MASTER OF SCIENCE IN LIFE SCIENCES

Hes·so
Haute Ecole Spécialisée de Suisse occidentale
Fachhochschule Westschweiz
University of Applied Sciences and Arts Western Switzerland

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

Laboratoire d'Automatique

Modeling, Regression and Optimization

Dr. Julien Billeter

Laboratoire d'Automatique
Ecole Polytechnique Fédérale de Lausanne (EPFL)

julien.billeter@epfl.ch

swissuniversities

MLS-S03 | 2015-2016

Table of Contents

A Survival Kit of Linear Algebra (Vocabulary)

- Scalars (written in *italics*)
  dimension (1×1)

- Vectors (written in **lowercase boldface**)
  dimension $(n \times 1)$ = n-dim array (column vector)

- Matrices (written in **UPPERCASE BOLDFACE**)
  dimension $(n \times m)$ = array of n rows and m columns

A Survival Kit of Linear Algebra (Grammar)

- Scalar multiplication $\alpha\mathbf{v}$ or $\alpha\mathbf{M}$
- Transposition $\mathbf{v}^{\mathrm{T}}$ or $\mathbf{M}^{\mathrm{T}}$
- Addition $\mathbf{u} + \mathbf{v}$ or $\mathbf{M} + \mathbf{N}$
- Vector and matrix multiplication $\mathbf{u}^{\mathrm{T}}\mathbf{v}$ or $\mathbf{M}\,\mathbf{N}$
- Inverse (existence of the identity) $\mathbf{M}^{-1}\mathbf{M} = \mathbf{M}\,\mathbf{M}^{-1} = \mathbf{I}$
- Rank $\mathrm{rank}(\mathbf{M})$
- Null space (or kernel) $\mathbf{M}\,\ker(\mathbf{M}) = \mathbf{0}$
- Rank-nullity theorem $\dim(\mathbf{M}) = \mathrm{rank}(\mathbf{M}) + \mathrm{nullity}(\mathbf{M})$

1. Dynamic and Static Models: Definitions

- Physical/chemical models are based on laws of conservation
- States variables: mass, concentration, temperature…
- Dynamic models use balance equations of **differential nature** (continuity equation, mole balances, heat balances) to describe the evolution of states **over time**
- Static models use physical laws (state equations) of **algebraic nature** (equilibrium relationships, rate expressions) to describe state variables at **one particular time**
- Combinations of dynamic and static models usually form physical/chemical models

1.1. Formulation of **Dynamic Models**

- **Balance equations**

$$Acc(t) = in(t) - out(t) + gen(t) - cons(t)$$

- **Conservation of mass**

Lavoisier (F-Chemist, 1743 - guillotined in 1794)

  - Balance equations: mass, **volume**, **numbers of moles**, **concentrations**, mole/mass/volume fractions…

- **Conservation of energy**

Joule (UK-Physicist, 1707-1783)

  - Balance equations: **energy**, **temperature**

1.1. Formulation of **Static Models** (State Equations)

- **Gas:** Ideal gas law (+ other derived state equations)
  Avogadro (I-Physicist, 1776-1856), Clapeyron (F-Physicist, 1779-1864)
  - Pressure, temperature, volume, amount of substance
- **Liquid:** Raoult's and Henry's laws (+ other derived equations)
  Raoult (F-Physicist, 1830-1901), Henry (UK-Physicist, 1774-1836)
  - Pressure, mole fraction/concentration, (volume, density)
- **Chemical reaction:** Kinetic rate law, Arrhenius/Eyring Equation
  Arrhenius (S-Chemist, 1859-1901), Trautz (D-Chemist, 1880-1960), Lewis (UK-Chemist, 1885-1956), Evans (UK-Chemist, 1904-1952), Eyring (US-Chemist, 1901-1981), Polanyi (UK-Mathematician, 1891-1976)
  - Reaction rate, equilibrium constants
- **Spectroscopy:** Beer's law
  Bouguer (F-Physicist, 1698-1758), Lambert (CH-Math., 1728-1777), Beer (D-Chemist, 1825-1863)
  - Absorbance, absorptivities, concentration

1.1. Method for Formulating a Problem

- **Structure** the problem (draw a sketch!)
- **Write** the equations (process/plant model)
- **Identify** the model parameters
- **Validate** the model (possible model mismatch?)

1.1. Example of Formulation

- Let consider a dynamic open reactor (with inlets and outlets) with the reaction scheme:

$$\mathrm{R1}: A + B \rightarrow C$$

$$\mathrm{R2}: A + C \rightarrow D$$

- Formulate a dynamic model describing the **total mass**, as well as the **numbers of moles** and **concentrations** of all species $(A, B, C, D, Solvent)$
- Formulate a generic expression of a dynamic model **valid for all types of reactors using matrix notation**

1.1. Expressions for **Isothermal** Chemical Reactors

- Numbers of moles

$$\dot{\mathbf{n}}(t) = \mathbf{N}^{\mathrm{T}} V(t)\mathbf{r}\left(\frac{\mathbf{n}(t)}{V(t)}\right) + \mathbf{C}_{in}\mathbf{q}_{in}(t) - \frac{q_{out}(t)}{V(t)}\mathbf{n}(t), \qquad \mathbf{n}(0) = \mathbf{n}_0$$

- Concentrations

$$\dot{\mathbf{c}}(t) \approx \mathbf{N}^{\mathrm{T}}\mathbf{r}(\mathbf{c}(t)) + \mathbf{C}_{in}\frac{\mathbf{q}_{in}(t)}{V(t)} - \frac{\sum_{i=1}^{p} q_{in,i}(t)}{V(t)}\mathbf{c}(t), * \qquad \mathbf{c}(0) = \mathbf{c}_0$$

$$\dot{\mathbf{c}}(t) = \frac{\dot{\mathbf{n}}(t)}{V(t)}, \quad \text{with } V(t) = \frac{m(t)}{\rho(\mathbf{c}(t))}, \qquad \mathbf{c}(0) = \mathbf{c}_0$$

- Total mass

$$\dot{m}(t) = \mathrm{diag}(\mathbf{M}_w)^{\mathrm{T}}\,\dot{\mathbf{n}}(t), \qquad m(0) = m_0$$

- Total volume

$$\dot{V}(t) = \frac{1}{\rho(t)}\sum_{i=1}^{p}\rho_{in,i}\,q_{in,i}(t) - q_{out}(t) - V(t)\frac{\dot{\rho}(t)}{\rho(t)}, \qquad V(0) = V_0$$

- Heat of reaction

$$q(t) = V(t)(-\Delta\mathbf{h}_r)^{\mathrm{T}}\,\mathbf{r}\left(\frac{\mathbf{n}(t)}{V(t)}\right), \quad \text{discounted from all other thermal effects} \qquad q(0) = 0$$

* : if (A1) the density is constant and (A2) the density of the inlet flows equals the density of the mixture

1.1. Expressions for **non-Isothermal** Chemical Reactors

- Numbers of moles

$$\dot{\mathbf{n}}(t) = \mathbf{N}^{\mathrm{T}} V(t)\mathbf{r}\left(\frac{\mathbf{n}(t)}{V(t)}, T(t)\right) + \mathbf{C}_{in}\mathbf{q}_{in}(t) - \frac{q_{out}(t)}{V(t)}\mathbf{n}(t), \qquad \mathbf{n}(0) = \mathbf{n}_0$$

- Concentrations

$$\dot{\mathbf{c}}(t) = \frac{\dot{\mathbf{n}}(t)}{V(t)}, \quad \text{with } V(t) = \frac{m(t)}{\rho(\mathbf{c}(t), T(t))}, \qquad \mathbf{c}(0) = \mathbf{c}_0$$

- Total volume

$$\dot{V}(t) = \frac{1}{\rho(t)}\sum_{i=1}^{p} \rho_{in,i}\, q_{in,i}(t) - q_{out}(t) - V(t)\frac{\dot{\rho}(t,T(t))}{\rho(t)}, \qquad V(0) = V_0$$

- Heat of reaction

$$q(t) = V(t)(-\Delta\mathbf{h}_r)^{\mathrm{T}}\,\mathbf{r}\left(\frac{\mathbf{n}(t)}{V(t)}, T(t)\right),$$ discounted from other thermal effects $q(0) = 0$

- Temperature

$$\dot{T}(t) = \frac{\dot{q}(t)}{m(t)\, c_p(t)},$$ discounted from all other thermal effects $T(0) = T_0$

1.1. Exercise of Formulation

- Let consider a fed-batch reactor filled with A and G, and fed with B, whose reaction scheme is:

$$A + 2B \rightarrow C \leftrightarrows D$$

$$A + C \rightarrow E + F$$

$$E + G \rightarrow 2H$$

- Formulate a dynamic model describing the state variables $\mathbf{n}(t)$ and $\mathbf{c}(t)$ of all species **using their generic matrix expressions**.
- What is the relation between the Mw's of all the species? What is the minimal number of Mw's you need to know all of them?

1.2. Integration of Dynamic Models

$$\dot{\mathbf{x}}(t) \coloneqq \frac{d}{dt}\mathbf{x}(t) = \mathbf{f}\big(t, \mathbf{x}(t)\big)$$

- Most nonlinear 1st order ODEs are not integrable analytically and require to be integrated **numerically**

$$\frac{d}{dt}\mathbf{x}(t) \approx \frac{\mathbf{x}(t+h) - \mathbf{x}(t)}{h} \Rightarrow \boxed{\mathbf{x}(t+h) = \mathbf{x}(t) + h\,\mathbf{f}\big(t, \mathbf{x}(t)\big)}$$

- Numerical integration methods are **explicit** if $\mathbf{f}$ is evaluated at t or **implicit** if $\mathbf{f}$ is evaluated at $t + h$
- Integration methods are **adaptative** if h is adapted over time to keep the integration error under a certain threshold
- Methods: **Euler's methods**, **Runge-Kutta's methods** (RK)

1.2. Euler's methods of integration

- **Explicit** Euler's method *

$$\mathbf{x}(t+h) = \mathbf{x}(t) + h\,\mathbf{f}\big(t, \mathbf{x}(t)\big)$$

Since $\mathbf{f}\big(t, \mathbf{x}(t)\big)$ is estimated at t, given $\mathbf{x}$ at time t allows integrating this equation forward

- **Implicit** Euler's method

$$\mathbf{x}(t+h) = \mathbf{x}(t) + h\,\mathbf{f}\big(t+h, \mathbf{x}(t+h)\big)$$

If $\mathbf{x}(t+h)$ cannot be factorized on the lhs, a numerical method (see Chap.1.3) is used to solve this equation at each time $t+h$.

* *Euler (CH-Mathematician, 1707-1783)*

1.2. Runge-Kutta's (RK) methods of integration

- Runge-Kutta's general scheme *

$$\mathbf{x}(t+h) = \mathbf{x}(t) + h \sum_{i=1}^{s} b_i \, \mathbf{k}_i$$

$$\mathbf{k}_i = \mathbf{f}\left(t + c_i h, \mathbf{x}(t) + h \sum_{j=1}^{s} a_{i,j} \, \mathbf{k}_j\right)$$

h the *step size*, s the number of *stages*,

$\mathbf{b}$ the s-dim vector of *weighting factors* $\left(\sum_{j=1}^{s} b_i = 1\right)$,

$\mathbf{c}$ the s-dim vector of *nodes*,

$\mathbf{A}$ an s-dim matrix of *coefficients* with $\left(\sum_{j=1}^{s} a_{i,j} = c_i\right)$,

i.e. the sum of each ith row of $\mathbf{A}$ equals c_i.

* *Runge (D-Math., 1856-1927), Kutta (D-Math., 1867-1944)*

1.2. **Explicit** 4 stages RK (RK4)

$$\mathbf{x}(t+h) = \mathbf{x}(t) + h \sum_{i=1}^{s} b_i \, \mathbf{k}_i \,, \qquad \mathbf{k}_i = \mathbf{f}\left(t + c_i h, \mathbf{x}(t) + h \sum_{j=1}^{s} a_{i,j} \, \mathbf{k}_j \right)$$

- RK4 explicit integration scheme:

$$\mathbf{A} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ \frac{1}{2} & 0 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}, \mathbf{b} = \begin{bmatrix} \frac{1}{6} \\ \frac{1}{3} \\ \frac{1}{3} \\ \frac{1}{6} \end{bmatrix}, \mathbf{c} = \begin{bmatrix} 0 \\ \frac{1}{2} \\ \frac{1}{2} \\ 1 \end{bmatrix}$$

$$\boxed{\mathbf{x}(t+h) = \mathbf{x}(t) + h\left(\tfrac{1}{6}\,\mathbf{k}_1 + \tfrac{1}{3}\,\mathbf{k}_2 + \tfrac{1}{3}\,\mathbf{k}_3 + \tfrac{1}{6}\,\mathbf{k}_4\right)}$$

with
$$\mathbf{k}_1 = \mathbf{f}\left(t, \mathbf{x}(t)\right)$$
$$\mathbf{k}_2 = \mathbf{f}\left(t + \tfrac{1}{2}h, \mathbf{x}(t) + h\,\tfrac{1}{2}\,\mathbf{k}_1\right),$$
$$\mathbf{k}_3 = \mathbf{f}\left(t + \tfrac{1}{2}h, \mathbf{x}(t) + h\,\tfrac{1}{2}\,\mathbf{k}_2\right)$$
$$\mathbf{k}_4 = \mathbf{f}\left(t + h, \mathbf{x}(t) + h\,\mathbf{k}_3\right)$$

1.2. **Implicit** 2 stages RK (RK2)

$$\mathbf{x}(t+h) = \mathbf{x}(t) + h \sum_{i=1}^{s} b_i \, \mathbf{k}_i \,, \qquad \mathbf{k}_i = \mathbf{f}\left(t + c_i h, \mathbf{x}(t) + h \sum_{j=1}^{s} a_{i,j} \, \mathbf{k}_j \right)$$

- RK2 implicit integration scheme (**trapezoidal rule**):

$$\mathbf{A} = \begin{bmatrix} 0 & 0 \\ \frac{1}{2} & \frac{1}{2} \end{bmatrix}, \mathbf{b} = \begin{bmatrix} \frac{1}{2} \\ \frac{1}{2} \end{bmatrix}, \mathbf{c} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$\mathbf{x}(t+h) = \mathbf{x}(t) + h\left(\tfrac{1}{2}\,\mathbf{k}_1 + \tfrac{1}{2}\,\mathbf{k}_2\right) \quad [1]$$

with $\mathbf{k}_1 = \mathbf{f}\big(t, \mathbf{x}(t)\big)$ [2]

