Weak pole placement in strong stabilization of LTI SISO systems

1st Abdul Hannan Faruqi
Department of Mechanical Engineering
Indian Institute of Technology Kanpur
Kanpur, India
ahannanf20@iitk.ac.in

***Abstract*—Youla and coworkers proposed a solution in 1974 for finding stable controllers that realize specified closed-loop transfer functions for single-input-single-output (SISO) systems which satisfy the parity interlacing property (PIP). That method is manual, iterative, and tedious for systems with several right half-plane zeros. Later papers have not progressed in the direction of achieving desired closed-loop transfer functions, although stabilization with stable controllers (called *strong stabilization*) has been examined from a related perspective. Strong stabilization requires an interpolant that is a stable and proper transfer function, whose inverse is also stable and proper. In recent work (Faruqi and Chatterjee, 2025), we have proposed an interpolant using a factored form that initially has real (non-integral) powers, followed by free-parameter adjustments that make these powers integers in the end. We have called this method "real to integer" (RTI) and implemented it for the strong stabilization problem. Here we present a modified RTI method, with a more efficient factored form to obtain stable controllers for PIP-satisfying plants, that also allows for desired pole-placement. This method places closed-loop poles from a set of specified poles, where the poles may appear in the closed-loop transfer function with multiplicity one or higher. Hence, we call it weak pole-placement. The contribution of this paper is thus a modification of a recently proposed interpolant for a different classical problem in control design, leading to a new algorithmic solution implementable on a computer. Several examples are provided.**

***Index Terms*—control theory, pole placement, strong stabilization**

I. Introduction

We consider classical control of linear time-invariant (LTI) and single-input single-output (SISO) systems, using transfer functions in the Laplace domain, with the aim of stabilization with stable controllers in the standard single loop feedback configuration, so as to realize desired closed loop dynamics. The classical single-loop feedback control architecture shown in Fig. 1 where $P(s)$ is the plant, $C(s)$ is a forward controller, and $F(s)$ is an optional feedback controller (sometimes replaced by a static gain K). In this paper, the transfer functions $P(s)$ and $C(s)$ are assumed to be rational and proper, while $F(s)$ is taken to be the constant 1. In what follows, we denote the complex right half plane along with the imaginary axis as the RHP; and the left half plane without the imaginary axis as the LHP. Within this framework, if the plant is unstable (i.e., $P(s)$ has one or more poles in the RHP), then the first goal of control is to stabilize the closed loop system. To this end, we require the closed loop transfer function, which is

$$\frac{C(s)P(s)}{1+F(s)C(s)P(s)},$$

to have no poles in the RHP.

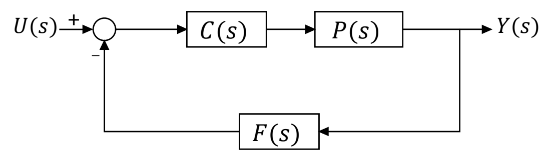

Fig. 1. Classical single-loop feedback control architecture.

In such stabilization, a second goal sometimes adopted is to achieve the stabilization using controllers $C(s)$ and $F(s)$ that are themselves stable, i.e., neither has poles in the RHP. Stable controllers are desirable because (i) they can be separately tested, (ii) the system can be more robust to sensor failures, and (iii) starting the system can be easier ([1], [2], [3]). It is well known (see [4], [5]) that stabilization with stable controllers as above, which is called *strong* stabilization, is possible if and only if the plant $P(s)$ satisfies the parity interlacing property (PIP): *between every pair of real zeros in the RHP, the number of real poles must be even.* Note that, in such control design, pole-zero cancellations in the RHP are not allowed.

An old problem, and one that is less frequently discussed in recent years, is the problem of strongly stabilizing a plant so as to obtain specified closed-loop dynamics. The classical paper by Youla *et al.* [4] presents one such method for obtaining any strongly realizable closed-loop transfer function using stable controllers $C(s)$ and $F(s)$. In recent years, there have been several papers devoted to strong stabilization of PIP-satisfying plants (see [6], [7], [8], etc.), but not so as to achieve arbitrarily specified strongly realizable closed loop transfer functions. Finally, in some recent work of our own [9], we have offered a way to construct stable stabilizing controllers that has some advantages over other methods presently available.

