Process Optimizazion: applications, methods and algorithms

Prof. Gabriele Pannocchia

Chemical Process Control Laboratory (CPCLab)
Department of Civil and Industrial Engineering
University of Pisa, Italy
Email: gabriele.pannocchia@unipi.it

GRICU/Nest PhD School – CAPE Forum 2025
Ischia (NA), Italy
September 18, 2025

Let’s start with a few questions

1. What is an optimization problem?
2. What is process optimization?

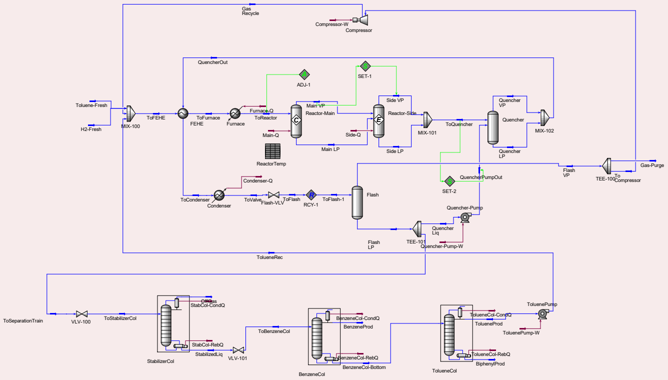
Outline

Outline

Optimization for process modeling (1/4)

Three shades of models

First-principles models (white)

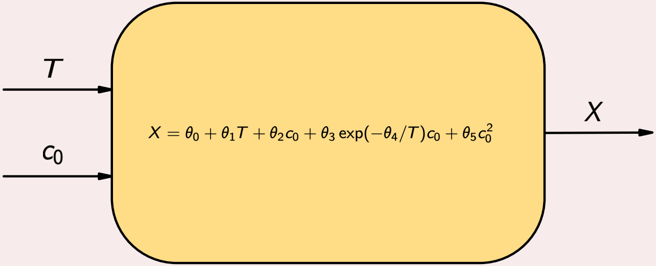


- Detailed models, based on first principles (mass, energy, momentum balances, etc.)
- Involve several parameters
- Possibly complex to solve

Optimization for process modeling (2/4)

Three shades of models

Black-box (statistical)



- Empirical models based on various expressions (linear, polynomial, etc.)
- Require experimental data from the plant
- The model parameters are obtained via regression techniques

Grey-box models

Similar to white-box models, but with (some) parameters determined from plant data

Optimization for process modeling (3/4)

Model classification

Linear vs nonlinear

- Most fundamental relations are nonlinear; however, they can be (locally) linearized
- Linear models are simple (simplistic) but often useful

Static vs dynamic

- Static models take some (static) inputs and produce the corresponding (static) output
- Dynamic models take some time-series input to produce the output at a given time
- Note that static models can also be used for dynamic purposes

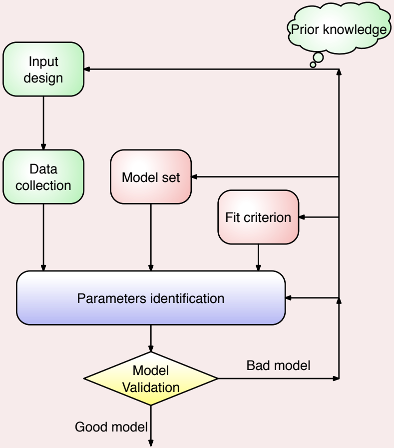
Deterministic vs stochastic

- Deterministic models produce the same output given the same set of inputs
- Stochastic models take into account the effect of some random inputs, hence generate a distribution of outputs given a set of deterministic inputs

Optimization for process modeling (4/4)

Data-driven modeling: an iterative loop

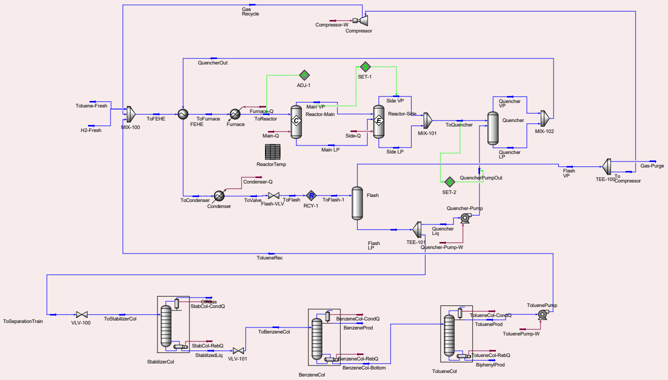
Conceptual steps to build a data-driven model



- Prior knowledge: from plant operators/engineers and some preliminary tests
- Input design: fundamental steps to cover the operating range of the plant
- Data collection: it is expensive so its duration should be limited
- Model set/Fit criterion: require experience and engineering sense
- Parameters identification: numerical algorithms are quite reliable nowadays
- Model validation: not easy to understand what went wrong

