Electronic Transactions on Numerical Analysis.
Volume 44, pp. 73–82, 2015.
Copyright © 2015, Kent State University.
ISSN 1068–9613.

ETNA
Kent State University
http://etna.math.kent.edu

REVISITING THE STABILITY OF COMPUTING THE ROOTS OF A QUADRATIC POLYNOMIAL[*]

NICOLA MASTRONARDI[†] AND PAUL VAN DOOREN[‡]

Abstract. We show in this paper that the roots x_1 and x_2 of a scalar quadratic polynomial $ax^2 + bx + c = 0$ with real or complex coefficients a, b, c can be computed in an element-wise mixed stable manner, measured in a relative sense. We also show that this is a stronger property than norm-wise backward stability but weaker than element-wise backward stability. We finally show that there does not exist any method that can compute the roots in an element-wise backward stable sense, which is also illustrated by some numerical experiments.

Key words. quadratic polynomial, roots, numerical stability

AMS subject classifications. 65G30, 65G50, 65H04

1. Introduction.

In this paper we consider the very simple problem of computing the two roots of a quadratic polynomial

$$p(x) := ax^2 + bx + c,$$

where the coefficients a, b, c are either in $\mathbb{R}$ or in $\mathbb{C}$, and where $a \neq 0$ in order for the equation to have indeed two roots. This is a very classical problem for which the solution is well known, namely

$$x_{1,2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}.$$

But the straightforward implementation of the above formula is quite often numerically unstable for special choices of the coefficients a, b, c. One would like, on the other hand, to have a computational scheme that produces computed roots $\hat{x}_1$ and $\hat{x}_2$ which correspond to an element-wise backward stable error, i.e., the relative backward errors are of the order of the unit roundoff u for each individual coefficient a, b, and c. In fact, we can assume that a is not perturbed in this process. We will call this *Element-wise Backward Stability (EBS)*:

$$\begin{aligned} &a(x-\hat{x}_1)(x-\hat{x}_2) = ax^2 + \hat{b}x + \hat{c} \\ &|b-\hat{b}| \leq \Delta|b|, \quad |c-\hat{c}| \leq \Delta|c|, \qquad \Delta = \mathcal{O}(u). \end{aligned}$$

We will see that this cannot be proven in the general case, but instead, we can obtain the slightly weaker result of *Element-wise Mixed Stability (EMS)*, which implies that the computed roots $\hat{x}_1$ and $\hat{x}_2$ satisfy

$$\begin{aligned} &a(x-\tilde{x}_1)(x-\tilde{x}_2) = ax^2 + \hat{b}x + \hat{c} \\ &|\hat{x}_1-\tilde{x}_1| \leq \Delta|\tilde{x}_1|, \quad |\hat{x}_2-\tilde{x}_2| \leq \Delta|\tilde{x}_2|, \\ &|b-\hat{b}| \leq \Delta|b|, \quad |c-\hat{c}| \leq \Delta|c|, \qquad \Delta = \mathcal{O}(u), \end{aligned}$$

[*]Received September 29, 2014. Accepted November 13, 2014. Published online on February 6, 2015. Recommended by W. B. Gragg. The work of the first author is partly supported by the GNCS INdAM project "Metodi di regolarizzazione per problemi di ottimizzazione vincolata" and by MIUR Progetto Premiale 2012 "Mathtech". The work of the second author is partly supported by the Belgian Network DYSCO (Dynamical Systems, Control, and Optimization), funded by the Interuniversity Attraction Poles Programme, initiated by the Belgian State, Science Policy Office. The scientific responsibility rests with its authors.

[†]Istituto per le Applicazioni del Calcolo "M. Picone", sede di Bari, Consiglio Nazionale delle Ricerche, Via G. Amendola, 122/D, I-70126 Bari, Italy (n.mastronardi@ba.iac.cnr.it).

[‡]Catholic University of Louvain, Department of Mathematical Engineering, Avenue Georges Lemaitre 4, B-1348 Louvain-la-Neuve, Belgium (paul.vandooren@uclouvain.be).

which means that the computed roots are close to roots of a nearby polynomial, all in a relative element-wise sense.

