Proyecciones Journal of Mathematics
Vol. 42, N° 2, pp. 407-431, April 2023.
Universidad Católica del Norte
Antofagasta - Chile

10.22199/issn.0717-6279-5447

Approximating roots by quadratic iteration

Alfredo Poirier
Pontificia Universidad Católica del Perú, Perú
and
Jesús Torres
Pontificia Universidad Católica del Perú, Perú
Received : May 2022. Accepted : December 2022

Abstract

We apply a coctel of elementary methods to the problem of finding the roots of an arbitrary polynomial. Specifically, we combine properties of the iteration $z \mapsto z^2 + c$ with rudimentary Galois theory in order to justify an algorithm to find the roots of a complex polynomial.

Keywords: *Roots of polynomials, iteration of quadratic polynomials, Complex Dynamics.*

MSC numbers, AMS2020: *Primary 37F10, Secondary 65H04.*

1. Introduction

In modern mathematics, the theory of complex dynamical systems has been very active for the last forty years. The purpose of this work is to apply the basic theory of iteration of quadratic maps to a practical problem: how to find a root of an arbitrary polynomial.

As a way of introduction, suppose we want to split $p(x) = x^3 - x^2 - 2x$ into linear factors. Of course, this is the same as finding its roots, which here are precisely $0, -1, 2$. Each of these can be placed in an iteration process of the form $z \mapsto z^2 + c$, where for reasons that will become clear later on we always start at 0. In this line, for $c = 0$ we have an orbit

$$0 \mapsto 0 \mapsto 0 \mapsto \dots,$$

for $c = -1$ we get

$$0 \mapsto -1 \mapsto 0 \mapsto -1 \mapsto \dots,$$

while for $c = 2$ we obtain

$$0 \mapsto 2 \mapsto 6 \mapsto 38 \mapsto \dots.$$

As you can see (and it can be rigorously proved) the orbit for $c = 2$ grows much more rapidly than the other two. The question is whether we can use this fact to our benefit in the task of finding the roots by an automatic procedure.

The idea is to consider $z \mapsto z^2 + x$ starting from 0 for all roots x of p simultaneously. In this way, we get a sequence of polynomials

$$0 \mapsto x \mapsto x^2 + x \mapsto (x^2 + x)^2 + x \mapsto ((x^2 + x)^2 + x)^2 + x \mapsto \dots.$$

It seems that the degree of the successive polynomials grow out of control, but that is not so if we remember that $p(x)$ is always 0. In other words, if we work modulo p, or, more formally, if we follow this process in the quotient ring $\mathbf{R}[x]/(\,p(x)\,)$, we have

$$0 \mapsto x \mapsto x^2 + x \mapsto 241x^2 + 241x \mapsto 348486x^2 + 348487x \mapsto \dots,$$

with only expressions of degree two to consider. Of course, if we plug in $x = 0, -1, 2$ we recover the same results as before.

Next we divide $241x^2 + 241x$ by its leading coefficient in order to obtain $x^2 + x$, a polynomial whose roots are precisely 0 and -1, the two roots with bounded orbit. Unfortunately this is not so simple: in the next step, the

same process yields $x^2 + 348486/3484867x$, where -1 is in this case only an *approximate root*. However, if we keep working, we will always get better and better approximations to the actual roots.

The bottom line is that one root of the polynomial has been lost somewhere. As we can determine —by elementary methods— which one it is, this is tantamount of finding a root of the original polynomial. The reason of this paper is to explain why these phenomena happen and to apply the same procedure to arbitrary polynomials.

In Section 2 we introduce the necessary background material on Complex Analysis. In Section 3 we recall basic ideas concerning the Vandermonde matrix. In Section 4 we prove a result from Number Theory as motivation for the methods borrowed from elsewhere. In Section 5 we present the main theorems needed as theoretical background for the construction of our algorithm in the case of one escaping root. In Section 6 we lift the restriction of a single escaping root. In Section 7 we tackle the case of no escaping roots. In Section 8 we focus on repeated roots. In Section 9 we see how to deal with conflicting leading roots. In Section 10 we wrap up our work into a single algorithm. In Section 11 we hint possible modifications and improvements.

The ideas behind this paper first appear in [4] and [5]. An implementation written in Python3 can be found following the link at the footnote[1].

2. The Mandelbrot set

In this section we recall the basic features of the Mandelbrot set. Key references for this material are [1] and [3].

Let us take a complex number c. We follow the iteration $z \mapsto z^2 + c$ starting from 0, the critical point. In the sequel we use $f^{\circ n}(z_0)$ to denote the n-th iterate of z_0; for most cases z_0 will be 0 or c.

The **critical orbit** (of c) is the set of iterates $f^{\circ n}(0)$ starting from 0. We use $\mathcal{O}_c = \{0, c, c^2 + c, ...\}$ for this set. Sometimes, stretching a little the notation we may even consider them as ordered sets.

For example, if we take c to be equal to -1, we see that $\mathcal{O}_{-1}$ is composed only of 0 and -1, a finite set. On the other hand, for $c = 1$, the orbit $\mathcal{O}_1 = \{0, 1, 2, 5, ...\}$ is infinite and is easy to see (by induction, for instance) that the elements become bigger and bigger.

We say that c **belongs to the Mandelbrot set** if $\mathcal{O}_c$, the orbit of c, is a bounded set. The set of parameters $c \in \mathbf{C}$ for which the orbit $\mathcal{O}_c$ is

[1] https://github.com/Jestefano/ApproximatingRoots

bounded is by definition the **Mandelbrot set** $\mathcal{M}$.

We have already seen that -1 belongs to $\mathcal{M}$. An easy computation yields the same for $0, -2$. On the other side, 1 does not belong to $\mathcal{M}$ as the successive iterates escape to ∞.

We can repeat this exercise with multiple numbers to obtain the plot of Figure 1. The shade is a way to measure how many iterations it takes for the orbit reach 16.

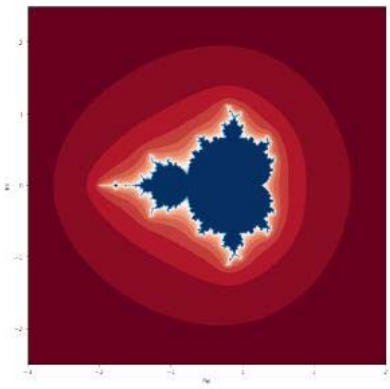

Figure 1. The Mandelbrot set surrounded by equipotentials.

Now we state a theorem that is extremely useful to identify elements that do not belong to $\mathcal{M}$.

