Discrete-time optimal control—direct approach

Fixed finite horizon and receding horizon (MPC)

Zdeněk Hurák
March 3, 2021

DISCRETE-TIME optimal control will not only serve us in our course as a natural transition from the mathematical domain of finite-dimensional (nonlinear) optimization to the control-theoretic domain of optimal control but also will offer us a practically useful control design tool. After all, vast majority of control systems implemented today run on digital computers, hence they run in discrete time. In this lecture we are going to approach the optimal control problem by its *direct transcription* as an optimization problem.

1 Optimal control for a discrete-time system on a fixed and finite horizon—general nonlinear problem

We start by considering the most general problem statement

$$\underset{\mathbf{u}_i,\ldots,\mathbf{u}_{N-1},(\mathbf{x}_i),\ldots,\mathbf{x}_N}{\text{minimize}} \quad \left(\phi(\mathbf{x}_N, N) + \sum_{k=i}^{N-1} L_k(\mathbf{x}_k, \mathbf{u}_k)\right) \tag{1}$$

$$\text{subject to} \quad \mathbf{x}_{k+1} = \mathbf{f}_k(\mathbf{x}_k, \mathbf{u}_k), \tag{2}$$

$$\mathbf{x}_i = \text{given}, \tag{3}$$

$$(\mathbf{x}_N = \text{given or not}), \tag{4}$$

$$\mathbf{u}_{\text{lb}} \leq \mathbf{u}_k \leq \mathbf{u}_{\text{ub}}, \tag{5}$$

$$\mathbf{x}_{\text{lb}} \leq \mathbf{x}_k \leq \mathbf{x}_{\text{ub}}. \tag{6}$$

Note that the $\phi()$ function penalizing the state at the final time could be incorporated as another $L_k()$ term within the sum in (1). True, but most often than not we have different requirements on the states during the transient period and at the end of control and the current form of the cost function makes this distinction conveniently explicit. For instance, all the L_ks could be set to zero and the cost is then given purely by the final state. On a formal side, these three possible formats of a cost function are labelled as Bolza, Lagrange and Mayer forms.

Also note that both the initial and the final states can be either fixed or free. As a matter of fact, the initial state $\mathbf{x}_i$ is in most circumstances assumed as given/fixed (although there might be reasonable problems where the initial state is also regarded

as and unknown variable that must to be determined through optimization). If the final state is fixed, it makes no sense to keep the term $\phi()$ in the cost function.

Clearly this optimal control problem is an instance of a general optimization problem

$$\begin{aligned} \underset{\mathbf{x}\in\mathbb{R}^{n(N-i)},\mathbf{u}\in\mathbb{R}^{m(N-i)}}{\text{minimize}} \quad & J(\mathbf{x},\mathbf{u}) && (7)\\ \text{subject to} \quad & \mathbf{g}(\mathbf{x},\mathbf{u})=0, && (8)\\ & \mathbf{h}(\mathbf{x},\mathbf{u})\leq\mathbf{0}, && (9) \end{aligned}$$

where $\mathbf{u}=(\mathbf{u}_i,\ldots,\mathbf{u}_{N-1})$ and $\mathbf{x}=(\mathbf{x}_{i+1},\ldots,\mathbf{x}_N)$ (note that if $\mathbf{x}_i$ is fixed, it does not have to be considered as an optimization variable).

2 Optimal control for a discrete-time system on a fixed and finite horizon—linear system and quadratic cost

There are two options: first, we may require $\mathbf{x}_N=\mathbf{r}_N$, in which case the first term in the optimization is redundant; second, we may require $\mathbf{x}_N\approx\mathbf{r}_N$, in which case the first term in the optimization criterion contains the weight of the final-time regulation error. The application context will give a guidance which of the two frameworks is the most appropriate.