This last property is also shown to be stronger than the so-called *Norm-wise Backward Stability (NBS)*, which only imposes that the vector of perturbed coefficients is close to the original vector in a relative norm sense:

$$\begin{aligned}&a(x-\hat{x}_1)(x-\hat{x}_2) = ax^2+\hat{b}x+\hat{c}\\ &\left\|\begin{bmatrix}a & b & c\end{bmatrix} - \begin{bmatrix}a & \hat{b} & \hat{c}\end{bmatrix}\right\| \le \Delta \left\|\begin{bmatrix}a & \hat{b} & \hat{c}\end{bmatrix}\right\|, \qquad \Delta = \mathcal{O}(u).\end{aligned}$$

This problem was studied already by several authors, but we could not find any conclusive answer to the EBS for any of the proposed algorithms.

In this paper, we will first consider the case of real coefficients since it is more frequently occurring and the results are slightly stronger. We then show how it extends to the case of complex coefficients. We end with a section on numerical experiments, where we also show that there does *not* exist a method that exhibits EBS for all quadratic polynomials.

2. Real coefficients.

Before handling the general case where all three coefficients are nonzero, we point out that when b and/or c are zero, then the proof of EBS is rather simple.

2.1. Case $c = 0$.

If $c = 0$, then the roots can be computed as follows

$$x_1 := -b/a, \quad x_2 = 0,$$

which is element-wise backward stable since under the IEEE floating point standard, we have that the *computed* roots satisfy

$$\hat{x}_1 = -\mathrm{fl}(b/a) = -b(1+\delta)/a = -\hat{b}/a, \quad \hat{x}_2 = 0, \qquad |\delta| \le u,$$

where u is the unit round-off of the IEEE floating point standard; see [1]. The backward error then indeed satisfies the relative element-wise bounds

$$|\hat{b}-b| \le u|b|, \quad |\hat{c}-c| = 0|c|.$$

2.2. Case $b = 0$.

If $b = 0$, then the roots can be computed as follows

$$x_1 = \sqrt{-c/a}, \quad x_2 := -x_1,$$

which is also element-wise backward stable since under the IEEE floating point standard, we have that the *computed* roots satisfy the element-wise bounds

$$\hat{x}_1 = \mathrm{fl}\left(\sqrt{\mathrm{fl}(-c/a)}\right) = \sqrt{-c(1+\eta)/a}, \quad \hat{x}_2 = -\hat{x}_1, \qquad |\eta| \le \gamma_3 := \frac{3u}{1-3u}.$$

Notice that if $\mathrm{sign}(c) = \mathrm{sign}(a)$, then the roots are purely imaginary. The backward error for this computation satisfies the relative element-wise bounds

$$|\hat{b}-b| \le 0|b|, \quad |\hat{c}-c| \le \gamma_3|c|.$$

We can thus assume now that all coefficients are nonzero. We start by reducing the problem to a simpler “standardized" form in order to simplify the computational steps.

2.3. Scaling the polynomial $p(x)$. We scale the coefficients so that the polynomial is monic, $b_1 := b/a$, $c_1 := c/a$. This step can be performed in a backward and forward stable way since we assumed $a \neq 0$. According to the IEEE floating point standard, we have that the computed values $\hat{b}_1 = \text{fl}(b_1)$ and $\hat{c}_1 = \text{fl}(c_1)$ satisfy the relative element-wise bounds

$$|b_1 - \hat{b}_1| \leq u|b_1|, \quad |c_1 - \hat{c}_1| \leq u|c_1|.$$

This implies that we can as well consider the monic polynomial

$$p_1(x) := p(x)/a = x^2 + b_1 x + c_1.$$

2.4. Scaling the variable x. We transform the variable x to $y := -x/\alpha$, where $|\alpha| := \sqrt{|c_1|}$ and $\text{sign}(\alpha) = \text{sign}(b_1)$, and consider the polynomial $p_1(-\alpha y)/\alpha^2$ (denoted as $q(y)$), which is now monic in y,

$$q(y) := y^2 - 2\beta y + e = 0,$$

and where $\beta \in \mathbb{R}_+$ and $e = \pm 1$. The formulas to compute α, β, and e are

$$\alpha := \text{sign}(b_1)\sqrt{|c_1|}, \quad \beta := |b_1|/(2\sqrt{|c_1|}), \quad e := \text{sign}(c_1) \cdot 1.$$

