Images of Julia sets that you can trust

Luiz Henrique de Figueiredo

with

Diego Nehab (IMPA) • Jorge Stolfi (UNICAMP) • João Batista Oliveira (PUCRS)
Can we trust this beautiful image?

Curtis McMullen
Julia sets

Study the dynamics of \( f(z) = z^2 + c \) for \( c \in \mathbb{C} \) fixed

\[
\begin{align*}
z_1 &= f(z_0), \\
z_2 &= f(z_1), \\
&\vdotswithin{=} \\
z_n &= f(z_{n-1}) = f^n(z_0)
\end{align*}
\]

What happens with the orbit of \( z_0 \in \mathbb{C} \) under \( f \)?
Julia sets

- Unbounded orbits
- Bounded orbits
Julia sets

- Unbounded orbits
- Bounded orbits

Attraction basin of \( \infty \) \( A(\infty) \)
Julia sets

Unbounded orbits

Bounded orbits

Attraction basin of $\infty$

Filled Julia set $A(\infty)$

Common boundary Julia set $K$
Julia sets

- Unbounded orbits
- Bounded orbits
- Common boundary
- Attraction basin of ∞: \( A(\infty) \)
- Filled Julia set: \( K \)
- Julia set: \( J \)
Julia set zoo

$c = 0.275$

$c = \frac{1}{4}$

$c = 0$

$c = -\frac{3}{4}$

$c = -1.312$

$c = -1.375$

$c = -2$

$c = i$

$c = (+0.285, +0.535)$

$c = (-0.125, +0.750)$

$c = (-0.500, +0.563)$

$c = (-0.687, +0.312)$
Julia set catalog: the Mandelbrot set

c ∈ \mathcal{M} \implies J_c \text{ is connected}
c \not\in \mathcal{M} \implies J_c \text{ is a Cantor set}

c ∈ \mathcal{M} \implies 0 \in K_c

Julia–Fatou dichotomy
Julia set catalog: the Mandelbrot set
demo...

$c \in \mathcal{M} := 0 \in K_c$

Julia–Fatou dichotomy
$c \in \mathcal{M} \Rightarrow J_c$ is connected
$c \notin \mathcal{M} \Rightarrow J_c$ is a Cantor set
Why distrust this beautiful image?

Curtis McMullen
Why distrust this beautiful image?

Escape-time algorithm

for $z_0$ in a grid of points in $\Omega$

$z \leftarrow z_0$

$n \leftarrow 0$

while $|z| \leq R$ and $n \leq N$ do

$z \leftarrow z^2 + c$

$n \leftarrow n + 1$

paint $z_0$ with color $n$
Escape-time algorithm

for $z_0$ in a grid of points in $\Omega$

\[
\begin{align*}
z &\leftarrow z_0 \\
n &\leftarrow 0 \\
\text{while } |z| \leq R \text{ and } n \leq N \text{ do} \\
& \quad z \leftarrow z^2 + c \\
& \quad n \leftarrow n + 1 \\
& \quad \text{paint } z_0 \text{ with color } n
\end{align*}
\]

escape radius
\[
R = \max(|c|, 2) \\
J \subset B(0, R)
\]
Lemma. If $z \in \mathbb{C}$ and $|z| > R = \max(|c|, 2) \Rightarrow |f^n(z)| \to \infty$ as $n \to \infty$.

Proof. The triangle inequality gives

$$|z^2| = |z^2 + c - c| \leq |z^2 + c| + |c|$$

and so

$$|f(z)| = |z^2 + c| \geq |z^2| - |c| = |z|^2 - |c| > |z|^2 - |z| = |z|(|z| - 1) > |z| > R$$

Iterating, we get $|f^n(z)| > |z|(|z| - 1)^n \to \infty$ because $|z| - 1 > 1$. \qed

Corollary. Every unbounded orbit escapes to $\infty$. \hspace{1cm} A(\infty)
Why distrust this beautiful image?

Escape-time algorithm

for \( z_0 \) in a grid of points in \( \Omega \)

\[
\begin{align*}
    z &\leftarrow z_0 \\
n &\leftarrow 0 \\
\text{while } |z| \leq R \text{ and } n \leq N \text{ do} \\
    &z \leftarrow z^2 + c \\
    &n \leftarrow n + 1 \\
paint \ z_0 \ with \ color \ n
\end{align*}
\]

escape radius

\[
R = \max(|c|, 2) \quad J \subset B(0, R)
\]
Why distrust this beautiful image?

