Problem with the ARRL "Gamma" Code for Some Parameters
N6MW 5/10/2010

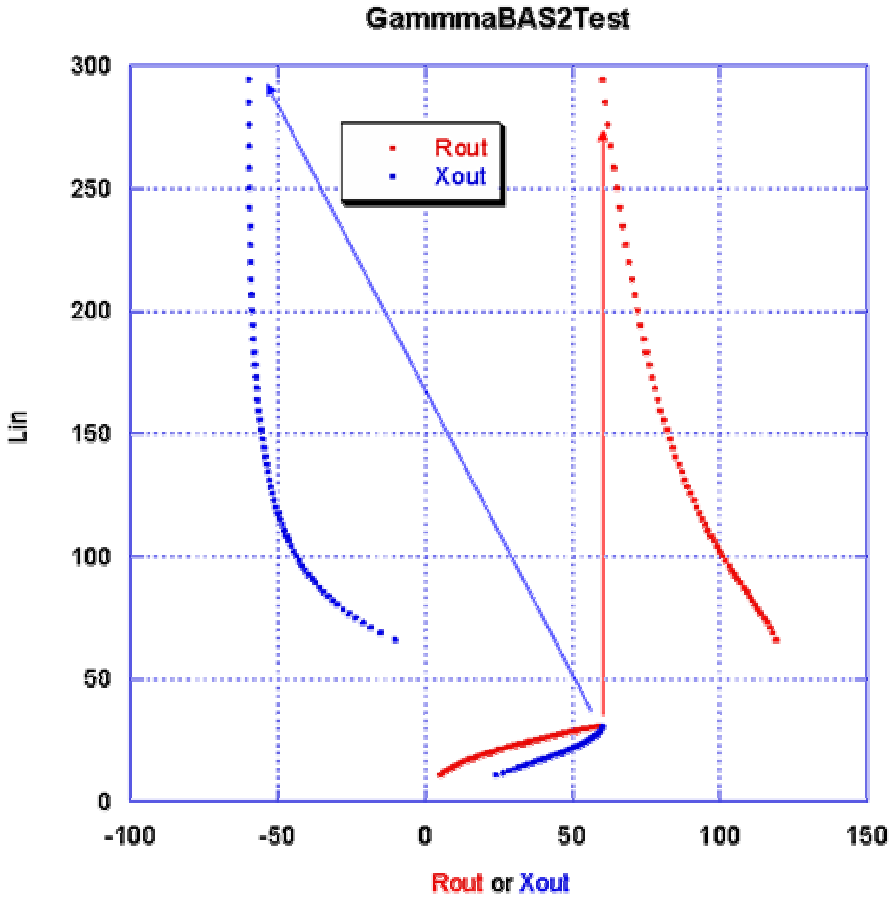
The Problem

There appears to be a legitimate parameter range of solutions to the gamma match transform equations that cannot be found using the ARRL code as it stands. This is shown for one example but the effect appears to be fairly general.

This example is for Zant = 15-j15, D=1, d=1, S=10 and f=10 MHz to provide round numbers, including a step up (SU) value of exactly 4. The first results shown are from "Gamma" as supplied with the ARRL Antenna Book (but with an added loop over Rout, the feedline impedance target, to speed the data collection process). The results were spot checked against the original ARRL-supplied executable over the full range for verification.

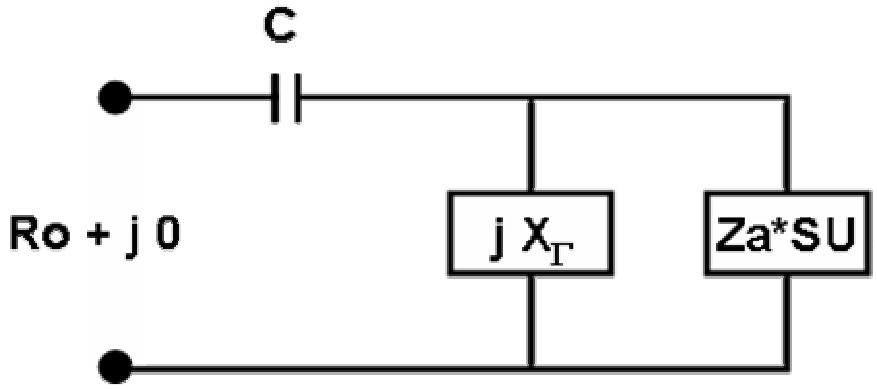
The code input Rout values from 5 to 120 ohms but with a +0.1 ohm offset to avoid using exactly 60 – the code produces an error at Rout of exactly 60. More on this later.