Optimization for process design

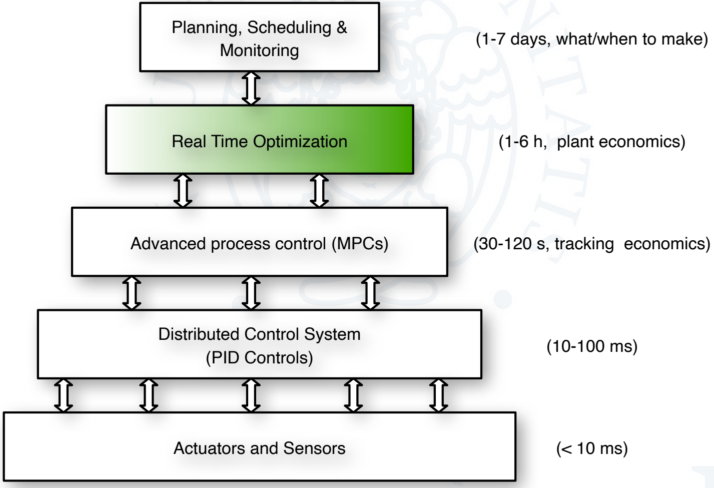
Design and optimization of a simulation model



- **Develop** the process and **simulate** it
- Choose the **process/unit parameters** (e.g., flow-rate, temperatures, pressures, etc.)
- Search for the **most profitable** (CAPEX+OPEX) process/unit parameters

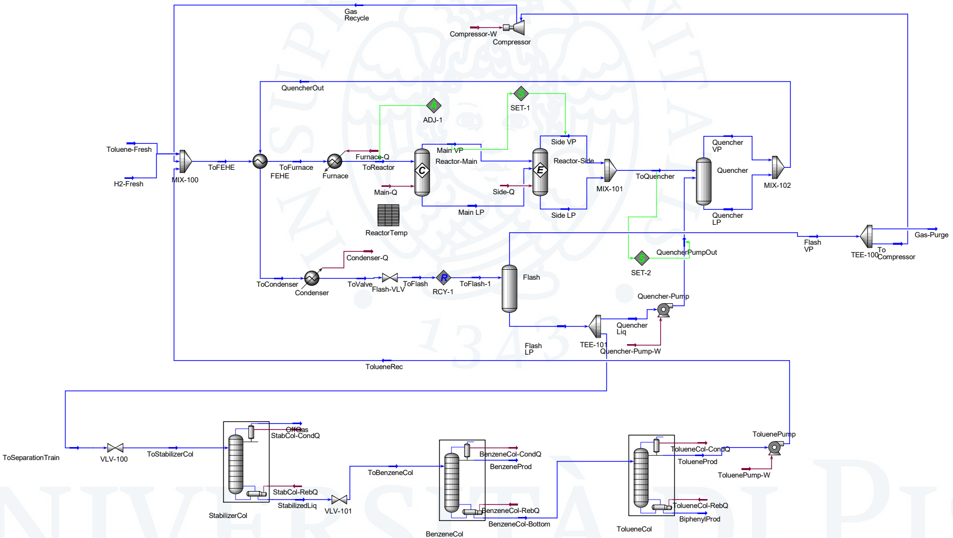
Optimization for Real-Time Optimization (1/4)

Hierarchical scheme of optimization, monitoring and control



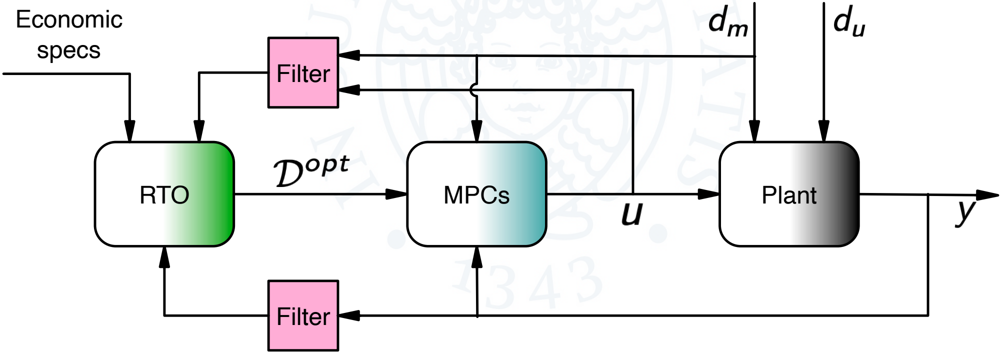
Optimization for Real-Time Optimization (2/4)

Steady-state process simulators



Optimization for Real-Time Optimization (3/4)

RTO and MPC: Bi-directional connection



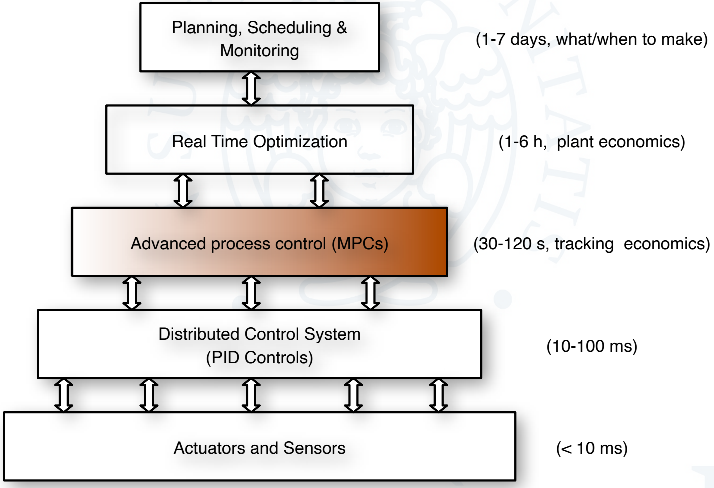
Optimization for Real-Time Optimization (4/4)

