QUADRATIC SPLINE SUBROUTINE PACKAGE

Water-Resources Investigations 82-41
A continuous piecewise quadratic function with continuous first derivative is devised for approximating a single-valued, but unknown, function represented by a set of discrete points. The quadratic is proposed as a treatment intermediate between using the angular (but reliable, easily constructed and manipulated) piecewise linear function and using the smoother (but occasionally erratic) cubic spline. Neither iteration nor the solution of a system of simultaneous equations is necessary to determining the coefficients. Several properties of the quadratic function are given.

A set of five short FORTRAN subroutines is provided for generating the coefficients (QSC), finding function value and derivatives (QSY), integrating (QSI), finding extrema (QSE), and computing arc length and the curvature-squared integral (QSK).
QUADRATIC SPLINE SUBROUTINE PACKAGE

by L. A. Rasmussen

Water-Resources Investigations 82-41
For additional information write to:

U.S. Geological Survey
Project Office - Glaciology
1201 Pacific Avenue - Suite 850
Tacoma, Washington 98402-4383
Abstract ................................................................. 1
Introduction .............................................................. 1
Constructing the Spline ............................................... 2
Properties ..................................................................... 5
The Subroutines .......................................................... 7
References ..................................................................... 11
Attachment A, Subroutine QSC ...................................... 12
Attachment B, Subroutine QSE ....................................... 13
Attachment C, Subroutine QSI ....................................... 14
Attachment D, Subroutine QSK ...................................... 15
Attachment E, Subroutine QSY ...................................... 16
A continuous piecewise quadratic function with continuous first derivative is devised for approximating a single-valued, but unknown, function represented by a set of discrete points. The quadratic is proposed as a treatment intermediate between using the angular (but reliable, easily constructed and manipulated) piecewise linear function and using the smoother (but occasionally erratic) cubic spline. Neither iteration nor the solution of a system of simultaneous equations is necessary to determining the coefficients. Several properties of the quadratic function are given.

A set of five short FORTRAN subroutines is provided for generating the coefficients (OSC), finding function value and derivatives (OSY), integrating (OSI), finding extrema (QSE), and computing arc length and the curvature-squared integral (QSK).

INTRODUCTION

The continuous piecewise quadratic function with continuous derivative is a member of the class of functions known as polynomial splines of zero deficiency; that is, a piecewise polynomial of degree $k$ with continuity of derivatives $0, 1, \ldots, k-1$. The widely used cubic spline and the very widely used continuous piecewise linear function are both members of this class.

Although splines of even degree are little used directly, because of the asymmetry of the required boundary conditions, there is much justification for the existence of algorithms for constructing and manipulating quadratic splines. The derivative of a cubic spline is a quadratic spline, and the integral of a
continuous piecewise linear function is a quadratic spline.

The quadratic spline generally has a jump discontinuity of the second derivative at the junction points, which in the function presented here are taken to be the data points. Although lacking continuity of the second derivative, it is free from the extraneous inflection points between data points, which sometimes occur with cubic splines (de Boor, 1978). The splines in tension, devised to avoid that defect, often concentrate the curvature near the data points (Ahlberg and others, 1967).

CONSTRUCTING THE SPLINE

Among the several criteria that could be used for generating a quadratic spline, the one chosen here leads to a compact, direct algorithm that does not require a boundary condition derivative. It maximizes the agreement of the quadratic spline with separately estimated values of the slopes at the given points.

Given $y_i$ at each of $x_1 < x_2 < ... < x_n$ on a differentiable, single-valued, but unknown function $\phi(x)$, a piecewise quadratic $y = F(x)$ that is continuous in function value and first derivative is constructed so that between each of the $n-1$ pairs of successive points $x_i, x_{i+1}$ the function is a separate quadratic

$$F(x) = \begin{cases} y_1 & (x = x_1) \\ f_i(x) & (x_i < x < x_{i+1}) \end{cases}$$

where

$$f_i(x) = a_i x^2 + b_i x + c_i$$

which has slope

$$f'_i(x) = df_i(x)/dx = 2a_i x + b_i$$

2.
The continuity conditions require for $2 \leq i < n-1$

\[ \begin{align*}
  f_{i-1}(x_i) &= f_i(x_i) = y_i \\
  f'_{i-1}(x_i) &= f'_i(x_i) = s_i
\end{align*} \]

(4)

where the slopes $s_i$ are to be determined, and the boundary conditions require

\[ \begin{align*}
  f_1(x_1) &= y_1 \\
  f_{n-1}(x_n) &= y_n
\end{align*} \]

(5)

Thus $F(x)$ is a polynomial spline of degree 2 and deficiency zero.

