Finding roots for *f(z) = z^z^z+1*
- an exploration going on over some years

1 Finding roots for $f(z)=z^{z^z}+1$

This question has been asked in the forum MSE in 2015 [1], and by opportunity I took that question to give it a chance. In the following I document the steps of explorations.

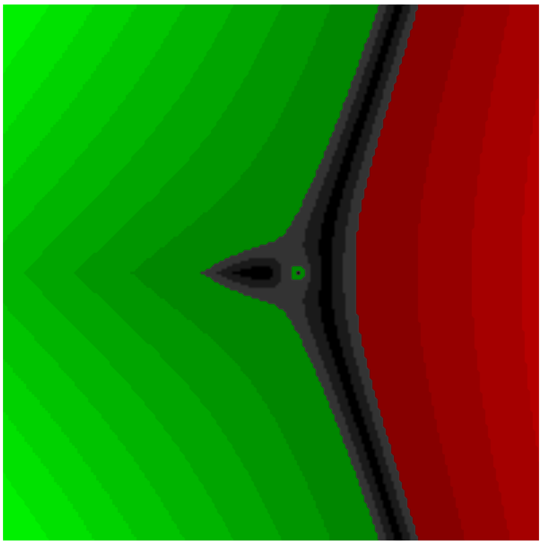
1.1 Start with roots for g(z)

To get a first insight in the problem I made computations easier by looking at the double logarithm of *f(z)*:

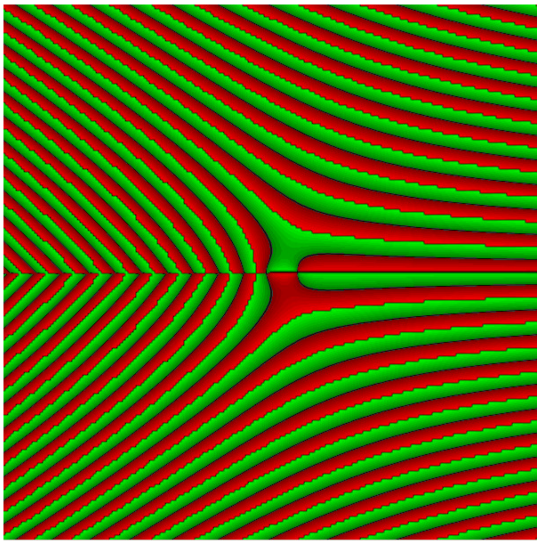
$$g(z) = \log(\log(z\hat{}z\hat{}z)) - \log(\log(-1)) \qquad \text{def: } lz=\log(z),\ ll\_1 = \log(\log(-1))$$
$$=\log(lz\cdot \exp(lz\cdot z))-ll\_1$$
$$=llz + lz\cdot z - ll\_1$$

The real and imaginary parts for the evaluation of *g(z)* at complex *z* are shown as distances from *zero*, where green color indicates negative values, red color positive value, and gray/black color values near *zero*. Darker grey is nearer *zero*, lighter green/red farther from *zero*.

REAL (g(z))



IMAG (g(z))



In each picture the respective zeros seem to be arranged in certain shapes because they must occur on the borders between green and red (thus in the black regions).

In the left picture, on the real values of *g(z)*, we see that the potential candidates for the roots, (numbers *z* for which the ***real part*** of *g(z)* becomes zero) are found in that black region which looks vertically like a boomerang having a triangular appendix to the left.

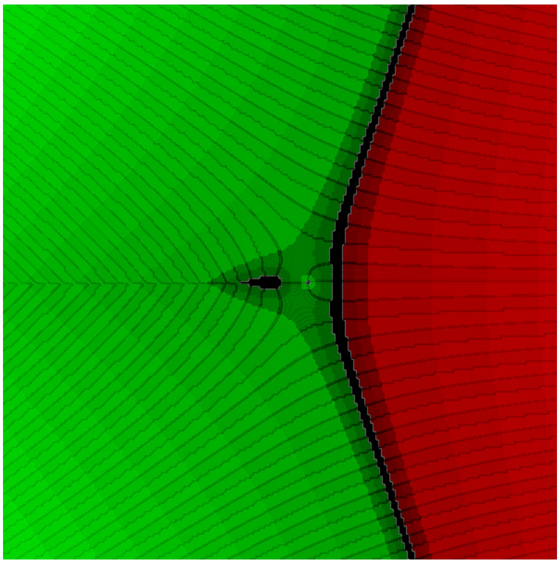
In the right picture, the candidate roots (which make the ***imaginary parts*** of *g(z)* zero) follow lines which are in the tendency horizontally (and appear also curved). They occur with some periodicity.

Because zeros of *g()* can only occur, where both pictures show their zeros ***simultanously***, an overlay of that two pictures should make it better visible, where roots of the function *g()* can reside at all: in that boomerang region and vertically with discrete distance and roughly periodically.

