

---

# Technical Note:

## Statistical methods for accuracy assesment of classified thematic maps

---

*D G Rossiter*  
*Department of Earth Systems Analysis*  
*International Institute for Geo-information Science & Earth*  
*Observation (ITC), Enschede (NL)*

26-April-2004

### Contents

<b>1 Motivation</b>	<b>3</b>
<b>2 Binomial tests of accuracy</b>	<b>5</b>
2.1 Small samples . . . . .	6
<b>3 Multinomial tests of accuracy</b>	<b>7</b>
3.1 Naïve measures of agreement . . . . .	9
3.2 The weighted confusion matrix . . . . .	12
3.3 The <i>kappa</i> index of agreement . . . . .	15
3.4 Conditional <i>kappa</i> . . . . .	19
3.5 Weighted <i>kappa</i> . . . . .	20
3.6 The <i>tau</i> index . . . . .	23

---

Copyright© 2004 D G Rossiter. All rights reserved. Reproduction and dissemination of the work as a whole (not parts) freely permitted if this original copyright notice is included. Sale or placement on a web site where payment must be made to access this document is strictly prohibited. To adapt or translate please contact the author ([rossiter@itc.nl](mailto:rossiter@itc.nl); <http://www.itc.nl/personal/rossiter>).

<b>4</b>	<b>Comparing maps</b>	<b>25</b>
4.1	Comparing the relative accuracy of two maps . . . . .	26
4.2	Comparing the agreement between two maps with <i>kappa</i> . . . . .	27
<b>5</b>	<b>Computing accuracy in R</b>	<b>28</b>
5.1	Naïve and <i>kappa</i> . . . . .	28
5.2	<i>Tau</i> . . . . .	30
5.3	Weighted naïve and <i>kappa</i> statistics . . . . .	32
5.4	Computing exact confidence intervals of the bino- mial distribution . . . . .	34
5.5	Numerical Example . . . . .	37
5.6	Computing normal scores with R . . . . .	42
<b>6</b>	<b>Further reading</b>	<b>43</b>
	<b>References</b>	<b>44</b>

# 1 Motivation

Classified thematic maps are produced for a wide variety of resources: soil types or properties, land cover, land use, forest inventory, and many more. These maps are not very useful without quantitative statements about their accuracy. Map *users* must know the quality of the map for their intended uses, and map *producers* must evaluate the success of their mapping efforts. Both users and producers may want to compare several maps to see which is best, or to see how well they agree.

The fundamental principles of thematic map accuracy are well-presented in a review paper by Stehman and Czaplewski [31], who divide accuracy assessment into three stages:

1. the *sampling design*: how to select the *reference sample*, i.e. the locations where *ground truth* will be determined;
2. the *response design*: how to determine the ground truth for each observation in the reference sample;
3. the *estimation and analysis protocol*: how to quantify the accuracy of the map, given the samples and their responses.

In this technical note we only deal with the statistical aspects of third stage, assuming that the crucial question of what is agreement between map and ground has been determined by a suitable response design.

There is nothing particularly novel in this note. It is hoped that by bringing all the techniques together with a common notation and a running numerical example, the reader may be helped to select the appropriate statistical technique for their map accuracy problem. The accompanying computer programs (§5) may also be useful.

There are four general questions that may be asked relating to the thematic accuracy of a map:

1. What is the *error frequency*: how often (i.e. over what proportion of the mapped area) does the map not agree with reality?
2. What is the *nature* of the errors: which classes are not mapped correctly, and with which other classes are they confused?
3. What is the *magnitude* of errors: how serious are they for a

decision-maker?

4. What is the *source* of the errors: why did they occur?

Here we deal with the first three questions; the map producer can use this information to investigate the fourth.

! → **Statistical validity is based on probability sampling** All the statistical methods discussed in this note are only valid if the reference sample is obtained with a *probability sampling* design [3]. This is a sampling scheme where:

1. Each potential sampling point has a non-zero probability of being selected;
2. These probabilities are known from the design.

If some areas could never be sampled (e.g. inaccessible locations, military zones), the resulting statistical statement is not applicable to them. It may be possible to argue that unsampled areas are similar to sampled ones, but this relies on external evidence, which may be more or less convincing. Stehman [29] reviews probability sampling designs for thematic map accuracy assessment.

! → A common practice in natural resources mapping is to locate sample points either *purposively* (where we want to examine the map) or *opportunistically* (where we are able to reach). In both these cases, any statistical measure of accuracy is highly suspect, not to say *prima facie* invalid.

Most of the statistical methods here assume that *every point has an equal probability of being selected*. This is true of *simple random sampling* and *proportional stratified random sampling*, where the number of points per stratum is proportional to its mapped area. However, the formulas can be adjusted for unequal proportions, as long as these are known.

For some other common sampling designs there are complications, especially in estimating the variance or error of the accuracy estimates. In *cluster sampling* we expect spatial dependence between nearby samples [27]. In *systematic sampling*, the accuracy assessment itself is not affected, but the variance estimates are usually too high [25]. These are not considered in this note.

## 2 Binomial tests of accuracy

The simplest method of accuracy assessment is a binomial test based on the number of agreements (‘true’, ‘success’) or disagreement (‘false’, ‘failure’) of a series of ground truth samples with the mapped class, attribute, or interpretation. Confidence intervals can be determined for the accuracy of the whole map or of any stratum, for example a single legend category. This method only answers the first question posed in §1, namely ‘What is the error frequency?’ over the whole map.

An important concept in both binomial and multinomial tests is the *null hypothesis*  $H_0$ , which here is that the true or population statistic, e.g. the overall map accuracy, is within a range that we estimate from the sample. We fix a *confidence level*  $(1 - \alpha)$  for this hypothesis, and this in turn allows us to compute the range. The parameter  $\alpha$  is the acceptable probability of a *Type I* error, that is, the chance that we are rejecting the null hypothesis when it is in fact true. The lower we set  $\alpha$ , the narrower the computed confidence intervals, but the less certain we are that the (unknown) population value is in fact in this interval; in addition, a larger sample is required to establish a given precision.

The statistics for the binomial test are covered in many sources [e.g. 10, 24]. Given an unbiased sample of size  $n$ , with  $n_t$  successes, the proportional accuracy and its standard deviation are estimated as:

$$p = n_t/n \tag{1}$$

$$s = \sqrt{\frac{p \cdot (1 - p)}{n}} \tag{2}$$

If the sample size is large enough, the confidence interval of the estimate may be approximated as:

$$p \pm \left[ s \cdot Z_{1-\alpha} + \frac{1}{2n} \right] \tag{3}$$

where  $Z_{1-\alpha}$  is the two-tailed normal score for the probability of Type I error  $\alpha$ ; this can be obtained from tables or computed (§5.6). The small-sample correction  $1/2n$  is usually ignored for sample sizes  $n > 50$ ; it is then less than 1%. The lower and upper limits as computed by Equation 3 are truncated at 0 and 1, respectively, if necessary.

For small samples, see §2.1, below.

## A numerical example of a binomial test

As an example, we summarize the confusion matrix of Table 3 (below); of the total observations  $n = 163$ , the diagonals, representing the correctly classified observations, sum to  $n_t = 86$ . Then we calculate  $p = 0.5276$  and  $s = 0.0391$ . For a 5% probability of a Type I error, the corresponding area under the normal curve is  $\Pr = 0.95$ , which is obtained for the two-tailed test with  $Z = 1.96$ . Then the 95% confidence interval for  $p$  is  $[0.4479 \dots 0.6073]$ . This is interpreted to mean that if we had repeated the same sampling scheme a large number of times, we expect that in 95% of these experiments the observed accuracy would be somewhere between 44.8% and 60.7%. We are taking a 5% risk that the true proportion is  $< 44.8\%$  or  $> 60.7\%$ .

We can narrow the confidence interval at the expense of a higher risk of Type I error. For example, increasing  $\alpha$  to 0.1, we obtain  $Z = 1.64$  and an interval for  $p$  of  $[0.4602 \dots 0.5950]$ , i.e. about 2.5% narrower. But now there is a 10% risk that the true proportion is outside this interval. Increasing  $\alpha$  to 0.2, i.e. a one in five chance of the true value being outside our calculated interval, we obtain  $Z = 1.28$  and an interval for  $p$  of  $[0.4744 \dots 0.5808]$ , now 5.3% narrower.

### 2.1 Small samples

For small samples, the confidence interval must be determined directly from the binomial distribution; the `EpiTools.net` website<sup>1</sup> provides R function `ci.binomial` for this purpose; these are reproduced in §5.4.

In our running numerical example, the 95% confidence interval for  $p$  computed by `ci.binomial` is  $[0.4480 \dots 0.6062]$  (count of successes 73 to 99 of 163), very close to the approximation computed above,  $[0.4479 \dots 0.6073]$ . However, for small sample sizes and for high or low proportions, the approximation is not suitable.

For example, if the sample size is only 20 (typical of many student projects), of which 16 agree with the map ( $p = 0.8$ ), the approximation (Equation 3), including the small sample correction, gives a 95% confidence interval of  $[0.5997 \dots 1.0003]$  (which must be truncated at the upper limit to 1), whereas the exact calculation gives

---

<sup>1</sup><http://www.medepi.org/epitools/rfunctions/index.html>

[0.5634...0.9427], about 3.5–5.5% lower and 2% narrower.<sup>2</sup>

### 3 Multinomial tests of accuracy

Binomial tests treat errors for all classes equally, and therefore can only estimate the accuracy of the map or stratum as whole. To investigate the errors associated with individual classes, *multinomial* tests must be used. These answer the first two questions posed in §1, namely ‘What is the error frequency?’ (as with binomial tests) and also ‘What is the *nature* of the errors: which classes or properties are not mapped correctly?’

Multinomial tests are based on the so-called *confusion matrix*. This is a table with *columns* representing the *reference* (observed) classes, and the *rows* the *classified* (mapped) classes. Each cell in the matrix contains the number of observations in the mapped class of its row that were in fact observed in the class of its column. Thus the diagonals represent agreement between map and ground (as in binomial tests), and off-diagonals represent different mis-classifications. Numerous authors [8, 17, 23, 14] explain confusion matrices.

Table 1 shows the basic notation of confusion matrices, and Table 2 shows the simple statistics that are computed from it.

---

<sup>2</sup>The interval computed without the finite-population correction is [0.6247...0.9753], which is unrealistically narrow.