Theorem 2.1. *A complex number c belongs to $\mathcal{M}$ if and only if the orbit $\mathcal{O}_c$ is bounded by 2. Furthermore, if c is outside $\mathcal{M}$, then from some iterate on the norm of its iterates is incremental and blows to infinity.*

Proof. For an elementary proof we refer to [1] or [3]. □

For us, more important than $\mathcal{M}$ is its complement. Thus, Theorem 2.1 states that once the orbit of a critical point goes beyond 2, there is no way back and it keeps growing and growing. As a matter of fact, when we add the point at infinity to this complement of $\mathcal{M}$, it becomes simply connected and thus can be uniformized: the complement of the Mandelbrot set is analytically equivalent to the complement of the unitary complex unit disk, and this mapping is "almost" explicit.

Theorem 2.2. *The sets $\hat{\mathbf{C}} - \mathcal{M}$ and $\hat{\mathbf{C}} - \mathbf{D}$ are conformally equivalent. In fact, the isomorphism is given by*

$$\psi(c) = \lim_{k\to\infty} \sqrt[2^k]{f_c^{\circ k}(c)} = c + \frac{a_{-1}}{c} + \ldots,$$

so ψ is tangent to the identity at ∞. ∎

The proof is beyond elementary methods and it can be found in [3].

For practical purposes, the theorem states that $|f_c^{\circ k}(c)|$ grows at a rate A^{2^k}, for some $A > 1$ that depends on c.

The **equipotential function** (of the Mandelbrot set) is defined by setting $\phi(c) = |\psi(c)|$ for $c \notin \mathcal{M}$ and $\phi(c) = 1$ for $c \in \mathcal{M}$. The level sets of ϕ are then called **equipotentials.** (Its logarithm is what is named the **Green function**.)

Notice that $\phi(c)$ is strictly greater than 1 for elements outside $\mathcal{M}$. We use this function to compare the escape growth for different parameters.

Lemma 2.3. *Whenever $\phi(c) > \phi(c')$ holds, then there exists $\lambda < 1$ such that for k big enough $\left|\dfrac{f_{c'}^{\circ k}(c')}{f_c^{\circ k}(c)}\right|$ is less than λ^{2^k}.*

Proof. We first assume $\phi(c') > 1$. By assumption we can choose r subject to $\lim \sqrt[2^k]{|f_c^{\circ k}(c)/f_{c'}^{\circ k}(c')|} > r > 1$. So, for k big we get $|f_c^{\circ k}(c)/f_{c'}^{\circ k}(c')| > r^{2^k} > 1$. To establish the lemma we need $\lambda = 1/r$.

Applying the same ideas as in the last paragraph we can find $\lambda < 1$ so that $|f_c^{\circ k}(c)| > 1/\lambda^{2^k}$. Since $\phi(c') = 1$ implies that c' belongs to the Mandelbot set, points in its orbit satisfy $|f_{c'}^{\circ k}(c')| \leq 2$. The expected result is now an easy Calculus exercise. □

Remark 2.4. *That ϕ is tangent to the identity near ∞ has as a consequence that as R increases, the equipotentials $\phi^{-1}(R)$ start looking more and more like true circles. This of course is not so for small values of R: the closer we get to the Mandelbrot set, the more wiggling the equipotentials behave.*

3. The Vandermonde matrix

In this section we recall well known basic features of the Vandermonde matrix. For extra details we refer the reader to [6].

Given $x_1, ..., x_n$, the **Vandermonde matrix of** $x_1, \ldots, x_n$ is by definition

$$V(x_1, \ldots, x_n) = \begin{bmatrix} 1 & x_1 & \ldots & x_1^{n-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & \ldots & x_n^{n-1} \end{bmatrix}.$$

This matrix will be helpful for switching coordinates from one space to another. In that direction we need the following lemma.

Lemma 3.1. *The determinant of the Vandermonde matrix* $V(x_1, \ldots, x_n)$ *is given by* $\prod_{i<j}(x_j - x_i)$. *As a consequence, the Vandermonde matrix is invertible if and only if all* x_i *are different.*

Proof. See [6]. □

4. A seemingly unrelated result

To justify our line of thought, we prove a Number Theory result applying Complex Dynamics methods. We recall first without proof a truly elementary result.

Lemma 4.1. *Consider two polynomials* $p(x) = a_n x^n + a_{n-1}x^{n-1} + \ldots + a_0$ *and* $d(x) = x^m + b_{m-1}x^{m-1} + \ldots + b_0$ *in* $\mathbf{Z}[x]$, *with* $d(x)$ *monic. Then there exist polynomials* $q(x)$ *and* $r(x)$ *in* $\mathbf{Z}[x]$ *so that* $p(x) = d(x)q(x) + r(x)$ *with* $\deg(r) < \deg(d) = m$. ∎

It is not hard to believe that if an algebraic integer α is such that it and all its conjugates have complex norm less than or equal to 1, then α must be a root of unity. However, the proof of this fact hardly appears in any text. Curiously, using the ideas that we want to develop, we can carry out a proof by an unorthodox method.

Lemma 4.2. *Let* α *be an algebraic integer. If* α *and all its conjugates have complex norm less than or equal to 1, then* α *is a root of unity.*

Proof. Suppose $p(x) = x^n + a_{n-1}x^{n-1} + \ldots + a_0$ is the minimal polynomial of α. Notice that $p(x)$ has no repeated roots, since it is minimal. Consider the orbit $\mathcal{O}_\alpha = \{\alpha, \alpha^2, \alpha^4, \ldots, \alpha^{2^m}, \ldots\}$ under the iteration $z \mapsto z^2$. Given an arbitrary element α^{2^m} of $\mathcal{O}_\alpha$, Lemma 4.1 yields

$$\alpha^{2^m} = p(\alpha)q_m(\alpha) + r_m(\alpha) = r_m(\alpha).$$

with $r_m(x) = r_{m0} + r_{m1}x + \ldots + r_{m(n-1)}x^{n-1} \in \mathbf{Z}[x]$ of degree at most $n-1$. Also, because of $|\alpha^{2^m}| \leq 1$, we have $|r_m(\alpha)| \leq 1$.

On the other side, if c_1, c_2, ... c_n are all the roots of p, then basic Galois theory commands

$$\begin{bmatrix} 1 & c_1 & \dots & c_1^{n-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & c_n & \dots & c_n^{n-1} \end{bmatrix} \begin{bmatrix} r_{m0} \\ r_{m1} \\ \vdots \\ r_{m(n-1)} \end{bmatrix} = V(c_1, c_2, \dots, c_n) \begin{bmatrix} r_{m0} \\ r_{m1} \\ \vdots \\ r_{m(n-1)} \end{bmatrix} = \begin{bmatrix} r_m(c_1) \\ r_m(c_2) \\ \vdots \\ r_m(c_n) \end{bmatrix}.$$

However, as $V(c_1, \dots, c_n)$ is invertible (because all c_i are different), we have

$$\begin{bmatrix} r_{m0} \\ r_{m1} \\ \vdots \\ r_{m(n-1)} \end{bmatrix} = V^{-1}(c_1, c_2, ..., c_n) \begin{bmatrix} r_m(c_1) \\ r_m(c_2) \\ \vdots \\ r_m(c_n) \end{bmatrix}.$$

Notice that all $r_m(c_i)$ belong to a compact set (due to $|r_m(c_i)| = |c_i^{2^m}| \leq 1$). Thus, under $V^{-1}(c_1, c_2, ..., c_n)$, a linear transformation, this compact set maps to a compact set. But then the n-tuples r_{mj} do not only belong to a fixed compact set but have integer entries. This is only possible if they eventually repeat: we get $r_m(x) = r_{\bar{m}}(x)$ for some $\bar{m} \neq m$, which implies $\alpha^{2^m} = \alpha^{2^{\bar{m}}}$. This only happens for roots of unity. □

5. One escaping root

We stress the fact that our goal has always been to factorize an arbitrary polynomial, which without much ado we assume to be monic.

