

Now suppose $p'(z)$ is also desired. Then use the b_k 's from above to form

$$q(x) \equiv b_1x^{n-1} + \cdots + b_{n-1}x + b_n.$$

One may verify that $q(x)$ and b_0 are the quotient and remainder when $p(x)$ is divided by $x - z$, i.e.,

$$p(x) = (x - z)q(x) + b_{n+1}.$$

Differentiating this representation for $p(x)$ and evaluating at $x = z$, we see that $p'(z) = q(z)$. Thus $p'(z)$ can be gotten by applying the nested multiplication algorithm to $q(x)$. Moreover, from the standpoint of the schema, the b_k 's are laid out in just the right way for doing this.

Schematic description of algorithm for evaluating $p(z)$, $p'(z)$:

$$\begin{array}{rcccccc}
 z] & a_1 & a_2 & \cdots & a_{n-1} & a_n & a_{n+1} \\
 & & zb_1 & & zb_{n-2} & zb_{n-1} & zb_n \\
 z] & \hline
 & b_1 & b_2 & \cdots & b_{n-1} & b_n & [b_{n+1} = p(z) \\
 & & zc_1 & & zc_{n-2} & zc_{n-1} & \\
 & \hline
 & c_1 & c_2 & \cdots & c_{n-1} & [c_n = p'(z)
 \end{array}$$

The following algorithm evaluates $p(z)$ and $p'(z)$ simultaneously, without creating any new arrays:

```

b ← a1
c ← b
for k = 2 : n
    b ← ak + zb
    c ← b + zc
b ← an+1 + zb
Then b = p(z) and c = p'(z).

```