Since the sign function is exact under relative perturbations, e is computed exactly. It then follows that α and β can be performed in a backward and forward stable way: the computed values $\hat{\alpha} = \text{fl}(\alpha)$ and $\hat{\beta} = \text{fl}(\beta)$ satisfy the relative element-wise bounds

$$|\alpha - \hat{\alpha}| \leq u|\alpha|, \quad |\hat{\beta} - \beta| \leq 2u|\beta|,$$

and e is computed exactly. This implies that we can as well consider the polynomial $q(y)$. We recapitulate this in a formal lemma.

LEMMA 2.1. *The transformations*

$$[\alpha, \beta] = g_a[b, c] \quad \textit{and} \quad [b, c] = g_a^{-1}[\alpha, \beta]$$

between the polynomial $p(x) = ax^2 + bx + c$, with $a \neq 0$, and the monic polynomial $p(-\alpha y)/(a\alpha^2) = q(y) = y^2 - 2\beta y + e$ defined by the forward and backward relations

$$\alpha := \text{sign}(b/a)\sqrt{|c/a|}, \quad \beta := |b/a|/(2\sqrt{|c/a|}),$$

and

$$b = -2a\beta\alpha, \quad c = ae\alpha^2,$$

where a and $e = \text{sign}(c/a) \cdot 1$ are not perturbed, are both element-wise well-conditioned maps.

Proof. If we define the perturbations for the forward map as

$$[\alpha(1 + \delta_\alpha), \beta(1 + \delta_\beta)] = g_a[b(1 + \delta_b), c(1 + \delta_c)],$$

then the above discussion reveals that the relative perturbations $\delta_\alpha, \delta_\beta$ of the result are $\mathcal{O}(u)$ if the relative perturbations of the data, δ_b, δ_c, are $\mathcal{O}(u)$. The same reasoning can be applied to the perturbation of the backward map

$$[b(1 + \delta_b), c(1 + \delta_c)] = g_a^{-1}[\alpha(1 + \delta_\alpha), \beta(1 + \delta_\beta)],$$

which now states that $\delta_b, \delta_c = \mathcal{O}(u)$ provided that $\delta_\alpha, \delta_\beta = \mathcal{O}(u)$ since only multiplications are involved in the backward relations. □

This lemma implies that relative small perturbations in the coefficients of $q(y)$ can be mapped to relative small perturbations in the coefficients of $p(x)$, both element-wise and norm-wise.

2.5. Calculating the roots. The roots of the polynomial $q(y) := y^2 - 2\beta y + e$ are given by

$$y_1 = \beta + \sqrt{\beta^2 - e}, \quad y_2 = \beta - \sqrt{\beta^2 - e}.$$

The way these roots are computed depends now on the values of β and e.
Case 1: $e = -1$ (real roots)

$$y_1 = \mathrm{fl}\left(\beta + \mathrm{fl}\left(\sqrt{\mathrm{fl}(\beta^2+1)}\right)\right), \quad y_2 = -\mathrm{fl}(1/y_1).$$

Case 2: $e = 1$ and $\beta \geq 1$ (real roots)

$$y_1 = \mathrm{fl}\left(\beta + \mathrm{fl}\left(\sqrt{\mathrm{fl}(\beta+1)(\beta-1)}\right)\right), \quad y_2 = \mathrm{fl}(1/y_1).$$

Case 3: $e = 1$ and $\beta < 1$ (complex conjugate roots)

$$y_1 = \beta + \jmath\,\mathrm{fl}\left(\sqrt{\mathrm{fl}(\beta+1)(1-\beta)}\right), \quad y_2 = \overline{y}_1.$$

