Linear-Quadratic Optimal Reinforcement Learning Using Hewer's Algorithm

Pavan datta Reddy, *ECE*

***Abstract*—In Linear-Quadratic (LQ) Optimal Controller problems in discrete time, we optimize the performance criterion by going backwards in time. This multi-stage optimization is performed in this way because the performance criterion has a terminal condition at final time, N, so we consider optimizing backwards. When we go through this process of optimization at each stage, we obtain an optimal control, which we plug back into the optimal performance criterion to yield a new value at each consecutive stage. After finding the optimal performance, we can solve for the Lyapunov equation, given that we have the initial conditions. Note that the Lyapunov equation for each stage can be found in the optimal performance criterion in that particular stage (when arranged in a certain way). With this procedure, we can find the stabilizing solutions to the Lyapunov equation, which then can be used to find the stabalizing gain. With that gain we can find the optimal feedback control for each stage. Then, we plug the optimal control into the performance critertion to get a new performance value, and the process repeats over and over again. In general solving these Lyapunov equations are much simpler, because for each iteration we converge to the unique stabilizing solution, which is actually the solution to the discrete-time Riccati equation (DARE). This algorithm that is described called Hewer's Algorithm, and we will apply this algorithm in a Boeing 747 flight controls systems LQ problem.**

CONTENTS

I. INTRODUCTION

THE general system of a linear-quadratic system in discrete-time is given by:

$$x(k+1) = Ax(k) + Bu(k), \quad x(0) = x_0 \tag{1}$$

Note that $x(k)$ is the state, or plant, which is an element of the set R^n, $u(k)$ is the control input, which is an element of the set, R^m, and A and B are constant matrices with their appropriate dimensions with respect to x(k) and u(k). In general, with respect to this system, we want to find the optimal performance criterion, and the way find this optimum is through minimization.

Specifically, the quadratic performance of this system can be represented as follows:

$$J = \frac{1}{2}\sum_{k=0}^{N-1}\left[x^T(k)Qx(k) + u^T(k)Ru(k)\right] + \frac{1}{2}x^T(N)Hx(N)$$
$$Q \geq 0, \quad R > 0, \quad H \geq 0 \tag{2}$$

Q and H are positive semi-definite, and R is positive definite to clarify on the notation above. At time $k = N$, which is the final time, we have a boundary condition established where $J = \frac{1}{2}x^T(N)Hx(N)$. In general, we want to look for the optimal control values u(k), which give us the optimal performance. Let us define the controls, as $u(k) = F(k)x(k)$, where $F(k)$ is basically just the gain. We consider the controls in terms of the gain because after the state feedback, we want to guarantee stability of the system. Rewriting the equation for the LQ system, we get this form using the given definition for the control:

$$x(k+1) = (A + BF)x(k), \;\; x(0) = x_0 \tag{3}$$

In general, stability entails that the matrix $A+BF$ has to have eigenvalues that are to the left-side of the complex plane, so we have to find a gain matrix, which contains gain values that ensure that the LQ system is stable. As a result, we can then find the optimal control with the optimal gain, which then can help us find the optimal performance in the system. We can find the optimal performance by picking random gains that will obviously ensure stability, but that is a long, tedious process. Instead, we can use algebraic discrete-time equations, which solve for a matrix P, and has this relation with the gain:

$$P = (A + BF)^T P(A + BF) + F^T + RF + Q \tag{4}$$

$$F = -(R + B^T PB)^{-1}B^T PA \tag{5}$$

The equation to get matrix P is known as Lyapunov equation. In general, we want to ensure Lyapunov stability in this equation, where the system is asymptotically stable. Like how we discussed about the relationship between the gain and stability before, the same sort of concept applies here, but we want the system to be bounded. At the same time, it should also goes through the origin, which enables this idea of being asymptotically stable. Note that when we plug in the gain into the solution of the Ricatti equation, we will get the Ricatti equation back:

$$P = A^T PA + Q - A^T PB(R + B^T PB)^{-1} B^T PA \quad (6)$$

In general, the way to solve LQ optimal problem is to use the Bellman Functional Equations in discrete time, where we do static optimizations at discrete time. We start off at k=N and go backwards in discrete time to find optimal control values and optimal performance criterion at each stage because of the boundary condition we established before: $J = \frac{1}{2}x^T(N)Hx(N)$. In general at any k = N-k, we can get the following information regarding the optimal control, optimal performance criterion, and the matrix P that satisfies the Lyapunov equation using Bellman's Function Equation in discrete time, where $k = N - k$:

$$u^{opt}(N-K) = F^{opt}(N-K)x(N-K) \quad (7)$$

$$J^{opt}_{N-K,N} = \frac{1}{2}x^T(N-K)P(K)x(N-K) \quad (8)$$

$$P(K) = (A + BF(N-K))^T P(A + BF(N-K) + F^T(N-K) + RF(N-K) + Q \quad (9)$$

where P(0) = H and K = 1,2,...,N. At k=0 where K=N, the optimal performance is $J^{opt}_{0,N} = \frac{1}{2}x^T(0)P(N)x(0)$. In each iteration, the system feedback becomes $A+BF(N-K)$.

