Optimal Control in Reducing Rotor Vibrations*

Sami KIVILUOTO**, Ying WU**, Kai ZENGER** and Xiao-Zhi GAO**

**Aalto University, School of Electrical Engineering, Department of Automation and Systems Technology, Espoo, Finland
E-mail: sami.kiviluoto@tkk.fi, yiwu@cc.hut.fi, kai.zenger@aalto.fi, gao@cc.hut.fi

Abstract
This paper presents three different LQ controllers, which are used to attenuate rotor vibrations in an electrical motor. The controllers are compared in performance and control effort. A method of solving the related Riccati equation is discussed to overcome the issue of finding the solution, when the model including disturbance dynamics is not stabilizable. One LQ controller is calculated utilizing particle swarm optimization, and the benefits of doing so are evaluated with respect to the cases when no optimization was used in tuning the controller.

Key words: Optimal Control, Rotor Vibration, Electrical Motor, LQ Controller, Particle Swarm Optimization

1. Introduction

Vibrations are present in practically all rotating machines. They have been studied quite extensively e.g. in helicopters due to their direct influence on the experience of pilots and passengers. Used designs for attenuating the vibrations include pole-placement (1), combining optimal control and Kalman filters (2) and the LQ approach (3).

In electrical machines vibrations are equally a well-known problem, which shortens the lifespan of the motor by introducing a mechanical stress on the machine parts, setting limits to motor usage. Vibrations are particularly harmful at the critical frequency of the system. At this operation point energy brought to the system amplifies the vibrations until the supporting structures give in. For these reasons it is beneficial to use control, which minimizes the vibrations.

This paper presents an optimal controller that minimizes a cost function measuring rotor vibrations. The controller is designed using a linearized state-space representation based on a nonlinear finite element (FE) model, which was identified for a real cage induction motor. In addition to the dynamics of rotor shaft, the linearized model includes a special actuator built into the motor. The optimal state feedback controller is obtained by solving the LQ optimal control problem based on knowledge of the system and additionally by utilizing particle swarm optimization, which is a stochastic algorithm. The purpose is to find the most beneficial controller tuning method and simultaneously to assess the capabilities of particle swarm optimization in a practical engineering problem.

In linear quadratic (LQ) approach, the effect of disturbance is minimized either in the model output (displacement of the rotor) or in the total force acting on the rotor. The total force is considered on one hand as the sum of disturbance and control force and on the other hand as the excitation of the rotor dynamics in the model. The fact that the process is not
stabilizable because of sinusoidal disturbance is of concern, because the algebraic Riccati equation is not solvable in that case. The problem is discussed using two different methods: 1) Slightly modifying the disturbance model and 2) Solving the Riccati equation as a finite time dynamical problem.

Particle swarm optimization (PSO) is presented as an alternative method to tune the parameters in the LQ problem. It is a population-based optimization strategy, which is inspired by social behaviors, such as bird flocking and fish schooling. Parameters are related to “particles” in the PSO, and each particle can fly in the search domain according to its changing velocity. Each particle is associated with a position, which stands for the possible solution to the problem under consideration. Previous experience and the present best particle are the guiding rules for updating the particles. In this paper, the weight matrix elements for the states and control outputs are formulated as particles. Firstly, the search range is defined, and the position and velocity of the particle population are randomly initialized. Secondly, the position and velocity of these particles are updated in each step of iteration until the particle with the best position is found so that the cost function can be minimized.

Analysis of the controller design and simulations are presented, and the control results are verified and compared. The focus is on operation at the critical frequency of the machine and in near frequency regions.

2. Model formulation

The model of the motor consists of three parts: actuator dynamics, rotor dynamics and the disturbance force. A diagram of the system is presented in Fig.1.

The actuator model and the rotor model are linearized state-space representations, which have been identified from a nonlinear FE model using the prediction error method. The feedback presented in the figure is inherent to the actuator since the displacement of rotor affects the air gap between rotor and stator. The smaller the air gap the stronger is the force produced by the actuator winding.
The combined model can be written in the state-space form (6):

\[
\begin{bmatrix}
\dot{x}(t) \\
y(t)
\end{bmatrix} =
\begin{bmatrix}
A_r & B_r C_a & B_r C_d \\
B_{a2} C_r & A_a & 0 \\
0 & 0 & A_d \\
\end{bmatrix}
\begin{bmatrix}
x(t) \\
u(t)
\end{bmatrix}
\begin{bmatrix}
0 \\
B_{a1}
\end{bmatrix}
\]

