Topology optimization of engineering systems

Florian Feppon

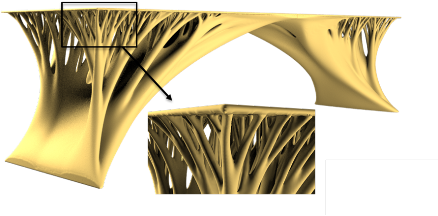
Spring 2022 – Seminar for Applied Mathematics

ETH zürich

What is topology optimization ?

(a) Siemens (2017)

(b) APWorks (2016)



(c) M2DO (Kambampati et. al. 2018)

(d) AIRBUS (2010)

What is topology optimization ?

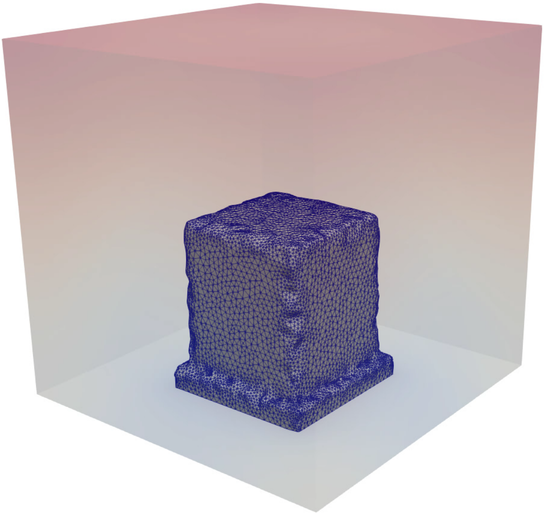


Figure: Minimization of the average temperature with a cooling material

Course outline

- 14 sessions from Feb 24th to June 2nd.

Course outline

- 14 sessions from Feb 24th to June 2nd.
- Evaluations: **pass** or **fail**. 20' oral presentation of a personal project or presentation of a journal article.

Course outline

Prospective structure of the course:

Session	Date	Online	Topic
1	24/02/2022	Presential	Nonlinear constrained optimization (part 1)
2	03/03/2022	Presential	Nonlinear constrained optimization (part 2)
3	10/03/2022	Online	Topology optimization and automated generative design : perspectives and applications in the context of additive manufacturing
4	17/03/2022	Online	Common physical models in mechanical and aeronautic engineering. PDE and variational forms. Formulation of shape optimization problems.
5	24/03/2022	Presential	Shape differential calculus. Shape derivatives of volume and surface functionals.
6	31/03/2022	Presential	Shape derivatives of PDE constrained functionals. The adjoint state.
7	07/04/2022	**Online / Need to change the date!**	Shape derivatives of arbitrary functionals. Introduction to FreeFEM / Implementation
8	14/04/2022	Presential	Numerical shape evolution algorithms : moving mesh methods, implicit surfaces and body-fitted meshes
9	21/04/2022	Presential	General results about shape optimization. Homogenization, relaxed designs. The SIMP method
10	28/04/2022	Presential	Projects: numerical implementation with python library
11	05/05/2022	Presential or online	Domain decomposition methods and parallel computing
12	12/05/2022	Online	Advanced methods: geometric constraints/topological derivative
13	19/05/2022	Online	Project presentations
	26/05/2022		No class on May 26th
14	02/06/2022	Presential	Project presentations

Course outline

Prospective structure of the course:

Session	Date	Online	Topic
1	24/02/2022	Presential	Nonlinear constrained optimization (part 1)
2	03/03/2022	Presential	Nonlinear constrained optimization (part 2)
3	10/03/2022	Online	Topology optimization and automated generative design : perspectives and applications in the context of additive manufacturing
4	17/03/2022	Online	Common physical models in mechanical and aeronautic engineering. PDE and variational forms. Formulation of shape optimization problems.
5	24/03/2022	Presential	Shape differential calculus. Shape derivatives of volume and surface functionals.
6	31/03/2022	Presential	Shape derivatives of PDE constrained functionals. The adjoint state.
7	07/04/2022	**Online / Need to change the date!**	Shape derivatives of arbitrary functionals. Introduction to FreeFEM / Implementation
8	14/04/2022	Presential	Numerical shape evolution algorithms : moving mesh methods, implicit surfaces and body-fitted meshes
9	21/04/2022	Presential	General results about shape optimization. Homogenization, relaxed designs. The SIMP method
10	28/04/2022	Presential	Projects: numerical implementation with python library
11	05/05/2022	Presential or online	Domain decomposition methods and parallel computing
12	12/05/2022	Online	Advanced methods: geometric constraints/topological derivative
13	19/05/2022	Online	Project presentations
	26/05/2022		No class on May 26th
14	02/06/2022	Presential	Project presentations

Check the webpage of the course!

https://people.math.ethz.ch/~ffeppon/teaching.html

Course outline

Course material:

- My PhD thesis:
  Feppon, F. *Shape and topology optimization of multiphysics systems* (2019). Thèse de doctorat de l'Université Paris-Saclay préparée à l'École polytechnique.
- Lecture notes prepared for the Von Karmann Institute:
  Feppon, F. *Shape and topology optimization applied to Compact Heat Exchangers* (2021).

Lecture 1: nonlinear constrained optimization. Null space gradient flows

Florian Feppon

Spring 2022 – Seminar for Applied Mathematics

Shape optimization problems

Shape/Topology optimization is the mathematical art of generating shapes that best fulfill a proposed objective.
Generically, a design optimization problem arises under the form

$$\min_{\Omega \subset D} \quad J(\Omega)$$
$$s.t. \begin{cases} G_i(\Omega) = 0, & 1 \leq i \leq p \\ H_j(\Omega) \leq 0, & 1 \leq j \leq q \end{cases}$$

where

Shape optimization problems

Shape/Topology optimization is the mathematical art of generating shapes that best fulfill a proposed objective.
Generically, a design optimization problem arises under the form

$$\min_{\Omega \subset D} \quad J(\Omega)$$
$$s.t. \begin{cases} G_i(\Omega) = 0, & 1 \leq i \leq p \\ H_j(\Omega) \leq 0, & 1 \leq j \leq q \end{cases}$$

where

- Ω is an **open domain** sought to be optimized

Shape optimization problems

Shape/Topology optimization is the mathematical art of generating shapes that best fulfill a proposed objective.
Generically, a design optimization problem arises under the form

$$\min_{\Omega \subset D} \quad J(\Omega)$$
$$s.t. \begin{cases} G_i(\Omega) = 0, & 1 \leq i \leq p \\ H_j(\Omega) \leq 0, & 1 \leq j \leq q \end{cases}$$

where

- Ω is an **open domain** sought to be optimized
- J is an **objective function** to minimize (corresponding to a measure of the performance)

Shape optimization problems

Shape/Topology optimization is the mathematical art of generating shapes that best fulfill a proposed objective.
Generically, a design optimization problem arises under the form

$$\min_{\Omega \subset D} \quad J(\Omega)$$
$$s.t. \begin{cases} G_i(\Omega) = 0, & 1 \leq i \leq p \\ H_j(\Omega) \leq 0, & 1 \leq j \leq q \end{cases}$$

where

- Ω is an **open domain** sought to be optimized
- J is an **objective function** to minimize (corresponding to a measure of the performance)
- G_i and H_j are respectively p and q **equality and inequality constraints** (corresponding e.g. to industrial specifications to meet)

Shape optimization problems

Shape/Topology optimization is the mathematical art of generating shapes that best fulfill a proposed objective.
Generically, a design optimization problem arises under the form

$$\min_{\Omega \subset D} \quad J(\Omega)$$
$$s.t. \begin{cases} G_i(\Omega) = 0, & 1 \leq i \leq p \\ H_j(\Omega) \leq 0, & 1 \leq j \leq q \end{cases}$$

where

- Ω is an **open domain** sought to be optimized
- J is an **objective function** to minimize (corresponding to a measure of the performance)
- G_i and H_j are respectively p and q **equality and inequality constraints** (corresponding e.g. to industrial specifications to meet)

Shape optimization problems

Shape/Topology optimization is the mathematical art of generating shapes that best fulfill a proposed objective.
Generically, a design optimization problem arises under the form

$$\min_{\Omega \subset D} \quad J(\Omega)$$
$$s.t. \begin{cases} G_i(\Omega) = 0, & 1 \leq i \leq p \\ H_j(\Omega) \leq 0, & 1 \leq j \leq q \end{cases}$$

where

- Ω is an **open domain** sought to be optimized
- J is an **objective function** to minimize (corresponding to a measure of the performance)
- G_i and H_j are respectively p and q **equality and inequality constraints** (corresponding e.g. to industrial specifications to meet)

In industrial applications, $J(\Omega)$, $G_i(\Omega)$ or $H_j(\Omega)$ involve the solution u_Ω defined with respect to a PDE model posed on Ω.

Shape optimization problems

Shape/Topology optimization is the mathematical art of generating shapes that best fulfill a proposed objective.
Generically, a design optimization problem arises under the form

$$\min_{\Omega \subset D} \quad J(\Omega)$$
$$s.t. \begin{cases} G_i(\Omega) = 0, & 1 \leq i \leq p \\ H_j(\Omega) \leq 0, & 1 \leq j \leq q \end{cases}$$

where

- Ω is an **open domain** sought to be optimized
- J is an **objective function** to minimize (corresponding to a measure of the performance)
- G_i and H_j are respectively p and q **equality and inequality constraints** (corresponding e.g. to industrial specifications to meet)

In industrial applications, $J(\Omega)$, $G_i(\Omega)$ or $H_j(\Omega)$ involve the solution u_Ω defined with respect to a PDE model posed on Ω.
In the next lectures, we will learn how to compute shape derivatives of the functionals $J(\Omega)$, $G_i(\Omega)$, $H_i(\Omega)$ with respect to **arbitrary** shape deformations.

Shape optimization problems

Shape/Topology optimization is the mathematical art of generating shapes that best fulfill a proposed objective.
Generically, a design optimization problem arises under the form

$$\min_{\Omega \subset D} \quad J(\Omega)$$

$$s.t. \begin{cases} G_i(\Omega) = 0, & 1 \leq i \leq p \\ H_j(\Omega) \leq 0, & 1 \leq j \leq q \end{cases}$$

where

- Ω is an **open domain** sought to be optimized
- J is an **objective function** to minimize (corresponding to a measure of the performance)
- G_i and H_j are respectively p and q **equality and inequality constraints** (corresponding e.g. to industrial specifications to meet)

In industrial applications, $J(\Omega)$, $G_i(\Omega)$ or $H_j(\Omega)$ involve the solution u_Ω defined with respect to a PDE model posed on Ω.
In the next lectures, we will learn how to compute shape derivatives of the functionals $J(\Omega)$, $G_i(\Omega)$, $H_i(\Omega)$ with respect to **arbitrary** shape deformations.
Today, we focus on numerical algorithms for solving such optimization programs.