---

[1] *This exercise was triggered by the question "**Solve x^x^x=-1**" on math.stackexchange.com 30. Aug 2015 , see* https://math.stackexchange.com/q/1415029/ *and improves and supersedes my there given answers!*

Overlay of the left picture with the black lines of the right picture (the black region of the right picture is also a bit more detailed)

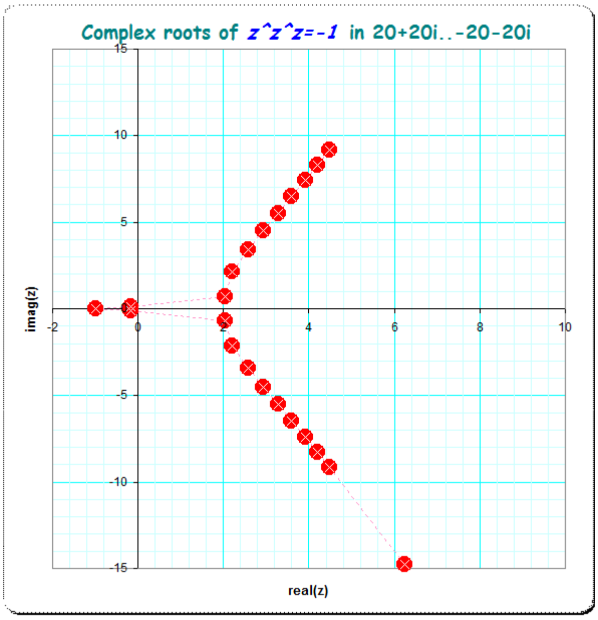


Important disclaimer: the pictures are not "exclusive" w.r.t the existence of roots for *f(z)* : I only compare ***the equality of the double-logarithms*** which occur in the evaluation of *g(z)* and because of multivaluedness of logarithms they form only a subset of all roots for *f(z)* !

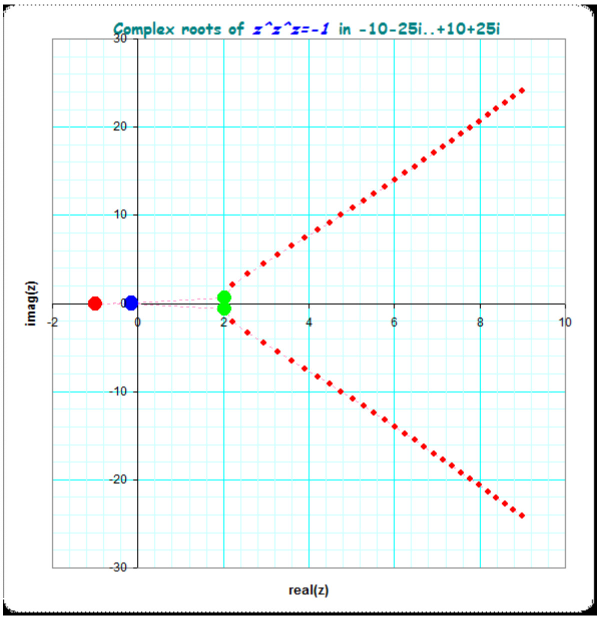
1.2 Tracing for more roots for g(z)

Based on that visible impression, I built a tracer which searches further roots for *g()*. First it found two more roots ρ_3,ρ_4 towards the right halfplane nearly on a linear line with ρ_2. From that it extrapolates the initial list of three roots ρ_2,ρ_3,ρ_4 based on the direction and distances of their occurences, and traces the way along the boomerang-region to outwards. That reduced the searchspace much and made the numerical search much more efficient. Quickly one could find first -say- *20* new roots, and getting trustful on the mathematical principle even simply new *200* roots, suggesting that this is an obvious principle generalizable for an infinite set of roots, all roughly (but increasingly!) linearly aligned.

```
      ρk
 -----------------------------------------
ρ0  = -1
ρ1  ≈ -0.15890875158 + 0.0968231909176*I
ρ2  ≈  2.03426954187 + 0.678025662373*I
ρ3  ≈  2.21022616044 + 2.14322152216*I
ρ4  ≈  2.57448299040 + 3.39212026316*I
ρ5  ≈  2.93597198855 + 4.49306256310*I
ρ6  ≈  3.27738123699 + 5.51072853255*I
ρ7  ≈  3.60013285730 + 6.47345617876*I
ρ8  ≈  3.90713751281 + 7.39619042452*I
ρ9  ≈  4.20091744993 + 8.28794173821*I
ρ10 ≈  4.48346951212 + 9.15465399776*I
ρ11 ≈  4.75636133031 + 10.0005052039*I
  ...              ...
  zk+1 ≈ ρk + (ρk - ρk-1)*0.96
              as initial value for the Newton algorithm on f(z)= ³z - (-1)
 Newton(zk+1) => ρk+1
```



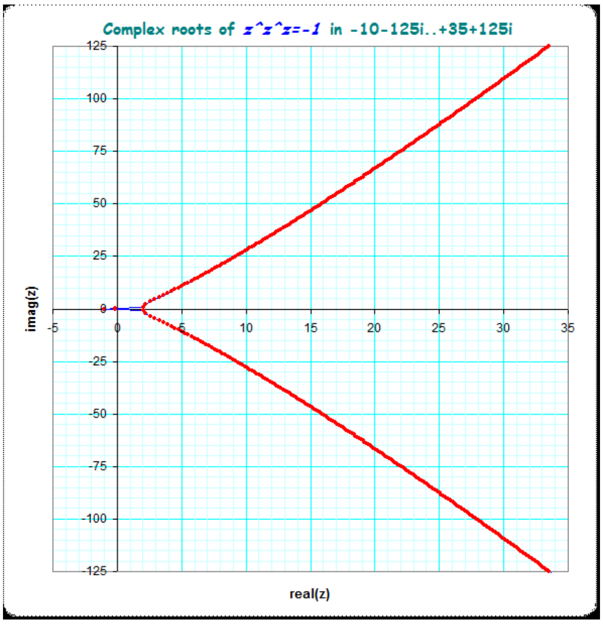
After this, proceeding with that idea one finds easily more roots



The first 21 (plus 20 conjugates) roots. The big points are that first three roots which I found initially using the New-ton-iteration on g(z).

Their colours indicate the initial regions from where the Newton-iteration discovers them (see next picture)

and even the suggestion of existence of infinitely many more ...



