

# Pythagoras Theorem on Complex Plane. Introduction of Complex Pythagoras Numbers and Complex Pythagoras Planes. Mandelbrot Set Cousins.

Wojciech Christopher Peter Fiałkiewicz

## Abstract

This Preprint shows how extending Pythagoras Theorem to lines with negative length can describe a version of Pythagoras' Theorem that exists on the Complex Plane. From that version of Pythagoras Theorem are being described numbers that have been called Complex Pythagoras Numbers and for each of such a number there exists a Complex Pythagoras Plane, with corresponding Mandelbrot Set Cousin present on that Plane.

## Introduction

Pythagoras' Theorem is in the form:  $a^2 + b^2 = c^2$ , where  $a$ ,  $b$ , and  $c$  are Real Numbers. From that equation, it can be obtained that:

$$c = (a^2 + b^2)^{0.5}$$

$$a = (c^2 - b^2)^{0.5}$$

$$b = (c^2 - a^2)^{0.5}$$

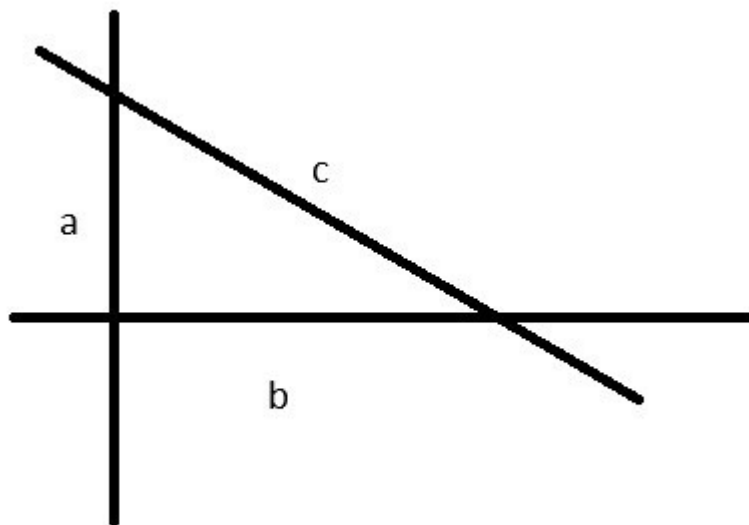


Illustration 1.0

In Illustration 1.0 can be seen that  $c$  is longer than  $b$  because  $a$  has a positive value.

$$b + \delta_a = c$$

$$\delta_a = c - b$$

$$\delta_a = (a^2 + b^2)^{0.5} - (c^2 - a^2)^{0.5}$$

similarly

$$a + \Delta b = c$$

$$\Delta b = c - a$$

$$\Delta b = (a^2 + b^2)^{0.5} - (c^2 - b^2)^{0.5}$$

so

$$c = c$$

$$a + (a^2 + b^2)^{0.5} - (c^2 - b^2)^{0.5} = b + (a^2 + b^2)^{0.5} - (c^2 - a^2)^{0.5}$$