Below we formulate the optimal *regulation*, that is, the goal is to bring the state either exactly or approximately to zero, that is, $\mathbf{x}_N=\mathbf{0}$ and $\mathbf{x}_N\approx\mathbf{0}$, respectively.

$$\begin{aligned} \underset{\mathbf{u}_0,\ldots,\mathbf{u}_{N-1},(\mathbf{x}_0),\ldots,\mathbf{x}_N}{\text{minimize}} \quad & \frac{1}{2}\mathbf{x}_N^T\mathbf{S}\mathbf{x}_N+\frac{1}{2}\sum_{k=0}^{N-1}\left(\mathbf{x}_k^T\mathbf{Q}\mathbf{x}_k+\mathbf{u}_k^T\mathbf{R}\mathbf{u}_k\right) && (10)\\ \text{subject to} \quad & \mathbf{x}_{k+1}=\mathbf{A}\mathbf{x}_k+\mathbf{B}\mathbf{u}_k, && (11)\\ & \mathbf{x}_0=\mathbf{r}_0, && (12)\\ & (\mathbf{x}_N=\mathbf{0}\text{ or not fixed}). && (13) \end{aligned}$$

Below we elaborate on the case of optimal regulation on a finite horizon, that is, the requried state at the end of the time interval is $\mathbf{x}_N\approx\mathbf{0}$. Modification to the fixed value is straightforward.

2.1 Simultaneous (sparse) formulation

We rewrite it in the matrix-vector format

$$\underset{\mathbf{u},\mathbf{x}}{\text{minimize}}\frac{1}{2}\left(\begin{bmatrix}\mathbf{x}_1^T & \mathbf{x}_2^T & \dots & \mathbf{x}_N^T\end{bmatrix}\underbrace{\begin{bmatrix}\mathbf{Q} & & & \\ & \mathbf{Q} & & \\ & & \ddots & \\ & & & \mathbf{S}\end{bmatrix}}_{\bar{\mathbf{Q}}}\underbrace{\begin{bmatrix}\mathbf{x}_1\\ \mathbf{x}_2\\ \vdots\\ \mathbf{x}_N\end{bmatrix}}_{\mathbf{x}}\right.$$
$$\left.+\begin{bmatrix}\mathbf{u}_0^T & \mathbf{u}_1^T & \dots & \mathbf{u}_{N-1}^T\end{bmatrix}\underbrace{\begin{bmatrix}\mathbf{R} & & & \\ & \mathbf{R} & & \\ & & \ddots & \\ & & & \mathbf{R}\end{bmatrix}}_{\bar{\mathbf{R}}}\underbrace{\begin{bmatrix}\mathbf{u}_0\\ \mathbf{u}_1\\ \vdots\\ \mathbf{u}_{N-1}\end{bmatrix}}_{\mathbf{u}}\right)+\underbrace{\frac{1}{2}\mathbf{x}_0^T\mathbf{Q}\mathbf{x}_0}_{\text{constant}} \quad (14)$$

subject to

$$\begin{bmatrix}\mathbf{x}_1\\ \mathbf{x}_2\\ \mathbf{x}_3\\ \vdots\\ \mathbf{x}_N\end{bmatrix}=\underbrace{\begin{bmatrix}\mathbf{0} & & & & \\ \mathbf{A} & \mathbf{0} & & & \\ & \mathbf{A} & \mathbf{0} & & \\ & & & \ddots & \\ & & & \mathbf{A} & \mathbf{0}\end{bmatrix}}_{\bar{\mathbf{A}}}\begin{bmatrix}\mathbf{x}_1\\ \mathbf{x}_2\\ \mathbf{x}_3\\ \vdots\\ \mathbf{x}_N\end{bmatrix}+\underbrace{\begin{bmatrix}\mathbf{B} & & & & \\ & \mathbf{B} & & & \\ & & \mathbf{B} & & \\ & & & \ddots & \\ & & & & \mathbf{B}\end{bmatrix}}_{\bar{\mathbf{B}}}\begin{bmatrix}\mathbf{u}_0\\ \mathbf{u}_1\\ \mathbf{u}_2\\ \vdots\\ \mathbf{u}_{N-1}\end{bmatrix}+\underbrace{\begin{bmatrix}\mathbf{A}\\ \mathbf{0}\\ \mathbf{0}\\ \vdots\\ \mathbf{0}\end{bmatrix}}_{\bar{\mathbf{A}}_0}\mathbf{x}_0. \quad (15)$$

Note that the last term in the cost function need not be considered because it is constant.

