Static Output Feedback Pole Placement via a Trust Region Approach

Kaiyang Yang*, *Member, IEEE,* and Robert Orsi†, *Member, IEEE*

***Abstract*— This paper presents two closely related algorithms for the problem of pole placement via static output feedback. The algorithms are based on two different trust region methods and utilize the derivatives of the closed loop poles. Extensive numerical experiments show the effectiveness of the algorithms in practice though convergence to a solution is not guaranteed for either algorithm. While desired poles must be distinct, strategies for dealing with repeated poles are also presented.**

***Index Terms*— Pole placement, static output feedback, trust region method, Newton's method, Levenberg-Marquardt method, eigenvalue derivatives.**

I. Introduction

Pole placement via static output feedback is a classical problem in systems and control theory. Given system matrices $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$, $C \in \mathbb{R}^{p\times n}$ and desired poles $\lambda^D \in \mathbb{C}^n$, the aim is find $K \in \mathbb{R}^{m\times p}$ such that $A+BKC$ has eigenvalues λ^D. (Desired poles with nonzero imaginary part are assumed to come in complex conjugate pairs.)

Instances of this problem are not always solvable. Furthermore the survey paper [1] states that sufficient conditions for solvability, such as Wang's condition (see below), are mainly theoretical in nature and that there are no good numerical algorithms available in many cases when a problem is known to be solvable. In fact determining solvability has recently been shown to be NP-hard [2]. In particular, this means that an efficient (i.e., polynomial time) algorithm that is able to correctly solve all instances of the problem cannot be expected.

There do exist some readily verifiable necessary or sufficient conditions for arbitrary pole placement. Necessary conditions include controllability and observability. One way to see this is to consider the associated system controllability and observability forms. Another necessary condition is that $mp \geq n$ [3], i.e., that the total number of variables is greater than or equal to the number of poles to be assigned. Regarding sufficient conditions, Kimura showed in 1975 that any controllable observable system satisfying $m + p > n$ is almost arbitrarily pole assignable [4]. Independently, a slightly weaker version of the result was given in [5]. Wang showed in 1992 that, for generic choices of system matrices, $mp > n$ is a sufficient condition for arbitrary pole placement [6]. (A simpler proof of this result was later given in [7].) For the borderline case, $mp = n$, arbitrary pole placement may not be possible. In [3] it is shown that for $m = p = 2$ and $n = 4$ there is a nonempty open set of system matrices for which arbitrary pole placement is not possible.

A great deal of research exists for the static output feedback pole placement problem. Starting points for further information are the survey papers [8], [9] and [1], as well as the more recent papers [10], [11], [12] and [13]. Regarding algorithms, we mention in particular the 'eigenstructure assignment' methods that can be used for systems meeting Kimura's condition; see [14], [15], [13] and the papers mentioned within. Approaches for the larger class of systems satisfying Wang's condition are fewer and include those presented in [16] and [17], both of which are based on 'linearization around a dependent compensator.' Practical use of the approach in [16] appears limited as produced are a family of controllers parameterized by a scalar variable ϵ, and while closed loop poles approach the desired poles as $\epsilon \to 0$, $K = K(\epsilon)$ may contain terms of the form $1/\epsilon$, resulting in asymptotically infinite entries. The numerical approach given in [17] may also produce K's with large entries and can involve difficult parameter tuning [13]. Another approach is to consider the expression $\det(sI - (A + BKC)) = \prod_i (s - \lambda_i^D)$. Equating coefficients gives a set of polynomial equations that characterize the solution set. In principle, one can then for example apply Gröbner basis techniques however the computational complexity of such methods limits their use to small dimensional problems [1]; see also [10]. Other approaches include [18], though convergence is not analyzed and it is not clear whether the method is even locally convergent; [15] and [19] which build on eigenstructure assignment methods; and [20] which is based on numerical homotopy methods.

In this paper we present two related numerical algorithms for solving static output feedback pole placement problems. Two trust region approaches are considered for solving the following unconstrained nonlinear least squares problem

$$\min_{K\in\mathbb{R}^{m\times p}} f(K) := \frac{1}{2}\|\lambda(A+BKC) - \lambda^D\|_2^2. \qquad (1)$$

Here $\lambda(A+BKC)$ denotes the vector of eigenvalues of $A+BKC$, with entries sorted to give the minimum norm. Trust region methods, which are well known in the optimization community, are a type of iterative method for minimizing nonconvex functions. The specific trust region methods we use are the trust region Newton method and the Levenberg-Marquardt method. In order to employ the trust region Newton method, at each iteration, the first and second derivatives

The second author acknowledges the support of the Australian Research Council through grant DP0450539.

