TTK4135 – Optimalization and Control

Sondre Bø Kongsgård
sondrebk@stud.ntnu.no

Contents

1 Definitions 1
2 Background Material 2
3 Tips & Tricks 2
4 Algorithms 3

1 Definitions

Convexity Used to describe a special case of the general constrained optimization problem in which

- The objective function is convex
- The equality constraint functions $c_i(\cdot)$, $i \in \mathcal{E}$ are linear, and
- The inequality constraint functions$c_i(\cdot)$, $i \in \mathcal{I}$ are concave

LICQ (Linear Independence Constraint Qualification) Given the point x and the active set $\mathcal{A}(x)$, we say that the LICQ holds if the set of active constraint gradients $\nabla c_i(x)$, $i \in \mathcal{A}(x)$ is linearly independent.

Search directions Several approaches to line search directions can be used:

Method	Formula
Steepest descent	$p_k = -\nabla f_k$
Newton direction	$p_k^N = -(\nabla^2 f_k)^{-1} \nabla f_k$
Quasi-Newton direction	$p_k = -B_k^{-1} \nabla f_k$
	For updating B_k
SR1(Symmetric-rank-one)	
BFGS	

The Wolfe conditions Are used in line-search methods to decide if the decrease in the objective function is sufficient.

$$f(x_k + \alpha p_k) <= f(x_k) + c_1 \alpha \nabla f_k^T p_k \tag{1}$$

There is also a second Wolfe condition, but we don't need it if we use backtracking.

Merit functions SQP methods often use merit functions to decide whether a trail step should be accepted. In line search methods, the merit function controls the size of the step.

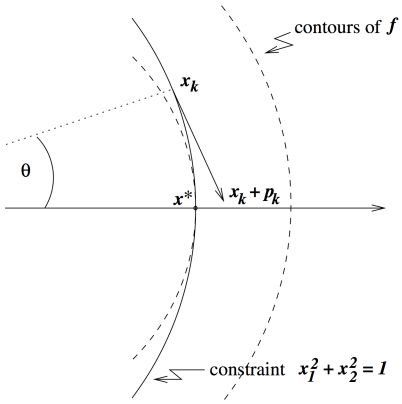
$$l_1: \quad \phi_1(x;\mu) = f(x) + \mu \sum_{i \in \mathcal{E}} |c_i(x)| + \mu \sum_{i \in \mathcal{I}} [c_i(x)]^- \tag{2}$$

Notation: $[z]^- = \max\{0, -z\}$

The positive scalar μ is the penalty parameter, which determines the weight that we assigned to constraint satisfaction relative to minimization of the objective.

Exact merit function: A merit function $\phi(x;\mu)$ is exact if there is a positive scalar μ^* such that for any $\mu > \mu^*$, any local solution of the nonlinear programming problem is a local minimizer of $\phi(x;\mu)$.

Maratos effect The phenomenon where a merit function prevents rapid convergence because steps that make good progress toward a solution are rejected.



Figur 1: Maratos Effect

Model predictive control (MPC) A form of control in which the current control action is obtained by solving, at each sampling instant, a finite horizon open-loop optimal control problem, using the current state of the plant as the initial state; the optimization yields an optimal control sequence and the first control in this sequence is applied to the plant.

Full-space VS Reduced-space formulation

- In full-space optimization, we include all inputs and all states in our objective function. The number of optimization variables is #steps · (#states + #inputs).
- In reduced-space optimization, we remove the states from the objective function by replacing them using the model ($x_{t+1} = A_t x_t + B_t u_t$). The number of optimization variables is #steps · #inputs.

	Pros	Cons
Full-space	Often sparsity in matrices	Many variables
Reduced-space	Less variables	Normally dense matrices

Stabilizability Tells us that we must be able to influence all unstable modes. Important for LQGC.

Detectability A milder form of observability. This implies that an observable system always is detectable. The opposite is however not necessarily true. Important for LQGC

2 Background Material

Matrix factorizations

Cholesky	LU	QR	Symmetric indefinite
$A = LL^\top$	$PA = LU$	$AP = QR$	$PAP^\top = LBL^\top$
	$Ax = b$ $L\underbrace{(Ux)}_{y} = b$ Triangular forward substitution $Ly = b$ for y Triangular backward substitution $Ux = y$ for x		
$A \in \mathbb{R}^{nxn}$ $A = A^T$ (Symmetric) $A \succ 0$	$A \in \mathbb{R}^{nxn}$	$A \in \mathbb{R}^{mxn}$ (only real A)	$A = A^T$ A can be indefinite