$$a - (c^2 - b^2)^{0.5} = b - (c^2 - a^2)^{0.5}$$

$$a - b = (c^2 - b^2)^{0.5} - (c^2 - a^2)^{0.5}$$

so

$$a = b + (c^2 - b^2)^{0.5} - (c^2 - a^2)^{0.5}$$

$$b = a + (c^2 - a^2)^{0.5} - (c^2 - b^2)^{0.5}$$

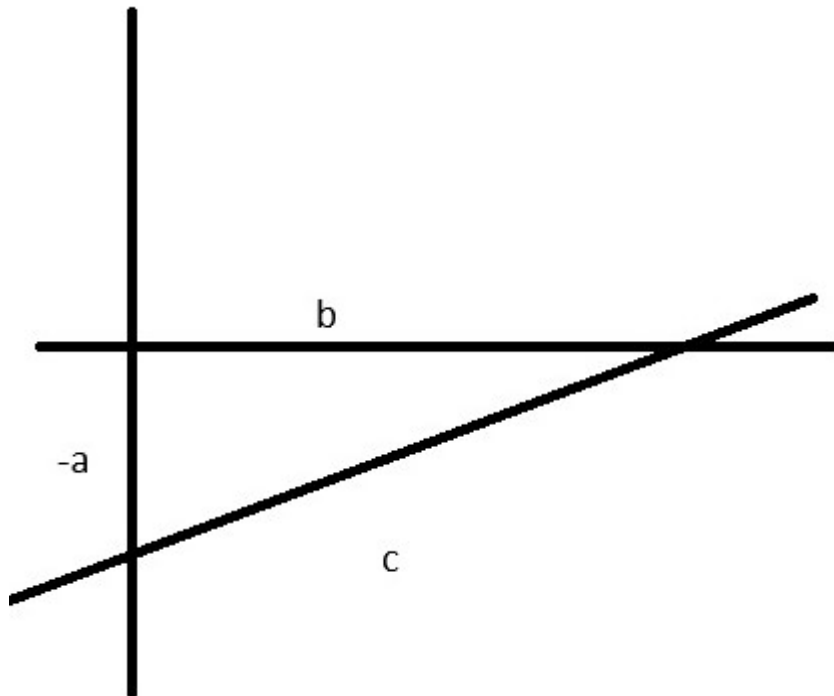


Illustration 1.1

In Illustration 1.1 can be imagined that  $c$  should have a shorter length because  $a$  has a negative value.

$$c = b - \Delta a$$

$$c = b - ((a^2 + b^2)^{0.5} - (c^2 - a^2)^{0.5})$$

$$\begin{aligned}
c^2 &= (b + (c^2 - a^2)^{0.5} - (a^2 + b^2)^{0.5})^2 \\
c^2 &= b^2 + b*(c^2 - a^2)^{0.5} - b*(a^2 + b^2)^{0.5} + b*(c^2 - a^2)^{0.5} + c^2 - a^2 - (c^2 - a^2)^{0.5}*(a^2 + b^2)^{0.5} - b*(a^2 + b^2)^{0.5} - (c^2 - a^2)^{0.5}*(a^2 + b^2)^{0.5} + a^2 + b^2 \\
0 &= 2*b^2 + 2*b*(c^2 - a^2)^{0.5} - 2*b*(a^2 + b^2)^{0.5} - 2*(c^2 - a^2)^{0.5}*(a^2 + b^2)^{0.5} \\
b^2 + b*(c^2 - a^2)^{0.5} &= b*(a^2 + b^2)^{0.5} + (c^2 - a^2)^{0.5}*(a^2 + b^2)^{0.5} \\
b*(c^2 - a^2)^{0.5} - (c^2 - a^2)^{0.5}*(a^2 + b^2)^{0.5} &= b*(a^2 + b^2)^{0.5} - b^2 \\
(c^2 - a^2)^{0.5}*(b - (a^2 + b^2)^{0.5}) &= b*(a^2 + b^2)^{0.5} - b^2 \\
(c^2 - a^2)^{0.5} &= (b*(a^2 + b^2)^{0.5} - b^2)/(b - (a^2 + b^2)^{0.5}) \\
c^2 - a^2 &= (b*(a^2 + b^2)^{0.5} - b^2)^2/(b - (a^2 + b^2)^{0.5})^2 \\
c^2 &= (b*(a^2 + b^2)^{0.5} - b^2)^2/(b - (a^2 + b^2)^{0.5})^2
\end{aligned}$$

so

$$c^2 = a^2 + b^2 * ((a^2 + b^2)^{0.5} - b)^2/((b - (a^2 + b^2)^{0.5}))^2 \text{ for } a < 0 \text{ and } b \geq 0$$

Similarly, we can obtain that

$$c^2 = a^2 * ((a^2 + b^2)^{0.5} - a)^2/(a - (a^2 + b^2)^{0.5})^2 + b^2 \text{ for } a \geq 0 \text{ and } b < 0$$