*School of Information Technology and Electrical Engineering, University of New South Wales at the Australian Defence Force Academy, Canberra ACT 2600, Australia. Email: kaiyang.yang@adfa.edu.au

†Research School of Information Sciences and Engineering, The Australian National University, Canberra ACT 0200, Australia. Email: robert.orsi@anu.edu.au

of the eigenvalues of $A + BKC$ must be calculated. The Levenberg-Marquardt based algorithm has the advantage of only requiring the first derivatives of the eigenvalues. Both resulting algorithms have the desirable property that, near solutions, they can converge quadratically.

A technicality that arises in the proposed approaches is that the eigenvalues of $A + BKC$ may not be differentiable everywhere. They will however be differentiable at all points at which $A + BKC$ has distinct eigenvalues. A consequence of this is that the algorithms are only appropriate for problems for which the desired poles are distinct. It turns out that the algorithms can still be used to solve problems whose desired poles are distinct but whose separation is quite small and hence this does not appear to be a serious limitation.

Aside from the issue of repeated desired poles, the approaches can be applied without further restrictions. In particular there are no restrictions on n, m and p. For problems that are not solvable, a local minimizer of the cost will be found. The approaches are fundamentally different to those in [16] and [17] and as a result K's are not biased towards having large entries. Other advantages include the possibility to find multiple solutions by using different initial conditions. Simple modifications allow the approaches to be applied to problems for which the entries of K are constrained, for example, pole placement by decentralized control, in which case K must be block diagonal. The approaches can be used for either continuous time or discrete time systems.

The idea of solving pole placement type problems by utilizing eigenvalue derivatives is not completely new. Related ideas have been used to solve a non-control related inverse eigenvalue problem involving symmetric matrices [21]. However, the methods presented in [21] all require one to start sufficiently close to a solution for them to converge. The use of a trust region methodology means that this is not the case for our algorithms; this is an important distinguishing feature. First derivatives of eigenvalues (though not second derivatives) have also been used to solve various control problems. In [22] they are used to try to achieve pole placement in certain convex regions. The methodology that is used there is quite different to the one used here and is based on convex programming techniques. It requires that the open loop poles are already quite close to the desired poles. Using the eigenvalue derivatives, the change in the eigenvalues due to a small change in K is modelled by a first order approximation. This linear approximation is used within a convex program to try to place the poles in nearby regions. Similar ideas based on linear approximations of eigenvalues and convex programs have also been used for robustness analysis and stabilization; see [23].

The paper is structured as follows. Section II contains an overview of trust region methods. In order to use these methods to solve the pole placement problem, the first and second derivatives of f are required. Details of these calculations, including how to calculate derivatives of the eigenvalues, are given in Section III. Trust region methods require the function to be minimized to be differentiable. While f will typically only be differentiable on an open dense set, this turns out to be sufficient. One of the contributions of this paper is the realization that trust region methods can be successfully applied to such functions. Such issues are addressed in Section IV. Section V contains computational results, including results for a number of problems from the literature. The paper ends with some concluding remarks.

II. Trust Region Methods

This section gives an overview of trust region methods. It is assumed that the function $f : \mathbb{R}^N \to \mathbb{R}$ to be minimized is (sufficiently) smooth. The actual f we wish to minimize is given in (1) and may not satisfy this assumption. Issues related to this fact are addressed in Section IV. Additional information on trust region methods can be found in [24] and [25].

A. Basic Methodology

Trust region methods can be used to minimize smooth nonconvex functions and are iterative in nature. Given a current iterate x_k (in this section the variable is denoted by a generic vector $x \in \mathbb{R}^N$ rather than by K), they construct a possibly nonconvex, quadratic approximation of the objective function about x_k. This model is only assumed to be a good approximation in a certain ball centered about x_k. This is the so-called 'trust region'. It turns out that, numerically, it is possible to readily minimize a quadratic function over a ball. Doing so gives a candidate step p_k. The step p_k is only accepted if the difference in the objective function, $f(x_k) - f(x_k + p_k)$, is sufficiently close to the difference predicted by the model. If p_k is not acceptable, the trust region radius is decreased and the process repeated. If the model gives a good prediction, the trust region radius may be increased, allowing a larger step in the next iteration.