Escape-time algorithm

for \( z_0 \) in a grid of points in \( \Omega \)

\[
\begin{align*}
  &z \leftarrow z_0 \\
  &n \leftarrow 0 \\
  &\text{while } |z| \leq R \text{ and } n \leq N \text{ do} \\
  &\quad z \leftarrow z^2 + c \\
  &\quad n \leftarrow n + 1 \\
  &\text{paint } z_0 \text{ with color } n
\end{align*}
\]
Why distrust this beautiful image?

Spatial sampling
need fine grid
what happens between samples?

Escape-time algorithm

for \( z_0 \) in a grid of points in \( \Omega \)

\[
\begin{align*}
  z & \leftarrow z_0 \\
  n & \leftarrow 0 \\
  \text{while } |z| \leq R \text{ and } n \leq N \text{ do} \\
  & \quad z \leftarrow z^2 + c \\
  & \quad n \leftarrow n + 1 \\
  \text{paint } z_0 \text{ with color } n
\end{align*}
\]
Why distrust this beautiful image?

- Spatial sampling
- Partial orbits
  program cannot run forever

>> Escape-time algorithm

for $z_0$ in a grid of points in $\Omega$

\[
\begin{align*}
  z & \leftarrow z_0 \\
  n & \leftarrow 0 \\
  \text{while } |z| \leq R \text{ and } n \leq N \text{ do} \\
  & z \leftarrow z^2 + c \\
  & n \leftarrow n + 1 \\
  \text{paint } z_0 \text{ with color } n
\end{align*}
\]
Why distrust this beautiful image?

- Spatial sampling
- Partial orbits
- Floating-point rounding errors
  - squaring needs double digits

Escape-time algorithm

for \( z_0 \) in a grid of points in \( \Omega \)

\[
\begin{align*}
  z &\leftarrow z_0 \\
  n &\leftarrow 0 \\
  \text{while } |z| \leq R \text{ and } n \leq N \text{ do} \\
  &\quad z \leftarrow z^2 + c \\
  &\quad n \leftarrow n + 1 \\
  \text{paint } z_0 \text{ with color } n
\end{align*}
\]
Why distrust this beautiful image?

- **Spatial sampling**
  Compute color on grid points
  Cannot be sure grid is fine enough
  Cannot be sure behavior at sample points is typical
  Finer grid ⇒ more detail

- **Partial orbits**
  Can only compute partial orbits
  Cannot be sure partial orbits are long enough
  Longer orbits ⇒ more detail

- **Floating-point errors**
  $z^2$ needs twice the number of digits that $z$ needs
  Do rounding errors during iteration influence classification of points?
  Multiple-precision ⇒ more detail (deep zoom)
You can trust our method

- No spatial sampling

- No orbits

- No floating-point errors
You can trust our method

- No spatial sampling
  Classify entire rectangles in the complex plane
  Spatial resolution limited by available memory
  Deeper quadtree $\Rightarrow$ more detail

- No orbits

- No floating-point errors
You can trust our method

- **No spatial sampling**
  Classify *entire rectangles* in the complex plane
  Spatial resolution limited by available memory
  Deeper quadtree $\Rightarrow$ more detail

- **No orbits**
  Evaluate $f$ once on each cell using interval arithmetic
  Perform *no function iteration* at all
  Use cell mapping and color propagation in graphs

- **No floating-point errors**
You can trust our method

- **No spatial sampling**
  Classify *entire rectangles* in the complex plane
  Spatial resolution limited by available memory
  Deeper quadtree ⇒ more detail

- **No orbits**
  Evaluate $f$ once on each cell using interval arithmetic
  Perform *no function iteration* at all
  Use cell mapping and color propagation in graphs

- **No floating-point errors**
  All numbers are dyadic fractions with restricted range and precision
  Use *error-free fixed-point* arithmetic
  Precision depends only on spatial resolution
  Standard double-precision floating-point enough for huge images
Our algorithm

quadtree for
\( \Omega = [-R, R] \times [-R, R] \)

- white rectangles contained in \( A(\infty) \)
- black rectangles contained in \( K \)
- gray rectangles contain \( J \)
Our algorithm

quadtree for
\[ \Omega = [-R, R] \times [-R, R] \]

- white rectangles contained in \( A(\infty) \)
- black rectangles contained in \( K \)
- gray rectangles contain \( J \)

certified decomposition
Our algorithm

quadtree for
\[ \Omega = [-R, R] \times [-R, R] \]

- refinement
- cell mapping
- color propagation
Our algorithm

quadtree for
\[ \Omega = [-R, R] \times [-R, R] \]

- refinement
- cell mapping
- color propagation
Quadtrees are a type of quadtree. For level 0, we have $c = -1$. 