Here are the first 201 roots (plus 200 conjugates)

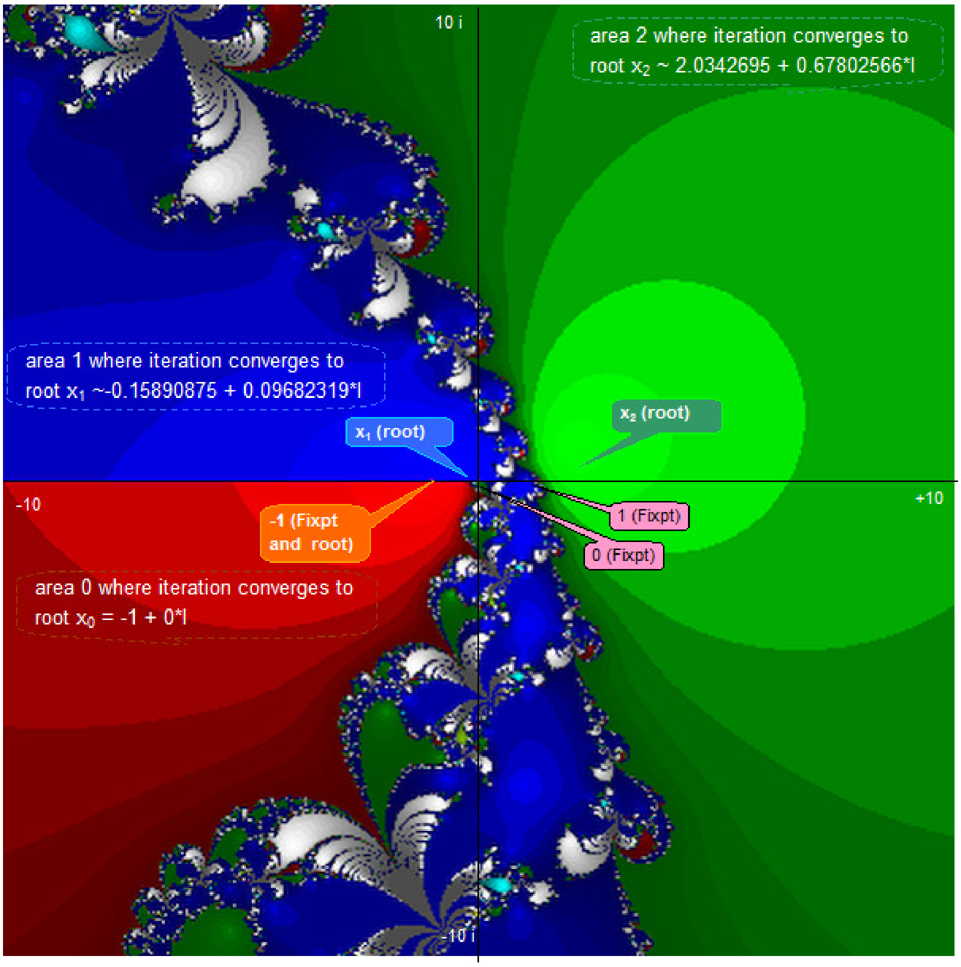
2 A Newton Fractal

The term "Newton-fractal" (https://de.wikipedia.org/wiki/Newtonfraktal) is a fairly common term which means just what I intuitively made here: it documents the orbit of some initial complex value z_0 when the Newton-iteration is applied to some function *f(z)* to find a root. Then the found root gets a certain color associated and this point of color is placed in a complex scatterplot at the coordinate of the initial value z_0. This gives then a visual impression of at which roots – starting from any z_0 – one arrives. There are pictures of impressive fractals available in the internet, so far as I have seen them only for *f(z)* being a polynomial, though.

Because I didn't know that this is also a well known and well studied process, and thus had to develop my algorithms myself, the results might not be the most accurate ones, but I think the following picture still gives a valueable impression of even a fractal structure.

2.1 Newton-fractal for *g(z)*

The first 3 roots known were ρ_0=-1, ρ_1~-0.1589 + 0.0968î, ρ_2~2.034+0.678î



This picture might be a bit misleading - while evaluating for *g(z)* it doesn't evaluate for the complex conjugates of the roots for *f(z)* which must also appear. I've not yet a properly updated version of this picture.