What follows describes the trust region method in greater detail. At each iteration, the quadratic approximation is assumed to be of the form $m_k(p) = f(x_k) + \nabla f(x_k)^T p + \frac{1}{2} p^T B_k p$. Here B_k is typically either the Hessian of f at x_k or some approximation of this Hessian. If B_k is the Hessian, then m_k is simply the 2nd order Taylor approximation of f at x_k. As will be discussed below, it may also be useful to consider other choices for B_k.

Each constrained minimization problem is of the form

$$\min_{p \in \mathbb{R}^N} m_k(p) \quad \text{s.t. } \|p\|_2 \leq \Delta_k, \tag{2}$$

where $\Delta_k > 0$ is the current trust region radius. The solution p_k of (2) gives a potential step. Whether or not it is a suitable step is assessed by considering the ratio of actual reduction of the objective to the predicted reduction:

$$\rho_k = \frac{f(x_k) - f(x_k + p_k)}{m_k(0) - m_k(p_k)}. \tag{3}$$

The overall trust region method is as follows.

Trust Region Method, Generic Algorithm ([24])

Given $\hat{\Delta} > 0$, $\Delta_0 \in (0, \hat{\Delta})$, and $\eta \in [0, \frac{1}{4})$:
for $k = 0, 1, 2, \ldots$
  Obtain p_k by (approximately) solving (2);
  Evaluate ρ_k from (3);
  if $\rho_k < \frac{1}{4}$
    $\Delta_{k+1} = \frac{1}{4}\Delta_k$

else
  if $\rho_k > \frac{3}{4}$ and $\|p_k\|_2 = \Delta_k$
    $\Delta_{k+1} = \min\{2\Delta_k, \hat{\Delta}\}$
  else
    $\Delta_{k+1} = \Delta_k$;
if $\rho_k > \eta$
  $x_{k+1} = x_k + p_k$
else
  $x_{k+1} = x_k$;
end(for).

Approximate solutions of the constrained quadratic minimization problem (2) can be obtained in a number of ways. One way is the nearly exact solution method described in [24, Section 4.2]. Without going into details we mention that this method is equivalent to solving a one dimensional root finding problem which can be solved using a Newton method.

Regarding the choice of B_k's, the Hessian of f at x_k is a natural choice. In this case, the method is called the *trust region Newton method*. When the objective function f is a least squares cost, say $f(x) = \frac{1}{2}\sum_{i=1}^{M} r_i^2(x)$ for some functions $r_i : \mathbb{R}^N \to \mathbb{R}$, as is the case in (1), there is another suitable choice. In this case, if we define $r(x) = (r_1(x), \ldots, r_M(x))^T$ and let $J(x)$ denote the Jacobian of $r(x)$, then $\nabla f(x) = J(x)^T r(x)$ and $\nabla^2 f(x) = J(x)^T J(x) + \sum_{i=1}^{M} r_i(x)\nabla^2 r_i(x)$, and a good choice for B_k is $J(x_k)^T J(x_k)$. The advantages of this choice for B_k include the fact that it does not require the calculation of the second derivatives of the r_i's and that it gives a good approximation of $\nabla^2 f(x_k)$ when $f(x_k)$ is small, that is, when each $r_i(x_k)$ is small. For this choice of B_k's, the method is called the *Levenberg-Marquardt method*.

B. Convergence Results

This subsection contains some general convergence results. These results are modified versions of those that are referenced. All assume the nearly exact solution method is used for the subproblems (2) and that the algorithm parameter η is non-zero, that is, $\eta \in (0, \frac{1}{4})$.

A simple but important property of trust region methods is that the cost is non-increasing from one iteration to the next: for all $k \geq 0$, $f(x_k) \geq f(x_{k+1})$.

Here are some less simple properties. The following result concerns global convergence to stationary points.

Theorem 1 ([24, Th 4.8]): Suppose that on the sublevel set $\{x \mid f(x) \leq f(x_0)\}$, f is twice continuously differentiable and bounded below, and that $\|B_k\| \leq \beta$ for some constant β. Then $\lim_{k\to\infty} \nabla f(x_k) = 0$.

Theorem 1 holds for both the trust region Newton method and the Levenberg-Marquardt method. For the former method, the following result also holds.

