Continuation Based Computation of Root-Locus for SISO Dead-Time Systems [1]

Suat Gumussoy *, **Wim Michiels** *

* *Department of Computer Science, K. U. Leuven, Celestijnenlaan 200A, 3001, Heverlee, Belgium (e-mails: {suat.gumussoy, wim.michiels}@cs.kuleuven.be).*

Abstract: We present a numerical method to plot the root-locus of Single-Input-Single-Output (SISO) dead-time systems on a given right half-plane up to a predefined controller gain. We compute the starting and intersection points of root-locus inside the region and we obtain the root-loci of each root based on a predictor-corrector type continuation method. The method is effective for high-order SISO dead-time systems.

Keywords: Root-locus, SISO dead-time systems, stability analysis, time delay.

1. INTRODUCTION

The root-locus method is an essential tool in modern control engineering for analysis and synthesis problems, Ogata (1997). This method is successfully implemented for finite dimensional SISO systems and becomes a fundamental tool in control education, Evans (2004); Krajewski and Viaro (2007).

The closed-loop of the SISO system with a time-delay and a constant gain has infinitely many poles in the complex-plane, Michiels and Niculescu (2007). Therefore the root-locus plot for dead-time systems is a difficult problem. Unlike the finite dimensional case, the root-locus equation contains a time-delay term and standard polynomial root-finding algorithms for the root computation are not available.

There are several approaches to plot the root-locus of SISO dead-time systems. The approximate root-locus can be plotted using the classical root-locus technique and the Padé approximation of time-delay term, Özbay (2006). This approach guarantees the accuracy of the approximation only inside a certain region and has numerical problems when high order Padé approximation is needed for larger regions. The direct approaches to which our method belongs, obtain the root-locus plot without approximating the time-delay term. A graphical method based on the computation of the root-locus gain and the closed-loop roots on various vertical lines in the complex-plane is presented in Huang and Li (1967). The root-locus on a rectangular region in the complex-plane is obtained by finding points on a rectangular grid satisfying the root-locus equation and connecting these points, Krall (1970). A continuation based method computes the root-locus on a rectangular region on the complex plane using the slope of the phase equation of the root-locus equation to determine a prediction step direction and a Newton-Raphson iteration to correct the prediction values, Ash and Ash (1968). The method detects the roots entering into the region by checking the sign of function values on the constant grid points of the region's boundary.

The root-locus equation of a SISO dead-time system is written as two real-valued equations, Yeung and Wong (1982). For a given constant imaginary part, the real part of the root on the root-locus is computed by finding the roots of a polynomial. By sweeping various values on the imaginary axis, the root-locus plot is obtained.

In Suh and Bien (1982), a continuation method for the root-locus analysis, Pan and Chao (1978), is extended to SISO dead-time systems. The root-loci branch is followed by computing the solution of non-linear differential equations with respect to the controller gain. The root-locus equation is transformed into another root-locus equation whose starting points are computed easily. Using differential equations for the second root-locus equation, these points are followed until the roots of both root-locus equations coincide. Since the common points are the initial points of the original root-locus equation, the root-locus plot is obtained for the original root-locus equations using its corresponding differential equations. In Nishioka et al. (1991) the same approach is used to compute the initial points of the roots entering into the region. Instead of computing the root-locus by solving differential equations, the controller gain is written as a function of other terms in the root-locus equation and the zeros of the imaginary part of the controller gain is computed by a triangulation method on the complex plane. The last two approaches are applicable to SISO time-delay systems with state-delays. However these methods follow the root-locus trajectories with respect to the controller gain which is numerically ill-posed due to the high sensitivity in the neighborhood of intersection points, which are characterized by the presence of multiple roots. The detection of asymptotic roots requires solving another root-locus problem and the number of required roots in a complex region is difficult to estimate.

[1] This article present results of the Belgian Programme on Interuniversity Poles of Attraction, initiated by the Belgian State, Prime Ministers Office for Science, Technology and Culture, the Optimization in Engineering Centre OPTEC of the K.U.Leuven, and the project STRT1-09/33 of the K.U.Leuven Research Foundation.

In this paper, we compute the root-locus plot of SISO dead-time systems on a given complex right half-plane up to a predefined controller gain. We calculate the starting points, the poles of SISO dead-time system and the roots entering into the region, and branching points of the root-locus inside the region. We follow the root trajectory based on a predictor-corrector type continuation method.

Our main contributions are the following:

- We compute all the roots entering into the region and their controller gains for an upper bound controller gain. By choosing the controller gain sufficiently large, the asymptotic behavior of the roots can be seen.
- Our continuation method estimates the next root on the complex plane by a simple linear predictor and corrects this prediction with a Newton method. Since the trajectory following method is based on a parameterization of curves in the (root,gain) space in terms of arclength, it is numerically robust.
- We use an adaptive step size in the prediction step depending on the convergence rate of the Newton method and the distance of the root from the root-locus trajectory. This makes our algorithm scalable by using different step sizes for different root-locus trajectories.
- Most of the methods in the literature requires many evaluations of the transfer function of the SISO dead-time system. The evaluation of the finite-dimensional transfer function is numerically expensive and the function values are not numerically stable due to the oscillation and exponential increase of the time-delay term in the imaginary axis and the positive real axis direction in the complex-plane respectively. We avoid these problems by transforming the root-locus equation into phase and logarithmic magnitude equations and evaluating in a numerically stable way. We need these evaluations only in the correction step.

