Journal of Universal Computer Science, vol. 26, no. 10 (2020), 1312-1330
submitted: 27/7/20, accepted: 27/10/20, appeared: 28/10/20

Interval Methods for Fixed and Periodic Points: Development and Visualization

José Eduardo de Almeida Ayres
(IMPA, Rio de Janeiro, Brazil
jeaayres@impa.br)

Luiz Henrique de Figueiredo
(IMPA, Rio de Janeiro, Brazil
lhf@impa.br)

Abstract: We describe the development of rigorous numerical methods based on interval analysis for finding all fixed points of a map and all attracting periodic points of a complex polynomial. We also discuss their performance with instructive visualizations.
Key Words: fixed points, interval analysis, computer-assisted proofs
Category: G.1.5, G.1.2

1 Introduction

Finding the fixed points of a function is important in many contexts. For instance, solving nonlinear equations is frequently cast as finding fixed points. Newton's method is the main example of this formulation. Fixed points and periodic points are also important in discrete dynamical systems, especially in complex dynamics, where periodic orbits play a key role [Branner 1989; Keen 1989].

In this paper, we describe the development of rigorous numerical methods for finding all fixed points of a map. The methods are based on interval analysis [Moore 1966, 1979; Moore et al. 2009; Tucker 2011]. There is a large literature on interval methods for solving nonlinear equations [Baker Kearfott 1996], but surprisingly very little that is specific to fixed points. We know only these papers: [Caprani and Madsen 1975, 1978; Rall 1982, 1987; Rihm 2001]. Although the two problems are mathematically equivalent, there are algorithmic opportunities to be exploited for computing fixed points. Our main contribution is a hybrid algorithm that switches to plain fixed-point iteration once it establishes the existence of an attracting fixed point using Banach's criterion before finding a small certified enclosure for the fixed point. After recalling the main definitions and facts about fixed points in §2, we briefly describe interval analysis in §3. Then in §4 we describe a series of interval algorithms for finding all fixed points of a map, culminating with our hybrid algorithm and its specialization for finding all attracting periodic points of a complex polynomial described in §5. Finally, in §6 we present results illustrating the performance of these algorithms, including novel, instructive visualizations of the progress of the algorithms.

2 Fixed points

Let $f\colon \Omega \subseteq \mathbf{R}^d \to \mathbf{R}^d$ be a continuous function defined on a box Ω (that is, a product of compact intervals). A *fixed point* of f is a point $x^* \in \Omega$ such that $f(x^*) = x^*$. A fixed point x^* is *attracting* if $\|f(x) - x^*\| < \|x - x^*\|$ for all points $x \neq x^*$ in a neighborhood of x^*. A fixed point x^* is *repelling* if $\|f(x) - x^*\| > \|x - x^*\|$ for all points $x \neq x^*$ in a neighborhood of x^*. For differentiable functions f, a fixed point x^* is attracting iff $\|f'(x^*)\| < 1$ and repelling iff $\|f'(x^*)\| > 1$. When $\|f'(x^*)\| = 1$, the fixed point x^* is *indifferent* or *neutral.* Here, $\|f'(x^*)\|$ is the norm of the Jacobian matrix of f at x^*.

We shall use only the classical results on fixed points, which we discuss below. A wealth of results on fixed points can be found in the books [Agarwal et al. 2001] and [Berinde 2007], among many others.

The most popular numerical method for finding a fixed point of f is *fixed-point iteration* (also known as Picard iteration):

$$x_{n+1} = f(x_n), \quad x_0 \in \Omega$$

When this sequence converges, its limit is a fixed point of f, because f is continuous. Convergence typically depends on choosing the initial point x_0 sufficiently close to a fixed point of f, which may not be easy to find. Fixed-point iteration is mostly suitable for finding attracting fixed points. Repelling fixed points cannot be found using fixed-point iteration, unless it starts at the fixed point itself. In this sense, repelling fixed points cannot be directly observed. Indifferent fixed points can sometimes be found using fixed-point iteration, but not from all nearby initial points. We shall see that interval methods can find *all* fixed points: attracting, repelling, and indifferent.

Brouwer's fixed-point theorem guarantees the existence (but not uniqueness) of fixed points when $f\colon K \to K$ is a map on a convex compact set $K \subseteq \mathbf{R}^d$, such as a box. It is the staple existence theorem, due to its fairly general hypotheses and immediate generalizations ("convex" can be replaced by "homeomorphic to a ball"). Brouwer's fixed-point theorem is stated as an existence result, but there are constructive formulations based on Sperner's lemma that can be used algorithmically [Scarf 1967; Todd 1976].

Banach's fixed-point theorem guarantees the existence and uniqueness of fixed points when $f\colon K \to K$ is a contraction map on a compact set $K \subseteq \mathbf{R}^d$. In this case, f has a unique fixed point $x^* \in K$ and fixed-point iteration converges to x^* for every initial point $x_0 \in K$. This important theorem (stated in its full generality for complete metric spaces) is crucially used in many proofs and in many numerical methods. Although apparently a global result, it is typically used locally, near a fixed point, because f need not be a global contraction. Indeed, Newton's method for finding zeros of a function is an instance of fixed-point iteration that converges provided the initial point is sufficiently near a fixed point (that is, a zero of the function). However, the global convergence properties of Newton's method are complicated, even for finding zeros of polynomials [von Haeseler and Peitgen 1988; Hubbard et al. 2001].

We shall see presently that the hypotheses of these classical theorems can be checked rigorously in a computer using interval analysis. A key point in this verification is finding an explicit box $X \subseteq \Omega$ such that $f(X) \subseteq X$. This guarantees the existence of fixed points of f in X by Brouwer's fixed-point theorem.