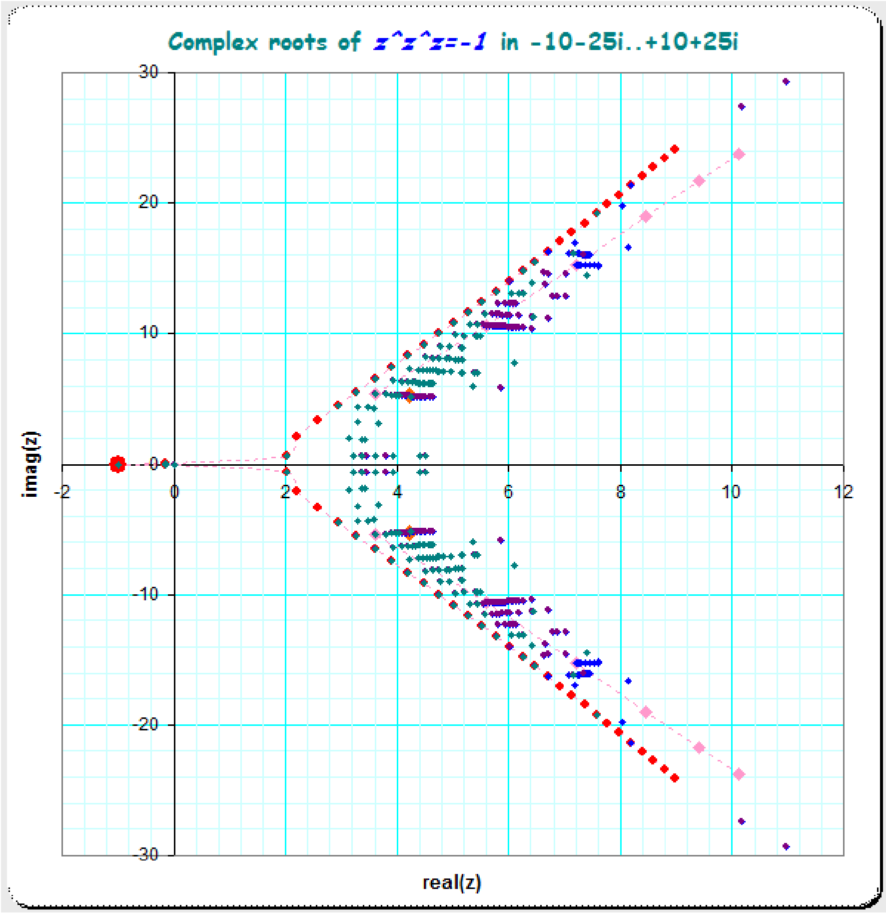
3 Roots *ρ* for *f()* (meaning *f(ρ)=0*)

After sneaking into the general situation via considering *g(z)* only, a second approach which applied the Newton-iteration for finding roots now actually for *f()* gave a set of additional roots *ρ*, such that *f(ρ)=0*. On first sight the localization of this set looked completely chaotic - besides the surprising observation that the new found roots occur only in the right-hand open parabola limited by the shape which has been found for roots of *g()* already.

Then searching with even finer intervals for the initial values z_0 for the Newton-iteration gave always more and more roots ρ_{z0}, and that increasing number of roots begin to show some new structure: see the dotted, but approximately linear lines, on which the roots seem to accumulate. This is not yet sufficiently studied though:

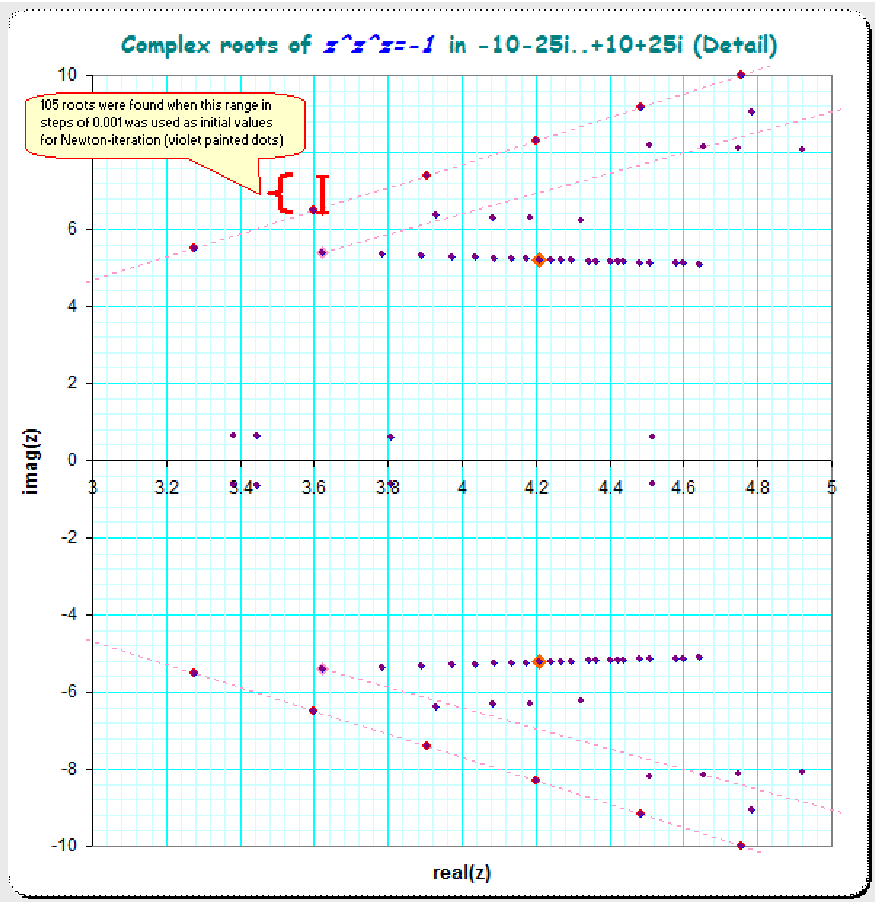
3.1 Complex roots *ρ* of *f()* such that *f(ρ)=0*

Remark: different colors are applied due to different searching episodes with slightly changed specific sets of initial values z_0 leading to according zeros ρ_{z0}. Red dots are the roots already known from $g(\rho_z)=0$