The paper is organized as follows. In Section 2 we formulate the problem. The starting and branching points of root-locus inside the region are computed in Section 3. The predictor-corrector based continuation method is given in Section 4. The overall algorithm for root-locus plot is presented in Section 5. Section 6 is devoted to a numerical example. In Section 7 some concluding remarks are presented.

Notation:

$\mathbb{C}, \mathbb{R}, \mathbb{Z}$	: fields of complex, real and integer numbers,
$\Re(u)$	: real part of a complex number u,
$\Im(u)$	: imaginary part of a complex number u,
$\|u\|, \angle u$	: magnitude and phase of a complex number u,
$\lfloor u \rfloor$	: the next smallest integer close to a real number u,
$\lceil u \rceil$	: the next largest integer close to a real number u,
u^T	: the transpose of the vector u,
$\text{sign}(u)$	: returns $+1, -1, 0$ given a real number u for $u > 0, u < 0, u = 0$ respectively.

2. PROBLEM FORMULATION

A SISO dead-time system is a proper SISO system with a constant input or output time-delay, $h \in \mathbb{R}, h > 0$ and it has transfer function representation

$$G(s)e^{-hs} = \alpha \frac{\prod_{r=1}^{m} s-(\sigma_{zr}+j\omega_{zr})}{\prod_{i=1}^{n} s-(\sigma_{pi}+j\omega_{pi})} e^{-hs} \tag{1}$$

where $\alpha \in \mathbb{R}$, $\sigma_{zr} + j\omega_{zr}$ $r = 1, \ldots, m$, $\sigma_{pi} + j\omega_{pi}$ $i = 1, \ldots, n$ are the system gain, zeros and poles of G respectively. The *root-locus equation* of a SISO dead-time system is

$$f(s,k) = 1 + kG(s)e^{-hs} = 0 \tag{2}$$

where $k \in \mathbb{R}$, $k \geq 0$ is the controller gain. We define the *root-locus region* as

$$\mathbb{C}_{\sigma_0} = \{s \in \mathbb{C} : \Re(s) \geq \sigma_0\} \tag{3}$$

and the *boundary* of the root-locus region is a *given* vertical line parallel to the imaginary axis, $\Re(s) = \sigma_0 < 0$, $\sigma_0 \in \mathbb{R}$ and its value depends on the analysis requirements. We consider the following root-locus problem:

Problem: Compute the root-locus of the SISO dead-time system (1) inside the root-locus region $\mathbb{C}_{\sigma_0}$ for $k \in [0, k_{\max}]$ where $k_{\max}$ is a given positive real number.

When a root s of (2) crosses the boundary $\Re(s) = \sigma_0$ at the controller gain k, we determine whether it enters into or leaves the region $\mathbb{C}_{\sigma_0}$ by computing its *crossing direction*, Michiels and Niculescu (2007) defined as

$$\mathcal{CD}(s,k) := \text{sign}\left(\Re\left(-\frac{\frac{\partial f}{\partial k}}{\frac{\partial f}{\partial s}}\bigg|_{f(s,k)=0}\right)\right). \tag{4}$$

Note that a root on the boundary $\Re(s) = \sigma_0$ enters into or leaves the region $\mathbb{C}_{\sigma_0}$ when $\mathcal{CD}(s,k) > 0$ or $\mathcal{CD}(s,k) < 0$ respectively.

3. COMPUTATION OF CRITICAL POINTS OF ROOT-LOCUS

The *critical points* of root-locus are:

- the starting points of the root-locus, the poles of G inside $\mathbb{C}_{\sigma_0}$, and the roots of (2) crossing the boundary of the root-locus region $\Re(s) = \sigma_0$ for some $k \in [0, k_{\max}]$,
- the branching points of the root-locus where two or more root-locus trajectories intersect inside the region $\mathbb{C}_{\sigma_0}$.

The computation of roots of (2) crossing the boundary $\Re(s) = \sigma_0$ and their crossing directions are given in Section 3.1.

A branching point s satisfies the root-locus equation (2) and

$$\frac{\partial f(s,k)}{\partial s} = k(G'(s) - G(s)h)e^{-hs} = 0. \tag{5}$$

Thus, the branching points are the zeros of $G'(s) - G(s)h$ inside the region $\mathbb{C}_{\sigma_0}$, satisfying the root-locus equation (2) for a controller gain $k \in \mathbb{R}$, $k > 0$. Since $G'(s) - G(s)h$ is a rational transfer function, its zeros can be computed by standard polynomial root-finding algorithms and the branching points can be determined accurately. The behavior of a root around a branching point is given in the following Lemma, Suh and Bien (1982).