The terms with the $\mathbf{x}$ vector can be combined and we get

$$\begin{bmatrix}\mathbf{0}\\ \mathbf{0}\\ \mathbf{0}\\ \vdots\\ \mathbf{0}\end{bmatrix}=\underbrace{\begin{bmatrix}-\mathbf{I} & & & & \\ \mathbf{A} & -\mathbf{I} & & & \\ & \mathbf{A} & -\mathbf{I} & & \\ & & & \ddots & \\ & & & \mathbf{A} & -\mathbf{I}\end{bmatrix}}_{\bar{\mathbf{A}}-\mathbf{I}}\begin{bmatrix}\mathbf{x}_1\\ \mathbf{x}_2\\ \mathbf{x}_3\\ \vdots\\ \mathbf{x}_N\end{bmatrix}+\underbrace{\begin{bmatrix}\mathbf{B} & & & & \\ & \mathbf{B} & & & \\ & & \mathbf{B} & & \\ & & & \ddots & \\ & & & & \mathbf{B}\end{bmatrix}}_{\bar{\mathbf{B}}}\begin{bmatrix}\mathbf{u}_0\\ \mathbf{u}_1\\ \mathbf{u}_2\\ \vdots\\ \mathbf{u}_{N-1}\end{bmatrix}+\underbrace{\begin{bmatrix}\mathbf{A}\\ \mathbf{0}\\ \mathbf{0}\\ \vdots\\ \mathbf{0}\end{bmatrix}}_{\bar{\mathbf{A}}_0}\mathbf{x}_0. \quad (16)$$

Upon stacking the two vectors we reformulate the constraint as

$$\mathbf{0}=\underbrace{\begin{bmatrix}(\bar{\mathbf{A}}-\mathbf{I}) & \bar{\mathbf{B}}\end{bmatrix}}_{\tilde{\mathbf{A}}}\underbrace{\begin{bmatrix}\mathbf{x}\\ \mathbf{u}\end{bmatrix}}_{\tilde{\mathbf{x}}}+\underbrace{\bar{\mathbf{A}}_0\mathbf{x}_0}_{\tilde{\mathbf{b}}}. \quad (17)$$

and the optimization cost as

$$\underset{\tilde{\mathbf{x}}\in\mathbb{R}^{2N}}{\text{minimize}}\quad \frac{1}{2}\underbrace{\begin{bmatrix}\mathbf{x}^T & \mathbf{u}^T\end{bmatrix}}_{\tilde{\mathbf{x}}^T}\underbrace{\begin{bmatrix}\bar{\mathbf{Q}} & \\ & \bar{\mathbf{R}}\end{bmatrix}}_{\tilde{\mathbf{Q}}}\underbrace{\begin{bmatrix}\mathbf{x}\\ \mathbf{u}\end{bmatrix}}_{\tilde{\mathbf{x}}} \quad (18)$$

We have reformulated the optimal control problem as the linearly constrained quadratic program

$$\boxed{\begin{array}{ll} \underset{\tilde{\mathbf{x}}\in\mathbb{R}^{2N}}{\text{minimize}} & \frac{1}{2}\tilde{\mathbf{x}}^T\tilde{\mathbf{Q}}\tilde{\mathbf{x}} \\ \text{subject to} & \tilde{\mathbf{A}}\tilde{\mathbf{x}}+\tilde{\mathbf{b}}=\mathbf{0}. \end{array}} \tag{19}$$

$$\text{subject to} \quad \tilde{\mathbf{A}}\tilde{\mathbf{x}}+\tilde{\mathbf{b}}=\mathbf{0}. \tag{20}$$