However, in this discrete-time case in regards to solving the algebraic discrete-time Riccati equations (DARE), for each iteration, these DAREs take time to solve. The only DAREs that seem simple to solve is in the special case where $N \to \infty$ and H=0 because we are optimizing over an infinite period of time and we do not have a boundary condition we have to consider. As a result, the gain matrix, F, as well as the matrix P to be found in the Lyapunov equation are usually constant for every iteration, but they can vary slightly over time, so we solve simpler DARE equations. If these conditions are not met, we, as a result, need a faster approach to find an optimal P that could give us an optimal gain, K, which helps us find an optimal control, which helps us find the optimal performance. Therefore, the Hewer's Algorithm is the most effective choice we have in finding a value P, and we will discuss this algorithm in the methodology.

Note that the systems for a time-varying and time-invariant case do not differ in the methodology for finding the optimal control and therefore the optimal performance. Just understand that instead the matrices are just functions of time in the time-varying systems. So, the linear-quadratic system does not really have to be specified in terms of being time-varying or time-invariant in our case, because we are just going to explain the general methodology of finding the optimal control using Hewer's algorithm. But, to be more specific, let us consider a time-invariant LQ system.

Also note that in an LQ system, we are going to have to consider some noise as well. Specifically, each state, which is in a Gaussian environment, will also have an added input white noise. So, the new equation becomes:

$$x(k+1) = Ax(k) + Bu(k) + w_d, \quad x(0) = x_0 \quad (10)$$

where $w(d)$ represents the external white noise we discussed. As a result, we now deal with a Linear-Quadratic-Gaussian (LQG) system, as the states are in a Gaussian environment as well as they receive external noise from the environment. As a result, the performance criterion of the LQ problem needs to be amended to account for the noise introduced into the system. As a result, we formulate a performance index as follows:

$$J = E[\sum_{k=0}^{N-1} \left[x^T(k)Qx(k) + u^T(k)Ru(k)\right] + x^T(N)Hx(N)] \quad (11)$$

which is just the expectation over the noise. Note that the LQG is just a combination of the LQ optimal state feedback but processed through a Kalman Filter. But, also the controls are fed into the Kalman Filter. As a result, the Kalman Filter will determine an estimated state of the solution, which will determine the optimal control to put into the system. In general, LQG basically consists of a linear quadratic regulator (LQR), who's methodology is same as LQ optimal controllers, and a Kalman Filter. For the LQR portion, we want the system to achieve some sort of Lyapunov stability, where we want the system to oscillate in small neighborhood but also converge to the origin, which also implies asymptotic stability. So, we want to find eigenvalues that are stable for this system, so they have to be on the left-side of the complex plane. But, more generally, these eigenvalues will help us find gains that ensure stability, and we find these gains once again by using the Riccati equation. In general, having an LQ system ensures controllablity in our system because we are able to maintain stability. For the Kalman Filter, we want to minimize the error between the observed value of the state and the actual value. Let's call this error as follows:

$$\epsilon = x - \hat{x} \quad (12)$$

In general, note in a Kalman Filter, observability is introduced because we need an observer to measure an estimated state. In the error equation above, x is the actual state while $\hat{x}$ is the observer's estimated state. As a result, we need to find eigenvalues for the system such that the estimated value of the state converges as fast as possible to true value of the state. As a result, the eigenvalues we find for the Kalman Filter help us obtain gains that would minimize this cost function for the Kalman Filter, which is just the covariance of the error:

$$J = E((x - \hat{x})^T(x - \hat{x})) \quad (13)$$

In reality, instead of finding arbitrary eigenvalues, we would use the Riccatti equation to find the appropriate gains that would help us minimize the cost function above. Also note that the noise, w_d, introduced in the system would also affect

where the eigenvalues would be placed to ensure stability. So, considering the LQR and the Kalman Filter, it seems that these two control systems have similar methodologies in finding gains that would stabilize their systems. Consequently, when finding the optimal control within an LQG system, we are trying to find the optimal gains for both the LQR and the Kalman Filter. But, more generally, the eigenvalues of both these control systems are maintained and contribute to the stability in the LQG. Note, that controllability and observability laws are also considered in LQG when using both of these systems.

Despite that fact that input noise can be introduced into a system, let us consider a system where there is not any noise for the sake of simplicity. In our scenario, with the Boeing 747, we will be using the Hewer's Algorithm to solve the LQ optimal control problem, and it will now be explored.

A. Methodology

In order to solve the problem given, which will be explained thoroughly later on, we will be using Hewer's Algorithm. In essence, the Hewer's Algorithm is similar to the Newton-Raphson method of finding roots to a function, where we converge closer and closer to a root starting from a given point on a function. But, in our case, when it comes to the Hewer's Algorithm we instead, for every iteration, find an optimal gain that stabilizes the system, which more specifically contains the stabilizing solution to a Lyapunov equation. These solutions to the Lyapunov equations for each iteration converge to a single stabilizing solution. So, just how we converge to the root with the Newton-Raphson method, we converge to the unique stabilizing solution. Note that stability refers to asymptotic stability of the feedback matrix where the eigenvalues are less than 1. But, instead of just explaining let us be more rigorous about this method. First let us define the convergent property of the Lyapunov solutions. Let's say we have a discrete time-invariant system that does not have any controls for the sake of simplicity

$$x(m+1) = Ax(m), \quad x(k) = x_k \tag{14}$$

where $m = k, k+1, k+2, ...$ In turn $x(m) = A^{m-k}x(k)$. As a result, the performance criterion of the system as a function of $x(k)$ can be written as follows:

$$J = \sum_{m=k}^{\infty} [x^T(m)Qx(m)] = x^T(k) (\sum_{m=k}^{\infty} [(A^{m-k})^T Q A^{m-k}])x(k) \tag{15}$$

