

Fractal Zoomer



Version: **1.0.6.7**

Author: **Christos Kalonakis**

Contact: hrkalona@gmail.com

File Menu

- **Starting Position**, restores the default position and size for each fractal.
- **Go To**, lets you, accurately, choose the center and the size.
When Julia option is selected for the first time, Go To lets you set the Julia's seed.
- **Zoom In**, lets you zoom in with a fixed zooming factor.
- **Zoom Out**, lets you zoom out with a fixed zooming factor.

- **Up**, lets you move one screen up.
- **Down**, lets you move one screen down.
- **Left**, lets you move one screen left.
- **Right**, lets you move one screen right.

- **Save Settings As**, saves the center, size, fractal color, palette, out coloring mode, in coloring mode, iterations, bailout, fractal function, plane, rotation, initial value, perturbation, julia settings, domain coloring, and image filters.
- **Load Settings**, loads the center, size, fractal color, palette, out coloring mode, in coloring mode, iterations, bailout, fractal function, plane, rotation, initial value, perturbation, julia settings, domain coloring, and image filters.

- **Save Image As**, lets you save a png image.
- **Save Settings and Image As**, lets you save the current settings and a png image.

- **Edit User Code**, opens the text editor and lets you edit the user code.
- **Compile User Code**, compiles the user code.

- **Exit**, exits the application.

Options Menu

- **Fractal Options Menu**, lets you set the fractal options.
- **Colors Menu**, lets you set the color options.
- **Iterations Menu**, lets you change the maximum iterations.
- **Bailout Conditions Menu**, lets you set the bailout condition.
- **Bailout**, lets you set the value that will be checked through the bailout test algorithm.

A complex number can either be bounded from the bailout value, or not bounded.

Bailout will reset to default every time use choose a new function, julia set or a plane transformation.

- **Rotation Menu**, lets you rotate the image by some degrees.
- **User Point**, lets you select a complex number, that can be used in user formulas and in all the plane transformations that use a transformation center.
- **Image Size**, lets you change the size of the image, both dimensions will be the same.
Increasing the size of the image will improve the image's quality, but its going to be needed more time, for the image, to draw.
3D option will be disabled for image sizes greater than 2000.
- **Stretch Factor**, lets you change the size ratio of the image. This option can be used, along with rotation, to reveal some hidden patterns, when the image is too noisy.
- **Zooming Factor**, lets you change the rate of each zoom in or zoom out.
- **Optimizations Menu**, lets you configure multithreading and enable some algorithms that will speed up the drawing process.
- **Tools Options Menu**, lets you change the options for the tools.

- **Filters Options**, lets you change the options for the image filters.
- **Options Overview**, creates a report that contains all the current active fractal options.
- **Window Menu**, lets you change the window options.

Fractal Options Menu

- **Fractal Functions Menu**, lets you choose the function generating the fractal.
- **Planes Menu**, lets you apply a transformation to the initial complex number, which then gets iterated through the normal process.
- **Initial Value**, lets you set a complex number as an initial value. This option also supports a variable formula to be assigned by the user. While using this option, Root Finding Methods, Sierpinski Gasket, Julia and Julia Map options will be disabled.
- **Perturbation**, lets you add a complex number to the initial value. This option also supports a variable formula to be assigned by the user. While using this option, Root Finding Methods, Sierpinski Gasket, Julia and Julia Map options will be disabled.

Fractal Functions Menu

- **Mandelbrot Type Menu**, contains some of the mandelbrot type variations.
- **Formulas Menu**, contains some of the formulas created by fractal enthusiasts.
- **Lambda**, lets you change the function to $z = cz(1 - z)$.
- **Magnet Type Menu**, contains the magnet type variations.
- **Root Finding Methods Menu**, lets you use some iterative methods to find the roots of a function.
- **Barnsley Type Menu**, contains the barnsley type variations.
- **Szegedi Butterfly Type Menu**, contains the szegedi butterfly type variations.
- **Nova**, lets you set the function to one of the root finding methods,

$$\text{Newton: } z = z - r \frac{p(z)}{p'(z)} + c$$

$$\text{Halley: } z = z - r \frac{2p(z)p'(z)}{2p'(z)^2 - p(z)p''(z)} + c$$

$$\text{Schroder: } z = z - r \frac{p(z)p'(z)}{p'(z)^2 - p(z)p''(z)} + c$$

$$\text{Householder: } z = z - r \frac{p(z)[2p'(z)^2 + p(z)p''(z)]}{2p'(z)^3} + c$$

$$\text{Secant: } z_{n+1} = z_n - r p(z_n) \frac{z_n - z_{n-1}}{p(z_n) - p(z_{n-1})} + c,$$

$$\text{Steffensen: } z = z - r \frac{p(z)^2}{p(z+p(z)) - p(z)} + c,$$

$$\text{Muller: } z_{n+1} = z_n - r(z_n - z_{n-1}) \frac{2C}{B \pm \sqrt{B^2 - 4AC}} + c,$$

$$\text{Parhalley: } z = z - r \frac{2f(z)}{f'(z) \pm \sqrt{f'(z)^2 - 2f(z)f''(z)}} + c,$$

Laguerre:
$$z = z - r \frac{f(z) \deg}{f'(z) \pm \sqrt{(\deg-1)^2 f'(z)^2 - \deg(\deg-1) f''(z) f(z)}} + c,$$

where $p(z) = z^w - 1$. The exponent and the relaxation are given by the user. The famous Nova fractal can be obtained by setting the initial value to $1 + 0i$. While using the Nova function, Bailout Conditions, Bailout, and Periodicity Checking options will be disabled.

- **Spider**, lets you change the function to $z = z^2 + c$, $c = \frac{c}{2} + z$.
- **Manowar**, lets you change the function to $t = z^2 + z_1 + c$, $z_1 = z$, $z = t$.
- **Phoenix**, lets you change the function to $t = z^2 + k$, $s = z$, $z = t$, where $k = (\text{Im}(c)\text{Re}(s) + \text{Re}(c)) + (\text{Im}(c)\text{Im}(s))i$.

- **Sierpinski Gasket**, lets you change the function to
if $\text{Im}(z) > 0.5$ then $z = 2\text{Re}(z) + (2\text{Im}(z) - 1)i$
else(if $\text{Re}(z) > 0.5$ then $z = 2\text{Re}(z) - 1 + 2\text{Im}(z)i$ else $z = 2z$).

While using this function Periodicity Checking, Perturbation, Initial Value, Julia, and Julia Map options will be disabled.

- **Frothy Basin**, lets you change the function to $z = z^2 - (1 + 1.028713768218725i)\bar{z} + c$.
- **Math Library Type Menu**, contains the math library type variations.
- **User Formulas Type Menu**, lets you define your own formulas.

Most of the above functions follow the same pattern, excluding Root Finding Methods and Magnet functions. A pixel will either stay bounded (norm < bailout), or it will escape to infinity (norm >= bailout).

While using any function, except Mandelbrot, Distance Estimator option will be

disabled.

Mandelbrot Type Menu

- **Mandelbrot** $z = z^2 + c$, is the most famous fractal. You can choose some variations of the function, choosing some of the default exponents 3 - 10.
- **Multibrot** $z = z^n + c$, lets you manually choose the exponent of z, to any real number.
- **Multibrot** $z = z^w + c$, lets you manually choose the exponent of z, to any complex number.
- **Multibrot Polynomial**, lets you insert your own polynomial input as p(z) (coefficients in degree descending order).

- **Burning Ship**, lets you change the function to $z = (|\operatorname{Re}(z)| + |\operatorname{Im}(z)|i)^k + c$, where k is any of the default, real or complex exponents. A polynomial input can also be used with burning ship. This function will only work for mandelbrot and multibrot functions.

- **Mandel Grass**, lets you change the function to $z = z^k + c$,
$$z = z + r \frac{z}{\sqrt{\operatorname{Re}(z)^2 + \operatorname{Im}(z)^2}},$$
where k is any of the default, real or complex exponents. A polynomial input can also be used with mandel grass. The factor r is given by the user. This function will only work for mandelbrot and multibrot functions.

- **Mandelbar**, lets you change the function to $z = \bar{z}^2 + c$, where \bar{z} is the conjugate of the complex number.

Formulas Menu

- **M-Like Type Menu**, contains some of the m-like type variations.
- **Kaliset Type Menu**, contains some of the kaliset type variations.
- **General Type Menu**, contains some general type variations.
- **Coupled Type Menu**, contains some coupled type variations.