We can introduce a (vector) Lagrange multiplier λ to form the Lagrange function

$$\mathcal{L}(\tilde{\mathbf{x}},\boldsymbol{\lambda})=\frac{1}{2}\tilde{\mathbf{x}}^T\tilde{\mathbf{Q}}\tilde{\mathbf{x}}+\boldsymbol{\lambda}(\tilde{\mathbf{A}}\tilde{\mathbf{x}}+\tilde{\mathbf{b}}), \tag{21}$$

for which the gradient(s) with respect to $\tilde{\mathbf{x}}$ and $\boldsymbol{\lambda}$ is (are)

$$\nabla_{\mathbf{x}}\mathcal{L}(\tilde{\mathbf{x}},\boldsymbol{\lambda})=\tilde{\mathbf{Q}}\tilde{\mathbf{x}}+\tilde{\mathbf{A}}^T\boldsymbol{\lambda}, \tag{22}$$

$$\nabla_{\boldsymbol{\lambda}}\mathcal{L}(\tilde{\mathbf{x}},\boldsymbol{\lambda})=\tilde{\mathbf{A}}\tilde{\mathbf{x}}+\tilde{\mathbf{b}}. \tag{23}$$

Setting the overall gradient to zero enforces the following KKT set of linear equations

$$\begin{bmatrix}\tilde{\mathbf{Q}} & \tilde{\mathbf{A}}^T\\ \tilde{\mathbf{A}} & \mathbf{0}\end{bmatrix}\begin{bmatrix}\tilde{\mathbf{x}}\\ \boldsymbol{\lambda}\end{bmatrix}=\begin{bmatrix}\mathbf{0}\\ -\tilde{\mathbf{b}}\end{bmatrix}. \tag{24}$$

Solving this could be accomplished by using some general solver for linear systems (by calling the backslash operator in Matlab) or by using some more tailored solver for symmetric indefinite systems (based on LDL factorization, for example ldl() in Matlab).

2.1.1 Adding constraints on controls and states

Practically speaking, we while solving a real(istic) optimal control problem, we should expect some inequality constraints on $\mathbf{u}$ and possibly on $\mathbf{x}$ as well, in which case the KKT system above would have to be augmented and we resort to some already finetuned numerical solver for quadratic programming (QP) instead.

2.2 Sequential (dense) formulation

We can get rid of the constraints by expressing $\mathbf{x}$ as a function of $\mathbf{u}$ and $\mathbf{x}_0$. This can be done in a straightforward way by using (16). Namely,

$$\mathbf{x}=(\mathbf{I}-\bar{\mathbf{A}})^{-1}\bar{\mathbf{B}}\mathbf{u}+(\mathbf{I}-\bar{\mathbf{A}})^{-1}\bar{\mathbf{A}}_0\mathbf{x}_0. \tag{25}$$

However, instead of solving the sets of equations, we can do this substitution in a more insightful way. We write down the state equation for several discrete times

$$\mathbf{x}_1=\mathbf{A}\mathbf{x}_0+\mathbf{B}\mathbf{u}_0 \tag{26}$$

$$\mathbf{x}_2=\mathbf{A}\mathbf{x}_0+\mathbf{B}\mathbf{u}_0 \tag{27}$$

$$=\mathbf{A}(\mathbf{A}\mathbf{x}_0+\mathbf{B}\mathbf{u}_0)+\mathbf{B}\mathbf{u}_0 \tag{28}$$

$$=\mathbf{A}^2\mathbf{x}_0+\mathbf{A}\mathbf{B}\mathbf{u}_0+\mathbf{B}\mathbf{u}_0 \tag{29}$$

$$\vdots \tag{30}$$

$$\mathbf{x}_k=\mathbf{A}^k\mathbf{x}_0+\mathbf{A}^{k-1}\mathbf{B}\mathbf{u}_0+\mathbf{A}^{k-2}\mathbf{B}\mathbf{u}_1+\ldots\mathbf{B}\mathbf{u}_{k-1}. \tag{31}$$