With the above background, we can now state the contribution of this paper. Noting that the algorithm given in [4] is manual, iterative, and tedious in the presence of several RHP plant zeros, we modify the more computationally oriented method of [9] so as to achieve a reduced objective (compared to [4]) using a relatively simple algorithm. In doing so, we only use the forward controller $C(s)$ and choose $F(s) = 1$. Our new method allows the control designer to specify desirable locations for the closed-loop poles, and the algorithm then places poles with multiplicity one or higher at a subset of the specified poles to minimize the controller order. We call this *weak* pole placement. While unable to realize a fully specified closed-loop transfer function as in [4], our method gives the designer significant room to adjust performance while being much more computationally efficient. In what follows, we will present the basic theory and follow up with numerical examples.

II. Brief theoretical background

For stabilization of a plant with co-prime factorization $P(s) = \dfrac{N(s)}{D(s)}$ (see [1], [2]), using a stable $C(s)$, we require that

$$U(s) = D(s) + N(s)C(s) \tag{1}$$

be a unit in the set of stable and proper transfer functions ($U(s)$ and its inverse both belong to this set), and

$$U(z_n) = D(z_n), \tag{2}$$

where z_n are the RHP zeros of $N(s)$, including zeros at infinity and counting multiplicities. Here, $N(s), D(s)$ are stable, proper transfer functions with no common zeros ($s = \infty$ included). Thus, $D(s)$ is biproper ($\lim_{s\to\infty} D(s) = 1$). Eq. (1) yields the controller

$$C(s) = \frac{U(s) - D(s)}{N(s)} \tag{3}$$

With this formulation, the closed-loop transfer function based on the architecture of Fig. 1 is

$$\begin{aligned} T(s) &= \frac{P(s)C(s)}{1 + P(s)C(s)} = \frac{\dfrac{N(s)}{D(s)}\,C(s)}{1 + \dfrac{N(s)}{D(s)}\,C(s)} \\ &= \frac{U(s) - D(s)}{U(s)} \end{aligned} \tag{4}$$

Thus, the closed-loop poles are the zeros of $U(s)$ and the poles of $D(s)$.

Several works in the literature recast this interpolation problem in different ways to utilize the Nevanlinna-Pick interpolation method [10], which finds an analytic function with magnitude bounded by unity on the unit disc to interpolate between points lying inside or on the unit disc. Among these, Ganesh and Pearson [11] have considered an exponential functional form for $U(s)$ wherein the function inside the exponent is constructed using Nevanlinna-Pick interpolation. Yücesoy and Özbay [5] have considered

$$U(s) = F(s)^{\ell}, \quad \|F\|_\infty < 1, \quad \ell \geq 1 \text{ is an integer,}$$

where

$$F(s) = \frac{G(s) + 1}{G(s) + \rho}, \quad \|G\|_\infty < 1, \quad \rho > 1 \text{ to be chosen,}$$

and where, finally, $G(s)$ is constructed using Nevanlinna-Pick interpolation. Large enough ℓ guarantees a solution [5]. In a recent paper, Gündeş and Özbay [7] have given formulas for controllers for systems with restrictions on the numbers and locations of RHP poles and/or zeros.

These methods are aimed at providing a unique interpolant $U(s)$ that is optimal in some way, although $D(s)$ can be chosen independently. Therefore, these methods do not allow placing all closed-loop poles at desired locations because the zeros of $U(s)$ cannot be pre-specified. Only the closed-loop poles coming from the poles of $D(s)$ can be placed. In contrast, in the RTI method [9], we define

$$U(s) = \prod_{k=1}^{r} \left(\frac{s + a_{2k-1}}{s + a_{2k}} \right)^{m_k} = f_k(s)^{m_k} \tag{5}$$

with a's ≥ 1 that are arbitrarily chosen. Herein lies the advantage of this method for placing closed-loop poles. For $N(s)$ with relative degree (δ_r) less than 2, i. e. zeros at infinity are not repeated, and having q distinct finite RHP zeros, we take the log of the interpolation equations (2) (substituting for $U(s)$ from eq. (5)) and put $r = q$ to obtain

$$\sum_{k=1}^{q} m_k \ln \left[\frac{s + a_{2k-1}}{s + a_{2k}} \right] = \ln \left[D\left(z_n\right) \right], \quad 1 \leq n \leq q. \tag{6}$$

Since $U(s) = 1$ at infinity, the interpolation condition at infinity is already satisfied. This system of equations is linear in the m's and is full rank, yielding real solutions whenever the plant satisfies PIP. For repeated RHP zeros, the redundant equations in the above system are supplemented with derivatives of those equations, which are also linear in the m's.

