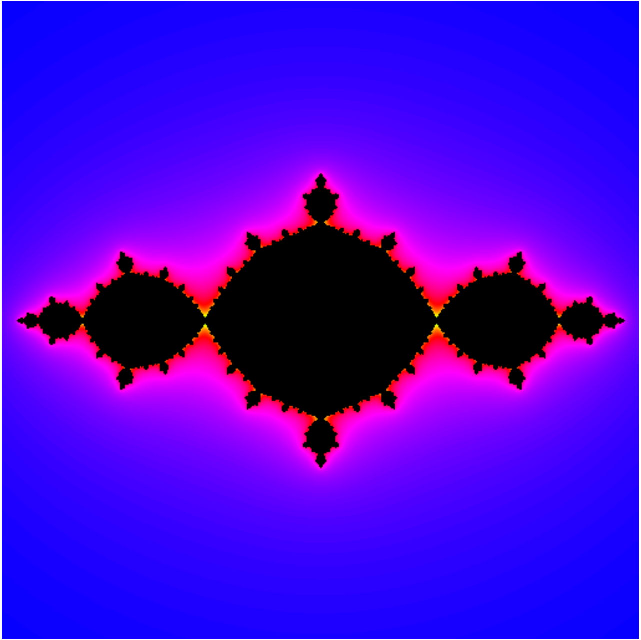
First Palis-Balzan International Symposium on
Dynamical Systems

Images of Julia sets that you can trust

Luiz Henrique de Figueiredo
IMPA

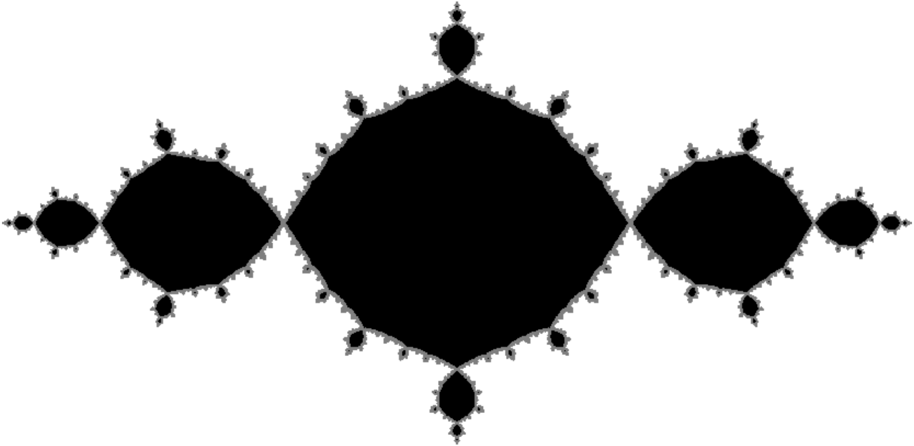
with
Diego Nehab (IMPA) • Jorge Stolfi (UNICAMP) • João Batista Oliveira (PUCRS)

Can we trust this beautiful image?



Curtis McMullen

Julia sets

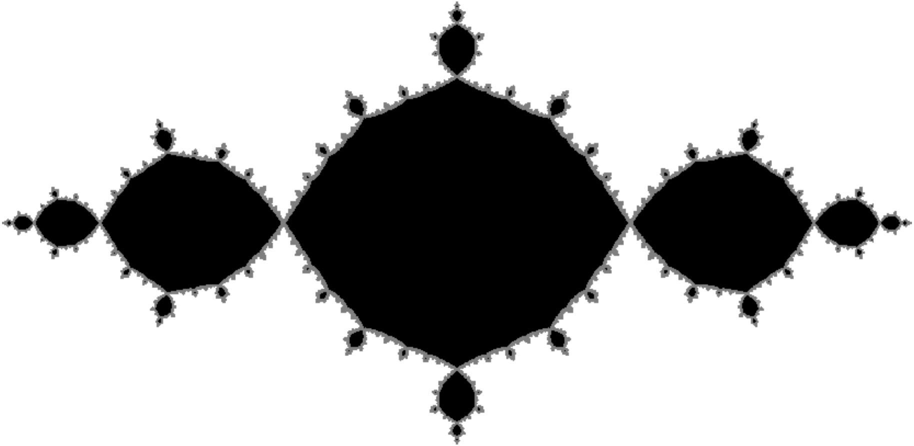


Describe the dynamics of $f(z) = z^2 + c$ for $c \in \mathbb{C}$ fixed

$$z_1 = f(z_0), \quad z_2 = f(z_1), \quad \ldots, \quad z_n = f(z_{n-1}) = f^n(z_0)$$

What happens with the orbit of $z_0 \in \mathbb{C}$ under f ?

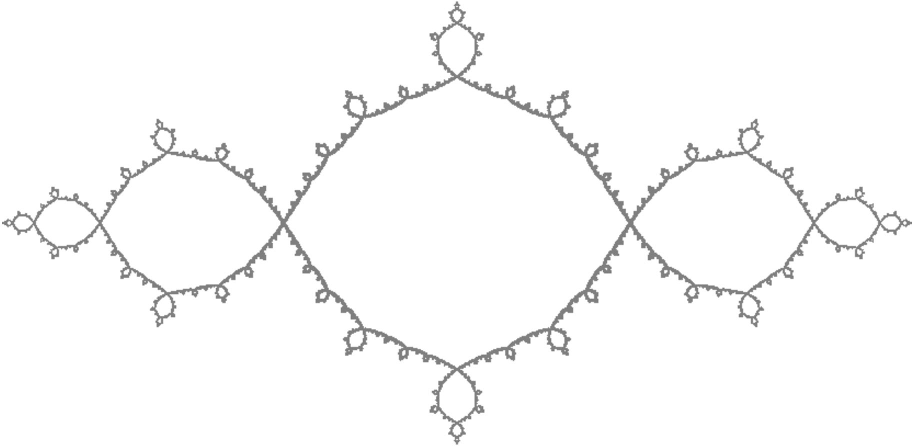
Julia sets



□ unbounded orbits = attraction basin of ∞ $A(\infty)$

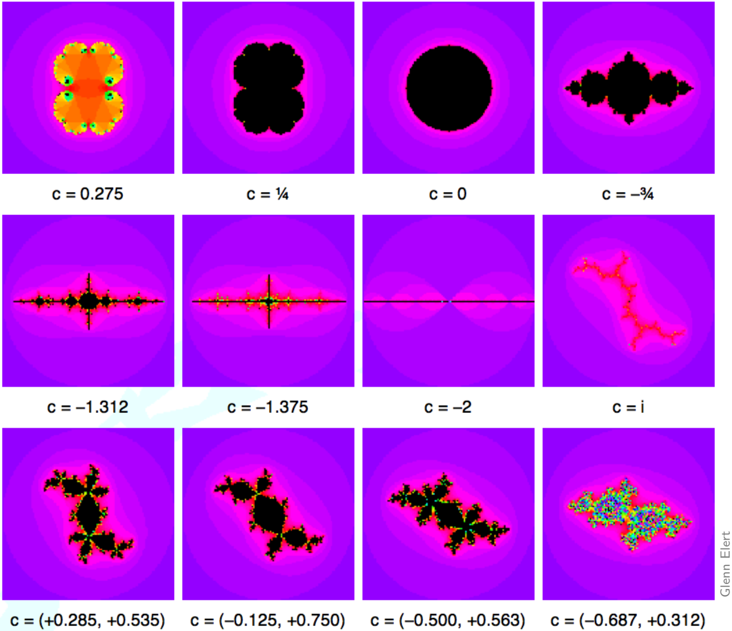
■ bounded orbits = filled Julia set K

Julia sets

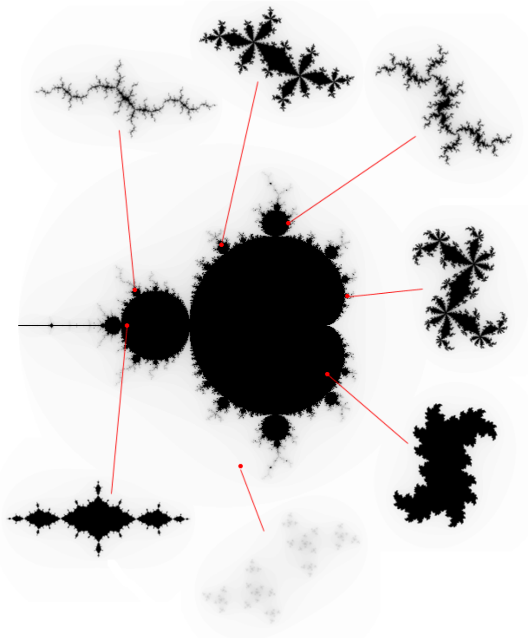


☐	unbounded orbits = attraction basin of ∞	$A(\infty)$
■	bounded orbits = filled Julia set	K
■	common boundary = Julia set	J

Julia set zoo



Julia set catalog: the Mandelbrot set



Paul Bourke

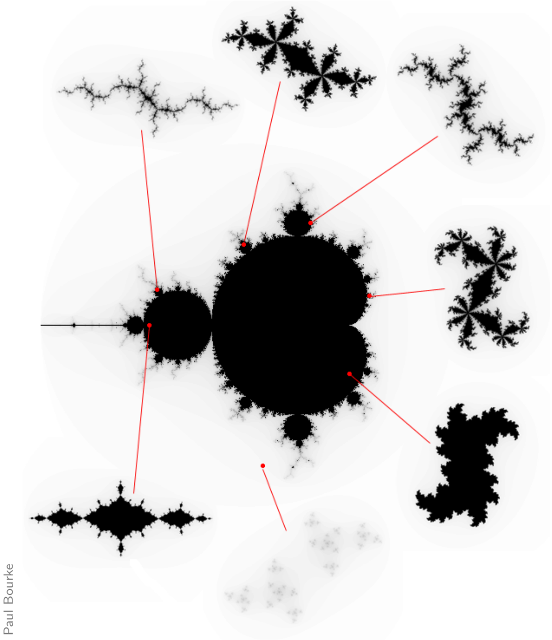
$c \in \mathcal{M} \;:=\; 0 \in K_c$

Julia–Fatou dichotomy

$c \in \mathcal{M} \;\Rightarrow\; J_c$ is connected

$c \notin \mathcal{M} \;\Rightarrow\; J_c$ is a Cantor set

Julia set catalog: the Mandelbrot set demo. . .



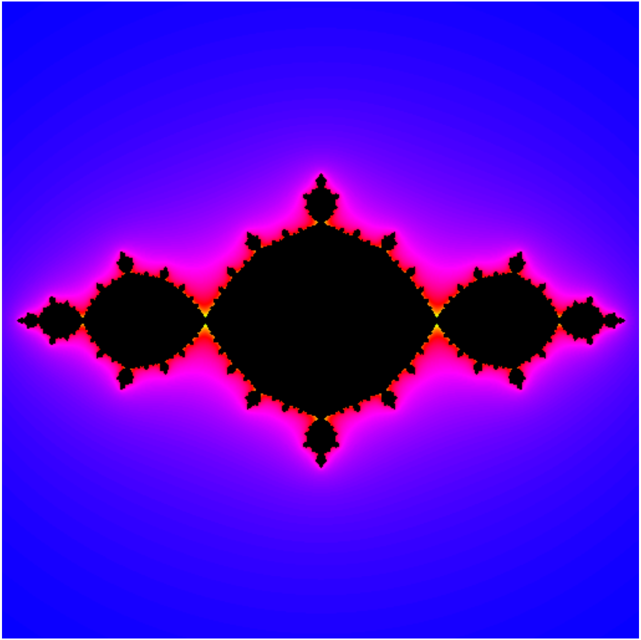
$c \in \mathcal{M} \ := \ 0 \in K_c$

Julia–Fatou dichotomy

$c \in \mathcal{M} \ \Rightarrow \ J_c$ is connected

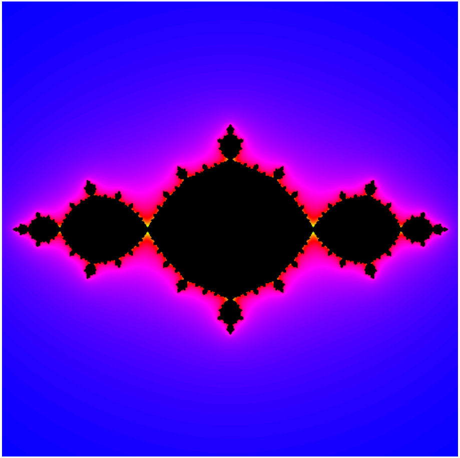
$c \notin \mathcal{M} \ \Rightarrow \ J_c$ is a Cantor set

Why distrust this beautiful image?



Curtis McMullen

Why distrust this beautiful image?


Curtis McMullen

Escape-time algorithm

for z_0 in a grid of points in Ω

$z \leftarrow z_0$

$n \leftarrow 0$

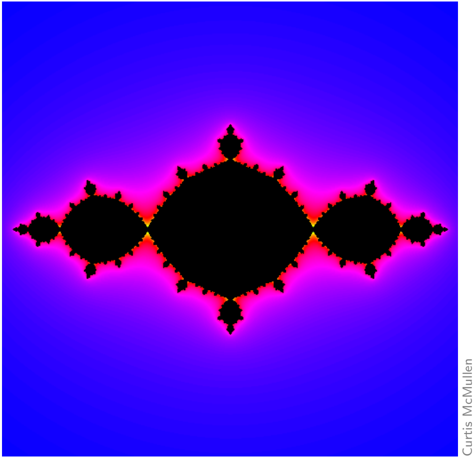
while $|z| \leq R$ and $n \leq N$ do

$z \leftarrow z^2 + c$

$n \leftarrow n + 1$

paint z_0 with color n

Why distrust this beautiful image?



Escape-time algorithm

for z_0 in a grid of points in Ω

$z \leftarrow z_0$

$n \leftarrow 0$

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 radius

$R = \max(|c|, 2) \qquad J \subset B(0, R)$

Why distrust this beautiful image?

- Spatial sampling
  what happens between samples?

Escape-time algorithm

for z_0 in a grid of points in Ω
  $z \leftarrow z_0$
  $n \leftarrow 0$
  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 radius

$R = \max(|c|, 2) \qquad J \subset B(0, R)$

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 Ω
  $z \leftarrow z_0$
  $n \leftarrow 0$
  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 radius

$R = \max(|c|, 2) \qquad J \subset B(0, R)$

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 Ω
    $z \leftarrow z_0$
    $n \leftarrow 0$
    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 radius

$R = \max(|c|, 2) \qquad J \subset B(0, R)$

Why distrust this beautiful image?

