# Optimal Control, Lecture 7: Approximation in Policy Space and Iterative Learning Control

Anders Hansson

Division of Automatic Control
Linkoping University

# Contents

- Approximation in policy space
- Iterative Learning Control (ILC)
- Iterative Feedback Tuning (IFT)
- Guest Lecture by Mikael Norrlöf, ABB Robotics

# Approximation in Policy Space

Assume that we are given a $Q$-function, and that we would like to solve

$$\mu(x) = \operatorname*{argmin}_u Q(x,u) \tag{1}$$

approximately. Define $\tilde{\mu}(a,u)$ using an ANN or a linear regression with $a$ as parameter vector. Solve the LS problem

$$\text{minimize } \frac{1}{2}\sum_{k=1}^{N} \|u_k - \tilde{\mu}(a,u_k)\|_2^2$$

with variable $a$, where $u_k$, $k \in \mathbb{N}_N$ are solutions to (1) for the samples $x = x_k$.

- Less optimization problems need to be solved when the policy is implemented in real-time
- Evaluating $\tilde{\mu}$ might be done much faster than solving the optimization problem.
- This approach can be used for any policy for which we know its values for a discrete number of samples.

# Iterative Learning Control (ILC)

Consider

$$\begin{aligned} &\text{minimize } \phi(x_N) + \sum_{k=0}^{N-1} f_k(x_k, u_k) \\ &\text{subject to } x_{k+1} = F_k(x_k, u_k), \quad k \in \mathbf{Z}_{N-1} \end{aligned} \tag{2}$$

Define $J(u)$ as the function defined by objective function when constraints used to substitute away $x_k$, i.e. $J$ is a function of $u = (u_0, \ldots, u_{N-1})$.

Similarly consider $x_k$ to be functions of $u$.

# Gradient of $J$

Chain rule gives

$$\frac{dJ(u)}{du^T} = \frac{d\phi(x_N)}{dx_N^T}\frac{dx_N}{du^T} + \sum_{k=0}^{N-1} \frac{\partial f_k(x_k, u_k)}{\partial x_k^T}\frac{dx_k}{du^T} + \frac{\partial f_k(x_k, u_k)}{\partial u_k^T}\frac{du_k}{du^T},$$

where

$$\frac{dx_{k+1}}{du_l^T} = \frac{\partial F_k(x_k, u_k)}{\partial x_k^T}\frac{dx_k}{du_l^T} + \frac{\partial F_k(x_k, u_k)}{\partial u_k^T}\delta(k-l),$$

and where

$$\delta(k) = \begin{cases} 1, & k = 0 \\ 0, & k \neq 0. \end{cases}$$

The initial value is zero. Notice that $\frac{du_k}{du^T}$ is a trivial matrix.

For the linear case when $F_k(x,u) = A_k x + B_k u$ we have

$$\frac{dx_{k+1}}{du_l^T} = A_k \frac{dx_k}{du_l^T} + B_k \delta(k-l).$$

# Gradient of $J$ ctd.

For the gradient one simulation or experiment has to be carried out for each value of $l$ and each component of the control signal.

In case $A_k$ and $B_k$ do not depend on $k$

$$\frac{dx_k}{du_l^T} = \frac{dx_{k-l}}{du_0^T},$$

Hence just needs to obtain the so-called impulse response of the dynamical system.

For nonlinear $F_k$, assuming $F_k(0,0) = 0$,

$$\frac{dx_{k+1}}{du_{i,j}} \approx F_k\left(\frac{dx_k}{du_{i,j}}, \frac{du_k}{du_{i,j}}\right).$$

Here index $i$ refers to stage index and index $j$ to component index.

# ILC Using Root-Finding

Consider

$$x_{k+1} = F_k(x_k, u_k)$$
$$y_k = G_k(x_k, u_k)$$

Assume $x_0 = 0$. Let $y = (y_0, y_1, \ldots, y_{N-1})$ and $u = (u_0, u_1, \ldots, u_{N-1})$. Define $H$ such that

$$y = H(u).$$

Find $u$ such that error signal $\epsilon$ defined as

$$\epsilon(u) = y - r = H(u) - r$$

is zero for given *reference value* $r$ for $y$.

# Root-Finding Algorithm

The following root-finding algorithm

$$u_{k+1} = u_k - t\epsilon(u_k)$$

is used.

Here sub-index $k$ refers to iteration index.

Evaluation of $\epsilon$ can be done with simulations or with experiments on a real dynamical system.