$\mathbf{k}_2 = \mathbf{f}(t+h, \mathbf{x}(t) + h\tfrac{1}{2}\,\mathbf{k}_1 + h\tfrac{1}{2}\,\mathbf{k}_2)$ [3]

Using [1] to substitute $h\tfrac{1}{2}\,\mathbf{k}_2$ by $\mathbf{x}(t+h) - \mathbf{x}(t) - h\tfrac{1}{2}\,\mathbf{k}_1$ in [3] yields

$$\mathbf{k}_2 = \mathbf{f}(t+h, \mathbf{x}(t) + h\tfrac{1}{2}\,\mathbf{k}_1 + \mathbf{x}(t+h) - \mathbf{x}(t) - h\tfrac{1}{2}\,\mathbf{k}_1) = \mathbf{f}(t+h, \mathbf{x}(t+h))$$

$$\boxed{\mathbf{x}(t+h) = \mathbf{x}(t) + \tfrac{1}{2}\,h\big[\mathbf{f}\big(t, \mathbf{x}(t)\big) + \mathbf{f}(t+h, \mathbf{x}(t+h))\big]}$$

1.2. Explicit **Adaptative** RK45

$$\mathbf{x}_4(t+h) = \mathbf{x}(t) + h\left(\frac{25}{216}\,\mathbf{k}_1 + \frac{1\,408}{2\,565}\,\mathbf{k}_2 + \frac{2\,197}{4\,104}\,\mathbf{k}_3 - \frac{1}{5}\,\mathbf{k}_4\right)$$
$$\mathbf{x}_5(t+h) = \mathbf{x}(t) + h\left(\frac{16}{135}\,\mathbf{k}_1 + \frac{6\,656}{12\,825}\,\mathbf{k}_3 + \frac{28\,561}{56\,430}\,\mathbf{k}_4 - \frac{9}{50}\,\mathbf{k}_5 + \frac{2}{55}\,\mathbf{k}_6\right)$$

with $\mathbf{k}_i = \mathbf{f}\big(t, \mathbf{x}(t)\big)$ given by $\mathbf{c}$ and $\mathbf{A}$ of the RK45 method

$$\boldsymbol{\varepsilon}_{LTE,45} := |\mathbf{x}_5(t+h) - \mathbf{x}_4(t+h)|\,,\ \boldsymbol{\varepsilon}_{LTE,45,rel} := \frac{\boldsymbol{\varepsilon}_{LTE,45}}{|\mathbf{x}_5(t+h)|}$$

RK45-Fehlberg* Method: *LTE = Local Truncation Error*

1. Compute $\mathbf{x}_4(t_i + h)$ and $\mathbf{x}_5(t_i + h)$
2. Compute $\boldsymbol{\varepsilon}_{LTE,45}$ and $\boldsymbol{\varepsilon}_{LTE,45,rel}$
   a) $\boldsymbol{\varepsilon}_{min} \le \boldsymbol{\varepsilon}_{LTE,45} \le \boldsymbol{\varepsilon}_{max} \Rightarrow$ step is acceptable, $\mathbf{x}(t_i + h) = \mathbf{x}_4(t_i + h),\ t_i + h \to t_i$
   b) $\boldsymbol{\varepsilon}_{LTE,45} < \boldsymbol{\varepsilon}_{min} \Rightarrow$ step is too small, return to point 1. with $h := \min(2h, h_{max})$
   c) $\boldsymbol{\varepsilon}_{LTE,45} > \boldsymbol{\varepsilon}_{max} \Rightarrow$ step is too large, return to point 1. with $h := max\left(\frac{1}{2}h, h_{min}\right)$

* *Fehlberg (D-Mathematician, 1911-1990)*

1.2. Common Integration Problems

- Problem: Discontinuities due to sudden events

  Solution: Integration by regions or use an events function

- Problem: Rates of different magnitude **at different times**

  Solution: Use a **variable step-size** ODE solver

- Problem: Rates of different magnitude **at the same time (stiff problem)**

  Solution: Use a **stiff** ODE solver (implicit method)

1.2. MATLAB ODE solvers

- **Explicit** adaptative (variable stepsize) ODE solvers:
  - `ode45` RK45-Felhberg method
  - `ode23` RK23 method
- **Implicit** adaptative **stiff** ODE solvers:
  - `ode15s` BDF methods *
  - `ode23t` Trapezoidal method
- **MATLAB ode solver call**:
  - `[tout,yout] = ode45(@odefun,tspan,y0,options,...)`
- **MATLAB ode options call**:
  - `options = odeset('name1',value1,'name2',value2)`

* Backward Differention Formula (BDF):

$\mathcal{O}(1): \mathbf{x}(t+h) = \mathbf{x}(t) + h\,\mathbf{f}\big(t+h, \mathbf{x}(t+h)\big)$ (Implicit Euler's method!)

$\mathcal{O}(2): \mathbf{x}(t+2h) = \frac{4}{3}\mathbf{x}(t+h) + \frac{1}{3}\mathbf{x}(t) + \frac{2}{3}h\,\mathbf{f}\big(t+2h, \mathbf{x}(t+2h)\big)$

BDFs are stable up to $\mathcal{O}(6)$ only!

1.2. Exercise about Numerical Integration

- Let consider a fed-batch reactor filled with A (and solvent) and fed with B during the 1st phase of the reaction, whose scheme is

$$\text{R1: } A + B \rightarrow C, \quad r_1(t) \coloneqq k_1 c_A(t) c_B(t)$$
$$\text{R2: } A + C \rightarrow D, \quad r_2(t) \coloneqq k_2 c_A(t) c_C(t)$$

- Formulate the dynamic model describing the state variables $\mathbf{n}(t)$ and $\mathbf{c}(t)$ of all species (including the solvent) **using the generic matrix expressions**
- Derive an expression for the volume assuming the additivity of volumes
- Integrate by regions this dynamic model using MATLAB `ode45`

1.3. Solution of Static Models

$$\underset{\mathbf{x}(t)}{\text{find}} \quad \mathbf{f}\big(\mathbf{x}(t)\big) = \mathbf{0}$$

- This problem consists in finding the root of $\mathbf{f}$
- This problem has an analytical solution if $\mathbf{f}$ is explicit in $\mathbf{x}$
- For implicit $\mathbf{f}$, an iterative method is required to find $\mathbf{x}(t)$

1.3. Bisection method (double false position)

Method:

1. Start by selecting two endpoints $a \coloneqq x_{i,0}$, $b \coloneqq x_{i,1}$, which bracket the root

$$x_{i,k+1} \coloneqq \frac{a+b}{2}, \ \forall k = 1, 2, \dots$$

2. Adjust a or b based on the following test:
   a) $f(a)f(x_{i,k+1}) < 0$ (opposite signs) $\Rightarrow b \coloneqq x_{i,k+1}$
   b) > 0 (same signs) $\Rightarrow a \coloneqq x_{i,k+1}$
   c) $= 0 \Rightarrow x_{i,k+1}$ is the root of f

Drawback: Estimation error is halved at each iteration

Order of convergence: 1 (linear)

1.3. Unidimensional Secant Method

Method (1 300 BC!):

1. Start with *two* initial points $x_{i,0}$ and $x_{i,1}$ (bracketing the root) and construct a line (secant) through the points $\{x_{i,0}, f(x_{i,0})\}$ and $\{x_{i,1}, f(x_{i,1})\}$, whose equation is

$$y_i = f(x_{i,1}) + f'(x_{i,1})(x_i - x_{i,1}) \text{ with } f'(x_{i,1}) \approx \frac{f(x_{i,1}) - f(x_{i,0})}{x_{i,1} - x_{i,0}} \text{ (backward)}$$

2. Find the zero of the secant, $y_i = 0 \Longrightarrow x_i = x_{i,1} - \frac{f(x_{i,1})}{f'(x_{i,1})}$.
3. $x_i \rightarrow x_{i,2}$, construct a line (secant) through $\{x_{i,1}, f(x_{i,1})\}$ and $\{x_{i,2}, f(x_{i,2})\}$ and find its zero... Hence, the recurrent relation:

$$x_{i,k+1} = x_{i,k} - \frac{f(x_{i,k})}{f'(x_{i,k})}, \ \forall k = 1, 2, \ldots \text{ with } f'(x_{i,k}) \approx \frac{f(x_{i,k}) - f(x_{i,k-1})}{x_{i,k} - x_{i,k-1}}$$

1.3. Multi-dimensional Secant Method

Method:

$$\mathbf{x}_{k+1} = \mathbf{x}_{i,k} - \mathbf{J}^{+}\left(\mathbf{x}_{i,k}\right)\mathbf{f}\left(\mathbf{x}_{i,k}\right),\ \forall k = 1, 2, \ldots$$

$$\text{with } \mathbf{J}(\mathbf{x}_k) := \frac{\mathbf{f}(\mathbf{x}_k) - \mathbf{f}(\mathbf{x}_{k-1})}{\mathbf{x}_k - \mathbf{x}_{k-1}} \text{ (backward)}$$

- Order of convergence: $\frac{1+\sqrt{5}}{2} \approx 1.618$ (less than quadratic!)
- The secant method does not check if two successive estimates $\mathbf{x}_k$ and $\mathbf{x}_{k-1}$ bracket the root (source of failure); **Solution**: use a double false position approach to guarantee the bracketing of the root
- Pseudo-inverse of the Jacobian $\mathbf{J}$ is $\mathbf{J}^{+} = (\mathbf{J}^{\mathrm{T}}\,\mathbf{J})^{-1}\,\mathbf{J}^{\mathrm{T}}$

1.3. Unidimensional Newton-Raphson method

Newton-Raphson* Method:

1. Start with *one* initial guess $x_{i,0}$ and construct the tangent using a truncated Taylor expansion, whose equation is

$$y_i = f(x_{i,0}) + f'(x_{i,0})(x_i - x_{i,0}) \text{ with } f'(x_{i,0}) \approx \frac{f(x_{i,0}+\delta x_{i,0}) - f(x_{i,0})}{\delta x_{i,0}}$$

(finite differences or analytical)

2. Find the zero of the tangent, $y_i = 0 \Longrightarrow x_i = x_{i,1} - \frac{f(x_{i,0})}{f'(x_{i,0})}$.

3. $x_{i,2} \rightarrow x_{i,1}$, construct the tangent and find its zero...
   Hence, the recurrent relation:

$$x_{i,k+1} = x_{i,k} - \frac{f(x_{i,k})}{f'(x_{i,k})}, \ \forall k = 0, 1, \ldots \text{ with } f'(x_{i,k}) \approx \frac{f(x_{i,k}+\delta x_{i,k}) - f(x_{i,k})}{\delta x_{i,k}}$$

$\delta x_{i,k} \longrightarrow 0$

* *Newton (UK-Math./Phys., 1643-1727), Raphson (UK-Mathematician, 1710-1761)*

1.3. Multi-dimensional Newton-Raphson method

Method:

$$\mathbf{x}_{k+1} = \mathbf{x}_{i,k} - \mathbf{J}^{+}(\mathbf{x}_{i,k})\,\mathbf{f}(\mathbf{x}_{i,k}),\ \forall k = 0, 1, 2, \ldots$$

with $\mathbf{J}(\mathbf{x}_k) := \frac{\mathbf{f}(\mathbf{x}_k + \boldsymbol{\delta}\mathbf{x}_k) - \mathbf{f}(\mathbf{x}_k)}{\boldsymbol{\delta}\mathbf{x}_k}$ (finite differences or analytical solution)

- Order of convergence: 2 (quadratic!)
- The Newton-Raphson method is sensitive to the initial guess…
- **Quasi-Newton methods**: the Jacobian $\mathbf{J}$ is only calculated for the initial guess (not even always!) and updated algebraically over the iterations (e.g. BFGS * algorithm)

* Broyden (UK-Math., 1933-2011), Fletcher (UK-Math., born in 1939), Goldfarb (US-Math., born in 1949), Shanno (US-Math., born in 1936)

1.3. MATLAB Root finders

- **Unidimensional** root finder:
  - `fzero` combination of bisection, secant and newton-raphson
- **Multidimensional** root finder
  - `None` except if formulated as an optimization problem $\min_{\mathbf{x}} |\mathbf{f}(\mathbf{x})|$
- **MATLAB fzero call**:
  - `[x,fval]=fzero(@fun,x0,options,...)`
- **MATLAB optim options call**:
  - `options = optimset('name1',value1,'name2',value2)`

2. Regression Problems

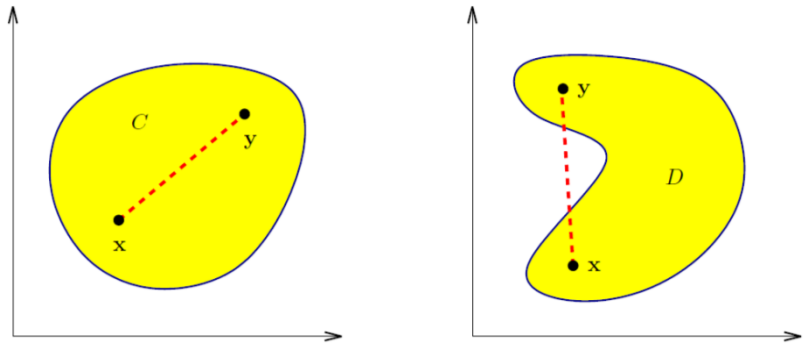
- **Regression problems**
  Mathematical problems in which *modeled* data are fitted to *measured* data by *estimating* the parameters (**parameter estimation**) of a *postulated* model (**model identification**).

- **Dichotomy of nested problems**
  Origin/cause: The model is *identified* simultaneously as the model parameters are *estimated*
  Consequence: in case of no good fit, is it a **problem of parameter estimation or of model identification** (wrong postulated model)?

