Abstract
The trapezoidal algorithm will be discussed as an approximation to numerical integration in the context of the analysis of data from human clinical trials. Several nonstandard characteristics of clinical data will be addressed. The algorithm will be implemented in SAS® code using explicitly subscripted arrays.

Background
In human clinical trials one often has to measure the concentration of a new investigational drug in a subject’s blood. To do this, data are measured at discrete time points over a specified interval. This is known as collecting information on the bioavailability of a pharmaceutical drug. Similarly, one might be interested in the amount of any number of substances, natural or artificial, typically found in blood, e.g., blood glucose levels. In some cases a stimulus might be applied to the subject. The measured data would represent the body’s reaction as evidenced by an increase or decrease in the concentration of a particular substance. How does one evaluate data such as this, which is collected multiple times during a specified interval?

One might analyze the data sampled at each point separately. Alternatively, the discrete measurements can represent the skeleton of a continuous curve (see Figure 1). Data points from an individual subject define a curve and can be evaluated as the subject’s response over a fixed period of time. In the standard case of a bioavailability study, the subject starts with a drug concentration level of zero, i.e., before drug administration. Later, as the drug is absorbed into the person's system, the concentration of the drug will be some value greater than zero. The curve represents this deviation from zero or baseline. The area under the curve is calculated as a measurement of the subject's total deviation from baseline.

If one knows the equation of the curve, then the area under it would be calculated by numerical integration. However, in the setting described above, one does not have that information and must use a numerical approximation. One such approximation to integration is the trapezoidal rule. The trapezoidal algorithm will be presented in detail later.

Instead of a typical bioavailability situation, one might apply a stimulus (e.g., candy bar) to the subject and record his reaction (e.g., change in blood glucose levels). In this case, the baseline value is defined as the initial concentration of a substance. Therefore, the baseline value may be greater than zero. The deviation from baseline in response to stimulus is represented by the curve. However, the curve may fall below baseline (see Figure 2). A curve with both positive and negative portions is numerically integrated in two sections. The same approach is used when applying the trapezoidal rule.

In order to apply the trapezoidal algorithm to these two sections of the curve, one needs to know the time at which the curve crosses baseline. This information may not be readily available since it may occur between two specified sampling points. This specific time can be calculated as will be demonstrated.

Finally, given that one has sequential data measured at several time points over a specified interval, the results can be represented as total deviation from the baseline value. A positive area under the curve is denoted AUC+ while in the negative direction it is AUC−. The trapezoidal rule approximates the area under the curve whether it is a deviation above or below the baseline value.

Solution
To approximate numerical integration, the trapezoidal rule divides the area bounded by the curve and baseline into a series of trapezoidal regions. The area of each trapezoid is calculated and by summing the series, one obtains an approximation of the area under the curve, AUC+ or AUC−.

At the point where the curve crosses baseline, there are two triangular shaped regions (see Figure 3). One needs to find the area of each triangle and add it to either AUC+ or AUC−. If the difference between time A and time B is relatively small, one can assume the section of the curve between points A and B is almost straight. One then needs to calculate the area of two similar triangles.

Let:

\[ a = \text{point A - baseline} \]
\[ b = \text{point B - baseline} \]
\[ \text{known} = \{ \text{time A - time B} \} \]
\[ x = (\text{time curve crosses baseline}) - \text{time A} \]
\[ y = \text{known} - x \]

Corresponding sides of similar triangles vary in length proportionally. Therefore,

\[ a:b = x:y \]
\[ a:b = x:(\text{known} - x) \]
\[ a(\text{known} - x) = b \times x \]
\[ (a \times \text{known}) - (a \times x) = b \times x \]
\[ x = a - \text{known} \]
\[ \frac{a}{a + b} \]
The area of a triangle is \( \frac{1}{2} \times \text{(base)} \times \text{(height)} \).

Thus, where A and B are points on opposite sides of baseline:

\[
\text{AUC}_+ = \frac{1}{2} \times a \times x \\
\text{AUC}_- = \frac{1}{2} \times (a \times \text{known}) \div (a + b)
\]

and \( \text{AUC}_- = \frac{1}{2} \times b \times y \)

\[
\text{AUC}_- = \frac{1}{2} \times b \times \left[ \text{known} \div (a + b) \right]
\]

The remaining area that needs to be approximated can be divided into trapezoidal regions. The base of each trapezoid is defined by the time span between two consecutive sampling points. This algorithm allows for unequal time spans between sampling points. To approximate the area under the curve between points A and B, multiply the base of the trapezoid by its height. Consider the midpoint of a line segment connecting points A and B. The distance of this midpoint from baseline is the height of the trapezoid. If points A and B are above baseline the area calculated is added to \( \text{AUC}_+ \). However, if both points are below baseline, the height is negative and thus a portion of \( \text{AUC}_- \) is calculated.

Using the trapezoidal rule described above, an equation for the area under the curve between points A and B is as follows.

\[
\text{AUC}_{AB} = \frac{1}{2} \times \left[ \text{(point A - baseline)} + \text{(point B - baseline)} \right] \times \text{time A - time B}
\]

The trapezoidal algorithm described here is efficiently implemented in SAS code using explicitly subscripted arrays. The case that the curve crosses baseline is accounted for in the code and thus both an \( \text{AUC}_+ \) and \( \text{AUC}_- \) can be calculated.

