

Methods for finding roots of polynomial equations .

Summary:

Methods used to determine the roots of (complex) polynomial functions are introduced. These methods use initial values (seed values) for the calculation of a root. Each root appears to be associated with a specific set of initial values. The article contains a short introduction of methods used to find the roots (solutions) of linear equations and more specifically the method of successive substitution, Wegstein's method, the method of Regula Falsi, Muller's method and the two Newton-Raphson methods.

Methods for finding the roots of linear equations .

Linear equations are equations of the form: $Y = F(X)$ where the function of X does not contain a factor or an element of the function (Y) itself.

An example of a linear functions is: $Y = 5X^2 - 3\sin(X) + 7X - 3$, while $Y = 5X^2 - 3\sin(Y) + 7XY - 3$ depicts a non-linear function. I have restricted myself to methods used to find the roots of linear functions. Those roots are defined by the solutions that satisfy the condition that $F(X) = 0$. Easy to use methods are those that use successive substitutions and the method of Wegstein. The method of successive substitution works as follows:

Suppose one has to find the (two) roots of the equation

$$F(X) = X^2 - X + C = 0. \quad (1)$$

In the method of successive substitution I use the equation $X_{n+1} = X_n^2 + C$ to find solutions for the equation (1). Starting with an appropriate initial or seed value for X_n , X_{n+1} can be calculated and its value substituted in X_n . Depending on the chosen initial value this process converges in many cases to a situation in which X_{n+1} does not change anymore. In that case one root of equation (1) has been found. It is possible to find the second root using a different initial value, but finding a different root is far from guaranteed. Wegstein proposed an alternative to the method of successive substitution. This variant will generate many more solutions than the strict way in which the newly found value X_{n+1} is inserted in the equation containing a function of X_n , as in

$$X_{n+1} = F(X_n) + C \quad (2)$$

The method of the successive substitution does not provide us with all of the possible solutions of the original equation (1)

$$F(X) = 0 \quad (3)$$

Wegstein's more successful method consists of inserting a value of X_{n+1} into equation (2), where X_{n+1} consists of a combination of the previous value of X and the newly calculated value of X .

That combination is achieved with the aid of a *relaxation factor*. The new value used to calculate the right hand side of equation (2) becomes $\rho \cdot X_n + (1-\rho) \cdot X_{n+1}$. In this case equation (2) is transformed into:

$$X_{n+1} = \rho \cdot X_n + (1-\rho) \cdot X_{n+1} \quad (4)$$

and in which where ρ is the relaxation factor. The difficulty in both the method of successive

substitution and Wegstein's method is the way in which multiple roots are to be found. The beauty of using the successive substitution is that it has a convergence speed that is often similar to that of the Newton-Raphson method without having to use the first derivative of the function I am trying to solve. The difficulty in applying this method consists of having to isolate X from its equation and bringing it to the right-hand side (or zero side) of the equation. This difficulty is best shown in solving the equation:

$$\mathbf{X^3 - 2 = 0} \tag{5}$$

This equation can be written as:

$$\mathbf{X = X^3 + X - 2} \tag{5a}, \text{ or}$$

$$\mathbf{X = (2 + 5X - X^3)/5} \tag{6}$$

Applying the method of successive substitution, equation (5) quickly diverges, no matter what initial value (or seed value) of X is being used. Equation (6) however generates a solution with an accuracy of 5 fractional digits in 4 steps. A similar problem may be encountered with the use of relaxation factors. Applying Wegstein's method to many different equations shows that most solutions could be produced by applying high relaxation factors. A more general and modified Wegstein method incorporates the modification of the equation as is shown in (6). Its representation is:

$$\mathbf{Z_{n+1} = \rho \cdot Z_n + (1-\rho) \cdot \{F(Z_n) + (\alpha-1) \cdot Z_n\} / \alpha} \tag{6a}$$

A third method is the method of Regula Falsi. This method is also an iterative method. The new value of X is calculated using the two previously calculated values of X. The method is formulated as:

$$\mathbf{X_{n+1} = [X_{n-1} \cdot F(X_n) - X_n \cdot F(X_{n-1})] / [F(X_n) - F(X_{n-1})]} \tag{7}$$

The difficulty is that one has to start with two seed values, one for X_n and one for X_{n-1} .