- Spatial sampling
  Compute color on grid points
  Cannot be sure behaviour at sample points is typical
  Finer grid $\Rightarrow$ more detail
- Partial orbits
  Can only compute partial orbits
  Cannot be sure partial orbits are long enough
  Longer orbits $\Rightarrow$ 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 $\Rightarrow$ 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 label 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 $\Rightarrow$ more detail

- No orbits

  Evaluate f once on each cell using interval arithmetic

  Perform no function iteration at all

  Use cell mapping and label propagation in graphs

- No floating-point errors

  All numbers are dyadic fractions with restricted range and precision

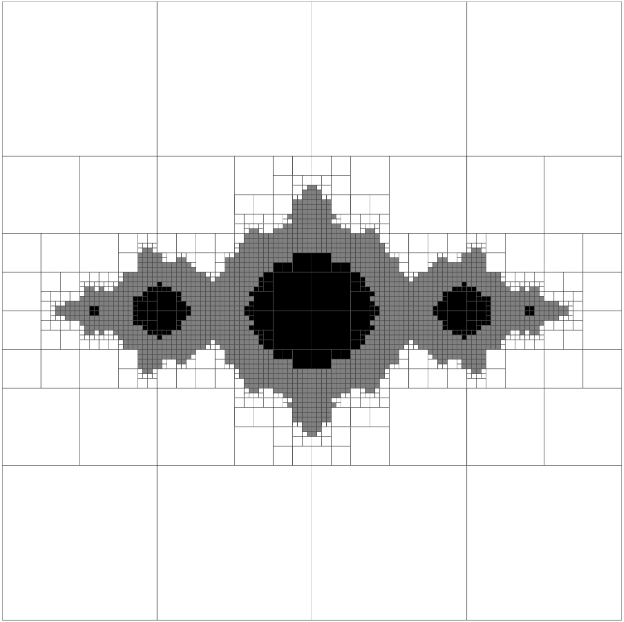
  Use error-free fixed-point arithmetic

  Precision depends only on spatial resolution

  Standard double-precision floating-point enough for huge images

Our algorithm

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



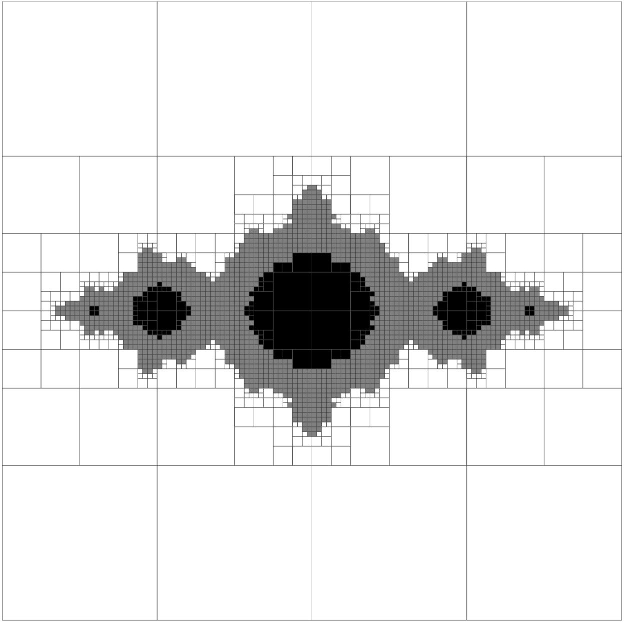
- white rectangles contained in $A(\infty)$
- black rectangles contained in K
- gray rectangles contain J

Our algorithm

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

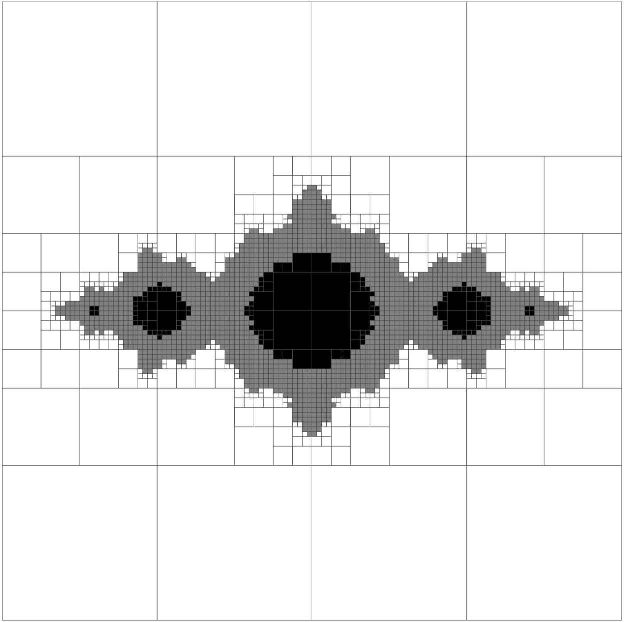
- white rectangles contained in $A(\infty)$
- black rectangles contained in K
- gray rectangles contain J

certified decomposition



Our algorithm

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

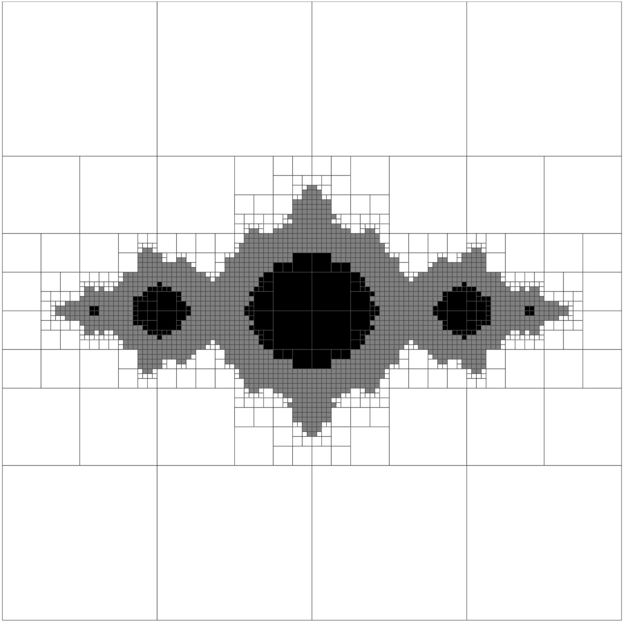


- refinement
- cell mapping
- label propagation

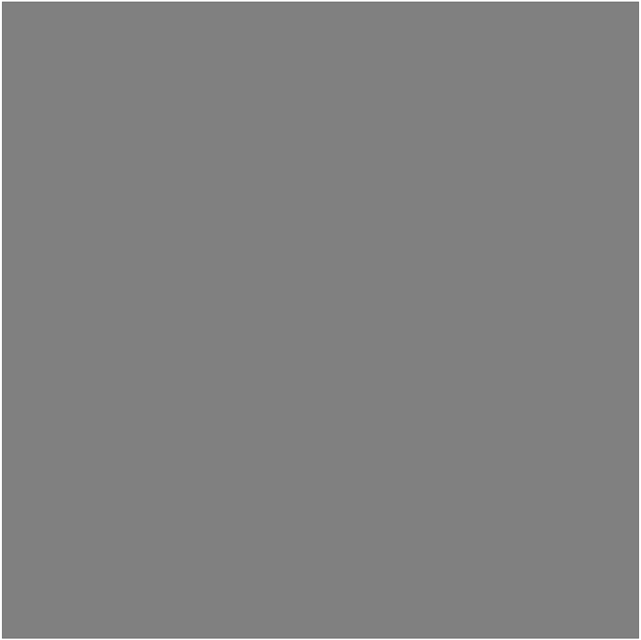
Our algorithm

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

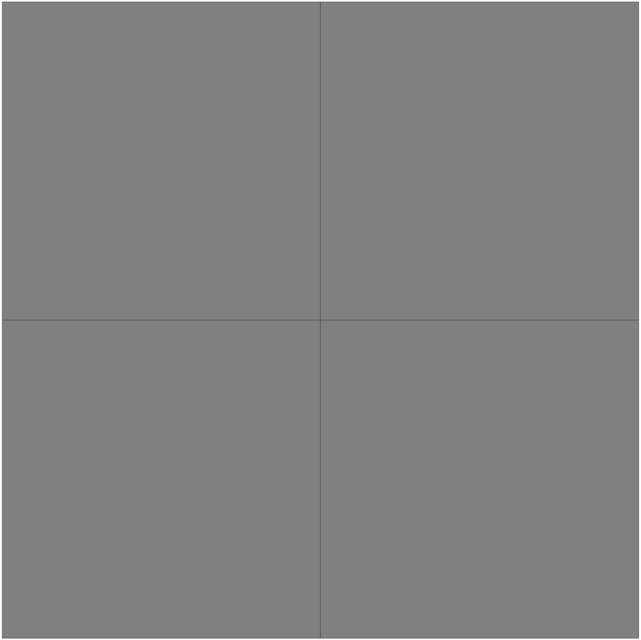
- refinement
- cell mapping
- label propagation



