# Solutions for Homework 3 Foundations of Computational Math 2 Spring 2012

# Problem 3.1

Show that given a set of points

$$x_0, \;\; x_1, \ldots, x_n$$

a Leja ordering can be computed in $O(n^2)$ operations.

**Solution:**

The following MATLAB code due to Higham (see Higham 2002 Second Edition) shows how a Leja ordering can be computed in $O(n^2)$ operations.

```
function [a,perm] = leja(a)
n = max(size(a));
perm= (1:n)';
[t,i] = max(abs(a));
if i ~= 1
   a([1 i]) = a([i 1]) ;
   perm([1 i]) = perm([i 1]) ;
end
p = ones(n,1);
for k=2:n-1
   for i=k:n
     p(i) = p(i) *(a(i)-a(k-1));
   end
   [t, i]=max(abs(p(k:n)));
   i = i+k-1;
   if i ~=k
      a([k i]) = a([i k]);
      p([k i]) = p([i k]);
      perm([k i]) = perm([i k]);
   end
end
```

# Problem 3.2

Consider a polynomial

$$p_n(x) = \alpha_0 + \alpha_1 x + \cdots + \alpha_n x^n$$

$p_n(x)$ can be evaluated using Horner's rule (written here with the dependence on the formal argument $x$ more explicitly shown)

$c_n(x) = \alpha_n$
for $i = n-1 : -1 : 0$
$\quad c_i(x) = xc_{i+1}(x) + \alpha_i$
end

$p_n(x) = c_0(x)$

If the roots of the polynomial are known we can use a recurrence based on

$$p_n(x) = \alpha_n(x - \rho_1) \cdots (x - \rho_n)$$

given by:

$d_0 = \alpha_n$
for $i = 1 : n$
$\quad d_i = d_{i-1} * (x - \rho_i)$
end

$p_n(x) = d_n$

This algorithm can be shown to compute $p_n(x)$ to high relative accuracy. Specifically,

$$d_n = p_n(x)(1 + \mu), \qquad |\mu| \leq \gamma_{2n} u$$

where $\gamma_k = ku/(1 - ku)$ and $u$ is the unit roundoff of the floating point system used.

## 3.2.a

An error analysis of Horner's rule shows that the computed value of the polynomial satisfies

$$\hat{c}_0 = (1 + \theta_1)\alpha_0 + (1 + \theta_3)\alpha_1 x + \cdots + (1 + \theta_{2n-1})\alpha_{n-1}x^{n-1} + (1 + \theta_{2n})\alpha_n x^n$$

where $|\theta_k| \leq \gamma_k$. (The pattern on the subscript is odd numbers, i.e., increment of 2, until the last which is even, i.e., last increment is 1.)

Let

$$\tilde{p}_n(x) = |\alpha_0| + |\alpha_1|x + \cdots + |\alpha_n|x^n$$

Show that

$$\frac{|p_n(x) - \hat{c}_0|}{|p_n(x)|} \leq \gamma_{2n} \frac{\tilde{p}_n(|x|)}{|p_n(x)|} \tag{1}$$

and therefore

$$\kappa_{rel} = \frac{\tilde{p}(|x|)}{|p(x)|}$$

is a relative condition number for perturbations to the coefficients bounded by $\gamma_{2n}$.

## 3.2.b

Equation 1 also yields an a priori bound on the forward error

$$|p_n(x) - \hat{c}_0|$$

that can be computed along with evaluating $p_n(x)$ with Horner's rule.

Write a code that evaluates $p_n(x)$ and the forward error bound using Horner's rule and $p_n(x)$ using the product form . Apply the code to the polynomial

$$p_9(x) = (x-2)^9$$
$$= x^9 - 18x^8 + 144x^7 - 672x^6 + 2016x^5 - 4032x^4 + 5376x^3 - 4608x^2 + 2304x - 512$$

to evaluate $p_9(x)$ in both forms and the a priori bound on forward error at several hundred points in the interval $[1.91, 2.1]$.

You may view $p_9(x)$ evaluated using the product form as exact for the purposes of this exercise. Plot the product form values across the interval and use the forward error bound to plot curves above and below the "exact" product form curve to show where the computed values must lie. (Recall, for IEEE single precision $u \approx 5.9 \times 10^{-8}$ and for IEEE double precision $u \approx 1.1 \times 10^{-16}$.) Then plot the values of $p_9(x)$ computed with Horner's rule and verify the correctness of the a priori bounding curves. Comment on the tightness of the bounds and the computed values of $p_9(x)$.