Results for Rout and Xout for the gamma rod length L (in inches) are plotted below.

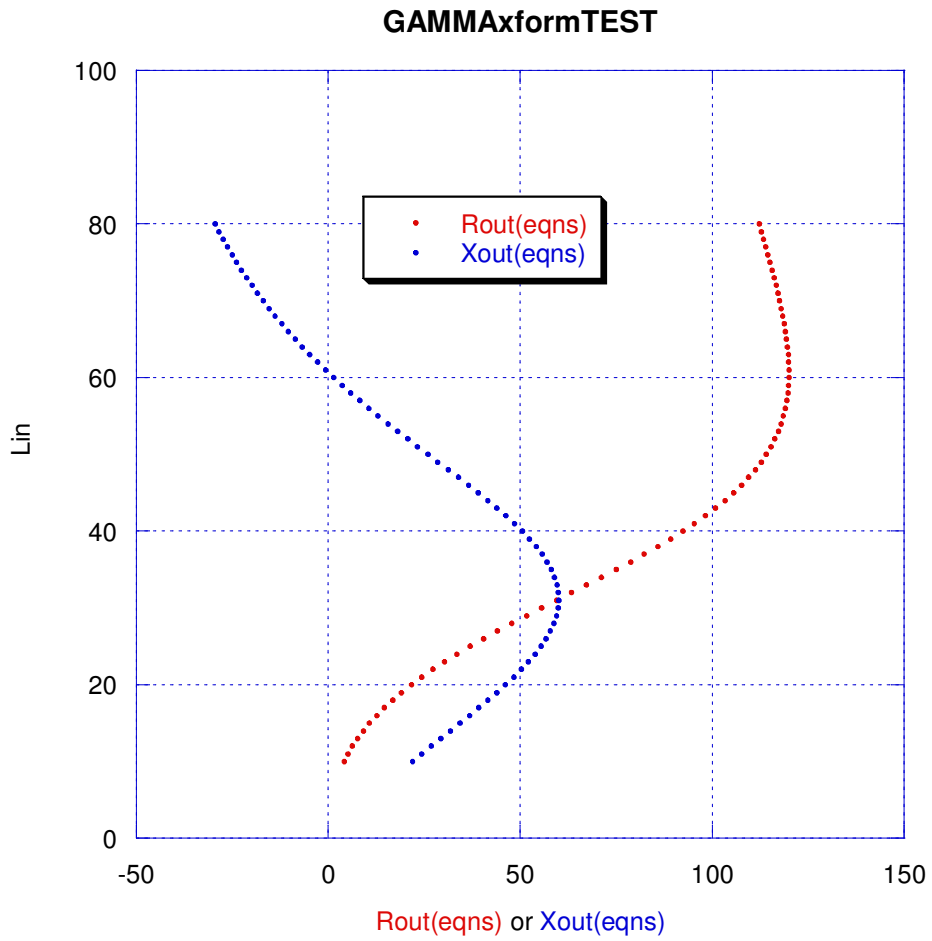


Note that for Rout crossing 60 there is a discontinuity (arrows) in the solutions for both L and Xout (the reactance of a series capacitor needed to cancel the gamma network reactance). Here Xout is negative meaning an inductive reactance is needed – not the standard desired outcome but still physical. For Rout=120 and greater, the code generates an error (illegal function call). At first blush, you might think that what is trying to say is for Rout>=60 there are no solutions. I believe the results generated for 120>Rout>60 are actually incorrect and not solutions to the original equations, but yet there do exist solutions. And here's why.

The full complex gamma transform equations from the circuit



do not readily lend themselves to solutions for L, the gamma rod length and C, given Ro=Rout. However, it is easy to plot up the values of Rout and Xout =1/(2π*f*C) for a range of L given d, D, f, and S. Obviously iteration methods could be used to solve for Rout(L) if desired. Here is the result of looping over a set of L values (10 to 80) to find Rout and Xout for the same d, D, f, and S used before.