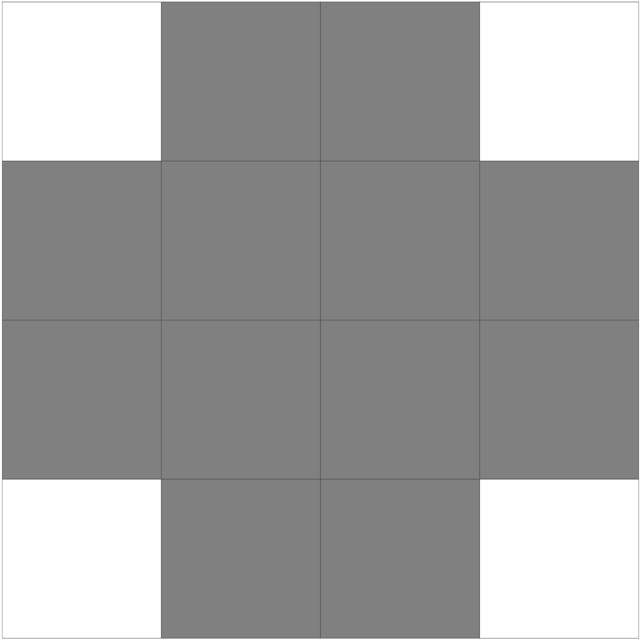
Quadtree $c = -1$ level 0



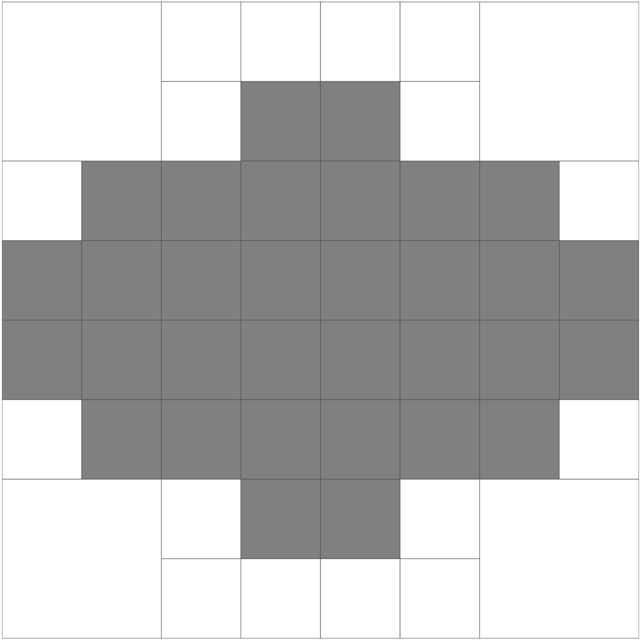
Quadtree $c = -1$ level 1



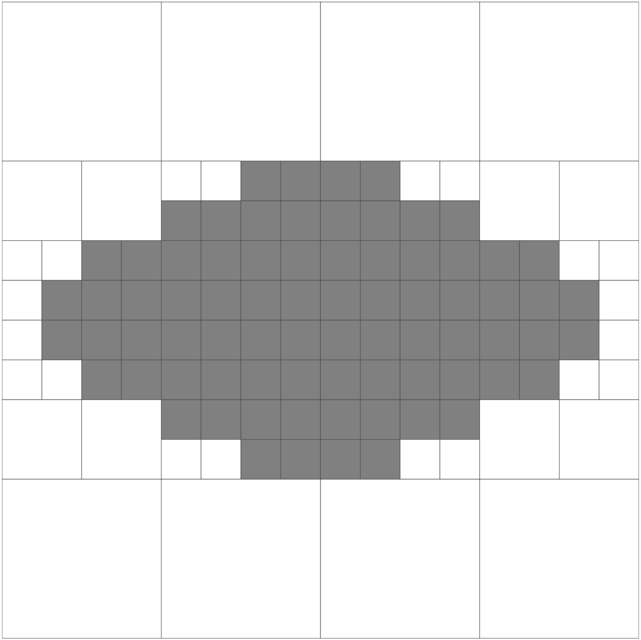
Quadtree $c = -1$ level 2



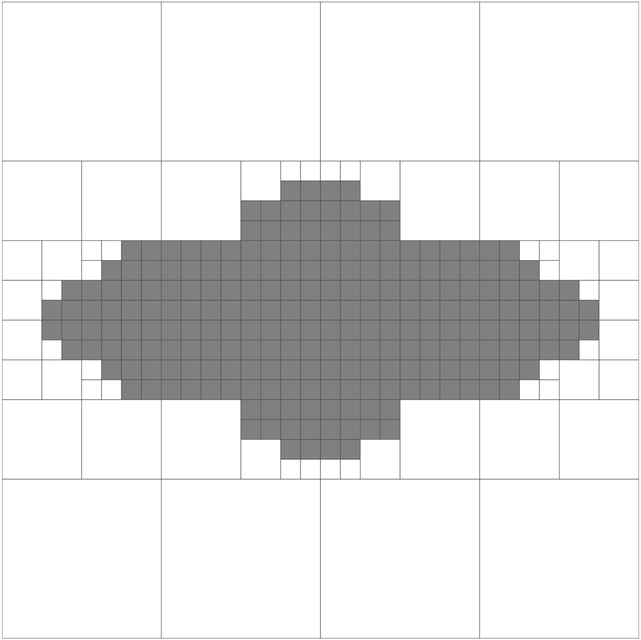
Quadtree $c = -1$ level 3



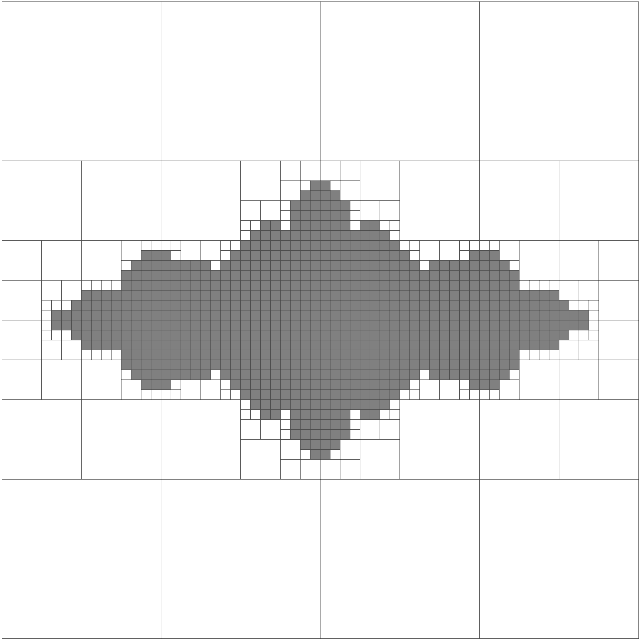
Quadtree $c = -1$ level 4



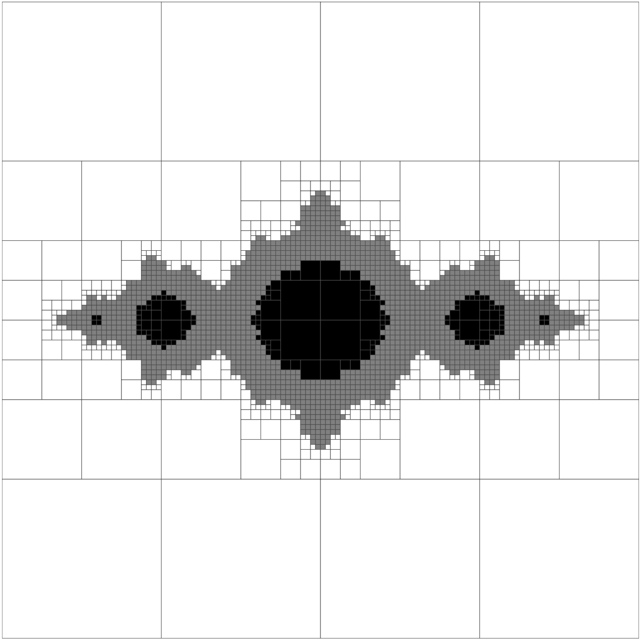
Quadtree $c = -1$ level 5



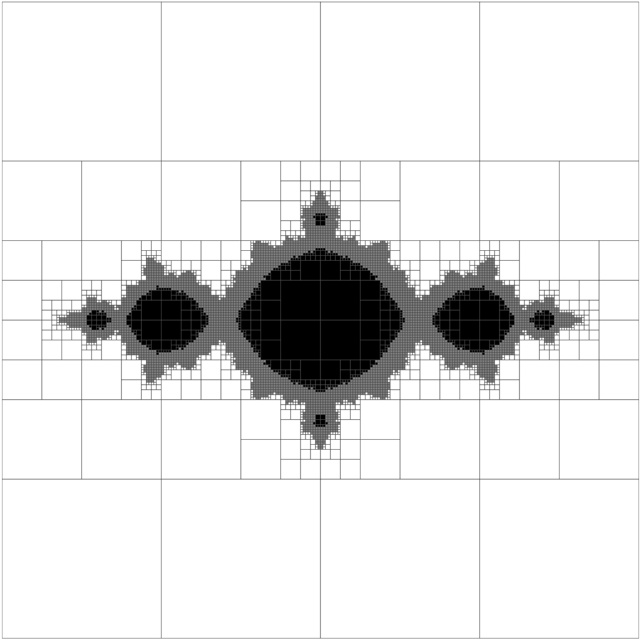
Quadtree $c = -1$ level 6



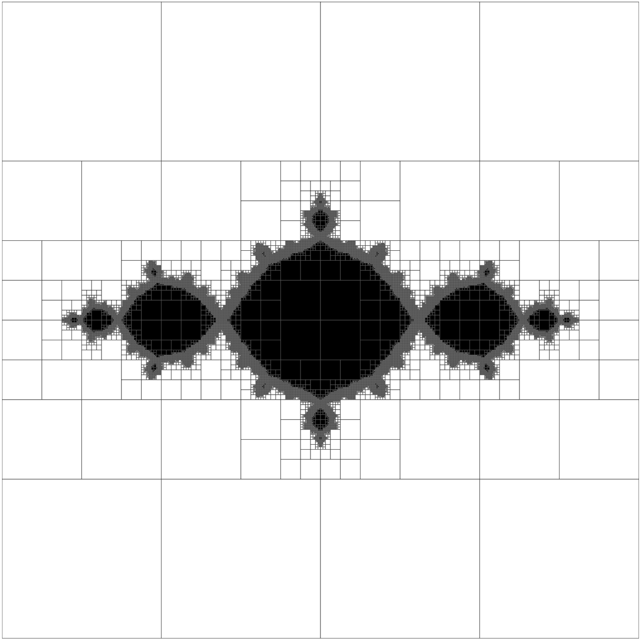
Quadtree $c = -1$ level 7



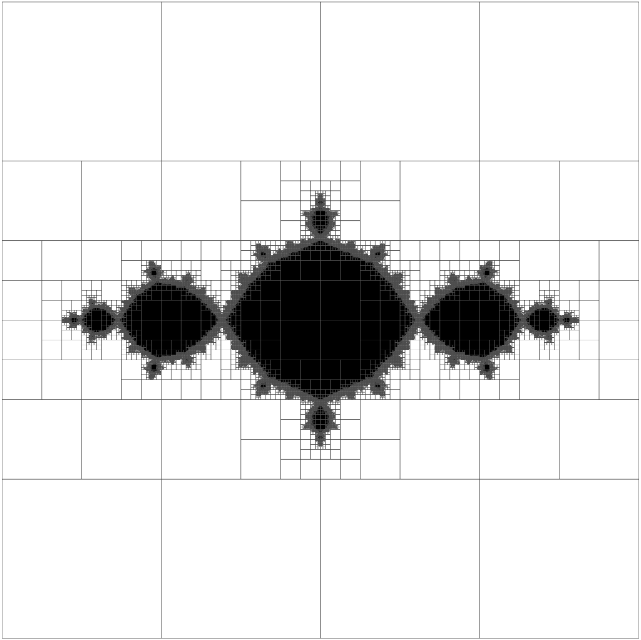
Quadtree $c = -1$ level 8



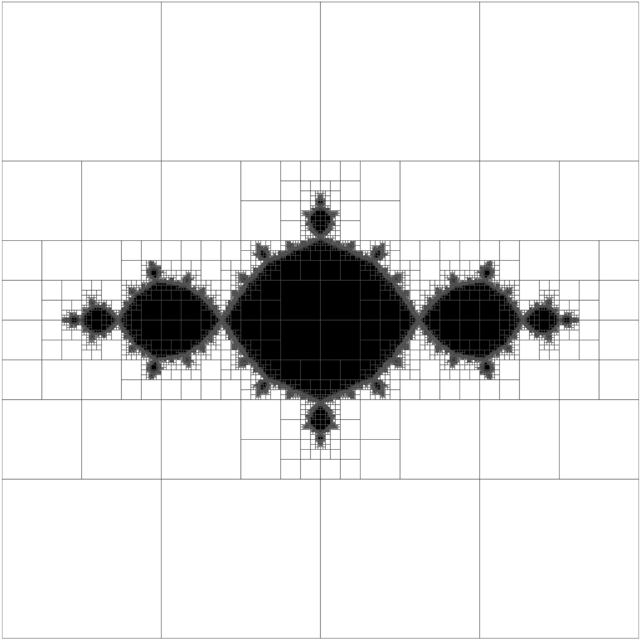
Quadtree $c = -1$ level 9



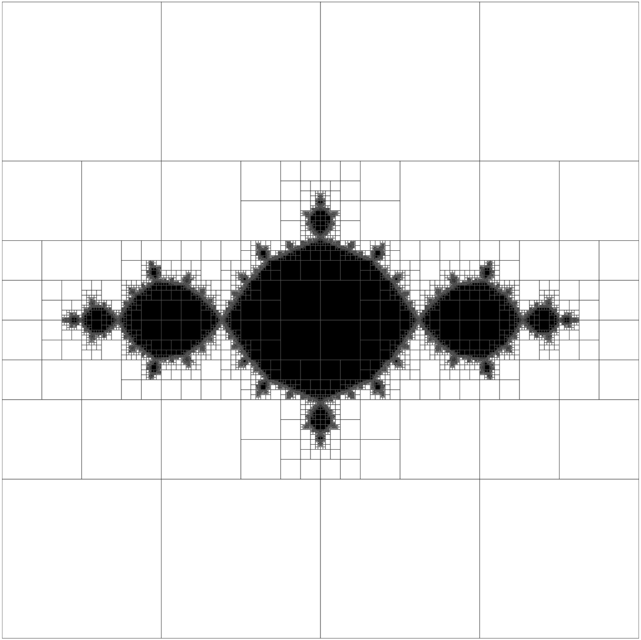
Quadtree $c = -1$ level 10



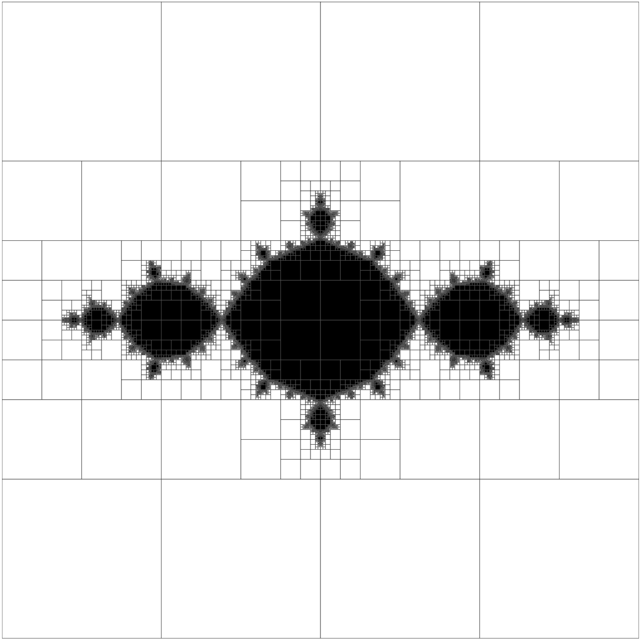
Quadtree $c = -1$ level 11



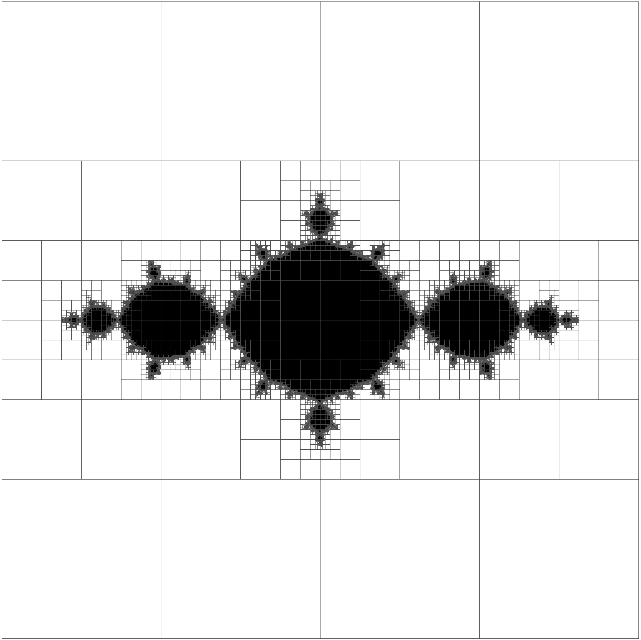
Quadtree $c = -1$ level 12



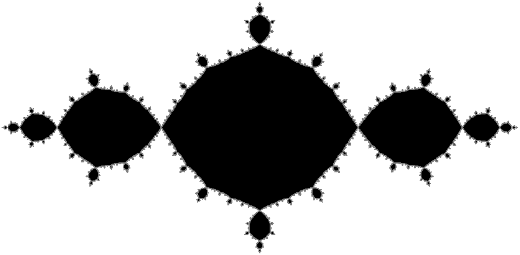
Quadtree $c = -1$ level 13



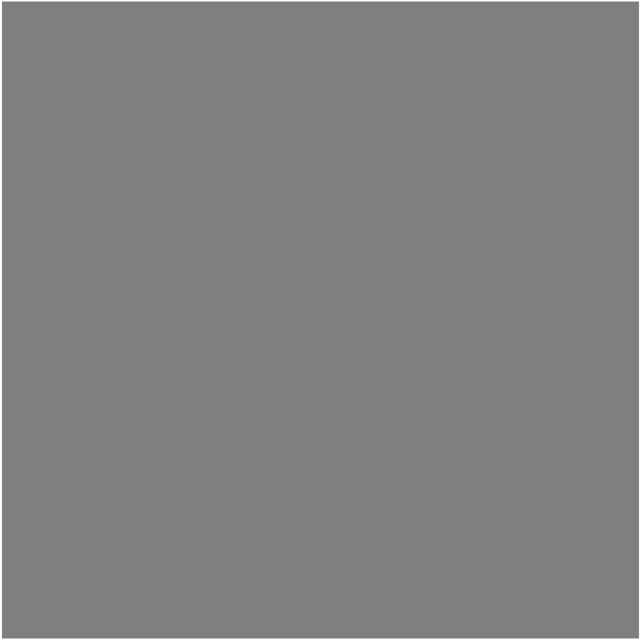
Quadtree $c = -1$ level 14



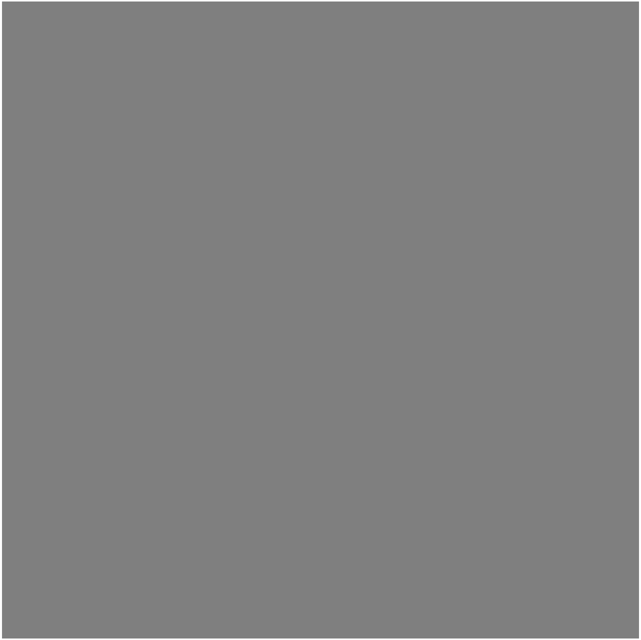
Adaptive approximation $c = -1$ level 14



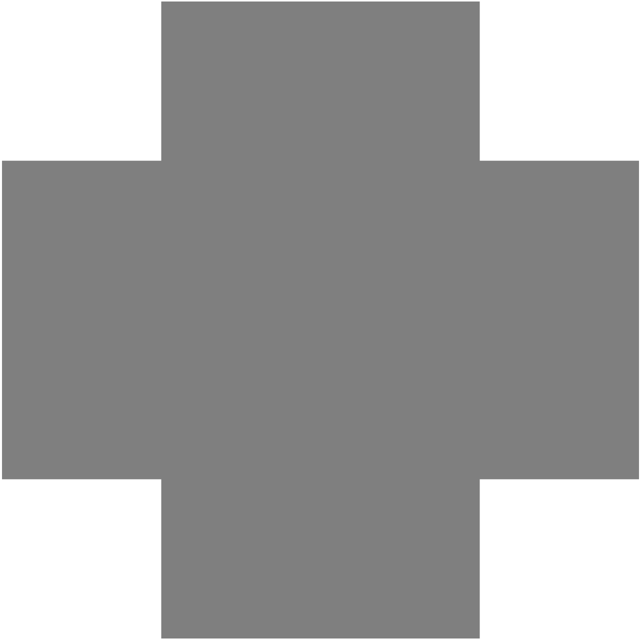
Adaptive approximation $c = -1$ level 0



Adaptive approximation $c = -1$ level 1



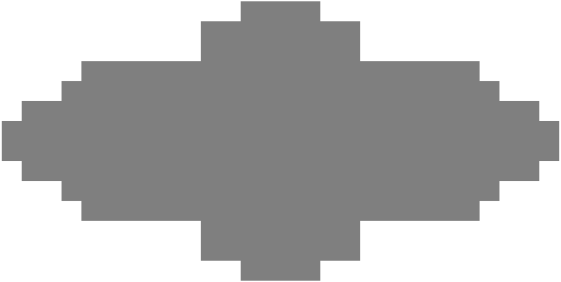
Adaptive approximation $c = -1$ level 2



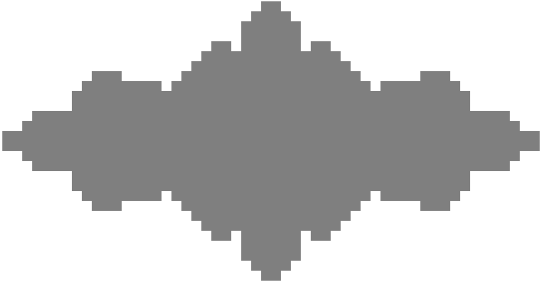
Adaptive approximation $c = -1$ level 3

Adaptive approximation $c = -1$ level 4

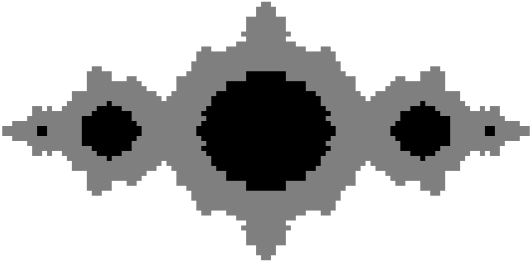
Adaptive approximation $c = -1$ level 5



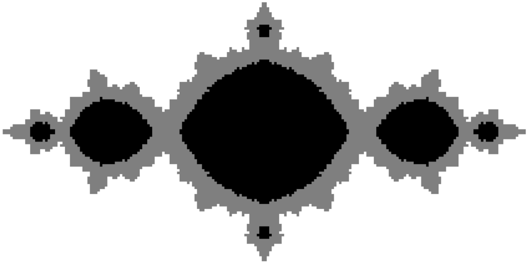
Adaptive approximation $c = -1$ level 6



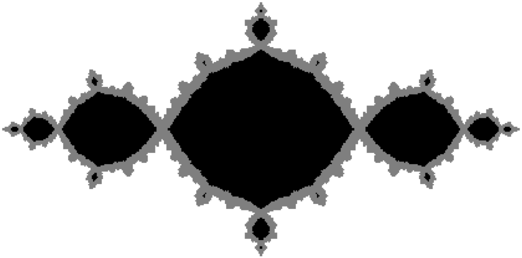
Adaptive approximation $c = -1$ level 7



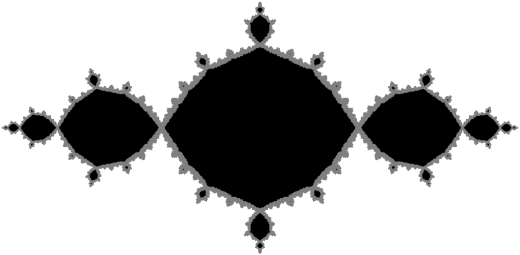
Adaptive approximation $c = -1$ level 8



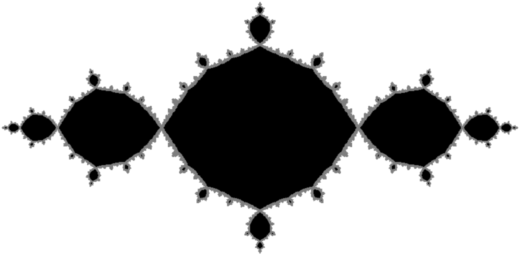
Adaptive approximation $c = -1$ level 9



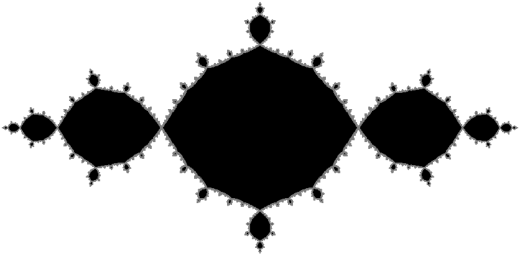
Adaptive approximation $c = -1$ level 10



