Automatic Generation of Root Locus Plots

Tomás Oliveira e Silva

***Abstract* – In this paper we present an algorithm capable of drawing automatically accurate root locus plots. It computes all the values of K for which at least one branch of the root locus changes direction abruptly (in that case the characteristic equation has multiple roots), subdivides each branch into segments according to these values of K, and draws separately each segment. Based of the first derivative of K as a function of s, appropriate values of K are computed in order to sample each branch of the root locus at approximately equidistant points. The output of our algorithm compares favorably with the output of `MATLAB`'s `rlocus` command.**

***Resumo* – Neste artigo é apresentado um algoritmo capaz de desenhar automaticamente o diagrama do lugar das raízes de um dado sistema.**

I. INTRODUCTION

The root locus is a classic technique used to study the location of the poles of the closed loop transfer function of a given linear system as a function of one of its parameters, usually a loop gain, given its open loop transfer function. It is commonly formulated for the simple negative feedback loop of Fig. 1, which has an equivalent transfer function of

$$F(s) = \frac{KG(s)}{1 + KG(s)H(s)}.$$

The poles of $F(s)$, which depend on K, are the zeros of the equation

$$1 + KG(s)H(s) = 0,$$

which is known as the characteristic equation of the system. We will assume that both $G(s)$ and $H(s)$ are proper rational transfer functions. In that case we have

$$G(s)H(s) = \frac{N(s)}{D(s)} = \frac{s^n + b_1 s^{n-1} + \cdots + b_n}{s^m + a_1 s^{m-1} + \cdots + a_m},$$

with $m \geq n$. To write this equation we have assumed, without loss of generality, that both $N(s)$ and $D(s)$ are monic polynomials; also, it is implicitly assumed that $N(s)$ and $D(s)$ do not have zeros in common. Using these polynomials in the characteristic equation yields

$$D(s) + KN(s) = 0. \tag{1}$$

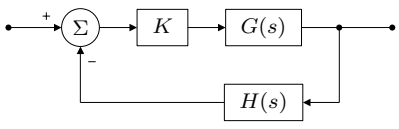

Fig. 1 - System configuration usually used in root locus plots.

As K goes from 0 to $+\infty$, or from 0 to $-\infty$, each one of the roots of this polynomial equation traces a curve in the extended complex plane. The collection of these curves (called branches) is known as the root locus of the system.

A large set of rules to sketch, by hand, the root locus of a given system can be found in most introductory textbooks of control theory. Although useful, these rules fall short of what is required to draw an accurate root locus automatically. In section II we present some extra useful rules, not usually found in textbooks, that help in this task. These rules form the backbone of our algorithm to draw the root locus, which is described in section III. We conclude the paper with section IV, where some examples are presented.

II. SOME USEFUL EQUATIONS

To draw automatically a root locus it is convenient to subdivide the range of values of K in intervals, in such a way that in each interval the branches of the root locus are smooth functions without derivative discontinuities. Since the branches begin ($K = 0$) at the zeros of $D(s)$, and end ($K = \pm\infty$) at either the zeros of $N(s)$ or at ∞, it suffices to find all finite values of K for which there are discontinuities in the derivatives of the branches (as functions of K).

It is possible to write the characteristic equation in the form

$$D(s) + K(s)N(s) = 0 \tag{2}$$

where now K is a function of s. With this new formulation a given point p of the complex plane belongs to the root locus if and only if $K(p) = -D(p)/N(p)$ is a real number. Note that instead of working with s as a function of K, say $s = s(K)$, we are working with the inverse function, i.e., with $K = K(s)$. Since

$$\frac{ds(K)}{dK}\,\frac{dK(s)}{ds} = 1,$$

and since $K(s)$ is a meromorphic function, $ds(K)/dK$ does not exist only when $dK(s)/ds = 0$. Thus, the exceptional values of K must satisfy the condition $K'(s) = 0$, which is equivalent to the condition

$$D'(s)N(s) - D(s)N'(s) = 0. \tag{3}$$

This last condition is usually found in the textbooks as the condition to determine the breakaway and breakin points of the root locus with respect to the real axis. It must be emphasized here that the same condition may hold outside of the real axis. What is important is that the values of K corresponding to the solutions of (3) be real (and of the proper sign).

