Summary of TTK4135 Optimization and Control

Morten Fyhn Amundsen

NTNU

May 28, 2015

Contents

1 Introduction

This document is intended as a summary of the course *TTK4135 Optimization and Control* at NTNU.

Jeg vil overhode ikke garantere for pdf-ens kvalitet.

1.1 Notation

The notation used here is the same as in the textbook *Numerical Optimization.* This means that a lowercase, italic, non-bold character (x) may represent either a scalar or a vector. Matrices are uppercase, italic, and also non-bold (X). One slight difference is in transposed matrices: X^T in the book, but X^{T} here. The latter follows IEEE style. '^{T}' is stupid.

1.2 Todo

- Convexity in the general case.
- LU decomposition.
- Section: Unconstrained optimisation.
- Section: Line search methods.
- Section: Quasi-Newton methods.
- Section: Derivative-free optimisation.
- Section: Open-loop dynamic optimisation.
- Section: MPC.
- Section: LQR.
- Section: SQP.
- 15.3 Elimination of variables.
- 15.4 Merit function.

2 Optimisation problems

2.1 Nonlinear program (NLP) formulation

$$\begin{aligned} \min_{x\in\mathbb{R}^n} \quad & f(x) \\ \text{s.t.} \quad & c_i(x) = 0, \quad i \in \mathcal{E} \\ & c_i(x) \geq 0, \quad i \in \mathcal{I} \end{aligned} \tag{1}$$

2.2 Requirements for convexity

- $f(x)$ is convex.
- $c_i(x) = 0, i \in \mathcal{E}$ are linear.
- $c_i(x) \geq 0, i \in \mathcal{I}$ are concave.

er de to siste punktene det samme som å si "the feasible set is convex"? se side 5 av Foss.

f is convex if its domain S is a convex set and if for any two points $x, y \in S$, the following is satisfied:

$$f(\alpha x + (1-\alpha)y)) \leq \alpha f(x) + (1-\alpha) f(y) \;\; \forall \;\; \alpha \in [0,1] \tag{2}$$

2.3 The feasible set

$$\Omega = \{x \in \mathbb{R}^n \mid c_i(x) = 0, i \in \mathcal{E} \;\; \wedge \;\; c_i(x) \geq 0, i \in \mathcal{I}\} \tag{3}$$

This is just the set of all x satisfying all equality and inequality constraints.

2.4 Global and local solutions

A feasible $x^* \in \Omega$ is a

- **global solution** if $f(x) \geq f(x^*) \;\; \forall \;\; x \in \Omega$;
  (I.e. $f(x)$ is never smaller than $f(x^*)$. A strictly convex problem has one unique global solution.)
- **local solution** if $f(x) \geq f(x^*) \;\; \forall \;\; x$ in some area around x^*;
  (I.e. No directions from x^* decrease $f(x)$.)
- **strict local solution** if $f(x) > f(x^*) \;\; \forall \;\; x \neq x^*$ in some area around x^*.
  (I.e. All directions from x^* increase $f(x)$.)

2.5 The Lagrangian function

Used on problems with only equality constraints.

At the optimal point x^*, the gradient of the constraint is parallel to the gradient of the objective function: $\nabla f(x^*) = \lambda_1^* \nabla c_1(x^*)$ for some λ_1. This is equivalent to

$$\nabla_x \mathcal{L}(x^*, \lambda_1^*) = 0 \tag{4}$$

when the Lagrangian function $\mathcal{L}$ for (1) is defined as

$$\mathcal{L}(x, \lambda) = f(x) - \sum_{i \in \mathcal{E} \cup \mathcal{I}} \lambda_i c_i(x). \tag{5}$$

The point here is that (4) is a necessary condition for an optimal solution. (This can be expanded for problems with several equality constraints, I believe.)

2.6 KKT conditions

If x^* is a local solution of (1), and λ^* is its Lagrange multiplier vector, then the following is satisfied:

$$\begin{aligned}
\nabla_x \mathcal{L}(x^*, \lambda^*) &= 0 && \text{(6a)}\\
c_i(x^*) &= 0 \;\; \forall \;\; i \in \mathcal{E} && \text{(6b)}\\
c_i(x^*) &\geq 0 \;\; \forall \;\; i \in \mathcal{I} && \text{(6c)}\\
\lambda_i^* &\geq 0 \;\; \forall \;\; i \in \mathcal{I} && \text{(6d)}\\
\lambda_i^* c_i(x^*) &= 0 \;\; \forall \;\; i \in \mathcal{E} \cup \mathcal{I} && \text{(6e)}
\end{aligned}$$

If the LICQ holds, λ^* is unique.