For plants with q finite RHP zeros and relative degree 2, consider

$$D(s) = \frac{s^p + c_1 s^{p-1} + \cdots}{s^p + d_1 s^{p-1} + \cdots},$$

for large s,

$$D(s) = 1 + \frac{c_1 - d_1}{s} + \cdots . \tag{7}$$

A similar large s expansion can be written for $U(s)$. Therefore, we pick any number $M > 0$ such that $c_1 - d_1 + M > 0$. Then

$$U(s) = \left(\frac{s + c_1 - d_1 + M}{s + M} \right) \prod_{k=1}^{q+1} f_k^{m_k}, \tag{8}$$

where an extra term is added to enforce the interpolation condition at $s \to \infty$. Again, we obtain the set of interpolation equations as

$$\sum_{k=1}^{q+1} m_k \ln f_k\left(z_n\right) = \ln D\left(z_n\right) - \ln \left(\frac{z_n + c_1 - d_1 + M}{z_n + M} \right), \tag{9}$$

$$\sum_{k=1}^{q+1} m_k \left(a_{2k-1} - a_{2k}\right) = 0, \tag{10}$$

where the last equation ensures that the f_k's do not contribute to the leading order (s^{-1}) in the large s expansion of $U(s)$, and the only contribution is from the pre-specified term that matches the leading order of $D(s)$. Detailed explanation may be found in [9].

The m's obtained at this stage are real and potentially large. Subsequently, the a's are varied until the m's are reduced to small integer values using optimization, and the final values are used to construct $U(s)$ using Eq. (5). The complete RTI algorithm is presented below (Algorithm 1)

Algorithm 1 Real-to-Integer (RTI)

Inputs: N, D. **Output**: U

$q \leftarrow$ number of finite right-half-plane (RHP) zeros of $N(s)$ (with multiplicities)

$\delta_r \leftarrow$ relative degree of $N(s)$

if $\delta_r < 2$ **then**

    Choose $2q$ parameters a_k

else if $\delta_r = 2$ **then**

    Choose $2q+2$ parameters a_k

    Choose M as in Section II.

end if

Minimize $\sum_k |m_k|$, with m_k obtained by solving the relevant equations (eqs. (6) or eqs. (9) and (10) depending on δ_r), over a's using unconstrained optimization.

Minimize $\sum_k \sin^2(m_k\pi)$ with a_k from the previous step as initial guesses again using unconstrained optimization.

Solve $\sin(\mathbf{m}\pi) = \mathbf{0}$ using modified Newton–Raphson iterations.

Finally, with the optimized a's

$$U(s) = \prod_k \left(\frac{s+a_{2k-1}}{s+a_{2k}}\right)^{m_k}.$$

Due to the optimization steps in the algorithm, the a's initially chosen to place closed-loop poles are changed. Realizing this and other limitations of the RTI method, we now present the Modified RTI method.

III. The Modified RTI Method

The main drawback of the RTI method is that $U(s)$ is made up of products of linear polynomials in the numerator and denominator (Eq. (5)), thus restricting it to have only real poles and zeros, which in addition occur in pairs of equal multiplicity. This prevents it from placing closed-loop poles anywhere in the complex left half-plane and may also prevent it from finding lower order interpolating functions. Also, as pointed out in the previous section, the RTI method loses the originally placed poles in the optimization process.

A. The modified form of $U(s)$

As discussed (see eq. (4) and the following discussion), the zeros of $U(s)$ contribute to the closed-loop poles. In order to be able to place the closed-loop poles anywhere in the RHP, $U(s)$ must include quadratic terms. Therefore, we define

$$U(s) = \frac{\prod_{j=1}^{n_c}(s^2+x_j s+y_j)^{M_j}}{\prod_{p=1}^{n_q}(s^2+a_{2p-1}s+a_{2p})^{M_{n_c+p}}} \times \frac{\prod_{k=1}^{n_r}(s+z_k)^{M_{n_c+n_q+k}}}{\prod_{q=1}^{n_l}(s+a_{2n_q+q})^{M_{n_c+n_q+n_r+q}}}, \tag{11}$$

