	Area = 1.517 hectares
Cell K:	Class = Grassland	Area = 1.517 hectares
Cell L:	Class = Grassland	Area = 0.850 hectares
Cell M:	Class = Pine	Area = 0.020 hectares
Cell N:	Class = Pine	Area = 0.644 hectares
Cell O:	Class = Grassland	Area = 0.716 hectares
Cell P:	Class = Mixed Conifer	Area = 0.065 hectares

Cell Areas per Class

Pine: 6.791 hectares
Grassland: 5.517 hectares
Mixed Conifer: 0.198 hectares
Oak: 0.060 hectares

Final Classification will be Pine

Our extension offers the option to use circular neighborhoods in the Kappa analysis (see p. 30 in this manual), as well as a function to calculate circular neighborhood values in a separate table (under the "Kappa Tools" menu in your View; see the discussion on *Additional Menu Functions* on p. 73 in this manual).

Additional Metrics

Several other metrics may be derived from the error matrix, which may be used to further describe model performance. These include model sensitivity and specificity, commission and omission error, and positive and negative predictive power. For a detailed description of these concepts, please refer to Fielding and Bell (1997) and Lurz et al (2001).

Computations are based on a "Confusion Matrix", reflecting the four possible ways a sample point may be classified and observed:

Confusion Matrix

		Actual Values	
		+	-
Predicted Values	+	*a*	*b*
	-	*c*	*d*

(a) = number of times a classification agreed with the observed value

(b) = number of times a point was classified as *X* when it was observed to not be *X*.

(c) = number of times a point was not classified as *X* when it was observed to be *X*.

(d) = number of times a point was not classified as *X* when it was not observed to be *X*.

Given this confusion matrix:

$$\text{Sensitivity} = \frac{a}{a+c} \qquad \text{(Equivalent to Producer's Accuracy)}$$

$$\text{Specificity} = \frac{d}{b+d}$$

$$\text{False Positive Rate (Commission Error)} = \frac{b}{b+d} = \left(1 - \text{Specificity}\right)$$

$$\text{False Negative Rate (Omission Error)} = \frac{c}{a+c} = \left(1 - \text{Sensitivity}\right)$$

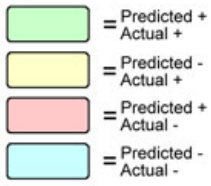
$$\text{Positive Predictive Power} = \frac{a}{a+b} \qquad \text{(Equivalent to User's Accuracy)}$$

$$\text{Negative Predictive Power} = \frac{d}{c+d}$$

Therefore, given a five -category error matrix, with classified values in rows and observed values in columns:

ID	1	2	3	4	5	SUM
1	14	4	11	3	2	34
2	4	120	13	15	22	174
3	24	4	80	3	8	119
4	0	0	0	0	0	0
5	1	7	19	1	7	35
SUM	43	135	123	22	39	362

individual sensitivity, specificity, positive and negative predictive power, user/producer error, and omission/commission error for each category may be derived:



Confusion Matrix for Class 1

		Actual Values	
		+	-
Predicted Values	+	14	20
	-	29	299

Sensitivity = 14 / (14 + 29) = 0.3256
Specificity = 299 / (20 + 299) = 0.9373
- Pred. Power = 299 / (29 + 299) = 0.9116
Omission Error = 29 / (14 + 29) = 0.6744
Commission Error = 20 / (20 + 299) = 0.0627

ID	1	2	3	4	5	SUM
1	14	4	11	3	2	34
2	4	120	13	15	22	174
3	24	4	80	3	8	119
4	0	0	0	0	0	0
5	1	7	19	1	7	35
SUM	43	135	123	22	39	362

Producer Accuracy = 14 / (14 + 4 + 24 + 0 + 1) = 0.3256
User Accuracy = 14 / (14 + 4 + 11 + 3 + 2) = 0.4118

Confusion Matrix for Class 2

		Actual Values	
		+	-
Predicted Values	+	120	54
	-	15	173

Sensitivity = 120 / (120 + 15) = 0.8889
Specificity = 173 / (54 + 173) = 0.7621
- Pred. Power = 173 / (15 + 173) = 0.9202
Omission Error = 15 / (120 + 15) = 0.1111
Commission Error = 54 / (54 + 173) = 0.2379

ID	1	2	3	4	5	SUM
1	14	4	11	3	2	34
2	4	120	13	15	22	174
3	24	4	80	3	8	119
4	0	0	0	0	0	0
5	1	7	19	1	7	35
SUM	43	135	123	22	39	362

Producer Accuracy = 120 / (4 + 120 + 4 + 0 + 7) = 0.8889
User Accuracy = 120 / (4 + 120 + 13 + 15 + 22) = 0.6897

Confusion Matrix for Class 3

		Actual Values	
		+	-
Predicted Values	+	80	39
	-	43	200

Sensitivity = 80 / (80 + 43) = 0.6504
Specificity = 200 / (39 + 200) = 0.8368
- Pred. Power = 200 / (43 + 200) = 0.8230
Omission Error = 43 / (80 + 43) = 0.3496
Commission Error = 39 / (39 + 200) = 0.1632

ID	1	2	3	4	5	SUM
1	14	4	11	3	2	34
2	4	120	13	15	22	174
3	24	4	80	3	8	119
4	0	0	0	0	0	0
5	1	7	19	1	7	35
SUM	43	135	123	22	39	362

Producer Accuracy = 80 / (11 + 13 + 80 + 0 + 19) = 0.6504
User Accuracy = 80 / (24 + 4 + 80 + 3 + 8) = 0.6723

Confusion Matrix for Class 4

		Actual Values	
		+	-
Predicted Values	+	0	0
	-	22	340

Sensitivity = 0 / (0 + 22) = 0.0000
Specificity = 340 / (0 + 340) = 1.0000
- Pred. Power = 340 / (22 + 340) = 0.9392
Omission Error = 22 / (0 + 22) = 1.0000
Commission Error = 0 / (0 + 340) = 0.0000

ID	1	2	3	4	5	SUM
1	14	4	11	3	2	34
2	4	120	13	15	22	174
3	24	4	80	3	8	119
4	0	0	0	0	0	0
5	1	7	19	1	7	35
SUM	43	135	123	22	39	362

Producer Accuracy = 0 / (3 + 15 + 3 + 0 + 1) = 0.0000
User Accuracy = 0 / (0 + 0 + 0 + 0 + 0) = null

Confusion Matrix for Class 5

		Actual Values	
		+	-
Predicted Values	+	7	28
	-	32	295

Sensitivity = 7 / (7 + 32) = 0.1795
Specificity = 295 / (28 + 295) = 0.9133
- Pred. Power = 295 / (32 + 295) = 0.9021
Omission Error = 32 / (7 + 32) = 0.8205
Commission Error = 28 / (28 + 295) = 0.0867

ID	1	2	3	4	5	SUM
1	14	4	11	3	2	34
2	4	120	13	15	22	174
3	24	4	80	3	8	119
4	0	0	0	0	0	0
5	1	7	19	1	7	35
SUM	43	135	123	22	39	362

Producer Accuracy = 7 / (2 + 22 + 8 + 0 + 7) = 0.1795
User Accuracy = 7 / (1 + 7 + 19 + 1 + 7) = 0.2000

Each of these additional metrics describes model performance in terms of that particular class or category. Now, weighted average model statistics may be generated by combining these metrics over all classes, weighting them according to the relative proportion of values in a given class.

Computationally, this may be calculated by collapsing these five per-category confusion matrices into a single confusion matrix by adding up the respective components, thereby producing an overall confusion matrix.

a: $\sum_{i=1}^{k} n_{ii}$ = Sum of diagonal elements

b: $\sum_{i=1}^{k}\sum_{j\neq i}^{k} n_{ij}$ = Sum of non-diagonal elements

c: $\sum_{j=1}^{k}\sum_{i\neq j}^{k} n_{ij}$ = Sum of non-diagonal elements

d: $\sum_{i=1}^{k}\sum_{i\neq j}^{k}\sum_{j\neq i}^{k} n_{ij}$ = Sum of non-row/column elements

Overall Confusion Matrix

		Actual Values	
		+	-
Predicted Values	+	$\sum_{i=1}^{k} n_{ii}$	$\sum_{i=1}^{k}\sum_{j\neq i}^{k} n_{ij}$
	-	$\sum_{j=1}^{k}\sum_{i\neq j}^{k} n_{ij}$	$\sum_{i=1}^{k}\sum_{i\neq j}^{k}\sum_{j\neq i}^{k} n_{ij}$

(Note $b = c$)

Therefore,

Overall Weighted Average Sensitivity is calculated as:

$$\frac{a}{a+c} = \frac{\sum_{i=1}^{k} n_{ii}}{\sum_{i=1}^{k} n_{ii} + \sum_{j=1}^{k}\sum_{i\neq j}^{k} n_{ij}} = \frac{(14+120+80+0+7)}{(14+120+80+0+7)+(29+15+43+22+32)} = 0.6105$$

Overall Weighted Average Specificity:

$$\frac{d}{b+d} = \frac{\sum_{i=1}^{k}\sum_{i\neq j}^{k}\sum_{j\neq i}^{k} n_{ij}}{\sum_{i=1}^{k}\sum_{j\neq i}^{k} n_{ij} + \sum_{i=1}^{k}\sum_{i\neq j}^{k}\sum_{j\neq i}^{k} n_{ij}} = \frac{(299+173+200+340+295)}{(20+54+39+0+28)+(299+173+200+340+295)} = 0.9026$$

Overall Weighted Average Omission Error:

$$\frac{c}{a+c} = \frac{\sum_{j=1}^{k}\sum_{i\neq j}^{k} n_{ij}}{\sum_{i=1}^{k} n_{ii} + \sum_{j=1}^{k}\sum_{i\neq j}^{k} n_{ij}} = \frac{(29+15+43+22+32)}{(14+120+80+0+7)+(29+15+43+22+32)} = 0.3896$$

Overall Weighted Average Commission Error:

$$\frac{b}{b+d} = \frac{\sum_{i=1}^{k}\sum_{j\neq i}^{k} n_{ij}}{\sum_{i=1}^{k}\sum_{j\neq i}^{k} n_{ij} + \sum_{i=1}^{k}\sum_{i\neq j}^{k}\sum_{j\neq i}^{k} n_{ij}} = \frac{(20+54+39+0+28)}{(20+54+39+0+28)+(299+173+200+340+295)} = 0.0974$$

2 x 2 Presence/Absence Models

IMPORTANT: Many researchers design a study to generate a simple two-category presence/absence predictive habitat model. In this case, the full error matrix is identical to the basic confusion matrix, and the "overall" weighted average statistics may not be suitable to the researcher. More often, the researcher will be interested in the statistics describing either the "presence" or the "absence" category.

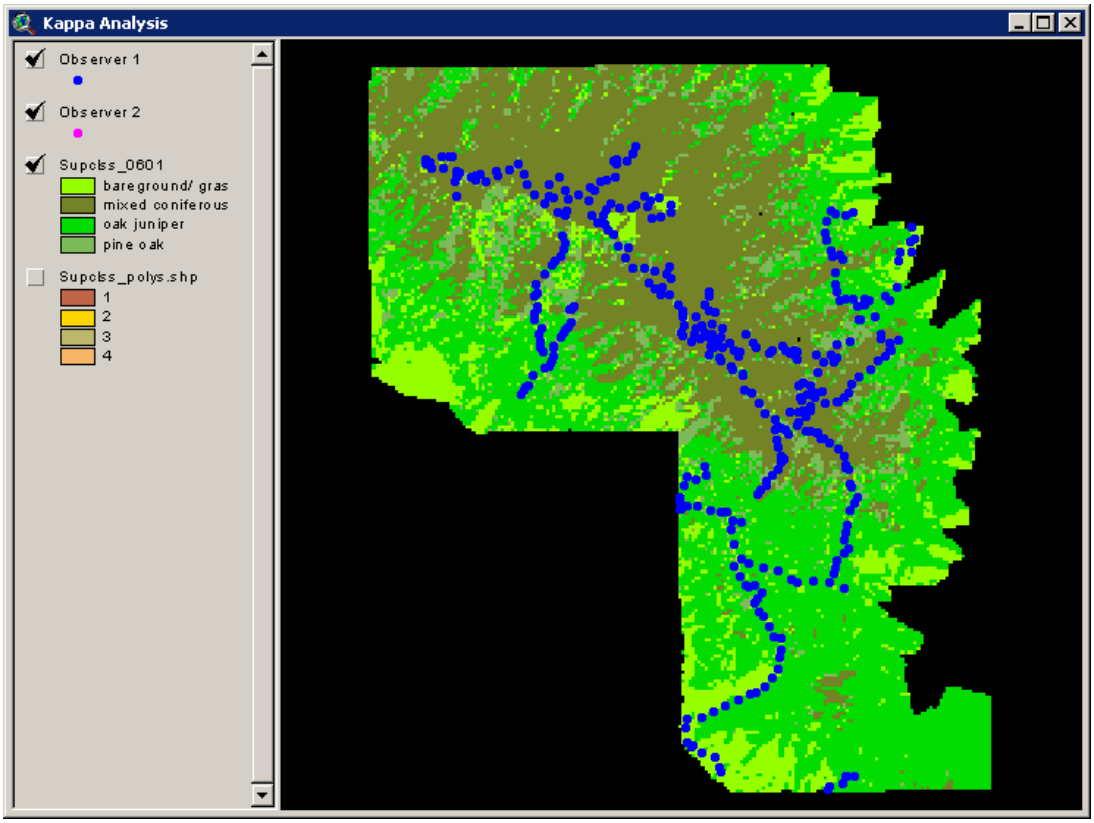
For example, the researcher may be interested in the sensitivity and specificity of the two-category model. With only 2 categories, the sensitivity of the "presence" category is equivalent to the specificity of the "absence", and vice versa. Thus, "overall" weighted average sensitivity and specificity will be the same value.

If one is interested in how well the model successfully predicts a species' habitat (or any "presence/absence" type of analysis), this information will be provided in the statistics for the "presence".

Conversely, if one is interested in how well the model successfully predicts where a species will not be, the user should refer to the statistics for the "absence" category.

Using This Extension to Generate the Kappa Statistic

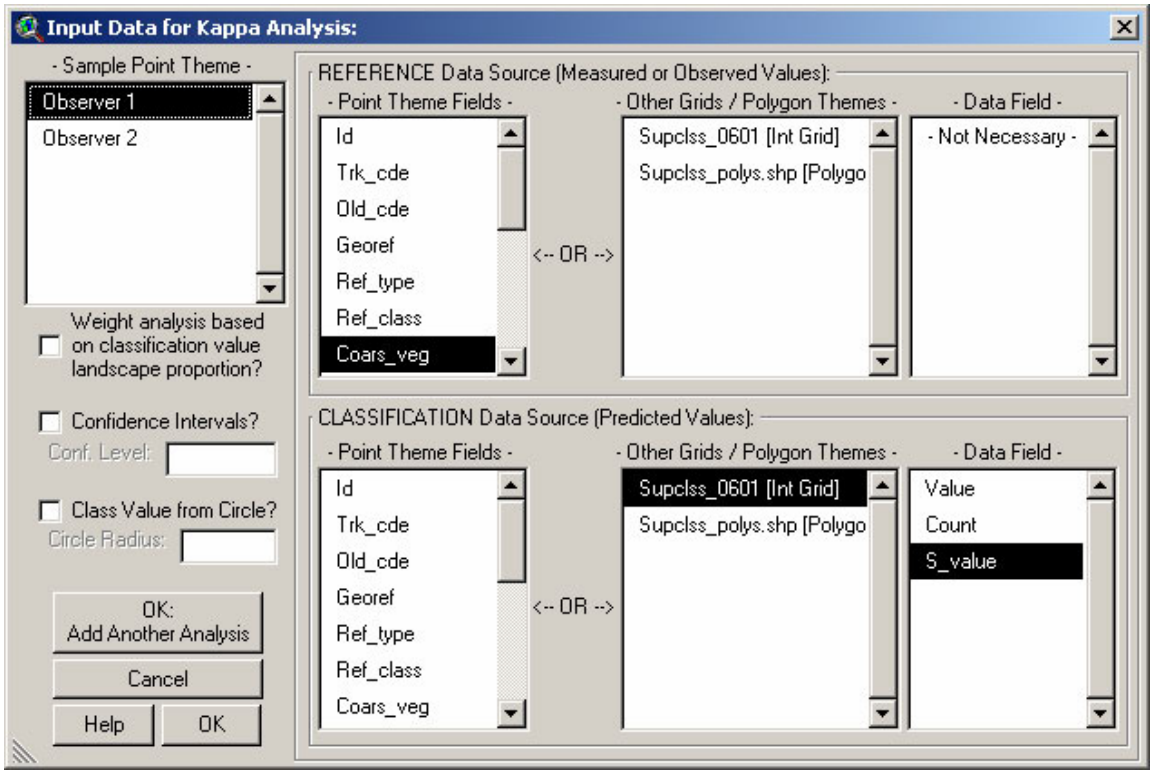
Given a view with reference and classification data:



In this case the grid named "Supclss_0601" has been derived from the predictive model and represents vegetation land cover classifications based on satellite imagery. The point theme "Observer 1" represents a set of points surveyed by an observer in the field, and the point theme "Observer 2" represents the same set of points surveyed by another observer in the field. The Kappa analysis in this example will measure the agreement between field observer 1 and the classification grid.

NOTE: The sample points used in this example are not randomly distributed. Ideally, sample points used in the Kappa Analysis should be as random as possible to minimize spatial autocorrelation, and usually selected using some type of systematic or random sampling design (see *Choosing Sample Locations* on p. 37) to ensure that all classifications are adequately sampled. In this example, the terrain was too extreme to allow us to sample random points either economically or safely, so we instead chose a semi-systematic sampling design. We collected data at ½ km intervals along all traversable roads and hiking trails spanning the elevational gradient of the mountain range. This dataset therefore represents a reasonable compromise between the ideal data for accuracy assessments (G.L. Berlin, Northern Arizona University, personal communication) and the practical reality of limited time and budgets faced by managers.

Click the  button to start analysis:



Choose the point theme containing your reference sampling locations (in this case, “Observer 1”) and the source containing the reference values (in this case, the field [Coars_veg] in “Observer 1”). In most cases the reference data will be saved in a field in the point theme. However, if necessary you have the option to extract the reference data from either grid or polygon themes.

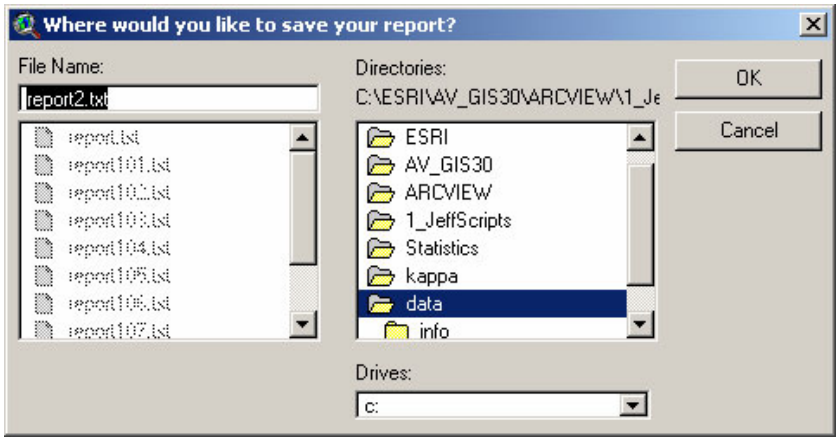
Next, choose the source for your classification data, derived from your predictive model (in this case, the field [S_value] in the grid “Supclss_0601”). These values may be selected from either another field in the point theme or from a grid or polygon theme in the view. Click “OK” when you have made your selections.

A Note regarding Reference and Classification Data Formats:

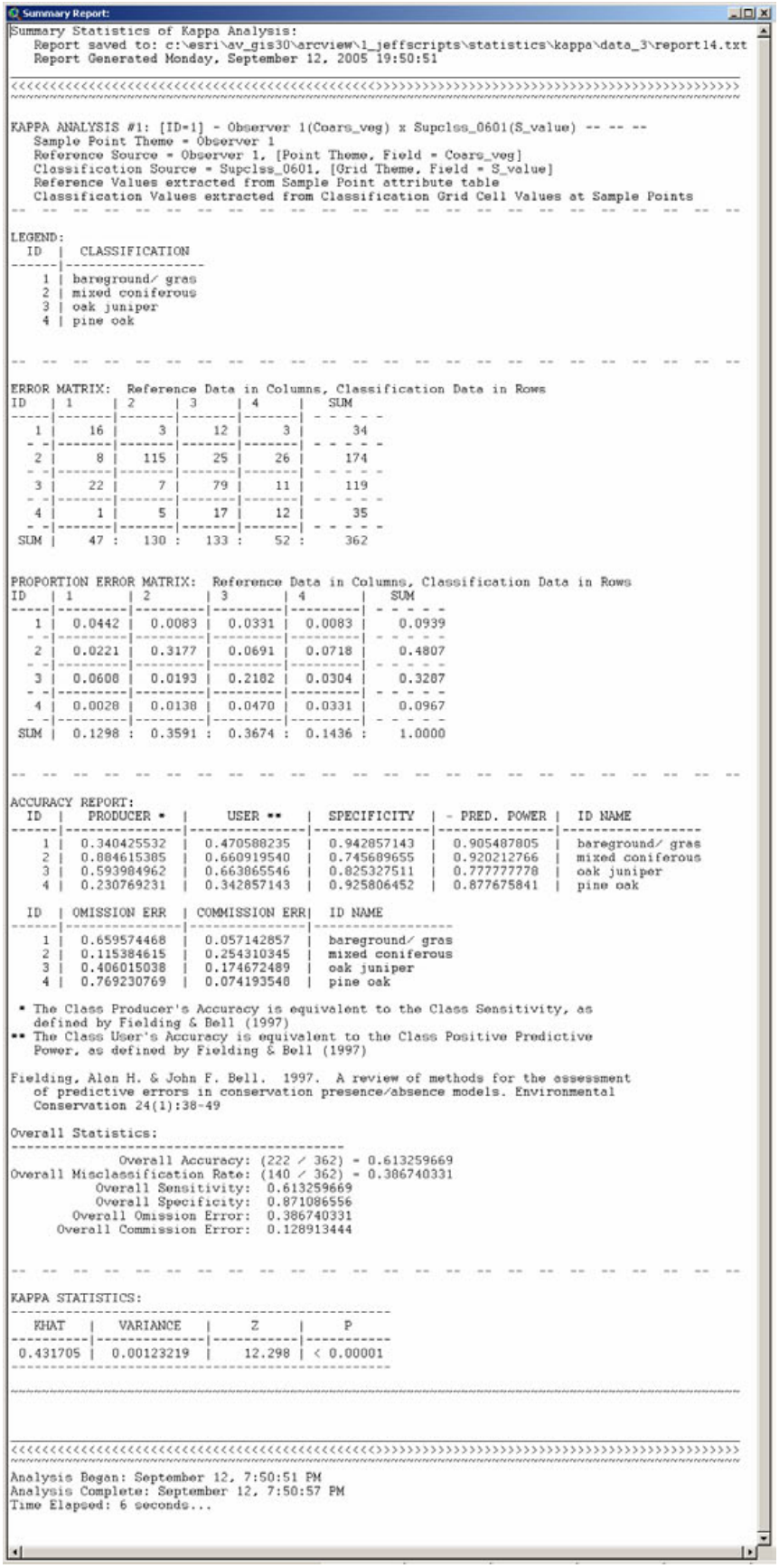
This extension compares a set of classification values to a set of reference values, and both sets of values can come from multiple potential sources. The example above is probably the most straightforward. The reference values are incorporated into the sample point theme attribute table and the classification values are extracted directly from the classification grid. However, this extension may also use reference data extracted from grids or polygon themes, as well as classification data incorporated in the point theme attribute table.

Our extension actually runs fastest if both classification and reference values are available in the point theme attribute table because it does not have to run time-consuming grid or polygon intersection operations. Large numbers of grid operations can also cause Spatial Analyst to crash (see *Troubleshooting* on p. 79 of this manual), so you may occasionally find it advantageous to add both classification and reference values to your sample point attribute table. For help with this operation, please refer to the discussions on *Generating Class Values* beginning on p. 73, and the section on linking tables on p. 77 of this manual.