3.2 Detail

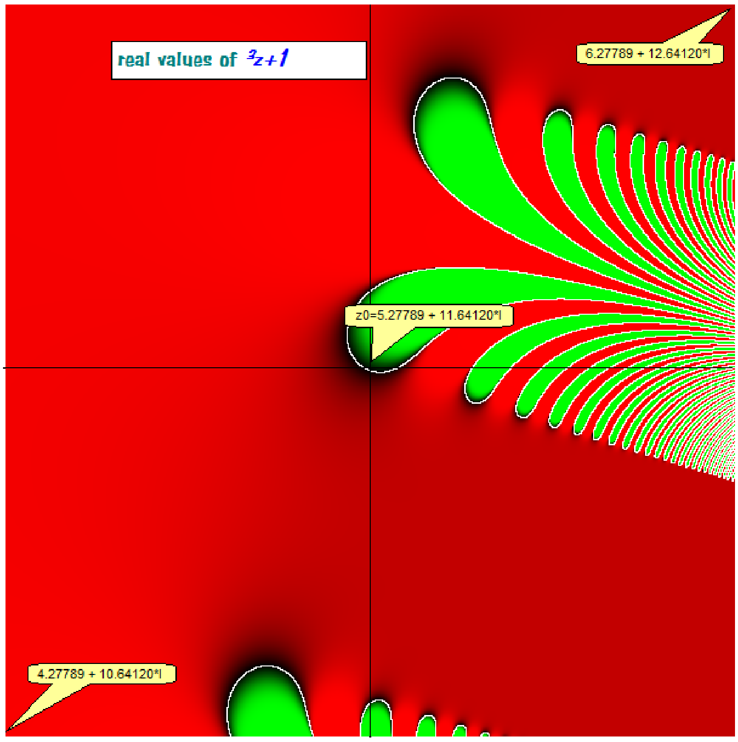
Detail which shows from what small interval of initial values z_0 for the Newton-iteration new roots ρ_{z0} can be found! Starting at some known root ~3.6+5.8î scanning vertically one imaginary unit upwards in steps of 0.001 finds 105 new roots!



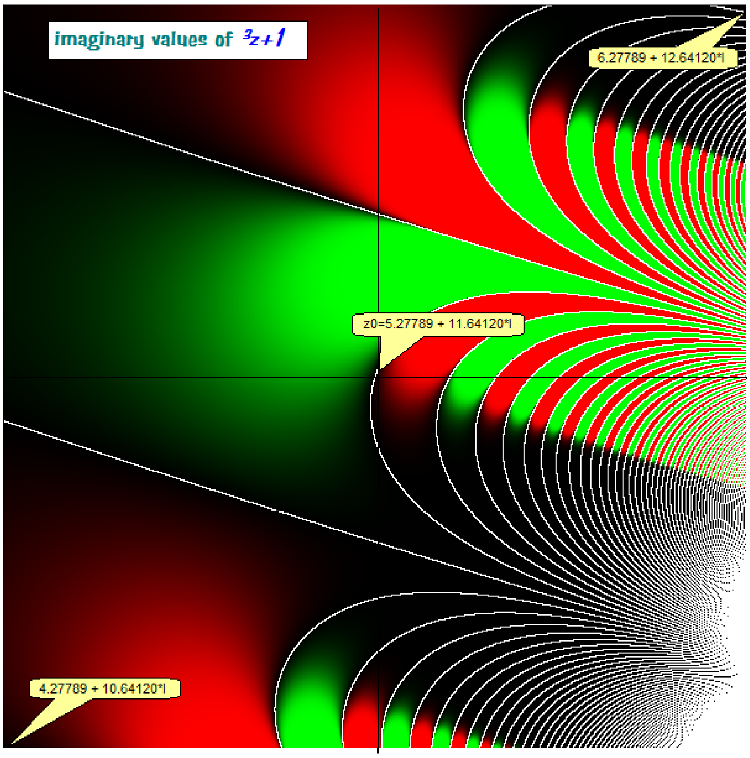
3.3 Contour lines of zeros in real and in imaginary-part (update 2020 feb (1))

The rough structure in the scattering of the roots for *f(z)* made me suspecting that the roots might lie somehow on ***dotted*** lines and looking at the imaginary and the real parts of *f(z)* separately the resp roots might be connected by ***continuous*** lines. This seems to be true; I show three pictures. This pictures show the neighbourhood of a known zero $z_0 \approx 5.277+11.641\hat{\imath}$ with ± one unit extension.

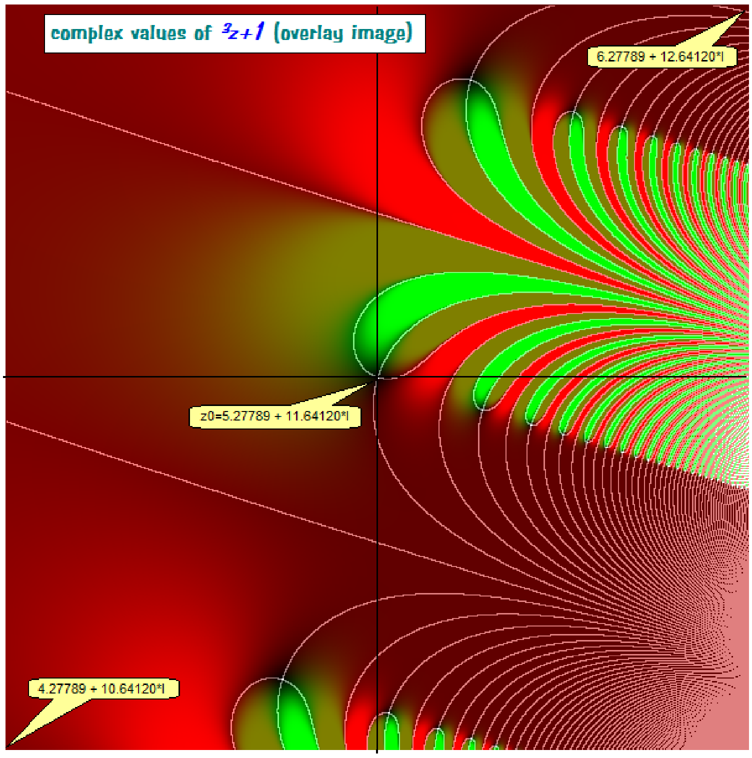
First picture *real(f(z))* : green color indicates negative values, red color positive values. Small absolute values dark, large absolute values light. Where neighboured dots have alternating sign I draw white points, indicating ***continuous lines*** of zero values:



second picture *imag(f(z))*. Again we find ***continuous lines*** with zero values:



Now the overlay shows ***discrete locations*** for the complex zeros of *f(z)* as met in the recent investigation; they are just on the points, where the white lines from the real and from the complex images intersect:



Interesting the further structure of the roots-locations: that slightly rotated rectangular shape towards the south-east of the picture.