M-Like Type Menu

- $z = 0.25z^2 + c + (z^2 + c)^{-2}$, is a m-like type variation.
- $z = cz^2 + (cz^2)^{-1}$, is a m-like type variation.
- $z = cz^2 + 1 + (cz^2 + 1)^{-1}$, is a m-like type variation.
- $z = z^2 + z^2c^{-1}$, is a m-like type variation.
- $z = (z^2) * e^{\frac{1}{z}} + c$, is a m-like type variation.
- $z = (-0.4 + 0.2i)cz^2 - \frac{0.275i}{c}z + c$, is a m-like type variation.
- $z = c(az^b + dz^e)$ **Menu**, contains some m-like type variations.
- $z = (c(az^b + dz^e) + f)^g$ **Menu**, contains some m-like type variations.
- $z = \frac{(z^a+b)}{(z^d+e)} + f + g$ **Menu**, contains some m-like type variations.

$z = c(az^b + dz^e)$ Menu

- $z = c(z + z^{-1})$
- $z = c(z^3 + z^{-3})$
- $z = c(5z^{-1} + z^5)$
- $z = c(z^5 - 5z)$
- $z = c(2z^2 - z^4)$
- $z = c(2z^{-2} - z^{-4})$
- $z = c(5z^{-1} - z^{-5})$
- $z = c(-3z^{-3} + z^{-9})$
- $z = c(z^{2.5} + z^{-2.5})$
- $z = c(z^{2i} + z^{-2})$
- $z = c(-3z^{-2} + 2z^{-3})$
- $z = c((z + 2)^6 + (z + 2)^{-6})$

$z = (c(az^b + dz^e) + f)^g$ Menu

- $z = (c(2z - z^2) + 1)^2$
- $z = (c(2z - z^2) + 1)^3$
- $z = (c(z - z^2) + 1)^2$
- $z = (c(z - z^2) + 1)^3$
- $z = (c(z - z^2) + 1)^4$
- $z = (c(z^3 - z^4) + 1)^5$

$$z = \frac{(z^a+b)}{(z^d+e)} + f + g \text{ Menu}$$

- $z = \frac{(z^{10}+c)}{(z^8+c)} + c - 2$
- $z = \frac{(z^{16}-10)}{(z^{14}-c)} + c - 6$

Kalaset Type Menu

- $z = |z^{-1}| + c$
- $z = |z^2| + c$
- $z = \frac{|z|}{|c|} + c$
- $z = \left| \frac{z}{c} \right| + c$
- $z = \left| \frac{z}{0.5+0.5i} \right| + c$
- $z = \frac{|z|}{c} + c$
- $z = |z|^{-3} + c$
- $z = |z^{-3}| + c$

All of the above formulas are using the absolute value function.

General Type Menu

- **Math Library Type Menu**, contains some math type variations.
- $z = z - \frac{z^n + c}{nz^{n-2}}$ **Menu**, contains some newton type variations.

Math Library Type Menu

- $z = \sin(z)c$
- $z = \cos(z)c$
- $z = z\log(z + 1) + c$
- $z = z\sin(z) + c$
- $z = z\sinh(z) + c$
- $z = 2 - 2\cos(z) + c$
- $z = 2\cosh(z) - 2 + c$
- $z = z\sin z(z) - c^2$

$z = z - \frac{z^n+c}{nz^{n-2}}$ Menu

- $z = z - \frac{z^3+c}{3z}$
- $z = z - \frac{z^4+c}{4z^2}$
- $z = z - \frac{z^5+c}{5z^3}$
- $z = z - \frac{z^8+c}{8z^6}$

Coupled Type Menu

- **Coupled Mandelbrot**, creates a coupled version of the mandelbrot set.
- **Coupled Mandelbrot-Burning Ship**, creates a coupled version of the mandelbrot set with the burning ship set.

Magnet Type Menu

- **Magnet 1**, lets you change the function to $z = \left(\frac{z^2+c-1}{2z+c-2}\right)^2$.
- **Magnet 2**, lets you change the function to $z = \left(\frac{z^3+3(c-1)z+(c-1)(c-2)}{3z^2+3(c-2)z+(c-1)(c-2)+1}\right)^2$.

Magnet functions follow 3 patterns. A pixel will either stay bounded (norm < bailout),

or it will escape to infinity (norm >= bailout), or it will converge to a single point (1 + 0i).

Newton Method Menu

- **Newton Method Menu**, lets you use Newton's iterative method for root finding.
- **Halley Method Menu**, lets you use Halley's iterative method for root finding.
- **Scroder Method Menu**, lets you use Schroder's iterative method for root finding.
- **Householder Method Menu**, lets you use Householders's iterative method for root finding.
- **Secant Method Menu**, lets you use Secant's iterative method for root finding.
- **Steffensen Method Menu**, lets you use Steffensen's iterative method for root finding.
- **Muller Method Menu**, lets you use Muller's iterative method for root finding.
- **Parhalley Method Menu**, lets you use Parabolic Halley's iterative method for root finding.
- **Laguerre Method Menu**, lets you use Laguerre's iterative method for root finding.

Root Finding Methods follow 2 patterns. A pixel will either converge to one of the roots of the function, or it wont converge at all.

Newton Method Menu

- **Newton's iterative method:** $z_{n+1} = z_n - \frac{f(z_n)}{f'(z_n)}$
- **Newton 3**, lets you change the function to $p(z) = z^3 - 1$.
- **Newton 4**, lets you change the function to $p(z) = z^4 - 1$.
- **Newton Generalized 3**, lets you change the function to $p(z) = z^3 - 2z + 2$.
- **Newton Generalized 8**, lets you change the function to $p(z) = z^8 + 15z^4 - 16$.
- **Newton Sin**, lets you change the function to $f(z) = \sin(z)$.
- **Newton Cos**, lets you change the function to $f(z) = \cos(z)$.
- **Newton Polynomial**, lets you insert your own polynomial input as $p(z)$ (coefficients in degree descending order).
- **Newton Formula**, lets you set a custom function.

While using the Newton functions, Bailout Conditions, Bailout, Periodicity Checking, most of the out coloring algorithms, Perturbation, Initial Value, Julia, and Julia Map options will be disabled.

Halley Method Menu

- **Halley's iterative method:**
$$z_{n+1} = z_n - \frac{2f(z_n)f'(z_n)}{2f'(z_n)^2 - f(z_n)f''(z_n)}$$
- **Halley 3**, lets you change the function to $p(z) = z^3 - 1$.
- **Halley 4**, lets you change the function to $p(z) = z^4 - 1$.
- **Halley Generalized 3**, lets you change the function to $p(z) = z^3 - 2z + 2$.
- **Halley Generalized 8**, lets you change the function to $p(z) = z^8 + 15z^4 - 16$.
- **Halley Sin**, lets you change the function to $f(z) = \sin(z)$.
- **Halley Cos**, lets you change the function to $f(z) = \cos(z)$.
- **Halley Polynomial**, lets you insert your own polynomial input as $p(z)$ (coefficients in degree descending order).
- **Halley Formula**, lets you set a custom function.

While using the Halley functions, Bailout Conditions, Bailout, Periodicity Checking,

most of the out coloring algorithms, Perturbation, Initial Value, Julia, and Julia Map options will be disabled.

Schroder Method Menu

- **Schroder's iterative method:** $z_{n+1} = z_n - \frac{f(z_n)f'(z_n)}{f'(z_n)^2 - f(z_n)f''(z_n)}$
- **Schroder 3**, lets you change the function to $p(z) = z^3 - 1$.
- **Schroder 4**, lets you change the function to $p(z) = z^4 - 1$.
- **Schroder Generalized 3**, lets you change the function to $p(z) = z^3 - 2z + 2$.
- **Schroder Generalized 8**, lets you change the function to $p(z) = z^8 + 15z^4 - 16$.
- **Schroder Sin**, lets you change the function to $f(z) = \sin(z)$.
- **Schroder Cos**, lets you change the function to $f(z) = \cos(z)$.
- **Schroder Polynomial**, lets you insert your own polynomial input as $p(z)$ (coefficients in degree descending order).
- **Schroder Formula**, lets you set a custom function.

While using the Schroder functions, Bailout Conditions, Bailout, Periodicity Checking, most of the out coloring algorithms, Perturbation, Initial Value, Julia, and Julia Map options will be disabled.