Lemma 1. Assume that $\tilde{s}$ is a root of the root-locus equation for a controller gain $\tilde{k}$, i.e., $f(\tilde{s}, \tilde{k}) = 0$ with multiplicity N, i.e.,

$$\frac{\partial^l f(s,\tilde{k})}{\partial s}\bigg|_{s=\tilde{s}} = 0,\ l = 1, \ldots, N-1,\ \text{and}\ \frac{\partial^N f(s,\tilde{k})}{\partial s}\bigg|_{s=\tilde{s}} \neq 0.$$

Then the root-locus has N intersecting trajectories at $s = \tilde{s}$ and the angle of direction change of a root incoming to and going from a branching point is 0 or $-\frac{\pi}{N}$ when N is odd or even respectively.

Since branching points inside the region $\mathbb{C}_{\sigma_0}$ and their multiplicities are computed before-hand, Lemma 1 allows us to determine the direction of a root-locus trajectory after a branching point.

3.1 Computation of Roots Crossing the Boundary of the Root-Locus Region

A root s on the boundary of root-locus region $\Re(s) = \sigma_0$ for the controller gain k satisfies the magnitude and phase equations of the root-locus equation (2). We first find the intervals on the boundary where the magnitude condition holds for some $k \in [0, k_{\max}]$. This is equivalent to finding the intervals on $\Re(s) = \sigma_0$ and $\omega \in [0, \infty)$ such that

$$K(\omega) := h\sigma_0 - \ln|G(\sigma_0 + j\omega)| \le \ln k_{\max}. \tag{6}$$

Lemma 2. Assume that G has no poles or zeros on the boundary of the root-locus region. The functions $K(\omega)$ and $K'(\omega)$ are continuous and the non-negative zeros of $K'(\omega)$ are the non-negative real roots of the polynomial,

$$\Gamma_z(\omega)\sum_{i=1}^{n}\Delta\omega_{pi}\Gamma_p^i(\omega) - \Gamma_p(\omega)\sum_{r=1}^{m}\Delta\omega_{zr}\Gamma_z^r(\omega) \tag{7}$$

where $\Delta\sigma_{zr} = (\sigma_0 - \sigma_{zr})$, $\Delta\omega_{zr} = (\omega - \omega_{zr})$, $\gamma_{zr}(\omega) = \Delta\sigma_{zr}^2 + \Delta\omega_{zr}^2$, $\Gamma_z^r(\omega) = \prod_{\substack{r_1=1 \\ r_1 \ne r}}^{m}\gamma_{zr}(\omega)$ for $r = 1, \dots, m$, $\Delta\sigma_{pi} = (\sigma_0 - \sigma_{pi})$, $\Delta\omega_{pi} = (\omega - \omega_{pi})$, $\gamma_{pi}(\omega) = \Delta\sigma_{pi}^2 + \Delta\omega_{pi}^2$, $\Gamma_p^i(\omega) = \prod_{\substack{i_1=1 \\ i_1 \ne k}}^{n}\gamma_{pi}(\omega)$ for $i = 1, \dots, n$, $\Gamma_z(\omega) = \prod_{r=1}^{m}\gamma_{zr}(\omega)$, $\Gamma_p(\omega) = \prod_{i=1}^{n}\gamma_{pi}(\omega)$.

Proof. Using the transfer function of G (1), the function $K(\omega)$ can be written as,

$$K(\omega) = h\sigma_0 - \ln|\alpha| + \tfrac{1}{2}\left(\sum_{i=1}^{n}\ln\gamma_{pi}(\omega) - \sum_{r=1}^{m}\ln\gamma_{zr}(\omega)\right). \tag{8}$$

The first derivative of $K(\omega)$ (8) is

$$K'(\omega) = \sum_{i=1}^{n}\frac{\Delta\omega_{pi}}{\gamma_{pi}(\omega)} - \sum_{r=1}^{m}\frac{\Delta\omega_{zr}}{\gamma_{zr}(\omega)}. \tag{9}$$

The functions $K(\omega)$ and $K'(\omega)$ are continuous except the points where $\gamma_{zr}(\omega)$ or $\gamma_{pi}(\omega)$ are equal to zero. These points are the poles or zeros of G on $\Re(s) = \sigma_0$. The continuity results in Lemma 2 follow from the assumption. The polynomial (7) is the numerator of the function $K'(\omega)$ in (9) and the result follows. □

Corollary 3. Assume that G has no poles or zeros on the boundary of the root-locus region. Then function $K(\omega)$ is monotonic on the intervals whose boundary points (without multiplicity) are successive non-negative zeros of $K'(\omega)$, 0 and ∞.

Proof. By Lemma 2, the function $K(\omega)$ is continuous since σ_0 is chosen such that there are no poles or zeros of G on $\Re(s) = \sigma_0$. Therefore it is monotonic inside the intervals determined by its extremum points and the end points of the boundary of the root-locus region, 0 and ∞. □

Since $K(\omega)$ is monotonic on each interval in Corollary 3, we find the subinterval in each interval where $K(\omega)$ satisfies (6). This is done as follows. If the values of $K(\omega)$ at the interval end points are smaller than $\ln k_{\max}$, then all $K(\omega)$ values in this interval are smaller than $\ln k_{\max}$ because $K(\omega)$ is monotonic. If one of the values of $K(\omega)$ at the interval end points is larger and the other one is smaller than $\ln k_{\max}$, we can find the point where $K(\omega)$ is equal to $\ln k_{\max}$ by a bisection algorithm and take the subinterval satisfying (6). If both values of $K(\omega)$ at the interval end points are larger than $\ln k_{\max}$, we discard that interval since all values of $K(\omega)$ are larger than $\ln k_{\max}$ and the condition (6) never holds. Based on this approach, we can compute the set of intervals I on the boundary of the root-locus region where the magnitude condition (6) is satisfied for some values of $0 \le k \le k_{\max}$.