Theorem 2 ([24, Th 4.9]): Suppose the set $\{x \mid f(x) \leq f(x_0)\}$ is compact, that f is twice continuously differentiable on this set, and that $B_k = \nabla^2 f(x_k)$. Then the x_k's have a limit point x^* that satisfies the first and second order necessary conditions for a local minima, $\nabla f(x^*) = 0$ and $\nabla^2 f(x^*)$ is positive semidefinite.

The following result for the trust region Newton method implies that near a strict local minimum the method reduces to a pure Newton method and is quadratically convergent.

Theorem 3 ([24, Th 6.4]): Suppose that $B_k = \nabla^2 f(x_k)$. Further, suppose the sequence of x_k's converges to a point x^* that satisfies the first and second order sufficient conditions for a strict local minima, $\nabla f(x^*) = 0$ and $\nabla^2 f(x^*)$ is positive definite, and that f is three times continuously differentiable in a neighborhood of x^*. Then the trust region bound Δ_k becomes inactive for all k sufficiently large.

The final result in this section shows that the Levenberg-Marquardt method can be locally quadratically convergent to points whose cost is zero. (Note that this may not be the case for local minima whose cost is not zero.)

Theorem 4 ([24, Section 10.2]): Suppose the r_i's that determine f are three times continuously differentiable in a neighborhood of a point x^* satisfying $f(x^*) = 0$. Suppose further that $J(x^*)^T J(x^*)$ is positive definite. Then the Levenberg-Marquardt method is locally quadratically convergent to x^*.

III. Derivative Calculations

In order to apply the trust region methods described in the prior section, we need to calculate the appropriate first and second derivatives. As already mentioned, the eigenvalues of $A+BKC$ may not be differentiable everywhere. For example, the eigenvalues of $\left[\begin{smallmatrix}4 & 0\\ 0 & 0\end{smallmatrix}\right] + \left[\begin{smallmatrix}k\\ l\end{smallmatrix}\right]\left[\begin{smallmatrix}1 & 1\end{smallmatrix}\right]$ are $2 \pm \sqrt{4+4k}$ when $l = -k$ and hence they are not differentiable at $(k,l) = (-1,1)$. The next result, which follows from a result in [26, Section 2.5.7], shows that lack of differentiability cannot occur at points at which the eigenvalues are distinct.

Theorem 5: Consider a matrix valued function $A : \mathbb{R}^N \to \mathbb{R}^{n\times n}$. Suppose $A(x)$ is k-times continuously differentiable in x in an open neighborhood Ω. Further, suppose that at each point in Ω, $A(x)$ has distinct eigenvalues. Then the eigenvalues of $A(x)$ are k-times continuously differentiable in Ω.

Suppose the conditions of Theorem 5 are satisfied with $k \geq 2$. Then for any $x \in \Omega$ we can write down explicit expressions for the first and second derivatives of the eigenvalues of $A(x)$. If λ_i denotes the ith eigenvalue of $A(x)$, suppose $D = \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$ and let $X \in \mathbb{C}^{n\times n}$ be such that $A(x)X = XD$. Then

$$\frac{\partial \lambda_i}{\partial x_k} = \left(X^{-1} \frac{\partial A(x)}{\partial x_k} X \right)_{ii}, \tag{4}$$

and if we define

$$P = X^{-1}\frac{\partial A(x)}{\partial x_k}X \quad \text{and} \quad Q = X^{-1}\frac{\partial A(x)}{\partial x_l}X, \tag{5}$$

then

$$\frac{\partial^2 \lambda_i}{\partial x_k \partial x_l} = \left(X^{-1}\frac{\partial^2 A(x)}{\partial x_k \partial x_l}X \right)_{ii} + \sum_{\substack{j=1\\ j\neq i}}^{n} \frac{P_{ij}Q_{ji} + P_{ji}Q_{ij}}{\lambda_i - \lambda_j}. \tag{6}$$

Results (4)–(6) extend those appearing in [27], which deal solely with the scalar variable case, i.e., $N = 1$. They can be proved by appropriately modifying the techniques used in that paper. Details are not presented here.