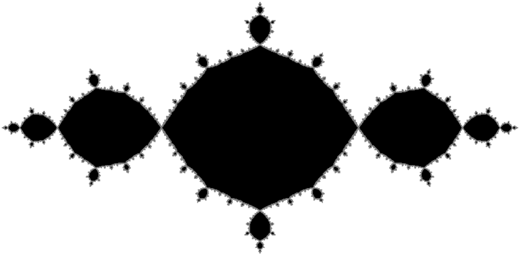
Adaptive approximation $c = -1$ level 11



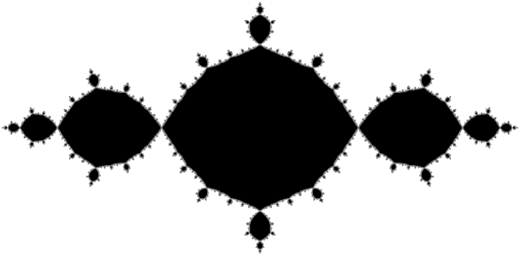
Adaptive approximation $c = -1$ level 12



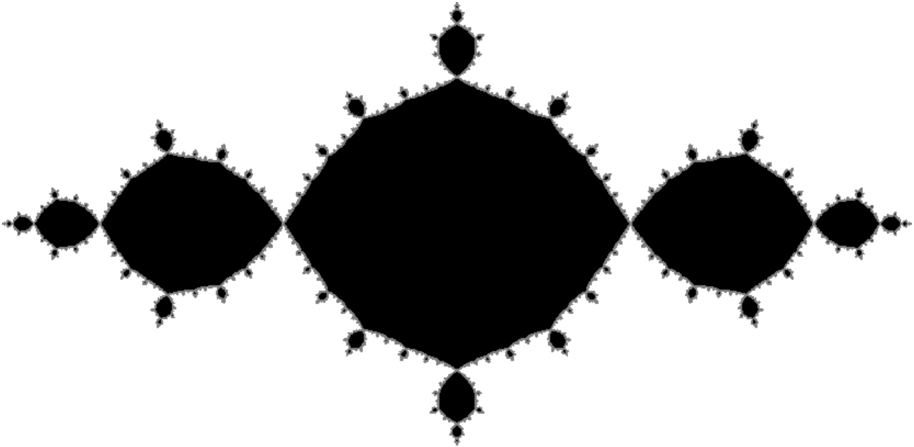
Adaptive approximation $c = -1$ level 13



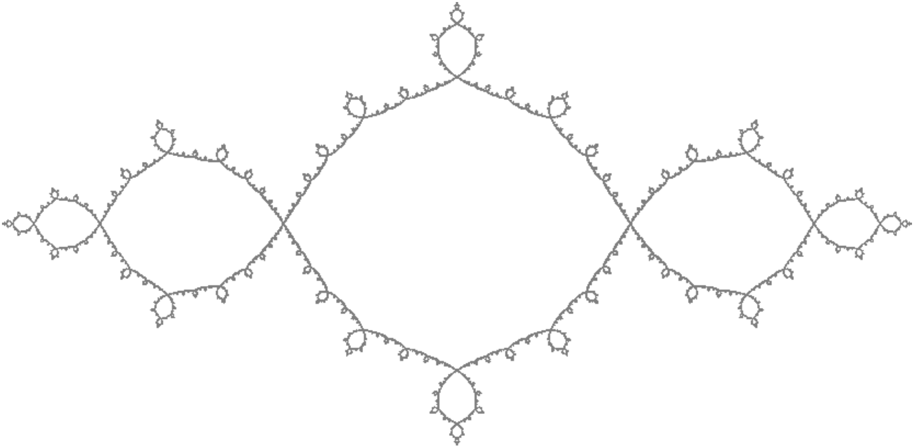
Adaptive approximation $c = -1$ level 14



Adaptive approximation $c = -1$



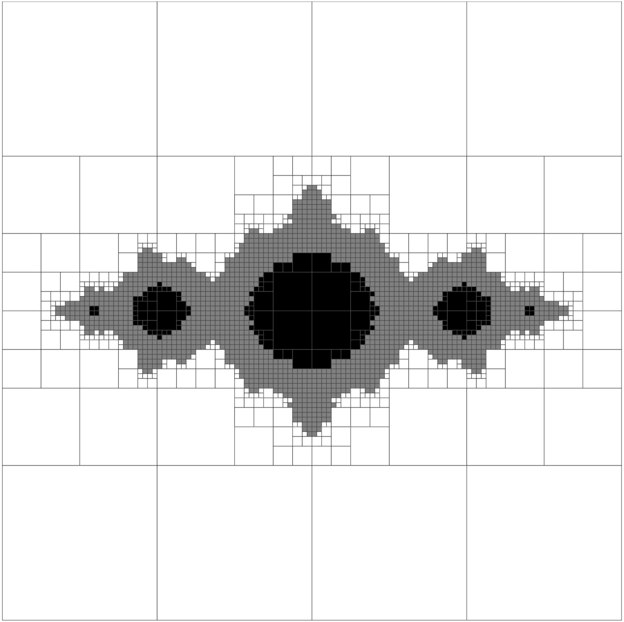
Adaptive approximation $\quad c = -1$



Our algorithm

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

- refinement
- cell mapping
- label propagation



Cell mapping

Directed graph on the leaves of the quadtree

- edges emanate from each leaf gray cell q
- color q white if $f(q)$ is outside $B(0,R)$
- add edge $q \to t$ for each leaf cell t that intersects $f(q)$

Cell mapping

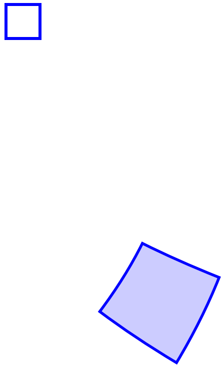
Directed graph on the leaves of the quadtree

- edges emanate from each leaf gray cell q
- color q white if $f(q)$ is outside $B(0, R)$
- add edge $q \to t$ for each leaf cell t that intersects $f(q)$

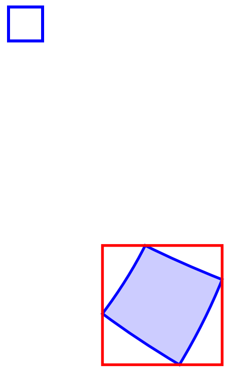
Conversative estimate of the dynamics

Avoid point sampling

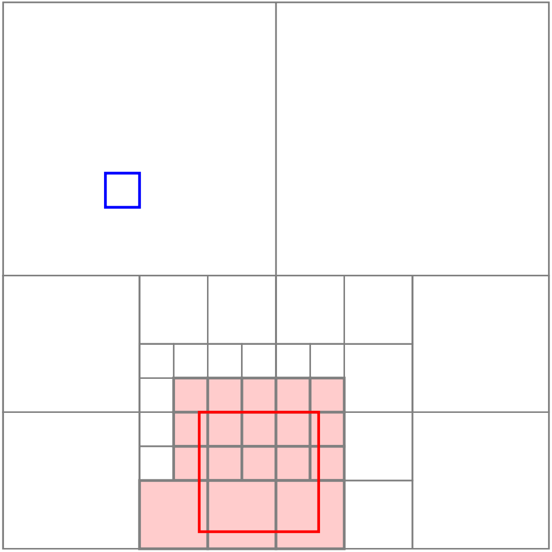
Cell mapping source cell leaf gray cell



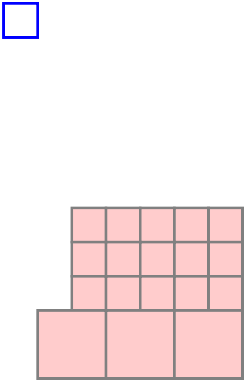
Cell mapping bounding box interval arithmetic



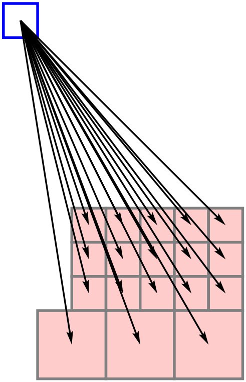
Cell mapping quadtree traversal



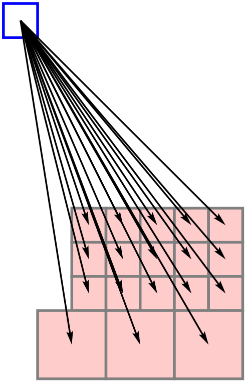
Cell mapping target cells contain exact image



Cell mapping edges



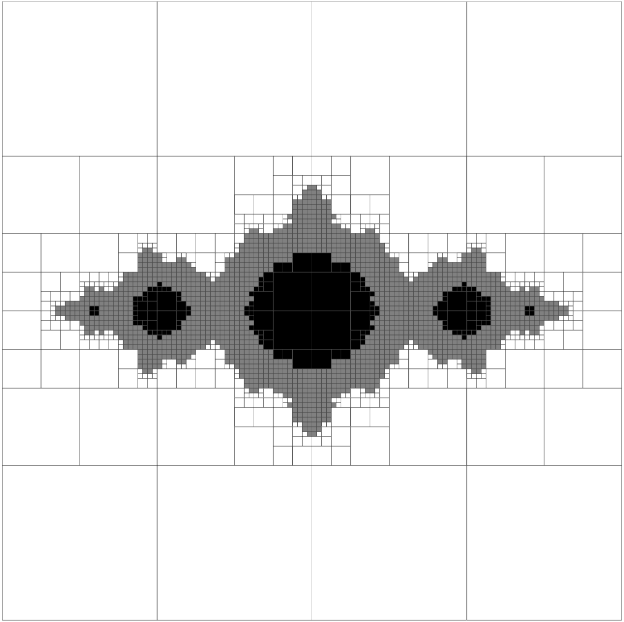
Cell mapping demo. . .



Our algorithm

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

- refinement
- cell mapping
- label propagation



Label 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

Label propagation

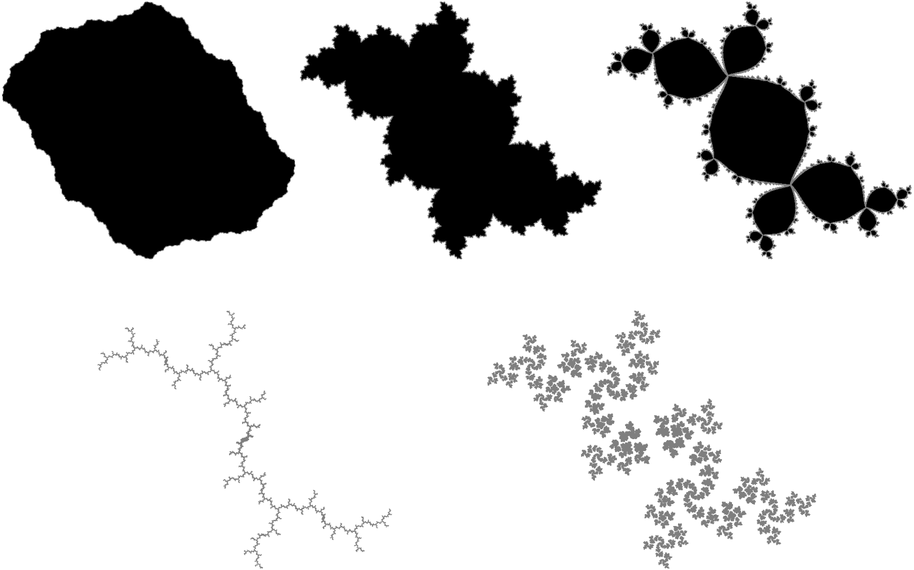
Propagate white and black to gray cells

- 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

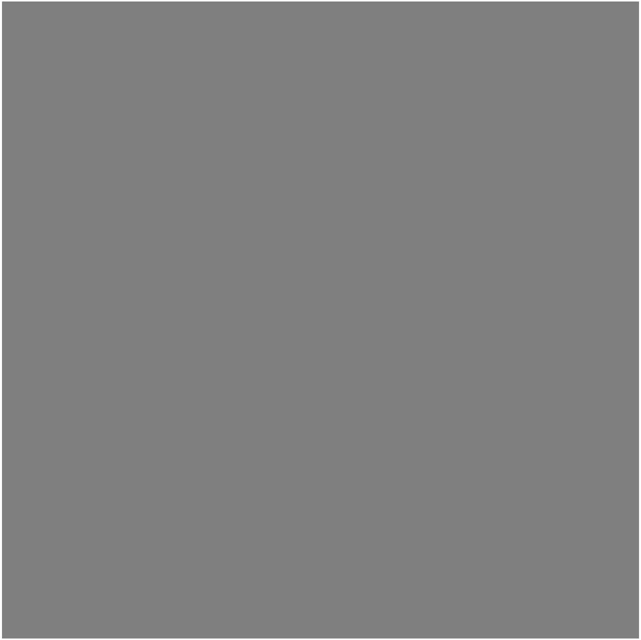
Graph traversals replace function iteration

Avoid floating-point errors

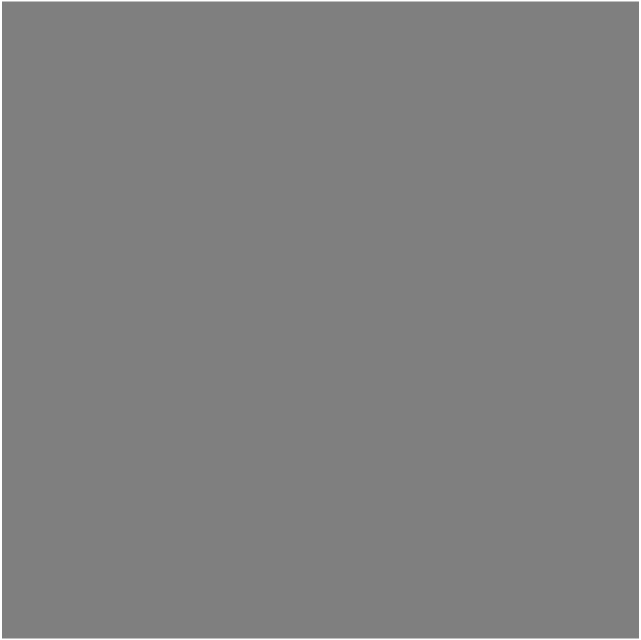
Adaptive approximation more examples



Adaptive approximation $c = 0.12 + 0.30\,i$ level 0



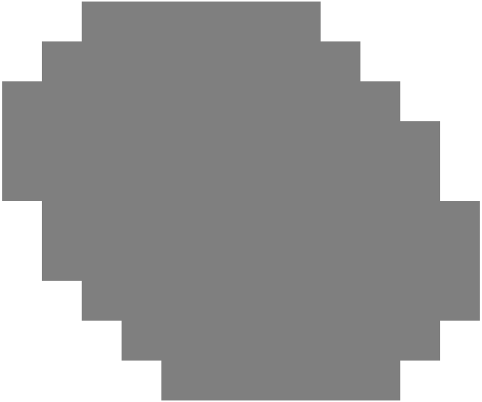
Adaptive approximation $\quad c = 0.12 + 0.30\,i \quad$ level 1



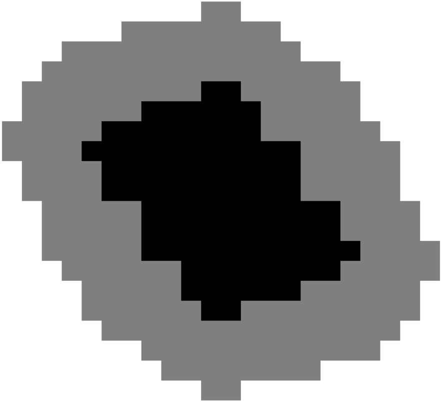
Adaptive approximation $c = 0.12 + 0.30\,i$ level 2

Adaptive approximation $c = 0.12 + 0.30\,i$ level 3

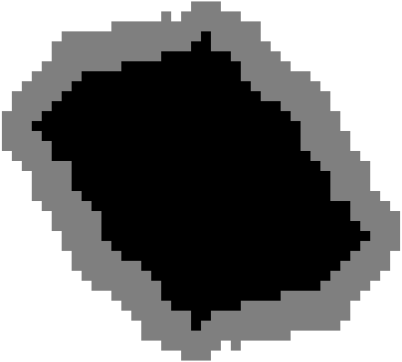
Adaptive approximation $c = 0.12 + 0.30\,i$ level 4