2. Null space gradient flows for constrained optimization

For the exposure, let us consider the general optimization problem

$$\min_{x\in\mathcal{X}} \quad J(x)$$
$$\text{s.t.} \begin{cases} \boldsymbol{g}(x) = 0 \\ \boldsymbol{h}(x) \leq 0, \end{cases}$$

with $J : \mathcal{X} \to \mathbb{R}$, $\boldsymbol{g} : \mathcal{X} \to \mathbb{R}^p$ and $\boldsymbol{h} : \mathcal{X} \to \mathbb{R}^q$ Fréchet differentiable.
The set $\mathcal{X}$ can be

- a finite dimensional vector space, $\mathcal{X} = \mathbb{R}^n$

2. Null space gradient flows for constrained optimization

For the exposure, let us consider the general optimization problem

$$\min_{x\in\mathcal{X}} \quad J(x)$$
$$\text{s.t.} \begin{cases} \boldsymbol{g}(x) = 0 \\ \boldsymbol{h}(x) \leq 0, \end{cases}$$

with $J : \mathcal{X} \to \mathbb{R}$, $\boldsymbol{g} : \mathcal{X} \to \mathbb{R}^p$ and $\boldsymbol{h} : \mathcal{X} \to \mathbb{R}^q$ Fréchet differentiable.
The set $\mathcal{X}$ can be

- a finite dimensional vector space, $\mathcal{X} = \mathbb{R}^n$
- a Hilbert space equipped with a scalar product $a(\cdot, \cdot)$, $\mathcal{X} = V$

For the exposure, let us consider the general optimization problem

$$\min_{x\in\mathcal{X}} \quad J(x)$$
$$\text{s.t.} \begin{cases} \boldsymbol{g}(x) = 0 \\ \boldsymbol{h}(x) \leq 0, \end{cases}$$

with $J : \mathcal{X} \to \mathbb{R}$, $\boldsymbol{g} : \mathcal{X} \to \mathbb{R}^p$ and $\boldsymbol{h} : \mathcal{X} \to \mathbb{R}^q$ Fréchet differentiable.
The set $\mathcal{X}$ can be

- a finite dimensional vector space, $\mathcal{X} = \mathbb{R}^n$
- a Hilbert space equipped with a scalar product $a(\cdot,\cdot)$, $\mathcal{X} = V$
- a "manifold", as in shape optimization:

$$\mathcal{X} = \{\Omega \subset D \,|\, \Omega \text{ Lipschitz } \}$$

Optimization algorithms aim at answering the question:

From a current guess $x_n \in \mathcal{X}$, how to select the next guess $x_{n+1} \in \mathcal{X}$ given objective J and constraints $\boldsymbol{g}$, $\boldsymbol{h}$?

Optimization algorithms aim at answering the question:

From a current guess $x_n \in \mathcal{X}$, how to select the next guess $x_{n+1} \in \mathcal{X}$ given objective J and constraints $\boldsymbol{g}$, $\boldsymbol{h}$?

Two challenges: decreasing $J(x_n)$, while making the constraints $\boldsymbol{g}(x_n) = 0$ and $\boldsymbol{h}(x_n) \leq 0$ better satisfied.

2. Null space gradient flows for constrained optimization

Optimization algorithms aim at answering the question:

From a current guess $x_n \in \mathcal{X}$, how to select the next guess $x_{n+1} \in \mathcal{X}$ given objective J and constraints $\boldsymbol{g}$, $\boldsymbol{h}$?

2. Null space gradient flows for constrained optimization

Optimization algorithms aim at answering the question:

From a current guess $x_n \in \mathcal{X}$, how to select the next guess $x_{n+1} \in \mathcal{X}$ given objective J and constraints $\boldsymbol{g}$, $\boldsymbol{h}$?

Two classes of algorithms, **black-box** and **gradient-based** algorithms:

- **Parameter explorations**: generate a random but hopefully "reasonable" population of sample guesses $(x_n)_{n\in I}$, compute $J(x_n)$, $g_i(x_n)$ and $h_i(x_n)$, and take the best candidate.

Optimization algorithms aim at answering the question:

From a current guess $x_n \in \mathcal{X}$, how to select the next guess $x_{n+1} \in \mathcal{X}$ given objective J and constraints $\boldsymbol{g}$, $\boldsymbol{h}$?

Two classes of algorithms, **black-box** and **gradient-based** algorithms:

- **Parameter explorations**: generate a random but hopefully "reasonable" population of sample guesses $(x_n)_{n\in I}$, compute $J(x_n)$, $g_i(x_n)$ and $h_i(x_n)$, and take the best candidate.
- **"Black-box" optimization**: compute several values of $J(x_n)$, $g_i(x_n)$, $h_i(x_n)$ for a "small" but "representative" population of $(x_n)_{n\in I}$, construct an approximate surrogate model of J, g_i, and h_i, and solve the optimization program with the surrogate model

Optimization algorithms aim at answering the question:

From a current guess $x_n \in \mathcal{X}$, how to select the next guess $x_{n+1} \in \mathcal{X}$ given objective J and constraints $\boldsymbol{g}$, $\boldsymbol{h}$?

Two classes of algorithms, **black-box** and **gradient-based** algorithms:

- **Parameter explorations**: generate a random but hopefully "reasonable" population of sample guesses $(x_n)_{n\in I}$, compute $J(x_n)$, $g_i(x_n)$ and $h_i(x_n)$, and take the best candidate.
- **"Black-box" optimization**: compute several values of $J(x_n)$, $g_i(x_n)$, $h_i(x_n)$ for a "small" but "representative" population of $(x_n)_{n\in I}$, construct an approximate surrogate model of J, g_i, and h_i, and solve the optimization program with the surrogate model
- **"gradient-based" algorithms**: use the knowledge of the derivatives of J, g_i and h_i to infer a descent direction $\boldsymbol{\xi}_n$. The update

$$x_{n+1} = x_n + h\boldsymbol{\xi}_n$$

for a sufficiently small $h > 0$ should lead a better candidate x_{n+1}.

2. Null space gradient flows for constrained optimization

Optimization algorithms aim at answering the question:

From a current guess $x_n \in \mathcal{X}$, how to select the next guess $x_{n+1} \in \mathcal{X}$ given objective J and constraints $\boldsymbol{g}$, $\boldsymbol{h}$?

Two classes of algorithms, **black-box** and **gradient-based** algorithms:

- **Parameter explorations**: generate a random but hopefully "reasonable" population of sample guesses $(x_n)_{n\in I}$, compute $J(x_n)$, $g_i(x_n)$ and $h_i(x_n)$, and take the best candidate.
- **"Black-box" optimization**: compute several values of $J(x_n)$, $g_i(x_n)$, $h_i(x_n)$ for a "small" but "representative" population of $(x_n)_{n\in I}$, construct an approximate surrogate model of J, g_i, and h_i, and solve the optimization program with the surrogate model
- **"gradient-based" algorithms**: use the knowledge of the derivatives of J, g_i and h_i to infer a descent direction $\boldsymbol{\xi}_n$. The update

$$x_{n+1} = x_n + h\boldsymbol{\xi}_n$$

for a sufficiently small $h > 0$ should lead a better candidate x_{n+1}.

2. Null space gradient flows for constrained optimization

Optimization algorithms aim at answering the question:

From a current guess $x_n \in \mathcal{X}$, how to select the next guess $x_{n+1} \in \mathcal{X}$ given objective J and constraints $\boldsymbol{g}$, $\boldsymbol{h}$?

Two classes of algorithms, **black-box** and **gradient-based** algorithms:

- **Parameter explorations**: generate a random but hopefully "reasonable" population of sample guesses $(x_n)_{n \in I}$, compute $J(x_n)$, $g_i(x_n)$ and $h_i(x_n)$, and take the best candidate.
- **"Black-box" optimization**: compute several values of $J(x_n)$, $g_i(x_n)$, $h_i(x_n)$ for a "small" but "representative" population of $(x_n)_{n \in I}$, construct an approximate surrogate model of J, g_i, and h_i, and solve the optimization program with the surrogate model
- **"gradient-based" algorithms**: use the knowledge of the derivatives of J, g_i and h_i to infer a descent direction $\boldsymbol{\xi}_n$. The update

$$x_{n+1} = x_n + h\boldsymbol{\xi}_n$$

for a sufficiently small $h > 0$ should lead a better candidate x_{n+1}.

For shape optimization, a single computation of $J(x_n)$, $g_i(x_n)$ and $h_i(x_n)$ requires to solve PDEs: it is very costly. Black box methods cannot be considered as good methods.

Gradient methods are very powerful and can be used to solve constrained shape optimization problems.

Gradient methods are very powerful and can be used to solve constrained shape optimization problems.

The price to pay is that they require the knowledge of the gradient.

Outline

Outline

Outline

Outline

1. Reminders on smooth constrained optimization
2. Gradient flows for unconstrained optimization
3. Constrained optimization:
   3.1 Extension to equality constrained optimization
   3.2 Extension to equality and inequality constrained optimization
4. Numerical implementation
5. Numerical examples

Outline

Differential vs. gradients

Definition 1

1. $\boldsymbol{g} : V \to \mathbb{R}^p$ is *differentiable* at $x \in V$ if there exists a continuous linear mapping $\mathrm{D}\boldsymbol{g}(x) : V \to \mathbb{R}^p$ such that

$$\boldsymbol{g}(x+h) = \boldsymbol{g}(x) + \mathrm{D}\boldsymbol{g}(x)h + o(h) \text{ with } \frac{o(h)}{||h||_V} \xrightarrow{h \to 0} 0.$$

Differential vs. gradients

Definition 1

1. $\boldsymbol{g} : V \to \mathbb{R}^p$ is *differentiable* at $x \in V$ if there exists a continuous linear mapping $\mathrm{D}\boldsymbol{g}(x) : V \to \mathbb{R}^p$ such that

$$\boldsymbol{g}(x+h) = \boldsymbol{g}(x) + \mathrm{D}\boldsymbol{g}(x)h + o(h) \text{ with } \frac{o(h)}{||h||_V} \xrightarrow{h \to 0} 0.$$

$\mathrm{D}\boldsymbol{g}(x)$ is called the **Fréchet derivative** of $\boldsymbol{g}$ at x.

Differential vs. gradients

Definition 1

1. $\boldsymbol{g} : V \to \mathbb{R}^p$ is *differentiable* at $x \in V$ if there exists a continuous linear mapping $\mathrm{D}\boldsymbol{g}(x) : V \to \mathbb{R}^p$ such that

$$\boldsymbol{g}(x+h) = \boldsymbol{g}(x) + \mathrm{D}\boldsymbol{g}(x)h + o(h) \text{ with } \frac{o(h)}{||h||_V} \xrightarrow{h\to 0} 0.$$

   $\mathrm{D}\boldsymbol{g}(x)$ is called the **Fréchet derivative** of $\boldsymbol{g}$ at x.

2. If $\boldsymbol{g} : V \to \mathbb{R}^p$ is differentiable, for any $\boldsymbol{\mu} \in \mathbb{R}^p$, there exists a unique vector $\mathrm{D}\boldsymbol{g}(x)^T\boldsymbol{\mu} \in V$ satisfying

$$\forall \boldsymbol{\mu} \in \mathbb{R}^p, \forall \boldsymbol{\xi} \in V, \ \langle \mathrm{D}\boldsymbol{g}(x)^T\boldsymbol{\mu}, \boldsymbol{\xi}\rangle_V = \boldsymbol{\mu}^T \mathrm{D}\boldsymbol{g}(x)\boldsymbol{\xi},$$

Differential vs. gradients

Definition 1

1. $\boldsymbol{g} : V \to \mathbb{R}^p$ is *differentiable* at $x \in V$ if there exists a continuous linear mapping $\mathrm{D}\boldsymbol{g}(x) : V \to \mathbb{R}^p$ such that

$$\boldsymbol{g}(x+h) = \boldsymbol{g}(x) + \mathrm{D}\boldsymbol{g}(x)h + o(h) \text{ with } \frac{o(h)}{||h||_V} \xrightarrow{h \to 0} 0.$$

   $\mathrm{D}\boldsymbol{g}(x)$ is called the **Fréchet derivative** of $\boldsymbol{g}$ at x.