$$= \prod_{j=1}^{n_c+n_q} e_j(s)^{m_j} \prod_{k=1}^{n_r+n_l} f_k(s)^{m_{n_c+n_q+k}}, \tag{12}$$

where the roots of $(s^2+x_j s+y_j)$ are the j^{th} pair of pre-specified complex conjugate closed-loop poles and $-z_k$ is the k^{th} pre-specified real closed-loop pole, and n_q and n_l are the pre-specified number of linear and quadratic terms to be used. In the simplified notation in Eq. (12), e's are the quadratic terms and f's are the linear terms. For this notation, the constraints are

$$\sum_{j=1}^{n_c+n_q} m_j = 0, \quad \sum_{k=1}^{n_r+n_l} m_{n_c+n_q+k} = 0, \tag{13}$$

$$m_j, m_{n_c+n_q+k} \geq 0, \quad j=[1,n_c], k=[1,n_r] \tag{14}$$

$$m_{n_c+p}, m_{n_c+n_q+n_r+q} \leq 0, \quad p=[1,n_q], q=[1,n_l], \tag{15}$$

$$a_p > 0, \quad q=[1,2n_q+n_l]. \tag{16}$$

Equation (13) ensures that $U(s)$ remains biproper, while inequalities (14) and (15) ensure that the pre-specified closed-loop poles remain as the zeros of $U(s)$ and no new zeros are added to $U(s)$ during the optimization process. As the pre-specified zeros of $U(s)$ lie in the left half-plane, the condition on a's in the inequality (16) ensures that $U(s)$ is a unit in stable, proper transfer functions.

In the next section, we will discuss the optimization procedure and show that integer m's can be obtained by varying only the a's. Thus, we shall be able to obtain the desired closed-loop poles, although with possibly higher multiplicities.

Remark 1: For this form of $U(s)$, $\lim_{s\to\infty} U(s) = 1$, and in general, we choose $D(s)$ such that $\lim_{s\to\infty} D(s) = 1$. Thus, $U(s) - D(s)$ has a single zero at infinity.

B. Linear programming problem formulation

Let the number of finite RHP zeros of $N(s)$ be q. Like Eq. (6), substituting for $U(s)$ from Eq. (12) in the interpolation conditions yields

$$\sum_{j=1}^{n_c+n_q} m_j \ln\left[e_j(z_r)\right] + \sum_{k=1}^{n_r+n_l} m_{n_c+n_q+k} \ln\left[f_k(z_r)\right] = \ln\left[D(z_r)\right], \tag{17}$$

$1 \leq r \leq q$. This system of equations is supplemented with

$$\sum_{j=1}^{n_c+n_q} m_j = 0, \tag{18}$$

$$\sum_{k=1}^{n_r+n_l} m_{n_c+n_q+k} = 0, \tag{19}$$

to ensure that $U(s)$ is biproper.

For $n_c + n + n_r + n_l = q + 2$, the above system will have a unique solution for distinct zeros (z_r). In case of repeated zeros, derivatives of the relevant equations are used in the same way as in the RTI method (Section IV-F of [9]). In general, multiplicity $p+1$ requires p^{th} derivatives. These equations are also linear in the m's.

With this, we have a linearly independent set of $q+2$ equations. In the M-RTI method, we allow $n_c+n+n_r+n_l \geq q+2$, and use the above system of equations as equality constraints. In addition, we implement the inequality constraints

$$\begin{aligned} -m_p \leq 0, \quad 1 \leq p \leq n_c \ \text{ and} \\ n_c + n_q < p \leq n_c + n_q + n_r \\ m_q \leq 0, \quad n_c < q \leq n_c + n_q \ \text{ and} \\ n_c + n_q + n_r < q \leq n_c + n_q + n_r + n_l \end{aligned}$$

Using these sets of constraints, we define a linear programming problem of dimension $n_c + n_q + n_r + n_l$ and minimize the objective function

$$F_{lp} = \sum_{j=1}^{n_c+n_q} 2m_j + \sum_{k=1}^{n_r+n_l} m_{n_c+n_q+k}. \tag{20}$$

Finally, this linear programming problem is solved for varying a's until small integer m's are obtained.