3 Interval analysis

Interval analysis is the main tool for rigorous numerical computation [Tucker 2011]. It is based on interval arithmetic, an extension of ordinary arithmetic operations and standard elementary functions to intervals [Moore 1966, 1979; Moore et al. 2009]. The basic fact in interval analysis is that for each function $f\colon \Omega \subseteq \mathbf{R}^d \to \mathbf{R}$ expressed by a formula or an algorithm, there is a computable function F automatically built from the expression of f, called the *natural interval extension* of f, such that $F(X)$ is an interval that estimates the whole range of values taken by f on a box $X \subseteq \Omega$:

$$F(X) \supseteq f(X) = \{f(x) : x \in X\}$$

Finding the exact range $f(X)$ is a hard problem in general [Traylor and Kreinovich 1995]. Therefore, the inclusion $F(X) \supseteq f(X)$ is usually proper and interval estimates are usually overestimates. Nevertheless, the estimates $F(X)$ get better as X shrinks to a point in the sense that $F(\{x\}) = \{f(x)\}$ for every $x \in \Omega$. More precisely, we have at least linear convergence for interval estimates: $\operatorname{diam}(F(X)) \le c \operatorname{diam}(X)$ for some $c > 0$ that depends only on f. Thus, as we shall see in §4, interval methods are typically divide-and-conquer methods that recursively explore the domain of f, getting better information about f as they refine the subdivision, and discarding boxes that cannot contain a solution. For instance, when finding the zeros of f in Ω, we can discard a box X whenever $0 \notin F(X)$. This is a computational proof that f has no zeros in X. However, because of overestimation, we cannot conclude that f has a zero in X when $0 \in F(X)$. In this case, we subdivide X and recursively test the pieces.

We extend interval estimates to functions $f\colon \Omega \subseteq \mathbf{R}^d \to \mathbf{R}^m$ by combining interval estimates for each component of f. More precisely, if $f = (f_1, \ldots, f_m)$ and F_i is an interval extension of $f_i\colon \Omega \subseteq \mathbf{R}^d \to \mathbf{R}$, then, for each box $X \subseteq \Omega$, the box $F(X) = F_1(X) \times \cdots \times F_m(X) \supseteq f(X)$ estimates the range of f on X.

Automatic differentiation [Moore 1966; Rall 1986; Moore et al. 2009; Tucker 2011] is the perfect companion for interval arithmetic and works in a similar fashion. It automatically converts an expression for f into an algorithm that simultaneously computes the value of f and of all its partial derivatives. When fed intervals instead of numbers, this algorithm computes interval estimates for the value of f and of all its partial derivatives. This allows us to reason reliably about both the range of values of f and its regions of monotonicity.

Interval arithmetic and automatic differentiation allow us to check the hypotheses of the fixed-point theorems rigorously in a computer. The existence of fixed points in a box X guaranteed by Brouwer's theorem follows whenever $F(X) \subseteq X$ because then $f(X) \subseteq F(X)$ implies $f(X) \subseteq X$. The existence of a unique fixed point in a box X guaranteed by Banach's theorem follows whenever $F(X) \subseteq X$ and $\|F'(X)\| < 1$ because these imply that f is a contraction in X, thanks to the mean value inequality. Here, F' is an interval extension of the Jacobian matrix of f, which can be computed with automatic differentiation.

4 Finding fixed points

We shall now describe a series of four incrementally refined interval algorithms for finding *all* fixed points of f in Ω. The series culminates with our hybrid algorithm, Algorithm 4.

0. The starting point is Algorithm 0, the standard divide-and-conquer interval method that performs adaptive subdivision of Ω to find all zeros of a function $g\colon \Omega \subseteq \mathbf{R}^d \to \mathbf{R}^d$ [Moore 1966].

Algorithm 0

```
procedure Explore(X)
  if 0 ∉ G(X) then
    discard X
  else if diam(X) < ε then
    accept X
  else
    SubExplore(X)
  end
end
```

```
procedure SubExplore(X)
  divide X into smaller subboxes
    X_1, ..., X_m
  for j = 1, ..., m do
    Explore(X_j)
  end
end
```

Given an interval extension G for g, we recursively explore Ω, discarding every box X that cannot possibly contain zeros of g, as proved by $0 \notin G(X)$. Otherwise, X *may* contain zeros of g. If X is small enough for our purposes, we accept X as containing a zero of g. Then the midpoint of X is an approximation for a zero of g within the given tolerance ε. Otherwise, we subdivide X into smaller subboxes and explore them as above to isolate the zeros of g. In low dimension (that is, $d \leq 3$), we typically split X at its center into 2^d subboxes, leading to quadtrees and octrees [Samet 1984]. In higher dimension, to avoid creating a huge number of subboxes, we typically split X at its center across its longest side into two subboxes, leading to a bintree [Samet and Tamminen 1985]. This strategy is also frequently used in low dimension for boxes with high aspect ratio. More sophisticated subdivision strategies exist [Baker Kearfott 1996, §4.3]. As mentioned in §3, the basis for this method is that interval estimates get better as the boxes reduce in size, converging to the actual function value when the boxes shrink to a point.

To find the fixed points of f in Ω, we use this method with $g(x) = f(x) - x$ and its interval extension $G(X) = F(X) - X$. Then the zeros of g in Ω are exactly the fixed points of f in Ω. The union of the accepted boxes contains all fixed points of f in Ω.

1. Algorithm 0 can be cast in the context of fixed points by noting that all fixed points of f in X must lie in $X \cap F(X)$. Thus, we can discard X if $X \cap F(X) = \varnothing$, because then f has no fixed points in X. Algorithm 1 uses this formulation. The test $X \cap F(X) = \varnothing$ is equivalent to the test $0 \notin G(X)$ used in Algorithm 0, but it reads better in the context of fixed points. Brouwer's theorem implies that Algorithm 1 (and its subsequent refinements) can certify the existence of fixed points in X whenever $F(X) \subseteq X$ (code omitted). In this case, X contains at least one fixed point, but may contain more: Brouwer's theorem guarantees existence but not uniqueness. Algorithm 1 subdivides large boxes hoping to *isolate* fixed points within the tolerance ε.