2. If $\boldsymbol{g} : V \to \mathbb{R}^p$ is differentiable, for any $\boldsymbol{\mu} \in \mathbb{R}^p$, there exists a unique vector $\mathrm{D}\boldsymbol{g}(x)^{\mathcal{T}}\boldsymbol{\mu} \in V$ satisfying

$$\forall \boldsymbol{\mu} \in \mathbb{R}^p, \forall \boldsymbol{\xi} \in V, \ \langle \mathrm{D}\boldsymbol{g}(x)^{\mathcal{T}}\boldsymbol{\mu}, \boldsymbol{\xi} \rangle_V = \boldsymbol{\mu}^T \mathrm{D}\boldsymbol{g}(x)\boldsymbol{\xi},$$

   The linear operator $\mathrm{D}\boldsymbol{g}(x)^{\mathcal{T}} : \mathbb{R}^p \to V$ thus defined is called the **linear transpose** of $\mathrm{D}\boldsymbol{g}(x)$.

Differential vs. gradients

Definition 1

1. $\boldsymbol{g} : V \to \mathbb{R}^p$ is *differentiable* at $x \in V$ if there exists a continuous linear mapping $\mathrm{D}\boldsymbol{g}(x) : V \to \mathbb{R}^p$ such that

$$\boldsymbol{g}(x+h) = \boldsymbol{g}(x) + \mathrm{D}\boldsymbol{g}(x)h + o(h) \text{ with } \frac{o(h)}{||h||_V} \xrightarrow{h\to 0} 0.$$

   $\mathrm{D}\boldsymbol{g}(x)$ is called the **Fréchet derivative** of $\boldsymbol{g}$ at x.

2. If $\boldsymbol{g} : V \to \mathbb{R}^p$ is differentiable, for any $\boldsymbol{\mu} \in \mathbb{R}^p$, there exists a unique vector $\mathrm{D}\boldsymbol{g}(x)^{\mathcal{T}}\boldsymbol{\mu} \in V$ satisfying

$$\forall \boldsymbol{\mu} \in \mathbb{R}^p, \forall \boldsymbol{\xi} \in V, \ \langle \mathrm{D}\boldsymbol{g}(x)^{\mathcal{T}}\boldsymbol{\mu}, \boldsymbol{\xi}\rangle_V = \boldsymbol{\mu}^T \mathrm{D}\boldsymbol{g}(x)\boldsymbol{\xi},$$

   The linear operator $\mathrm{D}\boldsymbol{g}(x)^{\mathcal{T}} : \mathbb{R}^p \to V$ thus defined is called the **linear transpose** of $\mathrm{D}\boldsymbol{g}(x)$.

3. If $J : V \to \mathbb{R}$ is a scalar function differentiable at $x \in V$, there exists a unique vector $\nabla J(x) \in V$ satisfying

$$\forall \boldsymbol{\xi} \in V, \ \langle \nabla J(x), \boldsymbol{\xi}\rangle_V = \mathrm{D}J(x)\boldsymbol{\xi}.$$

Differential vs. gradients

Definition 1

1. $\boldsymbol{g} : V \to \mathbb{R}^p$ is *differentiable* at $x \in V$ if there exists a continuous linear mapping $\mathrm{D}\boldsymbol{g}(x) : V \to \mathbb{R}^p$ such that

$$\boldsymbol{g}(x+h) = \boldsymbol{g}(x) + \mathrm{D}\boldsymbol{g}(x)h + o(h) \text{ with } \frac{o(h)}{||h||_V} \xrightarrow{h\to 0} 0.$$

   $\mathrm{D}\boldsymbol{g}(x)$ is called the **Fréchet derivative** of $\boldsymbol{g}$ at x.

2. If $\boldsymbol{g} : V \to \mathbb{R}^p$ is differentiable, for any $\boldsymbol{\mu} \in \mathbb{R}^p$, there exists a unique vector $\mathrm{D}\boldsymbol{g}(x)^{\mathcal{T}}\boldsymbol{\mu} \in V$ satisfying

$$\forall \boldsymbol{\mu} \in \mathbb{R}^p, \forall \boldsymbol{\xi} \in V,\ \langle \mathrm{D}\boldsymbol{g}(x)^{\mathcal{T}}\boldsymbol{\mu}, \boldsymbol{\xi}\rangle_V = \boldsymbol{\mu}^T \mathrm{D}\boldsymbol{g}(x)\boldsymbol{\xi},$$

   The linear operator $\mathrm{D}\boldsymbol{g}(x)^{\mathcal{T}} : \mathbb{R}^p \to V$ thus defined is called the **linear transpose** of $\mathrm{D}\boldsymbol{g}(x)$.

3. If $J : V \to \mathbb{R}$ is a scalar function differentiable at $x \in V$, there exists a unique vector $\nabla J(x) \in V$ satisfying

$$\forall \boldsymbol{\xi} \in V,\ \langle \nabla J(x), \boldsymbol{\xi}\rangle_V = \mathrm{D}J(x)\boldsymbol{\xi}.$$

   The vector $\nabla J(x) \in V$ is called the **gradient** of J at x.

Differential vs. gradients

- Do not confuse the **gradient** ∇J and the **differential** $\mathrm{D}J$!

Differential vs. gradients

- Do not confuse the **gradient** ∇J and the **differential** $\mathrm{D}J$!

Differential vs. gradients

- Do not confuse the **gradient** ∇J and the **differential** $\mathrm{D}J$!This difference will be important for shape optimization algorithms.

Differential vs. gradients

- Do not confuse the **gradient** ∇J and the **differential** $\mathrm{D}J$!This difference will be important for shape optimization algorithms.
- If $\boldsymbol{g}(x) = (g_i(x))_{1\leq i\leq p}$, then $\mathrm{D}g^{\mathcal{T}} = \begin{bmatrix} \nabla g_1 & \nabla g_2 & \dots & \nabla g_p \end{bmatrix}$

Differential vs. gradients

- Do not confuse the **gradient** ∇J and the **differential** $\mathrm{D}J$!This difference will be important for shape optimization algorithms.
- If $\boldsymbol{g}(x) = (g_i(x))_{1\leq i\leq p}$, then $\mathrm{D}g^{\mathcal{T}} = \begin{bmatrix}\nabla g_1 & \nabla g_2 & \dots & \nabla g_p\end{bmatrix}$
- If $V = \mathbb{R}^N$ and $\langle\cdot,\cdot\rangle_V$ is the usual Euclidean inner product, then $\mathrm{D}\boldsymbol{g} = (\partial_j g_i)_{1\leq i\leq p, 1\leq j\leq N}$ and $\mathrm{D}\boldsymbol{g}^{\mathcal{T}} = \mathrm{D}\boldsymbol{g}^T$.

Differential vs. gradients

- Do not confuse the **gradient** ∇J and the **differential** $\mathrm{D}J$!This difference will be important for shape optimization algorithms.
- If $\boldsymbol{g}(x) = (g_i(x))_{1\leq i\leq p}$, then $\mathrm{D}g^{\mathcal{T}} = \begin{bmatrix}\nabla g_1 & \nabla g_2 & \dots & \nabla g_p\end{bmatrix}$
- If $V = \mathbb{R}^N$ and $\langle\cdot,\cdot\rangle_V$ is the usual Euclidean inner product, then $\mathrm{D}\boldsymbol{g} = (\partial_j g_i)_{1\leq i\leq p, 1\leq j\leq N}$ and $\mathrm{D}\boldsymbol{g}^{\mathcal{T}} = \mathrm{D}\boldsymbol{g}^T$.

Differential vs. gradients

- Do not confuse the **gradient** ∇J and the **differential** $\mathrm{D}J$!This difference will be important for shape optimization algorithms.
- If $\boldsymbol{g}(x) = (g_i(x))_{1\leq i\leq p}$, then $\mathrm{D}g^{\mathcal{T}} = \begin{bmatrix}\nabla g_1 & \nabla g_2 & \dots & \nabla g_p\end{bmatrix}$
- If $V = \mathbb{R}^N$ and $\langle\cdot,\cdot\rangle_V$ is the usual Euclidean inner product, then $\mathrm{D}\boldsymbol{g} = (\partial_j g_i)_{1\leq i\leq p, 1\leq j\leq N}$ and $\mathrm{D}\boldsymbol{g}^{\mathcal{T}} = \mathrm{D}\boldsymbol{g}^T$. Careful: physicists usually write $\nabla\boldsymbol{g} = (\partial_j g_i)_{1\leq i\leq p, 1\leq j\leq N}$ for $\mathrm{D}\boldsymbol{g}$ though $\mathrm{D}\boldsymbol{g}$ is not a gradient.

Differential vs. gradients

- Do not confuse the **gradient** ∇J and the **differential** $\mathrm{D}J$!This difference will be important for shape optimization algorithms.
- If $\boldsymbol{g}(x) = (g_i(x))_{1\leq i\leq p}$, then $\mathrm{D}g^{\mathcal{T}} = \begin{bmatrix}\nabla g_1 & \nabla g_2 & \dots & \nabla g_p\end{bmatrix}$
- If $V = \mathbb{R}^N$ and $\langle\cdot,\cdot\rangle_V$ is the usual Euclidean inner product, then $\mathrm{D}\boldsymbol{g} = (\partial_j g_i)_{1\leq i\leq p, 1\leq j\leq N}$ and $\mathrm{D}\boldsymbol{g}^{\mathcal{T}} = \mathrm{D}\boldsymbol{g}^T$. Careful: physicists usually write $\nabla\boldsymbol{g} = (\partial_j g_i)_{1\leq i\leq p, 1\leq j\leq N}$ for $\mathrm{D}\boldsymbol{g}$ though $\mathrm{D}\boldsymbol{g}$ is not a gradient.
- If $V = V$ and $\langle\boldsymbol{\xi},\boldsymbol{\xi}\rangle_V := \boldsymbol{\xi}^T A\boldsymbol{\xi}$ for $A \in \mathbb{R}^{n\times n}$ a positive definite matrix, then $\mathrm{D}\boldsymbol{g}^{\mathcal{T}} = A^{-1}\mathrm{D}\boldsymbol{g}^T$.