Rewriting into matrix-vector form (and extending the time to the full interval)

$$\begin{bmatrix}\mathbf{x}_1\\ \mathbf{x}_2\\ \vdots\\ \mathbf{x}_N\end{bmatrix} = \underbrace{\begin{bmatrix}\mathbf{B} & & & \\ \mathbf{AB} & \mathbf{B} & & \\ \vdots & & \ddots & \\ \mathbf{A}^{N-1}\mathbf{B} & \mathbf{A}^{N-2}\mathbf{B} & & \mathbf{B}\end{bmatrix}}_{\hat{\mathbf{C}}} \begin{bmatrix}\mathbf{u}_0\\ \mathbf{u}_1\\ \vdots\\ \mathbf{u}_{N-1}\end{bmatrix} + \underbrace{\begin{bmatrix}\mathbf{A}\\ \mathbf{A}^2\\ \vdots\\ \mathbf{A}^N\end{bmatrix}}_{\hat{\mathbf{A}}} \mathbf{x}_0 \tag{32}$$

For convenience, let's rewrite the compact relation between $\mathbf{x}$ and $\mathbf{u}$ and $\mathbf{x}_0$

$$\mathbf{x} = \hat{\mathbf{C}}\mathbf{u} + \hat{\mathbf{A}}\mathbf{x}_0. \tag{33}$$

We can now substitute this into the original cost (which then becomes independent of $\mathbf{x}$, which we reflect by using a new name $\tilde{J}$)

$$\tilde{J}(\mathbf{u},\mathbf{x}_0) = \frac{1}{2}(\hat{\mathbf{C}}\mathbf{u} + \hat{\mathbf{A}}\mathbf{x}_0)^T\bar{\mathbf{Q}}(\hat{\mathbf{C}}\mathbf{u} + \hat{\mathbf{A}}\mathbf{x}_0) + \frac{1}{2}\mathbf{u}^T\bar{\mathbf{R}}\mathbf{u} + \frac{1}{2}\mathbf{x}_0^T\mathbf{Q}\mathbf{x}_0 \tag{34}$$

$$= \frac{1}{2}\mathbf{u}^T\hat{\mathbf{C}}^T\bar{\mathbf{Q}}\hat{\mathbf{C}}\mathbf{u} + \mathbf{x}_0^T\hat{\mathbf{A}}^T\bar{\mathbf{Q}}\hat{\mathbf{C}}\mathbf{u} + \frac{1}{2}\mathbf{x}_0^T\hat{\mathbf{A}}^T\bar{\mathbf{Q}}\hat{\mathbf{A}}\mathbf{x}_0 + \frac{1}{2}\mathbf{u}^T\bar{\mathbf{R}}\mathbf{u} + \frac{1}{2}\mathbf{x}_0^T\mathbf{Q}\mathbf{x}_0 \tag{35}$$

$$= \frac{1}{2}\mathbf{u}^T(\hat{\mathbf{C}}^T\bar{\mathbf{Q}}\hat{\mathbf{C}} + \bar{\mathbf{R}})\mathbf{u} + \mathbf{x}_0^T\hat{\mathbf{A}}^T\bar{\mathbf{Q}}\hat{\mathbf{C}}\mathbf{u} + \frac{1}{2}\mathbf{x}_0^T(\hat{\mathbf{A}}^T\bar{\mathbf{Q}}\hat{\mathbf{A}} + \mathbf{Q})\mathbf{x}_0. \tag{36}$$

From the point of view of finding the optimizer, the last term (the one independent of $\mathbf{u}$) can be omitted from the optimization. We will not introduce a new name for the cost function, though

$$\tilde{J}(\mathbf{u},\mathbf{x}_0) = \frac{1}{2}\mathbf{u}^T\underbrace{(\hat{\mathbf{C}}^T\bar{\mathbf{Q}}\hat{\mathbf{C}} + \bar{\mathbf{R}})}_{\mathbf{H}}\mathbf{u} + \mathbf{x}_0^T\underbrace{\hat{\mathbf{A}}^T\bar{\mathbf{Q}}\hat{\mathbf{C}}}_{\mathbf{F}^T}\mathbf{u}. \tag{37}$$