Next, specify the location to save the text report. This report will also open automatically after the analysis, but you have the option here to specify the name and location of the output file:



Once the calculations are complete, the analysis report will open:

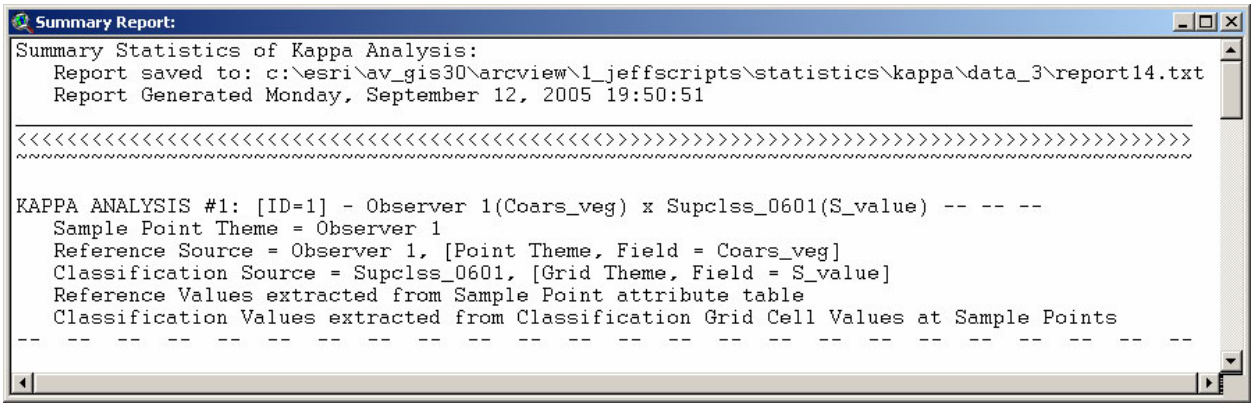


Summary Report:

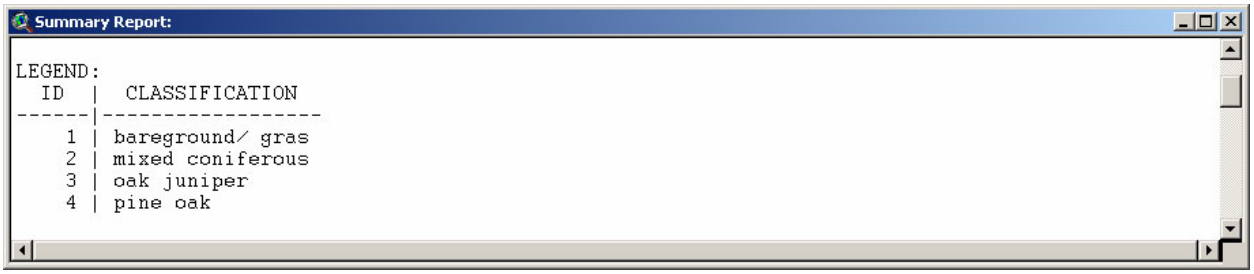
```
Summary Statistics of Kappa Analysis:
   Report saved to: c:\esri\av_gis30\arcview\1_jeffscripts\statistics\kappa\data_3\report14.txt
   Report Generated Monday, September 12, 2005 19:50:51

KAPPA ANALYSIS #1: [ID=1] - Observer 1(Coars_veg) x Supclss_0601(S_value) -- -- --
   Sample Point Theme = Observer 1
   Reference Source = Observer 1, [Point Theme, Field = Coars_veg]
   Classification Source = Supclss_0601, [Grid Theme, Field = S_value]
   Reference Values extracted from Sample Point attribute table
   Classification Values extracted from Classification Grid Cell Values at Sample Points

LEGEND:
  ID  |  CLASSIFICATION
------|-----------------
    1 | bareground/ gras
    2 | mixed coniferous
    3 | oak juniper
    4 | pine oak

ERROR MATRIX:  Reference Data in Columns, Classification Data in Rows
ID   | 1     | 2     | 3     | 4     |   SUM
-----|-------|-------|-------|-------| - - - - -
   1 |    16 |     3 |    12 |     3 |      34
   2 |     8 |   115 |    25 |    26 |     174
   3 |    22 |     7 |    79 |    11 |     119
   4 |     1 |     5 |    17 |    12 |      35
 SUM |    47 :   130 :   133 :    52 :     362

PROPORTION ERROR MATRIX:  Reference Data in Columns, Classification Data in Rows
ID   | 1       | 2       | 3       | 4       |   SUM
-----|---------|---------|---------|---------| - - - - -
   1 |  0.0442 |  0.0083 |  0.0331 |  0.0083 |    0.0939
   2 |  0.0221 |  0.3177 |  0.0691 |  0.0718 |    0.4807
   3 |  0.0608 |  0.0193 |  0.2182 |  0.0304 |    0.3287
   4 |  0.0028 |  0.0138 |  0.0470 |  0.0331 |    0.0967
 SUM |  0.1298 :  0.3591 :  0.3674 :  0.1436 :    1.0000

ACCURACY REPORT:
  ID  |   PRODUCER *  |     USER **   |  SPECIFICITY  | - PRED. POWER |  ID NAME
------|---------------|---------------|---------------|---------------|-----------------
    1 |  0.340425532  |  0.470588235  |  0.942857143  |  0.905487805  |  bareground/ gras
    2 |  0.884615385  |  0.660919540  |  0.745689655  |  0.920212766  |  mixed coniferous
    3 |  0.593984962  |  0.663865546  |  0.825327511  |  0.777777778  |  oak juniper
    4 |  0.230769231  |  0.342857143  |  0.925806452  |  0.877675841  |  pine oak

  ID  | OMISSION ERR  | COMMISSION ERR|  ID NAME
------|---------------|---------------|-----------------
    1 |  0.659574468  |  0.057142857  |  bareground/ gras
    2 |  0.115384615  |  0.254310345  |  mixed coniferous
    3 |  0.406015038  |  0.174672489  |  oak juniper
    4 |  0.769230769  |  0.074193548  |  pine oak

 * The Class Producer's Accuracy is equivalent to the Class Sensitivity, as
   defined by Fielding & Bell (1997)
** The Class User's Accuracy is equivalent to the Class Positive Predictive
   Power, as defined by Fielding & Bell (1997)

Fielding, Alan H. & John F. Bell.  1997.  A review of methods for the assessment
   of predictive errors in conservation presence/absence models. Environmental
   Conservation 24(1):38-49

Overall Statistics:
-------------------------------------------
             Overall Accuracy: (222 / 362) = 0.613259669
Overall Misclassification Rate: (140 / 362) = 0.386740331
          Overall Sensitivity:  0.613259669
          Overall Specificity:  0.871086556
        Overall Omission Error:  0.386740331
     Overall Commission Error:  0.128913444

KAPPA STATISTICS:
-----------------------------------------------
   KHAT   |   VARIANCE   |     Z     |     P
----------|--------------|-----------|----------
 0.431705 |  0.00123219  |    12.298 | < 0.00001
-----------------------------------------------

Analysis Began: September 12, 7:50:51 PM
Analysis Complete: September 12, 7:50:57 PM
Time Elapsed: 6 seconds...
```

The report is broken down into the following sections:

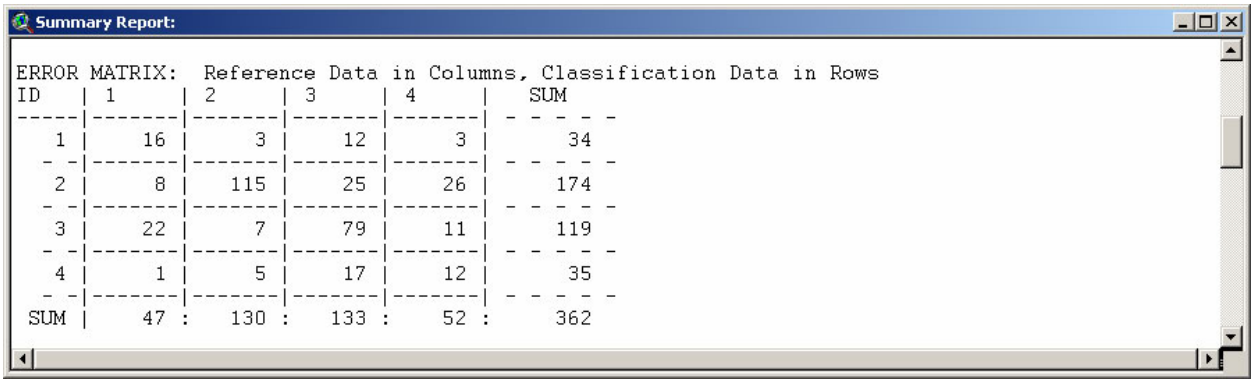
Header: Provides the time and date of the analysis, the hard-drive location of the text file, and a brief description of the data sources.



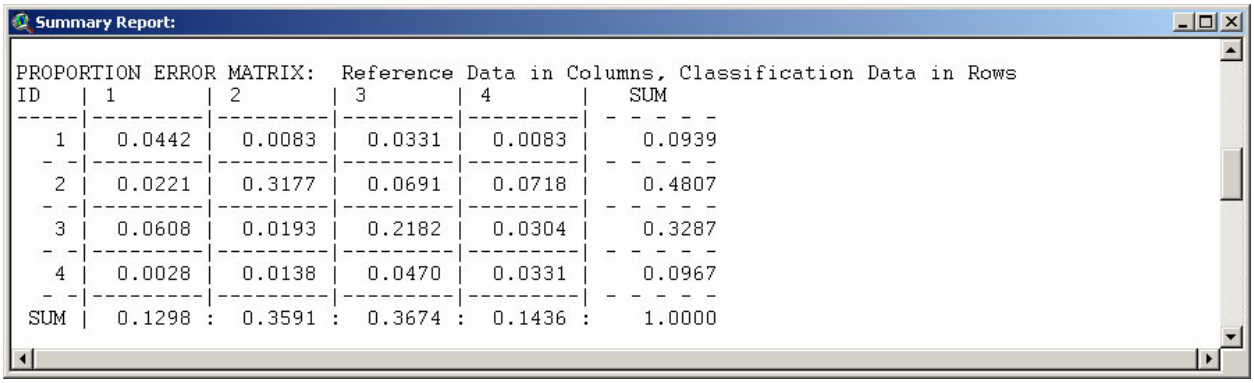
Legend: To simplify the report format, most of the report identifies the classification values by an index number. The legend specifies the classification type its corresponding index number.



Error matrix: This matrix shows the classification successes along the diagonal and the misclassifications in the upper and lower triangles. The reference values are in the columns and the classification values are in the rows. For example, the value "22" in the 1st column of the 3rd row should be interpreted to mean there were 22 cases where the model predicted those points to be class 3 (or "oak juniper" from the legend), but field observations identified them as class 1 (or "bareground/ gras"). Looking at the 3rd column of the 1st row, there were only 12 cases where this mistake occurred in the opposite direction.



Proportion Error Matrix: This is simply the error matrix rescaled into proportional values by dividing the value in each cell by the total sample size. This form of the error matrix is more useful for some purposes so the report includes both versions.

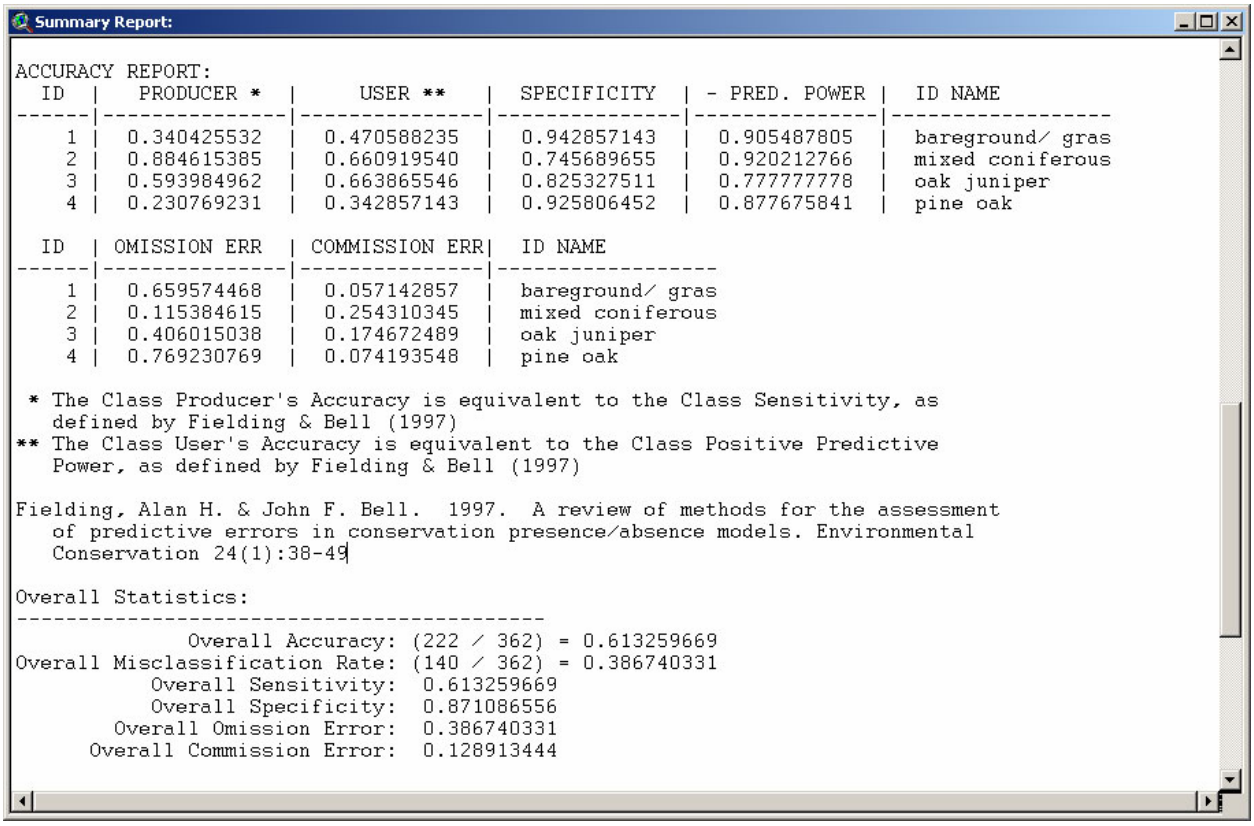


Summary Report:

PROPORTION ERROR MATRIX: Reference Data in Columns, Classification Data in Rows

ID	1	2	3	4	SUM
1	0.0442	0.0083	0.0331	0.0083	0.0939
2	0.0221	0.3177	0.0691	0.0718	0.4807
3	0.0608	0.0193	0.2182	0.0304	0.3287
4	0.0028	0.0138	0.0470	0.0331	0.0967
SUM	0.1298	0.3591	0.3674	0.1436	1.0000

Accuracy Report: The accuracy report summarizes the producer's accuracy, user's accuracy, sensitivity and the specificity of each class, as well as the overall accuracy, sensitivity, specificity, commission and omission error of the model (see pages 15 – 18).



Summary Report:

ACCURACY REPORT:

ID	PRODUCER *	USER **	SPECIFICITY	- PRED. POWER	ID NAME
1	0.340425532	0.470588235	0.942857143	0.905487805	bareground/ gras
2	0.884615385	0.660919540	0.745689655	0.920212766	mixed coniferous
3	0.593984962	0.663865546	0.825327511	0.777777778	oak juniper
4	0.230769231	0.342857143	0.925806452	0.877675841	pine oak

ID	OMISSION ERR	COMMISSION ERR	ID NAME
1	0.659574468	0.057142857	bareground/ gras
2	0.115384615	0.254310345	mixed coniferous
3	0.406015038	0.174672489	oak juniper
4	0.769230769	0.074193548	pine oak

* The Class Producer's Accuracy is equivalent to the Class Sensitivity, as defined by Fielding & Bell (1997)
** The Class User's Accuracy is equivalent to the Class Positive Predictive Power, as defined by Fielding & Bell (1997)

Fielding, Alan H. & John F. Bell. 1997. A review of methods for the assessment of predictive errors in conservation presence/absence models. Environmental Conservation 24(1):38-49

Overall Statistics:

Overall Accuracy: (222 / 362) = 0.613259669
Overall Misclassification Rate: (140 / 362) = 0.386740331
Overall Sensitivity: 0.613259669
Overall Specificity: 0.871086556
Overall Omission Error: 0.386740331
Overall Commission Error: 0.128913444

Category Statistics:

- Producer's Accuracy: The proportion of sample points correctly classified into class *X* divided by the number of points observed to be class *X*, and reflects the accuracy of the model from the perspective of the model (see Congalton and Green 1999:46 for an in-depth discussion). This value is equivalent to model Sensitivity.

- User's Accuracy: The proportion of sample points correctly classified into class *X* divided by the total number of points predicted to be class *X*, and reflects the accuracy of the model from the perspective of the model user (Congalton and Green 1999:46).
- Sensitivity: The probability that a sample point will be classified as *X* if it actually is *X*. This is conceptually similar and computationally identical to the concept of "Producer's Accuracy" (see Fielding and Bell 1997; Lurz et al 2001).
- Specificity: The probability that a sample point will not be classified as *X* if it is not *X* (Fielding and Bell 1997; Lurz et al 2001).
- Positive Predictive Power: The probability that a sample point is *X* if it has been classified as *X* (Fielding and Bell 1997; Lurz et al 2001). This is conceptually similar and computationally identical to the concept of user's accuracy.
- Negative Predictive Power: The probability a sample point is not *X* if it is not classified as *X* (Fielding and Bell 1997; Lurz et al 2001).
- Omission Error: The proportion of points incorrectly classified as not *X* when actually observed to be *X*. This is similar in concept to Type II statistical error, and may also be represented as:

  Omission Error = 1 – Sensitivity

- Commission Error: The proportion of points incorrectly classified as *X* when actually observed to not be *X*. This is similar in concept to Type I statistical error, and may also be represented as:

  Commission Error = 1 – Specificity

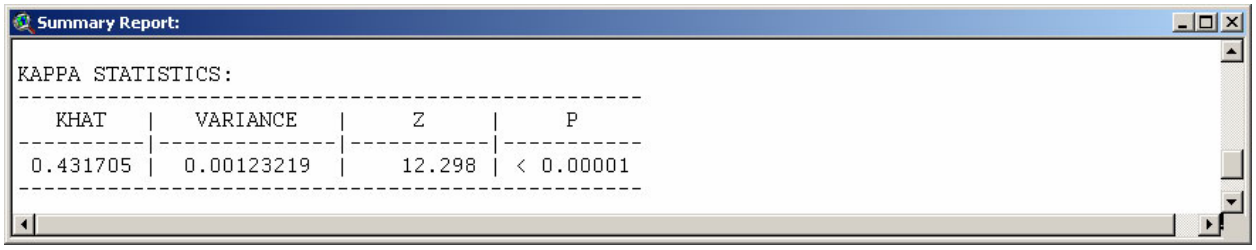
Overall Statistics:

- Overall Accuracy: This is simply the number of correctly-classified sample points divided by the total number of sample points.
- Overall Misclassification Rate: The number of incorrectly-classified sample points divided by the total number of sample points. This is the complement to overall accuracy, and may be represented as:

  Overall Misclassification Rate = 1 – Overall Accuracy

- Overall Sensitivity: The general ability of the model to classify sample points as *X* if they are actually *X*, identical to the Overall Accuracy.
- Overall Specificity: The general ability of the model to avoid misclassifying sample points as *X* if they are not *X*..
- Overall Omission Error: The general rate at which points failed to be classified into the correct category, combined over all categories, and the complement to sensitivity (see page 18).
- Overall Commission Error: The general rate at which points were misclassified, combined over all categories, and the complement to general specificity (see page 18).

Kappa Statistic: K_{HAT} (or $\hat{K}$) is the chance-corrected measure of model accuracy, based on the actual agreement between predicted and observed values and the chance agreement between the row and column totals for each classification (see Congalton and Green 1999:49 and Agresti 1990:366-367). The Z-score and associated *P*-value reflect the probability that this model performs better than random chance at predicting the distribution of vegetation classes on the landscape.



In this case, the *P*-value < 0.00001 suggests the model almost certainly predicts vegetation distribution better than random chance.

Calculating Confidence Intervals for $\hat{K}$

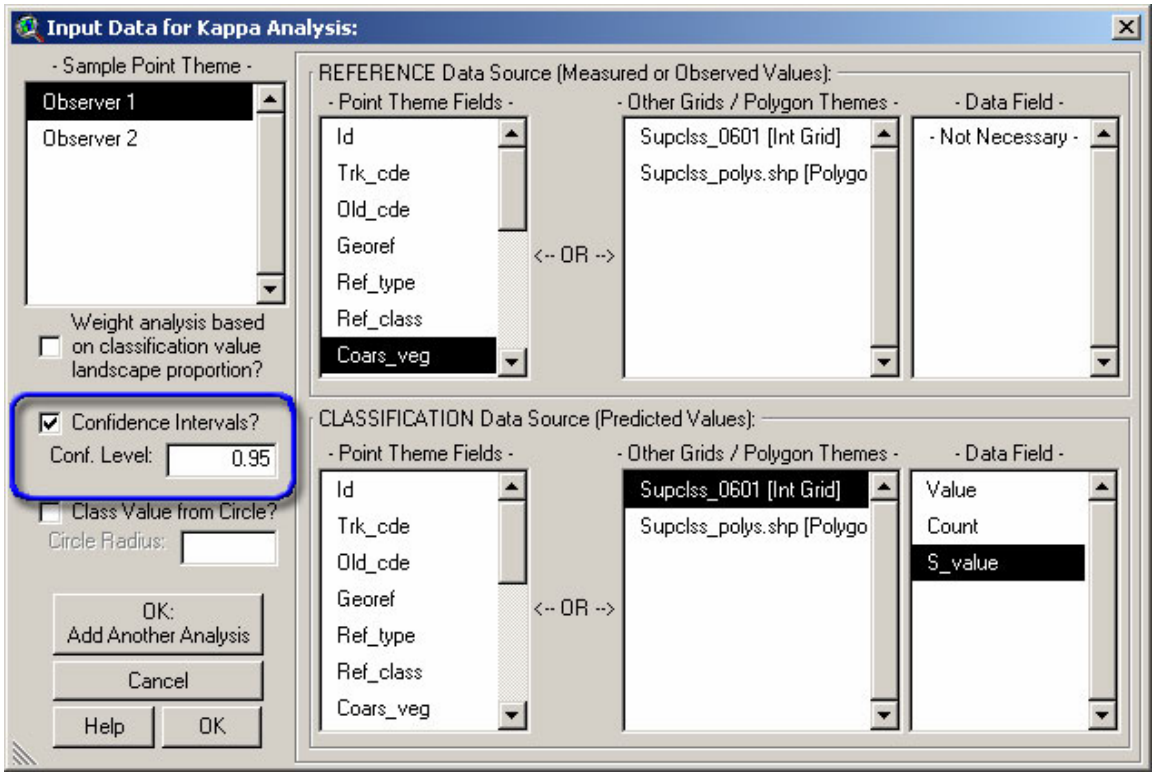
Using $\hat{K}$ and the associated variance, the confidence interval is calculated as:

$$CI = \hat{K} \pm Z_{\alpha/2}\sqrt{V(\hat{K})}$$

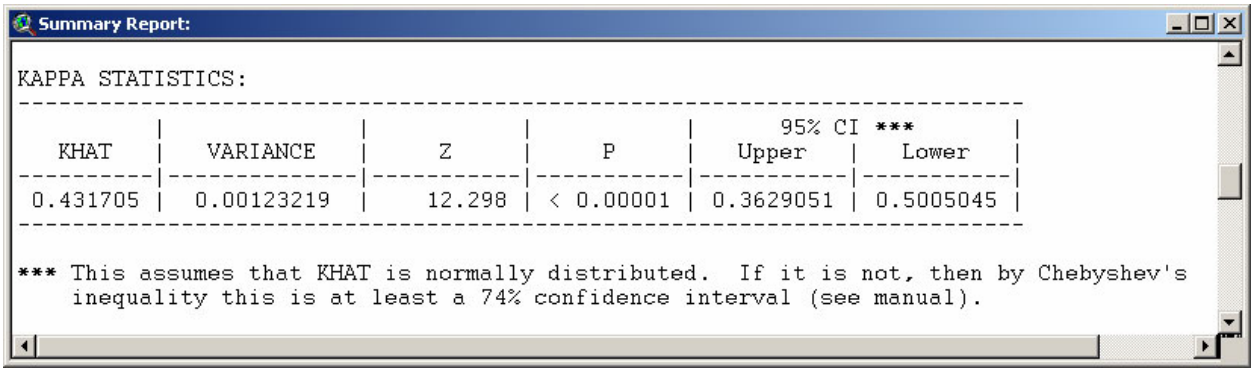
where $\alpha = 1 -$ confidence level

The $\hat{K}$ statistic for large samples is asymptotically normally distributed and therefore this confidence interval should be accurate. However, if the distribution is not normal, then Chebyshev's Inequality is used to determine the worst-case confidence level (see p. 34).

To calculate a confidence interval for $\hat{K}$, click the "Generate Confidence Intervals" option in the Kappa dialog:

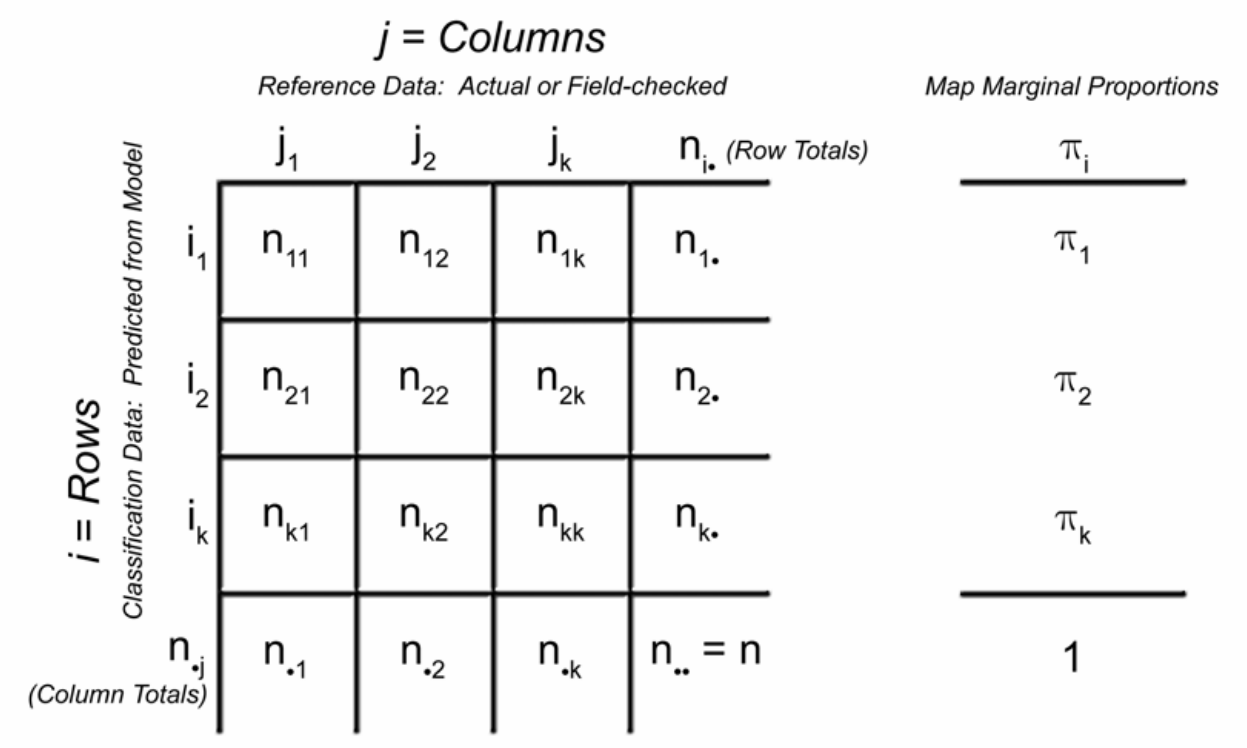


The report will now include the confidence interval in the Kappa Statistic section, with a note regarding the Chebyshev adjustment:



Calculating Variance and Confidence Intervals for Each Class

According to Congalton and Green (1999:59-63), it is possible to estimate variance and confidence intervals for overall accuracy, and user's and producer's accuracies within each category, and Card (1982) describes equations suitable for simple and stratified random sampling designs. Each method requires specification of the exact proportions π of the map which fall into each classification category, and the error matrix is then weighted by the relative proportions π_i:



In this case, the map-proportion-adjusted cell probabilities are estimated as follows:

Where π_i = map proportion of classification *i*:

$$\textit{Individual Cell Probability} = \hat{p}_{ij} = \frac{\pi_i n_{ij}}{n_{i\bullet}}$$

$$\text{Marginal proportions (column totals)} = \hat{p}_j = \sum_{i=1}^{r} \frac{\pi_i n_{ij}}{n_{\bullet j}}$$

Statistics describing overall accuracy are estimated as:

$$\textit{Overall Accuracy} = \hat{P}_c = \sum_{i=1}^{r} \frac{\pi_i n_{ii}}{n_{\cdot i}}$$

$$\textit{Variance (Simple Random Sampling):}\quad V\left(\hat{P}_c\right) = \sum_{i=1}^{r} \frac{p_{ii}\left(\pi_i - p_{ii}\right)}{\left(\pi_i n\right)}$$

$$\textit{Variance (Stratified Random Sampling):}\quad V\left(\hat{P}_c\right) = \sum_{i=1}^{r} \frac{p_{ii}\left(\pi_i - p_{ii}\right)}{n_{i\cdot}}$$

$$\textit{Confidence Interval:}\quad CI = \hat{P}_c \pm Z_{\alpha/2}\sqrt{V\left(\hat{P}_c\right)}$$

Statistics describing producer's accuracy for category *j* are estimated as:

$$\textit{Producer's Accuracy:}\quad \hat{\theta}_{jj} = \left(\frac{\pi_j}{\hat{p}_j}\right)\left(\frac{n_{jj}}{n_{\cdot j}}\right) = \frac{\hat{p}_{jj}}{\hat{p}_j}$$

$$\textit{Variance (Simple Random Sampling):}\quad V\left(\hat{\theta}_{jj}\right) = p_{jj}p_{\cdot j}^{-4}\left(p_{jj}\sum_{i\neq j}^{r}\frac{p_{ij}\left(\pi_i - p_{ij}\right)}{\pi_i n} + \frac{\left(\pi_j - p_{jj}\right)\left(p_{\cdot j} - p_{jj}\right)^2}{\pi_j n}\right)$$

$$\textit{Variance (Stratified Random Sampling):}\quad V\left(\hat{\theta}_{jj}\right) = p_{jj}p_{\cdot j}^{-4}\left(p_{jj}\sum_{i\neq j}^{r}\frac{p_{ij}\left(\pi_i - p_{ij}\right)}{n_{i\cdot}} + \frac{\left(\pi_j - p_{jj}\right)\left(p_{\cdot j} - p_{jj}\right)^2}{n_{\cdot j}}\right)$$

$$\text{Confidence Interval:}\quad CI = \hat{\theta}_{jj} \pm Z_{\alpha/2}\sqrt{V\left(\hat{\theta}_{jj}\right)}$$

Statistics describing user's accuracy for category *i* are estimated as:

$$\textit{User's Accuracy:}\quad \hat{\vartheta}_{ii} = \frac{n_{ii}}{n_{i\cdot}}$$

$$\textit{Variance (Simple Random Sampling):}\quad V\left(\hat{\vartheta}_{ii}\right) = \frac{p_{ii}\left(\pi_i - p_{ii}\right)}{\pi_i^{\,3} n}$$

$$\textit{Variance (Stratified Random Sampling):}\quad V\left(\hat{\vartheta}_{ii}\right) = \frac{p_{ii}\left(\pi_i - p_{ii}\right)}{\pi_i^{\,2} n_{i\cdot}}$$

$$\textit{Confidence Interval:}\quad CI = \hat{\vartheta}_{ii} \pm Z_{\alpha/2}\sqrt{V\left(\hat{\vartheta}_{ii}\right)}$$

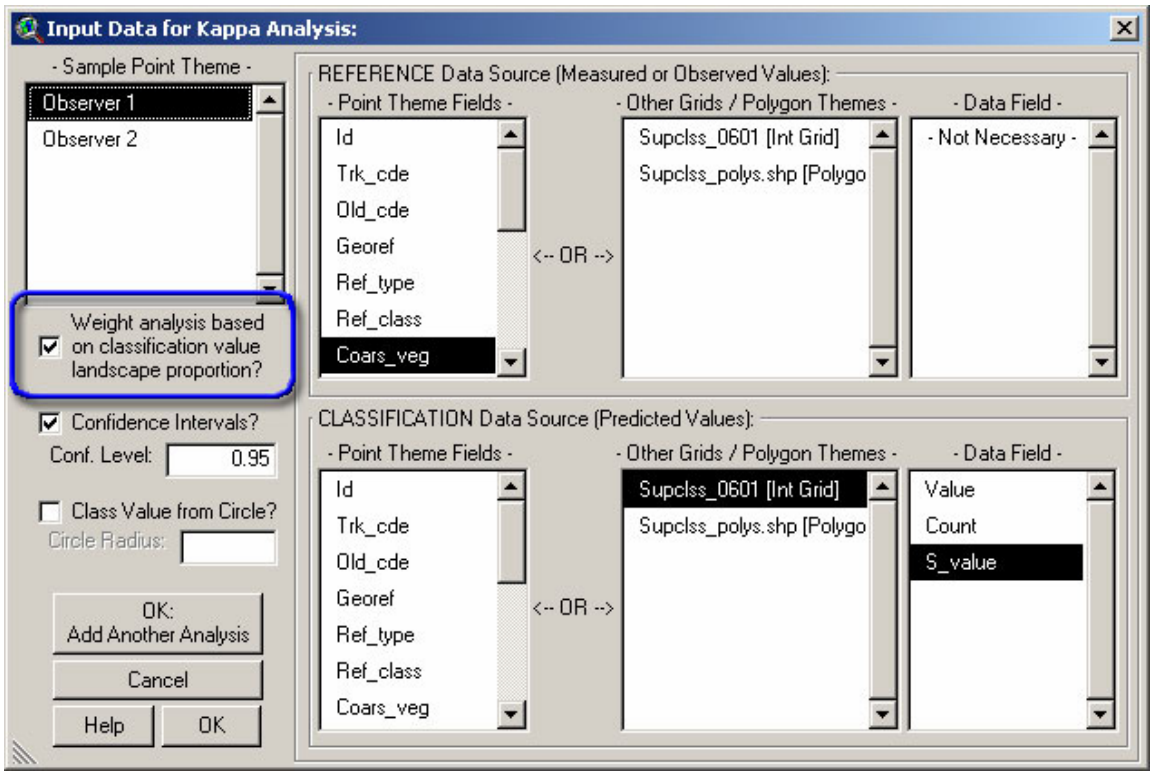
IMPORTANT: Congalton and Green (1999:63) discuss the confidence level of these intervals, pointing out these are accurate only if the statistics are normally distributed. Otherwise, by Chebyshev's Inequality (see p. 34) they are at least $\left(1 - \frac{1}{\left(Z_{\alpha/2}\right)^2}\right)$ confidence intervals.