The above results can be used to calculate the derivatives of our objective function $f(K) = \frac{1}{2}\sum_{i=1}^{n}(\lambda_i - \lambda_i^D)^*(\lambda_i - \lambda_i^D)$. Differentiating we have

$$\frac{\partial f(K)}{\partial K_{kl}} = \mathrm{Re}\left\{\sum_{i=1}^{n}(\lambda_i - \lambda_i^D)^*\frac{\partial \lambda_i}{\partial K_{kl}}\right\} \tag{7}$$

and

$$\frac{\partial^2 f(K)}{\partial K_{kl}\partial K_{pq}} = \mathrm{Re}\Bigg\{\sum_{i=1}^{n}\left(\frac{\partial \lambda_i}{\partial K_{kl}}\right)^*\left(\frac{\partial \lambda_i}{\partial K_{pq}}\right) + \sum_{i=1}^{n}(\lambda_i - \lambda_i^D)^*\frac{\partial^2 \lambda_i}{\partial K_{kl}\partial K_{pq}}\Bigg\}. \tag{8}$$

Note that '$\frac{\partial A(x)}{\partial x_k}$' is given by

$$\frac{\partial(A + BKC)}{\partial K_{kl}} = B_k C_l, \tag{9}$$

where B_k is the kth column of B and C_l is the lth row of C. Identity (9) implies that the first term appearing in (6) is always zero. Combining (4)–(9) we now have a complete characterization of the first and second derivatives of our cost (at points where $A + BKC$ has distinct eigenvalues).

For the Levenberg-Marquardt method, the approximate second derivatives are given by the real part of the first summation in (8).

IV. Additional Comments

When evaluating the cost $f(K)$, the eigenvalues of $A + BKC$ must be matched with the desired eigenvalues in a least squares sense. Suppose a given problem has distinct desired eigenvalues and that it is solvable. Then, near a solution of the problem, the eigenvalues of $A + BKC$ will be distinct, which eigenvalues of $A + BKC$ match to which desired eigenvalues will not change, and the eigenvalues of $A+BKC$ will depend smoothly on K. As a result, for problems that are solvable and have distinct desired eigenvalues, our objective function f will be smooth in a neighborhood of solutions. An important consequence of this is that the results from Subsection II-B regarding local convergence to solutions still apply. In particular, near solutions of problems with distinct eigenvalues, both our algorithms can converge quadratically.

The comments above address the behavior of the algorithms in a neighborhood of a solution. What about behavior far away from a solution? Are the algorithms even defined in such regions? Considering the steps involved, all that is required for the algorithms to be well defined is that, for each iterate, $A+BKC$ have distinct eigenvalues. If the desired eigenvalues are distinct and a generic initial condition is used, it is unlikely that for either algorithm, that for any iterate, $A + BKC$ has repeated eigenvalues. Hence, under these mild assumptions, the algorithms should be well defined and this is indeed what is observed in practice.

If the desired eigenvalues are not distinct, the cost may not be differentiable at a solution. This indicates that the requirement of distinct desired eigenvalues is also necessary. This does not limit the usefulness of the algorithms too much however as desired eigenvalues can always be perturbed slightly so that they are distinct. While having distinct but close eigenvalues does lead to a degree of ill-conditioning in our algorithms, the algorithms can still be effectively utilized in such cases, as will be shown in the numerical results section.

Regarding the global convergence properties of the algorithms, including behavior near non-optimal stationary points, a detailed analysis is required and will not be attempted in the present short paper. For the time being we simply mention that, if the desired eigenvalues are distinct, it is our belief that, modulo small changes, the global results of Subsection II-B should still hold, at least generically.

V. Computational Results

This section contains results for both random problems and problems from the literature. The random problem results include a comparison to the numerical homotopy based algorithm in [20], which is written in Ada and C, and is now part of the publicly available software PHCpack [28]. Our algorithms are coded using Matlab 7.1 [1]. All results were obtained using a 3.2 GHz Pentium 4 640 machine with 1 GB of RAM.

A. Random Problems

A 1000 random problems were created for each of a number of different choices for the system dimensions (n, m, p). As for all randomly generated matrices in the paper, entries in the system matrices were drawn from a normal distribution of zero mean and variance 1. Each λ^D was taken to be the spectrum of a randomly generated matrix, and a scalar was added to the entries of each λ^D to ensure $\max_i \mathrm{Re}\,\lambda_i^D = -0.1$. As each triple (n, m, p) was chosen to satisfy $mp > n$, Wang's sufficient condition ensured each problem was solvable.