To convert that equation to a Complex Plane we can achieve by multiplying the proper part of the equation by 1

$$\begin{aligned}
c^2 &= a^2 + b^2 * ((a^2 + b^2)^{0.5} - b)^2/((b - (a^2 + b^2)^{0.5}))^2 * 1 \\
c^2 &= a^2 + b^2 * ((a^2 + b^2)^{0.5} - b)^2/((b - (a^2 + b^2)^{0.5}))^2 * (-1/-1) \\
c^2 &= a^2 + b^2 * ((a^2 + b^2)^{0.5} - b)^2/((b - (a^2 + b^2)^{0.5}))^2 * (i^2/i^2)
\end{aligned}$$

and we obtain

$$c^2 = a^2 + b^2 * ((a^2 + b^2)^{0.5}*i - b*i)^2/((b*i - (a^2 + b^2)^{0.5}*i))^2 \text{ for } a < 0 \text{ and } b \geq 0$$

and similarly

$$c^2 = a^2 * ((a^2 + b^2)^{0.5}*i - a*i)^2/(a*i - (a^2 + b^2)^{0.5}*i)^2 + b^2 \text{ for } a \geq 0 \text{ and } b < 0$$

and obtaining both equations that exist on Complex Plane.

The class that realizes the implementation of Complex Numbers:

```
public class ComplexNumber
{
    public double a { get; set; }
    public double b { get; set; }

    public ComplexNumber()
    {

    }

    public ComplexNumber(ComplexNumber cn)
    {
        a = cn.a;
        b = cn.b;
    }

    public ComplexNumber(double real)
    {
        a = real;
        b = 0.0;
    }

    public ComplexNumber(ComplexPythagorasNumber cpn)
    {
        ComplexNumber cn = ComplexPythagorasNumber.ConvertToComplexNumber(cpn);

        a = cn.a;
        b = cn.b;
    }

    public static ComplexNumber operator +(ComplexNumber a, ComplexNumber b)
    {
        ComplexNumber ret = new ComplexNumber();

        ret.a = a.a + b.a;
        ret.b = a.b + b.b;

        return ret;
    }

    public static ComplexNumber operator +(ComplexNumber a, double b)
    {
        return a + (new ComplexNumber(b));
    }

    public static ComplexNumber operator +(double a, ComplexNumber b)
    {
        return (new ComplexNumber(a)) + b;
    }
}
```

```

public static ComplexNumber operator -(ComplexNumber a, ComplexNumber b)
{
    ComplexNumber ret = new ComplexNumber();

    ret.a = a.a - b.a;
    ret.b = a.b - b.b;

    return ret;
}

```

```

public static ComplexNumber operator -(ComplexNumber a, double b)
{
    return a - (new ComplexNumber(b));
}

```

```

public static ComplexNumber operator -(double a, ComplexNumber b)
{
    return (new ComplexNumber(a)) - b;
}

```

```

public static ComplexNumber operator *(ComplexNumber a, ComplexNumber b)
{
    ComplexNumber ret = new ComplexNumber();

    ret.a = a.a * b.a - a.b * b.b;
    ret.b = a.a * b.b + a.b * b.a;

    return ret;
}

```

```

public static ComplexNumber operator *(ComplexNumber a, double b)
{
    return a * (new ComplexNumber(b));
}

```

```

public static ComplexNumber operator *(double a, ComplexNumber b)
{
    return (new ComplexNumber(a)) * b;
}

```

```

public static ComplexNumber operator /(ComplexNumber a, ComplexNumber b)
{
    ComplexNumber ret = new ComplexNumber();

    ret.a = (a.a * b.a + a.b * b.b) / (b.a * b.a + b.b * b.b);
    ret.b = (a.b * b.a - a.a * b.b) / (b.a * b.a + b.b * b.b);

    return ret;
}

```

```

public static ComplexNumber operator /(ComplexNumber a, double b)
{

```

```

    return a / (new ComplexNumber(b));
}

public static ComplexNumber operator /(double a, ComplexNumber b)
{
    return (new ComplexNumber(a)) / b;
}

public double R()
{
    return Math.Sqrt(a * a + b * b);
}
}