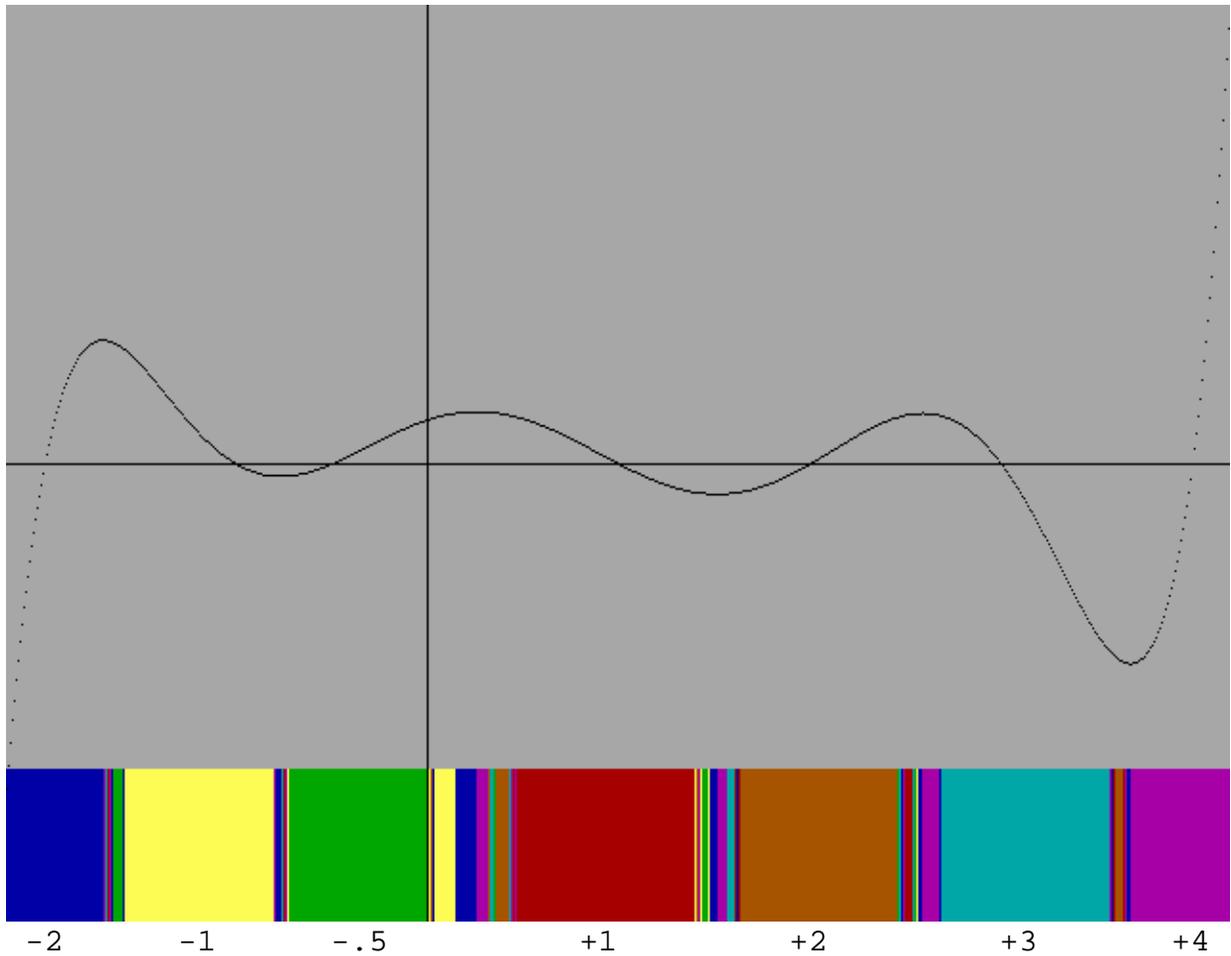
A more efficient approach is offered by the widely used Newton-Raphson method. In this iterative method the new value for X is calculated using the previously calculated value of X, F(X) and F'(X), the first derivative of F(X).

$$\mathbf{X_{n+1} = X_n - F(X_n) / F'(X_n)} \tag{8}$$

Where the methods of successive substitution and Wegstein are reasonably simple and straightforward, Regula Falsi and Newton-Raphson do need the use of resp. the difference between two calculated values and the first derivative of the function whose roots one tries to determine. The method of successive substitution is used widely in the creation of fractals. The method of Newton-Raphson is without any doubt the most used and useful method. I am using a simple equation to show how the method works for determining the real roots of higher order polynomials. The example employed here is the following; determine the roots of the equation:

$$\mathbf{X^7 - 6.5X^6 + 3.5X^5 + 38.5X^4 - 38.5X^3 - 56X^2 + 34X + 24 = 0} \tag{9}$$

The roots of this equation are: -2, -1, -0.5, +1, +2, +3, and +4 (colored blue, yellow, green, red, brown, cyan, and magenta). The equation is visualized in the figure below. The diagram shows that the function intersects the X-axis of an XY-coordinate system at the seven points that represent the



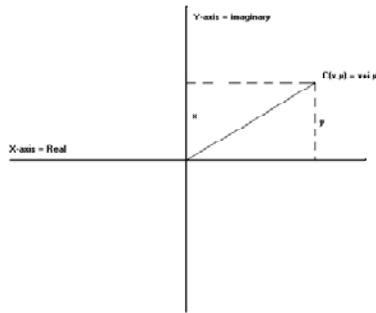
X-values for which the Y-value of the equation (9) is zero. The colored bar at the bottom of the figure is associated to the X-axis and represents the values of the seed values of X for which Newton-Raphson finds similar roots.

In finding the roots I used as seed values the X-coordinates between the minimum and maximum value of X and move along the X-axis with steps of the size $(\max.X - \min.X)/640$. (The number 640 represents the amount of horizontal pixels of the screen. Newton-Raphson takes a seed value and determines the associated point of the curve which is the intersection of a line perpendicular to the X-axis, has a point with the seed value on that line and the curve. It then constructs a line tangent to this intersecting point and determines the intersection of the tangential line with the X-axis. This is the new seed value. This process continues until the difference between the lastly found seed value of X and the previously determined value of X is smaller than a given accuracy. The figure shows how fast the method works in areas where the curve behaves like an almost straight line. As soon as the minimum and maximum portions of the curve (areas where the first derivative of the function start to deviate notably) are approached the tangential line will intersect with the X-axis at points (new seed values) that will result in an iterative process converging to a different root than the root which is closest to the initial seed values chosen or lastly calculated. There is one problem though. It is possible that the iteration will cycle around the solution. In that case the number of iterations might

exceed a specified number of maximum iterations and the search for a solution will end. Use of a relaxation factor may be needed. See the last chapter of this article.

I used the method of Newton- Raphson in order to demonstrate the beauty of patterns that are created when applying this method to finding roots of higher degree complex polynomial equations (See next chapter). For that specific purpose I will focus on finding the roots of complex polynomials, that is polynomials of the form:

$$a_n Z^n + a_{n-1} Z^{n-1} + \dots + a_2 Z^2 + a_1 Z + a_0 = 0 \tag{10}$$



Here Z is a complex number consisting of a real and an imaginary part: $Z = X + i.Y$ where $i = \sqrt{-1}$ and X and Y are real numbers. Complex numbers are represented in the Euclidian two-dimensional space as coordinates whose real number is represented by the value of the X-coordinate and whose imaginary number is represented by the value of the Y-coordinate. The absolute value of a complex number is its distance to the origin, being $\sqrt{X^2+Y^2}$. Real numbers coincide with its values of the X-axis and imaginary numbers with the Y-axis. The absolute value of complex numbers is often used to determine whether an iterative

process has reached a required accuracy. This is done with comparing the value of a newly calculated value with the previously computed value. If the difference between these values is smaller than a given accuracy, the iteration process comes to a halt and the lastly calculated value is supposed to represent the required solution. I choose however to compare the values of both the real and imaginary parts of the lastly and previously calculated complex number and the imaginary parts of these complex numbers. If one would calculate the roots of an equation like $Z^n - 1 = 0$ then the absolute values of all n roots would be the same (approximately 1) and it would not distinguish between the results of the lastly calculated complex number and the previously calculated number if these two refer to different roots of said equation.