2.7 Active set

$$\mathcal{A}(x) = \mathcal{E} \cup \{ i \in \mathcal{I} \mid c_i(x) = 0 \} \tag{7}$$

Or: $\mathcal{A}$ is the set of all indices of constraints that are active at the point x.

2.8 LICQ

The *linear independence constraint qualification* holds if the set of the gradients of all active constraints $\{\nabla c_i(x), i \in \mathcal{A}\}$ is linearly independent. This is necessary at the solution.

2.9 Linearised feasible direction

Given a feasible point $x \in \mathbb{R}^n$ and the active constraint set $\mathcal{A}(x)$, the set of linearised feasible directions $\mathcal{F}(x)$ is

$$\mathcal{F}(x) = \left\{ d \in \mathbb{R}^n \,\middle|\, \begin{array}{l} d^{\mathrm{T}} \nabla c_i(x) = 0 \ \forall \ i \in \mathcal{E} \\ d^{\mathrm{T}} \nabla c_i(x) \geq 0 \ \forall \ i \in \mathcal{A}(x) \cap \mathcal{I} \end{array} \right\} \tag{8}$$

2.10 The Hessian matrix

$$\nabla^2 f(x) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 x_2} & \cdots & \frac{\partial^2 f}{\partial x_1 x_n} \\ \frac{\partial^2 f}{\partial x_2 x_1} & \frac{\partial^2 f}{\partial x_2^2} & \cdots & \frac{\partial^2 f}{\partial x_2 x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n x_1} & \frac{\partial^2 f}{\partial x_n x_2} & \cdots & \frac{\partial^2 f}{\partial x_n^2} \end{bmatrix} \tag{9}$$

2.11 Positive definiteness

A matrix M is positive definite if $z^{\mathrm{T}} M z > 0 \ \forall \ z \neq 0$.

3 Linear programming

3.1 Primal/standard LP formulation

$$\begin{aligned} \min_x \quad & c^{\mathrm{T}} x \\ \text{s.t.} \quad & Ax = b \\ & x \geq 0 \end{aligned} \tag{10}$$

3.2 Slack variables

Inequality constraints on the form $x \leq u$ or $Ax \geq b$ can always be converted to equality constraints by adding slack variables:

$$\begin{aligned} x \leq u \Leftrightarrow x + w = u, &\quad w \geq 0 \\ Ax \geq b \Leftrightarrow Ax - y = b, &\quad y \geq 0 \end{aligned} \tag{11}$$

3.3 Dual LP formulation

$$\begin{aligned} \max_{\lambda,s} \quad & b^{\mathrm{T}}\lambda \\ \text{s.t.} \quad & A^{\mathrm{T}}\lambda + s = c \\ & s \geq 0 \end{aligned} \tag{12}$$

The dual LP problem has the same optimal value $c^{\mathrm{T}}x^* = b^{\mathrm{T}}\lambda^*$ as the primal, and by defining $s = c - A^{\mathrm{T}}\lambda$, the KKT conditions are identical! (And still sufficient.)

The optimal Lagrange multipliers λ^* in the primal problem are the optimal variables in the dual problem, while the optimal Lagrange multipliers x in the dual problem are the optimal variables in the primal problem. (This is cool.)

3.4 Lagrangian function for LP

$$\mathcal{L}(x,\lambda,s) = c^{\mathrm{T}}x - \lambda^{\mathrm{T}}(Ax - b) - s^{\mathrm{T}}x \tag{13}$$

where $\lambda \in \mathbb{R}^m$ is the multiplier vector for the constraints $Ax = b$, while $s \in \mathbb{R}^n$ is the multiplier vector for the constraints $x > 0$

3.5 KKT conditions for LP

$$\begin{aligned} A^{\mathrm{T}}\lambda^* + s^* &= c && \text{(14a)} \\ Ax^* &= b && \text{(14b)} \\ x^* &\geq 0 && \text{(14c)} \\ s^* &\geq 0 && \text{(14d)} \\ x_i^* s_i^* &= 0, \quad i = 1, 2, \ldots, n && \text{(14e)} \end{aligned}$$

For an LP, the KKT conditions are *sufficient* conditions for x^* to be a global solution. (14e) states that x_i^* and/or s_i^* must be zero for each i. Equivalent: $x^{\mathrm{T}*}s^* = 0$.