Symbol	Meaning	Computation
$\mathbf{X}$	confusion matrix → rows $[1 \dots r]$ are <i>classified</i> (“mapped”) data → columns $[1 \dots r]$ are <i>reference</i> (“true”) data	
$r$	number of rows and columns of $\mathbf{X}$	
$x_{ij}$	number of observations in row $i$ , column $j$ , i.e. in reference class $j$ but mapped as class $i$	as observed
$x_{i+}$ $x_{+j}$	marginal sum of row (mapped class) $i$ marginal sum of column (reference class) $j$	$\sum_{j=1}^r x_{ij}$ $\sum_{i=1}^r x_{ij}$
$n$	total number of observations	$\sum_{i=1}^r \sum_{j=1}^r x_{ij}$ ... or $\sum_{i=1}^r x_{i+}$ ... or $\sum_{j=1}^r x_{+j}$
$\mathbf{P}$	proportions matrix	$\mathbf{X} ./ n$
$p_{ij}$	proportion of observations in row $i$ , column $j$	$x_{ij}/n$
$p_{i+}$ $p_{+j}$	proportion of mapped data in row (class) $i$ ... or proportion of reference data in column (class) $j$ ... or	$x_{i+}/n$ $\sum_{j=1}^r p_{ij}$ $x_{+j}/n$ $\sum_{i=1}^r p_{ij}$

Table 1: Notation for the confusion matrix

Symbol	Statistic	Computation
$C_i$	User's 'accuracy', mapped class $i$	$x_{ii}/x_{i+}$
	... or	$p_{ii}/p_{i+}$
$\bar{C}_i$	Errors of commission, mapped class $i$	$1 - C_i$
$O_j$	Producer's 'reliability', reference class $j$	$x_{jj}/x_{+j}$
	... or	$p_{jj}/p_{+j}$
$\bar{O}_j$	Errors of omission, reference class $j$	$1 - O_j$
$A_o$	Overall accuracy	$\sum_{i=1}^r x_{ii}/n$
	... or	$\sum_{i=1}^r p_{ii}$
$\bar{A}_o$	Overall error	$1 - A_o$

Table 2: Statistics computed from the confusion matrix

The traditional terminology for errors is somewhat confusing, but well-established. It is expressed from the point of view of the mapper. Looking across the rows (classes as mapped), an error of *commission* is said to occur when the mapper incorrectly mapped this class at a reference (ground truth) site where it does not exist. That is, the mapper 'committed' the error of over-mapping a class. This leads to a lower user's 'accuracy'  $C_i$ . Conversely, looking down the columns (classes as found in the field), an error of *omission* is said to occur when the mapper failed to correctly identify this reference site in its true class. That is, the mapper 'omitted' to map the site in the correct class. This leads to a lower producer's 'reliability'  $O_j$ . The overall accuracy  $A_o$  is the same as the estimated binomial accuracy  $\hat{p}$  from §2, considering all off-diagonals together as classification failures.

These matrices may be evaluated with various statistics, which we now explain

1. naïve measures of agreement;
2. *kappa*; and
3. *tau*.

These can all be modified to give partial credit for mis-classifications, using *weighted* versions.

### 3.1 Naïve measures of agreement

The statistics from Table 2 may be used directly as measures of overall accuracy ( $A_o$ ) or per-class accuracy from the user's or

producer's point of view ( $C_i$  or  $O_j$ , respectively). Individual off-diagonals  $p_{ij}, i \neq j$  show the frequency with which mapped class  $i$  was in fact class  $j$  on the ground. These statistics are easy to interpret. The standard deviation of the overall accuracy is computed as in the binomial test (Equation 2) with the proportion of successes  $p$  in that equation replaced by the overall accuracy  $A_o$ .

The standard deviation of the per-class user's accuracy is calculated similarly, but with the number of successes and the total number of trials limited to a single row:

$$p_{i+} = x_{ii}/x_{i+} \quad (4)$$

$$s_{i+} = \sqrt{\frac{p_{i+} \cdot (1 - p_{i+})}{x_{i+}}} \quad (5)$$

That is,  $p_{i+} = C_i$ . The confidence interval for  $p_{i+}$  is calculated as in Equation 3, except that the finite-population correction refers to the number of observations in the row:

$$p_{i+} \pm \left[ s_{i+} \cdot Z_{1-\alpha} + \frac{1}{2x_{i+}} \right] \quad (6)$$

The per-class producer's reliability is computed similarly, using the column proportions and total instead of row proportions and totals:

$$p_{+i} = x_{ii}/x_{+i} \quad (7)$$

$$s_{+i} = \sqrt{\frac{p_{+i} \cdot (1 - p_{+i})}{x_{+i}}} \quad (8)$$

That is,  $p_{+i} = O_i$ . The confidence interval for  $p_{+i}$  is:

$$p_{+i} \pm \left[ s_{+i} \cdot Z_{1-\alpha} + \frac{1}{2x_{+i}} \right] \quad (9)$$

If the class totals are small, and especially if the proportion of agreement is near 0 or 1, the exact binomial probabilities must be calculated, as explained in 2.1.

### A numerical example of naïve statistics

Table 3 is a sample confusion matrix also used by Congalton et al. [9] and Skidmore [23]. The counts from the confusion matrix  $\mathbf{X}$  are shown in roman type, and the proportions from the corresponding proportions matrix  $\mathbf{P}$  are shown in in italics. Table

4 shows per-class user’s accuracies and producer’s reliabilities, along with their standard deviations and confidence intervals (both approximate and exact). Note that the exact confidence interval differs substantially from the approximate in the classes with small sample size, proportions near 0 or 1, or both.

		Reference Class				Total
		A	B	C	D	$p_{i+}$
Mapped Class	A	35 <i>0.2147</i>	14 <i>0.0859</i>	11 <i>0.0675</i>	1 <i>0.0061</i>	61 <i>0.3742</i>
	B	4 <i>0.0245</i>	11 <i>0.0675</i>	3 <i>0.0184</i>	0 <i>0.0000</i>	18 <i>0.1104</i>
	C	12 <i>0.0736</i>	9 <i>0.0552</i>	38 <i>0.2331</i>	4 <i>0.0245</i>	63 <i>0.3865</i>
	D	2 <i>0.0123</i>	5 <i>0.0307</i>	12 <i>0.0736</i>	2 <i>0.0123</i>	21 <i>0.1288</i>
Total $p_{+j}$		53 <i>0.3252</i>	39 <i>0.2393</i>	64 <i>0.3926</i>	7 <i>0.0429</i>	163 <i>1.0000</i>

Table 3: Example of a confusion matrix

Class	User’s accuracy			
	$C_i$	$s_i$	95% C.I.	Exact C.I.
A	0.5738	0.0633	0.4415 ... 0.7061	0.4406 ... 0.6696
B	0.6111	0.1149	0.3581 ... 0.8641	0.3575 ... 0.8270
C	0.6032	0.0616	0.4744 ... 0.7319	0.4720 ... 0.7243
D	0.0952	0.0641	0.0000 ... 0.2446	0.0117 ... 0.3038
Class	Producer’s reliability			
	$O_j$	$s_j$	95% C.I.	Exact C.I.
A	0.6604	0.0651	0.5234 ... 0.7973	0.5173 ... 0.7848
B	0.2821	0.0721	0.1280 ... 0.4361	0.1500 ... 0.4487
C	0.5938	0.0614	0.4656 ... 0.7219	0.4637 ... 0.7149
D	0.2857	0.1707	0.0000 ... 0.6918	0.0367 ... 0.7096

Overall accuracy:  $A_o = 0.5276$

$s = 0.0391$ ; 95% C.I.: 0.4479 ... 0.6073

Table 4: Naïve per-class statistics for the example confusion matrix

We can examine the matrix and statistics to gain insight on the magnitude and type of errors. Clearly, some classes are better

mapped than others. From the map user’s point of view, classes A, B, and C are all mapped with an accuracy of about 60%; however, because of the small number of reference sites located in map unit B, its confidence interval is quite wide. From the map producer’s point of view, only classes A and C were reliably mapped. The producer often mis-mapped class B as one of these two.

### 3.2 The weighted confusion matrix

Naïve accuracy statistics may also be computed for a *weighted* confusion matrix, which gives partial credit for incorrect classifications. This is attractive when not all mapping mistakes are equally serious for the map user. This idea was first developed by Cohen [5] as a modification for the *kappa* statistic, but may be applied directly to naïve statistics.

This method requires that the analyst make a second matrix containing the weights for each cell in the confusion matrix. The diagonals must be 1, and the off-diagonals must be in the range  $[0 \dots 1]$ . A value of 0 indicates that there is no credit for this mistake, and as the values increase towards 1, the mistake is considered decreasingly serious. A value of 1 means that the two classes are considered identical for accuracy assessment. There is no need for the weights matrix to be symmetric, since an error of commission may be considered more or less serious than an error of omission. If all off-diagonals are 0, this reduces to the unweighted case.

! → In this technical note we explain the computations and interpretation of results; the more interesting question of how to assign weights is not treated here.

The additional notation and statistics for the weighted confusion matrix are shown in Table 5. The statistics  $\bar{w}_{i+}$  and  $\bar{w}_{+j}$  are the weights in the mapped (rows) or reference (columns) class, themselves weighted by the proportion of each class in the mapped or reference population, respectively, i.e. from  $\mathbf{P}^T$ .

The sums of weighted averages ranges from  $[1 \dots r]$ , with 1 representing the unweighted case, and  $r$  (the number of classes) representing the (meaningless) case where no misclassification is important. The weighted overall agreement and per-class accuracies

Symbol	Meaning	Computation
<b>W</b>	weights matrix $w_{ii} = 1, \forall i$ $w_{ij} \in [0 \dots 1], \forall i \neq j$	$\equiv 1$ assigned by evaluator
$\bar{w}_{i+}$	weighted average of the weights of row $i$	$\sum_{j=1}^r w_{ij} \cdot p_{+j}$
$\bar{w}_{+j}$	weighted average of the weights of column $j$	$\sum_{i=1}^r w_{ij} \cdot p_{i+}$

Table 5: Notation for the weighted confusion matrix

are computed as:

$$A_{ow} = \sum_{i=1}^r \sum_{j=1}^r w_{ij} \cdot p_{ij} \quad (10)$$

$$C_{iw} = \frac{1}{p_{i+}} \cdot \sum_{j=1}^r w_{ij} \cdot p_{ij} \quad (11)$$

$$O_{jw} = \frac{1}{p_{+j}} \cdot \sum_{i=1}^r w_{ij} \cdot p_{ij} \quad (12)$$

If the off-diagonals are zero, these reduce to the formulas from Table 2.