- **Least squares problems**
  These problems are part of the family of quadratic problems of the general form: $\min_{\mathbf{x}} \frac{1}{2}\mathbf{x}^{\mathrm{T}}\mathbf{H}\,\mathbf{x} + \mathbf{f}^{\mathrm{T}}\mathbf{x}$
  QP problems: quadratic problems with linear equality/inequality constraints

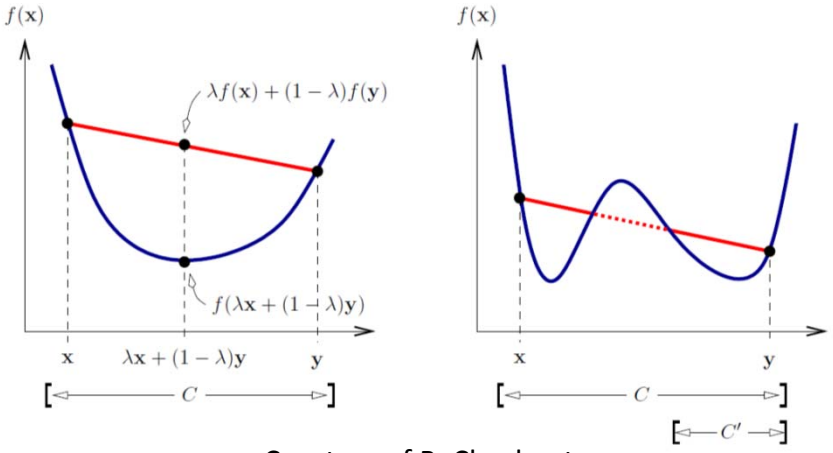
2. Notion of Convex Set and Convex Function

- Convex Set:



Courtesy of B. Chachuat

- Convex function:



Courtesy of B. Chachuat

2. Necessary Conditions of Optimality (NCO)

- **1st order NCO:** If $\mathbf{x}^*$ is a local minimum of a function $\phi: \mathcal{C} \longrightarrow \mathbb{R}$, then

$$\boxed{\nabla\phi(\mathbf{x}^*) = \mathbf{J}^{\mathrm{T}}(\mathbf{x}^*) = \mathbf{0} \Longleftrightarrow \mathbf{x}^* \text{ is a stationary point}}$$

gradient Jacobian

- **2nd order NCO:** If $\mathbf{x}^*$ is a local minimum of $\phi: \mathcal{C} \longrightarrow \mathbb{R}$, then

$$\boxed{\nabla^2\phi(\mathbf{x}^*) = \mathbf{H}(\mathbf{x}^*) \succcurlyeq \mathbf{0} \text{ (positive semidefinite)}}$$

Positive semidefinitness:

$$\mathbf{Hv} = \boldsymbol{\lambda}\mathbf{v} \Rightarrow (\mathbf{H} - \boldsymbol{\lambda}\mathbf{I}) = \mathbf{0} \Rightarrow p(\boldsymbol{\lambda}) = \det(\mathbf{H} - \boldsymbol{\lambda}\mathbf{I}) = 0 \text{ and all } \lambda'\text{s (eigenvalues)} \geq 0$$

- 1st and 2nd order NCO form **sufficient conditions of optimality (SCO)** if ϕ is a convex function defined on a convex set $\mathcal{C}$.

2. Exercise on NCO's

- Let consider

$$y = f(x) = 4x^5 - 8x^4 - 5x^3 + 10x^2 + x + 5$$
$$\text{on the domain } x \in \mathcal{C} = [-1, 2]$$

- Find analytically the stationary points of $y = f(x)$ using
  - 1st NCO: find x^* s.t. $J = \frac{df(x)}{dx} = 0$
- Qualify the stationary points (minima/maxima) using
  - 2nd NCO: find x^* s.t. $H = \frac{d^2 f(x)}{d^2 x} \geq 0$ (minimum)

$$\text{find } x^* \text{ s.t. } H = \frac{d^2 f(x)}{d^2 x} \leq 0 \text{ (maximum)}$$

- Is there another way to qualify the stationary points ?

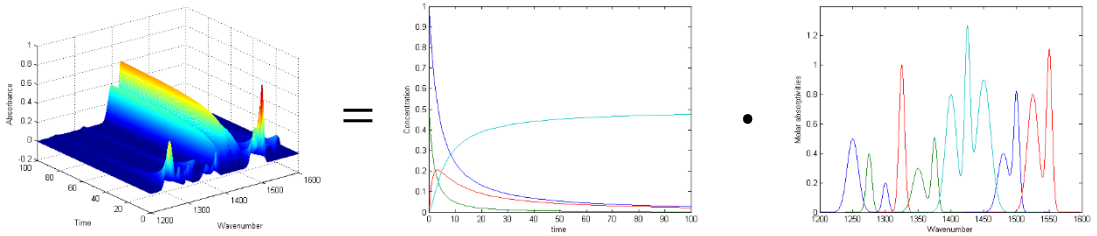
2.1. Concept of Output Function

- An **output function** translates the values of the internal states (numbers of moles/concentrations, not always *directly* measurable) into indirect measured quantities (**outputs**).
- Typical **indirect** measurements are
  - Spectroscopic measurements
    absorbance, reflectance/scattering data (isothermal cond.)
  - Calorimetric measurements
    as heat-flow data (isothermal conditions) or
    as heat-flow or temperature data (non-isothermal cond.)
  - Other indirect measurements
    as HPLC, GC, conductometric data, refraction index data…

2.1. Absorbance Data (Beer's law)

- "The absorbance of a solution is proportional to the product of its concentration and the distance light travels through it"
  Beer (D-Chemist, 1825-1863), Lambert (CH-Math., 1728-1777), Bouguer (F-Physicist, 1698-1758).

$$\mathbf{Y} = \mathbf{C}\,\mathbf{A}$$



$\mathbf{Y}$ $\mathbf{C}$ $\mathbf{A}$

with

$\mathbf{Y}(H \times W)$ the absorbance at H times and L wavelength/wavenumbers,
$\mathbf{C}(H \times S) = [\mathbf{c}^{\mathrm{T}}(t_0);\ \mathbf{c}^{\mathrm{T}}(t_1);\ \ldots;\ \mathbf{c}^{\mathrm{T}}(t_H)]$ the concentrations, and
$\mathbf{A}(S \times W) = \ell[\mathbf{a}(w_1)\ \mathbf{a}(w_2)\ \ldots\ \mathbf{a}(w_L)]$ the absorptivities/pure spectra

Unit conversion: $\mathrm{Abs} \coloneqq -\log_{10}(\mathrm{Trans})$, with $\mathrm{Trans} \coloneqq \frac{I}{I_0}$

2.1. Calorimetric Data

- Calorimetric signal* under isothermal or non-isothermal conditions

$$\mathbf{q} = \mathbf{R}_v(-\Delta\mathbf{h}_r)$$

with

$\mathbf{q}(H \times 1)$ the heat flow at H times (univariate data),

$\mathbf{R}_v(H \times R) = [V(t_0)\,\mathbf{r}^{\mathrm{T}}(t_0);\ \ldots\ ;\ V(t_H)\,\mathbf{r}^{\mathrm{T}}(t_H)]$ the reaction rates

$\Delta\mathbf{h}_r(R \times 1)$ the reaction enthalpies

* discounted from all other thermal effects

2.1. Formulation of Linear Regression Problems

- A systems of linear equations can be written in matrix

$$\begin{cases} a_{1,1}x_1 + \cdots + a_{1,n}x_n = y_1 \\ \vdots \quad \ddots \quad \vdots \quad \vdots \\ a_{m,1}x_1 + \cdots + a_{m,n}x_n = y_n \end{cases} \quad \Longrightarrow \quad \mathbf{A}\,\mathbf{x} = \mathbf{y}$$

with $\mathbf{A}(m \times n)$, and $\mathbf{x}(n \times 1)$ the **regressors** and $\mathbf{y}(m \times 1)$ the **regressands**

- The number of solutions of this linear system is:

∞	when $m < n$	underdetermined system
1	$m = n$	determined system
∞	$m > n$	overdetermined system

2.1. Exercise of Formulation

- Formulate a **matrix equation** for the following linear system:

$$\begin{cases} x_1 + x_2 + x_3 = 3 & (1) \\ x_1 - x_2 - x_3 = -1 & (2) \\ x_1 - x_2 + x_3 = 1 & (3) \\ x_1 + x_2 - x_3 = 0 & (4) \end{cases}$$

- How many solutions for the system of Eqs. 1 – 2?
- " " " " " " of Eqs. 1 – 3?
- " " " " " " of Eqs. 1 – 4?

2.1. Formulation of Nonlinear Regression Problems

- A **nonlinear regression problem** consists in
  - **Minimizing an objective function** $\phi(\cdot)$ (or cost function)
  - Expressing ϕ as a **difference between measured & modeled quantities**
  - Postulating a **dynamic model** $\mathbf{f}_{x,d}(\cdot)$ and (possibly) **a static model** $\mathbf{f}_{x,s}(\cdot)$
  - Using an **output (signal) model** $\mathbf{f}_{y}(\cdot)$
  - **Adjusting model parameters** $\boldsymbol{\theta}$ such that ϕ is minimal

$$[\boldsymbol{\theta}_1^* \quad \boldsymbol{\theta}_2^*] = \arg\left\{ \min_{\boldsymbol{\theta}_1,\boldsymbol{\theta}_2} \; \phi\left(\tilde{\mathbf{y}}(t), \mathbf{y}(t,\boldsymbol{\theta}_1,\boldsymbol{\theta}_2)\right) \right\} \qquad \text{Objective function}$$

$$\text{s.t.} \quad \dot{\mathbf{x}}(t) = \mathbf{f}_{x,d}(\mathbf{x}(t),\boldsymbol{\theta}_1) \qquad \text{Dynamic model}$$

$$\mathbf{x}(t) = \mathbf{f}_{x,s}(\mathbf{x}(t),\boldsymbol{\theta}_1) \qquad \text{Static model}$$

$$\mathbf{y}(t,\boldsymbol{\theta}_1,\boldsymbol{\theta}_2) = \mathbf{f}_{y}(\mathbf{x}(t,\boldsymbol{\theta}_1),\boldsymbol{\theta}_2) \qquad \text{Output model}$$

with $\boldsymbol{\theta}_1$ **nonlinear** parameters and $\boldsymbol{\theta}_2$ **linear** parameters

- Nonlinear problems have to be solved iteratively!

2.1. Linear versus Nonlinear Parameters

- Let $\mathrm{f}(\boldsymbol{\theta})$ be a function depending on a vector of parameters $\boldsymbol{\theta} = [\theta_1, \ldots, \theta_i, \ldots, \theta_j, \ldots, \theta_n]^{\mathrm{T}}$.
- θ_i is a nonlinear parameter if

$$\frac{\partial}{\partial \theta_i} \mathrm{f}(\boldsymbol{\theta}) = f'(\theta_i)$$

- θ_j is a linear parameter if

$$\frac{\partial}{\partial \theta_j} \mathrm{f}(\boldsymbol{\theta}) \neq f'(\theta_j)$$

2.1. Exercise of Formulation

- Formulate a **regression problem in the least-squares sense** for a batch reactor with the following reaction:

$$A \rightarrow B \rightleftarrows C$$

$$\text{with } r(t) = k c_A(t) \text{ and } K = \frac{c_C(t)}{c_B(t)}$$

Hint: there are 2 dynamic eqs. and 2 static equations!

- The content of the reactior is measured by absorbance spectroscopy at 800, 900 and 1000 cm^{-1}, with all the species absorbing at these wavenumbers.
- How many **nonlinear** parameters in this problem?
- How many **linear** parameters in this problem?

2.2. Generalized Inverse of a Matrix

$$\mathbf{A}\ (m \times n)$$

The inverse or generalized inverse only exists if $\mathbf{A}$ is **FULL RANK**

- If $n < m$: Left pseudo-inverse, s.t. $\mathbf{A}^{+}\mathbf{A} = \mathbf{I}_n$

$$\mathbf{A}^{+} = \underbrace{(\mathbf{A}^{\mathrm{T}}\mathbf{A})}_{n \times n}{}^{-1}\,\mathbf{A}^{\mathrm{T}}\ (n \times m)$$

- If $m < n$: Right pseudo-inverse, s.t. $\mathbf{A}\,\mathbf{A}^{+} = \mathbf{I}_m$

$$\mathbf{A}^{+} = \mathbf{A}^{\mathrm{T}}\underbrace{(\mathbf{A}\,\mathbf{A}^{\mathrm{T}})}_{m \times m}{}^{-1}\ (n \times m)$$

- If $m = n$: Inverse $\mathbf{A}^{+} = \mathbf{A}^{-1}\ (n \times n)$, s.t. $\mathbf{A}\,\mathbf{A}^{-1} = \mathbf{A}^{-1}\mathbf{A} = \mathbf{I}_n$

2.2. **Univariate** Solution of Linear Regression Problems

$$\boxed{\mathbf{y} = \mathbf{A}\,\mathbf{x}}$$ with $\mathbf{y}$ $(m \times 1)$, $\mathbf{A}$ $(m \times n)$ and $\mathbf{x}$ $(n \times 1)$

- If $n < m$, $\mathbf{A}^{\mathrm{T}}\mathbf{y} = \mathbf{A}^{\mathrm{T}}\mathbf{A}\,\mathbf{x}$ (normal equation) and the least-squares solution is

$$\boxed{\mathbf{x} = \mathbf{A}^{+}\mathbf{y}}$$, since $\mathbf{A}^{+}\mathbf{A}\,\mathbf{x} = \mathbf{I}_n\,\mathbf{x}$ (left pseudo-inverse)

- If $n = m$, the unique solution is

$$\boxed{\mathbf{x} = \mathbf{A}^{-1}\mathbf{y}}$$, since $\mathbf{A}^{-1}\mathbf{A}\,\mathbf{x} = \mathbf{I}_n\,\mathbf{x}$

2.2. **Multivariate** Solution of Linear Regression Problems

$$\boxed{\mathbf{Y} = \mathbf{A}\,\mathbf{B}} \quad \text{with } \mathbf{Y}\ (m \times p),\ \mathbf{A}\ (m \times n) \text{ and } \mathbf{B}\ (n \times p)$$

- If $n < m$, $\mathbf{A}^{\mathrm{T}}\mathbf{Y} = \mathbf{A}^{\mathrm{T}}\mathbf{A}\,\mathbf{X}$ (normal equation) and the **least-squares solution** is $\boxed{\mathbf{B} = \mathbf{A}^{+}\mathbf{Y}}$, since $\mathbf{A}^{+}\mathbf{A}\,\mathbf{B} = \mathbf{I}_n\,\mathbf{B}$ (left pseudo-inverse)
- If $n < p$, $\mathbf{Y}\,\mathbf{B}^{\mathrm{T}} = \mathbf{A}\,\mathbf{B}\,\mathbf{B}^{\mathrm{T}}$ (normal equation) and the **least-squares solution** is $\boxed{\mathbf{A} = \mathbf{Y}\,\mathbf{B}^{+}}$, since $\mathbf{A}\,\mathbf{B}\,\mathbf{B}^{+} = \mathbf{A}\,\mathbf{I}_n$ (right pseudo-inverse)
- If $n = m$, **the unique solution** for $\mathbf{B}$ is $\boxed{\mathbf{B} = \mathbf{A}^{-1}\mathbf{Y}}$, since $\mathbf{A}^{-1}\mathbf{A}\,\mathbf{B} = \mathbf{I}_n\,\mathbf{B}$
- if $n = p$, the **unique solution** for $\mathbf{A}$ is $\boxed{\mathbf{A} = \mathbf{Y}\,\mathbf{B}^{-1}}$, since $\mathbf{A}\,\mathbf{B}\,\mathbf{B}^{+} = \mathbf{A}\,\mathbf{I}_n$

2.2. Example of left and right pseudo-inverse

Let consider Beer's law: $\mathbf{Y}_{(H\times W)} = \mathbf{C}_{(H\times S)}\mathbf{A}_{(S\times W)}$

- Under which condition can $\mathbf{C}$ be computed **in the least-squares sense** and what is its solution?
- Under which condition can $\mathbf{C}$ be computed **uniquely** and what is its solution?
- Under which condition can $\mathbf{A}$ be computed **in the least-squares sense** and what is its solution?
- Under which condition can $\mathbf{A}$ be computed **uniquely** and what is its solution?