These formulas are adapted from Congalton and Green (1999:63) and Card (1982).

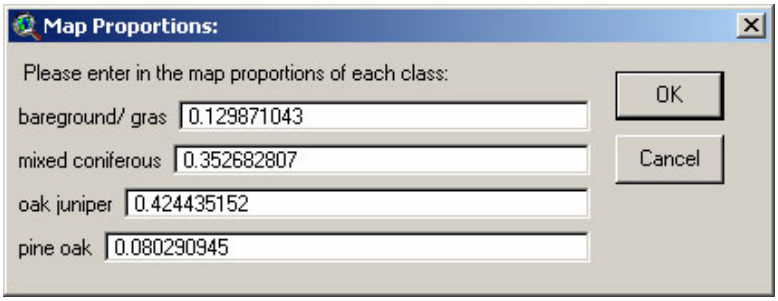
Using This Extension to Calculate Variances and Confidence Intervals

Variances and confidence intervals for each category require the marginal map proportions of each category, and this extension offers a means to specify map proportions for each classification value.

Simply check the option “Weight analysis based on classification value landscape proportion?” before clicking the “OK” button:

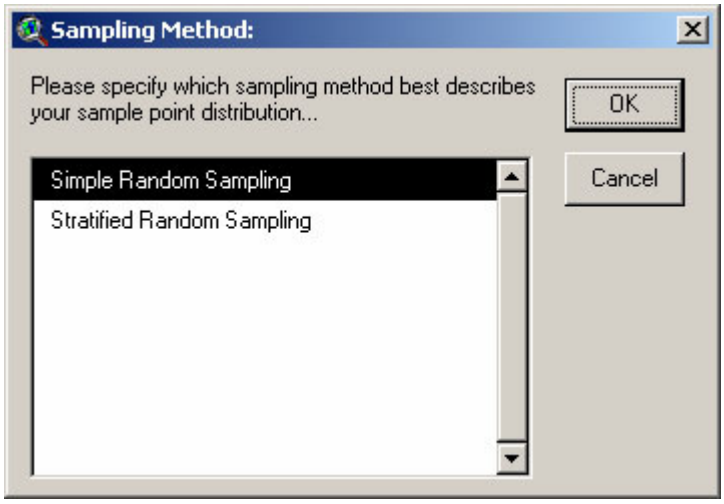


Notice that confidence intervals are optional. If you want confidence intervals for the overall and category accuracies, select the “Generate Confidence Intervals” option and specify your desired confidence level. After clicking “OK,” you will be prompted to specify the map proportions:

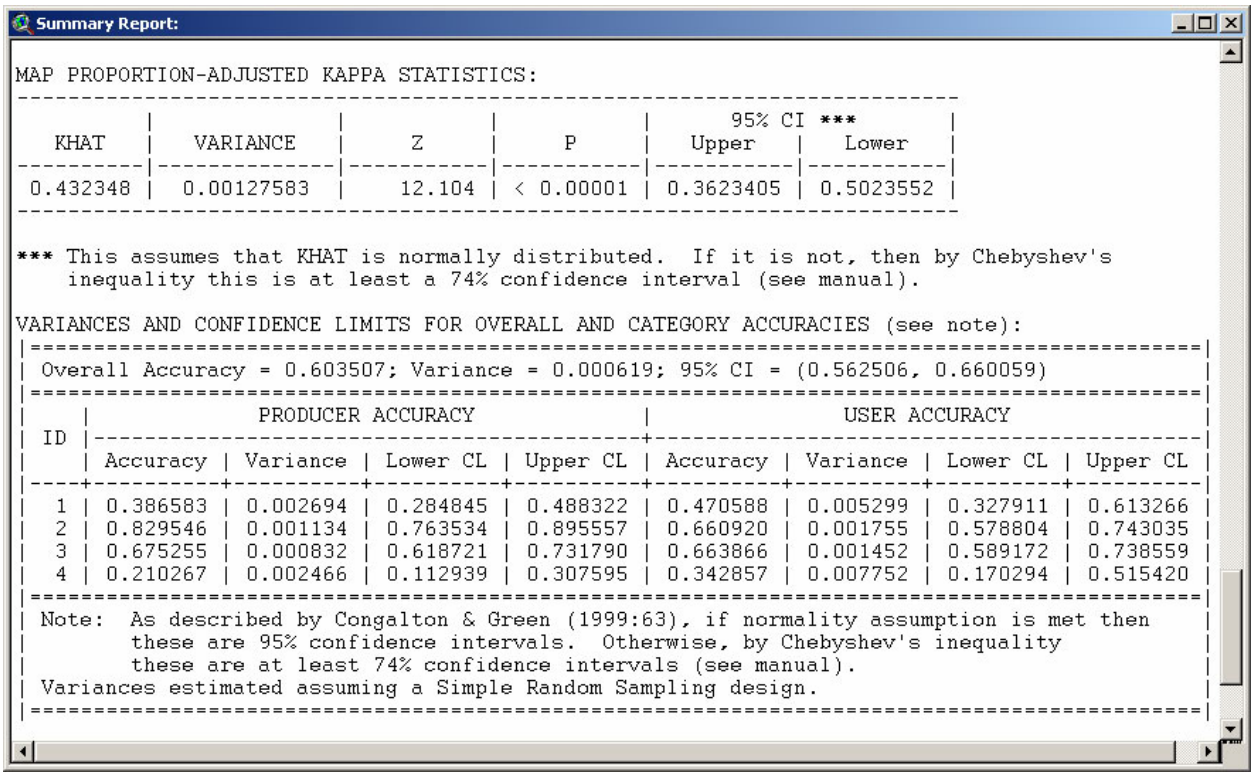


This extension will analyze your classification data and calculate the map proportions for you if your classification data source is either a polygon theme or a grid, and those values are entered into the dialog box automatically. You may either modify these values or simply click the “OK” button to accept. If your classification data source is a field in the point theme, this extension will estimate the relative map proportions based on the relative proportions of the classification data values.

You will next be asked which sampling method is closest to the method you used. The variance equations are slightly different depending on whether your sample points follow a simple or stratified random distribution (see equations above). The equations for stratified random sampling assume that you sampled rarely-occuring categories more often than you would have if you had followed a simple random sampling design, and therefore the variances will tend to be lower than with simple random sampling. Variances for commonly-occuring landscape types tend to be larger than with simple random sampling, though.



After the analysis is complete, the variances and confidence intervals will be added to the report:

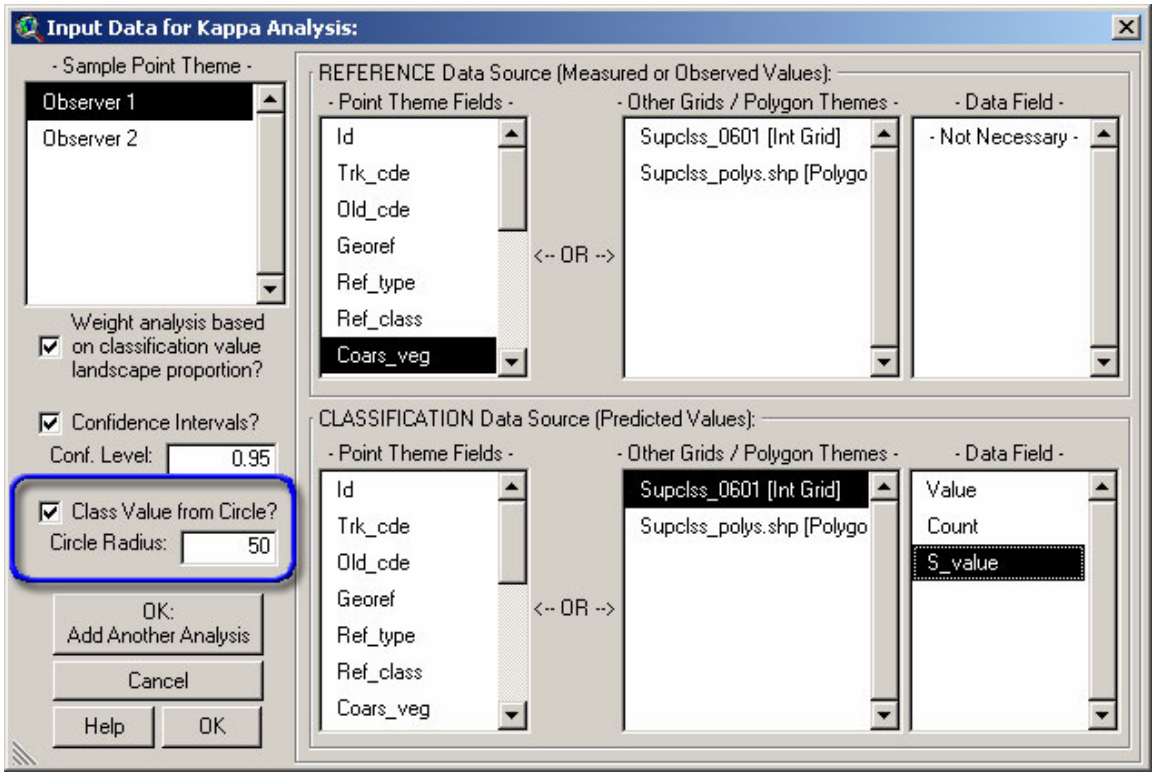


Adding Class Values from a Circular Area

If you feel that there is some uncertainty about the location accuracy of your sample points, and would therefore rather derive your classification values from the majority class value in a circular area around your sample point rather than from the exact class value at that point (as described in *Adjusting for Locational Uncertainty* on p. 13) , then click the “Class Value in Circle” option and enter a radius in the map units of your data.

Please note that this function can only apply if either your reference or class values are extracted from a grid or polygon theme. This function will have no effect if all values are extracted from fields in the point attribute table.

This extension runs much faster if both classification and reference values are extracted from fields in the attribute table. If you would like to add fields to your attribute table containing values extracted from circular areas, please refer to the discussions on *Generating Class Values* beginning on p. 73.



Comparing Different Analyses

Kappa statistics and associated variance values may be used to compare the predictive ability of different models, possibly derived from different predictive algorithms or datasets or even between different dates of imagery (Congalton and Mead 1983). Assuming Kappa statistics follow the standard normal distribution, the Z-score for each statistic is calculated as:

$$Z = \frac{\hat{K}_1}{\sqrt{\hat{\mathrm{var}}\left(\hat{K}_1\right)}}$$

In this case, H_0: $K_1 = 0$

H_1: $K_1 \neq 0$

The Z-score reflecting the difference between two separate analyses is calculated as:

$$Z = \frac{\left|\hat{K}_1 - \hat{K}_2\right|}{\sqrt{\hat{\mathrm{var}}\left(\hat{K}_1\right) + \hat{\mathrm{var}}\left(\hat{K}_2\right)}}$$

In this case, H_0: $(K_1 - K_2) = 0$

H_1: $(K_1 - K_2) \neq 0$

Assuming a 2-sided test (i.e., K_1 is different than K_2), then H_0 would be rejected if $Z \geq Z_{\alpha/2}$.

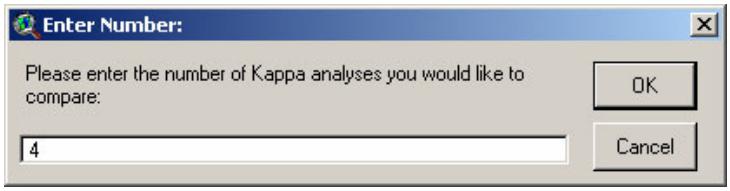
Multiple $\hat{K}$ values by can be tested for equal values by first estimating the supposed 'common' $\hat{K}$ value (as described by Fleiss [1981:222]):

$$\text{'Common' } \hat{K} = \frac{\sum_{m=1}^{g} \frac{\hat{K}_m}{Var\left(\hat{K}_m\right)}}{\sum_{m=1}^{g} \frac{1}{Var\left(\hat{K}_m\right)}}$$

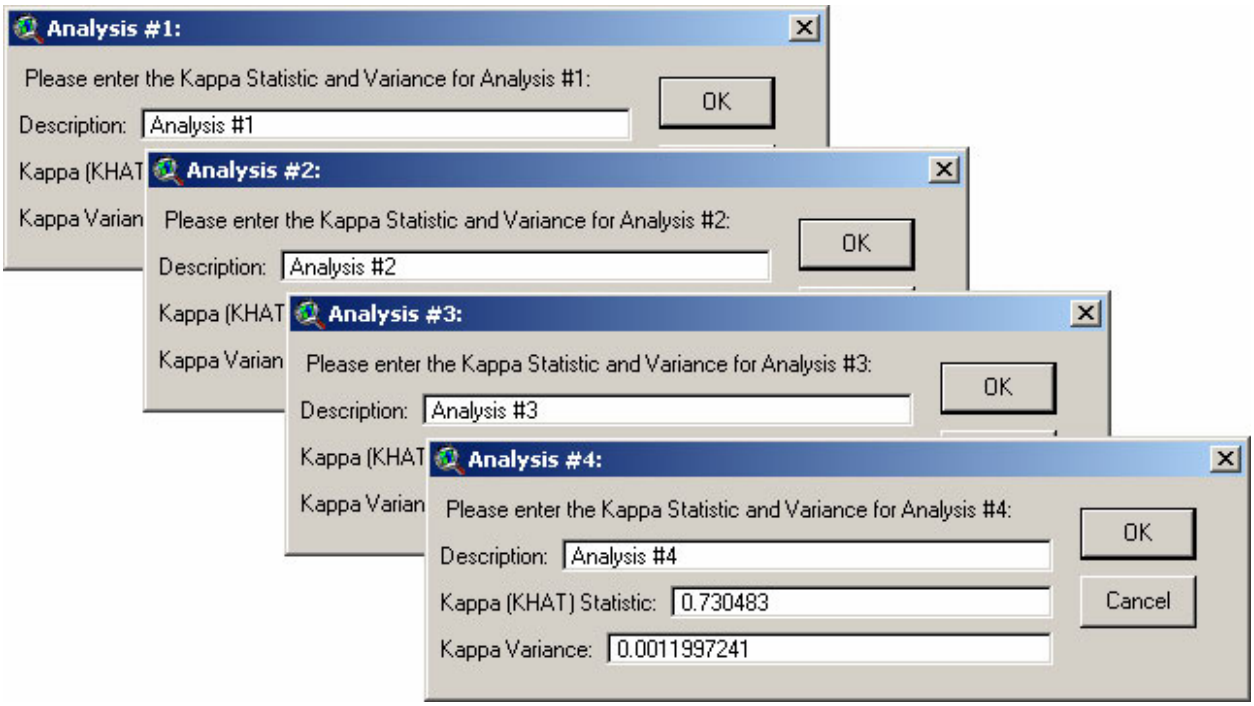
This 'common' $\hat{K}$ value can then be used to test for equal $\hat{K}$ values on the Chi-Square distribution with *g - 1* degrees of freedom:

$$\chi^2_{equal\ \hat{K}} = \sum_{m=1}^{g} \frac{\left(\hat{K}_m - \hat{K}_{'Common'}\right)^2}{Var\left(\hat{K}_m\right)}$$

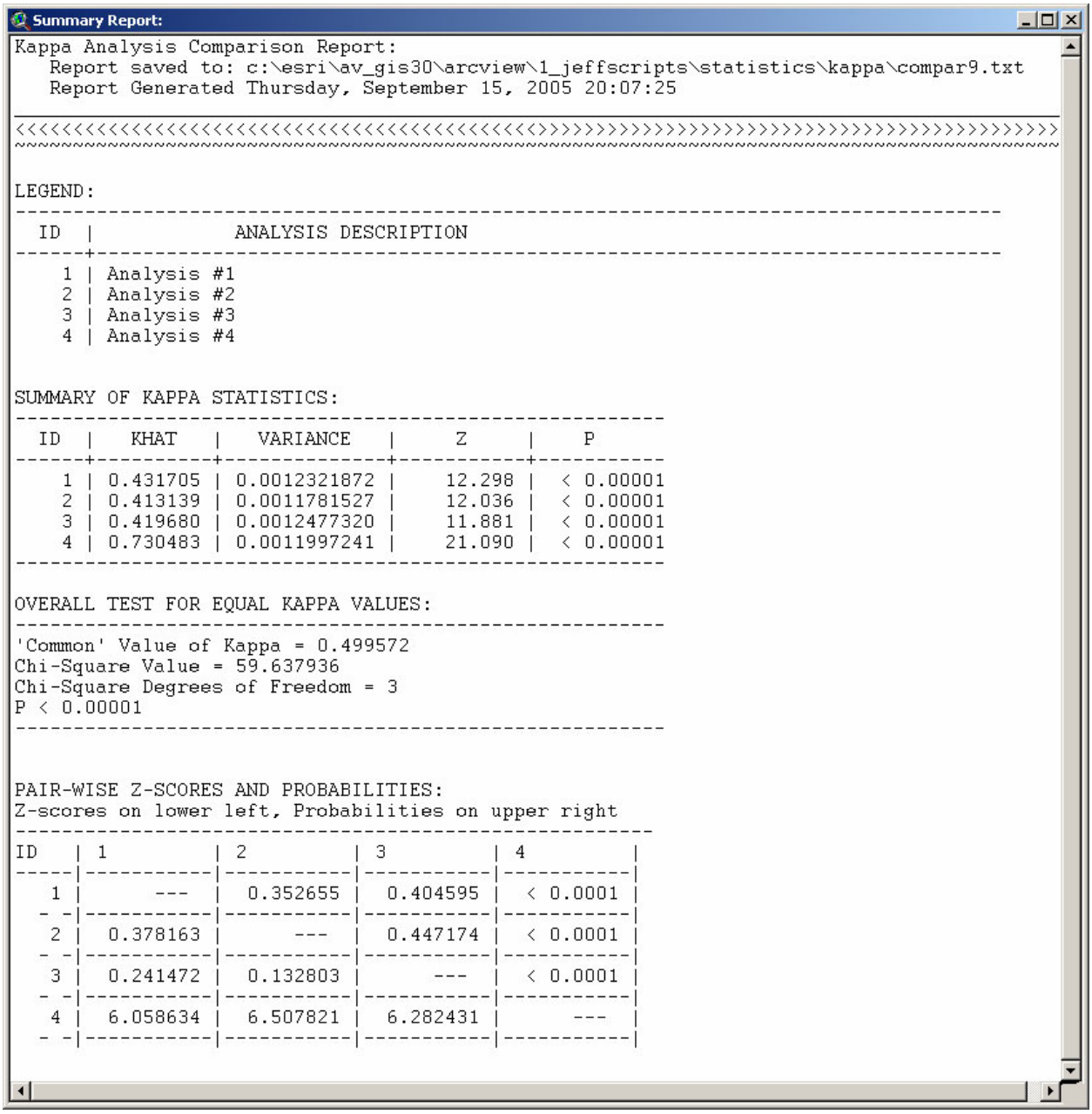
This extension provides two methods to compare different analyses. If you already have your $\hat{K}$ and Variance statistics worked out, you can use the  tool to calculate the comparisons directly. Click this button and you will first be asked for the number of Kappa analyses to compare:



You will then be asked to describe each analysis separately. For each analysis, enter a brief description (used for identification purposes in the output report), the $\hat{K}$ and the Variance values:



Your output report will be saved to the hard drive and displayed in a report window:



Summary Report:

```
Kappa Analysis Comparison Report:
   Report saved to: c:\esri\av_gis30\arcview\1_jeffscripts\statistics\kappa\compar9.txt
   Report Generated Thursday, September 15, 2005 20:07:25
```

LEGEND:

ID	ANALYSIS DESCRIPTION
1	Analysis #1
2	Analysis #2
3	Analysis #3
4	Analysis #4

SUMMARY OF KAPPA STATISTICS:

ID	KHAT	VARIANCE	Z	P
1	0.431705	0.0012321872	12.298	< 0.00001
2	0.413139	0.0011781527	12.036	< 0.00001
3	0.419680	0.0012477320	11.881	< 0.00001
4	0.730483	0.0011997241	21.090	< 0.00001

OVERALL TEST FOR EQUAL KAPPA VALUES:

```
'Common' Value of Kappa = 0.499572
Chi-Square Value = 59.637936
Chi-Square Degrees of Freedom = 3
P < 0.00001
```

PAIR-WISE Z-SCORES AND PROBABILITIES:

Z-scores on lower left, Probabilities on upper right

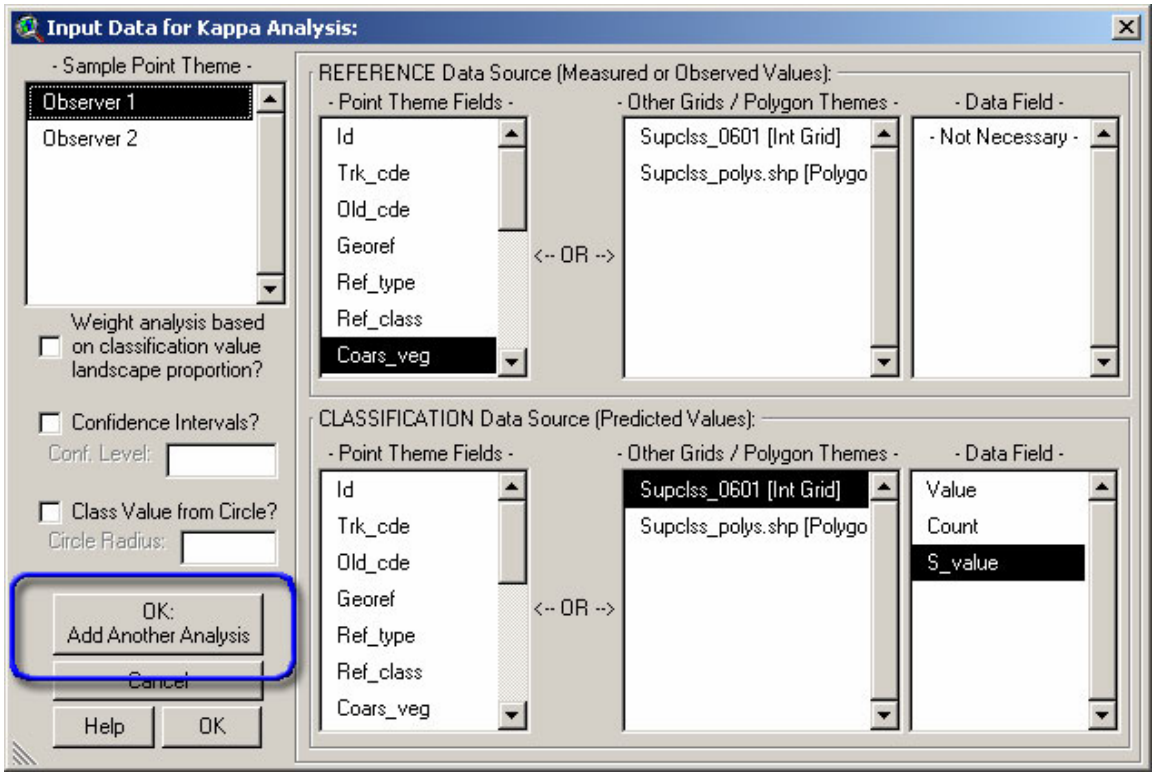
ID	1	2	3	4
1	---	0.352655	0.404595	< 0.0001
2	0.378163	---	0.447174	< 0.0001
3	0.241472	0.132803	---	< 0.0001
4	6.058634	6.507821	6.282431	---

Note that in this example, all methods produced results significantly better than chance, but analysis # 4 was very different than analyses # 1 – 3. The methods or personnel involved in analyses # 1 – 3 apparently produced generally equivalent output, but whatever was involved in analysis # 4 is in a class by itself.

Alternatively, you can select to compare Kappa analyses when you initially generate the Kappa statistics. If you elect to run multiple Kappa analyses simultaneously, then the resulting Kappa statistics will automatically be compared.