## 3.2.c

The computed value on step $i$ of Horner's rule satisfies

$$(1+\epsilon_i)\hat{c}_i = x\hat{c}_{i+1}(1+\delta_i) + \alpha_i, \quad |\delta_i| \leq u, \quad |\epsilon_i| \leq u$$

Define $\hat{c}_i = c_i + e_i$ with $e_n = 0$ and $c_i$ the exact value of the parameter in Horner's rule evaluated in exact arithmetic. Show that

$$e_i = xe_{i+1} + x\hat{c}_{i+1}\delta_i - \epsilon_i\hat{c}_i$$

$$|e_i| \leq u\beta_i$$

$$\beta_i = |x|\beta_{i+1} + |x||\hat{c}_{i+1}| + |\hat{c}_i|, \quad \beta_n = 0$$

and therefore we have the bound

$$|p_n(x) - \hat{c}_0| \leq u\beta_0$$

This bound is called a running error bound for Horner's rule and can also be easily incorporated into the code for simulatneous evaluation with the values above. Compare this bounds prediction with those of the a priori bound above.

**Solution:** We have from

$$\hat{c}_0 = (1+\theta_1)\alpha_0 + (1+\theta_3)\alpha_1 x + \cdots + (1+\theta_{2n})\alpha_n x^n$$

that

$$\hat{c}_0 - p_n(x) = \theta_1\alpha_0 + \theta_3\alpha_1 x + \cdots + \theta_{2n}\alpha_n x^n$$

Therefore we can generate the forward error bound as follows:

$$\begin{aligned} |\hat{c}_0 - p_n(x)| &= |\theta_1\alpha_0 + \theta_3\alpha_1 x + \cdots + \theta_{2n}\alpha_n x^n| \\ &\leq |\theta_1||\alpha_0| + |\theta_3||\alpha_1||x| + \cdots + |\theta_{2n}||\alpha_n||x^n| \\ &\leq \gamma_{2n}(|\alpha_0| + |\alpha_1||x| + \cdots + |\alpha_n||x^n|) \\ &= \gamma_{2n}\tilde{p}_n(|x|) \end{aligned}$$

where

$$\tilde{p}_n(x) = |\alpha_0| + |\alpha_1|x + \cdots + |\alpha_n|x^n$$

Dividing through by $|p_n(x)|$ yields relative error and condition number as desired for perturbations whose relative size is less than $\gamma_{2n}$.

The a priori bound clearly can be evaluated at the same time as $p_n(x)$. Figure 1 shows the product form of $p_9(x)$ evaluated over the interval of interest and the upper and lower apriori bounds. The computation was done in double precision. Note how large the bound is compared to unit roundoff for IEEE double precision $u \approx 1.1 \times 10^{-16}$. The product form gives the exact value of $p_9(x)$ to essentially working precision.

The more accurate running error bound provides an estimate of numerical error for each evaluation of $p_9(x)$. Starting from

$$(1+\epsilon_i)\hat{c}_i = x\hat{c}_{i+1}(1+\delta_i) + \alpha_i, \quad |\delta_i| \leq u, \quad |\epsilon_i| \leq u$$

Substituting $\hat{c}_i = c_i + e_i$ and using that the exat value satisfies $c_i = xc_{i+1} + \alpha_i$ we have

$$\begin{aligned} e_i &= xe_{i+1} + x\hat{c}_{i+1}\delta_i - \epsilon_i\hat{c}_i, \quad e_n = 0 \\ |e_i| &\leq |x||e_{i+1}| + |x||\hat{c}_{i+1}||\delta_i| + |\epsilon_i||\hat{c}_i| \\ |e_i| &\leq |x||e_{i+1}| + (|x||\hat{c}_{i+1}| + |\hat{c}_i|)u \end{aligned}$$

So since $e_n = 0$ we can take $\beta_n = 0$ and we have

$$|e_{n-1}| \leq (|x||\hat{c}_{i+1}| + |\hat{c}_i|)u = \beta_{n-1}u$$

Substituting and using induction we have

$$\begin{aligned} |e_i| &\leq |x||e_{i+1}| + (|x||\hat{c}_{i+1}| + |\hat{c}_i|)u \\ &\leq \beta_{i+1}u + (|x||\hat{c}_{i+1}| + |\hat{c}_i|)u = \beta_i u \end{aligned}$$

and the result follows.