Adaptive approximation $c = 0.12 + 0.30\, i$ level 5

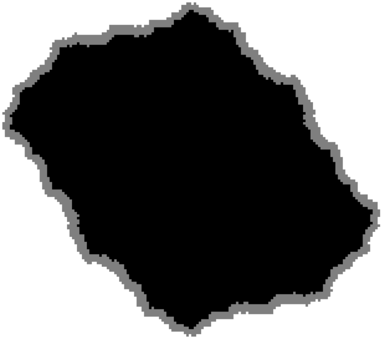


Adaptive approximation $\quad c = 0.12 + 0.30\,i \quad$ level 6

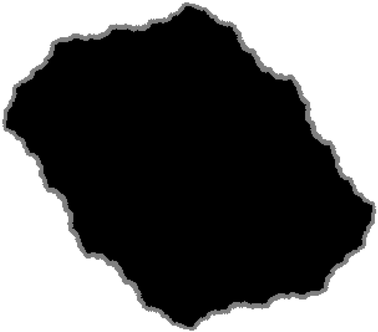


Adaptive approximation $c = 0.12 + 0.30\,i$ level 7

Adaptive approximation $c = 0.12 + 0.30\,i$ level 8



Adaptive approximation $c = 0.12 + 0.30\,i$ level 9



Adaptive approximation $c = 0.12 + 0.30\,i$ level 10

Adaptive approximation $\quad c = 0.12 + 0.30\,i \quad$ level 11

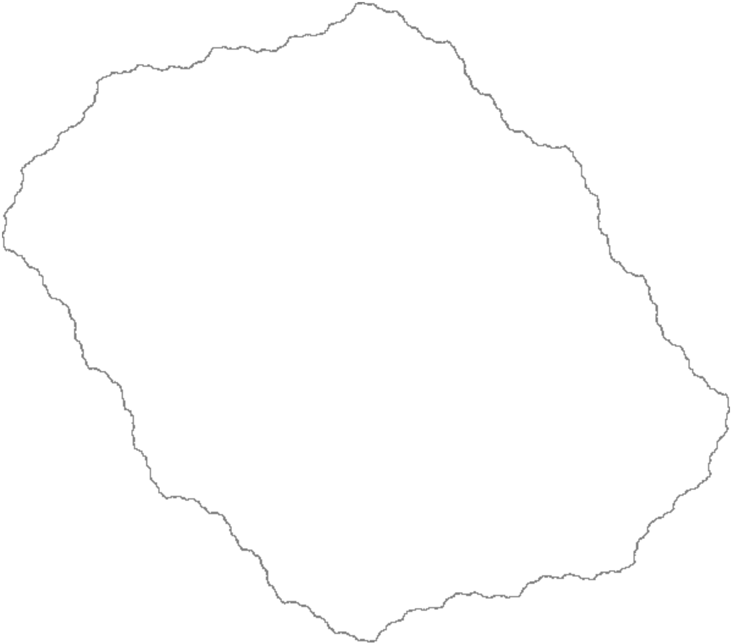
Adaptive approximation $c = 0.12 + 0.30\,i$ level 12

Adaptive approximation $c = 0.12 + 0.30\,i$ level 13

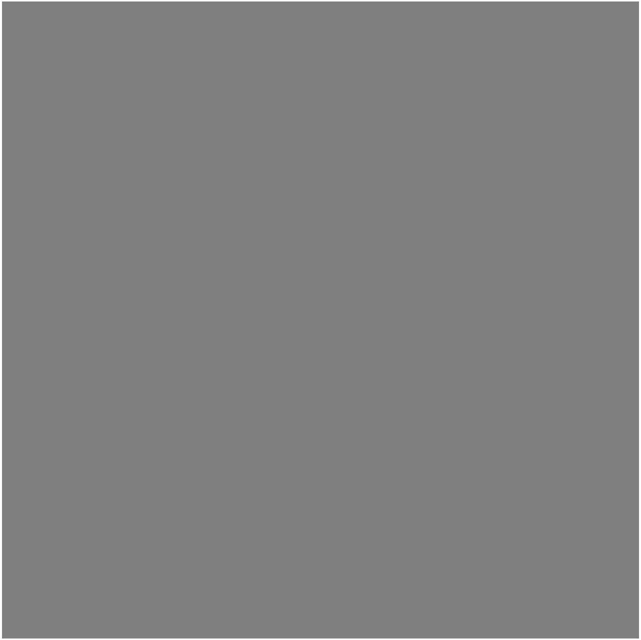
Adaptive approximation $c = 0.12 + 0.30\,i$ level 14

Adaptive approximation $c = 0.12 + 0.30\,i$

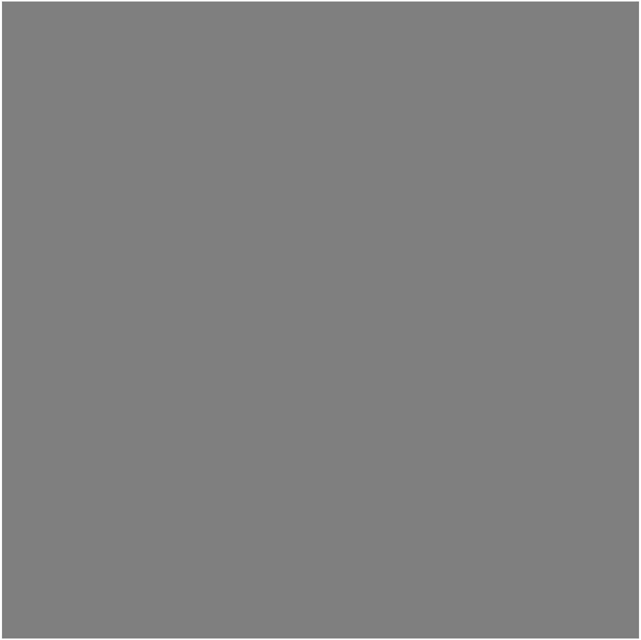
Adaptive approximation $c = 0.12 + 0.30\,i$



Adaptive approximation $c = -0.12 + 0.60\,i$ level 0

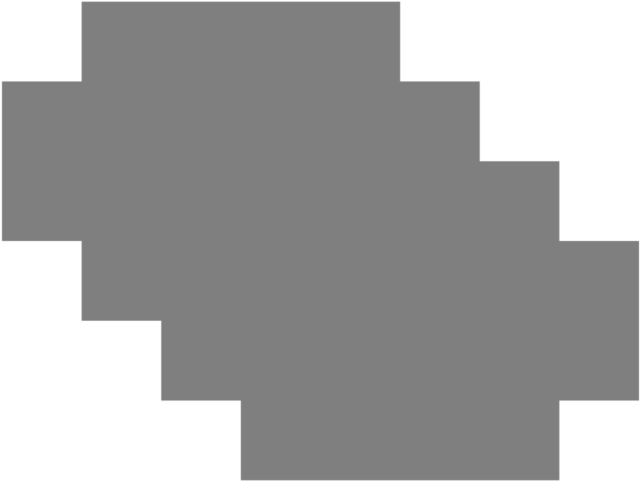


Adaptive approximation $c = -0.12 + 0.60\,i$ level 1

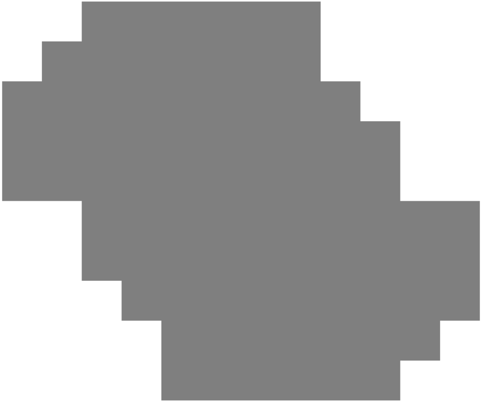


Adaptive approximation $c = -0.12 + 0.60\,i$ level 2

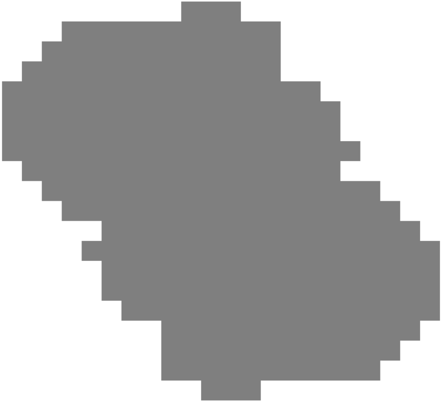
Adaptive approximation $c = -0.12 + 0.60\,i$ level 3



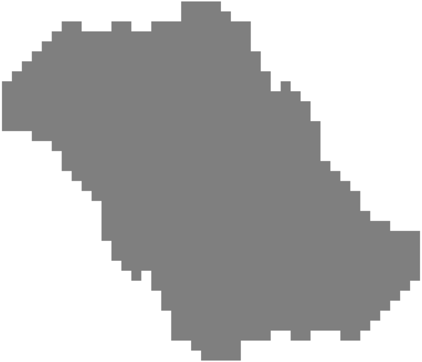
Adaptive approximation $c = -0.12 + 0.60\,i$ level 4



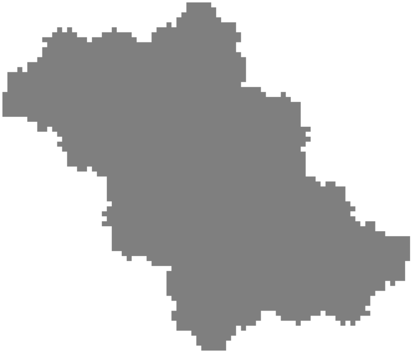
Adaptive approximation $c = -0.12 + 0.60\,i$ level 5



Adaptive approximation $c = -0.12 + 0.60\,i$ level 6

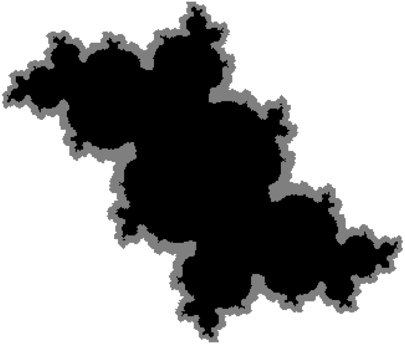


Adaptive approximation $c = -0.12 + 0.60\, i$ level 7

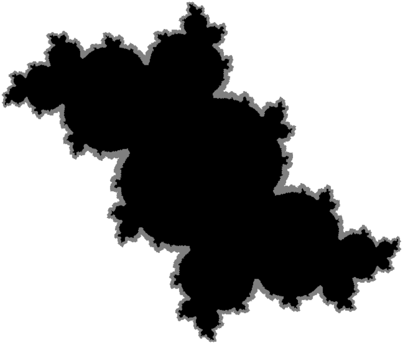


Adaptive approximation $c = -0.12 + 0.60\,i$ level 8

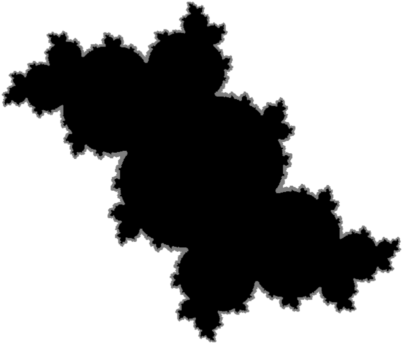
Adaptive approximation $c = -0.12 + 0.60\,i$ level 9



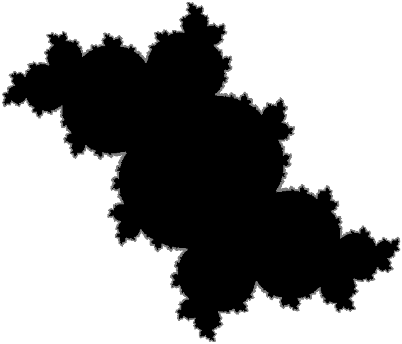
Adaptive approximation $c = -0.12 + 0.60\, i$ level 10



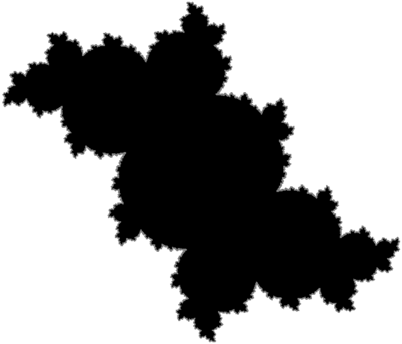
Adaptive approximation $c = -0.12 + 0.60\,i$ level 11



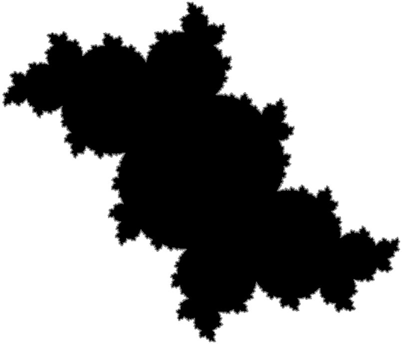
Adaptive approximation $c = -0.12 + 0.60\,i$ level 12



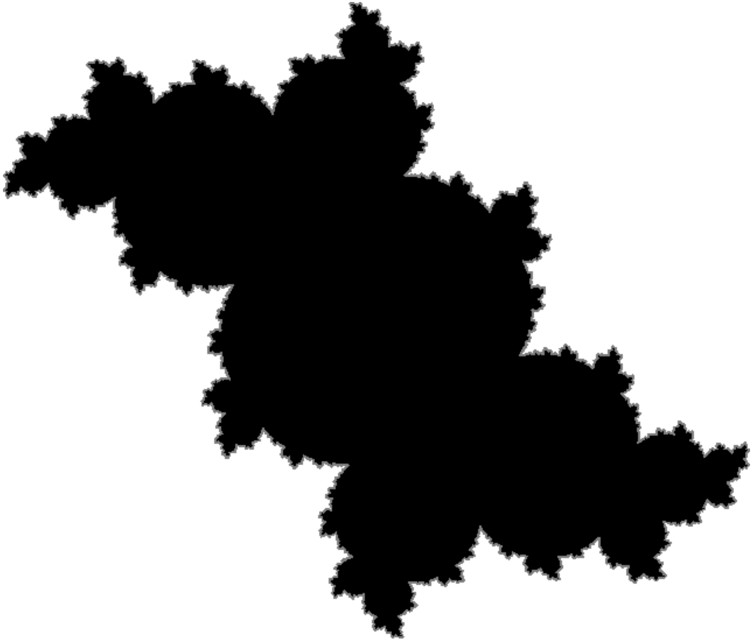
Adaptive approximation $c = -0.12 + 0.60\,i$ level 13



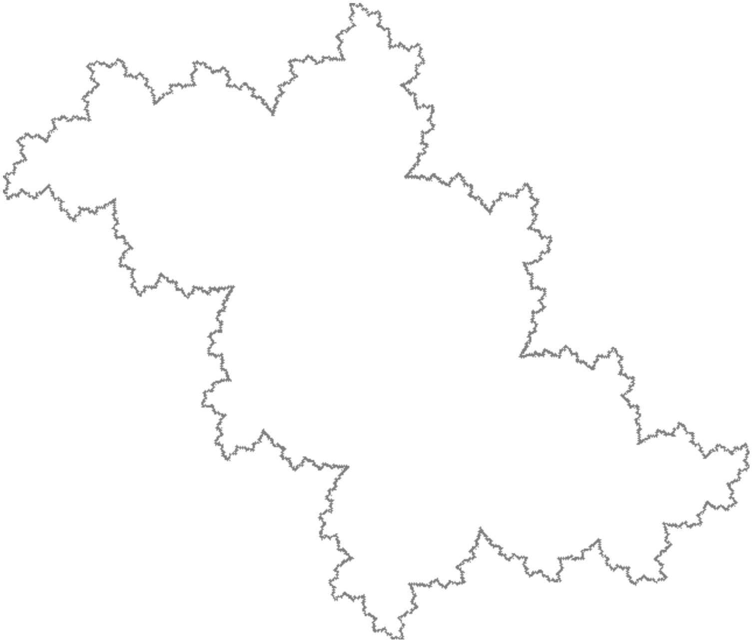
Adaptive approximation $c = -0.12 + 0.60\,i$ level 14



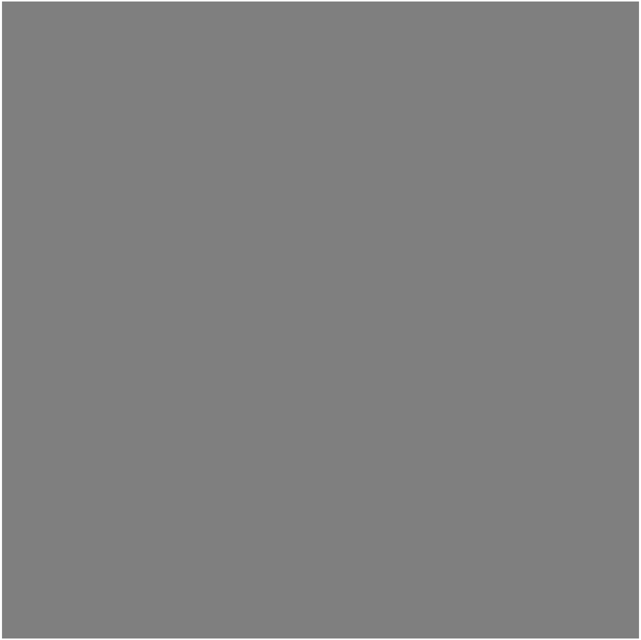
Adaptive approximation $\quad c = -0.12 + 0.60\,i$