For example, if you were unsure whether Observer 1 and Observer 2 were equally competent to conduct field surveys, you could have them survey the same points and compare their respective $\hat{K}$ values.

You can run multiple Kappa analyses by clicking the “OK – Add Another Analysis” button in the initial Input Data window:



Continue to click the "OK: Add Another Analysis" button until you have entered all analyses you wish to compare. Then, click the "OK" button. The extension will automatically generate comparison Z-scores and *P*-values for each pair-wise comparison and add these analyses to the output report.

Chebyshev's Inequality:

Calculating accurate confidence intervals requires the population distribution be known, which is usually not a problem in cases where the nature of the data leads it to follow a particular distribution (such as how sample means from a particular population tend to be normally distributed because of the Central Limit Theorem). A confidence interval will not be correct if the population does not exactly follow the expected distribution.

Chebyshev's Inequality is a probability theorem which can be used to determine the minimum probability that a sample value will be within *k* standard deviations from the population mean, regardless of the shape of the distribution:

$$P\left(|X-\mu| \geq k\sigma\right) \leq \frac{1}{k^2} \qquad \text{or} \qquad P\left(|X-\mu| < k\sigma\right) \geq \left(1-\frac{1}{k^2}\right)$$

where X = sample value

μ = population mean

σ = population standard deviation

k = number of standard deviations from the mean

Therefore, if a confidence interval for a particular statistic is calculated, Chebyshev's Inequality may be used to determine the worst-case confidence level given the specified confidence level. For example, if a 95% confidence interval for Kappa is desired, and the sample produced $\hat{K}$ = 0.43 and $Var\left(\hat{K}\right)$ = 0.012, then a 95% confidence interval would be calculated as:

$$\begin{aligned} \text{CI} &= \hat{K} \pm Z_{\alpha/2}\sqrt{Var\left(\hat{K}\right)} \\ &= 0.43 \pm 1.96(0.035) \\ &= (0.361,\ 0.499) \end{aligned}$$

If $\hat{K}$ is normally distributed, we may state that if we repeated this analysis an infinite number of times, then 95% of the time our confidence intervals would capture the true Kappa value. However, if $\hat{K}$ is not at all normally distributed, Chebyshev's inequality implies that we would capture the true Kappa value at least 74% of the time:

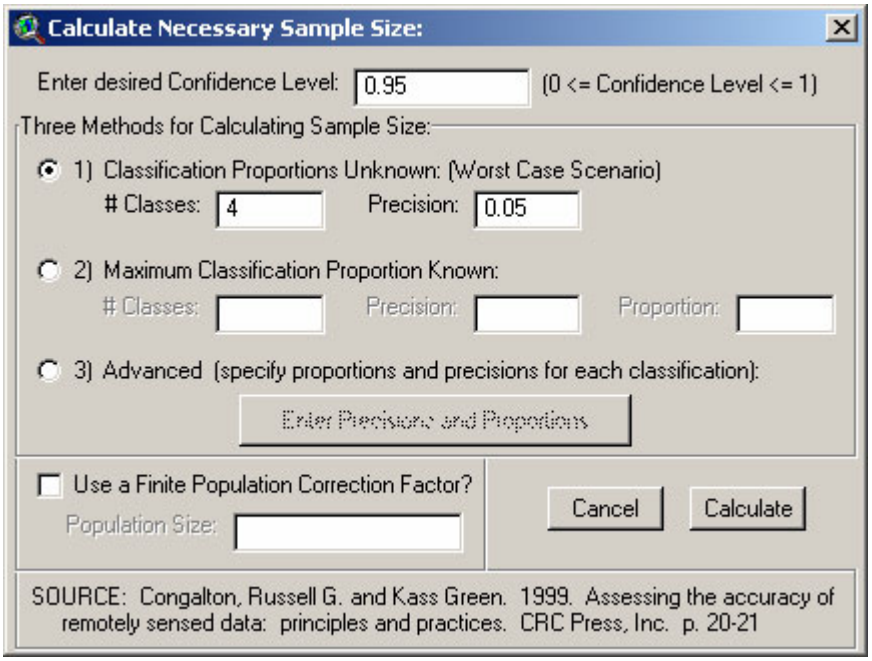
$$\left(1 - \frac{1}{\left(Z_{\alpha/2}\right)^2}\right) = \left(1 - \frac{1}{(1.96)^2}\right) = 0.74$$

See Spiegel et al (2000), Abramowitz and Stegun (1972) and Jeffrey (2000) for more in-depth discussions of Chebyshev's Inequality and its uses and implications.

Calculating Required Sample Size

Any good text on experimental design will stress the importance of sufficient sample size and of choosing the correct sampling scheme. Congalton and Green (1999:11-25) discuss sample design with a particular emphasis on classification accuracy assessment, and describe methods to estimate the necessary sample size for several possible scenarios. This extension provides a tool to automate those methods.

Click the  button to open the sample size estimation tool:



There are three options, each based on differing amounts of knowledge regarding the relative proportion of classification values in your classification dataset. The worst-case scenario will produce the highest sample size.

The confidence level reflects the probability the sample size will be sufficient to perform statistically valid accuracy assessments on the model. The precision reflects the acceptable risk when mistaken conclusions occur regarding the accuracy of a map based on the statistical analyses.

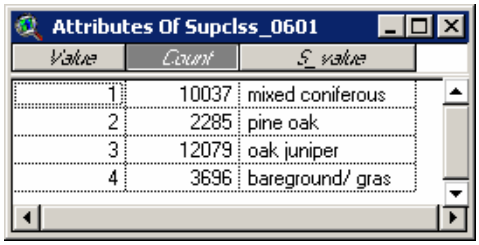
All three options can be calculated with or without using a Finite Population Correction factor (FPC). If the total number of possible sample points is infinite, then do not use the FPC option. However, if there are only a finite number of possible sample points available, you can use the finite population factor to properly estimate the necessary sample size.

Use Option 1 if you have no information regarding the relative map proportions of each classification. The worst case occurs if one of your classification proportions is exactly equal to 50%, and Option 1 will assume that case is true.

Use Option 2 if you know the proportion of the class with the highest proportion on the predicted landscape. In all cases, this proportion will be the one closest to 50% (i.e., the worst-case scenario), and this value will be used to adjust the estimated necessary sample size accordingly. If your classification dataset is grid-based, you can manually calculate the proportion values by doing the following:

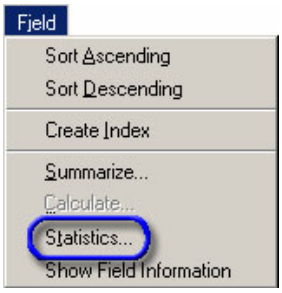
1) Make the classification grid active by clicking on the grid name in the View table of contents.

2) Open the grid attribute table by clicking the  button:

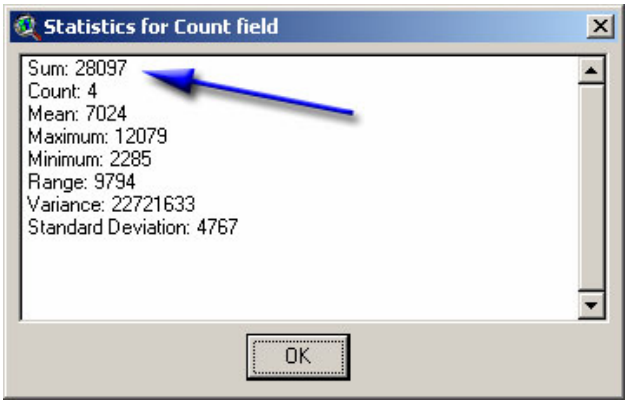


3) The *Count* field contains the number of cells in each category and can be used to calculate the proportions of each class. You need to add these up to get the sum, then divide the values by that sum to get the respective proportions.

4) Click the *Count* field name to select this field (see above). Then click the "Statistics…" menu item in the "Field" menu item:

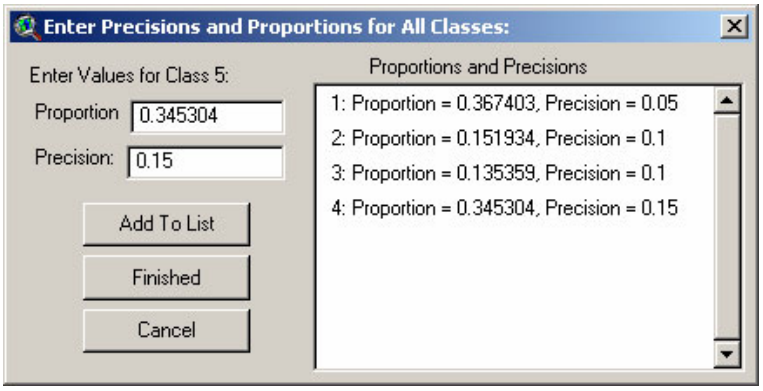


5) The output window will display several statistics. You only need the sum for this:

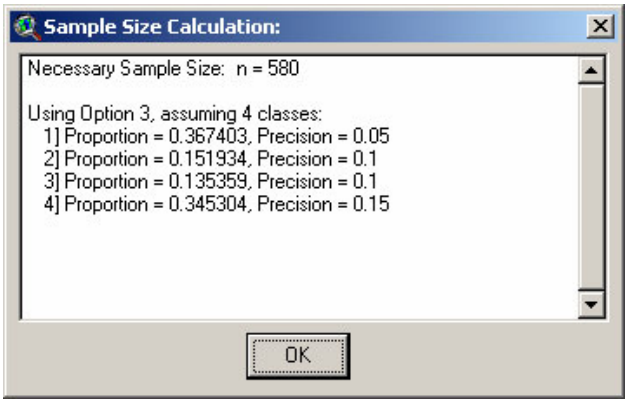


6) Next, divide each value by the sum (28,097 in this example) to get the relative proportions. Enter the largest proportion into the "Sample Size" dialog box.

Use "Option 3" if you know all the proportions and would like to enter separate precision levels for each class. For example, if you know all the proportions and if you were especially concerned about precision for "mixed coniferous" (Class 1), but not concerned about precision for "oak juniper" (Class 2), then you would select "Option 3" and click the "Enter Precisions and Proportions" button to specify these values:



Click the "Finished" button to return to the "Sample Size" dialog. Then, click "Calculate" to generate the estimated sample size:



For comparison purposes, the following table summarizes the estimated necessary sample size for a hypothetical grid with four classes, given a confidence level of 95% and a precision of 10%. For "Option 2," the maximum proportion is 36.7%. For "Option 3," the proportions and precision levels are identical to those in the illustration above:

Scenario	No Finite Population Factor	Population = 10,000	Population = 1,000
Option 1	n = 156	n = 154	n = 136
Option 2	n = 145	n = 143	n = 127
Option 3	n = 580	n = 549	n = 368

NOTE: Increasing precision for a given class (e.g., "mixed coniferous;" Class 1), increases sample size requirements.

Choosing Sample Locations:

Once you know the necessary sample size, you are then faced with the problem of how those sample points should be distributed. Statistically, the most representative method is a random point distribution

over the entire study area (*Simple Random Sampling*), but this can often be difficult or cost-prohibitive in difficult terrain. This method can also potentially under-sample rare classification types. Congalton and Green (1999:22-25) and Congalton (1988) discuss in depth the pros and cons of this and other common sample designs. The following is a brief overview which may be useful for the reader:

1) *Stratified Random Sampling*: Each class receives a set number of randomly-distributed sample points. This ensures that all classes will be sufficiently sampled, but this method may have the same problems as simple random sampling in difficult terrain.

2) *Cluster Sampling*: Sample points are grouped into clusters, where all pixels in that cluster are sampled. Congalton (1988) recommends that clusters should preferably be around 10 pixels in size, and never larger than 25 pixels. Kesh (1965: 148-181) provides a detailed discussion of issues involved with cluster sampling.

3) *Systematic Sampling*: The starting point is chosen at random, and all subsequent sampling points are taken at some regular interval. This method can be especially problematic if the landscape or model has some spatial periodicity.

4) *Stratified Systematic Unaligned Sampling*: A combination of random and systematic sampling, in which each sample point is randomly located within the stratification interval.

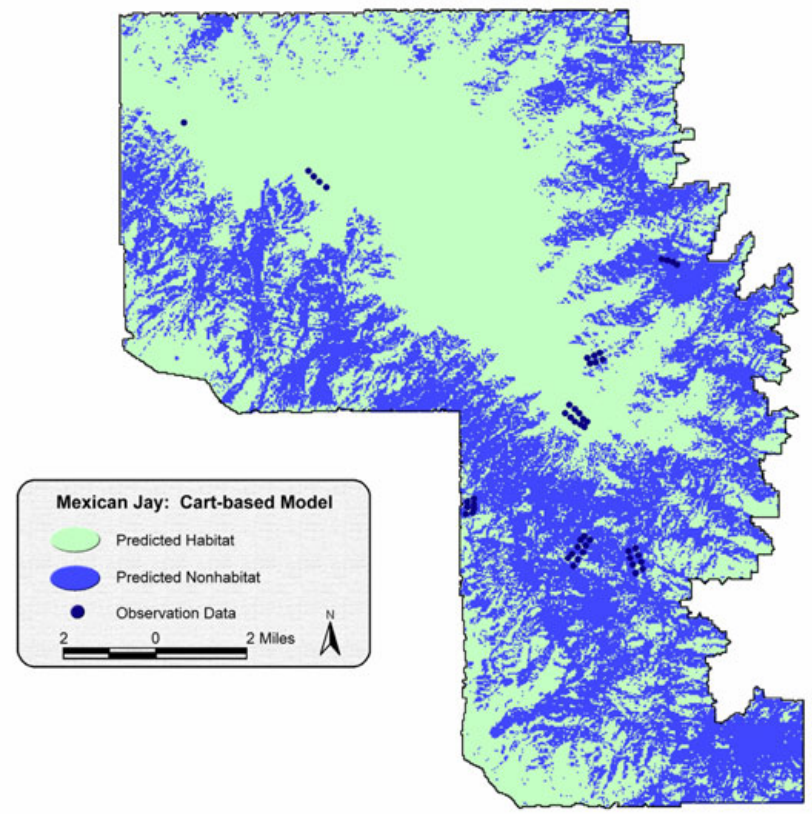
Case Study – Mexican Jay Distribution Surfaces, Pinaleños Mountains, Arizona

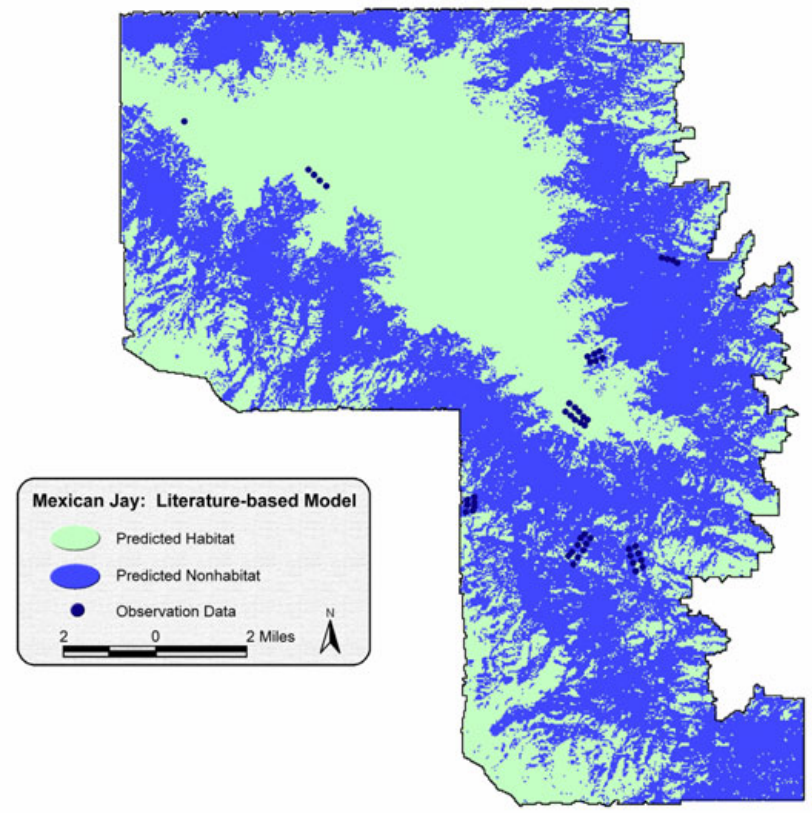
This case study was based on research conducted by Wynne (2003).

In the United States, Mexican jay (*Aphelocoma ultramarina*) occurs only in the southern-most extent of the southwestern United States. This species ranges from the Mogollon Rim extending southward into southeastern Arizona and the southeastern-most New Mexico (Edwards 1986) and in the Big Bend area of western Texas (Brewster County; Brown 1994). In the southwestern United States, this species has been intensively studied in the Gila and Burro Mountains, New Mexico (Edwards 1986) and Chiricahua Mountains, Arizona (Tanner and Hardy 1958; Balda 1970; Brown and Brown 1985; Brown 1994). From the U.S., Mexican jays range southward into Mexico to northern Chihuahua, northern Coahuila, central Nuevo León, west-central Tamaulipas south to the highlands of Colima, northern Michoacán, state of México, northern Morelos, Puebla and west-central Veracruz (Brown 1994). This case study is the first habitat investigation of this corvid on the Pinaleños Mountains, Arizona.

Two predictive habitat surface models will be discussed in this case study; a classification tree-based model and literature-based model. The classification tree -based model was developed using a 1993-95 retrospective dataset collected by Dr. William M. Block, USDA Forest Service, Rocky Mountain Research Station, Flagstaff. Literature-based information was derived from the Birds of North America species account (Brown 1994) and other peer-reviewed sources. Landscape scale habitat correlates used in model development include vegetation, elevation, slope, aspect and distance to streams and springs. The minimum mapping unit was 30 meters. Identification of the model with the best performance will be accomplished using a multi-criteria model selection, consisting of highest overall accuracy, lowest misclassification rate, highest $\hat{K}$ statistic, *P*-value (significant at ≤ 0.05), highest sensitivity and specificity, highest positive predictive power, and lowest commission and omission error rates.

Using an induced classification tree approach (refer to Wynne 2003), oak-juniper woodlands, elevation ≤ 1988 meters, and slope ≤ 24.5 were identified as habitat correlates. Results of the literature review identified oak-juniper woodlands, pine-oak woodlands (Brown 1994; Balda 1970), and elevation between 1460 to 2363 meters (Bailey 1923; Tanner and Hardy 1958; Brown and Brown 1985). See predictive habitat surfaces below.


Mexican Jay: Cart-based Model
Predicted Habitat
Predicted Nonhabitat
Observation Data
2 0 2 Miles
N


Mexican Jay: Literature-based Model
Predicted Habitat
Predicted Nonhabitat
Observation Data
2 0 2 Miles
N

A statistical comparison of the models suggests no significant difference exists between the literature-based and classification tree based models (Z=0.335, p = 0.369). Furthermore, neither model had statistically significant $\hat{K}$ *P*-values, implying neither model classified Mexican jay habitat better than random chance. However, using a multi-criteria model selection process, the classification tree-based model did perform better than the literature-based model (see Table below). The classification tree-based model had the highest overall accuracy (63.6%), lowest misclassification rate (0.364), and highest Kappa value ($\hat{K}$ = 0.22). Additionally, this model had highest specificity for predicting presence (0.636), highest model sensitivity for absence (0.636), highest predictive power for absence (0.84) and presence (0.368), lowest commission rates for presence (0.364) and lowest omission rates for absence (0.364).

Metric	Literature-based	Classification-tree based
Overall Accuracy	56.8	**63.6**
Misclassification Rate	43.18	**36.36**
$\hat{K}$	0.136	0.22
P-value	0.229	0.09
Sensitivity (Absence)	0.545	**0.636**
Sensitivity (Presence)	0.636	0.636
Specificity (Absence)	0.636	0.636
Specificity (Presence)	0.545	**0.636**
Positive Predictive Power (Absent)	0.818	**0.84**
Positive Predictive Power (Presence)	0.318	**0.368**
Commission Error (Absence)	0.364	0.364
Commission Error (Presence)	0.455	**0.364**
Omission Error (Absence)	0.455	**0.364**
Omission Error (Presence)	0.364	0.364

Bold text suggests better performance by model.

This case study provides a tractable method for identifying the model with the highest potential performance when statistical significance is lacking between models.

For additional examples regarding the use of the Cohen's Kappa and other statistics, please refer to Farber and Kadmon (2003) and Fielding (2002).

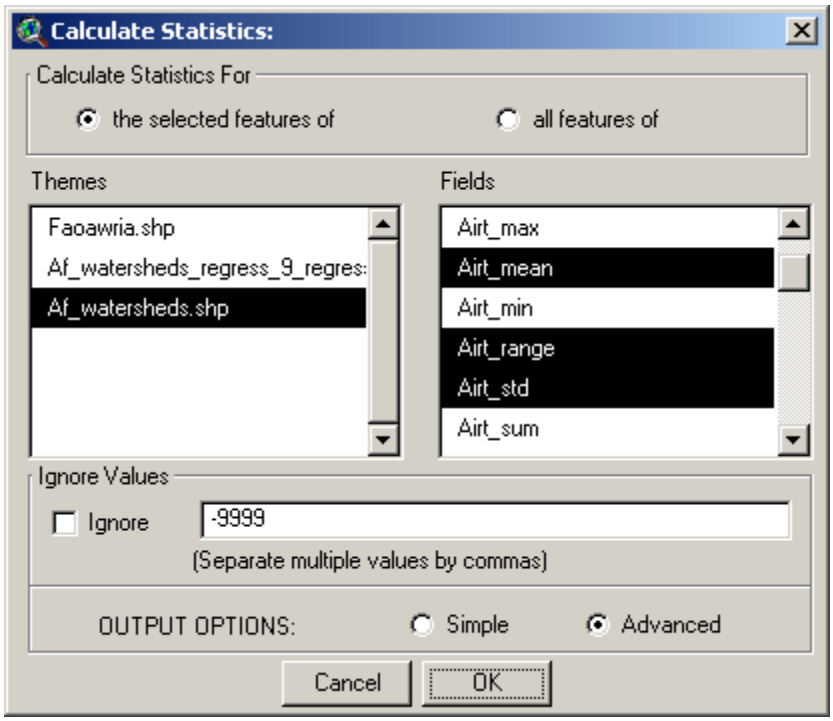
If you want to inspire confidence, give plenty of statistics. It does not matter that they should be accurate, or even intelligible, as long as there is enough of them.
Lewis Carroll

Field Summary Statistics:

This tool provides functions similar to those available in the basic ArcView "Statistics…" options under the standard "Field" menu item in the Table menu, with the exception that there are both more options and a higher level of precision used for any calculations. The tool may be used to generate statistics on either a theme in a view or a field in a table.

Summary Statistics on a Theme:

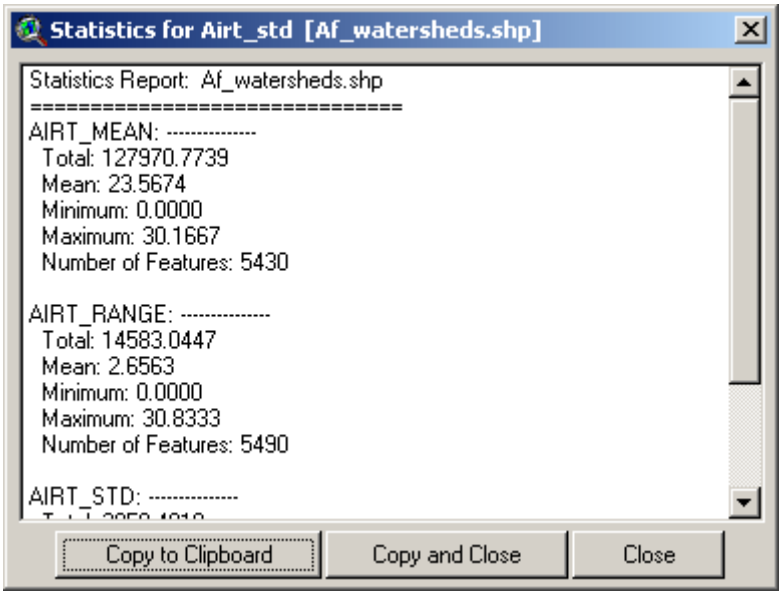
The  button will only be enabled if the user has at least one feature theme in the current View. When the user clicks the button, they will be prompted to identify the theme and/or fields for calculating the statistics.



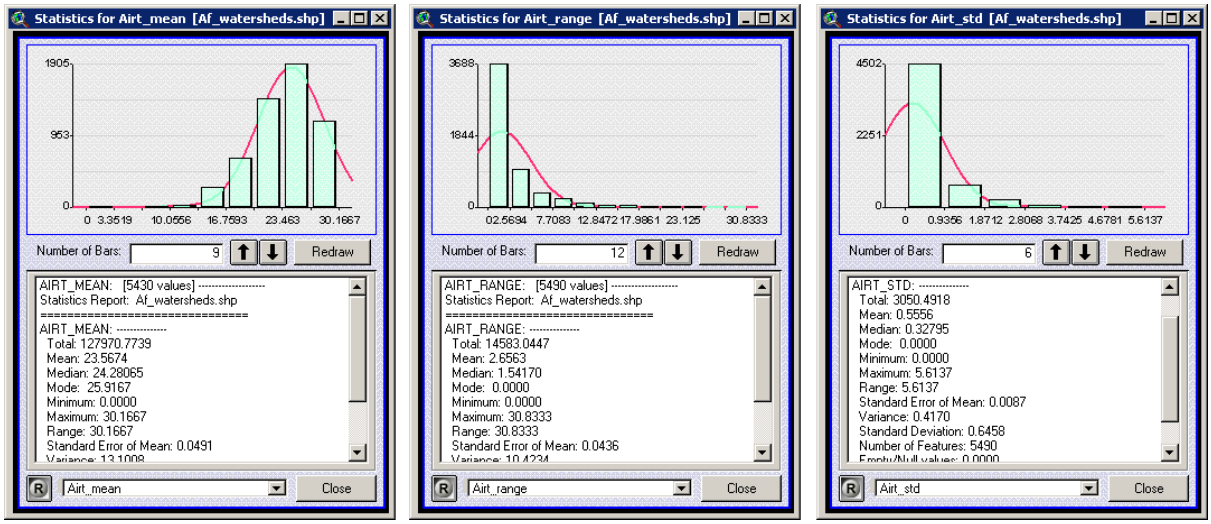
Also, the user may choose to calculate statistics on either all the features or only the selected features. If no features are selected, this tool will use all the features regardless of which option is chosen. The user also may choose to calculate statistics on multiple fields at one time.

The user may specify certain values they do not wish to include in the analysis. For example, it is common practice to designate some number to mean "No Data", or to identify values not involved in the analysis. Researchers often use -9999 or -99999 for this purpose, especially with datasets where such a value would be impossible (e.g., elevation, population, area, etc.) The user may designate as many of these values as desired by entering them into the "Ignore Values" section, and checking the "Ignore" box.

The user may choose between either Simple or Advanced output. Simple output includes the *Sum, Number of Features, Mean, Minimum* and *Maximum*, and is reported in a text box:



Advanced output includes the *Sum, Mean, Median, Mode(s), Minimum, Maximum, Range, Standard Error of Mean, Variance, Standard Deviation, Number of Features,* and *Number of Null Values*, and is reported in a histogram:



Although only one histogram window will be open, the user may choose which set of statistics to view by choosing the field from the drop-down box at the bottom of the window. Also, the user may change the number of histogram bars to display by clicking the up/down arrows and selecting “Redraw”. The red line behind the histogram bars demonstrates how the bars should be arranged if the data were normally distributed. In the above examples, the mean air temperature values follow the normal distribution better than the range and standard deviation of air temperature values. The “R” button at the window’s bottom left is the “Refresh” button, and can be used if the image becomes corrupted. Clicking this button will redraw the image .