The roots crossing the boundary of the root-locus region also satisfy the phase equation of (2):

$$(2l+1)\pi = \phi(\omega), \quad l \in \mathbb{Z}, \tag{10}$$

over the intervals I on $\Re(s) = \sigma_0$. Here the function $\phi(\omega)$ represents the *continuous* extension of the phase of the transfer function $G(s)e^{-hs}$ and their equivalence is

$$\mod\left(\angle\, G(s)e^{-hs}\big|_{s=\sigma_0+j\omega}, 2\pi\right) = \mod\left(\phi(\omega), 2\pi\right)$$

where $\omega \in [0, \infty)$. The left hand-side of the equation (10) represents constant functions of ω. If we partition the intervals I into the subintervals such that the function $\phi(\omega)$ is monotonic on each subinterval, we can compute the boundary crossing roots by a bisection algorithm. The following results allow us to compute the intervals on $\Re(s) = \sigma_0$ where the function $\phi(\omega)$ is monotonic.

Lemma 4. Assume that G has no poles or zeros on the boundary of the root-locus region. Then the functions $\phi(\omega)$ and $\phi'(\omega)$ are continuous and the non-negative zeros of $\phi'(\omega)$ are the non-negative real roots of the polynomial,

$$\Gamma_p(\omega)\sum_{r=1}^{m}\Delta\sigma_{zr}\Gamma_z^r(\omega) - \Gamma_z(\omega)\sum_{i=1}^{n}\Delta\sigma_{pi}\Gamma_p^i(\omega) - h\Gamma_z(\omega)\Gamma_p(\omega) \tag{11}$$

where the functions $\Gamma_z^r(\omega)$ for $r = 1, \dots, m$, $\Gamma_p^i(\omega)$ for $i = 1, \dots, n$, $\Gamma_z(\omega)$ and $\Gamma_p(\omega)$ are defined in Lemma 2.

Proof. Using the transfer function of G (1), the function $\phi(\omega)$ is written as

$$\phi(\omega) = \phi_1(\omega) + \phi_0 \tag{12}$$

where

$$\phi_1(\omega) := \sum_{r=1}^{m}\tan^{-1}\frac{\Delta\omega_{zr}}{\Delta\sigma_{zr}} - \sum_{i=1}^{n}\tan^{-1}\frac{\Delta\omega_{pi}}{\Delta\sigma_{pi}} - h\omega \tag{13}$$

and ϕ_0 is the offset difference, 0 or π between $\phi(\omega)$ and $\phi_1(\omega)$ (13) defined as $\phi_0 = \angle G(\sigma_0) - \phi_1(0)$.

The first derivative of the function $\phi(\omega)$ is

$$\phi'(\omega) = \sum_{r=1}^{m}\frac{\Delta\sigma_{zr}}{\gamma_{zr}(\omega)} - \sum_{i=1}^{n}\frac{\Delta\sigma_{pi}}{\gamma_{pi}(\omega)} - h. \tag{14}$$

Following the same arguments in Lemma 2, the functions $\phi(\omega)$ and $\phi'(\omega)$ are continuous by the assumption. The polynomial (11) is the numerator of the function $\phi'(\omega)$ (14) and the result follows. □

Corollary 5. Assume that G has no poles or zeros on the boundary of the root-locus region. Then the function $\phi(\omega)$ is monotonic on each interval in the set of intervals I_ϕ whose boundary points (without multiplicity) are successive non-negative zeros of $\phi'(\omega)$, 0 and ∞.

Proof. The function $\phi(\omega)$ is continuous. The monotonicity of $\phi(\omega)$ changes only at the points where $\phi'(\omega) = 0$. The assertion follows. □

The intersection of two sets of intervals I and I_ϕ partitions I into the subintervals, $I = \cup_{i=1}^{n_I} I_i$, where $\phi(\omega)$ is monotonic on each interval I_i. Each intersection of the function $\phi(\omega)$ and the constant functions in the left hand side of (10) over the intervals I corresponds to a boundary crossing root since any such point on I satisfies both the magnitude condition (6) and the phase equation of the root-locus equation (10) on $\Re(s) = \sigma_0$. Since the function $\phi(\omega)$ is monotonic on I_i, we can compute each intersection point by a bisection algorithm for $\phi(\omega)$ over the interval I_i. The value of the ω at the intersection point is the imaginary part of the boundary crossing root on the interval I_i and the corresponding controller gain is the value of $K(\omega)$ for this point. If there is no horizontal line intersecting $\phi(\omega)$ on I_i, we discard the interval since there is no root crossing this interval. We compute the roots of (2) crossing the boundary of the root-locus region by the following algorithm.