I also used the extended version of Newton-Raphson. This involves the first and second derivative of the polynomial equation. It is claimed that convergence of the iterative process is faster than that of the normal Newton-Raphson. However, I observed that the convergence speed is slowed down by the extra calculations needed for this approach. The method finds all the roots of higher-degree complex polynomials. Its formula reads as follows:

$$Z_{n+1} = Z_n - \frac{F(Z_n)}{F'(Z_n)} - \frac{F(Z_n)^2 * F''(Z_n)}{2 * F'(Z_n)^3} \tag{11}$$

I finally turned to Muller's method. The extensive calculations involved with that method made me assume that Muller's method would never be able to compete with the method of Newton-Raphson. Since this method is often referred to in the literature I decided to investigate its usefulness in the case of high degree polynomials with complex coefficients.

The use of Muller's method can be described as follows:

Determine the roots of:

$$F(Z, C) = \sum_{n=0}^{n=k} C_n Z^n ; C \text{ and } Z \text{ are complex numbers; degree of polynomial is } k$$

With Muller's method one defines a quadratic (parabolic) curve through three points on the curve defined by the complex polynomial function $F(Z)$.

$$F(Z) = \frac{(Z - Z_{n-2}) \cdot (Z - Z_{n-1})}{(Z_n - Z_{n-2}) \cdot (Z_n - Z_{n-1})} * F(Z_n) + \frac{(Z - Z_{n-2}) \cdot (Z - Z_n)}{(Z_{n-1} - Z_{n-2}) \cdot (Z_{n-1} - Z_n)} * F(Z_{n-1}) + \frac{(Z - Z_{n-1}) \cdot (Z - Z_n)}{(Z_{n-2} - Z_{n-1}) \cdot (Z_{n-2} - Z_n)} * F(Z_{n-2}) \quad (12)$$

Now the function $F(Z)$ is intersected with the X-axis. This means:

$$(Z - Z_{n-2}) \cdot (Z - Z_{n-1}) \cdot H_n + (Z - Z_{n-2}) \cdot (Z - Z_n) \cdot H_{n-1} + (Z - Z_{n-1}) \cdot (Z - Z_n) \cdot H_{n-2} = 0 \quad (13)$$

where:

$$H_n = F(Z_n) / \{(Z_n - Z_{n-2}) \cdot (Z_n - Z_{n-1})\} \quad (13a)$$

$$H_{n-1} = F(Z_{n-1}) / \{(Z_{n-1} - Z_{n-2}) \cdot (Z_{n-1} - Z_n)\} \quad (13b)$$

$$H_{n-2} = F(Z_{n-2}) / \{(Z_{n-2} - Z_{n-1}) \cdot (Z_{n-2} - Z_n)\} \quad (13c)$$

Multiplying (13a), (13b), and (13c) with $(Z_n - Z_{n-1}) \cdot (Z_n - Z_{n-2}) \cdot (Z_{n-1} - Z_{n-2})$ gives us

$$G_n = F(Z_n) \cdot (Z_{n-1} - Z_{n-2}) \quad (14a)$$

$$G_{n-1} = F(Z_{n-1}) \cdot (Z_{n-2} - Z_n) \quad (14b)$$

$$G_{n-2} = F(Z_{n-2}) \cdot (Z_n - Z_{n-1}) \quad (14c)$$

This gives

$$F(Z) = \{Z^2 - Z \cdot (Z_{n-2} + Z_{n-1}) + Z_{n-1} \cdot Z_{n-2}\} \cdot G_n + \{Z^2 - Z \cdot (Z_{n-2} + Z_n) + Z_n \cdot Z_{n-2}\} \cdot G_{n-1} + \{Z^2 - Z \cdot (Z_n + Z_{n-1}) + Z_{n-1} \cdot Z_n\} \cdot G_{n-2} \quad (15)$$

and (15) becomes zero when intersecting this function with the X-axis

$$(G_n + G_{n-1} + G_{n-2}) \cdot Z^2 - \{(Z_{n-2} + Z_{n-1}) \cdot G_n + (Z_{n-2} + Z_n) \cdot G_{n-1} + (Z_n + Z_{n-1}) \cdot G_{n-2}\} \cdot Z + \{Z_{n-1} \cdot Z_{n-2} \cdot G_n + Z_n \cdot Z_{n-2} \cdot G_{n-1} + Z_{n-1} \cdot Z_n \cdot G_{n-2}\} = 0 \quad (16)$$

Finally the complex numbers a, b and c are determined:

$$\mathbf{a} = \mathbf{G}_n + \mathbf{G}_{n-1} + \mathbf{G}_{n-2} \quad (17)$$

$$\mathbf{b} = (\mathbf{Z}_{n-2} + \mathbf{Z}_{n-1}) \cdot \mathbf{G}_n + (\mathbf{Z}_{n-2} + \mathbf{Z}_n) \cdot \mathbf{G}_{n-1} + (\mathbf{Z}_n + \mathbf{Z}_{n-1}) \cdot \mathbf{G}_{n-2} \quad (18)$$

$$\mathbf{c} = \mathbf{Z}_{n-1} \cdot \mathbf{Z}_{n-2} \cdot \mathbf{G}_n + \mathbf{Z}_n \cdot \mathbf{Z}_{n-2} \cdot \mathbf{G}_{n-1} + \mathbf{Z}_{n-1} \cdot \mathbf{Z}_n \cdot \mathbf{G}_{n-2} \quad (19)$$

and solve the equation:

$$\mathbf{a} \cdot \mathbf{Z}^2 - \mathbf{b} \cdot \mathbf{Z} + \mathbf{c} = \mathbf{0} \quad (20)$$

where:

$$\mathbf{Z} = \{\mathbf{b} \pm \sqrt{\mathbf{b}^2 - 4 \cdot \mathbf{a} \cdot \mathbf{c}}\} / 2\mathbf{a} \quad (21)$$

In the iteration process of Muller's method \mathbf{Z} is the new value \mathbf{Z}_{n+1} . \mathbf{Z}_{n-1} replaces \mathbf{Z}_{n-2} ; \mathbf{Z}_n replaces \mathbf{Z}_{n-1} ; and \mathbf{Z}_n now becomes the calculated \mathbf{Z}_{n+1} .