(1)

where

\[
x(t) = \begin{bmatrix} x_r(t) \\ x_a(t) \\ x_d(t) \end{bmatrix}, \quad u(t) = u_a(t), \quad y(t) = y_r(t)
\]

The subscripts \(r\), \(a\) and \(d\) refer to the state-space models of the rotor, actuator and disturbance. Thus the matrices \(A_r, B_r, C_r\) come from the state-space model describing the identified rotor dynamics and \(A_a, B_{a1}, B_{a2}\) and \(C_a\) are the matrices of the identified actuator model.

\[
\begin{bmatrix}
\dot{x}_a(t) \\
y_a(t)
\end{bmatrix} =
\begin{bmatrix}
A_a & 0 & 0 \\
B_{a2} & A_{a2} & 0 \\
0 & 0 & A_d \\
\end{bmatrix}
\begin{bmatrix}
x_a(t) \\ u_a(t) \\ y_a(t)
\end{bmatrix}
\]

(2)

where \(u_a(t)\) is the control voltage of the actuator, \(y_r(t)\) is the rotor displacement and \(y_a(t)\) is the control force produced by the actuator. \(A_d\) and \(C_d\) in the combined model Eq. (1) are the matrices of a disturbance model giving sinusoidal force as output. The dimension of the total model becomes 18. Four states describe the rotor dynamics in x and y directions, ten states describe the actuator dynamics and four last states form the disturbance as shown in the model.

An LQ controller is a state-feedback by design, but it is not possible to measure all states of the system in reality. Therefore a state observer Eq. (3) for the model is calculated as described in (6):

\[
\begin{bmatrix}
\dot{x}(t) \\
\dot{y}(t)
\end{bmatrix} =
\begin{bmatrix}
A - KC & Bu(t) + Ky(t) \\
B^T u(t) + K^T y(t)
\end{bmatrix}
\]

(3)

and the estimation error dynamics is:

\[
\dot{\hat{x}}(t) = (A - KC) \hat{x}(t) + (A^T - C^T K^T) \hat{y}(t).
\]

(4)

Matrices \(A, B\) and \(C\) are the matrices of the complete model and \(K\) defines the state-feedback from the measured states that are used to update the observer. By considering a general state-space representation in Eq. (5) describing a system controlled by a state-feedback controller, it is obvious that the estimation gain in Eq. (4) can be solved by an analogous control problem. For example the LQ approach can be used to that end.
\[
\begin{align*}
\dot{x}(t) &= (A - BL)x(t) \\
y(t) &= Cx(t)
\end{align*}
\] (5)

With an observer described by Eq. (3), it is possible to implement an LQ controller with the knowledge of rotor displacement and control signals only.

3. Control design

3.1. LQ controller

Considering the following system

\[
\begin{align*}
\dot{x}(t) &= Ax(t) + Bu(t) \\
y(t) &= Cx(t)
\end{align*}
\] (6)

the LQ controller is a state feedback

\[
u(t) = -Lx(t)
\] (7)

and the problem is to find \(L\) that minimizes the cost function:

\[
J = \frac{1}{2} \int_{0}^{t_f} \left( x(t)^T S(t) x(t) + \frac{1}{2} \int_{0}^{t_f} \left( (v(t))' Q v(t) + u(t)' R u(t) \right) dt \right)
\] (8)

The optimal feedback gain \(L\) is given by \(L = R^{-1}B^T S\), which requires solving the Riccati equation:

\[
A^T S(t) + S(t) A - S(t)BR^{-1}B^T S(t) + v^T Q v = -\dot{S}(t).
\] (9)

If \(t_f\) approaches infinity, the cost will become

\[
J = \int_{0}^{\infty} \left( (v(t))' Q v(t) + u(t)' R u(t) \right) dt
\] (10)

The derivative of \(S\) in Eq. (9) is then zero and the equation simplifies to the algebraic Riccati equation

\[
A^T S + SA - SB R^{-1} B^T S + v^T Q v = 0
\] (11)

In the application to be considered the matrix \(R\) is an identity matrix. Matrix \(Q\) is a two by two matrix (with large weight values, e.g. \(10^8\)) since the disturbance is modeled in x and y directions. Vector \(v\) is selected according to what is minimized. To compare different approaches, it is chosen to minimize: 1) Augmented states that are formed by integrating the sum of control force and disturbance over time in x and y directions given by Eq. (12). 2) The rotor displacement. 3) The input states of the rotor system. The augmented states added to the combined model are defined as
When the augmented states are minimized, the sum of actuator and disturbance forces is driven towards zero over time. Then \( \nu \) will be selected as:

\[
\nu = \begin{bmatrix}
0 & 0 & 0 & \ldots & 1 & 0 \\
0 & 0 & 0 & \ldots & 0 & 1 \\
\end{bmatrix}
\]