[Diagram of a quadtree]
Quadtree  \( c = -1 \)  level 2
Quadtrees: $c = -1$, level 3.
Quadtree

\[ c = -1 \]

level 4
Quadtree

\[ c = -1 \]  level 5
Quadtree

\[ c = -1 \quad \text{level 7} \]
Quadtree
c = −1
level 8
Quadtree $c = -1$ level 9
Quadtree $c = -1$ level 10
Quadtree $c = -1$ level 11
Quadtrees

\[ c = -1 \quad \text{level 12} \]
Quadtree $c = -1$ level 13
Quadtrees

\[ c = -1 \quad \text{level 14} \]
Adaptive approximation \( c = -1 \) level 14
Adaptive approximation $c = -1$ level 0
Adaptive approximation \quad c = -1 \quad \text{level 1}
Adaptive approximation  \( c = -1 \)  level 2
Adaptive approximation  \( c = -1 \)  level 3
Adaptive approximation $c = -1$ level 4
Adaptive approximation  $c = -1$  level 5
Adaptive approximation \[ c = -1 \] level 6
Adaptive approximation \[ c = -1 \] level 7
Adaptive approximation  \[ c = -1 \]  level 8
Adaptive approximation $c = -1$ level 9
Adaptive approximation  \( c = -1 \)  level 10
Adaptive approximation $c = -1$ level 11
Adaptive approximation $c = -1$ level 12
Adaptive approximation  $c = -1$  level 13
Adaptive approximation \( c = -1 \) level 14
Adaptive approximation $c = -1$
Adaptive approximation  \( c = -1 \)
Our algorithm

quadtree for
\( \Omega = [-R, R] \times [-R, R] \)

- refinement
- cell mapping
- color propagation
Cell mapping

Directed graph on the leaves of the quadtree and exterior

- edges emanate from each leaf gray cell $A$
- add edge $A \rightarrow B$ for each leaf cell $B$ that intersects $f(A)$

$$f(A) \subseteq \bigcup_{A \rightarrow B} B$$
Cell mapping

Directed graph on the leaves of the quadtree and exterior

- edges emanate from each leaf gray cell $A$

- add edge $A \rightarrow B$ for each leaf cell $B$ that intersects $f(A)$

$$f(A) \subseteq \bigcup_{A \rightarrow B} B$$

Conservative estimate of the dynamics

Avoid point sampling
<table>
<thead>
<tr>
<th>Cell mapping</th>
<th>source cell</th>
<th>leaf gray cell</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Cell mapping

exact image under $f$
Cell mapping  quadtree traversal
Cell mapping target cells contain exact image
Cell mapping edges demo...
Our algorithm

quadtree for
\( \Omega = [-R, R] \times [-R, R] \)

- refinement
- cell mapping
- color propagation
Color propagation

Propagate white and black to gray cells

- **new white cells**
  gray cells for which all paths end in white cells

- **new black cells**
  gray cells for which no path ends in a white cell
Color propagation

Propagate white and black to gray cells

- new white cells
  gray cells for which all paths end in white cells

- new black cells
  gray cells for which no path ends in a white cell

Graph traversals replace function iteration

Avoid floating-point errors
The algorithm refinement
The algorithm cell mapping
The algorithm new white cells...
The algorithm new white cells...
The algorithm  new white cells
The algorithm gray cells that reach white...
The algorithm gray cells that reach white...
The algorithm gray cells that reach white
The algorithm

new black cells
Adaptive approximation examples
Adaptive approximation \( c = 0.12 + 0.30i \) level 0
Adaptive approximation  \[ c = 0.12 + 0.30 \, i \]  level 1
Adaptive approximation  \( c = 0.12 + 0.30 i \)  level 2
Adaptive approximation  $c = 0.12 + 0.30i$  level 3
Adaptive approximation  \[ c = 0.12 + 0.30i \]  level 4
Adaptive approximation \[ c = 0.12 + 0.30 i \] level 5
Adaptive approximation $c = 0.12 + 0.30i$ level 6
Adaptive approximation $c = 0.12 + 0.30i$ level 7
Adaptive approximation