It turns out that the solutions of (3) also give rise to multiple roots of the characteristic equation. To see this let

$$\Delta(s) = D(s) + KN(s)$$

be the characteristic polynomial, and let p be such that $D'(p)N(p) - D(p)N'(p) = 0$ with $N(p) \neq 0$. Setting $K = -D(p)/N(p)$ forces p to be a root of $\Delta(s)$. Furthermore, the first condition makes p also a root of $\Delta'(s)$. Thus p is at least a double root of $\Delta(s)$. In the next paragraph we will determine exactly the multiplicity of this root. Nonetheless, it is worthwhile to point out in advance that if p is a solution of (3) with multiplicity $L-1$ then setting $K = -D(p)/N(p)$ makes p a root of $\Delta(s)$ with multiplicity L.

Let p be a point of the root locus. Thus p is a root of $\Delta(s)$. We wish to find the multiplicity of this root. First of all, notice that

$$\Delta^{(l)}(s) = \frac{d^l \Delta(s)}{ds^l} = D^{(l)}(s) + KN^{(l)}(s).$$

(In this equation K is *not* a function of s.) Next, notice that from (2) we have

$$\begin{aligned} 0 &= \big(D(s) + K(s)N(s)\big)^{(l)} \\ &= D^{(l)}(s) + \sum_{i=0}^{l} C_i^l K^{(i)}(s) N^{(l-i)}(s), \end{aligned}$$

where $C_i^l = \frac{l!}{i!(l-i)!}$. (Now K is a function of s.) This equation may be put in the form

$$\begin{aligned} D^{(l)}(s) + K(s)N^{(l)}(s) + K^{(l)}(s)N(s) = \\ - \sum_{i=1}^{l-1} C_i^l K^{(i)}(s) N^{(l-i)}(s). \end{aligned} \quad (4)$$

Assume for the moment that the right hand side of this equation is zero when $s = p$, which will certainly happen if $l = 1$. In this case

$$K^{(l)}(p) = -\frac{D^{(l)}(p) + K(p)N^{(l)}(p)}{N(p)}, \quad (5)$$

which implies that if $K^{(l)}(p) = 0$ then $\Delta^{(l)}(p) = 0$. Now suppose that $K^{(l)}(p) = 0$ for $l = 1, \ldots, L-1$, and that $K^{(L)}(s) \neq 0$; if $L = 1$ then no derivative of $K(s)$ need to be zero at $s = p$. Under these conditions it is clear that $\Delta^{(l)}(s)$ will vanish for $l = 1, \ldots, L-1$, and will not vanish for $l = L$. Thus, p is a root of $\Delta(s)$ with multiplicity L. Note that the derivatives of $K^{(l)}(s)$ can be computed for $l = 1, \ldots, L$ with (5), because the right hand side of (4) will always vanish.

Formula (5) is valid if all lower order derivatives of $K(s)$ vanish at $s = p$. As in the previous paragraph, let L be the first derivative of $K(s)$ that does not vanish at $s = p$. Clearly, $L \geq 1$. In this case p is as root of $\Delta(s)$ with multiplicity L. The Taylor series expansion of $K(s)$ in the neighborhood of $s = p$ has then the form

$$K(p+h) = K(p) + \frac{h^L}{L!} K^{(L)}(p) + \cdots$$

where $\cdots$ denotes the terms in h^{L+1}, h^{L+2}, and so on, of the Taylor series. This power series has a strictly positive radius of convergence if $N(p) \neq 0$, i.e., if $K \neq \infty$, because $K(s)$ is a meromorphic function. Thus, for small enough h it is possible to make the approximation

$$K(p+h) - K(p) \approx \frac{h^L}{L!} K^{(L)}(p). \quad (6)$$

This leads immediately to the condition

$$\underline{/K(p+h) - K(p)} \approx L\underline{/h} + \underline{/K^{(L)}(p)}. \quad (7)$$

For $p+h$ to belong to the root locus $K(p+h)$ must be a real number. In that case (7), with $\approx$ replaced by $=$, specifies the angles of departure (and of arrival) of the L branches of the root locus at $s = p$. For example, if K is increasing (root locus for $K > 0$) then $\underline{/K(p+h) - K(p)} = 2k\pi$ for the branch, or branches, departing from the point $s = p$, and $\underline{/K(p+h) - K(p)} = \pi + 2k\pi$ for those arriving at that point.[1] Also, it is clear that the angles between adjacent departing (or arriving) branches are all equal to $2\pi/L$, and that the angles between adjacent departing and arriving branches are all equal to π/L. This fact is often stated without proof in the textbooks.

