From Centralized to Decentralized Control of Complex Systems

In the Pursuit of Efficient Control Approaches

Ionela Prodan

Université Grenoble Alpes, Laboratoire de Conception et d'Intégration des Systèmes (LCIS EA 3747), Valence, F-26009, France,
ionela.prodan@lcis.grenoble-inp.fr,
https://sites.google.com/site/iprodanionela/

Fall semester 2019

Lecture 2

Outline

Basics on constrained optimization

Definition (Convex set)

A set S is convex if for all $x, y \in S$:

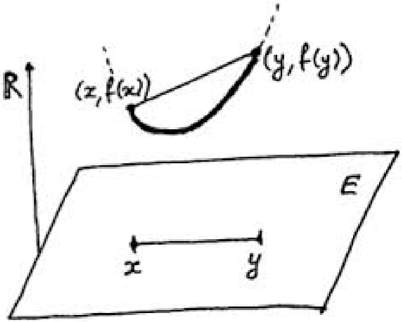
$$\lambda x + (1-\lambda)y \in S, \quad \textit{for all } \lambda \in [0,1]\,.$$

Basics on constrained optimization

Definition (Convex function)

A function $f : S \to \mathbb{R}$ is convex if S is convex and

$$f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda) f(y) \text{ for all } x, y \in S \text{ and } \lambda \in [0,1]\,.$$



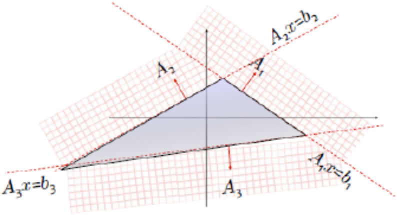
Basics on constrained optimization

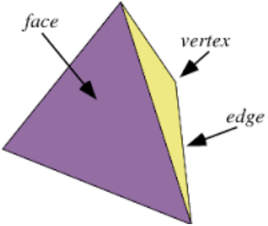
Definition (Polyhedron)

A convex polyhedron is the intersection of a finite set of halfspaces of $\mathbb{R}^n$.

Definition (Polytope)

A convex polytope is a bounded convex polyhedron (hyperplane representation: $Ax \leq b$).





[0] Next lecture will provide more details on set-theoretic methods.

Basics on constrained optimization

Optimization: the problem of choosing a set of parameters that maximize or minimize a given function.

Basics on constrained optimization

Convex optimization problem

$$\begin{aligned} &\text{minimize } f(x) \\ &\text{subject to } x \in C, \quad \text{with } f \text{ and } C \text{ convex.} \end{aligned}$$

- Very efficient numerical algorithm exists
- Global solution attained
- Extensive useful theory
- Often occurring in engineering problems
- Tractable in theory and practice .

[0] Excellent book Boyd and Vandenberghe (2004): Convex optimization. Cambridge university press, http://www.stanford.edu/~boyd/cvxbook/.

Basics on constrained optimization

Linear Program

George Dantzig

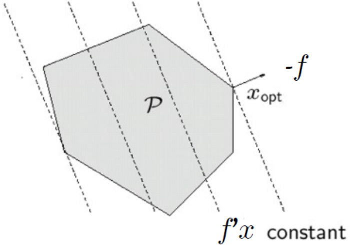
(1914-2005)

minimize $f'x$

subject to $Ax \leq b, \quad x \in \mathbb{R}^n$.

Transformation from max to min:

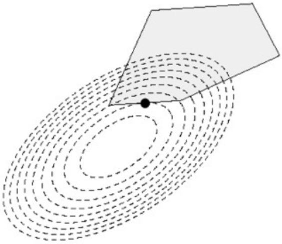
$$\max_x f'x = -(\min_x -f'x)$$



Basics on constrained optimization

Quadratic Program

$$\text{minimize } \frac{1}{2}x'Px + f'x$$
$$\text{subject to } Ax \le b, \quad x \in \mathbb{R}^n.$$



Convex optimization if $P \succeq 0$ (P positive semidefinite matrix).

Hard problem if $P \not\succeq 0$ (P indefinite matrix).

[0]See other papers and lectures of A. Bemporad (IMT Institute for Advanced Studies Lucca, Italy), E. Camacho (University of Seville, Spain).

Basics on constrained optimization

Mixed-Integer Linear Program

$$\begin{aligned} &\text{minimize } f'x + g'\alpha \\ &\text{subject to } Ax + B\alpha \leq c, \\ &\qquad\qquad x \in \mathbb{R}^n, \alpha \in \{0,1\}^m . \end{aligned}$$

Some variables are continous, some are discrete (i.e., $\{0, 1\}$).

In general, it is a NP-hard problem.

Rich variety of algorithms/solvers available.

Basics on constrained optimization

Various solvers and modeling languages in constrained optimization:

- AMPL (A Modeling Language for Mathematical Programming) it is particularly notable for the generality of its syntax and for the variety of its indexing operations.
- GAMS (General Algebraic Modeling System) it is one of the first modeling languages, is tailored for complex, large scale modeling applications.
- GNU MathProg is a subset of AMPL associated with the free package GLPK (GNU Linear Programming Kit).
- CVX is a Matlab-based modeling language (Stephen Boyd and Lieven Vandenberghe from Stanford) `http://stanford.edu/~boyd/cvxbook/`.
- CVXPY is a Python-embedded modeling language for convex optimization problems (Steven Diamond, Stephen Boyd, Eric Chu) `http://www.cvxpy.org/en/latest/`.
- Pyomo is a Python-based, open-source optimization modeling language with a diverse set of optimization capabilities `http://www.pyomo.org/`.
- Yalmip is another Matlab-based modeling language (Johan Lofberg) `http://users.isy.liu.se/johanl/yalmip/pmwiki.php?n=Main.WhatIsYALMIP`.
- CPLEX it comes from IBM, it combines an integrated development environment (IDE) with Optimization Programming Language (OPL) and high-performance ILOG CPLEX optimizer solvers `http://www-01.ibm.com/software/commerce/optimization/cplex-optimizer/`.