The difficulty is to determine what sign of formula (21) has to be used. The strategy I employed is to determine first of all the two values \mathbf{Z} may assume in elaborating formula (21) and then chose the value which is closest to \mathbf{Z}_n . As seed values I used a frame that contains the world coordinates that are associated with the pixels of the display screen (640 by 480 values). If not all the roots are found in the frame that contains the grid with the seed values that are associated to this grid, the process is repeated with the smallest value of \mathbf{Z}_{n+1} produced by formula (21). If still being unsuccessful one can once more run all the seed values defined by the values associated to the grid values and try the largest value of \mathbf{Z}_{n+1} .

The convergence exponents of the methods are:

- | | |
|-----------------------------|------|
| 1. Successive substitution: | 1 |
| 2. Wegstein | 1 |
| 3. Modified Wegstein | 1 |
| 4. Regula Falsi | 1.62 |
| 5. Muller | 1.84 |
| 6. Newton-Raphson | 2 |
| 7. Newton-Raphson extended | 3 |

The easiest of these methods are the first three. However, as I have demonstrated, it is difficult to find all the roots. In my used examples this method only found one root of a nine-degree complex equation. Methods 4 and 5 needs initial values of resp. 2 and 3 points, while for methods 6 and 7 resp. the first derivative, and the first and second derivative have to be determined.

Visualizing the loci of seed values for the investigated methods.

Since the method of Wegstein, and its modified version generates only one root, I did not pursue any further efforts in finding more roots with the methods of successive substitution (Wegstein and modified Wegstein) but focused entirely on Newton-Raphson, Regula Falsi and Muller.

For these methods I used the equation

$$F(Z) = Z^9 - C.Z^6 + C.Z^3 - Z + C = 0, C = -.37 - .63i \tag{22}$$

With Newton-Raphson one needs to determine the first derivative. This makes the calculation a little more complex than is the case with successive substitution. However, according to my investigations, NR always produces all the roots of complex polynomial equations.

The roots are calculated by assigning world coordinates for complex numbers to the screen coordinates (640x480) or pixels. The choice of those coordinates may influence the fact whether or not all roots can be found. It is safe to specify them as an area that would include those roots. In reality it turns out that most choices will always find all roots. Exceptions will be treated later. I applied the NR-method by using all the complex numbers that are assigned to the pixels. As soon as a root is found it will be placed in an array, and when finding similar roots, these will be discarded. This process stops as soon as all roots are found.

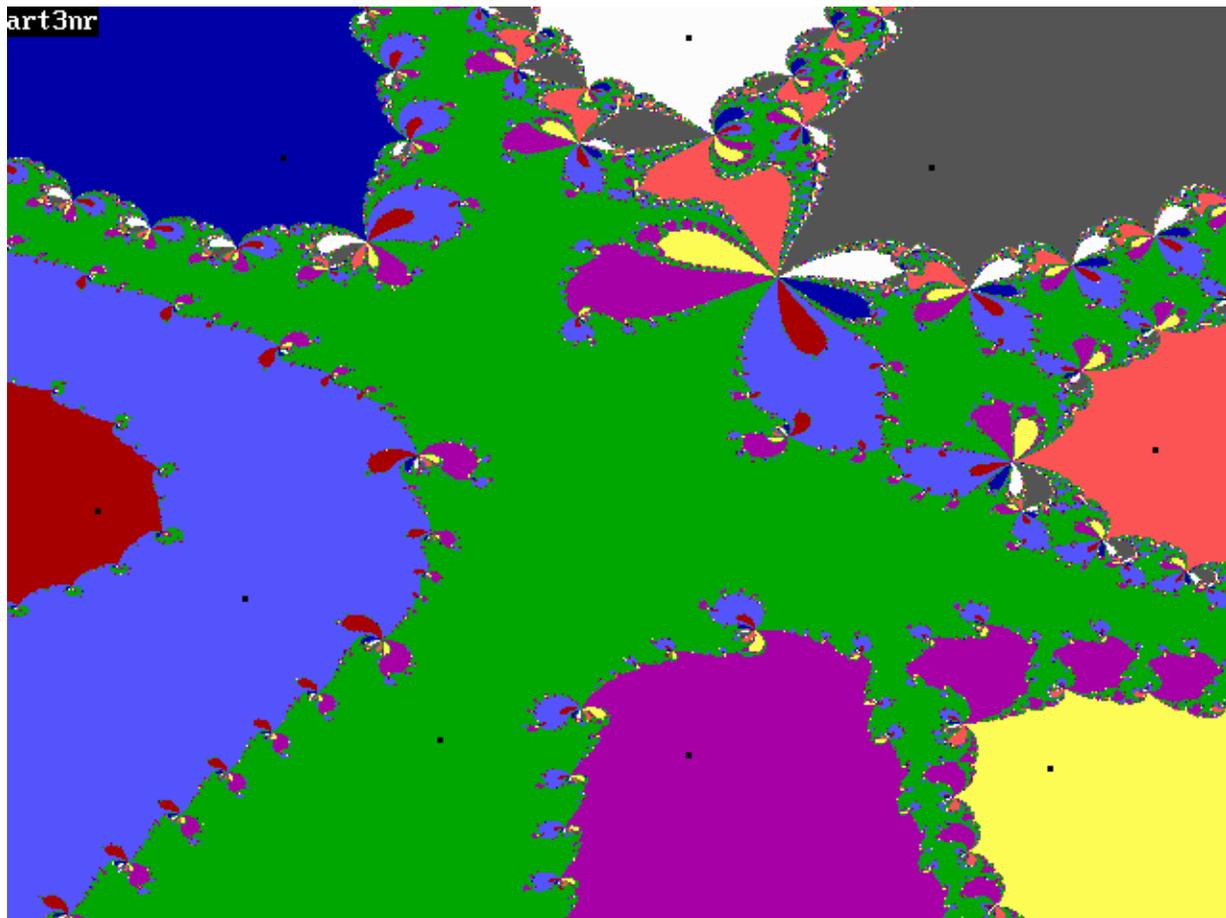
With NR the roots of formula (22) and using as seed values the world coordinates between (-1.2, 1.2, 1.2, -1.2) for respectively the minimum real value, the maximum real value, the maximum imaginary value, and the minimum imaginary value or (rmin, rmax, imax, imin), I found the following roots (stored in symbolic address rtart3)