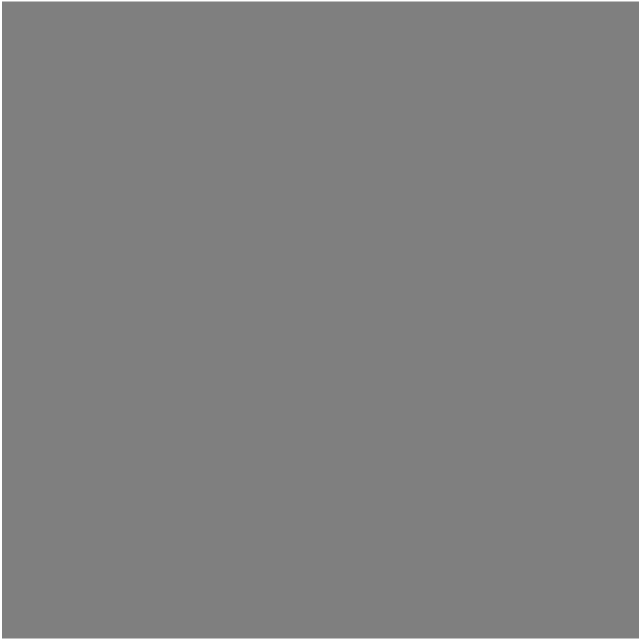
Adaptive approximation $c = -0.12 + 0.60\,i$



Adaptive approximation $c = -0.12 + 0.74\,i$ level 0

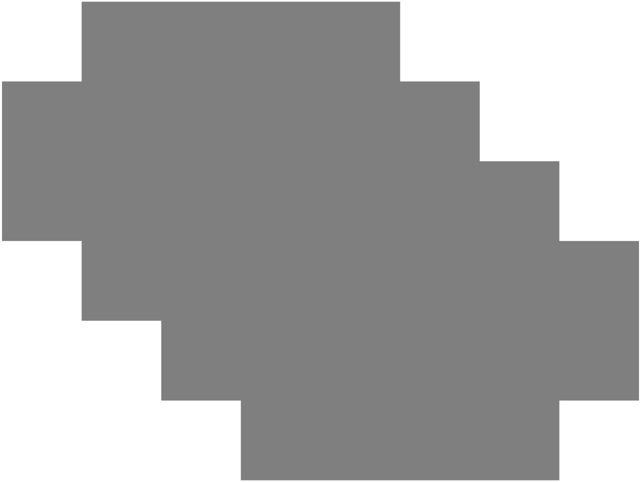


Adaptive approximation $c = -0.12 + 0.74\,i$ level 1

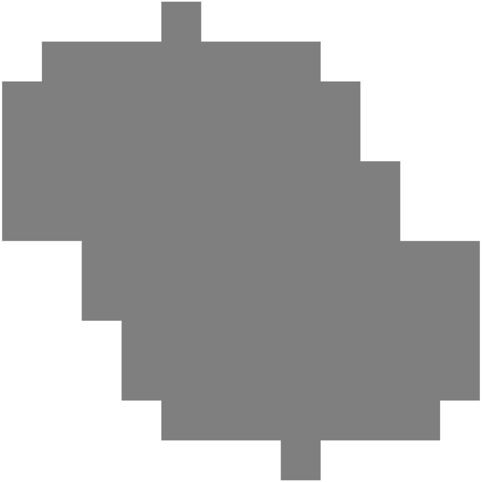


Adaptive approximation $c = -0.12 + 0.74\, i$ level 2

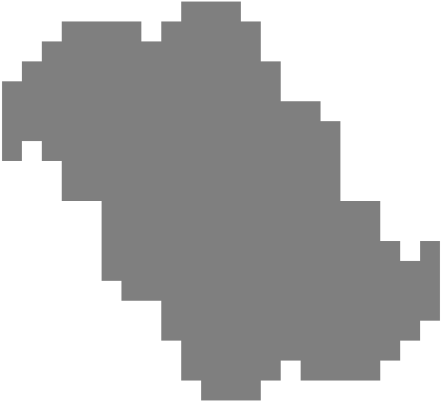
Adaptive approximation $c = -0.12 + 0.74\,i$ level 3



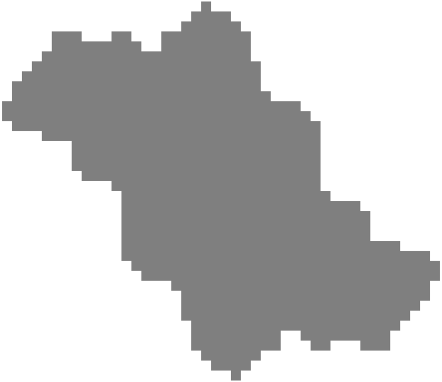
Adaptive approximation $c = -0.12 + 0.74\,i$ level 4



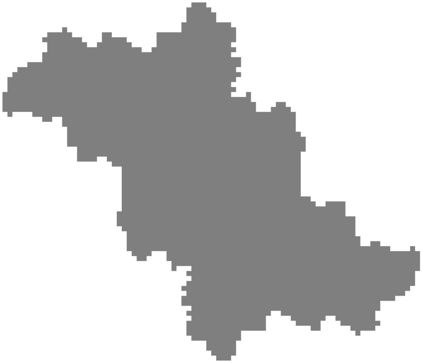
Adaptive approximation $c = -0.12 + 0.74\,i$ level 5



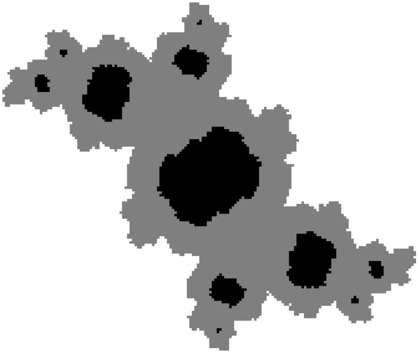
Adaptive approximation $c = -0.12 + 0.74\,i$ level 6



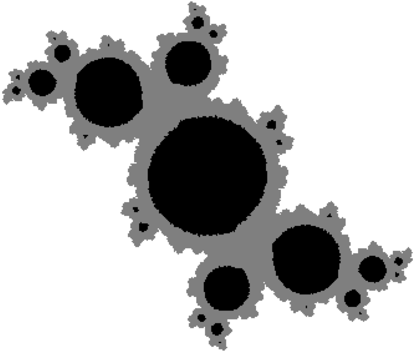
Adaptive approximation $c = -0.12 + 0.74\,i$ level 7



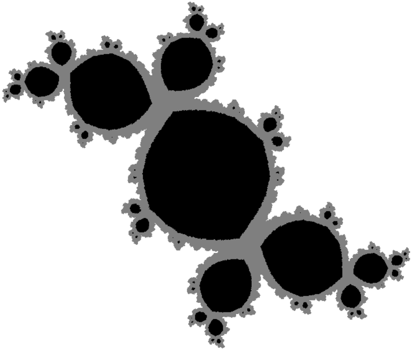
Adaptive approximation $\quad c = -0.12 + 0.74\,i \quad$ level 8



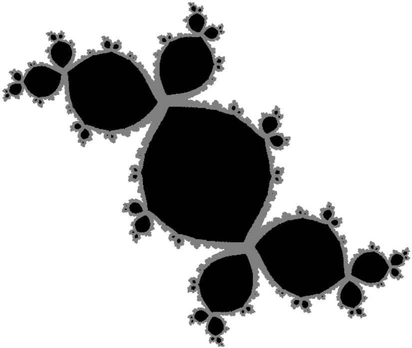
Adaptive approximation $c = -0.12 + 0.74\,i$ level 9



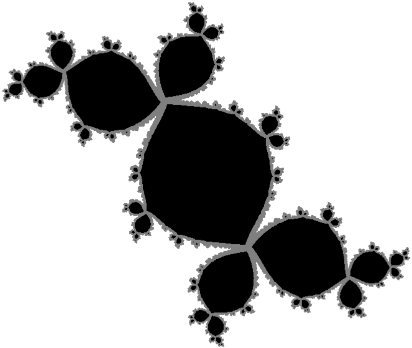
Adaptive approximation $c = -0.12 + 0.74\,i$ level 10



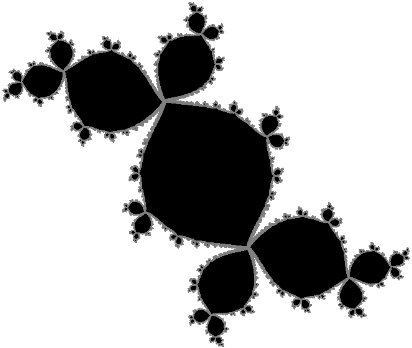
Adaptive approximation $c = -0.12 + 0.74\,i$ level 11



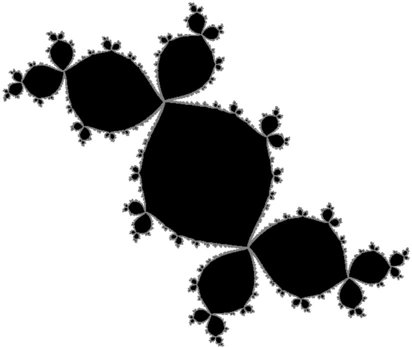
Adaptive approximation $c = -0.12 + 0.74\,i$ level 12



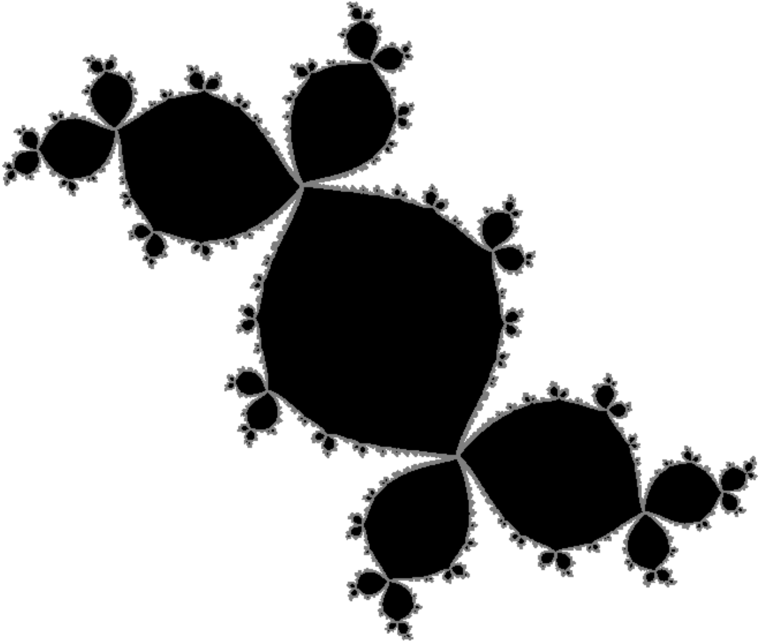
Adaptive approximation $c = -0.12 + 0.74\,i$ level 13



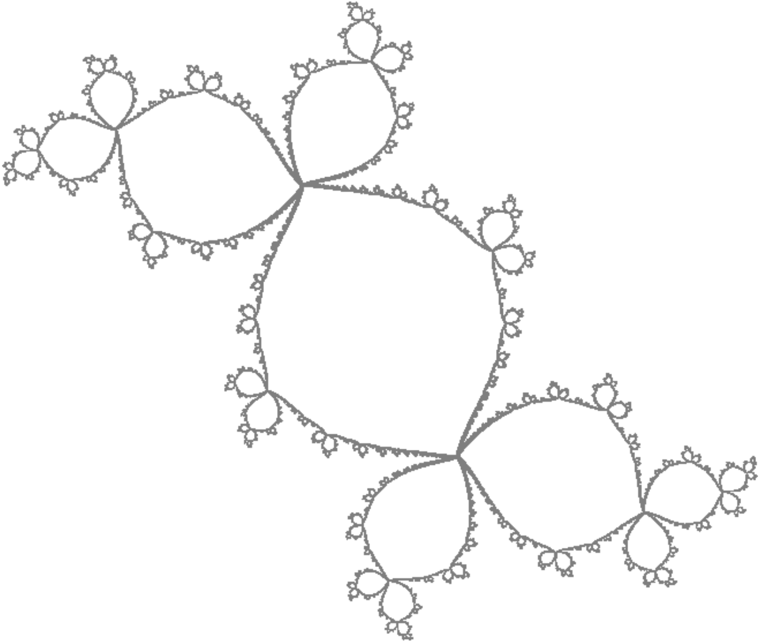
Adaptive approximation $c = -0.12 + 0.74\,i$ level 14



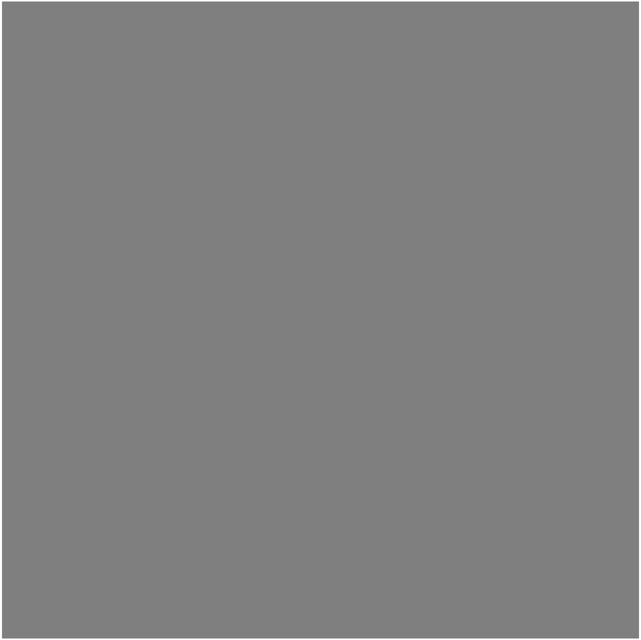
Adaptive approximation $c = -0.12 + 0.74\,i$



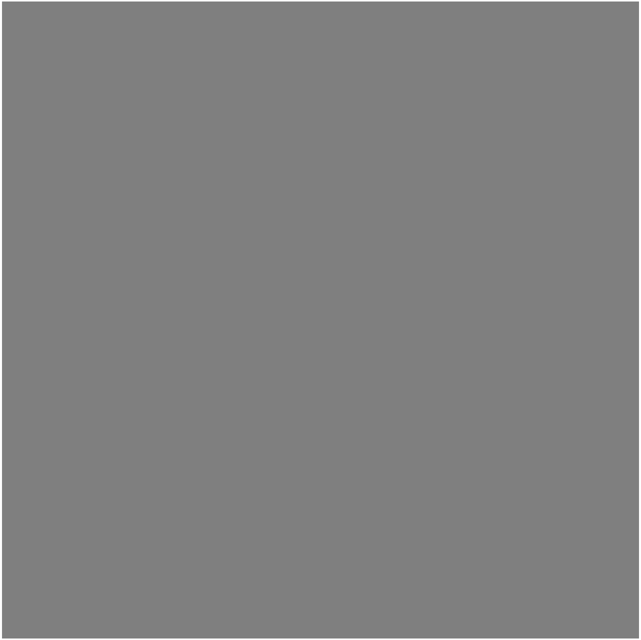
Adaptive approximation $c = -0.12 + 0.74\,i$