**Code**

```sas
*** CALCULATE AREA UNDER THE CURVE (AUC) ***;
*** USING THE TRAPEZOIDAL ALGORITHM. ***;
*** DEFINE FOUR EXPLICITLY SUBSCRIPTED ARRAYS: ***;
*** CONC -- MEASURED DATA ***;
*** TIMES -- TIME POINTS ***;
*** AUCP -- POSITIVE (+) TRAPEZOIDAL AREAS ***;
*** AUCN -- NEGATIVE (-) TRAPEZOIDAL AREAS ***;

ARRAY CONC[10] CONC1 - CONC10;
ARRAY TIMES[10] TIME1 - TIME10;
ARRAY AUCP[9] AUCP1 - AUCP9;
ARRAY AUCN[9] AUCN1 - AUCN9;

*** CHECK FOR MISSING VALUES IN FIRST AND LAST POSITION. IF FOUND, SET AREA TO MISSING. ***;
*** AREA UNDER CURVE. AUCPOS REPRESENTS ***;
*** AUC+ AND AUCNEG REPRESENTS AUC-. ***;

   AUCPOS = .;
   AUCNEG = .;
END /* IF */;

*** INTERPOLATE WHERE MISSING VALUES. ***;
*** DOES NOT HANDLE CASE OF TWO OR MORE CONSECUTIVE MISSING CONCENTRATIONS. ***;
ELSE DO;
   DO I = 2 TO 9;
      IF CONC[I] = . OR CONC[I+1] = . THEN DO;
      CONC[I] = (CONC[I-1] + CONC[I+1]) / 2;
      END /* DO */;

      *** SUBTRACT BASELINE CONCENTRATION FROM MEASURED CONCENTRATION AT EACH TIME POINT. ***;
      DO I = 2 TO 10;
         CONC[I] = CONC[I] - CONC[I-1];
      END /* DO */;

      *** SET VALUE OF BASELINE TO ZERO. ***;
      *** THIS IS EQUIVALENT TO SUBTRACTING THE BASELINE VALUE FROM ITSELF. ***;
      CONC[1] = 0;

      *** INITIALIZE VARIABLES AND ARRAYS. ***;
      AUCPOS = 0;
      AUCNEG = 0;
      DO I = 1 TO 9;
         AUCP[I] = 0;
         AUCN[I] = 0;
      END /* DO */;

      *** IMPLEMENT TRAPEZOIDAL ALGORITHM. ***;
      DO I = 1 TO 9;
         *** CALCULATE AREA OF TRAPEZOID ABOVE BASELINE. ***;
         IF CONC[I] > 0 AND CONC[I+1] > 0 THEN
            AUCP[I] = ((CONC[I] + CONC[I+1]) / 2) * ABS(TIMES[I] - TIMES[I+1]);
         END /* IF */;

         *** CALCULATE AREA OF TRAPEZOID BELOW BASELINE. ***;
         ELSE IF CONC[I] <= 0 AND CONC[I+1] <= 0 THEN
            AUCN[I] = ((CONC[I] + CONC[I+1]) / 2) * ABS(TIMES[I] - TIMES[I+1]);
         END /* ELSE */;

         *** IF THE CURVE_crosses BASELINE BETWEEN THE DATA POINTS. FIRST, ***;
         *** INITIALIZE VARIABLES. ***;
```

872
ELSE DO;
    BASE1 = 0;
    AUC1 = 0;
    BASE2 = 0;
    AUC2 = 0;
    BASE1 = (ABS(CONC[I])
        * ABS(TIMES[I] - TIMES[I+1]))
        / (ABS(CONC[I]) + ABS(CONC[I+1]));
    AUC1 = (CONC[I] * BASE1) / 2;
    BASE2 = ABS(TIMES[I] - TIMES[I+1])
        - BASE1;
    AUC2 = (CONC[I+1] * BASE2) / 2;
    IF AUC1 > 0 THEN
        AUCP[I] = AUC1;
    ELSE IF AUC1 < 0 THEN
        AUCN[I] = AUC1;
    IF AUC2 > 0 THEN
        AUCP[I] = AUC2;
    ELSE IF AUC2 < 0 THEN
        AUCN[I] = AUC2;
    END /* ELSE */;
**SUM THE SERIES OF TRAPEZOIDAL AREAS TO FIND AUC+ AND AUC-.**
AUCPOS = AUCPOS + AUCP[I];
AUCNEG = AUCNEG + AUCN[I];
END /* DO */;
END /* ELSE */;

Summary
The implementation of the trapezoidal algorithm in code is a good application of explicitly subscripted arrays. When one needs to do the same manipulation to many data points, arrays enable efficient coding, e.g., when calculating the area of individual trapezoidal regions. Use of the explicit subscript to access an array element before or after the current one makes it easy to interpolate where there are missing values. Furthermore, the use of two arrays to store collected data, concentration and time, sets up a one-to-one correspondence between the measured data and the sampling time and therefore easily handles cases where there are unequal time spans between sampling points. Finally, the application of explicitly subscripted arrays makes the unwieldy calculations of the algorithm easily manageable.

Conclusion
In the pharmaceutical industry one may have information consisting of several data points collected during a specified interval of time. This data may be used in bioavailability calculations or to measure the concentration of substances in blood. Each subject's results can be analyzed as the total deviation from baseline by integrating the curve represented by the data points. Since the equation of the curve is not usually known, the trapezoidal algorithm provides a good approximation to the area under the curve. The special case of the curve crossing baseline can be handled by a straightforward algorithm and efficiently implemented in SAS code using explicitly subscripted arrays.

For more information, contact the author:
Joan E. Baron
Syntex Research, A4-200
P.O. Box 10850
Palo Alto, CA 94303
SAS is the registered trademark of the SAS Institute Inc., Cary, NC, USA.
Standard Case Bioavailability Study: baseline = 0, equal time intervals

Figure 1
Nonstandard Case: baseline = initial concentration, unequal time intervals

Figure 2
Figure 3