short root finding method

This document describes the method used to solve nonlinear ODEs in MOONS. This was used to match the Robert's stretching parameter in the non-uniform grid. The equation to solve was

$$\underbrace{\frac{(\beta+2\alpha)\gamma_N-\beta+2\alpha}{(2\alpha+1)\{1+\gamma_N\}}}_{dimensionless}-\underbrace{\left\{\frac{(\beta+2\alpha)\gamma_{N-1}-\beta+2\alpha}{(2\alpha+1)\{1+\gamma_{N-1}\}}\right\}}_{dimensionless}=\frac{\Delta h}{(h_{max}-h_{min})}$$

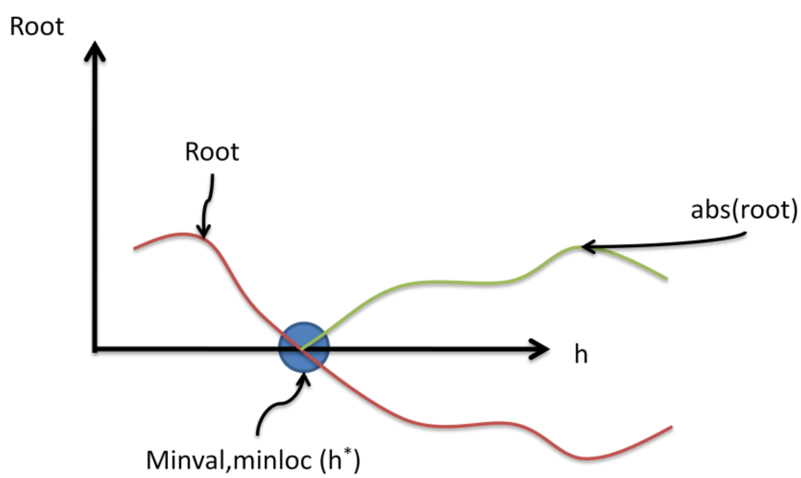
Where β is the unknown. Clearly, this equation is non-linear and must be solved numerically. The method I used is a sort of binary search method. For ease of notation, I will write this solution as would be done in matlab, it is very easily transferrable to Fortran with the minloc() function.

Problem statement

We may always write our equation by moving everything to one side:

$$f(h)=root(h)=0$$

The solution method can be pictorially described as follows:



Solution Method

This problem can be easily solved, in four lines:

$$h=linspace(h_{min},h_{max},N);$$

$$root(h)=\cdots$$

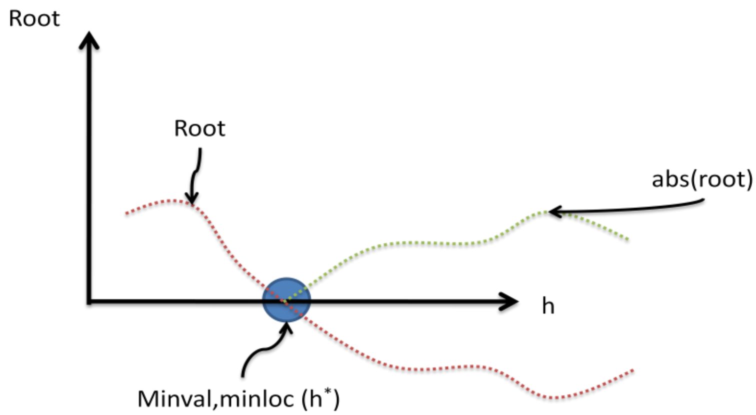
$$[minVal\ minLoc]=\min\Big(abs\big(root(h)\big)\Big);$$

$$h^*=h(minLoc);$$

Note that the accuracy depends on the range and number of points used in the array: h_{min}, h_{max}, N. Therefore, the resolution is

$$dh = \frac{h_{max} - h_{min}}{N}$$

Which means, pictorially,



This means that we may repeat this process with new boundaries that are at least $2\Delta h$ away from our initial solution.

Central search

$$h_{min}^{n+1} = h^* - 2\Delta h$$

$$h_{max}^{n+1} = h^* + 2\Delta h$$

Search Right

Note that if the solution is on the boundary, we would like to expand our boundaries in the direction of the minimum. If near the max we have

$$h_{min}^{n+1} = h^* - 2\Delta h$$

$$h_{max}^{n+1} = h^* + 2\Delta h$$

This doesn't allow for very efficient boundary searches, so we may alternatively expand our search on the same scale as the initial range:

$$h_{min}^{n+1} = h^* - 2\Delta h$$

$$h_{max}^{n+1} = h^* + 2N\Delta h$$

Search Left

Similarly, if near the min we have

$$h_{min}^{n+1} = h^* - 2\Delta h$$

$$h_{max}^{n+1} = h^* + 2N\Delta h$$

Steep asymptotic search Left (for $\beta \to 1$)

If we are searching near a *known* asymptote, where the root becomes nearly vertical, we may want to search a bit more carefully:

$$h_{min}^{n+1} = A - (A - h^*) \times (1 - \gamma)$$

$$h_{max}^{n+1} = h^* + 2\Delta h$$

Where A is the location of the asymptote and as $\gamma \to 1$, $h_{min}^{n+1} \to A$ and as $\gamma \to 0$, $h_{min}^{n+1} \to h_{min}^{n}$. I believe a value of $\gamma = 0.8$ should be a good conservative choice in order to slowly ascend/descend to the asymptote (in order to avoid NANs by truncation error).

Steep asymptotic search Right (for $\beta \to 1$)

Similarly, we have

$$h_{min}^{n+1} = h^* - 2\Delta h$$

$$h_{max}^{n+1} = A + (A - h^*) \times (1 - \gamma)$$

Iterations

This process can be repeated recursively in order to reach a very high precision. It has been my experience that, for a central search, this method is very effective at reaching high resolution in very short time.