Let us now check that the roots are computed in a forward stable manner. The error analysis for the operations performed in the IEEE floating point standard gives the following bounds.
Case 1: $e = -1$ (real roots)

$$\hat{y}_1 = \left(\beta + \sqrt{(\beta^2+1)}\right)(1+\eta_3), \quad \hat{y}_2 = -(1/\hat{y}_1)(1+\eta_1), \qquad |\eta_i| \leq \gamma_i.$$

Case 2: $e = 1$ and $\beta \geq 1$ (real roots)

$$\hat{y}_1 = \left(\beta + \sqrt{(\beta+1)(\beta-1)}\right)(1+\eta_4), \quad \hat{y}_2 = (1/\hat{y}_1)(1+\eta_1), \qquad |\eta_i| \leq \gamma_i.$$

Case 3: $e = 1$ and $\beta < 1$ (complex conjugate roots)

$$\hat{y}_1 = \beta + \jmath\left(\sqrt{(\beta+1)(1-\beta)}\right)(1+\eta_3), \quad \hat{y}_2 = \overline{\hat{y}}_1, \qquad |\eta_i| \leq \gamma_i.$$

Notice that these bounds imply forward stability for all these computations. Combining this with Lemma 2.1, we have thus shown the following theorem.

THEOREM 2.2. *The computed roots $\hat{y}_i, i = 1, 2,$ of the polynomial $q(y)$ satisfy the relative forward bounds*

$$|\hat{y}_1 - y_1| \leq \Delta|y_1|, \quad |\hat{y}_2 - y_2| \leq \Delta|y_2|, \qquad \Delta = \mathcal{O}(u),$$

and the transformed roots $\hat{x}_i = \mathrm{fl}(-\alpha\hat{y}_i)$, $i = 1, 2,$ satisfy the mixed bounds

$$\begin{aligned}
&a(x-\tilde{x}_1)(x-\tilde{x}_2) = ax^2 + \hat{b}x + \hat{c}\\
&|\hat{x}_1 - \tilde{x}_1| \leq \Delta|\hat{x}_1|, \quad |\hat{x}_2 - \tilde{x}_2| \leq \Delta|\hat{x}_2|,\\
&|b - \hat{b}| \leq \Delta|b|, \quad |c - \hat{c}| \leq \Delta|c|, \qquad \Delta = \mathcal{O}(u).
\end{aligned}$$

We can therefore also evaluate the backward bound by recomputing the sum and product of the computed roots. We first point out that the sum and product will be real because even when the two computed roots $\hat{y}_1$ and $\hat{y}_2$ are complex, they will be exactly complex conjugate.

Since the product of the exact roots is $e = \pm 1$ and the computed roots are forward stable, we obviously have that the product of the computed roots satisfies

$$\hat{y}_1 \hat{y}_2 = e\big(1 + \mathcal{O}(u)\big),$$

which is element-wise backward stable in a relative sense.

For the sum of the computed roots, the situation is more problematic. Since $|y_1| \geq |y_2|$ and both of these roots are computed in a forward stable way, we have that

$$\hat{y}_1 + \hat{y}_2 = \beta + \mathcal{O}(u)\hat{y}_1, \tag{2.1}$$

but $\hat{y}_1$ can be much larger than β, and the backward error will then be much larger than $\beta \cdot \mathcal{O}(u)$. Let us analyze the three cases. For case 3, the sum of the computed roots is *exactly* 2β since this is a representable number. In case 2, $\hat{y}_1 \leq 2\beta$, and (2.1) then implies backward stability for the element β. But when $\beta \ll 1$, we can *not* obtain a sufficiently small backward error for (2.1) since the recomputed sum has an error that is of the order of $\mathcal{O}(u)\hat{y}_1 \gg \mathcal{O}(u)\beta$. It happens in this special case that element-wise backward stability gets lost.

3. Complex coefficients.

The cases when b and/or c are zero are again easy to handle, but the relative error bounds are slightly larger. Since exact error bounds are more difficult to describe, we prefer to just indicate their order of magnitude. Let us first treat the case of zero values.