2.2. MATLAB inverse and pseudo-inverse

- **Inverse (square matrix `A`)**
  - `inv(A)`
  - `(A)^-1`
- **Pseudo-inverse (non-square matrix `B`)**
  - `pinv(B)` Left or right pseudo-inverse depending on the dimensions
  - `inv(B'*B)*B' or B'*inv(B*B')`
- **Linear regression by left-pseudo inverse (between `Y` and `B`)**
  - `pinv(B)*Y`
  - `B\Y`
- **Linear regression by right-pseudo inverse (between `Y` and `B`)**
  - `Y*pinv(B)`
  - `Y/B`

2.2. Exercise: Compute **Univariate** Absorptivities

- Consider the following **absorbance** measurements at 800 cm^{-1} and the corresponding **concentrations**:

#	$\mathbf{y}$ [-]	$\mathbf{c}_A$ [mol/L]	$\mathbf{c}_B$ [mol/L]	$\mathbf{c}_C$ [mol/L]	$\mathbf{c}_D$ [mol/L]
1	2.5	1	1	1	1
2	2.2	1	1	1	0
3	1.5	1	1	0	0
4	0.8	1	0	0	1
5	1.6	1	0	1	1

- Compute the **absorptivities** at 800 cm^{-1} of A, B, C and D.
  - *Can you solve this problem with measurements #1 to #3?*
  - *Estimate the absorptivities with measurements #1 to #4 and #1 to #5.*

2.2. Exercise: Compute **Multivariate** Absorptivities

- Consider the following **absorbance** measurements at different wavenumbers and the corresponding **concentrations**:

t	$\mathbf{y}_{800\ \mathrm{cm}^{-1}}$ [-]	$\mathbf{y}_{850\ \mathrm{cm}^{-1}}$ [-]	$\mathbf{y}_{900\ \mathrm{cm}^{-1}}$ [-]	$\mathbf{y}_{950\ \mathrm{cm}^{-1}}$ [-]	$\mathbf{c}_A(t)$ [L/mol]	$\mathbf{c}_B(t)$ [L/mol]	$\mathbf{c}_C(t)$ [L/mol]	$\mathbf{c}_D(t)$ [L/mol]
0	1.000	0.350	0.150	0.150	1.00	0.50	0.00	0.00
1	1.200	0.630	0.250	0.170	0.80	0.70	0.20	0.00
2	0.960	0.660	0.360	0.190	0.60	0.50	0.30	0.10
3	0.700	0.665	0.495	0.230	0.40	0.30	0.35	0.25
4	0.410	0.610	0.670	0.300	0.20	0.10	0.30	0.50

- Compute the **pure component spectra** of A, B, C and D.
  - *Can you solve this problem with measurements at times 0 to 2?*
  - *Estimate the pure spectra with measurements at times 0 to 3 and 0 to 4.*

2.2. Exercise: Compute Concentrations from **Multivariate** Data

- Consider the following **absorbance** measurements at different wavenumbers and the corresponding **absorptivities**:

#	cm^{-1}	$\mathbf{y}$ [-]	$\mathbf{a}_A$ [L/mol]	$\mathbf{a}_B$ [L/mol]	$\mathbf{a}_C$ [L/mol]	$\mathbf{a}_D$ [L/mol]
1	800	0.650	0.5	0.1	0.1	0.1
2	850	1.450	1.0	0.5	0.1	0.1
3	900	1.525	0.5	1.0	0.5	0.1
4	950	1.100	0.1	0.5	1.0	0.5
5	1000	0.700	0.1	0.1	0.5	1.0

- Compute the **concentrations** of A, B, C and D.
  - *Can you solve this problem with measurements #1 to #3 only?*
  - *Estimate the concentrations with measurements #1 to #4 and #1 to #5.*

2.2. Exercise: Compute Reaction Enthalpies from **Univariate** Data

- Consider the following **heat flow** measurements and the corresponding **rates of reaction**:

#	$\mathbf{q}$ [W]	$\mathbf{r}_{v,1}$ [mol/s]	$\mathbf{r}_{v,2}$ [mol/s]
1	19,000	0.9	0.1
2	18,000	0.8	0.2
3	17,000	0.6	0.4

- Compute the **enthalpies of reaction**.
  - *Can you solve this problem with only measurement #1 only?*
  - *Estimate the concentrations with measurements #1 to #2 and #1 to #3.*
- Why is it impossible to compute the **rates of reaction** from **heat flow** measurements and the knowledge of **enthalpies of reaction**?

2.2. Curve Fitting

- Consider the following data points:

- Find the best curves that approximates $y_1 = f(x)$ and $y_2 = g(x)$

x	y_1	y_2
0.0	2.0460	1.4616
0.1	2.0269	2.4021
0.2	2.1111	2.1336
0.3	2.1432	2.7800
0.4	2.2335	3.0366
0.5	2.2173	3.8411
0.6	2.3010	4.2199
0.7	2.3666	5.2970
0.8	2.4330	6.2218
0.9	2.4963	7.4636
1.0	2.5026	7.9418

2.3. Reminder: Nonlinear Regression (Chapt. 2.1)

Problem:

$$\min_{\boldsymbol{\theta}} \ \phi(t, \boldsymbol{\theta})$$

$$\min_{\boldsymbol{\theta}} \frac{1}{2} \underbrace{\left(\tilde{\mathbf{y}}(t) - \mathbf{y}(t, \boldsymbol{\theta})\right)^{\mathrm{T}}}_{\mathbf{r}(\boldsymbol{\theta})^{\mathrm{T}}} \underbrace{\left(\tilde{\mathbf{y}}(t) - \mathbf{y}(t, \boldsymbol{\theta})\right)}_{\mathbf{r}(\boldsymbol{\theta})}$$

$$\min_{\boldsymbol{\theta}} \ \frac{1}{2} \ \mathbf{r}(\boldsymbol{\theta})^{\mathrm{T}} \mathbf{r}(\boldsymbol{\theta}) = \min_{\boldsymbol{\theta}} \ \frac{1}{2} ssq(\boldsymbol{\theta})$$

$$\text{s.t. dynamic, static and output (signal) models}$$

2.3. Reminder: Meaning of the Gradient (Analysis)

- The gradient $\boldsymbol{\nabla}\phi(\mathbf{x}) = \mathbf{J}^{\mathrm{T}}(\mathbf{x})$ of a multi-variable function $\phi(\mathbf{x})$ indicates the **tangent** at point $\mathbf{x}$.
- The gradient $\boldsymbol{\nabla}\phi(\mathbf{x})$ is a vector that points towards the direction of an increase of the function ϕ.
- Hence, following the opposite direction of the gradient, namely $-\boldsymbol{\nabla}\phi(\mathbf{x})$, allows pointing towards a direction of decreasing ϕ.
- This is the mathematical basis of the steepest descent method for minimizing a function, with $\boxed{\mathbf{J}(\boldsymbol{\theta}) \coloneqq \frac{\partial \mathbf{r}(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}}$, $\mathbf{r}$ the residuals and $\boldsymbol{\theta}$ the adjustable parameters.

2.3. Steepest (Gradient) Descent Method

Method:

- **Recurrence relation for finding the minimum of ϕ**

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \gamma_k \, \boldsymbol{\Delta\theta}_k \;\text{ with }\; \boldsymbol{\Delta\theta}_k = -\mathbf{J}(\boldsymbol{\theta}_k)^{\mathrm{T}} \mathbf{r}(\boldsymbol{\theta}_k)$$

- The Gradient (Jacobian) direction does not give an indication about the length of the step to apply.
- To correct that, the stepsize is adapted using the parameter γ_k which is computed according a **Line Search Method** (e.g. Goldstein-Armijo's method) to maximize the stepsize, while minimizing ϕ.

2.3. Minimizing using the 1st NCO (Chapt. 2)

- 1st order NCO: If $\boldsymbol{\theta}^*$ is a local minimum of ϕ, then

$$\boxed{\boldsymbol{\nabla}\phi(\boldsymbol{\theta}^*) = \mathbf{J}^{\mathrm{T}}(\boldsymbol{\theta}^*) = \mathbf{0} \Leftrightarrow \boldsymbol{\theta}^* \text{ is a stationary point}}$$

- The 1st order NCO gives a method to find a fixed (stationary) point of ϕ as follows:
  - Make a *truncated* Taylor development of the residuals as

$$\mathbf{r}(\boldsymbol{\theta}_k + \boldsymbol{\Delta\theta}_k) = \mathbf{r}(\boldsymbol{\theta}_k) + \mathbf{J}(\boldsymbol{\theta}_k)\boldsymbol{\Delta\theta}_k + \mathcal{O}(2) \text{ with } \mathbf{J}(\boldsymbol{\theta}_k) := \frac{\partial \mathbf{r}(\boldsymbol{\theta}_k)}{\partial \boldsymbol{\theta}_k}$$

  - Minimizing $\mathbf{r}(\boldsymbol{\theta}_k + \boldsymbol{\Delta\theta}_k)$ implies the stepsize:

$$\boldsymbol{\Delta\theta}_k = -\mathbf{J}(\boldsymbol{\theta}_k)^{+}\mathbf{r}(\boldsymbol{\theta}_k)$$

which is **Newton-Raphson** applied to the residuals! (cf. Chapt. 1.3.)

2.3. Newton-Gauss method

Method:

- **Recurrence relation for finding the minimum of ϕ**

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \boldsymbol{\Delta\theta}_k \text{ with } \boldsymbol{\Delta\theta}_k = -\mathbf{J}(\boldsymbol{\theta}_k)^{+}\mathbf{r}(\boldsymbol{\theta}_k)$$

- The Newton-Gauss stepsize is known to be usually too long and the decrease in the residuals is not always guaranteed.
- To correct that, the stepsize is usually adapted using a **Line Search Method** to work at the highest stepsize, while minimizing the residuals.

2.3. Levenberg-Marquardt Modification

- The Levenberg-Marquardt* modification allows switching between Newton-Gauss and the steepest descent method
- **Recurrence relation for finding the minimum of ϕ**

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \boldsymbol{\Delta\theta}_k \text{ with } \boldsymbol{\Delta\theta}_k = -(\mathbf{H}(\boldsymbol{\theta}_k) + \lambda_k \mathbf{I})\, \mathbf{J}(\boldsymbol{\theta}_k)^{\mathrm{T}} \mathbf{r}(\boldsymbol{\theta}_k)$$

with $\mathbf{H}(\boldsymbol{\theta}_k) \approx \mathbf{J}(\boldsymbol{\theta}_k)^{\mathrm{T}} \mathbf{J}(\boldsymbol{\theta}_k)$ according to Newton-Gauss

- $\lambda_k \geq 0$ is the Marquardt parameter

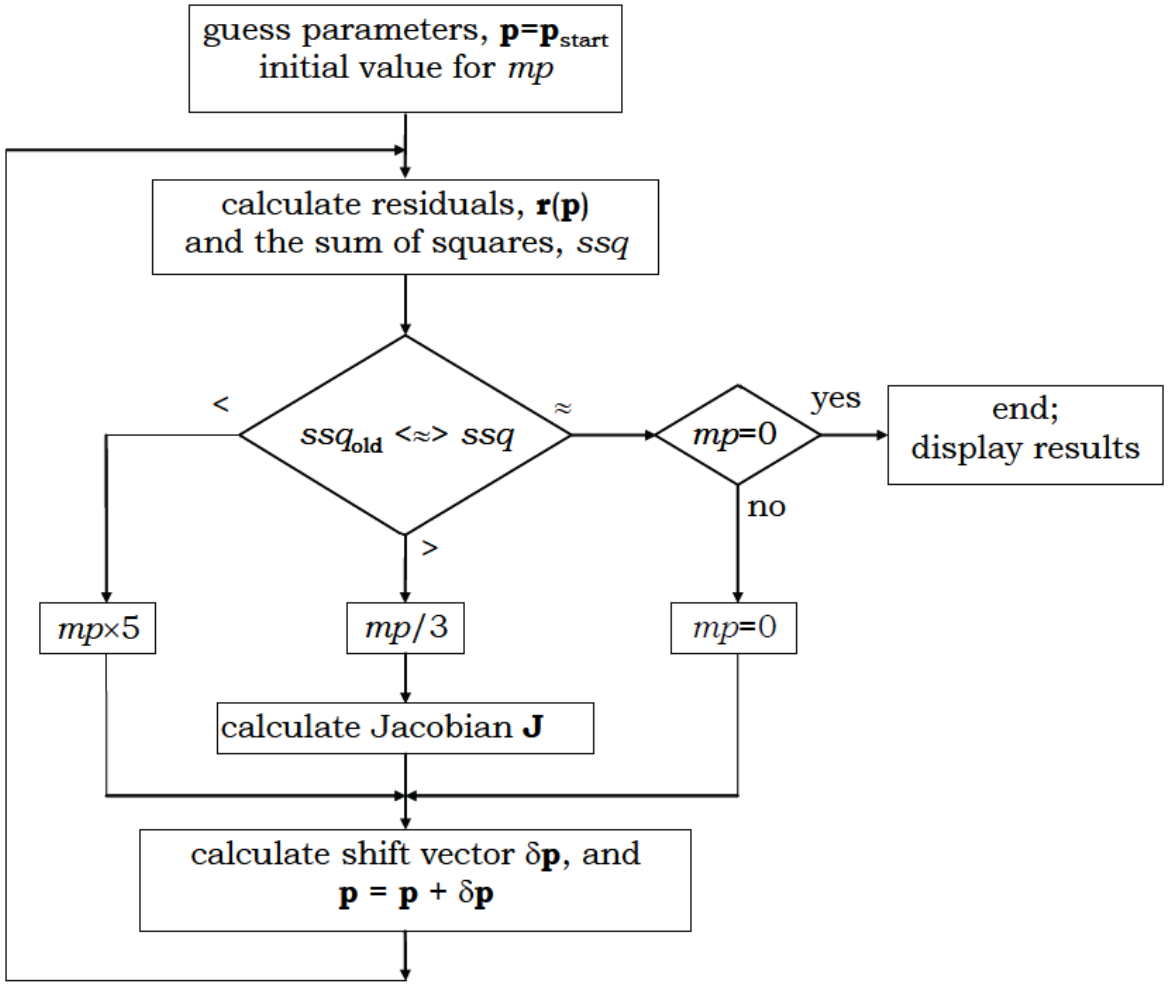
  $\lambda_k = 0 \quad \Rightarrow$ Newton-Gauss,

  $\lambda_k \to \infty \Rightarrow$ Steepest Descent (shorter stepsize)

- λ_k is adapted according to heuristic arguments to avoid divergence due to a bad choice of the initial guesses.