Householder Method Menu

- **Householder's iterative method:**
$$z_{n+1} = z_n - \frac{f(z_n)[2f'(z_n)^2 + f(z_n)f''(z_n)]}{2f'(z_n)^3}$$
- **Householder 3**, lets you change the function to $p(z) = z^3 - 1$.
- **Householder 4**, lets you change the function to $p(z) = z^4 - 1$.
- **Householder Generalized 3**, lets you change the function to $p(z) = z^3 - 2z + 2$.
- **Householder Generalized 8**, lets you change the function to $p(z) = z^8 + 15z^4 - 16$.
- **Householder Sin**, lets you change the function to $f(z) = \sin(z)$.
- **Householder Cos**, lets you change the function to $f(z) = \cos(z)$.
- **Householder Polynomial**, lets you insert your own polynomial input as $p(z)$ (coefficients in degree descending order).
- **Householder Formula**, lets you set a custom function.

While using the Householder functions, Bailout Conditions, Bailout, Periodicity Checking, most of the out coloring algorithms, Perturbation, Initial Value, Julia, and Julia Map options will be disabled.

Secant Method Menu

- **Secant's iterative method:** $z_{n+1} = z_n - f(z_n) \frac{z_n - z_{n-1}}{f(z_n) - f(z_{n-1})}$
- **Secant 3**, lets you change the function to $p(z) = z^3 - 1$.
- **Secant 4**, lets you change the function to $p(z) = z^4 - 1$.
- **Secant Generalized 3**, lets you change the function to $p(z) = z^3 - 2z + 2$.
- **Secant Generalized 8**, lets you change the function to $p(z) = z^8 + 15z^4 - 16$.
- **Secant Cos**, lets you change the function to $f(z) = \cos(z)$.
- **Secant Polynomial**, lets you insert your own polynomial input as $p(z)$ (coefficients in degree descending order).
- **Secant Formula**, lets you set a custom function.

While using the Secant functions, Bailout Conditions, Bailout, Periodicity Checking, most of the out coloring algorithms, Perturbation, Initial Value, Julia, and Julia Map options will be disabled.

Steffensen Method Menu

- **Steffensen's iterative method:** $z_{n+1} = z_n - \frac{f(z_n)^2}{f(z_n + f(z_n)) - f(z_n)}$
- **Steffensen 3**, lets you change the function to $p(z) = z^3 - 1$.
- **Steffensen 4**, lets you change the function to $p(z) = z^4 - 1$.
- **Steffensen Generalized 3**, lets you change the function to $p(z) = z^3 - 2z + 2$.
- **Steffensen Polynomial**, lets you insert your own polynomial input as $p(z)$ (coefficients in degree descending order).
- **Steffensen Formula**, lets you set a custom function.

While using the Steffensen functions, Bailout Conditions, Bailout, Periodicity Checking, most of the out coloring algorithms, Perturbation, Initial Value, Julia, and Julia Map options will be disabled.

Muller Method Menu

- **Muller's iterative method:**

$$q = \frac{z_n - z_{n-1}}{z_{n-1} - z_{n-2}}$$

$$A = qf(z_n) - q(q + 1)f(z_{n-1}) + q^2f(z_{n-2})$$

$$B = (1 + 2q)f(z_n) - (1 + q)^2f(z_{n-1}) + q^2f(z_{n-2})$$

$$C = (1 + q)f(z_n)$$

$$z_{n+1} = z_n - (z_n - z_{n-1}) \frac{2C}{B \pm \sqrt{B^2 - 4AC}}$$

- **Muller 3**, lets you change the function to $p(z) = z^3 - 1$.
- **Muller 4**, lets you change the function to $p(z) = z^4 - 1$.
- **Muller Generalized 3**, lets you change the function to $p(z) = z^3 - 2z + 2$.
- **Muller Generalized 8**, lets you change the function to $p(z) = z^8 + 15z^4 - 16$.
- **Muller Sin**, lets you change the function to $f(z) = \sin(z)$.
- **Muller Cos**, lets you change the function to $f(z) = \cos(z)$.
- **Muller Polynomial**, lets you insert your own polynomial input as $p(z)$ (coefficients in degree descending order).
- **Muller Formula**, lets you set a custom function.

While using the Muller functions, Bailout Conditions, Bailout, Periodicity Checking, most of the out coloring algorithms, Perturbation, Initial Value, Julia, and Julia Map options will be disabled.

Parhalley Method Menu

- **Parabolic Halley's iterative method:**

$$z_{n+1} = z_n - \frac{2f(z_n)}{f'(z_n) \pm \sqrt{f'(z_n)^2 - 2f(z_n)f''(z_n)}}$$

- **Parhalley 3**, lets you change the function to $p(z) = z^3 - 1$.
- **Parhalley 4**, lets you change the function to $p(z) = z^4 - 1$.
- **Parhalley Generalized 3**, lets you change the function to $p(z) = z^3 - 2z + 2$.
- **Parhalley Generalized 8**, lets you change the function to $p(z) = z^8 + 15z^4 - 16$.
- **Parhalley Sin**, lets you change the function to $f(z) = \sin(z)$.
- **Parhalley Cos**, lets you change the function to $f(z) = \cos(z)$.
- **Parhalley Polynomial**, lets you insert your own polynomial input as $p(z)$ (coefficients in degree descending order).
- **Parhalley Formula**, lets you set a custom function.

While using the Parabolic Halley functions, Bailout Conditions, Bailout, Periodicity Checking, most of the out coloring algorithms, Perturbation, Initial Value, Julia, and Julia Map options will be disabled.

Laguerre Method Menu

- **Laguerre's iterative method:**

$$z_{n+1} = z_n - \frac{f(z_n) \deg}{f'(z_n) \pm \sqrt{(\deg-1)^2 f'(z_n)^2 - \deg(\deg-1) f''(z_n) f(z_n)}}$$

- **Laguerre 3**, lets you change the function to $p(z) = z^3 - 1$.
- **Laguerre 4**, lets you change the function to $p(z) = z^4 - 1$.
- **Laguerre Generalized 3**, lets you change the function to $p(z) = z^3 - 2z + 2$.
- **Laguerre Generalized 8**, lets you change the function to $p(z) = z^8 + 15z^4 - 16$.
- **Laguerre Sin**, lets you change the function to $f(z) = \sin(z)$.
- **Laguerre Cos**, lets you change the function to $f(z) = \cos(z)$.
- **Laguerre Polynomial**, lets you insert your own polynomial input as $p(z)$ (coefficients in degree descending order).
- **Laguerre Formula**, lets you set a custom function.

While using the Laguerre functions, Bailout Conditions, Bailout, Periodicity Checking,

most of the out coloring algorithms, Perturbation, Initial Value, Julia, and Julia Map

options will be disabled.

Barnsley Type Menu

- **Barnsley 1**, lets you change the function to
if $\operatorname{Re}(z) \geq 0$ then $z = (z - 1)c$ else $z = (z + 1)c$.
- **Barnsley 2**, lets you change the function to
if $\operatorname{Re}(z)\operatorname{Im}(c) + \operatorname{Re}(c)\operatorname{Im}(z) \geq 0$ then $z = (z + 1)c$ else $z = (z - 1)c$.
- **Barnsley 3**, lets you change the function to
if $\operatorname{Re}(z) > 0$ then $z = z^2 - 1$ else $z = z^2 + \operatorname{Re}(c)\operatorname{Re}(z) + \operatorname{Im}(c)\operatorname{Re}(z) - 1$.

Szegedi Butterfly Type Menu

- **Szegedi Butterfly 1**, lets you change the function to

$$z = (\operatorname{Im}(z)^2 - \sqrt{|\operatorname{Re}(z)|}) + (\operatorname{Re}(z)^2 - \sqrt{|\operatorname{Im}(z)|})i + c.$$

- **Szegedi Butterfly 2**, lets you change the function to

$$z = (\operatorname{Re}(z)^2 - \sqrt{|\operatorname{Im}(z)|}) + (\operatorname{Im}(z)^2 - \sqrt{|\operatorname{Re}(z)|})i + c.$$

Math Library Type Menu

- $z = e^z + c$, lets you use the classic math, exp function.
- $z = \log(z) + c$, lets you use the classic math, log function.
- $z = \sin(z) + c$, lets you use the classic math, sin function.
- $z = \cos(z) + c$, lets you use the classic math, cos function.
- $z = \tan(z) + c$, lets you use the classic math, tan function.
- $z = \cot(z) + c$, lets you use the classic math, cot function.
- $z = \sinh(z) + c$, lets you use the classic math, sinh function.
- $z = \cosh(z) + c$, lets you use the classic math, cosh function.
- $z = \tanh(z) + c$, lets you use the classic math, tanh function.
- $z = \coth(z) + c$, lets you use the classic math, coth function.