Differential vs. gradients

- Do not confuse the **gradient** ∇J and the **differential** $\mathrm{D}J$!This difference will be important for shape optimization algorithms.
- If $\boldsymbol{g}(x) = (g_i(x))_{1\le i\le p}$, then $\mathrm{D}\boldsymbol{g}^{\mathcal{T}} = \begin{bmatrix}\nabla g_1 & \nabla g_2 & \dots & \nabla g_p\end{bmatrix}$
- If $V = \mathbb{R}^N$ and $\langle\cdot,\cdot\rangle_V$ is the usual Euclidean inner product, then $\mathrm{D}\boldsymbol{g} = (\partial_j g_i)_{1\le i\le p, 1\le j\le N}$ and $\mathrm{D}\boldsymbol{g}^{\mathcal{T}} = \mathrm{D}\boldsymbol{g}^T$. Careful: physicists usually write $\nabla\boldsymbol{g} = (\partial_j g_i)_{1\le i\le p, 1\le j\le N}$ for $\mathrm{D}\boldsymbol{g}$ though $\mathrm{D}\boldsymbol{g}$ is not a gradient.
- If $V = V$ and $\langle\boldsymbol{\xi},\boldsymbol{\xi}\rangle_V := \boldsymbol{\xi}^T A\boldsymbol{\xi}$ for $A\in\mathbb{R}^{n\times n}$ a positive definite matrix, then $\mathrm{D}\boldsymbol{g}^{\mathcal{T}} = A^{-1}\mathrm{D}\boldsymbol{g}^T$.
- The matrix $\mathrm{D}\boldsymbol{g}\mathrm{D}\boldsymbol{g}^{\mathcal{T}}\in\mathbb{R}^{p\times p}$ has entries

$$(\mathrm{D}\boldsymbol{g}\mathrm{D}\boldsymbol{g}^{\mathcal{T}})_{ij} = \langle\nabla g_i, \nabla g_j\rangle_V = \mathrm{D}g_i(x)(\nabla g_j(x)).$$

Consider the optimization problem

$$\min_{x \in V} \quad J(x) \\ \text{s.t.} \begin{cases} \boldsymbol{g}(x) = 0 \\ \boldsymbol{h}(x) \leq 0, \end{cases} \tag{1}$$

- The set $\{x \in V \,|\, \boldsymbol{g}(x) = 0 \text{ and } \boldsymbol{h}(x) \leq 0\}$ is called **the feasible domain**.

Consider the optimization problem

$$\min_{x \in V} \quad J(x) \qquad \text{s.t.} \begin{cases} \boldsymbol{g}(x) = 0 \\ \boldsymbol{h}(x) \le 0, \end{cases} \tag{1}$$

- The set $\{x \in V \,|\, \boldsymbol{g}(x) = 0 \text{ and } \boldsymbol{h}(x) \le 0\}$ is called **the feasible domain**.
- x^* is called a **local minimizer** if there is an open neighborhood $\mathcal{U}$ such that x^* solves the minimization problem

$$\min_{x \in \mathcal{U}} \quad J(x) \qquad \text{s.t.} \begin{cases} \boldsymbol{g}(x) = 0 \\ \boldsymbol{h}(x) \le 0, \end{cases}$$

Consider the optimization problem

$$\min_{x \in V} \quad J(x) \\ \text{s.t.} \begin{cases} \boldsymbol{g}(x) = 0 \\ \boldsymbol{h}(x) \leq 0, \end{cases} \tag{1}$$

- The set $\{x \in V \mid \boldsymbol{g}(x) = 0 \text{ and } \boldsymbol{h}(x) \leq 0\}$ is called **the feasible domain**.
- x^* is called a **local minimizer** if there is an open neighborhood $\mathcal{U}$ such that x^* solves the minimization problem

$$\min_{x \in \mathcal{U}} \quad J(x) \\ \text{s.t.} \begin{cases} \boldsymbol{g}(x) = 0 \\ \boldsymbol{h}(x) \leq 0, \end{cases}$$

- if x^* is solution to eq. (1), then x^* is called a **global minimizer**.

Consider the optimization problem

$$\min_{x \in V} \quad J(x)$$
$$\text{s.t.} \begin{cases} \boldsymbol{g}(x) = 0 \\ \boldsymbol{h}(x) \leq 0, \end{cases}$$

- A constraint g_i or h_j is called **violated** at $x \in V$ if $g_i(x) \neq 0$ or $h_j(x) > 0$, is called **satisfied** otherwise;

Consider the optimization problem

$$\min_{x \in V} \quad J(x)$$
$$\text{s.t.} \begin{cases} \boldsymbol{g}(x) = 0 \\ \boldsymbol{h}(x) \leq 0, \end{cases}$$

- A constraint g_i or h_j is called **violated** at $x \in V$ if $g_i(x) \neq 0$ or $h_j(x) > 0$, is called **satisfied** otherwise;
- A constraint h_j is called **active** at $x \in V$ if $h_j(x) = 0$

Differential vs. gradients

Consider the optimization problem

$$\min_{x\in V} \quad J(x)$$
$$\text{s.t.} \begin{cases} \boldsymbol{g}(x) = 0 \\ \boldsymbol{h}(x) \leq 0, \end{cases}$$

Differential vs. gradients

Consider the optimization problem

$$\min_{x \in V} \quad J(x)$$
$$\text{s.t.} \begin{cases} \boldsymbol{g}(x) = 0 \\ \boldsymbol{h}(x) \leq 0, \end{cases}$$

Denote by $\widetilde{I}(x)$ the set of **active** or **violated** inequality constraints:

$$\widetilde{I}(x) = \{i \in \{1, \ldots, q\} \mid h_i(x) \geqslant 0\},$$

Differential vs. gradients

Consider the optimization problem

$$\min_{x \in V} \quad J(x)$$
$$\text{s.t.} \begin{cases} \boldsymbol{g}(x) = 0 \\ \boldsymbol{h}(x) \leq 0, \end{cases}$$

Denote by $\widetilde{I}(x)$ the set of **active** or **violated** inequality constraints:

$$\widetilde{I}(x) = \{i \in \{1, \dots, q\} \,|\, h_i(x) \geqslant 0\},$$

and

$$\boldsymbol{C}_{\widetilde{I}(x)} = \left[\boldsymbol{g}(x) \quad | \quad (h_i(x))_{i \in \widetilde{I}(x)}\right]^T, \qquad \widetilde{q}(x) := \#\widetilde{I}(x)$$

the vector of corresponding constraints and their number.

Differential vs. gradients

Consider the optimization problem

$$\min_{x \in V} \quad J(x)$$
$$\text{s.t.} \begin{cases} \boldsymbol{g}(x) = 0 \\ \boldsymbol{h}(x) \leq 0, \end{cases}$$

Denote by $\widetilde{I}(x)$ the set of **active** or **violated** inequality constraints:

$$\widetilde{I}(x) = \{i \in \{1, \ldots, q\} \,|\, h_i(x) \geqslant 0\},$$

and

$$\boldsymbol{C}_{\widetilde{I}(x)} = \left[\boldsymbol{g}(x) \quad | \quad (h_i(x))_{i \in \widetilde{I}(x)}\right]^T, \qquad \widetilde{q}(x) := \#\widetilde{I}(x)$$

the vector of corresponding constraints and their number.

We say that the constraints are **qualified** at $x \in V$ if they are linearly independent:

$$\operatorname{rank}(\mathrm{D}C_{\widetilde{I}(x)}) = p + \widetilde{q}(x).$$

Differential vs. gradients

Consider the optimization problem

$$\min_{x \in V} \quad J(x)$$
$$\text{s.t.} \begin{cases} \boldsymbol{g}(x) = 0 \\ \boldsymbol{h}(x) \leq 0, \end{cases}$$

Denote by $\widetilde{I}(x)$ the set of **active** or **violated** inequality constraints:

$$\widetilde{I}(x) = \{i \in \{1, \dots, q\} \mid h_i(x) \geqslant 0\},$$

and

$$\boldsymbol{C}_{\widetilde{I}(x)} = \Big[\boldsymbol{g}(x) \quad | \quad (h_i(x))_{i \in \widetilde{I}(x)}\Big]^T, \qquad \widetilde{q}(x) := \#\widetilde{I}(x)$$

the vector of corresponding constraints and their number.

We say that the constraints are **qualified** at $x \in V$ if they are linearly independent:

$$\text{rank}(\mathrm{D}C_{\widetilde{I}(x)}) = p + \widetilde{q}(x).$$

This is equivalent to

$$\mathrm{D}C_{\widetilde{I(x)}} \mathrm{D}C^{\mathcal{T}}_{\widetilde{I(x)}} \text{ is invertible.}$$

Differential vs. gradients

Proposition 1

Assume that J, $\boldsymbol{g}$ and $\boldsymbol{h}$ are $\mathcal{C}^1$ functions and that the constraints are qualified.

Differential vs. gradients

Proposition 1

Assume that J, $\boldsymbol{g}$ and $\boldsymbol{h}$ are $\mathcal{C}^1$ functions and that the constraints are qualified. Then if x^* is a local minimizer, then there exist $(\boldsymbol{\lambda}^*, \boldsymbol{\mu}^*) \in \mathbb{R}^d \times \mathbb{R}_+^{\widetilde{q}(x)}$ such that

$$\nabla J(x^*) + \mathrm{D}g(x^*)^{\mathcal{T}} \boldsymbol{\lambda}^* + \mathrm{D}h_{\widetilde{I}(x^*)}(x^*)^{\mathcal{T}} \boldsymbol{\mu}^* = 0 \tag{2}$$

Differential vs. gradients

Proposition 1

Assume that J, $\boldsymbol{g}$ and $\boldsymbol{h}$ are $\mathcal{C}^1$ functions and that the constraints are qualified. Then if x^* is a local minimizer, then there exist $(\boldsymbol{\lambda}^*, \boldsymbol{\mu}^*) \in \mathbb{R}^d \times \mathbb{R}_+^{\widetilde{q}(x)}$ such that

$$\nabla J(x^*) + \mathrm{D}g(x^*)^{\mathcal{T}} \boldsymbol{\lambda}^* + \mathrm{D}h_{\widetilde{I}(x^*)}(x^*)^{\mathcal{T}} \boldsymbol{\mu}^* = 0 \tag{2}$$

- eq. (2) is called the Karush, Kuhn and Tucker condition;

Differential vs. gradients

Proposition 1

Assume that J, $\boldsymbol{g}$ and $\boldsymbol{h}$ are $\mathcal{C}^1$ functions and that the constraints are qualified. Then if x^* is a local minimizer, then there exist $(\boldsymbol{\lambda}^*, \boldsymbol{\mu}^*) \in \mathbb{R}^d \times \mathbb{R}_+^{\widetilde{q}(x)}$ such that

$$\nabla J(x^*) + \mathrm{D}g(x^*)^{\mathcal{T}} \boldsymbol{\lambda}^* + \mathrm{D}h_{\widetilde{I}(x^*)}(x^*)^{\mathcal{T}} \boldsymbol{\mu}^* = 0 \tag{2}$$

- eq. (2) is called the Karush, Kuhn and Tucker condition;
- if there are no constraints, it reduces to the standard first order optimality condition

$$\nabla J(x^*) = 0$$

Differential vs. gradients

Proposition 1

Assume that J, $\mathbf{g}$ and $\mathbf{h}$ are $\mathcal{C}^1$ functions and that the constraints are qualified. Then if x^* is a local minimizer, then there exist $(\boldsymbol{\lambda}^*, \boldsymbol{\mu}^*) \in \mathbb{R}^d \times \mathbb{R}_+^{\widetilde{q}(x)}$ such that

$$\nabla J(x^*) + \mathrm{D}g(x^*)^{\mathcal{T}} \boldsymbol{\lambda}^* + \mathrm{D}h_{\widetilde{I}(x^*)}(x^*)^{\mathcal{T}} \boldsymbol{\mu}^* = 0 \tag{2}$$

- eq. (2) is called the Karush, Kuhn and Tucker condition;
- if there are no constraints, it reduces to the standard first order optimality condition

$$\nabla J(x^*) = 0$$

- for equality and inequality constraints, we shall interpret eq. (2) as the nullity of the gradient **projected tangentially to the constraints.**

Outline

Unconstrained optimization

Consider the unconstrained minimization problem

$$\min_{x \in V} \qquad J(x),$$

with $J : V \to \mathbb{R}$ differentiable.