For our algorithms, an attempt was made to solve each problem using up to 5 randomly chosen initial conditions and a maximum of 2000 iterations per initial condition. The convergence condition used was $\|\lambda(A + BKC) - \lambda^D\|_2 < \epsilon$, $\epsilon = 10^{-3}$. For the numerical homotopy algorithm, as it is able to find multiple solutions, some of which may be complex rather than real, it was limited to finding at most 10 solutions and was deemed successful if one of these was real. Results are given in Table I.

TABLE I

A comparison of performance for different n, m and p. S.R. denotes the success rate, T denotes the average solution time in CPU seconds, and i the average number of iterations. T and i are based only on those problems that were successfully solved. $\epsilon = 10^{-3}$.

	(n,m,p)	(3,2,2)	(6,4,3)	(9,5,5)
Trust Region Newton	S.R. (%)	100	100	92
	T	0.11	4.4	84
	i	44	260	910
Levenberg-Marquardt	S.R. (%)	100	100	99
	T	0.04	0.12	0.72
	i	41	110	380
Numerical Homotopy [20]	S.R. (%)	80	79	-
	T	0.87	22	-

[1]The code is available from http://rsise.anu.edu.au/∼robert/pole/.

Not surprisingly, given the reduced computation required for its implementation, the Levenberg-Marquardt based algorithm is faster than the trust region Newton based algorithm; notice in particular the large difference in T for the $(9,5,5)$ problems. What is perhaps surprising is that the Levenberg-Marquardt based algorithm is more likely to find a solution. Regarding the performance of the homotopy algorithm, for the $(9,5,5)$ problems, no solutions could be found due to memory limitations. For the other problems, the homotopy algorithm had lower success rates and was slower than our algorithms.

The problems in Table I are 'easy' in the sense that they actually satisfy Kimura's condition, $m+p>n$, and in most cases the number of variables mp is significantly larger than n. Results for some harder problems are presented in Table II. For these problems, $mp-n=1$. For our algorithms, maximum iterations per initial condition were increased to 5000.

TABLE II

SOME HARDER RANDOM PROBLEMS. $\epsilon = 10^{-3}$.

	(n,m,p)	(5,3,2)	(7,2,4)	$(9,2,5)$
Trust Region Newton	S.R. (%)	99	92	65
	T	1.9	15	51
	i	540	1800	2800
Levenberg-Marquardt	S.R. (%)	100	95	79
	T	0.25	1.3	4.0
	i	480	1300	2700
Numerical Homotopy [20]	S.R. (%)	100	98	89
	T	0.47	6.0	22

For these more difficult problems, the Levenberg-Marquardt based algorithm again performed better than the trust region Newton based algorithm. Overall, success rates for both algorithms decreased. The homotopy algorithm, compared to the better performing of the two trust region based algorithms, had slightly greater success rates at the expense of longer solution times.

B. Particular Problems

For the Levenberg-Marquardt based algorithm, Table III below contains results for particular problems from the literature. In order to present a number of results, rather than presenting the details of each problem, only references are given.

For each problem, 500 random initial conditions were tested. Maximum iterations per initial condition was 1000 (expect for Problem 8 for which 1500 was used). To highlight the accuracy that is achievable, the termination parameter ϵ was reduced to $\epsilon = 10^{-6}$.

Performance was again very good. Solutions could be found from many different initial conditions, and aside from Problem 8, for those initial conditions that lead to a solution, average convergence times were less than 0.3 CPU seconds.

Problem 8 was the most sensitive to initial conditions. In fact, its results in the table are based on choosing the entries of initial K's from a normal distribution of zero mean and variance 100. (Choosing initial conditions in the same manner as for all the other problems lead to a low success rate of 6%.)

TABLE III

PARTICULAR PROBLEMS. $\epsilon = 10^{-6}$.

No.	references	(n, m, p)	T	i	S.R. (%)
1	[5]	(3, 2, 2)	0.03	11	100
2	[13, case 1]	(5, 3, 2)	0.06	24	89
3	[13, case 2]	(5, 3, 2)	0.05	18	100
4	[14], [15, ex 2], [19, ex 3]	(5, 2, 4)	0.14	81	95
5	[15, ex 1, case 1]	(4, 2, 2)	0.14	100	78
6	[15, ex 1, case 2]	(4, 2, 2)	0.12	77	91
7	[18, ex 1]	(4, 2, 2)	0.08	41	96
8	[18, ex 2]	(6, 3, 2)	0.41	250	53
9	[18, ex 3]	(5, 3, 2)	0.22	130	100
10	[29, ex 1]	(4, 3, 2)	0.03	12	100
11	[29, ex 2]	(3, 1, 2)	0.02	4.0	100
12	[29, ex 3]	(4, 2, 2)	0.03	8.8	100
13	[30]	(8, 4, 3)	0.28	130	90