3.6 Feasible polytope

Figure 1: Feasible Politoed

Theorem. All basic feasible points of a standard LP are vertices of the feasible polytope $\{x \mid Ax = b,\ \ x \geq 0\}$.

3.7 The Simplex algorithm

Iterates between *basic feasible points*. A feasible point x is basic if there is a subset $\mathcal{B}$ of the index set $\{1, 2, \ldots, n\}$ such that

- $\mathcal{B}$ contains m indices.
- $i \notin \mathcal{B} \implies x_i = 0$
- The $m \times m$ matrix $B = [A_i]_{i \in \mathcal{B}}$ is nonsingular (A_i is the ith column of A).

Then $\mathcal{B}$ is a *basis* for (10), and B is the *basis matrix*. The nonbasic set $\mathcal{N} = \{1, 2, \ldots, n\} \backslash \mathcal{B}$ is the complement of $\mathcal{B}$, and the nonbasic matrix is $N = [A_i]_{i \in \mathcal{N}}$.

Features:

- Active set method.
- Usually iterates between adjacent vertices.
  (One index in the basis is changed at each iteration.)
- $c^{\mathrm{T}} x$ will decrease in most iterations.
- Finding a feasible starting point is a significant challenge.

4 Quadratic programming

A quadratic program (QP) has a quadratic objective function and linear constraints.

4.1 Standard formulation

$$\begin{aligned} \min_{x} \quad & \frac{1}{2}x^{\mathrm{T}}Gx + c^{\mathrm{T}}x \\ \text{s.t.} \quad & a_i^{\mathrm{T}}x = b_i, \quad i \in \mathcal{E} \\ & a_i^{\mathrm{T}}x \geq b_i, \quad i \in \mathcal{I} \end{aligned} \tag{15}$$

4.2 Convexity

- $G > 0 \implies$ **strictly convex** (e.g. a bowl shape)
- $G \geq 0 \implies$ **convex** (e.g. a valley shape)

4.3 KKT

The KKT conditions for a *convex* QP are sufficient conditions, and they define a global solution.

4.4 Critical cone

The *critical cone* is defined as:

$$w \in \mathcal{C}(x^*, \lambda^*) \Leftrightarrow \begin{cases} \nabla c_i(x^*)^{\mathrm{T}}w = 0 \;\; \forall \;\; i \in \mathcal{E} \\ \nabla c_i(x^*)^{\mathrm{T}}w = 0 \;\; \forall \;\; i \in \mathcal{A}(x^*) \cap \mathcal{I} \text{ with } \lambda_i^* > 0 \\ \nabla c_i(x^*)^{\mathrm{T}}w \geq 0 \;\; \forall \;\; i \in \mathcal{A}(x^*) \cap \mathcal{I} \text{ with } \lambda_i^* = 0 \end{cases} \tag{16}$$

4.5 Second-order sufficient conditions

If

$$w^{\mathrm{T}}\nabla^2_{xx}\mathcal{L}(x^*, \lambda^*)w > 0, \;\; \forall \;\; w \in \mathcal{C}(x^*, \lambda^*), \;\; w \neq 0 \tag{17}$$

where $x^* \in \mathbb{R}^n$ is a feasible point with a Lagrange multiplier λ^* such that the KKT conditions are satisfied, then x^* is a strict local solution. What is w?

4.6 Equality constrained QP

$$\begin{bmatrix} G & -A^{\mathrm{T}} \\ A & 0 \end{bmatrix} \begin{bmatrix} x^* \\ \lambda^* \end{bmatrix} = \begin{bmatrix} -c \\ b \end{bmatrix} \tag{18}$$

or

$$\underbrace{\begin{bmatrix} G & A^{\mathrm{T}} \\ A & 0 \end{bmatrix}}_{K} \begin{bmatrix} -p \\ \lambda^* \end{bmatrix} = \begin{bmatrix} g \\ h \end{bmatrix} \tag{19}$$

where $p = x^* - x$ and K is the *KKT matrix*. This has a unique global solution if $Z^{\mathrm{T}}GZ$ is pos. def. and A has full rank. Z is a matrix of full rank, and $AZ = 0$.

