Chapter 14. Statistical Description of Data

14.0 Introduction

In this chapter and the next, the concept of *data* enters the discussion more prominently than before.

Data consist of numbers, of course. But these numbers are fed into the computer, not produced by it. These are numbers to be treated with considerable respect, neither to be tampered with, nor subjected to a numerical process whose character you do not completely understand. You are well advised to acquire a reverence for data that is rather different from the "sporty" attitude that is sometimes allowable, or even commendable, in other numerical tasks.

The analysis of data inevitably involves some trafficking with the field of *statistics*, that gray area which is not quite a branch of mathematics — and just as surely not quite a branch of science. In the following sections, you will repeatedly encounter the following paradigm:

- apply some formula to the data to compute "a statistic"
- compute where the value of that statistic falls in a probability distribution that is computed on the basis of some "null hypothesis"
- if it falls in a very unlikely spot, way out on a tail of the distribution, conclude that the null hypothesis is *false* for your data set

If a statistic falls in a *reasonable* part of the distribution, you must not make the mistake of concluding that the null hypothesis is "verified" or "proved." That is the curse of statistics, that it can never prove things, only disprove them! At best, you can substantiate a hypothesis by ruling out, statistically, a whole long list of competing hypotheses, every one that has ever been proposed. After a while your adversaries and competitors will give up trying to think of alternative hypotheses, or else they will grow old and die, and *then your hypothesis will become accepted.* Sounds crazy, we know, but that's how science works!

In this book we make a somewhat arbitrary distinction between data analysis procedures that are *model-independent* and those that are *model-dependent*. In the former category, we include so-called *descriptive statistics* that characterize a data set in general terms: its mean, variance, and so on. We also include statistical tests that seek to establish the "sameness" or "differentness" of two or more data sets, or that seek to establish and measure a degree of *correlation* between two data sets. These subjects are discussed in this chapter.

In the other category, model-dependent statistics, we lump the whole subject of fitting data to a theory, parameter estimation, least-squares fits, and so on. Those subjects are introduced in Chapter 15.

Section 14.1 deals with so-called *measures of central tendency*, the moments of a distribution, the median and mode. In §14.2 we learn to test whether different data sets are drawn from distributions with different values of these measures of central tendency. This leads naturally, in §14.3, to the more general question of whether two distributions can be shown to be (significantly) different.

In §14.4–§14.7, we deal with *measures of association* for two distributions. We want to determine whether two variables are "correlated" or "dependent" on one another. If they are, we want to characterize the degree of correlation in some simple ways. The distinction between parametric and nonparametric (rank) methods is emphasized.

Section 14.8 introduces the concept of data smoothing, and discusses the particular case of Savitzky-Golay smoothing filters.

This chapter draws mathematically on the material on special functions that was presented in Chapter 6, especially §6.1–§6.4. You may wish, at this point, to review those sections.

CITED REFERENCES AND FURTHER READING:

Bevington, P.R. 1969, *Data Reduction and Error Analysis for the Physical Sciences* (New York: McGraw-Hill).

Stuart, A., and Ord, J.K. 1987, *Kendall's Advanced Theory of Statistics*, 5th ed. (London: Griffin and Co.) [previous eds. published as Kendall, M., and Stuart, A., *The Advanced Theory of Statistics*].

Norusis, M.J. 1982, *SPSS Introductory Guide: Basic Statistics and Operations*; and 1985, *SPSS-X Advanced Statistics Guide* (New York: McGraw-Hill).

Dunn, O.J., and Clark, V.A. 1974, *Applied Statistics: Analysis of Variance and Regression* (New York: Wiley).

14.1 Moments of a Distribution: Mean, Variance, Skewness, and So Forth

When a set of values has a sufficiently strong central tendency, that is, a tendency to cluster around some particular value, then it may be useful to characterize the set by a few numbers that are related to its *moments*, the sums of integer powers of the values.

Best known is the *mean* of the values $x_1, \ldots, x_N$,

$$\overline{x} = \frac{1}{N} \sum_{j=1}^{N} x_j \tag{14.1.1}$$

which estimates the value around which central clustering occurs. Note the use of an overbar to denote the mean; angle brackets are an equally common notation, e.g., $\langle x \rangle$. You should be aware that the mean is not the only available estimator of this

quantity, nor is it necessarily the best one. For values drawn from a probability distribution with very broad "tails," the mean may converge poorly, or not at all, as the number of sampled points is increased. Alternative estimators, the *median* and the *mode*, are mentioned at the end of this section.

Having characterized a distribution's central value, one conventionally next characterizes its "width" or "variability" around that value. Here again, more than one measure is available. Most common is the *variance*,

$$\mathrm{Var}(x_1 \dots x_N) = \frac{1}{N-1} \sum_{j=1}^{N} (x_j - \overline{x})^2 \tag{14.1.2}$$

or its square root, the *standard deviation*,

$$\sigma(x_1 \dots x_N) = \sqrt{\mathrm{Var}(x_1 \dots x_N)} \tag{14.1.3}$$

Equation (14.1.2) estimates the mean squared deviation of x from its mean value. There is a long story about why the denominator of (14.1.2) is $N-1$ instead of N. If you have never heard that story, you may consult any good statistics text. Here we will be content to note that the $N-1$ *should* be changed to N if you are ever in the situation of measuring the variance of a distribution whose mean $\overline{x}$ is known *a priori* rather than being estimated from the data. (We might also comment that if the difference between N and $N-1$ ever matters to you, then you are probably up to no good anyway — e.g., trying to substantiate a questionable hypothesis with marginal data.)

As the mean depends on the first moment of the data, so do the variance and standard deviation depend on the second moment. It is not uncommon, in real life, to be dealing with a distribution whose second moment does not exist (i.e., is infinite). In this case, the variance or standard deviation is useless as a measure of the data's width around its central value: The values obtained from equations (14.1.2) or (14.1.3) will not converge with increased numbers of points, nor show any consistency from data set to data set drawn from the same distribution. This can occur even when the width of the peak looks, by eye, perfectly finite. A more robust estimator of the width is the *average deviation* or *mean absolute deviation*, defined by

$$\mathrm{ADev}(x_1 \dots x_N) = \frac{1}{N} \sum_{j=1}^{N} |x_j - \overline{x}| \tag{14.1.4}$$

One often substitutes the sample median x_{med} for $\overline{x}$ in equation (14.1.4). For any fixed sample, the median in fact minimizes the mean absolute deviation.

Statisticians have historically sniffed at the use of (14.1.4) instead of (14.1.2), since the absolute value brackets in (14.1.4) are "nonanalytic" and make theorem-proving difficult. In recent years, however, the fashion has changed, and the subject of *robust estimation* (meaning, estimation for broad distributions with significant numbers of "outlier" points) has become a popular and important one. Higher moments, or statistics involving higher powers of the input data, are almost always less robust than lower moments or statistics that involve only linear sums or (the lowest moment of all) counting.

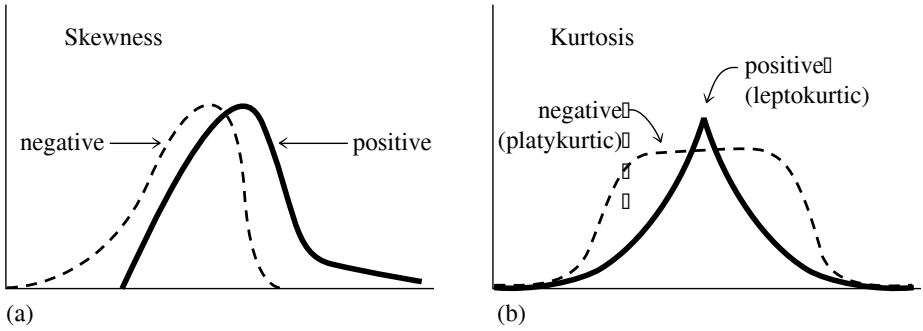

Figure 14.1.1. Distributions whose third and fourth moments are significantly different from a normal (Gaussian) distribution. (a) Skewness or third moment. (b) Kurtosis or fourth moment.

That being the case, the *skewness* or *third moment*, and the *kurtosis* or *fourth moment* should be used with caution or, better yet, not at all.

The skewness characterizes the degree of asymmetry of a distribution around its mean. While the mean, standard deviation, and average deviation are *dimensional* quantities, that is, have the same units as the measured quantities x_j, the skewness is conventionally defined in such a way as to make it *nondimensional*. It is a pure number that characterizes only the shape of the distribution. The usual definition is

$$\text{Skew}(x_1 \dots x_N) = \frac{1}{N} \sum_{j=1}^{N} \left[\frac{x_j - \overline{x}}{\sigma} \right]^3 \tag{14.1.5}$$

where $\sigma = \sigma(x_1 \dots x_N)$ is the distribution's standard deviation (14.1.3). A positive value of skewness signifies a distribution with an asymmetric tail extending out towards more positive x; a negative value signifies a distribution whose tail extends out towards more negative x (see Figure 14.1.1).

Of course, any set of N measured values is likely to give a nonzero value for (14.1.5), even if the underlying distribution is in fact symmetrical (has zero skewness). For (14.1.5) to be meaningful, we need to have some idea of *its* standard deviation as an estimator of the skewness of the underlying distribution. Unfortunately, that depends on the shape of the underlying distribution, and rather critically on its tails! For the idealized case of a normal (Gaussian) distribution, the standard deviation of (14.1.5) is approximately $\sqrt{15/N}$ when $\overline{x}$ is the true mean, and $\sqrt{6/N}$ when it is estimated by the sample mean, (14.1.1). In real life it is good practice to believe in skewnesses only when they are several or many times as large as this.

The kurtosis is also a nondimensional quantity. It measures the relative peakedness or flatness of a distribution. Relative to what? A normal distribution, what else! A distribution with positive kurtosis is termed *leptokurtic*; the outline of the Matterhorn is an example. A distribution with negative kurtosis is termed *platykurtic*; the outline of a loaf of bread is an example. (See Figure 14.1.1.) And, as you no doubt expect, an in-between distribution is termed *mesokurtic*.

The conventional definition of the kurtosis is

$$\text{Kurt}(x_1 \dots x_N) = \left\{ \frac{1}{N} \sum_{j=1}^{N} \left[\frac{x_j - \overline{x}}{\sigma} \right]^4 \right\} - 3 \tag{14.1.6}$$

where the -3 term makes the value zero for a normal distribution.

The standard deviation of (14.1.6) as an estimator of the kurtosis of an underlying normal distribution is $\sqrt{96/N}$ when σ is the true standard deviation, and $\sqrt{24/N}$ when it is the sample estimate (14.1.3). However, the kurtosis depends on such a high moment that there are many real-life distributions for which the standard deviation of (14.1.6) as an estimator is effectively infinite.

Calculation of the quantities defined in this section is perfectly straightforward. Many textbooks use the binomial theorem to expand out the definitions into sums of various powers of the data, e.g., the familiar

$$\text{Var}(x_1 \dots x_N) = \frac{1}{N-1}\left[\left(\sum_{j=1}^{N} x_j^2\right) - N\overline{x}^2\right] \approx \overline{x^2} - \overline{x}^2 \tag{14.1.7}$$

but this can magnify the roundoff error by a large factor and is generally unjustifiable in terms of computing speed. A clever way to minimize roundoff error, especially for large samples, is to use the *corrected two-pass algorithm* [1]: First calculate $\overline{x}$, then calculate $\text{Var}(x_1 \dots x_N)$ by

$$\text{Var}(x_1 \dots x_N) = \frac{1}{N-1}\left\{\sum_{j=1}^{N}(x_j - \overline{x})^2 - \frac{1}{N}\left[\sum_{j=1}^{N}(x_j - \overline{x})\right]^2\right\} \tag{14.1.8}$$

The second sum would be zero if $\overline{x}$ were exact, but otherwise it does a good job of correcting the roundoff error in the first term.

```
#include <math.h>

void moment(float data[], int n, float *ave, float *adev, float *sdev,
    float *var, float *skew, float *curt)
Given an array of data[1..n], this routine returns its mean ave, average deviation adev,
standard deviation sdev, variance var, skewness skew, and kurtosis curt.
{
    void nrerror(char error_text[]);
    int j;
    float ep=0.0,s,p;

    if (n <= 1) nrerror("n must be at least 2 in moment");
    s=0.0;                                  First pass to get the mean.
    for (j=1;j<=n;j++) s += data[j];
    *ave=s/n;
    *adev=(*var)=(*skew)=(*curt)=0.0;       Second pass to get the first (absolute), sec-
    for (j=1;j<=n;j++) {                        ond, third, and fourth moments of the
        *adev += fabs(s=data[j]-(*ave));        deviation from the mean.
        ep += s;
        *var += (p=s*s);
        *skew += (p *= s);
        *curt += (p *= s);
    }
    *adev /= n;
    *var=(*var-ep*ep/n)/(n-1);              Corrected two-pass formula.
    *sdev=sqrt(*var);                       Put the pieces together according to the con-
    if (*var) {                                 ventional definitions.
        *skew /= (n*(*var)*(*sdev));
        *curt=(*curt)/(n*(*var)*(*var))-3.0;
    } else nrerror("No skew/kurtosis when variance = 0 (in moment)");
}
```

Semi-Invariants

The mean and variance of independent random variables are additive: If x and y are drawn independently from two, possibly different, probability distributions, then

$$\overline{(x+y)} = \overline{x} + \overline{y} \qquad \text{Var}(x+y) = \text{Var}(x) + \text{Var}(x) \tag{14.1.9}$$

Higher moments are not, in general, additive. However, certain combinations of them, called *semi-invariants*, are in fact additive. If the centered moments of a distribution are denoted M_k,

$$M_k \equiv \left\langle (x_i - \overline{x})^k \right\rangle \tag{14.1.10}$$

so that, e.g., $M_2 = \text{Var}(x)$, then the first few semi-invariants, denoted I_k are given by

$$\begin{aligned} I_2 &= M_2 \qquad I_3 = M_3 \qquad I_4 = M_4 - 3M_2^2 \\ I_5 &= M_5 - 10M_2M_3 \qquad I_6 = M_6 - 15M_2M_4 - 10M_3^2 + 30M_2^3 \end{aligned} \tag{14.1.11}$$

Notice that the skewness and kurtosis, equations (14.1.5) and (14.1.6) are simple powers of the semi-invariants,

$$\text{Skew}(x) = I_3/I_2^{3/2} \qquad \text{Kurt}(x) = I_4/I_2^2 \tag{14.1.12}$$

A Gaussian distribution has all its semi-invariants higher than I_2 equal to zero. A Poisson distribution has all of its semi-invariants equal to its mean. For more details, see [2].

Median and Mode

The median of a probability distribution function $p(x)$ is the value x_{med} for which larger and smaller values of x are equally probable:

$$\int_{-\infty}^{x_{\text{med}}} p(x)\,dx = \frac{1}{2} = \int_{x_{\text{med}}}^{\infty} p(x)\,dx \tag{14.1.13}$$

The median of a distribution is estimated from a sample of values $x_1, \ldots, x_N$ by finding that value x_i which has equal numbers of values above it and below it. Of course, this is not possible when N is even. In that case it is conventional to estimate the median as the mean of the unique *two* central values. If the values x_j $j = 1, \ldots, N$ are sorted into ascending (or, for that matter, descending) order, then the formula for the median is

$$x_{\text{med}} = \begin{cases} x_{(N+1)/2}, & N \text{ odd} \\ \frac{1}{2}(x_{N/2} + x_{(N/2)+1}), & N \text{ even} \end{cases} \tag{14.1.14}$$

If a distribution has a strong central tendency, so that most of its area is under a single peak, then the median is an estimator of the central value. It is a more robust estimator than the mean is: The median fails as an estimator only if the area in the tails is large, while the mean fails if the first moment of the tails is large; it is easy to construct examples where the first moment of the tails is large even though their area is negligible.

To find the median of a set of values, one can proceed by sorting the set and then applying (14.1.14). This is a process of order $N \log N$. You might rightly think

that this is wasteful, since it yields much more information than just the median (e.g., the upper and lower quartile points, the deciles, etc.). In fact, we saw in §8.5 that the element $x_{(N+1)/2}$ can be located in of order N operations. Consult that section for routines.

The *mode* of a probability distribution function $p(x)$ is the value of x where it takes on a maximum value. The mode is useful primarily when there is a single, sharp maximum, in which case it estimates the central value. Occasionally, a distribution will be *bimodal*, with two relative maxima; then one may wish to know the two modes individually. Note that, in such cases, the mean and median are not very useful, since they will give only a "compromise" value between the two peaks.