Examples of RTO and MPC variables

RTO variables		MPC variables	
Constraints	Decisions to MPC	Constraints	Manipulated setpoints (in DCS)
Reactor conversion	Desired targets	Temperature	Flow
Production rates	Min/max limits	Level	Temperature
MPC constraints	Costs/economic priorities	Composition	Pressure
		Column DP	Valve positions
		Compressor power	
		Valve positions (PID outputs)	

Optimization for Advanced Process Control (1/5)

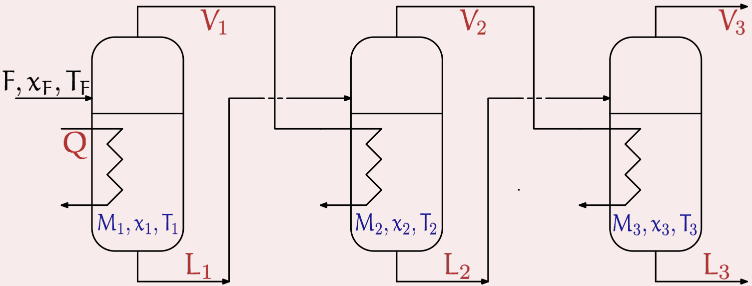
Hierarchical scheme of optimization, monitoring and control



Optimization for Advanced Process Control (2/5)

Process flow diagram of multistage evaporation

Forward feed triple effect arrangement

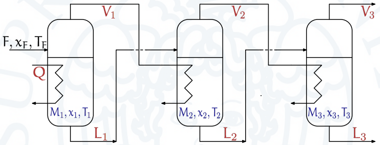


Conditions for heat integration

- For heat transfer to be possible: $T_1 > T_2 > T_3$
- This is achieved by operating at decreasing pressures: $p_1 > p_2 > p_3$

Optimization for Advanced Process Control (3/5)

Mass and energy balances of multi-stage evaporation



First evaporator

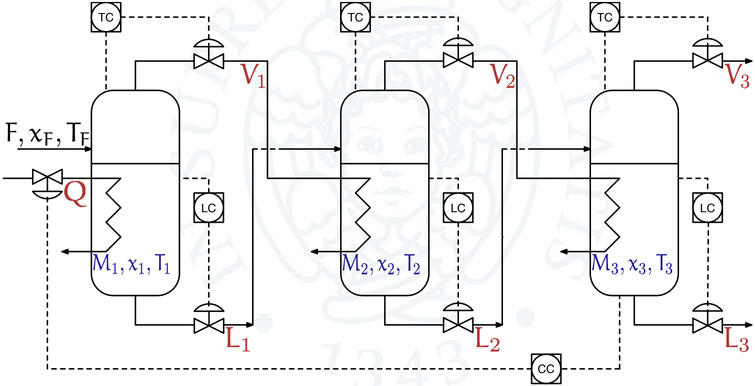
$$
\begin{cases}
\text{Overall mass balance:} & \frac{dM_1}{dt} = F - L_1 - V_1 \\
\text{Solute mass balance:} & M_1 \frac{dx_1}{dt} = F x_F + (V_1 - F)\, x_1 \\
\text{Energy balance:} & M_1 c_p \frac{dT_1}{dt} = F c_p \,(T_F - T_1) - V_1 \lambda + Q
\end{cases}
$$

$i-$th evaporator ($i = 2, 3$)

$$
\begin{cases}
\text{Overall mass balance:} & \frac{dM_i}{dt} = L_{i-1} - L_i - V_i \\
\text{Solute mass balance:} & M_i \frac{dx_i}{dt} = L_{i-1} x_{i-1} + (V_i - L_{i-1})\, x_i \\
\text{Energy balance:} & M_i c_p \frac{dT_i}{dt} = L_{i-1} c_p \,(T_{i-1} - T_i) - V_i \lambda_i + V_{i-1} \lambda_{i-1}
\end{cases}
$$

Optimization for Advanced Process Control (4/5)

Conventional control architecture



Decentralized control structure

- Each controlled variable is **paired** with a manipulated variable
- A **SISO PID controller** is used for each pairing

Multivariable system features

- **Interactions**: each manipulated variable affects more than one controlled variable
- **Directionality**: it is easier to "move" the system in certain "directions" than in others
- Both manipulated and controlled variables should **satisfy** certain (safety, quality, operation) **constraints**

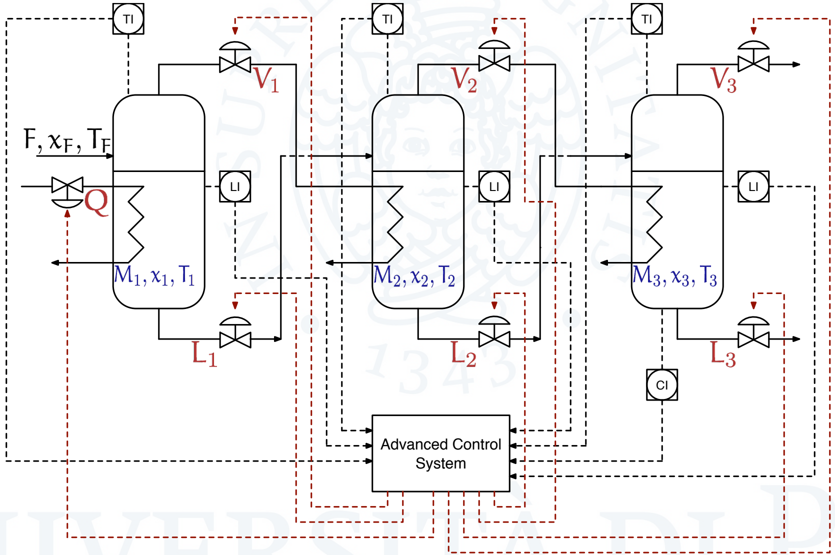
Opportunities

These needs coupled with economic reasons call(ed) for the adoption of advanced optimization-based control techniques, able to:

- Control all variables adjusting all manipulated variables **simultaneously**
- **Minimize** energy and cost
- **Respect** constraints

Optimization for Advanced Process Control (5/5)

Advanced control architecture



Outline

What is an optimization problem?

Optimization is used in all quantitative sciences and engineering. Its aim is to minimize (or maximize) an objective function depending on decision variables subject to constraints

The three ingredients

1. $w \in \mathbb{R}^n$: vector of variables
2. $F : \mathbb{R}^n \to \mathbb{R}$: scalar objective function to be minimized
3. $G : \mathbb{R}^n \to \mathbb{R}^m$: vector function of m **equality** constraints
   $H : \mathbb{R}^n \to \mathbb{R}^p$: vector function of p **inequality** constraints

Optimization problem

$$\min_{w \in \mathbb{R}^n} F(w) \quad \text{s.t.}$$

$$\begin{aligned} G_i(w) &= 0 \qquad i = 1, \ldots, m \\ H_i(w) &\leq 0 \qquad i = 1, \ldots, p \end{aligned}$$

- Only in few special cases a closed-form solution exists
- When F, G, H are nonlinear and smooth, we speak of a nonlinear programming problem (NLP)
- Usually we need iterative algorithms to find an approximate solution
- In RTO and APC, the problem depends on parameters that change every sampling time

Example of an optimization problem

Example and standard form

- Starting problem

$$\min_{w_1, w_2} (w_1 - 2)^2 + (w_2 - 1)^2 \qquad \text{subject to} \begin{cases} w_1^2 - w_2 & \leq 0 \\ w_1 + w_2 & \leq 2 \end{cases}$$

- Rewritten in standard form

$$F(w) = (w_1 - 2)^2 + (w_2 - 1)^2, \qquad w = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix}$$

$$G(w) = [\,], \qquad H(w) = \begin{bmatrix} g_1(w) \\ g_2(w) \end{bmatrix} = \begin{bmatrix} w_1^2 - w_2 \\ w_1 + w_2 - 2 \end{bmatrix}$$

Classification of optimization problems

Possible criteria of classification

1. Optimization problems can be: **constrained** or **unconstrained**
2. Optimization problems can be: **continuous** or **discrete** (or mixed)
3. Optimization problems can be: **global** or **local**

Our focus

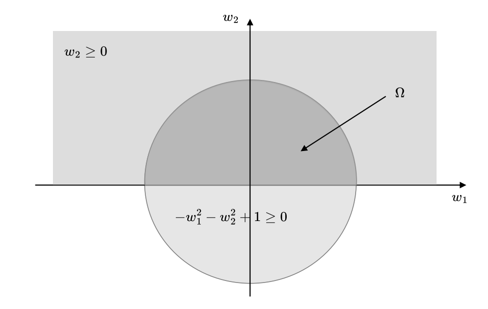
We here address, mainly, **constrained**, **continuous**, **local** optimization problems and algorithms

Optimization problems: Basic definitions

Feasible set and minimizers

The feasible set

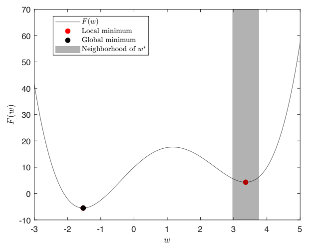
The feasible set of the optimization problem is:
$\Omega\{w \in \mathbb{R}^n | G(w) = 0, \; H(w) \leq 0\}$



The feasible set is the intersection of the two grey areas (halfspace and circle)

Global and local minimizer

- A point $w^\star \in \Omega$ is a global minimizer if $F(w^\star) \leq F(w)$ for all $w \in \Omega$
- A point $w^\star \in \Omega$ is a local minimizer if there exists a neighborhood $\mathcal{N}$ of $w^\star$ such that $F(w^\star) \leq F(w)$ for all $w \in \mathcal{N}$



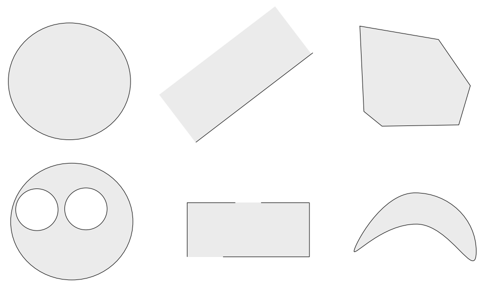
$$F(w) = \frac{1}{2}w^4 - 2w^3 - 3w^2 + 12w + 10$$