4.7 Convex inequality constrained QP

Solved with the active set method (chapter 16.5).

The Lagrangean is

$$\mathcal{L}(x^*, \lambda^*) = \frac{1}{2}x^{\mathrm{T}}Gx + x^{\mathrm{T}}c - \sum_{i \in \mathcal{E} \cup \mathcal{I}} \lambda_i(a_i^{\mathrm{T}}x - b_i) \tag{20}$$

and the KKT conditions become

$$Gx^* + c - \sum_{i \in \mathcal{A}(x^*)} \lambda_i^* a_i = 0 \tag{21a}$$

$$a_i^{\mathrm{T}}x^* = b_i, \quad i \in \mathcal{A}(x^*) \tag{21b}$$

$$a_i^{\mathrm{T}}x^* > b_i, \quad i \in \mathcal{I} \setminus \mathcal{A}(x^*) \tag{21c}$$

$$\lambda_i^* \geq 0, \quad i \in \mathcal{I} \cap \mathcal{A}(x^*) \tag{21d}$$

$$\lambda_i^* \left(a_i^{\mathrm{T}}x^* - b_i\right) = 0, \quad i \in \mathcal{I} \tag{21e}$$

The active-set method is outlined as follows:

- Get direction p_k from solving an equality-constrained QP.
- Alter the working set $\mathcal{W}_k$ of active constraints at each iteration.
- Always feasible (require feasible starting point, found by solving a Phase I problem).
- Phase I problem 'as hard as' the original problem.

5 Unconstrained optimisation

$$\min_{x \in \mathbb{R}^n} \quad f(x) \tag{22}$$

5.1 What is a solution?

Theorem 2.2 (First order necessary conditions). If x^* is a local minimiser and f is cont. diff.able in an open area around x^*, then $\nabla f(x^*) = 0$.

Theorem 2.3 (Second order necessary conditions). If x^* is a local minimiser of f and $\nabla^2 f$ exists and is cont. in an area around x^*, then $\nabla f(x^*) = 0$ and $\nabla^2 f(x^*)$ is pos. semidef.

Theorem 2.4 (Second order sufficient conditions). Suppose that $\nabla^2 f$ is cont. in an area around x^* and that $\nabla f(x^*) = 0$ and $\nabla^2 f(x^*)$ is pos. def. Then x^* is a strict local minimiser of f.

6 Line search methods

Concept: Choose a new direction and distance to move for each iteration:

$$x_{k+1} = x_k + \alpha_k p_k \tag{23}$$

Usually, the direction must be a descent direction, and it is often on the form

$$p_k = -B_k^{-1} \nabla f_k \tag{24}$$

where B_k is symmetric and nonsingular. Three choices are common:

- Steepest descent: $B_k = I$
- Newton's method: $B_k = \nabla^2 f(x_k)$
- Quasi-Newton: $B_k \approx \nabla^2 f(x_k)$

6.1 Step length

We want an α_k that gives a good reduction of f without too much computation. The best choice is of course the length that gives the greatest possible reduction of f when moving in the chosen direction, i.e. the *global* minimiser of

$$\phi(\alpha) = f(x_k + \alpha p_k), \tag{25}$$

but that is hard to compute.

6.1.1 Wolfe conditions

The Wolfe conditions are requirements that try to assure a good choice of step length.

$$f(x_k + \alpha_k p_k) \leq f(x_k) + c_1 \alpha_k \nabla f_k^{\mathrm{T}} p_k \tag{26a}$$

$$\nabla f(x_k + \alpha_k p_k)^{\mathrm{T}} p_k \geq c_2 \nabla f_k^{\mathrm{T}} p_k \tag{26b}$$

with $0 < c_1 < c_2 < 1$.

- (26a) is the *sufficient decrease condition.* It assures a certain decrease of f, proportional to the step length. (c_1 is often small, i.e. 10^{-4}.)
- (26b) is the *curvature condition.* It makes sure that you keep going (longer step length) if f goes steeply downhill. If f is steep, it's better to move further. (c_2 is typically 0.9 for Newton and Quasi-Newton methods.)