Algorithm 1

```
procedure Explore(X)
  if X ∩ F(X) = ∅ then          [Brouwer certifies existence if F(X) ⊆ X]
    discard X
  else if diam(X) < ε then
    accept X
  else
    SubExplore(X)
  end
end
```

2. The formulation of Algorithm 1 in terms of $X \cap F(X)$ immediately motivates Algorithm 2, which recursively explores $X' = X \cap F(X)$, instead of X. No such reduction is available for Algorithm 0, which solves generic nonlinear equations. This tiny change brings a qualitative improvement: Algorithm 1 is *spatially adaptive* because its search is guided by the *location* of the fixed points of f. Algorithm 2 is also *analytically adaptive* because its search is also guided by the *nature* of the fixed points of f. Indeed, near an attracting fixed point $x^* \in X$, we typically have $F(X) \subset X$, strictly, and so $X' = F(X) \subset X$, strictly. The stronger the attraction of x^*, the smaller X' is, and the faster Algorithm 2 converges to x^*. Conversely, near a repelling fixed point, we typically have $F(X) \supseteq X$, and so $X' = X$ signals that we need to subdivide X.

Algorithm 2

```
procedure Explore(X)
  X' ← X ∩ F(X)
  if X' = ∅ then
    discard X
  else if diam(X') < ε then
    accept X'
  else
    SubExplore(X')
  end
end
```

Algorithm 3

```
procedure Explore(X)
  X' ← X ∩ F(X)
  if X' = ∅ then
    discard X
  else if diam(X') < ε then
    accept X'
  else if diam(X') < λ diam(X) then
    Explore(X')
  else
    SubExplore(X')
  end
end
```

Algorithm 2 can be seen as an *interval iteration* in the sense of [Rall 1982, 1987]:

$$X_{n+1} = X_n \cap F(X_n), \quad X_0 = \Omega$$

When $X_n = \varnothing$ for some n, the sequence *diverges* and there are no fixed points of f in Ω. Otherwise, (X_n) is a sequence of nested nonempty boxes and so *converges* to the box $X^* = \bigcap_{n=0}^{\infty} X_n \neq \varnothing$, by Cantor's intersection theorem. Moreover, X^* contains *all* fixed points of f in Ω. In the outward-rounded floating-point arithmetic used in interval arithmetic, the sequence converges in finite time, because there are finitely many intervals with floating-point numbers as extremes. Algorithm 2 goes beyond convergence in pure interval iteration in the sense of [Rall 1982, 1987] by subdividing large boxes to isolate fixed points.

3. Recursively exploring $X' = X \cap F(X)$ instead of X in Algorithm 2 is not always a clear advantage. It may happen that X' is not much smaller than X, because we are near a weakly attracting fixed point (or near a repelling fixed point, when $X' = X$). Algorithm 3 compares the diameters of X and X' to ensure that good linear convergence is preserved in these cases, while still taking advantage of strongly attracting fixed points, when X' is much smaller than X. We use $\lambda = \frac{1}{2}$ for comparing diameters, which corresponds to the reduction in quadtree subdivision. More sophisticated tests exist [Hansen and Walster 2003, §11.7]. Algorithm 3 outputs a certified enclosure for each fixed point of f in Ω, attracting, repelling, and indifferent. This is as far as one can go using just the continuity of f.

4. When f is differentiable, we use Algorithm 4, which is our final method: it outputs a small certified enclosure for each attracting fixed point of f in Ω. Algorithm 4 incorporates Banach's criterion for attracting fixed points, as mentioned in §3. As soon as we establish the existence of a unique attracting fixed point in a box, we switch to plain fixed-point Picard iteration, because it is faster than interval iteration. Upon convergence to an approximate fixed point $\hat{x}$, we find a small box X around $\hat{x}$ such that $F(X) \subseteq X$ using interval inflation. (We used $\eta = 10^{-15}$ and $h = 0.25$ in our experiments.) Finally, we refine this box using pure interval iteration to output a certified enclosure for the fixed point x^*.

Interval inflation was proposed by [Caprani and Madsen 1978], outside the context of adaptive subdivision interval methods. They argued that enlarging a box X around an approximate fixed point often increases the chances that $F(X) \subseteq X$. In more general forms, this strategy is known as ε-inflation [Mayer 1995; Rump 1998]; it also helps to avoid clusters of boxes around a single solution [Baker Kearfott 1996, §4.2], a nuisance in subdivision methods. Clustering may affect methods like Algorithm 3 that cannot prove uniqueness of solution in a box, but not Algorithm 4.

Algorithm 4

```
procedure Explore(X)
  X′ ← X ∩ F(X)
  if X′ = ∅ then
    discard X
  else if diam(X′) < ε then
    accept X′
  else if F(X) ⊆ X and ‖F′(X)‖ < 1 then
    ExploreAttracting(X′)          [Banach]
  else if diam(X′) < λ diam(X) then
    Explore(X′)
  else
    SubExplore(X′)
  end
end

procedure ExploreAttracting(X)
  x̂ ← mid(X)
  repeat          [Picard]
    x̂ ← f(x̂)
  until convergence
  X ← [x̂ − η, x̂ + η]
  repeat          [inflation]
    X ← X + diam(X)[−h, h]
  until F(X) ⊆ X
  repeat          [interval iteration]
    X ← F(X)
  until convergence
  accept X
end
```

5 Finding attracting periodic points

A *periodic point* of $f\colon \Omega \subseteq \mathbf{R}^d \to \mathbf{R}^d$ is a point $x^* \in \Omega$ such that $f^n(x^*) = x^*$, where $f^n = f \circ \cdots \circ f$ (n times) is the n-th iterate of f. The *period* of a periodic point x^* is the smallest positive integer n such that $f^n(x^*) = x^*$. Periodic points of period n are thus fixed points of f^n. Conversely, the fixed points of f^n are the periodic points of f with period dividing n. Clearly, the periodic points of period 1 are exactly the fixed points of f. The *orbit* of a point $x_0 \in \Omega$ is the set of the images of x_0 under the iterates of f, that is, $\{x_0, f(x_0), f^2(x_0), \dots\}$. Periodic points are exactly those that have finite orbits; the size of the orbit is the period. A *periodic orbit* is the orbit of a periodic point.