As a result, J becomes:

$$J(x(k)) = x^T(k)Px(k) \tag{16}$$

where $P = \sum_{m=k}^{\infty}[(A^{m-k})^T Q A^{m-k}]$ and as a result satisfies the Lyapunov equation $P - A^T P A = Q$. Also note that this system is asymptotically stable where the eigenvalues of A are less than 1 ($|\lambda(A_i) < 1|, \forall i$). As a result, let us now solve a similar problem, but this time consider a time-variant quadratic linear system.

$$x(k+1) = Ax(k) + Bu(k), \quad x(k) = x_k \tag{17}$$

where the performance criterion is

$$J(x(k)) = \frac{1}{2}\sum_{m=k}^{\infty} \left[x^T(m)Qx(m) + u^T(m)Ru(m)\right] \quad Q = Q^T \geq 0, \quad R = R^T > 0 \tag{18}$$

Note that the matrices Q and R have to be symmetric matrices for us to use this algorithm, so be aware of this subtlety when picking appropriate weights. In essence, let start with an initial control with the stabilizing gain F_0:

$$u_0(x(k)) = -F_0 x(k) \tag{19}$$

$$x(k+1) = (A - BF_0)x(k) \tag{20}$$

So, we just substituted some values, specifically the stabilizing gain, in the system. However, let us rewrite the system in terms of m where $m = k, k+1, k+2, ...$, so we could consider the summation across all time k for each iteration.

$$x(m) = (A - BF_0)^{m-k}x(k) \tag{21}$$

.

Let us go through the first iteration now. The performance criterion of the first iteration is...

$$J_0(x(k)) = \frac{1}{2}\sum_{m=k}^{\infty}[x^T(m)(Q + F_0^T R F_0)x(m)] \tag{22}$$

the performance criterion can be simplified to...

$$J_0(x(k)) = \frac{1}{2}x^T(k)P_1 x(k) \tag{23}$$

where P_1 satisfies the Lyapunov equation...

$$P_1 - ((A - BF_0)^T P_1 (A - BF_0)) = Q + F_0^T R F_0 \tag{24}$$

which gives us another approximate optimal gain F_1, which more specifically gives us an optimal feedback control $u_1(x(k)) = -F_1 x(k)$. F_1 is important because it solved for in Bellman's functional equation in discrete-time when minimizing the performance criterion to find the optimal $u_1(k)$ in our second iteration of finding the approximate stabilizing solution. In general, we find the partial derivative in respect to u_1 of the performance criterion and minimize that...

$$\frac{\partial}{\partial u_1} = Ru_1(k) + B^T P_2 B u_1(k) + B^T P_2 A x(k) = 0 \tag{25}$$

As a result our approximate optimal control, which gives us the approximate optimal for our second iteration is...

$$u_1(k) = -(R + B^T P_1 B)^{-1} B^T P_1 A x(k) = -F_1 x(k) \quad (26)$$

As a result, we can plug the approximate gain for the second iteration back into our performance criterion $J_1(x(k))$ to get the approximate optimum.

$$J_1(x(k)) = \frac{1}{2} \sum_{m=k}^{\infty} [x^T(m)(Q + F_1^T R F_1) x(m)] \quad (27)$$

which can be simplified to...

$$J_1(x(k)) = \frac{1}{2} x^T(k) P_2 x(k) \quad (28)$$

where P_2 satisfies the Lyapunov equation...

$$P_2 - ((A - BF_1)^T P_1 (A - BF_1)) = Q + F_1^T R F_1 \quad (29)$$

After two iterations, the process for each iteration becomes obvious. First, we find the gain from using Bellman's functional equation in discrete-time to find the optimal control. Then we use this approximate gain to find the optimal performance, which gives us an approximate stabilizing solution to a discrete-time Lyapunov equation.

$$u_i(k) = -F_i x(k) \quad (30)$$

$$J(x(k)) = \frac{1}{2} x^T(k) P_{i+1} x(k) \quad (31)$$

$$P_{i+1} + (A - BF_i)^T P_{i+1} (A - BF_i) = Q + F_i^T R F_i \quad (32)$$

Consequently, as we go through every iteration, we approximate closer to the unique stabilizing solution, which we can call P. In general, we converge to that solution as follows: $P_1 \geq P_2 \geq ... \geq P$. This P is actually a solution to the DARE equation that we talked about before in equation (6). But, we just do not only converge to the unique stabilizing solution. With the unique stabilizing solution, we converge to an unique gain, which further gives us a unique or optimal control. But, most importantly we converge to the optimum performance critertion through each iteration: $J_0(x(k)) \geq J_1(x(k)) \geq J_2(x(k)) \geq ... \geq J_{opt}(x(k))$. So, in general as the unique stabilizing solution converges, so does the the unique stabilizing gain, the optimal control, and the most importantly the optimal performance.