Algorithm converges if $\epsilon$ is Lipschitz continuous with Lipschitz constant $\beta$ and strongly monotone with dissipation $\alpha$ for $t \in (0, 2/(\alpha + \beta)]$

# System Theory Interpretation of Convergence Criteria

Lipschitz constant $\beta$ is *incremental gain* of the dynamical system, i.e. smallest $\beta$ such that

$$\|H(u) - H(v)\|_2 \leq \beta \|u - v\|_2, \quad \forall u, v \in \mathbf{R}^{Nm}.$$

The strong monotonicity condition is the same as saying that the dynamical system is *incrementally strictly passive* with dissipation $\alpha$, i.e.

$$(H(u) - H(v))^T (u - v) \geq \alpha \|u - v\|_2^2, \quad \forall u, v \in \mathbf{R}^{Nm}.$$

# Linear System

Assume $F_k(x,u) = Ax + Bu$ and $G_k = Cx + Du$. Then $H$ is a linear function and we may write $y = Hu$, where

$$H = \begin{bmatrix} h_0 & 0 & \cdots & 0 \\ h_1 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ h_{N-1} & \cdots & h_1 & h_0 \end{bmatrix},$$

where $h_0 = D$ and $h_i = CA^iB \in \mathbf{R}^m$ for $i \in \mathbb{N}$.

# Convergence Criterion

Lipschitz constant is $\beta = \|H\|_2$, and criterion for strong monotonicity is

$$(u-v)H^T(u-v) \geq \alpha\|u-v\|_2^2, \quad \forall u, v \in \mathbf{R}^{Nm}.$$

equivalent to

$$\frac{1}{2}x^T\left(H+H^T\right)x \geq \alpha\|x\|_2^2, \quad \forall x \in \mathbf{R}^{Nm},$$

equivalent to $\lambda_{\min} = \lambda_{\min}(H+H^T) > 2\alpha$.

If we then denote largest eigenvalue of $H^TH$ with $\lambda_{\max}$ it follows that the algorithm converges for $t \in (0, 4/(\lambda_{\min} + 2\lambda_{\max})]$, assuming that $\lambda_{\min} > 0$.

# Transformation of $\epsilon$

Let $T$ be such that $T(\epsilon) = 0$ if and only if $\epsilon = 0$, and apply the root finding algorithm to $T(\epsilon(u))$.

One possible choice is to take $T$ as a linear function defined by an invertible matrix $T$, i.e. we consider $Te(u) = 0$.

For the linear case the matrix $H$ above is replace by $TH$, and it is for this matrix we need to compute $\alpha$ and $\beta$.

In case we know the impulse response we may take $T = H^{-1}$ and we obtain convergence in one step with $t_k = 1$.

Also possible to use feedback in order to make $H$ itself close to the identity matrix.

# Iterative Feedback Tuning (IFT)

Consider control signal given by policy $\mu$, i.e. $u_k = \mu(x_k, a)$, where $a$ parameter that we want to learn. It should solve

$$\text{minimize } \phi(x_N) + \sum_{k=0}^{N-1} f_k(x_k, u_k) \tag{3}$$

$$\text{subject to } x_{k+1} = F_k(x_k, u_k), \quad k \in \mathbf{Z}_{N-1} \tag{4}$$

$$u_k = \mu(x_k, a), \quad k \in \mathbf{Z}_{N-1} \tag{5}$$

with variable $a$ for a given initial value $x_0$, where $\phi$, $f_k$ and $F_k$ are defined as before.

Define $J(a)$ as the function defined by (3) when (4— 5) are used to substitute away $x_k$ and $u_k$. Similarly consider $x_k$ and $u_k$ to be functions of $a$.

# Gradients of $J$

Using the chain rule

$$\frac{dJ(a)}{da^T} = \frac{d\phi(x_N)}{dx_N^T}\frac{dx_N}{da^T} + \sum_{k=0}^{N-1} \frac{\partial f_k(x_k, u_k)}{\partial x_k^T}\frac{dx_k}{da^T} + \frac{\partial f_k(x_k, u_k)}{\partial u_k^T}\frac{du_k}{da^T},$$

where

$$\begin{aligned}
\frac{dx_{k+1}}{da^T} &= \frac{\partial F(x_k, u_k)}{\partial x_k^T}\frac{dx_k}{da^T} + \frac{\partial F(x_k, u_k)}{\partial u_k^T}\frac{du_k}{da^T} \\
\frac{du_k}{da^T} &= \frac{\partial \mu(x_k, a)}{\partial x_k^T}\frac{dx_k}{da^T} + \frac{\partial \mu(x_k, a)}{\partial a^T}.
\end{aligned}$$

The initial value is zero.

# Linear System

Let $F_k(x,u) = A_k x + B_k u$, and let $\mu(x,a) = Lx$, where $L \in \mathbf{R}^{m\times n}$ with $a^T = \begin{bmatrix} L_1 & \cdots & L_m \end{bmatrix}$, where $L_i$ are the rows of $L$, we have

$$\begin{aligned} \frac{dx_{k+1}}{da^T} &= A_k \frac{dx_k}{da^T} + B_k \frac{du_k}{da^T} \\ \frac{du_k}{da^T} &= L \frac{dx_k}{da^T} + \mathrm{bdiag}(x_k^T). \end{aligned}$$

Derivatives are obtained by simulation of closed loop system or from experiments involving closed loop system with current $x_k$ as an additional input.