```

The class that realizes the implementation of Complex Pythagoras Numbers:

```

public class ComplexPythagorasNumber : ComplexNumber
{
    public static ComplexNumber ConvertToComplexNumber(ComplexPythagorasNumber cpns)
    {
        ComplexNumber cn = new ComplexNumber();

        cn.a = cpn.a;
        cn.b = (Math.Sqrt(cpn.a * cpn.a + cpn.b * cpn.b) - cpn.a) / (cpn.a - Math.Sqrt(cpn.a * cpn.a +
cpn.b * cpn.b));

        return cn;
    }

    public ComplexPythagorasNumber()
    {
    }

    public ComplexPythagorasNumber(ComplexNumber cn)
    {
        a = cn.a;
        b = cn.b;
    }

    public static ComplexPythagorasNumber operator +(ComplexPythagorasNumber a,
ComplexPythagorasNumber b)
    {
        return new ComplexPythagorasNumber(ConvertToComplexNumber(a) +
ConvertToComplexNumber(b));
    }

    public static ComplexPythagorasNumber operator -(ComplexPythagorasNumber a,
ComplexPythagorasNumber b)
    {
        return new ComplexPythagorasNumber(ConvertToComplexNumber(a) -
ConvertToComplexNumber(b));
    }
}

```

```

    }

    public static ComplexPythagorasNumber operator *(ComplexPythagorasNumber a,
ComplexPythagorasNumber b)
    {
        return new ComplexPythagorasNumber(ConvertToComplexNumber(a) *
ConvertToComplexNumber(b));
    }

    public static ComplexPythagorasNumber operator /(ComplexPythagorasNumber a,
ComplexPythagorasNumber b)
    {
        return new ComplexPythagorasNumber(ConvertToComplexNumber(a) /
ConvertToComplexNumber(b));
    }
}

```

The function that calculates the Fractal:

```

protected override DisplayNumber DetermineNumber(ComplexNumber z, ComplexNumber c, int
x, int y)
{
    DisplayNumber dn = new DisplayNumber();
    ComplexPythagorasNumber pzn, pz, pc;

    dn.x = x;
    dn.y = y;
    pz = new ComplexPythagorasNumber(z);
    pc = new ComplexPythagorasNumber(c);
    pzn = pz;

    for (int i = 0; i <= Amount; i++)
    {
        double tr = 0.0;

        pzn = pzn * pzn + pc;
        tr = ComplexPythagorasNumber.ConvertToComplexNumber(pzn).R();
        if (!(double.IsNaN(tr) || double.IsNegativeInfinity(tr) || double.IsPositiveInfinity(tr) ||
double.IsInfinity(tr)))
        {
            dn.R = tr;
            dn.Nr = i;
        }
        if (dn.R > 2.0)
        {
            if (!(dn.Escaped))
            {
                dn.r = dn.R;
                dn.nr = dn.Nr;
                dn.Escaped = true;
            }
        }
    }
}

```

```

        pzn = new
ComplexPythagorasNumber(ComplexPythagorasNumber.ConvertToComplexNumber(pzn));
    }
    return dn;
}

```

If the function present in the class ComplexPythagorasNumber has the following content:

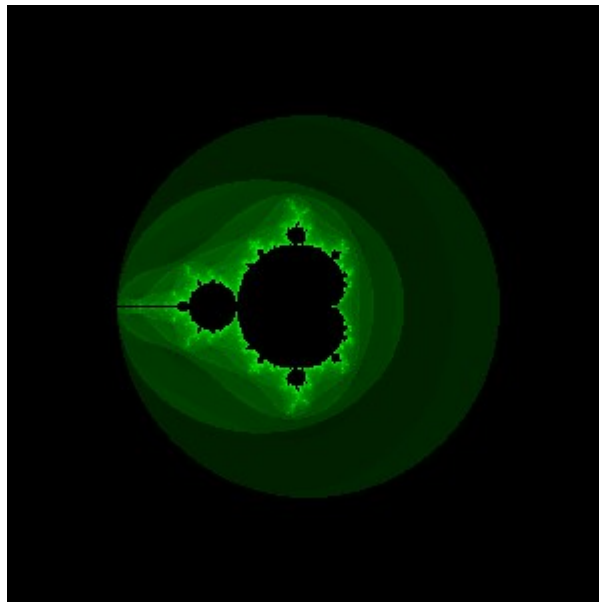
```

public static ComplexNumber ConvertToComplexNumber(ComplexPythagorasNumber cpns)
{
    ComplexNumber cn = new ComplexNumber();

    cn.a = cpn.a;
    cn.b = cpn.b;

    return cn;
}