Optimal and Constrained Control - An Overview

Optimization-based control: the problem of choosing a set of parameters for a control law so that some performance condition is satisfied.

- Dynamic programming[1]
- Pontryagin's maximum principle[2]
- Model predictive control[3]
- Vertex control[4]
- Interpolation-based control[5]

[1] Bellman, R. (1952): On the theory of dynamic programming. Proceedings of the National Academy of Sciences of the United States of America 38(8), 716.

[2] Clarke F., Pontryagin, L., Gamkrelidze, R. (1986): The mathematical theory of optimal processes, vol. 4.

[3] A.I. Propoi (1963): Use of linear programming methods for synthesizing sampled-data automatic systems. Automation and Remote Control, 24(7):837–844.

[4] Gutman, P., Cwikel, M. (1986): Admissible sets and feedback control for discrete-time linear dynamical systems with bounded controls and states. IEEE Transactions on Automatic Control, 31(4), 373–376.

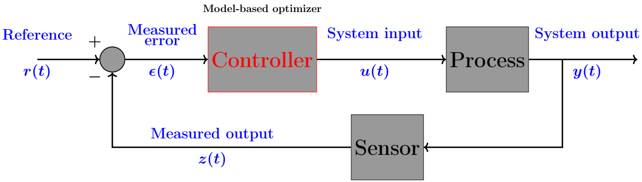
[5] Rossiter, J.A. and Kouvaritakis, B. and Bacic, M. (2004): Interpolation based computationally efficient predictive control, International Journal of Control, vol 77(3), pp. 290–301.

Outline

1. Optimal and Constrained Control - An Overview

2. Receding horizon strategy
   - Unconstrained MPC
   - Constrained MPC
   - Disturbance Handling in MPC

Model Predictive Control



Use the dynamical model of the process to predict its future evolution and optimize the control signal.

Receding horizon philosophy

- At time step t solve an optimal control problem over a finite future horizon of N steps

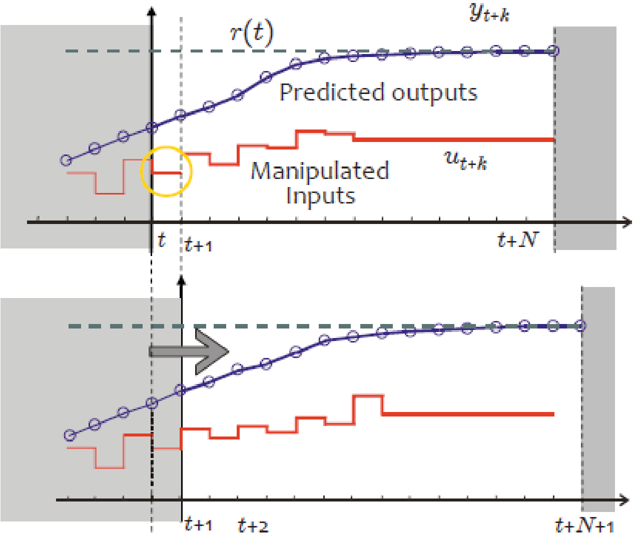
$$\arg \min_{u(t),\dots,u(t+k+N-1)} \sum_{k=0}^{N-1} ||y(t+k)-r(t)||_Q + ||u(t+k)-u_r(t)||_R$$

$$\text{such that: } \begin{cases} x(t+k+1) = f(x(t+k), u(t+k)), \\ y(t+k) = g(x(t+k), u(t+k)), \\ u(t+k) \in \mathcal{U}, y(t+k) \in \mathcal{Y}, k = 0, \dots, N-1. \end{cases}$$

- Apply only the first optimal move u^*
- At time $t+1$ get new measurements, repeat the optimization. And so on...

Advantage of repeated on-line optimization: FEEDBACK.

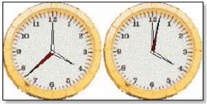
Receding horizon philosophy



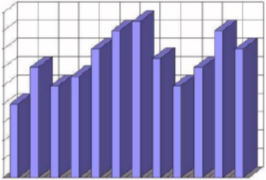
[5] See other papers and lectures of A. Bemporad (IMT Institute for Advanced Studies Lucca, Italy) and M. Alamir (GIPSA lab, France).

Receding horizon - Examples

MPC is like playing chess



"Rolling horizon" policies are used frequently in finance



[5]See other papers and lectures of A. Bemporad (IMT Institute for Advanced Studies Lucca, Italy).

MPC - Examples

prediction model how a vehicle/UAV moves on the map

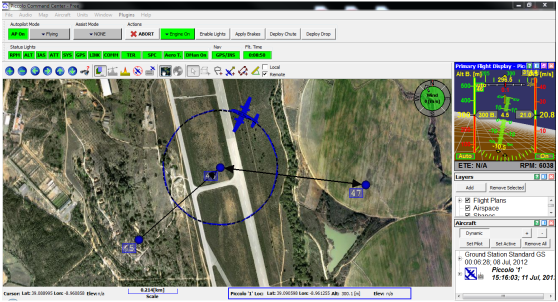
constraints have a certain roll angle, maintain a constant velocity, drive on roads, respect one way roads.

disturbances mainly driver's inattention!, wind, waves etc.,

set point desired location

cost function minimum time, minimum distance, minimum energy, etc.

receding horizon mechanism event-based (optimal route re-planned when path is lost).