Therefore, in an LQG problem, when finding optimal gains for both the Kalman Filter and the LQR systems, we have to do Hewer's Algorithm on both of these systems. So, we get two unique stabilizing solutions, which are solutions to two DAREs. In the problem, we are solving, we are not considering any noise, so we will just be solving an LQ optimal control problem, but using Hewer's Algorithm.

II. Problem Formulation

NOW the problem we have to solve at hand is in regards to the Boeing 747 flight control systems using Hewer's Algorithm. The aircraft flies at a level of 40,000 feet at a speed of 774 ft/s. Our system will consider the aircraft at discrete times intervals of 2 second. In general, the aicraft system can be represented as a linear-quadratic...

$$x(k+1) = Ax(k) + Bu(k) \quad (33)$$

where the matrices A and B are as follows with the following initial condition...

$$A = \begin{bmatrix} 0.99 & 0.03 & -0.02 & -0.32 \\ 0.01 & 0.47 & 4.7 & 0.0 \\ 0.02 & -0.06 & 0.4 & 0.0 \\ 0.01 & -0.04 & 0.72 & 0.99 \end{bmatrix},$$

$$B = \begin{bmatrix} 0.01 & 0.99 \\ -3.44 & 1.66 \\ -0.83 & 0.44 \\ -0.47 & 0.25 \end{bmatrix}.$$

$$x(t_0) = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

For our weighted Q and R matrices, they have to be symmetric for us to use Hewer's Algorithm. So let us define these weighted matrices as $Q = I_4$ and $R = I_2$. In general, we want to use Hewer's Algorithm to find the approximate feedback controls as well as the approximate system state trajectories obtained across ten iterations. Hewer's Algorithm allows us to find these approximate values after a certain amount of iterations because we start off with a worst approximation, and through each iteration, we find a find a better approximation. Although we can find a better approximation if we did more iterations, the problem we are looking at should require us to just use six to ten iterations to find an accurate approximation of the control and system state trajectory. Note that the Hewer's algorithm has a quadratic rate of convergence, so we should be able to find our answer within a couple of iterations. Also, since we are using Hewer's Algorithm, let us start off with an approximate gain value F_0. Since the system should be asymptotically stable, meaning that the system's reaches a stable state of 0 after some time, let us assume that we start of with the zero matrix for the gain. Once again the values that are we are trying to find are the optimal controls and optimal state trajectories after about ten iterations.

III. Results and Discussion

SO, considering our objectives of obtaining the optimal controls and optimal states after a certain amount of iterations, let us now talk about the results obtained.

Let us depict the system when we have an optimal gain as reference when we do Hewer's Algorithm. We can find the optimal gain, which can call $F_{optimal}$ which also gives us our optimal stabilizing solution, $P_{optimal}$ for our given problem using $dlqr(A, B, Q, R)$ in MATLAB. Once we find our optimal values, then we can find the state response of

our system. The state response of all stables variable matrix entries at each discrete time from x_1 to x_4 are depicted in Fig. 1. But, to make sure that our results with the optimal value are correct, let us also simulate the state response.

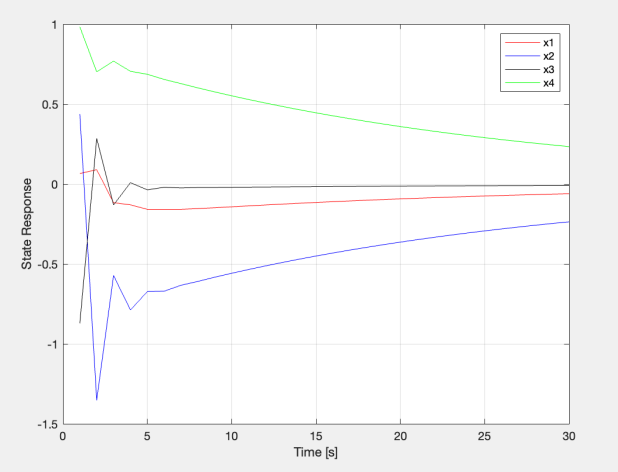

Fig. 1. State Response with $F_{optimal}$ and $P_{optimal}$ solution. Along thirty iterations.

We can model the LQ system as depicted in Fig. 2, and simulate it across the same amount of iterations as in Fig. 1, which is thirty. So, when simulating this model, we get the following step response in Fig. 3. Note that the step response in Fig. 3 also depicts the states at $k = 0$, while Fig. 2, does not, as it starts from $k = 1$. However, besides that discrepancy, Fig. 1 and Fig. 2 have the same values, so we can determine that we have the step response at the ideal gain and solution.