\[ c = 0.12 + 0.30i \]

level 8
Adaptive approximation $c = 0.12 + 0.30i$ level 9
Adaptive approximation  \( c = 0.12 + 0.30i \)  level 10
Adaptive approximation  \[ c = 0.12 + 0.30 \, i \]  level 11
Adaptive approximation \[ c = 0.12 + 0.30i \] level 12
Adaptive approximation $c = 0.12 + 0.30i$  level 13
Adaptive approximation  $c = 0.12 + 0.30 \, i$  level 14
Adaptive approximation  \( c = 0.12 + 0.30 i \)
Adaptive approximation  \[ c = 0.12 + 0.30 i \]
Adaptive approximation $c = -0.12 + 0.60i$ level 0
Adaptive approximation \( c = -0.12 + 0.60i \) level 1
Adaptive approximation \( c = -0.12 + 0.60 \, i \) level 2
Adaptive approximation \[ c = -0.12 + 0.60i \] level 3
Adaptive approximation \( c = -0.12 + 0.60 \, i \) level 4
Adaptive approximation

\[ c = -0.12 + 0.60i \]

level 5
Adaptive approximation $c = -0.12 + 0.60i$ level 6
Adaptive approximation $c = -0.12 + 0.60i$ level 7
Adaptive approximation  \[ c = -0.12 + 0.60i \]  level 8
Adaptive approximation  \[ c = -0.12 + 0.60i \]  level 9
Adaptive approximation \( c = -0.12 + 0.60i \) level 10
Adaptive approximation $c = -0.12 + 0.60i$ level 11
Adaptive approximation  \[ c = -0.12 + 0.60i \]  level 12
Adaptive approximation $c = -0.12 + 0.60i$ level 13
Adaptive approximation  \[ c = -0.12 + 0.60i \]  level 14
Adaptive approximation \[ c = -0.12 + 0.60i \]
Adaptive approximation \[ c = -0.12 + 0.60 \, i \]
Adaptive approximation \[ c = -0.12 + 0.74i \] level 0
Adaptive approximation $c = -0.12 + 0.74i$ level 1
Adaptive approximation  \( c = -0.12 + 0.74i \)  level 2
Adaptive approximation \[ c = -0.12 + 0.74i \] level 3
Adaptive approximation \( c = -0.12 + 0.74i \) level 4
Adaptive approximation \( c = -0.12 + 0.74i \) level 5
Adaptive approximation  \( c = -0.12 + 0.74i \)  level 7
Adaptive approximation $c = -0.12 + 0.74i$ level 8
Adaptive approximation $c = -0.12 + 0.74i$ level 9
Adaptive approximation

$$c = -0.12 + 0.74i$$

level 10
Adaptive approximation \[ c = -0.12 + 0.74i \] level 11
Adaptive approximation  \[ c = -0.12 + 0.74i \]  level 12
Adaptive approximation $c = -0.12 + 0.74i$  level 13
Adaptive approximation $c = -0.12 + 0.74 i$ level 14
Adaptive approximation $c = -0.12 + 0.74i$
Adaptive approximation $c = -0.12 + 0.74i$
Adaptive approximation  \( c = i \)  level 0
Adaptive approximation \( c = i \) level 1
Adaptive approximation  \[ c = i \]  level 2
Adaptive approximation $c = i$ level 3
Adaptive approximation $c = i$ level 4
Adaptive approximation $c = i$ level 5
Adaptive approximation  $c = i$  level 6
Adaptive approximation $c = i$ level 7
Adaptive approximation $c = i$ level 8
Adaptive approximation  \( c = i \)  level 9
Adaptive approximation $c = i$ level 10
Adaptive approximation \[ c = i \] level 11
Adaptive approximation \( c = i \) level 12
Adaptive approximation $c = i$ level 13
Adaptive approximation $c = i$ level 14
Adaptive approximation  \( c = i \)
Adaptive approximation $c = i$
Adaptive approximation \[ c = -0.25 + 0.74i \] level 0
Adaptive approximation \[ c = -0.25 + 0.74i \] level 1
Adaptive approximation \[ c = -0.25 + 0.74i \] level 2
Adaptive approximation \( c = -0.25 + 0.74 \, i \) level 3
Adaptive approximation \[ c = -0.25 + 0.74i \] level 4
Adaptive approximation \[ c = -0.25 + 0.74i \] level 5
Adaptive approximation  \[ c = -0.25 + 0.74i \]  level 6
Adaptive approximation \[ c = -0.25 + 0.74i \] level 7
Adaptive approximation $c = -0.25 + 0.74i$ level 8
Adaptive approximation: $c = -0.25 + 0.74i$  level 9
Adaptive approximation \( c = -0.25 + 0.74i \) level 10
Adaptive approximation \[ c = -0.25 + 0.74i \] level 11
Adaptive approximation \( c = -0.25 + 0.74i \) level 12
Adaptive approximation $c = -0.25 + 0.74 i$ level 13
Adaptive approximation  \( c = -0.25 + 0.74i \)  level 14
Adaptive approximation \( c = -0.25 + 0.74i \)
Adaptive approximation \( c = -0.25 + 0.74 i \)
Applications