C. Repeated Eigenvalues

Each of the problems considered in the prior subsection (as well as all the random problems) had distinct desired eigenvalues. In this subsection we consider what can be achieved if the desired eigenvalues are not distinct. We consider the second example from [19]. For this problem $(n,m,p) = (6,2,3)$ and the desired poles are $\lambda^D = [-3,-3,-2,-2,-1,-1]^T$. The algorithms do not provide a way to exactly solve this problem. However, a fairly good approximate solution can be found by considering a slightly perturbed desired spectrum with distinct entries. For example, suppose λ^D is replaced with $\lambda^D_\delta = [-3-\delta,-3,-2-\delta,-2,-1-\delta,-1]^T$, $\delta = 10^{-4}$. Then this perturbed problem can often be solved.

An alternative strategy is to solve a series of perturbed problems with decreasing δ's. First solve a perturbed problem with $\delta = 10^{-1}$. Then, setting $\delta = 10^{-2}$ and using the solution of the prior problem as an initial condition, solve this new perturbed problem. Continue with $\delta = 10^{-3}$ and $\delta = 10^{-4}$.

Using the Levenberg-Marquardt based algorithm, the first strategy lead to a solution for 56% of initial conditions tried, with average convergence time of 0.59 CPU seconds. The second strategy was successful in 59% of cases, with average convergence time of 0.29 CPU seconds.

The main issue we encountered in solving these problems was not convergence to local minima, though this can occur, but rather that near solutions the Hessian of the cost can have very large eigenvalues. This leads to numerical issues when trying to solve the constrained quadratic subproblems (2). The code we have implemented for these subproblems works very well in the vast majority of cases though we expect it could still be improved further and hence that even better results may be achievable.

VI. CONCLUSION

In this paper two related numerical methods for the static output feedback pole placement problem have been presented. Both algorithms are well behaved globally and local convergence to solutions can occur quadratically. Extensive computational results presented in the paper indicate that the algorithms can be highly effective in practice. While it is required that the desired poles are distinct, the algorithms can still be successfully utilized for problems with repeated poles if small perturbations to the desired poles are allowed.

VII. Acknowledgements

The authors are very grateful to Joachim Rosenthal for pointing out [20], and Jan Verschelde for very helpful comments regarding this work.

References

[1] J. Rosenthal and J. C. Willems, "Open problems in the area of pole placement," in *Open problems in mathematical systems and control theory*, V. D. Blondel, E. D. Sontag, M. Vidyasager, and J. C. Willems, Eds. Springer Verlag, 1998, ch. 37, pp. 181–191.

[2] M. Fu, "Pole placement via static output feedback is NP-hard," *IEEE Trans. Automatic Control*, vol. 49, no. 5, pp. 855–857, 2004.

[3] J. C. Willems and W. H. Hesselink, "Generic properties of the pole placement problem," in *Proc. 7th IFAC Congress*, 1978, pp. 1725–1729.

[4] H. Kimura, "Pole assignment by gain output feedback," *IEEE Trans. Automatic Control*, vol. 20, pp. 509–516, 1975.

[5] E. J. Davison and S. H. Wang, "On pole assignment in linear multivariable systems using output feedback," *IEEE Trans. Automatic Control*, vol. 20, no. 4, pp. 516–518, 1975.

[6] X. A. Wang, "Pole placement by static output feedback," *J. Math. Systems, Estimation, and Control*, vol. 2, pp. 205–218, 1992.

[7] J. Rosenthal, J. M. Schumacher, and J. C. Willems, "Generic eigenvalue assignment by memoryless real output feedback," *Systems & Control Letters*, vol. 26, pp. 253–260, 1995.

[8] C. I. Byrnes, "Pole assignment by output feedback," in *Three decades of mathematical system theory*, ser. Lecture notes in control and inform. sci. 135, H. Nijmeijer and J. M. Schumacher, Eds. Springer Verlag, 1989, pp. 31–78.

[9] V. L. Syrmos, C. T. Abdallah, P. Dorato, and K. Grigoriadis, "Static output feeback – a survey," *Automatica*, vol. 33, no. 2, pp. 125–137, 1997.