CITED REFERENCES AND FURTHER READING:

Bevington, P.R. 1969, *Data Reduction and Error Analysis for the Physical Sciences* (New York: McGraw-Hill), Chapter 2.

Stuart, A., and Ord, J.K. 1987, *Kendall's Advanced Theory of Statistics*, 5th ed. (London: Griffin and Co.) [previous eds. published as Kendall, M., and Stuart, A., *The Advanced Theory of Statistics*], vol. 1, §10.15

Norusis, M.J. 1982, *SPSS Introductory Guide: Basic Statistics and Operations*; and 1985, *SPSS-X Advanced Statistics Guide* (New York: McGraw-Hill).

Chan, T.F., Golub, G.H., and LeVeque, R.J. 1983, *American Statistician*, vol. 37, pp. 242–247. [1]

Cramér, H. 1946, *Mathematical Methods of Statistics* (Princeton: Princeton University Press), §15.10. [2]

14.2 Do Two Distributions Have the Same Means or Variances?

Not uncommonly we want to know whether two distributions have the same mean. For example, a first set of measured values may have been gathered before some event, a second set after it. We want to know whether the event, a "treatment" or a "change in a control parameter," made a difference.

Our first thought is to ask "how many standard deviations" one sample mean is from the other. That number may in fact be a useful thing to know. It does relate to the strength or "importance" of a difference of means *if that difference is genuine*. However, by itself, it says nothing about whether the difference *is* genuine, that is, statistically significant. A difference of means can be very small compared to the standard deviation, and yet very significant, if the number of data points is large. Conversely, a difference may be moderately large but not significant, if the data are sparse. We will be meeting these distinct concepts of *strength* and *significance* several times in the next few sections.

A quantity that measures the significance of a difference of means is not the number of standard deviations that they are apart, but the number of so-called *standard errors* that they are apart. The standard error of a set of values measures the accuracy with which the sample mean estimates the population (or "true") mean. Typically the standard error is equal to the sample's standard deviation divided by the square root of the number of points in the sample.

Student's t-test for Significantly Different Means

Applying the concept of standard error, the conventional statistic for measuring the significance of a difference of means is termed *Student's* t. When the two distributions are thought to have the same variance, but possibly different means, then Student's t is computed as follows: First, estimate the standard error of the difference of the means, s_D, from the "pooled variance" by the formula

$$s_D = \sqrt{\frac{\sum_{i\in A}(x_i - \overline{x_A})^2 + \sum_{i\in B}(x_i - \overline{x_B})^2}{N_A + N_B - 2}\left(\frac{1}{N_A} + \frac{1}{N_B}\right)} \tag{14.2.1}$$

where each sum is over the points in one sample, the first or second, each mean likewise refers to one sample or the other, and N_A and N_B are the numbers of points in the first and second samples, respectively. Second, compute t by

$$t = \frac{\overline{x_A} - \overline{x_B}}{s_D} \tag{14.2.2}$$

Third, evaluate the significance of this value of t for Student's distribution with $N_A + N_B - 2$ degrees of freedom, by equations (6.4.7) and (6.4.9), and by the routine `betai` (incomplete beta function) of §6.4.

The significance is a number between zero and one, and is the probability that $|t|$ could be this large or larger just by chance, for distributions with equal means. Therefore, a small numerical value of the significance (0.05 or 0.01) means that the observed difference is "very significant." The function $A(t|\nu)$ in equation (6.4.7) is one minus the significance.

As a routine, we have

```
#include <math.h>

void ttest(float data1[], unsigned long n1, float data2[], unsigned long n2,
    float *t, float *prob)
Given the arrays data1[1..n1] and data2[1..n2], this routine returns Student's t as t,
and its significance as prob, small values of prob indicating that the arrays have significantly
different means. The data arrays are assumed to be drawn from populations with the same
true variance.
{
    void avevar(float data[], unsigned long n, float *ave, float *var);
    float betai(float a, float b, float x);
    float var1,var2,svar,df,ave1,ave2;

    avevar(data1,n1,&ave1,&var1);
    avevar(data2,n2,&ave2,&var2);
    df=n1+n2-2;                                    Degrees of freedom.
    svar=((n1-1)*var1+(n2-1)*var2)/df;             Pooled variance.
    *t=(ave1-ave2)/sqrt(svar*(1.0/n1+1.0/n2));
    *prob=betai(0.5*df,0.5,df/(df+(*t)*(*t)));     See equation (6.4.9).
}
```

which makes use of the following routine for computing the mean and variance of a set of numbers,

```
void avevar(float data[], unsigned long n, float *ave, float *var)
Given array data[1..n], returns its mean as ave and its variance as var.
{
    unsigned long j;
    float s,ep;

    for (*ave=0.0,j=1;j<=n;j++) *ave += data[j];
    *ave /= n;
    *var=ep=0.0;
    for (j=1;j<=n;j++) {
        s=data[j]-(*ave);
        ep += s;
        *var += s*s;
    }
    *var=(*var-ep*ep/n)/(n-1);          Corrected two-pass formula (14.1.8).
}
```

The next case to consider is where the two distributions have significantly different variances, but we nevertheless want to know if their means are the same or different. (A treatment for baldness has caused some patients to *lose* all their hair and turned others into werewolves, but we want to know if it helps cure baldness *on the average*!) Be suspicious of the unequal-variance t-test: If two distributions have very different variances, then they may also be substantially different in shape; in that case, the difference of the means may not be a particularly useful thing to know.

To find out whether the two data sets have variances that are significantly different, you use the *F-test*, described later on in this section.

The relevant statistic for the unequal variance t-test is

$$t = \frac{\overline{x_A} - \overline{x_B}}{[\mathrm{Var}(x_A)/N_A + \mathrm{Var}(x_B)/N_B]^{1/2}} \tag{14.2.3}$$

This statistic is distributed *approximately* as Student's t with a number of degrees of freedom equal to

$$\frac{\left[\dfrac{\mathrm{Var}(x_A)}{N_A} + \dfrac{\mathrm{Var}(x_B)}{N_B}\right]^2}{\dfrac{[\mathrm{Var}(x_A)/N_A]^2}{N_A - 1} + \dfrac{[\mathrm{Var}(x_B)/N_B]^2}{N_B - 1}} \tag{14.2.4}$$

Expression (14.2.4) is in general not an integer, but equation (6.4.7) doesn't care.

The routine is

```
#include <math.h>
#include "nrutil.h"

void tutest(float data1[], unsigned long n1, float data2[], unsigned long n2,
    float *t, float *prob)
Given the arrays data1[1..n1] and data2[1..n2], this routine returns Student's t as t, and
its significance as prob, small values of prob indicating that the arrays have significantly differ-
ent means. The data arrays are allowed to be drawn from populations with unequal variances.
{
    void avevar(float data[], unsigned long n, float *ave, float *var);
    float betai(float a, float b, float x);
    float var1,var2,df,ave1,ave2;
```

```
    avevar(data1,n1,&ave1,&var1);
    avevar(data2,n2,&ave2,&var2);
    *t=(ave1-ave2)/sqrt(var1/n1+var2/n2);
    df=SQR(var1/n1+var2/n2)/(SQR(var1/n1)/(n1-1)+SQR(var2/n2)/(n2-1));
    *prob=betai(0.5*df,0.5,df/(df+SQR(*t)));
}
```

Our final example of a Student's t test is the case of *paired samples*. Here we imagine that much of the variance in *both* samples is due to effects that are point-by-point identical in the two samples. For example, we might have two job candidates who have each been rated by the same ten members of a hiring committee. We want to know if the means of the ten scores differ significantly. We first try `ttest` above, and obtain a value of `prob` that is not especially significant (e.g., > 0.05). But perhaps the significance is being washed out by the tendency of some committee members always to give high scores, others always to give low scores, which increases the apparent variance and thus decreases the significance of any difference in the means. We thus try the paired-sample formulas,

$$\mathrm{Cov}(x_A, x_B) \equiv \frac{1}{N-1}\sum_{i=1}^{N}(x_{Ai} - \overline{x_A})(x_{Bi} - \overline{x_B}) \tag{14.2.5}$$

$$s_D = \left[\frac{\mathrm{Var}(x_A) + \mathrm{Var}(x_B) - 2\mathrm{Cov}(x_A, x_B)}{N}\right]^{1/2} \tag{14.2.6}$$

$$t = \frac{\overline{x_A} - \overline{x_B}}{s_D} \tag{14.2.7}$$

where N is the number in each sample (number of pairs). Notice that it is important that a particular value of i label the corresponding points in each sample, that is, the ones that are paired. The significance of the t statistic in (14.2.7) is evaluated for $N - 1$ degrees of freedom.

The routine is

```
#include <math.h>

void tptest(float data1[], float data2[], unsigned long n, float *t,
    float *prob)
```

Given the paired arrays `data1[1..n]` and `data2[1..n]`, this routine returns Student's t for paired data as `t`, and its significance as `prob`, small values of `prob` indicating a significant difference of means.

```
{
    void avevar(float data[], unsigned long n, float *ave, float *var);
    float betai(float a, float b, float x);
    unsigned long j;
    float var1,var2,ave1,ave2,sd,df,cov=0.0;

    avevar(data1,n,&ave1,&var1);
    avevar(data2,n,&ave2,&var2);
    for (j=1;j<=n;j++)
        cov += (data1[j]-ave1)*(data2[j]-ave2);
    cov /= df=n-1;
    sd=sqrt((var1+var2-2.0*cov)/n);
    *t=(ave1-ave2)/sd;
    *prob=betai(0.5*df,0.5,df/(df+(*t)*(*t)));
}
```

F-Test for Significantly Different Variances

The *F-test* tests the hypothesis that two samples have different variances by trying to reject the null hypothesis that their variances are actually consistent. The statistic F is the ratio of one variance to the other, so values either $\gg 1$ or $\ll 1$ will indicate very significant differences. The distribution of F in the null case is given in equation (6.4.11), which is evaluated using the routine `betai`. In the most common case, we are willing to disprove the null hypothesis (of equal variances) by either very large or very small values of F, so the correct significance is *two-tailed*, the sum of two incomplete beta functions. It turns out, by equation (6.4.3), that the two tails are always equal; we need compute only one, and double it. Occasionally, when the null hypothesis is strongly viable, the identity of the two tails can become confused, giving an indicated probability greater than one. Changing the probability to two minus itself correctly exchanges the tails. These considerations and equation (6.4.3) give the routine

```
void ftest(float data1[], unsigned long n1, float data2[], unsigned long n2,
    float *f, float *prob)
```
Given the arrays `data1[1..n1]` and `data2[1..n2]`, this routine returns the value of `f`, and its significance as `prob`. Small values of `prob` indicate that the two arrays have significantly different variances.
```
{
    void avevar(float data[], unsigned long n, float *ave, float *var);
    float betai(float a, float b, float x);
    float var1,var2,ave1,ave2,df1,df2;

    avevar(data1,n1,&ave1,&var1);
    avevar(data2,n2,&ave2,&var2);
    if (var1 > var2) {                Make F the ratio of the larger variance to the smaller
        *f=var1/var2;                     one.
        df1=n1-1;
        df2=n2-1;
    } else {
        *f=var2/var1;
        df1=n2-1;
        df2=n1-1;
    }
    *prob = 2.0*betai(0.5*df2,0.5*df1,df2/(df2+df1*(*f)));
    if (*prob > 1.0) *prob=2.0-*prob;
}
```

CITED REFERENCES AND FURTHER READING:

von Mises, R. 1964, *Mathematical Theory of Probability and Statistics* (New York: Academic Press), Chapter IX(B).

Norusis, M.J. 1982, *SPSS Introductory Guide: Basic Statistics and Operations*; and 1985, *SPSS-X Advanced Statistics Guide* (New York: McGraw-Hill).

14.3 Are Two Distributions Different?

Given two sets of data, we can generalize the questions asked in the previous section and ask the single question: Are the two sets drawn from the same distribution function, or from different distribution functions? Equivalently, in proper statistical language, "Can we disprove, to a certain required level of significance, the null hypothesis that two data sets are drawn from the same population distribution function?" Disproving the null hypothesis in effect proves that the data sets are from different distributions. Failing to disprove the null hypothesis, on the other hand, only shows that the data sets can be *consistent* with a single distribution function. One can never *prove* that two data sets come from a single distribution, since (e.g.) no practical amount of data can distinguish between two distributions which differ only by one part in 10^{10}.

Proving that two distributions are different, or showing that they are consistent, is a task that comes up all the time in many areas of research: Are the visible stars distributed uniformly in the sky? (That is, is the distribution of stars as a function of declination — position in the sky — the same as the distribution of sky area as a function of declination?) Are educational patterns the same in Brooklyn as in the Bronx? (That is, are the distributions of people as a function of last-grade-attended the same?) Do two brands of fluorescent lights have the same distribution of burn-out times? Is the incidence of chicken pox the same for first-born, second-born, third-born children, etc.?

These four examples illustrate the four combinations arising from two different dichotomies: (1) The data are either continuous or binned. (2) Either we wish to compare one data set to a known distribution, or we wish to compare two equally unknown data sets. The data sets on fluorescent lights and on stars are continuous, since we can be given lists of individual burnout times or of stellar positions. The data sets on chicken pox and educational level are binned, since we are given tables of numbers of events in discrete categories: first-born, second-born, etc.; or 6th Grade, 7th Grade, etc. Stars and chicken pox, on the other hand, share the property that the null hypothesis is a known distribution (distribution of area in the sky, or incidence of chicken pox in the general population). Fluorescent lights and educational level involve the comparison of two equally unknown data sets (the two brands, or Brooklyn and the Bronx).

One can always turn continuous data into binned data, by grouping the events into specified ranges of the continuous variable(s): declinations between 0 and 10 degrees, 10 and 20, 20 and 30, etc. Binning involves a loss of information, however. Also, there is often considerable arbitrariness as to how the bins should be chosen. Along with many other investigators, we prefer to avoid unnecessary binning of data.

The accepted test for differences between binned distributions is the *chi-square test*. For continuous data as a function of a single variable, the most generally accepted test is the *Kolmogorov-Smirnov test*. We consider each in turn.

Chi-Square Test

Suppose that N_i is the number of events observed in the ith bin, and that n_i is the number expected according to some known distribution. Note that the N_i's are

integers, while the n_i's may not be. Then the chi-square statistic is

$$\chi^2 = \sum_i \frac{(N_i - n_i)^2}{n_i} \tag{14.3.1}$$

where the sum is over all bins. A large value of χ^2 indicates that the null hypothesis (that the N_i's are drawn from the population represented by the n_i's) is rather unlikely.

Any term j in (14.3.1) with $0 = n_j = N_j$ should be omitted from the sum. A term with $n_j = 0,\ N_j \neq 0$ gives an infinite χ^2, as it should, since in this case the N_i's cannot possibly be drawn from the n_i's!

The *chi-square probability function* $Q(\chi^2|\nu)$ is an incomplete gamma function, and was already discussed in §6.2 (see equation 6.2.18). Strictly speaking $Q(\chi^2|\nu)$ is the probability that the sum of the squares of ν random *normal* variables of unit variance (and zero mean) will be greater than χ^2. The terms in the sum (14.3.1) are not individually normal. However, if either the number of bins is large ($\gg 1$), or the number of events in each bin is large ($\gg 1$), then the chi-square probability function is a good approximation to the distribution of (14.3.1) in the case of the null hypothesis. Its use to estimate the significance of the chi-square test is standard.

The appropriate value of ν, the number of degrees of freedom, bears some additional discussion. If the data are collected with the model n_i's fixed — that is, not later renormalized to fit the total observed number of events ΣN_i — then ν equals the number of bins N_B. (Note that this is *not* the total number of *events*!) Much more commonly, the n_i's are normalized after the fact so that their sum equals the sum of the N_i's. In this case the correct value for ν is $N_B - 1$, and the model is said to have one constraint (`knstrn=1` in the program below). If the model that gives the n_i's has additional free parameters that were adjusted after the fact to agree with the data, then each of these additional "fitted" parameters decreases ν (and increases `knstrn`) by one additional unit.