Optimization problems: Basic definitions

Convexity

Convex sets

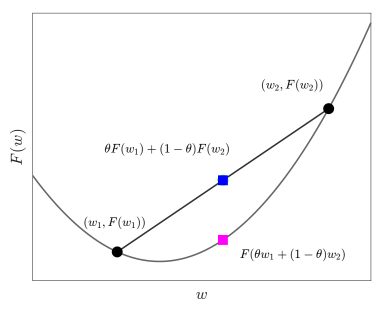
A set Ω is convex if for any $w_1, w_2 \in \Omega$ and any $\theta \in [0, 1]$, it holds: $\theta w_1 + (1 - \theta) w_2 \in \Omega$



Convex functions

A function $F : \Omega \to \mathbb{R}$ is convex if for any $w_1, w_2 \in \Omega$ and any $\theta \in [0, 1]$, it holds:

$$F(\theta w_1 + (1 - \theta) w_2) \leq \theta F(w_1) + (1 - \theta) F(w_2)$$



Optimization problems: Basic definitions

Convex optimization problems

A convex optimization problem

$$\min_{w \in \mathbb{R}^n} F(w) \quad \text{s.t.}$$
$$G(w) = 0$$
$$H(w) \leq 0$$

in which:

- $F(\cdot)$ is a convex function
- $\Omega\{w \in \mathbb{R}^n | G(w) = 0,\ H(w) \leq 0\}$ is a convex set

Properties of convex problems

- For convex problems, every locally optimal solution is globally optimal
- First order conditions are necessary and sufficient
- "...in fact, the great watershed in optimization isn't between linearity and nonlinearity, but convexity and nonconvexity." R. T. Rockafellar, SIAM Review, 1993

Unconstrained optimization: preliminary definitions

Definitions

- A point $w^\star \in \mathbb{R}^n$ is a global minimum if $F(w^\star) \leq F(w)$ for all $w \in \mathbb{R}^n$ (or in the domain for which $F(\cdot)$ is defined)
- A point $w^\star \in \mathbb{R}^n$ is a local minimum if there exists a neighborhood $\mathcal{N}$ of $w^\star$ such that $F(w^\star) \leq F(w)$ for all $w \in \mathcal{N}$

Fundamental result 1

If $F(\cdot)$ is differentiable, then for any $p \in \mathbb{R}^n$:

$$F(w+p) = F(w) + \nabla F(w+tp)^T p \quad \text{with } t \in (0,1)$$

Fundamental result 2

If $F(\cdot)$ is twice differentiable, then for any $p \in \mathbb{R}^n$:

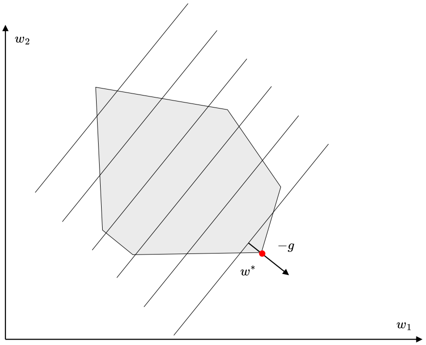
$$F(w+p) = F(w) + \nabla F(w)^T p + \frac{1}{2} p^T \nabla^2 F(x+tp) p \quad \text{with } t \in (0,1)$$

Optimization problems: common classes

Class 1: Linear programming (LP)

Linear Program (LP)

$$\min_{w \in \mathbb{R}^n} g^\top w \quad \text{s.t.}$$
$$Aw - b = 0$$
$$Cw - d \leq 0$$



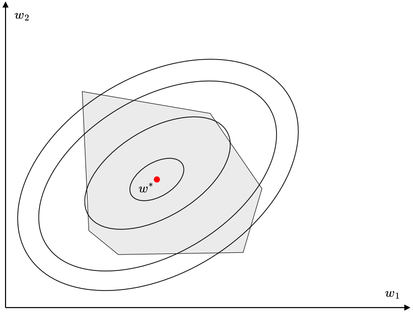
- Convex optimization problem
- 1947: simplex method by G. Dantzig
- A solution is always at a vertex of the feasible set (possibly a whole facet if nonunique)
- Very mature and reliable, with many applications in planning/scheduling problems

Optimization problems: common classes

Class 2: Quadratic programming (QP)

Quadratic Program (QP)

$$\min_{w \in \mathbb{R}^n} \frac{1}{2} w^\top Q w + g^\top w \quad \text{s.t.}$$
$$Aw - b = 0$$
$$Cw - d \leq 0$$



- Convex optimization problem when Q positive definite
- Solved online in Linear Model Predictive Control
- Many good solvers
- Subproblems in nonlinear optimization

Optimization problems: common classes

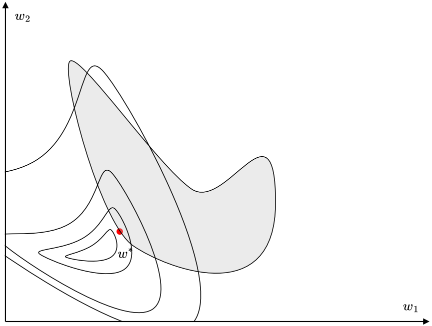
Class 3: Nonlinear programming (NLP)

NonLinear Program (NLP)

$$\min_{w \in \mathbb{R}^n} F(w) \quad \text{s.t.}$$

$$G(w) = 0$$

$$H(w) \leq 0$$



- Can be convex or nonconvex
- Solved online in Nonlinear Model Predictive Control
- Several good solvers, usually based on Newton-type algorithms