* Levenberg (US-Math., 1919-1973), Marquardt (US-Math., 1929-1997)

2.3. NGLM Algorithm



2.3. Elimination of Linear Parameters

- Certain output models can be written as a **product** of a function $\mathbf{f}_{y,\boldsymbol{\theta}_1}$ depending only on nonlinear parameters $\boldsymbol{\theta}_1$ and the linear parameters $\boldsymbol{\theta}_2$ as:

$$\mathbf{Y}(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2) = \mathbf{f}_y(\mathbf{x}(t, \boldsymbol{\theta}_1), \boldsymbol{\theta}_2) := \mathbf{f}_{y,\boldsymbol{\theta}_1}\left(\mathbf{x}(t, \boldsymbol{\theta}_1)\right) \boldsymbol{\theta}_2$$

- For these output models, the linear parameters $\boldsymbol{\theta}_2$ can be eliminated by linear regression using the **measurements** as

$$\widehat{\boldsymbol{\theta}}_2 = \mathbf{f}_{y,\boldsymbol{\theta}_1}\left(\mathbf{x}(t, \boldsymbol{\theta}_1)\right)^+ \widetilde{\mathbf{Y}} \;\Rightarrow \mathbf{Y}(\boldsymbol{\theta}_1) = \mathbf{f}_{y,\boldsymbol{\theta}_1}\left(\mathbf{x}(t, \boldsymbol{\theta}_1)\right) \mathbf{f}_{y,\boldsymbol{\theta}_1}\left(\mathbf{x}(t, \boldsymbol{\theta}_1)\right)^+ \widetilde{\mathbf{Y}}$$

- Hence, the linear parameters disappear of the regression problem:

$$[\boldsymbol{\theta}_1^* \;\; \boldsymbol{\theta}_2^*] = \arg\left\{ \min_{\boldsymbol{\theta}_1, \boldsymbol{\theta}_2} \; \phi\left(\widetilde{\mathbf{Y}}, \mathbf{Y}(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2)\right)\right\} \Rightarrow$$

$$\boldsymbol{\theta}_1^* = \arg\left\{ \min_{\boldsymbol{\theta}_1} \; \phi\left(\widetilde{\mathbf{Y}}, \mathbf{Y}(\boldsymbol{\theta}_1)\right)\right\} \text{ with } \widehat{\boldsymbol{\theta}}_2^* = \mathbf{f}_{y,\boldsymbol{\theta}_1}\left(\mathbf{x}(t, \boldsymbol{\theta}_1^*)\right)^+ \widetilde{\mathbf{Y}}$$

2.3. Examples of Elimination of Linear Parameters

- **Spectroscopic Data**

$$\mathbf{Y}(\boldsymbol{\theta}, \mathbf{A}) = \mathbf{f}_y(\mathbf{c}(t, \boldsymbol{\theta}), \mathbf{A}) \coloneqq \mathbf{C}(\boldsymbol{\theta})\, \mathbf{A}$$

$$\text{Elimination: } \widehat{\mathbf{A}} = \mathbf{C}(\boldsymbol{\theta})^{+} \widetilde{\mathbf{Y}} \;\Rightarrow \mathbf{Y}(\boldsymbol{\theta}) = \mathbf{C}(\boldsymbol{\theta})\, \mathbf{C}(\boldsymbol{\theta})^{+} \widetilde{\mathbf{Y}}$$

$$\boldsymbol{\theta}^{*} = \arg\left\{ \min_{\boldsymbol{\theta}} \; \sum_{i=1}^{H} \sum_{j=1}^{W} (\tilde{y}_{i,j} - y(\boldsymbol{\theta})_{i,j})^2 \right\}$$

$$\text{with } \widehat{\mathbf{A}}^{*} = \mathbf{C}(\boldsymbol{\theta}^{*})^{+} \widetilde{\mathbf{Y}}$$

- **Calorimetric Data**

$$\mathbf{q}(\boldsymbol{\theta}, \Delta\mathbf{H}_r) = f_y(\mathbf{r}_v(t, \boldsymbol{\theta}), \Delta\mathbf{H}_r) \coloneqq \mathbf{R}_v(\boldsymbol{\theta})(-\Delta\mathbf{H}_r)$$

$$\text{Elimination: } \Delta\widehat{\mathbf{H}}_r = -\mathbf{R}_v(\boldsymbol{\theta})^{+} \widetilde{\mathbf{q}} \;\Rightarrow\; \mathbf{q}(\boldsymbol{\theta}) = \mathbf{R}_v(\boldsymbol{\theta})\, \mathbf{R}_v(\boldsymbol{\theta})^{+} \widetilde{\mathbf{q}}$$

$$\boldsymbol{\theta}^{*} = \arg\left\{ \min_{\boldsymbol{\theta}} \; \sum_{i=1}^{H} (\tilde{q}_i - q(\boldsymbol{\theta})_i)^2 \right\}$$

$$\text{with } \Delta\widehat{\mathbf{H}}_r^{*} = -\mathbf{R}_v(\boldsymbol{\theta}^{*})^{+} \widetilde{\mathbf{q}}$$

2.3. Statistical Information provided by Gradient Methods

- Degree of Freedom: number of redundant information in the data during the minimization

$$df \coloneqq \dim(\mathbf{Y}, 1) \cdot \dim(\mathbf{Y}, 2) - (\dim(\boldsymbol{\theta}_1) + \dim(\boldsymbol{\theta}_2))$$

- Residual variance: variance of the residuals comparable to the variance of the measurements

$$\sigma_r^2 \coloneqq \frac{ssq(\boldsymbol{\theta}^*)}{df} = \frac{\mathbf{r}(\boldsymbol{\theta}^*)^{\mathrm{T}}\mathbf{r}(\boldsymbol{\theta}^*)}{df}$$

- Variance-covariance matrix: indicates the variance in the fitted parameters and their covariance with the other parameters

$$\boldsymbol{\Sigma}_{\boldsymbol{\theta}^*} \coloneqq \sigma_r^2 \ \mathbf{H}(\boldsymbol{\theta}^*)^{-1} \approx \sigma_r^2 \left(\mathbf{J}(\boldsymbol{\theta}^*)^{\mathrm{T}}\mathbf{J}(\boldsymbol{\theta}^*)\right)^{-1}$$

- Correlation matrix: variance-covariance matrix normalized to 1

$$\mathbf{P}_{\boldsymbol{\theta}^*} \coloneqq \mathbf{b}\, \boldsymbol{\Sigma}_{\boldsymbol{\theta}^*} \mathbf{b} \ \text{ with } \mathbf{b} = \left(\mathrm{Diag}\left(\mathrm{diag}\left(\boldsymbol{\Sigma}_{\boldsymbol{\theta}^*}^{1/2} \right) \right) \right)^{-1}$$

2.3. MATLAB Nonlinear Optimizers

- **Optimization of one variable**
  - `fminbnd` Minimum on an interval
- **Optimization of several variables**
  - `fminunc` Unconstrained minimization
  - `fmincon` Constrained minimization (not detailed here, see Chapter 3)
- **MATLAB fminbnd**:
  - `[x,fval,exitflag] = fminbnd(fun,x1,x2,options,...)`
  - `options = optimset('name1',value1,'name2',value2)`
- **MATLAB fminunc**:
  - `[x,fval,exitflag] = fminunc(fun,x0,options,...)`
  - `options = optimoptions(SolverName,'name1',value1)`

2.3. Exercise on Uni/Multivariate Regression

- Consider the **last exercise of Chapter 1.2**.
- **Simulated Reality:** Simulate noisy spectroscopic and calorimetric measurements based on the reaction scheme.
- **Fit the 'measured' spectroscopic data** in the least squares sense **by adjusting the two rate constants**. Estimate their respective uncertainties and correlations. Eliminate the pure component spectra and estimate them at the end.
- **Fit the 'measured' calorimetric data** in the least squares sense **by adjusting the two rate constants**. Estimate their respective uncertainties and correlations. Eliminate the enthalpies of reaction and estimate them at the end.

3. Optimization Problems (OP)

- **Optimization problems**
  Mathematical problems in which the **optimum** (min or max) **of an objective/cost function** is found by adjusting **decision variables** (d.v., $\mathbf{u}$).
- **Requirements**
  Optimization relies on the **knowledge of a mathematical model and model parameters** (e.g. identified/estimated by regression)
- **Constrained vs Unconstrained optimization**
  Optimization problems can be **constrained** (equality and inequality constraints) or **unconstrained**
- **Dynamic vs Static optimization**
  Optimization problems can be **dynamic** (**dynamic model and dynamic d.v.,** $\mathbf{u}(t)$) or **static** (**static model and static d.v.,** $\mathbf{u}$).

3. Solution of **Dynamic Optimization** Problems

Two approaches exist to solve **dynamic** optimization problems:

- First optimize, then discretize (difficult) $\Rightarrow \mathbf{u}(t)$
  First optimize the function f of the decision variables (d.v., $\mathbf{u}$) among an infinite set of functions (called a *functional*) on the entire time interval, **then discretize** the time to compute the optimal $\mathbf{u}$.
- First discretize, then optimize (more common) $\Rightarrow \mathbf{u}(t_i)$
  First discretize the time and define a set of decision variables per interval ($\mathbf{u}(t_i)$), **then optimize** the problem and find the optimal decision variables on all the intervals.
- **Discretization methods** (for first discretize, then optimize)
  Decision variables can be *piecewise constant* (1 d.v./interval), *piecewise linear* (2 d.v./interval), *piecewise polynomial* (3 d.v./interval)...

3.1. Formulation of **Dynamic Optimization** Problems

- A **dynamic optimization problem** consists in
  - **Minimizing an objective function** $\phi(\cdot)$ (or cost function)
  - Knowing a **dynamic model** $\mathbf{f}_{x,d}(\cdot)$ and (possibly) **a static model** $\mathbf{f}_{x,s}(\cdot)$
  - Using an **output (signal) model** $\mathbf{f}_y(\cdot)$
  - Defining equality $\mathbf{h}(\cdot)$ and inequality $\mathbf{g}(\cdot)$ constraints (if constrained)
  - Defining bounds on the decision variables: $\mathbf{u}^-$ and $\mathbf{u}^+$
  - **Adjusting the decision variables** $\mathbf{u}(t)$ such that ϕ is minimal

$$\mathbf{u}^*(t) = \arg\left\{ \min_{\mathbf{u}(t)} \ \phi\big(\mathbf{y}(\mathbf{u}(t))\big) \right\} \qquad \text{Objective function}$$

$$\text{s.t.} \quad \dot{\mathbf{x}}(t) = \mathbf{f}_{x,d}(\mathbf{x}(t), \mathbf{u}(t), \boldsymbol{\theta}) \qquad \text{Dynamic model}$$

$$\mathbf{x}(t) = \mathbf{f}_{x,s}(\mathbf{x}(t), \mathbf{u}(t), \boldsymbol{\theta}) \qquad \text{Static model}$$

$$\mathbf{y}(\mathbf{u}(t)) = \mathbf{f}_y(\mathbf{x}(t, \mathbf{u}(t))) \qquad \text{Output model}$$

$$\mathbf{g}(\mathbf{y}(t, \mathbf{u}(t))) \leq \mathbf{0} \qquad \text{Inequality constraints}$$

$$\mathbf{h}(\mathbf{y}(t, \mathbf{u}(t))) = \mathbf{0} \qquad \text{Equality constraints}$$

$$\mathbf{u}^- \leq \mathbf{u}(t) \leq \mathbf{u}^+ \qquad \text{Bounds on } \mathbf{u}$$

3.1. **Reformulation** with *First Discretize, then Optimize*

- The *continuous decision variables* $\mathbf{u}(t)$ of the **dynamic optimization problem** are discretized on H time intervals using a discretization method (e.g. piecewise constant).