6.1.2 Sufficient decrease and backtracking

The curvature condition is used because the sufficient decrease condition alone may not assure a good choice of step length. The *backtracking* approach allows us to tell (26b) to fuck right off.

```
Choose $\bar{\alpha} > 0, \rho \in (0,1), c \in (0,1)$.
$\alpha \leftarrow \bar{\alpha}$
**repeat**
    $\alpha \leftarrow \rho\alpha$
**until** $f(x_k + \alpha p_k) \leq f(x_k) + c\alpha \nabla f_k^{\mathrm{T}} p_k$
Terminate with $\alpha_k = \alpha$.
```

6.2 Newton's method with Hessian modification

Chapter 3.4. Syllabus: First two pages until Section *Eigenvalue Modification.*

6.3 Step-length selection algorithms

Chapter 3.5. Syllabus: Down to Section *A Line Search Algorithm for the Wolfe Conditions*

7 Newton and Quasi-Newton methods

Solves unconstrained, nonlinear optimisation problems iteratively:

$$x_{k+1} = x_k + \alpha_k p_k \tag{27}$$

7.1 Newton's method

(Notation: $g_k = \nabla f(x_k)$ and $H_k = \nabla^2 f(x_k)$.)

The second-order Taylor expansion of the objective function near x_k is

$$f(x_k + \Delta x) \approx f(x_k) + \Delta x^{\mathrm{T}} g_k + \frac{1}{2}\Delta x^{\mathrm{T}} H_k \Delta x. \tag{28}$$

The derivative of (28) is

$$\frac{\partial f(x_k + \Delta x)}{\partial \Delta x} = g_k + H_k \Delta x \tag{29}$$

which is zero for Δx that are global minima of $f(x_k + \Delta x)$. Solving for Δx gives

$$\Delta x = -H_k^{-1} g_k \tag{30}$$

This suggests that we should move x_k towards $H_k^{-1} g_k$. In practice, we use

$$x_{k+1} = x_k - \alpha H_k^{-1} g_k \tag{31}$$

and find α with some line search method.

There is a huge drawback: Computing the Hessian takes forever in large systems.

7.2 BFGS Quasi-Newton method

Start at an initial position x_0 and the inverse of an approximated and pos. def. Hessian H_0. Repeat until x_k converges to the solution:

1. Compute search direction:

$$p_k = -H_k \nabla f(x_k) \tag{32}$$

2. Find a step length α_k with some line search method, so that it satisfies the Wolfe conditions (26).

3. Update position:

$$x_{k+1} = x_k + \alpha_k p_k \tag{33}$$

4. Define s_k as the distance moved:

$$s_k = x_{k+1} - x_k = \alpha_k p_k \tag{34}$$

5. Define y_k as the change in the gradient:
$$y_k = \nabla f(x_{k+1}) - \nabla f(x_k) \tag{35}$$
6. Approximate next Hessian inverse (always pos. def.):
$$H_{k+1} = \left(I - \rho_k s_k y_k^{\mathrm{T}}\right) H_k \left(I - \rho_k y_k s_k^{\mathrm{T}}\right) + \rho_k s_k s_k^{\mathrm{T}} \tag{36}$$
where $\rho_k = \left(y_k^{\mathrm{T}} s_k\right)^{-1}$.

Another, slower version exists that works on the (not inverse) Hessian approximation B_k, using the update formula
$$B_{k+1} = B_k + \frac{y_k y_k^{\mathrm{T}}}{y_k^{\mathrm{T}} s_k} - \frac{B_k s_k s_k^{\mathrm{T}} B_k}{s_k^{\mathrm{T}} B_k s_k}. \tag{37}$$

Alternatives to BFGS are SR1 and DFP.
The update formulas are derived from
$$\begin{aligned} \min_{H} \quad & \|H - H_k\| \\ \text{s.t.} \quad & H = H^{\mathrm{T}} \\ & H y_k = s_k. \end{aligned} \tag{38}$$

8 Approximating derivatives

8.1 Finite difference derivative approximation

The *one-sided difference* approximation is
$$\frac{\partial f(x)}{\partial x_i} \approx \frac{f(x + \epsilon e_i) - f(x)}{\epsilon} \tag{39}$$
where ϵ is a small, positive number and e_i is a unit vector. This is quick but not very accurate.