The standard deviations and confidence intervals for the overall accuracy are calculated with Equations 2 and 3, respectively, as in the unweighted case; however, the proportion of successes  $p$  in those equations is not limited to the diagonals (as in Equation 1), but instead includes the weighted contribution from the off-diagonals, i.e. it is the overall weighted accuracy  $A_{ow}$  from Equation 10, above.

Similarly, the standard deviations and confidence intervals for the *per-class user's accuracy* are calculated with Equations 5 and 6, with the proportion of successes  $p_{i+}$  in those equations replaced by the *per-class weighted user's accuracy*  $C_{iw}$  from Equation 11, above. The *per-class producer's reliabilities* are calculated with Equations 8 and 9, with the proportion of successes  $p_{+i}$  in those equations replaced by the *per-class weighted producer's reliability*  $O_{iw}$  from Equation 11.

A numerical example for the weighted confusion matrix

This example, shown in Table 6 uses the confusion matrix of Table 3, and in addition a hypothetical (arbitrary) 4x4 weights matrix.

The confusion matrix  $\mathbf{X}$  is presented here as the proportions matrix  $\mathbf{P}$ , since these are used in the formulas of the previous section.

The example weights matrix gives full credit for each class correctly (1's on the diagonal), for mapping class D as A or C, and for mapping class A as C. It gives substantial partial credit (0.91) for mapping class A as class D, and somewhat less for mapping class C as A (0.67) or D (0.61). There is no partial credit for mis-mapping class B.

	Reference Class				$p_{i+}$	$\bar{w}_{i+}$	$C_{iw}$
	A	B	C	D			
A	0.2147	0.0859	0.0675	0.0061	0.3742	0.6312	0.7110
	<i>1</i>	<i>0</i>	<i>0.67</i>	<i>1</i>			
B	0.0245	0.0675	0.0184	0.0000	0.1104	0.2393	0.6111
<b>Mapped</b>	<i>0</i>	<i>1</i>	<i>0</i>	<i>0</i>			
<b>Class C</b>	0.0736	0.0552	0.2331	0.0245	0.3865	0.7607	0.8571
	<i>1</i>	<i>0</i>	<i>1</i>	<i>1</i>			
D	0.0123	0.0307	0.0736	0.0123	0.1288	0.5783	0.5305
	<i>0.91</i>	<i>0</i>	<i>0.61</i>	<i>1</i>			
$p_{+j}$	0.3252	0.2393	0.3926	0.0429	1	2.5938	
$\bar{w}_{+j}$	0.9880	0.1104	0.7158	0.8896	2.2095		
$O_{jw}$	0.9211	0.2821	0.8233	1.0000			

Overall weighted accuracy  $A_{ow}$ : 0.7332

Table 6: Example weighted confusion matrix

In this example,  $r = 4$ , which is then the maximum possible for the sums of weighted averages. The producer's sum is  $\approx 2.2$  and the user's  $\approx 2.6$ , indicating that many classes in the map are similar, especially from the user's point of view.

The weighted per-class user's accuracies range from moderate ( $\approx 53\%$  for class D) to high ( $\approx 86\%$  for class C). Producer's reliabilities range from very low ( $\approx 28\%$  for class B) to perfect (for class D). In most cases, these are significantly higher than the corresponding unweighted figures from Table 4. The exception is class B, which is so unlike the other classes that partial credit was not given; in this case the weighted and unweighted figures are identical.

The overall utility of the map to the user is measured by the overall weighted accuracy  $A_{ow}$ . In this example it is quite high,  $\approx 73\%$ , as compared to the unweighted accuracy  $A_o \approx 53\%$ .

Per-class weighted accuracies, their standard deviations, and confidence intervals for this example are shown in Table 7, which may be compared to the unweighted equivalents in Table 4.

Class	User's accuracy		
	$C_{i_w}$	$s_{i_w}$	95% C.I.
A	0.7110	0.0580	0.5890 ... 0.8329
B	0.6111	0.1149	0.3581 ... 0.8641
C	0.8571	0.0441	0.7628 ... 0.9515
D	0.5305	0.1089	0.2932 ... 0.7677

  

Class	Producer's reliability		
	$O_{j_w}$	$s_{j_w}$	95% C.I.
A	0.9211	0.0370	0.8391 ... 1.0000
B	0.2821	0.0721	0.1280 ... 0.4361
C	0.8233	0.0477	0.7220 ... 0.9245
D	1.0000	0.0000	0.9286 ... 1.0000

Overall accuracy:  $A_{o_w} = 0.7332$

$s_w = 0.03464$ ; 95% C.I.: 0.6622 ... 0.8042

Table 7: Naïve statistics for the example confusion and weight matrices

The standard deviations for the weighted case are in general lower, because the accuracies are in general further from 0.5. Only for Class D's user's accuracy, where the weighted accuracy increases to near 0.5, is the standard deviation higher.

### 3.3 The *kappa* index of agreement

The *kappa* index of agreement for categorical data was developed by Cohen [4, 5] and associates [11] in the context of psychology and psychiatric diagnosis. The specific question they addressed is whether two classifications of the same group of subjects agree or not, in other words, whether the two persons performing a diagnosis agree in general. If not, it suggests that one or both of the diagnosticians are in error, or that the categories of the classification can not reliably be distinguished by competent professionals.

*Kappa* was subsequently adopted by the remote sensing community as a useful measure of classification accuracy, first in the obvious analogy to psychiatric diagnosis where the two diagnosticians are two human photo-interpreters or two automatic classifiers. It was then extended to the case where one diagnostician is

the automatic classifier (the computer) and the other is the manual classifier (a human), by tradition with the manual classifier considered as the columns or ‘reference’ to which the automatic classifier or ‘map’ is to be compared.

*Kappa* is explained by Congalton [6, 7], Congalton and Green [8] and Skidmore [23], among others. A formula for its variance was developed by Fleiss et al. [11]. Correct formulas for *kappa* and its variance are presented by Hudson and Ramm [15]. A clear mathematical derivation is given by Bishop et al. [1, §11.4.2].

The basic idea behind *kappa* is that some of the apparent classification accuracy given by naïve measures could be due to chance. This is especially relevant when some classes are more likely to be encountered during field sampling than others. As a simple illustration, consider a map where class *A* is mapped over 90% of the survey area, and class *B* over the other 10%. This can be used as an *a priori* estimate of the actual coverage, in which case a randomly-selected field sampling point would have a 90% chance of being in fact class *A*. The joint probability is then 0.81 of a this point being correctly mapped as class *A*, strictly by chance. A similar argument for class *B* gives a joint probability of 0.01, so the map would be expected to have an overall accuracy of 82% simply by a chance assignment of ground truth points to mapped classes, irrespective of the actual pattern shown on the map, so long as the mapped proportions are as given. This makes it hard to distinguish a good map from one that is simply lucky.

This is not important to the map user, who only cares about the quality of the single map in hand is or the relative quality of several maps. However, to the producer who wants to compare mapping methods or mappers can not use naïve statistics. In particular, a map with a few large classes may appear more accurate than one with many classes, simply because of the simpler legend. The *kappa* statistic somewhat compensates for this.

Use of *kappa* instead of naïve accuracy measures becomes less important as the number of classes increases. For example, for a map with twenty classes, each with  $p_{i+} = p_{+i} = 0.05$ , chance agreement is only  $20 \cdot (0.05)^2 = 0.05$ . It also becomes less important as the balance between classes becomes more even. For two equally-likely classes which are each mapped over half of the area, the chance overall accuracy is 50%, compared to 82% for the 90% – 10% split calculated above.

*Kappa* as presented here is based on the assumption that the marginal proportions are known to the classifier before assigning observations to classes [2, 20]. This has a clear interpretation in psychiatric diagnosis, where the prior probabilities of various conditions are known from previous studies with similar populations. In land resource surveys or remote sensing interpretations, an estimate of the proportions of each class is established by the survey itself, that is, we assume that the survey is correct before collecting ground truth to verify it. The resulting row (map) proportions  $p_{i+}$  can be reasonably considered as prior probabilities for the column (reference) proportions  $p_{+j}$ , if we assume that the mapper is at least somewhat competent. That is, before the accuracy assessment takes place, the evaluator already knows the proportions that ‘should’ be found. But, *kappa* also assumes that the mapper knows the marginal proportions. This is certainly not the case where the mapper is an automatic classifier. In §3.6 we discuss the case where prior knowledge or ignorance of marginal proportions is explicitly considered.

**Simple vs. stratified random sampling** The formulas presented in this section are only valid if samples were from a simple random sample or proportional stratified sampling, that is, where the number of samples in a stratum is proportional to its mapped area. For a stratified random sample, they must be adjusted. These were developed by Stehman [26]. However, he demonstrates that using ordinary *kappa* (as presented here) on stratified samples is not too bad except when classes are quite unequal in size or samples are far from proportional to class size.

### 3.3.1 Unweighted *kappa*

The usual form of *kappa* is where all errors are equally important. We define four intermediate coefficients  $\theta_1 \dots \theta_4$  as follows:

$$\theta_1 = \frac{1}{n} \sum_{i=1}^r x_{ii} = \sum_{i=1}^r p_{ii} \quad (13)$$

$$\theta_2 = \frac{1}{n^2} \sum_{i=1}^r x_{i+} \cdot x_{+i} = \sum_{i=1}^r p_{i+} \cdot p_{+i} \quad (14)$$

$$\theta_3 = \frac{1}{n^2} \sum_{i=1}^r x_{ii} \cdot (x_{i+} + x_{+i}) = \sum_{i=1}^r p_{ii} \cdot (p_{i+} + p_{+i}) \quad (15)$$

$$\theta_4 = \frac{1}{n^3} \sum_{i=1}^r \sum_{j=1}^r x_{ij} \cdot (x_{+i} + x_{+j})^2 = \sum_{i=1}^r \sum_{j=1}^r p_{ij} \cdot (p_{+i} + p_{+j})^2 \quad (16)$$

The first coefficient,  $\theta_1$ , is the overall accuracy  $A_o$ . The second,  $\theta_2$ , is the expected overall accuracy if there were chance agreement between reference and mapped data. The other two coefficients are used in the computation of variance. Note that in the formula for  $\theta_4$ , the *reference* total  $x_{+i}$  for mapped class  $i$ , is added to the *mapped* total  $x_{j+}$  for reference class  $j$ . That is, these are not the marginal totals for the cell, but rather those from the transpose  $\mathbf{X}^T$ .