Periodic orbits are important in discrete dynamical systems, because they represent stationary states. Attracting periodic orbits are especially important because they represent the fate of nearby points. Periodic orbits play a key role in the dynamics of complex rational maps [Branner 1989; Keen 1989]. Typically, the basins of attraction of attracting periodic orbits divide the Riemann sphere into regions sharing a common boundary, the Julia set. The Julia set is the closure of the repelling periodic points. When f is a polynomial, the set of points having bounded orbits is called the filled Julia set; its boundary is the Julia set. In particular, all periodic points are in the filled Julia set.

We shall adapt Algorithm 4 to find all attracting periodic points of a complex polynomial $f\colon \mathbf{C} \to \mathbf{C}$. The periodic points of period n are found among the fixed points of f^n, that is, the roots of $f^n(z) = z$. This is a polynomial equation and so can in principle be solved numerically using one of several standard methods. However, if f has degree d, then the equation $f^n(z) = z$ has degree d^n and we do not want to find that many zeros only to choose the few that are attracting periodic points. Moreover, most standard methods need polynomial equations expressed in monomial form, and we would rather not expand $f^n(z)$ into monomial form.

Algorithm 5 is an adaptation of Algorithm 4 to find the fixed points of f^n, trying to discard its repelling fixed points and to detect its attracting fixed points, as soon as possible. Since the repelling fixed points are on the Julia set, we can discard boxes that are away from the Julia set, that is, outside a disk centered at the origin and containing the Julia set. Such a disk is called an *escape disk* because all orbits starting outside it escape to infinity. The radius of an escape disk can be explicitly computed from the coefficients of $f(z) = a_n z^n + a_{n-1} z^{n-1} + \cdots + a_0$ [McClure 2019, §4.1]:

$$R = \max\left(2|a_n|, 2\frac{|a_{n-1}| + \cdots + |a_0|}{|a_n|}\right)$$

This formula generalizes the well-known formula $R = \max(2, |c|)$ for $f(z) = z^2 + c$.

Algorithm 5 computes interval estimates for $f^n(X)$ and $(f^n)'(X)$ iteratively, using the chain rule. As soon as an intermediate estimate $F^k(X)$ is outside the escape disk, we have proved that all orbits starting at X will escape to infinity. In this case, X cannot contain any attracting periodic points of f, because those are in the filled Julia set. This test is not strictly needed, but it helps to reduce the density of the quadtree. Without it, the large overestimation in the interval estimates for $f^n(X)$ due to the wrapping effect [Moore 1979] leads to needless subdivisions. We can definitely discard a box X when $|(f^n)'(X)| \geq 1$, because then any periodic points of period n in X will be repelling or indifferent.

Algorithm 5

procedure *Explore*(X)
  $W, W' \leftarrow X, 1$
  for $k = 1$ to n **do**
    $W, W' \leftarrow F(W), F'(W)W'$
    if W is outside the escape disk **then**
      discard X
    end
  end
  $X' \leftarrow X \cap W$
  if $X' = \varnothing$ or $\|W'\| \geq 1$ **then**
    discard X
  else if $\mathrm{diam}(X') < \varepsilon$ **then**
    accept X'
  else if $W \subseteq X$ and $\|W'\| < 1$ **then**
    ExploreAttracting(X')
  else if $\mathrm{diam}(X') < \lambda\,\mathrm{diam}(X)$ **then**
    Explore(X')
  else
    SubExplore(X')
  end
end

6 Numerical experiments

We now present the results of some numerical experiments that illustrate the performance of the algorithms discussed above. We find periodic points of $f(z) = z^2 + c$ for several $c \in \mathbf{C}$ and several periods (see Table 1). The fourth column gives $\mu = |(f^n)'(x^*)|$, which measures the nature of the periodic point x^*: *super attracting* when $\mu = 0$, *strongly attracting* when $\mu < 1$ and is close to 0, and *weakly attracting* when $\mu < 1$ but is close to 1. The domain is $\Omega = [-2.01, 1.99] \times [-1.99, 2.01] \subseteq \mathbf{R}^2 \cong \mathbf{C}$. The tolerance used for termination is $\varepsilon = 10^{-12}$.

Pictures of the subdivision behavior of the algorithms can only show the early steps, due to limited spatial resolution. The interesting behavior happens in very small boxes. Therefore, we focus on how the algorithms find a single periodic point. The graphs shown below describe the main steps taken by an algorithm until it accepts a box containing a certified enclosure for a periodic point. The horizontal axis shows the sequence of steps. The vertical axis shows the decimal

Figure	period	x^*	μ	c
2a	3	0	0	$-0.122561166876654 + 0.744861766619744\,i$
2b	1	$-0.15 - 0.11\,i$	0.38	$-0.16 - 0.15\,i$
2c	2	0.17	0.80	-1.2
3a	8	-1.10	0	-1.3815474844320614695
3b	4	-1.29	0.18	1.3
3c	8	-1.35	0.91	-1.393
4a	5	-0.79	0	$-0.504340175446244 + 0.562765761452981\,i$
4b	1	$-0.15 - 0.11\,i$	0.38	$-0.16 - 0.15\,i$
4c	8	-0.19	0.91	-1.393
5a	8	-1.10	0	-1.381547484432061
5b	3	$-0.07 + 0.011\,i$	0.35	$-0.10 + 0.72\,i$
5c	2	-1.17	0.80	-1.2
6	1	$-0.15 - 0.11\,i$	0.38	$-0.16 - 0.15\,i$
7	3	$-0.0099 + 0.0052\,i$	0.058	$-0.12 + 0.74\,i$
8	4	0.38	0.18	-1.3

Table 1: Data for our numerical experiments.