Unconstrained optimization

Consider the unconstrained minimization problem

$$\min_{x \in V} \qquad J(x),$$

with $J : V \to \mathbb{R}$ differentiable.
The gradient $\nabla J(x)$ has two roles:

Unconstrained optimization

Consider the unconstrained minimization problem

$$\min_{x \in V} \quad J(x),$$

with $J : V \to \mathbb{R}$ differentiable.

The gradient $\nabla J(x)$ has two roles:

Lemma 2

- *$-\nabla J(x)$ is the "best descent direction" at x in the sense that*

$$-\frac{\nabla J(x)}{||\nabla J(x)||_V} = \begin{array}{c} \arg\min_{\boldsymbol{\xi} \in V} \quad \mathrm{D}J(x) \cdot \boldsymbol{\xi} \\ s.t. ||\boldsymbol{\xi}||_V \leq 1. \end{array}$$

Unconstrained optimization

Consider the unconstrained minimization problem

$$\min_{x \in V} \quad J(x),$$

with $J : V \to \mathbb{R}$ differentiable.
The gradient $\nabla J(x)$ has two roles:

Lemma 2

- *$-\nabla J(x)$ is the "best descent direction" at x in the sense that*

$$-\frac{\nabla J(x)}{||\nabla J(x)||_V} = \begin{array}{cc} \arg\min\limits_{\boldsymbol{\xi} \in V} & \mathrm{D}J(x) \cdot \boldsymbol{\xi} \\ & s.t. ||\boldsymbol{\xi}||_V \leq 1. \end{array}$$

- *If x is a local minimizer of J, then $\nabla J(x) = 0$.*

Unconstrained optimization

Consider the unconstrained minimization problem

$$\min_{x \in V} \qquad J(x),$$

with $J : V \to \mathbb{R}$ differentiable.

Unconstrained optimization

Consider the unconstrained minimization problem

$$\min_{x \in V} \quad J(x),$$

with $J : V \to \mathbb{R}$ differentiable.

The fixed step gradient method is

$$x_{n+1} = x_n - h\nabla J(x_n). \tag{3}$$

Unconstrained optimization

Consider the unconstrained minimization problem

$$\min_{x \in V} \qquad J(x),$$

with $J : V \to \mathbb{R}$ differentiable.

The fixed step gradient method is

$$x_{n+1} = x_n - h\nabla J(x_n). \tag{3}$$

For h sufficiently small, $J(x_{n+1}) = J(x_n) - h||\nabla J(x_n)||^2 + o(h) < J(x_n)$,

Unconstrained optimization

Consider the unconstrained minimization problem

$$\min_{x \in V} \qquad J(x),$$

with $J : V \to \mathbb{R}$ differentiable.

The fixed step gradient method is

$$x_{n+1} = x_n - h\nabla J(x_n). \tag{3}$$

For h sufficiently small, $J(x_{n+1}) = J(x_n) - h||\nabla J(x_n)||^2 + o(h) < J(x_n)$, J has decreased!

Unconstrained optimization

Consider the unconstrained minimization problem

$$\min_{x \in V} \quad J(x),$$

with $J : V \to \mathbb{R}$ differentiable.

The fixed step gradient method is

$$x_{n+1} = x_n - h\nabla J(x_n). \tag{3}$$

For h sufficiently small, $J(x_{n+1}) = J(x_n) - h||\nabla J(x_n)||^2 + o(h) < J(x_n)$, J has decreased!

The convergence analysis of the discrete scheme eq. (3) is delicate, it can be done for convex functions.

Unconstrained optimization

Consider the unconstrained minimization problem

$$\min_{x \in V} \qquad J(x),$$

with $J : V \to \mathbb{R}$ differentiable.

The fixed step gradient method is

$$x_{n+1} = x_n - h\nabla J(x_n). \tag{3}$$

For h sufficiently small, $J(x_{n+1}) = J(x_n) - h||\nabla J(x_n)||^2 + o(h) < J(x_n)$, J has decreased!

The convergence analysis of the discrete scheme eq. (3) is delicate, it can be done for convex functions. On the other hand, eq. (3) can be interpreted as the Euler method for the **gradient flow**

$$\frac{\mathrm{d}x}{\mathrm{d}t} = -\nabla J(x). \tag{4}$$

Unconstrained optimization

Consider the unconstrained minimization problem

$$\min_{x \in V} \quad J(x),$$

with $J : V \to \mathbb{R}$ differentiable.

The fixed step gradient method is

$$x_{n+1} = x_n - h\nabla J(x_n). \tag{3}$$

For h sufficiently small, $J(x_{n+1}) = J(x_n) - h||\nabla J(x_n)||^2 + o(h) < J(x_n)$, J has decreased!

The convergence analysis of the discrete scheme eq. (3) is delicate, it can be done for convex functions. On the other hand, eq. (3) can be interpreted as the Euler method for the **gradient flow**

$$\frac{\mathrm{d}x}{\mathrm{d}t} = -\nabla J(x). \tag{4}$$

It is easier to analyse eq. (4):

- $\frac{\mathrm{d}J(x)}{\mathrm{d}t} = -||\nabla J(x)||^2 < 0$ so $t \mapsto J(x(t))$ decreases along the trajectory $t \mapsto x(t)$;

Unconstrained optimization

Consider the unconstrained minimization problem

$$\min_{x \in V} \qquad J(x),$$

with $J : V \to \mathbb{R}$ differentiable.

The fixed step gradient method is

$$x_{n+1} = x_n - h\nabla J(x_n). \tag{3}$$

For h sufficiently small, $J(x_{n+1}) = J(x_n) - h||\nabla J(x_n)||^2 + o(h) < J(x_n)$, J has decreased!

The convergence analysis of the discrete scheme eq. (3) is delicate, it can be done for convex functions. On the other hand, eq. (3) can be interpreted as the Euler method for the **gradient flow**

$$\frac{\mathrm{d}x}{\mathrm{d}t} = -\nabla J(x). \tag{4}$$

It is easier to analyse eq. (4):

- $\frac{\mathrm{d}J(x)}{\mathrm{d}t} = -||\nabla J(x)||^2 < 0$ so $t \mapsto J(x(t))$ decreases along the trajectory $t \mapsto x(t)$;
- $\frac{\mathrm{d}J(x)}{\mathrm{d}t} = 0 \Leftrightarrow \nabla J(x) = 0$: $J(x(t))$ decreases strictly except at a critical point.

Consider the unconstrained minimization problem

$$\min_{x \in V} \qquad J(x),$$

with $J : V \to \mathbb{R}$ differentiable.

The fixed step gradient method is

$$x_{n+1} = x_n - h\nabla J(x_n). \tag{3}$$

For h sufficiently small, $J(x_{n+1}) = J(x_n) - h||\nabla J(x_n)||^2 + o(h) < J(x_n)$, J has decreased!

The convergence analysis of the discrete scheme eq. (3) is delicate, it can be done for convex functions. On the other hand, eq. (3) can be interpreted as the Euler method for the **gradient flow**

$$\frac{\mathrm{d}x}{\mathrm{d}t} = -\nabla J(x). \tag{4}$$

It is easier to analyse eq. (4):

- $\dfrac{\mathrm{d}J(x)}{\mathrm{d}t} = -||\nabla J(x)||^2 < 0$ so $t \mapsto J(x(t))$ decreases along the trajectory $t \mapsto x(t)$;
- $\dfrac{\mathrm{d}J(x)}{\mathrm{d}t} = 0 \Leftrightarrow \nabla J(x) = 0$: $J(x(t))$ decreases strictly except at a critical point.

Unconstrained optimization

Consider the unconstrained minimization problem

$$\min_{x \in V} \qquad J(x),$$

with $J : V \to \mathbb{R}$ differentiable.

The fixed step gradient method is

$$x_{n+1} = x_n - h\nabla J(x_n). \tag{3}$$

For h sufficiently small, $J(x_{n+1}) = J(x_n) - h||\nabla J(x_n)||^2 + o(h) < J(x_n)$, J has decreased!

The convergence analysis of the discrete scheme eq. (3) is delicate, it can be done for convex functions. On the other hand, eq. (3) can be interpreted as the Euler method for the **gradient flow**

$$\frac{\mathrm{d}x}{\mathrm{d}t} = -\nabla J(x). \tag{4}$$

It is easier to analyse eq. (4):

- $\frac{\mathrm{d}J(x)}{\mathrm{d}t} = -||\nabla J(x)||^2 < 0$ so $t \mapsto J(x(t))$ decreases along the trajectory $t \mapsto x(t)$;
- $\frac{\mathrm{d}J(x)}{\mathrm{d}t} = 0 \Leftrightarrow \nabla J(x) = 0$: $J(x(t))$ decreases strictly except at a critical point.

Under mild regularity assumptions, **Morse theory** says that almost all the trajectories of eq. (4) converge to a local minimizer of J.

Outline

1. Reminders on smooth constrained optimization
2. Gradient flows for unconstrained optimization
3. Constrained optimization:
   3.1 Extension to equality constrained optimization
   3.2 Extension to equality and inequality constrained optimization
4. Numerical implementation
5. Numerical examples

Extension to constrained optimization problems ?

Consider the constrained optimization problem

$$\min_{x \in V} \quad J(x)$$

$$\text{s.t.} \begin{cases} \boldsymbol{g}(x) = 0 \\ \boldsymbol{h}(x) \leq 0, \end{cases}$$

Extension to constrained optimization problems ?

Consider the constrained optimization problem

$$\min_{x\in V} \quad J(x)$$
$$\text{s.t.} \begin{cases} \boldsymbol{g}(x) = 0 \\ \boldsymbol{h}(x) \leq 0, \end{cases}$$

- Many "iteratives" methods in literature:

Extension to constrained optimization problems ?

Consider the constrained optimization problem

$$\min_{x \in V} \quad J(x)$$
$$\text{s.t.} \begin{cases} \boldsymbol{g}(x) = 0 \\ \boldsymbol{h}(x) \leq 0, \end{cases}$$

- Many "iteratives" methods in literature:
  - Penalty methods (like Augmented Lagrangian Method)

Extension to constrained optimization problems ?

Consider the constrained optimization problem

$$\min_{x \in V} \quad J(x)$$
$$\text{s.t.} \begin{cases} \boldsymbol{g}(x) = 0 \\ \boldsymbol{h}(x) \leq 0, \end{cases}$$

- Many "iteratives" methods in literature:
  - Penalty methods (like Augmented Lagrangian Method)
  - Linearization methods : SLP, SQP, MMA, MFD

Constrained optimization problems

Consider the optimization problem

$$\min_{x \in V} \quad J(x)$$
$$\text{s.t.} \begin{cases} \boldsymbol{g}(x) = 0 \\ \boldsymbol{h}(x) \leq 0, \end{cases} \tag{5}$$

Consider the optimization problem

$$\begin{aligned} \min_{x \in V} \quad & J(x) \\ \text{s.t.} & \begin{cases} \boldsymbol{g}(x) = 0 \\ \boldsymbol{h}(x) \leq 0, \end{cases} \end{aligned} \tag{5}$$

Penalty methods (like Augmented Lagrangian Method): replace eq. (6) with

$$\min_{x_n \in V} J(x) + \Lambda_n^T C(x) + \frac{\alpha_n}{2} ||C(x)||^2$$

for a sequence of penalty parameters $(\Lambda_n)_{n\in\mathbb{N}}, (\alpha_n)_{n\in\mathbb{N}}$.