These coefficients are then combined to compute *kappa*, and its variance as follows:

$$\hat{k} = \frac{\theta_1 - \theta_2}{1 - \theta_2} \quad (17)$$

$$s^2[\hat{k}] = \frac{1}{n} \left[ \frac{\theta_1(1 - \theta_1)}{(1 - \theta_2)^2} + \frac{2(1 - \theta_1)(2\theta_1\theta_2 - \theta_3)}{(1 - \theta_2)^3} + \frac{(1 - \theta_1)^2(\theta_4 - 4\theta_2^2)}{(1 - \theta_2)^4} \right] \quad (18)$$

In Equation 17, the denominator is the proportion of disagreement that is predicted by the marginal totals. The numerator is the actual proportion of disagreement. So  $\hat{k}$  is the proportion of disagreements expected by chance that did *not* occur. If it is 1, this means that there were no disagreements. If it is 0, this means that the disagreements that occur are exactly those expected by chance. If it is less than 0, this means that the classifier actually has more disagreement with the reference data than would be expected from a pure random assignment of classified pixels to classes.

The *kappa* statistic may be difficult to explain to map users. The zero point is easy enough to understand, but the increasing accuracy of the map as *kappa* increases is not a direct relation as with overall accuracy. In particular, it depends on the marginal proportions.

Related to this is the question of what is a ‘good’ value of *kappa*. This is best answered by reference to previous studies of similar themes and landscapes.

It would be quite disturbing for *kappa* to be not significantly different from zero, as this would indicate that the mapper was incompetent; fortunately, in practice this is rarely the case.

Using the estimated *kappa* and its variance, we can compute a confidence interval for *kappa*. We can also compare two classifications to see if one is significantly better than the other [e.g 19], in the same way as was explained for the binomial test (§4.1), substituting  $\hat{k}$  and  $s^2[\hat{k}]$  for  $p$  and  $s^2$ , respectively, in Equation 33.

#### A numerical example for unweighted *kappa*

This example continues the analysis of the data in Table 3. The coefficients and statistics computed from this confusion matrix are shown in Table 8.

Statistic	Value
$\theta_1$	0.5276
$\theta_2$	0.3054
$\theta_3$	0.3575
$\theta_4$	0.4037
$\hat{k}$	0.3199
$s^2[\hat{k}]$	0.00274
$s[\hat{k}]$	0.05234
C.V.	16.4%
95% C.I.	0.2143 ... 0.4256

Table 8: *Kappa* statistics for the example confusion matrix

In this example,  $\hat{k}$  ( $\approx 32\%$ ) is substantially lower than the naïve measure of agreement, namely overall accuracy  $A_o = \theta_1$  ( $\approx 53\%$ ), indicating that a large portion of the apparent classification accuracy could be due to chance agreement. Because of the small sample size, the confidence interval for  $\hat{k}$  is wide. This is also shown by the high coefficient of variability. These are decreased as the square root of the increase in sample size. For example to double the precision, the sample size must be quadrupled.

### 3.4 Conditional *kappa*

*Kappa* may also be calculated for each class, either by row (user’s accuracy) or column (producer’s reliability) [20, 1]. This allows us to determine which classes are well-mapped. The calculation follows the same logic as overall *kappa*, but is restricted to one

row (Equation 19) or column (Equation 20):

$$\hat{k}_{i+} = \frac{(p_{ii}/p_{i+}) - p_{+i}}{1 - p_{+i}} = \frac{C_i - p_{+i}}{1 - p_{+i}} \quad (19)$$

$$\begin{aligned} &= \frac{p_{ii} - (p_{i+} \cdot p_{+i})}{p_{i+} - (p_{i+} \cdot p_{+i})} \\ \hat{k}_{+j} &= \frac{(p_{jj}/p_{+j}) - p_{j+}}{1 - p_{j+}} = \frac{O_j - p_{j+}}{1 - p_{j+}} \quad (20) \\ &= \frac{p_{jj} - (p_{+j} \cdot p_{j+})}{p_{+j} - (p_{+j} \cdot p_{j+})} \end{aligned}$$

Conditional *kappa* is in the interval  $[0 \dots p_{+i}]$  (user's) or  $[0 \dots p_{j+}]$  (producers's). It is thus a downward adjustment of the naïve measure of accuracy to take into account chance agreement. In general,  $\hat{k}_{i+} = C_i$  iff  $C_i = 1$ , i.e. perfect user's accuracy for the class. For a given agreement  $C_i$ ,  $\hat{k}_{i+}$  is lower to the extent that the marginal proportion  $p_{+i}$  of the reference class increases. That is, as the reference class becomes more likely, the apparent agreement represented by  $C_i$  must be adjusted more.

The variance of conditional *kappa* for user's class  $i$  is calculated as [1, Eqn. 11.4-10]:

$$s^2[\hat{k}_{i+}] = \frac{1}{n} \cdot \left\{ \frac{p_{i+} - p_{ii}}{p_{i+}^3 \cdot (1 - p_{+i})^3} \cdot [(p_{i+} - p_{ii})(p_{i+} \cdot p_{+i} - p_{ii}) + p_{ii} \cdot (1 - p_{i+} - p_{+i} + p_{ii})] \right\} \quad (21)$$

For producer's class  $i$ , Equation 21 is permuted by interchanging the row and column summaries, i.e.  $p_{i+}$  and  $p_{+i}$ .

### A numerical example for conditional *kappa*

Table 9 shows conditional *kappa* and their variances for the four classes of the sample confusion matrix of §3.3.1 (Table 3).

As expected, both user's and producer's conditional *kappa* are smaller than the naïve measures of accuracy. Class D is clearly very poorly mapped. We can see an important discrepancy between user's and producer's accuracy both for this class and class B. The coefficients of variation are all higher than for overall  $\hat{k}$ , due to the small sample sizes.

### 3.5 Weighted *kappa*

*Kappa* may also be computed for the weighted confusion matrix presented in §3.2, which gives partial credit for incorrect classifi-

Class	User's accuracy			
	$C_i$	$\hat{k}_{i+}$	$s[\hat{k}_{i+}]$	C.V.,%
A	0.5738	0.3684	0.0763	20.7
B	0.6111	0.4888	0.1440	29.5
C	0.6032	0.3466	0.0824	23.8
D	0.0952	0.0546	0.0603	110.3

  

Class	Producer's reliability			
	$O_j$	$\hat{k}_{+j}$	$s[\hat{k}_{+j}]$	C.V.,%
A	0.6604	0.4573	0.0899	19.6
B	0.2821	0.1929	0.0673	34.9
C	0.5938	0.3378	0.0806	23.9
D	0.2857	0.1801	0.1906	105.8

Table 9: Conditional *kappa* statistics for the example confusion matrix

cations. This was first developed by Cohen [5] and then refined by Fleiss et al. [11]. Its practical application in remote sensing accuracy assessment was explored by Næsset [21]. The formulas presented here are equivalent to, but computationally simpler than, those from these sources.

From the confusion matrix and the notation of Table 5, we define several intermediate coefficients:

$$\theta_{w1} = \sum_{i=1}^r \sum_{j=1}^r w_{ij} \cdot p_{ij} \quad (22)$$

$$\theta_{w2} = \sum_{i=1}^r \sum_{j=1}^r w_{ij} \cdot p_{i+} \cdot p_{+j} \quad (23)$$

$$\theta_{w4} = \sum_{i=1}^r \sum_{j=1}^r p_{ij} \cdot [w_{ij} \cdot (1 - \theta_{w2}) - (\bar{w}_{i+} + \bar{w}_{+j}) \cdot (1 - \theta_{w1})]^2 \quad (24)$$

The coefficients  $\theta_{w1}$  and  $\theta_{w2}$  are the weighted equivalents of  $\theta_1$  and  $\theta_2$  from Equations 13 and 14, and have the same meaning, namely actual and chance agreement, but here with ‘agreement’ including partial agreements. These are then combined to produce weighted *kappa* and its variance, as follows:

$$\hat{k}_w = \frac{\theta_{w1} - \theta_{w2}}{1 - \theta_{w2}} \quad (25)$$

$$s^2[\hat{k}_w] = \frac{\theta_{w4} - (\theta_{w1} \cdot \theta_{w2} - 2\theta_{w2} + \theta_{w1})^2}{n \cdot (1 - \theta_{w2})^4} \quad (26)$$

If  $\mathbf{W} = \mathbf{I}$ , i.e. the diagonals are 1 and the off-diagonals 0, these formulas give the same results as the formulas for unweighted *kappa* (Equations 17 and 18).

#### A numerical example for weighted *kappa*

This example continues that of §3.2. We calculate the coefficients and statistics shown in Table 10.

Statistic	Value
$\theta_{w1}$	0.7332
$\theta_{w2}$	0.6312
$\theta_{w4}$	0.0187
$\hat{k}_w$	0.2776
$s^2[\hat{k}_w]$	0.004741
$s[\hat{k}_w]$	0.06886
C.V.	24.1%
95% C.I.	0.1386 ... 0.4146

Table 10: Weighted *Kappa* statistics for the example confusion and weight matrices

As in the unweighted case,  $\hat{k}_w$  is substantially lower than the naïve measure of agreement, here the *weighted* overall accuracy  $A_{ow} = \theta_{w1}$ . In this example, there is even a greater discrepancy than for the unweighted example, because, first, the apparent agreement is much higher due to the partial credit given to incorrect answers, and second, the weighted chance agreement is higher. Weighted *kappa* may be higher or lower than unweighted *kappa*, depending on the confusion and weight matrices. In the present case, the large proportions of classes A and C in both the reference and mapped population, along with the high partial credit given to the misclassification between these two, leads to a high chance agreement.

Weighted *kappa* gives insight into the value of the map as compared to a random assignment of areas to map units with the specified proportions, when some classes are similar. In this example it is quite low, about 28%. Thus the map is quite good for the user (as shown by  $A_{ow} \approx 73\%$  from Table 6), but the producer didn't improve the pre-survey knowledge very much.