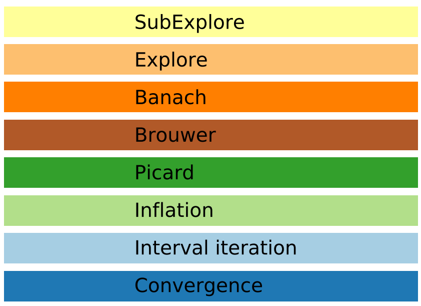
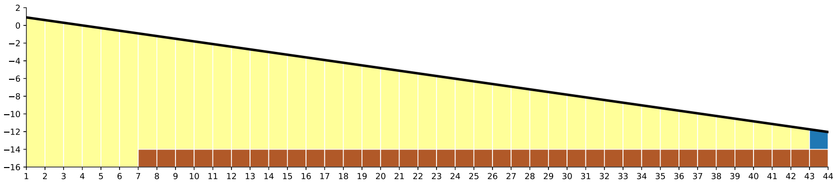


Figure 1: The meaning of the color bars in the graphs.

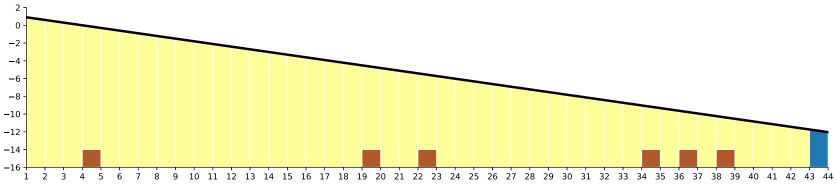
logarithm of the diameter of the current enclosure for the point; it indicates the number of correct decimal places in the current approximation. The bars are colored according to the legend in Figure 1. The certification of existence of fixed points via Brouwer's theorem is marked with a small brown box, when possible.

6.1 Individual performance

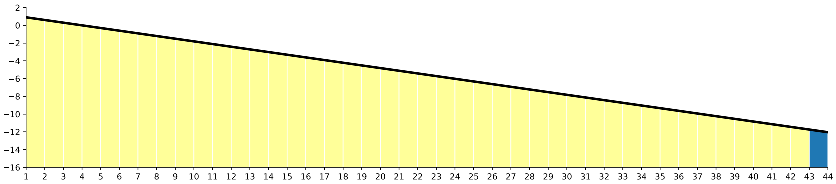
1. Figure 2a shows the steps taken by Algorithm 1 for finding a super attracting periodic point of period 3 for $c \approx -0.12 + 0.74\,i$. Note the constant decrease in diameter, due solely to subdivision; Algorithm 1 cannot exploit the nature of this point. Nevertheless, Algorithm 1 proves early on that there is a fixed point in a certain box using Brouwer's theorem. This certification persists until convergence. Figure 2b shows the steps taken by Algorithm 1 for finding a strongly attracting fixed point for $c = -0.16 - 0.15\,i$. Again, Algorithm 1 cannot exploit the nature of this point. Moreover, certification via Brouwer's theorem only happens for a



(a) super attracting point



(b) strongly attracting point

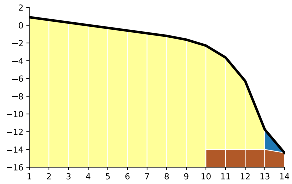


(c) weakly attracting point

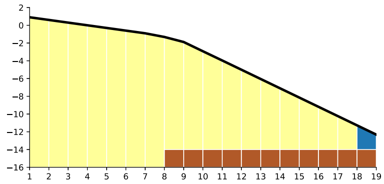
Figure 2: Performance of Algorithm 1.

few boxes. Figure 2c shows the steps taken by Algorithm 1 for finding a weakly attracting fixed point for $c = -1.2$. As a consequence, Algorithm 1 cannot certify any boxes via Brouwer's theorem. Figure 2 is typical of the behavior of Algorithm 1 for finding an attracting point. It subdivides steadily at a fixed rate but it cannot exploit the nature of the point to go any faster.

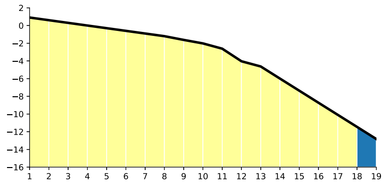
2. Figure 3a shows the steps taken by Algorithm 2 for finding a super attracting periodic point of period 8 for $c \approx -1.38$. Note the sharp decrease in diameter after step 10, when it also started certifying boxes using Brouwer's theorem. This reflects the nature of this point, because X' is much smaller than X. Figure 3b shows the steps taken by Algorithm 2 for finding a strongly attracting periodic point of period 4 for $c = -0.16 - 0.15\,i$. The nature of this point is reflected in the decrease in diameter after step 8, when it also started certifying boxes using Brouwer's theorem. The change of slope reflects the magnitude of μ. Figure 3c shows the steps taken by Algorithm 2 for finding a weakly attracting periodic point of period 8 for $c = -1.393$. As a consequence, Algorithm 2 cannot certify any boxes via Brouwer's theorem, but there is still a decrease in diameter after step 13. Again, the change of slope reflects the magnitude of μ. Figure 3 is typical



(a) super attracting point



(b) strongly attracting point



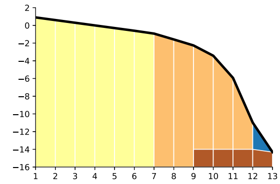
(c) weakly attracting point

Figure 3: Performance of Algorithm 2.

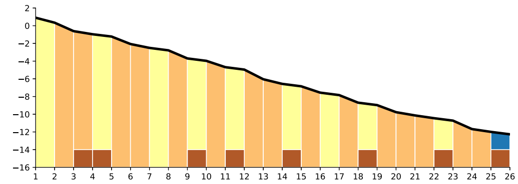
of the behavior of Algorithm 2 for finding an attracting point. It subdivides steadily at a rate that changes near the point to reflect the magnitude of μ.