The running bound clearly can be evaluated at the same time as $p_n(x)$ and the a priori bound. Figure 2 shows the computed value of $p_9(x)$ using Horner's rule evaluated over the interval of interest and the upper and lower running bounds. The computation was done in double precision. The computed values are connected via a solid line. The running bounds are much tighter than the a priori bounds (note the change in scale on this figure). The computed values do, in fact, lie in the bounded range. Nevertheless the bound is still quite large compared to unit roundoff for IEEE double precision $u \approx 1.1 \times 10^{-16}$. Most disturbing however is the way the computed values oscillate between positive and negative. So if we were looking for the root $x = 2$ in this interval we have no hope due to roundoff contaminating the polynomials values. Consider how a simple algorithm like bisection would be completely unreliable over this interval. Essentially this says that monitoring running bounds can alert us to the situation that no further accuracy is possible. In this case, the root cannot be found with less uncertainty than the size of the interval where numerical error is at least as large as the value of $p_9(x)$.

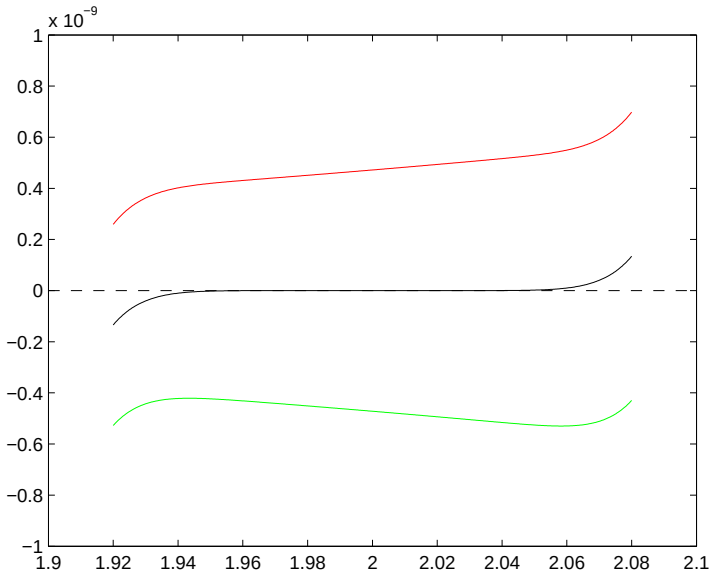


Figure 1: Exact value of $p_9(x)$ and a priori upper and lower bounds on computed value $\hat{c}_0$.

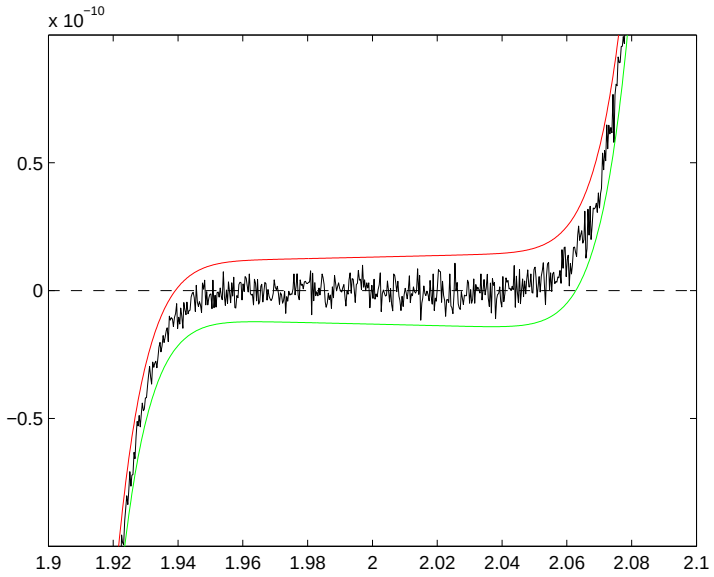


Figure 2: Computed value of $p_9(x)$, i.e., $\hat{c}_0$, and running upper and lower bounds on computed value.

# Problem 3.3

Text exercise 8.10.9 on page 377

**Solution:**

Let $H_3(x) = \alpha_0 + \alpha_1 x + \alpha_2 x^2 + \alpha_3 x^3$ be a cubic polynomial and therefore $H_3'(x) = \alpha_1 + 2\alpha_2 x + 3\alpha_3 x^2$.