Good models for (MPC) control

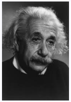
Computation complexity and theorethical properties (e.g., stability, robustness and the like) depend on chosen model/objective/constraints.

Good models for MPC:

- Descriptive enough to caputre the most significant dynamics of the system.

TRADE OFF

- Simple enough for solving the optimization problem.



"Make everything as simple as possible, but not simpler".

Albert Einstein (14 March 1879 - 18 April 1955)

[5] See other papers and lectures of A. Bemporad (IMT Institute for Advanced Studies Lucca, Italy).

MPC in industry

(S.Joe Qin and Thomas A. Badgwell (2003): *A survey of industrial model predictive control technology*)

Area	DMC Corp.	Setpoint Inc.	Honeywell Profimatics	Adersa	Treiber Controls	Total
Refining	360	320	290	280	250	1500
Petrochemicals	210	40	40	-	-	290
Chemicals	10	20	10	3	150	193
Pulp and Paper	10	-	30	-	5	45
Gas	-	-	5	-	-	5
Utility	-	-	2	-	-	2
Air Separation	-	-	-	-	5	5
Mining/Metallurgy	-	2	-	7	6	15
Food Processing	-	-	-	41	-	41
Furnaces	-	-	-	42	-	42
Aerospace/Defense	-	-	-	13	-	13
Automotive	-	-	-	7	-	7
Other	10	20	-	45	-	75
Total	600	402	377	438	416	2233
First App	DMC:1985	IDCOM-M:1987 SMCA:1993	PCT:1984 RMPCT:1991	IDCOM:1973 HIECON:1986	OPC:1987	
Largest App	603x283	35x28	28x20	-	24x19	

Why is MPC so successful?

General reason:

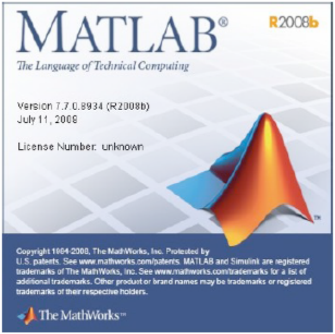
- MPC is most general way of posing the control problem in the time domain (optimal control, stochastic control, known references, measurable disturbances, multivariable, dead time, constraints, uncertainties).

Economical reason:

- MPC can be used to optimize operating points (economic objectives). Optimum usually at the intersection of a set of constraints.
- Obtaining smaller variance and taking constraints into account allow to operate closer to constraints (and optimum).

MPC Toolbox

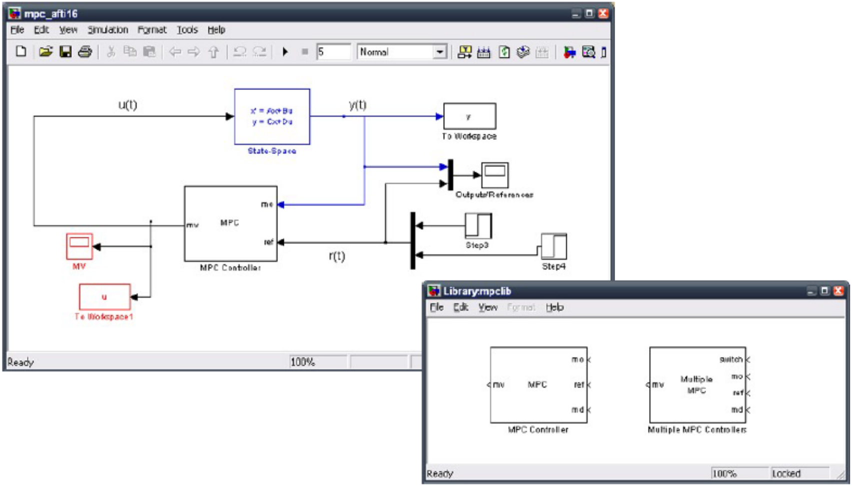
- MPC Toolbox 3.0 (Bemporad, Ricker, Morari, 1998-today):
  - Object-oriented implementation (MPC object)
  - MPC Simulink Library
  - MPC Graphical User Interface
  - RTW extension (code generation) [xPC Target, dSpace, etc.]
  - Linked to OPC Toolbox v2.0.1



Only linear models are handled

http://www.mathworks.com/products/mpc/

MPC Simulink Library



Unconstrained MPC

- Linear model: $\begin{cases} x(t+1) = Ax(t) + Bu(t) \\ y(t) = Cx(t) \end{cases}$ $x \in \mathbb{R}^n$, $u \in \mathbb{R}^m$, $y \in \mathbb{R}^p$

- Goal: find $u^*(0),\ u^*(1),\ \ldots,\ u^*(N-1)$

$$\boxed{J(x(0),U) = \sum_{k=0}^{N-1} \left[x'(k)Qx(k) + u'(k)Ru(k)\right] + x'(N)Px(N)}$$

$$U = [u'(0)\ u'(1)\ \ldots\ u'(N-1)]'$$

$u^*(0),\ u^*(1),\ \ldots,\ u^*(N-1)$ is the input sequence that steers the state to the origin "optimally"