But I think I'll go not further in this matter -with that findings someone interested and talented might dive deeper in this and give bigger pictures and/or more insight.

(data can be shared on request; the excel-sheet has the data in about six digits precision which are enough to reconstruct better approximations easily using them as initial values for the Newton-rootfinder, for instance with Pari/GP and arbitrary precision)

Gottfried Helms, Feb 2020 (last three pictures inserted Aug 2022)

4 Countably infinitely many roots by $2\cdot\mathbb{N} \times \mathbb{N}$ - each "having an address"

(It seems that) all roots can be adressed by a pair of numbers $B=[b_0, b_1]$ (where $b_0 \in 2\cdot\mathbb{N}$, $b_1 \in \mathbb{N}$) .

While there is no direct formula for $B \to \rho_B$, there is a convergent iterative procedure such that such address B allows ***to find*** (with a small need of computation) its according root $\rho_B = \rho_{b_0,b_1}$ `= rho_from_B( [b0, b1] )`.

4.1 Computation[1] of a root from the given address $B=[b_0, b_1]$

a) Consider the branched logarithm to some base z :

```
4.1a1   logz(x, branch=0, z)= (log(x) + branch · π·î)/log(z)
```

Next we define a two-step function of this, beginning at $x= -1$ as first argument, allowing ***two*** branch indexes - which we might write as $B=[b_0, b_1]$ for convenience:

```
4.1a2   LLz(B, z) = logz( logz(-1,b0,z), b1, z)          where b0 ∈ 2·ℕ, b1 ∈ ℕ
```

With this we have the following equality - with a given root ρ_B :

$$z = \rho_B$$

$$\text{4.1a3} \quad \rho_B = z = LL_z(B, z)$$

b) Moreover, given such fixed address $B=[b_0, b_1]$ this equality is ***attracting under iteration*** - so if we start with some z (only in the near of the expected root ρ_B) we can run the loop

```
4.1b    z=2+0.68*I  \\ initialize with some complex value from first quadrant
        approx(B,z,maxit=40) = for(it=1,maxit, z = (z + LLz(B,z))/2  ); return(z)
```

This iteration on z will converge towards the root ρ_B .

c) Having iterated to acceptably approximation to the root, we can improve that approximation by Newton-iteration based on the standard formula:

$$err = (z\hat{}z\hat{}z+1)/(z\hat{}z\hat{}z)'$$

$$z \leftarrow z - err$$

and the implementation

```
4.1c1   Newton(z,maxit=100) = for(it=1,maxit,err=(z^z^z+1)/(z^z^z)'; z=z - err ); return(z)
```

or better/more explicite, because the derivative-symbol in the denominator is likely not implemented:

```
4.1c2   Newton(z) = my(lz,zz,zzz,err);
                for(it=1,100,
                        lz  = log(   z );
                        zz  = exp(lz·z );
                        zzz = exp(lz·zz);
                        err = (zzz+1) / (zzz · zz · (lz^2+lz+1/z));
                     z = z - err ;
                   );
                return(z);
```

d) Finally the concatenation *rho_from_B()*:

```
4.1d    rho_from_B( B, z0=2+0.67·I)=my(z); z = approx(B, z0); z = Newton(z); return(z)
```

With that, by *rho_from_B(B)*, we get the B-indexed root ρ_B to arbitrary many digits precision.

[1] all program-snippets below are pseudocode near Pari/GP. For readability no errorchecks, stopcriteria etc. documented here.

4.2 Heuristics

Note, that the following two (three) roots are not covered by this:

$\rho^* = -1$

$\rho^{**} \approx -0.15890875158 + 0.0968231909176 \cdot I$ (and its conjugate)

however, here ρ^{**} can be computed with a **negative** real part in the initial value z_0:

*ρ** = rho_from_B([0,0], -0.1 + 0.1·I) ;*

but there are no further roots with addresses *rho_from_B([2·b₀, 0], -0.1 + 0.1·I)* available.