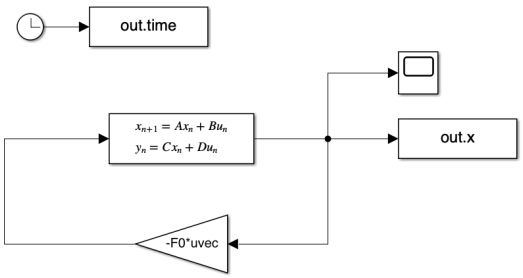

Fig. 2. Block Diagram model of Linear Quadratic system

Now, let us look at Hewer's Algorithm, where we have stabilizing solutions, P_k at every iteration where $k = 1, 2, 3, 4, 5$ and so on when solving the corresponding Lyapunov equation. As a result, because of these stabilizing solutions at each iteration, we also get the respective gains at each iteration which we can refer to as F_k. We will talk about the exact values of the stabilizing solutions to the Lyapunov equations and the gains later on, but let us consider the controls first. Since there is a direct relationship between the gain and control, we can find the corresponding controls at each iteration. Let us say we

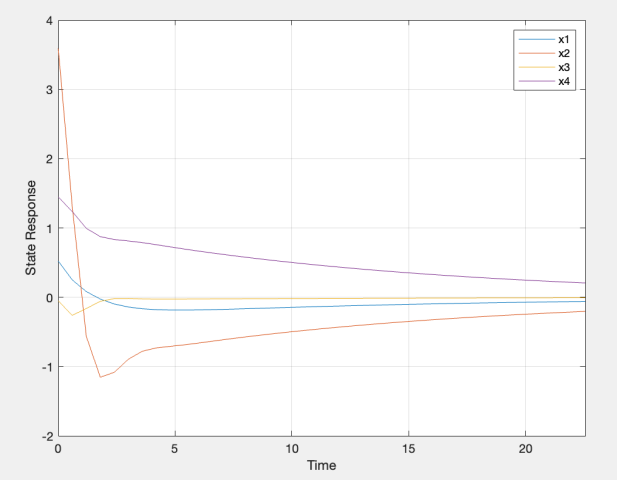

Fig. 3. Step Response Simulation. Starts from $k = 0$, where initial conditions are depicted.

do ten iterations using Hewer's Algorithm to find the control inputs.

Iteration	u_1	u_2
1	2.0314	-0.6992
2	1.4199	-1.1251
3	1.2260	-0.8900
4	1.1342	-0.7535
5	1.1089	-0.7143
6	1.1066	-0.7108
7	1.1066	-0.7108
8	1.1066	-0.7108
9	1.1066	-0.7108
10	1.1066	-0.7108

Fig. 4. Controls u_1 and u_2 after 10 iterations

As you can see, in Fig. 4, the controls converge to $u_1 = 1.1066$ and $u_2 = -0.7108$. As a result, we can say that the optimal control would be at the last iteration, which is the values that we converge to. In this case, however, the optimal control seems to have been reached at iteration six. So, in reality we only needed six iterations to get the optimal control. We can see the controls at Fig. 5 converging to the respective values we mentioned before. Besides, the optimal control we obtain, we still have to consider the state variables and most importantly the optimal performance criterion that we get to.

Now, let us consider the optimal state trajectories that we are to get after going through five iterations of Hewer's Algorithm. Since we get five different gains at each of the iterations, and as a result of that five different controls, then we should get five different system trajectories for the state response. Fig. 6 through Fig. 10 depict the state response at these different iterations, and you can see that the graphs converge to their approximate trajectories.

However, we also want to see how the individual elements

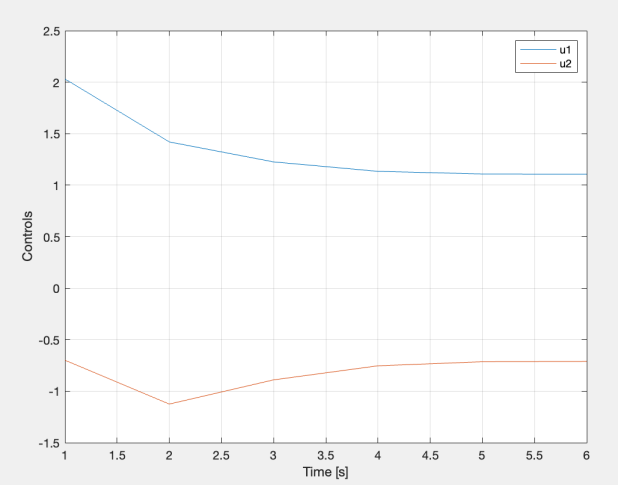

Fig. 5. Controls, u_1 and u_2 at each iteration at k = 1, 2, 3, 4, 5, 6.

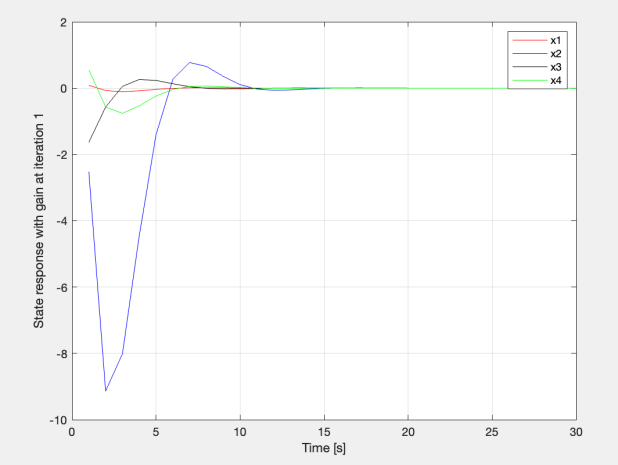

Fig. 6. State Response at gain F_1.