This selection results in a control method later called LQR\(_{\text{aug}}\). The height of \( \nu \) is two, again due to modeling the rotor displacement in \( x \) and \( y \) directions. The width is 20, which is the dimension of the model including the two augmented states. In case of LQ controller that minimizes the displacement of the rotor, \( \nu = C \) so that the cost minimizes displacement of the rotor in both directions. Selecting \( \nu = C \) gives a controller called LQR\(_y\) later in this paper. The vector \( \nu \) that minimizes the input states of the rotor is

\[
\nu = \begin{bmatrix}
1 & 0 & 0 & \ldots & 0 & 0 \\
0 & 0 & 1 & \ldots & 0 & 0 \\
\end{bmatrix}
\]

Augmentation is not required when \( \nu \) selects the states of the rotor model that are influenced by the forces (i.e. cause the displacement). So the width is 18. This method is from now on called LQR\(_u\).

The problem in calculating the optimal feedback for this system is that the model includes the sinusoidal disturbance, and the related dynamics is not controllable (the whole system is not stabilizable). Therefore the Schur-type method \(^{(7)}\), used by MATLAB for example, cannot be used to solve the Riccati equation. One way to overcome this issue is to add a small damping to the disturbance, which moves the corresponding poles to the open left-hand plane from imaginary axis. The damping doesn’t distort the results if it is set to be small enough so that it doesn’t affect the rotor displacement significantly.

### 3.2. Simulations

The observer and controllers are calculated at 49.5 Hz, which was the operating frequency when data was collected for identification of the model parts. Fig.2 shows the displacements in time domain when controller is switched on after two seconds. Fig.3 in turn shows the amplitude of the rotor in frequency domain when the disturbance frequency is altered. Attenuation of the vibration around the disturbance frequency under focus is displayed more closely in Fig.4. It is clear that all LQ regulators significantly reduce the amplitude of the rotor vibration at the disturbance frequency under focus. The peak around 40 Hz is higher with augmented states and lowest with LQR\(_u\), although LQR\(_u\) does have slightly less damping on the lowest and the highest frequencies. All methods provide similar results when the disturbance frequency increases. In general LQR\(_u\) is the best regulator of these three when the highest amplitudes are considered.
Fig. 2. The rotor displacement when LQ controller is switched on at $t = 2$ s.

Fig. 3. Amplitude of the rotor vibration in frequency domain. Control is not needed below 30 Hz (vertical line).

Fig. 4. Attenuation near the frequency for which the observer and controller were designed.
Control signals are also an important factor in comparison of the control results. In Figs. 5 and 6 the signal levels are displayed when the controller is switched on to illustrate the range of control signals, and during a shorter period that shows the signal values when the operation is steady.

![Fig.5 Full control signals by the three controllers (blue: x direction, green: y direction)](image1)

![Fig.6 Control signals in steady operation (blue: x direction, green: y direction)](image2)

The control voltage with LQR_y is significantly lower than with the other controllers. This would allow setting the weights in $Q$ even higher so that the control result would be closer to LQR_u without higher amplitudes at lowest frequencies. This, however, would result in an unfair comparison since the range of control voltage ($\pm14$ V) would allow setting the weights even higher for the other controllers as well.

### 3.3. Solving the Riccati equation dynamically

Solving the Riccati equation Eq. (9) with final condition $S(t_f)$ is equal to solving Eq. (13) with initial conditions $S(0) = S(t)$. By doing so it is possible to start from zero-state and calculate the matrix differential equation forward and let it converge to its solution $^{(9)}$. 
\[ A^T S(t) + S(t) A - S(t) B R^{-1} B^T S(t) + \nu^T Q \nu = \dot{S}(t) \]  

(13)

As the time span gets longer, the result will seem as if the element were constant throughout the period. The steady state value of \( S \) can then be considered as a possible solution of the algebraic Riccati equation Eq. (11). Furthermore, the candidate has to be positive-definite, which is easy to verify by calculating the eigenvalues of \( S \).