### 3.6 The *tau* index

The various *kappa* indices all assume that both the user's and producer's marginal probabilities for each class are known before the classification. In remote sensing applications, this is not the case for the user's marginals (rows of the confusion matrix), which are not known until the automatic classifier completes its work. The producer's marginals (columns), by contrast, are considered to be known before the classification, since they are independently-determined by field sampling. Ma and Redmond [18] proposed the *tau* index of agreement for this situation. Their work was extended by Næsset [20] to assess per-class producer's reliability; the per-class user's accuracy is the same as conditional *kappa*. *Tau* may be especially appropriate for assessing soil maps made by automatic classifiers [e.g. 32].

To assess overall accuracy, Ma and Redmond [18] define (using the notation of Table 2 and §3.3.1):

$$\tau = \frac{\theta_1 - \theta'_2}{1 - \theta'_2}, \text{ where} \quad (27)$$

$$\theta_1 = \sum_{i=1}^r p_{ii} \quad (28)$$

$$\theta'_2 = \sum_{i=1}^r p_i \cdot p_{+i} \quad (29)$$

$$\theta'_3 = \sum_{i=1}^r p_{ii} \cdot (p_i + p_{+i}) \quad (30)$$

$$\theta'_4 = \sum_{i=1}^r \sum_{j=1}^r p_{ij} \cdot (p_{+i} + p_j)^2 \quad (31)$$

Equation 27 has the same form as the definition of  $\hat{k}$  in Equation 17, except that  $\theta_2$  from Equation 14, representing chance agreement, is replaced by  $\theta'_2$  in Equation 27. This equation in turn uses  $p_i$  instead of  $p_{i+}$  to represent the row proportions. These  $p_i$  are the *prior* probabilities of class membership. In the absence of specific information, they may all be set to  $1/r$ , i.e. equal probability for each of the  $r$  classes. In this situation,  $\tau$  is the same as the modified  $\hat{k}$  proposed by Foody [12]. *Tau* can also be applied with any other prior assignment of classes, for example, an estimate of the distribution of soil classes based on a detailed map of a sample area.

! → The key issue when using *tau* is the assignment of the prior prob-

abilities. As will be shown in the numerical example below (§3.6), this can have a large effect on the resulting value of  $\tau$ .

The coefficients  $\theta'_3$  and  $\theta'_4$  are modifications of Equations 15 and 16, with prior row proportions substituted for actual. They can then be used to calculate the variance  $s^2[\tau]$  with a variation of Equation 18:

$$s^2[\tau] = \frac{1}{n} \left[ \frac{\theta_1(1-\theta_1)}{(1-\theta_2)^2} + \frac{2(1-\theta_1)(2\theta_1\theta_2 - \theta'_3)}{(1-\theta_2)^3} + \frac{(1-\theta_1)^2(\theta'_4 - 4\theta_2^2)}{(1-\theta_2)^4} \right] \quad (32)$$

and this variance can be used to compute confidence intervals as for  $\kappa$  or the binomial test (Equation 3). We can also compare two classifications to see if one is significantly better than the other, substituting  $\tau$  and  $s^2[\tau]$  for  $p$  and  $s^2$ , respectively, in Equation 33.

#### A numerical example for $\tau$

Table 11 shows the computed coefficients and statistics using the data of Table 3, for three cases: (1) assuming that each class is equally likely *a priori*, i.e.  $p_i = 0.25, \forall i$  (left column); (2) with unequal priors, in this case  $\{0.1, 0.4, 0.1, 0.4\}$ , i.e. with the highest expectation for the classes that are not mapped over the largest areas (center column); and (3) with unequal priors, in this case  $\{0.4, 0.1, 0.4, 0.1\}$ , i.e. with the highest expectations closely matching the largest mapped areas (right column).

With equal priors (left column),  $\theta'_2$  is substantially lower than the chance agreement  $\theta_2$  computed for  $\hat{k}$ . This leads to a value of  $\tau$  that is significantly higher than  $\hat{k}$  (0.3199) and closer to the overall accuracy (0.5276). This is because, in this example, the marginal proportions for the rows are far from equal, and furthermore the two largest match the corresponding row proportions fairly closely. The standard deviation for  $\tau$  is somewhat lower than for  $\kappa$  (0.05234).

If the prior probabilities are quite different from the actual proportions (center column),  $\tau$  is even closer to the overall accuracy; this is because the map is not what was expected, so agreement is a pleasant surprise. In this case, as shown in the right-

Statistic	Value <sup>†</sup>	Value <sup>‡</sup>	Value <sup>*</sup>
$\theta_1$	0.5276	0.5276	0.5276
$\theta_2'$	0.2500	0.1847	0.3153
$\theta_3'$	0.3099	0.2547	0.3651
$\theta_4'$	0.3209	0.2667	0.4202
$\tau$	0.3701	0.4206	0.3100
$s^2[\tau]$	0.00239	0.002064	0.002186
$s[\tau]$	0.0489	0.04543	0.05307
C.V.	13.2%	10.8%	17.1%
95% C.I.	0.2712 ... 0.4691	0.3285 ... 0.5127	0.2030 ... 0.4171

<sup>†</sup>Equal priors, all 0.25

<sup>‡</sup>Priors {0.1, 0.4, 0.1, 0.4}

<sup>\*</sup>Priors {0.4, 0.1, 0.4, 0.1}

Actual proportions {0.3742, 0.1104, 0.3865, 0.1288}

Table 11: Tau statistics for the example confusion matrix

most column of Table 11, there is a smaller chance agreement, the index of agreement is closer to the overall accuracy, and the variability of the estimate is lower. The map is providing a large increase in information over the prior state of knowledge.

If the prior probabilities are the same as the actual row proportions, *tau* is the same as *kappa*. This situation is closely approximated in the rightmost column of Table 11, where the statistics are quite close to those shown in the example for unweighted *kappa* of Table 8.

**Weighted *tau*** It should be possible to weight the *tau* statistic, as was done for *kappa* (§3.5); however, I have not yet worked out the analysis.

## 4 Comparing maps

A common problem for both map users and producers is to *compare* several maps to see which has higher accuracy. Producers are generally comparing different classification algorithms, while users are deciding which map is better for a given application. This latter decision depends on the *utility* of the map with respect to the intended use Stehman [30]; accuracy statistics can, of course, inform it.

A second problem is to see how well two maps agree with each other. They may be equally accurate when considered separately, but their errors may not be the same. In other words, do they provide the same information?

#### 4.1 Comparing the relative accuracy of two maps

**Binomial tests** The simplest comparison between maps is which of them has a higher binomial accuracy; that is, which has the lower error frequency over the whole map. If we assume that the two samples have the same true (unknown) variance, we can pool these to calculate the test  $Z$  statistic for the difference between the two binomial accuracies:

$$Z = \frac{|p_1 - p_2|}{\sqrt{s_1^2 + s_2^2}} \quad (33)$$

This  $Z$ -score is the ordinate on the cumulative normal distribution, i.e. the number of standard deviations from the mean. These can be found in tables or computed in most statistical programs (§5.6). The tabulated value is then the one-tailed probability that the difference between the classifications is due to chance. In general the test is two-tailed, since neither map is preferred *a priori*, so the probability obtained from the table should be doubled.

(The assumption of equal variances is generally reasonable if the sample sizes and assessment methods are similar. Otherwise, the two variances must be weighted [3].)

Continuing the numerical example of §2, suppose we have a second map that appears to be more accurate, with  $p_2 = 0.65$  and a somewhat higher estimated standard deviation of the accuracy,  $s_2 = 0.045$ . The 95% confidence interval for this second classification computed by Equation 3 is [0.5618 ... 0.7382]. The  $Z$ -score for the difference between the classifications would be, according to Equation 33:

$$Z = \frac{|0.5276 - 0.65|}{\sqrt{(0.0391)^2 + (0.045)^2}} = 2.0532$$

The one-tailed probability corresponding to a  $Z$ -score of 2.0532 is  $\approx 0.020$ , so the two-tailed probability that the second map would in fact be better than the first is  $\approx (1 - (2 \cdot 0.020)) = 96.0\%$ . That is, if we decide that it is justified to pool the variances, the test suggests that the second map would in fact be better than the first, with  $\text{Pr} = 0.960$ .

**Multinomial tests** The user's or producer's accuracy of two maps for a single class can be compared with binomial tests, only taking into account the proportion of successful classifications in the relevant row or column. The two maps can be compared over-all by listing the comparisons for each class.

The *kappa* or *tau* statistics for an entire map can be compared; in the second case, the same prior probabilities must be used for each assessment. It seems that Equation 33 could be used to test whether apparent differences are likely to be due to chance, using the variance for each estimate of *kappa* or *tau*, but I am not sure whether this approach is valid.

## 4.2 Comparing the agreement between two maps with *kappa*

The original use of the *kappa* statistic was to compare two independent psychiatric diagnoses [4]. By analogy, we can use it to compare two maps with the same thematic classification. Both rows and columns represent mapped data (that is, the columns are not ground truth). We must assume that both mappers had the same prior information about the proportions of each class to be mapped; this may be true of human mappers, who have prior experience in an area, but is difficult to establish for automatic classifiers.

Here the *kappa* statistic refers to agreement between *maps*, not a map and the ground truth. A sampling grid must be set up which covers the common mapped area, and the class of each map at the sample point recorded. If both maps are discretized ('raster' maps) on a common grid, a cross-tabulation operation in the GIS will provide the confusion matrix.

A *kappa* value of 0 implies that the two maps are unrelated: knowing the class at a given point of one map gives no information about the class at that point on the other map. A *kappa* value of 1 implies complete agreement; a value less than 0 implies more disagreement than a random assignment of classes to one of the maps. The advantage of using *kappa* rather than the naïve statistics is that chance agreement is accounted for; this is most important with only a few classes or if some classes cover a large proportion of the maps.

## 5 Computing accuracy in R

This section presents functions for computing accuracy measures in the R environment for statistical computing and visualisation [16]. If you are unfamiliar with R, you should consult the monograph “Introduction to the R Project for Statistical Computing for use at ITC” [22]; this explains how to use R on the ITC network, how to obtain and install your own (free and legal) copy, and how to use the documentation and tutorials.

The functions presented here may be cut-and-pasted at the R command line; they are also available as digital files which may run with the `source()` method.