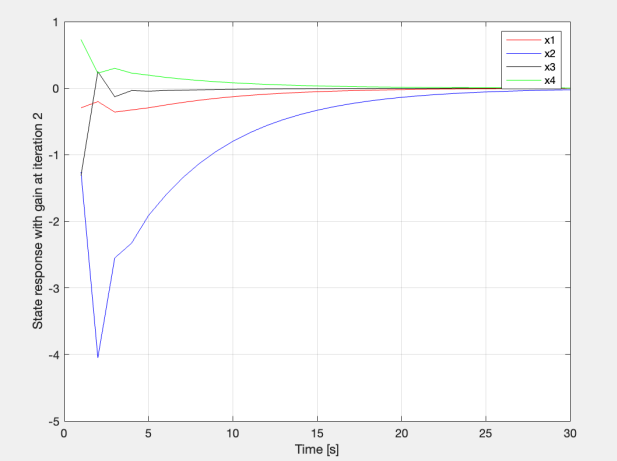

Fig. 7. State Response at gain F_2.

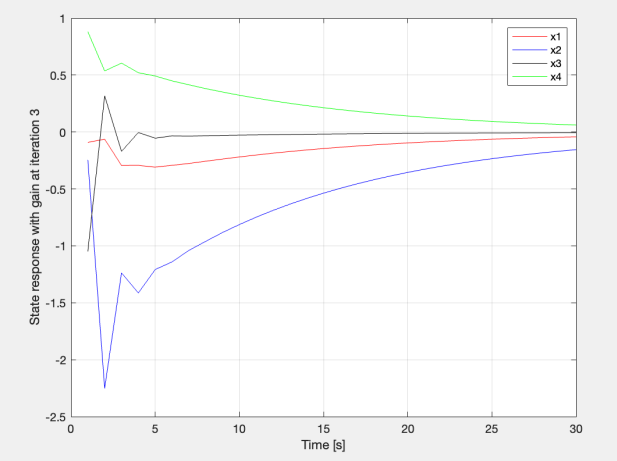

Fig. 8. State Response at gain F_3.

of the state, x_1, x_2, x_3, and x_4, converge to their optimal trajectories at different iterations. Let us say that we want to find the state trajectories after five iterations. So, we will have five different trajectories for one state variable, as in each iteration we have a different gain value. Fig. 11 - Fig. 14 depict these state trajectories respective to the state variables x_1 through x_4 We want to find the best state response at a certain iteration that closely resembles the optimal trajectories that we found in Fig. 1.

Considering the state response of x_1 in Fig. 11, we see that the best trajectory out of the five iterations, is at iteration four. It is the gain, F_4 and more specifically the solution P_4 that get us this solutions. We see that trajectory of x_1 at iteration 4 closely resembles the trajectory of x_1 in Fig. 1, which is the optimal. Likewise, for the state response of x_2, the trajectory at the fifth iteration is the best. For state response x_3, it is the fourth iteration. And for the state response x_4 it is the third iteration that give us the best trajectory that resembles the optimal trajectory depicted in Fig. 1. However, let us determine, which state response has the worst convergence. More specifically, let us explore which state variable has the worst performance. To determine the worst performance, we have to see which state variable has the most distance between its different trajectories. Consequently, once we compare all of the graphs from Fig. 11 - Fig. 14, we see that x_1 performs the worst out of all the state variables. While the other state variables' trajectories seem to converge quickly within the five iterations we designated, the trajectories at x_1 seem to converge slowly. Although we have an optimal trajectory out of the five iterations, it seems like we could do better with the convergence if we ran more iterations, specifically for x_1. But, this conclusion entails that we would have to run more iterations for every state variable, as running more iterations for one state variable element means that you have to run iterations for the rest of the elements in the matrix.

Now finally, let us consider the most important value we care about, which is the performance criterion or an equivalent term, the reward, in reinforcement learning. As you can see in Fig. 15, the performance criterion values seem to converge between 10 and 15. It is hard to determine with the scale of the graph, so let us look at Fig. 16 where we actually have

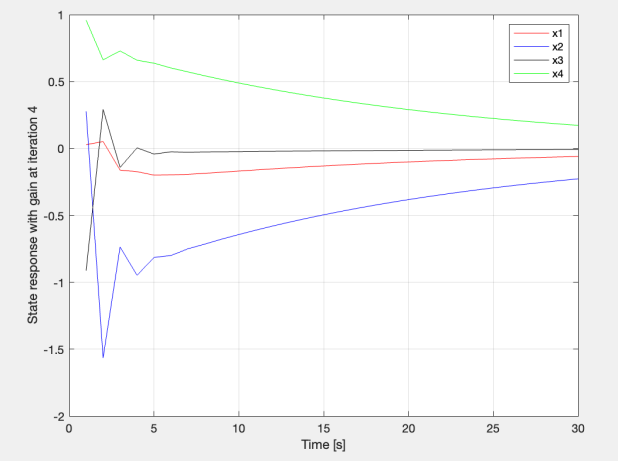

Fig. 9. State Response at gain F_4.

