CLASSICAL MECHANICS-2 TERM PAPER

ANALYTICAL MECHANICS IN OPTIMAL CONTROL THEORY

Shobhit Saheb Dey
Guided By - Prof. Tirtha Sankar Ray
Department of Physics
Indian Institute of Technology Kharagpur

ABSTRACT

The Lagrangian, Hamiltonian and Hamilton-Jacobi equations of motion evolve from the optimization of the action over time. Optimal control policies also stem from different methods of optimization of cost over trajectory with different cost functions used for different scenarios. So to optimally control a system, with known dynamics, an optimal control policy is designed through equations similar to Hamiltonian equations(Pontryagin's principle). Linear Quadratic Regulator is derived through this principle also coded and tested to control an autonomously driven car on a simulator.

INTRODUCTION

The Lagrangian formulation of dynamics based on principle of extremal action gives the Euler-Lagrange equation of motion. But the Lagrangian itself(whose time integral is to be extremized) has to be constructed for different physical frameworks(Newtonian, special relativistic, quantum field theoretic etc.) based on their fundamental requirements(certain covariance, algebraic form of quantities, normalizations etc.). Once a suitable Lagrangian is constructed for a certain class of canonical phase spaces, the trajectories extremizing the action could be found by either Euler-Lagrange equation, Hamilton's equations or Hamilton-Jacobi equation. In the engineering problems relevant in innumerable areas like robotics, aerospace, automation, electronics, photonics etc. a common problem statement occurs: i.e actuating the system to a desired state $\mathbf{x}$ in configuration space optimally and keeping it stable there in spite of external disturbances. The metric of optimality in controls is a Lagrangian that penalizes the system for being away from the target state and the amount/force/energy of actuation given. The cost function is the time integral of this Lagrangian. Penalizing the system for being away from target state is trivial to interpret given the ultimate reason of deploying a control system in first place. But consider a cruise control in car; if it had near to infinite acceleration and retardation capabilities then it could reach any desired speed within no time and the integral(cost function) would be 0. Hence due to system constraints, power bounds and fuel economy the control system has to trade off between error(desired state - current state) and actuation. Adhering to these, we construct a suitable Lagrangian for the Linear Quadratic Regulator(LQR). Before that(similar to classical mechanics) we first derive the control policy for an arbitrary Lagrangian analogous to Hamilton's equations of motion.

Variational optimization of cost function

If $L(\mathbf{x}, \dot{\mathbf{x}}, \mathbf{u}, t)$ is the Lagrangian and the dynamics of the system is governed by the equation $\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u}, t)$ then this equation acts as a **constraint** for the optimization problem. Here $\mathbf{x}$ is the n-dimensional state and $\mathbf{u}$ is the m-dimensional actuation vector representing m actuators powering and controlling the system. We introduce n Lagrange multipliers comprising a **costate vector** represented as $\mathbf{p}(t)$. So

the problem reduces to minimizing the following cost function:

$$J = \int_{t_0}^{\infty} (L(\mathbf{x}, \dot{\mathbf{x}}, \mathbf{u}, t) + \mathbf{p}(t)^T (f(\mathbf{x}, \mathbf{u}, t) - \dot{\mathbf{x}}))\, dt \tag{1}$$

Defining the quantity **Hamiltonian** inspired from mechanics as:

$$H = \mathbf{p}(t)^T \dot{\mathbf{x}} - L \tag{2}$$

Then the cost function becomes:

$$J = \int_{t_0}^{\infty} (\mathbf{p}(t)^T f(\mathbf{x}, \mathbf{u}, t) - H)\, dt \tag{3}$$

Introducing variations in all the independent, free variables $\delta\mathbf{x}, \delta\mathbf{u}, \delta\mathbf{p}$ and $\delta\dot{\mathbf{x}}$ which is not an independent variable but varies due to the dynamics $\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u}, t)$ resulting in the variation δJ:

$$\delta J = \int_{t_0}^{\infty} (\delta\mathbf{p}^T \dot{\mathbf{x}} - \delta\mathbf{x}^T \dot{\mathbf{p}} - \delta\mathbf{x}^T \frac{\partial H}{\partial \mathbf{x}} - \delta\mathbf{p}^T \frac{\partial H}{\partial \mathbf{p}} - \delta\mathbf{u}^T \frac{\partial H}{\partial \mathbf{u}})\, dt \tag{4}$$