As an aside, note that (7) is valid when $K(p) = 0$, i.e., it can be used to compute the angles of departure from the poles of the branches of the root locus. In order to make the connection of this formula with the usual rule used for this purpose, suppose that p is a zero of $D(s)$ with multiplicity L, i.e., $D(s) = (s-p)^L D_1(s)$ with $D_1(p) \neq 0$. In this case it is an elementary exercise to verify that $D^{(L)}(p) = L!\, D_1(p)$. Since $K = 0$ it follows from (5) that $\underline{/K^{(L)}(p)} = \underline{/-D_1(p)/N(p)}$. If

$$D(s) = \prod_{k=1}^{m} (s - p_k) \quad \text{and} \quad N(s) = \prod_{k=1}^{n} (s - z_k)$$

we have

$$\underline{/D_1(p)} = \sum_{\substack{k=1 \\ p_k \neq p}}^{m} \underline{/p - p_k} \quad \text{and} \quad \underline{/N(p)} = \sum_{k=1}^{n} \underline{/p - z_k},$$

and so (7) becomes the formula found in most textbooks for the angles of departure of the branches. (Most textbooks treat only the case $L = 1 \ldots$)

If $m = n$ there exists one extra exceptional value of K, not usually mentioned in the textbooks. This exceptional value of K is of course $K = -1$, for which $\Delta(s)$ becomes a polynomial of degree smaller that m. In this special case some, or all, of the branches of the root locus go to infinity. The number of branches that go to infinity is equal to the degree reduction of $\Delta(s)$, which is equal to m minus the degree of $D(s) - N(s)$. One way to show this consists in rewriting the characteristic equation in the form

$$\underbrace{N(s)}_{\mathsf{D}(s)} + \underbrace{\frac{1}{1+K}}_{\mathsf{K}} \underbrace{\big(D(s) - N(s)\big)}_{\mathsf{N}(s)} = 0, \qquad K \neq -1.$$

This new equation can be interpreted as the characteristic equation of another system [cf. (1)]. When K tends to -1

[1] Note that k can be any integer; it suffices to use $k = 0, \ldots, L-1$.

from above (or below), K tends to $+\infty$ (or $-\infty$). The behavior of the branches of the root locus near $K = -1$ will then be the same as the asymptotic behavior of the branches of the root locus of $\mathsf{D}(s) + \mathsf{KN}(s) = 0$ when K tends to $\pm\infty$. (An elementary analysis of this well known case is presented in the appendix.) Note that the two root locus plots are graphically the same when drawn for K, and K, ranging from $-\infty$ to $+\infty$, although the calibration of the two plots will certainly be different. Finally, note that `MATLAB` does not handle this exceptional case, with ugly consequences ...

III. THE ALGORITHM

The input data of the algorithm is:

- a monic polynomial $D(s)$;
- a monic polynomial $N(s)$;
- a boolean flag, indicating if positive (true) or negative (false) values of K are desired;
- a bounding box, specifying the region of the root locus we are interested in;
- information of where to place one or more arrows in each segment (the default is one arrow in the middle of each segment).

Some optional flags may also be given, such as, for example, a flag indicating if the asymptote is to be drawn for each branch going to, or coming from, the point at infinity. The bounding box information is used to compute the desired distance δ between consecutive points of each segment of the root locus. By default $\delta = \max(\mathrm{width}, \mathrm{height})/500$, where width and height are the width and height of the bounding box.

The output data is an encapsulated `PostScript` file, holding the commands necessary to draw the root locus with an aspect ratio of 1.