  This reformulation transforms the n_u decision variables $\mathbf{u}(t)$ *continuous in time* into $n_u \cdot H$ decisions variables $\mathbf{u}(t_i)$, $i = 1, \dots, H$, *discrete in time*.

$$[\mathbf{u}^*(t_1), \dots, \mathbf{u}^*(t_H)] = \arg\left\{ \min_{\mathbf{u}(t_1),\dots,\mathbf{u}(t_H)} \sum_{i=1}^{H} \phi\big(\mathbf{y}(\mathbf{u}(t_i))\big) \right\} \qquad t_{i-1} \le t \le t_i$$

$$\begin{aligned}
\text{s. t.} \quad & \dot{\mathbf{x}}(t) = \mathbf{f}_{x,d}(\mathbf{x}(t), \mathbf{u}(t_i), \boldsymbol{\theta}) \\
& \mathbf{x}(t) = \mathbf{f}_{x,s}(\mathbf{x}(t), \mathbf{u}(t_i), \boldsymbol{\theta}) \\
& \mathbf{y}(\mathbf{u}(t_i)) = \mathbf{f}_{y}(\mathbf{x}(t, \mathbf{u}(t_i))) \\
& \mathbf{g}(\mathbf{y}(\mathbf{u}(t_i))) \le \mathbf{0} \\
& \mathbf{h}(\mathbf{y}(\mathbf{u}(t_i))) = \mathbf{0} \\
& \mathbf{u}^- \le \mathbf{u}(t_i) \le \mathbf{u}^+
\end{aligned}$$

3.1. Formulation of **Static Optimization** Problems

- A **static optimization problem** consists in
  - **Minimizing an objective function** $\phi(\cdot)$ (or cost function)
  - Knowing a **dynamic model** $\mathbf{f}_{x,d}(\cdot)$ and (possibly) **a static model** $\mathbf{f}_{x,s}(\cdot)$
  - Using an **output (signal) model** $\mathbf{f}_y(\cdot)$
  - Defining equality $\mathbf{h}(\cdot)$ and inequality $\mathbf{g}(\cdot)$ constraints (if constrained)
  - Defining bounds on the decision variables: $\mathbf{u}^-$ and $\mathbf{u}^+$
  - **Adjusting the decision variables** $\mathbf{u}$ such that ϕ is minimal

$$\mathbf{u}^* = \arg\left\{ \min_{\mathbf{u}} \ \phi\big(\mathbf{y}(\mathbf{u})\big) \right\} \qquad \text{Objective function}$$

$$\begin{aligned}
\text{s.t.} \quad & \dot{\mathbf{x}}(t) = \mathbf{f}_{x,d}(\mathbf{x}(t), \mathbf{u}, \boldsymbol{\theta}) && \text{Dynamic model} \\
& \mathbf{x}(t) = \mathbf{f}_{x,s}(\mathbf{x}(t), \mathbf{u}, \boldsymbol{\theta}) && \text{Static model} \\
& \mathbf{y}(\mathbf{u}) = \mathbf{f}_y(\mathbf{x}(t, \mathbf{u})) && \text{Output model} \\
& \mathbf{g}(\mathbf{y}(\mathbf{u})) \leq \mathbf{0} && \text{Inequality constraints} \\
& \mathbf{h}(\mathbf{y}(\mathbf{u})) = \mathbf{0} && \text{Equality constraints} \\
& \mathbf{u}^- \leq \mathbf{u} \leq \mathbf{u}^+ && \text{Bounds on } \mathbf{u}
\end{aligned}$$

3.1. **Unconstrained Optimization** Problems

- **Unconstrained dynamic** optimization problems

$$[\mathbf{u}^*(t_1), \dots, \mathbf{u}^*(t_H)] = \arg\left\{ \min_{\mathbf{u}(t_1),\dots,\mathbf{u}(t_H)} \sum_{i=1}^{H} \phi\left(\mathbf{y}(\mathbf{u}(t_i))\right) \right\} \quad t_{i-1} \le t \le t_i$$

$$\begin{aligned} \text{s.t.} \quad & \dot{\mathbf{x}}(t) = \mathbf{f}_{x,d}(\mathbf{x}(t), \mathbf{u}(t_i), \boldsymbol{\theta}) \\ & \mathbf{x}(t) = \mathbf{f}_{x,s}(\mathbf{x}(t), \mathbf{u}(t_i), \boldsymbol{\theta}) \\ & \mathbf{y}(\mathbf{u}(t_i)) = \mathbf{f}_{y}(\mathbf{x}(t, \mathbf{u}(t_i))) \end{aligned}$$

- **Unconstrained static** optimization problems

$$\mathbf{u}^* = \arg\left\{ \min_{\mathbf{u}} \ \phi(\mathbf{y}(\mathbf{u})) \right\}$$

$$\begin{aligned} \text{s.t.} \quad & \dot{\mathbf{x}}(t) = \mathbf{f}_{x,d}(\mathbf{x}(t), \mathbf{u}, \boldsymbol{\theta}) \\ & \mathbf{x}(t) = \mathbf{f}_{x,s}(\mathbf{x}(t), \mathbf{u}, \boldsymbol{\theta}) \\ & \mathbf{y}(\mathbf{u}) = \mathbf{f}_{y}(\mathbf{x}(t, \mathbf{u})) \end{aligned}$$

3.1. NCO's for **Unconstrained Optimization** Problems

The NCO's defined in Chapter 2 remain valid for unconstrained optimization problems

- **1st order NCO:** If $\mathbf{u}^*$ is a local minimum of a function $\phi: \mathcal{C} \longrightarrow \mathbb{R}$, then

$$\boxed{\nabla\phi(\mathbf{u}^*) = \mathbf{J}^{\mathrm{T}}(\mathbf{u}^*) = \mathbf{0} \Leftrightarrow \mathbf{u}^* \text{ is a stationary point}}$$

gradient Jacobian

- **2nd order NCO:** If $\mathbf{u}^*$ is a local minimum of $\phi: \mathcal{C} \longrightarrow \mathbb{R}$, then

$$\boxed{\nabla^2\phi(\mathbf{u}^*) = \mathbf{H}(\mathbf{u}^*) \succcurlyeq \mathbf{0} \ \text{(positive semidefinite)}}$$

- 1st and 2nd order NCO form **sufficient conditions of optimality (SCO)** if ϕ is a convex function defined on a convex set $\mathcal{C}$.

3.1. **Constrained Optimization** Problems

- **Lagrange** function $\mathcal{L}(\cdot)$, a.k.a. **Lagrangian**
  - Dynamic optimization problems

$$\mathcal{L}\left(\mathbf{u}(t_1), \dots, \mathbf{u}(t_H)\right) := \sum_{i=1}^{H} \phi\left(\mathbf{y}\left(t, \mathbf{u}(t_i)\right)\right) + \sum_{i=1}^{H} \boldsymbol{\nu}_i^{\mathrm{T}} \mathbf{g}\left(\mathbf{y}\left(t, \mathbf{u}(t_i)\right)\right) + \sum_{i=1}^{H} \boldsymbol{\mu}_i^{\mathrm{T}} \mathbf{h}\left(\mathbf{y}\left(t, \mathbf{u}(t_i)\right)\right) + \sum_{i=1}^{H} \boldsymbol{\lambda}_{+,i}^{\mathrm{T}} \left(\mathbf{u}(t_i) - \mathbf{u}^{+}\right) + \sum_{i=1}^{H} \boldsymbol{\lambda}_{-,i}^{\mathrm{T}} \left(\mathbf{u}^{-} - \mathbf{u}(t_i)\right), \quad \text{with } t_{i-1} \le t \le t_i$$

  - Static optimization problems

$$\mathcal{L}(\mathbf{u}) := \phi\left(\mathbf{y}(\mathbf{u})\right) + \boldsymbol{\nu}^{\mathrm{T}} \mathbf{g}\left(\mathbf{y}(\mathbf{u})\right) + \boldsymbol{\mu}^{\mathrm{T}} \mathbf{h}\left(\mathbf{y}(\mathbf{u})\right) + \boldsymbol{\lambda}_{+}^{\mathrm{T}} \left(\mathbf{u} - \mathbf{u}^{+}\right) + \boldsymbol{\lambda}_{-}^{\mathrm{T}} \left(\mathbf{u}^{-} - \mathbf{u}\right)$$

  - $\boldsymbol{\nu}$ $(\boldsymbol{\nu}_i)$, $\boldsymbol{\mu}$ $(\boldsymbol{\mu}_i)$ and $\boldsymbol{\lambda}_{+}, \boldsymbol{\lambda}_{-}$ $\left(\boldsymbol{\lambda}_{+,i}, \boldsymbol{\lambda}_{-,i}\right)$ are the **Lagrange multipliers**

3.1. Active vs Inactive Constraints

- **Inequality constraints**
  - **Active** $\boxed{\mathbf{g}_i(\mathbf{y}(\mathbf{u}^*)) \overset{!}{=} \mathbf{0}}$, $i \in \mathcal{A}(\mathbf{u}^*)$ $\Rightarrow$ $\boxed{\boldsymbol{\nu}_i \overset{!}{>} \mathbf{0}}$
    These constraints **play** a role in the minimum of $\mathcal{L}(\cdot)$
  - **Inactive** $\boxed{\mathbf{g}_j(\mathbf{y}(\mathbf{u}^*)) \overset{!}{<} \mathbf{0}}$, $j \notin \mathcal{A}(\mathbf{u}^*)$ $\Rightarrow$ $\boxed{\boldsymbol{\nu}_j \overset{!}{=} \mathbf{0}}$
    These constraints **do not play** any role in the minimum of $\mathcal{L}(\cdot)$
- **Equality constraints** are always **active**
  - **Alway active** $\boxed{\mathbf{h}(\mathbf{y}(\mathbf{u}^*)) \overset{!}{=} \mathbf{0}}$ $\Rightarrow$ $\boxed{\boldsymbol{\mu} \overset{!}{>} \mathbf{0}}$
    These constraints **always play** a role in the minimum of $\mathcal{L}(\cdot)$
- **Finding the minimum of $\mathcal{L}(\cdot)$ consists in following all the active inequality constraints $\mathbf{g}_i, i \in \mathcal{A}(\mathbf{u}^*)$, and equality constraints $\mathbf{h}$**

3.1. Interpretation of the **Lagrange Multipliers**

- The **Lagrange multipliers** represent the **sensitivity** of the objective function with respect to a change in the constraints. They indicate how much the optimal cost would change, if the constraints were perturbed.
- Obviously, the Lagrange multipliers of inactive constraints are zero because any change in the value of these constraints keep the optimal value unchanged.
- In economics, the Lagrange multipliers are viewed as the **marginal costs of the constraints**, and are referred to as the shadow prices.

3.1. NCO's for **Constrained Optimization** Problems

KKT conditions*:

- **1st order KKT:** If $\mathbf{u}^*$ is a local minimum of a function $\mathcal{L}: \mathcal{C} \longrightarrow \mathbb{R}$, then

$$\mathbf{g}(\mathbf{u}^*) \leq \mathbf{0}, \mathbf{h}(\mathbf{u}^*) = \mathbf{0} \text{ and } \mathbf{u}^- \leq \mathbf{u}^* \leq \mathbf{u}^+ \qquad \text{Primal feasibility}$$

$$\mathbf{j} := \frac{\partial \mathcal{L}(\mathbf{u}^*)}{\partial \mathbf{u}} = \frac{\partial \phi(\mathbf{u}^*)}{\partial \mathbf{u}} + \boldsymbol{\nu}^{\mathrm{T}} \frac{\partial \mathbf{g}(\mathbf{u}^*)}{\partial \mathbf{u}} + \boldsymbol{\mu}^{\mathrm{T}} \frac{\partial \mathbf{h}(\mathbf{u}^*)}{\partial \mathbf{u}} + \boldsymbol{\lambda}_+^{\mathrm{T}} - \boldsymbol{\lambda}_-^{\mathrm{T}} = \mathbf{0} \qquad \text{Dual feasibility}$$

$$\nu_i, \mu_i, \lambda_i \geq 0 \qquad \text{Dual feasibility}$$

$$\boldsymbol{\nu}^{\mathrm{T}} \mathbf{g}(\mathbf{u}^*) = 0, \boldsymbol{\mu}^{\mathrm{T}} \mathbf{h}(\mathbf{u}^*) = 0, \boldsymbol{\lambda}_+^{\mathrm{T}} (\mathbf{u} - \mathbf{u}^+) = \boldsymbol{\lambda}_-^{\mathrm{T}} (\mathbf{u}^- - \mathbf{u}) = \mathbf{0} \qquad \text{Complementary slackness}$$

- **2nd order KKT:** If $\mathbf{u}^*$ is a local minimum of $\mathcal{L}: \mathcal{C} \longrightarrow \mathbb{R}$, then

$$\boxed{\nabla^2 \mathcal{L}(\mathbf{u}^*) = \mathbf{H}(\mathbf{u}^*) \succcurlyeq \mathbf{0} \text{ (positive semidefinite)}}$$

- KKT conditions are **sufficient conditions** if ϕ and $\mathbf{g}$ are convex, and $\mathbf{h}$ are affine functions, all defined on a convex set $\mathcal{C}$.