### 5.1 Naïve and *kappa*

First is an R function to compute naïve and *kappa* statistics, overall and per-class for confusion matrix (formal argument `CM`), which returns its results in a list. The function checks that the matrix is square (equal number of field & mapped classes). The results may be printed with `summary.kappa` (see below); you can calculate other statistics directly from the returned list. For example, to compute the average user’s and producer’s accuracy :

---

#### R code:

```
z<-kappa(x)
sum(z$user.naive)/nrow(x); sum(z$prod.naive)/ncol(x)
```

---

---

## R code:

```
kappa <- function(CM) {
  #convert both data frames and vectors to matrices
  cmx<-as.matrix(CM)
  #try to convert a vector to a square matrix
  if (ncol(cmx) == 1)
    cmx<-matrix(cmx, byrow=TRUE, nrow=sqrt(nrow(cmx)))
  nr<-nrow(cmx); nc<-ncol(cmx)
  if (nr != nc)
    { print("Error: matrix is not square"); break }
  n<-sum(cmx)
  d<-diag(cmx); dsum<-sum(d); th1<-dsum/n
  th1v<-((th1*(1-th1))/n)
  csum<-apply(cmx,2,sum); rsum<-apply(cmx,1,sum)
  ua<-d/rsum; pa<-d/csum
  th2 <- sum(rsum*csum) / n^2; kh <- (th1-th2)/(1-th2)
  th3 <- sum( (csum + rsum) * d ) / n^2;
  th4 <- 0; for (i in 1:nr) for (j in 1:nc)
    th4 <- th4 + (cmx[i,j] * ((csum[i] + rsum[j])^2));
  th4 <- th4 / n^3;
  th1c <- 1 - th1; th2c <- 1 - th2;
  khv <- 1/n *
    (
      ( ( th1 * th1c ) / th2c^2 )
      + ( ( 2 * th1c * ((2*th1*th2) - th3) ) / th2c^3 )
      + ( ( th1c^2 * ( th4 - (4 * th2^2) ) ) / th2c^4 )
    )
  #per-class kappa, user's accuracy...
  p <- cmx/n; uap <- apply(p,1,sum); pap <- apply(p,2,sum); dp<-diag(p);
  kpu <- (dp/uap - pap)/(1 - pap);
  #...and its variance
  t1 <- uap-dp; t2 <- (pap*uap)-dp; t3 <- dp*(1 - uap - pap + dp);
  kpuv <- ( (t1/(uap^3 * (1-pap)^3)) * ((t1*t2) + t3) )/n;
  #per-class kappa, producer's reliability...
  kpp <- (dp/pap - uap)/(1 - uap);
  #...and its variance
  t1 <- (pap-dp);
  kppv <- ( (t1/(pap^3 * (1-uap)^3)) * ((t1*t2) + t3) )/n;
  #return all statistics as a list
  list(sum.n=n, sum.naive=th1, sum.var=th1v, sum.kappa=kh, sum.kvar=khv,
    user.naive=ua, prod.naive=pa,
    user.kappa=kpu, user.kvar=kpuv, prod.kappa=kpp, prod.kvar=kppv)
}
```

---

## Second, an R function to print the summary statistics:

---

### R code:

```
summary.kappa <- function(kappa, alpha=0.05) {
  ciw<-function(var, n) {
    qnorm(1-(alpha/2))*sqrt(var) + (1/(2*n))
  }
  print(paste("Number of observations:", kappa$sum.n), quote=F)
  print("Summary of naive statistics", quote=F)
  print(paste(
    "Overall accuracy, stdev, CV%:",
    round(kappa$sum.naive, 4), ",",
    round(sqrt(kappa$sum.var), 4), ",",
    round((sqrt(kappa$sum.var)/kappa$sum.naive)*1000,0)/10,
    quote=F)
  )
  w<-ciw(kappa$sum.var, kappa$sum.n)
  print(paste(
    round((1-alpha)*100,0), "% confidence limits for accuracy:",
    round((kappa$sum.naive-w), 4), "...",
    round((kappa$sum.naive+w), 4)), quote=F, sep="")
  print("User's accuracy", quote=F); print(round(kappa$user.naive, 4));
  print("Producer's reliability:", quote=F); print(round(kappa$prod.naive, 4));
  print("Summary of kappa statistics", quote=F)
  print(paste("Overall kappa, stdev, & CV%:",
    round(kappa$sum.kappa, 4), ",",
    round(sqrt(kappa$sum.kvar), 4), ",",
    round((sqrt(kappa$sum.kvar)/kappa$sum.kappa)*1000,0)/10, quote=F)
  )
  w<-ciw(kappa$sum.kvar, kappa$sum.n)
  print(paste(
    round((1-alpha)*100,0), "% confidence limits for kappa:",
    round((kappa$sum.kappa-w), 4), "...",
    round((kappa$sum.kappa+w), 4)), quote=F, sep="")
  print("Per-class kappa, stdev, & CV%, for user's accuracy:", quote=F)
  print(round(kappa$user.kappa, 4), quote=F);
  print(round(sqrt(kappa$user.kvar), 4), quote=F);
  print(round((sqrt(kappa$user.kvar)/kappa$user.kappa)*1000,0)/10, quote=F);
  print("Per-class kappa, stdev, & CV%, for producer's reliability:", quote=F)
  print(round(kappa$prod.kappa, 4), quote=F);
  print(round(sqrt(kappa$prod.kvar), 4), quote=F);
  print(round((sqrt(kappa$prod.kvar)/kappa$prod.kappa)*1000,0)/10, quote=F);
}
```

---

## 5.2 *Tau*

Second, two procedures to compute and print *tau*. In addition to the confusion matrix CM, the `tau()` function takes an optional second argument `P`, which is a vector with the prior probability of

each class; if this is not given, each class is given an equal prior.

---

### **R code:**

```
tau <- function(CM, P) {
  #convert both data frames and vectors to matrices
  cmx<-as.matrix(CM)
  #try to convert a vector to a square matrix
  if (ncol(cmx) == 1)
    cmx<-matrix(cmx, byrow=TRUE, nrow=sqrt(nrow(cmx)))
  nr<-nrow(cmx); nc<-ncol(cmx)
  if (nr != nc)
    { print("Error: matrix is not square"); break }
  #check P and create if necessary
  if (missing(P))
    P<-rep(1/nr, nr)
  if (length(P) != nc)
    { print("Error: prior probabilities vector has wrong length"); break }
  if (abs(1-sum(P)) > 0.0001)
    { print("Error: prior probabilities must sum to 1"); break }
  n<-sum(cmx)
  d<-diag(cmx); dsum<-sum(d); th1<-dsum/n
  csum<-apply(cmx,2,sum); th2<-(csum%*%P)/n
  tau<-(th1-th2)/(1-th2);
  th3<-sum( (csum + (P*n)) * diag(cmx) ) / n^2;
  rsum<-apply(cmx,1,sum)
  ua<-d/rsum; pa<-d/csum
  th4 <- 0; for (i in 1:nr) for (j in 1:nc)
    th4 <- th4 + (cmx[i,j] * ((csum[i] + P[j]*n)^2));
  th4 <- th4 / n^3;
  th1c <- 1 - th1; th2c <- 1 - th2;
  tv <- 1/n *
  (
    ( th1 * th1c ) / th2c^2 )
  + ( ( 2 * th1c * ((2*th1*th2) - th3) ) / th2c^3 )
  + ( ( th1c^2 * ( th4 - (4 * th2^2) ) ) / th2c^4 )
  )
  list(prior=P, obs=rsum, ref=csum, n=n, tau=tau, tvar=tv, coeff=c(th1, th2, th3, th4))
}
```

---

---

**R code:**

```
summary.tau <- function(tau, alpha=0.05) {
  ciw<-function(var, n) {
    qnorm(1-(alpha/2))*sqrt(var) + (1/(2*n))
  }
  print(paste("Number of observations:", tau$n), quote=F)
  print("Prior class probabilities:", quote=F)
  print(tau$prior, quote=F)
  print("Observed class proportions:", quote=F)
  print(round(tau$obs/tau$n,4), quote=F)
  print("Reference class proportions:", quote=F)
  print(round(tau$ref/tau$n,4), quote=F)
  print(paste("Tau, stdev, & CV%:",
    round(tau$tau,4), ",",
    round(sqrt(tau$stvar),4), ",",
    round((sqrt(tau$stvar)/tau$tau)*1000,0)/10), quote=F)
  w<-ciw(tau$stvar, tau$n)
  print(paste(round((1-alpha)*100,0),"% confidence limits for tau:",
    round((tau$tau-w),4), "...", round((tau$tau+w),4), sep=""), quote=F)
}
```

---

### 5.3 Weighted naïve and *kappa* statistics

Third, an R function to compute *weighted* naïve and kappa statistics, overall and per-class for a square confusion matrix  $CM$  and a weights matrix  $W$ , which has 1's on the diagonals, and [0..1) on the off-diagonals. The results are returned in a list, which can be printed with `summary.kw()`.

