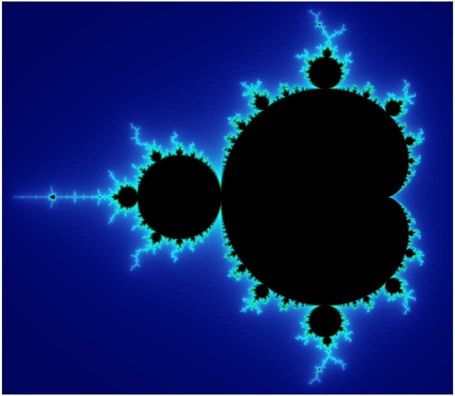
The Mandelbrot set

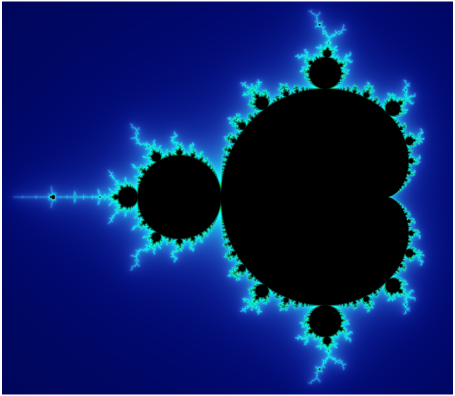
- The Mandelbrot set is a famous example of a fractal subset of $\mathbb{C}$ with a very intricate structure. It is the black part of the image below.



- We will explore how to make pictures of this set using Python.

The Mandelbrot set

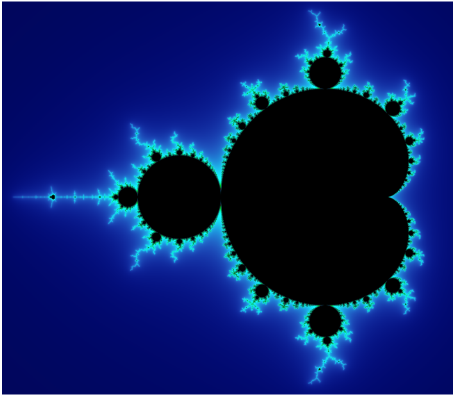
- The Mandelbrot set is a famous example of a fractal subset of $\mathbb{C}$ with a very intricate structure. It is the black part of the image below.



- We will explore how to make pictures of this set using Python.

The Mandelbrot set

- The Mandelbrot set is a famous example of a fractal subset of $\mathbb{C}$ with a very intricate structure. It is the black part of the image below.



- We will explore how to make pictures of this set using Python.

The Mandelbrot set

- For a number $c \in \mathbb{C}$, put $q_c(z) = z^2 + c$.
- Then
$$q_c(0) = c$$
$$q_c^2(0) = q_c(c) = c^2 + c$$
$$q_c^3(c) = q_c(c^2 + c) = c^4 + 2c^3 + c^2 + c$$
  and so on.
- Given a value c, we can look at the sequence of values $q_c^n(0)$ as $n \to \infty$.
- One can show that there are only two possibilities:
  (a) $|q_c^n(0)| \leq 2$ for all n; or
  (b) $|q_c^n(0)| \to \infty$ as $n \to \infty$.
- The Mandelbrot set M is the set of values c for which case (a) occurs, i.e.
$$M = \{c \in \mathbb{C} \mid |q_c^n(0)| \leq 2 \text{ for all } n \geq 0\}.$$
- To calculate the fine structure of M in Python, we need a very large number of calculations. To make this fast and efficient, we need to use vectorized operations with numpy arrays.
- If $c \notin M$ then the numbers $q_c^n(0)$ become very big causing overflow errors in Python. We need trickery to fix this without breaking vectorization.