3. Figure 4a shows the steps taken by Algorithm 3 for finding a super attracting periodic point of period 5 for $c \approx -0.50 + 0.56\,i$. Note that the switch to exploring X' after step 7 and the sharp decrease in diameter after step 9, when it also started certifying boxes using Brouwer's theorem. This reflects the nature of this point. Figure 4b shows the steps taken by Algorithm 3 for finding a strongly attracting fixed point for $c = -0.16 - 0.15\,i$. The nature of this point is reflected in the alternation of exploring and subdividing X', because μ is close to 0.5. Figure 4c shows the steps taken by Algorithm 3 for finding a weakly attracting periodic point of period 8 for $c = -1.393$. As a consequence, it needs to subdivide X' more often. Figure 4 is typical of the behavior of Algorithm 3 for finding an attracting point. It alternates between exploring and subdividing X'. The number of consecutive explore steps reflects the magnitude of μ.

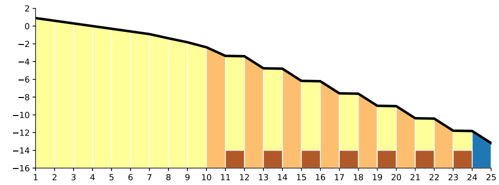
4. Figure 5a shows the steps taken by Algorithm 4 for finding a super attracting periodic point of period 8 for $c \approx -1.38$. Note that it certified existence and



(a) super attracting point



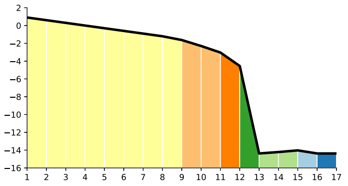
(b) strongly attracting point



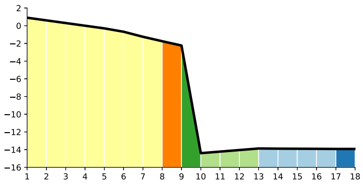
(c) weakly attracting point

Figure 4: Performance of Algorithm 3.

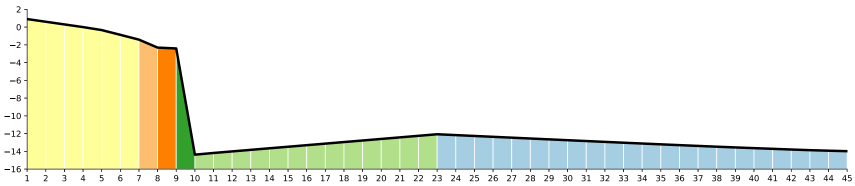
uniqueness using Banach's theorem at step 11 and so switched to Picard iteration at step 12. Inflation and refinement took just a couple of steps. This reflects the nature of this point. Figure 5b shows the steps taken by Algorithm 4 for finding a strongly attracting periodic point of period 3 for $c = -0.10 + 0.72\,i$. It switched to Picard iteration at step 9. Inflation and refinement took a few steps. This reflects the nature of this point. Figure 5c shows the steps taken by Algorithm 4 for finding a weakly attracting periodic point of period 2 for $c = -1.2$. As a consequence, inflation took several steps. Refinement did not succeed, but the output box was close to the tolerance. This reflects the nature of this point and the magnitude of $\mu = 0.8$. Figure 5 is typical of the behavior of Algorithm 4 for finding an attracting point. It subdivides until Banach's certification holds and then quickly finds a certified small box around the point using inflation. The number of steps in inflation and refinement depends on the nature of the point. In the hybrid approach of Algorithm 4, plain fixed-point iteration replaces many subdivisions steps in previous algorithms.



(a) super attracting point



(b) strongly attracting point



(c) weakly attracting point

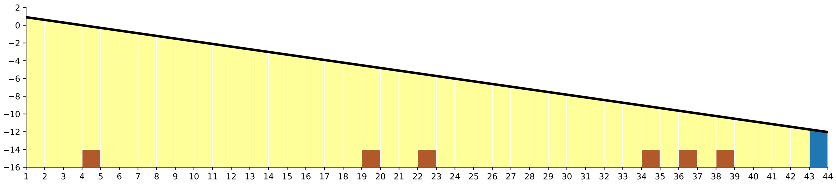
Figure 5: Performance of Algorithm 4.

6.2 Comparative performance

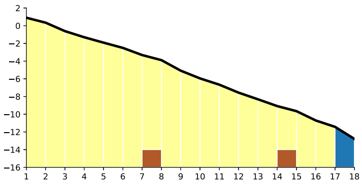
Figure 6 compares the performances of Algorithm 1 and Algorithm 2 for finding a strongly attracting fixed point for $c = -0.16 - 0.15\,i$. The main difference is the rate of diameter decrease, shown by the slope of the graph. As expected, near a strongly attracting point, X' is much smaller than X.

Figure 7 compares the performances of Algorithm 2 and Algorithm 3 for finding a strongly attracting periodic point of period 3 for $c = -0.12 + 0.74\,i$. Although Algorithm 3 takes slightly longer to find this point, it is overall faster. Algorithm 2 visits 1553 boxes to maximum depth 15 whereas Algorithm 3 visits 1389 boxes to maximum depth 18.

Figure 8 compares the performances of Algorithm 3 and Algorithm 4 for finding a strongly attracting periodic point of period 4 for $c = -1.3$. Algorithm 3 steadily approximates the point, and certifies existence using Brouwer's theorem from step 7 onward. Algorithm 4 fares much better. It certifies both existence and uniqueness using Banach's theorem at the same step as Algorithm 3 certifies existence only. From then on, it quickly finds a small certified enclosure for the point.

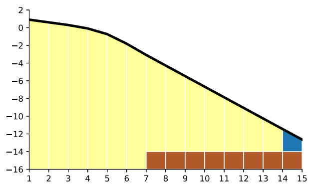


(a) Algorithm 1

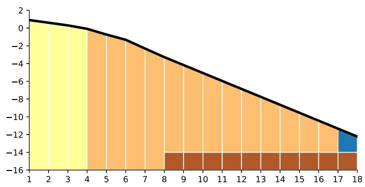


(b) Algorithm 2

Figure 6: Performance of Algorithm 1 vs. Algorithm 2.