Summary Statistics on a Field in a Table:

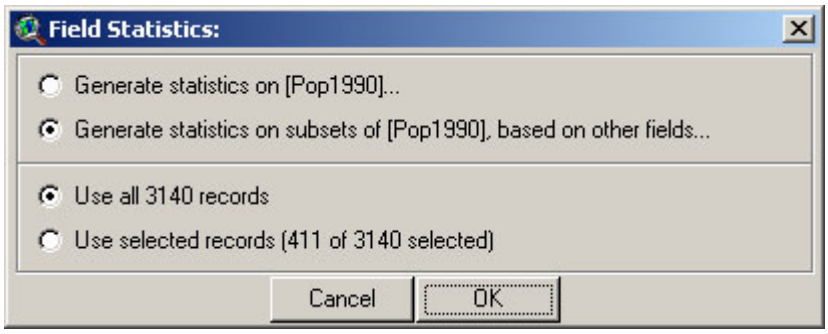
The  button in the Table button bar will be enabled only if a numeric field has been selected. This tool will allow the user to generate a large number of statistics on the values within a given field. The user may choose from: mean; standard error of the mean; confidence intervals; minimum; 1st quartile; median; 3rd quartile; maximum; range; variance; standard deviation; average absolute deviation; skewness,

normal and Fisher's G1; kurtosis, normal and Fisher's G2; number of records; number of null values; mode; and total sum for any attribute field(s) within a set of selected records.

This tool also allows users to break up the dataset into subsets based on one or more additional fields and generate multiple statistics for each subset of data. For example, if a person had a table of county-level statistics for all the counties in the United States, this tool would let them calculate a single set of statistics for all counties combined, or separate sets of statistics for each state or region.

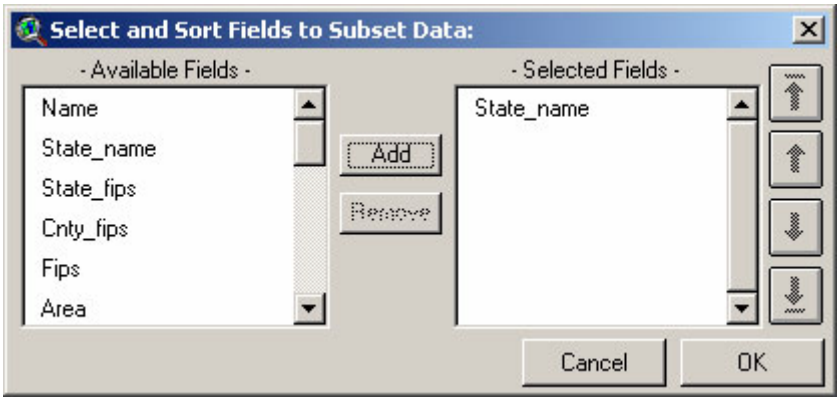
Users can use a single or multiple fields as classification fields to divide the data into subsets. If the user chooses multiple fields, then this extension will develop a separate set of statistics for each unique combination of classification values.

Begin by selecting the field containing your data, clicking the  to open the "Field Statistics" dialog, and setting your preferences:



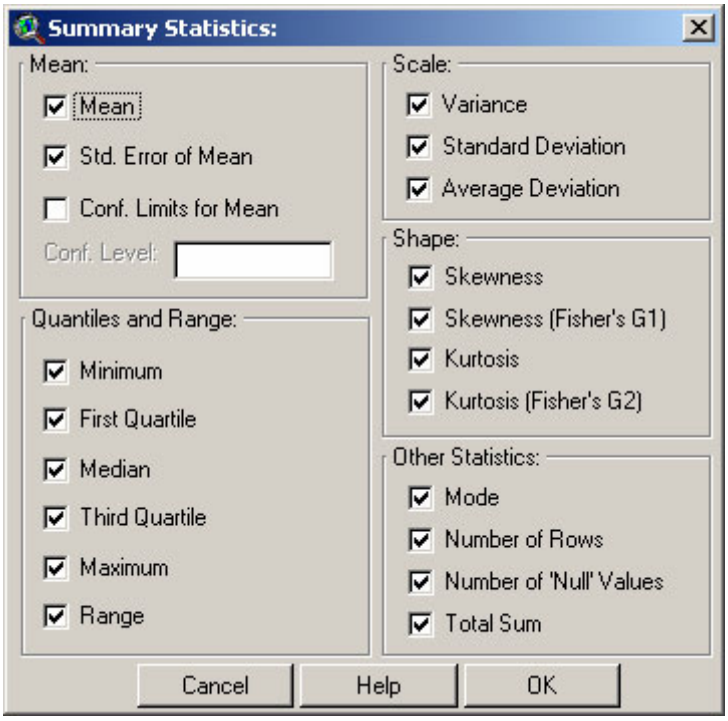
Generating Statistics on Multiple Subsets of Data:

Note that you have options to generate statistics on all data in the field or on subsets of that data. If you choose to generate statistics on subsets of the data, you will next be prompted to specify the fields containing your classification values. For example, if you wanted to analyze county statistics by state, then you would need to specify the field containing the state names.

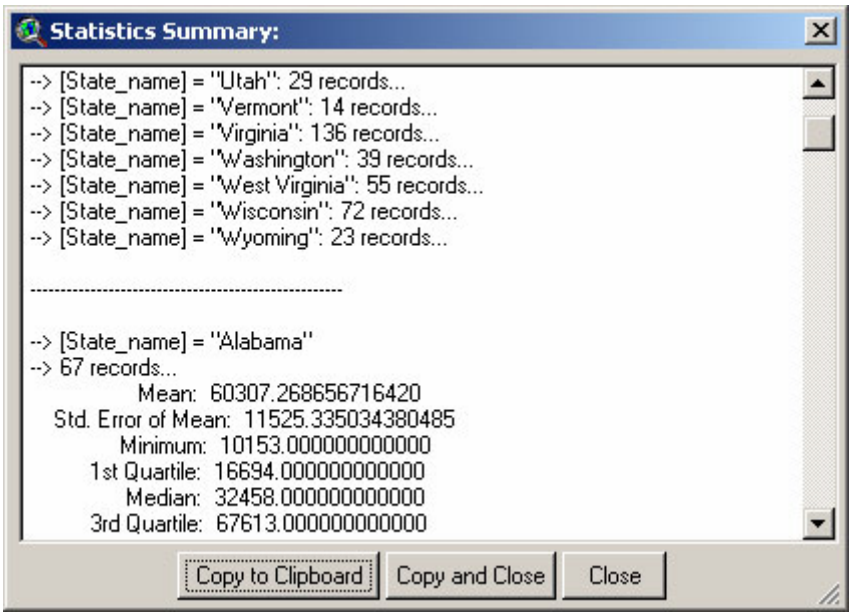


Note that you may choose multiple fields and change their order. If you choose multiple fields, then this extension will generate statistics for each unique combination of field values. The field order will not change the statistics produced, but will change the order they are presented in the final report.

Click 'OK' and specify the statistics you would like to generate. These statistics are described in detail below:

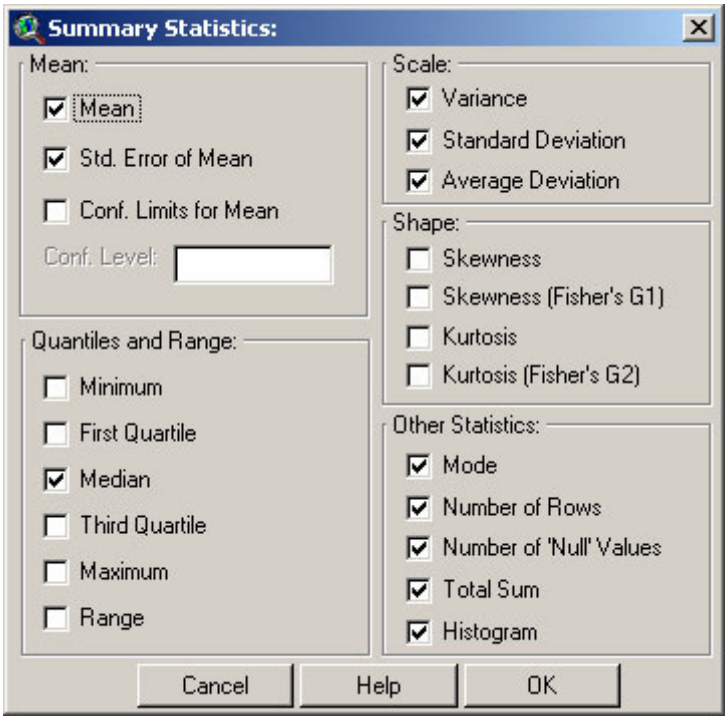


Click ‘OK’ and the extension will generate a report:

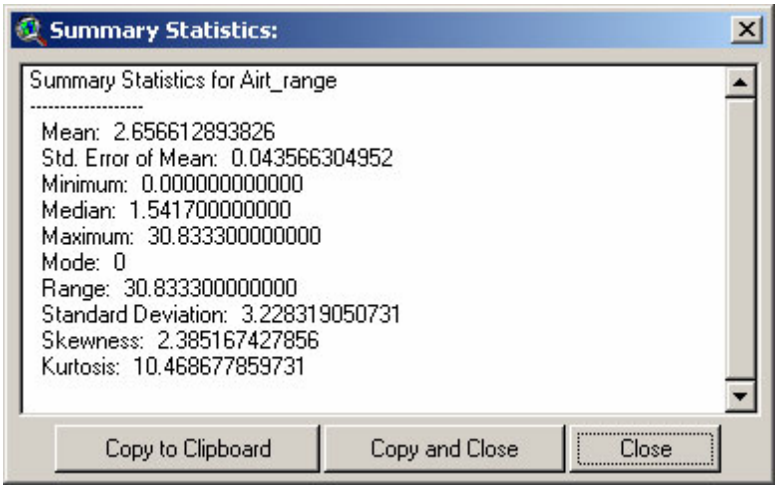


Generating Statistics on a Single Dataset:

If you choose to calculate statistics on all data in that field, you will next prompted to specify your statistics in the Summary Statistics dialog. This version is slightly different in that here you have the option to create a histogram if you wish.



Choose the desired statistics and then click "OK." If you selected the Histogram option, the output will appear in a histogram as illustrated above. If the Histogram option was not selected, the output will appear in a report window:



This tool may also be accessed with Avenue code, which enables more advanced users to pass these statistics to variables, and then use the calculated values in other places. Please review *Calculating Summary Statistics with Avenue* on p. 51 for details on accessing the Avenue script directly.

1) Mean: Calculated as: $\frac{\sum x}{n}$

2) Standard (Std) Error of the Mean: Calculated as: $s\{\bar{Y}\} = \frac{s}{\sqrt{n}}$, where s = sample standard deviation

3) Confidence Interval: The confidence limits for population mean μ with a confidence coefficient (1 – α), given a sample population mean $\bar{X}$, are calculated as:

$$\bar{X} \pm t_{\left(1-\alpha/2;n-1\right)} s\{\bar{Y}\},$$

where $s\{\bar{Y}\} = \frac{s}{\sqrt{n}}$, s = Sample Standard Deviation

and $t_{\left(1-\alpha/2;n-1\right)}$ = value from the t distribution at $p = \left(1-\alpha/2\right)$ and $n-1$ degrees of freedom.

4) Minimum: The lowest value in the data set.

5) Quartiles and Median: Those values, at which at most (P)% of the data lie below the value, and at most (1 – P)% of the data lie above the value. There are different ways to calculate quartile values which produce similar yet different results. Some methods draw the quartile values from the data set itself, so that the value called the "quartile" will always be found in the data. This script uses a different method which occasionally calculates a quartile value which represents the midpoint between two values from the data set, using the following algorithm:

Assuming the data have been sorted from lowest to highest:

$$\text{Quartile 1 Index} = (N+1)\times 0.25 = Q(1)$$
$$\text{Quartile 2 Index} = (N+1)\times 0.50 = Q(2)$$
$$\text{Quartile 3 Index} = (N+1)\times 0.75 = Q(3)$$

If $Q(N)$ is an integer, then:

$$\textit{Quartile} = Q(N)^{th}\ \textit{Value}$$

If $Q(N)$ is not an integer, then:

R = that value at which $R < N < R+1$, and

$$\textit{Quartile} = \frac{\left(Q(R)^{th}\ \textit{Value}\right) + \left(Q(R+1)^{th}\ \textit{Value}\right)}{2}$$

6) Maximum: The highest value in the data set.

7) Range: Maximum - Minimum

8) Variance: Calculated as: $\text{Variance} = \frac{\sum_{i=1}^{n}(y_i - \bar{y})^2}{n-1}$

9) Standard Deviation: Calculated as: $\text{Std. Deviation} = \sqrt{\frac{\sum_{i=1}^{n}(y_i - \bar{y})^2}{n-1}}$

10) Average (Avg) Deviation: Calculated as: $\text{Avg. Deviation} = \frac{\sum_{i=1}^{n}|y_i - \bar{y}|}{n}$

11) Skewness: Measures the degree of asymmetry of the sample data around the mean.

$$\text{Skewness} = \frac{m_3}{m_2^{3/2}},$$

$$\text{where: } m_2 = 2^{\text{nd}} \text{ moment} = \frac{\sum_{i=1}^{n}(y_i - \bar{y})^2}{n}$$

$$\text{and: } m_3 = 3^{\text{rd}} \text{ moment} = \frac{\sum_{i=1}^{n}(y_i - \bar{y})^3}{n}$$

12) Skewness (Fisher's G1): There are alternative methods to calculate *skewness* measures of the data. S-PLUS uses the *Fisher's G1* variation, calculated as:

$$\text{Fisher's G1} = \frac{b_1\sqrt{n(n-1)}}{n-2}$$

$$\text{where: } b_1 = \frac{m_3}{m_2^{3/2}} \quad \text{(standard measure of skewness)},$$

$$\text{and: } m_2 = 2^{\text{nd}} \text{ moment} = \frac{\sum_{i=1}^{n}(y_i - \bar{y})^2}{n}$$

$$\text{and: } m_3 = 3^{\text{rd}} \text{ moment} = \frac{\sum_{i=1}^{n}(y_i - \bar{y})^3}{n}$$

13) Kurtosis: Measures the "peakedness" or "pointedness" in a distribution, and calculated as:

$$\text{Kurtosis} = \frac{m_4}{m_2^{2}},$$

$$\text{where: } m_2 = 2^{\text{nd}} \text{ moment} = \frac{\sum_{i=1}^{n}(y_i - \bar{y})^2}{n}$$

$$\text{and: } m_4 = 4^{\text{th}} \text{ moment} = \frac{\sum_{i=1}^{n}(y_i - \bar{y})^4}{n}$$

14) Kurtosis (Fisher's G2): As with *Skewness*, there are alternative ways to calculate kurtosis. S-PLUS uses the *Fisher's G2* variation, calculated as:

$$\text{Fisher's G2} = \frac{(n+1)(n-1)}{(n-2)(n-3)}\left[b_2 - \frac{3(n-1)}{n+1}\right]$$

$$\text{where: } b_2 = \frac{m_4}{m_2^{2}} \quad \text{(standard measure of kurtosis)},$$

$$\text{and: } m_2 = 2^{\text{nd}} \text{ moment} = \frac{\sum_{i=1}^{n}(y_i - \bar{y})^2}{n}$$

$$\text{and: } m_4 = 4^{\text{th}} \text{ moment} = \frac{\sum_{i=1}^{n}(y_i - \bar{y})^4}{n}$$

15) Mode: The value occurring most often. There could be multiple modes or no modes within a given dataset. If no value in the dataset is found more than once, this option will report no modes were found. If no value occurs more than once, no mode is returned.

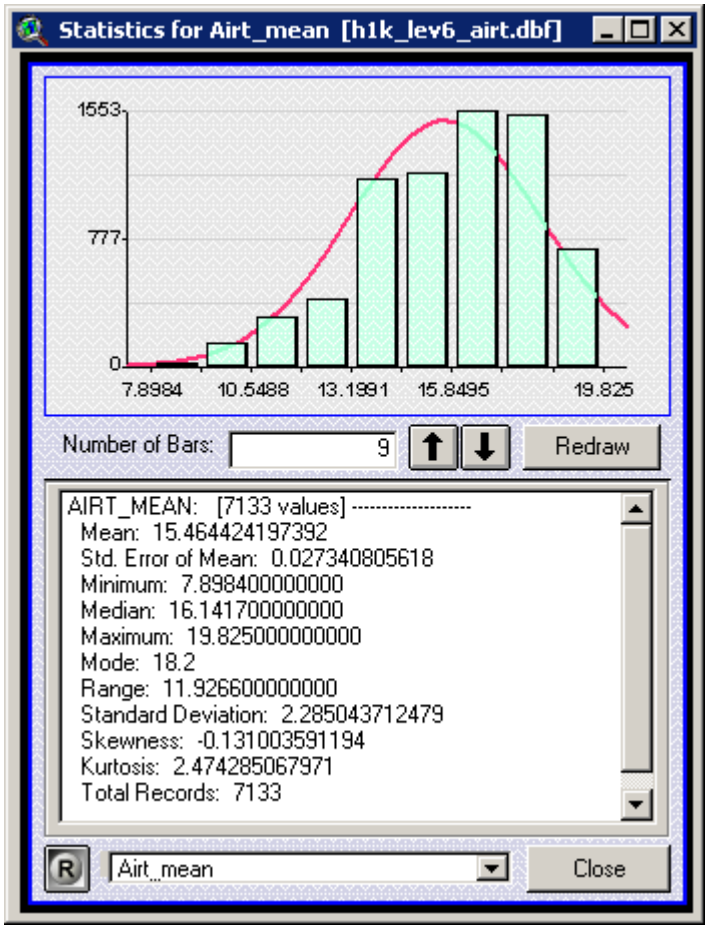
16) Number of Rows: The total number of rows of data examined during the analysis.

17) Number of 'Null' Values: The total number of cases of missing data. These are represented as "null" numbers in the table, which are different than zeros.

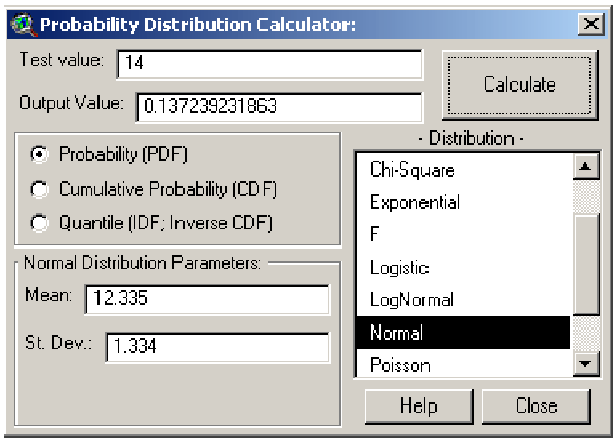
18) Total Sum: Sum of all non-null values, calculated as: $\sum x$

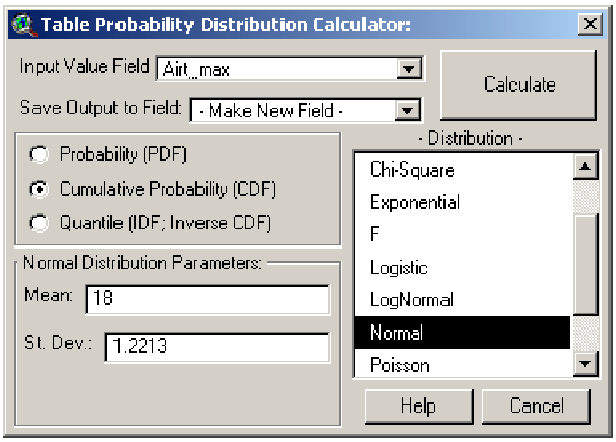
19) Histogram: This is a graphic illustrating the shape of the data and is useful for visually determining if the data are normally distributed. You may change the number of vertical bars by clicking the up/down arrows and then the "Redraw" button. The red line behind the bars shows how the data would appear if they were normally distributed. The drop-down box at the bottom of the illustration (containing the words "Airt_mean" in this example) shows the field from which the statistics were calculated. If you generated this histogram from a theme in a View, you may have selected multiple fields to calculate the statistics. This option allows you to choose which set of statistics to view.

The R" button in the lower left-hand corner is the Refresh button. If the histogram window becomes corrupted, click the "R" button to regenerate it.



Probability Distribution Calculators:





This extension includes two versions of a Probability Distribution Calculator, each of which calculate distribution data based on a variety of distributions and parameters. The **Probability Distribution Calculator** is started from within a View, and is opened by clicking on the  button in the View toolbar. You simply enter the input and parameter values, specify whether you are calculating Probability, Cumulative Probability or Quantile values, click "Calculate", and the result appears in the "Output Value" window. This calculator remains open until you close it, and you can leave the calculator open as you conduct other ArcView routines.

The "Table Probability Distribution Calculator" is designed to work on all selected records in a table, applying the distribution parameters to each value and saving the results to a field within the table. This calculator is opened from within a Table by clicking on the  button in the Table toolbar. Select the field containing the "Input" values, then decide whether to create a new field or use an existing field to save the "Output" values. Then, click "Calculate" to generate distribution values for all selected records. The window stays open until you click "Calculate" or "Cancel".

Distribution functions included within this extension may be grouped in three categories; *Probability Density Functions*, *Cumulative Distribution Functions* and *Quantile Functions*. In general, the *Probability Density Functions* return the probability the Test Value = *X* given that particular distribution. The *Cumulative Distribution Functions* return the probability the Test Value ≤ *X*, given that particular distribution. The *Quantile Functions* (sometimes referred to as *Inverse Density Functions* or *Percent Point Functions*) return the Value *X* at which *P*(*X*) = [specified probability], given that particular distribution.

Functions and Probability Distributions			
Distribution	**Probability Density Function**	**Cumulative Distribution Function**	**Quantile Function**
Beta	PDF_Beta	CDF_Beta	IDF_Beta
Binomial	PDF_Binomial	CDF_Binomial	IDF_Binomial
Cauchy	PDF_Cauchy	CDF_Cauchy	IDF_Cauchy
Chi-Square	PDF_ChiSquare	CDF_ChiSquare	IDF_ChiSquare

Exponential	PDF_Exp	CDF_Exp	IDF_Exp
F	PDF_F	CDF_F	IDF_F
Logistic	PDF_Logistic	CDF_Logistic	IDF_Logistic
LogNormal	PDF_LogNormal	CDF_LogNormal	IDF_LogNormal
Normal	PDF_Normal_Simpsons	CDF_Normal	IDF_Normal
Poisson	PDF_Poisson	CDF_Poisson	IDF_Poisson
Student's T	PDF_StudentsT	CDF_StudentsT	IDF_StudentsT
Weibull	PDF_Weibull	CDF_Weibull	IDF_Weibull

Equations for each function are included in the *Distribution Functions, Parameters and Usages* (p. 53), but some of them do not have closed formulas which can be calculated and therefore must be computed numerically. Those interested should refer to the references to find source code and computational methods of calculating these functions. We recommend Croarkin & Tobias (date unknown) and McLaughlin (2001) for illustrations of the various distributions, and Press et al. (1988-1997) and Burkardt (2001) for computational methods. All of these sources are available on-line.

The descriptions in *Functions, Parameters and Usages* (p. 53) include four methods of utilizing each function. The first method describes how to use the *Probability Distribution Calculators* to calculate values. There are three additional methods available for programmers who may want to access the functions through Avenue code. Simply copy the line of code exactly as written, substituting your parameter variable names in the proper places.

Avenue Functions:

1) The first *Avenue* option sends your parameters to a central script called "KappaStats.DistFunc", which checks for possible errors in the parameters (e.g. using a negative value for Degrees of Freedom). If the script finds errors, it will halt operation and alert you to the problem. If it doesn't find errors, it forwards your parameters to the appropriate script and returns the result. Users may want to review the script "KappaStats.ProbDlogCalculate" for an example of this option. IMPORTANT: Users should be aware this script only checks whether the input values follow the rules described in *Functions, Parameters and Usages* on p. 53. It doesn't check for programming errors, such as sending a non-numeric value to the script.

2) The second *Avenue* option is similar to the first. It sends your parameters to a central script to check for errors (in this case, "KappaStats.TableDistFunc"), but it doesn't halt operation if it finds an error. Rather, it returns an error message (in *String* format) detailing the problem. We recommend this option for cases in which the user wants to conduct calculations on a series of values (i.e. records in a table), but doesn't want the function to stop if it finds an illegal value (e.g., possibly a record with no data). This option would allow the user to insert an "if-then" statement in their code to check if the result is a String or a Number. Numerical responses would indicate successful calculations while String responses could be appended to a running report of unsuccessful calculations. Users may want to review the script "KappaStats.ProbTabDlogCalculate" for an example of this option.

3) If you'd like to skip the error-checking routines, use the third *Avenue* option to send your parameters to the relevant script directly.

Calculating Summary Statistics with Avenue

The Summary Statistics tool collects a series of True/False and numerical parameters from the user and sends them to a script called "KappaStats.CalcFieldStats." This script executes the necessary calculations and returns a list of results. The tool then prints those results up in a Report window for the user.

Avenue programmers may bypass the dialog and send values to the script directly, and then they will have the desired statistics directly available within a list. For example, many statistical calculations require metrics such as means, standard deviations, variances, quartiles, etc. The user may want to generate these values early in a script and then use them in later calculations. The "KappaStats.CalcFieldStats" script makes it simple to generate such values from data in a table.

This option is simpler than the standard Avenue method for generating statistics, which is to create a new file on the hard drive and then use the "Summarize" request to save statistics to the file. It also offers a larger variety of statistical output, including confidence intervals, standard error of the mean, average deviation, and kurtosis/skewness values. This option is also slower on large datasets. However, it doesn't divide up the dataset into subsets like the "Summarize" function.

The function may be used with just a few lines of code:

```
ListOfResults = av.Run("KappaStats.CalcFieldStats", {ListOfInputParameters,
       theVTab, theField})
```

The object "`theVTab`" is a VTab object containing your data, and "`theField`" is a Field object in the VTab, reflecting the field in which you want to calculate the statistics.

The "`ListOfInputParameters`" must contain 22 values, most of which are Boolean (e.g. true/false) reflecting whether you want a particular statistic calculated. Note the last value should be set to "False".

```
ListOfInputParameters = {CalcMean, CalcSEMean, CalcConInt, Con_Level,
   CalcMinimum, Calc1stQuart, CalcMedian, Calc3rdQuart, CalcMaximum,
   CalcVariance, CalcStandDev, CalcAvgDev, CalcSkewness, CalcSkewFish,
   CalcKurtosis, CalcKurtFish, CalcCount, CalcNumNull, CalcSum, CalcRange,
   CalcMode, False}
```

Where:

`CalcMean:` **Boolean**, *True* if you want to calculate the mean.

`CalcSEMean:` **Boolean**, *True* if you want to calculate the standard error of the mean.

`CalcConInt:` **Boolean**, *True* if you want to calculate confidence intervals of the mean.

`Con_Level:` **Number**, $0 \le p \le 1$, where p = probability = $(1 - \alpha)$

`CalcMinimum:` **Boolean**, *True* if you want to calculate the minimum value.

`Calc1stQuart:` **Boolean**, *True* if you want to calculate the 1st quartile.

`CalcMedian:` **Boolean**, *True* if you want to calculate the median.

`Calc3rdQuart:` **Boolean**, *True* if you want to calculate the 3rd quartile.

`CalcMaximum:` **Boolean**, *True* if you want to calculate the maximum value.

`CalcVariance:` **Boolean**, *True* if you want to calculate the variance.

`CalcStandDev:` **Boolean**, *True* if you want to calculate the standard deviation.

`CalcAvgDev:` **Boolean**, *True* if you want to calculate the absolute average deviation.