1	-.6596836704294642	.8045808075081935
2	-.354127539113212	-.7182412317552311 *)
3	-.7348577419930283	-.3524344430808881
4	-1.023330372933215	-.1204092667014445
5	.1359958049606741	-.7578194705211022
6	.8427186349667669	-.7933247215131565
7	.1336872350533339	1.119929073163079
8	.6102546527271762	.7788095958421002
9	1.049342996739447	.0389096570005479

I also calculated the roots based on formula (6a). For $\rho = 0$ and $\alpha = 2, 3, 5$ and 6 only one root is found in the window of the world coordinates (-1, 1, 1, -1). For $\rho = .9$ also the same root is found. Other tried combinations of ρ and α were so far unsuccessful. The found root is the 2nd root from the abovementioned list, indicated with *)

I next generated a picture in which nine different colors will be assigned to the found roots. This also

means that one has to determine the roots using all seed values assigned to all 640x480 pixels. I used here the same world coordinates as used in finding the roots $(-1.2, 1.2, 1.2, -1.2)$. The generated pictures are placed in symbolic address art3nr. The result is graph art3nr. The black spots indicate the location of the nine roots. The large colored areas would extend

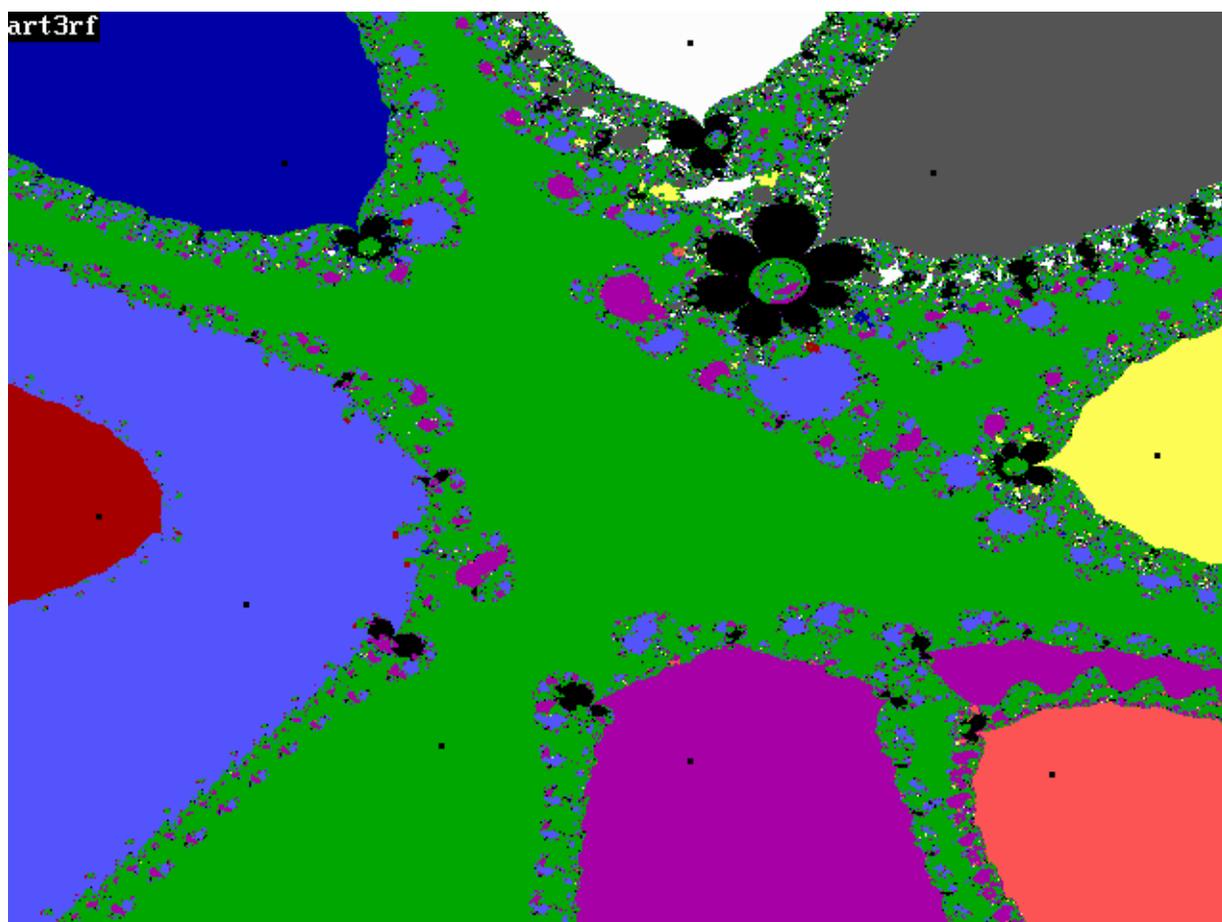


indefinitely if the window of the world coordinates would be made larger and it would then seem as if the boundaries of these areas follow straight lines. Each area with the same color is the locus of all seed values that, when applying the NR-method, would converge to the same root. I refer to these areas as “leafs” and discriminate between high-order leafs (the large areas of the picture) and lower-order leafs (the smaller areas in the picture). In many articles the pictures generated with this type of approach are referred to as fractals. In reality none of these is a result of fractional dimensions but is a result of the behavior of the iteration process which can jump from leaf to leaf with the same color. It is a behavior that one would also find when finding the roots of a polynomial equation where these roots would be real (not complex) numbers. The generated picture consists thus of a set of loci of seed values that will iterate to a given root with the Newton-Raphson method, called here an NR-polynomiograph (The term polynomiograph was coined by Prof. Bahman Kaltari). The method of Regula Falsi. This method is also an iterative method. The new value of Z is calculated using two

previously calculated or specified values of Z .