Formally, we work the process of iteration for polynomials modulo p given by $z \mapsto z^2 + x \in \mathbf{C}[x]/(\,p(x)\,)$, starting from 0. The result at the k-th step is denoted by $f^{\circ k}(x)$. The representative modulo p of smaller degree (in practice the only representative of degree smaller than $\deg(p)$) is the **k-th reduced polynomial (under iteration)**.

However —compare the introduction— the coefficients of $f^{\circ k}(x)$ tend to get huge really fast (cf. Lemma 5.1, below). Therefore, we choose $b_k \in \{b_{k1}, \dots, b_{k(n-1)}, 1\}$ subject to $|b_k| = \max_i(|b_{ki}|, 1)$. For technical reasons we introduce the **normalized reduced polynomial** as

$$F_k(x) = \frac{f^{\circ k}(x)}{b_k} = \sum_{j=0}^{n-1} \frac{b_{kj}}{b_k} x^j,$$

Most times $|b_k|$ tends to ∞, but in revenge the coefficients of F_k belong to the unit box, a compact set.

Lemma 5.1. *Consider the k-th iterated polynomials* $f^{\circ k}(x) = \sum_i b_{ki}x^i$. *With the notation above, set* $|b_k| = \max_i(|b_{ki}|, 1)$. *If the collection* $\{|b_k|\}$ *is bounded, then all roots of* $p(x)$ *lay inside the Mandelbrot set* $\mathcal{M}$. *In fact, when at least one root is outside the Mandelbrot set, then necessarily* $|b_k| \to \infty$. *If the roots of* $p(x)$ *are different, the reciprocal is also true.*

Proof. Let c_1, c_2, ..., c_n be the roots of $p(x)$. Each $f^{\circ k}(c_i)$ equals $\sum_j b_{kj}c_i^j$, so we can express this relationship in matrix form as

$$\begin{bmatrix} f^{\circ k}(c_1) \\ \vdots \\ f^{\circ k}(c_n) \end{bmatrix} = \begin{bmatrix} 1 & c_1 & \dots & c_1^{n-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & c_n & \dots & c_n^{n-1} \end{bmatrix} \begin{bmatrix} b_{k0} \\ \vdots \\ b_{k(n-1)} \end{bmatrix}.$$

Since the coefficients $\{b_{kj}\}$ are mapped to the actual iterations of the critical points by a linear transformation, compact sets are mapped to compact sets. Hence, when $\{b_k\}$ is bounded, by definition all c_i belong to the Mandelbrot set.

If all c_i are distinct, the Vandermonde matrix turns out to be invertible, and the reciprocal follows as when the $f^{\circ k}(c_i)$ belong to a compact set, and so do their images. □

To move forward, we fix $p(x) = x^n + a_{n-1}x^{n-1} + ... + a_0 \in \mathbf{C}[x]$ of degree at least 2 with different roots. As a way to make our ideas transparent, we assume that all but one root belong to the Mandelbrot set. We replicate the method hinted in the introduction in the hope that we rescue the remaining roots. The general case will be dealt in subsequent sections along this paper.

The normalized reduced polynomials F_k belong to a compact subset. Any convergent limit of a subsequence is a polynomial of degree at most $n - 1$. Its projective class, that is, any non trivial scalar multiple of it, is referred to from now on as a **horizon polynomial**. When $|b_k|$ goes to ∞, by definition F_k has always at least one coefficient equal to 1, therefore the same holds for any horizon polynomial. Thus horizon polynomials are non-vanishing. The big question is what is the relation between roots of the original polynomials and that of the horizons.

Lemma 5.2. *Suppose* $p(x)$ *has exactly one root off the Mandelbrot set* $\mathcal{M}$. *Then the other roots of* p *are also roots of the horizon polynomials.*

As a consequence, if all roots are different and only one is escaping, then the normalized reduced polynomials converge to a unique horizon polynomial. This is a polynomial of degree $n - 1$ *which divides* p.

Proof. Let the roots be $c_1, \ldots, c_n$, with c_1 escaping; we necessarily have $|b_k| \to \infty$. As $c_2, \ldots, c_n$ are non escaping, Theorem 2.1 tells us that all $|f^{\circ k}(c_i)|$ are bounded by 2. Therefore, for $i = 2, 3, \ldots, n$ we have

$$|F_k(c_i)| = \frac{|f^{\circ k}(c_i)|}{|b_k|} \leq \frac{2}{|b_k|}.$$

By construction, at least one coefficient of $F_k(x)$ is 1. Since it has only n coefficients, one index should repeat infinitely often. Hence, there is a subsequence k_l such that F_{k_l} has only 1's as its j-th coefficient. Working along this subsequence, we can consider a limit F, a polynomial of degree at most $n-1$, whose i-th coefficient is the limit of the i-th coefficient of F_{k_l}. Thus, the j-th coefficient of F is 1, where j is given above. It follows easily from the displayed inequality that in the limit we get $F(c_2) = F(c_3) = \ldots = F(c_n) = 0$. With these two conditions F is uniquely determined, and indeed it is of degree $n-1$.

Now, any horizon polynomial in sight has at least one coefficient equal to 1 and $c_2, \ldots, c_n$ as roots. Therefore it must be a degree $n-1$ polynomial whose projective class agrees with that of F.

Finally, this polynomial divides p as they share $c_2, \ldots, c_n$ as roots. □

We remark that the rate of convergence to the horizon polynomial relays on the speed of the escaping root. When the root is far from $\mathcal{M}$ the polynomials will stabilize soon. In sharp contrast, if the escaping root is close to $\mathcal{M}$ it might take a lot of iterations to find suitable approximations.

Take $p(x) = x^4 - 26.1534x^3 - 50.8349x^2 - 24.2565x - 1.6835$, with roots -1.1176, -0.6455, -0.0833 and 28. As $\mathcal{M} \cap \mathbf{R}$, the real part of the Mandelbrot set, is equal to $[-2, 0.25]$, we conclude that only 28 escapes.

Let us see what happens with the first few iterations. We have displayed in Table 5.1 the iterated polynomial $f^{\circ k}(x)$, since we are concerned about the coefficients, together with $f^{\circ k}(x)/b_{k(n-1)}$. We are considering this variant instead of the k-th reduced polynomial for computational reasons. Monic polynomials are easier to handle computationally. In turn, reduced polynomials are useful when we want to prove statements. We only depict four decimal places in the table to fit the data, a programming language, of course, can manage way more than that.