`CalcSkewness:` **Boolean**, *True* if you want to calculate the standard skewness.

`CalcSkewFish:` **Boolean**, *True* if you want to calculate the Fisher's G1 skewness.

`CalcKurtosis:` **Boolean**, *True* if you want to calculate the standard kurtosis.

`CalcKurtFish:` **Boolean**, *True* if you want to calculate the Fisher's G2 kurtosis.

`CalcCount:` **Boolean**, *True* if you want to calculate the total number of rows of data.

`CalcNumNull:` **Boolean**, *True* if you want to calculate the number of null values.

`CalcSum:` **Boolean**, *True* if you want to calculate the sum.

`CalcRange:` **Boolean**, *True* if you want to calculate the Range.

`CalcMode:` **Boolean**, *True* if you want to calculate the Mode.

`ForHistogram:` **False**, intended only for internal use.

When the script finishes, it will return a list of 19 values to you which represent the various requested statistics. If you did not request a particular statistic, then it will not be calculated and the return list will contain a "nil" object in its place. NOTE: if you requested a confidence interval, the upper and lower levels are returned as a separate list (3rd object in the Return List).

```
Return list:  {Mean, Standard Error of Mean, {Lower Confidence Level,
              Upper Confidence Level}, Minimum, 1st Quartile, Median, 3rd
              Quartile, Maximum, Variance, Standard Deviation, Skewness,
              Fisher's GI Skewness, Kurtosis, Fisher's G2 Kurtosis,
              Record Count, Number of Null Values, Sum, Range, Mode}
```

For example: If you had a table of population demographic data containing a field of Annual Income values, and you were interested in the mean annual income plus a 95% confidence interval around that mean, the code would be set up as:

```
theDemographyVTab = YourTable.GetVTab
theField = theDemographyVTab.FindField("Income")
theInputParameters = {True, False, True, 0.95, False, False, False, False,
       False, False, False, False, False, False, False, False, False, False,
       False, False, False}
theReturnList = av.Run("KappaStats.CalcFieldStats", {theInputParameters,
       theDemographyVTab , theField})
theMeanIncome = theReturnList.Get(0)
theLowerConfidenceLimit = theReturnList.Get(2).Get(0)
theUpperConfidenceLimit = theReturnList.Get(2).Get(1)
```

All the objects in "`theReturnList`" will be "nil" objects except for the ones at indices 0 and 2. The Mean will be at index 0, the Lower 95% Confidence Limit will be the first item in index 2, and the Upper 95% Confidence Limit will be the second item in index 2.

In general, all the possible statistics may be obtained with the following lines of code. Simply copy and paste the appropriate lines into your script:

```
theMean = theReturnList.Get(0)
theSEMean = theReturnList.Get(1)
if (Calculating_Confidence_Intervals)
       LowerCI = theReturnList.Get(2).Get(0)
       UpperCI = theReturnList.Get(2).Get(1)
end
theMinimum = theReturnList.Get(3)
theQ1 = theReturnList.Get(4)
theMedian = theReturnList.Get(5)
theQ3 = theReturnList.Get(6)
theMaximum = theReturnList.Get(7)
theVar = theReturnList.Get(8)
theStdDev = theReturnList.Get(9)
theAvgDev = theReturnList.Get(10)
theSkew = theReturnList.Get(11)
theFisherSkew = theReturnList.Get(12)
theKurt = theReturnList.Get(13)
theFisherKurt = theReturnList.Get(14)
theCount = theReturnList.Get(15)
theNumberNull = theReturnList.Get(16)
theSum = theReturnList.Get(17)
theRange = theReturnList.Get(18)
theMode = theReturnList.Get(19)
```

Functions, Parameters and Usages

Probability Density Functions:

1. **PDF_Beta:** This function returns the probability that the Test Value = *X*, assuming a Beta distribution and the specified Shape parameters. This is the standardized Beta function, where Location = 0 and Scale (upper bound) = 1. According to McLaughlin (2001), parameters *Shape1* and *Shape2* can be any positive value, but rarely exceed 10. The function becomes nearly flat if the values becomes much larger.

   a) Parameters:

      i) Test Value: Number

      ii) Shape1: Number > 0

      iii) Shape2: Number > 0

   b) Usages:

      i) From "Probability Distribution Calculator", select "Probability (PDF)" and *Beta* distribution.

      ii) (*Avenue*): theProb = av.Run("KappaStats.DistFunc", {"PDF_Beta", {Test Value, Shape1, Shape2}})

      iii) (*Avenue*): theProb = av.Run("KappaStats.TableDistFunc", {"PDF_Beta", {Test Value, Shape1, Shape2}})

      iv) (*Avenue*): theProb = av.Run("KappaStats.PDF_Beta", {Test Value, Shape1, Shape2})

   c) Function:

$$\text{Beta PDF} = \frac{\Gamma(S_1 + S_2)}{\Gamma(S_1)\Gamma(S_2)} y^{S_1 - 1}(1-y)^{S_2 - 1}$$

where: y = Test Value, S_1 = Shape 1, S_2 = Shape 2

and: $\Gamma(x) = \int_0^\infty t^{x-1} e^{-1} dt$

2. **PDF_Binomial:** The Binomial distribution is used when there are exactly two mutually exclusive outcomes of a trial. This function returns the probability of getting *X* successes out of *N* trials, given a probability of success = *P*.

   a) Parameters:

      i) # Successes: Integer ≥ 0

      ii) # Trials: Integer ≥ 2, # Successes

      iii) Probability of Success: Number ($0 \geq p \geq 1$)

   b) Usages:

      i) From "Probability Distribution Calculator", select "Probability (PDF)" and *Binomial* distribution.

      ii) (*Avenue*): theProb = av.Run("KappaStats.DistFunc", {"PDF_Binomial", {#Success, #Trials, Probability of Success}})

      iii) (*Avenue*): theProb = av.Run("KappaStats.TableDistFunc", {"PDF_Binomial", {#Success, #Trials, Probability of Success}})

      iv) (*Avenue*): theProb = av.Run("KappaStats.PDF_Binomial", {#Success, #Trials, Probability of Success}})

c) Function: Binomial PDF $= \left(\frac{B!}{y!(B-y)!} \right) A^y (1-A)^{B-y}$

where: y = #Successes, A = Probability of Success, B = #Trials

3. PDF_Cauchy: This function returns the probability that the Test Value = *X*, assuming a *Cauchy* distribution with the specified mean and standard deviation. The *Standardized Cauchy distribution* is that with *Location = 0* and *Scale = 1.*

a) Parameters:

i) Test Value: Number

ii) Location: Number

iii) Scale: Number > 0

b) Usages:

i) From "Probability Distribution Calculator", select "Probability (PDF)" and *Cauchy* distribution.

ii) (*Avenue*): theProb = av.Run("KappaStats.DistFunc", {"PDF_Cauchy", {Test Value, Location, Scale}})

iii) (*Avenue*): theProb = av.Run("KappaStats.TableDistFunc", {"PDF_Cauchy", {Test Value, Location, Scale}})

iv) *(Avenue*): theProb = av.Run("KappaStats.PDF_Cauchy", {Test Value, Location, Scale})

c) Function: Cauchy PDF $= \frac{1}{\pi B \left[1 + \left(\frac{y-A}{B} \right)^2 \right]}$

where: y = Test Value, A = Location, B = Scale

4. PDF_ChiSquare: This function returns the probability that the Test Value = *X*, assuming a *Chi-Square* distribution with the specified Degrees of Freedom. The *Chi-Square* distribution results when *v* (where *v* = Degrees of Freedom) independent variables with standard normal distributions are squared and summed (Croarkin & Tobias, Date unknown).

a) Parameters:

i) Test Value: Number ≥ 0

ii) Degrees of Freedom: Number > 0

b) Usages:

i) From "Probability Distribution Calculator", select "Probability (PDF)" and *Chi-Square* distribution.

ii) (*Avenue*): theProb = av.Run("KappaStats.DistFunc", {"PDF_ChiSquare", {Test Value, DF}})

iii) (*Avenue*): theProb = av.Run("KappaStats.TableDistFunc", {"PDF_ChiSquare", {Test Value, DF}})

iv) *(Avenue*): theProb = av.Run("KappaStats.PDF_ChiSquare", {Test Value, DF})

c) Function: Chi-Square PDF $= \dfrac{e^{\frac{-y}{2}} x^{\frac{v}{2}-1}}{2^{\frac{v}{2}}\Gamma\left(\frac{v}{2}\right)}$

where: y = Test Value, S_1 = Shape 1, S_2 = Shape 2

and: $\Gamma(x) = \int_0^\infty t^{x-1}e^{-1}dt$

5. PDF_Exp: This function returns the probability that the Test Value = *X*, assuming an *Exponential* distribution with the specified mean. This script uses the 1-parameter version of the equation (i.e. assuming *Location* = 0). The *Standard Exponential Distribution* is that which has Mean = *1*.

a) Parameters:

i) Test Value: Number ≥ 0

ii) Mean: Number > 0

b) Usages:

i) From "Probability Distribution Calculator", select "Probability (PDF)" and *Exponential* distribution.

ii) (*Avenue*): theProb = av.Run("KappaStats.DistFunc", {"PDF_Exp", {Test Value, Mean}})

iii) (*Avenue*): theProb = av.Run("KappaStats.TableDistFunc", {"PDF_Exp", {Test Value, Mean}})

iv) *(Avenue*): theProb = av.Run("KappaStats.PDF_Exp", {Test Value, Mean})

c) Function: Exponential PDF $= \dfrac{1}{\beta}e^{-\frac{x}{\beta}}$

where: x = Test Value, β = Mean (or Scale Parameter)

6. PDF_F: This function returns the probability that the Test Value = *X*, assuming an *F* distribution with the specified Degrees of Freedom. The *F* distribution is the ratio of two *Chi-Square* distributions with ratios v_1 and v_2 respectively.

a) Parameters:

i) Test Value: Number ≥ 1

ii) 1st Degrees of Freedom: Number > 1

iii) 2nd Degrees of Freedom: Number > 1

b) Usages:

i) From "Probability Distribution Calculator", select "Probability (PDF)" and *F* distribution.

ii) (*Avenue*): theProb = av.Run("KappaStats.DistFunc", {"PDF_F", {Test Value, DF1, DF2}})

iii) (*Avenue*): theProb = av.Run("KappaStats.TableDistFunc", {"PDF_F", {Test Value, DF1, DF2}})

iv) *(Avenue*): theProb = av.Run("KappaStats.PDF_F", {Test Value, DF1, DF2})

c) Function: $$\text{F PDF} = \frac{\Gamma\left(\frac{v_1+v_2}{2}\right)\left(\frac{v_1}{v_2}\right)^{\frac{v_1}{2}} x^{\frac{v_1}{2}-1}}{\Gamma\left(\frac{v_1}{2}\right)\Gamma\left(\frac{v_1}{2}\right)\left(1+\frac{v_1 x}{v_2}\right)^{\frac{v_1+v_2}{2}}}$$

where: $x = \text{Test Value}$, $v_1 = \text{DF1}$, $v_2 = \text{DF2}$

and: $\Gamma(x) = \int_0^\infty t^{x-1} e^{-1} dt$

7. PDF_Logistic: This function returns the probability that the Test Value = *X*, assuming a *Logistic* distribution with the specified mean and scale.

a) Parameters:

i) Test Value: Number

ii) Mean: Number

iii) Scale: Number > 0

b) Usages:

i) From "Probability Distribution Calculator", select "Probability (PDF)" and *Logistic* distribution.

ii) (*Avenue*): theProb = av.Run("KappaStats.DistFunc", {"PDF_Logistic", {Test Value, Mean, Scale}})

iii) (*Avenue*): theProb = av.Run("KappaStats.TableDistFunc", {"PDF_Logistic", {Test Value, Mean, Scale}})

iv) *(Avenue*): theProb = av.Run("KappaStats.PDF_Logistic", {Test Value, Mean, Scale})

c) Function: $$\text{Logistic PDF} = \frac{1}{B}\frac{\exp\left(\frac{y-A}{B}\right)}{\left[1+\exp\left(\frac{y-A}{B}\right)\right]^2}$$

where: $y = \text{Test Value}$, $A = \text{Mean}$, $B = \text{Scale}$

8. PDF_LogNormal: This function returns the probability that the Test Value = *X*, assuming a *LogNormal* distribution with the specified mean and scale. A *LogNormal* distribution occurs when natural logarithms of variable *X* are normally distributed. The *Standard LogNormal Distribution* is that with Mean = 0 and Scale = 1. The *2-Parameter LogNormal Distribution* is that with Mean = 0.

a) Parameters:

i) Test Value: Number ≥ 0

ii) Mean: Number > 0

iii) Scale: Number > 0

b) Usages:

i) From "Probability Distribution Calculator", select "Probability (PDF)" and *LogNormal* distribution.

ii) (*Avenue*): theProb = av.Run("KappaStats.DistFunc", {"PDF_LogNormal", {Test Value, Mean, Scale}})

iii) (*Avenue*): theProb = av.Run("KappaStats.TableDistFunc", {"PDF_LogNormal", {Test Value, Mean, Scale}})

iv) *(Avenue*): theProb = av.Run("KappaStats.PDF_LogNormal", {Test Value, Mean, Scale})

c) Function: LogNormal PDF $= \frac{1}{B\sqrt{2\pi}} \exp\left(-\frac{1}{2}\left(\frac{\ln(y) - A}{B} \right)^2 \right)$

where: $y =$ Test Value, $A =$ Mean, $B =$ Scale

9. PDF_Normal: This function returns the probability that the Test Value = *X*, assuming a *Normal* distribution with the specified mean and standard deviation. The *Standard Normal Distribution* is that with Mean = 0 and Standard Deviation = 1.

a) Parameters:

i) Test Value: Number

ii) Mean: Number

iii) Standard Deviation: Number > 0

b) Usages:

i) From "Probability Distribution Calculator", select "Probability (PDF)" and *Normal* distribution.

ii) (*Avenue*): theProb = av.Run("KappaStats.DistFunc", {"PDF_Normal", {Test Value, Mean, St. Dev.}})

iii) (*Avenue*): theProb = av.Run("KappaStats.TableDistFunc", {"PDF_Normal", {Test Value, Mean, St. Dev.}})

iv) *(Avenue*): theProb = av.Run("KappaStats.PDF_Normal", {Test Value, Mean, St. Dev.})

c) Function: Normal PDF $= \frac{1}{B\sqrt{2\pi}} \exp\left(-\frac{1}{2}\left(\frac{y - A}{B} \right)^2 \right)$

where: $y =$ Test Value, $A =$ Mean, $B =$ Scale

10. PDF_Poisson: This function returns the probability that the Specified Number of Events = *X*, assuming a *Poisson* distribution with the specified mean.

a) Parameters:

i) # Events: Integer ≥ 0

ii) Mean: Number > 0

b) Usages:

i) From "Probability Distribution Calculator", select "Probability (PDF)" and *Poisson* distribution.

ii) (*Avenue*): theProb = av.Run("KappaStats.DistFunc", {"PDF_Poisson", {# Events, Mean}})

iii) (*Avenue*): theProb = av.Run("KappaStats.TableDistFunc", {"PDF_Poisson", {# Events, Mean}})

iv) *(Avenue)*: theProb = av.Run("KappaStats.PDF_Poisson", {# Events, Mean})

c) Function: Poisson PDF $= \dfrac{\exp^{-A} A^{y}}{y!}$

where: $y =$ Test value, $A =$ Expectation (mean)

11. PDF_StudentsT: This function returns the probability that the Test Value = *X*, assuming a *Students T* distribution with the specified Degrees of Freedom. A *Student's T* distribution with 1df is a *Cauchy* Distribution, and it approaches a *Normal* distribution when DF>30. Various sources recommend using the *Normal* distribution if DF>100.

a) Parameters:

i) Test Value: Number

ii) Degrees of Freedom: Number > 0

b) Usages:

i) From "Probability Distribution Calculator", select "Probability (PDF)" and *Student's T* distribution.

ii) (*Avenue*): theProb = av.Run("KappaStats.DistFunc", {"PDF_StudentsT", {Test Value, DF}})

iii) (*Avenue*): theProb = av.Run("KappaStats.TableDistFunc", {"PDF_StudentsT", {Test Value, DF}})

iv) *(Avenue*): theProb = av.Run("KappaStats.PDF_StudentsT", {Test Value, DF})

c) Function: Student's T PDF $= \dfrac{\Gamma\left(\dfrac{v+1}{2}\right)}{\sqrt{\pi v}\,\Gamma\left(\dfrac{v}{2}\right)}\left[1+\dfrac{y^2}{v}\right]^{-\frac{v+1}{2}}$

where: $y =$ Test Value, $v =$ Degrees of Freedom

and: $\Gamma(x) = \int_0^{\infty} t^{x-1} e^{-1} dt$

12. PDF_Weibull: This function returns the probability that the Test Value = *X*, assuming a *Weibull* distribution with the specified Location, Scale and Shape parameters. The *Standardized Weibull Distribution* is that with *Location = 0* and *Scale = 1*. The *2-Parameter Weibull Distribution* is that with *Location = 0*.

a) Parameters:

i) Test Value: Number > Location

ii) Location: Number

iii) Scale: Number > 0

iv) Shape: Number > 0

b) Usages:

i) From "Probability Distribution Calculator", select "Probability (PDF)" and *Weibull* distribution.

ii) (*Avenue*): theProb = av.Run("KappaStats.DistFunc", {"PDF_Weibull", {Test Value, Location, Scale, Number}})

iii) (*Avenue*): theProb = av.Run("KappaStats.TableDistFunc", {"PDF_Weibull", {Test Value, Location, Scale, Number}})

iv) *(Avenue*): theProb = av.Run("KappaStats.PDF_Weibull", {Test Value, Location, Scale, Number})

c) Function: Weibull PDF $= \left(\frac{C}{B}\right)\left(\frac{y-A}{B}\right)^{C-1} \exp^{\left(-\left(\frac{y-A}{b}\right)^2\right)}$

where: y = Test Value, A = Location, B = Scale, C = Shape

Cumulative Distribution Functions:

1. **CDF_Beta:** This function returns the probability that the Test Value ≤ *X*, assuming a Beta distribution with the specified Shape parameters. This is the *Standardized Beta function*, where Location = 0 and Scale (upper bound) = 1. According to McLaughlin (2001), parameters *Shape1* and *Shape2* can be any positive value, but they rarely exceed 10. The function becomes nearly flat if the values get much larger than this.

   a) Parameters:

      i) Test Value: Number

      ii) Shape1: Number > 0

      iii) Shape2: Number > 0

   b) Usages:

      i) From "Probability Distribution Calculator", select "Cumulative Probability (CDF)" and *Beta* distribution.

      ii) (*Avenue*): theProb = av.Run("KappaStats.DistFunc", {"CDF_Beta", {Test Value, Shape1, Shape2}})

      iii) (*Avenue*): theProb = av.Run("KappaStats.TableDistFunc", {"CDF_Beta", {Test Value, Shape1, Shape2}})

      iv) *(Avenue*): theProb = av.Run("KappaStats.CDF_Beta", {Test Value, Shape1, Shape2})

   Beta CDF = $I(y,S_1,S_2)$ (From Press et al, 1997)

   c) Function: where: y = Test Value, S_1 = Shape 1, S_2 = Shape 2

   and: $I(x,a,b) \equiv \frac{B_x(a,b)}{B(a,b)} \equiv \frac{1}{B(a,b)}\int_0^x t^{a-1}(1-t)^{b-1}dt$

   and: $B(a,b) = \int_0^1 t^{a-1}(1-t)^{b-1}dt$

2. **CDF_Binomial:** The *Binomial* distribution is used when there are exactly two mutually exclusive outcomes of a trial. This function returns the probability of getting ≤ *X* successes out of *N* trials, given a probability of success= *P*.

   a) Parameters:

      i) # Successes: Integer ≥ 0

      ii) # Trials: Integer ≥ 2, # Successes

iii) Probability of Success: Number ($0 \geq p \geq 1$)

b) Usages:

i) From "Probability Distribution Calculator", select "Cumulative Probability (CDF)" and *Binomial* distribution.

ii) (*Avenue*): theProb = av.Run("KappaStats.DistFunc", {"CDF_Binomial", {#Success, #Trials, Probability of Success}})

iii) (*Avenue*): theProb = av.Run("KappaStats.TableDistFunc", {"CDF_Binomial", {#Success, #Trials, Probability of Success}})

iv) *(Avenue*): theProb = av.Run("KappaStats.CDF_Binomial", {#Success, #Trials, Probability of Success})

c) Function: Binomial CDF $= \sum_{i=1}^{y} \left(\frac{B!}{i!(B-i)!} \right) A^i (1-A)^{B-i}$

where: $y =$ #Successes, $A =$ Probability of Success, $B =$ #Trials

3. CDF_Cauchy: This function returns the probability that the Test Value ≤ *X*, assuming a *Cauchy* distribution with the specified Location and Scale parameters. The *Standardized Cauchy distribution* has *Location = 0* and *Scale = 1.*

a) Parameters:

i) Test Value: Number

ii) Location: Number

iii) Scale: Number > 0

b) Usages:

i) From "Probability Distribution Calculator", select "Cumulative Probability (CDF)" and *Cauchy* distribution.

ii) (*Avenue*): theProb = av.Run("KappaStats.DistFunc", {"CDF_Cauchy", {Test Value, Location, Scale}})

iii) (*Avenue*): theProb = av.Run("KappaStats.TableDistFunc", {"CDF_Cauchy", {Test Value, Location, Scale}})

iv) *(Avenue*): theProb = av.Run("KappaStats.CDF_Cauchy", {Test Value, Location, Scale})

c) Function: Cauchy CDF $= \frac{1}{2} + \frac{1}{\pi} \tan^{-1} \left(\frac{y-A}{B} \right)$

where: $y =$ Test Value, $A =$ Location, $B =$ Scale

4. CDF_ChiSquare: This function returns the probability that the Test Value ≤ *X*, assuming a *Chi-Square* distribution with the specified Degrees of Freedom. The *Chi-Square* distribution results when *v* (where *v* = Degrees of Freedom) independent variables with standard normal distributions are squared and summed (Croarkin & Tobias, Date unknown).

a) Parameters:

i) Test Value: Number ≥ 0

ii) Degrees of Freedom: Number > 0

b) Usages:

i) From "Probability Distribution Calculator", select "Cumulative Probability (CDF)" and *Chi-Square* distribution.

ii) (*Avenue*): theProb = av.Run("KappaStats.DistFunc", {"CDF_ChiSquare", {Test Value, DF}})

iii) (*Avenue*): theProb = av.Run("KappaStats.TableDistFunc", {"CDF_ChiSquare", {Test Value, DF}})

iv) *(Avenue*): theProb = av.Run("KappaStats.CDF_ChiSquare", {Test Value, DF})

$$\text{Chi-Square CDF} = \frac{\gamma\left(\frac{v}{2}, \frac{x}{2}\right)}{\Gamma\left(\frac{v}{2}\right)}$$

c) Function: where: y = Test Value, S_1 = Shape 1, S_2 = Shape 2

and: $\Gamma(x) = \int_0^\infty t^{x-1} e^{-1} dt$

and: $\gamma(x,y) = \int_0^y t^{x-1} e^{-1} dt$

5. CDF_Exp: This function returns the probability that the Test Value ≤ *X*, assuming an *Exponential* distribution with the specified mean. This script uses the 1-parameter version of the equation (i.e. assuming *Location* = 0). The *Standard Exponential Distribution* is that which has *Mean = 1*.

a) Parameters:

i) Test Value: Number ≥ 0

ii) Mean: Number > 0

b) Usages:

i) From "Probability Distribution Calculator", select "Cumulative Probability (CDF)" and *Exponential* distribution.

ii) (*Avenue*): theProb = av.Run("KappaStats.DistFunc", {"CDF_Exp", {Test Value, Mean}})

iii) (*Avenue*): theProb = av.Run("KappaStats.TableDistFunc", {"CDF_Exp", {Test Value, Mean}})

iv) *(Avenue*): theProb = av.Run("KappaStats.CDF_Exp", {Test Value, Mean})

c) Function: Exponential CDF $= 1 - e^{-\frac{x}{\beta}}$

where: x = Test value, β = Mean (or Scale Parameter)

6. CDF_F: This function returns the probability that the Test Value ≤ *X*, assuming an *F* distribution with the specified Degrees of Freedom. The *F* distribution is the ratio of two *Chi-Square* distributions with ratios v_1 and v_2 respectively.

a) Parameters:

i) Test Value: Number ≥ 1

ii) 1st Degrees of Freedom: Number > 1

iii) 2nd Degrees of Freedom: Number > 1

b) Usages:

i) From "Probability Distribution Calculator", select "Cumulative Probability (CDF)" and *F* distribution.

ii) (*Avenue*): theProb = av.Run("KappaStats.DistFunc", {"CDF_F", {Test Value, DF1, DF2}})

iii) (*Avenue*): theProb = av.Run("KappaStats.TableDistFunc", {"CDF_F", {Test Value, DF1, DF2}})

iv) *(Avenue*): theProb = av.Run("KappaStats.CDF_F", {Test Value, DF1, DF2})

$$\text{F CDF} = 1 - I\left(k, \frac{v_1}{2}, \frac{v_2}{2}\right)$$

$$\text{where: } k = \left(\frac{v_2}{v_2 + v_1 y}\right)$$

c) Function: and: y = Test Value, S_1 = Shape 1, S_2 = Shape 2

$$\text{and: } I(x,a,b) \equiv \frac{B_x(a,b)}{B(a,b)} \equiv \frac{1}{B(a,b)} \int_0^x t^{a-1}(1-t)^{b-1} dt$$

$$\text{and: } B(a,b) = \int_0^1 t^{a-1}(1-t)^{b-1} dt$$

(From Croarkin & Tobias, Date Unknown; Press et al, 1997)

7. CDF_Logistic: This function returns the probability that the Test Value ≤ *X*, assuming a *Logistic* distribution with the specified mean and scale.

a) Parameters:

i) Test Value: Number

ii) Mean: Number

iii) Scale: Number > 0

b) Usages:

i) From "Probability Distribution Calculator", select "Cumulative Probability (CDF)" and *Logistic* distribution.

ii) (*Avenue*): theProb = av.Run("KappaStats.DistFunc", {"CDF_Logistic", {Test Value, Mean, Scale}})

iii) (*Avenue*): theProb = av.Run("KappaStats.TableDistFunc", {"CDF_Logistic", {Test Value, Mean, Scale}})

iv) *(Avenue*): theProb = av.Run("KappaStats.CDF_Logistic", {Test Value, Mean, Scale})

c) Function: $\text{Logistic CDF} = \dfrac{1}{1 + \exp\left(\dfrac{A - y}{B}\right)}$

where: y = Test Value, A = Mean, B = Scale

8. **CDF_LogNormal:** This function returns the probability that the Test Value ≤ *X*, assuming a *LogNormal* distribution with the specified mean and scale. A *LogNormal* distribution occurs when natural logarithms of variable *X* are normally distributed. The *Standard LogNormal Distribution* is that with Mean = 0 and Scale = 1. The *2-Parameter LogNormal Distribution* is that with Mean = 0.

   a) Parameters:

      i) Test Value: Number ≥ 0

      ii) Mean: Number > 0

      iii) Scale: Number > 0

   b) Usages:

      i) From "Probability Distribution Calculator", select "Cumulative Probability (CDF)" and *LogNormal* distribution.