Computation of the cost function

$$J(x(0),U) = x'(0)Qx(0) + \begin{bmatrix} x'(1) & x'(2) & \dots & x'(N-1) & x'(N) \end{bmatrix} \overbrace{\begin{bmatrix} Q & 0 & 0 & \dots & 0 \\ 0 & Q & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & \dots & 0 & Q & 0 \\ 0 & 0 & \dots & 0 & P \end{bmatrix}}^{\bar{Q}} \cdot$$

$$\cdot \begin{bmatrix} x(1) \\ x(2) \\ \vdots \\ x(N-1) \\ x(N) \end{bmatrix} + \begin{bmatrix} u'(0) & u'(1) & \dots & u'(N-1) \end{bmatrix} \underbrace{\begin{bmatrix} R & 0 & \dots & 0 \\ 0 & R & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \dots & 0 & R \end{bmatrix}}_{\bar{R}} \begin{bmatrix} u(0) \\ u(1) \\ \vdots \\ u(N-1) \end{bmatrix}$$

$$\begin{bmatrix} x(1) \\ x(2) \\ \vdots \\ x(N) \end{bmatrix} = \overbrace{\begin{bmatrix} B & 0 & \dots & 0 \\ AB & B & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ A^{N-1}B & A^{N-2}B & \dots & B \end{bmatrix}}^{\bar{S}} \begin{bmatrix} u(0) \\ u(1) \\ \dots \\ u(N-1) \end{bmatrix} + \underbrace{\begin{bmatrix} A \\ A^2 \\ \vdots \\ A^N \end{bmatrix}}_{\bar{T}} x(0)$$

$$\begin{aligned} J(x(0),U) &= x'(0)Qx(0) + (\bar{S}U + \bar{T}x(0))'\bar{Q}(\bar{S}U + \bar{T}x(0)) + U'\bar{R}U \\ &= \frac{1}{2}U' \underbrace{2(\bar{R} + \bar{S}'\bar{Q}\bar{S})}_{H} U + x'(0) \underbrace{2\bar{T}'\bar{Q}\bar{S}}_{F} U + \frac{1}{2}x'(0) \underbrace{2(Q + \bar{T}'\bar{Q}\bar{T})}_{Y} x(0) \end{aligned}$$

Unconstrained MPC

Solve on-line the optimization problem

$$\min_U J(x(t), U) = \frac{1}{2} U^T H U + x^T(t) F U$$

Solution: $\nabla_U J(x(t), U) = HU + F^T x(t) = 0,$

$$U^* = -H^{-1} F^T x(t)$$

$$u(t) = -\left[I \ 0 \ldots \ 0\right] H^{-1} F^T x(t)$$

$$\boxed{u(t) \triangleq K x(t),}$$

Unconstrained MPC is nothing else than a standard linear state-feedback law.

Constrained MPC

- Linear model: $\begin{cases} x(t+1) = Ax(t) + Bu(t) \\ y(t) = Cx(t) \end{cases}$ $\quad x \in \mathbb{R}^n,\ u \in \mathbb{R}^m$, $\ y \in \mathbb{R}^p$

- Constraints: $\begin{cases} u_{\min} \le u(t) \le u_{\max} \\ y_{\min} \le y(t) \le y_{\max} \end{cases}$

- Constrained optimal control problem (quadratic performance index):

$$
\begin{aligned}
\min_{u(0),\dots,u(N-1)} \quad & \sum_{k=0}^{N-1} \left[x'(k)Qx(k) + u'(k)Ru(k) \right] + x'(N)Px(N) \\
\text{s.t.} \quad & u_{\min} \le u(k) \le u_{\max},\ k = 0,\dots,N-1 \\
& y_{\min} \le y(k) \le y_{\max},\ k = 1,\dots,N
\end{aligned}
$$

$$Q = Q' \succeq 0,\ R = R' \succ 0,\ P \succeq 0$$

Constrained MPC

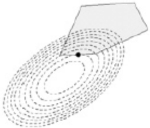
- Optimization problem:

$$\boxed{\begin{aligned} V(x(0)) = \tfrac{1}{2}x'(0)Yx(0) + \min_{U}\ & \tfrac{1}{2}U'HU + x'(0)FU \\ \text{s.t.}\ & GU \leq W + Sx(0) \end{aligned}}$$

(quadratic)

(linear)

Convex QUADRATIC PROGRAM (QP)

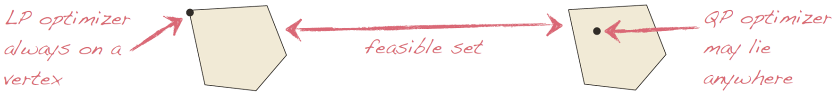


- $U \triangleq [u'(0)\ \dots\ u'(N-1)]' \in \mathbb{R}^s$, $s \triangleq Nm$, is the optimization vector
- $H = H' \succ 0$, and H, F, Y, G, W, S depend on weights Q, R, P, upper and lower bounds $u_{\min}$, $u_{\max}$, $y_{\min}$, $y_{\max}$, and model matrices A, B, C

[5] See other papers and lectures of A. Bemporad (IMT Institute for Advanced Studies Lucca, Italy).

MPC Computations

- The on-line optimization problem is a Quadratic Program (QP) or Linear Program (LP).
- Algorithms:
  - Active set mehods (small/medium size)
  - Interior point methods (large size)
- Benchmarks on commercial/public domain QP/LP solvers
  http://plato.la.asu.edu/bench.html.
- Remark: using Linear Programming (LP) for $1-$ or $\infty-$norms, control action may be less smooth than with QP.