Optimization problems: common classes

Class 4: Continuous-Time Optimal Control (CTOC)

Optimal Control Problem (OCP)

$$\min_{x(\cdot),u(\cdot)} \int_0^T \ell_c(x(t),u(t))dt + V(x(T)) \quad \text{s.t.}$$

$$x(0) = \bar{x}$$

$$\dot{x}(t) = f_c(x(t),u(t)), \quad t \in [0,T]$$

$$h(x(t),u(t)) \leq 0, \quad t \in [0,T]$$

$$r(x(T)) \leq 0$$

- Decision variables $x(\cdot)$, $u(\cdot)$ in infinite dimensional function space
- Infinitely many constraints $(t \in [0,T])$
- More general dynamic model can be used (DAE, PDE, nonsmooth or stochastic ODE)
- Can be convex or nonconvex
- All or some components of $u(t)$ may take integer values (mixed-integer OCP)

Constrained optimization: example 1

Solve

$$\min\ w_1 + w_2 \qquad \text{s. t.}\ \ w_1^2 + w_2^2 = 2$$

Standard notation, feasibility region and solution

- In standard notation: $F(w) = w_1 + w_2$, $G(w) = [2 - w_1^2 - w_2^2]$, $H(w) = [\,]$
- Feasibility region: circle of radius $\sqrt{2}$, only the border
- Solution: $w^\star = [-1, -1]^T$

Observation

$$\nabla F(w^\star) = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \qquad \nabla G_1(w^\star) = \begin{bmatrix} 2 \\ 2 \end{bmatrix}$$

The normal vector at the constraint, $\nabla G_1(w^\star)$, is parallel to the cost function gradient $\nabla F(w^\star)$:

$$\nabla F(w^\star) + \mu_1^\star \nabla G_1(w^\star) = 0 \qquad \text{with } \mu_1^\star = -\frac{1}{2}$$

Constrained optimization: example 2

Solve

$$\min\ w_1 + w_2 \qquad \text{s. t.}\ \ w_1^2 + w_2^2 \leq 2$$

Standard notation, feasibility region and solution

- In standard notation: $F(w) = w_1 + w_2$, $G(w) = [\,]$, $H(w) = [w_1^2 + w_2^2 - 2]$
- Feasibility region: circle of radius $\sqrt{2}$, including the interior
- Solution: $w^\star = [-1, -1]^T$

Observation

$$\nabla F(w^\star) = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \qquad \nabla H_1(x^\star) = \begin{bmatrix} -2 \\ -2 \end{bmatrix}$$

The normal vector at the constraint, $\nabla H_1(x^\star)$, is parallel to the cost function gradient $\nabla f(x^\star)$:

$$\nabla F(x^\star) + \lambda_1^\star \nabla H_1(w^\star) = 0 \qquad \text{with } \lambda_1^\star = \frac{1}{2}$$

Optimality conditions for constrained optimization

KKT conditions

Lagrangian Function

$$\mathcal{L}(w, \mu, \lambda) = F(w) + \mu^\top G(w) + \lambda^\top H(w)$$

Karush-Kuhn-Tucker (KKT) necessary conditions

If w^* is a (local) minimizer, there exists vectors μ^* and λ^* such that (w^*, μ^*, λ^*) satisfies:

$$\begin{aligned}
\nabla_w \mathcal{L}(w^*, \mu^*, \lambda^*) &= 0 \\
G(w^*) &= 0 \\
H(w^*) &\leq 0 \\
\lambda^* &\geq 0 \\
\lambda_i^* H_i(w^*) &= 0, \quad i = 1, \ldots, p
\end{aligned}$$

Outline

Unconstrained optimization: general strategies

Iterative algorithms

Starting from an initial point w_0, optimization algorithms generate a sequence of iterates $\{w_k\}_{k\geq 0}$, and terminate when

1. No further progresses are possible or
2. It "seems" that the current iterate is a good approximation of the solution

Step computation

In deciding how to move from w_k to w_{k+1}, numerical algorithms use information about f at the current iterate w_k and often at previous iterates $(w_{k-1}, w_{k-2}, \ldots)$

Fundamental strategies

- "Line search" methods
- "Trust region" methods

"Line search" strategy (1/2)

Basic idea

The "line search" strategy chooses a step direction p_k and then try to minimize $F(\cdot)$ over the segment connecting w_k and $w_k + p_k$, i.e.

$$\min_{\alpha > 0} F(w_k + \alpha p_k)$$

Most common directions

- The most intuitive direction is the so-called "steepest descent", i.e.:

$$p_k = -\nabla F(w_k)$$

- Newton step (often the most effective):

$$p_k^N = -[\nabla^2 F(w_k)]^{-1} \nabla F(w_k)$$

- "quasi-Newton" step (cheaper to compute):

$$p_k = -B_k^{-1} \nabla F(w_k) \qquad \text{with } B_k \approx \nabla^2 F(w_k)$$

“Line search” strategy (2/2)

Sufficient decrease condition

- Instead of solving the unidimensional problem

$$\min_{\alpha>0} F(w_k + \alpha p_k)$$

  we look for a step length α_k that satisfies certain conditions
- A **sufficient decrease** condition is called Wolfe condition:

$$F(w_k + \alpha_k p_k) \leq F(w_k) + c_1 \alpha_k \nabla F(w_k)^T p_k$$

  with c_1 small ($\sim 10^{-4}$)