Figure 7 illustrates the amplitude of rotor vibration at different frequencies when the Riccati equation is solved either dynamically or by adding an artificial damping to the disturbance model. There is practically no difference between the two. Even so, solving the Riccati equation dynamically allows us to keep the model unmodified and the solution fulfills Eq. (11) much more accurately (asymptotic behavior) than resorting to artificial damping, which provides a solution that does not satisfy the equation as well. The difference in controller performance is not noticeable, but the fact nonetheless removes the benefit of the doubt giving stronger foundation for mathematical analysis.

![Fig. 7 Control results with LQ controller given by the Schur method and by calculating the Riccati equation dynamically](image)

Fig. 7 Control results with LQ controller given by the Schur method and by calculating the Riccati equation dynamically

Figures 8 and 9 show again the control signals, which turn out to be similar to what was presented earlier. (The roughness of the control signal appears to be a numerical problem in the simulation because all states individually look smooth like the produced force.)
The control signal of LQR_y is clearly much smaller than the other two controllers. The reason is the phase difference of control signal and disturbance (Fig.10). With LQR_y the difference is 180 degrees, which is ideal for cancelling the disturbance. The other control signals are quite close to the disturbance in phase and therefore require more effort. The major phase difference doesn’t cause instability because the control signals don’t describe forces directly. For LQR_{aug} and LQR_u there is transient when the control is switched on (see e.g. Fig.8). After the transient has passed, the control forces and control signals are not in the same phase. The couplings between x and y directions in the actuator model can lead the system to this kind of state.
4. Particle Swarm Optimization

When computing an LQ controller, it is assumed that the designer knows how to set the weights for the states and controls. Obviously, selecting the weights differently results in different performance. Section 3 has presented controllers that are based on knowledge of how the vibrations manifest themselves in the system. However, there may be better ways to tune the controller, but it is impractical to investigate this by repeating the process manually. A method called Particle Swarm Optimization (PSO) was chosen to perform this task for the purpose of studying its potential for parameter optimization in our research.

The Particle Swarm Optimization (PSO) is a population-based optimization strategy, which is inspired by social behaviors, such as bird flocking and fish schooling. It was firstly developed by Kennedy and Eberhart in 1995 (9), and has become one of the most efficient techniques for solving difficult optimization problems. The distinguishing features of the PSO are its computational efficiency and algorithm simplicity.

Suppose there are \( n \) particles in the PSO, which are initialized randomly, and each particle can fly in the search domain (\( D \) dimensions) according to its own velocity \( V_i = (v_{i1}, v_{i2}, \ldots, v_{id}) \). The particle is associated with a position \( X_i = (x_{i1}, x_{i2}, \ldots, x_{id}) \), which stands for the possible solution to the problem under consideration. During the iterations, each particle can update the position on the basis of its own previous best position \( P_i = (p_{i1}, p_{i2}, \ldots, p_{id}) \) and the global best position \( P_{gd} \) of the whole swarm. The position update of these particles is as follows:

\[
\begin{align*}
V_{id}^{t+1} &= \lambda(wV_{id}^t + c_1r_1(p_{id} - x_{id}^t) + c_2r_2(P_{gd} - x_{id}^t)) \\
x_{id}^{t+1} &= x_{id}^t + V_{id}^{t+1}
\end{align*}
\]

where \( w \) is the inertial factor, which is usually chosen to linearly decrease from 0.9 to 0.4. Its function is to balance the local and global abilities of the PSO, since large \( w \) makes more contributions to global search (10); \( c_1 \) and \( c_2 \) are the learning factors, through which particles can share the information with each other; \( r_1 \) and \( r_2 \) are random numbers uniformly...
distributed over [0, 1]; λ is a constriction factor used to limit the maximum velocity value given by Eq. (11). The ranges of $v_{id}$ and $x_{id}$ are defined as $[-v_{max}, v_{max}]$ and $[x_{min}, x_{max}]$ respectively, and they can be set according to the domain knowledge.