In terms of the slope $s_i$, the coefficients of $f_i(x)$ are given, in turn, by

\[ \begin{align*}
  a_i &= (R_i - s_i)/(x_{i+1} - x_i) \\
  b_i &= s_i - 2a_i x_i \\
  c_i &= y_i - (a_i x_i + b_i) x_i
\end{align*} \]

(6)

where

\[ R_i = (y_{i+1} - y_i)/(x_{i+1} - x_i) \]

(7)

When equation (3) is applied at $x_{i+1}$ to $f_i(x)$, in which $a_i$ and $b_i$ are taken from equation (6), there occurs the recurrence relation

\[ s_{i+1} = 2R_i - s_i \]

(8)

which can be used to give all subsequent $s_i$ linearly in terms of $s_1$

\[ s_i = (-1)^{i+1} \left[ s_1 + 2 \sum_{j=1}^{i-1} (-1)^j R_j \right] \]

(9)

If equation (9) is written in the form

\[ s_i = g_i s_1 + h_i \]

(10)

in which $g_1=1$ and $h_1=0$, subsequent $g_i$ and $h_i$ obey the convenient recurrence relations

\[ \begin{align*}
  g_{i+1} &= -g_i \\
  h_{i+1} &= 2R_i - h_i
\end{align*} \]

(11)
The construction of $F(x)$ therefore has one degree of freedom, which through equations (6,9) may be expressed in terms of $s_1$. Chosen here as the property of $F(x)$ to be optimized, since it leads to a closed form expression for $s_1$, is a measure of the agreement between the $s_i$ and separately estimated values $z_i$ of the slopes of $q(x)$ at the $x_i$. The $z_i$ are obtained from $n-2$ auxiliary quadratics $T_2(x)$, $T_3(x)$, ..., $T_{n-1}(x)$ where $T_i(x)$ passes through $(x_{i-1}, y_{i-1})$, $(x_i, y_i)$, and $(x_{i+1}, y_{i+1})$; it has coefficients

$$T_i(x) = a_i x^2 + b_i x + c_i$$

which are given, in turn, by

$$a_i = (R_i - R_{i-1})/(x_{i+1} - x_{i-1})$$
$$b_i = R_i - (x_i + x_{i+1})a_i$$
$$c_i = y_i - (a_i x_i + b_i) x_i$$

The $z_i$ are obtained by differentiating equation (12)

$$T_i'(x) = dT_i(x)/dx = 2a_i x + b_i$$

and setting

$$z_i = \begin{cases} T_2'(x_1) & (i=1) \\ T_i'(x_i) & (2 \leq i \leq n-1) \\ T_{n-1}'(x_n) & (i=n) \end{cases}$$

The quantity to be minimized is

$$I = \sum_{i=1}^{n} \left( \frac{s_i - z_i}{1 + z_i^2} \right)^2$$

which, when substituting for $s_i$ from equation (10) and requiring $\partial I/\partial s_1 = 0$, yields

$$s_1 = \sum_{i=1}^{n} \frac{g_i (z_i - h_i)}{(1 + z_i^2)^2} / \sum_{i=1}^{n} \frac{1}{(1 + z_i^2)^2}$$

Finally, with $s_1$ determined, equations (6-8) are used to get the coefficients of the $n-1$ quadratics $f_i(x)$. 

4.
Properties

Equations (1-17) constitute a compact, direct algorithm for constructing \( F(x) \), as neither numerical iteration nor the solution of a system of simultaneous linear equations is necessary. This is a consequence of the chosen norm, equation (16). Although closed form expressions for arc length

\[
A = \sum_{i=1}^{n-1} p_i \left[ \frac{1}{4a_i} \left[ s \sqrt{1+s^2} + \ln(s+\sqrt{1+s^2}) \right] \right] s_i^{s_{i+1}+1} (a \neq 0),
\]