We have, then, the following program:

```
void chsone(float bins[], float ebins[], int nbins, int knstrn, float *df,
    float *chsq, float *prob)
```

Given the array `bins[1..nbins]` containing the observed numbers of events, and an array `ebins[1..nbins]` containing the expected numbers of events, and given the number of constraints `knstrn` (normally one), this routine returns (trivially) the number of degrees of freedom `df`, and (nontrivially) the chi-square `chsq` and the significance `prob`. A small value of `prob` indicates a significant difference between the distributions `bins` and `ebins`. Note that `bins` and `ebins` are both `float` arrays, although `bins` will normally contain integer values.

```
{
    float gammq(float a, float x);
    void nrerror(char error_text[]);
    int j;
    float temp;

    *df=nbins-knstrn;
    *chsq=0.0;
    for (j=1;j<=nbins;j++) {
        if (ebins[j] <= 0.0) nrerror("Bad expected number in chsone");
        temp=bins[j]-ebins[j];
        *chsq += temp*temp/ebins[j];
    }
    *prob=gammq(0.5*(*df),0.5*(*chsq));          Chi-square probability function. See §6.2.
}
```

Next we consider the case of comparing *two* binned data sets. Let R_i be the number of events in bin i for the first data set, S_i the number of events in the same bin i for the second data set. Then the chi-square statistic is

$$\chi^2 = \sum_i \frac{(R_i - S_i)^2}{R_i + S_i} \tag{14.3.2}$$

Comparing (14.3.2) to (14.3.1), you should note that the denominator of (14.3.2) is *not* just the average of R_i and S_i (which would be an estimator of n_i in 14.3.1). Rather, it is twice the average, the sum. The reason is that each term in a chi-square sum is supposed to approximate the square of a normally distributed quantity with unit variance. The variance of the difference of two normal quantities is the sum of their individual variances, not the average.

If the data were collected in such a way that the sum of the R_i's is necessarily equal to the sum of S_i's, then the number of degrees of freedom is equal to one less than the number of bins, $N_B - 1$ (that is, `knstrn` $= 1$), the usual case. If this requirement were absent, then the number of degrees of freedom would be N_B. Example: A birdwatcher wants to know whether the distribution of sighted birds as a function of species is the same this year as last. Each bin corresponds to one species. If the birdwatcher takes his data to be the first 1000 birds that he saw in each year, then the number of degrees of freedom is $N_B - 1$. If he takes his data to be all the birds he saw on a random sample of days, the same days in each year, then the number of degrees of freedom is N_B (`knstrn` $= 0$). In this latter case, note that he is also testing whether the birds were more numerous overall in one year or the other: That is the extra degree of freedom. Of course, any additional constraints on the data set lower the number of degrees of freedom (i.e., increase `knstrn` to *more positive* values) in accordance with their number.

The program is

```
void chstwo(float bins1[], float bins2[], int nbins, int knstrn, float *df,
    float *chsq, float *prob)
Given the arrays bins1[1..nbins] and bins2[1..nbins], containing two sets of binned
data, and given the number of constraints knstrn (normally 1 or 0), this routine returns the
number of degrees of freedom df, the chi-square chsq, and the significance prob. A small value
of prob indicates a significant difference between the distributions bins1 and bins2. Note that
bins1 and bins2 are both float arrays, although they will normally contain integer values.
{
    float gammq(float a, float x);
    int j;
    float temp;

    *df=nbins-knstrn;
    *chsq=0.0;
    for (j=1;j<=nbins;j++)
        if (bins1[j] == 0.0 && bins2[j] == 0.0)
            --(*df);                          No data means one less degree of free-
        else {                                    dom.
            temp=bins1[j]-bins2[j];
            *chsq += temp*temp/(bins1[j]+bins2[j]);
        }
    *prob=gammq(0.5*(*df),0.5*(*chsq));       Chi-square probability function. See §6.2.
}
```

Equation (14.3.2) and the routine `chstwo` both apply to the case where the total number of data points is the same in the two binned sets. For unequal numbers of data points, the formula analogous to (14.3.2) is

$$\chi^2 = \sum_i \frac{(\sqrt{S/R}R_i - \sqrt{R/S}S_i)^2}{R_i + S_i} \tag{14.3.3}$$

where

$$R \equiv \sum_i R_i \qquad S \equiv \sum_i S_i \tag{14.3.4}$$

are the respective numbers of data points. It is straightforward to make the corresponding change in `chstwo`.

Kolmogorov-Smirnov Test

The Kolmogorov-Smirnov (or *K–S*) test is applicable to unbinned distributions that are functions of a single independent variable, that is, to data sets where each data point can be associated with a single number (lifetime of each lightbulb when it burns out, or declination of each star). In such cases, the list of data points can be easily converted to an unbiased estimator $S_N(x)$ of the *cumulative* distribution function of the probability distribution from which it was drawn: If the N events are located at values $x_i,\ i = 1, \ldots, N$, then $S_N(x)$ is the function giving the fraction of data points to the left of a given value x. This function is obviously constant between consecutive (i.e., sorted into ascending order) x_i's, and jumps by the same constant $1/N$ at each x_i. (See Figure 14.3.1.)

Different distribution functions, or sets of data, give different cumulative distribution function estimates by the above procedure. However, all cumulative distribution functions agree at the smallest allowable value of x (where they are zero), and at the largest allowable value of x (where they are unity). (The smallest and largest values might of course be $\pm\infty$.) So it is the behavior between the largest and smallest values that distinguishes distributions.

One can think of any number of statistics to measure the overall difference between two cumulative distribution functions: the absolute value of the area between them, for example. Or their integrated mean square difference. The Kolmogorov-Smirnov D is a particularly simple measure: It is defined as the *maximum value* of the absolute difference between two cumulative distribution functions. Thus, for comparing one data set's $S_N(x)$ to a known cumulative distribution function $P(x)$, the K–S statistic is

$$D = \max_{-\infty<x<\infty} |S_N(x) - P(x)| \tag{14.3.5}$$

while for comparing two different cumulative distribution functions $S_{N_1}(x)$ and $S_{N_2}(x)$, the K–S statistic is

$$D = \max_{-\infty<x<\infty} |S_{N_1}(x) - S_{N_2}(x)| \tag{14.3.6}$$

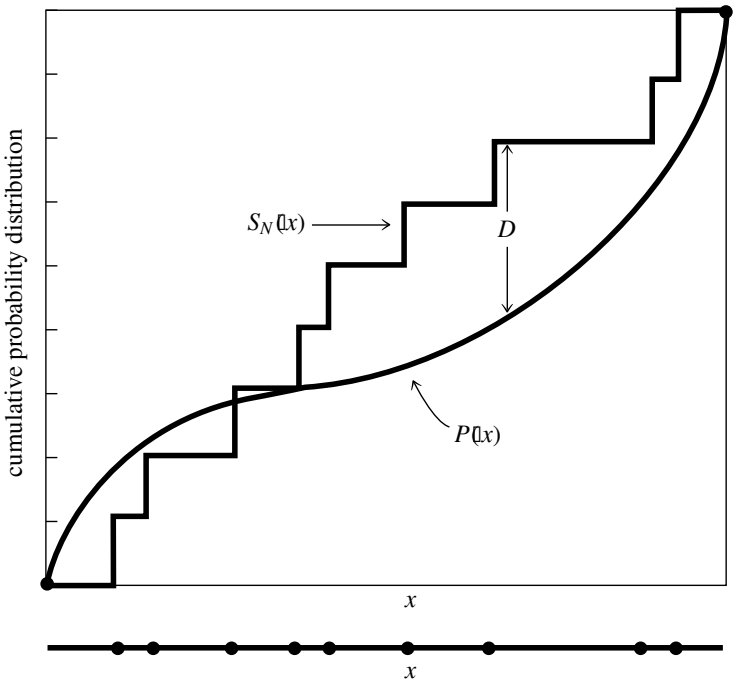

Figure 14.3.1. Kolmogorov-Smirnov statistic D. A measured distribution of values in x (shown as N dots on the lower abscissa) is to be compared with a theoretical distribution whose cumulative probability distribution is plotted as $P(x)$. A step-function cumulative probability distribution $S_N(x)$ is constructed, one that rises an equal amount at each measured point. D is the greatest distance between the two cumulative distributions.

What makes the K–S statistic useful is that *its* distribution in the case of the null hypothesis (data sets drawn from the same distribution) can be calculated, at least to useful approximation, thus giving the significance of any observed nonzero value of D. A central feature of the K–S test is that it is invariant under reparametrization of x; in other words, you can locally slide or stretch the x axis in Figure 14.3.1, and the maximum distance D remains unchanged. For example, you will get the same significance using x as using $\log x$.

The function that enters into the calculation of the significance can be written as the following sum:

$$Q_{KS}(\lambda) = 2\sum_{j=1}^{\infty}(-1)^{j-1}\, e^{-2j^2\lambda^2} \tag{14.3.7}$$

which is a monotonic function with the limiting values

$$Q_{KS}(0) = 1 \qquad Q_{KS}(\infty) = 0 \tag{14.3.8}$$

In terms of this function, the significance level of an observed value of D (as a disproof of the null hypothesis that the distributions are the same) is given approximately [1] by the formula

$$\text{Probability}\ (D > \text{observed}\ ) = Q_{KS}\left(\left[\sqrt{N_e} + 0.12 + 0.11/\sqrt{N_e}\right]\, D\right) \tag{14.3.9}$$

where N_e is the effective number of data points, $N_e = N$ for the case (14.3.5) of one distribution, and

$$N_e = \frac{N_1 N_2}{N_1 + N_2} \tag{14.3.10}$$

for the case (14.3.6) of two distributions, where N_1 is the number of data points in the first distribution, N_2 the number in the second.

The nature of the approximation involved in (14.3.9) is that it becomes asymptotically accurate as the N_e becomes large, but is already quite good for $N_e \geq 4$, as small a number as one might ever actually use. (See [1].)

So, we have the following routines for the cases of one and two distributions:

```
#include <math.h>
#include "nrutil.h"

void ksone(float data[], unsigned long n, float (*func)(float), float *d,
    float *prob)
```

Given an array `data[1..n]`, and given a user-supplied function of a single variable `func` which is a cumulative distribution function ranging from 0 (for smallest values of its argument) to 1 (for largest values of its argument), this routine returns the K–S statistic `d`, and the significance level `prob`. Small values of `prob` show that the cumulative distribution function of `data` is significantly different from `func`. The array `data` is modified by being sorted into ascending order.

```
{
    float probks(float alam);
    void sort(unsigned long n, float arr[]);
    unsigned long j;
    float dt,en,ff,fn,fo=0.0;

    sort(n,data);                                  If the data are already sorted into as-
    en=n;                                              cending order, then this call can be
    *d=0.0;                                            omitted.
    for (j=1;j<=n;j++) {                           Loop over the sorted data points.
        fn=j/en;                                   Data's c.d.f. after this step.
        ff=(*func)(data[j]);                       Compare to the user-supplied function.
        dt=FMAX(fabs(fo-ff),fabs(fn-ff));          Maximum distance.
        if (dt > *d) *d=dt;
        fo=fn;
    }
    en=sqrt(en);
    *prob=probks((en+0.12+0.11/en)*(*d));          Compute significance.
}
```

```
#include <math.h>

void kstwo(float data1[], unsigned long n1, float data2[], unsigned long n2,
    float *d, float *prob)
```

Given an array `data1[1..n1]`, and an array `data2[1..n2]`, this routine returns the K–S statistic `d`, and the significance level `prob` for the null hypothesis that the data sets are drawn from the same distribution. Small values of `prob` show that the cumulative distribution function of `data1` is significantly different from that of `data2`. The arrays `data1` and `data2` are modified by being sorted into ascending order.

```
{
    float probks(float alam);
    void sort(unsigned long n, float arr[]);
    unsigned long j1=1,j2=1;
    float d1,d2,dt,en1,en2,en,fn1=0.0,fn2=0.0;
```

```
    sort(n1,data1);
    sort(n2,data2);
    en1=n1;
    en2=n2;
    *d=0.0;
    while (j1 <= n1 && j2 <= n2) {                          If we are not done...
        if ((d1=data1[j1]) <= (d2=data2[j2])) fn1=j1++/en1;  Next step is in data1.
        if (d2 <= d1) fn2=j2++/en2;                         Next step is in data2.
        if ((dt=fabs(fn2-fn1)) > *d) *d=dt;
    }
    en=sqrt(en1*en2/(en1+en2));
    *prob=probks((en+0.12+0.11/en)*(*d));                   Compute significance.
}
```

Both of the above routines use the following routine for calculating the function Q_{KS}:

```
#include <math.h>
#define EPS1 0.001
#define EPS2 1.0e-8

float probks(float alam)
Kolmogorov-Smirnov probability function.
{
    int j;
    float a2,fac=2.0,sum=0.0,term,termbf=0.0;

    a2 = -2.0*alam*alam;
    for (j=1;j<=100;j++) {
        term=fac*exp(a2*j*j);
        sum += term;
        if (fabs(term) <= EPS1*termbf || fabs(term) <= EPS2*sum) return sum;
        fac = -fac;             Alternating signs in sum.
        termbf=fabs(term);
    }
    return 1.0;                 Get here only by failing to converge.
}
```

Variants on the K–S Test

The sensitivity of the K–S test to deviations from a cumulative distribution function $P(x)$ is not independent of x. In fact, the K–S test tends to be most sensitive around the median value, where $P(x) = 0.5$, and less sensitive at the extreme ends of the distribution, where $P(x)$ is near 0 or 1. The reason is that the difference $|S_N(x) - P(x)|$ does not, in the null hypothesis, have a probability distribution that is independent of x. Rather, its variance is proportional to $P(x)[1 - P(x)]$, which is largest at $P = 0.5$. Since the K–S statistic (14.3.5) is the maximum difference over all x of two cumulative distribution functions, a deviation that might be statistically significant at *its own* value of x gets compared to the expected chance deviation at $P = 0.5$, and is thus discounted. A result is that, while the K–S test is good at finding *shifts* in a probability distribution, especially changes in the median value, it is not always so good at finding *spreads*, which more affect the tails of the probability distribution, and which may leave the median unchanged.

One way of increasing the power of the K–S statistic out on the tails is to replace D (equation 14.3.5) by a so-called *stabilized* or *weighted* statistic [2-4], for example the *Anderson-Darling statistic*,

$$D^* = \max_{-\infty<x<\infty} \frac{|S_N(x) - P(x)|}{\sqrt{P(x)[1 - P(x)]}} \tag{14.3.11}$$

Unfortunately, there is no simple formula analogous to equations (14.3.7) and (14.3.9) for this statistic, although Noé [5] gives a computational method using a recursion relation and provides a graph of numerical results. There are many other possible similar statistics, for example

$$D^{**} = \int_{P=0}^{1} \frac{[S_N(x) - P(x)]^2}{P(x)[1 - P(x)]} dP(x) \tag{14.3.12}$$

which is also discussed by Anderson and Darling (see [3]).

Another approach, which we prefer as simpler and more direct, is due to Kuiper [6,7]. We already mentioned that the standard K–S test is invariant under reparametrizations of the variable x. An even more general symmetry, which guarantees equal sensitivities at all values of x, is to wrap the x axis around into a circle (identifying the points at $\pm\infty$), and to look for a statistic that is now invariant under all shifts and parametrizations on the circle. This allows, for example, a probability distribution to be "cut" at some central value of x, and the left and right halves to be interchanged, without altering the statistic or its significance.

Kuiper's statistic, defined as

$$V = D_+ + D_- = \max_{-\infty<x<\infty} [S_N(x) - P(x)] + \max_{-\infty<x<\infty} [P(x) - S_N(x)] \tag{14.3.13}$$

is the sum of the maximum distance of $S_N(x)$ *above and below* $P(x)$. You should be able to convince yourself that this statistic has the desired invariance on the circle: Sketch the indefinite integral of two probability distributions defined on the circle as a function of angle around the circle, as the angle goes through several times 360°. If you change the starting point of the integration, D_+ and D_- change individually, but their sum is constant.

Furthermore, there is a simple formula for the asymptotic distribution of the statistic V, directly analogous to equations (14.3.7)–(14.3.10). Let

$$Q_{KP}(\lambda) = 2\sum_{j=1}^{\infty} (4j^2\lambda^2 - 1)e^{-2j^2\lambda^2} \tag{14.3.14}$$

which is monotonic and satisfies

$$Q_{KP}(0) = 1 \qquad Q_{KP}(\infty) = 0 \tag{14.3.15}$$

In terms of this function the significance level is [1]

$$\text{Probability } (V > \text{observed }) = Q_{KP}\left(\left[\sqrt{N_e} + 0.155 + 0.24/\sqrt{N_e}\right] V\right) \tag{14.3.16}$$

Here N_e is N in the one-sample case, or is given by equation (14.3.10) in the case of two samples.

Of course, Kuiper's test is ideal for any problem originally defined on a circle, for example, to test whether the distribution in longitude of something agrees with some theory, or whether two somethings have different distributions in longitude. (See also [8].)

We will leave to you the coding of routines analogous to `ksone`, `kstwo`, and `probks`, above. (For $\lambda < 0.4$, don't try to do the sum 14.3.14. Its value is 1, to 7 figures, but the series can require many terms to converge, and loses accuracy to roundoff.)