The Mandelbrot set

- For a number $c \in \mathbb{C}$, put $q_c(z) = z^2 + c$.
- Then
$$\begin{aligned} q_c(0) &= c \\ q_c^2(0) &= q_c(c) = c^2 + c \\ q_c^3(c) &= q_c(c^2 + c) = c^4 + 2c^3 + c^2 + c \end{aligned}$$
  and so on.
- Given a value c, we can look at the sequence of values $q_c^n(0)$ as $n \to \infty$.
- One can show that there are only two possibilities:
  (a) $|q_c^n(0)| \leq 2$ for all n; or
  (b) $|q_c^n(0)| \to \infty$ as $n \to \infty$.
- The Mandelbrot set M is the set of values c for which case (a) occurs, i.e.
$$M = \{c \in \mathbb{C} \mid |q_c^n(0)| \leq 2 \text{ for all } n \geq 0\}.$$
- To calculate the fine structure of M in Python, we need a very large number of calculations. To make this fast and efficient, we need to use vectorized operations with numpy arrays.
- If $c \notin M$ then the numbers $q_c^n(0)$ become very big causing overflow errors in Python. We need trickery to fix this without breaking vectorization.

The Mandelbrot set

- For a number $c \in \mathbb{C}$, put $q_c(z) = z^2 + c$.
- Then
$$\begin{aligned} q_c(0) &= c \\ q_c^2(0) &= q_c(c) = c^2 + c \\ q_c^3(c) &= q_c(c^2 + c) = c^4 + 2c^3 + c^2 + c \end{aligned}$$
and so on.
- Given a value c, we can look at the sequence of values $q_c^n(0)$ as $n \to \infty$.
- One can show that there are only two possibilities:
  (a) $|q_c^n(0)| \leq 2$ for all n; or
  (b) $|q_c^n(0)| \to \infty$ as $n \to \infty$.
- The Mandelbrot set M is the set of values c for which case (a) occurs, i.e.
$$M = \{c \in \mathbb{C} \mid |q_c^n(0)| \leq 2 \text{ for all } n \geq 0\}.$$
- To calculate the fine structure of M in Python, we need a very large number of calculations. To make this fast and efficient, we need to use vectorized operations with numpy arrays.
- If $c \notin M$ then the numbers $q_c^n(0)$ become very big causing overflow errors in Python. We need trickery to fix this without breaking vectorization.

The Mandelbrot set

- For a number $c \in \mathbb{C}$, put $q_c(z) = z^2 + c$.
- Then
$$\begin{aligned} q_c(0) &= c \\ q_c^2(0) &= q_c(c) = c^2 + c \\ q_c^3(c) &= q_c(c^2 + c) = c^4 + 2c^3 + c^2 + c \end{aligned}$$
  and so on.
- Given a value c, we can look at the sequence of values $q_c^n(0)$ as $n \to \infty$.
- One can show that there are only two possibilities:
  (a) $|q_c^n(0)| \leq 2$ for all n; or
  (b) $|q_c^n(0)| \to \infty$ as $n \to \infty$.
- The Mandelbrot set M is the set of values c for which case (a) occurs, i.e.
$$M = \{c \in \mathbb{C} \mid |q_c^n(0)| \leq 2 \text{ for all } n \geq 0\}.$$
- To calculate the fine structure of M in Python, we need a very large number of calculations. To make this fast and efficient, we need to use vectorized operations with numpy arrays.
- If $c \notin M$ then the numbers $q_c^n(0)$ become very big causing overflow errors in Python. We need trickery to fix this without breaking vectorization.

The Mandelbrot set

- For a number $c \in \mathbb{C}$, put $q_c(z) = z^2 + c$.
- Then
$$\begin{aligned} q_c(0) &= c \\ q_c^2(0) &= q_c(c) = c^2 + c \\ q_c^3(c) &= q_c(c^2 + c) = c^4 + 2c^3 + c^2 + c \end{aligned}$$
  and so on.
- Given a value c, we can look at the sequence of values $q_c^n(0)$ as $n \to \infty$.
- One can show that there are only two possibilities:
  (a) $|q_c^n(0)| \leq 2$ for all n; or
  (b) $|q_c^n(0)| \to \infty$ as $n \to \infty$.
- The Mandelbrot set M is the set of values c for which case (a) occurs, i.e.
$$M = \{c \in \mathbb{C} \mid |q_c^n(0)| \leq 2 \text{ for all } n \geq 0\}.$$
- To calculate the fine structure of M in Python, we need a very large number of calculations. To make this fast and efficient, we need to use vectorized operations with numpy arrays.
- If $c \notin M$ then the numbers $q_c^n(0)$ become very big causing overflow errors in Python. We need trickery to fix this without breaking vectorization.