The difficulty is that one has to start with two seed values, one for Z_n and one for Z_{n-1} . Since we are working here with the first difference between the functions of these two values, Regula Falsi has an iteration pattern that is similar to that of Newton-Raphson where the first derivative of the function of Z has to be used. Using the same roots of formula (22) and the same world coordinates as used in the previous chapter the RF-polynomiograph (Regula Falsi polynomiograph) is similar to the NR-polynomiograph.

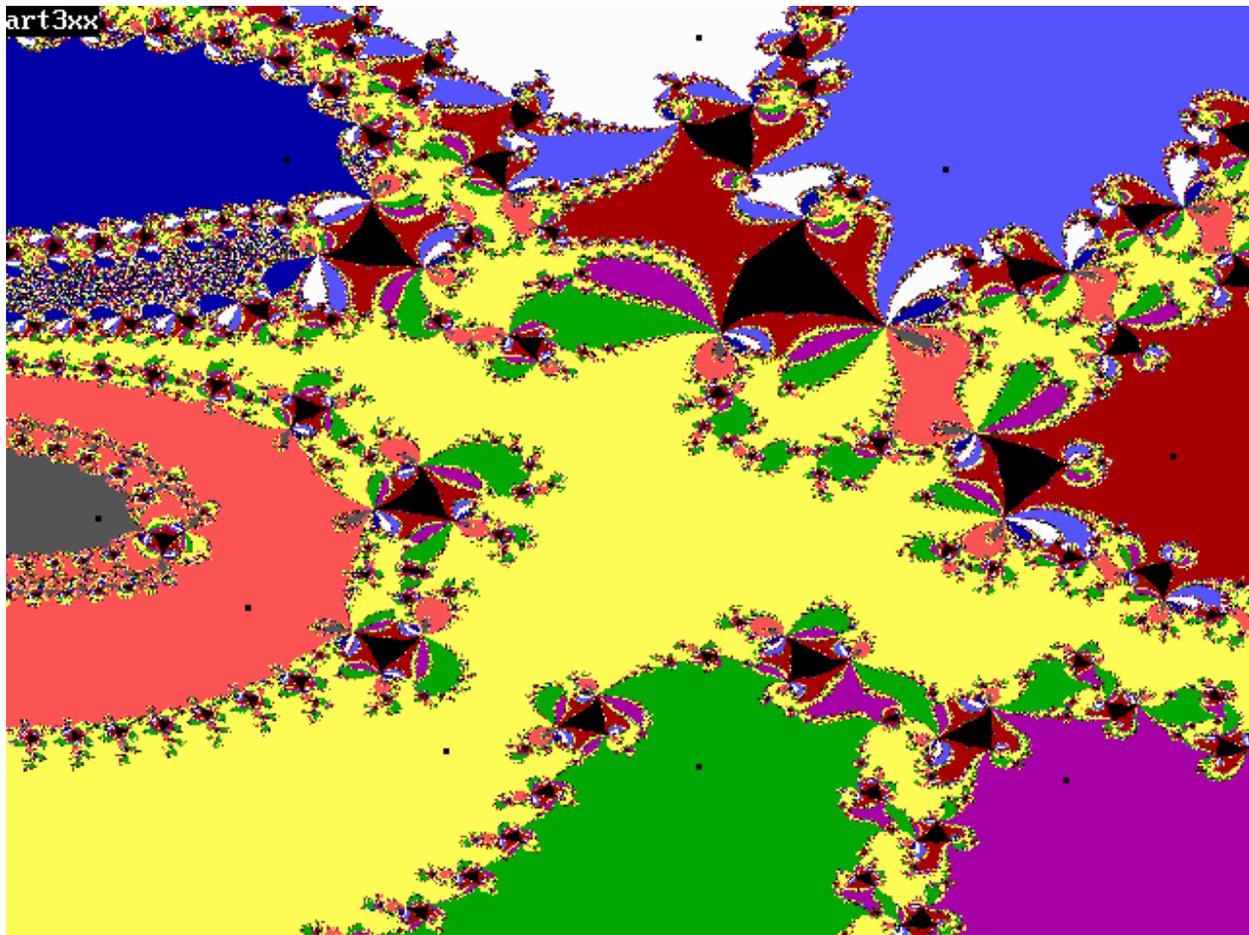
In this RF-polynomiograph the black leafs (black areas other than the small black squares that indicate the locations of the roots) means that Regula Falsi did not converge there to any root. This may have been influenced by the choice of the initial two seed values of Z_n and Z_{n-1} . The RF-polynomiograph is stored under the name art3rf.



In the extended Newton-Raphson method a term with the second derivative has been added to the original Newton-Raphson formula.

This method is also applied to the equation of formula (22). In a real/imaginary window of coordinates defined by $(-1.2, 1.2, 1.2, -1.2)$ (rmin, rmax, imax, imin) and a screen of 640x480 pixels all 9 roots were found. The roots are placed in a list, and the visual representation in are placed in a storage address with symbolic name art3xx. The roots are the same roots as shown on page 7, but they were found in a different order. The little black square dots indicate the location of the roots. The larger black areas are areas of seed values with which a root was not found.

The NRX-polynomiograph (Newton-Raphson eXtended polynomiograph) is shown below. These roots are the same roots as shown on page 7.



Muller's method is the most elaborate of all the methods that I investigated with the objective of finding the roots of high degree complex polynomials and generating polynomiographs of the results.