User Formulas Type Menu

- **User Formula**, lets you define your own formulas.
- **User Formula Iteration Based**, lets you define your own formulas. Different formula will be evaluated in every iteration.
- **User Formula Conditional**, lets you define your own formulas. Different formula will be evaluated based on the given condition.
- **User Formula Coupled**, lets you define your own formulas. The formulas will be coupled using linear interpolation.

Planes Menu

- **General Planes Menu**, contains all the general transformations.
- **Fold Planes Menu**, contains all the fold transformations.
- **Newton Planes Menu**, contains all the newton transformations.
- **Math Planes Menu**, contains all the math library transformations.
- **User Planes**, lets you define your own transformation.
- **Apply Planes on Julia Seed**, lets you enable the application of the plane transformation to the julia seed.
- **Apply Planes on Julia Plane**, lets you enable the application of the plane transformation to the julia plane.

After the chosen transformation has taken place, the transformed complex number is iterated through the normal process.

General Planes Menu

- μ , is the default plane (no transformation is applied).
- μ^2 , transforms the initial complex number, by using the transformation z^2 .
- μ^{2i} , transforms the initial complex number, by using the transformation z^{2i} .
- $\frac{1}{\mu}$, transforms the initial complex number, by using the transformation $\frac{1}{z}$.
- $\frac{1}{\mu+0.25}$, transforms the initial complex number, by using the transformation $\frac{1}{z} + 0.25$.
- $\frac{1}{\mu-1.40115}$, transforms the initial complex number, by using the transformation $\frac{1}{z} - 1.40115$.
- $\frac{1}{\mu-2}$, transforms the initial complex number, by using the transformation $\frac{1}{z} - 2$.
- $\frac{\mu^2}{\mu^4-0.25}$, transforms the initial complex number, by using the transformation $\frac{z^2}{z^4-0.25}$.
- λ , transforms the initial complex number, by using the transformation $z(1-z)$.
- $\frac{1}{\lambda}$, transforms the initial complex number, by using the transformation $\frac{1}{z}(1-\frac{1}{z})$.
- $\frac{1}{\lambda-1}$, transforms the initial complex number, by using the transformation $0.25 - \frac{1}{z^2}$.
- **Bipolar**, transforms the initial complex number, by using the bipolar coordinates conversion.
- **Inverse Bipolar**, transforms the initial complex number, by using the inverse bipolar coordinates conversion.
- **Circle Inversion**, transforms the initial complex number, by using the circle inversion.
- **Inflection**, transforms the initial complex number, by using the inflection transformation.

Fold Planes Menu

- **Fold up**, transforms the initial complex number, by using the fold up transformation.
- **Fold down**, transforms the initial complex number, by using the fold down transformation.
- **Fold right**, transforms the initial complex number, by using the fold right transformation.
- **Fold left**, transforms the initial complex number, by using the fold left transformation.
- **Fold in**, transforms the initial complex number, by using the fold in transformation.
- **Fold out**, transforms the initial complex number, by using the fold out transformation.

Distort Planes Menu

- **Twirl**, transforms the initial complex number, by using the twirl transformation.
- **Shear**, transforms the initial complex number, by using the shear transformation.
- **Kaleidoscope**, transforms the initial complex number, by using the kaleidoscope transformation.
- **Pinch**, transforms the initial complex number, by using the pinch transformation.
- **Ripples**, transforms the initial complex number, by using the ripples transformation.

Newton Planes Menu

- **Newton's iterative method:** $z_{n+1} = z_n - \frac{f(z_n)}{f'(z_n)}$
- **Newton 3**, transforms the initial complex number, by using the transformation $p(z) = z^3 - 1$.
- **Newton 4**, transforms the initial complex number, by using the transformation $p(z) = z^4 - 1$.
- **Newton Generalized 3**, transforms the initial complex number, by using the transformation $p(z) = z^3 - 2z + 2$.
- **Newton Generalized 8**, transforms the initial complex number, by using the transformation $p(z) = z^8 + 15z^4 - 16$.

Only the first 5 iterations of Newton's method are computed.

The transformed complex number equals to the complex number created by the 5th iteration.

Math Planes Menu

- e^z , transforms the initial complex number, by using the transformation e^z .
- $\log(z)$, transforms the initial complex number, by using the transformation $\log(z)$.
- \sqrt{z} , transforms the initial complex number, by using the transformation \sqrt{z} .
- $|z|$, transforms the initial complex number, by using the transformation $|z|$ (absolute value).
- $\Gamma(z)$, transforms the initial complex number, by using the gamma function transformation.
- $z!$, transforms the initial complex number, by using the factorial transformation.
- $\operatorname{erf}(z)$, transforms the initial complex number, by using the error function transformation.
- $\zeta(z)$, transforms the initial complex number, by using the riemann zeta function transformation.

- **Trigonometric Planes Menu**, contains all the trigonometric transformations.

- **Inverse Trigonometric Planes Menu**, contains all the inverse trigonometric transformations.

Trigonometric Planes Menu

- **$\sin(z)$** , transforms the initial complex number, by using the transformation $\sin(z)$.
- **$\cos(z)$** , transforms the initial complex number, by using the transformation $\cos(z)$.
- **$\tan(z)$** , transforms the initial complex number, by using the transformation $\tan(z)$.
- **$\cot(z)$** , transforms the initial complex number, by using the transformation $\cot(z)$.
- **$\sinh(z)$** , transforms the initial complex number, by using the transformation $\sinh(z)$.
- **$\cosh(z)$** , transforms the initial complex number, by using the transformation $\cosh(z)$.
- **$\tanh(z)$** , transforms the initial complex number, by using the transformation $\tanh(z)$.
- **$\coth(z)$** , transforms the initial complex number, by using the transformation $\coth(z)$.
- **$\sec(z)$** , transforms the initial complex number, by using the transformation $\sec(z)$.
- **$\csc(z)$** , transforms the initial complex number, by using the transformation $\csc(z)$.
- **$\operatorname{sech}(z)$** , transforms the initial complex number, by using the transformation $\operatorname{sech}(z)$.
- **$\operatorname{csch}(z)$** , transforms the initial complex number, by using the transformation $\operatorname{csch}(z)$.

[Inverse Trigonometric Planes Menu](#)

- **asin(z)**, transforms the initial complex number, by using the transformation $\text{asin}(z)$.
- **acos(z)**, transforms the initial complex number, by using the transformation $\text{acos}(z)$.
- **atan(z)**, transforms the initial complex number, by using the transformation $\text{atan}(z)$.
- **acot(z)**, transforms the initial complex number, by using the transformation $\text{acot}(z)$.
- **asinh(z)**, transforms the initial complex number, by using the transformation $\text{asinh}(z)$.
- **acosh(z)**, transforms the initial complex number, by using the transformation $\text{acosh}(z)$.
- **atanh(z)**, transforms the initial complex number, by using the transformation $\text{atanh}(z)$.
- **acoth(z)**, transforms the initial complex number, by using the transformation $\text{acoth}(z)$.
- **asec(z)**, transforms the initial complex number, by using the transformation $\text{asec}(z)$.
- **acsc(z)**, transforms the initial complex number, by using the transformation $\text{acsc}(z)$.
- **asech(z)**, transforms the initial complex number, by using the transformation $\text{asech}(z)$.
- **acsch(z)**, transforms the initial complex number, by using the transformation $\text{acsch}(z)$.

Colors Menu

- **Out Coloring Mode Menu**, lets you set how to display the areas outside the set.
- **In Coloring Mode Menu**, lets you set how to display the areas inside the set.
- **Processing Menu**, lets you set some post-processing algorithms.
- **Blending Menu**, lets you set some image blending methods.
- **Fractal Color**, lets you set the colors corresponding to the maximum iterations, distance estimation and some color algorithms.
- **Palette Menu**, lets you set a palette corresponding to how fast the norm of the complex number is getting greater than the bailout value, or how fast it converges to some value.
- **Random Palette**, randomizes the palette.
- **Palette Shifting Menu**, lets you apply an offset to the selected palette.
- **Transfer Functions Menu**, lets you change the function of color mapping, that is applied to the palette.
- **Color Intensity**, sets the intensity of the current palette.