Matrix calculus

Derivative

$$\begin{aligned}\nabla(c^\top \mathbf{x}) &= c \\ \nabla(\mathbf{x}^\top c) &= c\end{aligned} \tag{3}$$

$$\nabla\left(\frac{1}{2}\mathbf{x}^\top G\mathbf{x}\right) = \frac{1}{2}G\mathbf{x} + \frac{1}{2}G^\top\mathbf{x} \tag{4}$$

Gradient

$$\nabla f(\mathbf{x}) = \left[\frac{\partial f}{\partial \mathbf{x}}^\top\right] = \begin{bmatrix}\frac{\partial f}{\partial x_1} \\ \vdots \\ \frac{\partial f}{\partial x_n}\end{bmatrix} \tag{5}$$

Hessian

$$\nabla_{xx} f(\mathbf{x}) = \begin{bmatrix}\frac{\partial f}{\partial x_1^2} & \frac{\partial f}{\partial x_1 \partial x_2} & \cdots \\ \frac{\partial f}{\partial x_2 \partial x_1} & \frac{\partial f}{\partial x_2^2} & \\ \vdots & & \ddots\end{bmatrix} \tag{6}$$

Jacobian

$$Jf(\mathbf{x}) = \begin{bmatrix}\frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \\ \vdots & & \ddots\end{bmatrix} \tag{7}$$

Elements of Analysis

Lipschitz continuous

$$||f(x_1) - f(x_0)|| \leq L||x_1 - x_0||, \quad \forall\, x_0, x_1 \in \mathcal{N} \tag{8}$$

Mean value theorem

$$f(x+p) = f(x) + \nabla f(x + \alpha p)^\top p \tag{9}$$

3 Tips & Tricks

LP Remember to read the task closely, and see whether it asks for the given resources to be *fully utilized*. If not, then introduce slack variables in your constraints.

Prediction horizon Prediction horizon Generally, the prediction horizon N should be chosen such that

$$\text{dominant dynamics} < N < \text{control interval} \tag{10}$$

Slack variables We normally have tighter bounds on the state variables than reality since some of the constraints are hard and must always be satisfied. Therefore, the state constraints (A.9d) may be violated for all the time. In this case a feasible point may not exist and a control input may not be available.

Fix: (soften the constraints by using slack variables)

$$\min_{z \in \mathbb{R}^n} f(z) = \cdots + p^\top \epsilon + \frac{1}{2}\epsilon^\top S \epsilon \tag{11}$$

$$\begin{gathered} x^{\text{low}} < x_t < x^{\text{high}} \\ \Downarrow \\ x^{\text{low}} - \epsilon_t < x_t < x^{\text{high}} + \epsilon_t \end{gathered} \tag{12}$$

Linearization of a constraint Use the following formula to linearize a constraint (eg. for use in a QP problem):

$$\nabla c_i(x_i)^\top p + c_1(x_i) \tag{13}$$

4 Algorithms

...

Active set methods Maintain estimates of the active and inactive index sets that are updated at each step of the algorithm. Examples: the simplex method and active-set method.

Simplex

One Step of Simplex()

1 Given $\mathcal{B}, \mathcal{N}, x_B = B^{-1}b \geq, x_N = 0$;
2 Solve $B^\top \lambda = C_B for \lambda$,
3 Compute $s_N = c_N - N^\top \lambda$; (* pricing *)
4 **if** $s_N \geq 0$
5 **stop**; (* optimal point found *)
6 Select $q \in \mathcal{N}$ with $s_q < 0$ as the entering index;
7 Solve $Bd = A_q$ for d;
8 **if** $d \leq 0$
9 **stop**; (* problem is unbounded *)
10 Calculate $x_q^+ = \min_{i|d_i>0}(x_B)_i/d_i$, and use p to denote the minimizing i;
11 Update $x_B^+ = x_B - dx_q^+, x_N^+ = (0, \ldots, 0, x_q^+, 0, \ldots, 0)^\top$;
12 Change $\mathcal{B}$ by adding q and removing the basic variable corresponding to column p of B

MPC-based controllers MPC merges feedback control with dynamic optimization.

Name	Optimization problem solved	Note
MPC	Non-specified (any)	
Linear MPC	QP with linear equality constraints and inequalities on x, u and Δu	
LQ	Convex QP with only linear equality constraints	*Finite-horizon:* $u_t = K_t x_t$ *Infinite-horizon:* $u_t = Kx_t$
LQGC	Same as LQ	Combined with a Kalman filter
NMPC	Non-linear optimization problems	

State feedback MPC procedure()

1 **for** $T = 0, 1, 2, \ldots$ **do**
2 Get the current state x_t
3 Solve a dynamic optimization problem on the prediction horizon from t to $t + N$ with x_t as the initial condition
4 Apply the first control move u_t from the solution above

For an output feedback MPC procedure, line 2 in the algorithm above changes to:

2 Compute an estimate of the current state $\hat{x}_t$ based on the measured data up until time t