Two final cautionary notes: First, we should mention that all varieties of K–S test lack the ability to discriminate some kinds of distributions. A simple example is a probability distribution with a narrow "notch" within which the probability falls to zero. Such a distribution is of course ruled out by the existence of even one data point within the notch, but, because of its cumulative nature, a K–S test would require many data points in the notch before signaling a discrepancy.

Second, we should note that, if you estimate any parameters from a data set (e.g., a mean and variance), then the distribution of the K–S statistic D for a cumulative distribution function $P(x)$ that *uses the estimated parameters* is no longer given by equation (14.3.9). In general, you will have to determine the new distribution yourself, e.g., by Monte Carlo methods.

CITED REFERENCES AND FURTHER READING:

von Mises, R. 1964, *Mathematical Theory of Probability and Statistics* (New York: Academic Press), Chapters IX(C) and IX(E).

Stephens, M.A. 1970, *Journal of the Royal Statistical Society*, ser. B, vol. 32, pp. 115–122. [1]

Anderson, T.W., and Darling, D.A. 1952, *Annals of Mathematical Statistics*, vol. 23, pp. 193–212. [2]

Darling, D.A. 1957, *Annals of Mathematical Statistics*, vol. 28, pp. 823–838. [3]

Michael, J.R. 1983, *Biometrika*, vol. 70, no. 1, pp. 11–17. [4]

Noé, M. 1972, *Annals of Mathematical Statistics*, vol. 43, pp. 58–64. [5]

Kuiper, N.H. 1962, *Proceedings of the Koninklijke Nederlandse Akademie van Wetenschappen*, ser. A., vol. 63, pp. 38–47. [6]

Stephens, M.A. 1965, *Biometrika*, vol. 52, pp. 309–321. [7]

Fisher, N.I., Lewis, T., and Embleton, B.J.J. 1987, *Statistical Analysis of Spherical Data* (New York: Cambridge University Press). [8]

14.4 Contingency Table Analysis of Two Distributions

In this section, and the next two sections, we deal with *measures of association* for two distributions. The situation is this: Each data point has two or more different quantities associated with it, and we want to know whether knowledge of one quantity gives us any demonstrable advantage in predicting the value of another quantity. In many cases, one variable will be an "independent" or "control" variable, and another will be a "dependent" or "measured" variable. Then, we want to know if the latter variable *is* in fact dependent on or *associated* with the former variable. If it is, we want to have some quantitative measure of the strength of the association. One often hears this loosely stated as the question of whether two variables are *correlated* or *uncorrelated*, but we will reserve those terms for a particular kind of association (linear, or at least monotonic), as discussed in §14.5 and §14.6.

Notice that, as in previous sections, the different concepts of significance and strength appear: The association between two distributions may be very significant even if that association is weak — if the quantity of data is large enough.

It is useful to distinguish among some different kinds of variables, with different categories forming a loose hierarchy.

- A variable is called *nominal* if its values are the members of some unordered set. For example, "state of residence" is a nominal variable that (in the U.S.) takes on one of 50 values; in astrophysics, "type of galaxy" is a nominal variable with the three values "spiral," "elliptical," and "irregular."
- A variable is termed *ordinal* if its values are the members of a discrete, but ordered, set. Examples are: grade in school, planetary order from the Sun (Mercury = 1, Venus = 2, ...), number of offspring. There need not be any concept of "equal metric distance" between the values of an ordinal variable, only that they be intrinsically ordered.
- We will call a variable *continuous* if its values are real numbers, as are times, distances, temperatures, etc. (Social scientists sometimes distinguish between *interval* and *ratio* continuous variables, but we do not find that distinction very compelling.)

A continuous variable can always be made into an ordinal one by binning it into ranges. If we choose to ignore the ordering of the bins, then we can turn it into

	1. red	2. green	. . .	
1. male	# of red males N_{11}	# of green males N_{12}	. . .	# of males $N_{1\cdot}$
2. female	# of red females N_{21}	# of green females N_{22}	. . .	# of females $N_{2\cdot}$
. . .	. . .	. . .	. . .	. . .
	# of red $N_{\cdot 1}$	# of green $N_{\cdot 2}$	. . .	total # N

Figure 14.4.1. Example of a contingency table for two nominal variables, here sex and color. The row and column marginals (totals) are shown. The variables are "nominal," i.e., the order in which their values are listed is arbitrary and does not affect the result of the contingency table analysis. If the ordering of values has some intrinsic meaning, then the variables are "ordinal" or "continuous," and correlation techniques (§14.5-§14.6) can be utilized.

a nominal variable. Nominal variables constitute the lowest type of the hierarchy, and therefore the most general. For example, a set of *several* continuous or ordinal variables can be turned, if crudely, into a single nominal variable, by coarsely binning each variable and then taking each distinct combination of bin assignments as a single nominal value. When multidimensional data are sparse, this is often the only sensible way to proceed.

The remainder of this section will deal with measures of association between *nominal* variables. For any pair of nominal variables, the data can be displayed as a *contingency table*, a table whose rows are labeled by the values of one nominal variable, whose columns are labeled by the values of the other nominal variable, and whose entries are nonnegative integers giving the number of observed events for each combination of row and column (see Figure 14.4.1). The analysis of association between nominal variables is thus called *contingency table analysis* or *crosstabulation analysis.*

We will introduce two different approaches. The first approach, based on the chi-square statistic, does a good job of characterizing the significance of association, but is only so-so as a measure of the strength (principally because its numerical values have no very direct interpretations). The second approach, based on the information-theoretic concept of *entropy*, says nothing at all about the significance of association (use chi-square for that!), but is capable of very elegantly characterizing the strength of an association already known to be significant.

Measures of Association Based on Chi-Square

Some notation first: Let N_{ij} denote the number of events that occur with the

first variable x taking on its ith value, and the second variable y taking on its jth value. Let N denote the total number of events, the sum of all the N_{ij}'s. Let $N_{i\cdot}$ denote the number of events for which the first variable x takes on its ith value regardless of the value of y; $N_{\cdot j}$ is the number of events with the jth value of y regardless of x. So we have

$$N_{i\cdot} = \sum_j N_{ij} \qquad N_{\cdot j} = \sum_i N_{ij}$$
$$N = \sum_i N_{i\cdot} = \sum_j N_{\cdot j} \tag{14.4.1}$$

$N_{\cdot j}$ and $N_{i\cdot}$ are sometimes called the *row and column totals* or *marginals*, but we will use these terms cautiously since we can never keep straight which are the rows and which are the columns!

The null hypothesis is that the two variables x and y have no association. In this case, the probability of a particular value of x given a particular value of y should be the same as the probability of that value of x regardless of y. Therefore, in the null hypothesis, the expected number for any N_{ij}, which we will denote n_{ij}, can be calculated from only the row and column totals,

$$\frac{n_{ij}}{N_{\cdot j}} = \frac{N_{i\cdot}}{N} \qquad \text{which implies} \qquad n_{ij} = \frac{N_{i\cdot} N_{\cdot j}}{N} \tag{14.4.2}$$

Notice that if a column or row total is zero, then the expected number for all the entries in that column or row is also zero; in that case, the never-occurring bin of x or y should simply be removed from the analysis.

The chi-square statistic is now given by equation (14.3.1), which, in the present case, is summed over all entries in the table,

$$\chi^2 = \sum_{i,j} \frac{(N_{ij} - n_{ij})^2}{n_{ij}} \tag{14.4.3}$$

The number of degrees of freedom is equal to the number of entries in the table (product of its row size and column size) minus the number of constraints that have arisen from our use of the data themselves to determine the n_{ij}. Each row total and column total is a constraint, except that this overcounts by one, since the total of the column totals and the total of the row totals both equal N, the total number of data points. Therefore, if the table is of size I by J, the number of degrees of freedom is $IJ - I - J + 1$. Equation (14.4.3), along with the chi-square probability function (§6.2), now give the significance of an association between the variables x and y.

Suppose there is a significant association. How do we quantify its strength, so that (e.g.) we can compare the strength of one association with another? The idea here is to find some reparametrization of χ^2 which maps it into some convenient interval, like 0 to 1, where the result is not dependent on the quantity of data that we happen to sample, but rather depends only on the underlying population from which the data were drawn. There are several different ways of doing this. Two of the more common are called *Cramer's V* and the *contingency coefficient C*.

The formula for Cramer's V is

$$V = \sqrt{\frac{\chi^2}{N \min(I-1, J-1)}} \tag{14.4.4}$$

where I and J are again the numbers of rows and columns, and N is the total number of events. Cramer's V has the pleasant property that it lies between zero and one inclusive, equals zero when there is no association, and equals one only when the association is perfect: All the events in any row lie in one unique column, and vice versa. (In chess parlance, no two rooks, placed on a nonzero table entry, can capture each other.)

In the case of $I = J = 2$, Cramer's V is also referred to as the *phi* statistic.

The contingency coefficient C is defined as

$$C = \sqrt{\frac{\chi^2}{\chi^2 + N}} \tag{14.4.5}$$

It also lies between zero and one, but (as is apparent from the formula) it can never achieve the upper limit. While it can be used to compare the strength of association of two tables with the same I and J, its upper limit depends on I and J. Therefore it can never be used to compare tables of different sizes.

The trouble with both Cramer's V and the contingency coefficient C is that, when they take on values in between their extremes, there is no very direct interpretation of what that value means. For example, you are in Las Vegas, and a friend tells you that there is a small, but significant, association between the color of a croupier's eyes and the occurrence of red and black on his roulette wheel. Cramer's V is about 0.028, your friend tells you. You know what the usual odds against you are (because of the green zero and double zero on the wheel). Is this association sufficient for you to make money? Don't ask us!

```
#include <math.h>
#include "nrutil.h"
#define TINY 1.0e-30                    A small number.

void cntab1(int **nn, int ni, int nj, float *chisq, float *df, float *prob,
    float *cramrv, float *ccc)
Given a two-dimensional contingency table in the form of an integer array nn[1..ni][1..nj],
this routine returns the chi-square chisq, the number of degrees of freedom df, the significance
level prob (small values indicating a significant association), and two measures of association,
Cramer's V (cramrv) and the contingency coefficient C (ccc).
{
    float gammq(float a, float x);
    int nnj,nni,j,i,minij;
    float sum=0.0,expctd,*sumi,*sumj,temp;

    sumi=vector(1,ni);
    sumj=vector(1,nj);
    nni=ni;                             Number of rows
    nnj=nj;                             and columns.
    for (i=1;i<=ni;i++) {               Get the row totals.
        sumi[i]=0.0;
        for (j=1;j<=nj;j++) {
            sumi[i] += nn[i][j];
            sum += nn[i][j];
```

```
        }
        if (sumi[i] == 0.0) --nni;          Eliminate any zero rows by reducing the num-
    }                                           ber.
    for (j=1;j<=nj;j++) {                   Get the column totals.
        sumj[j]=0.0;
        for (i=1;i<=ni;i++) sumj[j] += nn[i][j];
        if (sumj[j] == 0.0) --nnj;          Eliminate any zero columns.
    }
    *df=nni*nnj-nni-nnj+1;                  Corrected number of degrees of freedom.
    *chisq=0.0;
    for (i=1;i<=ni;i++) {                   Do the chi-square sum.
        for (j=1;j<=nj;j++) {
            expctd=sumj[j]*sumi[i]/sum;
            temp=nn[i][j]-expctd;
            *chisq += temp*temp/(expctd+TINY);      Here TINY guarantees that any
        }                                               eliminated row or column will
    }                                                   not contribute to the sum.
    *prob=gammq(0.5*(*df),0.5*(*chisq));    Chi-square probability function.
    minij = nni < nnj ? nni-1 : nnj-1;
    *cramrv=sqrt(*chisq/(sum*minij));
    *ccc=sqrt(*chisq/(*chisq+sum));
    free_vector(sumj,1,nj);
    free_vector(sumi,1,ni);
}
```

Measures of Association Based on Entropy

Consider the game of "twenty questions," where by repeated yes/no questions you try to eliminate all except one correct possibility for an unknown object. Better yet, consider a generalization of the game, where you are allowed to ask multiple choice questions as well as binary (yes/no) ones. The categories in your multiple choice questions are supposed to be mutually exclusive and exhaustive (as are "yes" and "no").

The value to you of an answer increases with the number of possibilities that it eliminates. More specifically, an answer that eliminates all except a fraction p of the remaining possibilities can be assigned a value $-\ln p$ (a positive number, since $p < 1$). The purpose of the logarithm is to make the value additive, since (e.g.) one question that eliminates all but 1/6 of the possibilities is considered as good as two questions that, in sequence, reduce the number by factors 1/2 and 1/3.

So that is the value of an answer; but what is the value of a question? If there are I possible answers to the question $(i = 1, \ldots, I)$ and the fraction of possibilities consistent with the ith answer is p_i (with the sum of the p_i's equal to one), then the value of the question is the expectation value of the value of the answer, denoted H,

$$H = -\sum_{i=1}^{I} p_i \ln p_i \tag{14.4.6}$$

In evaluating (14.4.6), note that

$$\lim_{p \to 0} p \ln p = 0 \tag{14.4.7}$$

The value H lies between 0 and $\ln I$. It is zero only when one of the p_i's is one, all the others zero: In this case, the question is valueless, since its answer is preordained.

H takes on its maximum value when all the p_i's are equal, in which case the question is sure to eliminate all but a fraction $1/I$ of the remaining possibilities.

The value H is conventionally termed the *entropy* of the distribution given by the p_i's, a terminology borrowed from statistical physics.

So far we have said nothing about the association of two variables; but suppose we are deciding what question to ask next in the game and have to choose between two candidates, or possibly want to ask both in one order or another. Suppose that one question, x, has I possible answers, labeled by i, and that the other question, y, as J possible answers, labeled by j. Then the possible outcomes of asking both questions form a contingency table whose entries N_{ij}, when normalized by dividing by the total number of remaining possibilities N, give all the information about the p's. In particular, we can make contact with the notation (14.4.1) by identifying

$$\begin{aligned} p_{ij} &= \frac{N_{ij}}{N} \\ p_{i\cdot} &= \frac{N_{i\cdot}}{N} \qquad \text{(outcomes of question } x \text{ alone)} \\ p_{\cdot j} &= \frac{N_{\cdot j}}{N} \qquad \text{(outcomes of question } y \text{ alone)} \end{aligned} \tag{14.4.8}$$

The entropies of the questions x and y are, respectively,

$$H(x) = -\sum_i p_{i\cdot} \ln p_{i\cdot} \qquad H(y) = -\sum_j p_{\cdot j} \ln p_{\cdot j} \tag{14.4.9}$$

The entropy of the two questions together is

$$H(x,y) = -\sum_{i,j} p_{ij} \ln p_{ij} \tag{14.4.10}$$

Now what is the entropy of the question y *given* x (that is, if x is asked first)? It is the expectation value over the answers to x of the entropy of the restricted y distribution that lies in a single column of the contingency table (corresponding to the x answer):

$$H(y|x) = -\sum_i p_{i\cdot} \sum_j \frac{p_{ij}}{p_{i\cdot}} \ln \frac{p_{ij}}{p_{i\cdot}} = -\sum_{i,j} p_{ij} \ln \frac{p_{ij}}{p_{i\cdot}} \tag{14.4.11}$$

Correspondingly, the entropy of x given y is

$$H(x|y) = -\sum_j p_{\cdot j} \sum_i \frac{p_{ij}}{p_{\cdot j}} \ln \frac{p_{ij}}{p_{\cdot j}} = -\sum_{i,j} p_{ij} \ln \frac{p_{ij}}{p_{\cdot j}} \tag{14.4.12}$$

We can readily prove that the entropy of y given x is never more than the entropy of y alone, i.e., that asking x first can only reduce the usefulness of asking

y (in which case the two variables are *associated*!):

$$\begin{aligned} H(y|x) - H(y) &= -\sum_{i,j} p_{ij} \ln \frac{p_{ij}/p_{i\cdot}}{p_{\cdot j}} \\ &= \sum_{i,j} p_{ij} \ln \frac{p_{\cdot j} p_{i\cdot}}{p_{ij}} \\ &\le \sum_{i,j} p_{ij} \left(\frac{p_{\cdot j} p_{i\cdot}}{p_{ij}} - 1 \right) \\ &= \sum_{i,j} p_{i\cdot} p_{\cdot j} - \sum_{i,j} p_{ij} \\ &= 1 - 1 = 0 \end{aligned} \tag{14.4.13}$$

where the inequality follows from the fact

$$\ln w \le w - 1 \tag{14.4.14}$$

We now have everything we need to define a measure of the "dependency" of y on x, that is to say a measure of association. This measure is sometimes called the *uncertainty coefficient* of y. We will denote it as $U(y|x)$,

$$U(y|x) \equiv \frac{H(y) - H(y|x)}{H(y)} \tag{14.4.15}$$

This measure lies between zero and one, with the value 0 indicating that x and y have no association, the value 1 indicating that knowledge of x completely predicts y. For in-between values, $U(y|x)$ gives the fraction of y's entropy $H(y)$ that is lost if x is already known (i.e., that is redundant with the information in x). In our game of "twenty questions," $U(y|x)$ is the fractional loss in the utility of question y if question x is to be asked first.