Disturbance Handling in MPC

Consider the LTI discrete system affected by disturbances $w \in \mathbb{R}^n$:

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k) + Gw(k), \\ y(k) &= Cx(k). \end{aligned}$$

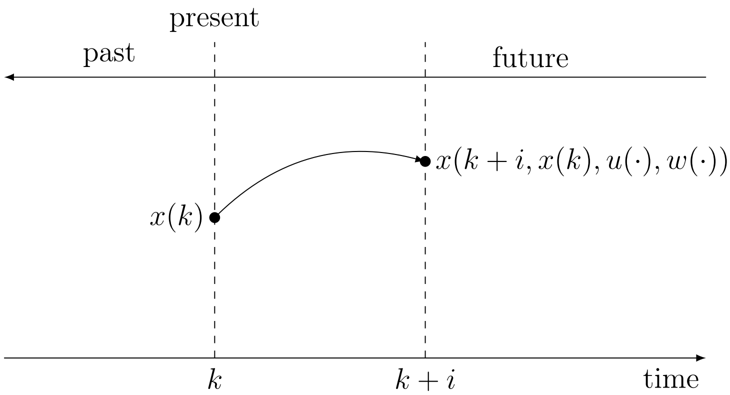
The disturbance vector $w \in \mathbb{R}^n$ can be:

- measured with known dynamics
- measured with unknown dynamics
- unmeasured with known dynamics
- unmeasured with unknown dynamics

In case where the disturbance is not modeled, predictive control need to use some assumption on the future behavior of w during the future prediction horizon.

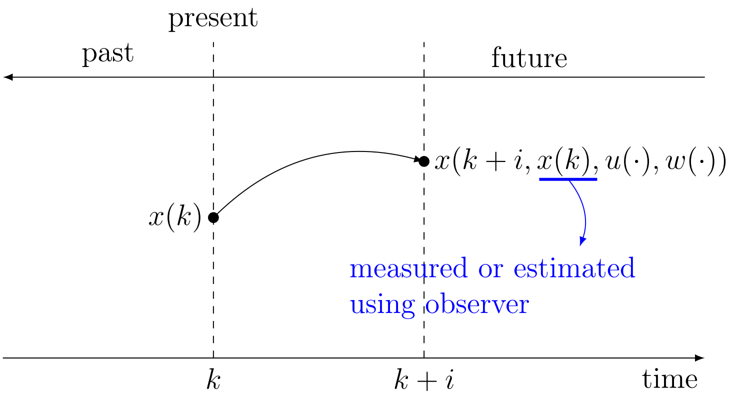
[5] See other papers and lectures of M. Alamir (GIPSA Lab, Grenoble INP, France).

Disturbance Handling in MPC



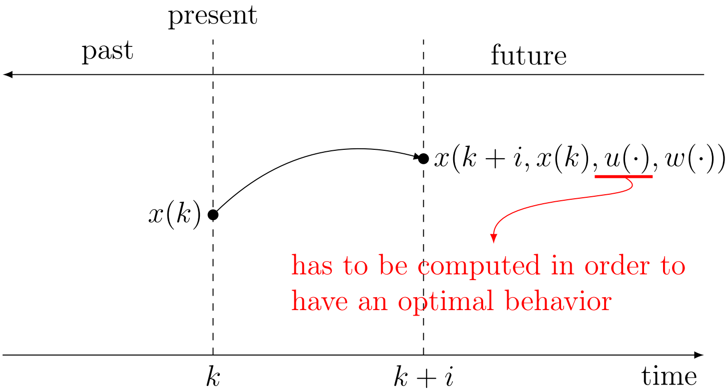
[5] See other papers and lectures of M. Alamir (GIPSA Lab, Grenoble INP, France).

Disturbance Handling in MPC



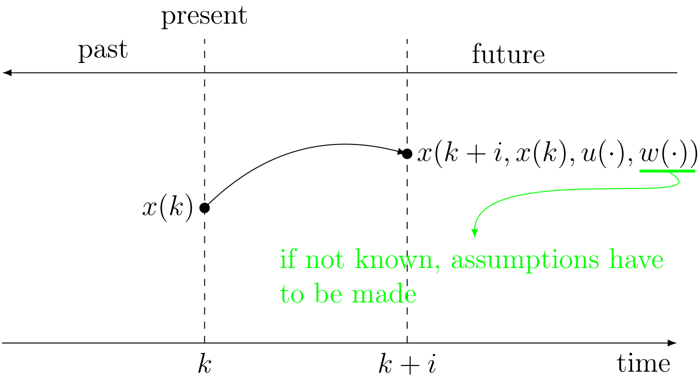
[5] See other papers and lectures of M. Alamir (GIPSA Lab, Grenoble INP, France).

Disturbance Handling in MPC



[5] See other papers and lectures of M. Alamir (GIPSA Lab, Grenoble INP, France).

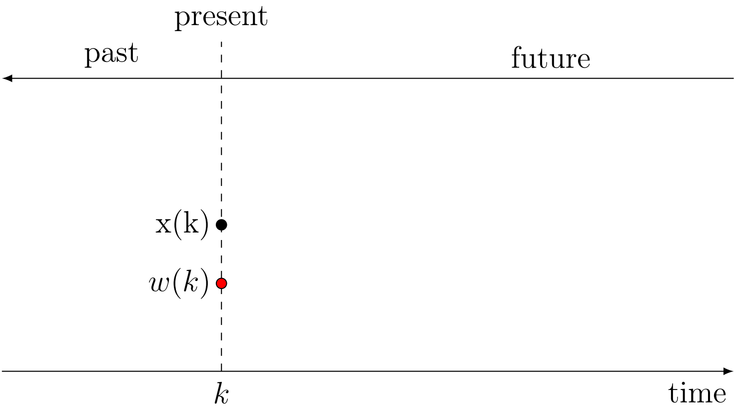
Disturbance Handling in MPC



[5] See other papers and lectures of M. Alamir (GIPSA Lab, Grenoble INP, France).