- Image generation
- Point and box classification
- Fractal dimension of Julia set
- Area of filled Julia set
- Diameter of Julia set
Applications certified numerical results

- **Image generation**
  - large images
  - smaller images with anti-aliasing

- **Point and box classification**
  - quadtree traversal + one function evaluation if gray

- **Fractal dimension of Julia set**
  - upper bound
  \[
  \dim_H = 1 + \frac{|c|^2}{4 \log 2} + \cdots \tag{Ruelle}
  \]

- **Area of filled Julia set**
  - lower and upper bounds
  \[
  \pi (1 - |p_1(c)|^2 - 3|p_2(c)|^2 - 5|p_3(c)|^2 - \cdots) \tag{Milnor}
  \]

- **Diameter of Julia set**
  - lower and upper bounds
Inverse Böttcher map \( \psi: \mathbb{C} \setminus \mathbb{D} \to \mathbb{C} \setminus \mathbb{K} \)

\[
\psi(w^2) = \psi(w)^2 + c
\]

Laurent series near \( \infty \)

\[
\psi(w) = w \left( 1 + \frac{a_2}{w^2} + \frac{a_4}{w^4} + \frac{a_6}{w^6} + \cdots \right)
\]

\[
a_2 = -\frac{c}{2} \quad a_{2n} = \frac{1}{2}(a_n - a_n^2) - \sum_{2 \leq j < n \atop j \text{ even}} a_j a_{2n-j} \quad a_{2n+1} = 0
\]

Gronwall's area theorem

\[
\text{area}(\mathbb{K}) = \pi(1 - |a_2|^2 - 3|a_4|^2 - 5|a_6|^2 - \cdots)
\]

Truncating series gives upper bounds

Quadtree gives both lower and upper bounds

series converges slowly
This follows, since we can choose 
\[ \psi : \mathbb{U} \to \mathbb{U} \]
mapping the origin to any given point of \( \mathbb{U} \), and since the Poincaré metric at the center of \( \psi \) is \( 2(\mu - 1) \).

As an example, if \( \mathbb{U} \) is a half-plane, then the Poincaré metric precisely agrees with the \((l/r)-\)metric \( Idz^2/lz^2 \).

**Figure 45.** Upper bounds for the area of the filled Julia set for \( f_c(z) = z^2 + c \) in the range \(-2 \leq c \leq .25\).
Area of filled Julia set \(-1.25 \leq c \leq 0.25\)
Area of filled Julia set \(-1.25 \leq c \leq 0.25\) level 19
Area of filled Julia set \(-1.25 \leq c \leq 0.25\) level 19
Area of filled Julia set $-1.25 \leq c \leq 0.25 \quad \text{level 19}$
Limitations

- Memory

- Need to explore $\Omega \supseteq [-R, R] \times [-R, R]$
Limitations

- **Memory**
  depth of quadtree and size of cell graph limited by available memory
currently spatial resolution $\approx 4 \times 10^{-6}$
cannot reach 20 levels

- **Need to explore** $\Omega \supseteq [-R, R] \times [-R, R]$
even if region of interest is smaller
limited amount of zoom
limitation inherent to using cell mapping because $f$ is transitive on $J$
Future work higher-degree polynomials

- Escape radius

- Bounding box
Future work: higher-degree polynomials

- **Escape radius**

  \[ R = \frac{1 + |a_d| + \cdots + |a_0|}{|a_d|} \]

  is an escape radius for \( f(z) = a_d z^d + \cdots + a_0 \) (Douady)

