Letting $q_{n-1}(x) = \gamma_0 x^{n-1} + \gamma_1 x^{n-2} + \cdots + \gamma_{n-1}$, substituting into Eq. (4.21), and equating like powers of x, we easily see (Problem 3) that

$$q_{n-1}(x) = b_0 x^{n-1} + b_1 x^{n-2} + \cdots + b_{n-1}$$

where $\{b_j\}_{j=0}^{n-1}$ are given by the synthetic division algorithm. Thus this algorithm generates the coefficients of $q_{n-1}(x)$ as well as $p(\alpha)$. By the same reasoning as above, the algorithm can be repeated and will yield $q_{n-1}(\alpha)$ and the coefficients of $q_{n-2}(x)$ (of degree $n - 2$) where

$$q_{n-1}(x) = (x - \alpha)q_{n-2}(x) + r_1, \qquad r_1 = q_{n-1}(\alpha). \tag{4.22}$$

Substituting (4.22) into (4.21), we obtain

$$p(x) = (x - \alpha)^2 q_{n-2}(x) + r_1(x - \alpha) + r_0. \tag{4.23}$$

Differentiating (4.23), we see that $p'(\alpha) = q_{n-1}(\alpha) = r_1$. Obviously this procedure may be continued, using synthetic division on each successive $q_k(x)$, $k = n - 2, n - 3, \ldots, 0$ to yield

$$p(x) = r_n(x - \alpha)^n + r_{n-1}(x - \alpha)^{n-1} + \cdots + r_1(x - \alpha) + r_0. \tag{4.24}$$

Equating (4.24) with the unique Taylor's expansion (4.18), we finally obtain our desired result,

$$r_m = p^{(m)}(\alpha)/m!, \qquad 0 \le m \le n.$$

EXAMPLE 4.6. Let $p(x) = x^6 + 5x^5 + 4x^4 + 3x^3 + 2x^2 + x + 1$ and let $\alpha = 2$. We illustrate synthetic division in Table 4.6 and find $p^{(j)}(\alpha)/j!$ for $j = 0, 1, \ldots, 6$. The entries in any row are the coefficients for $q_{6-j}(x)$, with the last entry in the row being $p^{(j)}(\alpha)/j!$. Thus, $p'(2) = 765$, $p''(2)/2! = 756$, etc. In this illustration, we see, for example, that

$$p(x) = (x - 2)(x^5 + 7x^4 + 18x^3 + 39x^2 + 80x + 161) + 323$$

and

$$p(x) = (x - 2)^6 + 17(x - 2)^5 + 114(x - 2)^4 + 395(x - 2)^3 + 756(x - 2)^2 + 765(x - 2) + 323.$$

TABLE 4.6

$p(x)$	1	5	4	3	2	1	1
$q_5(x)$	1	7	18	39	80	161	323
$q_4(x)$	1	9	36	111	302	765	
$q_3(x)$	1	11	58	227	756		
$q_2(x)$	1	13	84	395			
$q_1(x)$	1	15	114				
$q_0(x)$	1	17					
	1						

It is obvious how Newton's method should utilize synthetic division. In each iteration, let $x_m = \alpha$ and generate $p(x_m) \equiv r_0$ and $p'(x_m) \equiv r_1$ in exact analogy to the procedure in Example 4.6. Then proceed by setting $x_{m+1} = x_m - p(x_m)/p'(x_m)$. There are many other methods (such as the secant method) that involve evaluation of $p(x)$ or its derivatives at specific points, and synthetic division can be utilized by these methods as well. Once again we emphasize that the initial guess of Newton's method, x_0, should be "close" to r, a zero of $p(x)$, in order to ensure convergence. Section 4.4.3 will consider the problem of the approximate location of the zeros of $p(x)$, and will be of immense aid in making a good choice for x_0.

If our goal is to find *all* the zeros of an nth degree polynomial, $p(x)$, then it seems reasonable first to find one zero, say r_1, of $p(x)$. We can next say that $p(x) = (x - r_1)p_1(x)$, and hence any zero of the $(n - 1)$st degree polynomial $p_1(x)$ is a zero of $p(x)$. We now search for a zero of $p_1(x)$, say r_2, write $p_1(x) = (x - r_2)p_2(x)$, and then search for a zero of $p_2(x)$, etc. This process of finding a zero and dividing it out is called *deflation*. Polynomial deflation must be used judiciously, for large errors can result from our inability to find roots exactly. Subroutine POLRT, listed in Fig. 4.9, uses Newton's method and deflation to find all the real roots of a polynomial $p(x)$. Synthetic division is used for evaluation and for finding the coefficients of the deflated polynomials. An initial guess of $x_0 = 0$ is used for Newton's method at each stage of the deflation. Note that subroutine POLRT cannot determine complex roots, but modifications are easily made that enable the subroutine to find complex roots (see Problem 1).