Algorithm 1.

```
For each interval in $I_i = [\omega_i^L, \omega_i^R]$ of $I = \cup_{i=1}^{n_I} I_i$,
  (1) Compute $\phi_i^{\max}$, $\phi_i^{\min}$, the maximum and minimum of $\phi(\omega_i^L)$, $\phi(\omega_i^R)$.
  (2) Compute $l_i^{\max} = \left\lfloor \frac{\phi_i^{\max}}{2\pi} - \frac{1}{2} \right\rfloor$ and $l_i^{\min} = \left\lceil \frac{\phi_i^{\min}}{2\pi} - \frac{1}{2} \right\rceil$.
  (3) If ($l_i^{\min} > l_i^{\max}$) discard the interval $I_i$,
      else
      for $l = l_i^{\min}$ to $l_i^{\max}$
        • find the imaginary part of the boundary crossing root $\omega_{cr}$ at the intersection of the horizontal line $(2l+1)\pi$ and $\phi(\omega)$ (12) by a bisection algorithm over the interval $I_i$.
        • compute the corresponding controller gain for the boundary crossing root, $K_{cr} = K(\omega_{cr})$.
```

Note that the controller gain K_{cr} is equal to $K_{cr} = \ln k_{cr}$ where k_{cr} is controller gain in the root-locus equation (2). In the remainder of the paper, we use capital K and the small k for the controller gain in logarithmic base and the original one in (2).

By Algorithm 1, we compute all roots crossing the boundary of the root-locus region $\Re(s) = \sigma_0$ and the corresponding controller gain values for $k \in [0, k_{\max}]$. The crossing directions of these roots are determined using the following theorem.

Theorem 6. The crossing direction of a boundary crossing root, $s_{cr} = \sigma_0 + j\omega_{cr}$ only depends on the imaginary part ω_{cr} on the boundary of root-locus region $\Re(s) = \sigma_0$ and is equal to

$$\mathcal{CD}(s_{cr}, k_{cr}) = -\text{sign}\left(\phi'(\omega_{cr})\right). \qquad (15)$$

Proof. Using the transfer function representation in (1) and (9,14), we obtain

$$G'(s_{cr})G^{-1}(s_{cr}) - h = \phi'(\omega_{cr}) + jK'(\omega_{cr}). \qquad (16)$$

By (4) and (16), the crossing direction of s_{cr} at $k = k_{cr}$ is equal to

$$\mathcal{CD}(s_{cr}, k_{cr}) = \text{sign}\left(\Re\left(\left(k_{cr}\left(h - \frac{G'(s_{cr})}{G(s_{cr})}\right)\right)^{-1}\right)\right)$$
$$= -\text{sign}(\phi'(\omega_{cr})). \qquad □$$

By Theorem 6, the crossing directions of roots crossing $\Re(s) = \sigma_0$ are the same when their imaginary parts are inside the same interval of I_ϕ in Corollary 5. Using this result, we determine the crossing directions of boundary crossing roots from their imaginary parts. We group the boundary crossing roots according to their crossing directions and define the sets W^{in} and W^{out} as

$$W^{in} = \{s_\nu^I, K_\nu^I\}_{\nu=1}^{n_i} \text{ and } W^{out} = \{s_\nu^O, K_\nu^O\}_{\nu=1}^{n_o}$$

where $s_\nu^I = \sigma_0 + j\omega_\nu^I, K_\nu^I$ for $\nu = 1, \ldots, n_i$, $s_\nu^O = \sigma_0 + j\omega_\nu^O, K_\nu^O$ for $\nu = 1, \ldots, n_o$ are the boundary crossing roots entering into or leaving the root-locus region and their controller gains respectively.

Remark 1: The crossing direction formula (4) is well-posed (either $+1$ or -1) if there are no poles or zeros of G or branch points on the boundary of the root-locus region.
Remark 2: If the plant G is bi-proper (i.e., $d := G(\infty) \neq 0$), then the controller gain must be bounded as $k_{\max} < \frac{e^{h\sigma_0}}{|d|}$. For larger controller gains, the root-locus region $\mathbb{C}_{\sigma_0}$ always has infinitely many roots.

4. COMPUTING A ROOT-LOCUS TRAJECTORY

The starting points of the root-locus are the poles of G inside $\mathbb{C}_{\sigma_0}$ at $k = 0$ and the roots of the root-locus equation (2) entering into the root-locus region, $\sigma_0 + j\omega_\nu^I$ at $k = k_\nu^I := e^{K_\nu^I}$ for $\nu = 1, \ldots, n_i$. We compute each root-locus trajectory by a secant-predictor, Newton-corrector continuation method, Allgower and Georg (2003). In the prediction step, a line passing through the last two computed roots and controller gains is used to estimate the next root and controller gain at a certain distance (steplength) in the (root,gain) parameter space. This estimate is corrected using Newton's method in the correction step. The next iteration continues in a similar way, though the step length is adaptive.