If $c = 0$, then the roots can be computed as follows

$$x_1 := -b/a, \quad x_2 = 0,$$

which is element-wise backward stable since under the IEEE floating point standard, we have that the *computed* roots satisfy (see [1])

$$\hat{x}_1 = -\mathrm{fl}(b/a) = -b(1+\delta)/a = -\hat{b}/a, \quad \hat{x}_2 = 0, \qquad |\delta| = \mathcal{O}(u).$$

The backward error then indeed satisfies the relative element-wise bounds

$$|\hat{b} - b| \leq |\delta||b|, \quad |\hat{c} - c| \leq 0|c|, \qquad |\delta| = \mathcal{O}(u).$$

If $b = 0$, then the roots can be computed as follows

$$x_1 = \sqrt{-c/a}, \quad x_2 := -x_1,$$

which is also element-wise backward stable since under the IEEE floating point standard, we have that the *computed* roots satisfy (see [1])

$$\hat{x}_1 = \mathrm{fl}\left(\sqrt{\mathrm{fl}(-c/a)}\right) = \sqrt{-c(1+\eta)/a}, \quad \hat{x}_2 = -\hat{x}_1, \qquad |\eta| = \mathcal{O}(u).$$

The backward error then satisfies the relative element-wise bounds

$$|\hat{b} - b| \leq 0|b|, \quad |\hat{c} - c| \leq |\eta|.|c|, \qquad |\eta| = \mathcal{O}(u).$$

When there are no zero values, we again first apply a scaling of the problem.

3.1. Scaling the polynomial $p(x)$.

As in the real case, we scale the coefficients as follows: $b_1 := b/a$, $c_1 := c/a$. This can be performed in a backward and forward stable way since $a \neq 0$. According to the IEEE floating point standard, we have indeed that

$$|b - b_1| \leq |\Delta||b|, \quad |c - c_1| \leq |\Delta||c|, \qquad |\Delta| = \mathcal{O}(u).$$

This implies that we can as well look at the monic polynomial $p(x)/a = p_1(x) = x^2 + b_1 x + c_1$.

3.2. Scaling the variable x. This becomes more complicated for the case of complex coefficients. We now have that $y := -x/\alpha$, where $|\alpha| := \sqrt{|c_1|}$ and $\arg(\alpha) = \arg(b_1)$. This implies that we can again consider the polynomial

$$q(y) = y^2 - 2\beta y + e = 0, \tag{3.1}$$

where $\beta \in \mathbb{R}_+$ and $|e| = 1$. The formulas to compute α, β, and e are

$$b_1 = |b_1|e_b, \quad c_1 = |c_1|e_c, \quad \alpha := e_b\sqrt{|c_1|}, \quad \beta := |b_1|/(2\sqrt{|c_1|}), \quad e := e_c/(e_b)^2,$$

where $e_b := \arg(b_1)$ and $e_c := \arg(c_1)$. For computational reasons, we also compute the square root f of e, i.e., $f^2 = e$.

We have again a similar lemma describing the transformation between the coefficients of the polynomials

LEMMA 3.1. *The transformations*

$$[\alpha, \beta, f] = h_a[b, c] \quad \textit{and} \quad [b, c] = h_a^{-1}[\alpha, \beta, f]$$

between the polynomial $p(x) = ax^2 + bx + c$, with $a \neq 0$, and the monic polynomial $p(-\alpha y)/(a\alpha^2) = q(y) = y^2 - 2\beta y + f^2$ defined by the forward and backward relations

$$\alpha := \arg(b/a)\sqrt{|c/a|}, \quad \beta := |b/a|/(2\sqrt{|c/a|}), \quad f = \sqrt{\arg(b/a)}/\arg(c/a)$$

and

$$b = -2a\beta\alpha, \quad c = af^2\alpha^2,$$

where a is not perturbed, are both element-wise well-conditioned maps.

Proof. The proof is very similar to that of Lemma 2.1 with the difference that all quantities are complex except for β, which is real, and f, which can be parameterized by a real angle. ☐

This lemma implies again that relative small perturbations in the coefficients of $q(y)$ can be mapped to relative small perturbations in the coefficients of $p(x)$, both element-wise and norm-wise.