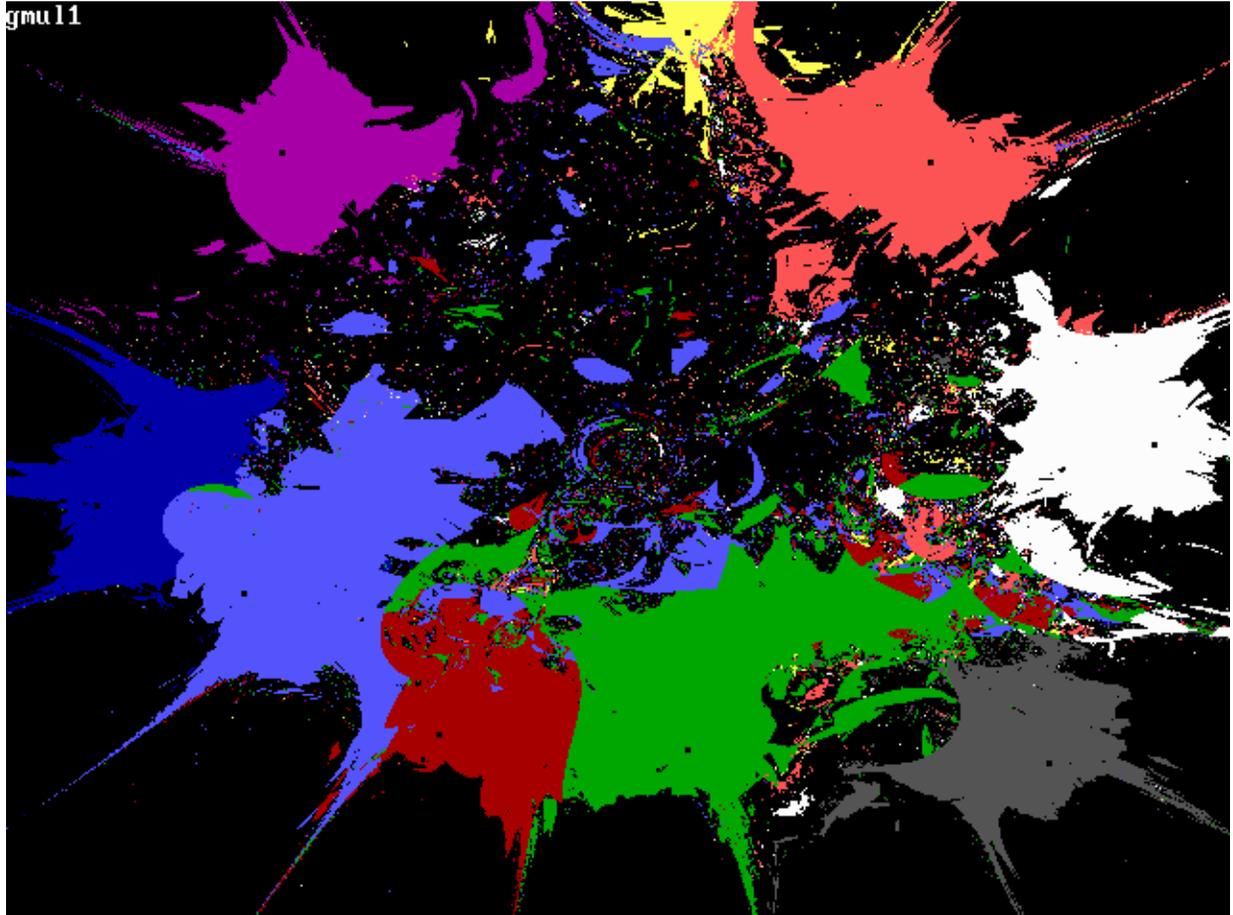
The values of Z are calculated by assigning world coordinates for complex numbers to the screen coordinates (640x480) in the same way as I did in the case of the previous methods. In this way I examined all 640x480 seed values for Z_n . The values of Z_{n-1} and Z_{n-2} are derived from Z_n by making them slightly different from Z_n .

This quadratic equation results in two values for Z , which is the new seed value in the iteration process for finding one of the roots of the polynomial. I chose one of these values by comparing them to the lastly found Z and continue with the value which is closest to that lastly found value. In case one doesn't find all the roots the iteration process has to be repeated, starting from the initially chosen seed value(s) and pick the Z value which is the result of applying the positive sign in the quadratic equation (12). One can exhaust the search for all the roots by repeating the iteration process once more and choosing the other sign of the quadratic equation.

With Muller's method I have found the roots of formula (22) with $C = -.37-.63i$, and using world coordinates between (-1.2, 1.2, 1.2, -1.2) for resp. r_{min} , r_{max} , i_{max} , i_{min} . As expected, the roots are similar to those found with Newton-Raphson or Regula Falsi.

The generated polynomiograph (MU-polynomiograph) is totally different from the NR-polynomiograph and RF-polynomiograph. The boundaries between the colored areas are sharp and the picture exhibits a rather chaotic character. Many seed values did not result in finding any root (the black areas in the MU-polynomiograph with the symbolic name $gmull$). This may have been caused by the limit that was set in the number of allowed iteration steps. It is also possible that the used strategies where I had to make a choice in determining which value of the two values that are the result of evaluating formula (21) had to be used. A random selection of the use of the plus- or minus-sign in calculating Z in formula (21) might give different results. Due to the complication of making these choices the time necessary for generating the MU-polynomiograph is extremely long. However, the program that calculates the roots is almost as fast as the program used in the Newton-Raphson approach, despite the more elaborate computations that are involved in Muller's method.

The MU-polynomiograph for the complex equation of formula (22) is shown below. Also here the little black square boxes in the larger colored patterns indicate the location of the roots.



More about relaxation factors.