4.1 Prediction Step

The predicted root and the controller gain computation in the prediction step requires the previous root, the controller gain, a direction and a step length. For each root-locus trajectory, the starting point s_0 is available. The direction of the prediction step d_i is computed as follows:

- Initial directions $d_0^s \in \mathbb{C}$ for the poles of G inside $\mathbb{C}_{\sigma_0}$ and the boundary crossing roots are computed by the phase equation of root-locus (2) and by the phase of the derivative of the roots with respect to the controller gain for boundary crossing roots, i.e.,

$$\left.\frac{\partial s}{\partial k}\right|_{(s,k)=(s_l^i,k_l^i)} = -\left.\left(k\left(\frac{G'(s)}{G(s)} - h\right)\right)^{-1}\right|_{(s,k)=(s_l^i,k_l^i)},$$
$$= -\left(k_l^i\left(\phi(\omega_l^i) + jK(\omega_l^i)\right)\right)^{-1}.$$

  Set the root-locus direction as $d_i = [\,\Re(d_0^s)\ \Im(d_0^s)\ 1\,]^T$ and normalize to 1.
- The directions in other iterations are computed using the real and imaginary parts of the last two corrected roots and the controller gains, $\tilde{s}_i^c = [\,\sigma_i^c\ \omega_i^c\ K_i^c\,]^T$, $\tilde{s}_{i-1}^c = [\,\sigma_{i-1}^c\ \omega_{i-1}^c\ K_{i-1}^c\,]^T$, as

$$d_i = \frac{\tilde{s}_i^c - \tilde{s}_{i-1}^c}{\|\tilde{s}_i^c - \tilde{s}_{i-1}^c\|},\ i \geq 1. \qquad (17)$$

The real and imaginary parts of the predicted root and the controller gain $\tilde{s}^p_{i+1} = \left[\, \sigma^p_{i+1}\ \omega^p_{i+1}\ K^p_{i+1}) \,\right]^T$ are computed using a line equation with a step length h_i

$$\tilde{s}^p_{i+1} = \tilde{s}^c_i + d_i h_i,\ i \geq 0. \tag{18}$$

The initial step length h_0 is fixed. The step lengths in other iterations are calculated adaptively based on previous values, as we outline later on.

4.2 Correction Step

We use Newton's method to solve a set of nonlinear equations to find the real and imaginary parts of the corrected root and the corrected controller gain $\tilde{s}^c_{i+1} = \left[\, \sigma^c_{i+1}\ \omega^c_{i+1}\ K^c_{i+1}) \,\right]^T$. These equations are given by

$$M(\sigma^c_{i+1}, \omega^c_{i+1}, K^c_{i+1}) = 0 \tag{19}$$

$$P(\sigma^c_{i+1}, \omega^c_{i+1}) = 0 \tag{20}$$

$$(\tilde{s}^c_{i+1} - \tilde{s}^p_{i+1}) d_i = h_i \tag{21}$$

where

$$\begin{aligned} M(\sigma,\omega,K) &= \ln|\alpha| + \tfrac{1}{2}\textstyle\sum_{r=1}^m \left(\ln(\sigma-\sigma_{zr})^2 + (\omega-\omega_{zr})^2\right) \\ &\quad - \tfrac{1}{2}\textstyle\sum_{i=1}^n \left(\ln(\sigma-\sigma_{pi})^2 + (\omega-\omega_{pi})^2\right) - h\sigma + K, \quad (22) \\ P(\sigma,\omega) &= \angle\alpha + \textstyle\sum_{r=1}^m \tan^{-1}\frac{\omega-\omega_{zr}}{\sigma-\sigma_{zr}} - \textstyle\sum_{i=1}^n \tan^{-1}\frac{\omega-\omega_{pi}}{\sigma-\sigma_{pi}} \\ &\quad - h\omega - \pi, \quad (23) \end{aligned}$$

and $P(\sigma,\omega)$ has a range $(-\pi,\pi]$.

The functions M (22) and P (23) are equivalent representations of the magnitude and phase equations of the root-locus equation (2). The arctangent functions in $P(\sigma,\omega)$ are implemented as two argument function atan2 with the range $(-\pi,\pi]$. The equation (21) guarantees that the (linearized) distance of the corrected root and the controller gain $\tilde{s}^c_{i+1}$ from the predicted root and the controller gain $\tilde{s}^p_{i+1}$ is equal to the step size h_i.