Notice that from the fifth iteration on the coefficients of $f^{\circ k}(x)$ get unmanageable. However, for $f^{\circ k}(x)/b_{k(n-1)}$, starting with the forth iteration the coefficients do not change their first four decimal places.

Table 5.1: Data for $x^4 - 26.1534x^3 - 50.8349x^2 - 24.2565x - 1.6835$.

Number of iterations	$f^{\circ k}(x)/b_{k(n-1)}$	$f^{\circ k}(x)$
0	$b_{00} = 0$	$d_{00} = 0$
1	$b_{10} = 0$ $b_{11} = 1$	$d_{10} = 0$ $d_{11} = 1$
2	$b_{20} = 0$ $b_{21} = 1$ $b_{22} = 1$	$d_{20} = 0$ $d_{21} = 1$ $d_{22} = 1$
3	$b_{30} = 0.0597$ $b_{31} = 0.8971$ $b_{32} = 1.8411$ $b_{33} = 1$	$d_{30} = 1.6835$ $d_{31} = 25.2565$ $d_{32} = 51.8349$ $d_{33} = 28.1534$
4	$b_{40} = 0.0601$ $b_{41} = 0.8684$ $b_{42} = 1.8465$ $b_{43} = 1$	$d_{40} = 1116002.4067$ $d_{41} = 16119267.9435$ $d_{42} = 34273597.7652$ $d_{43} = 18560881.8449$
5	$b_{50} = 0.0601$ $b_{51} = 0.8684$ $b_{52} = 1.8465$ $b_{53} = 1$	$d_{50} = 4.8520 * 10^{17}$ $d_{51} = 7.0081 * 10^{18}$ $d_{52} = 1.4901 * 10^{19}$ $d_{53} = 8.0697 * 10^{18}$

The lost root is thus $26.1534 - (-1.8465)$ which is equal to 27.9999, no doubt a good approximation considering that we only needed five iterations to grab correctly four decimal places.

As a second example, we consider a negative root. Take

$$p(x) = x^4 + 4.1444x^3 + 3.20668265x^2 + 0.08227144975x - 0.262911670125,$$

with roots $-3.14, -0.8125, -0.431$, and 0.2391. Once again, the only root outside $[-2, 0.25]$ is -3.14.

We focus in small differences with respect to the previous example. We see that it took longer this time for the polynomial to stabilize. This cannot be otherwise as its leading root lays closer to $\mathcal{M}$. Finally, again we recover the lost root using the same method. We compute $1.0044 - 4.1444$, which yields -3.1400.

Table 5.2: Iterates for $x^4 + 4.14x^3 + 3.20x^2 + 0.082x - 0.26$.

$b_{20} = 0$	$d_{20} = 0$
$b_{21} = 1$	$d_{21} = 1$
$b_{22} = 1$	$d_{22} = 1$
$b_{30} = -0.1226$	$d_{30} = 0.2629116$
$b_{31} = -0.4279$	$d_{31} = 0.91772$
$b_{32} = 1.0290$	$d_{32} = -2.20668265$
$b_{33} = 1$	$d_{33} = -2.1444$
$b_{40} = -0.0837$	$d_{40} = 6.8912255$
$b_{41} = 0.0385$	$d_{41} = -3.1745610$
$b_{42} = 0.9902$	$d_{42} = -81.527683$
$b_{43} = 1$	$d_{43} = -82.3319018$
$b_{50} = -0.0837$	$d_{50} = 12200.3961$
$b_{51} = 0.0528$	$d_{51} = -7702.1686$
$b_{52} = 1.0044$	$d_{52} = -146351.2820$
$b_{53} = 1$	$d_{53} = -145710.0801$
$b_{60} = -0.0837$	$d_{60} = 37875472409.5000$
$b_{61} = 0.0528$	$d_{61} = -23914407973.3069$
$b_{62} = 1.0044$	$d_{62} = -454343737751.865$
$b_{63} = 1$	$d_{63} = -452353382871.255$

6. Several escaping roots

It does not really matter whether we have one or one hundred roots outside the Mandelbrot set $\mathcal{M}$. As long as there is one leading root, we should not worry. The following lemma explains this claim.

Lemma 6.1. *If $p(x)$ has two roots c_1, c_2 that satisfy $f^{\circ k}(c_2)/f^{\circ k}(c_1) \to 0$, then c_2 is a root of any horizon polynomial.*

Proof. Iteration evaluated at c_1 has the expansion $f^{\circ k}(c_1) = \sum_{j=0}^{n-1} b_{kj} c_1^j$. By definition we have $|b_{kj}/b_k| \le 1$, thus the bound $\left| \dfrac{f^{\circ k}(c_1)}{b_k} \right| \le \displaystyle\sum_{j=0}^{n-1} |c_1^j| = A$ independent of k. Next we multiply both sides by $|f^{\circ k}(c_2)/f^{\circ k}(c_1)|$ in order

to reach

$$\left|\frac{f^{\circ k}(c_2)}{b_k}\right| \leq A \left|\frac{f^{\circ k}(c_2)}{f^{\circ k}(c_1)}\right| \to 0.$$

Therefore, $f^{\circ k}(c_2)/b_k = \sum_{j=0}^{n-1} b_{kj} c_2^j / b_k$ tends to zero as well. □

A couple of remarks regarding the statement. First, the hypothesis might look a little bit far fetched, but this is not the case as long as we remember Lemma 2.2. We only need one root to be in an equipotential larger than the other for this to happen. Also notice that the stability of the polynomial coefficients depends on the escape rate of the leading root. The bottom line is that in comparison to the single root case, nothing has changed but —perhaps— for the rate of convergence.

Table 6.1: Data for $x^4 + 10.9942x^3 + 3.7174x^2 - 6.7229x - 0.4461$.

$b_{30} = -0.0496$	$d_{30} = 0.4461$
$b_{31} = -0.8586$	$d_{31} = 7.7229$
$b_{32} = 0.3021$	$d_{32} = -2.7174$
$b_{33} = 1$	$d_{33} = -8.9942$
$b_{40} = -0.0421$	$d_{40} = 3930.1620$
$b_{41} = -0.6312$	$d_{41} = 58851.0545$
$b_{42} = 0.4108$	$d_{42} = -38301.74679$
$b_{43} = 1$	$d_{43} = -93222.6392$
$b_{50} = -0.0421$	$d_{50} = 414997230107.2569$
$b_{51} = -0.6312$	$d_{51} = 6214122130105.688$
$b_{52} = 0.4109$	$d_{52} = -4044916688845.686$
$b_{53} = 1$	$d_{53} = -9844041588843.117$
$b_{60} = -0.0421$	$d_{60} = 4.62750008344 * 10^{27}$
$b_{61} = -0.6312$	$d_{61} = 6.92916689302 * 10^{28}$
$b_{62} = 0.4109$	$d_{62} = -4.51035596319 * 10^{28}$
$b_{63} = 1$	$d_{63} = -1.097677284787 * 10^{29}$

Consider

$$p(x) = x^4 + 10.9942x^3 + 3.7174x^2 - 6.7229x - 0.4461,$$

with roots -1 and -0.0645 inside $\mathcal{M}$, and -10.5833 and 0.6536 outside. In Table 6.1 we present the behavior of the first few iterations.