where

\[
p_i = \begin{cases} 
\frac{1}{4a_i} \left[ s \sqrt{1+s^2} + \ln(s+\sqrt{1+s^2}) \right] s_i^{s_{i+1}+1} & (a \neq 0) \\
(x_{i+1} - x_i) \sqrt{1 + b_i^2} & (a = 0)
\end{cases}
\] (18)

and for the integral of the square of curvature

\[
C = \sum_{i=1}^{n-1} q_i \left[ \frac{3}{4} \tan^{-1}s + s \left[ 3 + 2/(1+s^2) \right] / (1+s^2) \right] \frac{s_i^{s_{i+1}+1}}{s_i} (a \neq 0),
\] (19)

where

\[
q_i = \frac{a_i}{4} \left[ 3\tan^{-1}s + s \left[ 3 + 2/(1+s^2) \right] / (1+s^2) \right] \frac{s_i^{s_{i+1}+1}}{s_i}
\]

may be written, minimizing either \( A \) or \( C \) requires solving a nonlinear equation in \( s_i \).

The quantity in equation (16) roughly resembles the angle \( \theta \) between \( F(x) \) and the slope imputed through equations (13-15) to \( \phi(x) \). From the identity

\[
\tan \theta = \frac{s - z}{1 + sz}
\]

may be obtained

\[
\frac{s - z}{1 + z^2} = \frac{\tan \theta}{1 - z \tan \theta}
\]

which, under the small angle approximation \( \tan \theta \approx \theta \), shows that as the goodness of fit between the \( s_i \) and \( z_i \) increases, \( I \) tends to \( \sum \theta^2 \).
All the following properties can be derived from equations (1-17). Because those derivations cannot be expressed concisely here, and because the results can be easily verified experimentally, the properties are only stated.

If \( \phi(x) \) is a polynomial of degree less than three, \( F(x) \) is identical with \( \phi(x) \).

If the \( x_i \) are equally spaced and if \( n \) is odd, the integral of \( F(x) \) coincides with the result of applying Simpson's rule.

Because of the constant in the denominator of equation (16), \( F(x) \) is not independent of scaling; that is, for the same set of \( x_i \), where \( \lambda \) is a scalar, the \( F(x) \) constructed from the ordinates \( \lambda y_1, \lambda y_2, \ldots, \lambda y_n \) is not \( \lambda \) times the \( F(x) \) constructed from the ordinates \( y_1, y_2, \ldots, y_n \). Therefore, \( F(x) \) is not additive; that is, for the same set of \( x_i \), the \( F(x) \) constructed from the ordinates \( y_1 + \hat{y}_1, y_2 + \hat{y}_2, \ldots, y_n + \hat{y}_n \) is not the sum of the \( F(x) \) constructed from the ordinates \( \hat{y}_1, \hat{y}_2, \ldots, \hat{y}_n \) and the \( F(x) \) constructed from the ordinates \( y_1, y_2, \ldots, y_n \).

A convex \( F(x) \) does not exist for certain convex \( (x_i, y_i) \). This, in fact, afflicts all quadratic splines, regardless of how \( S_i \) may be chosen. If the \( R_i \) are monotonic, there may exist no \( S_i \) satisfying equation (8) that are monotonic. Consider the five points \((0, 0), (10, 16), (20, 28), (30, 32), (40, 34)\), which from equation (7) give \( R_i = 1.6, 1.2, 0.4, 0.2 \); then, from equation (9), \( S_i = S_1, 3.2 - S_1, S_1 - 0.8, 1.6 - S_1, S_1 - 1.2 \). The requirement that the \( S_i \) be decreasing, because the \( R_i \) are decreasing, leads to the incompatible set of inequalities \( S_1 > 1.6, S_1 < 2.0, S_1 > 1.2, S_1 < 1.4 \) of which the first and last are contradictory.

\( F(x) \) cannot be made periodic.
THE SUBROUTINES

The five subroutines presented here are only the nucleus of a quadratic spline subroutine package. A coefficient-forming subroutine simpler than QSC could be produced by requiring the user to supply the slope $S_1$; this would be consistent with the usual design of cubic spline generators, which require that two boundary condition derivatives be supplied. Subroutine QSC was designed for situations in which the user has no additional information for determining a slope. However, such a coefficient forming subroutine that requires $S_1$ to be supplied would be independent of scaling; it could also be used together with an optimization algorithm to form a quadratic spline with minimum curvature-squared integral or with minimum arc length. Other possibilities are subroutines for finding the intersections of a quadratic spline and a linear function, for integrating the product of two quadratic splines, for finding the point on a quadratic spline nearest a specified external point, etc. All of these suggested subroutines could be written to be compatible with the five included here, as long as they used the same data storage scheme.