C. Existence of integer solutions

For each supplied real numerator term (closed-loop poles) in $U(s)$ in eq. (11), we can write

$$t_k(s) = \frac{(s+z_k)^{M_k}}{\prod_{q=1}^{N_k}(s+a_q)^{M_q}} = \frac{(s+\alpha_k-\epsilon)^{M_k}}{(s+\alpha_k+\epsilon)^{\sum_{q=1}^{N_k} M_q}}, \tag{21}$$

by letting the linear denominator terms have $a_q = z_k + 2\epsilon = \alpha_k + \epsilon$ with $\epsilon \ll 1$, for q from 1 to $N_k < n_l$ and

$$\sum_{q=1}^{N_k} M_q = M_k. \tag{22}$$

Let

$$\sum_k N_k = N_R < n_l. \tag{23}$$

Then, the remaining $n_l - N_R$ terms can have any value for the parameter a with the corresponding M_q's being 0.

Similarly, for each pair of complex conjugate closed-loop poles, i.e. the quadratic numerator terms in $U(s)$, we can write

$$t_j = \frac{(s^2+x_j s+y_j)^{M_j}}{\prod_{p=1}^{N_j}(s^2+a_{2p-1}s+a_{2p})^{M_p}} \tag{24}$$

$$= \frac{[(s+(\beta_j-\epsilon)+\gamma_j i)(s+(\beta_j-\epsilon)-\gamma_j i)]^{M_j}}{[(s+(\beta_j+\epsilon)+\gamma_j i)(s+(\beta_j+\epsilon)-\gamma_j i)]^{\sum_{p=1}^{N_j} M_p}} \tag{25}$$

by letting the quadratic denominator term parameters be $2(\beta_j+\epsilon)$ and $(\beta_j+\epsilon)^2+\gamma_j^2$, for p from 1 to $N_j < n_q$, and

$$\sum_{p=1}^{N_j} M_p = M_j. \tag{26}$$

Again, let

$$\sum_j N_j = N_C < n_q. \tag{27}$$

The remaining $n_q - N_C$ terms can have any parameter values with the corresponding M_p's being 0.

Finally, substituting in eq. (11), we obtain

$$\begin{aligned} U(s) = &\prod_{j=1}^{n_c} \frac{[(s+(\beta_j-\epsilon)+\gamma_j i)(s+(\beta_j-\epsilon)-\gamma_j i)]^{M_j}}{[(s+(\beta_j+\epsilon)+\gamma_j i)(s+(\beta_j+\epsilon)-\gamma_j i)]^{\sum_{p=1}^{N_j} M_p}} \\ &\times \prod_{p=N_C+1}^{n_q} \frac{1}{(s^2+a_{2p-1}s+a_{2p})^0} \\ &\times \prod_{k=1}^{n_r} \frac{(s+\alpha_k-\epsilon)^{M_k}}{(s+\alpha_k+\epsilon)^{\sum_{q=1}^{N_k} M_q}} \times \prod_{q=N_R+1}^{n_l} \frac{1}{(s+a_{2n_q+q})^0}. \end{aligned}$$

which simplifies to

$$\begin{aligned} U(s) = &\prod_{j=1}^{n_c} \left[\frac{(s+(\beta_j-\epsilon)+\gamma_j i)(s+(\beta_j-\epsilon)-\gamma_j i)}{(s+(\beta_j+\epsilon)+\gamma_j i)(s+(\beta_j+\epsilon)-\gamma_j i)}\right]^{M_j} \\ &\times \prod_{k=1}^{n_r} \left[\frac{(s+\alpha_k-\epsilon)}{(s+\alpha_k+\epsilon)}\right]^{M_k} \end{aligned} \tag{28}$$

The rest of the proof follows from Section IV-D of [9].

IV. Optimization for integer exponents

Since the controller is non-unique, any computational approach will involve some *ad hoc* or arbitrary choices, and for simplicity we have presented them as such instead of conducting additional heuristic studies. The main theoretical idea of our paper could be realized with other implementation details.

For any chosen vector $\mathbf{a}$ (set of closed-loop poles), we can compute a set of indices m_k. Now we define the preconditioning objective function

$$F_0(\mathbf{a}) = \sum_{j=1}^{n_c+n_q} 2\,|m_j| + \sum_{k=1}^{n_r+n_l} |m_{n_c+n_q+k}|, \tag{29}$$

and minimize it with respect to $\mathbf{a}$ to get the indices down to smaller values. At this stage the indices are real and not integers.