[10] J. Rosenthal and F. Sottile, "Some remarks on real and complex output feedback," *Systems & Control Letters*, vol. 33, pp. 73–80, 1998.

[11] A. Eremenko and A. Gabrielov, "Pole placement by static output feedback for generic linear systems," *SIAM J. Control Optimization*, vol. 41, no. 1, pp. 303–312, 2002.

[12] K. H. Kiritsis, "A necessary condition for pole assignment by constant output feedback," *Systems & Control Letters*, vol. 45, pp. 317–320, 2002.

[13] O. Bachelier, J. Bosche, and D. Mehdi, "On pole placement via eigenstructure assignment approach," *IEEE Trans. Automatic Control*, vol. 51, no. 9, pp. 1554–1558, 2006.

[14] V. L. Syrmos and F. L. Lewis, "Output feedback eigenstructure assignment using two sylvester equations," *IEEE Trans. Automatic Control*, vol. 38, no. 3, pp. 495–499, 1993.

[15] A. T. Alexandridis and P. N. Paraskevopoulos, "A new approach to eigenstructure assignment by output feedback," *IEEE Trans. Automatic Control*, vol. 41, no. 7, pp. 1046–1050, 1996.

[16] J. Leventides and N. Karcanias, "Global asymptotic linearisation of the pole placement map: a closed-from solution for the constant output feedback problem." *Automatica*, vol. 31, no. 9, pp. 1303–1309, 1995.

[17] X. A. Wang, "Grassmannian, central projection and output feedback pole assignment of linear systems," *IEEE Trans. Automatic Control*, vol. 41, no. 6, pp. 786–794, 1996.

[18] T. H. Lee, Q. G. Wan, and E. K. Koh, "An iterative algorithm for pole placement by output feedback," *IEEE Trans. Automatic Control*, vol. 39, no. 3, pp. 565–568, 1994.

[19] L. Carotenuto, C. Franzè, and P. Muraca, "New computational procedure to the pole placement problem by static output feedback," *IEE Proceedings Control Theory & Applications*, vol. 148, no. 6, pp. 466–471, 2001.

[20] J. Verschelde and Y. Wang, "Computing dynamic output feedback laws," *IEEE Trans. Automatic Control*, vol. 49, no. 8, pp. 1393–1397, 2004.

[21] S. Friedland, J. Nocedal, and M. L. Overton, "The formulation and analysis of numerical methods for inverse eigenvalue problems," *SIAM J. Numer. Anal.*, vol. 24, no. 3, pp. 634–667, 1987.

[22] A. Hassibi, J. How, and S. Boyd, "Low-authority controller design via convex optimization," *AIAA J. of Guidance, Control, and Dynamics*, vol. 22, no. 6, pp. 862–872, 1999.

[23] B. T. Polyak and P. S. Shcherbakov, "Hard problems in linear control theory: possible approaches to solution," *Automation and Remote Control*, vol. 66, no. 5, pp. 681–718, 2005, translated from Avtomatika i Telemekhanika, no. 5, 2005, pp. 7–46.

[24] J. Nocedal and S. J. Wright, *Numerical Optimization*. New York: Springer, 1999.

[25] A. R. Conn, N. I. M. Gould, and P. L. Toint, *Trust-Region Methods*, ser. MPS-SIAM Series on Optimization. Philadelphia: SIAM, 2000.

[26] T. Kato, *A Short Introduction to Perturbation Theory for Linear Operators*. New York: Springer-Verlag, 1982.

[27] P. Lancaster, "On eigenvalues of matrices dependent on a parameter," *Numer. Math.*, vol. 6, pp. 377–387, 1964.

[28] J. Verschelde, "Algorithm 795: PHCpack: A general-purpose solver for polynomial systems by homotopy continuation," *ACM Transactions on Mathematical Software*, vol. 25, no. 2, pp. 251–276, 1999, software available at `http://www.math.uic.edu/~jan/download.html`.

[29] B. Sridhar and D. P. Lindorff, "Pole placement with constraint gain output feedback," *Int. J. Control*, vol. 18, no. 5, pp. 993–1003, 1973.

[30] H. Kimura, "A further result on the problem of pole assignment by output feedback," *IEEE Trans. Automatic Control*, vol. 22, no. 3, pp. 458–463, 1977.