3.3. Calculating the roots. The roots of the polynomial (3.1) are now given by

$$y_1 = \beta + \sqrt{\beta^2 - f^2}, \quad y_2 = \beta - \sqrt{\beta^2 - f^2}.$$

But we only need to consider the case where $e = f^2$ is not real since otherwise we can apply the analysis of the previous section. The algorithm for calculating the two roots involves first computing y_1 as the root of largest module and then computing y_2 using $y_2 = f^2/y_1$. If we compute the square root of the complex number $\beta^2 - f^2$ as

$$\gamma = \sqrt{(\beta - f)(\beta + f)},$$

then the roots of the polynomial are given by

$$y_1 := \beta + \text{sign}(\text{real}(\gamma))\gamma, \quad y_2 = f^2/y_1.$$

The rounding errors can be written as follows

$$\begin{aligned}
\hat{\gamma} &= \sqrt{\beta^2 - f^2}(1 + \delta_1), \\
\hat{y}_1 &= (\beta + |\text{real}(\hat{\gamma})|)\,(1 + \delta_2) + \jmath\,\text{sign}(\text{real}(\hat{\gamma}))\,\text{imag}(\hat{\gamma})\,, \\
\hat{y}_2 &= f^2(1 + \delta_3)/\hat{y}_1,
\end{aligned}$$

where all $|\delta_i|, i = 1, 2, 3$, are of the order of the unit round-off u. These formulas yield that y_1 and y_2 can be computed in a forward stable way.

The backward error analysis of these operations will be a problem when β is much smaller than $|f|$. This leads to the same conclusions as in the case of real coefficients: when the sum of the roots is much smaller than the roots themselves, then the relative backward error of the sum can be large despite the fact that the forward errors in the computation as a function of β and f are small.

4. Comparing the different types of stability.

In this section we compare the different types of stability in terms of the constraints that they impose on the computed roots. First of all, it is obvious that EBS implies EMS since EMS follows from EBS simply by choosing

$$\tilde{x}_1 = \hat{x}_1, \quad \text{and} \quad \tilde{x}_2 = \hat{x}_2.$$

We now prove that EBS implies NBS, which is slightly more involved.

LEMMA 4.1. *Let the computed roots $\hat{x}_1$ and $\hat{x}_2$ of $p(x) = ax^2 + bx + c$ satisfy*

$$\begin{aligned} &a(x - \tilde{x}_1)(x - \tilde{x}_2) = ax^2 + \hat{b}x + \hat{c}, \\ &|\hat{x}_1 - \tilde{x}_1| \le \Delta|\tilde{x}_1|, \quad |\hat{x}_2 - \tilde{x}_2| \le \Delta|\tilde{x}_2|, \\ &|b - \hat{b}| \le \Delta|b|, \quad |c - \hat{c}| \le \Delta|c|, \qquad \Delta = \mathcal{O}(u). \end{aligned}$$

Then they also satisfy the norm-wise bound

$$\begin{aligned} &a(x - \hat{x}_1)(x - \hat{x}_2) = ax^2 + \hat{\hat{b}}x + \hat{\hat{c}}, \\ &\left\| \begin{bmatrix} a & b & c \end{bmatrix} - \begin{bmatrix} a & \hat{\hat{b}} & \hat{\hat{c}} \end{bmatrix} \right\| \le 3\Delta \left\| \begin{bmatrix} a & \hat{b} & \hat{c} \end{bmatrix} \right\|, \qquad \Delta = \mathcal{O}(u). \end{aligned}$$

Proof. It follows from the EMS constraints that

$$\hat{\hat{b}} = \hat{b} + a(\hat{x}_1 - \tilde{x}_1 + \hat{x}_2 - \tilde{x}_2) \quad \text{and} \quad \hat{\hat{c}} = \hat{c} + a(\tilde{x}_1\tilde{x}_2 - \hat{x}_1\hat{x}_2),$$

which yields the bounds

$$\begin{aligned} |b - \hat{\hat{b}}| &\le |b - \hat{b}| + |a|\big(|\hat{x}_1 - \tilde{x}_1| + |\hat{x}_2 - \tilde{x}_2|\big), \\ |c - \hat{\hat{c}}| &\le |c - \hat{c}| + |a|\big(|\tilde{x}_1\tilde{x}_2 - \hat{x}_1\hat{x}_2|\big). \end{aligned}$$