The algorithm first computes the roots of $D(s)$ and of $N(s)$, if these polynomials where not given in factored form. All solutions of (3) are then computed. Since multiple roots of $D(s)$ or of $N(s)$ are also solutions of this equation, these roots are first removed, by deflation, from $D'(s)N(s) - D(s)N'(s)$ before searching for other solutions. This avoids, in most cases, the factorization of a polynomial that has multiple roots.[2] For each solution found, the corresponding value of K is computed, and those that are real and of the proper sign are sorted (repetitions are removed) and stored for posterior use. $K = 0$, $K = \pm\infty$, and, if necessary, $K = -1$, are also added to this list. For each of these values of K all (finite) roots of $\Delta(s)$ are computed. Of course, the known roots of $\Delta(s)$ are removed first, mainly to avoid factoring a polynomial with multiple roots. These known roots are either roots of $D(s)$, for $K = 0$, of $N(s)$, for $K = \infty$, or are solutions of (3), for all other values of K except possibly $K = -1$.

The rest of the algorithm is concerned with the generation of each segment of the root locus. First K is set to 0 and the corresponding roots of $\Delta(s)$ are used as the initial points of each of the m segments (one for each branch). Next, points are appended to each segment, in the way described below, until at least one segment reaches a value of K stored in the list of exceptional values. Those segments are then closed and sent to the output routine, and new ones are opened in their place unless the last exceptional value of K has been reached, in which case the algorithm terminates.

The proximity of an exceptional value of K is checked by comparing the respective values of K, and, in the case of branches not going to infinity, by also comparing the roots of the characteristic equation.

Equations (5) and (6) are used to increment the value of K. This is done by finding the smallest value of

$$\left|K(p+h) - K(p)\right| \approx \frac{\delta^L}{L!}\left|\frac{D^{(L)}(p) + K(p)N^{(L)}(p)}{N(p)}\right|$$

among the roots of $\Delta(s)$, for $K = K(p)$, that are *inside* the bounding box.[3] If one of the roots crosses the bounding box boundary, then a bisection strategy is used to compute a good approximation to the exact value of K for which that happens. The roots of $\Delta(s)$ for the new value of K are then computed, using the previous ones as starting values.[4] A simple proximity test is then used to assign each root to a segment. Arrival (and departure) angles are computed for each point.

The current point of each segment may be in one of three states: inside the bounding box, on the boundary, or outside the bounding box. To improve the quality of the generated `PostScript` code, and to decrease its size, each visible (inside or on the boundary) part of the segment is re-parametrized. This is done by selectively throwing away points that are closer than 4δ. (This makes the average distance between points to be between 4δ and 5δ.) A cubic spline (`PostScript`'s `curveto` command) is then drawn between every two consecutive visible points, using the departing and arrival angles to compute the two extra points required to specify uniquely the spline. It is also at this stage that arrows are drawn. Their position is specified as a fraction of the total length of each visible part of a segment. Using again a bisection strategy, the value of K where the arrow will be placed is computed. With this information, its position and orientation are also computed, and the arrow is drawn.

After all segments are drawn, it only remains to draw some decorative stuff, such as the real and imaginary axis, the asymptotes (if requested), and the poles (crosses) and zeros (circles).

Future implementations of the algorithm may provide the following extra facilities:

- automatic generation of a reasonable default bounding box;
- inclusion of calibration data (values of K) on interesting points of the plot.

[2] Multiple roots give some trouble to many root finding algorithms.

[3] If all roots are outside of the bounding box, an increment of K based of the smallest distance between the roots and the bounding box is used.

[4] Note that at this stage it is never necessary to deal with multiple roots. Using the previous roots as starting approximations of the roots of the new characteristic polynomial is a very effective way of reducing the execution time of the algorithm. More sophisticated, and accurate, guesses of the positions of these roots can also be easily computed, using (6) to estimate h.

IV. EXAMPLES

Figures 2 to 4 present examples of some root locus plotted with our algorithm. Contrary to MATLAB, all points where two or more branches touch are treated correctly. Figure 2 shows a "simple" root locus plot, used by the author to illustrate, in his control theory classes, the classical rules to sketch by hand a root locus. Figure 3 shows an example where two branches touch outside of the real axis; this example is simple enough to be used in the classroom (if time permits). Figure 4 shows an example where the exceptional case $K = -1$, for monic $D(s)$ and $N(s)$, must be taken in consideration. Note that in the even simpler case $D(s) = s + 1$, $N(s) = s + 2$, and $K < 0$, for example, the root locus drawn by MATLAB includes the line segment between the pole and the zero (together with other parts of the real axis), which is clearly wrong!