The interpolation conditions desired are:

$$
\begin{gathered}
H_3(x) = f(x_0) \\
H_3'(x) = f'(x_0) \\
H_3'(x) = f'(x_1) \\
H_3(x) = f(x_2) \\
H_3(x_0) = \alpha_0 + \alpha_1 x_0 + \alpha_2 x_0^2 + \alpha_3 x_0^3 \\
H_3'(x_0) = \alpha_1 + 2\alpha_2 x_0 + 3\alpha_3 x_0^2 \\
H_3'(x_1) = \alpha_1 + 2\alpha_2 x_1 + 3\alpha_3 x_1^2 \\
H_3(x_2) = \alpha_0 + \alpha_1 x_2 + \alpha_2 x_2^2 + \alpha_3 x_2^3 \\
\begin{pmatrix} 1 & x_0 & x_0^2 & x_0^3 \\ 0 & 1 & 2x_0 & 3x_0^2 \\ 0 & 1 & 2x_1 & 3x_1^2 \\ 1 & x_2 & x_2^2 & x_2^3 \end{pmatrix} \begin{pmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{pmatrix} = \begin{pmatrix} f(x_0) \\ f'(x_0) \\ f'(x_1) \\ f(x_2) \end{pmatrix} \\
x_0 = -1, \quad x_1 = 1, \quad x_2 = 2 \\
f(-1) = 1, \quad f'(-1) = 1, \quad f'(1) = 2, \quad f(2) = 1
\end{gathered}
$$

Note that the matrix depends only on the points $x_i$ and the degree of the polynomial while the righthand-side vector depends only on the values of $f$ and $f'$. The linear system therefore represents the entire interpolation problem and the existence and uniqueness of $H_3(x)$ for this data.

Substituing in the data yields:

$$
\begin{gathered}
H_3(-1) = f(-1) = 1 \rightarrow \alpha_0 - \alpha_1 + \alpha_2 - \alpha_3 = 1 \\
H_3'(-1) = f'(-1) = 1 \rightarrow \alpha_1 - 2\alpha_2 + 3\alpha_3 = 1 \\
H_3'(1) = f'(1) = 2 \rightarrow \alpha_1 + 2\alpha_2 + 3\alpha_3 = 2 \\
H_3(2) = f(2) = 1 \rightarrow \alpha_0 + 2\alpha_1 + 4\alpha_2 + 8\alpha_3 = 1
\end{gathered}
$$

This yields the particular linear system of equations

$$
\begin{gathered}
\begin{pmatrix} 1 & -1 & 1 & -1 \\ 0 & 1 & -2 & 3 \\ 0 & 1 & 2 & 3 \\ 1 & 2 & 4 & 8 \end{pmatrix} \begin{pmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 2 \\ 1 \end{pmatrix} \\
Ma = b
\end{gathered}
$$

There exists a unique $H_3(x)$ if and only if the matrix $M$ is nonsingular. If it is singular then further work must be done to show that no solution exists rather than an infinite number.

The matrix $M$ is singular. There are various ways to show this. One is to derive the factorization

$$M = LU$$

$$L = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 3 & 9/4 & 1 \end{pmatrix}$$

$$U = \begin{pmatrix} 1 & -1 & 1 & -1 \\ 0 & 1 & -2 & 3 \\ 0 & 0 & 4 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

$L$ is nonsingular due to its nonzero diagonal and lower triangular structure. $U$ is singular due to the row of 0 elements and the singularity of $M$ follows. Therefore, there is no unique $H_3(x)$ that satisfies the interpolation conditions.

To see that no $H_3(x)$ exists that satisfies the interpolation conditions we must show that $b \notin \mathcal{R}(M)$. This is the same as showing that $c = L^{-1}b \notin \mathcal{R}(U)$. We have

$$Lc = b$$

$$L = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 3 & 9/4 & 1 \end{pmatrix}$$

$$b = \begin{pmatrix} 1 \\ 1 \\ 2 \\ 1 \end{pmatrix}$$

$$c = \begin{pmatrix} 1 \\ 1 \\ 1 \\ -21/4 \end{pmatrix}$$

Notice that any vector $v \in \mathcal{R}(U)$ must have its last component equal to 0. The vector $c$ does not satisfy this constraint therefore $c = L^{-1}b \notin \mathcal{R}(U)$. It follows that $b \notin \mathcal{R}(M)$ and therefore no $H_3(x)$ exists that satisfies the interpolation conditions.