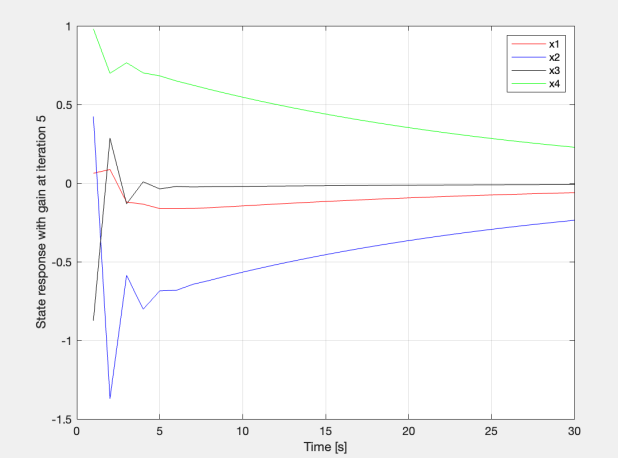

Fig. 10. State Response at gain F_5.

the values of the performance criterion. As you can see the performance criterion seems to be converging to 13.1696. If we were to do more that ten iterations, most likely our optimal performance criterion would hover around the value we obtained at the tenth iteration. As a result, $J_{opt} = 13.1716$ after using Hewer's Algorithm for ten iterations. Also, it seems we reached our optimal performance at iteration seven, so it seems that we also could do less than ten iterations to achieve optimum performance.

Besides, the state feedback values, let us look at at the stabilizing solutions, P, during five iterations when we solve for Lyapunov equations. Note that finding these values in the first place help us find the state trajectories and performance; Specifically the values we obtain for P for every iteration help us get a better gain value, which in turn helps us achieve a more optimal performance. In general, let's look at the values that we obtain at the fifth iteration, which are P_5 and F_5.

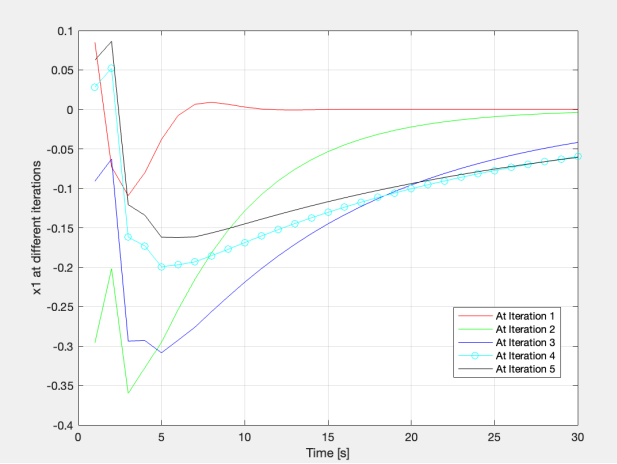

Fig. 11. State Response x_1 at different gains. The dotted line is the best trajectory. Performs the worst out of the rest of the state variables.

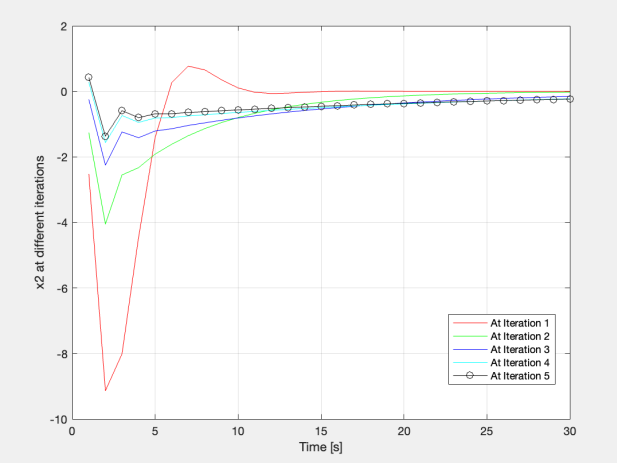

Fig. 12. State Response x_2 at different gains. The dotted line is the best trajectory.

$$P_5 = \begin{bmatrix} 1.8345 & 0.0247 & -0.3138 & -0.3924 \\ 0.0247 & 1.3462 & 0.3458 & -2.3602 \\ -0.3138 & -0.3458 & 3.9766 & 1.1263 \\ -0.3924 & -2.3602 & 1.1263 & 22.7944 \end{bmatrix}$$

$$F_5 = \begin{bmatrix} 0.2936 & -0.0429 & -1.0448 & -0.3149 \\ 0.5961 & 0.0455 & 0.2612 & -0.1885 \end{bmatrix}$$

As you can see, both the stabilizing solutions and the stabilizing gains converge to specific values after five. But, note that if we were to do more than five iterations, we would get matrices that change in value but hover around the fifth iteration. We should realize that doing more iterations might help us converge to the optimal solution and optimal performance, which could help getting a more accurate performance for the state variable x_1.