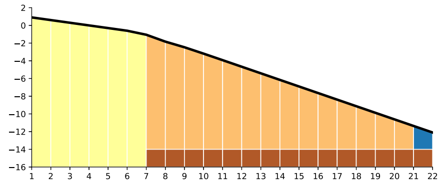
(a) Algorithm 2



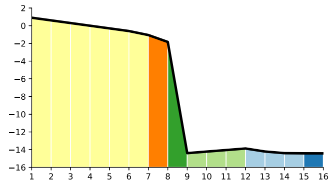
(b) Algorithm 3

Figure 7: Performance of Algorithm 2 vs. Algorithm 3.

Figure 9 compares the domain subdivisions performed by Algorithm 1 and Algorithm 2 for finding all fixed points for $c = -0.16 - 0.15\,i$. The maximum depth is 5, low on purpose for illustration. There is a strongly attracting fixed point on the left of the domain and a repelling fixed point on the right. This is reflected in the decomposition of Algorithm 2, which approaches the fixed points much faster than Algorithm 1 does, even at that low depth. The nature of the repelling fixed point is captured by the cluster of accepted boxes around it; such a cluster appears at all depths.



(a) Algorithm 3



(b) Algorithm 4

Figure 8: Performance of Algorithm 3 vs. Algorithm 4.

Figure 10 compares the domain subdivisions performed by Algorithm 4 and Algorithm 5 for finding all periodic points of period 6 for $c = -0.60 - 0.66\,i$. The maximum depth is 17. The subdivision performed by Algorithm 4 is everywhere dense, because of the repelling points on the Julia set, resulting in a large number of boxes accepted at that depth (1073 boxes corresponding to repelling points). The subdivision performed by Algorithm 5 is dense only near the Julia set, as expected, because it avoids boxes outside the escape disk. Both algorithms certified existence and uniqueness using Banach's theorem of all six attracting periodic points, shown in light blue. However, Algorithm 4 explored 234189 boxes while Algorithm 5 explored 20085 boxes, a gain of an order of magnitude (also reflected in the total execution times, which includes graphics output).

6.3 Execution times

We wrote prototype implementations of our algorithms in the scripting language Lua. Our programs were meant to support the qualitative analysis given above and were instrumented for that purpose. The programs were not meant to support quantitative analysis of execution time. Nevertheless, as a rough indication, Table 2 gives the execution times for the computations illustrated in the figures. Note that the experiments do not always solve the same instance or even the same problem: they were chosen to display different convergence behaviors. In particular, Algorithm 4 finds only attracting fixed points, whereas the previous algorithms find all fixed points. Algorithm 5 is specialized to complex polynomials and exploits the nature of complex dynamics.

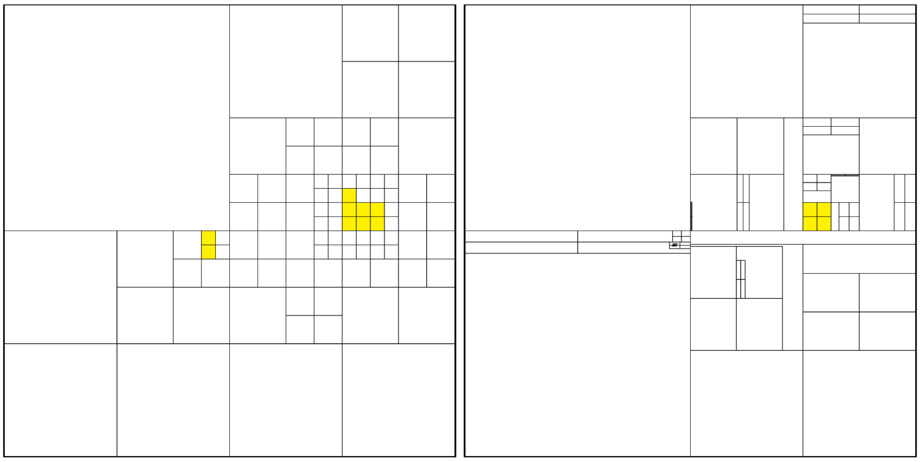


Figure 9: Decompositions of Algorithm 1 (left) and Algorithm 2 (right).

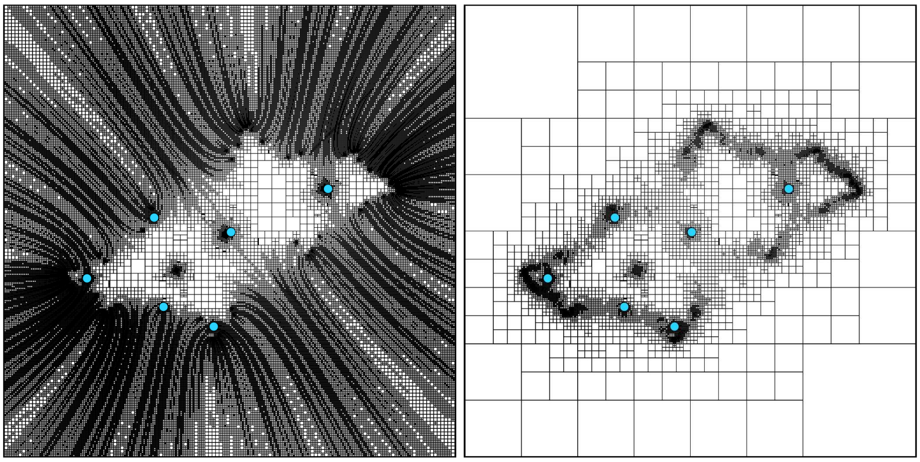


Figure 10: Decompositions of Algorithm 4 (left) and Algorithm 5 (right).

Figure	a	b	c
2	0.15567	0.01762	0.02860
3	19.46442	0.23579	18.64248
4	2.20107	2.20107	19.02098
5	19.42642	0.14639	0.01707
6	0.01762	0.01049	
7	0.13420	0.11114	
8	0.21073	0.23416	

Table 2: Execution times in seconds for our numerical experiments.