As expected, the terms do not change up to the fourth decimal place after the first five iterations. Here, as in the previous section, it is easy to recover the lost root. We do that by computing $0.4109 - 10.9942$ which amounts to -10.5833, exactly the escaping root. This information allows us to conclude that $\phi(-10.5833)$ is at least as big as $\phi(0.6536)$, where here ϕ is the equipotential function of the complement of the Mandelbrot set.

7. No escaping roots

Even if Theorem 2.1 helps us decide whether a point lives outside the Mandelbrot set or not, it does not tells us when the threshold 2 is reached. In other words, a parameter can be outside the Mandelbrot set $\mathcal{M}$, yet take 1,000 steps for us to notice. Of course we cannot wait forever.

However, if after, say, ten iterates the coefficients of the iteration polynomials remain small, then even under no evidence that the roots belong to $\mathcal{M}$, it is clear that they are too close to $\mathcal{M}$ for the algorithm in its present state to be of any use.

As the Mandelbrot set is located in a vertical strip that spreads from real part -2 to real part 1, what we can do is shift the roots several units to the right or to the left. In our case it is enough to consider a shift of 3 units. Once this is performed, all roots that were close to $\mathcal{M}$ lay outside it now.

Let us consider an example, for instance

$$p(x) = x^4 + 1.5027x^3 + 0.5618x^2 + 1.5027x - 0.4381.$$

The first few iterations are listed below.

Table 7.1: Iterations for $x^4 + 1.5027x^3 + 0.5618x^2 + 1.5027x - 0.4381$.

$d_{20} = 0$	$d_{21} = 1$	$d_{22} = 1$	
$d_{30} = 0.4381$	$d_{31} = -0.5027$	$d_{32} = 0.4381$	$d_{33} = 0.4973$
$d_{40} = -0.0460$	$d_{41} = 1.4041$	$d_{42} = 0.9539$	$d_{43} = 0.40411$
$d_{50} = 0.5118$	$d_{51} = -0.6471$	$d_{52} = 0.5118$	$d_{53} = 0.3528$
$d_{60} = 0.0313$	$d_{61} = 1.2046$	$d_{62} = 1.031$	$d_{63} = 0.2046$

As we can see, the coefficients got stuck in a small set. Therefore it is reasonable go ahead and shift the polynomial 3 units.

Now we proceed to compute the iterations once again but for the modified polynomial

$$\hat{p}(x) = p(x+3) = x^4 + 13.5027x^3 + 68.0861x^2 + 153.4465x + 130.6992.$$

Table 7.2: Table for $x^4 + 13.502x^3 + 68.086x^2 + 153.446x + 130.699$.

$b_{20} = 0$	$d_{20} = 0$
$b_{21} = 1$	$d_{21} = 1$
$b_{22} = 1$	$d_{22} = 1$
$b_{30} = 11.3624$	$d_{30} = -130.6992$
$b_{31} = 13.2531$	$d_{31} = -152.4465$
$b_{32} = 5.8322$	$d_{32} = -67.0861$
$b_{33} = 1$	$d_{33} = -11.5027$
$b_{40} = 25.3192$	$d_{40} = -281337.0308$
$b_{41} = 25.0834$	$d_{41} = -278717.1851$
$b_{42} = 8.5183$	$d_{42} = -94652.2830$
$b_{43} = 1$	$d_{43} = -11111.5966$
$b_{50} = 27.5395$	$d_{50} = -32594000193.6498$
$b_{51} = 26.5256$	$d_{51} = -31393997104.1491$
$b_{52} = 8.7540$	$d_{52} = -10360743821.7405$
$b_{53} = 1$	$d_{53} = -1183533802.8959$
$b_{60} = 27.5000$	$d_{60} = -3.1338 * 10^{20}$
$b_{61} = 26.5000$	$d_{61} = -3.0198 * 10^{20}$
$b_{62} = 8.7500$	$d_{62} = -9.9713 * 10^{19}$
$b_{63} = 1$	$d_{63} = -1.1395 * 10^{19}$

We recover the lost root by computing $8.7500 - 13.5027$, which amounts to -4.7527. After that, we have to remember to undo the translation. So we add 3 to our last answer. This renders -1.7527. Once we replace this number in our original polynomial $p(x)$ we conclude that, indeed, it is a root.

As a final comment, we remark that we should only consider this case once there is nothing else we can do. This presupposes that we have already found all roots outside $\mathcal{M}$. Otherwise we might be reinstating roots inside $\mathcal{M}$ that are truly outside. Fortunately, our algorithm has a natural preference: it starts computing roots the farthest away from the Mandelbrot set

and then proceeds inward. At the end we will have to deal anyway with potential roots in $\mathcal{M}$.

8. Repeated roots

Now we drop unnecessary hypothesis and see how some techniques can be refined. We start by avoiding one of the strongest assumptions so far, the fact that all roots are distinct.

In most algorithms dealing with factorization, multiple (*i.e.*, repeated) roots are a big concern. This algorithm is no exception.

In order to avoid such problem we consider $p/\gcd(p', p)$, where p' stands for the derivative of p. In practice you may rather consider p'/n instead of p' in order to keep both polynomials monic.

Theorem 8.1. *Suppose $p(x)$, of degree n, has distinct roots $c_1, \ldots, c_m$ of multiplicity $\alpha_1, \ldots, \alpha_m$, respectively. If α_i is 1, then p' does not have c_i as root. Else, if α_i is greater than 1, then c_i is a root of p' of multiplicity $\alpha_i - 1$.*

As a consequence $p/gcd(p', p)$ has all c_i as roots of multiplicity 1 (and only them).

Proof. All this is trivial and well known. □

Therefore we have switched our original polynomial for a polynomial with no repeated roots. As a side note, notice that if all roots of $p(x)$ are simple, nothing changes at all.