IV. Conclusion

In general, using Hewer's Algorithm seems like a great tool to find the optimal values of an LQR problem. In general, considering the computational power and difficulty needed to solve non-linear algebraic Ricatti equations, we needed a better

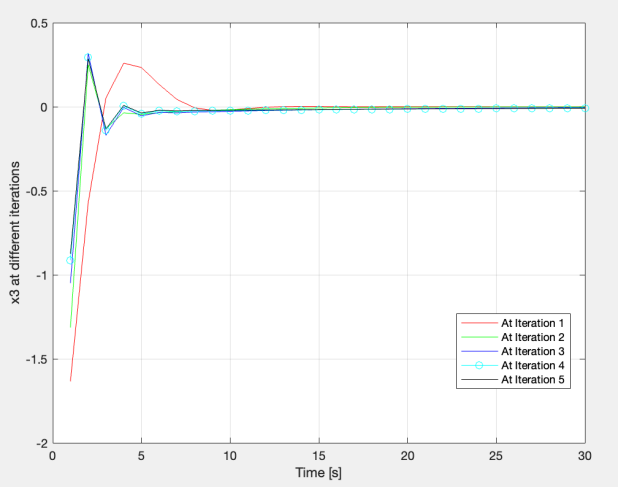

Fig. 13. State Response x_3 at different gains. The dotted line is the best trajectory.

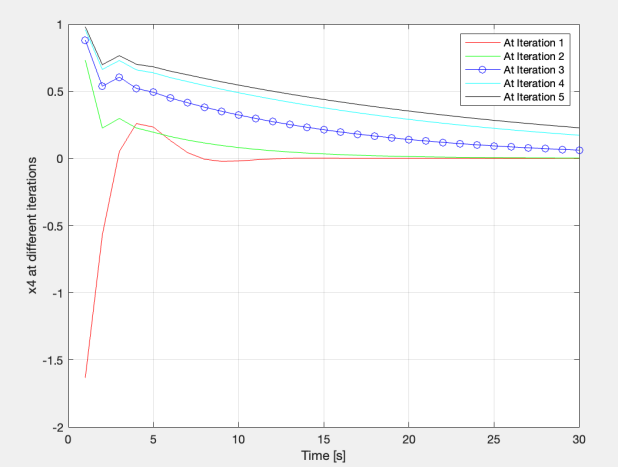

Fig. 14. State Response x_4 at different gains. The dotted line is the best trajectory.

way to find the optimal P without solving these complex equations. Consequently, that is where Hewer's Algorithm comes from, so we can avoid these problems. Instead, we do feedback iterations where we solve Lyapunov equations to find the approximate P, which we use to find the approximate feedback gain to find the approximate control. And these approximate controls can be used to find the approximate state trajectories and the optimum performance criterion. From the scenario we analyzed, we used five to six iterations of Hewer's Algorithm to find the optimal state trajectories and optimal control. We also noted that we could do more iterations to get a more precise optimal results. For example, for the state trajectories, we could do more iterations to get better state trajectories for x_1. But, for the most part, for other values, such as the performance criterion, the more and iterations we do, the optimal values will hover around a certain value. It is just an exercise to determine how many iterations we need to obtain an optimal performance. Therefore, for us to obtain optimal performance in our scenario, we would want to do actually less than ten iterations. Most likely we should do

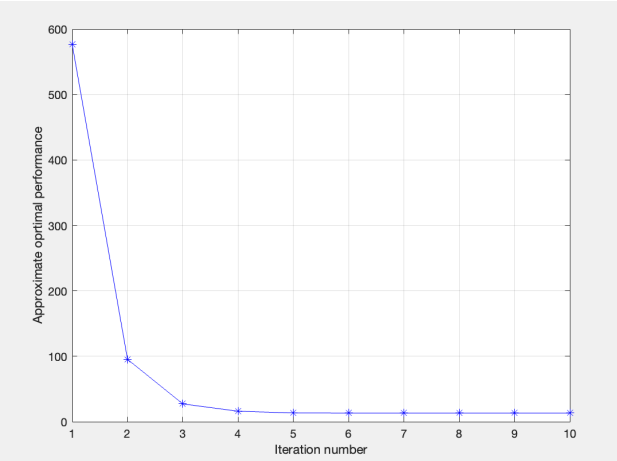

Fig. 15. Performance Criterion convergence after ten iterations.

Iteration	J_{opt}
1	576.4004
2	95.2524
3	27.2344
4	16.0064
5	13.4063
6	13.1716
7	13.1696
8	13.1696
9	13.1696
10	13.1696

Fig. 16. Performance Criterion values after ten iterations.

seven iterations if we go off of Fig. 16. In general, considering the computational intensity of solving certain equations using the simple principles of a Linear Quadratic Regulator problem in discrete time, we need algorithms that reduce this computational difficulty to solve for stabilizing values. In our case, Hewer's Algorithm offers just that solution. We solve for simpler Lyapunov equations as we go through more and more iterations. Hewer's Algorithm offers us the ability to reduce the amount of computation needed to solve for non-linear DAREs, which is significant within itself because we can obtain optimal results using simpler mathematics over successive iterations.

REFERENCES

[1] G.Hewer, "An iterative technique for the computation of the steady state gains for the discrete optimal regulator," *IEEE Transactions on Automatic Control*, Vol. AC-16, 382-384, 1971.

[2] G.Hewer, "Analysis of a discrete matrix Riccati equation of linear control and Kalman filter," *Journal of Mathematical Analysis and Applications,*, Vol. 42, 226-236, 1973.

[3] G. Goel and B. Hassibi, "Competitive Control," *IEEE Transactions on Automatic Control*, Vol. 68, 3162-3173, 2023.

[4] P. Reddy "HewersAlgorithm" *GitHub*, 04 Nov. 2024 https://github.com/pdvr99/HewersAlgorithm