Out Coloring Mode Menu

- **Escape Time**, is the default way of coloring areas. This means that the number of iterations needed, for a complex number to be greater than the bailout
or for a complex number to converge in a root (Root Finding Methods), are used.
- **Binary Decomposition**, uses the number of iterations if $\text{Im}(z) \geq 0$ else iterations + 50.
- **Binary Decomposition 2**, uses the number of iterations if $\text{Re}(z) \geq 0$ else iterations + 50.
- **Escape Time + Re**, uses the number of iterations + $\text{Re}(z)$.
While using this option all the Root Finding Methods will be disabled.
- **Escape Time + Im**, uses the number of iterations + $\text{Im}(z)$.
While using this option all the Root Finding Methods will be disabled.
- **Escape Time + Re/Im**, uses the number of iterations + $\text{Re}(z)/\text{Im}(z)$.
While using this option all the Root Finding Methods will be disabled.
- **Escape Time + Re + Im + Re/Im**, uses the number of iterations + $\text{Re}(z) + \text{Im}(z) + \text{Re}(z)/\text{Im}(z)$.
While using this option all the Root Finding Methods will be disabled.
- **Biomorph**, uses the number of iterations if $\text{Re}(z)$ or $\text{Im}(z)$ is inside the rectangular boundaries
of bailout else iterations + 50.
While using this option all the Root Finding Methods will be disabled.
- **Color Decomposition**, uses $(\text{atan2}(\text{Im}(z), \text{Re}(z)) / 2\pi + 0.75) * 59\pi$.
Its great for the Root Finding Methods, since every root gets a different color.
- **Escape Time + Color Decomposition**, uses the number of iterations + $(\text{atan2}(\text{Im}(z), \text{Re}(z)) / 2\pi + 0.75) * 59\pi$.
Its great for the Root Finding Methods, since every root gets a different color, plus the iterations for the convergence are taken into account.
- **Escape Time + Gaussian Integer**, uses the number of iterations plus the distance to the closest gaussian integer, iterations + $\text{norm}(z - \text{GI}(z)) * 90$, where $\text{GI}(z) = \text{round}(\text{Re}(z), \text{Im}(z))$.
While using this option all the Root Finding Methods will be disabled.
- **Escape Time + Gaussian Integer 2**, uses iterations + $\text{atan}(\text{Im}(w), \text{Re}(w)) * 5$, where $w = z - \text{GI}(z)$, $\text{GI}(z) = \text{round}(\text{Re}(z), \text{Im}(z))$.
While using this option all the Root Finding Methods will be disabled.
- **Escape Time + Gaussian Integer 3**, uses iterations + $\text{Re}(w)$, where $w = z - \text{GI}(z)$, $\text{GI}(z) = \text{round}(\text{Re}(z), \text{Im}(z))$.

While using this option all the Root Finding Methods will be disabled.

- **Escape Time + Gaussian Integer 4**, uses iterations + $\text{Re}(w) + \text{Im}(w)$, where $w = z - \text{GI}(z)$, $\text{GI}(z) = \text{round}(\text{Re}(z), \text{Im}(z))$.

While using this option all the Root Finding Methods will be disabled.

- **Escape Time + Gaussian Integer 5**, uses iterations + $\text{Re}(w) + \text{Im}(w) + \text{Re}(w)/\text{Im}(w)$,

where $w = z - \text{GI}(z)$, $\text{GI}(z) = \text{round}(\text{Re}(z), \text{Im}(z))$.

While using this option all the Root Finding Methods will be disabled.

- **Escape Time + Algorithm**, uses iterations + $\text{atan}(\text{Im}(w), \text{Re}(w)) * 4$, where $w = z_{(n)} - z_{(n-1)}$.

- **Escape Time + Algorithm 2**, uses iterations + $\text{atan}(\text{Im}(w), \text{Re}(w)) * 8$, where $w = z - \sin(z)$.

While using this option all the Root Finding Methods will be disabled.

- **Distance Estimator**, uses the distance to the nearest point on the boundary.

While using this option all the Functions, except Mandelbrot, will be disabled.

While using this option, Initial value, and Perturbation option will be disabled.

- **Escape Time + Escape Radius**, uses iterations, the norm, and the argument.

While using this option all the Root Finding Methods will be disabled.

- **Escape Time + Grid**, uses iterations, the norm, and the argument.

While using this option all the Root Finding Methods will be disabled.

- **Banded**, uses an iteration based color algorithm.

While using this option all the Root Finding Methods will be disabled.

- **Escape Time + Field Lines**, uses iterations, the norm, and the argument.

While using this option all the Root Finding Methods will be disabled.

- **Escape Time + Field Lines 2**, uses iterations, the norm, and the argument.

While using this option all the Root Finding Methods will be disabled.

- **User Out Coloring Method**, lets you define your own coloring algorithm.

While using this option, every time you select a new fractal function, the formula will be reseted.

In Coloring Mode Menu

- **Maximum Iterations**, is the default way of coloring areas. This means that the color assigned to the maximum number of iterations (usually black) will be used.
- **norm(z)**, takes into account the norm of z in the last iteration to colorize the area.
- **Decomposition Like**, uses $(\text{atan2}(\text{Im}(z), \text{Re}(z)) / 2\pi + 0.75) * 59\pi$.
- **Re / Im**, uses $\text{Re}(z) / \text{Im}(z)$ in the last iteration to colorize the area.
- **cos(norm(z))**, uses $\cos(\text{Re}(z) * \text{Im}(z) * |\text{Re}(z)| * |\text{Im}(z)|)$ if $(\text{norm}(z) * 10) \% 2$ is 1, else $\sin(\text{Re}(z) * \text{Im}(z) * |\text{Re}(z)| * |\text{Im}(z)|)$.
- **norm(z) * cos(Re^2)**, uses $\text{norm}(z) * \cos(\text{Re}(z)^2)$ in the last iteration to colorize the area.
- **sin(Re^2 - Im^2)**, uses $\sin(\text{Re}(z)^2 - \text{Im}(z)^2)$ in the last iteration to colorize the area.
- **atan(Re * Im * |Re| * |Im|)**, uses $\text{atan}(\text{Re}(z) * \text{Im}(z) * |\text{Re}(z)| * |\text{Im}(z)|)$ in the last iteration to colorize the area.
- **Squares**, uses $(\text{atan2}(\text{Im}(z), \text{Re}(z)) / 2\pi + 0.75) * 59\pi$ if $(|\text{Re}(z) * 40| \% 2) \text{ XOR } (|\text{Im}(z) * 40| \% 2)$ is 1, else $(\text{atan2}(\text{Re}(z), \text{Im}(z)) / 2\pi + 0.75) * 59\pi$.
- **Squares 2**, is variation of the Squares algorithm, using different coloring method.
- **User In Coloring Method**, lets you define your own coloring algorithm.

While using any but the default in-coloring algorithm, Periodicity Checking option will be disabled.

Processing Menu

- **Smoothing**, smooths the image, so the color transitions are not easily visible. You should use smoothing for Bump Mapping, Fake Distance Estimation, Entropy Coloring and Rainbow Palette.
- **Distance Estimation**, sets some points near the boundary of the set to the maximum iterations value. While using this option all the Functions, except Mandelbrot, will be disabled. While using this option, Initial value, Perturbation, Burning Ship, and Mandelgrass option will be disabled.
- **Fake Distance Estimation**, emulates the effect of distance estimation without using derivatives.
- **Entropy Coloring**, calculates the entropy of nearby points to create a coloring effect.
- **Offset Coloring**, applies a palette offset to areas with similar iteration gradients.
- **Rainbow Palette**, uses the gradients of the fractal to create a rainbow effect.
- **Greyscale Coloring**, transforms the image to greyscale, in areas with similar iteration gradients.
- **Bump Mapping**, emulates a light source to create a pseudo 3d image, including shadows.

Blending Menu

- **Normal**, blends the image using linear interpolation.
- **Multiply**, blends the image using multiply blending.
- **Divide**, blends the image using divide blending.
- **Addition**, blends the image using addition blending.
- **Subtraction**, blends the image using subtraction blending.
- **Difference**, blends the image using difference blending.
- **Value**, blends the image using value blending.
- **Saturation**, blends the image using saturation blending.
- **Color**, blends the image using color blending.
- **Overlay**, blends the image using overlay blending.
- **Screen**, blends the image using screen blending.
- **Dodge**, blends the image using dodge blending.
- **Burn**, blends the image using burn blending.
- **Darken Only**, blends the image using darken only blending.
- **Lighten Only**, blends the image using lighten only blending.
- **Hard Light**, blends the image using hard light blending.
- **Grain Extract**, blends the image using grain extract blending.
- **Grain Merge**, blends the image using grain merge blending.