* **K**arush (US-Math., 1917-1997), **K**uhn (US-Math., 1925-2014), **T**ucker (US-Math., 1905-1995)

3.1. Alternative 1st NCO for **Constrained** Problems

1st NCO:

- $\frac{\partial \mathcal{L}(\mathbf{u}^*)}{\partial \mathbf{u}} = \mathbf{0}$
- $\frac{\partial \mathcal{L}(\mathbf{u}^*)}{\partial \boldsymbol{\nu}} = \mathbf{0}$
- $\frac{\partial \mathcal{L}(\mathbf{u}^*)}{\partial \boldsymbol{\mu}} = \mathbf{0}$
- $\frac{\partial \mathcal{L}(\mathbf{u}^*)}{\partial \boldsymbol{\lambda}_+} = \mathbf{0}$
- $\frac{\partial \mathcal{L}(\mathbf{u}^*)}{\partial \boldsymbol{\lambda}_-} = \mathbf{0}$

Solve for the unknowns $\mathbf{u}$, $\boldsymbol{\nu}$, $\boldsymbol{\mu}$, $\boldsymbol{\lambda}_+$ and $\boldsymbol{\lambda}_-$

$n_{\mathbf{u}} + n_{\mathbf{g}} + n_{\mathbf{h}} + 2n_{\mathbf{u}}$ eqs with as many unknowns

+ all 1st order KKT conditions to **rule out contradictory solutions**

3.1. **Constraint Qualification** (CQ)

Not every (local) minimum is a KKT point
(there might be more minima than KKT points)

but…

- Applying a **Constraint Qualification (CQ)** ensures that all (local) minima satisfy the KKT conditions.
- **L**inear **I**ndependence **C**onstraint **Q**ualification (**LICQ**)
  a point $\mathbf{u}^*$ is said to be a regular point *if the gradients of the active constraints are independent (= full rank)*
- If LICQ applies, the Lagrange Multipliers are unique
- **KKT are sufficient conditions** if the objective function and the active constraints are **convex** functions (as mentioned earlier)

3.1. Example: **Unconstrained Optimization**

- Let consider

$$\phi(x) = \frac{1}{4}x^3 + \frac{3}{4}x^2 - \frac{3}{2}x - 2$$

$$\text{on the domain } x \in \mathcal{C} = [-5, 3]$$

- Find analytically the stationary points of $\phi(x)$ using
  - 1st NCO: find x^* s.t. $J = \frac{d\phi(x)}{dx} = 0$
- Qualify the stationary points (minima/maxima) using
  - 2nd NCO: find x^* s.t. $H = \frac{d^2\phi(x)}{d^2x} \geq 0$ (minimum)

$$\text{find } x^* \text{ s.t. } H = \frac{d^2\phi(x)}{d^2x} \leq 0 \text{ (maximum)}$$

3.1. Example: **Constrained Optimization**

- Let consider

$$\min_{x_1,x_2} \ \phi(x_1,x_2) = x_1^2 + x_2^2$$

$$\text{s.t.} \quad h(x_1,x_2):\ x_1+x_2 = 1$$

$$g(x_1,x_2):\ x_2 \leq a$$

- Find analytically the unconstrained minimum of ϕ
- Find analytically the minimum of ϕ constrained by h
- Find analytically the minimum of ϕ constrained by h and g, and discuss the influence of a on the solution

3.2. **Solving** Optimization Problems (OP)

Simple static optimization:

1. Solution of optimization problems with explicit equality constraints
2. Graphical solution of linear optimization problems with a limited number of decision variables (max 3) and a limited number of explicit constraints

Dynamic optimization and more complex static problems

- Solution obtained numerically
  - **Penalty function** (reformulation in an unconstrained problem, no use of KKT's)
  - **Interior point methods** (reformulation in an unconstrained problem, no KKT's)
  - **Newton-like methods** (Sequential Quadratic Programming, SQP) (use of KKT's)

3.2. OP with **Explicit Equality Constraints**

- If the equality constraints are *explicit* and *independent*, n_h decision variables $\mathbf{u}_h$ can be replaced by the expression of their equality constraint in the objective function ϕ.
- The optimization problem is then reduced to finding $n_d = (n_u - n_h)$ decision variables $\mathbf{u}_d$ that minimize ϕ.

Before:

$$\mathbf{u}^* = \arg\left\{ \min_{\mathbf{u}} \ \phi\big(\mathbf{y}(\mathbf{u})\big) \right\}$$
$$\text{s.t.} \quad \mathbf{x} = \mathbf{f}_{x,s}(\mathbf{x}, \mathbf{u}, \boldsymbol{\theta})$$
$$\mathbf{y}(\mathbf{u}) = \mathbf{f}_y(\mathbf{x}(\mathbf{u}))$$
$$\mathbf{h}(\mathbf{y}(\mathbf{u})) = \mathbf{0}$$
$$\mathbf{u}^- \leq \mathbf{u} \leq \mathbf{u}^+$$

n_u decision variables $\mathbf{u}$

After:

$$\mathbf{u}_d^* = \arg\left\{ \min_{\mathbf{u}_d} \ \phi\big(\mathbf{y}(\mathbf{u}_d)\big) \right\}$$
$$\text{s.t.} \quad \mathbf{x} = \mathbf{f}_{x,s}(\mathbf{x}, \mathbf{u}_d, \boldsymbol{\theta})$$
$$\mathbf{y}(\mathbf{u}_d) = \mathbf{f}_y(\mathbf{x}(\mathbf{u}_d))$$
$$\mathbf{u}_h = \mathbf{h}_d\,(\mathbf{y}(\mathbf{u}_d))$$
$$\mathbf{u}_d^- \leq \mathbf{u}_d \leq \mathbf{u}_d^+$$

$n_u - n_h$ decision variables $\mathbf{u}_d$

3.2. Example: OP with **Explicit Equality Constraints**

- Let consider

$$\min_{x_1,x_2} \ \phi(x_1,x_2) = x_1^2 + x_2^2 + 4$$
$$\text{s.t.} \ \ h\text{:}\ x_1 + x_2 = 1$$

- Find analytically the minimum of ϕ constrained by h using h to eliminate x_2 from ϕ.

- Let consider

$$\min_{x_1,x_2,,x_3} \ \phi(x_1,x_2) = x_1^2 + x_2^2 + x_3^2 + 4$$
$$\text{s.t.} \ \ h\text{:}\ x_1 + x_2 = 1$$

- Find analytically the minimum of ϕ constrained by h using the elimination of x_2 by h.

3.2. **Graphical Solution of Linear OP** + Example

For linear optimization problems, the minimum always lie in one of the vertices (corners) of the feasible region.

Example: A manufacturer has to produce pants (x) and jackets (y). For materials, the manufacturer has 750 m^2 of cotton and 1 000 m^2 of polyester. Every pair of pants (1 unit) needs 1 m^2 of cotton and 2 m^2 of polyester. Every jacket needs 1.5 m^2 of cotton and 1 m^2 of polyester. The price of the pants is fixed at 50 $ and the jacket at 40 $. Note that, for obvious reasons, the manufacturer must produce ($x > 0$ and $y > 0$).

What is the number of pants and jackets that the manufacturer must produce to obtain a maximum profit?

3.2. **Penalty Function** (shown for Static OP)

Original Constrained Optimization Problem:

$$\mathbf{u}^* = \arg\left\{ \min_{\mathbf{u}} \ \phi(\mathbf{y}(\mathbf{u})) \right\}$$
$$\text{s.t} \quad \mathbf{g}(\mathbf{y}(\mathbf{u})) \leq \mathbf{0}$$
$$\mathbf{h}(\mathbf{y}(\mathbf{u})) = \mathbf{0}$$
$$\mathbf{u}^- \leq \mathbf{u} \leq \mathbf{u}^+$$

Reformulated Unconstrained Optimization Problem:

$$\mathbf{u}^* = \arg\left\{ \min_{\mathbf{u}} \ \phi(\mathbf{y}(\mathbf{u})) + \mu\, \alpha(\mathbf{y}(\mathbf{u})) \right\}$$

with the *auxiliary function*:

$$\alpha(\mathbf{y}(\mathbf{u})) := \sum_{i=1}^{n_g} (\max[0, g_i(\mathbf{y}(\mathbf{u}))])^2 + \sum_{j=1}^{n_h} |h_j(\mathbf{y}(\mathbf{u}))|^2$$

more generally: $\alpha(\mathbf{y}(\mathbf{u})) := \sum_{i=1}^{n_g} (\max[0, g_i(\mathbf{y}(\mathbf{u}))])^p + \sum_{j=1}^{n_h} |h_j(\mathbf{y}(\mathbf{u}))|^p$

3.2. Example: **Penalty Function**

- Let consider the problem of minimizing $\phi(x) := x$, subject to $g(x) := 2 - x \leq 0$.
- The obvious solution to this problem is $x^* = 2$ with $\phi(x^*) = 2$.
- Show that the solution of the Penalty problem can be made arbitrarily close to the solution of the original problem, by choosing the value of the penalty parameter μ sufficiently large.

3.2. A Simple **Algorithm for Penalty Function**

- Define $\varepsilon > 0$
- Choose an initial guess $\mathbf{u}_0$
- Initialize $\mu_0 > 0$
- Define $\beta > 1$ (**increasing** effect of the penalty)
- Set $k = 0$, then

1. Solve $\mathbf{u}_{k+1} = \arg\left\{ \min_{\mathbf{u}} \ \phi\big(\mathbf{y}(\mathbf{u})\big) + \mu_k \, \alpha\big(\mathbf{y}(\mathbf{u})\big) \right\}$
2. If $\mu_k \, \alpha\big(\mathbf{y}(\mathbf{u}_{k+1})\big) < \varepsilon$, stop;
   otherwise $\mu_{k+1} = \beta \mu_k$, $k \leftarrow k + 1$ and
   go back to Step 1.

3.2. **Interior Point Methods** (shown for Static OP)

Original Constrained Optimization Problem:

$$\mathbf{u}^* = \arg\left\{ \min_{\mathbf{u}} \ \phi(\mathbf{y}(\mathbf{u})) \right\}$$
$$\text{s.t} \ \ \mathbf{g}(\mathbf{y}(\mathbf{u})) \leq \mathbf{0}$$

Reformulated Unconstrained Optimization Problem:

$$\mathbf{u}^* = \arg\left\{ \min_{\mathbf{u}} \ \phi(\mathbf{y}(\mathbf{u})) + \mu \, b(\mathbf{y}(\mathbf{u})) \right\}$$

with the *auxiliary function*: $\boxed{b(\mathbf{y}(\mathbf{u})) := -\sum_{i=1}^{n_g} \frac{1}{g_i(\mathbf{y}(\mathbf{u}))}}$

or as an alternative: $b(\mathbf{y}(\mathbf{u})) := -\sum_{i=1}^{n_g} \ln\left(-g_i(\mathbf{y}(\mathbf{u}))\right)$

The auxiliary function represents *a barrier function* that enforces staying within the feasible region, namely, $\mathbf{g}(\mathbf{y}(\mathbf{u})) < \mathbf{0}$

3.2. Example: **Interior Point Methods**

- Let consider the problem of minimizing $\phi(x) := x$, subject to $g(x) := 2 - x \leq 0$.
- The obvious solution to this problem is $x^* = 2$ with $\phi(x^*) = 2$.
- Show that the solution of the Barrier Function can be made arbitrarily close to the solution of the original problem, by choosing the value of the barrier parameter μ sufficiently close to 0.

3.2. A Simple **Algorithm for Interior Point Method**

- Define $\varepsilon > 0$
- Choose an initial guess $\mathbf{u}_0$
  in the feasible region $\mathbf{g}(\mathbf{y}(\mathbf{u}_0)) < \mathbf{0}$
- Initialize $\mu_0 > 0$
- Define $\beta \in [0,1]$ (**reducing** effect of the barrier)
- Set $k = 0$, then

1. Solve $\mathbf{u}_{k+1} = \arg\left\{ \min_{\mathbf{u}} \ \phi(\mathbf{y}(\mathbf{u})) + \mu_k \, b(\mathbf{y}(\mathbf{u})) \right\}$
2. If $\mu_k \, b(\mathbf{y}(\mathbf{u}_{k+1})) < \varepsilon$, stop;
   otherwise $\mu_{k+1} = \beta \mu_k$, $k \leftarrow k + 1$ and
   go back to Step 1.

3.2. Lagrange Multipliers vs Penalty/Barrier Parameters

Penalty Function:

- $\nu_i := -\mu \frac{\partial}{\partial \mathbf{u}} [(\max[0, g_i(\mathbf{y}(\mathbf{u}))])^2]$

Interior Point Method (Barrier Function):

- $\nu_i := -\mu \frac{\partial}{\partial \mathbf{u}} \left[-\frac{1}{g_i(\mathbf{y}(\mathbf{u}))} \right]$

Example:

Compute the Lagrange multiplier as a function of μ for the previous example, both for the Penalty Function and for the Barrier Function…

3.2. Sequential Quadratic Programming (SQP)

- An SQP is a **Newton-like or Quasi-Newton Method** that uses the **KKT conditions** to **minimize a quadratic approximation of the Lagrange function subject to a linear approximation of the constraints**.
- **Only the active inequality constraints** (set $\mathcal{A}$) **are of interest** since the inactive inequality constraints have no influence on the objective function.
- For the sake of conciseness, **the upper and lower bounds are assumed to be treated as additional inequality constraints:** $\mathbf{u}^- - \mathbf{u} \leq \mathbf{0}$ and $\mathbf{u} - \mathbf{u}^+ \leq \mathbf{0}$
- For the sake of conciseness, $\mathbf{y}(\mathbf{u})$ will just be written as $\mathbf{u}$

3.2. SQP – **Lagrange Function and KKT Conditions**

The **Lagrange Function** is defined as:

- $\mathcal{L}(\mathbf{u}^*, \boldsymbol{\nu}_{\mathcal{A}}^*, \boldsymbol{\mu}^*) = \phi(\mathbf{u}^*) + \boldsymbol{\nu}_{\mathcal{A}}^{*,\mathrm{T}}\, \mathbf{g}_{\mathcal{A}}(\mathbf{u}) + \boldsymbol{\mu}^{*,\mathrm{T}}\mathbf{h}(\mathbf{u})$

The **KKT conditions** imply:

- $\frac{\partial}{\partial \mathbf{u}}\mathcal{L}(\mathbf{u}^*, \boldsymbol{\nu}_{\mathcal{A}}^*, \boldsymbol{\mu}^*) = \frac{\partial \phi(\mathbf{u}^*)}{\partial \mathbf{u}} + \frac{\partial \mathbf{g}_{\mathcal{A}}(\mathbf{u}^*)}{\partial \mathbf{u}}\boldsymbol{\nu}_{\mathcal{A}}^* + \frac{\partial \mathbf{h}(\mathbf{u}^*)}{\partial \mathbf{u}}\boldsymbol{\mu}^* \overset{!}{=} \mathbf{0}_{n_u}$
- $\frac{\partial}{\partial \boldsymbol{\nu}_{\mathcal{A}}}\mathcal{L}(\mathbf{u}^*, \boldsymbol{\nu}_{\mathcal{A}}^*, \boldsymbol{\mu}^*) = \mathbf{g}_{\mathcal{A}}(\mathbf{u}^*) \overset{!}{=} \mathbf{0}_{n_{\mathcal{A}}}$
- $\frac{\partial}{\partial \boldsymbol{\mu}}\mathcal{L}(\mathbf{u}^*, \boldsymbol{\nu}_{\mathcal{A}}^*, \boldsymbol{\mu}^*) = \mathbf{h}(\mathbf{u}^*) \overset{!}{=} \mathbf{0}_{n_\mu}$

This describes a system of $n_u + n_{\mathcal{A}} + n_\mu$ equations with as many unknown $(\mathbf{u}^*, \boldsymbol{\nu}_{\mathcal{A}}^*, \boldsymbol{\mu}^*)$.