Note that, instead of the sum of absolute values, we could have used the sum of squares as well, or any of many other possible objective functions. The choice is *ad hoc* and the controller is non-unique.

As a practical matter, we have taken three steps to make the preconditioning optimization robust. First, for unconstrained input $\tilde{a}_k$, we specified

$$a_n = |\tilde{a}_n|, \tag{30}$$

so that optimization searches do not wander into regions with negative a's. Second, we penalized extremely large values of the a's, to make sure that optimization searches do not wander off to infinity. We penalized $\max|m_k|$ approaching too-small values ($U(s) \to 1$), so that the interpolation is not lost when we finally round-off the m's. Finally, we penalized any root of the quadratic terms whose real part approaches zero so that the closed loop remains stable under small perturbations. Thus, mapping $\tilde{a}$'s to a's as in Eq. (30), we adjusted Eq. (29) and actually minimized

$$F_1(\mathbf{a}) = F_0(\mathbf{a}) + Q + Q_{\text{offset}}.$$

In the above

$$Q = 10\left(1 - \max_k |m_k|\right) \cdot H\left(1 - \max_k |m_k|\right) + 0.005 \sum_n a_n^2,$$

where H denotes the Heaviside function ($H(x) = 1$ if $x > 0$, and $H(x) = 0$ otherwise) and where the coefficients 10 and 0.005 (large and small, respectively) are arbitrarily chosen. Also, the Q_{offset} is used to offset the real part of the roots by a pre-chosen value δ. We take $\delta = 0.5$ for the examples. Thus, we define

$$Q_{\text{offset}} = w \sum_k (\delta - p_k) \cdot H(\delta - p_k),$$

$$p_k = \begin{cases} \dfrac{a_{2_k - 1}}{2} & \Im(\Delta_k) > 0, k \le n_q \\ \dfrac{a_{2k-1} - \Delta_k}{2} & \Im(\Delta_k) = 0, k \le n_q \\ a_k, & k > 2n_q \end{cases},$$

where finally $\Delta_k = \sqrt{a_{2k-1}^2 - 4a_{2k}}$ is the square root of the k^{th} discriminant and the the weight w is chosen to be 100 arbitrarily.

After lowering the magnitudes of the indices (m's), we nudge them towards integer values by a second optimization calculation using the objective function

$$F_2(\mathbf{a}) = \sum_{k=1}^{n_c+n_q+n_r+n_l} \sin^2(m_k \pi) + Q_{\text{offset}},$$

where again we use a's from Eq. (30) and Q_{offset} is used (with a reduced w) to ensure that the offset remains robust. In the above, it is clear that $F_2 \ge 0$, with $F_2 = 0$ only when every m_k is an integer.

The two optimization calculations above are unconstrained and do not require sophisticated algorithms. We have used Matlab's built-in `fminsearch`. At the end of these optimization calculations, we obtain m's that are quite close to integers (up to 12 or more decimal places).

V. Examples

Example 1: Example 1 from Youla *et al.* [4].
Consider

$$P(s) = \frac{(1-s)(s+2)}{s(s-3)(s-4)}, \tag{31}$$

The co-prime factorization is done as

$$N(s) = \frac{(s-1)}{(s+2)(s+1)}, \quad D(s) = \frac{s(s-4)(s-3)}{(s+2)^2(s+1)}$$

We choose the set $\{-1, -2\}$ for the closed-loop poles and use our method with 2 linear and 2 quadratic terms, which yields the interpolant

$$U(s) = \frac{(s+1)^2}{(s+3.143)(s+1.897)}, \tag{32}$$

Thus, we expect the closed-loop poles to be present at -1 and -2, i.e. the zeros of $U(s)$ and the poles of $D(s)$. The controller obtained is

$$C(s) = \frac{-8.9606(s+3.229)(s-0.07289)}{(s+3.143)(s+2)} \tag{33}$$

and the closed-loop transfer function is

$$T(s) = \frac{8.9606(s+3.229)(s-1)(s-0.07289)}{(s+2.097)(s+0.8831)(s^2+2.124s+1.139)}, \tag{34}$$

which has poles close to -1 and -2 but they are slightly displaced due to numerical errors.