---

## R code:

```
kw <- function(CM, W = diag(sqrt(length(as.matrix(CM)))) ) {
  cmx<-as.matrix(CM); wx<-as.matrix(W)
  #try to convert a vector to a square matrix
  if (ncol(cmx) == 1)
    cmx<-matrix(cmx, byrow=TRUE, nrow=sqrt(nrow(cmx)))
  if (ncol(wx) == 1)
    wx<-matrix(wx, byrow=TRUE, nrow=sqrt(nrow(wx)))
  nr<-nrow(cmx); nc<-ncol(cmx)
  if (nr != nc) { print("Error: confusion matrix is not square"); break }
  if (dim(wx) != dim(cmx))
    { print("Weight and Confusion Matrices are not the same size"); break }
  #summarize cmx
  n<-sum(cmx); rs<-apply(cmx,1,sum); cs<-apply(cmx,2,sum)
  # confusion matrix and marginals as proportions
  p <- cmx/n; cp <- cs/n; rp <- rs/n;
  if ( round((sum(rp) + sum(cp))/2, 2) != 1)
    { print("Error: Bad checksum in row proportions"); break }
  # expected proportions
  pp<- rp %o% cp
  # weighted weights
  wr <- wx%*%cp; wc <- t(t(wx)%*%rp);
  # marginal accuracy
  # rows = user's
  ua <- apply(wx*p,1,sum)/rp; uasd<-sqrt(ua*(1-ua)/rs);
  # columns = producer's
  pa <- apply(wx*p,2,sum)/cp; pasd<-sqrt(pa*(1-pa)/cs);
  thw1 <- sum(sum(p * wx)); thw1v<-((thw1*(1-thw1))/n)
  thw2 <- sum(sum(pp * wx));
  khw <- (thw1-thw2)/(1-thw2);
  thw1c <- 1 - thw1; thw2c <- 1 - thw2;
  thw4 <- 0; for (i in 1:nr) for (j in 1:nc)
    thw4 <- thw4 + (p[i,j]*((wx[i,j]*thw2c - (wr[i]+wc[j]) * thw1c)^2 ))
  khwv <- (thw4 - (thw1*thw2 - 2*thw2 + thw1)^2) / (n * thw2c^4)
  list(
    sum.n=n,
    sum.kappa=khw, sum.kvar=khwv, theta=c(thw1,thw2,thw4),
    sum.naive=thw1, sum.var=thw1v,
    user.wa=ua, prod.wa=pa,
    user.wsd=uasd, prod.wsd=pasd,
    weights.row=wr, weights.col=wc, expected=pp)
}
```

---

The following function reports the summary statistics computed by `kw()`:

---

## R code:

```
summary.kw <- function(kw, alpha=0.05) {
  ciw<-function(var, n) {
    qnorm(1-(alpha/2))*sqrt(var) + (1/(2*n))
  }
  print(paste("Number of observations:", kw$sum.n), quote=F)
  print(paste("Sum of weighted sum of row, column weights:",
    round(sum(kw$weights.row), 2), ",",
    round(sum(kw$weights.col), 2) ), quote=F)
  print("Summary of weighted naive statistics", quote=F)
  print(paste(
    "Overall accuracy, stdev, CV%:",
    round(kw$sum.naive, 4), ",", round(sqrt(kw$sum.var), 4), ",",
    round((sqrt(kw$sum.var)/kw$sum.naive)*1000,0)/10,
    quote=F)
  w<-ciw(kw$sum.var, kw$sum.n)
  print(paste(
    round((1-alpha)*100,0), "% confidence limits for accuracy:",
    round((kw$sum.naive-w), 4), "...",
    round((kw$sum.naive+w), 4), sep=""), quote=F)
  print("User's weighted accuracy", quote=F)
  print(round(kw$user.wa, 4));
  print("Producer's weighted reliability:", quote=F)
  print(round(kw$prod.wa, 4));
  print("Summary of weighted kappa statistics", quote=F)
  print(paste("Overall weighted kappa, stdev, & CV%:",
    round(kw$sum.kappa, 4), ",",
    round(sqrt(kw$sum.kvar), 4), ",",
    round((sqrt(kw$sum.kvar)/kw$sum.kappa)*1000,0)/10), quote=F)
  w<-ciw(kw$sum.kvar, kw$sum.n)
  print(paste(
    round((1-alpha)*100,0), "% confidence limits for weighted kappa:",
    round((kw$sum.kappa-w), 4), "...",
    round((kw$sum.kappa+w), 4), sep=""), quote=F)
}
```

---

## 5.4 Computing exact confidence intervals of the binomial distribution

The following methods are copied (by permission of the author, Dr. Tomás Aragón) from the EpiTools.net website<sup>3</sup>. The third-listed method (`ci.binomial`), is designed to be called by the end user. This calls the first two listed methods (which must be defined prior to being called by the third) to compute the upper (`uci.binomial`) and lower (`lci.binomial`) confidence intervals

<sup>3</sup> <http://www.medepi.org/epitools/rfunctions/index.html>

for binomial counts. By default, the 95% confidence intervals are calculated; the optional third argument can be used to compute other intervals.

---

**R code:**

```
uci.binomial <- function(x,n,a=0,b=1,alpha=.05){
# Finds the exact upper confidence limit for binomial count
# R v0.90.1, Tomas Aragon, created 01/01/2000, edited
# x = observed count (non-negative integer)
# n = number of Bernoulli trials
# a = default lower bound for bisection method
# b = default upper bound for bisection method
# alpha = .05 (default), for 95% confidence interval
# This function is used by ci.binomial()
#
if(x==0) answer <- 1-(alpha)^(1/n)
else{
  answer <- (a+b)/2
  if(abs(a-b) >= 0.00000001){
    lefta <- pbinom(x,n,a)-alpha/2
    centera <- pbinom(x,n,answer)-alpha/2
    righta <- pbinom(x,n,b)-alpha/2
    if(lefta*centera < 0){
      answer <- uci.binomial(x,n,a,answer)
    }
    else{
      answer <- uci.binomial(x,n,answer,b)
    }
  }
}
}
answer
}
```

---

---

**R code:**

```
lci.binomial <- function(x,n,a=0,b=1,alpha=.05){
# Finds the exact lower confidence limit for binomial count
# R v0.90.1, Tomas Aragon, created 01/01/2000, edited
# x = observed count (non-negative integer)
# n = number of Bernoulli trials
# a = default lower bound for bisection method
# b = default upper bound for bisection method
# alpha = .05 (default), for 95% confidence interval
# This function is used by ci.binomial()
#
if(x==0) answer <- 0
else{
  answer <- (a+b)/2
  if(abs(a-b) >= 0.00000001){
    lefta <- 1-pbinom(x,n,a)+dbinom(x,n,a)-alpha/2
    centera <- 1-pbinom(x,n,answer)+dbinom(x,n,answer)-alpha/2
    righta <- 1-pbinom(x,n,b)+dbinom(x,n,b)-alpha/2
    if(lefta*centera < 0){
      answer <- lci.binomial(x,n,a,answer)
    }
    else{
      answer <- lci.binomial(x,n,answer,b)
    }
  }
}
answer
}
```

---

---

**R code:**

```
ci.binomial <- function(x,n,alpha=0.05){
# Finds the exact 95% confidence interval for binomial count
# R v0.90.1, Tomas Aragon, created 01/01/2000, edited
# x = observed count (non-negative integer)
# number of Bernoulli trials
# alpha = .05 (default), for 95% confidence interval
# Output includes x and CIs, and p=x/n and CIs
# Uses lci.binomial() and uci.binomial() for bisection method
#
xn <- cbind(x,n)
ci.pois <- function(xx,aa=alpha){
lci <- lci.binomial(xx[1],xx[2],alpha=aa)
uci <- uci.binomial(xx[1],xx[2],alpha=aa)
c(lci=lci,uci=uci)
}
prob <- t(apply(xn,1,ci.pois))
ans <- cbind(as.vector(x),n,prob*n,as.vector(x/n),prob)
dimnames(ans)[[2]] <- c("x","n","x.lci","x.uci","p=x/n","p.lci","p.uci")
ans
}
```

---

## 5.5 Numerical Example

First, the confusion matrix must be created as an R object.

One way to create the matrix is to read it directly from the console into object `cm` with the `scan()` method, and then use the `matrix()` method to convert it to a matrix. Since we chose to enter the values in row-major order (i.e. by mapped class), we must use the optional `byrow=T` argument to the `matrix()` method. Then, we name the rows and columns with the class. Finally, we examine the result.

---

**R code:**

```
cm<-scan()
35 14 11 1 4 11 3 0 12 9 38 4 2 5 12 2

cm<-matrix(cm, 4, 4, byrow=T)
rownames(cm)<-c("A","B","C","D")
colnames(cm)<-c("A","B","C","D")
str(cm)
```

---

---

**R console output:**

```
  A  B  C  D
A 35 14 11 1
B  4 11  3 0
C 12  9 38 4
D  2  5 12 2
```

---

Another way to create the confusion matrix is to prepare it in a text editor or Excel as a comma-separated values (“CSV”) file, with row and column labels, and then import it with the `read.csv()` method.

Here is the CSV file for our example matrix, as prepared in a text editor; note that the first row has the column names (reference classes) and each of the other rows begins with the row name (mapped class); there is no entry for the very first column of the first row:

```
, "A", "B", "C", "D"
"A", 35, 14, 11, 1
"B", 4, 11, 3, 0
"C", 12, 9, 38, 4
"D", 2, 5, 12, 2
```

This can be read in as follows; note that we inform R that the row names are in the first column:

---

**R code:**

```
cm<-read.csv("cm.csv", row.names=1)
```

---

With the confusion matrix in hand, we can now compute the naïve, *kappa*, and *tau* statistics, the latter for various prior probabilities, and print the results with various probabilities of Type I error:

---

**R code:**

```
x<-kappa(cm); summary.kappa(x)
summary.kappa(x, alpha=0.1)
x<-tau(cm); summary.tau(x)
x<-tau(cm, c(.1,.4,.1,.4)); summary.tau(x)
```

---

---

**R console output:**

```
[1] Number of observations: 163
[1] Summary of naive statistics
[1] Overall accuracy, stdev, CV%: 0.5276 , 0.0391 , 7.4
[1] 95 % confidence limits for accuracy: 0.4479 ... 0.6073
[1] User's accuracy
      A      B      C      D
0.5738 0.6111 0.6032 0.0952
[1] Producer's reliability:
      A      B      C      D
0.6604 0.2821 0.5938 0.2857
[1] Summary of kappa statistics
[1] Overall kappa, stdev, & CV%: 0.3199 , 0.0523 , 16.4
[1] 95 % confidence limits for kappa: 0.2143 ... 0.4256
[1] Per-class kappa, stdev, & CV%, for user's accuracy:
      A      B      C      D
0.3684 0.4888 0.3466 0.0546
      A      B      C      D
0.0763 0.1440 0.0824 0.0603
      A      B      C      D
20.7 29.5 23.8 110.3
[1] Per-class kappa, stdev, & CV%, for producer's reliability:
      A      B      C      D
0.4573 0.1929 0.3378 0.1801
      A      B      C      D
0.0899 0.0673 0.0806 0.1906
      A      B      C      D
19.6 34.9 23.9 105.8
```