Constrained optimization problems

Consider the optimization problem

$$\min_{x \in V} \quad J(x)$$
$$\text{s.t.} \begin{cases} \boldsymbol{g}(x) = 0 \\ \boldsymbol{h}(x) \leq 0, \end{cases} \tag{5}$$

Linearization methods (SLP, SQP, MMA, MFD): replace eq. (6) with the sequence of linear subproblems

$$\min_{x_{n+1} \in V} \quad J(x_{n+1})$$
$$s.t. \begin{cases} \boldsymbol{g}(x_n) + \mathrm{D}\boldsymbol{g}(x_n) \cdot (x_{n+1} - x_n) = 0 \\ \boldsymbol{h}(x_n) + \mathrm{D}\boldsymbol{h}(x_n) \cdot (x_{n+1} - x_n) \leq 0 \\ ||x_{n+1} - x_n||_\infty \leq h, \end{cases}$$

for h a small “time-step”.

Constrained optimization problems

These methods suffer from:

- the need for tuning unintuitive parameters.

Constrained optimization problems

These methods suffer from:

- the need for tuning unintuitive parameters.
- The augmented Lagrangien method **worsens** the solution if x_n is optimal but the multiplier Λ_n is not "correct".

Constrained optimization problems

These methods suffer from:

- the need for tuning unintuitive parameters.
- The augmented Lagrangien method **worsens** the solution if x_n is optimal but the multiplier Λ_n is not "correct".

These methods suffer from:

- the need for tuning unintuitive parameters.
- The augmented Lagrangien method **worsens** the solution if x_n is optimal but the multiplier Λ_n is not "correct". The objective objective function may not decrease **even if constraints are satisfied**.

Constrained optimization problems

These methods suffer from:

- the need for tuning unintuitive parameters.
- The augmented Lagrangien method **worsens** the solution if x_n is optimal but the multiplier Λ_n is not "correct". The objective objective function may not decrease **even if constraints are satisfied**.
- "inconsistencies" when $h \to 0$: SLP, SQP, MFD subproblems may not have a solution if h too small;

Constrained optimization problems

These methods suffer from:

- the need for tuning unintuitive parameters.
- The augmented Lagrangien method **worsens** the solution if x_n is optimal but the multiplier Λ_n is not "correct". The objective objective function may not decrease **even if constraints are satisfied**.
- "inconsistencies" when $h \to 0$: SLP, SQP, MFD subproblems may not have a solution if h too small;
- these schemes cannot be interpreted as a discretization of some ODE.

Constrained optimization problems

These methods suffer from:

- the need for tuning unintuitive parameters.
- The augmented Lagrangien method **worsens** the solution if x_n is optimal but the multiplier Λ_n is not "correct". The objective objective function may not decrease **even if constraints are satisfied**.
- "inconsistencies" when $h \to 0$: SLP, SQP, MFD subproblems may not have a solution if h too small;
- these schemes cannot be interpreted as a discretization of some ODE.

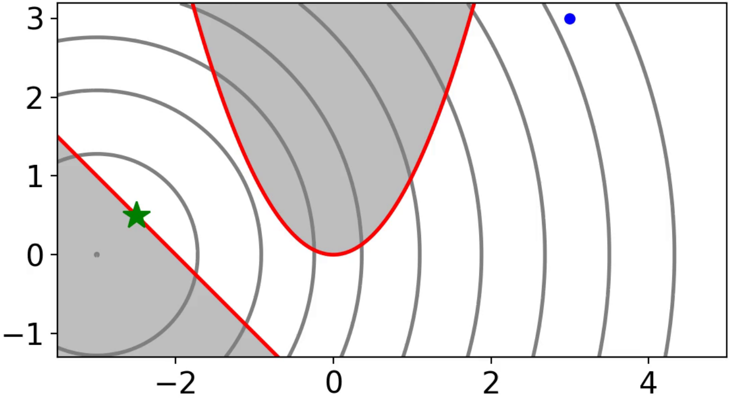
These methods suffer from:

- the need for tuning unintuitive parameters.
- The augmented Lagrangien method **worsens** the solution if x_n is optimal but the multiplier Λ_n is not "correct". The objective objective function may not decrease **even if constraints are satisfied**.
- "inconsistencies" when $h \to 0$: SLP, SQP, MFD subproblems may not have a solution if h too small;
- these schemes cannot be interpreted as a discretization of some ODE.

In what follows, we consider an extension of the gradient flow $\dot{x} = -\nabla J(x)$ for constrained optimization.

$$\min_{(x_1,x_2)\in\mathbb{R}^2} \quad J(x_1,x_2) = x_1^2 + (x_2+3)^2$$

$$\text{s.t.} \quad \begin{cases} h_1(x_1,x_2) = -x_1^2 + x_2 & \leq 0 \\ h_2(x_1,x_2) = -x_1 - x_2 - 2 & \leq 0 \end{cases}$$



Outline

Equality constrained optimization

Consider the optimization problem

$$\begin{aligned} \min_{x \in V} \quad & J(x) \\ \text{s.t. } & \boldsymbol{g}(x) = 0 \end{aligned} \tag{6}$$

Equality constrained optimization

Consider the optimization problem

$$\min_{x \in V} \quad J(x) \\ \text{s.t. } \boldsymbol{g}(x) = 0 \tag{6}$$

Assume that $\operatorname{rank}(\mathrm{D}\boldsymbol{g}(x)\mathrm{D}\boldsymbol{g}(x)^{\mathcal{T}}) = p$.

Equality constrained optimization

Consider the optimization problem

$$\min_{x \in V} \quad J(x) \\ \text{s.t. } \boldsymbol{g}(x) = 0 \tag{6}$$

Assume that $\text{rank}(\mathrm{D}\boldsymbol{g}(x)\mathrm{D}\boldsymbol{g}(x)^{\mathcal{T}}) = p$.

Definition 3

The *null space* and *range space* directions $\boldsymbol{\xi}_J(x)$ and $\boldsymbol{\xi}_C(x)$ are defined by:

$$\boldsymbol{\xi}_J(x) := (I - \mathrm{D}\boldsymbol{g}^{\mathcal{T}}(\mathrm{D}\boldsymbol{g}\mathrm{D}\boldsymbol{g}^{\mathcal{T}})^{-1}\mathrm{D}\boldsymbol{g})\nabla J(x),$$
$$\boldsymbol{\xi}_C(x) := \mathrm{D}\boldsymbol{g}^{\mathcal{T}}(\mathrm{D}\boldsymbol{g}\mathrm{D}\boldsymbol{g}^{\mathcal{T}})^{-1}\boldsymbol{g}(x).$$

Equality constrained optimization

The following properties hold for the null space direction $\boldsymbol{\xi}_J(x)$:

Lemma 4

1. $V = \mathrm{Ker}(\mathrm{D}\boldsymbol{g}(x)) \oplus \mathsf{Ran}(\mathrm{D}\boldsymbol{g}(x)^{\mathcal{T}})$, *where* $\mathsf{Ran}(\mathrm{D}\boldsymbol{g}(x)^{\mathcal{T}}) := \{\mathrm{D}\boldsymbol{g}(x)^{\mathcal{T}}\boldsymbol{\lambda} \,|\, \boldsymbol{\lambda} \in \mathbb{R}^p\}$ *of* $\mathrm{D}\boldsymbol{g}(x)^{\mathcal{T}}$.

Equality constrained optimization

The following properties hold for the null space direction $\boldsymbol{\xi}_J(x)$:

Lemma 4

1. $V = \mathrm{Ker}(\mathrm{D}\boldsymbol{g}(x)) \oplus \mathsf{Ran}(\mathrm{D}\boldsymbol{g}(x)^{\mathcal{T}})$, *where* $\mathsf{Ran}(\mathrm{D}\boldsymbol{g}(x)^{\mathcal{T}}) := \{\mathrm{D}\boldsymbol{g}(x)^{\mathcal{T}}\boldsymbol{\lambda} \,|\, \boldsymbol{\lambda} \in \mathbb{R}^p\}$ *of* $\mathrm{D}\boldsymbol{g}(x)^{\mathcal{T}}$.
2. *The operator* $\Pi_{g(x)} : V \to V$ *defined by*

$$\Pi_{g(x)} = I - \mathrm{D}\boldsymbol{g}^{\mathcal{T}}(\mathrm{D}\boldsymbol{g}\mathrm{D}\boldsymbol{g}^{\mathcal{T}})^{-1}\mathrm{D}\boldsymbol{g}(x)$$

   is the orthogonal projection onto $\mathrm{Ker}(\mathrm{D}\boldsymbol{g}(x))$ *with* $\mathrm{Ker}(\Pi_{g(x)}) = \mathsf{Ran}(\mathrm{D}\boldsymbol{g}(x)^{\mathcal{T}})$.

Equality constrained optimization

The following properties hold for the null space direction $\boldsymbol{\xi}_J(x)$:

Lemma 4

1. $V = \mathrm{Ker}(\mathrm{D}\boldsymbol{g}(x)) \oplus \mathrm{Ran}(\mathrm{D}\boldsymbol{g}(x)^{\mathcal{T}})$, *where* $\mathrm{Ran}(\mathrm{D}\boldsymbol{g}(x)^{\mathcal{T}}) := \{\mathrm{D}\boldsymbol{g}(x)^{\mathcal{T}}\boldsymbol{\lambda} \,|\, \boldsymbol{\lambda} \in \mathbb{R}^p\}$ *of* $\mathrm{D}\boldsymbol{g}(x)^{\mathcal{T}}$.
2. *The operator* $\Pi_{g(x)} : V \to V$ *defined by*

$$\Pi_{g(x)} = I - \mathrm{D}\boldsymbol{g}^{\mathcal{T}}(\mathrm{D}\boldsymbol{g}\mathrm{D}\boldsymbol{g}^{\mathcal{T}})^{-1}\mathrm{D}\boldsymbol{g}(x)$$

   is the orthogonal projection onto $\mathrm{Ker}(\mathrm{D}\boldsymbol{g}(x))$ *with* $\mathrm{Ker}(\Pi_{g(x)}) = \mathrm{Ran}(\mathrm{D}\boldsymbol{g}(x)^{\mathcal{T}})$.
3. *When* $\Pi_{g(x)}(\nabla J(x)) \neq 0$, $-\boldsymbol{\xi}_J(x) = -\Pi_{g(x)}(\nabla J(x))$ *is the* **best feasible descent direction** *for* J *in the sense that*

$$-\frac{\boldsymbol{\xi}_J(x)}{||\boldsymbol{\xi}_J(x)||_V} = \arg\min_{\boldsymbol{\xi}\in V} \quad \mathrm{D}J(x)\boldsymbol{\xi} \quad \text{s.t.} \begin{cases} \mathrm{D}\boldsymbol{g}(x)\boldsymbol{\xi} = 0 \\ \langle \boldsymbol{\xi}, \boldsymbol{\xi} \rangle_V \leq 1. \end{cases} \tag{7}$$

Equality constrained optimization

Lemma 5

The null space direction $\boldsymbol{\xi}_J(x) = \Pi_{g(x)}(\nabla J(x))$ *is the closest least squares approximation to* $\nabla J(x)$ *within the space* $\mathrm{Ker}(\mathrm{D}\boldsymbol{g}(x))$*:*

$$\boldsymbol{\xi}_J(x) = \arg \min_{\boldsymbol{\xi} \in \mathrm{Ker}(\mathrm{D}\boldsymbol{g}(x))} ||\nabla J(x) - \boldsymbol{\xi}||_V.$$