The gradient of this cost function with respect to $\mathbf{u}$ (the initial state $\mathbf{x}_0$ is regarded as a fixed parameter) is

$$\nabla_{\mathbf{u}}\tilde{J} = \mathbf{H}\mathbf{u} + \mathbf{F}\mathbf{x}_0. \tag{38}$$

Setting it to zero we have to solve

$$\boxed{\mathbf{H}\mathbf{u} = -\mathbf{F}\mathbf{x}_0} \tag{39}$$

for $\mathbf{u}$. Formally, we write the solution as

$$\mathbf{u} = -\mathbf{H}^{-1}\mathbf{F}\mathbf{x}_0 \tag{40}$$

but it is understood that direct computation of the inverse is not a recommended practice (for example, in Matlab we will use the backslash operator, which invokes a solver based on a QR factorization).

What a surprising simplicity! The whole problem of optimal control was reformulated as the problem of solving a set of linear equations.

2.2.1 Adding the constraints on controls

Adding constraints on $\mathbf{u}$ is straightforward. It is just that instead of a linear system we will have a linear system with additional inequality constraints. Let's get one.

$$\underset{\mathbf{u}}{\text{minimize}} \quad \frac{1}{2}\mathbf{u}^T\mathbf{H}\mathbf{u} + \mathbf{x}_0^T\mathbf{F}^T\mathbf{u} \tag{41}$$

$$\text{subject to} \quad \mathbf{u}_k \leq \mathbf{u}_{\max} \tag{42}$$

$$\mathbf{u}_k \geq \mathbf{u}_{\min}, \tag{43}$$

which we can rewrite more explicitly (in the matrix-vector format) as

$$\underset{\mathbf{u}}{\text{minimize}} \quad \frac{1}{2}\mathbf{u}^T\mathbf{H}\mathbf{u} + \mathbf{x}_0^T\mathbf{F}^T\mathbf{u} \tag{44}$$

$$\text{subject to} \begin{bmatrix} \mathbf{I} & & & \\ & \mathbf{I} & & \\ & & \ddots & \\ & & & \mathbf{I} \\ -\mathbf{I} & & & \\ & -\mathbf{I} & & \\ & & \ddots & \\ & & & -\mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{u}_0 \\ \mathbf{u}_1 \\ \vdots \\ \mathbf{u}_{N-1} \end{bmatrix} \leq \begin{bmatrix} \mathbf{u}_{\max} \\ \mathbf{u}_{\max} \\ \vdots \\ \mathbf{u}_{\max} \\ -\mathbf{u}_{\min} \\ -\mathbf{u}_{\min} \\ \vdots \\ -\mathbf{u}_{\min} \end{bmatrix}. \tag{45}$$

2.2.2 Adding the constraints on states

We might feel a little bit uneasy about loosing an immediate access to $\mathbf{x}$. But the game is not lost. We just need to express $\mathbf{x}$ as a function of $\mathbf{u}$ and $\mathbf{x}_0$ and impose the constrain on this expression. But we already have such expression, don't we? See (33). Therefore, we can formulate the constraint, say, an upper bound on $\mathbf{x}$

$$\mathbf{x}_k \leq \mathbf{x}_{\max} \tag{46}$$

as

$$\hat{\mathbf{C}}\mathbf{u} + \hat{\mathbf{A}}\mathbf{x}_0 \leq \mathbf{x}_{\max}. \tag{47}$$

3 Optimal control for a discrete-time system on a receding horizon—model predictive control

In the previous section we provided a computational framework for determining a finite optimal sequence of controls that minimize a certain cost. However nice the simulation results look, the major shortcoming is that the control is designed and realized in open loop. In other words, as an outcome of an offline design we get a sequence $u_0, u_1, \ldots, u_{N-1}$. How about adjusting this sequence in every period of the discrete-time controller?

[TBD]

4 Further reading

...