If we wish to view x as the dependent variable, y as the independent one, then interchanging x and y we can of course define the dependency of x on y,

$$U(x|y) \equiv \frac{H(x) - H(x|y)}{H(x)} \tag{14.4.16}$$

If we want to treat x and y symmetrically, then the useful combination turns out to be

$$U(x,y) \equiv 2 \left[\frac{H(y) + H(x) - H(x,y)}{H(x) + H(y)} \right] \tag{14.4.17}$$

If the two variables are completely independent, then $H(x,y) = H(x) + H(y)$, so (14.4.17) vanishes. If the two variables are completely dependent, then $H(x) = H(y) = H(x,y)$, so (14.4.16) equals unity. In fact, you can use the identities (easily proved from equations 14.4.9–14.4.12)

$$H(x,y) = H(x) + H(y|x) = H(y) + H(x|y) \tag{14.4.18}$$

to show that

$$U(x,y) = \frac{H(x)U(x|y) + H(y)U(y|x)}{H(x) + H(y)} \tag{14.4.19}$$

i.e., that the symmetrical measure is just a weighted average of the two asymmetrical measures (14.4.15) and (14.4.16), weighted by the entropy of each variable separately.

Here is a program for computing all the quantities discussed, $H(x)$, $H(y)$, $H(x|y)$, $H(y|x)$, $H(x,y)$, $U(x|y)$, $U(y|x)$, and $U(x,y)$:

```
#include <math.h>
#include "nrutil.h"
#define TINY 1.0e-30                          A small number.

void cntab2(int **nn, int ni, int nj, float *h, float *hx, float *hy,
    float *hygx, float *hxgy, float *uygx, float *uxgy, float *uxy)
```

Given a two-dimensional contingency table in the form of an integer array `nn[i][j]`, where `i` labels the x variable and ranges from 1 to `ni`, `j` labels the y variable and ranges from 1 to `nj`, this routine returns the entropy `h` of the whole table, the entropy `hx` of the x distribution, the entropy `hy` of the y distribution, the entropy `hygx` of y given x, the entropy `hxgy` of x given y, the dependency `uygx` of y on x (eq. 14.4.15), the dependency `uxgy` of x on y (eq. 14.4.16), and the symmetrical dependency `uxy` (eq. 14.4.17).

```
{
    int i,j;
    float sum=0.0,p,*sumi,*sumj;

    sumi=vector(1,ni);
    sumj=vector(1,nj);
    for (i=1;i<=ni;i++) {                     Get the row totals.
        sumi[i]=0.0;
        for (j=1;j<=nj;j++) {
            sumi[i] += nn[i][j];
            sum += nn[i][j];
        }
    }
    for (j=1;j<=nj;j++) {                     Get the column totals.
        sumj[j]=0.0;
        for (i=1;i<=ni;i++)
            sumj[j] += nn[i][j];
    }
    *hx=0.0;                                  Entropy of the x distribution,
    for (i=1;i<=ni;i++)
        if (sumi[i]) {
            p=sumi[i]/sum;
            *hx -= p*log(p);
        }
    *hy=0.0;                                  and of the y distribution.
    for (j=1;j<=nj;j++)
        if (sumj[j]) {
            p=sumj[j]/sum;
            *hy -= p*log(p);
        }
    *h=0.0;
    for (i=1;i<=ni;i++)                       Total entropy: loop over both x
        for (j=1;j<=nj;j++)                   and y.
            if (nn[i][j]) {
                p=nn[i][j]/sum;
                *h -= p*log(p);
            }
    *hygx=(*h)-(*hx);                         Uses equation (14.4.18),
    *hxgy=(*h)-(*hy);                         as does this.
    *uygx=(*hy-*hygx)/(*hy+TINY);             Equation (14.4.15).
    *uxgy=(*hx-*hxgy)/(*hx+TINY);             Equation (14.4.16).
    *uxy=2.0*(*hx+*hy-*h)/(*hx+*hy+TINY);     Equation (14.4.17).
    free_vector(sumj,1,nj);
    free_vector(sumi,1,ni);
}
```

CITED REFERENCES AND FURTHER READING:

Dunn, O.J., and Clark, V.A. 1974, *Applied Statistics: Analysis of Variance and Regression* (New York: Wiley).

Norusis, M.J. 1982, *SPSS Introductory Guide: Basic Statistics and Operations*; and 1985, *SPSS-X Advanced Statistics Guide* (New York: McGraw-Hill).

Fano, R.M. 1961, *Transmission of Information* (New York: Wiley and MIT Press), Chapter 2.

14.5 Linear Correlation

We next turn to measures of association between variables that are ordinal or continuous, rather than nominal. Most widely used is the *linear correlation coefficient*. For pairs of quantities (x_i, y_i), $i = 1, \ldots, N$, the linear correlation coefficient r (also called the product-moment correlation coefficient, or *Pearson's* r) is given by the formula

$$r = \frac{\sum_i (x_i - \overline{x})(y_i - \overline{y})}{\sqrt{\sum_i (x_i - \overline{x})^2} \sqrt{\sum_i (y_i - \overline{y})^2}} \tag{14.5.1}$$

where, as usual, $\overline{x}$ is the mean of the x_i's, $\overline{y}$ is the mean of the y_i's.

The value of r lies between -1 and 1, inclusive. It takes on a value of 1, termed "complete positive correlation," when the data points lie on a perfect straight line with positive slope, with x and y increasing together. The value 1 holds independent of the magnitude of the slope. If the data points lie on a perfect straight line with negative slope, y decreasing as x increases, then r has the value -1; this is called "complete negative correlation." A value of r near zero indicates that the variables x and y are *uncorrelated*.

When a correlation is known to be significant, r is one conventional way of summarizing its strength. In fact, the value of r can be translated into a statement about what residuals (root mean square deviations) are to be expected if the data are fitted to a straight line by the least-squares method (see §15.2, especially equations 15.2.13 – 15.2.14). Unfortunately, r is a rather poor statistic for deciding *whether* an observed correlation is statistically significant, and/or whether one observed correlation is significantly stronger than another. The reason is that r is ignorant of the individual distributions of x and y, so there is no universal way to compute its distribution in the case of the null hypothesis.

About the only general statement that can be made is this: If the null hypothesis is that x and y are uncorrelated, and if the distributions for x and y each have enough convergent moments ("tails" die off sufficiently rapidly), and if N is large (typically > 500), then r is distributed approximately normally, with a mean of zero and a standard deviation of $1/\sqrt{N}$. In that case, the (double-sided) significance of the correlation, that is, the probability that $|r|$ should be larger than its observed value in the null hypothesis, is

$$\text{erfc}\left(\frac{|r|\sqrt{N}}{\sqrt{2}}\right) \tag{14.5.2}$$

where $\text{erfc}(x)$ is the complementary error function, equation (6.2.8), computed by the routines `erffc` or `erfcc` of §6.2. A small value of (14.5.2) indicates that the

two distributions are significantly correlated. (See expression 14.5.9 below for a more accurate test.)

Most statistics books try to go beyond (14.5.2) and give additional statistical tests that can be made using r. In almost all cases, however, these tests are valid only for a very special class of hypotheses, namely that the distributions of x and y jointly form a *binormal* or *two-dimensional Gaussian* distribution around their mean values, with joint probability density

$$p(x,y)\,dxdy = \text{const.} \times \exp\left[-\frac{1}{2}(a_{11}x^2 - 2a_{12}xy + a_{22}y^2)\right]\,dxdy \tag{14.5.3}$$

where a_{11}, a_{12}, and a_{22} are arbitrary constants. For this distribution r has the value

$$r = -\frac{a_{12}}{\sqrt{a_{11}a_{22}}} \tag{14.5.4}$$

There are occasions when (14.5.3) may be known to be a good model of the data. There may be other occasions when we are willing to take (14.5.3) as at least a rough and ready guess, since many two-dimensional distributions do resemble a binormal distribution, at least not too far out on their tails. In either situation, we can use (14.5.3) to go beyond (14.5.2) in any of several directions:

First, we can allow for the possibility that the number N of data points is not large. Here, it turns out that the statistic

$$t = r\sqrt{\frac{N-2}{1-r^2}} \tag{14.5.5}$$

is distributed in the null case (of no correlation) like Student's t-distribution with $\nu = N - 2$ degrees of freedom, whose two-sided significance level is given by $1 - A(t|\nu)$ (equation 6.4.7). As N becomes large, this significance and (14.5.2) become asymptotically the same, so that one never does worse by using (14.5.5), even if the binormal assumption is not well substantiated.

Second, when N is only moderately large (≥ 10), we can compare whether the difference of two significantly nonzero r's, e.g., from different experiments, is itself significant. In other words, we can quantify whether a change in some control variable significantly alters an existing correlation between two other variables. This is done by using *Fisher's z-transformation* to associate each measured r with a corresponding z,

$$z = \frac{1}{2}\ln\left(\frac{1+r}{1-r}\right) \tag{14.5.6}$$

Then, each z is approximately normally distributed with a mean value

$$\overline{z} = \frac{1}{2}\left[\ln\left(\frac{1+r_{\rm true}}{1-r_{\rm true}}\right) + \frac{r_{\rm true}}{N-1}\right] \tag{14.5.7}$$

where $r_{\rm true}$ is the actual or population value of the correlation coefficient, and with a standard deviation

$$\sigma(z) \approx \frac{1}{\sqrt{N-3}} \tag{14.5.8}$$

Equations (14.5.7) and (14.5.8), when they are valid, give several useful statistical tests. For example, the significance level at which a measured value of r differs from some hypothesized value $r_{\rm true}$ is given by

$$\text{erfc}\left(\frac{|z-\overline{z}|\sqrt{N-3}}{\sqrt{2}}\right) \tag{14.5.9}$$

where z and $\overline{z}$ are given by (14.5.6) and (14.5.7), with small values of (14.5.9) indicating a significant difference. (Setting $\overline{z}=0$ makes expression 14.5.9 a more accurate replacement for expression 14.5.2 above.) Similarly, the significance of a difference between two measured correlation coefficients r_1 and r_2 is

$$\text{erfc}\left(\frac{|z_1-z_2|}{\sqrt{2}\sqrt{\frac{1}{N_1-3}+\frac{1}{N_2-3}}}\right) \tag{14.5.10}$$

where z_1 and z_2 are obtained from r_1 and r_2 using (14.5.6), and where N_1 and N_2 are, respectively, the number of data points in the measurement of r_1 and r_2.

All of the significances above are two-sided. If you wish to disprove the null hypothesis in favor of a one-sided hypothesis, such as that $r_1 > r_2$ (where the sense of the inequality was decided *a priori*), then (i) if your measured r_1 and r_2 have the *wrong* sense, you have failed to demonstrate your one-sided hypothesis, but (ii) if they have the right ordering, you can multiply the significances given above by 0.5, which makes them more significant.

But keep in mind: These interpretations of the r statistic can be completely meaningless if the joint probability distribution of your variables x and y is too different from a binormal distribution.

```
#include <math.h>
#define TINY 1.0e-20                Will regularize the unusual case of complete correlation.

void pearsn(float x[], float y[], unsigned long n, float *r, float *prob,
    float *z)
Given two arrays x[1..n] and y[1..n], this routine computes their correlation coefficient
r (returned as r), the significance level at which the null hypothesis of zero correlation is
disproved (prob whose small value indicates a significant correlation), and Fisher's z (returned
as z), whose value can be used in further statistical tests as described above.
{
    float betai(float a, float b, float x);
    float erfcc(float x);
    unsigned long j;
    float yt,xt,t,df;
    float syy=0.0,sxy=0.0,sxx=0.0,ay=0.0,ax=0.0;

    for (j=1;j<=n;j++) {            Find the means.
        ax += x[j];
        ay += y[j];
    }
    ax /= n;
    ay /= n;
    for (j=1;j<=n;j++) {            Compute the correlation coefficient.
        xt=x[j]-ax;
        yt=y[j]-ay;
        sxx += xt*xt;
        syy += yt*yt;
```

```
            sxy += xt*yt;
        }
        *r=sxy/(sqrt(sxx*syy)+TINY);
        *z=0.5*log((1.0+(*r)+TINY)/(1.0-(*r)+TINY));      Fisher's z transformation.
        df=n-2;
        t=(*r)*sqrt(df/((1.0-(*r)+TINY)*(1.0+(*r)+TINY)));      Equation (14.5.5).
        *prob=betai(0.5*df,0.5,df/(df+t*t));                    Student's t probability.
/*      *prob=erfcc(fabs((*z)*sqrt(n-1.0))/1.4142136)     */
For large n, this easier computation of prob, using the short routine erfcc, would give approx-
imately the same value.
}
```

CITED REFERENCES AND FURTHER READING:

Dunn, O.J., and Clark, V.A. 1974, *Applied Statistics: Analysis of Variance and Regression* (New York: Wiley).

Hoel, P.G. 1971, *Introduction to Mathematical Statistics*, 4th ed. (New York: Wiley), Chapter 7.

von Mises, R. 1964, *Mathematical Theory of Probability and Statistics* (New York: Academic Press), Chapters IX(A) and IX(B).

Korn, G.A., and Korn, T.M. 1968, *Mathematical Handbook for Scientists and Engineers*, 2nd ed. (New York: McGraw-Hill), §19.7.

Norusis, M.J. 1982, *SPSS Introductory Guide: Basic Statistics and Operations*; and 1985, *SPSS-X Advanced Statistics Guide* (New York: McGraw-Hill).

14.6 Nonparametric or Rank Correlation

It is precisely the uncertainty in interpreting the significance of the linear correlation coefficient r that leads us to the important concepts of *nonparametric* or *rank correlation*. As before, we are given N pairs of measurements (x_i, y_i). Before, difficulties arose because we did not necessarily know the probability distribution function from which the x_i's or y_i's were drawn.

The key concept of nonparametric correlation is this: If we replace the value of each x_i by the value of its *rank* among all the other x_i's in the sample, that is, $1, 2, 3, \ldots, N$, then the resulting list of numbers will be drawn from a perfectly known distribution function, namely uniformly from the integers between 1 and N, inclusive. Better than uniformly, in fact, since if the x_i's are all distinct, then each integer will occur precisely once. If some of the x_i's have identical values, it is conventional to assign to all these "ties" the mean of the ranks that they would have had if their values had been slightly different. This *midrank* will sometimes be an integer, sometimes a half-integer. In all cases the sum of all assigned ranks will be the same as the sum of the integers from 1 to N, namely $\frac{1}{2}N(N+1)$.

Of course we do exactly the same procedure for the y_i's, replacing each value by its rank among the other y_i's in the sample.

Now we are free to invent statistics for detecting correlation between uniform sets of integers between 1 and N, keeping in mind the possibility of ties in the ranks. There is, of course, some loss of information in replacing the original numbers by ranks. We could construct some rather artificial examples where a correlation could be detected parametrically (e.g., in the linear correlation coefficient r), but could not

be detected nonparametrically. Such examples are very rare in real life, however, and the slight loss of information in ranking is a small price to pay for a very major advantage: When a correlation is demonstrated to be present nonparametrically, then it is really there! (That is, to a certainty level that depends on the significance chosen.) Nonparametric correlation is more robust than linear correlation, more resistant to unplanned defects in the data, in the same sort of sense that the median is more robust than the mean. For more on the concept of robustness, see §15.7.

As always in statistics, some particular choices of a statistic have already been invented for us and consecrated, if not beatified, by popular use. We will discuss two, the *Spearman rank-order correlation coefficient* (r_s), and *Kendall's tau* (τ).

Spearman Rank-Order Correlation Coefficient

Let R_i be the rank of x_i among the other x's, S_i be the rank of y_i among the other y's, ties being assigned the appropriate midrank as described above. Then the rank-order correlation coefficient is defined to be the linear correlation coefficient of the ranks, namely,

$$r_s = \frac{\sum_i (R_i - \overline{R})(S_i - \overline{S})}{\sqrt{\sum_i (R_i - \overline{R})^2} \sqrt{\sum_i (S_i - \overline{S})^2}} \tag{14.6.1}$$

The significance of a nonzero value of r_s is tested by computing

$$t = r_s \sqrt{\frac{N-2}{1-r_s^2}} \tag{14.6.2}$$

which is distributed approximately as Student's distribution with $N-2$ degrees of freedom. A key point is that this approximation does not depend on the original distribution of the x's and y's; it is always the same approximation, and always pretty good.