Adaptive approximation $\quad c = i \quad$ level 0

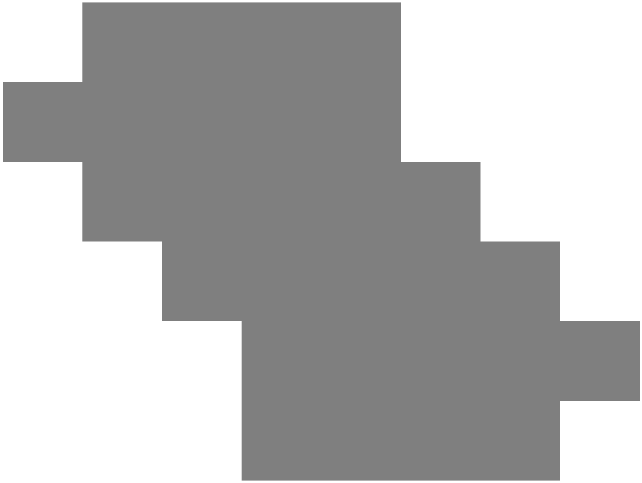


Adaptive approximation $c = i$ level 1

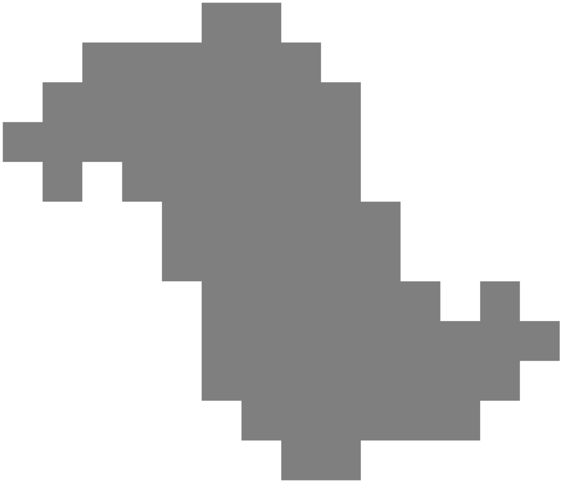


Adaptive approximation $c = i$ level 2

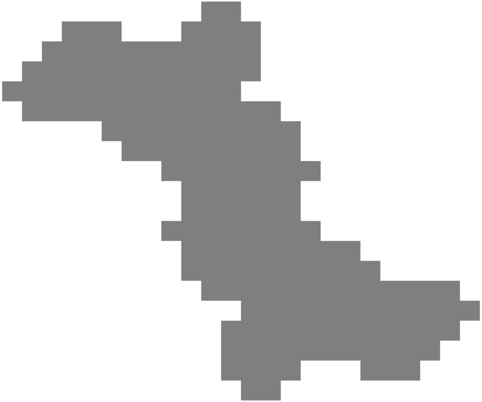
Adaptive approximation $c = i$ level 3



Adaptive approximation $c = i$ level 4

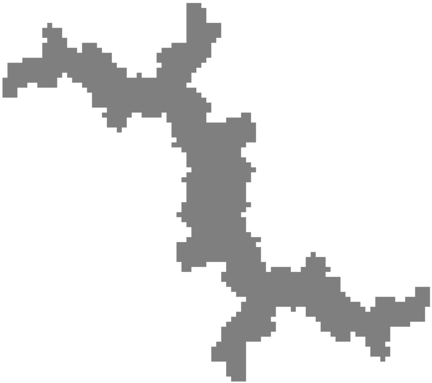


Adaptive approximation $c = i$ level 5

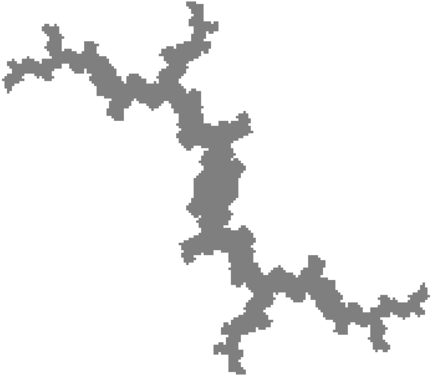


Adaptive approximation $\quad c = i \quad$ level 6

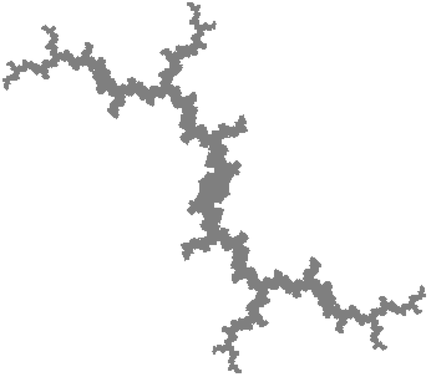
Adaptive approximation $\quad c = i \quad$ level 7



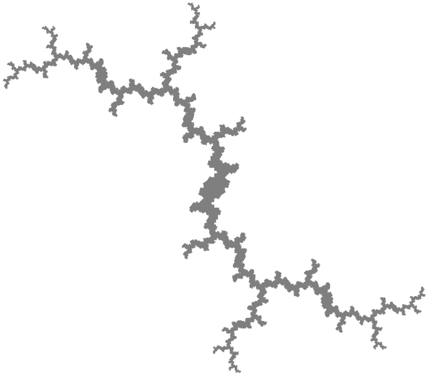
Adaptive approximation $c = i$ level 8



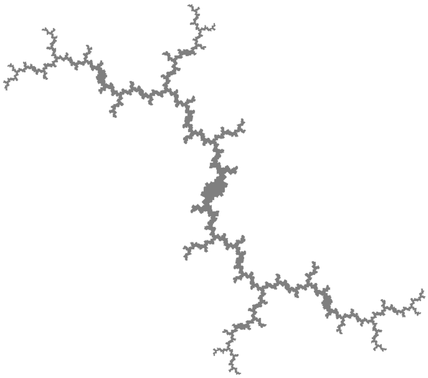
Adaptive approximation $c = i$ level 9



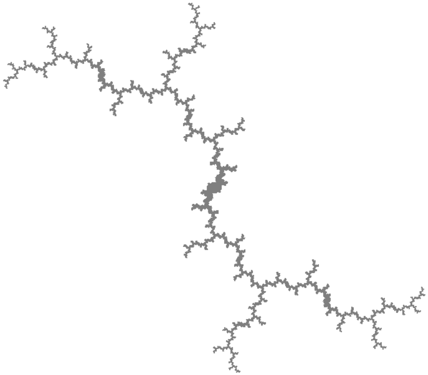
Adaptive approximation $c = i$ level 10



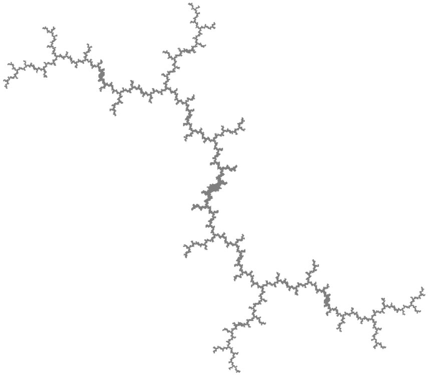
Adaptive approximation $c = i$ level 11



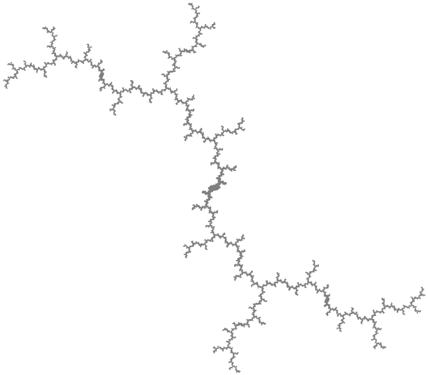
Adaptive approximation $c = i$ level 12



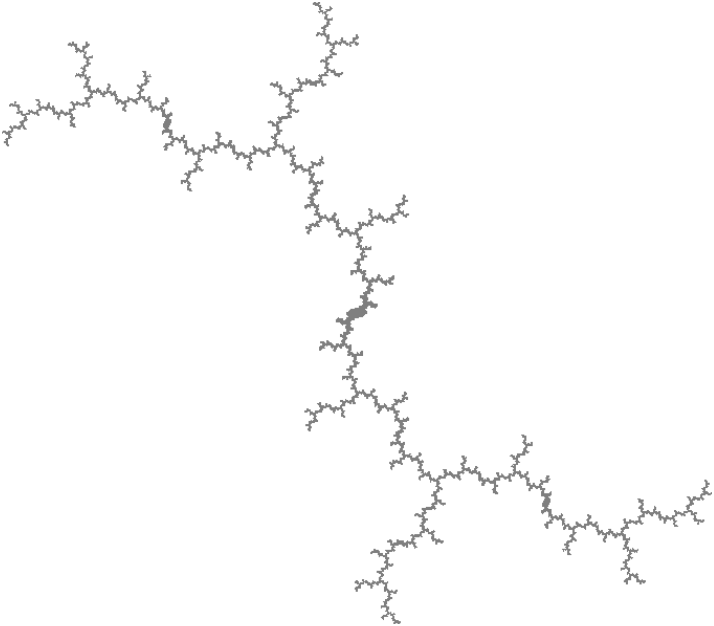
Adaptive approximation $c = i$ level 13



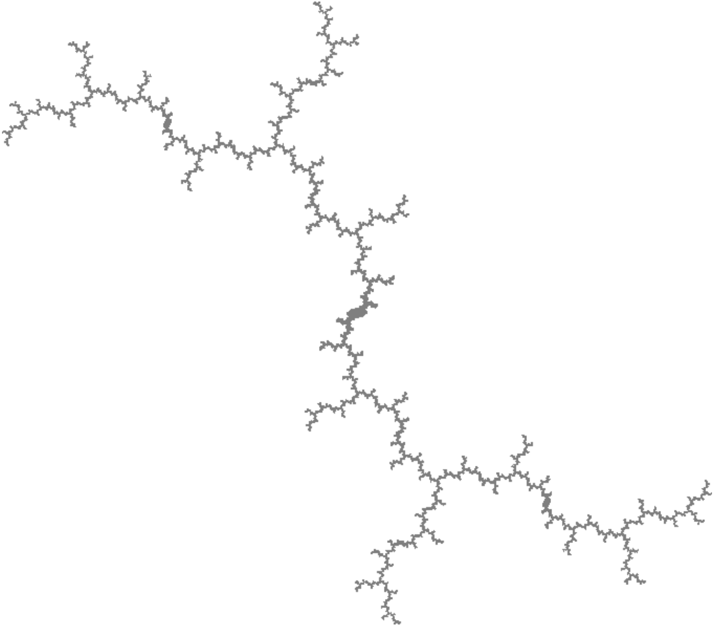
Adaptive approximation $c = i$ level 14



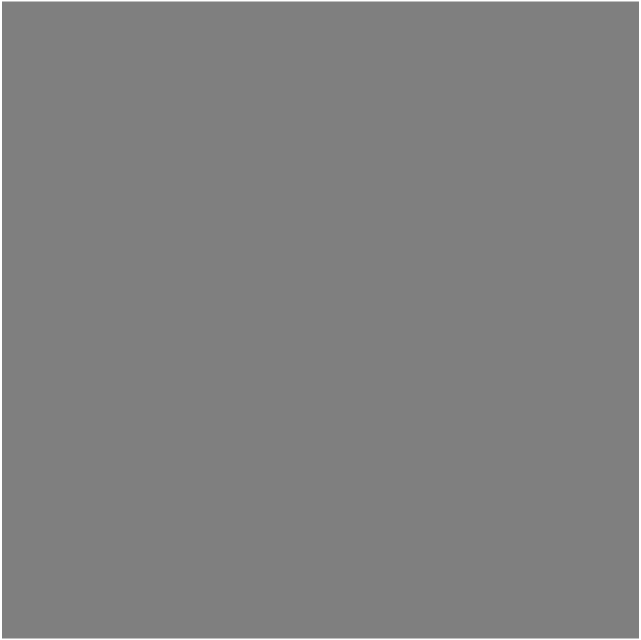
Adaptive approximation $c = i$



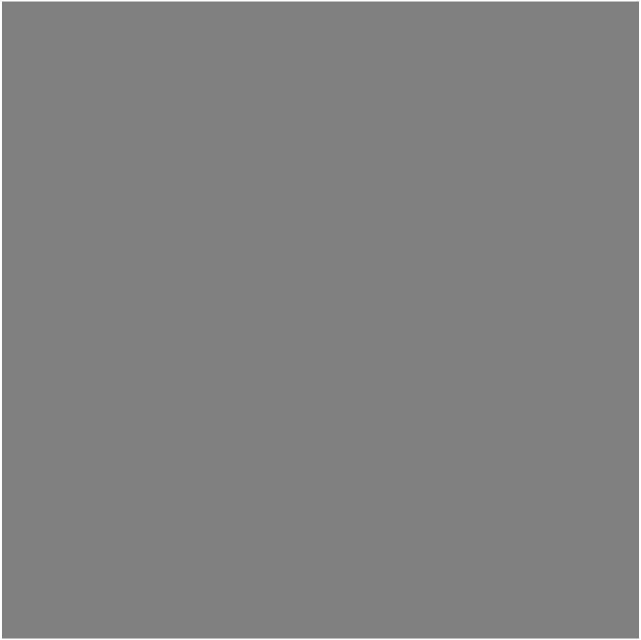
Adaptive approximation $c = i$



Adaptive approximation $c = -0.25 + 0.74\,i$ level 0

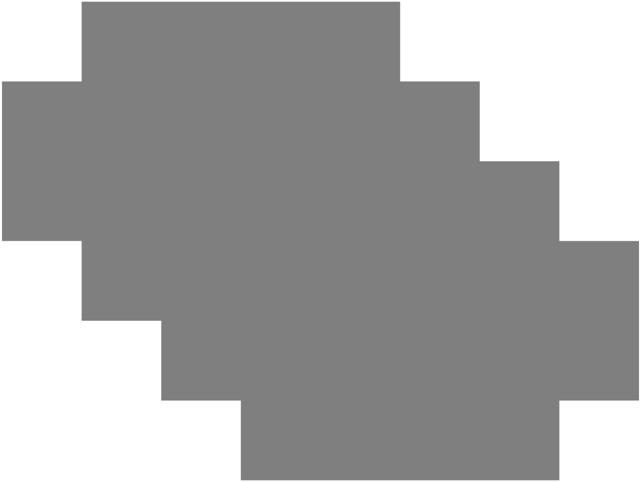


Adaptive approximation $c = -0.25 + 0.74\,i$ level 1

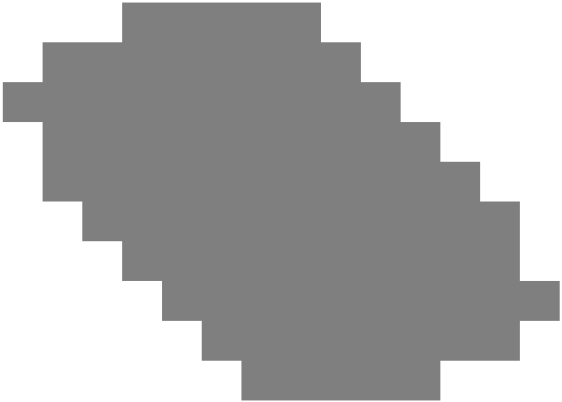


Adaptive approximation $\quad c = -0.25 + 0.74\,i \quad$ level 2

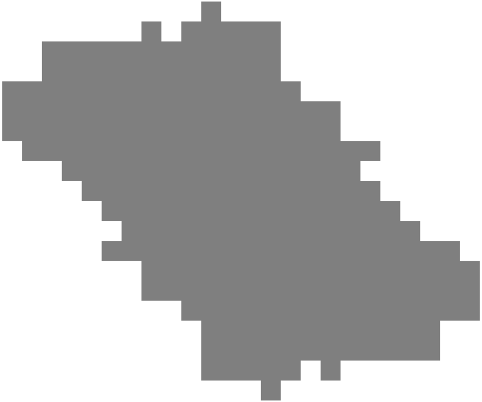
Adaptive approximation $c = -0.25 + 0.74\,i$ level 3



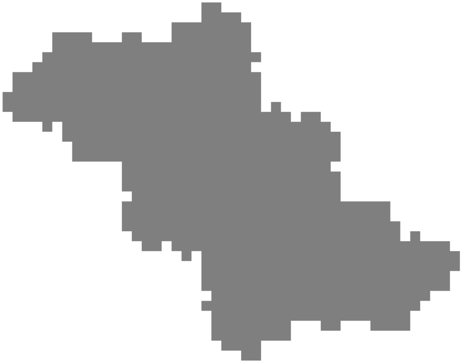
Adaptive approximation $c = -0.25 + 0.74\,i$ level 4