EXAMPLE 4.7. Subroutine POLRT was used to find the zeros of $p(x) = x^5 + x^4 - 9x^3 - x^2 + 20x - 12$ where $p(x) = (x - 2)(x + 2)(x + 3)(x - 1)^2$. A tolerance TOL = 0.00001 was used in this example. The program found the following estimates to the roots, listed in the order found:

0

0.9998991E 01 ?
0.2000000E 01
0.1000101E 01
−0.2000031E 01
−0.2999979E 01.

To illustrate the effect of not having these roots precisely, we listed the coefficients of the deflated polynomial of degree 4, found after the first root was divided out. These coefficients were

0.1000000E	01	(coefficient of x^4)
0.1999899E	01	(coefficient of x^3)
−0.7000303E	01	(coefficient of x^2)
−0.7999597E	01	(coefficient of x)
0.1200121E	02	(constant term).

These coefficients indicate that the deflated polynomial does not have precisely the remaining zeros of $p(x)$ as its zeros. For example, the constant term, 12.00121, is the product of the roots of the deflated polynomial whereas the product of 2, −2, −3, and 1 is 12.

```
      SUBROUTINE POLRT(A,ROOT,TOL,N,MTOL,NROOT)
      DIMENSION A(21),B(21),C(21),ROOT(20)
C
C  SUBROUTINE POLRT USES NEWTON'S METHOD AND DEFLATION TO FIND THE
C  REAL ROOTS OF A POLYNOMIAL.  THE CALLING PROGRAM MUST SUPPLY THE
C  DEGREE OF THE POLYNOMIAL, N, THE COEFFICIENTS A(I) (WHERE A(I) IS
C  THE COEFFICIENT OF X**(N+1-I)), A TOLERANCE TOL, AND AN INTEGER
C  MTOL.  THE SUBROUTINE RETURNS AN INTEGER NROOT=NUMBER OF REAL ROOTS
C  FOUND AND THE ROOTS (IN AN ARRAY ROOT).  AT EACH STAGE OF THE
C  DEFLATION, NEWTON'S METHOD TERMINATES WHEN MTOL ITERATIONS HAVE
C  BEEN EXECUTED OR WHEN TWO SUCCESSIVE ITERATES ARE LESS THAN TOL
C  IN ABSOLUTE VALUE.  N MUST BE GREATER THAN 2.
C
      NROOT=0
      NP1=N+1
    1 ITR=0
      X=0.
C
C  SYNTHETIC DIVISION ALGORITHM TO EVALUATE POLYNOMIAL AND ITS
C  DERIVITIVE
C          A                        C = derivative
    2 B(1)=A(1)
      C(1)=B(1)
      DO 3 I=2,N
      B(I)=X*B(I-1)+A(I)
    3 C(I)=X*C(I-1)+B(I)
      B(NP1)=X*B(N)+A(NP1)
C
C  NEWTON'S METHOD UPDATE TO OLD ESTIMATE OF ROOT
C
      XCORR=B(NP1)/C(N)
      X=X-XCORR
      IF(ABS(XCORR).LT.TOL)   GO TO 4
      ITR=ITR+1
      IF(ITR.LT.MTOL)   GO TO 2
      RETURN
    4 NROOT=NROOT+1
      ROOT(NROOT)=X
C
C  SET UP COEFFICIENTS OF DEFLATED POLYNOMIAL
C
      NP1=N
      N=N-1
      DO 5 I=1,NP1
    5 A(I)=B(I)
C
C  USE QUADRATIC FORMULA WHEN DEFLATED POLYNOMIAL HAS DEGREE 2
C
      IF(N.GT.2)   GO TO 1
      DISCRM=A(2)*A(2)-4.*A(1)*A(3)
      IF(DISCRM.GE.0.)   GO TO 6
      RETURN
    6 ROOT(NROOT+1)=(-A(2)+SQRT(DISCRM))/2.
      ROOT(NROOT+2)=(-A(2)-SQRT(DISCRM))/2.
      NROOT=NROOT+2
      RETURN
      END
```

Figure 4.9 Subroutine POLRT.