Disturbance Handling in MPC

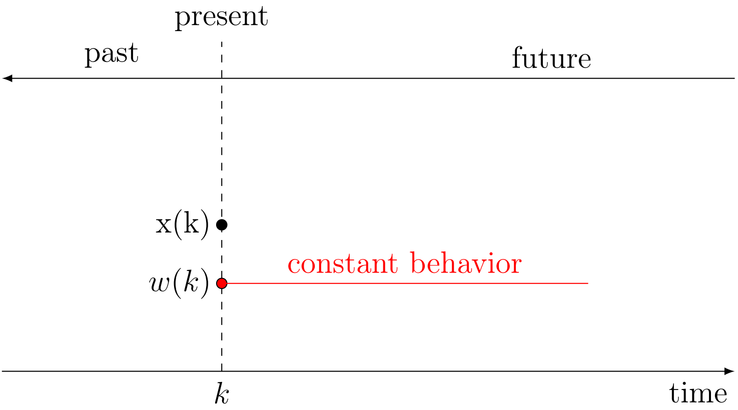
Standard assumptions on future disturbance behavior



[5] See other papers and lectures of M. Alamir (GIPSA Lab, Grenoble INP, France).

Disturbance Handling in MPC

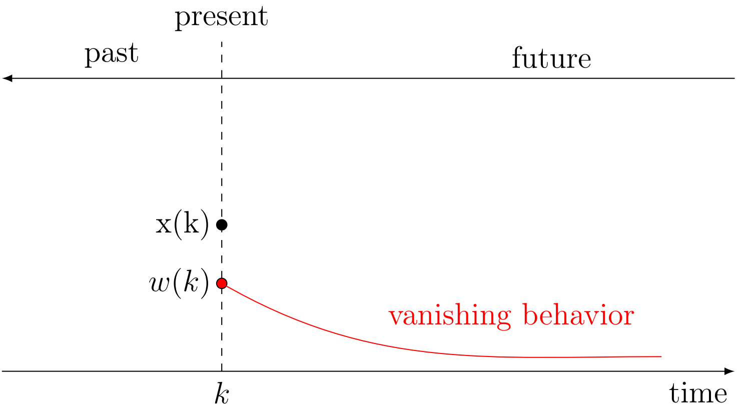
Standard assumptions on future disturbance behavior



[5]See other papers and lectures of M. Alamir (GIPSA Lab, Grenoble INP, France).

Disturbance Handling in MPC

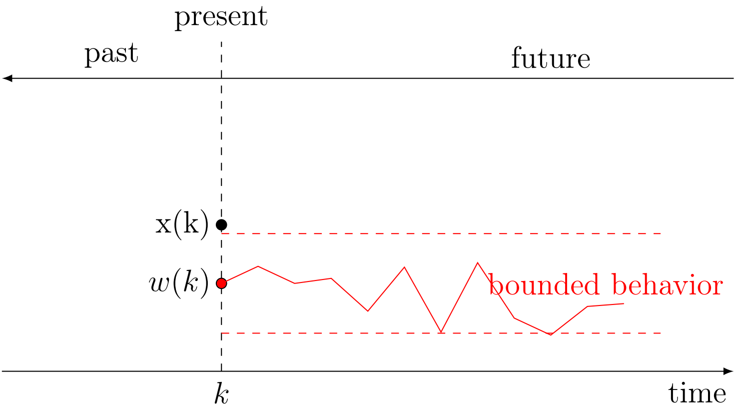
Standard assumptions on future disturbance behavior



[5]See other papers and lectures of M. Alamir (GIPSA Lab, Grenoble INP, France).

Disturbance Handling in MPC

Standard assumptions on future disturbance behavior



[5] See other papers and lectures of M. Alamir (GIPSA Lab, Grenoble INP, France).

Notions on observer design

For a LTI discrete system with $x \in \mathbb{R}^n$, $u \in \mathbb{R}^n$ and $y \in \mathbb{R}^m$

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k), \\ y(k) &= Cx(k). \end{aligned}$$

The observation paradigm is to know how to estimate the state $x(k)$ using the past measurements:

$$y(k), y(k-1), \cdots, y(k-N_0+1)$$

WHEN is this possible?

If this is possible, HOW to design the corresponding algorithm ?

Notions on observer design

For a LTI discrete system with $x \in \mathbb{R}^n$, $u \in \mathbb{R}^n$ and $y \in \mathbb{R}^m$

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k), \\ y(k) &= Cx(k). \end{aligned}$$

The observation paradigm is to know how to estimate the state $x(k)$ using the past measurements:

$$y(k), y(k-1), \cdots, y(k-N_0+1)$$

WHEN is this possible?

if the following condition is satisfied:

$$rank\left(\begin{bmatrix} C \\ CA \\ \vdots \\ CA^{n-1} \end{bmatrix}\right) = n \text{ (dimension of the state} x \in \mathbb{R}^n\text{)}$$

Notions on observer design

For a LTI discrete system with $x \in \mathbb{R}^n$, $u \in \mathbb{R}^n$ and $y \in \mathbb{R}^m$

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k), \\ y(k) &= Cx(k). \end{aligned}$$

The observation paradigm is to know how to estimate the state $x(k)$ using the past measurements:

$$y(k), y(k-1), \cdots, y(k-N_0+1)$$

HOW to design the corresponding algorithm ?

The dynamic observer equation is given by:

$$\hat{x}(k+1) = (A-LC)\hat{x}(k) + Bu(k) + Ly(k),$$

where $L \in \mathbb{R}^{n \times n_y}$ is computed such that:

$$\max_{i=1:N} |\lambda_i(A-LC)| < 1$$

.

(for instance, use $L = dlqr(A^T, C^T, Q, R, N)$).