```



Generated Fractal looks like in Illustration 2.1

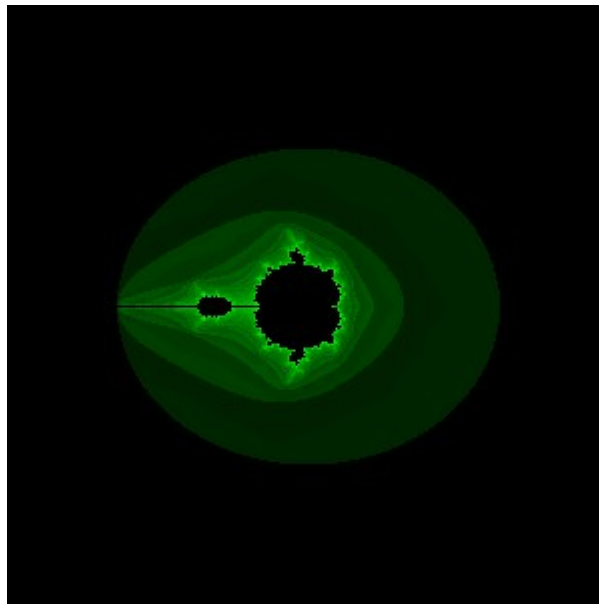


If the function present in the class ComplexPythagorasNumber has the following content:

```
public static ComplexNumber ConvertToComplexNumber(ComplexPythagorasNumber cpns)
{
    ComplexNumber cn = new ComplexNumber();

    cn.a = cpn.a;
    cn.b = cpn.b + 0.1 * cpn.b;

    return cn;
}
```



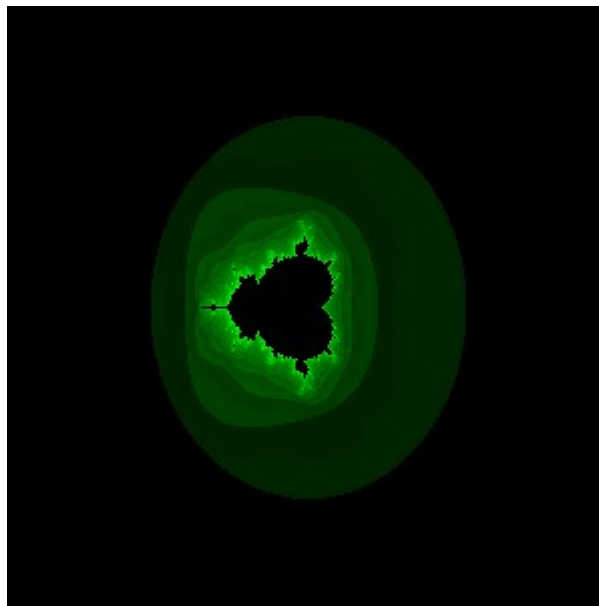
Generated Fractal looks like in Illustration 2.2

If the function present in the class ComplexPythagorasNumber has the following content:

```
public static ComplexNumber ConvertToComplexNumber(ComplexPythagorasNumber cpns)
{
    ComplexNumber cn = new ComplexNumber();

    cn.a = cpn.a + 0.1 * cpn.a;
    cn.b = cpn.b;

    return cn;
}
```



Generated Fractal looks like in Illustration 2.3

## References

<http://net.sonofgalaxy.shop>