To analyze the effects of not knowing r_1 exactly as in Example 4.7, suppose the Newton iterates $\{x_n\}_{n=0}^{\infty}$ are converging to a zero, r_1, of $p(x)$. No matter how large n may be, we can expect that $x_n \neq r_1$. Thus we must settle for some $x_n \equiv r'_1$ as an approximation for r_1. (In Section 4.4.3 we will say more about when to terminate an iteration.) By Theorem 4.9, there is a polynomial $p_1(x)$ of degree $n-1$ such that $p(x) = (x - r_1')p_1(x) + p(r_1')$. Even if we had no round-off error in computing the coefficients of $p_1(x)$, $p(r_1')$ is probably not zero. Thus the zeros of $p_1(x)$ are not, in general, exactly the same as the remaining zeros, $\{r_j\}_{j=2}^{n}$, of $p(x)$. As a matter of fact, it can happen that even though r_1' is extremely close to r_1, the zeros of $p_1(x)$ can be quite different from $\{r_j\}_{j=2}^{n}$. If this phenomenon occurs, $p(x)$ is said, as before, to be ill-conditioned (once again we refer to Wilkinson's example of ill-conditioning given in Section 3.3.2). Therefore the most practical strategy to find r_2 is to use the Newton iteration on $p_1(x)$ until it seems to be converging to some approximate zero, r_2'', of $p_1(x)$. Then we let $x_0 = r_2''$, and use Newton's iteration on the *original* polynomial $p(x)$ to generate a corrected approximation, r_2', for the actual zero, r_2. This correction process should be repeated for all of the other approximates as well. This correction strategy is useful in any root-finding method that finds roots one by one.

When deflation is employed, the error (caused both by rounding errors and by truncation of the Newton iteration) incurred by accepting r_1' as an approximation to r_1 not only influences the approximation $r_2' \approx r_2$ but also influences the approximation $r_3' \approx r_3$ and all successive approximations. Furthermore, the error incurred in generating r_2' also influences $r_3' \approx r_3$ and all successive approximations. We can also see that the smaller roots (in magnitude) are more sensitive to error than the larger ones. For example, let r_1 and r_2 be two roots of $p(x)$ with $|r_1| < |r_2|$, and assume that r_1' and r_2' can be found such that $|r_1 - r_1'| = \varepsilon$ and $|r_2 - r_2'| = \varepsilon$. Then the relative errors (which are of more practical importance than the absolute errors) satisfy

$$\frac{|r_1 - r_1'|}{|r_1|} > \frac{|r_2 - r_2'|}{|r_2|}.$$

Thus we see that the accumulation of error with respect to the deflation process has less effect on our approximations if we are able to approximate the smaller roots first. Once again this approximation requires some *a priori* knowledge of the approximate location of the roots, which is the subject of Section 4.4.3. However, an intuitive approach to finding the smallest zero first would be to select the initial guess quite close to 0.

EXAMPLE 4.8. To illustrate some of the effects of deflation, the polynomial $p(x) = x^4 - 4x^3 + 6x^2 - 4x + 1$ was run using subroutine POLRT with TOL = 0.00001. [In this case, $p(x) = (x - 1)^4$.] After 19 iterations, Newton's method provided $r_1' = 1.021412$ as one root. The program deflated $p(x)$ using r_1' as an assumed root, and after 23 iterations provided $r_2'' = 0.9776155$ as a second root. The next deflated polynomial was $x^2 - 2.000972x + 1.001454$, which is seen to have no real zeros (in fact, the zeros of this quadratic are approximately $1.000486 \pm 0.021949i$). Newton's method applied to this quadratic gave a sequence of iterates that exhibited a characteristic oscillatory behavior. The first 20 iterates are shown in Table 4.7.

TABLE 4.7

x_i	$f(x_i)$
0.0000000E 00	0.1001454E 01
0.5004840E 00	0.2504845E 00
0.7509677E 00	0.6274182E−01
0.8766937E 00	0.1580685E−01
0.9405380E 00	0.4076481E−02
0.9745383E 00	0.1156807E−02
0.9968296E 00	0.4968643E−03
0.1064779E 01	0.4615963E−02
0.1028881E 01	0.1289368E−02
0.1006177E 01	0.5149841E−03
0.9609319E 00	0.2047658E−02
0.9868165E 00	0.6704330E−03
0.1011340E 01	0.6008148E−03
0.9836637E 00	0.7667542E−03
0.1006454E 01	0.5187988E−03
0.9629857E 00	0.1888931E−02
0.9881714E 00	0.6341934E−03
0.1013921E 01	0.6637573E−03
0.9892181E 00	0.6103516E−03
0.1016302E 01	0.7333755E−03