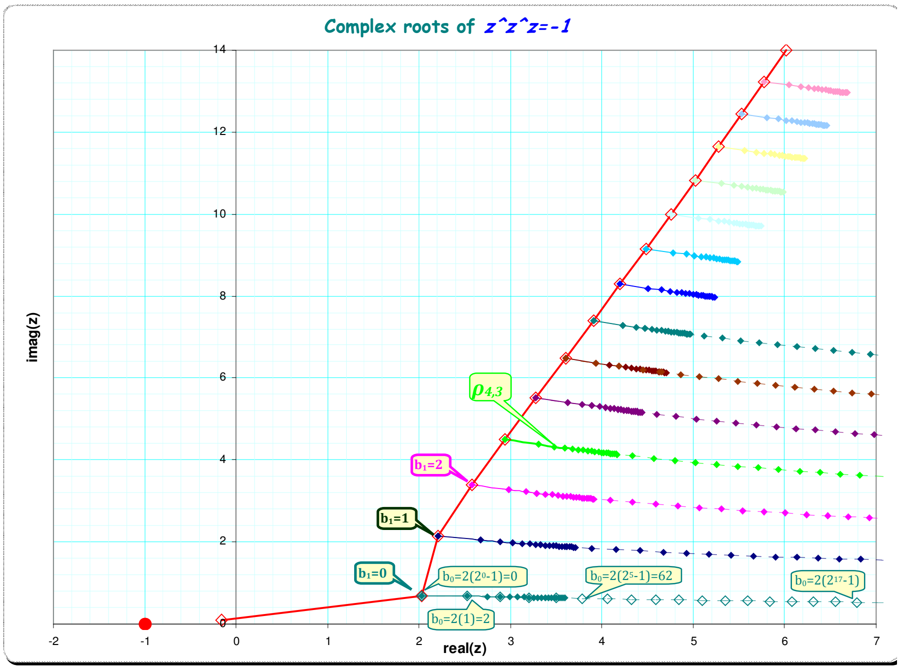
4.2a Table of complex roots ρ_{b_0,b_1} of $f(z) = z^{z^z} + 1$ (top left of the two-way infinite table):

-2 -1		ρ* = -1 ρ** ≈ -0.15890875158 + 0.0968231909176·I						
		col=0	col=1	col=2	col=3	col=4	col=5	
rov	b_1=	b_0=0	b_0=2	b_0=4	b_0=6	b_0=8	b_0=10	...
0	0	2.0342695+0.6780257*I	2.5241534+0.6738209*I	2.7446112+0.6662407*I	2.8876109+0.6605098*I	2.9933381+0.6560276*I	3.0771206+0.6523794*I	...
1	1	2.2102262+2.1432215*I	2.6782070+2.0401727*I	2.8897920+2.0001296*I	3.0271924+1.9758012*I	3.1288527+1.9585498*I	3.2094584+1.9452889*I	...
2	2	2.5744830+3.3921203*I	2.9798201+3.2710770*I	3.1678220+3.2201417*I	3.2913155+3.1882948*I	3.3833384+3.1653438*I	3.4566704+3.1475106*I	...
3	3	2.9359720+4.4930626*I	3.3051399+4.3750253*I	3.4772344+4.3234512*I	3.5906454+4.2906006*I	3.6753559+4.2666338*I	3.7429868+4.2478411*I	...
4	4	3.2773812+5.5107285*I	3.6235336+5.3980844*I	3.7850414+5.3479700*I	3.8915628+5.3157421*I	3.9711816+5.2920736*I	4.0347844+5.2734206*I	...
5	5	3.6001329+6.4734562*I	3.9300369+6.3659409*I	4.0839561+6.3176192*I	4.1854888+6.2863723*I	4.2613923+6.2633351*I	4.3220375+6.2451252*I	...
6	6	3.9071375+7.3961904*I	4.2247399+7.2931683*I	4.3728806+7.2465674*I	4.4705981+7.2163271*I	4.5436504+7.1939767*I	4.6020195+7.1762754*I	...
7	7	4.2009174+8.2879417*I	4.5087489+8.1888067*I	4.6522899+8.1437663*I	4.7469645+8.1144680*I	4.8177392+8.0927768*I	4.8742875+8.0755748*I	...
8	8	4.4834695+9.1546540*I	4.7832685+9.0588950*I	4.9230257+9.0152492*I	5.0151952+8.9868085*I	5.0840932+8.9657260*I	5.1391400+8.9489907*I	...
9	9	4.7563613+10.000505*I	5.0493849+9.9077026*I	5.1859508+9.8653019*I	5.2760065+9.8376359*I	5.3433205+9.8171085*I	5.3970997+9.8008019*I	...
10	10	5.0208361+10.828573*I	5.3080299+10.738383*I	5.4418508+10.697098*I	5.5300884+10.670132*I	5.5960397+10.650109*I	5.6487280+10.634194*I	...
11	11	5.2778940+11.641209*I	5.5599919+11.553347*I	5.6914145+11.513066*I	5.7780637+11.486734*I	5.8428244+11.467171*I	5.8945598+11.451614*I	...
...	...	...	...	...	...	...	...	...

Legend: the column $b_0=0$ (in red) contains the "primary" roots (except ρ^*, ρ^{**}). That values lay in the picture below on the (roughly) antidiagonal red line from bottom left to top right.

In each row of the table follow then the roots which are in the picture connected to the leading root roughly horizontally towards right (→ positive real infinity). So the value in row $row = b_1=3$, and column $col=2, b_0=4$, is $\rho_{4,3}=3.4772344+4.3234512 \cdot I$ and that root is in the picture the 3^{rd} point on the 4^{th} (=green) line (this point is identifyed by a legend).

4.2b Picture of roots of small indexes



Remark: the sequences of roots on the (nearly) horizontal lines densify exponentially with their index b_0. So the first 16 roots $\rho_{2k,0}$ are the first 16 points on the seagreen line. But to see more of the ***tendency*** of the sequence of that roots, a dotted line was underlayed and each 2^k-1'th root was marked; so we have the orbit of the first 2^{17}-1 roots $\rho_{2k,0}$ characterized by that (nearly) straight line.

4.3 Getting $B = [b_0,b_1]$ from a given root

Part of the computating apparatus should be the possibility of reversing the process: given a root *z* find its address *B* by a calculation.