      ii) (*Avenue*): theProb = av.Run("KappaStats.DistFunc", {"CDF_LogNormal, {Test Value, Mean, Scale}})

      iii) (*Avenue*): theProb = av.Run("KappaStats.TableDistFunc", {"CDF_LogNormal, {Test Value, Mean, Scale}})

      iv) *(Avenue*): theProb = av.Run("KappaStats.CDF_LogNormal", {Test Value, Mean, Scale})

   c) Function: $$\text{LogNormal CDF} = \Phi\left(\frac{\ln(y) - A}{B}\right)$$

   where: y = Test Value, A = Mean, B = Scale

   and: $\Phi(x)$ = Cumulative Distribution Function of the Normal Distribution

9. **CDF_Normal_Simpsons:** This function returns the probability that the Test Value ≤ *X*, assuming a *Normal* distribution with the specified mean and standard deviation. Because the formula for this function does not exist in a closed form, it must be computed numerically. This script uses the *Simpson's* approximation method (Stewart 1998, p. 421-424) to calculate a highly accurate estimate of the Normal cumulative distribution function (accuracy to > 12 decimal places). The *Standard Normal Distribution* is that with Mean = 0 and Standard Deviation = 1.

   a) Parameters:

      i) Test Value: Number

      ii) Mean: Number

      iii) Standard Deviation: Number > 0

   b) Usages:

      i) From "Probability Distribution Calculator", select "Cumulative Probability (CDF)" and *Normal* distribution.

      ii) (*Avenue*): theProb = av.Run("KappaStats.DistFunc", {"CDF_Normal_Simpsons, {Test Value, Mean, St. Dev.}})

      iii) (*Avenue*): theProb = av.Run("KappaStats.TableDistFunc", {"CDF_Normal_Simpsons, {Test Value, Mean, St. Dev.}})

      iv) *(Avenue*): theProb = av.Run("KappaStats.CDF_Normal_Simpsons", {Test Value, Mean, St. Dev.})

c) Function: $\text{Normal CDF} = \Phi\left(\frac{y-A}{B}\right)$

where: y = Test Value, A = Mean, B = Scale

and: $\Phi(x)$ = Cumulative Distribution Function of the Normal Distribution

10. CDF_Poisson: This function returns the probability that the specified Number of Events will be ≤ *X*, assuming a *Poisson* distribution with the specified mean.

a) Parameters:

i) # Events: Integer ≥ 0

ii) Mean: Number > 0

b) Usages:

i) From "Probability Distribution Calculator", select "Cumulative Probability (CDF)" and *Poisson* distribution.

ii) (*Avenue*): theProb = av.Run("KappaStats.DistFunc", {"CDF_Poisson, {# Events, Mean}})

iii) (*Avenue*): theProb = av.Run("KappaStats.TableDistFunc", {"CDF_Poisson, {# Events, Mean}})

iv) *(Avenue*): theProb = av.Run("KappaStats.CDF_Poisson", {# Events, Mean})

$$\text{Poisson CDF} = \frac{\gamma(y,A)}{\Gamma(y)}$$

c) Function: where: y = Test value, A = Expectation (mean)

and: $\Gamma(x) = \int_0^\infty t^{x-1}e^{-1}dt$

and: $\gamma(x,y) = \int_0^y t^{x-1}e^{-1}dt$

11. CDF_StudentsT: This function returns the probability that the Test Value ≤ *X*, assuming a *Students T* distribution with the specified Degrees of Freedom. A *Student's T* distribution with 1df is a *Cauchy* Distribution, and it approaches a *Normal* distribution when DF>30. Various sources recommend using the *Normal* distribution if DF>100.

a) Parameters:

i) Test Value: Number

ii) Degrees of Freedom: Number > 0

b) Usages:

i) From "Probability Distribution Calculator", select "Cumulative Probability (CDF)" and *Student's T* distribution.

ii) (*Avenue*): theProb = av.Run("KappaStats.DistFunc", {"CDF_StudentsT, {Test Value, DF}})

iii) (*Avenue*): theProb = av.Run("KappaStats.TableDistFunc", {"CDF_StudentsT, {Test Value, DF}})

iv) *(Avenue*): theProb = av.Run("KappaStats.CDF_StudentsT", {Test Value, DF})

c) Function: The CDF_StudentsT T Function is dependent on whether the test value is positive or negative:

$$\text{Student's T CDF} = \begin{cases} \frac{1}{2} I\left(\frac{v}{v+y^2}, \frac{v}{2}, \frac{1}{2}\right), & t \equiv y \le 0 \\ 1 - \frac{1}{2} I\left(\frac{v}{v+y^2}, \frac{v}{2}, \frac{1}{2}\right), & t \equiv y > 0 \end{cases}$$

where: $y = \text{Test Value}$, $v = \text{Degrees of Freedom}$

and: $I(x,a,b) \equiv \frac{B_x(a,b)}{B(a,b)} \equiv \frac{1}{B(a,b)} \int_0^x t^{a-1}(1-t)^{b-1} dt$

and: $B(a,b) = \int_0^1 t^{a-1}(1-t)^{b-1} dt$

12. CDF_Weibull: This function returns the probability that the Test Value ≤ *X*, assuming a *Weibull* distribution with the specified Location, Scale and Shape parameters. The *Standardized Weibull Distribution* is that with *Location = 0* and *Scale = 1*. The *2-Parameter Weibull Distribution* is that with *Location = 0*.

a) Parameters:

i) Test Value: Number > Location

ii) Location: Number

iii) Scale: Number > 0

iv) Shape: Number > 0

b) Usages:

i) From "Probability Distribution Calculator", select "Cumulative Probability (CDF)" and *Weibull* distribution.

ii) (*Avenue*): theProb = av.Run("KappaStats.DistFunc", {"CDF_Weibull, {Test Value, Location, Scale, Number}})

iii) (*Avenue*): theProb = av.Run("KappaStats.TableDistFunc", {"CDF_Weibull, {Test Value, Location, Scale, Number}})

iv) *(Avenue*): theProb = av.Run("KappaStats.CDF_Weibull", {Test Value, Location, Scale, Number})

c) Function: $\text{Weibull CDF} = 1 - \exp\left(-\left(\frac{y-A}{B}\right)^C\right)$

where: $y = \text{Test Value}$, $A = \text{Location}$, $B = \text{Scale}$, $C = \text{Shape}$

Quantiles (also referred to as Inverse Density Functions or Percent Point Functions).

1. IDF_Beta: This function takes the specified probability and returns the value *X*, such that *P*(*X*) = *P-value*, given the Beta distribution with the two specified Shape parameters. Because the formula for this function does not exist in a closed form, it must be computed numerically. This script arrives at the *X*-value through an iterative process, repeatedly testing *X*-values with the *CDF_Beta* function until it arrives at *P-value* that is within $1x10^{-12}$ units from the specified *P-value* (this usually takes between 30-60 iterations).

a) Parameters:

i) P-value: Number ($0 \geq p \geq 1$)

ii) Shape1: Number > 0

iii) Shape2: Number > 0

b) Usages:

i) From "Probability Distribution Calculator", select "Quantile (IDF; Inverse CDF)" and *Beta* distribution.

ii) (*Avenue*): theX = av.Run("KappaStats.DistFunc", {"IDF_Beta, {P-value, Shape1, Shape2}})

iii) (*Avenue*): theX = av.Run("KappaStats.TableDistFunc", {"IDF_Beta, {P-value, Shape1, Shape2}})

iv) *(Avenue*): theX = av.Run("KappaStats.IDF_Beta", {P-value, Shape1, Shape2})

c) Function:

$$\text{Beta IDF} = \int_0^y \frac{\Gamma(S_1 + S_2)}{\Gamma(S_1)\Gamma(S_2)} y^{S_1 - 1}(1-y)^{S_2 - 1}$$

where: y = Test Value, S_1 = Shape 1, S_2 = Shape 2

and: $\Gamma(x) = \int_0^\infty t^{x-1} e^{-1} dt$

2. IDF_Binomial: This function takes the specified probability and returns the value *X* such that the Probability of getting ($X - 1$) successes ≤ the Specified Probability. This function takes an iterative approach to finding the correct *X* value, repeatedly trying different values of *X* until it reaches the correct one. This iterative process rarely takes more than 25 repetitions.

a) Parameters:

i) P-value = Number ($0 \geq p \geq 1$)

ii) # Trials = Integer ≥ 2

iii) Probability of Success = Number ($0 \geq p \geq 1$)Usages:

b) Usages:

i) From "Probability Distribution Calculator", select "Quantile (IDF; Inverse CDF)" and *Binomial* distribution.

ii) (*Avenue*): theX = av.Run("KappaStats.DistFunc", {"IDF_Binomial, {P-value, NumTrials, Probability of Success}})

iii) (*Avenue*): theX = av.Run("KappaStats.TableDistFunc", {"IDF_Binomial, {P-value, NumTrials, Probability of Success}})

iv) *(Avenue*): theX = av.Run("KappaStats.IDF_Binomial", {P-value, NumTrials, Probability of Success})

Binomial IDF: Iterative Process, repeatedly testing values of *y*, such that:

c) Function:

$$p = \sum_{i=1}^{y} \left(\frac{B!}{i!(B-i)!} \right) A^i (1-A)^{B-i}$$

where: y = #Successes, A = Probability of Success, B = #Trials

Until: $P(y-1) \leq$ User-Specified probability

3. **IDF_Cauchy:** This function takes the specified probability and returns the value *X*, such that *P*(*X*) = *P-value*, given the Cauchy distribution with the specified location and scale parameters. The *Standardized Cauchy distribution* has *Location = 0* and *Scale = 1.*

   a) Parameters:

      i) P-value: Number ($0 \geq p \geq 1$)

      ii) Location: Number

      iii) Scale: Number > 0

   b) Usages:

      i) From "Probability Distribution Calculator", select "Quantile (IDF; Inverse CDF)" and *Cauchy* distribution.

      ii) (*Avenue*): theX = av.Run("KappaStats.DistFunc", {"IDF_Cauchy, {P-value, location, Scale}})

      iii) (*Avenue*): theX = av.Run("KappaStats.TableDistFunc", {"IDF_Cauchy, {P-value, location, Scale}})

      iv) *(Avenue*): theX = av.Run("KappaStats.IDF_Cauchy", {P-value, Location, Scale})

   c) Function: Cauchy IDF $= A - \dfrac{B}{\tan(\pi p)}$

      where: $A =$ Location, $B =$ Scale, $p =$ Probability

4. **IDF_ChiSquare:** This function takes the specified probability and returns the value *X*, such that *P*(*X*) = *P-value*, given the Chi-Square distribution with the specified Degrees of Freedom. Because the formula for this function does not exist in a closed form, it must be computed numerically. This script arrives at the *X*-value through an iterative process, repeatedly testing *X*-values with the *CDF_ChiSquare* function until it arrives at *P-value* that is within $1x10^{-12}$ units from the specified *P-value* (this usually takes between 30-60 iterations). The *Chi-Square* distribution results when *v* (where *v* = Degrees of Freedom) independent variables with standard normal distributions are squared and summed (Croarkin & Tobias, Date unknown).

   a) Parameters:

      i) P-value: Number ($0 \geq p \geq 1$)

      ii) Degrees of Freedom: Number

   b) Usages:

      i) From "Probability Distribution Calculator", select "Quantile (IDF; Inverse CDF)" and *Chi-Square* distribution.

      ii) (*Avenue*): theX = av.Run("KappaStats.DistFunc", {"IDF_ChiSquare, {P-Value, F}})

      iii) (*Avenue*): theX = av.Run("KappaStats.TableDistFunc", {"IDF_ChiSquare, {P-Value, F}})

      iv) *(Avenue*): theX = av.Run("KappaStats.IDF_ChiSquare", {P-Value, DF})

c) Function: $$\text{Chi-Square IDF} = \int_0^y \frac{e^{\frac{-y}{2}} x^{\frac{v}{2}-1}}{2^{\frac{v}{2}}\Gamma\left(\frac{v}{2}\right)} dy$$

where: y = Test Value, S_1 = Shape 1, S_2 = Shape 2

and: $\Gamma(x) = \int_0^\infty t^{x-1} e^{-1} dt$

5. IDF_Exp: This function takes the specified probability and returns the value *X*, such that *P*(*X*) = *P-value*, given the Exponential distribution with the specified mean. This script uses the 1-parameter version of the equation (i.e. assuming *Location* = 0). The *Standard Exponential Distribution* is that which has *Mean = 1*.

a) Parameters:

i) P-value: Number ($0 \geq p \geq 1$)

ii) Mean: Number > 0

b) Usages:

i) From "Probability Distribution Calculator", select "Quantile (IDF; Inverse CDF)" and *Exponential* distribution.

ii) (*Avenue*): theX = av.Run("KappaStats.DistFunc", {"IDF_Exp, {P-value, Mean}})

iii) (*Avenue*): theX = av.Run("KappaStats.TableDistFunc", {"IDF_Exp, {P-value, Mean}})

iv) *(Avenue*): theX = av.Run("KappaStats.IDF_Exp", {P-value, Mean})

c) Function: $$\text{Exponential IDF} = -\beta \ln(1-p)$$

where: β = Mean (or Scale Parameter)

and: p = Specified Probability

6. IDF_F: This function takes the specified probability and returns the value *X*, such that *P*(*X*) = *P-value*, given the *F* distribution with the specified Degrees of Freedom. Because the formula for this function does not exist in a closed form, it must be computed numerically. This script arrives at the *X*-value through an iterative process, repeatedly testing *X*-values with the *CDF_F* function until it arrives at *P-value* that is within $1x10^{-12}$ units from the specified *P-value* (this usually takes between 30-60 iterations). The *F* distribution is the ratio of two *Chi-Square* distributions with ratios v_1 and v_2 respectively.

a) Parameters:

i) Test Value: Number ≥ 1

ii) 1st Degrees of Freedom: Number > 1

iii) 2nd Degrees of Freedom: Number > 1

b) Usages:

i) From "Probability Distribution Calculator", select "Quantile (IDF; Inverse CDF)" and *F* distribution.

ii) (*Avenue*): theX = av.Run("KappaStats.DistFunc", {"IDF_F, {P-value, DF1, DF2}})

iii) (*Avenue*): theX = av.Run("KappaStats.TableDistFunc", {"IDF_F, {P-value, DF1, DF2}})

iv) *(Avenue*): theX = av.Run("KappaStats.IDF_F", {P-value, DF1, DF2})

c) Function: $$\text{F IDF} = \int_0^x \left(\frac{\Gamma\left(\frac{v_1+v_2}{2}\right)\left(\frac{v_1}{v_2}\right)^{\frac{v_1}{2}} x^{\frac{v_1}{2}-1}}{\Gamma\left(\frac{v_1}{2}\right)\Gamma\left(\frac{v_1}{2}\right)\left(1+\frac{v_1 x}{v_2}\right)^{\frac{v_1+v_2}{2}}} \right) dx$$

where: $x = \text{Test Value}$, $v_1 = \text{DF1}$, $v_2 = \text{DF2}$

and: $\Gamma(x) = \int_0^\infty t^{x-1} e^{-1} dt$

7. IDF_Logistic: This function takes the specified probability and returns the value *X*, such that *P*(*X*) = *P-value*, given the *Logistic* distribution with the specified mean and scale parameters.

a) Parameters:

i) P-value: Number ($0 \geq p \geq 1$)

ii) Mean: Number

iii) Scale: Number > 0

b) Usages:

i) From "Probability Distribution Calculator", select "Quantile (IDF; Inverse CDF)" and *Logistic* distribution.

ii) (*Avenue*): theX = av.Run("KappaStats.DistFunc", {"IDF_Logistic, {P-value, Mean, Scale}})

iii) (*Avenue*): theX = av.Run("KappaStats.TableDistFunc", {"IDF_Logistic, {P-value, Mean, Scale}})

iv) *(Avenue*): theX = av.Run("KappaStats.IDF_Logistic", {P-value, Mean, Scale})

c) Function: $\text{Logistic IDF} = A + B \ln\left(\frac{p}{1-p}\right)$

where: $p = \text{Probability}$, $A = \text{Mean}$, $B = \text{Scale}$

8. IDF_LogNormal: This function takes the specified probability and returns the value *X*, such that *P*(*X*) = *P-value*, given the *LogNormal* distribution with the specified mean and scale parameters. Because the formula for this function does not exist in a closed form, it must be computed numerically. This script arrives at the *X*-value through an iterative process, repeatedly testing *X*-values with the *CDF_LogNormal* function until it arrives at *P-value* that is within $1x10^{-12}$ units from the specified *P-value* (this usually takes between 30-60 iterations). A *LogNormal* distribution occurs when natural logarithms of variable *X* are normally distributed. The *Standard LogNormal Distribution* is that with Mean = 0 and Scale = 1. The *2-Parameter LogNormal Distribution* is that with Mean = 0.

a) Parameters:

i) P-value: Number ($0 \geq p \geq 1$)

ii) Mean: Number > 0

iii) Scale: Number > 0

b) Usages:

i) From "Probability Distribution Calculator", select "Quantile (IDF; Inverse CDF)" and *LogNormal* distribution.

ii) (*Avenue*): theX = av.Run("KappaStats.DistFunc", {"IDF_LogNormal, {P-value, Mean, Scale}})

iii) (*Avenue*): theX = av.Run("KappaStats.TableDistFunc", {"IDF_LogNormal, {P-value, Mean, Scale}})

iv) *(Avenue*): theX = av.Run("KappaStats.IDF_LogNormal", {P-value, Mean, Scale})

c) Function: LogNormal IDF $= \int_0^y \left(\frac{1}{B\sqrt{2\pi}} \exp\left(-\frac{1}{2}\left(\frac{\ln(y) - A}{B} \right)^2 \right) \right) dy$

where: y = Test Value, A = Mean, B = Scale

9. IDF_Normal: This function takes the specified probability and returns the value *X*, such that *P*(*X*) = *P-value*, given the *Normal* distribution with the specified mean and standard deviation. Because the formula for this function does not exist in a closed form, it must be computed numerically. This script arrives at the *X*-value through an iterative process, repeatedly testing *X*-values with the *CDF_Normal_Simpsons* function until it arrives at *P-value* that is within $1x10^{-12}$ units from the specified *P-value* (this usually takes between 30-60 iterations). Furthermore, there is no closed formula for calculating the Normal cumulative distribution function, so this script uses the *Simpson's* approximation method (Stewart 1998, p. 421-424) to calculate a highly accurate estimate (accuracy to > 12 decimal places). The *Standard Normal Distribution* is that with Mean = 0 and Standard Deviation = 1.

a) Parameters:

i) P-value: Number ($0 \geq p \geq 1$)

ii) Mean: Number

iii) Standard Deviation: Number > 0

b) Usages:

i) From "Probability Distribution Calculator", select "Quantile (IDF; Inverse CDF)" and *Normal* distribution.

ii) (*Avenue*): theX = av.Run("KappaStats.DistFunc", {"IDF_Normal, {P-value, Mean, St. Dev.}})

iii) (*Avenue*): theX = av.Run("KappaStats.TableDistFunc", {"IDF_Normal, {P-value, Mean, St. Dev.}})

iv) *(Avenue*): theX = av.Run("KappaStats.IDF_Normal", {P-value, Mean, St. Dev.})

c) Function: Normal IDF $= \int_0^y \left(\frac{1}{B\sqrt{2\pi}} \exp\left(-\frac{1}{2}\left(\frac{y - A}{B} \right)^2 \right) \right) dy$

where: y = Test Value, A = Mean, B = Scale

10. **IDF_Poisson:** This function takes the specified probability and returns the value *X* such that the Probability of getting (*X* – 1) events ≤ the Specified Probability. This function takes an iterative

approach to finding the correct *X* value, repeatedly trying different values of *X* until it reaches the correct one.

a) Parameters:

   i) P-value: Number ($0 \geq p \geq 1$)

   ii) Mean: Number > 0

b) Usages:

   i) From "Probability Distribution Calculator", select "Quantile (IDF; Inverse CDF)" and *Poisson* distribution.

   ii) (*Avenue*): theProb = av.Run("KappaStats.DistFunc", {"IDF_Poisson, {P-value, Mean}})

   iii) (*Avenue*): theProb = av.Run("KappaStats.TableDistFunc", {"IDF_Poisson, {P-value, Mean}})

   iv) *(Avenue*): theProb = av.Run("KappaStats.IDF_Poisson", {P-value, Mean})

Poisson IDF: Iterative Process, repeatedly testing values of *y*, such that:

$$p = \frac{\gamma(y, A)}{\Gamma(y)}$$

c) Function: where: $y =$ Test value, $A =$ Expectation (mean)

and: $\Gamma(x) = \int_0^\infty t^{x-1} e^{-1} dt$

and: $\gamma(x, y) = \int_0^y t^{x-1} e^{-1} dt$

Until: $P(y-1) \leq$ User-Specified probability

11. IDF_StudentsT: This function takes the specified probability and returns the value *X*, such that *P*(*X*) = *P-value*, given the *Student's T* distribution with the specified Degrees of Freedom. Because the formula for this function does not exist in a closed form, it must be computed numerically. This script arrives at the *X*-value through an iterative process, repeatedly testing *X*-values with the *CDF_StudentsT* function until it arrives at *P-value* that is within $1x10^{-12}$ units from the specified *P-value* (this usually takes between 30-60 iterations). A *Student's T* distribution with 1df is a *Cauchy* Distribution, and it approaches a *Normal* distribution when DF>30. Various sources (esp. McLaughlin 2001) recommend using the *Normal* distribution if DF>100.

a) Parameters:

   i) P-value: Number ($0 \geq p \geq 1$)

   ii) Degrees of Freedom: Number > 0

b) Usages:

   i) From "Probability Distribution Calculator", select "Quantile (IDF; Inverse CDF)" and *Student's T* distribution.

   ii) (*Avenue*): theX = av.Run("KappaStats.DistFunc", {"IDF_StudentsT, {P-value, DF}})

   iii) (*Avenue*): theX = av.Run("KappaStats.TableDistFunc", {"IDF_StudentsT, {P-value, DF}})

   iv) *(Avenue*): theX = av.Run("KappaStats.IDF_StudentsT", {P-value, DF})

c) Function: $\text{Student's T IDF} = \int_{-\infty}^{y} \frac{\Gamma\left(\frac{v+1}{2}\right)}{\sqrt{\pi v}\,\Gamma\left(\frac{v}{2}\right)}\left[1+\frac{y^2}{v}\right]^{-\frac{v+1}{2}} dy$

where: $y = \text{Test Value}$, $v = \text{Degrees of Freedom}$

and: $\Gamma(x) = \int_0^{\infty} t^{x-1}e^{-1}dt$

12. IDF_Weibull: This function takes the specified probability and returns the value *X*, such that *P*(*X*) = *P-value*, given the *Weibull* distribution with the specified Location, Scale and Shape parameters. The *Standardized Weibull Distribution* is that with *Location = 0* and *Scale = 1*. The *2-Parameter Weibull Distribution* is that with *Location = 0*.

a) Parameters:

i) Test Value: Number > Location

ii) Location: Number

iii) Scale: Number > 0

iv) Shape: Number > 0

b) Usages:

i) From "Probability Distribution Calculator", select "Quantile (IDF; Inverse CDF)" and *Weibull* distribution.

ii) (*Avenue*): theX = av.Run("KappaStats.DistFunc", {"IDF_Weibull, {P-value, Location, Scale, Number}})

iii) (*Avenue*): theX = av.Run("KappaStats.TableDistFunc", {"IDF_Weibull, {P-value, Location, Scale, Number}})

iv) *(Avenue*): theX = av.Run("KappaStats.IDF_Weibull", {P-value, Location, Scale, Number})

c) Function: $\text{Weibull IDF} = A + B\sqrt[C]{-\ln p}$

where: $p = \text{Probability}$, $A = \text{Location}$, $B = \text{Scale}$, $C = \text{Shape}$

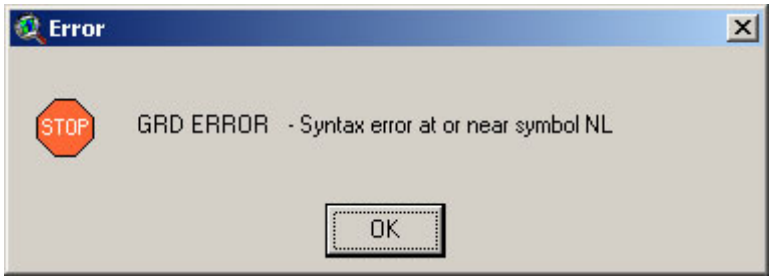
Additional Menu Functions:

Generating Separate Classification Tables:

This extension provides four options for generating stand-alone classification table based on a shapefile of sample points and a grid or polygon classification theme. These classification tables may be useful for general statistical purposes, but there are two additional important situations that may also lead you to generate stand-alone tables:

1) *Kappa Analysis runs faster:* Extracting the classification data from polygons or grids, especially from circular neighborhoods, can be time-consuming. In general the Kappa analysis will always run much faster if the data is available in the table. If you plan on running the analysis more than once, it may be advantageous to add the data to a table before you start.

2) *Large Numbers of Grid Requests Crash ArcView:* ArcView Spatial Analyst has a bug, which triggers a crash after approximately 32,500 grid operations in a single ArcView session. If you have a sample point theme with > 32,500 points, or if you use a circular neighborhood or conduct multiple analyses which require > 32,500 cell value requests, ArcView will crash with the message:



Please refer to the *Troubleshooting* section on p. 79 for more information. If you are encountering this problem, you can avoid it by the following steps:

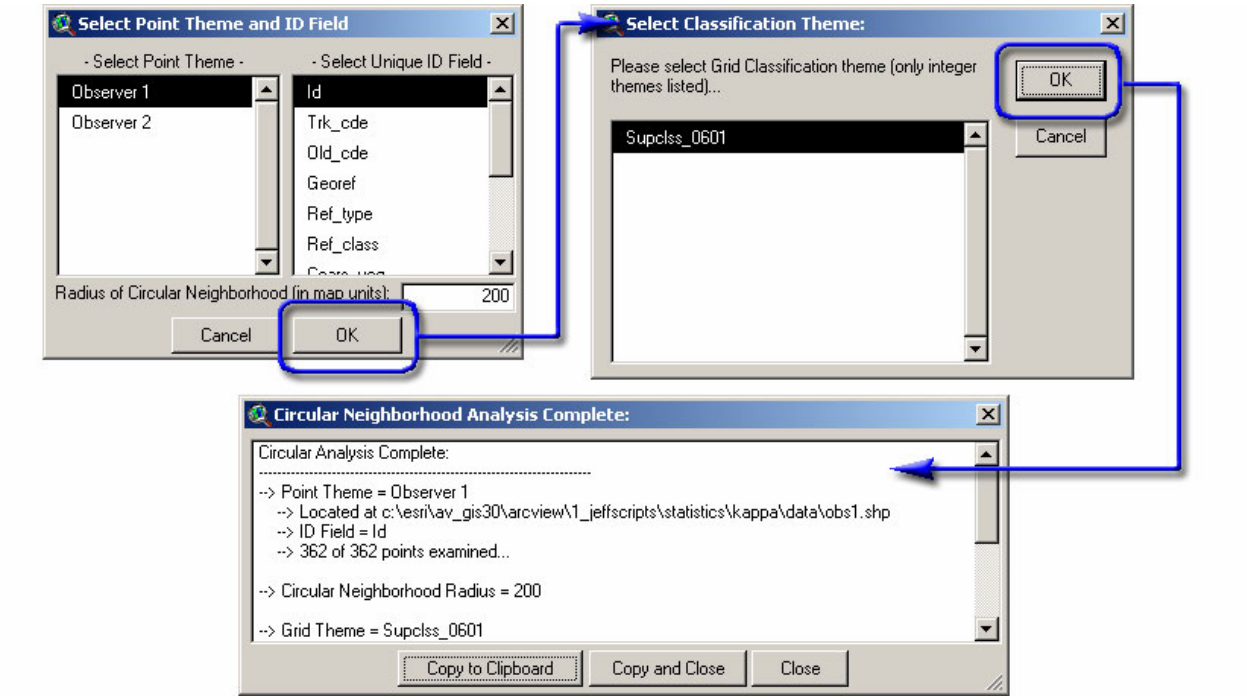
a) Restart ArcView. Spatial Analyst is unstable after you see this error message and needs to be restarted.

b) Select a series of subsets of your sample points and generate classification tables for each subset. If ArcView crashes again while you do this, restart ArcView and pick up where you left off.

c) After you have generated your data subsets, link each of them to your original sample point theme and transfer the class values to a field in the point theme attribute table (see *Linking and Joining Tables* on p. 77). Make sure you unlink each table before you move on to the next one.