Using the constraints of EMS, we then also obtain

$$\begin{aligned} |b - \hat{\hat{b}}| &\le \Delta|b| + \Delta|a|\big(|\tilde{x}_1| + |\tilde{x}_2|\big), \\ |c - \hat{\hat{c}}| &\le \Delta|c| + \Delta|a|\big(|\tilde{x}_1||\tilde{x}_2|\big). \end{aligned}$$

Switching to norms and using the triangle inequality then yields

$$\left\| \begin{bmatrix} 0 & |b - \hat{\hat{b}}| & |c - \hat{\hat{c}}| \end{bmatrix} \right\|_2 \le \Delta \left\| \begin{bmatrix} 0 & |\hat{b}| & |\hat{c}| \end{bmatrix} \right\|_2 + \Delta|a| \left\| \begin{bmatrix} 0 & |\tilde{x}_1| + |\tilde{x}_2| & |\tilde{x}_1\tilde{x}_2| \end{bmatrix} \right\|_2.$$

Because of Lemma A.1 in Appendix A, we also have

$$\Delta|a| \left\| \begin{bmatrix} 0 & |\tilde{x}_1| + |\tilde{x}_2| & |\tilde{x}_1\tilde{x}_2| \end{bmatrix} \right\|_2 \le \sqrt{3}\Delta \left\| \begin{bmatrix} a & |\hat{b}| & |\hat{c}| \end{bmatrix} \right\|_2,$$

and we finally obtain the norm-wise bound

$$\left\| \begin{bmatrix} 0 & |b - \hat{\hat{b}}| & |c - \hat{\hat{c}}| \end{bmatrix} \right\|_2 \le 3\Delta \left\| \begin{bmatrix} a & \hat{b} & \hat{c} \end{bmatrix} \right\|_2. \qquad \square$$

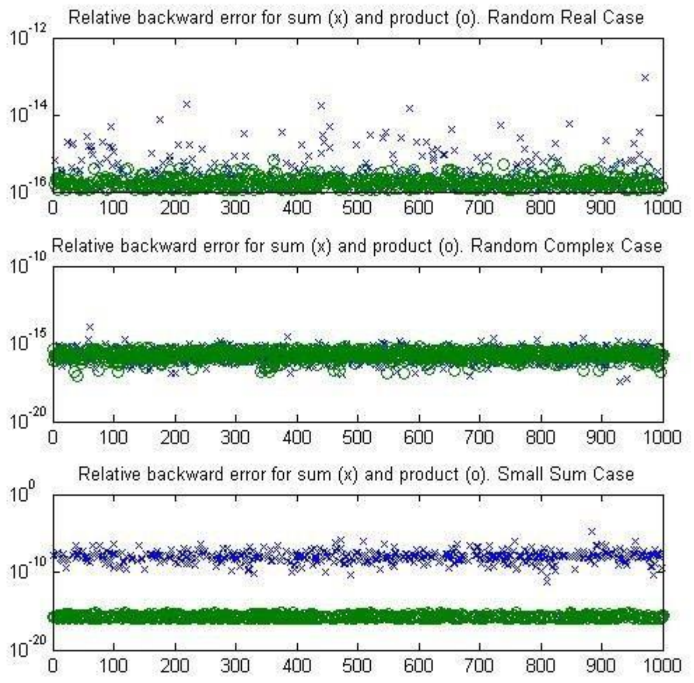

FIG. 5.1. *Backward errors for different cases.*

We also show that in general, EBS cannot always be satisfied, i.e., there does *not* exist any algorithm that achieves this. A counterexample is given by the polynomial

$$y^2 - 2\beta - 1, \quad \text{where } \beta = 2^{-t} + 2^{-2t}, \quad 2^{-2t} \leq u/2, \quad 2^{-t} \approx \sqrt{u}.$$