Disturbance Estimation

Consider the LTI discrete system:

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k) + Gw(k), \\ y(k) &= Cx(k). \end{aligned}$$

This can be rewritten equivalently as follows:

$$\begin{aligned} \begin{bmatrix} x(k+1) \\ w(k+1) \end{bmatrix} &= \underbrace{\begin{bmatrix} A & G \\ 0_{n_w \times n} & I_{n_w} \end{bmatrix}}_{\tilde{A}} \underbrace{\begin{bmatrix} x(k) \\ w(k) \end{bmatrix}}_{\tilde{x}} + \underbrace{\begin{bmatrix} B \\ 0_{n_u} \end{bmatrix}}_{\tilde{B}} u(k) \\ y(k) &= \underbrace{\begin{bmatrix} C & 0_{n_y \times n_w} \end{bmatrix}}_{\tilde{C}} \underbrace{\begin{bmatrix} x(k) \\ w(k) \end{bmatrix}}_{\tilde{x}} \end{aligned}$$

The same analysis and design can be done on the extended system:

$$rank\left(\begin{bmatrix} \tilde{C} \\ \tilde{C}\tilde{A} \\ \vdots \\ \tilde{C}\tilde{A}^{n-1} \end{bmatrix}\right) = n + n_w.$$

Disturbance Estimation

Generally, it is assumed that the disturbance has a constant dynamic:

$$
\begin{aligned}
x(k+1) &= Ax(k) + Bu(k) + Gw(k), \\
w(k+1) &= w(k), \\
y(k) &= Cx(k).
\end{aligned}
$$

This can be rewritten equivalently as follows:

$$
\begin{aligned}
\begin{bmatrix} x(k+1) \\ w(k+1) \end{bmatrix} &= \underbrace{\begin{bmatrix} A & G \\ 0_{n_w \times n} & I_{n_w} \end{bmatrix}}_{\tilde{A}} \underbrace{\begin{bmatrix} x(k) \\ w(k) \end{bmatrix}}_{\tilde{x}} + \underbrace{\begin{bmatrix} B \\ 0_{n_u} \end{bmatrix}}_{\tilde{B}} u(k) \\
y(k) &= \underbrace{\begin{bmatrix} C & 0_{n_y \times n_w} \end{bmatrix}}_{\tilde{C}} \underbrace{\begin{bmatrix} x(k) \\ w(k) \end{bmatrix}}_{\tilde{x}}
\end{aligned}
$$

The same analysis and design can be done on the extended system:

$$
rank\left(\begin{bmatrix} \tilde{C} \\ \tilde{C}\tilde{A} \\ \vdots \\ \tilde{C}\tilde{A}^{n-1} \end{bmatrix}\right) = n + n_w.
$$

Tube MPC

Consider the LTI discrete time in $\mathbb{R}^n$

$$x_d(k+1) = Ax_d(k) + Bu_d(k) + w(k), \quad w \in \mathbb{W} \subset \mathbb{R}^n,$$

with (A, B) stabilizable.

The nominal model of the system

$$x(k+1) = Ax(k) + Bu(k).$$

.

[5] See also Rawlings and Mayne (2009)

Tube MPC

Consider the LTI discrete time in $\mathbb{R}^n$

$$x_d(k+1) = Ax_d(k) + Bu_d(k) + w(k), \quad w \in \mathbb{W} \subset \mathbb{R}^n,$$

with (A, B) stabilizable.

The nominal model of the system

$$x(k+1) = Ax(k) + Bu(k).$$

Construct a RPI set for the system such that

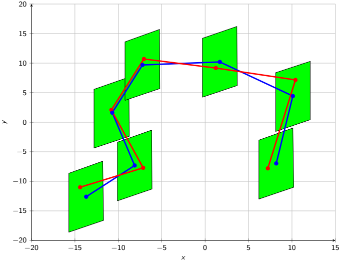
$$\mathcal{S} \triangleq x(k) \oplus \Omega_{UB},$$

where Ω_{UB} is a RPI set for the dynamics

$$z(k+1) = (A + BK)z(k) + w(k),$$

with $z(k) = x_d(k) - x(k) \in \mathcal{S}$

and $u_d(k) = u(k) + Kz(k)$ a stabilizing controller.



.

[5] See also Rawlings and Mayne (2009)

Tube MPC

Consider the LTI discrete time in $\mathbb{R}^n$

$$x_d(k+1) = Ax_d(k) + Bu_d(k) + w(k), \quad w \in \mathbb{W} \subset \mathbb{R}^n,$$

with (A, B) stabilizable.

The nominal model of the system

$$x(k+1) = Ax(k) + Bu(k).$$

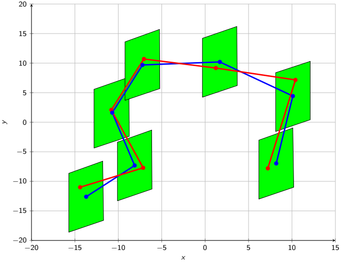
Construct a RPI set for the system such that

$$\mathcal{S} \triangleq x(k) \oplus \Omega_{UB},$$

where Ω_{UB} is a RPI set for the dynamics

$$z(k+1) = (A+BK)z(k) + w(k),$$

with $z(k) = x_d(k) - x(k) \in \mathcal{S}$
and $u_d(k) = u(k) + Kz(k)$ a stabilizing controller.



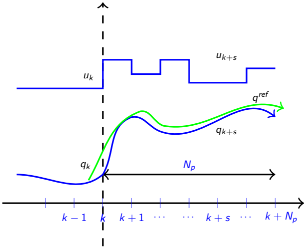
.