Integrating by parts the 1st integrand and after some manipulation we get:

$$\delta J = \int_{t_0}^{\infty} (\delta\mathbf{p}^T (\dot{\mathbf{x}} - \frac{\partial H}{\partial \mathbf{p}}) - \delta\mathbf{x}^T (\dot{\mathbf{p}} + \frac{\partial H}{\partial \mathbf{u}}) - \delta\mathbf{u}^T \frac{\partial H}{\partial \mathbf{u}})\, dt \tag{5}$$

To extremize J, δJ should be 0 for any arbitrary variation. Hence this gives rise to **Pontryagin's principle of optimal control**[Gee07], reminiscent of Hamilton's equations of motion:

$$\dot{\mathbf{x}} = \frac{\partial H}{\partial \mathbf{p}} \tag{6a}$$

$$\dot{\mathbf{p}} = -\frac{\partial H}{\partial \mathbf{x}} \tag{6b}$$

$$\frac{\partial H}{\partial \mathbf{u}} = \mathbf{0} \tag{6c}$$

Linear Quadratic Regulator

System dynamics and the Lagrangian

This control system works best for linear systems(although any non-linear system could be linearized) governed by the linear differential equation:

$$\dot{\mathbf{x}} = A\mathbf{x} + B\mathbf{u} \tag{7}$$

Here A is an $n \times n$ and B is an $n \times m$ matrix. Hence we are considering m actuators to control an n dimensional system. Consider that our target state is $\mathbf{x} = \mathbf{0}$(easily generalizable to any arbitrary target state by the substitution $\mathbf{x} \rightarrow \mathbf{x} - \mathbf{x}_f$ in the rest of derivation). we define the Lagragian to **quadratically penalize** the error($\mathbf{x} - \mathbf{0}$) and actuation with the resulting Hamiltonian(using Eq. (2) and (7)) as:

$$L = \frac{1}{2}\mathbf{x}^T Q\mathbf{x} + \frac{1}{2}\mathbf{u}^T R\mathbf{u} \tag{8a}$$

$$H = \mathbf{p}^T(A\mathbf{x} + B\mathbf{u}) - \frac{1}{2}\mathbf{x}^T Q\mathbf{x} - \frac{1}{2}\mathbf{u}^T R\mathbf{u} \tag{8b}$$

Here Q and R are positive definite matrices to ensure positive contributions to L.

Optimal control policy from Pontryagin's principle

Taking gradient of Eq. (8b) w.r.t $\mathbf{u}$ and using Eq. (6c) of Pontryagin's principle:

$$\mathbf{u} = R^{-1}B^T\mathbf{p} \tag{9}$$

Taking gradient of Eq. (8b) w.r.t $\mathbf{x}$ and using Eq. (6b) of Pontryagin's principle:

$$\dot{\mathbf{p}} = Q\mathbf{x} - A^{-1}\mathbf{p} \tag{10}$$

Here we introduce a transformation matrix $K(t)$ defined as $\mathbf{p} = K\mathbf{x}$. Then using Eq. (7), (9) and (10) and substituting $\mathbf{p} = K\mathbf{x}$ we get:

$$-\dot{K}\mathbf{x} = (KA + KBR^{-1}B^TK - Q + A^TK)\mathbf{x} \tag{11}$$

Now for control problems with infinite time horizons as in the considered case $\dot{K} = 0$ and this gives the **Continuous Algebraic Riccati Equation** or **CARE**:

$$K = -A^{-1}(KBR^{-1}B^TK - Q + A^TK) \tag{12}$$

Numerically, this equation can be solved using fixed point iteration method since it resembles the problem of finding x for $\phi(x) = x$. Now, once K is computed, the control policy nicely reduces to:

$$\mathbf{u} = R^{-1}B^T K\mathbf{x} \tag{13}$$

This intuitively suggests that the actuation given in *linear opposition* to error which in this case is $\mathbf{x}$ since the target state is $\mathbf{0}$. So, if we were to deploy LQR in a 1-dimensional system like cruise control where the state to be controlled in speed and actuation is acceleration, then the acceleration will be proportional to error in speed.

LQR for path tracking

To demonstrate LQR, a path tracking example is demonstrated in relevance to self-driving cars. In the simulator Gazebo, consider a problem statement of autonomously controlling the steering wheel to track a sinusoidal path at a constant speed.