Generating Class Values from Circular Region: Grid Source

This function allows you to extract a classification value from all grid cells intersected by a circular region around each sample point (refer to *Adjusting for Locational Uncertainty* on p. 13 for a discussion of what this does and why you may want to do it). This function will work on only the selected set of points (if any are selected), or all points (if none are selected). To run this function, click the menu item and identify your point theme and unique ID field:



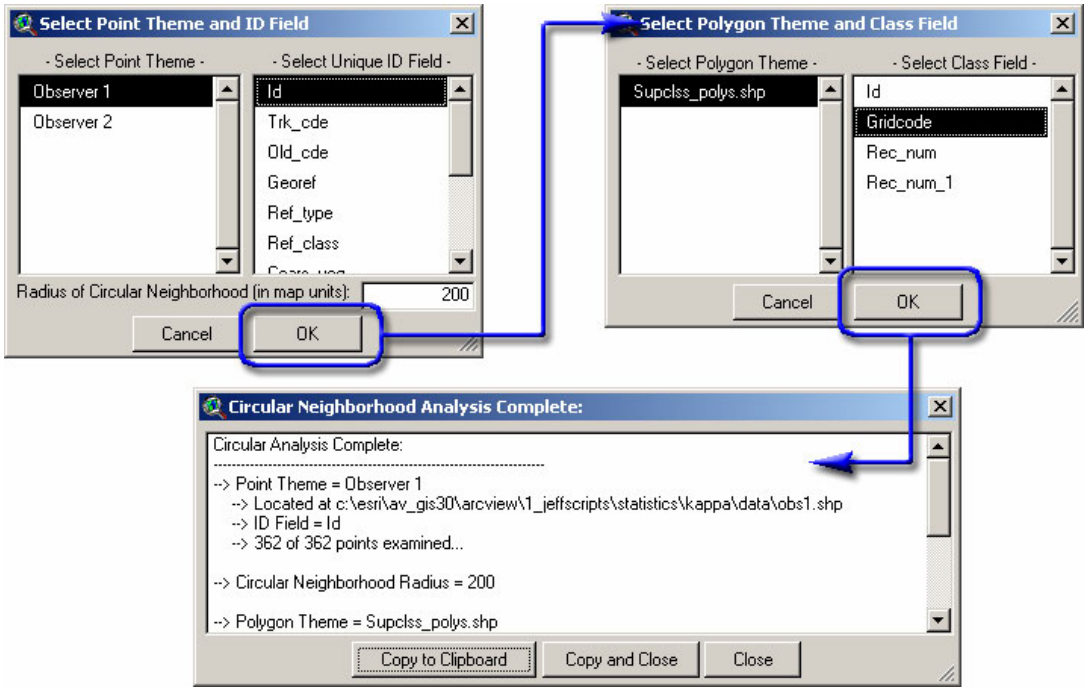
Upon completion, this function will add a table to your project that looks something like the following:

circular_classes25.dbf

Unique_id	Id	Class_val	Comment
0	3.00000	3	Class 3: Area = 100.00% of circle
1	9.00000	3	Class 3: Area = 59.14% of circle
2	76.00000	1	Class 1: Area = 96.94% of circle
3	119.00000	4	Class 4: Area = 67.17% of circle
4	142.00000	1	Class 1: Area = 100.00% of circle
5	156.00000	1	Class 1: Area = 95.44% of circle
6	157.00000	1	Class 1: Area = 74.69% of circle
7	165.00000	3	Class 3: Area = 98.42% of circle
8	176.00000	3	Class 3: Area = 60.20% of circle
9	182.00000	3	Class 3: Area = 54.65% of circle
10	191.00000	3	Class 3: Area = 84.39% of circle

Generating Class Values from Circular Region: Polygon Source

This function allows you to extract a classification value from all polygons intersected by a circular region around each sample point (refer to *Adjusting for Locational Uncertainty* on p. 13 for a discussion of this procedure). This function will work on only the selected set of points (if any are selected), or all points (if none are selected). Click the menu item and identify your point theme and unique ID field:



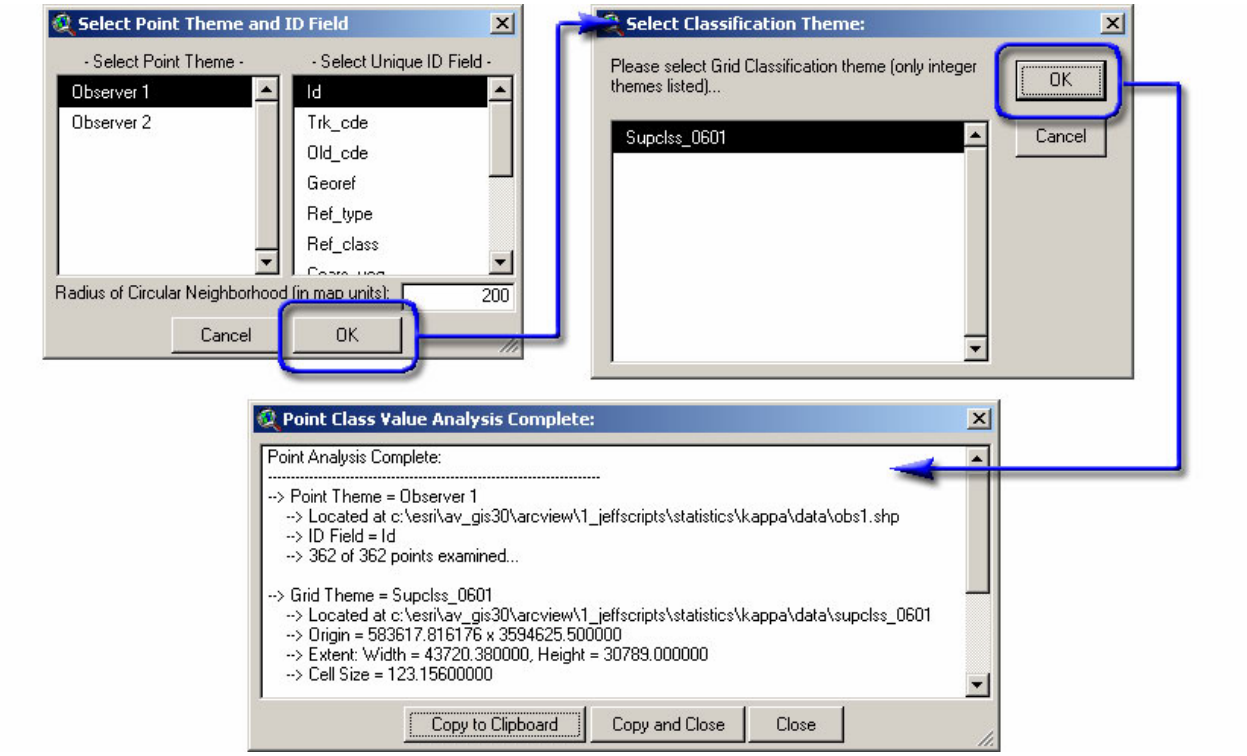
Upon completion, this function will add a table to your project that looks something like the following:

circular_classes26.dbf

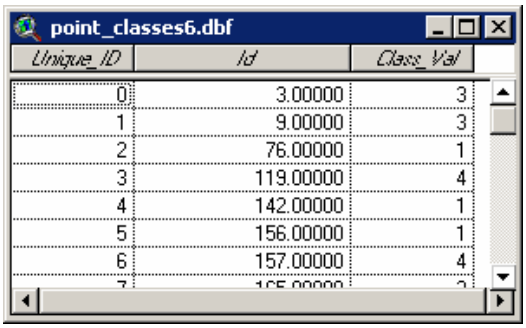
Unique_id	Id	Class_val	Comment
0	3.00000	3	Class 3: Area = 100.00% of circle
1	9.00000	3	Class 3: Area = 58.11% of circle
2	76.00000	1	Class 1: Area = 97.10% of circle
3	119.00000	4	Class 4: Area = 67.11% of circle
4	142.00000	1	Class 1: Area = 100.00% of circle
5	156.00000	1	Class 1: Area = 98.02% of circle
6	157.00000	1	Class 1: Area = 82.16% of circle
7	165.00000	3	Class 3: Area = 99.17% of circle
8	176.00000	3	Class 3: Area = 73.29% of circle
9	182.00000	3	Class 3: Area = 61.54% of circle
10	191.00000	3	Class 3: Area = 88.78% of circle

Generating Class Values from Point: Grid Source

This function allows you to extract a classification value from a grid cell by intersecting the sample point. This output is similar to what one may find using the standard ArcView Zonal Statistics function with a Zone field containing unique ID values. This function will work on only the selected set of points (if any are selected), or all points (if none are selected). Click the menu item and identify your point theme and unique ID field:



Upon completion, this function will add a table to your project that looks something like the following:

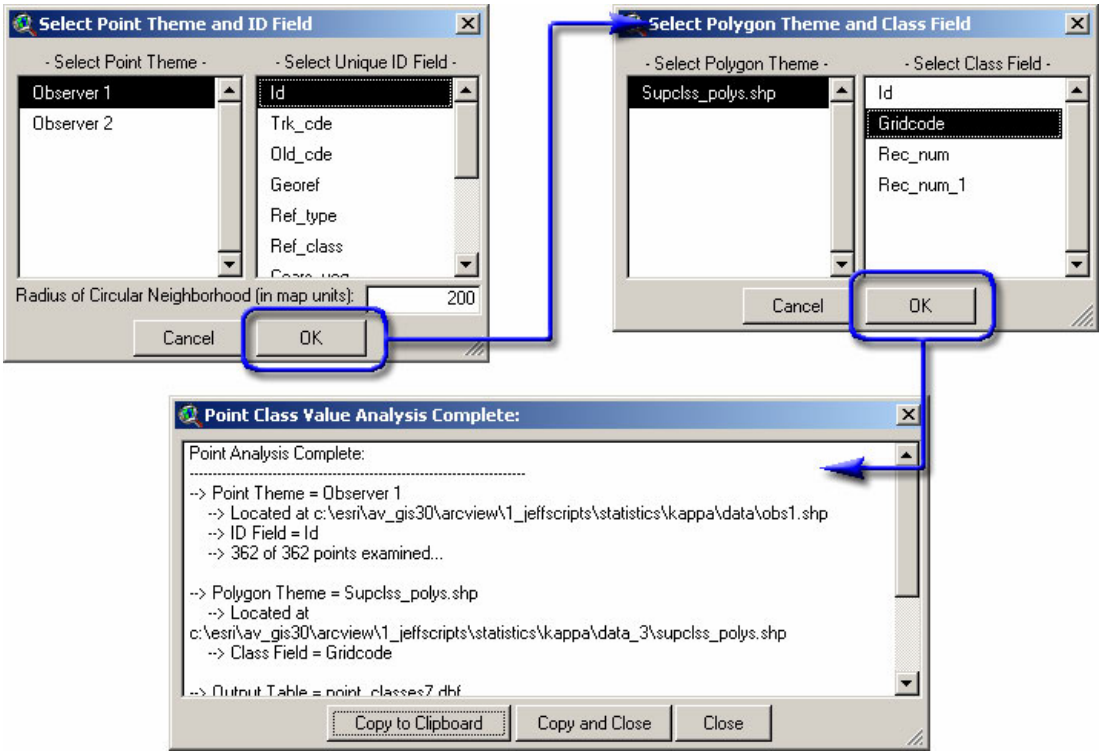


point_classes6.dbf

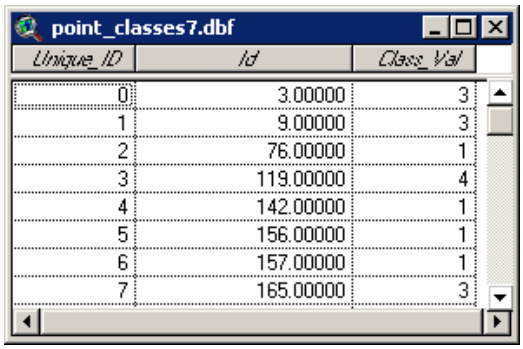
Unique_ID	Id	Class_Val
0	3.00000	3
1	9.00000	3
2	76.00000	1
3	119.00000	4
4	142.00000	1
5	156.00000	1
6	157.00000	4

Generating Class Values from Circular Point: Polygon Source

This function allows you to extract a classification value from the polygon by intersecting it with the sample point. The output is similar to what you might get if you do a spatial join on the polygon and point attribute fields. IMPORTANT: This function assumes there are no overlapping polygons (which should be the case in a properly-classified polygon theme). If there are multiple polygons at any particular point, this extension will only extract the classification value from the first one it encounters. This function will work on only the selected set of points (if any are selected), or all points (if none are selected). Click the menu item and identify your point theme and unique ID field:



Upon completion, this function will add a table to your project that looks something like the following:



point_classes7.dbf

Unique_ID	Id	Class_Val
0	3.00000	3
1	9.00000	3
2	76.00000	1
3	119.00000	4
4	142.00000	1
5	156.00000	1
6	157.00000	1
7	165.00000	3

Linking and Joining Classification Tables with Sample Point Theme:

If you use the standard Table Link button  to link your classification table with your point theme attribute table (using the unique ID Field from the attribute table and the equivalent field from the classification table; see the standard ArcView documentation on linking tables for more info), then all the classification table fields will appear in the Kappa Input Data dialog and may be used in the analysis. If you wish to transfer the classification data to the sample point theme permanently, do the following:

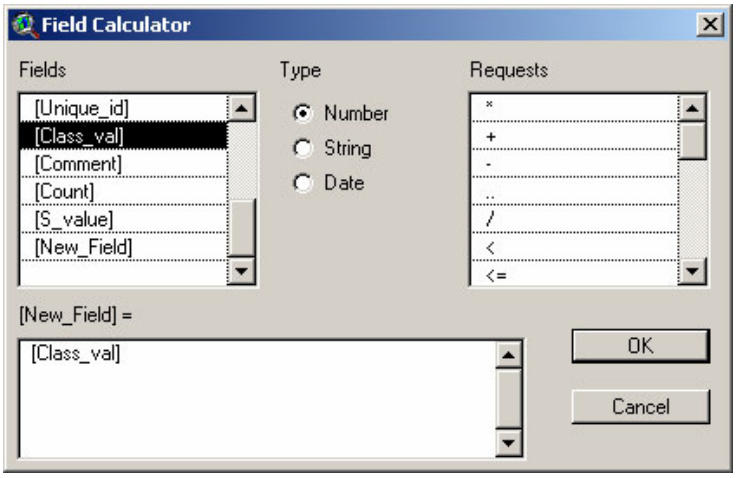
1) After the tables have been linked, set your point theme attribute table to *Editable* by clicking the "Table" menu, then "Start Editing".

2) Add a new field to your attribute table by clicking the "Edit" menu, then "Add Field". Make sure your new field is the correct type (i.e., don't make it a numeric field if your classification values are strings).

3) Clear any current selection in your attribute table by clicking the Clear Selection button  .

4) Select your new field by clicking on the field name at the top of the table. It should have an inset appearance.

5) Click the Calculate button  to open the Field Calculator and transfer the data. If your newly-created classification field was named [New_Field] and your class field from your joined classification table was named [Class_val], then the Field Calculator dialog would be filled out as follows:



6) Save edits by clicking the "Table" menu, then "Stop Editing".

7) If you wish, you may unjoin your tables by clicking the "Table" menu, then "Remove All Joins".

Generating Unique ID Values:

This extension includes several functions for identifying grid or polygon values at sample points. Each of these functions creates a separate table containing sample point ID values plus the grid or polygon theme classification value. Because these functions create separate tables rather than modifying the original sample point shapefile, you will need some unique ID value in your sample point attribute table.

In most cases, you will already have such a field available. In the rare case that a unique identifier field is lacking, you can quickly generate one using the menu item "Add Unique Record Number Field" under the "Kappa Tools" menu. This option will only be available if you have a single feature theme (point, line or polygon) active in your view. After clicking this option, a message will appear similar to the following:

IMPORTANT: This function will modify your original dataset and there is no "Undo" option. The change is permanent.

Troubleshooting:

If you encounter some strange crash, please click the menu item “Check Kappa Statistic Scripts” in either the View, Table or Project Help menu. Click this as soon as you are able to following the crash. With any luck, that function will generate a report with enough information for the author to find and fix the problem.

Otherwise, the problem may be found and explained below:

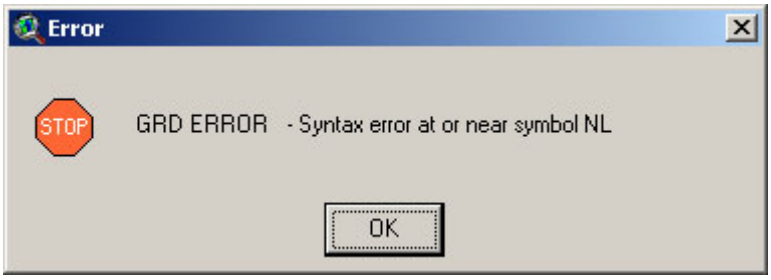
Problem: The Extension does not load when the following error message occurs:

Solution: This problem is caused by an outdated version of the Dialog Designer. For some reason, some versions of ArcView 3 were shipped with an older version of Dialog Designer which did not support this "LISTBOX_SELECTION_MULTIROW" option (which means a listbox on a dialog is set so that you can select multiple items from the list).

ESRI has a newer version of the Dialog Designer available on their website for free download. To obtain the most recent version of the Dialog Designer, point your browser to:

http://support.esri.com/index.cfm?fa=downloads.patchesServicePacks.viewPatch&PID=25&MetaID=483

Problem: Extension crashes in mid-operation, producing an obscure message stating there is a syntax error at or near symbol NL:



This is sometimes followed by the infamous “Segmentation Violation!” message:

Sometimes ArcView crashes completely and vanishes without showing these messages, while other times it vanishes after showing these messages.

Solution: There is no simple solution to this problem. It is due to a bug in Spatial Analyst which causes SA to crash after approximately 32,500 grid operations or if SA tries to hold > 50 grids in memory at one time.

We are unaware of a simple way to resolve this problem. If possible, use smaller point data sets or try to use fewer grids in your analysis. Alternatively, ArcGIS 9 is expected to run these operations without encountering this problem.

Problem: Unable to find grid in a directory, even though you know it is there.

Solution: This is probably due to a space or invalid character in the pathname. Spatial Analyst fails to recognize a grid if it lies in a folder with a space or period in it. For example, if you store your grids in the standard default Windows directory "My Documents" or even "My Docs," you will not see the grid listed in the "Add Theme" dialog. The "Add Theme" dialog will show you all the shapefiles and images, but no grids. This may be resolved by either renaming the folder (e.g., "my_docs") or moving the grid file to a different file location where it does not lie in a path with invalid characters.

Problem: You load your grids but you are unable to conduct any calculations on them (i.e., the grids aren't acting like grids).

Solution: These files may have been loaded as images. Grids may be loaded as either images or grids. If they are loaded as images, no grid functions can be performed. Remove the grid and then re-add them to you're the view by selecting "Grid Data Source" rather than "Image Data Source".

Problem: Extension crashes in mid-calculation with the message:

Solution: This error may be caused by either a corrupt INFO directory or if the working directory pathname is too long. We are unaware of the exact pathname size that triggers the error, but it is somewhere around 80 characters. If you have over 80 characters in your pathname and you see this error, then you can probably avoid it by changing your work directory to someplace closer to the root. We often create temporary GIS directories directly below the drive name (e.g., C:/temp_GIS).

References:

Abramowitz, M. and I.A. Stegun. 1972. Handbook of Mathematical Functions. Dover. 1046 pp.

Agresti, A. 1990. Categorical data analysis. John Wiley & Sons. 558 pp.

Bailey, F.M. 1923. Birds recorded from the Santa Rita Mountains in southern Arizona. Pacific Coast Avifauna 15: 60 pp.

Balda, R.P 1970. Effects of spring leaf-fall on composition and density of breeding birds in two southern Arizona woodlands. Condor 72: 325-331.

Boone, R.B. and W.B. Krohn. 2002. Modeling tools and accuracy assessment. *In* Predicting Species Occurrences: Issues of Accuracy and Scale (J. Scott, R. Heglund, M. Morrison et al.). Island Press, Washington. p. 265-270.

Brown, J.L. 1994. Mexican jay (Aphelocoma ultramarina). *In* Birds of North America 118 (A. Poole and F. Gill, eds). Academy of Natural Sciences, Philadelphia, and American Ornithologists' Union, Washington D.C. 15 pp.

Brown, J.L. and E.R. Brown. 1985. Ecological correlates of group size in a communal breeding jay. Condor 87: 309-315.

Burkardt, J. 2001. PROB - Probability Density Functions. Sample Fortran code for calculating density functions. http://www.psc.edu/~burkardt/src/prob/prob.html. Date Accessed: July 2002

Card, D.H. 1982. Using known map category marginal frequencies to improve estimates of thematic map accuracy. Photogrammetric Engineering and Remote Sensing. 48(3): 431-439.

Congalton, R.G. 1988. A comparison of sampling schemes used in generating error matrices for assessing the accuracy of maps generated from remotely sensed data. Photogrammetric Engineering and Remote Sensing. American Society of Photogrammetry and Remote Sensing. 54(5): 593-600.

Congalton, R.G. 1991. A review of assessing the accuracy of classifications of remotely sensed data. Remote Sensing of Environment. 37: 35-46

Congalton, R.G. and K. Green. 1999. Assessing the Accuracy of Remotely Sensed Data: Principles and Practices. Lewis Publishers. 137 pp.

Congalton, R.G. and R. A. Mead. A quantitative method to test for consistency and correctness in photointerpretation. Photogrammetric Engineering and Remote Sensing. American Society of Photogrammetry and Remote Sensing. 49(1): 69-74.

Croarkin, C. and P. Tobias. Date Unknown. Engineering and Statistics Handbook. National Institute of Standards and Technologyhttp://www.itl.nist.gov/div898/handbook/. Date Accessed: July 2002

Csuti, B. and P.J. Crist. 1998. Methods for assessing accuracy of animal distribution maps. *In* Gap Analysis Program report (Cassidy, K., E.O. Garton, W.B. Krohn, L.S. Mills, J.M. Scott, K. Williams and B. Csuti, editors). University of Idaho, Moscow.

Drost, C.A., S.R. Jacobs, and K.A. Thomas. 1999. Accuracy assessment of vertebrate distribution models for Arizona Gap. USGS-Colorado Plateau Research Station technical report. Flagstaff, Arizona. pp. 58.

Edwards, T.C. 1986. Ecological distribution of the Gray-breasted jay: the role of habitat. Condor 88: 456-460.

Farber, O. and R. Kadmon. 2003. Assessment of alternative approaches for bioclimatic modeling with special emphasis on the M-distance. Ecological Modeling 160: 115-130.

Fielding, A. 2002. What are the appropriate characteristics of an accuracy assessment? Pp. 271-280 in Predicting Species Occurrences: Issues of Accuracy and Scale (Scott, J. M., P. J. Heglund, and M. L. Morrison, editors). Island Press.

Fielding, A.H. and J.F. Bell. 1997. A review of methods for the assessment of prediction errors in conservation presence/absence models. Environmental Conservation 24: 38–49.

Fleiss, J.L.. 1981. Statistical methods for rates and proportions. 2nd Edition. John Wiley & Sons. 321 pp.

Hess, G.R. 1994. Pattern and error in landscape ecology: a commentary. Landscape ecology 9: 3-5.

Hess, G.R. and J.M. Bay. 1997. Generating confidence intervals for composition-based landscape indexes. Landscape Ecology 12: 309-320.

Jeffrey, A. 2000. Handbook of mathematical formulas and integrals (2nd Ed). Academic Press

Kish, L. 1965. Survey Sampling. John Wiley & Sons. 643 pp.

Logsdon, T. The navstar global positioning system. Van Nostrand Reinhold, New York. 256 pp.

Luck, M. and J. Wu 2002. A gradient analysis of urban landscape pattern: a case study from the Phoenix metropolitan region, Arizona, USA. Landscape Ecology 17: 327-339.

Lurz, P.W.W., S.P. Rushton,, L.A. Wauters, S. Bertolino, I. Currado, P. Mazzoglio, and M.D.F. Shirley. 2001. Predicting grey squirrel expansion in North Italy: a spatially explicit modeling approach. Landscape Ecology 16: 407–420.

McLaughlin, M. P. 2001. Regress+, Appendix A. A Compendium of Common Probability Distributions (version 2.3). http://www.causaScientia.org/math_stat/Dists/Compendium.pdf Date Accessed: July 2002

Ott, R. L. 1993. An Introduction to Statistical Methods and Data Analysis, Fourth Edition. Duxbury Press.

Press, W.H., Teukolsky, S.A., Vetterling W.T., and Flannery, B.P. 1988-1997. Numerical Recipes in C; the Art of Scientific Computing (2nd Ed). Cambridge University Press. (ISBN 0-521-43108-5). (http://lib-www.lanl.gov/numerical/bookcpdf.html - See Chapter 6, sections 1-4)

Slocum, T. A., R. B. McMaster, F. C. Kessler and H. H. Howard. 2005. Thematic cartography and geographic visualization. 2nd Ed. Prentice-Hall, Inc. 518 pp..

Spiegel, M.R, Schiller, J., and Srinivasan, R.A. 2000. Schaum's outline of probability and statistics. 2nd Ed. McGraw-Hill. 408 pp.

S-Plus Help Files 2001. S-Plus 6 for Windows.

http://www.insightful.com/support/splus60win/default.asp Date Accessed: July 2002

S-PLUS 6 for Windows Guide to Statistics, Volume 1, Insightful Corporation, Seattle, WA.

SPSS Help Files. 1999. SPSS for Windows Release 9.

http://www.spss.com/ Date Accessed: July 2002

Stewart, J. 1998. Calculus, Concepts and Contexts. Brooks/Cole Publishing Company. p. 421-424.

Story, M. and R.G. Congalton. 1986. Accuracy assessment: a user's perspective. Photogrammetric Engineering and Remote Sensing. American Society of Photogrammetry and Remote Sensing. 52(3) 397-399.

Tanner, J.T. and J.W. Hardy. 1958. Summer birds of the Chiricahua Mountains, Arizona. American Museum Novitates 1866, 12 pp.

Wynne, J.J. 2003. Landscape-scale modeling of vegetation land cover and songbird habitat, Pinaleños Mountains, Arizona, M.S. thesis. Northern Arizona University, Flagstaff. 105 pp. Available at http://www.jennessent.com/share/Wynne_thesis.pdf. Date Accessed: April 2004

Index

S

T

U

V

W

Z

Enjoy! If you have any questions, or find bugs in the software, please contact the authors at:

Jeff Jenness
Jenness Enterprises
3020 N. Schevene Blvd.
Flagstaff, AZ 86004
USA
jeffj@jennessent.com
http://www.jennessent.com
(928) 607-4638

USDA Forest Service
Rocky Mountain Research Station
2500 S. Pine Knoll Dr.
Flagstaff, AZ 86005
jjenness@fs.fed.us
(928) 556-2012

J. Judson Wynne
United States Geologic Survey
Southwest Biological Science Center
Colorado Plateau Research Station
2255 N. Gemini Drive
Flagstaff, AZ 86011
jwynne@usgs.gov
Fax (928) 556-7092
Tel (928) 556-7172

Updates to this extension and an on-line version of this manual are available at

http://www.jennessent.com/arcview/kappa_stats.htm