Backtracking

1. Set $\alpha_k = \alpha_{\max}$ (usually $\alpha_{\max} = 1$)
2. Check the Wolfe condition. If satisfied, go to Step 4. Else, go to Step 3
3. Decrease α_k, and go to Step 2
4. Define new iterate: $w_{k+1} = w_k + \alpha_k p_k$

“Trust region” strategy (1/2)

Basic idea: a model function

- Build a model function $m_k(\cdot)$ which approximates $F(\cdot)$ near w_k
- Compute the step that minimizes $m_k(\cdot)$ with a trust region

$$\min_{p_k} m_k(w_k + p_k) \qquad \text{where } w_k + p_k \text{ is in the trust region}$$

Quadratic model and step evaluation

- The most common model function is quadratic. Hence, we solve:

$$\min_{p_k \in \mathbb{R}^n} m_k(p_k) = F(w_k) + \nabla F(w_k)^T p_k + \frac{1}{2} p_k^T B_k p_k \qquad \text{s.t. } \|p_k\| \leq \Delta_k$$

- Very often we use an approximate solution (a.k.a. Cauchy point). Let $g_k = \nabla F(w_k)$. Then:

$$p_k = -g_k \frac{\Delta_k}{\|g_k\|} \tau^\star, \qquad \text{with } \tau^\star = \begin{cases} 1 & \text{if } g_k^T B_k g_k \leq 0 \\ \min\left(\frac{\|g_k\|^3}{\Delta_k g_k^T B_k g_k}, 1\right) & \text{otherwise} \end{cases}$$

”Trust region” strategy (2/2)

Step acceptance/rejection and trust-region size changes

- In order to decide whether the step is acceptable or not, evaluate:

$$\rho_k = \frac{F(w_k) - F(w_k + p_k)}{m_k(0) - m_k(p_k)}$$

- Rules of thumb:
  - if $\rho_k \leq \frac{1}{4}$: reject p_k, i.e. $w_{k+1} = w_k$, and reduce the trust-region $\Delta_{k+1} = \frac{1}{4}\Delta_k$
  - if $\rho_k \geq \frac{1}{4}$, accept p_k, i.e. $w_{k+1} = w_k + p_k$
  - if $\rho_k \geq \frac{3}{4}$ and $\|p_k\| = \Delta_k$, accept p_k, i.e. $w_{k+1} = w_k + p_k$, and enlarge the trust-region $\Delta_{k+1} = 2\Delta_k$

Linear programming (LP): introduction

LP in **standard form**

$$\min_{w} g^T x \qquad \text{subject to } Aw = b,\; w \geq 0$$

where: $g \in \mathbb{R}^n$, $w \in \mathbb{R}^n$, $A \in \mathbb{R}^{p \times n}$, $b \in \mathbb{R}^p$ and $\text{rank}(A) = p$

Optimality conditions

- Lagrangian function: $\mathcal{L}(w, \mu, \lambda) = g^T w + \mu^T (Aw - b) - \lambda^T w$
- If $w^\star$ is solution of the linear program, then:

$$\begin{aligned} g + A^T \mu^\star - \lambda^\star &= 0 \\ Aw^\star &= b \\ w^\star &\geq 0 \\ \lambda^\star &\geq 0 \\ w_i^\star \lambda_i^\star &= 0, \qquad i = 1, \ldots, n \end{aligned}$$

Linear programming (LP): the simplex method

The "base points"

A point $w \in \mathbb{R}^n$ is a base point if

- $Aw = b$ and $w \geq 0$
- At most p components of w are nonzero
- The columns of A corresponding to the nonzero elements are linearly independent

Fundamental aspects of the simplex method

- Base points are vertices of the feasibility region
- The solution is a base point
- The simplex method iterates from a base point w_k to another one w_{k+1}, and stops when all components of λ_k are nonnegative
- When a component of λ_k is negative, a new base point w_{k+1} in which the corresponding element of w_k is nonzero is selected

Quadratic Programming (QP)

Introduction

Standard form

$$\min_{w \in \mathbb{R}^n} \frac{1}{2} w^T Q w + g^T w$$

subject to:

$$Aw - b = 0$$
$$Cw - d \leq 0$$

The **active set**

$$\mathcal{A}(w) = \{i \in \{1, \ldots, m\} \mid (Cw - d)_i = 0\}$$

Quadratic Programming (QP)

Optimality conditions

Lagrangian function

$$\mathcal{L}(x, \mu, \lambda) = \frac{1}{2} w^T Q w + g^T w + \mu^T (Aw - b) + \lambda^T (Cw - d)$$

Optimality conditions (KKT)

$$\begin{aligned} Qw^\star + g + A^T \mu^\star + C^T \lambda^\star &= 0 \\ Aw^\star - b &= 0 \\ Cw^\star - d &\leq 0 \\ \lambda_i^\star &\geq 0 \qquad i = 1, \dots, m \\ \lambda_i &= 0 \qquad i \notin \mathcal{A}(w^\star) \end{aligned}$$

Quadratic Programming (QP)