The *central-difference* formula is
$$\frac{\partial f(x)}{\partial x_i} \approx \frac{f(x + \epsilon e_i) - f(x - \epsilon e_i)}{2\epsilon}. \tag{40}$$
This one is roughly twice as expensive to calculate, but also more precise.

9 Derivative-free optimisation

Sometimes we cannot calculate or approximate the derivative, so some guys made methods that do not require any differentiation.

9.1 Finite differences and noise

Fuck this.

9.2 Nelder-Mead method

(First of all: No relation to the Simplex method.)

Keeps track of a *simplex* S with $n + 1$ points in n dimensions (e.g. a triangle on a plane). For each iteration, it removes a bad vertex (large objective value) and finds a better vertex to replace it.

The simplest version replaces the worst vertex with a point reflected through the centroid of the remaining points. If the new point is better than the best current point, we can try moving even further. If it is not much better than the previous, we can try to move shorter (shrinking the simplex).

10 Open loop dynamic optimization

A quadratic objective function is often used for various problems. The objective function

$$f(z) = \sum_{t=0}^{N-1} \frac{1}{2} x_{t+1}^{\mathrm{T}} Q_{t+1} x_{t+1} + d_{xt+1} x_{t+1} + \frac{1}{2} u_t^{\mathrm{T}} R_t u_t + d_{ut} u_t \tag{41}$$

leads to the QP problem in (42) below:

$$\min_{z \in \mathbb{R}^n} \quad f(z) \tag{42a}$$

subject to

$$\begin{aligned}
x_0, u_{-1} &\quad \text{given} && \text{(42b)}\\
x_{t+1} &= A_t x_t + B_t u_t && \text{(42c)}\\
x^{\text{low}} &\leq x_t \leq x^{\text{high}} && \text{(42d)}\\
u^{\text{low}} &\leq u_t \leq u^{\text{high}} && \text{(42e)}\\
-\Delta u^{\text{high}} &\leq \Delta u_t \leq \Delta u^{\text{high}} && \text{(42f)}\\
Q_t &\geq 0 && \text{(42g)}\\
R_t &\geq 0 && \text{(42h)}
\end{aligned}$$

where

$$\Delta u_t = u_t - u_{t-1} \tag{42i}$$

$$z^{\mathrm{T}} = \left(x_1^{\mathrm{T}}, \dots, x_N^{\mathrm{T}}, u_0^{\mathrm{T}}, \dots, u_{N-1}^{\mathrm{T}}\right) \tag{42j}$$

$$n = N(n_x + n_u) \tag{42k}$$

This is an open-loop optimisation problem because there is no feedback present in the solution. We use only the initial state when solving over the event horizon.

11 Model predictive control

General idea: At each sample time, solve a bladi bladi bla.

Can enforce hard bounds on input and output values (which is pretty useful).

11.1 General formulation

11.2 Linear MPC

$$f(z) = \sum_{t=0}^{N-1} \frac{1}{2} x_{t+1}^{\mathrm{T}} Q_{t+1} x_{t+1} + \frac{1}{2} u_t^{\mathrm{T}} R_t u_t + \frac{1}{2} \Delta u_t^{\mathrm{T}} R_{\Delta t} \Delta u_t + d_{xt+1} x_{t+1} + d_{ut} u_t \tag{43}$$

Here the term $\frac{1}{2}\Delta u_t^{\mathrm{T}} R_{\Delta t} \Delta u_t$ is added to penalise control action. This reduces wear on actuators. Otherwise the objective is the same as (42).

12 Linear quadratic control

No bounds on input/output!

12.1 Finite horizon

$$\min_{z \in \mathbb{R}^n} \quad f(z) = \sum_{t=0}^{N-1} \frac{1}{2} x_{t+1}^{\mathrm{T}} Q_{t+1} x_{t+1} + \frac{1}{2} u_t^{\mathrm{T}} R_t u_t \tag{44a}$$

subject to

$$x_{t+1} = A_t x_t + B_t u_t \tag{44b}$$

$$x_0 = \text{given} \tag{44c}$$

where

$$z^{\mathrm{T}} = \left(x_1^{\mathrm{T}}, \dots, x_N^{\mathrm{T}}, u_0^{\mathrm{T}}, \dots, u_{N-1}^{\mathrm{T}}\right) \tag{44d}$$
$$n = N(n_x + n_u) \tag{44e}$$