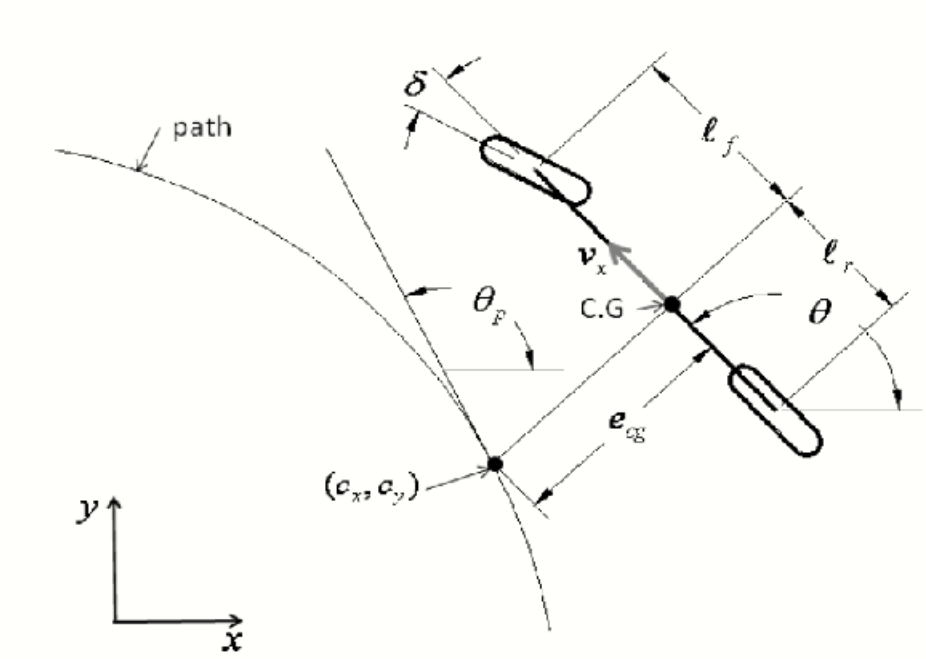

Figure 1: Dynamic model of car for autonomous path tracking

With the objective being to keep the car on track, the state vector $\mathbf{x} = (e_{cg}, \dot{e_{cg}}, \theta_e, \dot{\theta_e})^T$ where e_{cg} is called *cross track error* as clear from Fig:1, while θ_{cg} is the error in orientation of car given by $\theta_p - \theta$ from Fig:1. These two components have to be steered to 0. Now their time derivatives are also included in the state vector because the car is supposed to be stable at $e_{cg} = 0$ and $\theta_e = 0$, hence their derivatives will also be steered to 0. The only actuator we have here is the steering wheel, so $\mathbf{u} = \delta$ i.e 1-dimensional actuation vector. The system is actually non-linear but it is linearized in the form of

Eq.(7) as[Sni+09]:

$$\begin{pmatrix} \dot{e_{cg}} \\ \ddot{e_{cg}} \\ \dot{\theta_e} \\ \ddot{\theta_e} \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & -\frac{c_f+c_r}{mv} & \frac{c_f+c_r}{m} & \frac{l_f c_f - l_r c_r}{mv} \\ 0 & 0 & 0 & 1 \\ 0 & \frac{l_r c_r - l_f c_f}{I_z v} & \frac{l_f c_f - l_r c_r}{I_z} & -\frac{l_f^2 c_f + l_r^2 c_r}{I_z v} \end{pmatrix} \begin{pmatrix} e_{cg} \\ \dot{e_{cg}} \\ \theta_e \\ \dot{\theta_e} \end{pmatrix} + \begin{pmatrix} 0 \\ \frac{c_f}{m} \\ 0 \\ \frac{l_f c_f}{I} \end{pmatrix} \delta \tag{14}$$

Here m is mass, I is moment of inertia about vertical axis through COM, v is speed, c_f and c_r are coefficients of slip force generated by front and rear wheels respectively. δ is the steering angle.
While running the simulation the input is the path(an array of coordinates) and the instantaneous position and velocity of the car(which is maintained at 5 kmph using a separate PID controller). The algorithm summarized would be:

Data: Path array, Constants of Eq.(14), Q and R positive definite matrices
while *System Running* **do**
  Get car's position (s_x, s_y) and orientation θ
  Search point of closest distance i.e (c_x, c_y)
  Compute θ_p from path array using finite difference
  Compute magnitudes of the components of $\mathbf{x} = (e_{cg}, \dot{e_{cg}}, \theta_e, \dot{\theta_e})$
  if $s_x c_y - s_y c_x < 0$ **then**
    $e_{cg} \rightarrow -e_{cg}$
    if $\theta_e < 0$ **then**
      $\dot{e_{cg}} \rightarrow -\dot{e_{cg}}$
    else
      $\dot{e_{cg}} \rightarrow \dot{e_{cg}}$
    end
  else
    $e_{cg} \rightarrow e_{cg}$
    if $\theta_e > 0$ **then**
      $\dot{e_{cg}} \rightarrow -\dot{e_{cg}}$
    else
      $\dot{e_{cg}} \rightarrow \dot{e_{cg}}$
    end
  end
end

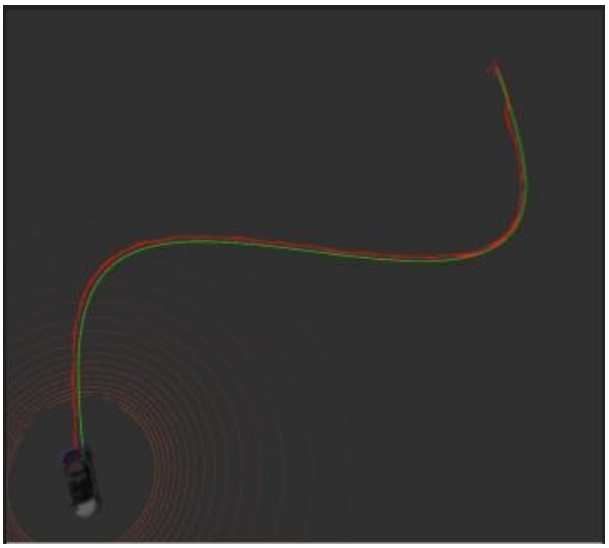

Figure 2: Visualization of simulation in Rviz

When this algorithm is implemented along with a PID controller for speed, result visualized in Fig 2 is obtained. The result is nearly optimal and there are some fluctuations and steady state error due to non-linearities in model, sensor and actuator noises etc. These could be rectified in the upgraded version of LQR called i-LQG(iterative-Linear Quadratic Gaussian) that assumes sensor noises to be Brownian and filters them, while non-linearity is handled by iterative procedure instead of instantaneous one that we have done here. Although it works on the same principle as LQR.

The complete code can be found in [Sho19a] and a video of running simulation in [Sho19b]. To get an intuition of why the Pontryagin's principle automatically gives a closed loop feedback control law in the form $\mathbf{u} = \mathbf{Cx}$ consider the same system without any actuation. Then Eq (7) reduces to:

$$\dot{\mathbf{x}} = \mathbf{Ax} \tag{15}$$

Solution of this system of linear differential equation is:

$$\mathbf{x(t)} = e^{\mathbf{A}t}\mathbf{x(0)} \tag{16}$$

Diagonalizing the matrix $\mathbf{A}$ and exponentiating with time we get:

$$\mathbf{x(t)} = \mathbf{P}\begin{pmatrix} e^{\lambda_1 t} & 0 & 0 & 0 \\ 0 & e^{\lambda_2 t} & 0 & 0 \\ 0 & 0 & ... & 0 \\ 0 & 0 & 0 & e^{\lambda_n t} \end{pmatrix}\mathbf{P}^{-1}\mathbf{x(0)} \tag{17}$$

This shows that if *any* of the eigenvalues λ_i has a *positive real part* then the system will blow up to infinity in configuration space in the direction along the corresponding eigenvector $\mathbf{e}_i$. But now if we deploy a system, with the help of actuators, a control law $\mathbf{u} = \mathbf{Cx}$ then Eq (7) reduces to:

$$\dot{\mathbf{x}} = (\mathbf{A} + \mathbf{BC})\mathbf{x} \tag{18}$$

Hence we have *virtually changed* the system and now we can *steer* all the eigenvalues λ_i wherever we want, favourably keeping all the real parts negative and imaginary parts small(to reduce oscillatory behaviour). And this is exactly what LQR does!

References

[Gee07] Hans P Geering. *Optimal control with engineering applications*. Springer, 2007.

[Sni+09] Jarrod M Snider et al. "Automatic steering methods for autonomous automobile path tracking". In: *Robotics Institute, Pittsburgh, PA, Tech. Rep. CMU-RITR-09-08* (2009).

[Sho19a] Kartik Punjabi Shobhit Saheb Dey. *controls*. Dey-Coded. 2019.

[Sho19b] Kartik Punjabi Shobhit Saheb Dey. *controls*. Drive Link. 2019.