In the coding of the subroutines, only a small subset of FORTRAN was used in the hope of permitting ready implementation on a wide variety of computer systems. Execution speed was sacrificed slightly to reduce storage requirements and to simplify use of the subroutines. They were written under the assumption that the stated restrictions will be observed; thus, they do not expend time and storage in checking the data supplied them.

The five subroutines are independent of each other, except that the coefficients generated by QSC may be used by any of the other, but none of the set directly uses any other. They rely on the system on which they are used to supply elementary function generators for real-valued square root, natural
logarithm, and single-argument inverse tangent with result in radians; for
these they assume FORTRAN FUNCTION subprograms named, respectively, SQRT,
 ALOG, and ATAN. On a large scale CDC computer their storage requirements in
central memory words (decimal) is QSC 165, QSY 72, QSI 137, OSE 125, and OSK 124.
The execution time of the coefficient forming subroutine OSC is proportional to
n, which is true also for direct methods for forming cubic splines; but only
about half as many operations are needed to obtain the quadratic coefficients
as are needed to obtain the cubic coefficients.

The usage of the subroutines must follow

\[
\text{DIMENSION } X(N), Y(N), Q(3,N), T(N)
\]

\[
N = n > 2
\]

\[
X = x_1 < x_2 < \ldots < x_n
\]

\[
Y = y_1, y_2, \ldots, y_n
\]

\[
\begin{align*}
Q(3,i) &= a_i \quad (i = 1, \ldots, n) \quad \text{coefficients for cubic splines} \\
Q(2,i) &= b_i \quad (1 < i < n) \quad \text{coefficients for quadratic splines} \\
Q(1,i) &= c_i \quad (i = 1, \ldots, n) \quad \text{temp. scratch storage}
\end{align*}
\]

\[
Q(1,n), Q(2,n), Q(3,n) \text{ are used as temporary scratch storage by subroutine}
\]

\[
\text{QSC.}
\]

To form the coefficients of \( F(x) \)

\[
\text{CALL OSC (N, X, Y, Q)}
\]

Input: \( N, X, Y \)

Output: \( Q \)
To find function value and derivatives of $F(x)$ at $x = U$

CALL QSY (N, X, Q, U, YU, YPU, YPPU)

Input: $N, X, Q, U$
Output: $YU = F(U)$
$YPU = \frac{dF(U)}{dx}$
$YPPU = \frac{d^2F(U)}{dx^2}$

Note: if $U < x_1$, then $x_1$ is used for $U$
if $U > x_n$, then $x_n$ is used for $U$

To integrate $F(x)$ from $x = U$ to $x = V$

CALL QSI (N, X, Q, U, V, T, P)

Input: $N, X, Q, U, V$
Output: $T, P$

Note: if $U < x_1$, then $x_1$ is used for $U$
if $U > x_n$, then $x_n$ is used for $U$
if $V < x_1$, then $x_1$ is used for $V$
if $V > x_n$, then $x_n$ is used for $V$

Note: on the first call $T(N)$ must be preset to zero, and subroutine QSI will form $T(i) = t_i$ for $1 \leq i \leq n-1$ and set $T(N) = 1$; on subsequent calls the $T$-array is used for speedy computation of the integral

Method:

$P = \int_{U}^{V} F(x)dx = \int_{x_1}^{V} F(x)dx - \int_{x_1}^{U} F(x)dx$

using

$\int_{x_1}^{x} F(x)dx = t_i + w_i(x)$ for $x_i \leq x \leq x_{i+1}$

where

$t_i = \begin{cases} -w_1(x_1) & i=1 \\ \sum_{j=1}^{i-1} w_j(x_{j+1}) - \sum_{j=1}^{i} w_j(x_j) & 2 \leq i \leq n-1 \end{cases}$
in which \[ w_i(x) = \int_0^x f_i(x) \, dx = c_i x + b_i x^2/2 + a_i x^3/3 \]

To find the extreme values of \( F(x) \)