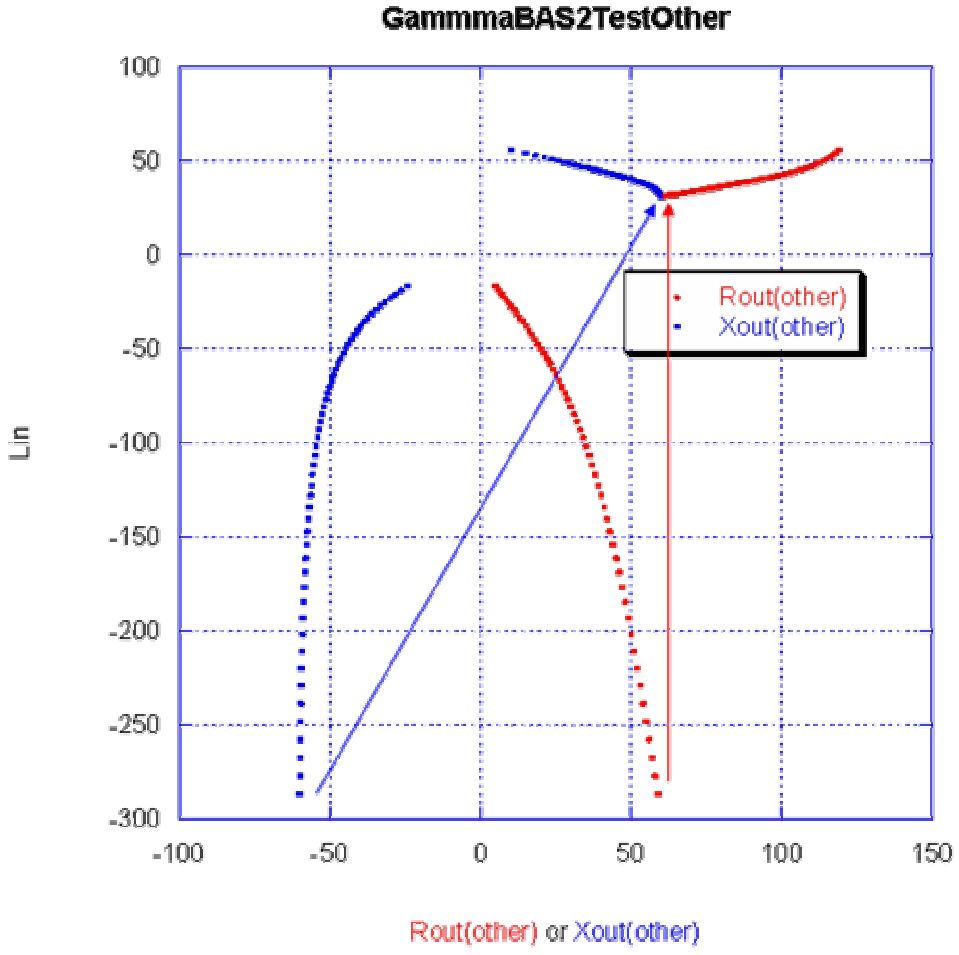
Note that there are no discontinuities, the results for Rout beyond 60 are smooth and with Xout values that are positive. Xout peaks at Rout=60. The maximum value for Rout is 120 ohms where Xout is zero. If L is made larger, Rout becomes < 120 and the Xout <0. - there may be special situations where this portion of the parameter space could be useful. But the key matter is that there exist solutions for 60<Rout<120 which are not found with the "Gamma" code.

As a side note, the ON4UN code (in the "double your Za" vertical mode only) reproduces (within numerical accuracy) the full equation results both for Rout<60 and for 60<Rout<120 for the Xout>0 cases. For Rout >120 the ON4UN code says it cannot find a solution but it is happy to do the Rout exactly 60 case.

So what is the issue with the "Gamma" code? An analysis of the actual BASIC code for "Gamma" indicates that the quadratic equation being solved has been set up for any unknown that is proportional to a factor of