12.2 Infinite horizon

$$\underset{z\in\mathbb{R}^n}{f^\infty(z)} = \sum_{t=0}^{\infty} \frac{1}{2} x_{t+1}^{\mathrm{T}} Q_{t+1} x_{t+1} + \frac{1}{2} u_t^{\mathrm{T}} R_t u_t \tag{45}$$

12.2.1 State feedback

(45) is defined by

$$x_{t+1} = Ax_t + Bu_t \tag{46}$$
$$x_0 = \text{given} \tag{47}$$
$$Q \geq 0 \tag{48}$$
$$R > 0 \tag{49}$$

and its solution is

$$u_t = -Kx_t \tag{50}$$

where

$$K = R^{-1}B^{\mathrm{T}}P\left(I + BR^{-1}B^{\mathrm{T}}P\right)^{-1}A \tag{51}$$
$$P = Q + A^{\mathrm{T}}P\left(I + BR^{-1}B^{\mathrm{T}}P\right)^{-1}A \tag{52}$$
$$P = P^{\mathrm{T}} \geq 0. \tag{53}$$

The closed-loop system (45)–(46) is stable if (A, B) is stabilisable[1], and (A, D) is detectable[2], where $Q = D^{\mathrm{T}}D$.

12.2.2 Linear-quadratic-Gaussian control

A combination of an LQR and a Kalman filter. Usually necessary in practice, as we cannot measure all states directly.

$$\xi_{t+1} = \begin{bmatrix} x_{t+1} \\ \widetilde{x}_{t+1} \end{bmatrix} = \begin{bmatrix} A - BK & BK \\ 0 & A - K_F C \end{bmatrix} \xi_t \tag{54}$$
$$\widetilde{x}_t = x_t - \hat{x}_t \tag{55}$$
$$\xi_0 = \text{given} \tag{56}$$

[1]All uncontrollable modes are asymptotically stable.
[2]All unobservable modes are asymptotically stable.

where ξ (xi) is the augmented state, i.e. the true states combined with the estimated states, K is the controller gain, and K_F is the Kalman gain.

Luckily, the controller and the filter are separated: They can be tuned independently.

13 Sequential quadratic programming

Used for nonlinearly constrained optimisation.

13.1 Equality-constrained NLP

$$\begin{aligned} \min \quad & f(x) && (57a) \\ \text{s.t.} \quad & c(x) = 0 && (57b) \end{aligned}$$

A QP approximation is solved at each iteration to compute direction p_k:

$$\begin{aligned} \min_p \quad & f_k + \nabla f_k^{\mathrm{T}} p + \frac{1}{2} p^{\mathrm{T}} \nabla^2_{xx} \mathcal{L}_k p && (58a) \\ \text{s.t.} \quad & A_k p + c_k = 0 && (58b) \end{aligned}$$

13.2 Inequality-constrained NLP

$$\begin{aligned} \min \quad & f(x) && (59a) \\ \text{s.t.} \quad & c_i(x) = 0, \quad i \in \mathcal{E} && (59b) \\ & c_i(x) \geq 0, \quad i \in \mathcal{I} && (59c) \end{aligned}$$

QP approximation:

$$\begin{aligned} \min_p \quad & f_k + \nabla f_k^{\mathrm{T}} p + \frac{1}{2} p^{\mathrm{T}} \nabla^2_{xx} \mathcal{L}_k p && (60a) \\ \text{s.t.} \quad & \nabla c_i(x_k)^{\mathrm{T}} p + c_i(x_k) = 0, \quad i \in \mathcal{E} && (60b) \\ & \nabla c_i(x_k)^{\mathrm{T}} p + c_i(x_k) \geq 0, \quad i \in \mathcal{I} && (60c) \end{aligned}$$

13.3 Merit functions

Used for backtracking line search (finding a step length α). Must evaluate both the decrease in $f(x)$ and constraint violation.

Most common: l_1-penalty function:

$$\phi_1(x;\mu) = f(x) + \mu \sum_{i \in \mathcal{E}} |c_i(x)| + \mu \sum_{i \in \mathcal{I}} \left[c_i(x)\right]^- \tag{61}$$

where

$$\big[c_i(x)\big]^- := \max(0, -c_i(x)). \tag{62}$$

Exactness: A merit function is *exact* if for all μ above some threshold, any local solution of the NLP is a local minimiser of the merit function.