CALL QSE \((N, X, Y, Q, XMAX, YMAX, XMIN, YMIN)\)

Input: \( N, X, Y, Q \)
Output: \( XMAX = x_{\text{max}} \)
\( YMAX = F(x_{\text{max}}) \)
\( XMIN = x_{\text{min}} \)
\( YMIN = F(x_{\text{min}}) \)

where
\[
\begin{align*}
F(x_{\text{max}}) & > F(x) & x_{\text{max}} < x & < x_{\text{max}} \\
F(x_{\text{min}}) & < F(x) & x_{\text{min}} < x & < x_{\text{min}} \\
\end{align*}
\]

To compute the arc length and the curvature-squared integral of \( F(x) \)

CALL QSK \((N, X, Q, A, C)\)

Input: \( N, X, Q \)
Output: \( A, C \)

Method: Equation (18) is used for the arc length \( A \), and equation (19) is used for the curvature-squared integral \( C \).
REFERENCES


SUBROUTINE QSC (N, X, Y, Q)

C

find coefficients Q of N-1 piecewise quadratic with continuous first derivative, through N given points (x,y) with increasing x

C

dimension x(n), y(n), q(3,n)

NM1 = N-1

C

form r(1) in q(1,1), form z(1) in q(2,1)

C

q(1,1) = (y(2)-y(1))/(x(2)-x(1))

d = 1.00

dx = x(2) - x(1)

q(1,1) = (y(2)-y(1))/dx

acar = \( q(1,1) - q(2,1) \)/\( x(2) - x(1) \)

bcar = \( q(1,1) + q(3,1) \)/2

w(i) = 2.0 * w(i - 1) - w(i - 2)

v = u + c * (w(i) - i * h + 1) * w

v = v + w

if (i - n) 2100, 2300, 2300

2100 g = -g

2200 h = 2.0 * q(i,1) - h

2300 x = u/v

C

form N-1 sets of quadratic coefficients

c(1) in q(1,1), b(1) in q(2,1), a(1) in q(3,1)

C

du 3100 i = 2, n

q(i,1) = (q(i-1,1) - 3)/(x(i+1) - x(i))

q(i,1) = 2.0 * x(i) * q(i,1)

r = 2.0 * w(i) - 5

3100 q(i,1) = y(i) - x(i) * (w(2,1) + x(i) = w(3,1))

C

return

end
SUBROUTINE QSE (N, X, Y, Q, XMAX, YMAX, XMIN, YMIN)

GIVEN STORED IN Q THE N-1 PIECEWISE QUADRATIC Y=F(X) THAT PASSES THROUGH N POINTS (X, Y) WITH INCREASING X, FIND (XMAX, YMAX) THE POINT WITH MAXIMUM Y, AND (XMIN, YMIN) THE POINT WITH MINIMUM Y.

OF TWO POINTS WITH THE SAME Y, THE ONE WITH SMALLER X IS CHOSEN

DIMENSION X(N), Y(N), Q(3,N)

NMI = N-1

ASSUME Y(1) IS BOTH GLOBAL MAXIMUM AND GLOBAL MINIMUM

YMAX = Y(1)
XMAX = X(1)
YMIN = Y(1)
XMIN = X(1)

CHECK EACH OF N-1 INTERVALS FOR EXTREME VALUES

DO 1000 I = 1,NMI

CHECK IF LAST POINT IN INTERVAL IS EXTREME SO FAR

100 IF ((Y(I+1)-YMAX) 200, 200, 100
100 YMAX = Y(I+1)
XMAX = X(I+1)
GO TO 400

200 IF ((Y(I+1)-YMIN) 300, 400, 400
300 YMIN = Y(I+1)
XMIN = X(I+1)

Determine if extremum of i-th quadratic is in the interval and, if so, if is extreme value so far

400 IF (Q(3, I)) 500, 1000, 500
500 XE = (-Q(2, I)/Q(3, I)) 1000, 1000, 600
600 YE = Q(1, I) + XE*(Q(2, I) + XE*Q(3, I))
700 IF (YE-YMAX) 800, 1000, 700
700 YMAX = YE
XMAX = XE
GO TO 1000

800 IF (YE-YMIN) 900, 1000, 1000
900 YMIN = YE
XMIN = XE

1000 CONTINUE

RETURN
END
SUBROUTINE QSI (N,X,Q,U,V,T,P)

GIVEN STORED IN Q THE N-1 PIECEWISE QUADRATIC Y=F(X) THAT PASSES
THROUGH N POINTS (X,Y) WITH INCREASING X, AND GIVEN U AND V,
FIND THE INTEGRAL P OF F(X) FROM X=U TO X=V. IF U OR V IS NOT
IN CLOSED INTERVAL X(1),X(N) IT IS CORRECTED TO NEARER ENDPOINT.