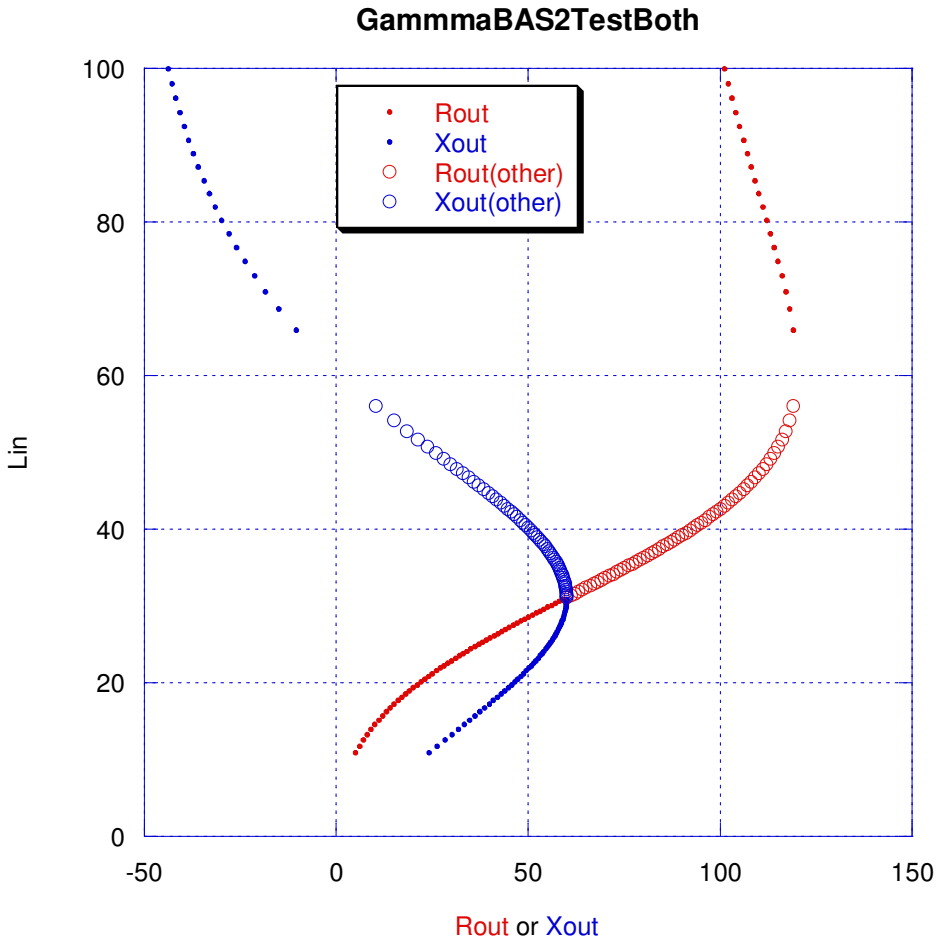
1-Rant*SU/Rout .

The solution found is then used to compute Rout using this relation. However for the current example SU=4 and Rant=15 so for Rout=60, the above factor is zero and it appears that leads to a divide by zero error only when Rout is exactly 60. Obviously for Rout>60 the factor changes sign. This appears to result in the need for the choice of the +/- root of the quadratic solution to change, which it does not. If the "Gamma" source code is changed to select the other root (i.e., Q = A + SQR(A * A + B) becomes Q = A - SQR(A * A + B)) the results for running the modified code over a range of Rout generates the following.



Again there is the discontinuity (arrows) at Rout=60 and the "solutions" for Rout<60, showing negative L and negative Xout, are not physical. However for Rout>60 the solutions are exactly those found when the full equations were used. To illustrate this, the last plot here is taken from "Gamma" for 0<Rout <60 and from "Gamma (other root)"

for 60<Rout<120 and put the same scale as used for the GAMMAxformTEST plot (the 2nd plot earlier) that uses the complex transform equations.



The lack of points near L=60 has no deep meaning - it was dictated by the choice of the increment of one between Rout values.

The full complex equations have the feature that the transformed impedance Zout (with no C added) has an imaginary part proportional to (1/Xg +Xant/SU/|Zant|^2), where Xg is the inductive impedance of the shunt gamma section. For negative Xant, as done in the current example, increasing L increases Xg so at some point Xout goes to zero. At this point the value of Rout becomes SU*|Zant|^2/Rant which for the current example (Xant = - Rant) means Rout goes to SU*(2Rant^2/Rant) which gives the 120 ohm maximum value seen since SU=4 and Rant=15 ohms. Other examples will change the details of the calculations but the fundamental issue will remain.

Resolution?

The ARRL "Gamma" code could be stopped from providing incorrect solutions by adding code that restricts Rout to be < Rant*SU – however this will still not allow the

user to find those fully proper solutions for larger Rout. It may also be possible to provide another branch in the code that is selected if Rant*SU <Rout < SU*|Zantl^2/Rant where the other root of the quadratic equation is used. In this case something would still need to be done to avoid the Rout=Rant*SU singularity.

This singularity appears to have been avoided in the ON4UN code, probably by setting up the quadratic equation to have a different unknown, one that does not vanish in this artificial manner over a useful part of the parameter range.