The blending methods affect Bump Mapping, Rainbow Palette, Entropy Coloring, Offset Coloring, Contours in Domain Coloring, and 3D mode.

Palette Menu

- **Default**, is the Default coloring palette.
 - **Spectrum**, is a coloring palette, based on spectrum.
 - **Alternative**, is palette based on software, Fractal Extreme.
 - **Alternative 2**, is an alternative coloring palette.
 - **Alternative 3**, is an alternative coloring palette.
 - **Alternative 4**, is an alternative coloring palette.
 - **Alternative 5**, is an alternative coloring palette.
 - **Alternative 6**, is an alternative coloring palette.
 - **Alternative 7**, is an alternative coloring palette.
 - **Alternative 8**, is an alternative coloring palette.
 - **Alternative 9**, is an alternative coloring palette.
 - **Dusk**, is a palette based on colors of dusk.
 - **Gray Scale**, is a palette based on gray scale.
 - **Earth Sky**, is a palette based on the colors of earth and sky.
 - **Hot Cold**, is a palette based on the colors of hot and cold.
 - **Hot Cold 2**, is a palette based on color temperature.
 - **Fire**, is a palette based on the colors of fire.
 - **Jet**, is a palette based on the colors of hot and cold.
-
- **Custom Palette**, is a palette, custom made by the user.
Through the custom palette editor, you can save, load, select the interpolation used, select the color space, or randomize your palettes.

Palette Shifting Menu

- **Shift Palette**, lets you shift the selected palette by a specific offset.
- **Shift Palette Forward**, lets you shift the selected palette forward by one.
- **Shift Palette Backward**, lets you shift the selected palette backward by one.

Transfer Functions Menu

- **Linear**, applies a linear mapping, n .
- **Square Root**, applies a square root mapping, $\sqrt{n + 1}$.
- **Cube Root**, applies a cube root mapping, $\sqrt[3]{n + 1}$.
- **Fourth Root**, applies a fourth root mapping, $\sqrt[4]{n + 1}$.
- **Logarithm**, applies a logarithmic mapping, $\log(n + 1)$.
- **Log Log**, applies a logarithm of the logarithm mapping, $\log(\log(n + 1) + 1)$.
- **Atan**, applies an arc tangent mapping, $\text{atan}(n)$.

Iterations Menu

- **Set Iterations**, lets you set the maximum iterations.
- **Increase Iterations**, lets you increase the maximum iterations by one.
- **Decrease Iterations**, lets you decrease the maximum iterations by one.

Increasing the iterations will slow down the drawing process, but in some cases will increase the quality of the produced image.

Bailout Conditions Menu

- **Circle (Euclidean norm)**, uses $\sqrt{\operatorname{Re}(z)^2 + \operatorname{Im}(z)^2} \geq \text{bailout}$ for the bailout condition.
- **Square (Infinity norm)**, uses $\max(|\operatorname{Re}(z)|, |\operatorname{Im}(z)|) \geq \text{bailout}$ for the bailout condition.
- **Rhombus (One norm)**, uses $|\operatorname{Re}(z)| + |\operatorname{Im}(z)| \geq \text{bailout}$ for the bailout condition.
- **N-Norm**, uses $\sqrt[n]{|\operatorname{Re}(z)|^n + |\operatorname{Im}(z)|^n} \geq \text{bailout}$ for the bailout condition.
- **Strip**, uses $|\operatorname{Re}(z)| \geq \text{bailout}$ for the bailout condition.
- **Halfplane**, uses $\operatorname{Re}(z) \geq \text{bailout}$ for the bailout condition.
- **Field Lines**, uses $\frac{\operatorname{Re}(z_n)}{\operatorname{Re}(z_{n-1})} \geq \text{bailout}$ and $\frac{\operatorname{Im}(z_n)}{\operatorname{Im}(z_{n-1})} \geq \text{bailout}$ for the bailout condition.
- **User Bailout Condition**, lets the user set the bailout condition.

Rotation Menu

- **Set Rotation**, lets you set the rotation number in degrees, and the rotation center.
- **Increase Rotation**, lets you increase the rotation by one degree.
- **Decrease Rotation**, lets you decrease the rotation by one degree.

While the rotation value is not -360 or 0 or 360, the Show Grid option will be disabled.

Optimizations Menu

- **Threads**, lets you set the number of threads, $n \times n$, in a 2-dimensional grid. The drawing process will split in parallel drawing pieces. You need to use the same or more threads than your computer's processors.
- **Greedy Drawing Algorithms**, lets you enable some algorithms that calculate only some parts of the image, either by tracing the boundaries or using divide and conquer. These algorithms can significantly speedup the drawing process, but they can introduce errors. Some customization options are also provided to visualize the effect of the algorithms.
- **Periodicity Checking**, detects periodical orbits, so the iterating process can escape from them in fewer iterations, speeding up the drawing. It is recommended to use this option when the image contains a lot of bounded areas and you are using a high number of iterations. In the opposite case the drawing will be slowed down. While using this option, In Coloring Mode options will be disabled.
- **Quick Draw Tiles**, lets you set the size of the tile. This size affects the zooming with mouse Scroll, Moving with Ctrl + Left Mouse Click, and Rotating with Shift + Left Mouse Click. The higher the tile size, the quicker the image renders, as less calculation are being performed at the expense of lesser image quality.

Tools Options Menu

- **Orbit Menu**, lets you change the options about orbit.
- **Julia Preview Image Filters**, will enable the image filters that you use on the normal image, so Julia's preview will also be filtered by them. Using the filters will make the preview slower.
- **Julia Map Options**, lets you change the julia map options.
- **3D Options**, lets you change the 3D options.
- **Polar Projection Options**, lets you change the polar projection options.
- **Domain Coloring Options**, lets you change the domain coloring options.
- **Color Cycling Options**, lets you change the color cycling options.
- **Grid Menu**, lets you change the options of the grid.
- **Zoom Window**, lets you change the zoom window options.
- **Boundaries Menu**, lets you change the boundaries options.

Tools Menu

- **Orbit**, lets you draw the orbit of a complex number.

If the orbit option is selected, clicking the left mouse click or dragging on the image will generate the orbit of the complex number corresponding to the mouse's location on the image. You can also set the orbit from Go To in the File Menu.

While using this option, 3D and Julia Map option will be disabled.

The Julia option will be disabled while the orbit option is selected.

You need to have a drawn Julia set to generate its orbit.

- **Julia**, lets you draw an image based on a complex number (seed).

When Julia option is selected, you can preview the Julia set corresponding to the seed generated from your mouse's location, by moving the mouse. If you left click to

that location or set the seed from Go To in the File Menu, the Julia set will be drawn, more detailed.

The previews are generated, using only 250 iterations.

All the basic options will be disabled while waiting for you to pick a seed.

Deselecting the option will enable them.

While using this option, all the Root Finding Methods, Julia Map, Perturbation and Initial Value options will be disabled.

- **Julia Map**, lets you split your image into a 2 dimensional grid, and then display the corresponding julia set on every grid cell. Each julia set is generated when taking

the middle point of its grid cell as a seed value. When you increase the julia slices,

you will notice that most of the fractal functions will have a direct connection to their julia sets.

While using this option, all the Root Finding Methods, Julia, 3D, Domain Coloring,

and Orbit option will be disabled.

- **3D**, lets you create a 3 dimensional version of the image, using height data (heightmap) in order to achieve that transformation.

While using this option, Julia, Julia Map, Color Cycling, Grid, Orbit option and Boundary Tracing option will be disabled.

You need to have a drawn Julia set to generate the 3D version of the image.

- **Polar Projection**, projects the complex plane into polar coordinates, where the x-axis represents the magnitude, normalized with the exp function, and the y-axis represents the angle. While using this option, Grid, and Zoom Window option will be disabled.

- **Domain Coloring**, generates an image of the complex plane, based on the angle and the norm of a complex number. While using this option, Color Cycling, Julia, Julia Map, Bailout Conditions, Bailout, Periodicity Checking and Boundary Tracing option will be disabled. If the current palette is used then Out Coloring Mode, In Coloring Mode Distance Estimation, Fake Distance Estimation, Bump Mapping, Rainbow Palette, Entropy Coloring, Offset Coloring, Greyscale Coloring, and Fractal Color options will be disabled. In the opposite case all color options will be disabled. You need to have a drawn Julia set to generate the domain coloring version of the image.

- **Color Cycling**, lets you animate your image, by continuously cycling your active palette. While using this option, all the basic options will be disabled.