It turns out that r_s is closely related to another conventional measure of nonparametric correlation, the so-called *sum squared difference of ranks*, defined as

$$D = \sum_{i=1}^{N} (R_i - S_i)^2 \tag{14.6.3}$$

(This D is sometimes denoted D^{**}, where the asterisks are used to indicate that ties are treated by midranking.)

When there are no ties in the data, then the exact relation between D and r_s is

$$r_s = 1 - \frac{6D}{N^3 - N} \tag{14.6.4}$$

When there are ties, then the exact relation is slightly more complicated: Let f_k be the number of ties in the kth group of ties among the R_i's, and let g_m be the number of ties in the mth group of ties among the S_i's. Then it turns out that

$$r_s = \frac{1 - \frac{6}{N^3 - N}\left[D + \frac{1}{12}\sum_k (f_k^3 - f_k) + \frac{1}{12}\sum_m (g_m^3 - g_m)\right]}{\left[1 - \frac{\sum_k (f_k^3 - f_k)}{N^3 - N}\right]^{1/2} \left[1 - \frac{\sum_m (g_m^3 - g_m)}{N^3 - N}\right]^{1/2}} \tag{14.6.5}$$

holds exactly. Notice that if all the f_k's and all the g_m's are equal to one, meaning that there are no ties, then equation (14.6.5) reduces to equation (14.6.4).

In (14.6.2) we gave a t-statistic that tests the significance of a nonzero r_s. It is also possible to test the significance of D directly. The expectation value of D in the null hypothesis of uncorrelated data sets is

$$\overline{D} = \frac{1}{6}(N^3 - N) - \frac{1}{12}\sum_k (f_k^3 - f_k) - \frac{1}{12}\sum_m (g_m^3 - g_m) \tag{14.6.6}$$

its variance is

$$\begin{aligned} \mathrm{Var}(D) = & \frac{(N-1)N^2(N+1)^2}{36} \\ & \times \left[1 - \frac{\sum_k (f_k^3 - f_k)}{N^3 - N}\right]\left[1 - \frac{\sum_m (g_m^3 - g_m)}{N^3 - N}\right] \end{aligned} \tag{14.6.7}$$

and it is approximately normally distributed, so that the significance level is a complementary error function (cf. equation 14.5.2). Of course, (14.6.2) and (14.6.7) are not independent tests, but simply variants of the same test. In the program that follows, we calculate both the significance level obtained by using (14.6.2) and the significance level obtained by using (14.6.7); their discrepancy will give you an idea of how good the approximations are. You will also notice that we break off the task of assigning ranks (including tied midranks) into a separate function, `crank`.

```
#include <math.h>
#include "nrutil.h"

void spear(float data1[], float data2[], unsigned long n, float *d, float *zd,
    float *probd, float *rs, float *probrs)
```

Given two data arrays, `data1[1..n]` and `data2[1..n]`, this routine returns their sum-squared difference of ranks as D, the number of standard deviations by which D deviates from its null-hypothesis expected value as `zd`, the two-sided significance level of this deviation as `probd`, Spearman's rank correlation r_s as `rs`, and the two-sided significance level of its deviation from zero as `probrs`. The external routines `crank` (below) and `sort2` (§8.2) are used. A small value of either `probd` or `probrs` indicates a significant correlation (`rs` positive) or anticorrelation (`rs` negative).

```
{
    float betai(float a, float b, float x);
    void crank(unsigned long n, float w[], float *s);
    float erfcc(float x);
    void sort2(unsigned long n, float arr[], float brr[]);
    unsigned long j;
    float vard,t,sg,sf,fac,en3n,en,df,aved,*wksp1,*wksp2;

    wksp1=vector(1,n);
    wksp2=vector(1,n);
    for (j=1;j<=n;j++) {
        wksp1[j]=data1[j];
        wksp2[j]=data2[j];
    }
    sort2(n,wksp1,wksp2);          Sort each of the data arrays, and convert the entries to
    crank(n,wksp1,&sf);            ranks. The values sf and sg return the sums $\sum(f_k^3 - f_k)$
    sort2(n,wksp2,wksp1);          and $\sum(g_m^3 - g_m)$, respectively.
    crank(n,wksp2,&sg);
    *d=0.0;
    for (j=1;j<=n;j++)             Sum the squared difference of ranks.
        *d += SQR(wksp1[j]-wksp2[j]);
```

```
    en=n;
    en3n=en*en*en-en;
    aved=en3n/6.0-(sf+sg)/12.0;                          Expectation value of D,
    fac=(1.0-sf/en3n)*(1.0-sg/en3n);
    vard=((en-1.0)*en*en*SQR(en+1.0)/36.0)*fac;          and variance of D give
    *zd=(*d-aved)/sqrt(vard);                            number of standard devia-
    *probd=erfcc(fabs(*zd)/1.4142136);                   tions and significance.
    *rs=(1.0-(6.0/en3n)*(*d+(sf+sg)/12.0))/sqrt(fac);    Rank correlation coefficient,
    fac=(*rs+1.0)*(1.0-(*rs));
    if (fac > 0.0) {
        t=(*rs)*sqrt((en-2.0)/fac);                      and its t value,
        df=en-2.0;
        *probrs=betai(0.5*df,0.5,df/(df+t*t));           give its significance.
    } else
        *probrs=0.0;
    free_vector(wksp2,1,n);
    free_vector(wksp1,1,n);
}
```

```
void crank(unsigned long n, float w[], float *s)
Given a sorted array w[1..n], replaces the elements by their rank, including midranking of ties,
and returns as s the sum of f^3 - f, where f is the number of elements in each tie.
{
    unsigned long j=1,ji,jt;
    float t,rank;

    *s=0.0;
    while (j < n) {
        if (w[j+1] != w[j]) {                Not a tie.
            w[j]=j;
            ++j;
        } else {                             A tie:
            for (jt=j+1;jt<=n && w[jt]==w[j];jt++);      How far does it go?
            rank=0.5*(j+jt-1);               This is the mean rank of the tie,
            for (ji=j;ji<=(jt-1);ji++) w[ji]=rank;       so enter it into all the tied
            t=jt-j;                                          entries,
            *s += t*t*t-t;                   and update s.
            j=jt;
        }
    }
    if (j == n) w[n]=n;                      If the last element was not tied, this is its rank.
}
```

Kendall's Tau

Kendall's τ is even more nonparametric than Spearman's r_s or D. Instead of using the numerical difference of ranks, it uses only the relative ordering of ranks: higher in rank, lower in rank, or the same in rank. But in that case we don't even have to rank the data! Ranks will be higher, lower, or the same if and only if the values are larger, smaller, or equal, respectively. On balance, we prefer r_s as being the more straightforward nonparametric test, but both statistics are in general use. In fact, τ and r_s are very strongly correlated and, in most applications, are effectively the same test.

To define τ, we start with the N data points (x_i, y_i). Now consider all $\frac{1}{2}N(N-1)$ *pairs* of data points, where a data point cannot be paired with itself, and where the points in either order count as one pair. We call a pair *concordant*

if the relative ordering of the ranks of the two x's (or for that matter the two x's themselves) is the same as the relative ordering of the ranks of the two y's (or for that matter the two y's themselves). We call a pair *discordant* if the relative ordering of the ranks of the two x's is opposite from the relative ordering of the ranks of the two y's. If there is a tie in either the ranks of the two x's or the ranks of the two y's, then we don't call the pair either concordant or discordant. If the tie is in the x's, we will call the pair an "extra y pair." If the tie is in the y's, we will call the pair an "extra x pair." If the tie is in both the x's and the y's, we don't call the pair anything at all. Are you still with us?

Kendall's τ is now the following simple combination of these various counts:

$$\tau = \frac{\text{concordant} - \text{discordant}}{\sqrt{\text{concordant} + \text{discordant} + \text{extra-}y}\ \sqrt{\text{concordant} + \text{discordant} + \text{extra-}x}} \tag{14.6.8}$$

You can easily convince yourself that this must lie between 1 and -1, and that it takes on the extreme values only for complete rank agreement or complete rank reversal, respectively.

More important, Kendall has worked out, from the combinatorics, the approximate distribution of τ in the null hypothesis of no association between x and y. In this case τ is approximately normally distributed, with zero expectation value and a variance of

$$\text{Var}(\tau) = \frac{4N + 10}{9N(N-1)} \tag{14.6.9}$$

The following program proceeds according to the above description, and therefore loops over all pairs of data points. Beware: This is an $O(N^2)$ algorithm, unlike the algorithm for r_s, whose dominant sort operations are of order $N \log N$. If you are routinely computing Kendall's τ for data sets of more than a few thousand points, you may be in for some serious computing. If, however, you are willing to bin your data into a moderate number of bins, then read on.

```
#include <math.h>

void kendl1(float data1[], float data2[], unsigned long n, float *tau,
    float *z, float *prob)
```

Given data arrays `data1[1..n]` and `data2[1..n]`, this program returns Kendall's τ as `tau`, its number of standard deviations from zero as `z`, and its two-sided significance level as `prob`. Small values of `prob` indicate a significant correlation (`tau` positive) or anticorrelation (`tau` negative).

```
{
    float erfcc(float x);
    unsigned long n2=0,n1=0,k,j;
    long is=0;
    float svar,aa,a2,a1;

    for (j=1;j<n;j++) {                          Loop over first member of pair,
        for (k=(j+1);k<=n;k++) {                 and second member.
            a1=data1[j]-data1[k];
            a2=data2[j]-data2[k];
            aa=a1*a2;
            if (aa) {                            Neither array has a tie.
                ++n1;
```

```
                ++n2;
                aa > 0.0 ? ++is : --is;
            } else {                                    One or both arrays have ties.
                if (a1) ++n1;                           An "extra x" event.
                if (a2) ++n2;                           An "extra y" event.
            }
        }
    }
    *tau=is/(sqrt((double) n1)*sqrt((double) n2));      Equation (14.6.8).
    svar=(4.0*n+10.0)/(9.0*n*(n-1.0));                  Equation (14.6.9).
    *z=(*tau)/sqrt(svar);
    *prob=erfcc(fabs(*z)/1.4142136);                    Significance.
}
```

Sometimes it happens that there are only a few possible values each for x and y. In that case, the data can be recorded as a contingency table (see §14.4) that gives the number of data points for each contingency of x and y.

Spearman's rank-order correlation coefficient is not a very natural statistic under these circumstances, since it assigns to each x and y bin a not-very-meaningful midrank value and then totals up vast numbers of identical rank differences. Kendall's tau, on the other hand, with its simple counting, remains quite natural. Furthermore, its $O(N^2)$ algorithm is no longer a problem, since we can arrange for it to loop over pairs of contingency table entries (each containing many data points) instead of over pairs of data points. This is implemented in the program that follows.

Note that Kendall's tau can be applied only to contingency tables where both variables are *ordinal*, i.e., well-ordered, and that it looks specifically for monotonic correlations, not for arbitrary associations. These two properties make it less general than the methods of §14.4, which applied to *nominal*, i.e., unordered, variables and arbitrary associations.

Comparing `kendl1` above with `kendl2` below, you will see that we have "floated" a number of variables. This is because the number of events in a contingency table might be sufficiently large as to cause overflows in some of the integer arithmetic, while the number of individual data points in a list could not possibly be that large [for an $O(N^2)$ routine!].

```
#include <math.h>

void kendl2(float **tab, int i, int j, float *tau, float *z, float *prob)
```

Given a two-dimensional table `tab[1..i][1..j]`, such that `tab[k][l]` contains the number of events falling in bin `k` of one variable and bin `l` of another, this program returns Kendall's τ as `tau`, its number of standard deviations from zero as `z`, and its two-sided significance level as `prob`. Small values of `prob` indicate a significant correlation (`tau` positive) or anticorrelation (`tau` negative) between the two variables. Although `tab` is a `float` array, it will normally contain integral values.

```
{
    float erfcc(float x);
    long nn,mm,m2,m1,lj,li,l,kj,ki,k;
    float svar,s=0.0,points,pairs,en2=0.0,en1=0.0;

    nn=i*j;                                     Total number of entries in contingency table.
    points=tab[i][j];
    for (k=0;k<=nn-2;k++) {                     Loop over entries in table,
        ki=(k/j);                               decoding a row,
        kj=k-j*ki;                              and a column.
        points += tab[ki+1][kj+1];              Increment the total count of events.
        for (l=k+1;l<=nn-1;l++) {               Loop over other member of the pair,
```

```
            li=l/j;                     decoding its row
            lj=l-j*li;                  and column.
            mm=(m1=li-ki)*(m2=lj-kj);
            pairs=tab[ki+1][kj+1]*tab[li+1][lj+1];
            if (mm) {                   Not a tie.
                en1 += pairs;
                en2 += pairs;
                s += (mm > 0 ? pairs : -pairs);       Concordant, or discordant.
            } else {
                if (m1) en1 += pairs;
                if (m2) en2 += pairs;
            }
        }
    }
    *tau=s/sqrt(en1*en2);
    svar=(4.0*points+10.0)/(9.0*points*(points-1.0));
    *z=(*tau)/sqrt(svar);
    *prob=erfcc(fabs(*z)/1.4142136);
}
```

CITED REFERENCES AND FURTHER READING:

Lehmann, E.L. 1975, *Nonparametrics: Statistical Methods Based on Ranks* (San Francisco: Holden-Day).

Downie, N.M., and Heath, R.W. 1965, *Basic Statistical Methods*, 2nd ed. (New York: Harper & Row), pp. 206–209.

Norusis, M.J. 1982, *SPSS Introductory Guide: Basic Statistics and Operations*; and 1985, *SPSS-X Advanced Statistics Guide* (New York: McGraw-Hill).

14.7 Do Two-Dimensional Distributions Differ?

We here discuss a useful generalization of the K–S test (§14.3) to *two-dimensional* distributions. This generalization is due to Fasano and Franceschini [1], a variant on an earlier idea due to Peacock [2].

In a two-dimensional distribution, each data point is characterized by an (x, y) pair of values. An example near to our hearts is that each of the 19 neutrinos that were detected from Supernova 1987A is characterized by a time t_i and by an energy E_i (see [3]). We might wish to know whether these measured pairs (t_i, E_i), $i = 1 \ldots 19$ are consistent with a theoretical model that predicts neutrino flux as a function of both time and energy — that is, a two-dimensional probability distribution in the (x, y) [here, (t, E)] plane. That would be a one-sample test. Or, given two sets of neutrino detections, from two comparable detectors, we might want to know whether they are compatible with each other, a two-sample test.