Active set methods for convex QP problem

Fundamental steps

1. Given a feasible w_k, we evaluate its active set and build:
$$C_{ac} = C[i,:], \qquad d_{ac} = d[i], \qquad \text{for all } i \in \mathcal{A}(w_k)$$
2. Express next iterate as $w_k + p_k$ and solve the KKT linear system:
$$\begin{bmatrix} Q & A^T & C_{ac}^T \\ A & 0 & 0 \\ C_{ac} & 0 & 0 \end{bmatrix} \begin{bmatrix} p_k \\ \mu \\ \lambda_{ac} \end{bmatrix} = - \begin{bmatrix} Qw_k + g \\ Aw_k - b \\ C_{ac}w_k - d_{ac} \end{bmatrix}$$
3. If $\|p_k\| \leq \rho$ and $\lambda_{ac} \geq 0$, stop: w_k is the solution.
4. If $\|p_k\| \leq \rho$ and $\lambda_{ac} \not\geq 0$ for some components, remove from the new active set $\mathcal{A}(w_{k+1})$ such indices. $k \leftarrow k+1$ and go to 2.
5. If $\|p_k\| > \rho$, define $w_{k+1} = w_k + \alpha_k p_k$, where α_k is the largest scalar in $(0, 1]$ such that no inequality constraint is violated. When a blocking constraint is found, it is included in the new active set $\mathcal{A}(w_{k+1})$. $k \leftarrow k+1$ and go to 1.

NonLinear Programming

Problem formulation

$$\min_{w \in \mathbb{R}^n} F(w) \quad \text{s.t.}$$
$$G(w) = 0$$
$$H(w) \leq 0$$

Lagrangian Function

$$\mathcal{L}(w, \mu, \lambda) = F(w) + \mu^\top G(w) + \lambda^\top H(w)$$

NLP solution algorithms

Sequential Quadratic Programming (SQP)

By linearizing all functions and setting $w^+ = w^k + \Delta w$, $\mu^+ = \mu^k + \Delta\mu$, $\lambda^+ = \lambda^k + \Delta\lambda$, we obtain the KKT conditions of the following Quadratic Program (QP):

Inequality constrained Quadratic Program within SQP method

$$\min_{\Delta w \in \mathbb{R}^n} \frac{1}{2} w^\top A^k w + \nabla F(w^k)^\top \Delta w \quad \text{s.t.}$$

$$G(w^k) + \nabla G(w^k) \Delta w = 0$$

$$H(w^k) + \nabla H(w^k) \Delta w \leq 0$$

with: $A^k = \nabla_w^2 \mathcal{L}(w^k, \mu^k, \lambda^k)$

Its solution delivers the next SQP iterate: $\Delta w^k, \mu^+, \lambda^+$

Three basic families

- Dynamic Programming / Hamilton-Jacobi-Bellmann Equation
- Indirect Methods / Calculus of Variations / Pontryagin's Maximum Principle
- Direct Methods, i.e., discretization combined with nonlinear programming

Pros and Cons

Dynamic programming

+ Searches whole state space (global minimum)
+ Optimization feedback control precomputed
+ Analytical solution possible in some cases
- But, in general, intractable ("curse of dimensionality")

Indirect methods

+ Boundary value problems with only $2n_x$ ODE
- Need explicit expressions for controls
- ODE strongly nonlinear and unstable
- inequalities lead to ODE with state dependent switches

Direct methods

+ Can use state-of-the-art NLP solvers
+ Can treat inequality constraints
- Obtain only suboptimal/approximate solutions

Nowadays, most widely used

Solution methods for Optimal Control Problems

Direct methods

Continuous-Time OCP

$$\min_{x(\cdot),u(\cdot)} \int_0^T \ell_c(x(t),u(t))dt + V(x(T)) \quad \text{s.t.}$$

$$x(0) = \bar{x}$$

$$\dot{x}(t) = f_c(x(t),u(t)), \quad t \in [0,T]$$

$$h(x(t),u(t)) \leq 0, \quad t \in [0,T]$$

$$r(x(T)) \leq 0$$

Direct methods first discretize, then optimize

Discrete-Time OCP

$$\min_{x(\cdot),u(\cdot)} \sum_{k=0}^{N-1} \ell(x_k,u_k) + V(x_N) \quad \text{s.t.}$$

$$x_0 = \bar{x}$$

$$x_{k+1} = f(x_k,u_k), \quad k = 0,\dots,N-1$$

$$h(x_k,u_k) \leq 0, \quad k = 0,\dots,N-1$$

$$r(x_N) \leq 0$$

Variables $x = (x_0,\dots,x_N)$, $u = (u_0,\dots,u_{N-1})$ can be lumped in vector $w = (x,u) \in \mathbb{R}^n$

$$\min_{w\in\mathbb{R}^n} F(w) \quad \text{s.t.}$$

$$G(w) = 0$$

$$H(w) \leq 0$$

Outline

Conclusions

General comments

- Numerical optimization is **powerful** and **useful** for may areas of science and engineering
- The most **reliable** and **efficient** algorithms are for linear programming (LP) and quadratic programming (QP) problems
- **Nonlinear** programming (NLP) problems are solved by means of Sequential Quadratic Programming (SQP) algorithms, which are very effective (**global solution** can be found) only for **convex** problems

Numerical optimization in Aspen Hysys/UniSim Design

- UniSim Design contains a general purpose **SQP optimizer**
- One can use the optimizer to find the **operating conditions** which minimize (or maximize) an objective function subject to **constraints**