One easily checks that β is a representable number and that the roots of the polynomial are given by the expansion

$$y_1 = 1 + \beta + \beta^2/2 - \beta^4/8 + \ldots, \quad y_2 = -1 + \beta - \beta^2/2 + \beta^4/8 + \ldots$$

Their exactly rounded values are given by the representable numbers

$$\hat{y}_1 = 1 + 2^{-t}, \quad \hat{y}_2 = -1 + 2^{-t},$$

which gives a sum equal to the representable number

$$\hat{y}_1 + \hat{y}_2 = 2.2^{-t}.$$

But this yields a relative error of the order of $\sqrt{u}$! Moreover, all other representable numbers in the neighborhood of y_1 and y_2 are on a grid of width u, and all possible combinations of their sums will still have a comparable relative error. It is thus impossible to find representable numbers that would satisfy the EBS property.

5. Numerical results.

We tested the routine given in Appendix B for the relative backward errors on three sets of 1000 random quadratic polynomials. We first took random real polynomials, then random complex polynomials, and finally random real polynomials with a very small sum of their roots (of the order of $\sqrt{\epsilon}$). The test results are displayed in Figure 5.1.

The first plot clearly indicates EBS since the relative errors of the recomputed sums and products of the roots is of the order of the unit round-off u. The second plot shows the same results for polynomials with complex coefficients. The third plot shows that for real polynomials $q(y)$ with a very small (but non-zero) coefficient β, EBS cannot be ensured by our algorithm. This is consistent with our analysis, which shows that there does not exist any algorithm to ensure EBS for such polynomials.

Appendix A.

LEMMA A.1. *For any real numbers a, b, and c, we have the inequality*

$$(|a|+|b|)^2+|ab|^2 \leq 2c^2+(a+b)^2+(1+2/c^2)(ab)^2,$$

which also implies for $c^2=3/2$ that

$$\left\|\begin{bmatrix}0 & |a|+|b| & |ab|\end{bmatrix}\right\|_2^2 \leq 3\left\|\begin{bmatrix}1 & a+b & ab\end{bmatrix}\right\|_2^2.$$

Proof. The first inequality follows from

$$(|a|+|b|)^2=(|a|-|b|)^2+4|ab| \leq (a+b)^2+4|ab|,$$

and

$$2(c-|ab|/c)^2 \geq 0 \quad \Longrightarrow \quad 4|ab| \leq 2c^2+2(ab)^2/c^2. \qquad \square$$

Appendix B.

```
function [x1,x2,beta,e,scale] = quadroot(a,b,c)
% Function [x1,x2,beta,e] = quadroot(a,b,c) computes the
% two roots  x1 and x2 of a quadratic polynomial
%  ax^2+bx+c=0 in a stable manner

beta=[];e=[];scale=[];
% special cases of zero elements
if a==0, return, else b1=b/a;c1=c/a; end
if b==0, x1=sqrt(-c1);x2=-x1; return, end
if c==0, x1=-b1; x2=0; return, end
% generic case
if isreal([b1,c1]),
    % with real coefficients
    c1abs=abs(c1);
    scale=sqrt(c1abs)*sign(b1);
    beta=b1/(2*scale);
    e=sign(c1);
    % computing the roots
    if e==-1, y1=beta+sqrt(beta^2+1);y2=-1/y1;
    else,
        if beta >= 1
          y1=beta+sqrt((beta+1)*(beta-1));
          y2=1/y1;
        else
```

```
          im=sqrt((beta+1)*(1-beta));
          y1=beta+j*im;y2=beta-j*im;
        end
    end
 else,
    % with complex coefficients
    scale=sign(b1)*(sqrt(abs(c1)));
    beta=abs(b1)/(2*sqrt(abs(c1)));
    f=sqrt(sign(c1))/sign(b1);
    gamma=sqrt((beta-f)*(beta+f));
    y1=beta+sign(real(gamma))*gamma;
    y2=f^2/y1;
end
x1=-y1*scale;x2=-y2*scale;
```

REFERENCES

[1] N. HIGHAM, *Accuracy and Stability of Numerical Algorithms*, 2nd. ed., SIAM, Philadelphia, 2002.