The following procedure is possible:

```
B_from_z(z)=my(lz=log(z),b0,b1,wb0,wb1);
                wb1=z·lz;
      b1=imag(wb1) \ Pi ;  \\ integer division!
                wb0=lz·exp(z·lz - b1 · Pi·I);
      b0 = imag(wb0) \ (2·Pi); \\ integer division!
      b0 = 2·b0;
      return([b0,b1]);
```

Note, that by the software internal imprecision of the data errors can occur, especially when such an imprecise value *z* is integer-divided by imprecise value *Pi*. So it is recommended to always work with high precision and recalculate/recheck the values with an even higher precision, especially when the adress *B* has high indexes b_0 or b_1.

Finally, to have coordinates *(r,c)* into a matrix which might contain the roots, the address *B* must be recalculated and renormed to match the Pari/GP-convention of index-base *1* for rows & columns:

```
rc_from_z(z)=my(b0,b1,r,c);
      [b0,b1] = B_from_z(z);
      [r,c] = [b1+1, b0/2+1]; \\ the +1 to correct for Pari/GP matrix-indexes
      return([r,c]);
```

5 Generalization to finding roots of $f_3(z, q=-1) = {}^3z - q$ and $f_4(z,q=-1) = {}^4z - q$

The generalization of the finding of roots from $f_3(\ )$ to $f_4(\ )$ is rather straight forward. Even the possibility to search for roots in $^3z = -1$ to that in $^3z = +1$ needs only change on a simple parameter. This is then equivalent in the problems $^4z = -1$ and $^4z = +1$.

Of course, this calls for generalizations to $f_m(z,q) = {}^mz - q$, but I've not yet done the required elaboration & checks.

5.1 Introducing the form $^3x = +1$ or roots of $f_3(x, +1)$

We have seen in the introduction-paragraph in chap 3, that for the finding of roots of $f_3(x)$ $(= f_3(x,-1))$ in the adress $B=[b_0, b_1]$ the values for b_0 must be taken from $2\cdot\mathbb{N}$.

If we take the values from $2\cdot\mathbb{N} + 1$ we get immediately the roots for $f_3(x, +1) = x^{x^x}-1$. One adaption must be taken in the *Newton()*-approximation; in the numerator we must replace the "$+1$" - term by "$-q$" and apply this q as parameter for the function-call:

```
5.1a    Newton(z,q=-1) = my(lz,zz,zzz,err);
                for(it=1,100,
                        lz  = log(   z );
                        zz  = exp(lz·z );
                        zzz = exp(lz·zz);
                        err = (zzz - q) / (zzz · zz · (lz^2+lz+1/z));
                     z = z - err ;
                   );
                return(z);
```

Finally we must adapt the concatenation *rho_from_B()*:

```
5.1b    rho_from_B( B, q=-1, z0=2+0.67·I)=my(z); z = approx(B, z0); z = Newton(z,q); return(z)
```

5.2 Introducing the form $^4z = q$ or roots of $f_4(z, q)$

Finding 4^{th} superroots, or roots of $f_4(z, q)$, requires only a little schematic extension of the functions given in chap 4.1.

a) Consider again the branched logarithm to some base z :

```
5.2a1   logz(x, branch=0, z)= (log(x) + branch · π·î)/log(z)
```

Next we define a ***three***-step function of this, beginning at $x = -1$ as first argument, allowing ***three*** branch indexes - which we might write as $B=[b_0, b_1, b_2]$ for convenience:

```
5.2a2   LLLz(B, z) = logz( logz( logz(-1,b0,z), b1, z), b2, z)
                        where b0 ∈ 2·ℕ, b1 ∈ ℕ ,b2 ∈ 2·ℕ
```

b) As before, given such fixed address $B=[b_0, b_1, b_2]$ this equality is ***attracting under iteration*** - so if we start with some z (only in the near of the expected root ρ_B) we can run the loop

```
5.2b    z=2+0.68*I  \\ initialize with some complex value from first quadrant
        approx(B,z,maxit=40) = for(it=1,maxit, z = (z + LLLz(B,z))/2  ); return(z)
```

This iteration on z will converge towards the root ρ_B .

c) Having iterated to acceptably approximation to the root, we can improve that approximation by Newton-iteration based on the standard formula:

$$err = (z\text{^}z\text{^}z\text{^}z - q)/(z\text{^}z\text{^}z\text{^}z)'$$
$$z \leftarrow z - err$$

and the implementation with explicite derivative in the denominator:

```
5.2c    Newton(z,q=-1) = my(lz,zz,zzz,err,nu,de);
            for(it=1,100,
                lz   = log(   z  );
                zz   = exp(lz·z  );
                zzz  = exp(lz·zz );
                zzzz = exp(lz·zzz);
                nu = zzzz -q ;
                    de =  zz·lz^2·(lz + 1) + (zz·lz + 1)/z;
                de = zzzz·zzz·de;
                err = nu  / de ;
              z = z - err ;
             );
            return(z);
```

d) Finally the concatenation *rho_from_B()*:

```
5.2d    rho_from_B( B, z0=2+0.67·I)=my(z,q);
            z = approx(B, z0);
              if(B[1] % 2, q=1, q=-1 );
            z = Newton(z,q);
            return(z)
```

With that, by *rho_from_B(B)*, we get the *B*-indexed root ρ_B to arbitrary many digits precision, and depending on index b_0 the root of the ^4z*-1* or ^4z*+1* - versions:

- if we want the roots of $^4z = -1$ we need the adress by $B \in [2 \cdot \mathbb{N}, \mathbb{N}, 2 \cdot \mathbb{N}]$

- if we want the roots of $^4z = +1$ we need the adress by $B \in [2 \cdot \mathbb{N} + 1, \mathbb{N}, 2 \cdot \mathbb{N}]$.

Gottfried Helms, 28.8.2022