Lemma 5

The null space direction $\boldsymbol{\xi}_J(x) = \Pi_{g(x)}(\nabla J(x))$ *is the closest least squares approximation to* $\nabla J(x)$ *within the space* $\mathrm{Ker}(\mathrm{D}\boldsymbol{g}(x))$*:*

$$\boldsymbol{\xi}_J(x) = \arg \min_{\boldsymbol{\xi} \in \mathrm{Ker}(\mathrm{D}\boldsymbol{g}(x))} ||\nabla J(x) - \boldsymbol{\xi}||_V.$$

It alternatively reads

$$\boldsymbol{\xi}_J(x) = \nabla J(x) + \mathrm{D}\boldsymbol{g}(x)^{\mathcal{T}} \boldsymbol{\lambda}^*(x),$$

where the Lagrange multiplier $\boldsymbol{\lambda}^*(x) := -(\mathrm{D}\boldsymbol{g}\mathrm{D}\boldsymbol{g}^{\mathcal{T}})^{-1}\mathrm{D}\boldsymbol{g}\nabla J(x)$ *is the unique solution to the following least squares problem that is the dual of eq.* (7)*:*

$$\boldsymbol{\lambda}^*(x) = \arg \min_{\boldsymbol{\lambda} \in \mathbb{R}^p} ||\nabla J(x) + \mathrm{D}\boldsymbol{g}(x)^{\mathcal{T}} \boldsymbol{\lambda}||_V.$$

Lemma 5

The null space direction $\boldsymbol{\xi}_J(x) = \Pi_{g(x)}(\nabla J(x))$ *is the closest least squares approximation to* $\nabla J(x)$ *within the space* $\mathrm{Ker}(\mathrm{D}\boldsymbol{g}(x))$*:*

$$\boldsymbol{\xi}_J(x) = \arg \min_{\boldsymbol{\xi} \in \mathrm{Ker}(\mathrm{D}\boldsymbol{g}(x))} ||\nabla J(x) - \boldsymbol{\xi}||_V.$$

It alternatively reads

$$\boldsymbol{\xi}_J(x) = \nabla J(x) + \mathrm{D}\boldsymbol{g}(x)^{\mathcal{T}} \boldsymbol{\lambda}^*(x),$$

where the Lagrange multiplier $\boldsymbol{\lambda}^*(x) := -(\mathrm{D}\boldsymbol{g}\mathrm{D}\boldsymbol{g}^{\mathcal{T}})^{-1}\mathrm{D}\boldsymbol{g}\nabla J(x)$ *is the unique solution to the following least squares problem that is the dual of eq.* (7)*:*

$$\boldsymbol{\lambda}^*(x) = \arg \min_{\boldsymbol{\lambda} \in \mathbb{R}^p} ||\nabla J(x) + \mathrm{D}\boldsymbol{g}(x)^{\mathcal{T}} \boldsymbol{\lambda}||_V.$$

Remark 1

- $\boldsymbol{\lambda}^*(x)$ is defined for any x such that $\mathrm{D}\boldsymbol{g}\mathrm{D}\boldsymbol{g}^{\mathcal{T}}(x)$ is invertible;

Lemma 5

The null space direction $\boldsymbol{\xi}_J(x) = \Pi_{g(x)}(\nabla J(x))$ *is the closest least squares approximation to* $\nabla J(x)$ *within the space* $\mathrm{Ker}(\mathrm{D}\boldsymbol{g}(x))$*:*

$$\boldsymbol{\xi}_J(x) = \arg \min_{\boldsymbol{\xi} \in \mathrm{Ker}(\mathrm{D}\boldsymbol{g}(x))} ||\nabla J(x) - \boldsymbol{\xi}||_V.$$

It alternatively reads

$$\boldsymbol{\xi}_J(x) = \nabla J(x) + \mathrm{D}\boldsymbol{g}(x)^{\mathcal{T}} \boldsymbol{\lambda}^*(x),$$

where the Lagrange multiplier $\boldsymbol{\lambda}^*(x) := -(\mathrm{D}\boldsymbol{g}\mathrm{D}\boldsymbol{g}^{\mathcal{T}})^{-1}\mathrm{D}\boldsymbol{g}\nabla J(x)$ *is the unique solution to the following least squares problem that is the dual of eq.* (7)*:*

$$\boldsymbol{\lambda}^*(x) = \arg \min_{\boldsymbol{\lambda} \in \mathbb{R}^p} ||\nabla J(x) + \mathrm{D}\boldsymbol{g}(x)^{\mathcal{T}} \boldsymbol{\lambda}||_V.$$

Remark 1

- $\boldsymbol{\lambda}^*(x)$ is defined for any x such that $\mathrm{D}\boldsymbol{g}\mathrm{D}\boldsymbol{g}^{\mathcal{T}}(x)$ is invertible;
- $\boldsymbol{\xi}_J(x) = 0$ if and only if x satisfies the KKT condition;

Equality constrained optimization

Lemma 5

The null space direction $\boldsymbol{\xi}_J(x) = \Pi_{g(x)}(\nabla J(x))$ *is the closest least squares approximation to* $\nabla J(x)$ *within the space* $\mathrm{Ker}(\mathrm{D}\boldsymbol{g}(x))$*:*

$$\boldsymbol{\xi}_J(x) = \arg \min_{\boldsymbol{\xi} \in \mathrm{Ker}(\mathrm{D}\boldsymbol{g}(x))} ||\nabla J(x) - \boldsymbol{\xi}||_V.$$

It alternatively reads

$$\boldsymbol{\xi}_J(x) = \nabla J(x) + \mathrm{D}\boldsymbol{g}(x)^{\mathcal{T}} \boldsymbol{\lambda}^*(x),$$

where the Lagrange multiplier $\boldsymbol{\lambda}^*(x) := -(\mathrm{D}\boldsymbol{g}\mathrm{D}\boldsymbol{g}^{\mathcal{T}})^{-1}\mathrm{D}\boldsymbol{g}\nabla J(x)$ *is the unique solution to the following least squares problem that is the dual of eq.* (7)*:*

$$\boldsymbol{\lambda}^*(x) = \arg \min_{\boldsymbol{\lambda} \in \mathbb{R}^p} ||\nabla J(x) + \mathrm{D}\boldsymbol{g}(x)^{\mathcal{T}} \boldsymbol{\lambda}||_V.$$

Remark 1

- $\boldsymbol{\lambda}^*(x)$ is defined for any x such that $\mathrm{D}\boldsymbol{g}\mathrm{D}\boldsymbol{g}^{\mathcal{T}}(x)$ is invertible;
- $\boldsymbol{\xi}_J(x) = 0$ if and only if x satisfies the KKT condition;
- In that case, $\boldsymbol{\lambda}^*(x)$ is **the** Lagrange multiplier of the KKT condition $\nabla J(x) + \mathrm{D}g(x^*)^{\mathcal{T}} \boldsymbol{\lambda}^* = 0$.

The range space step:

Lemma 6

1. *The range space step* $\boldsymbol{\xi}_C(x) := \mathrm{D}\boldsymbol{g}^{\mathcal{T}}(\mathrm{D}\boldsymbol{g}\mathrm{D}\boldsymbol{g}^{\mathcal{T}})^{-1}\boldsymbol{g}(x)$ *is orthogonal to* $\mathrm{Ker}(\mathrm{D}\boldsymbol{g}(x))$*:*

$$\forall \boldsymbol{\xi} \in \mathrm{Ker}(\mathrm{D}\boldsymbol{g}(x)),\ \ \langle \boldsymbol{\xi}_C(x), \boldsymbol{\xi} \rangle_V = 0.$$

The range space step:

Lemma 6

1. *The range space step* $\boldsymbol{\xi}_C(x) := \mathrm{D}\boldsymbol{g}^{\mathcal{T}}(\mathrm{D}\boldsymbol{g}\mathrm{D}\boldsymbol{g}^{\mathcal{T}})^{-1}\boldsymbol{g}(x)$ *is orthogonal to* $\mathrm{Ker}(\mathrm{D}\boldsymbol{g}(x))$*:*
$$\forall \boldsymbol{\xi} \in \mathrm{Ker}(\mathrm{D}\boldsymbol{g}(x)),\ \ \langle \boldsymbol{\xi}_C(x), \boldsymbol{\xi} \rangle_V = 0.$$
2. $-\boldsymbol{\xi}_C(x)$ *is a descent direction for the violation of the constraints:*
$$\mathrm{D}\boldsymbol{g}(x)(-\boldsymbol{\xi}_C(x)) = -\boldsymbol{g}(x).$$

Equality constrained optimization

The range space step:

Lemma 6

1. *The range space step* $\boldsymbol{\xi}_C(x) := \mathrm{D}\boldsymbol{g}^{\mathcal{T}}(\mathrm{D}\boldsymbol{g}\mathrm{D}\boldsymbol{g}^{\mathcal{T}})^{-1}\boldsymbol{g}(x)$ *is orthogonal to* $\mathrm{Ker}(\mathrm{D}\boldsymbol{g}(x))$*:*

$$\forall \boldsymbol{\xi} \in \mathrm{Ker}(\mathrm{D}\boldsymbol{g}(x)), \;\; \langle \boldsymbol{\xi}_C(x), \boldsymbol{\xi} \rangle_V = 0.$$

2. $-\boldsymbol{\xi}_C(x)$ *is a descent direction for the violation of the constraints:*

$$\mathrm{D}\boldsymbol{g}(x)(-\boldsymbol{\xi}_C(x)) = -\boldsymbol{g}(x).$$

3. *The set of solutions to the Gauss-Newton program*

$$\min_{\boldsymbol{\xi} \in V} ||\boldsymbol{g}(x) + \mathrm{D}\boldsymbol{g}(x)\boldsymbol{\xi}||^2$$

is the affine subspace $\{-\boldsymbol{\xi}_C(x) + \boldsymbol{\zeta} \mid \boldsymbol{\zeta} \in \mathrm{Ker}(\mathrm{D}\boldsymbol{g}(x))\}$ *of* V*.*

Equality constrained optimization

The range space step:

Lemma 6

1. *The range space step* $\boldsymbol{\xi}_C(x) := \mathrm{D}\boldsymbol{g}^{\mathcal{T}}(\mathrm{D}\boldsymbol{g}\mathrm{D}\boldsymbol{g}^{\mathcal{T}})^{-1}\boldsymbol{g}(x)$ *is orthogonal to* $\mathrm{Ker}(\mathrm{D}\boldsymbol{g}(x))$*:*

$$\forall \boldsymbol{\xi} \in \mathrm{Ker}(\mathrm{D}\boldsymbol{g}(x)), \ \ \langle \boldsymbol{\xi}_C(x), \boldsymbol{\xi} \rangle_V = 0.$$

2. $-\boldsymbol{\xi}_C(x)$ *is a descent direction for the violation of the constraints:*

$$\mathrm{D}\boldsymbol{g}(x)(-\boldsymbol{\xi}_C(x)) = -\boldsymbol{g}(x).$$

3. *The set of solutions to the Gauss-Newton program*

$$\min_{\boldsymbol{\xi} \in V} ||\boldsymbol{g}(x) + \mathrm{D}\boldsymbol{g}(x)\boldsymbol{\xi}||^2$$

is the affine subspace $\{-\boldsymbol{\xi}_C(x) + \boldsymbol{\zeta} \mid \boldsymbol{\zeta} \in \mathrm{Ker}(\mathrm{D}\boldsymbol{g}(x))\}$ *of* V*.*