PROBLEMS, SECTION 4.4.1

1. A number of modifications of subroutine POLRT are possible and desirable. Either modify this program, or write your own root-finding program that incorporates two obvious improvements.

a) Use the root-refinement idea; take each zero of a deflated polynomial and use it as an initial guess for Newton's method applied to the *original* polynomial $p(x)$. This technique will refine both the root and the deflated polynomial.

b) If after a number of iterations, Newton's method does not appear to be converging, it is possible that $p(x)$ has a complex zero. Include a provision to start the iteration with a complex initial guess, (say $x_0 = i$), when this situation happens. The program will need a capability to perform complex arithmetic if a complex initial guess is used.

Test your program with $p(x) = (x-1)^4$ and with $p(x) = x^6 - 2x^5 + 5x^4 - 6x^3 + 2x^2 + 8x - 8$ (which has zeros $1+i$, $1-i$, 1, -1, $2i$, $-2i$).

2. Let $p(x)$ be given by (4.12) and assume that none of its coefficients is zero. Show that evaluation of $p(\alpha)$ by direct substitution requires at least $2n - 1$ multiplications.
3. Establish that the number, b_n, generated by the synthetic division algorithm satisfies $b_n = p(\alpha)$. [Hint: In (4.16), let $p(x) = P(x)$ and $Q(x) = (x - \alpha)$. Let the coefficients of $Q(x)$ be $b_0, b_1, \ldots, b_{n-1}$ and equate like powers on both sides of (4.16).]
4. Start with Eq. (4.23) and show that $r_2 = p''(\alpha)/2$ where r_2 is obtained in the same manner as r_0 and r_1.
5. Let $p(x) = x^6 + 5x^5 + 4x^4 + 3x^3 + 2x^2 + x + 1$. Utilize synthetic division to find $\{\gamma_j\}_{j=0}^6$, and write $p(x)$ in the form

$$p(x) = \gamma_0(x + 2)^6 + \gamma_1(x + 2)^5 + \gamma_2(x + 2)^4 + \gamma_3(x + 2)^3 + \gamma_4(x + 2)^2 + \gamma_5(x + 2) + \gamma_6.$$

What does this value of γ_6 along with the last entry in the first row of Table 4.6 tell you?

4.4.2. Bairstow's Method

The basic purpose of Bairstow's method is to find a quadratic factor of a polynomial, $p(x)$. Let $p(x)$ be given by (4.12) and let u and v be any two real numbers. Then $p(x)$ can be written in the form

$$p(x) = (x^2 - ux - v)q(x) + b_{n-1}(x - u) + b_n \tag{4.25}$$

$$q(x) \equiv b_0x^{n-2} + b_1x^{n-3} + \cdots + b_{n-3}x + b_{n-2}. \tag{4.26}$$

We first note that $b_0 = a_0$ and the degree of $q(x)$ is $n - 2$. We also emphasize that each b_k is actually a function of u and v that we could find explicitly, should we desire (see Problem 1). Obviously for n very large, the explicit calculation of each b_k in terms of u and v could become quite cumbersome. Fortunately however, we can derive an algorithm that calculates each b_k quite efficiently.

Let $p(x)$ be given by (4.12), let u and v be arbitrary real numbers, and let $b_{-2} = b_{-1} = 0$. Then generate $\{b_k\}_{k=0}^n$ by

$$b_k = a_k + ub_{k-1} + vb_{k-2}, \qquad 0 \le k \le n. \tag{4.27}$$

We leave to the reader (Problem 2) to verify that the $\{b_k\}_{k=0}^n$ given by (4.27) satisfy (4.26) and (4.25). [This is merely a problem of comparing like powers of x in (4.25).] Before we lose sight of our principal problem of finding zeros of $p(x)$, we state the following theorem.

Theorem 4.10

Let u, v, and $p(x)$ be given as above, and let $r(x) = x^2 - ux - v = (x - s_1)(x - s_2)$. Then s_1 and s_2 are zeros of $p(x)$ if and only if $p(x) = r(x)q(x)$. [We see that finding a quadratic factor $r(x)$ of $p(x)$ is equivalent to finding u and v so that $b_{n-1} = b_n = 0$.]