The Mandelbrot set

- For a number $c \in \mathbb{C}$, put $q_c(z) = z^2 + c$.
- Then
$$\begin{aligned} q_c(0) &= c \\ q_c^2(0) &= q_c(c) = c^2 + c \\ q_c^3(c) &= q_c(c^2 + c) = c^4 + 2c^3 + c^2 + c \end{aligned}$$
  and so on.
- Given a value c, we can look at the sequence of values $q_c^n(0)$ as $n \to \infty$.
- One can show that there are only two possibilities:
  (a) $|q_c^n(0)| \leq 2$ for all n; or
  (b) $|q_c^n(0)| \to \infty$ as $n \to \infty$.
- The Mandelbrot set M is the set of values c for which case (a) occurs, i.e.
$$M = \{c \in \mathbb{C} \mid |q_c^n(0)| \leq 2 \text{ for all } n \geq 0\}.$$
- To calculate the fine structure of M in Python, we need a very large number of calculations. To make this fast and efficient, we need to use vectorized operations with numpy arrays.
- If $c \notin M$ then the numbers $q_c^n(0)$ become very big causing overflow errors in Python. We need trickery to fix this without breaking vectorization.

The Mandelbrot set

- For a number $c \in \mathbb{C}$, put $q_c(z) = z^2 + c$.
- Then
$$
\begin{aligned}
q_c(0) &= c \\
q_c^2(0) &= q_c(c) = c^2 + c \\
q_c^3(c) &= q_c(c^2 + c) = c^4 + 2c^3 + c^2 + c
\end{aligned}
$$
  and so on.
- Given a value c, we can look at the sequence of values $q_c^n(0)$ as $n \to \infty$.
- One can show that there are only two possibilities:
  (a) $|q_c^n(0)| \leq 2$ for all n; or
  (b) $|q_c^n(0)| \to \infty$ as $n \to \infty$.
- The Mandelbrot set M is the set of values c for which case (a) occurs, i.e.
$$M = \{c \in \mathbb{C} \mid |q_c^n(0)| \leq 2 \text{ for all } n \geq 0\}.$$
- To calculate the fine structure of M in Python, we need a very large number of calculations. To make this fast and efficient, we need to use vectorized operations with numpy arrays.
- If $c \notin M$ then the numbers $q_c^n(0)$ become very big causing overflow errors in Python. We need trickery to fix this without breaking vectorization.

The Mandelbrot set

- For a number $c \in \mathbb{C}$, put $q_c(z) = z^2 + c$.
- Then
$$\begin{aligned} q_c(0) &= c \\ q_c^2(0) &= q_c(c) = c^2 + c \\ q_c^3(c) &= q_c(c^2 + c) = c^4 + 2c^3 + c^2 + c \end{aligned}$$
  and so on.
- Given a value c, we can look at the sequence of values $q_c^n(0)$ as $n \to \infty$.
- One can show that there are only two possibilities:
  (a) $|q_c^n(0)| \leq 2$ for all n; or
  (b) $|q_c^n(0)| \to \infty$ as $n \to \infty$.
- The Mandelbrot set M is the set of values c for which case (a) occurs, i.e.
$$M = \{c \in \mathbb{C} \mid |q_c^n(0)| \leq 2 \text{ for all } n \geq 0\}.$$
- To calculate the fine structure of M in Python, we need a very large number of calculations. To make this fast and efficient, we need to use vectorized operations with numpy arrays.
- If $c \notin M$ then the numbers $q_c^n(0)$ become very big causing overflow errors in Python. We need trickery to fix this without breaking vectorization.