Example 2: Example 2 from Youla *et al.* [4].
Consider

$$P(s) = \frac{5(s-1)(s-2)}{s(s-3)(s-4)}, \tag{35}$$

with

$$N(s) = \frac{(s-2)(s-1)}{(s+3)(s+2)(s+1)},$$
$$D(s) = \frac{s(s-4)(s-3)}{(s+3)(s+2)(s+1)}.$$

This time we consider the set $\{-1, -2, -3\}$ for the closed-loop poles and using 4 linear and 4 quadratic terms, we obtain

$$U(s) = \frac{(s+3)^4(s+2)^4(s+1)^4}{(s+14.62)^5(s+0.05205)^7}, \tag{36}$$

which yields

$$C(s) = \frac{\begin{array}{c} -36.473(s-5.643)(s^2+0.5848s+0.109)\times \\ (s+0.3197)(s^2+0.3785s+0.1377)\times \\ (s^2-0.1762s+0.2414)(s^2+20.24s+105)\times \\ (s^2+21.49s+155.4) \end{array}}{(s+14.62)^5(s+0.05205)^7}.$$

The interpolant reported by Youla *et al.* [4] has order 8 while but, they have not reported the transfer functions of the controllers. By using their reported sensitivity function, we were not able to obtain stable controllers, as the cancellation was not exact. In our case, the controller is 12^{th} order and the closed-loop transfer function comes out to be

$$T(s) = \frac{\begin{array}{c} -36.473(s-5.643)(s-2)(s-1)\times \\ (s^2+0.3785s+0.1377)(s^2+20.24s+105)\times \\ (s^2-0.1762s+0.2414)(s^2+21.49s+155.4)\times \\ (s+0.3197)(s^2+0.5848s+0.109) \end{array}}{\begin{array}{c} (s+3)^5(s+2)^5(s+0.9746)\times \\ (s^2+2.043s+1.044)(s^2+1.982s+0.9827) \end{array}} \tag{37}$$

Example 3: Example 2 from Faruqi and Chatterjee [9].

Consider

$$P(s) = \frac{(s-0.3)(s-0.2)}{(s-5)(s-4)}, \quad (38)$$

with

$$N(s) = \frac{(s-0.3)(s-0.2)}{(s+5)(s+4)}, \quad D(s) = \frac{(s-5)(s-4)}{(s+5)(s+4)}.$$

Choosing the set $\{-4, -5\}$ for the closed-loop poles and using our method with 2 linear and 2 quadratic terms, we obtain

$$U(s) = \frac{(s+4)^5}{(s+9.517)^3(s+2.334)(s+0.473)} \quad (39)$$

which yields

$$C(s) = \frac{6.6413(s+22.01)(s+2.336)(s^2+16.69s+70.53)}{(s+9.517)^3(s+2.334)(s+0.473)}. \quad (40)$$

The corresponding closed-loop transfer function is

$$T(s) = \frac{\begin{gathered}6.6413(s+22.01)(s+2.336)\times \\ (s-0.3)(s-0.2)(s^2+16.69s+70.53)\end{gathered}}{(s+5)(s+4)^6} \quad (41)$$

which places the poles at the desired places. In Faruqi and Chatterjee [9], the controller which was 3rd order but it placed the closed-loop poles arbitrarily.

Example 4: Example 3 from Faruqi and Chatterjee [9]. Consider

$$P(s) = \frac{(s^2-2s+5)}{(s-2.5)(s^2+2s+5)}, \quad (42)$$

with

$$N(s) = \frac{(s^2-2s+5)}{(s^2+2s+5)(s+1)}, \quad D(s) = \frac{(s-2.5)}{(s+1)}. \quad (43)$$

For the closed-loop poles, we specify the set $\{-2.5, -1.5000 \pm 1.6583i\}$, where the latter are the roots of s^2+3s+5. Using 2 linear and 2 quadratic terms, we obtain

$$U(s) = \frac{(s^2+3s+5)^3}{(s^2+1.595s+6.581)(s^2+0.6571s+17.38)^2}$$

which yields

$$C(s) = \frac{\begin{gathered}11.091(s^2+2s+5) \\ \times(s^2+1.681s+6.409)(s^2+2.835s+14.86)\end{gathered}}{(s^2+1.595s+6.581)(s^2+0.6571s+17.38)^2} \quad (44)$$

The resulting closed-loop transfer function is

$$T(s) = \frac{\begin{gathered}11.091(s^2-2s+5)\times \\ (s^2+1.681s+6.409)(s^2+2.835s+14.86)\end{gathered}}{\begin{gathered}(s+2.404)(s^2+3.58s+5.173)\times \\ (s^2+2.457s+4.138)(s^2+3.06s+6.058)\end{gathered}} \quad (45)$$

which has the poles approximately where we had placed. The controller in [9] is 5th order but it doesn't offer desired closed-loop performance. A comparison of the step responses of the two controllers is shown in Fig. 2.