- **Plane Visualization**, lets you preview the current active plane transformation.

- **Show Grid**, will draw a coordinated grid. While using this option, 3D, and Polar Projection option will be disabled. While using this option, it is recommended to use image size greater than 600 or the grid numbers will not be easily distinguishable. The grid option, will only be enabled for the default rotations (0, 360, -360).

- **Show Zoom Window**, will draw a window indicating the contents of the location when zoomed into. While using this option, Julia, Julia Map, 3D, Color Cycling,

Grid, Orbit, and Polar Projection option will be disabled.

- **Show Boundaries**, will draw the boundaries of the plane for different magnitudes, and it will also indicate the bailout boundary with dashed line.

While using this option, 3D option will be disabled.

The boundaries option, will only be enabled for the default rotations (0, 360, -360).

The boundaries option, will be disabled after a certain zoom depth.

Details Menu

- **Anti-Aliasing**, is a filter that smooths the image, using supersampling.
- **Edge Detection**, is a filter that detects edges of the image.
- **Sharpness**, is a filter that makes the edges of the image more sharp.
- **Blurring**, is a filter that smooths the image by applying a blurring filter.
- **Emboss**, is a filter that edits the image, so bright areas are raised and dark ones are carved,
This filter gives some 3D properties to the image.
- **Glow**, is a filter that smooths the image by applying a glow with a blurring effect.
- **Noise**, is a filter that adds random noise to the image.

The options of the filters can be changed via the filters options.

Colors Menu

- **Histogram Equalization**, is a filter that changes the contrast of the image based on its histogram.
- **Posterization**, is a filter that reduces the color tones of the image based on a level threshold.
- **Contrast/Brightness**, is a filter that changes the contrast/brightness of the image.
- **Gain/Bias**, is a filter that changes the gain/bias of the image.
- **Gamma**, is a filter that changes the gamma of the image.
- **Exposure**, is a filter that changes the exposure of the image.
- **Color Temperature**, is a filter that changes the temperature of the colors in the image.
- **Inverted Colors**, is a filter that inverts the colors of the image.
- **Solarization**, is a filter that emulates the effect if the sun on a photographic print that is wholly or partially reversed in tone.
- **Mask Color Channel**, is a filter that removes a specific color channel of the image.
- **Color Channel Swapping**, is a filter that swaps the color channels of the image.
- **Color Channel Swizzling**, is a filter that swizzles the color channels of the image.
- **Color Channel Mixing**, is a filter that mixes the color channels of the image.
- **Color Channel Adjusting**, is a filter that adjusts the color channels of the

image.

- **Color Channel HSB Adjusting**, is a filter that adjusts the HSB color channels of the image.
- **Color Channel Scaling**, is a filter that scales the color channels of the image.
- **Grayscale**, is a filter that converts the image to grayscale.
- **Fade Out**, is a filter that applies a fade out effect to the image.

The options of the filters can be changed via the filters options.

Textures Menu

- **Dither**, is a filter that creates a dithering effect by reducing the color tones of the image.
- **Crystallize**, is a filter that creates a crystallization effect based on the image.
- **Pointillize**, is a filter that creates a pointillization effect based on the image.
- **Oil**, is a filter that creates an oil paint effect based on the image.
- **Marble**, is a filter that creates a marble effect on the image.
- **Weave**, is a filter that creates a weave effect based on the image.
- **Sparkle**, is a filter that creates a glowing/shining on the image.
- **Mirror**, is a filter that creates a mirror effect with transparency.

The options of the filters can be changed via the filters options.

Lighting Menu

- **Light Effects**, is a filter that add light effects to the image.

The options of the filters can be changed via the filters options.

Mouse Functionality

- **Left Mouse Click**, lets you zoom in with a fixed zooming factor.
If Julia option is selected for the first time, left mouse click lets you choose the seed.
If Orbit option is selected, left mouse click or drag lets you draw the selected complex number orbit.
If 3D option is selected, dragging the image will rotate the 3D model.
It is recommended to use less 3D details to achieve a smoother movement.
- **Right Mouse Click**, lets you zoom out with a fixed zooming factor.
- **Scroll Wheel**, lets you zoom in/out with fixed zooming factor.
The image will be generated in a lower resolution to speed up the process.
- **Ctrl + Left Mouse Click**, lets you move the image.
The image will be generated in a lower resolution to speed up the process.
- **Shift + Left Mouse Click**, lets you rotate the image.
- **Alt + Left Mouse Click**, lets you select the User Point.

The image will be generated in a lower resolution to speed up the process.

While zooming in or out, the image will recenter to the chosen position.

User Formulas

Variables

- **z**: current sequence point.
- **c**: plane point.
- **s**: starting sequence point.
- **p**: previous sequence point.
- **pp**: second to previous sequence point.
- **n**: current iteration.
- **maxn**: current maximum iterations.
- **bail**: current bailout.
- **cbail**: convergent bailout.
- **center**: current center.
- **size**: current size.
- **sizei**: current image size (size / sizei will yield the pixel size).
- **v1**: placeholder variable 1.
- **v2**: placeholder variable 2.
- **v3**: placeholder variable 3.
- **v4**: placeholder variable 4.
- **v5**: placeholder variable 5.
- **v6**: placeholder variable 6.
- **v7**: placeholder variable 7.
- **v8**: placeholder variable 8.
- **v9**: placeholder variable 9.
- **v10**: placeholder variable 10.
- **v11**: placeholder variable 11.
- **v12**: placeholder variable 12.
- **v13**: placeholder variable 13.
- **v14**: placeholder variable 14.
- **v15**: placeholder variable 15.
- **v16**: placeholder variable 16.
- **v17**: placeholder variable 17.
- **v18**: placeholder variable 18.
- **v19**: placeholder variable 19.
- **v20**: placeholder variable 20.
- **point**: a value that can be picked by the user.

All the placeholder variables are meant to be used inside the UserDefinedFunctions.java file, in order to store some temporary values that can be used later on, at a user defined out-coloring or in-coloring method. These variables are focused on experienced users that might have some computer science background.

You must use the mutable versions of the complex functions or use the assign function on these variables, or else the value that you want to save will be lost. For instance when you want to set the value of a variable do not use **v = new Complex(1, 2)** but you should use **v.assign(new Complex(1, 2))** Initially these variables are set to 0.

Consider the following scenario, you call **m1(z, c, v1)** and inside the body of m1, defined in UserDefinedFunctions.java you set a value to the 3rd argument of the m1 function based on some condition, defined by you. When the iteration terminates the out-coloring or in-coloring method is called, for example **cos(3*v1)**, and upon this point the value of **v1** that was set during the iteration step will be used. You can have a similar result if you use it inside a user bailout formula that calls a function defined in UserDefinedFunctions.java

Operations/Symbols

- **+**: addition.
- **-**: subtraction.
- *****: multiplication.
- **/**: division.
- **%**: remainder.
- **^**: exponentiation.
- **(**: left parenthesis.
- **)**: right parenthesis.
- **,**: comma.

Constants

- **pi**: 3.141592653589793.
- **e**: 2.718281828459045.

- **phi**: 1.618033988749895.
- **c10 (Champernowne's)**: 0.1234567891011121314.
- **alpha (Feigenbaum's)**: 2.5029078750958928.
- **delta (Feigenbaum's)**: 4.669201609102990.

Complex Numbers

- **a + bi**: a = Real, b = Imaginary.

Trigonometric Functions: f(z)

- **sin**: sine function.
- **cos**: cosine function.
- **tan**: tangent function.
- **cot**: cotangent function.
- **sec**: secant function.
- **csc**: cosecant function.
- **sinh**: hyperbolic sine function.
- **cosh**: hyperbolic cosine function.
- **tanh**: hyperbolic tangent function.
- **coth**: hyperbolic cotangent function.
- **sech**: hyperbolic secant function.
- **csch**: hyperbolic cosecant function.
- **asin**: arc sine function.
- **acos**: arc cosine function.
- **atan**: arc tangent function.
- **acot**: arc cotangent function.
- **asec**: arc secant function.
- **acsc**: arc cosecant function.
- **asinh**: hyperbolic arc sine function.
- **acosh**: hyperbolic arc cosine function.
- **atanh**: hyperbolic arc tangent function.
- **acoth**: hyperbolic arc cotangent function.
- **asech**: hyperbolic arc secant function.
- **acsch**: hyperbolic arc cosecant function.
- **vsin**: versine function.
- **vcos**: vercosine function.