APPENDIX

In this appendix we will analyze the asymptotic behavior of the root locus when K goes to $\pm\infty$ for the case $m > n$. Since we are interested in the $m - n$ roots of the characteristic equation that go to infinity, it is convenient to write this equation in the form

$$\frac{D(s)}{N(s)} = s^{m-n}\frac{1 + a_1 s^{-1} + \cdots + a_m s^{-m}}{1 + b_1 s^{-1} + \cdots + b_n s^{-n}} = -K.$$

When $1 \gg |s^{-1}| \gg |s^{-2}|$ we can approximate this equation by

$$s^{m-n}\big(1 + (a_1 - b_1)s^{-1}\big) \approx -K.$$

Under the same conditions another standard approximation gives

$$s^{m-n}\left(1 + \frac{a_1 - b_1}{m - n}s^{-1}\right)^{m-n} \approx -K.$$

Thus, as $|K|$ goes to ∞ the $m-n$ roots of the characteristic equation that also go to infinity are given by

$$s + \frac{a_1 - b_1}{m - n} \approx \sqrt[m-n]{-K}.$$

(There are $m-n$ different values of $\sqrt[m-n]{-K}$, one for each branch of the root locus that goes to ∞.) The approximation gets better and better as $|s|$ increases, i.e., as $|K|$ increases (asymptotes).

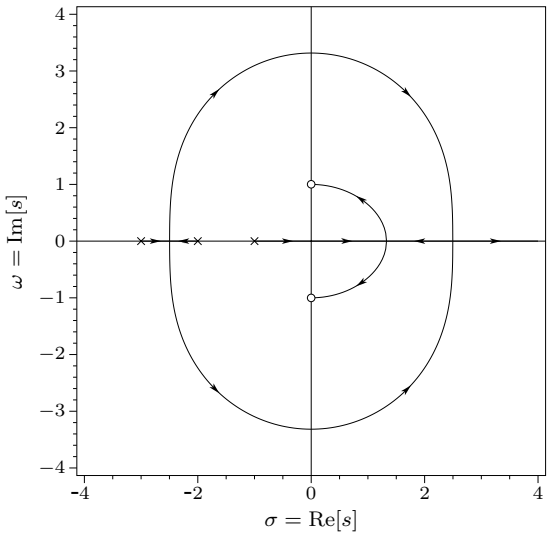

Fig. 2 - $D(s) = (s + 1)(s + 2)(s + 3)$, $N(s) = s^2 + 1$, and $K < 0$.

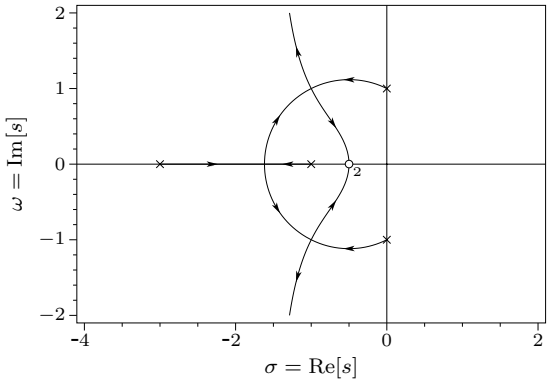

Fig. 3 - $D(s) = (s + 1)(s + 3)(s^2 + 1)$, $N(s) = 4s^2 + 4s + 1$, and $K > 0$.

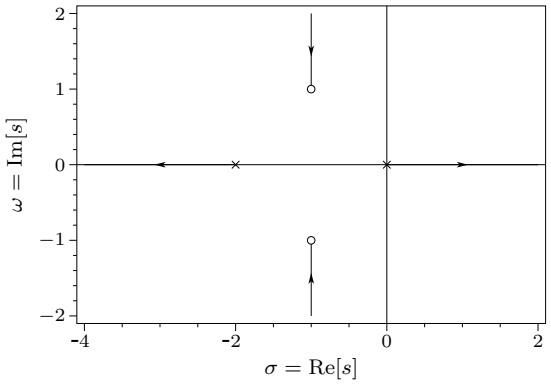

Fig. 4 - $D(s) = s(s+2)$, $N(s) = s^2+2s+2$, and $K < 0$. MATLAB does not draw correctly this root locus (it connects the horizontal and vertical segments with ugly diagonal lines).