In the spirit of the tried-and-true, one-dimensional K–S test, we want to range over the (x, y) plane in search of some kind of maximum *cumulative* difference between two two-dimensional distributions. Unfortunately, cumulative probability distribution is not well-defined in more than one dimension! Peacock's insight was that a good surrogate is the *integrated probability in each of four natural quadrants* around a given point (x_i, y_i), namely the total probabilities (or fraction of data) in $(x > x_i, y > y_i)$, $(x < x_i, y > y_i)$, $(x < x_i, y < y_i)$, $(x > x_i, y < y_i)$. The two-dimensional K–S statistic D is now taken to be the maximum difference (ranging both over data points and over quadrants) of the corresponding integrated probabilities. When comparing two data sets, the value of D may depend on which data set is ranged over. In that case, define an effective D as the average

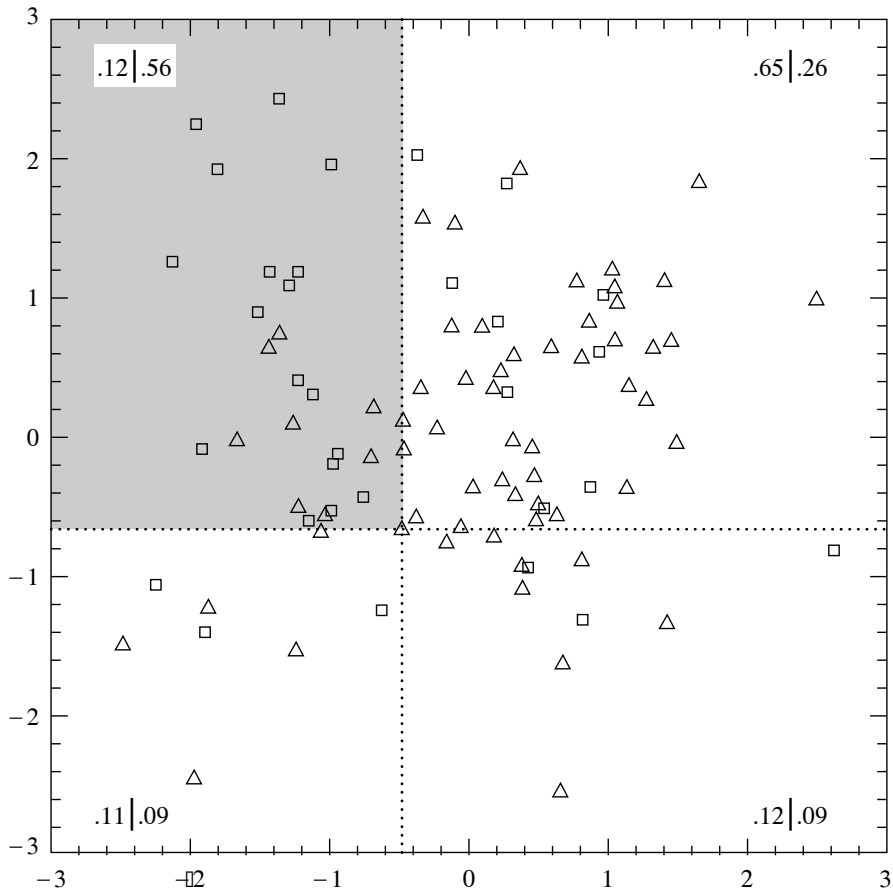

Figure 14.7.1. Two-dimensional distributions of 65 triangles and 35 squares. The two-dimensional K–S test finds that point one of whose quadrants (shown by dotted lines) maximizes the difference between fraction of triangles and fraction of squares. Then, equation (14.7.1) indicates whether the difference is statistically significant, i.e., whether the triangles and squares must have different underlying distributions.

of the two values obtained. If you are confused at this point about the exact definition of D, don't fret; the accompanying computer routines amount to a precise algorithmic definition.

Figure 14.7.1 gives a feeling for what is going on. The 65 triangles and 35 squares seem to have somewhat different distributions in the plane. The dotted lines are centered on the triangle that maximizes the D statistic; the maximum occurs in the upper-left quadrant. That quadrant contains only 0.12 of all the triangles, but it contains 0.56 of all the squares. The value of D is thus 0.44. Is this statistically significant?

Even for fixed sample sizes, it is unfortunately not rigorously true that the distribution of D in the null hypothesis is independent of the shape of the two-dimensional distribution. In this respect the two-dimensional K–S test is not as natural as its one-dimensional parent. However, extensive Monte Carlo integrations have shown that the distribution of the two-dimensional D is *very nearly* identical for even quite different distributions, as long as they have the same coefficient of correlation r, defined in the usual way by equation (14.5.1). In their paper, Fasano and Franceschini tabulate Monte Carlo results for (what amounts to) the distribution of D as a function of (of course) D, sample size N, and coefficient of correlation r. Analyzing their results, one finds that the significance levels for the two-dimensional K–S test can be summarized by the simple, though approximate, formulas,

$$\text{Probability}\ (D > \text{observed}\ ) = Q_{KS}\left(\frac{\sqrt{N}\,D}{1+\sqrt{1-r^2}(0.25-0.75/\sqrt{N})}\right) \qquad (14.7.1)$$

for the one-sample case, and the same for the two-sample case, but with

$$N = \frac{N_1 N_2}{N_1 + N_2}. \tag{14.7.2}$$

The above formulas are accurate enough when $N \gtrsim 20$, and when the indicated probability (significance level) is less than (more significant than) 0.20 or so. When the indicated probability is > 0.20, its value may not be accurate, but the implication that the data and model (or two data sets) are not significantly different is certainly correct. Notice that in the limit of $r \to 1$ (perfect correlation), equations (14.7.1) and (14.7.2) reduce to equations (14.3.9) and (14.3.10): The two-dimensional data lie on a perfect straight line, and the two-dimensional K–S test becomes a one-dimensional K–S test.

The significance level for the data in Figure 14.7.1, by the way, is about 0.001. This establishes to a near-certainty that the triangles and squares were drawn from different distributions. (As in fact they were.)

Of course, if you do not want to rely on the Monte Carlo experiments embodied in equation (14.7.1), you can do your own: Generate a lot of synthetic data sets from your model, each one with the same number of points as the real data set. Compute D for each synthetic data set, using the accompanying computer routines (but ignoring their calculated probabilities), and count what fraction of the time these synthetic D's exceed the D from the real data. That fraction is your significance.

One disadvantage of the two-dimensional tests, by comparison with their one-dimensional progenitors, is that the two-dimensional tests require of order N^2 operations: Two nested loops of order N take the place of an $N \log N$ sort. For small computers, this restricts the usefulness of the tests to N less than several thousand.

We now give computer implementations. The one-sample case is embodied in the routine `ks2d1s` (that is, 2-dimensions, 1-sample). This routine calls a straightforward utility routine `quadct` to count points in the four quadrants, and it calls a user-supplied routine `quadvl` that must be capable of returning the integrated probability of an analytic model in each of four quadrants around an arbitrary (x, y) point. A trivial sample `quadvl` is shown; realistic `quadvl`s can be quite complicated, often incorporating numerical quadratures over analytic two-dimensional distributions.

```
#include <math.h>
#include "nrutil.h"

void ks2d1s(float x1[], float y1[], unsigned long n1,
    void (*quadvl)(float, float, float *, float *, float *, float *),
    float *d1, float *prob)
```

Two-dimensional Kolmogorov-Smirnov test of one sample against a model. Given the x and y coordinates of `n1` data points in arrays `x1[1..n1]` and `y1[1..n1]`, and given a user-supplied function `quadvl` that exemplifies the model, this routine returns the two-dimensional K-S statistic as `d1`, and its significance level as `prob`. Small values of `prob` show that the sample is significantly different from the model. Note that the test is slightly distribution-dependent, so `prob` is only an estimate.

```
{
    void pearsn(float x[], float y[], unsigned long n, float *r, float *prob,
        float *z);
    float probks(float alam);
    void quadct(float x, float y, float xx[], float yy[], unsigned long nn,
        float *fa, float *fb, float *fc, float *fd);
    unsigned long j;
    float dum,dumm,fa,fb,fc,fd,ga,gb,gc,gd,r1,rr,sqen;

    *d1=0.0;
    for (j=1;j<=n1;j++) {                    Loop over the data points.
        quadct(x1[j],y1[j],x1,y1,n1,&fa,&fb,&fc,&fd);
        (*quadvl)(x1[j],y1[j],&ga,&gb,&gc,&gd);
        *d1=FMAX(*d1,fabs(fa-ga));
        *d1=FMAX(*d1,fabs(fb-gb));
        *d1=FMAX(*d1,fabs(fc-gc));
```

```
            *d1=FMAX(*d1,fabs(fd-gd));
            For both the sample and the model, the distribution is integrated in each of four
            quadrants, and the maximum difference is saved.
        }
        pearsn(x1,y1,n1,&r1,&dum,&dumm);        Get the linear correlation coefficient r1.
        sqen=sqrt((double)n1);
        rr=sqrt(1.0-r1*r1);
        Estimate the probability using the K-S probability function probks.
        *prob=probks(*d1*sqen/(1.0+rr*(0.25-0.75/sqen)));
}
```

```
void quadct(float x, float y, float xx[], float yy[], unsigned long nn,
    float *fa, float *fb, float *fc, float *fd)
```

Given an origin (x, y), and an array of nn points with coordinates xx[1..nn] and yy[1..nn], count how many of them are in each quadrant around the origin, and return the normalized fractions. Quadrants are labeled alphabetically, counterclockwise from the upper right. Used by ks2d1s and ks2d2s.

```
{
    unsigned long k,na,nb,nc,nd;
    float ff;
    na=nb=nc=nd=0;
    for (k=1;k<=nn;k++) {
        if (yy[k] > y) {
            xx[k] > x ? ++na : ++nb;
        } else {
            xx[k] > x ? ++nd : ++nc;
        }
    }
    ff=1.0/nn;
    *fa=ff*na;
    *fb=ff*nb;
    *fc=ff*nc;
    *fd=ff*nd;
}
```

```
#include "nrutil.h"

void quadvl(float x, float y, float *fa, float *fb, float *fc, float *fd)
```

This is a sample of a user-supplied routine to be used with ks2d1s. In this case, the model distribution is uniform inside the square $-1 < x < 1$, $-1 < y < 1$. In general this routine should return, for any point (x, y), the fraction of the total distribution in each of the four quadrants around that point. The fractions, fa, fb, fc, and fd, must add up to 1. Quadrants are alphabetical, counterclockwise from the upper right.

```
{
    float qa,qb,qc,qd;

    qa=FMIN(2.0,FMAX(0.0,1.0-x));
    qb=FMIN(2.0,FMAX(0.0,1.0-y));
    qc=FMIN(2.0,FMAX(0.0,x+1.0));
    qd=FMIN(2.0,FMAX(0.0,y+1.0));
    *fa=0.25*qa*qb;
    *fb=0.25*qb*qc;
    *fc=0.25*qc*qd;
    *fd=0.25*qd*qa;
}
```

The routine ks2d2s is the two-sample case of the two-dimensional K–S test. It also calls quadct, pearsn, and probks. Being a two-sample test, it does not need an analytic model.

```
#include <math.h>
#include "nrutil.h"

void ks2d2s(float x1[], float y1[], unsigned long n1, float x2[], float y2[],
    unsigned long n2, float *d, float *prob)
```

Two-dimensional Kolmogorov-Smirnov test on two samples. Given the x and y coordinates of the first sample as `n1` values in arrays `x1[1..n1]` and `y1[1..n1]`, and likewise for the second sample, `n2` values in arrays `x2` and `y2`, this routine returns the two-dimensional, two-sample K-S statistic as `d`, and its significance level as `prob`. Small values of `prob` show that the two samples are significantly different. Note that the test is slightly distribution-dependent, so `prob` is only an estimate.

```
{
    void pearsn(float x[], float y[], unsigned long n, float *r, float *prob,
        float *z);
    float probks(float alam);
    void quadct(float x, float y, float xx[], float yy[], unsigned long nn,
        float *fa, float *fb, float *fc, float *fd);
    unsigned long j;
    float d1,d2,dum,dumm,fa,fb,fc,fd,ga,gb,gc,gd,r1,r2,rr,sqen;

    d1=0.0;
    for (j=1;j<=n1;j++) {                    First, use points in the first sample as ori-
        quadct(x1[j],y1[j],x1,y1,n1,&fa,&fb,&fc,&fd);    gins.
        quadct(x1[j],y1[j],x2,y2,n2,&ga,&gb,&gc,&gd);
        d1=FMAX(d1,fabs(fa-ga));
        d1=FMAX(d1,fabs(fb-gb));
        d1=FMAX(d1,fabs(fc-gc));
        d1=FMAX(d1,fabs(fd-gd));
    }
    d2=0.0;
    for (j=1;j<=n2;j++) {                    Then, use points in the second sample as
        quadct(x2[j],y2[j],x1,y1,n1,&fa,&fb,&fc,&fd);    origins.
        quadct(x2[j],y2[j],x2,y2,n2,&ga,&gb,&gc,&gd);
        d2=FMAX(d2,fabs(fa-ga));
        d2=FMAX(d2,fabs(fb-gb));
        d2=FMAX(d2,fabs(fc-gc));
        d2=FMAX(d2,fabs(fd-gd));
    }
    *d=0.5*(d1+d2);                          Average the K-S statistics.
    sqen=sqrt(n1*n2/(double)(n1+n2));
    pearsn(x1,y1,n1,&r1,&dum,&dumm);         Get the linear correlation coefficient for each
    pearsn(x2,y2,n2,&r2,&dum,&dumm);             sample.
    rr=sqrt(1.0-0.5*(r1*r1+r2*r2));
    Estimate the probability using the K-S probability function probks.
    *prob=probks(*d*sqen/(1.0+rr*(0.25-0.75/sqen)));
}
```

CITED REFERENCES AND FURTHER READING:

Fasano, G. and Franceschini, A. 1987, *Monthly Notices of the Royal Astronomical Society*, vol. 225, pp. 155–170. [1]

Peacock, J.A. 1983, *Monthly Notices of the Royal Astronomical Society*, vol. 202, pp. 615–627. [2]

Spergel, D.N., Piran, T., Loeb, A., Goodman, J., and Bahcall, J.N. 1987, *Science*, vol. 237, pp. 1471–1473. [3]

14.8 Savitzky-Golay Smoothing Filters

In §13.5 we learned something about the construction and application of digital filters, but little guidance was given on *which particular* filter to use. That, of course, depends on what you want to accomplish by filtering. One obvious use for *low-pass* filters is to smooth noisy data.

The premise of data smoothing is that one is measuring a variable that is both slowly varying and also corrupted by random noise. Then it can sometimes be useful to replace each data point by some kind of local average of surrounding data points. Since nearby points measure very nearly the same underlying value, averaging can reduce the level of noise without (much) biasing the value obtained.

We must comment editorially that the smoothing of data lies in a murky area, beyond the fringe of some better posed, and therefore more highly recommended, techniques that are discussed elsewhere in this book. If you are fitting data to a parametric model, for example (see Chapter 15), it is almost always better to use raw data than to use data that has been pre-processed by a smoothing procedure. Another alternative to blind smoothing is so-called "optimal" or Wiener filtering, as discussed in §13.3 and more generally in §13.6. Data smoothing is probably most justified when it is used simply as a graphical technique, to guide the eye through a forest of data points all with large error bars; or as a means of making initial *rough* estimates of simple parameters from a graph.

In this section we discuss a particular type of low-pass filter, well-adapted for data smoothing, and termed variously *Savitzky-Golay* [1], *least-squares* [2], or *DISPO* (Digital Smoothing Polynomial) [3] filters. Rather than having their properties defined in the Fourier domain, and then translated to the time domain, Savitzky-Golay filters derive directly from a particular formulation of the data smoothing problem in the time domain, as we will now see. Savitzky-Golay filters were initially (and are still often) used to render visible the relative widths and heights of spectral lines in noisy spectrometric data.

Recall that a digital filter is applied to a series of equally spaced data values $f_i \equiv f(t_i)$, where $t_i \equiv t_0 + i\Delta$ for some constant sample spacing Δ and $i = \ldots -2, -1, 0, 1, 2, \ldots$. We have seen (§13.5) that the simplest type of digital filter (the nonrecursive or finite impulse response filter) replaces each data value f_i by a linear combination g_i of itself and some number of nearby neighbors,

$$g_i = \sum_{n=-n_L}^{n_R} c_n f_{i+n} \tag{14.8.1}$$

Here n_L is the number of points used "to the left" of a data point i, i.e., earlier than it, while n_R is the number used to the right, i.e., later. A so-called *causal* filter would have $n_R = 0$.

As a starting point for understanding Savitzky-Golay filters, consider the simplest possible averaging procedure: For some fixed $n_L = n_R$, compute each g_i as the average of the data points from f_{i-n_L} to f_{i+n_R}. This is sometimes called *moving window averaging* and corresponds to equation (14.8.1) with constant $c_n = 1/(n_L + n_R + 1)$. If the underlying function is constant, or is changing linearly with time (increasing or decreasing), then no bias is introduced into the result. Higher points at one end of the averaging interval are on the average balanced by lower points at the other end. A bias is introduced, however, if the underlying function has a nonzero second derivative. At a local maximum, for example, moving window averaging always reduces the function value. In the spectrometric application, a narrow spectral line has its height reduced and its width increased. Since these parameters are themselves of physical interest, the bias introduced is distinctly undesirable.

Note, however, that moving window averaging does preserve the area under a spectral line, which is its zeroth moment, and also (if the window is symmetric with $n_L = n_R$) its mean position in time, which is its first moment. What is violated is the second moment, equivalent to the line width.

The idea of Savitzky-Golay filtering is to find filter coefficients c_n that preserve higher moments. Equivalently, the idea is to approximate the underlying function within the moving window not by a constant (whose estimate is the average), but by a polynomial of higher order, typically quadratic or quartic: For each point f_i, we least-squares fit a polynomial to all

M	n_L	n_R	Sample Savitzky-Golay Coefficients											
2	2	2				−0.086	0.343	0.486	0.343	−0.086				
2	3	1			−0.143	0.171	0.343	0.371	0.257					
2	4	0		0.086	−0.143	−0.086	0.257	0.886						
2	5	5	−0.084	0.021	0.103	0.161	0.196	0.207	0.196	0.161	0.103	0.021	−0.084	
4	4	4		0.035	−0.128	0.070	0.315	0.417	0.315	0.070	−0.128	0.035		
4	5	5	0.042	−0.105	−0.023	0.140	0.280	0.333	0.280	0.140	−0.023	−0.105	0.042	

$n_L + n_R + 1$ points in the moving window, and then set g_i to be the value of that polynomial at position i. (If you are not familiar with least-squares fitting, you might want to look ahead to Chapter 15.) We make no use of the value of the polynomial at any other point. When we move on to the next point f_{i+1}, we do a whole new least-squares fit using a shifted window.