7 Conclusion

We proposed a hybrid interval algorithm that finds all attracting fixed points of a map. The algorithm is a novel combination of classical mathematical results and standard interval techniques. We also specialized this algorithm to find all attracting periodic points of a complex polynomial exploiting well-known facts of complex dynamics.

While the correctness of interval methods is assured, their efficiency depend on the quality of the interval estimates. Natural interval extensions provide first-order interval estimates. High-order estimates exist, but they are naturally more expensive to compute [Cornelius and Lohner 1984]. A natural direction for further work is to study the effect of high-order estimates on the overall efficiency of the algorithms presented here. Of special interest are second-order estimates, such as mean-value forms [Caprani and Madsen 1980] and affine arithmetic [de Figueiredo and Stolfi 2004], which may provide a good cost-benefit balance. Affine arithmetic will also probably help mitigate the wrapping effect.

Acknowledgements

The first author was partially supported by CNPq and FAPERJ doctoral scholarships. The second author is partially supported by a CNPq research grant. This research was done in the Visgraf Computer Graphics laboratory at IMPA. Visgraf is supported by the funding agencies FINEP, CNPq, and FAPERJ, and also by gifts from IBM Brasil, Microsoft, NVIDIA, and other companies.

References

[Agarwal et al. 2001] Agarwal, R. P., Meehan, M., and O'Regan, D. (2001). *Fixed point theory and applications.* Cambridge University Press.

[Baker Kearfott 1996] Baker Kearfott, R. (1996). *Rigorous global search: Continuous problems.* Kluwer.

[Berinde 2007] Berinde, V. (2007). *Iterative approximation of fixed points*, volume 1912 of *Lecture Notes in Mathematics.* Springer.

[Branner 1989] Branner, B. (1989). The Mandelbrot set. In [Devaney and Keen 1989], pages 75–105.

[Caprani and Madsen 1975] Caprani, O. and Madsen, K. (1975). Contraction mappings in interval analysis. *BIT*, 15(4):362–366.

[Caprani and Madsen 1978] Caprani, O. and Madsen, K. (1978). Iterative methods for interval inclusion of fixed points. *BIT*, 18(1):42–51.

[Caprani and Madsen 1980] Caprani, O. and Madsen, K. (1980). Mean value forms in interval analysis. *Computing*, 25(2):147–154.

[Cornelius and Lohner 1984] Cornelius, H. and Lohner, R. (1984). Computing the range of values of real functions with accuracy higher than second order. *Computing*, 33(3-4):331–347.

[de Figueiredo and Stolfi 2004] de Figueiredo, L. H. and Stolfi, J. (2004). Affine arithmetic: concepts and applications. *Numerical Algorithms*, 37(1):147–158.

[Devaney and Keen 1989] Devaney, R. L. and Keen, L., editors (1989). *Chaos and fractals: The mathematics behind the computer graphics.* Proc. Symposia in Applied Mathematics 39. AMS.

[Hansen and Walster 2003] Hansen, E. and Walster, G. W. (2003). *Global optimization using interval analysis.* CRC Press.

[Hubbard et al. 2001] Hubbard, J., Schleicher, D., and Sutherland, S. (2001). How to find all roots of complex polynomials by Newton's method. *Inventiones Mathematicae*, 146(1):1–33.

[Keen 1989] Keen, L. (1989). Julia sets. In [Devaney and Keen 1989], pages 57–74.

[Mayer 1995] Mayer, G. (1995). Epsilon-inflation in verification algorithms. *Journal of Computational and Applied Mathematics*, 60(1-2):147–169.

[McClure 2019] McClure, M. (2019). *Basic complex dynamics: A computational approach.*

[Moore 1966] Moore, R. E. (1966). *Interval analysis.* Prentice-Hall.

[Moore 1979] Moore, R. E. (1979). *Methods and applications of interval analysis.* SIAM.

[Moore et al. 2009] Moore, R. E., Baker Kearfott, R., and Cloud, M. J. (2009). *Introduction to interval analysis.* SIAM.

[Rall 1982] Rall, L. B. (1982). A theory of interval iteration. *Proceedings of the American Mathematical Society*, 86(4):625–631.

[Rall 1986] Rall, L. B. (1986). The arithmetic of differentiation. *Mathematics Magazine*, 59(5):275–282.

[Rall 1987] Rall, L. B. (1987). Interval methods for fixed-point problems. *Numerical Functional Analysis and Optimization*, 9(1-2):35–59.

[Rihm 2001] Rihm, R. (2001). Acceleration of iteration methods for interval fixed point problems. *Linear Algebra and its Applications*, 324(1-3):189–207.

[Rump 1998] Rump, S. M. (1998). A note on epsilon-inflation. *Reliable Computing*, 4(4):371–375.

[Samet 1984] Samet, H. (1984). The quadtree and related hierarchical data structures. *Computing Surveys*, 16(2):187–260.

[Samet and Tamminen 1985] Samet, H. and Tamminen, M. (1985). Bintrees, CSG trees, and time. *Computer Graphics*, 19(3):121–130. (Proceedings of SIGGRAPH '85.).

[Scarf 1967] Scarf, H. (1967). The approximation of fixed points of a continuous mapping. *SIAM Journal on Applied Mathematics*, 15(5):1328–1343.

[Todd 1976] Todd, M. J. (1976). *The computation of fixed points and applications*, volume 124 of *Lecture Notes in Economics and Mathematical Systems.* Springer.

[Traylor and Kreinovich 1995] Traylor, B. and Kreinovich, V. (1995). A bright side of NP-hardness of interval computations: interval heuristics applied to NP-problems. *Reliable Computing*, 1(3):343–359.

[Tucker 2011] Tucker, W. (2011). *Validated numerics: A short introduction to rigorous computations.* Princeton University Press.

[von Haeseler and Peitgen 1988] von Haeseler, F. and Peitgen, H.-O. (1988). Newton's method and complex dynamical systems. *Acta Applicandae Mathematicae*, 13(1-2):3–58.