Remark 8.2. *In order to compute the greatest common divisor of two polynomials we can apply the Euclidean algorithm and make use of the identity $gcd(a, b) = gcd(b, r)$, where r fits as "residue" in any expression of the form $a = d \cdot b + r$.*

Example 8.3. *Take*

$$p(x) = x^4 + 8.05x^3 - 32.2848x^2 - 155.11392x + 458.98085$$

with "normalized" derivative

$$\frac{p'(x)}{4} = x^3 + 6.0375x^2 - 16.1424x - 38.77848.$$

We compute the gcd as shown:

$$\begin{aligned} gcd(p, p'/4) &= gcd(p'/4, p(x) - xp'(x)/4) \\ &= gcd(p'/4, x^3 - 8.0211x^2 - 57.8064x + 228.0650) \\ &= gcd(x^3 - 8.0211x^2 - 57.8064x + 228.0650, x^2 + 2.9636x - 18.9808) \\ &= gcd(x^2 + 2.9636x - 18.9808, x^2 + 3.5345x - 20.7621) \\ &= gcd(x^2 + 3.5345x - 20.7621, x - 3.1199) \\ &= gcd(x - 3.1199, x - 3.1200). \end{aligned}$$

We can see that the last polynomial is $x - 3.12$*, the greatest common divisor of* p *and* p'*. Thus* 3.12 *is the only multiple root of* p*. Therefore, in order to make our work comfortable we consider instead* $p/gcd(p', p) = x^3 + 11.17x^2 + 2.5656x - 147.109248$*.*

The bottom line is that we must find the roots of $p/\gcd(p', p)$ and afterwards by some means or another check the corresponding multiplicities.

9. Several leading roots in the same equipotential

In the previous section we focused on how to reinforce our algorithm to prevent pitfalls when dealing with repeated roots. Now we focus on roots that share an equipotential without being equal.

Before presenting methods that will help us understand the situation, we explain how to detect if this were the case. As discussed previously, a requisite for this to occur pops when iterations blow up but do not converge: this because of Lemma 6.1.

Once we notice that this is happening, we split our original polynomial in two: one that grabs leading roots, and one that takes care of the remaining ones. In order to achieve this we must first extract the greatest common divisor between the horizon iteration and our polynomial, as this conveys information about the slow growing roots. Then we divide $p(x)$ by this factor to have a hold on the leading roots.

Example 9.1. *Consider*

$$p(x) = x^4 + 2.15241x^3 + 2.44106x^2 + 8.02121x - 4.79073.$$

First we check whether it has multiple roots or not. A simple inspection shows that p *and* $p'(x) = 4x^3 + 6.45723x^2 + 4.88212x + 8.02121$ *are relatively prime, which means that they do not share roots.*

Now we are ready to proceed with the iterations.

Table 9.1: Here $p(x) = x^4 + 2.1524x^3 + 2.441x^2 + 8.0212x - 4.7907$.

$b_{30} = -31.4321229016$	$d_{30} = 4.7907346029$
$b_{31} = 46.0663437705$	$d_{31} = -7.0212129108$
$b_{32} = 9.4548474058$	$d_{32} = -1.4410628507$
$b_{33} = 1$	$d_{33} = -0.1524152415$
$b_{40} = 3.7059136534$	$d_{40} = 38.867616982840666$
$b_{41} = -8.6820999959$	$d_{41} = -91.0578520738803$
$b_{42} = 2.3234825168$	$d_{42} = 24.368681241957834$
$b_{43} = 1$	$d_{43} = 10.487998539180962$
$b_{50} = 18.0901979472$	$d_{50} = -8910.701415741842$
$b_{51} = -23.7242681233$	$d_{51} = 11685.879290527198$
$b_{52} = -28.060787214$	$d_{52} = 13821.92152259368$
$b_{53} = 1$	$d_{53} = -492.5706972209647$
$b_{60} = -12.5472238187$	$d_{60} = 1082453609.542889$
$b_{61} = 22.6662248956$	$d_{61} = -1955423550.861008$
$b_{62} = 5.8686163529$	$d_{62} = -506287689.29195577$
$b_{63} = 1$	$d_{63} = -86270367.4680827$
$b_{70} = 4.9047943636$	$d_{70} = 3.1934222659758 * 10^{18}$
$b_{71} = -11.1761015126$	$d_{71} = -7.276556114574 * 10^{18}$
$b_{72} = 1.7830781105$	$d_{72} = 1.16092967776599 * 10^{18}$
$b_{73} = 1$	$d_{73} = 6.5108178431794 * 10^{17}$
$b_{80} = -8.141961013$	$d_{80} = -3.987589461369 * 10^{37}$
$b_{81} = 8.2155621576$	$d_{81} = 4.02363620099677 * 10^{37}$
$b_{82} = 16.9678211535$	$d_{82} = 8.3101239009331 * 10^{37}$
$b_{83} = 1$	$d_{83} = 4.89757867303324 * 10^{36}$

We see that the coefficients are big and, therefore, we expect the iterations to converge fast (see Lemma 6.1). But, surprise, the reader can check by inspecting the table that this is not so. Hence something unexpected is happening as there is more than one leading root.

Now we must factor this polynomial into the leading part and the non-leading part. In order to do so we take the greatest common divisor between $p(x)$ *and the horizon polynomial:*

$$\begin{aligned} &gcd(x^4 + 2.1524x^3 + 2.4410x^2 + 8.0212x - 4.7907, \\ &\qquad\qquad x^3 + 16.9678x^2 + 8.2155x - 8.1419) \\ &= gcd(x^3 + 16.967x^2 + 8.215x - 8.149, x^2 + 0.560x - 0.487) \\ &= gcd(x^2 + 0.560x - 0.487, x - 0.273) \\ &= gcd(x - 0.273, x - 0.343) = 1. \end{aligned}$$

This is annoying as it tells us that all the roots are leading ones. Therefore we must somehow modify the algorithm (or the polynomial).

Mathematically, we have just verified that the roots of our polynomial all lay in the same equipotential of the complement of the Mandelbrot set. Going back to Remark 2.4 we can ask whether this equipotential is almost round or not. Round here is subjective, so you better decide where big starts. Peeking at Figure 1, we can accept that $R = 2$ *is not a bad choice.*

If all roots of a polynotmial lay in a "big equipotential", then they belong to the "same circle", so all have more or less the same absolute value. Since for a monic polynomial the product of all roots is $(-1)^n p(0)$*, we conclude that the possible value for the equipotential is* $|p(0)|^{1/n}$*.*

Returning to our example, the possible value for the equipotential turns out to be $\sqrt[4]{4.7907} = 1.479$*. However, this is not what we agreed to call small! In some sense —by contradiction if you wish— we have convinced ourselves that the roots stand in a small equipotential. But small equipotentials are not too far from the Mandelbrot set, so at the end we can apply the same method as in Section 7: we shift* 3 *units the polynomial, in order to work with*

$$\hat{p}(x) = p(x + 3) = x^4 + 14.1524x^3 + 75.8126x^2 + 188.782x + 180.3567$$

instead.

Table 9.2: Iterations for $\hat{p}$, the shifted polynomial.