- **cvsin**: coversine function.
- **cvcos**: covercosine function.
- **hvsin**: haversine function.
- **hvcos**: havercosine function.
- **hcvsin**: hacoversine function.
- **hcvcos**: hacovercosine function.
- **exsec**: exsecant function.
- **excsc**: excosecant function.
- **avsin**: arc versine function.
- **avcos**: arc vercosine function.
- **acvsin**: arc coversine function.
- **acvcos**: arc covercosine function.
- **ahvsin**: arc haversine function.
- **ahvcos**: arc havercosine function.
- **ahcvsin**: arc hacoversine function.
- **ahcvcos**: arc hacovercosine function.
- **aexsec**: arc exsecant function.
- **aexcsc**: arc excosecant function.

Other Functions: f(z)

- **exp**: exp function.
- **log**: natural logarithm function.
- **log10**: base 10 logarithm function.
- **log2**: base 2 logarithm function.
- **sqrt**: square root function.
- **abs**: absolute value function (applied both on real and imaginary parts).
- **absre**: absolute value function (applied only on real part).
- **absim**: absolute value function (applied only on imaginary part).
- **conj**: conjugate number function.
- **re**: real part function.
- **im**: imaginary part function.
- **norm**: norm function.
- **arg**: argument function.
- **gamma**: gamma function.
- **fact**: factorial function.
- **erf**: error function.
- **rzeta**: riemann zeta function.

- **gi**: gaussian integer function.
- **rec**: reciprocal function.
- **flip**: flip real to imaginary function.
- **round**: math round function.
- **ceil**: math ceil function.
- **floor**: math floor function.
- **trunc**: math truncate function.
- **deta**: dirichlet eta function.
- **f1**: user function f1 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **f2**: user function f2 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **f3**: user function f3 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **f4**: user function f4 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **f5**: user function f5 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **f6**: user function f6 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **f7**: user function f7 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **f8**: user function f8 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **f9**: user function f9 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **f10**: user function f10 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **f11**: user function f11 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **f12**: user function f12 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **f13**: user function f13 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **f14**: user function f14 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **f15**: user function f15 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **f16**: user function f16 which can be changed in UserDefinedFunctions.java

(Compilation is required).

- **f17**: user function f17 which can be changed in UserDefinedFunctions.java

(Compilation is required).

- **f18**: user function f18 which can be changed in UserDefinedFunctions.java

(Compilation is required).

- **f19**: user function f19 which can be changed in UserDefinedFunctions.java

(Compilation is required).

- **f20**: user function f20 which can be changed in UserDefinedFunctions.java

(Compilation is required).

- **f21**: user function f21 which can be changed in UserDefinedFunctions.java

(Compilation is required).

- **f22**: user function f22 which can be changed in UserDefinedFunctions.java

(Compilation is required).

- **f23**: user function f23 which can be changed in UserDefinedFunctions.java

(Compilation is required).

- **f24**: user function f24 which can be changed in UserDefinedFunctions.java

(Compilation is required).

- **f25**: user function f25 which can be changed in UserDefinedFunctions.java

(Compilation is required).

- **f26**: user function f26 which can be changed in UserDefinedFunctions.java

(Compilation is required).

- **f27**: user function f27 which can be changed in UserDefinedFunctions.java

(Compilation is required).

- **f28**: user function f28 which can be changed in UserDefinedFunctions.java

(Compilation is required).

- **f29**: user function f29 which can be changed in UserDefinedFunctions.java

(Compilation is required).

- **f30**: user function f30 which can be changed in UserDefinedFunctions.java

(Compilation is required).

- **f31**: user function f31 which can be changed in UserDefinedFunctions.java

(Compilation is required).

- **f32**: user function f32 which can be changed in UserDefinedFunctions.java

(Compilation is required).

- **f33**: user function f33 which can be changed in UserDefinedFunctions.java

(Compilation is required).

- **f34**: user function f34 which can be changed in UserDefinedFunctions.java

(Compilation is required).

- **f35**: user function f35 which can be changed in UserDefinedFunctions.java

(Compilation is required).

- **f36**: user function f36 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **f37**: user function f37 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **f38**: user function f38 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **f39**: user function f39 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **f40**: user function f40 which can be changed in UserDefinedFunctions.java (Compilation is required).

The f1 - f40 functions are focused on experienced users that might have some computer science background.

Two Argument Functions: f(z, w)

- **logn**: log base n (argument 1: z, argument 2: base).
- **bipol**: bipolar function.
- **ibipol**: inversed bipolar function.
- **inflect**: inflection function.
- **foldu**: fold up function.
- **foldd**: fold down function.
- **foldl**: fold left function.
- **foldr**: fold right function.
- **foldi**: fold in function.
- **foldo**: fold out function.
- **shear**: shear function.
- **cmp**: compare function (returns: -1 when $z > w$, 1 when $z < w$, 0 when equal).
- **fuzz**: adds a random pertubation, based on the w argument used.
- **normn**: n-norm function.
- **g1**: user function g1 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **g2**: user function g2 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **g3**: user function g3 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **g4**: user function g4 which can be changed in UserDefinedFunctions.java (Compilation is required).

- **g5**: user function g5 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **g6**: user function g6 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **g7**: user function g7 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **g8**: user function g8 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **g9**: user function g9 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **g10**: user function g10 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **g11**: user function g11 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **g12**: user function g12 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **g13**: user function g13 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **g14**: user function g14 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **g15**: user function g15 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **g16**: user function g16 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **g17**: user function g17 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **g18**: user function g18 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **g19**: user function g19 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **g20**: user function g20 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **g21**: user function g21 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **g22**: user function g22 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **g23**: user function g23 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **g24**: user function g24 which can be changed in UserDefinedFunctions.java

(Compilation is required).

- **g25**: user function g25 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **g26**: user function g26 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **g27**: user function g27 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **g28**: user function g28 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **g29**: user function g29 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **g30**: user function g30 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **g31**: user function g31 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **g32**: user function g32 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **g33**: user function g33 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **g34**: user function g34 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **g35**: user function g35 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **g36**: user function g36 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **g37**: user function g37 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **g38**: user function g38 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **g39**: user function g39 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **g40**: user function g40 which can be changed in UserDefinedFunctions.java (Compilation is required).

The g1 - g40 functions are focused on experienced users that might have some computer science background.

[Multiple Argument User Functions: f\(z1, ... z10\)](#)

All the following functions can have up to 10 arguments, for example you can use them like this: $m_1(z, 3+2i)$ or $m_2(1+2i, z, c, 3)$.

All the missing arguments will be defaulted to 0.

- **m1**: user function m1 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **m2**: user function m2 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **m3**: user function m3 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **m4**: user function m4 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **m5**: user function m5 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **m6**: user function m6 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **m7**: user function m7 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **m8**: user function m8 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **m9**: user function m9 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **m10**: user function m10 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **m11**: user function m11 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **m12**: user function m12 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **m13**: user function m13 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **m14**: user function m14 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **m15**: user function m15 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **m16**: user function m16 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **m17**: user function m17 which can be changed in UserDefinedFunctions.java (Compilation is required).

- **m18**: user function m18 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **m19**: user function m19 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **m20**: user function m20 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **m21**: user function m21 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **m22**: user function m22 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **m23**: user function m23 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **m24**: user function m24 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **m25**: user function m25 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **m26**: user function m26 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **m27**: user function m27 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **m28**: user function m28 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **m29**: user function m29 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **m30**: user function m30 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **m31**: user function m31 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **m32**: user function m32 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **m33**: user function m33 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **m34**: user function m34 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **m35**: user function m35 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **m36**: user function m36 which can be changed in UserDefinedFunctions.java (Compilation is required).
- **m37**: user function m37 which can be changed in UserDefinedFunctions.java

(Compilation is required).

- **m38**: user function m38 which can be changed in UserDefinedFunctions.java

(Compilation is required).

- **m39**: user function m39 which can be changed in UserDefinedFunctions.java

(Compilation is required).

- **m40**: user function m40 which can be changed in UserDefinedFunctions.java

(Compilation is required).

The m1 - m40 functions are focused on experienced users that might have some computer science background.

Additional Two Argument Functions : $f(z, w)$

- **add**: addition.
- **sub**: subtraction.
- **mult**: multiplication.
- **div**: division.
- **rem**: remainder.
- **pow**: exponentiation.

General Guidelines:

If you set the User Out Coloring or User In Coloring algorithm to the value of -maxn, then the corresponding color for Distance Estimation will be used.

If you set the User Out Coloring or User In Coloring algorithm to a negative value, then the corresponding color for Special Color will be used.