All these least-squares fits would be laborious if done as described. Luckily, since the process of least-squares fitting involves only a linear matrix inversion, the coefficients of a fitted polynomial are themselves linear in the values of the data. That means that we can do all the fitting in advance, for fictitious data consisting of all zeros except for a single 1, and then do the fits on the real data just by taking linear combinations. This is the key point, then: There are particular sets of filter coefficients c_n for which equation (14.8.1) "automatically" accomplishes the process of polynomial least-squares fitting inside a moving window.

To derive such coefficients, consider how g_0 might be obtained: We want to fit a polynomial of degree M in i, namely $a_0 + a_1 i + \cdots + a_M i^M$ to the values $f_{-n_L}, \ldots, f_{n_R}$. Then g_0 will be the value of that polynomial at $i = 0$, namely a_0. The design matrix for this problem (§15.4) is

$$A_{ij} = i^j \qquad i = -n_L, \ldots, n_R, \quad j = 0, \ldots, M \tag{14.8.2}$$

and the normal equations for the vector of a_j's in terms of the vector of f_i's is in matrix notation

$$(\mathbf{A}^T \cdot \mathbf{A}) \cdot \mathbf{a} = \mathbf{A}^T \cdot \mathbf{f} \qquad \text{or} \qquad \mathbf{a} = (\mathbf{A}^T \cdot \mathbf{A})^{-1} \cdot (\mathbf{A}^T \cdot \mathbf{f}) \tag{14.8.3}$$

We also have the specific forms

$$\left\{\mathbf{A}^T \cdot \mathbf{A}\right\}_{ij} = \sum_{k=-n_L}^{n_R} A_{ki} A_{kj} = \sum_{k=-n_L}^{n_R} k^{i+j} \tag{14.8.4}$$

and

$$\left\{\mathbf{A}^T \cdot \mathbf{f}\right\}_{j} = \sum_{k=-n_L}^{n_R} A_{kj} f_k = \sum_{k=-n_L}^{n_R} k^j f_k \tag{14.8.5}$$

Since the coefficient c_n is the component a_0 when $\mathbf{f}$ is replaced by the unit vector $\mathbf{e}_n$, $-n_L \le n < n_R$, we have

$$c_n = \left\{(\mathbf{A}^T \cdot \mathbf{A})^{-1} \cdot (\mathbf{A}^T \cdot \mathbf{e}_n)\right\}_0 = \sum_{m=0}^{M} \left\{(\mathbf{A}^T \cdot \mathbf{A})^{-1}\right\}_{0m} n^m \tag{14.8.6}$$

Note that equation (14.8.6) says that we need only one row of the inverse matrix. (Numerically we can get this by LU decomposition with only a single backsubstitution.)

The function `savgol`, below, implements equation (14.8.6). As input, it takes the parameters `nl` $= n_L$, `nr` $= n_R$, and `m` $= M$ (the desired order). Also input is `np`, the physical length of the output array `c`, and a parameter `ld` which for data fitting should be zero. In fact, `ld` specifies which coefficient among the a_i's should be returned, and we are here interested in a_0. For another purpose, namely the computation of numerical derivatives (already mentioned in §5.7) the useful choice is `ld` ≥ 1. With `ld` $= 1$, for example, the filtered first derivative is the convolution (14.8.1) divided by the stepsize Δ. For `ld` $= k > 1$, the array `c` must be multiplied by $k!$ to give derivative coefficients. For derivatives, one usually wants `m` $= 4$ or larger.

```
#include <math.h>
#include "nrutil.h"

void savgol(float c[], int np, int nl, int nr, int ld, int m)
```

Returns in `c[1..np]`, in wrap-around order (N.B.!) consistent with the argument `respns` in routine `convlv`, a set of Savitzky-Golay filter coefficients. `nl` is the number of leftward (past) data points used, while `nr` is the number of rightward (future) data points, making the total number of data points used $\texttt{nl}+\texttt{nr}+1$. `ld` is the order of the derivative desired (e.g., $\texttt{ld}=0$ for smoothed function). `m` is the order of the smoothing polynomial, also equal to the highest conserved moment; usual values are $\texttt{m}=2$ or $\texttt{m}=4$.

```
{
    void lubksb(float **a, int n, int *indx, float b[]);
    void ludcmp(float **a, int n, int *indx, float *d);
    int imj,ipj,j,k,kk,mm,*indx;
    float d,fac,sum,**a,*b;

    if (np < nl+nr+1 || nl < 0 || nr < 0 || ld > m || nl+nr < m)
    nrerror("bad args in savgol");
    indx=ivector(1,m+1);
    a=matrix(1,m+1,1,m+1);
    b=vector(1,m+1);
    for (ipj=0;ipj<=(m << 1);ipj++) {      Set up the normal equations of the desired
        sum=(ipj ? 0.0 : 1.0);                 least-squares fit.
        for (k=1;k<=nr;k++) sum += pow((double)k,(double)ipj);
        for (k=1;k<=nl;k++) sum += pow((double)-k,(double)ipj);
        mm=IMIN(ipj,2*m-ipj);
        for (imj = -mm;imj<=mm;imj+=2) a[1+(ipj+imj)/2][1+(ipj-imj)/2]=sum;
    }
    ludcmp(a,m+1,indx,&d);                 Solve them: LU decomposition.
    for (j=1;j<=m+1;j++) b[j]=0.0;
    b[ld+1]=1.0;
    Right-hand side vector is unit vector, depending on which derivative we want.
    lubksb(a,m+1,indx,b);                  Get one row of the inverse matrix.
    for (kk=1;kk<=np;kk++) c[kk]=0.0;      Zero the output array (it may be bigger than
    for (k = -nl;k<=nr;k++) {                  number of coefficients).
        sum=b[1];                          Each Savitzky-Golay coefficient is the dot
        fac=1.0;                               product of powers of an integer with the
        for (mm=1;mm<=m;mm++) sum += b[mm+1]*(fac *= k);     inverse matrix row.
        kk=((np-k) % np)+1;                Store in wrap-around order.
        c[kk]=sum;
    }
    free_vector(b,1,m+1);
    free_matrix(a,1,m+1,1,m+1);
    free_ivector(indx,1,m+1);
}
```

As output, `savgol` returns the coefficients c_n, for $-n_L \le n \le n_R$. These are stored in `c` in "wrap-around order"; that is, c_0 is in `c[1]`, c_{-1} is in `c[2]`, and so on for further negative indices. The value c_1 is stored in `c[np]`, c_2 in `c[np-1]`, and so on for positive indices. This order may seem arcane, but it is the natural one where causal filters have nonzero coefficients in low array elements of `c`. It is also the order required by the function `convlv` in §13.1, which can be used to apply the digital filter to a data set.

The accompanying table shows some typical output from `savgol`. For orders 2 and 4, the coefficients of Savitzky-Golay filters with several choices of n_L and n_R are shown. The central column is the coefficient applied to the data f_i in obtaining the smoothed g_i. Coefficients to the left are applied to earlier data; to the right, to later. The coefficients always add (within roundoff error) to unity. One sees that, as befits a smoothing operator, the coefficients always have a central positive lobe, but with smaller, outlying corrections of both positive and negative sign. In practice, the Savitzky-Golay filters are most useful for much larger values of n_L and n_R, since these few-point formulas can accomplish only a relatively small amount of smoothing.

Figure 14.8.1 shows a numerical experiment using a 33 point smoothing filter, that is,

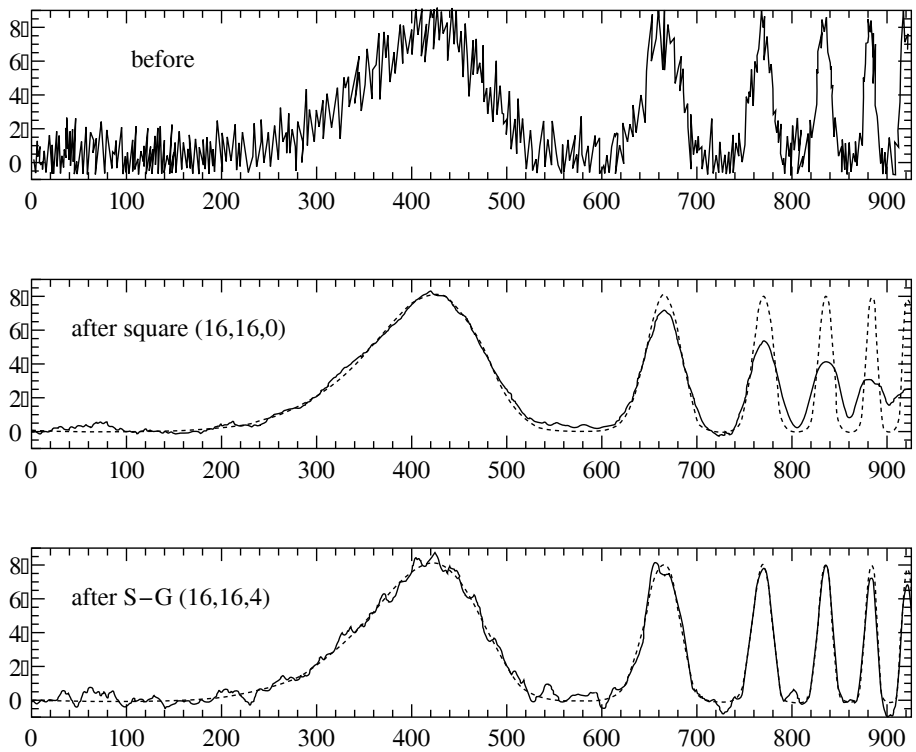

Figure 14.8.1. Top: Synthetic noisy data consisting of a sequence of progressively narrower bumps, and additive Gaussian white noise. Center: Result of smoothing the data by a simple moving window average. The window extends 16 points leftward and rightward, for a total of 33 points. Note that narrow features are broadened and suffer corresponding loss of amplitude. The dotted curve is the underlying function used to generate the synthetic data. Bottom: Result of smoothing the data by a Savitzky-Golay smoothing filter (of degree 4) using the same 33 points. While there is less smoothing of the broadest feature, narrower features have their heights and widths preserved.

$n_L = n_R = 16$. The upper panel shows a test function, constructed to have six "bumps" of varying widths, all of height 8 units. To this function Gaussian white noise of unit variance has been added. (The test function without noise is shown as the dotted curves in the center and lower panels.) The widths of the bumps (full width at half of maximum, or FWHM) are 140, 43, 24, 17, 13, and 10, respectively.

The middle panel of Figure 14.8.1 shows the result of smoothing by a moving window average. One sees that the window of width 33 does quite a nice job of smoothing the broadest bump, but that the narrower bumps suffer considerable loss of height and increase of width. The underlying signal (dotted) is very badly represented.

The lower panel shows the result of smoothing with a Savitzky-Golay filter of the identical width, and degree $M = 4$. One sees that the heights and widths of the bumps are quite extraordinarily preserved. A trade-off is that the broadest bump is less smoothed. That is because the central positive lobe of the Savitzky-Golay filter coefficients fills only a fraction of the full 33 point width. As a rough guideline, best results are obtained when the full width of the degree 4 Savitzky-Golay filter is between 1 and 2 times the FWHM of desired features in the data. (References [3] and [4] give additional practical hints.)

Figure 14.8.2 shows the result of smoothing the same noisy "data" with broader Savitzky-Golay filters of 3 different orders. Here we have $n_L = n_R = 32$ (65 point filter) and $M = 2, 4, 6$. One sees that, when the bumps are too narrow with respect to the filter size, then even the Savitzky-Golay filter must at some point give out. The higher order filter manages to track narrower features, but at the cost of less smoothing on broad features.

To summarize: Within limits, Savitzky-Golay filtering does manage to provide smoothing

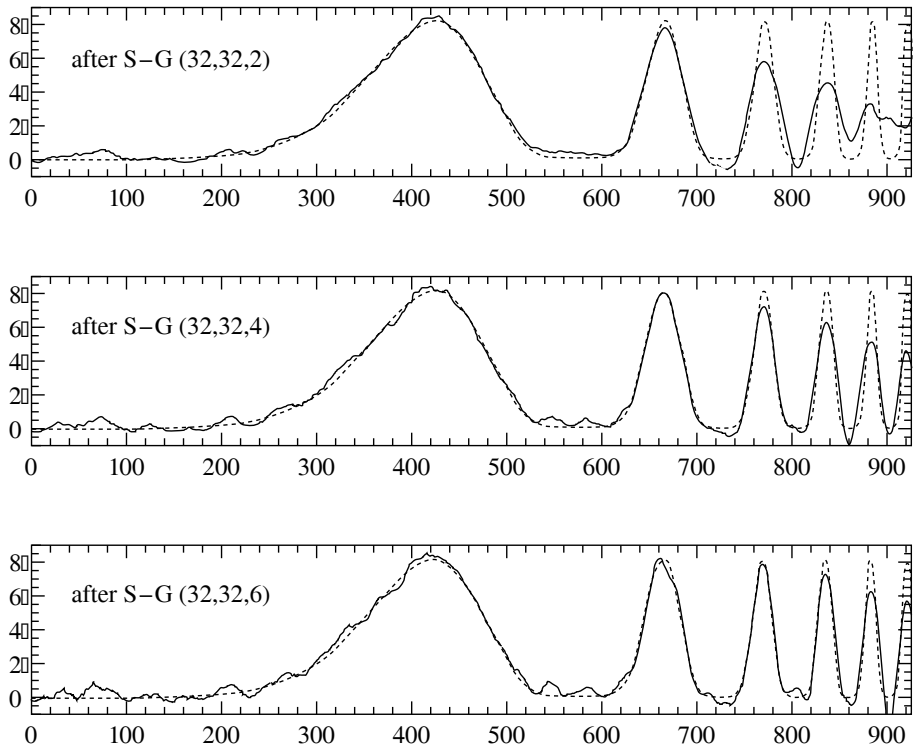

Figure 14.8.2. Result of applying wider 65 point Savitzky-Golay filters to the same data set as in Figure 14.8.1. Top: degree 2. Center: degree 4. Bottom: degree 6. All of these filters are inoptimally broad for the resolution of the narrow features. Higher-order filters do best at preserving feature heights and widths, but do less smoothing on broader features.

without loss of resolution. It does this by assuming that relatively distant data points have some significant redundancy that can be used to reduce the level of noise. The specific nature of the assumed redundancy is that the underlying function should be locally well-fitted by a polynomial. When this is true, as it is for smooth line profiles not too much narrower than the filter width, then the performance of Savitzky-Golay filters can be spectacular. When it is not true, then these filters have no compelling advantage over other classes of smoothing filter coefficients.

A last remark concerns irregularly sampled data, where the values f_i are not uniformly spaced in time. The obvious generalization of Savitzky-Golay filtering would be to do a least-squares fit within a moving window around each data point, one containing a fixed number of data points to the left (n_L) and right (n_R). Because of the irregular spacing, however, there is no way to obtain universal filter coefficients applicable to more than one data point. One must instead do the actual least-squares fits for each data point. This becomes computationally burdensome for larger n_L, n_R, and M.

As a cheap alternative, one can simply pretend that the data points *are* equally spaced. This amounts to virtually shifting, within each moving window, the data points to equally spaced positions. Such a shift introduces the equivalent of an additional source of noise into the function values. In those cases where smoothing is useful, this noise will often be much smaller than the noise already present. Specifically, if the location of the points is approximately random within the window, then a rough criterion is this: If the change in f across the full width of the $N = n_L + n_R + 1$ point window is less than $\sqrt{N/2}$ times the measurement noise on a single point, then the cheap method can be used.

CITED REFERENCES AND FURTHER READING:

Savitzky A., and Golay, M.J.E. 1964, *Analytical Chemistry*, vol. 36, pp. 1627–1639. [1]

Hamming, R.W. 1983, *Digital Filters*, 2nd ed. (Englewood Cliffs, NJ: Prentice-Hall). [2]

Ziegler, H. 1981, *Applied Spectroscopy*, vol. 35, pp. 88–92. [3]

Bromba, M.U.A., and Ziegler, H. 1981, *Analytical Chemistry*, vol. 53, pp. 1583–1586. [4]