$b_{20} = 0$	$d_{20} = 0$
$b_{21} = 1$	$d_{21} = 1$
$b_{22} = 1$	$d_{22} = 1$
$b_{30} = 14.8412412363$	$d_{30} = -180.3567$
$b_{31} = 15.4522563444$	$d_{31} = -187.782$
$b_{32} = 6.1561995984$	$d_{32} = -74.8126$
$b_{33} = 1.0$	$d_{33} = -12.1524$
$b_{40} = 32.5489481287$	$d_{40} = -474370.01130034815$
$b_{41} = 28.3951821834$	$d_{41} = -413832.81696078187$
$b_{42} = 8.6566754358$	$d_{42} = -126162.8243123506$
$b_{43} = 1.0$	$d_{43} = -14574.05042477287$
$b_{50} = 31.9639949993$	$d_{50} = -203288663938.84018$
$b_{51} = 27.7982865423$	$d_{51} = -176795063667.73987$
$b_{52} = 8.5132997602$	$d_{52} = -54143962105.17086$
$b_{53} = 1.0$	$d_{53} = -6359926659.462562$
$b_{60} = 31.9811487913$	$d_{60} = -4.317698063507066 * 10^{22}$
$b_{61} = 27.8041858956$	$d_{61} = -3.7537763381210383 * 10^{22}$
$b_{62} = 8.5129314405$	$d_{62} = -1.149310421433755 * 10^{22}$
$b_{63} = 1.0$	$d_{63} = -1.3500759749682811 * 10^{21}$
$b_{70} = 31.9811628657$	$d_{70} = -1.948893851813995 * 10^{45}$
$b_{71} = 27.8041964056$	$d_{71} = -1.6943545066541352 * 10^{45}$
$b_{72} = 8.5129333941$	$d_{72} = -5.1876798921494836 * 10^{44}$
$b_{73} = 1.0$	$d_{73} = -1.948893851813995 * 10^{43}$

After some toiling (compare Table 9.2), we find that the roots of the original polynomial are (from left to right) -2.6395, $\pm 1.9304i$, 0.4870 .

But what if the shared equipotential is big, say $R \geq 2$? Then we can rescale by means of the change of variables $X = 1.33x/R$ (for example) together with which the polynomial

$$P(X) = \left(\frac{1.33}{R}\right)^n p(RX/1.33)$$

has all roots with absolute value close to $R' = 1.33$, thus small. Since the actual (Mandelbrot) equipotential por $R = 1.33$ is rather irregular to the eye (as opposed to a round circle), we can apply our standard method to this new polynomial with an almost certainty that the algorithm will work.

Remark 9.2 (A note on complex conjugate roots). *When we deal with real polynomials, more than often we finish up with complex conjugate roots. Since the Mandelbrot set is symmetric by complex conjugation, a pair of complex conjugated roots share equipotential.*

Consider $p(x) = x^3 + 2x^2 + 2x + 1$, and let us see how the iterations behave (compare Table 9.3). The quantities involved in the iterations take longer to start growing. This is so because the roots are close to the Mandelbrot set, but fear not, we can increase the number of iterations.

Table 9.3: Iterations for $p(x) = x^3 + 2x^2 + 2x + 1$.

$b_{30} = 0$ $b_{31} = 0$ $b_{32} = 1$	$d_{30} = 0$ $d_{31} = 0$ $d_{32} = -1$
$b_{40} = 1$ $b_{41} = 2$ $b_{42} = 1$	$d_{40} = 2$ $d_{41} = 4$ $d_{42} = 2$
$b_{50} = -4$ $b_{51} = -3$ $b_{52} = 0$	$d_{50} = -4$ $d_{51} = -3$ $d_{52} = 0$
$b_{60} = 1.77777$ $b_{61} = 2.777777$ $b_{62} = 1$	$d_{60} = 16$ $d_{61} = 25$ $d_{62} = 9$
$\vdots$	$\vdots$
$b_{90} = -1.1235$ $b_{91} = -0.1235$ $b_{92} = 1$	$d_{90} = 853764348$ $d_{91} = 93916029$ $d_{92} = -759848320$
$b_{10,0} = 13.3715$ $b_{10,1} = 14.3715$ $b_{10,2} = 1$	$d_{10,0} = 2.0263 * 10^{18}$ $d_{10,1} = 2.1779 * 10^{18}$ $d_{10,2} = 1.5154 * 10^{17}$

The iterations do not stabilize, even though the terms get wild. This is reason enough to suspect that there are multiple roots in the leading equipotential, since otherwise they should have already stabilized according to Lemma 6.1.

As discussed earlier in this section, we must take the greatest common divisor between the last iteration and the original polynomial:

$$\begin{aligned}\gcd(x^3+2x^2+2x+1, x^2+14.3715x+13.3715) &= \\ &= \gcd(x^2+14.3715x+13.3715, x^2+0.9191x-0.0808) \\ &= \gcd(x^2+0.9191x-0.0808, x+1) \\ &= \gcd(x+1, x+1).\end{aligned}$$

As we can see, the polynomial associated with the roots off the leading equipotential is $x+1$, the greatest common divisor between a big iteration and the original polynomial.

We also learn that $p(x)/(x+1) = x^2+x+1$ is the polynomial tied to the biggest equipotential. This polynomial happens to have two complex conjugated roots (as discussed before). We now can easily get them by elementary methods.

10. The complete algorithm at work

Let us consider

$$p(x) = x^5 - 14.8592x^4 + 135.3351x^3 - 651.1201x^2 - 411.0283x + 13.8152.$$

As we have done previously, we start iterating to see in which category it falls.

As there is growth but not convergence, there are, one way or the other, multiple leading roots. According to the algorithm, we work through the greatest common divisor first:

$$\begin{aligned}&\gcd(x^5 - 14.8592x^4 + 135.3351x^3 - 651.1201x^2 - 411.0283x + 13.8152, \\ &\qquad x^4 + 2.7623x^3 - 92.4231x^2 + 52.2895x + 1.7678) \\ &= \gcd(x^4 + 2.7623x^3 - 92.4231x^2 + 52.2895x + 1.7678, \\ &\qquad x^3 - 8.0578x^2 - 4.8265x + 0.1626) \\ &= x^3 - 8.0212x^2 - 4.8060736x + 0.161975296\end{aligned}$$

Hence, the true horizon polynomial, given by this greatest common divisor, permits us recover the leading roots as those of $x^2 - 6.838x + 85.29223364$, the quotient. You are invited to check that $3.419 \pm 8.5792i$ are in fact roots of p.

Table 10.1: For $x^5 - 14.85x^4 + 135.33x^3 - 651.12x^2 - 411.02x + 13.81$.