3.2. SQP – **Approximate the KKT Conditions**

Quadratic approx. of $\mathcal{L}$, **linear approx.** of $\mathbf{g}_{\mathcal{A}}$ and $\mathbf{h}$:

- $$\frac{\partial}{\partial \mathbf{u}}\mathcal{L}(\mathbf{u}^*, \boldsymbol{\nu}_{\mathcal{A}}^*, \boldsymbol{\mu}^*) \overset{!}{=} \mathbf{0}_{n_u} \approx \frac{\partial \mathcal{L}(\mathbf{u}_k, \boldsymbol{\nu}_{\mathcal{A},k}, \boldsymbol{\mu}_k)}{\partial \mathbf{u}} + \frac{\partial^2 \mathcal{L}(\mathbf{u}_k, \boldsymbol{\nu}_{\mathcal{A},k}, \boldsymbol{\mu}_k)}{\partial \mathbf{u}^2}(\mathbf{u}_{k+1} - \mathbf{u}_k) + \frac{\partial \mathbf{g}_{\mathcal{A}}(\mathbf{u}_k)}{\partial \mathbf{u}}(\boldsymbol{\nu}_{\mathcal{A},k+1} - \boldsymbol{\nu}_{\mathcal{A},k}) + \frac{\partial \mathbf{h}(\mathbf{u}_k)}{\partial \mathbf{u}}(\boldsymbol{\mu}_{k+1} - \boldsymbol{\mu}_k)$$
- $$\frac{\partial}{\partial \boldsymbol{\nu}_{\mathcal{A}}}\mathcal{L}(\mathbf{u}^*, \boldsymbol{\nu}_{\mathcal{A}}^*, \boldsymbol{\mu}^*) \overset{!}{=} \mathbf{0}_{n_{\mathcal{A}}} \approx \mathbf{g}_{\mathcal{A}}(\mathbf{u}_k) + \frac{\partial \mathbf{g}_{\mathcal{A}}(\mathbf{u}_k)}{\partial \mathbf{u}}(\mathbf{u}_{k+1} - \mathbf{u}_k)$$
- $$\frac{\partial}{\partial \boldsymbol{\mu}}\mathcal{L}(\mathbf{u}^*, \boldsymbol{\nu}_{\mathcal{A}}^*, \boldsymbol{\mu}^*) \overset{!}{=} \mathbf{0}_{n_\mu} \approx \mathbf{h}(\mathbf{u}_k) + \frac{\partial \mathbf{h}(\mathbf{u}_k)}{\partial \mathbf{u}}(\mathbf{u}_{k+1} - \mathbf{u}_k)$$

3.2. SQP – **Define the Shift Vectors**

Quadratic approx. of $\mathcal{L}$, **linear approx.** of $\mathbf{g}_{\mathcal{A}}$ and $\mathbf{h}$:

$$\begin{cases} \dfrac{\partial \mathcal{L}(\mathbf{u}_k, \boldsymbol{\nu}_{\mathcal{A},k}, \boldsymbol{\mu}_k)}{\partial \mathbf{u}} + \dfrac{\partial^2 \mathcal{L}(\mathbf{u}_k, \boldsymbol{\nu}_{\mathcal{A},k}, \boldsymbol{\mu}_k)}{\partial \mathbf{u}^2} \Delta\mathbf{u}_k + \dfrac{\partial \mathbf{g}_{\mathcal{A}}(\mathbf{u}_k)}{\partial \mathbf{u}} \Delta\boldsymbol{\nu}_{\mathcal{A},k} + \dfrac{\partial \mathbf{h}(\mathbf{u}_k)}{\partial \mathbf{u}} \Delta\boldsymbol{\mu}_k = \mathbf{0}_{n_u} \\ \mathbf{g}_{\mathcal{A}}(\mathbf{u}_k) + \dfrac{\partial \mathbf{g}_{\mathcal{A}}(\mathbf{u}_k)}{\partial \mathbf{u}} \Delta\mathbf{u}_k = \mathbf{0}_{n_{\mathcal{A}}} \\ \mathbf{h}(\mathbf{u}_k) + \dfrac{\partial \mathbf{h}(\mathbf{u}_k)}{\partial \mathbf{u}} \Delta\mathbf{u}_k = \mathbf{0}_{n_\mu} \end{cases}$$

with $\Delta\mathbf{u}_k := (\mathbf{u}_{k+1} - \mathbf{u}_k)$,

$\Delta\boldsymbol{\nu}_{\mathcal{A},k} := (\boldsymbol{\nu}_{\mathcal{A},k+1} - \boldsymbol{\nu}_{\mathcal{A},k})$,

$\Delta\boldsymbol{\mu}_k := (\boldsymbol{\mu}_{k+1} - \boldsymbol{\mu}_k)$

$\Rightarrow$ Writing this system in matrix notation and passing the first term of each equation on the rhs yields…

3.2. SQP – **Rewrite in Matrix Notation**

$$\begin{bmatrix} \frac{\partial^2 \mathcal{L}(\mathbf{u}_k, \boldsymbol{\nu}_{\mathcal{A},k}, \boldsymbol{\mu}_k)}{\partial \mathbf{u}^2} & \frac{\partial \mathbf{g}_{\mathcal{A}}(\mathbf{u}_k)^{\mathrm{T}}}{\partial \mathbf{u}} & \frac{\partial \mathbf{h}(\mathbf{u}_k)^{\mathrm{T}}}{\partial \mathbf{u}} \\ \frac{\partial \mathbf{g}_{\mathcal{A}}(\mathbf{u}_k)}{\partial \mathbf{u}} & \mathbf{0}_{n_{\mathcal{A}} \times n_{\mathcal{A}}} & \mathbf{0}_{n_{\mathcal{A}} \times n_{\mu}} \\ \frac{\partial \mathbf{h}(\mathbf{u}_k)}{\partial \mathbf{u}} & \mathbf{0}_{n_{\mu} \times n_{\mathcal{A}}} & \mathbf{0}_{n_{\mu} \times n_{\mu}} \end{bmatrix} \begin{bmatrix} \Delta \mathbf{u}_k \\ \Delta \boldsymbol{\nu}_{\mathcal{A},k} \\ \Delta \boldsymbol{\mu}_k \end{bmatrix} = - \begin{bmatrix} \frac{\partial \mathcal{L}(\mathbf{u}_k, \boldsymbol{\nu}_{\mathcal{A},k}, \boldsymbol{\mu}_k)}{\partial \mathbf{u}} \\ \mathbf{g}_{\mathcal{A}}(\mathbf{u}_k) \\ \mathbf{h}(\mathbf{u}_k) \end{bmatrix}$$

Or using Hessian and Jacobian notation:

$$\underbrace{\begin{bmatrix} \mathbf{H}_{\mathcal{L}}(\mathbf{u}_k, \boldsymbol{\nu}_{\mathcal{A},k}, \boldsymbol{\mu}_k) & \mathbf{J}_{\mathbf{g}_{\mathcal{A}}}(\mathbf{u}_k)^{\mathrm{T}} & \mathbf{J}_{\mathbf{h}}(\mathbf{u}_k)^{\mathrm{T}} \\ \mathbf{J}_{\mathbf{g}_{\mathcal{A}}}(\mathbf{u}_k) & \mathbf{0} & \mathbf{0} \\ \mathbf{J}_{\mathbf{h}}(\mathbf{u}_k) & \mathbf{0} & \mathbf{0} \end{bmatrix}}_{\mathcal{H}\ (n_u + n_{\mathcal{A}} + n_{\mu} \times n_u + n_{\mathcal{A}} + n_{\mu})} \begin{bmatrix} \Delta \mathbf{u}_k \\ \Delta \boldsymbol{\nu}_{\mathcal{A},k} \\ \Delta \boldsymbol{\mu}_k \end{bmatrix} = - \begin{bmatrix} \mathbf{J}_{\mathcal{L}}(\mathbf{u}_k, \boldsymbol{\nu}_{\mathcal{A},k}, \boldsymbol{\mu}_k)^{\mathrm{T}} \\ \mathbf{g}_{\mathcal{A}}(\mathbf{u}_k) \\ \mathbf{h}(\mathbf{u}_k) \end{bmatrix}$$

$\Rightarrow$ Inverting matrix $\mathcal{H}$ allows computing the shift vector…

3.2. SQP – **Compute the Shift Vectors (Newton's step)**

- The shift vector can be calculated as:

$$\begin{bmatrix} \Delta \mathbf{u}_k \\ \Delta \boldsymbol{\nu}_{\mathcal{A},k} \\ \Delta \boldsymbol{\mu}_k \end{bmatrix} = -\mathcal{H}^{-1} \begin{bmatrix} \mathbf{J}_{\mathcal{L}}(\mathbf{u}_k, \boldsymbol{\nu}_{\mathcal{A},k}, \boldsymbol{\mu}_k)^{\mathrm{T}} \\ \mathbf{g}_{\mathcal{A}}(\mathbf{u}_k) \\ \mathbf{h}(\mathbf{u}_k) \end{bmatrix}$$

with $\begin{bmatrix} \mathbf{u}_{k+1} \\ \boldsymbol{\nu}_{\mathcal{A},k+1} \\ \boldsymbol{\mu}_{k+1} \end{bmatrix} = \begin{bmatrix} \mathbf{u}_k \\ \boldsymbol{\nu}_{\mathcal{A},k} \\ \boldsymbol{\mu}_k \end{bmatrix} + \begin{bmatrix} \Delta \mathbf{u}_k \\ \Delta \boldsymbol{\nu}_{\mathcal{A},k} \\ \Delta \boldsymbol{\mu}_k \end{bmatrix}$ (applying the shift vector)

and $\mathcal{H} := \begin{bmatrix} \mathbf{H}_{\mathcal{L}}(\mathbf{u}_k, \boldsymbol{\nu}_{\mathcal{A},k}, \boldsymbol{\mu}_k) & \mathbf{J}_{\mathbf{g}_{\mathcal{A}}}(\mathbf{u}_k)^{\mathrm{T}} & \mathbf{J}_{\mathbf{h}}(\mathbf{u}_k)^{\mathrm{T}} \\ \mathbf{J}_{\mathbf{g}_{\mathcal{A}}}(\mathbf{u}_k) & \mathbf{0} & \mathbf{0} \\ \mathbf{J}_{\mathbf{h}}(\mathbf{u}_k) & \mathbf{0} & \mathbf{0} \end{bmatrix}$

- In practice, a line search is required to reduce the length of the shift vector (similarly to NG-method in Chapter 2.3.)
- How to efficiently compute $\mathbf{H}_{\mathcal{L}}$ and hence $\mathcal{H}$? (BFGS method)

3.2. Hessian Estimation by BFGS

- The Hessian is usually time consuming to compute via finite differences. That is why, **the Hessian is estimated using an algebraic expression based on a line search and the knowledge of the Jacobian**.
- The most commonly used method to estimate a Hessian matrix is the **BFGS*** method:

$$\mathbf{B}_{k+1} = \mathbf{B}_k + \mathbf{f}_{BFGS}(\mathbf{J}_{k+1}, \mathbf{J}_k, \mathbf{B}_k, \alpha_k) \quad \text{with } \mathbf{B}_0 = \mathbf{I}$$

$$\text{so that } \mathbf{B}_{k+1} \approx \mathbf{H}_{k+1}$$

* **B**royden (UK-Math., 1933-2011), **F**letcher (UK-Math., born in 1939), **G**oldfarb (US-Math, born in 1949), **S**hanno (US-Math., born in 1936)

3.2. MATLAB Nonlinear Optimizers for **one variable**

- **Optimization of one variable**
  - `fminbnd` Minimum on an interval
- **MATLAB fminbnd**:
  - `[x,fval,exitflag] = fminbnd(fun,x1,x2,options,...)`
- **MATLAB optimset**:
  - `options = optimset('name1',value1,'name2',value2)`

3.2. MATLAB Nonlinear Optimizers for **multiple variables**

- **Optimization of multiple variables**
  - `fminunc` Unconstrained minimization (see description in Chapter 2.3)
  - `fmincon` Constrained minimization
  - `quadprog` Constrained QP minimization
- **MATLAB fmincon**: $\mathbf{A}\,\mathbf{x} \leq \mathbf{b}$, $\mathbf{A}_{eq}\,\mathbf{x} = \mathbf{b}_{eq}$, $\mathbf{lb} \leq \mathbf{x} \leq \mathbf{ub}$
  - `[x,fval,exitflag,output,lambda,J,H] = fminunc(fun,x0,A,b,Aeq,beq,lb,ub,nonlcon,options,...)`
- **MATLAB quadprog**: $\min_{\mathbf{x}} \ \phi(\mathbf{x}) = \frac{1}{2}\mathbf{x}^{\mathrm{T}}\mathbf{H}\,\mathbf{x} + \mathbf{f}^{\mathrm{T}}\mathbf{x}$
  - `[x,fval,exitflag,output,lambda] = quadprog(H,f,A,b,Aeq,beq,lb,ub,x0,options,...)`
- **MATLAB optimoptions**:
  - `options = optimoptions(SolverName,'name1',value1)`

3.2. Exercise: **Static Optimization**

- Consider the **last exercise of Chapter 1.2**.
- **Dynamic model:** Assume that the dynamic model is known from the last exercise of Chapter 2.3.
- **Find the optimal flowrate[1] of species *B* in the 1st phase of reaction** so that the profit at the end of the batch is maximum[2].
- **Find the optimal flowrate[1] of species *B* in the 1st phase of reaction** so that the profit at the end of the batch is maximum[2] and the concentration of side product *C* at the end is ≤ 0.6 mol/L.
- **Verify with a response surface that the profit is maximum**.

1. Physical limits: $q_{in,A} \in [0,10]$ L/ut;
2. Prices: *A*: -10, *B*: -20, *C*: 0, *D*: 50, *Solvent*: 0 (USD per mol/L)

3.2. Exercise: **Dynamic Optimization**

- Consider the **last exercise of Chapter 1.2**.
- **Dynamic model:** Assume that the dynamic model is known from the last exercise of Chapter 2.3.
- **Find the optimal flowrate profile[1] of species *B* all along the reaction** so that the profit at the end of the batch is maximum[2].
- **Find the optimal flowrate profile[1] of species *B* all along the reaction** so that the profit at the end of the batch is maximum[2] and the concentration of side product *C* at the end is less or equal to 0.6 mol/L.
- **Find the optimal flowrate profile of species *B* all along the reaction** so that the profit at the end of the batch is maximum[2] and the concentrations of dosed *B* and side product *C* all along the reaction are ≤ 0.2 and ≤ 0.6 mol/L, respectively.

1. Physical limits: $q_{in,A} \in [0,10]$ L/ut;
2. Prices: *A*: -10, *B*: -20, *C*: 0, *D*: 50, *Solvent*: 0 (USD per mol/L)