5 See also Rawlings and Mayne (2009)

Model Predictive Control (MPC)

Propoi (1963), Cutler et al. (2007), Richalet and O'Donovan (2009)

- Optimization-based control law
- Implicit (on-line) vs. explicit (off-line) implementation
- Constraints handling
- Can be implemented in a distributed fashion



model of the system

reference trajectory

$$\arg\min_{\mathbf{u}} \; V_f(x(k+N_p), q(k+N_p)) + \sum_{s=1}^{N_p-1} V_n(x(k+s), u(k+s), q(k+s))$$

$$\text{subject to:} \begin{cases} x(k+s+1) = f(x(k+s), u(k+s)), & s = 0 : N_p - 1, \\ h(x(k+s), u(k+s), q(k+s)) \leq 0, & s = 1 : N_p - 1. \end{cases}$$

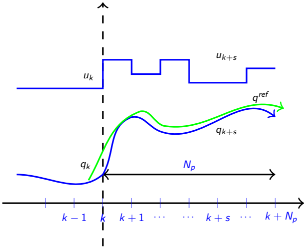
constraint sets representation

optimization solver

Model Predictive Control (MPC)

Propoi (1963), Cutler et al. (2007), Richalet and O'Donovan (2009)

- Optimization-based control law
- Implicit (on-line) vs. explicit (off-line) implementation
- Constraints handling
- Can be implemented in a distributed fashion



model of the system

reference trajectory

$$\arg\min_{\mathbf{u}} \; V_f(x(k+N_p), q(k+N_p)) + \sum_{s=1}^{N_p-1} V_n(x(k+s), u(k+s), q(k+s))$$

$$\text{subject to: } \begin{cases} x(k+s+1) = f(x(k+s), u(k+s)), & s = 0 : N_p - 1, \\ h(x(k+s), u(k+s), q(k+s)) \leq 0, & s = 1 : N_p - 1. \end{cases}$$

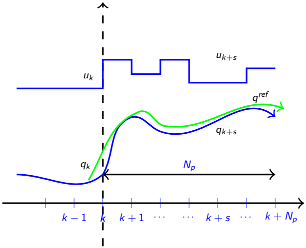
constraint sets representation

optimization solver

Model Predictive Control (MPC)

Propoi (1963), Cutler et al. (2007), Richalet and O'Donovan (2009)

- Optimization-based control law
- Implicit (on-line) vs. explicit (off-line) implementation
- Constraints handling
- Can be implemented in a distributed fashion



model of the system

reference trajectory

$$\arg\min_{\mathbf{u}} \; V_f(x(k+N_p), q(k+N_p)) + \sum_{s=1}^{N_p-1} V_n(x(k+s), u(k+s), q(k+s))$$

$$\text{subject to:} \begin{cases} x(k+s+1) = f(x(k+s), u(k+s)), & s = 0 : N_p - 1, \\ h(x(k+s), u(k+s), q(k+s)) \leq 0, & s = 1 : N_p - 1. \end{cases}$$

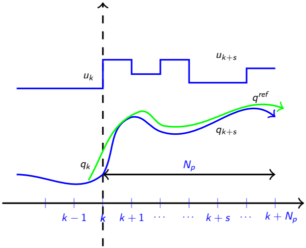
constraint sets representation

optimization solver

Model Predictive Control (MPC)

Propoi (1963), Cutler et al. (2007), Richalet and O'Donovan (2009)

- Optimization-based control law
- Implicit (on-line) vs. explicit (off-line) implementation
- Constraints handling
- Can be implemented in a distributed fashion



model of the system

reference trajectory

$$\underset{\mathbf{u}}{\arg\min}\ V_f(x(k+N_p), q(k+N_p)) + \sum_{s=1}^{N_p-1} V_n(x(k+s), u(k+s), q(k+s))$$

$$\text{subject to: } \begin{cases} x(k+s+1) = f(x(k+s), u(k+s)), & s = 0 : N_p - 1, \\ h(x(k+s), u(k+s), q(k+s)) \leq 0, & s = 1 : N_p - 1. \end{cases}$$

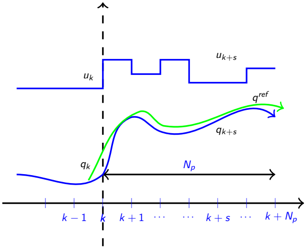
constraint sets representation

optimization solver

Model Predictive Control (MPC)

Propoi (1963), Cutler et al. (2007), Richalet and O'Donovan (2009)

- Optimization-based control law
- Implicit (on-line) vs. explicit (off-line) implementation
- Constraints handling
- Can be implemented in a distributed fashion



model of the system

reference trajectory

$$\arg\min_{\mathbf{u}} \; V_f(x(k+N_p), q(k+N_p)) + \sum_{s=1}^{N_p-1} V_n(x(k+s), u(k+s), q(k+s))$$

$$\text{subject to: } \begin{cases} x(k+s+1) = f(x(k+s), u(k+s)), & s = 0 : N_p-1, \\ h(x(k+s), u(k+s), q(k+s)) \leq 0, & s = 1 : N_p-1. \end{cases}$$

constraint sets representation

optimization solver

References I

S. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

C.R. Cutler et al. Method for removal of pid dynamics from mpc models, 2007. US Patent 7,263,473.

A.I. Propoi. Use of linear programming methods for synthesizing sampled-data automatic systems. *Automation and Remote Control*, 24(7):837–844, 1963.

J.B. Rawlings and D.Q. Mayne. *Model Predictive Control: Theory and Design*. 2009.

J. Richalet and D. O'Donovan. *Predictive Functional Control: Principles and Industrial Applications*. Springer, 2009.