Based on the set of equations in (19-21), we implement Newton's method as

$$(J^m_{i+1})\left(\tilde{s}^{m+1}_{i+1} - \tilde{s}^m_{i+1}\right) = -\tilde{f}^m_{i+1},\ m = 0,1,\ldots,m_{\tilde{s}} \tag{24}$$

where $\tilde{s}^m_{i+1} = \left[\, \sigma^m_{i+1}\ \omega^m_{i+1}\ K^m_{i+1} \,\right]^T$ is the vector of the real and imaginary part of the corrected root and the controller gain at m^{th} Newton iteration. The function $\tilde{f}^m_{i+1}$ and its Jacobian J^m_{i+1} are defined as

$$\tilde{f}^m_{i+1} = \begin{pmatrix} M(\sigma,\omega,K) \\ P(\sigma,\omega) \\ (\tilde{s}^m_{i+1} - \tilde{s}^p_{i+1})d_i - h_i \end{pmatrix},\ J^m_{i+1} = \begin{pmatrix} \frac{\partial M}{\partial\sigma} & \frac{\partial M}{\partial\omega} & \frac{\partial M}{\partial K} \\ \frac{\partial P}{\partial\sigma} & \frac{\partial P}{\partial\omega} & 0 \\ & d_i^T & \end{pmatrix}$$

where $(\sigma,\omega,K) = (\sigma^m_{i+1}, \omega^m_{i+1}, K^m_{i+1})$. The initial point $\tilde{s}^0_{i+1}$ for the correction step is the vector of the real and imaginary part of the predicted root and the controller gain from the prediction step $\tilde{s}^0_{i+1} = \tilde{s}^p_{i+1}$.

The Newton iterations continue until the root and the controller gain converge to a point within a predefined tolerance and the corrected root and the controller gain are set to the last iteration value in Newton method, $s^c_{i+1} = \tilde{s}^{m_{\tilde{s}}}_{i+1}$.

4.3 Adaptive Step Length

The step length computation for the next prediction step depends on two factors, Allgower and Georg (2003)

- the *contraction rate* of the first two successive Newton steps in the corrector step, i.e.,
$$\kappa_{i+1} := \frac{\|J^0_{i+1}\tilde{f}^1_{i+1}\|}{\|J^0_{i+1}\tilde{f}^0_{i+1}\|};$$
- the *distance* to the root-locus
$$\delta_{i+1} = \left\|1 - e^{M(\sigma^c_{i+1},\omega^c_{i+1},K_{i+1}) + jP(\sigma^c_{i+1},\omega^c_{i+1})}\right\|.$$

The individual deceleration factors are calculated as

$$\kappa_{df} = \sqrt{\tfrac{\kappa_{i+1}}{\tilde{\kappa}}},\ \text{and}\ \delta_{df} = \sqrt{\tfrac{\delta_{i+1}}{\tilde{\delta}}}$$

where $\tilde{\kappa}$, $\tilde{\delta}$ are the nominal contraction rate and the distance. The overall deceleration factor of the step length is computed as

$$h_{df} := \max\{\kappa_{df}, \delta_{df}\}$$

and limited to $[\frac{1}{2}, 2]$,

$$\bar{h}_{df} := \max\{\min\{h_{df}, 2\}, \tfrac{1}{2}\}.$$

Note that if $\bar{h}_{df} = 2$, the predictor step is repeated with a reduced step length. This check is done inside the corrector step to avoid unnecessary Newton iterations (see Allgower and Georg (2003) for further details). The step length for the next prediction step is

$$h_{i+1} = h_i / \bar{h}_{df}. \tag{25}$$

5. ALGORITHM

We compute the root-locus of the SISO dead-time system (1) inside the root-locus region $\mathbb{C}_{\sigma_0}$ by the following algorithm.

(1) Compute the critical points of root-locus as explained in Section 3, i.e. the starting and branching points of root-locus trajectories,
(2) For each root-locus trajectory computation:
  (a) Using the starting point as an initial root, compute the next root as explained in Section 4 by computing
    (i) the predicted root and the controller gain, $\tilde{s}^p_{i+1}$,
    (ii) the corrected root and the controller gain, $\tilde{s}^c_{i+1}$,
    (iii) the step length computation for the next prediction step, h_{i+1}.
  until that the trajectory reaches a branching point or the controller gain of the root exceeds $\ln k_{\max}$ or the trajectory leaves the root-locus region $\mathbb{C}_{\sigma_0}$.
  (b) When the trajectory reaches a branching point, go to Step $2-a$) and continue to compute the roots and the controller gains where the starting point is the branching point and the initial direction is calculated by Lemma 1.
  (c) When the controller gain exceeds $\ln k_{\max}$ or the trajectory leaves the root-locus region, stop the computation of the roots and the controller gains for this trajectory (If the trajectory leaves $\mathbb{C}_{\sigma_0}$, check that it crosses one of the points in the set

W^{out}). Go to Step $2-a$) and start to compute another root-locus trajectory by choosing another starting point.

Note that if the root-locus trajectory is on the real axis, we can continue from the next branching point with the controller gain less than $\ln k_{\max}$ or compute the root on the real axis whose controller gain is equal to $\ln k_{\max}$ by a bisection algorithm.

Remark 1: The algorithm can be modified to include negative controller gains, $k \in [-k_{\max}, k_{\max}]$. Then the constant functions on the left-hand side of the phase equation in (10) should be $l\pi, l \in \mathbb{Z}$. The computation of root-locus trajectories for boundary crossing roots remains the same. The poles of G inside $\mathbb{C}_{\sigma_0}$ are traced in two steps, first from $k = 0$ to $k = k_{\max}$, then from $k = 0$ to $k = -k_{\max}$.

Remark 2: The algorithm can be extended to the case where $\Re(s) = \sigma_0 > 0$. The value of $k_{\max}$ can be chosen such that the asymptotic properties of the roots are observed.