### 4.1. Optimization of LQ controller using PSO algorithm

In this section, PSO algorithm is used to determine the optimized $Q$ and $R$ matrices to minimize the new cost function $J_{opt}$ similar to Eq. (10):

$$J_{opt} = \int_0^T (y_{opt}(\tau)^T Q_{opt} y_{opt}(\tau)) d\tau$$

where $Q_{opt}$ is chosen as diag([1,1]). When using PSO, the matrices $Q$ and $R$ in Eq. (10) are chosen as diagonal matrices with non-negative real elements. The size of $Q$ matches the dimension of the system and states are not selected using vector $v$. Instead this task is given to the PSO algorithm in form of setting the elements in $Q$ (note that the elements $r_1$ and $r_2$ here are not the same as in Eq. (14)):

$$Q = \text{diag}(q_1, q_2, \ldots, q_{16}), \quad R = \text{diag}(r_1, r_5)$$

The disturbance model included in the system is problematic, as the sine-wave is a periodic signal that never converges. One way to deal with this is to set the weights for the states in question to zero, that is:

$$q_{15} = q_{16} = q_{17} = q_{18} = 0.$$

So there are 16 undefined elements (range set from one to $10^6$) left in $Q$ and $R$ that form particles in the PSO algorithm. The flow chart of the PSO algorithm applied is shown in Fig.11.
The steps of PSO algorithm can be explained as follows:
1. Select the size of the population $N$, and set the dimension and the upper and lower bounds of the search space.
2. Initialize the particles with random initial positions and velocities.
3. Solve the LQ problem and define the cost function in Eq. (16) as the fitness function in PSO. Compute the fitness value for each particle, so that the best position of the $i^{th}$ particle and the best position of the whole swarm can be found.
4. Update the particles according to Eqs. (14) and (15).
5. Update $p_{id}$ and $P_{gd}$:
   \[\text{if } \text{fitness}(X_i) < \text{fitness}(P_i), \text{ then } P_i = X_i\]  
   \[\text{if } \text{fitness}(P_i) < \text{fitness}(P_{gd}), \text{ then } P_{gd} = P_i\]  
6. Go back to step 4 until a criterion is met.

4.2. Simulation results

The parameters of the PSO algorithm are chosen as $N = 20$, $c_1 = c_2 = 2.05$, $w$ decays linearly from 0.9 to 0.2, $\lambda = 0.729$, $x_{min} = 1$, $x_{max} = 1e+8$, $v_{max} = 5e+7$.

As shown in Fig.12, the PSO algorithm has found the optimized $Q$ and $R$ matrices during 30 iterations, and then the corresponding controller can be computed through Eqs. (7), (11) and (16).

The simulation results show that it is convenient to get an optimized LQ controller using PSO. The comparison with the other presented controllers in Fig.13 indicates that the PSO-optimized controller is almost as good as LQR, in general and even better at the lower frequencies. When the control signal of PSO is compared with the control signals of other methods (Fig.14), it is apparent that it is not larger than with LQR-mg or LQR-u. Therefore the performance can be regarded as very good.
Fig. 13 PSO-optimized LQ controller compared with other LQ controllers. Controller is not necessary below 30 Hz (vertical line).

Fig. 14 The control signal levels including controlled obtained with PSO (blue: x direction, green: y direction).

5. Conclusions

Controlling the displacement of rotor in electrical machine is possible using an LQ controller, which minimizes augmented states (LQR_{aug}), rotor displacement directly (LQR_y), or the input states of the rotor system (LQR_u). The most efficient damping is given in minimizing the force acting on the rotor, but with a trade-off of poorer control result on the lowest frequencies and lower robustness against noise. It should be noted that each controller could be further analyzed in regard to how weights in Q and R change the damping curves as a function of disturbance frequency.
Additionally, the task of minimizing the cost function can be given to an optimization algorithm. The PSO-optimized LQ controller gives nearly as good a result as LQR u without the trade-off. The PSO is promising since it already gives a good result, although there are further modifications to be done to the algorithm to make it better suitable for optimization tasks with higher number of parameters. The modifications will on one hand allow more freedom to the algorithm and on the other hand will give the possibility to optimize the feedback gains directly.

The conclusion based on the simulations is that controlling rotor vibrations by minimizing the output (displacement of the rotor) is the best choice for practical implementations because it is effective and always feasible. PSO performed well with the given optimization task, which is a positive sign considering its use in more challenging parameter optimization problems. In fact, the results presented in this paper have been utilized later to implement an LQ controller to reduce rotor vibrations in an electric motor. The parametric electromechanical model in that implementation was identified using PSO and its derivatives. Practical tests carried out in real process environment have shown good performance.

Acknowledgements

The research has been funded by Academy of Finland and supported by Tekes– the Finnish Funding Agency for Technology.

Ying Wu's work was funded by the National Natural Science Foundation of China (NSFC) under Grant No. 60874084. Xiao-Zhi Gao's work was also funded by the Academy of Finland under Grant 135225.

References