The use of relaxation factors was introduced in the beginning of this article where I discussed Wegstein's method. It is possible that solutions can not be found with the methods of Newton-Raphson. (See the black holes in figure art3xx on the next page) In those cases relaxation factors can be introduced to force solutions. The relevant formula for those two NR-methods will then become:

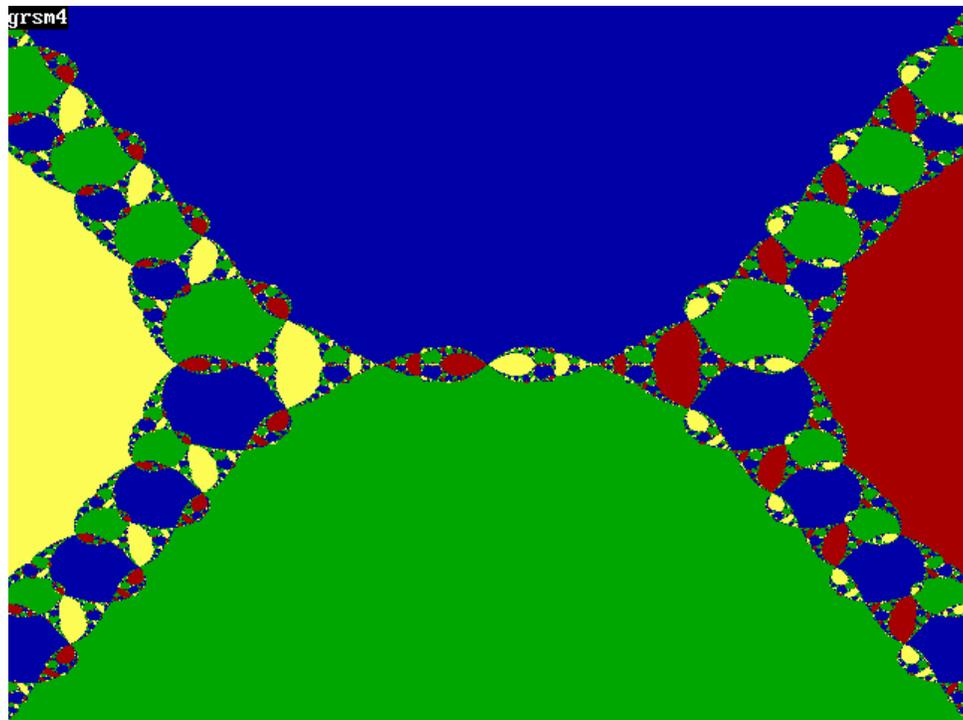
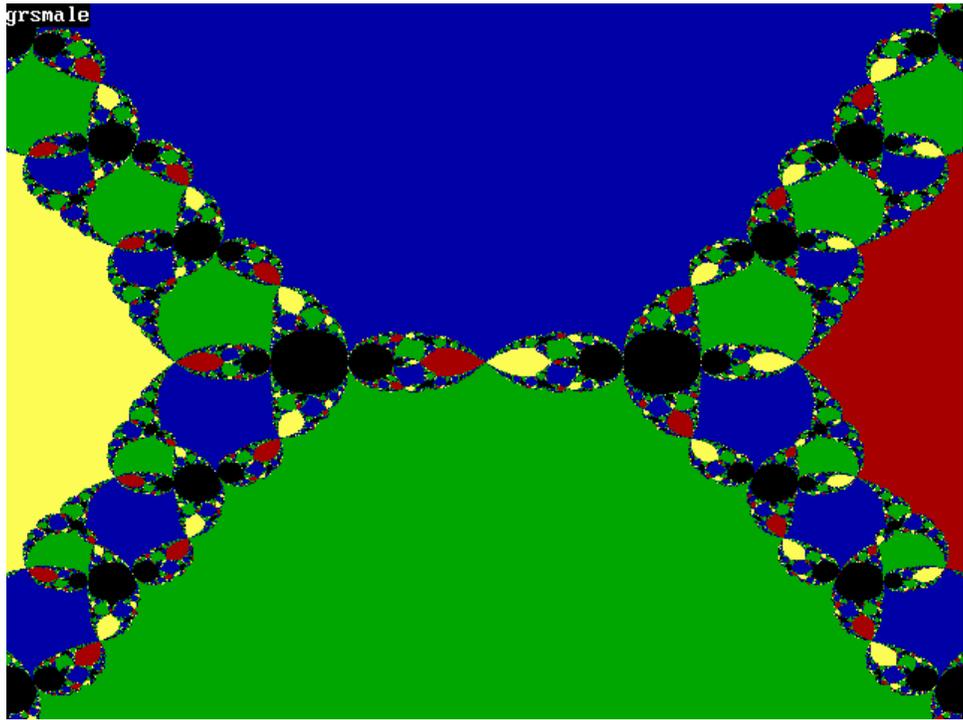
$$X_{n+1} = X_n - \rho \times F(X_n) / F'(X_n) \quad (8a) \text{ and}$$

$$Z_{n+1} = Z_n - \rho \{ F(Z_n) / F'(Z_n) - F(Z_n)^2 * F''(Z_n) / 2 * F'(Z_n)^3 \} \quad (11a)$$

where ρ denotes the relaxation factor that must have a value greater than zero and smaller than 1. The effects of the use of a relaxation factor in the NR-method of eq. 8a is shown in the two following polynomiographs. They represent the loci of the seed values of the four roots of the equation

$$Z^4 - 0.84Z^2 - 0.16 = 0 \quad (23)$$

The names of the polynomiographs are resp. grsmalez and grsm4z. $\rho = 0.75$ in the second graph.



References.

- (1) **Metamagical Themas** by *Douglas R. Hofstadter*.
Bantam Books 1985
ISBN 0-553-34279-7
- (2) **Introduction to Numerical Analysis** by *Carl-Erik Fröberg*
Addison-Wesley Publishing Company, Inc. Reading - Palo Alto - London, 1965
Library of Congress No. 64-22152
- (3) **Research Notes** of *Jakob Vlietstra*
Philips Research, Eindhoven, 1963-1966
- (4) **Handbook of Mathematical Functions** by Milton Abramowitz and Irene A. Stegun
Dover Publications, Library of Congress Card 65-122/53
1964
- (5) **A New Visual Art Medium: Polynomiography** by *Bahman Kalantari*,
in *Computer Graphics*, Volume 38 Number 3 August 2004.