---

---

**R console output:**

```
[1] Number of observations: 163
[1] Summary of naive statistics
[1] Overall accuracy, stdev, CV%: 0.5276 , 0.0391 , 7.4
[1] 90 % confidence limits for accuracy: 0.4602 ... 0.595
[1] User's accuracy
      A      B      C      D
0.5738 0.6111 0.6032 0.0952
[1] Producer's reliability:
      A      B      C      D
0.6604 0.2821 0.5938 0.2857
[1] Summary of kappa statistics
[1] Overall kappa, stdev, & CV%: 0.3199 , 0.0523 , 16.4
[1] 90 % confidence limits for kappa: 0.2308 ... 0.4091
[1] Per-class kappa, stdev, & CV%, for user's accuracy:
      A      B      C      D
0.3684 0.4888 0.3466 0.0546
      A      B      C      D
0.0763 0.1440 0.0824 0.0603
      A      B      C      D
20.7 29.5 23.8 110.3
[1] Per-class kappa, stdev, & CV%, for producer's reliability:
      A      B      C      D
0.4573 0.1929 0.3378 0.1801
      A      B      C      D
0.0899 0.0673 0.0806 0.1906
      A      B      C      D
19.6 34.9 23.9 105.8
```

---

**R console output:**

```
[1] Number of observations: 163
[1] Prior class probabilities:
[1] 0.25 0.25 0.25 0.25
[1] Observed class proportions:
      A      B      C      D
0.3742 0.1104 0.3865 0.1288
[1] Reference class proportions:
      A      B      C      D
0.3252 0.2393 0.3926 0.0429
[1] Tau, stdev, & CV%: 0.3701 , 0.0489 , 13.2
[1] 95% confidence limits for tau: 0.2712 ... 0.4691
```

---

---

**R console output:**

```
[1] Number of observations: 163
[1] Prior class probabilities:
[1] 0.1 0.4 0.1 0.4
[1] Observed class proportions:
      A      B      C      D
0.3742 0.1104 0.3865 0.1288
[1] Reference class proportions:
      A      B      C      D
0.3252 0.2393 0.3926 0.0429
[1] Tau, stdev, & CV%: 0.4206 , 0.0454 , 10.8
[1] 95% confidence limits for tau: 0.3285 ... 0.5127
```

---

Finally, we want to compute weighted accuracy statistics. So, we read in a weights matrix, also by row, and convert it to the required form. Then we apply method `kw()`.

---

**R code:**

```
w<-scan()
1 0 .67 1 0 1 0 0 1 0 1 1 .91 0 .61 1

w<-matrix(w, 4, 4, byrow=T)
x<-kw(cm,w)
summary.kw(x, 0.01)
```

---

**R console output:**

```
[1] Number of observations: 163
[1] Sum of weighted sum of row, column weights: 2.21 , 2.59
[1] Summary of weighted naive statistics
[1] Overall accuracy, stdev, CV%: 0.7332 , 0.0346 , 4.7
[1] 99% confidence limits for accuracy:0.6409...0.8255
[1] User's weighted accuracy
      A      B      C      D
0.7110 0.6111 0.8571 0.5305
[1] Producer's weighted reliability:
      A      B      C      D
0.9211 0.2821 0.8233 1.0000
[1] Summary of weighted kappa statistics
[1] Overall weighted kappa, stdev, & CV%: 0.2766 , 0.0689 , 24.9
[1] 99% confidence limits for weighted kappa:0.0962...0.4571
```

---

## 5.6 Computing normal scores with R

The Z-score is the ordinate on the cumulative normal distribution, i.e. the number of standard deviations from the mean. For known values of  $1 - \alpha$ , they can be computed with the `qnorm()` method, which takes one required argument, the one-tailed probability:

---

### R code:

```
qnorm(.95)
qnorm(c(.8, .9, .95, .99, .999))
```

---

### R console output:

```
[1] 1.644854
[1] 0.8416212 1.2815516 1.6448536 2.3263479 3.0902323
```

---

The probabilities of Type I error  $\alpha$  are generally two-tailed, i.e. there is no reason to believe that one of the two values to compare is *a priori* higher than the other. Then the Z-score is computed as `qnorm(1 - (alpha/2))`, e.g.:

---

### R code:

```
alpha<-.05; qnorm(1 - (alpha/2))
```

---

### R console output:

```
[1] 1.959964
```

---

In the reverse direction, i.e. with known Z-score, the corresponding *p*-value is computed with the `pnorm()` method, which takes one required argument, the two-tailed Z-score:

---

### R code:

```
dnorm(2.0532)
alpha<-.05; pnorm(qnorm(1 - (alpha/2)))
```

---

---

**R console output:**

```
[1] 1.959964  
[1] 0.975
```

---

## 6 Further reading

The fundamental principles of thematic map accuracy are well-presented in a review paper by Stehman and Czaplewski [31]. The same author has also discussed many of the issues in the present note [28].

Accuracy assessment has been extensively studied for various kinds of thematic maps, most notably land use and land cover maps produced by satellite remote sensing [7, 8, 23], and a wide variety of statistical techniques have been proposed for this purpose [9, 18, 20, 13].

## References

- [1] Y Bishop, S Fienberg, and P Holland. *Discrete multivariate analysis: theory and practice*. MIT Press, Cambridge, MA, 1975.
- [2] R L Brennan and D J Prediger. Coefficient kappa: some uses, misuses, and alternatives. *Educational and Psychological Measurement*, 41:687–699, 1981.
- [3] W G Cochran. *Sampling Techniques*. John Wiley, New York, 3rd edition, 1977.
- [4] J Cohen. A coefficient of agreement for nominal scales. *Educational & Psychological Measurement*, 20:37–46, 1960.
- [5] J Cohen. Weighted kappa: Nominal scale agreement with provision for scaled disagreement or partial credit. *Psychological Bulletin*, 70:426–443, 1968.
- [6] R G Congalton. A comparison of sampling schemes used in generating error matrices for assessing the accuracy of maps generated from remotely sensed data. *Photogrammetric Engineering & Remote Sensing*, 54:593–600, 1988.
- [7] R G Congalton. A review of assessing the accuracy of classifications of remotely sensed data. *Remote Sensing of Environment*, 37:35–46, 1991.
- [8] R G Congalton and K Green. *Assessing the accuracy of remotely sensed data: principles and practices*. Lewis, Boca Raton, FL, 1999.
- [9] R G Congalton, R G Oderwald, and R A Mead. Assessing landsat classification accuracy using discrete multivariate-analysis statistical techniques. *Photogrammetric Engineering & Remote Sensing*, 49(12):1671–1678, 1983.
- [10] W Feller. *An introduction to probability theory and its applications*, volume 1. John Wiley & Sons, New York, 3rd edition, 1968.
- [11] J L Fleiss, J Cohen, and B S Everitt. Large sample standard errors of kappa and weighted kappa. *Psychological Bulletin*, 72(5):323–327, 1969.
- [12] G M Foody. On the compensation for chance agreement in

- image classification and accuracy assessment. *Photogrammetric Engineering & Remote Sensing*, 58:1459–1460, 1992.
- [13] S Gopal and C Woodcock. Theory and methods for accuracy assessment of thematic maps using fuzzy sets. *Photogrammetric Engineering & Remote Sensing*, 60(2):181–188, 1994.
- [14] B Gorte. *Probabilistic segmentation of remotely sensed images*. ITC Publication 63. International Institute for Aerospace Survey & Earth Sciences (ITC), Enschede, the Netherlands, 1998.
- [15] W D Hudson and C W Ramm. Correct formulation of the Kappa coefficient of agreement (in remote sensing). *Photogrammetric Engineering & Remote Sensing*, 53(4):421–422, 1987.
- [16] R Ihaka and R Gentleman. R: A language for data analysis and graphics. *Journal of Computational and Graphical Statistics*, 5(3):299–314, 1996.
- [17] T M Lillesand and R W Kiefer. *Remote sensing and image interpretation*. John Wiley & Sons, New York, 3rd edition, 1994.
- [18] Z Ma and R L Redmond. Tau coefficients for accuracy assessment of classification of remote sensing data. *Photogrammetric Engineering & Remote Sensing*, 61(4):435–439, 1995.
- [19] R A Monserud and R Leemans. Comparing global vegetation maps with the Kappa statistic. *Ecological Modelling*, 62(4):275–293, 1992.
- [20] E Næsset. Conditional tau coefficient for assessment of producer’s accuracy of classified remotely sensed data. *ISPRS Journal of Photogrammetry & Remote Sensing*, 51(2):91–98, 1996.
- [21] E Næsset. Use of the weighted Kappa coefficient in classification error assessment of thematic maps. *International Journal of Geographical Information Systems*, 10(5):591–603, 1996.
- [22] D G Rossiter. *Introduction to the R Project for Statistical Computing for use at ITC*. International Institute for Geo-information Science & Earth Observation (ITC), En-

schede (NL), 2003. URL: [http://www.itc.nl/personal/rossiter/teach/R/RIntro\\_ITC.pdf](http://www.itc.nl/personal/rossiter/teach/R/RIntro_ITC.pdf).

- [23] A K Skidmore. Accuracy assessment of spatial information. In A Stein, F van der Meer, and B G F Gorte, editors, *Spatial statistics for remote sensing*, pages 197–209. Kluwer Academic, Dordrecht, 1999.
- [24] R G D Steel and J H Torrie. *Principles and procedures of statistics: a biometrical approach*. McGraw-Hill, New York, 2nd edition, 1980.
- [25] S V Stehman. Comparison of systematic and random sampling for estimating the accuracy of maps generated from remotely-sensed data. *Photogrammetric Engineering & Remote Sensing*, 58:1343–1350, 1992.
- [26] S V Stehman. Estimating the kappa coefficient and its variance under stratified random sampling. *Photogrammetric Engineering & Remote Sensing*, 62(4):401–407, 1996.
- [27] S V Stehman. Estimating standard errors of accuracy assessment statistics under cluster sampling. *Remote Sensing of Environment*, 60(3):258–269, 1997.
- [28] S V Stehman. Selecting and interpreting measures of thematic classification accuracy. *Remote Sensing of Environment*, 62(1):77–89, 1997.
- [29] S V Stehman. Basic probability sampling designs for thematic map accuracy assessment. *International Journal of Remote Sensing*, 20(12):2423–2441, 1999.
- [30] S V Stehman. Comparing thematic maps based on map value. *International Journal of Remote Sensing*, 20(12):2347–2366, 1999.
- [31] S V Stehman and R L Czaplewski. Design and analysis for thematic map accuracy assessment: Fundamental principles. *Remote Sensing of Environment*, 64(3):331–344, 1998.
- [32] S J Ventura and B J Irvin. Automated landform classification methods for soil-landscape studies. In J P Wilson and J Gallant, editors, *Terrain analysis : principles and applications*, pages 267–294. Wiley & Sons, New York, 2000.