Adaptive approximation $c = -0.25 + 0.74\,i$ level 5

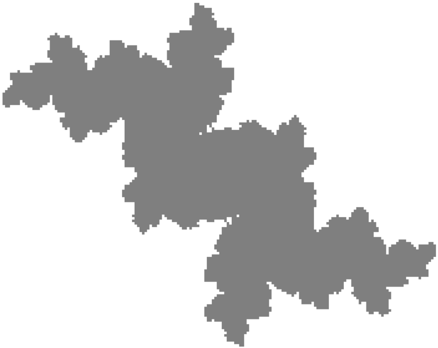


Adaptive approximation $c = -0.25 + 0.74\,i$ level 6

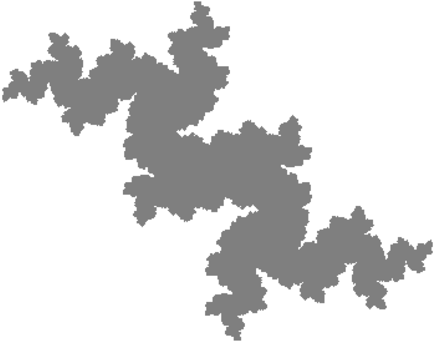


Adaptive approximation $c = -0.25 + 0.74\,i$ level 7

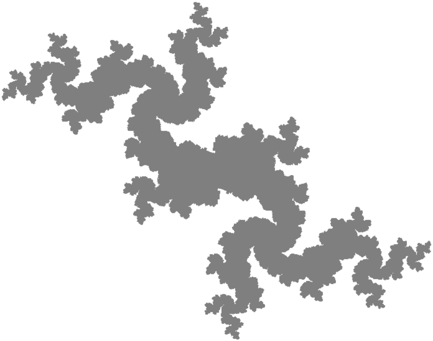
Adaptive approximation $c = -0.25 + 0.74\,i$ level 8



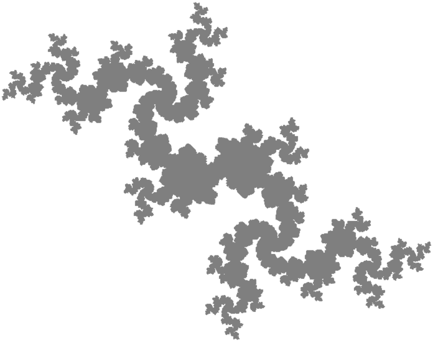
Adaptive approximation $c = -0.25 + 0.74\,i$ level 9



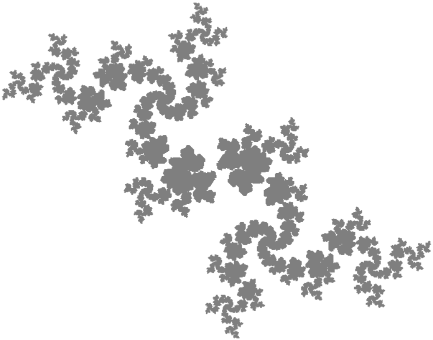
Adaptive approximation $c = -0.25 + 0.74\,i$ level 10



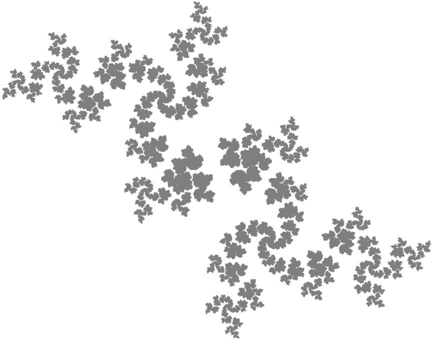
Adaptive approximation $c = -0.25 + 0.74\,i$ level 11



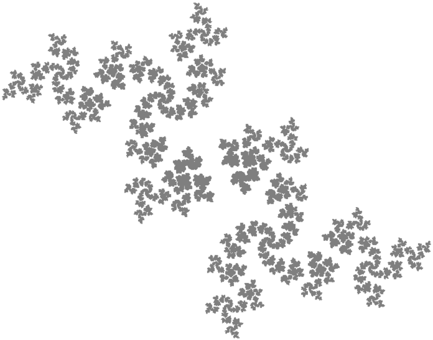
Adaptive approximation $c = -0.25 + 0.74\,i$ level 12



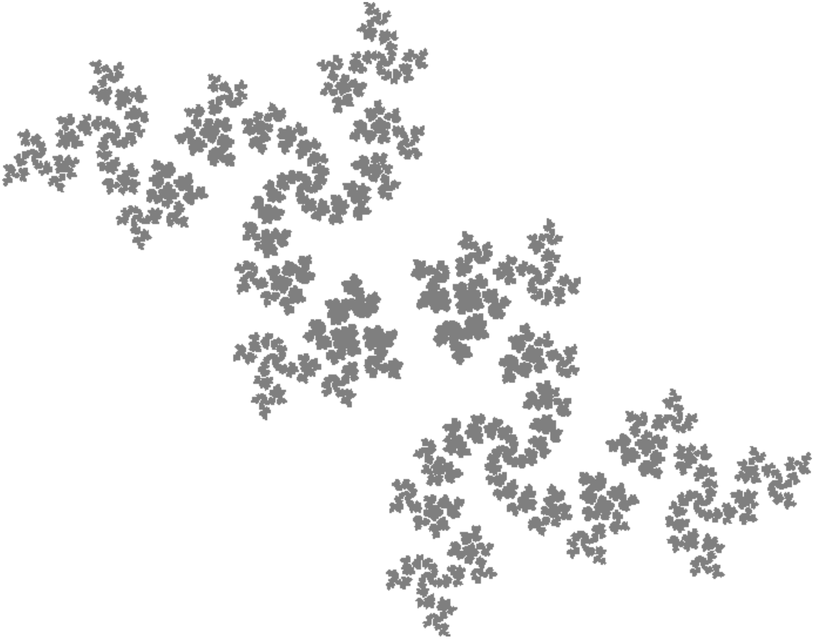
Adaptive approximation $c = -0.25 + 0.74\,i$ level 13



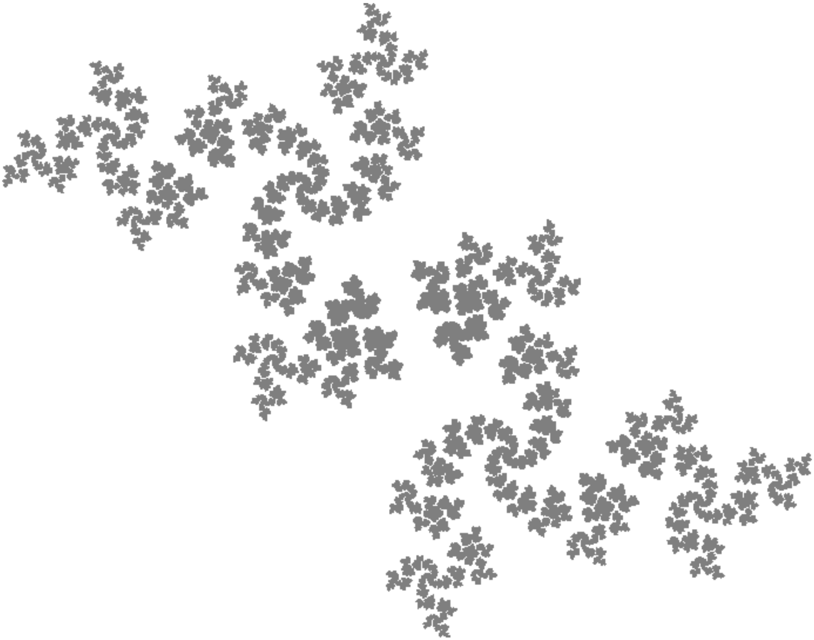
Adaptive approximation $c = -0.25 + 0.74\,i$ level 14



Adaptive approximation $c = -0.25 + 0.74\,i$



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 (Ruelle)
  upper bound $\dim_H = 1 + \frac{|c|^2}{4\log 2} + \cdots$

- Area of filled Julia set (Milnor)
  lower and upper bounds $\pi(1 - |p_1(c)|^2 - 3|p_2(c)|^2 - 5|p_1(c)|^2 - \cdots)$

- Diameter of Julia set
  lower and upper bounds

Area of filled Julia set after Milnor

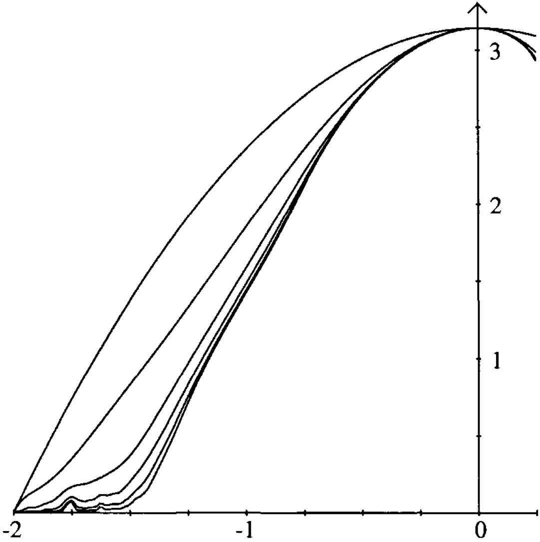
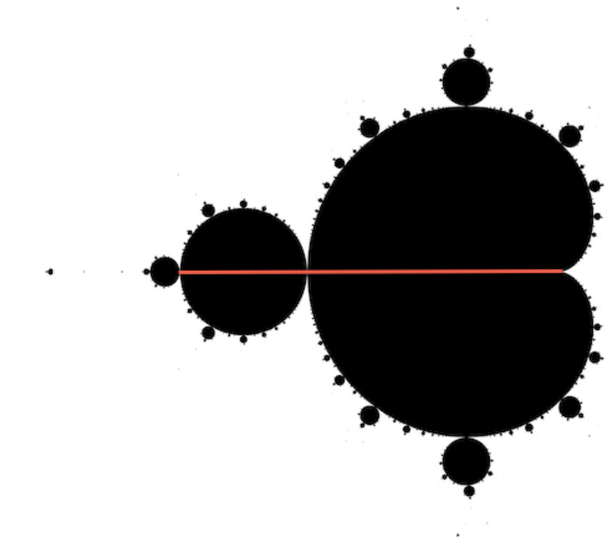


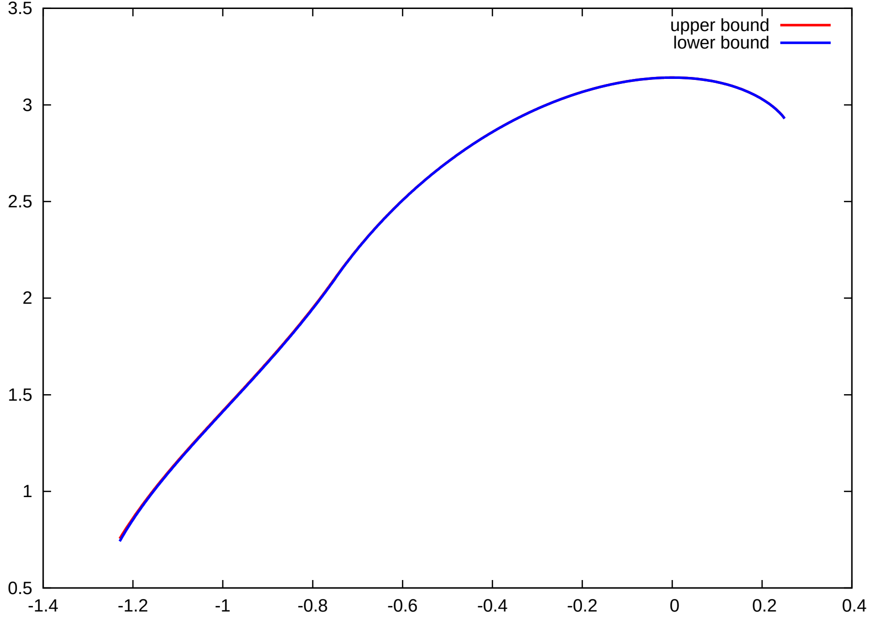
Figure 45. Upper bounds for the area of the filled Julia set for $f_c(z) = z^2 + c$ *in the range* $-2 \leq c \leq .25$.

J. Milnor, *Dynamics in one complex variable*

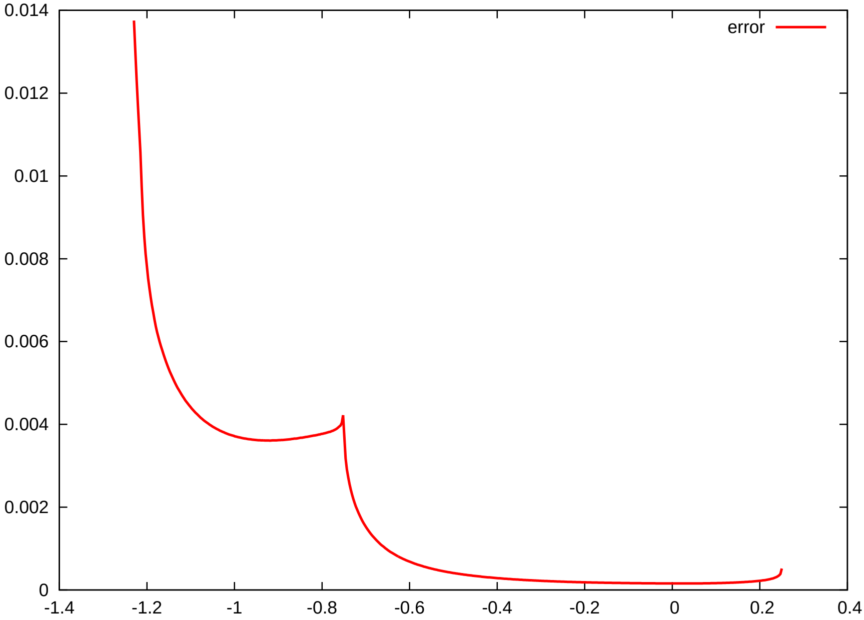
Area of filled Julia set $\quad -1.25 \leq c \leq 0.25$



Area of filled Julia set $-1.25 \le c \le 0.25$ level 19



Area of filled Julia set $-1.25 \leq c \leq 0.25$ level 19



Limitations

- Memory
- Need to explore $\Omega \supseteq [-R, R] \times [-R, R]$
- No proof of convergence

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
- No proof of convergence
  do approximations for J always decrease with the resolution?

Future work

- Julia sets for other polynomials

- Julia sets for Newton's method

Future work

- Julia sets for other polynomials

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

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

- Julia sets for Newton's method

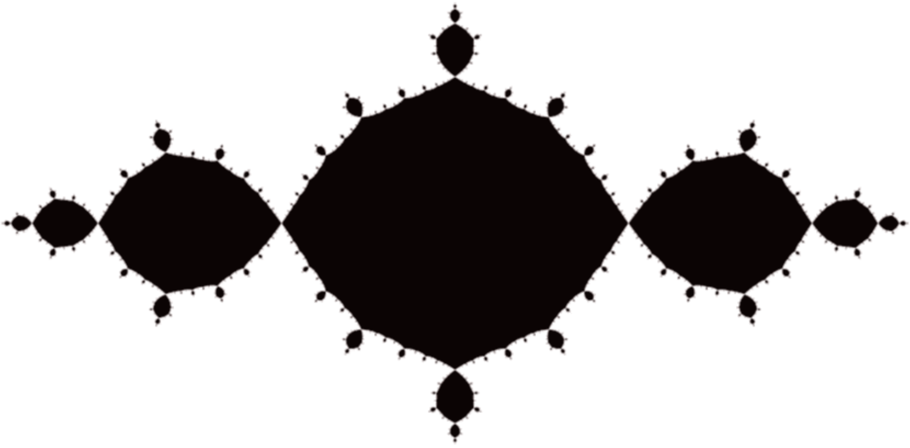
  no escape radius

  need to find explicit attracting regions around zeros

Julia set panorama

http://monge.visgraf.impa.br/panorama/julia-256GP/julia.htm

Images of Julia sets that you can trust



Thanks!

Related work

- M. Braverman and M. Yampolsky. *Computability of Julia sets*, volume 23 of *Algorithms and Computation in Mathematics*. Springer-Verlag, 2009.
- M. Dellnitz and A. Hohmann. A subdivision algorithm for the computation of unstable manifolds and global attractors. *Numerische Mathematik*, 75(3):293–317, 1997.
- C. S. Hsu. *Cell-to-cell mapping: A method of global analysis for nonlinear systems*. Springer-Verlag, 1987.
- J. Milnor. *Dynamics in one complex variable*, volume 160 of *Annals of Mathematics Studies*. Princeton University Press, third edition, 2006.
- R. E. Moore. *Interval Analysis*. Prentice-Hall, 1966.
- R. Rettinger and K. Weihrauch. The computational complexity of some Julia sets. In *Proceedings of the 35th Annual ACM Symposium on Theory of Computing*, pages 177–185. ACM, 2003.
- D. Saupe. Efficient computation of Julia sets and their fractal dimension. *Phys. D*, 28(3):358–370, 1987.

Area of filled Julia set $-1.25 \leq c \leq 0.25$ level 19