- **Bounding box**

  needs interval arithmetic with directed rounding
Cubic Julia set \( z^3 + 0.38 \)
Cubic Julia set $z^3 + 0.38$
Cubic Julia set \[ z^3 + 0.41 \]
Cubic Julia set \( z^3 - 3a^2 + b \) level 0
Cubic Julia set $z^3 - 3a^2 + b$ level 1
Cubic Julia set

\[ z^3 - 3a^2 + b \]

level 2
Cubic Julia set $z^3 - 3a^2 + b$ level 3
Cubic Julia set \[ z^3 - 3a^2 + b \] level 4
Cubic Julia set \[ z^3 - 3a^2 + b \] level 5
Cubic Julia set \[ z^3 - 3a^2 + b \] level 6
Cubic Julia set $z^3 - 3a^2 + b$ level 7
Cubic Julia set $z^3 - 3a^2 + b$ level 8
Cubic Julia set \[ z^3 - 3a^2 + b \] level 9
Cubic Julia set \( z^3 - 3a^2 + b \) level 10
Cubic Julia set $z^3 - 3a^2 + b$ level 11
Cubic Julia set \( z^3 - 3a^2 + b \) level 12
Cubic Julia set \[ z^3 - 3a^2 + b \] level 13
Cubic Julia set $z^3 - 3a^2 + b$ level 14
Cubic Julia set $z^3 - 3a^2 + b$
Cubic Julia set \[ z^3 - 3a^2 + b \]
Future work Newton’s method

- Which points converge to which root?
  Cayley (1879)

- Points that do not converge form the Julia set

- No escape radius

- Need to find explicit attracting regions around roots?
Future work

Newton’s method

\[ z^3 = 1 \]
Julia set panorama

http://monge.visgraf.impa.br/panorama/viewer/index.html?
img=../julia-256GP/julia.xml
Images of Julia sets that you can trust

Thanks!


Area of filled Julia set $-1.25 \leq c \leq 0.25$ level 19
Interval arithmetic \( f(z) = z^2 + c \)

\( (x,y) \mapsto (x^2 - y^2 + a, 2xy + b) \)

function f(xmin,xmax,ymin,ymax)
    local x2min,x2max=isqr(xmin,xmax)
    local y2min,y2max=isqr(ymin,ymax)
    local xymin,xymax=imul(xmin,xmax,ymin,ymax)
    return x2min-y2max+a,x2max-y2min+a,2*xymin+b,2*xymax+b
end

function imul(xmin,xmax,ymin,ymax)
    local mm=xmin*ymin
    local mM=xmin*ymax
    local Mm=xmax*ymin
    local MM=xmax*ymax
    local m,M=mm,mm
    if m>mM then m=mM elseif M<mM then M=mM end
    if m>Mm then m=Mm elseif M<Mm then M=Mm end
    if m>MM then m=MM elseif M<MM then M=MM end
    return m,M
end
**Interval arithmetic**  

\[ f(z) = z^2 + c \]

\[(x, y) \mapsto (x^2 - y^2 + a, 2xy + b)\]

**function** f(xmin,xmax,ymin,ymax)

  local x2min,x2max=isqr(xmin,xmax)
  local y2min,y2max=isqr(ymin,ymax)
  local xymin,xymax=imul(xmin,xmax,ymin,ymax)
  return x2min-y2max+a,x2max-y2min+a,2*xymin+b,2*xymax+b

end

**function** isqr(xmin,xmax)

  local u=xmin^2
  local v=xmax^2
  if xmin<0 and 0<=xmax then
    if u<v then return 0,v else return 0,u end
  else
    if u<v then return u,v else return v,u end
  end

end
Interval arithmetic

\[ f(z) = z^3 + c \]

\((x, y) \mapsto (x^3 - 3xy^2 + a, -y^3 + 3x^2y + b)\)

function f(xmin,xmax,ymin,ymax)

    local x2min,x2max=isqr(xmin,xmax)
    local y2min,y2max=isqr(ymin,ymax)
    local xy2min,xy2max=imul(xmin,xmax,y2min,y2max)
    local x2ymin,x2ymax=imul(x2min,x2max,ymin,ymax)
    local x3min,x3max=icub(xmin,xmax)
    local y3min,y3max=icub(ymin,ymax)
    return x3min-3*xy2max+a, x3max-3*xy2min+a,
           -y3max+3*x2ymin+b,-y3min+3*x2ymax+b

end

function icub(xmin,xmax)

    return xmin^3,xmax^3

end