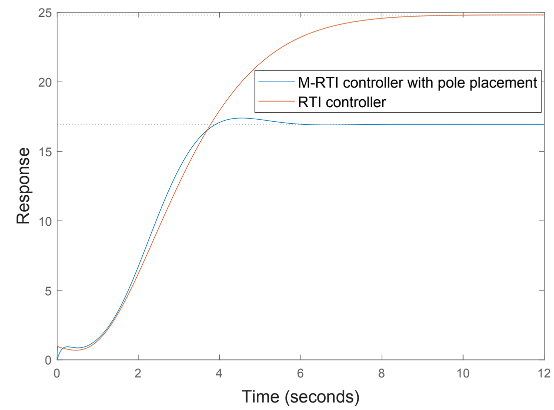

Fig. 2. Comparison of the step response of the controller obtained in Example 4 of this paper with the corresponding controller in Example 3 of [9].

VI. Conclusions

An extension of the RTI method of Faruqi and Chatterjee (2025), for strong stabilization, is presented. The RTI method lends itself to closed-loop pole placement through a theoretically small but practically significant modification, validated by numerical examples. As expected, not all specified poles end up being a part of the closed-loop transfer function, but the actual closed-loop poles do lie in close proximity of a subset of the pre-specified poles. Exploration of the sensitivity of the placed poles is left for future work. The multiplicity of the placed poles is sometimes more than one, as discussed in the theory. This has been achieved using only a stable controller in all examples. Therefore, the name 'weak pole placement in strong stabilization' is justified by the results.

References

[1] J. C. Doyle, B. A. Francis, and A. R. Tannenbaum, *Feedback Control Theory*. New York: Macmillan Publishing Co., 1992.

[2] M. Vidyasagar, *Control Systems Synthesis: A Factorization Approach, Part I*. Switzerland: Springer, 2022.

[3] M. Wakaiki, Y. Yamamoto, and H. Özbay, "Stable controllers for robust stabilization of systems with infinitely many unstable poles," *Systems & Control Letters*, vol. 62, no. 6, pp. 511–516, 2013.

[4] D. C. Youla, J. J. Bongiorno Jr, and C. N. Lu, "Single-loop feedback-stabilization of linear multivariable dynamical plants," *Automatica*, vol. 10, no. 2, pp. 159–173, 1974.

[5] V. Yücesoy and H. Özbay, "On the real, rational, bounded, unit interpolation problem in $\mathcal{H}_\infty$ and its applications to strong stabilization," *Transactions of the Institute of Measurement and Control*, vol. 41, p. 014233121875959, 2018.

[6] D. U. Campos-Delgado and K. Zhou, "A parametric optimization approach to $\mathcal{H}_\infty$ and $\mathcal{H}_2$ strong stabilization," *Automatica*, vol. 39, no. 7, pp. 1205–1211, 2003.

[7] A. N. Gündeş and H. Özbay, "Strong stabilization of high order plants," *Automatica*, vol. 140, p. 110256, 2022.

[8] X. Xin, Z. Wang, and Y. Liu, "Parameterization of minimal order strongly stabilizing controllers for two-link underactuated planar robot with single sensor," *Automatica*, vol. 158, p. 111280, 2023.

[9] A. H. Faruqi and A. Chatterjee, "New method for SISO strong stabilization with advantages over Nevanlinna–Pick interpolation," *IEEE Transactions on Automatic Control*, vol. 70, no. 7, pp. 4774–4779, 2025.

[10] H. Özbay, S. Gümüşsoy, K. Kashima, and Y. Yamamoto, *Frequency Domain Techniques for $\mathcal{H}_\infty$ Control of Distributed Parameter Systems*. Philadelphia: Society for Industrial and Applied Mathematics, 2018.

[11] C. Ganesh and J. Pearson, "Design of optimal control systems with stable feedback," in *1986 American Control Conference*, 1986, pp. 1969–1973.