$b_{40} = 1.8657$	$d_{40} = -4794.2190$
$b_{41} = -54.6984$	$d_{41} = 140553.0443$
$b_{42} = -111.9700$	$d_{42} = 287717.9737$
$b_{43} = -22.9715$	$d_{42} = 59027.7072$
$b_{44} = 1$	$d_{42} = -2569.5982$
$b_{50} = -0.6523$	$d_{50} = -4794.2190$
$b_{51} = 19.4435$	$d_{51} = 15039216632779.37$
$b_{52} = 29.7448$	$d_{52} = 23007106140498.695$
$b_{53} = -8.0276$	$d_{53} = -6209252417808.318$
$b_{54} = 1$	$d_{54} = 773482515899.5808$
$b_{60} = 0.2500$	$d_{60} = 4.8959 * 10^{26}$
$b_{61} = -7.3211$	$d_{61} = -1.4332 * 10^{28}$
$b_{62} = -15.3365$	$d_{62} = -3.0024 * 10^{28}$
$b_{63} = -3.1546$	$d_{63} = -6.1758 * 10^{27}$
$b_{64} = 1$	$d_{64} = 1.9577 * 10^{27}$
$b_{70} = -4.6913$	$d_{70} = 4.5268 * 10^{58}$
$b_{71} = 139.4187$	$d_{71} = -1.3452 * 10^{60}$
$b_{72} = 225.8715$	$d_{72} = -2.1794 * 10^{60}$
$b_{73} = -39.9329$	$d_{73} = 3.8532 * 10^{59}$
$b_{74} = 1$	$d_{74} = -9.6492 * 10^{57}$
$b_{80} = 1.7678$	$d_{80} = -7.3706 * 10^{120}$
$b_{81} = 52.2895$	$d_{81} = 2.1801 * 10^{122}$
$b_{82} = -92.4231$	$d_{82} = 3.8534 * 10^{122}$
$b_{83} = 2.7623$	$d_{83} = -1.1517 * 10^{121}$
$b_{84} = 1$	$d_{84} = -4.1693 * 10^{120}$

Now we should continue our discussion with $x^3-8.0212x^2-4.8060736x+0.161975296$. Of course, we iterate it.

Table 10.2: The process for $p(x) = x^3 - 8.02x^2 - 4.80x + 0.16$.

$b_{30} = -0.0188$ $b_{31} = 0.5685$ $b_{32} = 1$	$d_{30} = -1.6231$ $d_{31} = 49.0006$ $d_{32} = 86.1881$
$b_{40} = -0.0188$ $b_{41} = 0.5580$ $b_{42} = 1$	$d_{40} = -11016.7342$ $d_{41} = 325601.5213$ $d_{42} = 583514.2898$
$b_{50} = -0.0188$ $b_{51} = 0.5580$ $b_{52} = 1$	$d_{50} = -503802575865.4391$ $d_{51} = 14889927824837.625$ $d_{52} = 26684458467449.418$

We see that already at the fifth iteration we get convergence, so we have here one leading root: namely $0.558 + 8.0212 = 8.5792$.

The roots of the leftover polynomial $x^2 + 0.558x - 0.0188$ can be easily found, either by this algorithm or any other.

11. Final remarks

All our ideas relay on the fact that the complement of the Mandelbrot set has level sets that can be (rigorously) approximated dynamically. However, you can try your luck with any other similar process, either in the parameter space (like ours along this work) or in the dynamical space.

Fix for instance $C \in \mathbf{C}$. Then in the iteration process $z \mapsto z^2 + C$, for some *seeds* z_0 the succesive iterates

$$z_0 \mapsto z_1 = z_0^2 + C \mapsto z_2 \mapsto \dots$$

remain bounded, while for some they will be attracted to ∞. Again here you can play with the fact that some are attracted faster to ∞ than others (if attracted at all).

For instance we can take $z \mapsto z^2$, who has circular equipotentials and is easy to implement, for the polynomial

$$x^4 + 2.1524x^3 + 2.4410x^2 + 8.0212x - 4.7907,$$

an old friend from Example 9.1. (Recall that for this polynomial each root is *as far from the Mandelbrot set as the others are*, when measured

Table 11.1: Squaring $p(x) = x^4 + 2.15x^3 + 2.4x^2 + 8.02x - 4.79$.

$b_{20} = 0$	$d_{20} = 0$
$b_{21} = 0$	$d_{21} = 0$
$b_{22} = 1$	$d_{22} = 1$
$b_{30} = -2.2257483178$	$d_{30} = 4.7907346029$
$b_{31} = 3.7266103474$	$d_{31} = -8.0212129108$
$b_{32} = 1.1341040537$	$d_{32} = -2.4410628507$
$b_{33} = 1$	$d_{33} = -2.1524152415$
$b_{40} = -1.9588963031$	$d_{40} = 157.21097271448988$
$b_{41} = 3.7266103474$	$d_{41} = -299.0786376473024$
$b_{42} = 0.1192166276$	$d_{42} = -9.567715225644205$
$b_{43} = 1$	$d_{43} = -80.25487232886249$
$b_{50} = -1.8283009781$	$d_{50} = 306929.39197400087$
$b_{51} = 3.7266103474$	$d_{51} = -625611.571489818$
$b_{52} = -0.4311493973$	$d_{52} = 72379.99867829758$
$b_{53} = 1$	$d_{53} = -167876.84066154412$
$b_{60} = -1.8151306293$	$d_{60} = 1672127163290.7983$
$b_{61} = 3.7266103473$	$d_{61} = -3433012637344.8604$
$b_{62} = -0.4866697847$	$d_{62} = 448327934906.6706$
$b_{63} = 1$	$d_{63} = -921215881823.9551$
$b_{70} = -1.8151306292$	$d_{70} = 5.148943959513387 * 10^{25}$
$b_{71} = 3.7266103473$	$d_{71} = -1.0571722902933055 * 10^{26}$
$b_{72} = -0.4870487048$	$d_{72} = 1.3816694766192075 * 10^{25}$
$b_{73} = 1$	$d_{73} = -2.836820036863101 * 10^{25}$

with respect to the canonical equipotentials of the complement of $\mathcal{M}$.) The squaring process now takes advantage that root with biggest norm is attracted faster to ∞ than the rest.

From the data on Table 11.1 we can read that one root is $-0.48705 - 2.1524 = -2.63945$ like before. For the others, help yourself.

References

[1] L. Carleson and T. Gamelin, *Complex Dynamics*. Springer, 2013.

[2] S. Lang, *Complex Analysis*. Addison-Wesley, 1977.

[3] J. Milnor, *Dynamics in One Complex Variable: Introductory Lectures*. Vieweg, 2000.

[4] A. Poirier, "Approximating square roots". *ProMathematica*, vol. 9, no. 17-18, pp. 95-98, 1995. [On line]. Available: https://bit.ly/3l1rCIE

[5] J. Torres, "Approximating roots of polynomials", Tesis de Licenciatura. Pontificia Universidad Católica del Perú, 2021.

[6] L. R. Turner, *Inverse of the Vandermonde matrix with applications*. National Aeronautics and Space Administration, 1966. [On line]. Available: https://go.nasa.gov/3mDqlIn

Alfredo Poirier
Departamento de Ciencias
Sección Matemáticas
Pontificia Universidad Católica del Perú
Perú
e-mail: apoirie@pucp.edu.pe
Corresponding author

and

Jesús Torres
Facultad de Ciencias e Ingeniería
Pontificia Universidad Católica del Perú
Perú
e-mail: jstorres@pucp.pe