6. EXAMPLE

We consider the following SISO dead-time system (1)

$$\left(\frac{s^2-10s+50}{s^3+4s^2+4.25s+1.25}\right)e^{-s}.$$

The nominal contraction rate and distance are set to $\tilde{\kappa} = 1.1$, $\tilde{\delta} = 10^{-3}$ and the tolerance for the corrector step is 10^{-6}. The root-locus trajectories inside the root-locus region $\Re(s) \geq -3.5$ for the controller gain interval $k \in [0, 5]$ are given in Figure 1. The boundary crossing roots enter the root-locus region and their corresponding trajectories can be seen. The root-locus trajectories of the poles of G inside the root-locus region, $s = -0.5, -1, -2.5$, are shown in Figure 2. The trajectories of $s = -0.5$ and $s = -0.1$ have a branching point at $s = -0.7$, then they converge to the zeros of G, $s = 5 \pm 5i$. The trajectory of $s = -2.5$ leaves the root-locus region.

7. CONCLUDING REMARKS

A continuation method to compute the root-locus of SISO dead-time systems within a root-locus region, a given right complex half-plane, is given. The method calculates the starting points of root-locus trajectories including the ones crossing the boundary of root-locus region. The roots on each root-locus trajectory are predicted by a secant method where the step length is adaptive and the predicted values are corrected by Newton's method. The implementation is numerically stable and effective for high-order systems.

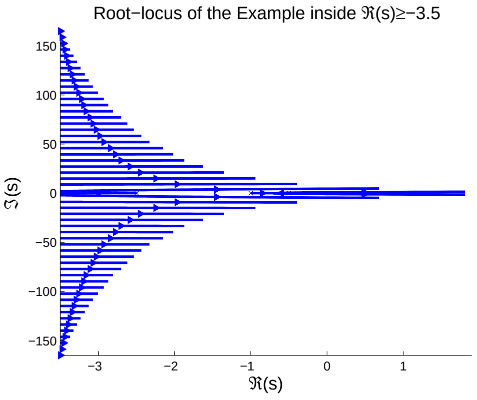

Fig. 1. The root-locus trajectories inside $\Re(s) \geq -3.5$

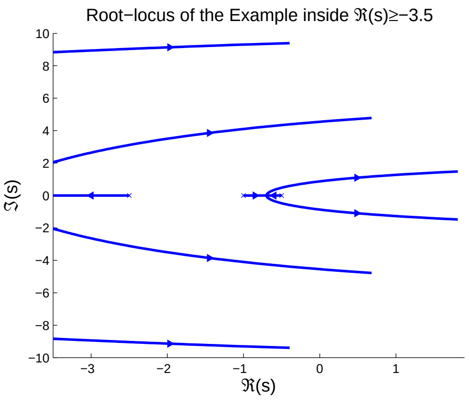

Fig. 2. The root-locus trajectories (zoomed)

REFERENCES

Allgower, E.L. and Georg, K. (2003). *Introduction to Numerical Continuation Methods*, volume 45 of *Classics in Applied Mathematics*. SIAM.

Ash, R.H. and Ash, G.R. (1968). Numerical computation of root loci using newton-raphson technique. *IEEE Transactions on Automatic Control*, AC13(5), 576–582.

Evans, G.W. (2004). Bringing root locus to the classrom. the story of walter r. evans and his textbook *control-sytem dynamics*. *IEEE Control Systems Magazine*, 24(6), 74–81.

Huang, I.B. and Li, L.L.C. (1967). Root locus determination of linear systems with transport lag. *IEEE Transactions on Automatic Control*, AC12(5), 632–634.

Krajewski, W. and Viaro, U. (2007). Root-locus invariance - exploiting alternative arrival and departure points. *IEEE Control Systems Magazine*, 27(1), 36–43.

Krall, A.M. (1970). Root locus method: A survey. *SIAM Review*, 12(1), 64–72.

Michiels, W. and Niculescu, S.I. (2007). *Stability and stabilization of time-delay systems. An eigenvalue based approach*, volume 12 of *Advances in design and control*. SIAM, Philadelphia.

Nishioka, K., Adachi, N., and Takeuchi, K. (1991). Simple pivoting algorithm for root-locus method of linear-systems with delay. *International Journal of Control*, 53(4), 951–966.

Ogata, K. (1997). *Modern control engineering*. Prentice Hall, New Jersey, 3rd edition.

Özbay, H. (2006). *The Root Locus Method*. Systems, Controls, Embedded Systems, Energy and Machines - The Electrical Engineering Handbook. CRC Press, Taylor & Francis Group, Boca Raton, FL, 3rd edition.

Pan, C. and Chao, K.S. (1978). A computer-aided root-locus method. *IEEE Transactions on Automatic Control*, 23(5), 856–860.

Suh, I.H. and Bien, Z. (1982). A root-locus technique for linear-systems with delay. *IEEE Transactions on Automatic Control*, 27(1), 205–208.

Yeung, K.S. and Wong, W.T. (1982). Root-locus plot of systems with time-delay. *Electronics Letters*, 18(11), 480–481.