DIMENSION X(N),Q(3,N),T(N)
NM1=N-1
UU=AMAX1(X(1),AMIN1(X(N),U))
VV=AMAX1(X(1),AMIN1(X(N),V))
IF (T(N)) 2000,1000,2000

1000 T(1)=0.
   DU 1100 I=1,NM1
   T(I)=T(I)-X(I)*(Q(1,I)+X(I)*(0.5*Q(2,I)+X(I)*Q(3,I)/3.))
1100 T(I+1)=T(I)+X(I+1)*(Q(1,I)+X(I+1)*(0.5*Q(2,I)+X(I+1)*Q(3,I)/3.))
   T(N)=1.
   FORM T-ARRAY (FIRST TIME CALLED WITH THIS Q)

2000 DU 2200 I=1,NM1
   IF (UU-X(1+1)) 2100,2100,2200
   2100 P(U)=T(I)+UU*(Q(I,I)+UU*(0.5*Q(2,I)+UU*Q(3,I)/3.))
   GO TO 3000
2200 CONTINUE

INTEGRAL OF F(X) FROM X(1) TO U

3000 DU 3200 I=1,NM1
   IF (VV-X(1+1)) 3100,3100,3200
   3100 P(V)=T(I)+VV*(Q(I,I)+VV*(0.5*Q(2,I)+VV*Q(3,I)/3.))
   GO TO 4000
3200 CONTINUE

INTEGRAL OF F(X) FROM X(1) TO V

4000 P=P(V)-P(U)

RETURN
END
SUBROUTINE WSK (N, X, Q, A, C)
C
GIVEN STORED IN Q THE N-1 PIECEWISE QUADRATIC Y=F(X) THAT PASSES
THROUGH N POINTS (X,Y) WITH INCREASING X, FIND FROM X(1) TO X(N)
BOTH THE ARC LENGTH AND THE INTEGRAL OF THE CURVATURE SQUARED.
C
DIMENSION X(N), Q(3,N), DA(2), DC(2)
NM1=N-1
A=0.
C=0.
C
DO 4000 I=1,NM1
IF (Q(3,I)) 2000,1000,2000
1000 A=A+(X(I+1)-X(I))**2
GO TO 4000
C
2000 I=1
DO 3000 J=1,2
Y=Q(3,I)*X(I)+Q(2,I)
U=1.+Y*Y
V=SQR(T(U)
DC(J)=3.*ATAN(Y)+(3.+2.*U)*Y/U
DA(J)=ALOG(Y+V)+Y*V
3000 I=I+1
C
C=C+0.25*(DC(2)-DC(1))*Q(3,I)
A=A+0.25*(DA(2)-DA(1))/Q(3,I)
4000 CONTINUE
C
RETURN
END
SUBROUTINE QSY (N, X, Q, U, YU, YPU, YPPU)

C GIVEN X=U AND N-1 PIECEWISE QUADRATIC Y=F(X) STORED IN Q AND
C PASSING THROUGH N POINTS (X, Y) WITH INCREASING X, FIND AT U
C ORIGINATE YU=F(U), FIRST DERIVATIVE YPU, AND SECOND DERIVATIVE YPPU
C IF U IS NOT IN CLOSED INTERVAL X(1), X(N) NEAREST ENDPOINT IS USED
C
C DIMENSION X(N), Q(3,N)
C UU=AMAX1(X(1), AMIN1(X(N), U))
C
C DD 2000 I=2,N
C IF (UU-X(I)) 1000,1000,2000
C
C 1000 YU=Q(I, I-1)+UU*(Q(I+1, I-1)+UU*Q(I+2, I-1))
C YPPU=2.*Q(I+3, I-1)
C YPU=UU*YPPU+Q(I+2, I-1)
C GO TO 9000
C
C 2000 CONTINUE
C
C 9000 RETURN
C
END