Table 29 Generation of an "**equivalent**" **ellipsoidal** meridional shape for a BIGBOSOR4 model of this multi-segment shell of revolution (Fig.2). These computations are carried out in SUBROUTINE **x3y3**, which is included with the **bosdec** library listed in Table a15.

```
    ==================================================================
c This version of SUBROUTINE BOSDEC is for an "equivalent" ellipsoidal
c head. The "equivalent" ellipsoidal head is constructed because BOSOR4
c (bigbosor4) finite elements tend to "lock up" for shells of revolution
c in which the meridional curvature varies significantly within a single
c shell segment.
c
c The "equivalent" ellipsoidal head consists of a user-defined number of
c toroidal segments that match as well as possible the contour of the
c ellipsoidal head. The meridional curvature of each toroidal segment
c is constant in that segment. Therefore, there is no problem of finite
c element "lock up" in a segmented model of this type.
c
c For each toroidal segment, bigbosor4 needs three points for input:
c (x1,y1), (x2,y2), and (x3,y3). (x1,y1) and (x2,y2) lie on the
c ellipsoidal contour and are the (x,y) coordinates at the two ends of
c the toroidal segment. (x3,y3) is the center of meridional curvature
c of the toroidal segment. The trick is to obtain (x3,y3) so that the
c toroidal segment best fits the ellipsoidal contour in that segment.
c
c We use the following procedure to get (x3,y3):
c
c 1. The equation of the ellipse is
c
c     x^2/a^2 + y^2/b^2 = 1.0                                   (1)
c
c 2. The equation for the normal to the ellipse at (x1,y1) is:
c
c     y - y1 = (y1/x1)(a^2/b^2)(x - x1)                         (2)
c
c 3. The equation for the normal to the ellipse at (x2,y2) is:
c
c     y - y2 = (y2/x2)(a^2/b^2)(x - x2)                         (3)
c
c 4. These two straight lines in (x,y) space intersect at (x03,y03),
c    with (x03,y03) are given by:
c    x03 = (b2 - b1)/(a1 - a2);   y03 = (a2*b1 - a1*b2)/(a2 - a1) (4)
c    in which a1, b1 and a2, b2 are:
c
c     a1 = (y1/x1)(a^2/b^2);      b1 = -a1*x1 + y1              (5)
c     a2 = (y2/x2)(a^2/b^2);      b2 = -a2*x2 + y2              (6)
c
c 5. For an ellipse the distance from the point (x03,y03) to (x1,y1) is
```

```
c    different than the distance from the point (x03,y03) to (x2,y2)
c    because the meridional curvature varies along the contour of the
c    ellipse. We wish to find a new point (x3,y3) in the neighborhood
c    of (x03,y03) for which the distance from (x3,y3) to (x1,y1) equals
c    the distance from (x3,y3) to (x2,y2). For such a point the
c    "equivalent" segment will be a toroidal segment in which the
c    meridional curvature is constant along the segment arc.
c
c 6. The square of the distances from (x03,y03) to (x1,y1) and to (x2,y2)
c    are:
c
c     d1sq = (x1 - x03)**2 + (y1 - y03)**2                         (7)
c     d2sq = (x2 - x03)**2 + (y2 - y03)**2                         (8)
c
c    and the difference of these is:
c
c     delsq = d1sq - d2sq                                          (9)
c
c 7. We determine the location of the center of meridional curvature of
c    the "equivalent" torioidal segment by allocating half of delsq to
c    each (distance)**2, d1sq and d2sq. We then have two (distance)^2
c    that are equal:
c
c     (x1 - x03)**2 + (y1 - y03)**2 - delsq/2                     (10)
c     (x2 - x03)**2 + (y2 - y03)**2 + delsq/2                     (11)
c
c 8. Suppose we let
c
c     x3 = x03 + dx  ;             y3 = y03 + dy                  (12)
c
c    Then we have two nonlinear equations for the unknowns (dx,dy):
c
c    [x1 - (x03+dx)]**2 + [y1 - (y03+dy)]**2 =
c                         (x1 - x03)**2 +(y1 - y03)**2 -delsq/2  (13)
c
c    [x2 - (x03+dx)]**2 + [y2 - (y03+dy)]**2 =
c                         (x2 - x03)**2 +(y2 - y03)**2 +delsq/2  (14)
c
c    These two equations say that the square of the distance from
c    (x3,y3) to (x1,y1) Eq.(13) is equal to that from (x3,y3) to (x2,y2)
c    Eq.(14).
c
c 9. We use Newton's method to solve the two simultaneous nonlinear
c    equations for (dx,dy):
c
c    For the ith Newton iteration, let
c
c     dx(i) = dx(i-1) + u                                         (15)
```

```
c     dy(i) = dy(i-1) + v                                            (16)
c
c    Then we develop two linear equations for u and v for the ith
c    Newton iteration:
c
c     u*2.*(x03-x1+dx(i-1)) +v*2.*(y03-y1 +dy(i-1)) = f1pp       (17)
c     u*2.*(x03-x2+dx(i-1)) +v*2.*(y03-y2 +dy(i-1)) = f2pp       (18)
c
c    in which the right-hand sides, f1pp and f2pp, are rather long
c    expressions given in SUBROUTINE x3y3, where the Newton iterations
c    occur.
=========================================================================
```