Equality constrained optimization

The range space step:

Lemma 6

1. *The range space step* $\boldsymbol{\xi}_C(x) := \mathrm{D}\boldsymbol{g}^{\mathcal{T}}(\mathrm{D}\boldsymbol{g}\mathrm{D}\boldsymbol{g}^{\mathcal{T}})^{-1}\boldsymbol{g}(x)$ *is orthogonal to* $\mathrm{Ker}(\mathrm{D}\boldsymbol{g}(x))$*:*

$$\forall \boldsymbol{\xi} \in \mathrm{Ker}(\mathrm{D}\boldsymbol{g}(x)),\ \ \langle \boldsymbol{\xi}_C(x), \boldsymbol{\xi}\rangle_V = 0.$$

2. $-\boldsymbol{\xi}_C(x)$ *is a descent direction for the violation of the constraints:*

$$\mathrm{D}\boldsymbol{g}(x)(-\boldsymbol{\xi}_C(x)) = -\boldsymbol{g}(x).$$

3. *The set of solutions to the Gauss-Newton program*

$$\min_{\boldsymbol{\xi}\in V} ||\boldsymbol{g}(x) + \mathrm{D}\boldsymbol{g}(x)\boldsymbol{\xi}||^2$$

is the affine subspace $\{-\boldsymbol{\xi}_C(x) + \boldsymbol{\zeta} \mid \boldsymbol{\zeta} \in \mathrm{Ker}(\mathrm{D}\boldsymbol{g}(x))\}$ *of* V*.*

Remark 2

The range space and null space steps are orthogonal: $\langle \boldsymbol{\xi}_J(x), \boldsymbol{\xi}_C(x)\rangle_V = 0$

Proposition 2

Assume that the constraints $\boldsymbol{g}$ are qualified and consider the flow

$$\begin{cases} \dot{x} = -\alpha_J(I - \mathrm{D}\boldsymbol{g}^{\mathcal{T}}(\mathrm{D}\boldsymbol{g}\mathrm{D}\boldsymbol{g}^{\mathcal{T}})^{-1}\mathrm{D}\boldsymbol{g}(x))\nabla J(x) - \alpha_C \mathrm{D}\boldsymbol{g}^{\mathcal{T}}(\mathrm{D}\boldsymbol{g}\mathrm{D}\boldsymbol{g}^{\mathcal{T}})^{-1}\boldsymbol{g}(x) \\ x(0) = x_0 \end{cases} \tag{8}$$

for some $\alpha_J, \alpha_C > 0$. Then the following properties hold true:

Proposition 2

Assume that the constraints $\boldsymbol{g}$ *are qualified and consider the flow*

$$\begin{cases} \dot{x} = -\alpha_J(I - \mathrm{D}\boldsymbol{g}^{\mathcal{T}}(\mathrm{D}\boldsymbol{g}\mathrm{D}\boldsymbol{g}^{\mathcal{T}})^{-1}\mathrm{D}\boldsymbol{g}(x))\nabla J(x) - \alpha_C \mathrm{D}\boldsymbol{g}^{\mathcal{T}}(\mathrm{D}\boldsymbol{g}\mathrm{D}\boldsymbol{g}^{\mathcal{T}})^{-1}\boldsymbol{g}(x) \\ x(0) = x_0 \end{cases} \tag{8}$$

for some $\alpha_J, \alpha_C > 0$. *Then the following properties hold true:*

1. *The violation of the constraints decreases exponentially:*

$$\forall t \in [0, T],\ \boldsymbol{g}(x(t)) = e^{-\alpha_C t}\boldsymbol{g}(x_0).$$

Equality constrained optimization

Proposition 2

Assume that the constraints $\mathbf{g}$ *are qualified and consider the flow*

$$\begin{cases} \dot{x} = -\alpha_J(I - \mathrm{D}\mathbf{g}^{\mathcal{T}}(\mathrm{D}\mathbf{g}\mathrm{D}\mathbf{g}^{\mathcal{T}})^{-1}\mathrm{D}\mathbf{g}(x))\nabla J(x) - \alpha_C \mathrm{D}\mathbf{g}^{\mathcal{T}}(\mathrm{D}\mathbf{g}\mathrm{D}\mathbf{g}^{\mathcal{T}})^{-1}\mathbf{g}(x) \\ x(0) = x_0 \end{cases} \tag{8}$$

for some $\alpha_J, \alpha_C > 0$*. Then the following properties hold true:*

1. *The violation of the constraints decreases exponentially:*

$$\forall t \in [0, T],\ \mathbf{g}(x(t)) = e^{-\alpha_C t}\mathbf{g}(x_0).$$

2. $J(x(t))$ *decreases "as soon as the violation of the constraints is sufficiently small":*

$$\forall t \in [0, T],\ ||\Pi_{g(x)}(\nabla J(x(t)))||_V^2 > Ce^{-\alpha_C t} \Rightarrow \frac{\mathrm{d}}{\mathrm{d}t}J(x(t)) < 0.$$

Equality constrained optimization

Proposition 2

Assume that the constraints $\boldsymbol{g}$ *are qualified and consider the flow*

$$\begin{cases} \dot{x} = -\alpha_J(I - \mathrm{D}\boldsymbol{g}^{\mathcal{T}}(\mathrm{D}\boldsymbol{g}\mathrm{D}\boldsymbol{g}^{\mathcal{T}})^{-1}\mathrm{D}\boldsymbol{g}(x))\nabla J(x) - \alpha_C \mathrm{D}\boldsymbol{g}^{\mathcal{T}}(\mathrm{D}\boldsymbol{g}\mathrm{D}\boldsymbol{g}^{\mathcal{T}})^{-1}\boldsymbol{g}(x) \\ x(0) = x_0 \end{cases} \tag{8}$$

for some $\alpha_J, \alpha_C > 0$. *Then the following properties hold true:*

1. *The violation of the constraints decreases exponentially:*

$$\forall t \in [0, T],\ \boldsymbol{g}(x(t)) = e^{-\alpha_C t}\boldsymbol{g}(x_0).$$

2. $J(x(t))$ *decreases "as soon as the violation of the constraints is sufficiently small":*

$$\forall t \in [0, T],\ ||\Pi_{g(x)}(\nabla J(x(t)))||_V^2 > Ce^{-\alpha_C t} \Rightarrow \frac{\mathrm{d}}{\mathrm{d}t}J(x(t)) < 0.$$

3. *Any stationary point* x^* *of eq.* (8) *satisfies the first-order KKT conditions, that is:*

$$\begin{cases} \boldsymbol{g}(x^*) = 0 \\ \exists \boldsymbol{\lambda}^* \in \mathbb{R}^p,\ \nabla J(x^*) + \mathrm{D}\boldsymbol{g}^{\mathcal{T}}(x^*)\boldsymbol{\lambda}^* = \Pi_{g(x^*)}(\nabla J(x^*)) = 0. \end{cases}$$

Equality constrained optimization

- $\alpha_J > 0$ and $\alpha_C > 0$ controls the trade off between decreasing $J(x)$ and $||\boldsymbol{g}(x)||$.

Equality constrained optimization

- $\alpha_J > 0$ and $\alpha_C > 0$ controls the trade off between decreasing $J(x)$ and $||\boldsymbol{g}(x)||$.
- Consider the Euler scheme:

$$x_{n+1} = x_n - \Delta t(\alpha_J \boldsymbol{\xi}_J(x_n) + \alpha_C \boldsymbol{\xi}_C(x_n)),$$

Equality constrained optimization

- $\alpha_J > 0$ and $\alpha_C > 0$ controls the trade off between decreasing $J(x)$ and $||\boldsymbol{g}(x)||$.
- Consider the Euler scheme:

$$x_{n+1} = x_n - \Delta t(\alpha_J \boldsymbol{\xi}_J(x_n) + \alpha_C \boldsymbol{\xi}_C(x_n)),$$

1. At first order, the constraints decrease with a geometric rate: $\boldsymbol{g}(x_{n+1}) = (1 - \alpha_C \Delta t)\boldsymbol{g}(x_n) + o(\Delta t)$.

Equality constrained optimization

- $\alpha_J > 0$ and $\alpha_C > 0$ controls the trade off between decreasing $J(x)$ and $||\boldsymbol{g}(x)||$.
- Consider the Euler scheme:

$$x_{n+1} = x_n - \Delta t(\alpha_J \boldsymbol{\xi}_J(x_n) + \alpha_C \boldsymbol{\xi}_C(x_n)),$$

1. At first order, the constraints decrease with a geometric rate: $\boldsymbol{g}(x_{n+1}) = (1 - \alpha_C \Delta t)\boldsymbol{g}(x_n) + o(\Delta t)$.
2. An accumulation point x^* of the sequence $(x_n)_{n\in\mathbb{N}}$ satisfies $\boldsymbol{g}(x^*) = 0$ and the KKT conditions.

Equality constrained optimization

- $\alpha_J > 0$ and $\alpha_C > 0$ controls the trade off between decreasing $J(x)$ and $||\boldsymbol{g}(x)||$.
- Consider the Euler scheme:

$$x_{n+1} = x_n - \Delta t(\alpha_J \boldsymbol{\xi}_J(x_n) + \alpha_C \boldsymbol{\xi}_C(x_n)),$$

  1. At first order, the constraints decrease with a geometric rate: $\boldsymbol{g}(x_{n+1}) = (1 - \alpha_C \Delta t)\boldsymbol{g}(x_n) + o(\Delta t)$.
  2. An accumulation point x^* of the sequence $(x_n)_{n\in\mathbb{N}}$ satisfies $\boldsymbol{g}(x^*) = 0$ and the KKT conditions.
- The range space step $\boldsymbol{\xi}_C(x_n)$ corrects numerical errors on the violation of the constraint ($\boldsymbol{\xi}_J(x_n)$ preserves the constraint only at first order).

Exercise: solve an equality constrained optimization problem

- Install the nullspace optimizer python package:

  `https://people.math.ethz.ch/~ffeppon/topopt_course/install_software.html`

Exercise: solve an equality constrained optimization problem

- Install the nullspace optimizer python package:

  `https://people.math.ethz.ch/~ffeppon/topopt_course/install_software.html`

- Write an optimization program to solve the constrained minimization problem on the hyperbola:

$$\min_{(x_1,x_2)\in\mathbb{R}^2} \quad x_1 + x_2$$

$$s.t.\ x_1 x_2 = 1.$$

  Use $(0.1, 0.1), (4, 0.25), (4, 1)$ as initialisations.

Exercise: solve an equality constrained optimization problem

- Install the nullspace optimizer python package:

  `https://people.math.ethz.ch/~ffeppon/topopt_course/install_software.html`

- Write an optimization program to solve the constrained minimization problem on the hyperbola:

$$\min_{(x_1,x_2)\in\mathbb{R}^2} \quad x_1 + x_2$$

$$s.t.\ x_1 x_2 = 1.$$

  Use $(0.1, 0.1), (4, 0.25), (4, 1)$ as initialisations.

- Do the same to solve

$$\max_{(x_1,x_2)\in\mathbb{R}^2} \quad x_2$$

$$s.t. \begin{cases} (x_1 - 0.5)^2 + x_2^2 = 2 \\ (x_1 + 0.5)^2 + x_2^2 = 2. \end{cases}$$