Nonlinear Programming

- Basic Notions of Nonlinear Programming
- Necessary Conditions of Optimality
- Interpretation of Lagrange Multipliers
- Minimal Primer on Algorithms for NLPs
- Computation of Derivatives

Contents

Nonlinear Program (NLP)

Problem formulation:

$$\min_{x \in \mathbb{R}^{n_x}} \quad f(x) \quad \text{subject to} \quad \begin{cases} h(x) = 0 \\ g(x) \leq 0 \end{cases}$$

- Objective $f : \mathbb{R}^{n_x} \to \mathbb{R}$;
- Equality constraints $h(x) : \mathbb{R}^{n_x} \to \mathbb{R}^{n_h}$, $h(x) = [h_1(x), ..., h_{n_h}(x)]^\top$;
- Inequality constraints $g(x) : \mathbb{R}^{n_x} \to \mathbb{R}^{n_g}$, $g(x) = [g_1(x), ..., g_{n_g}(x)]^\top$.

Nonlinear Program (NLP)

Problem formulation:

$$\min_{x \in \mathbb{R}^{n_x}} \quad f(x) \quad \text{subject to} \quad \begin{cases} h(x) = 0 \\ g(x) \leq 0 \end{cases}$$

- Objective $f : \mathbb{R}^{n_x} \to \mathbb{R}$;
- Equality constraints $h(x) : \mathbb{R}^{n_x} \to \mathbb{R}^{n_h}$, $h(x) = [h_1(x), ..., h_{n_h}(x)]^\top$;
- Inequality constraints $g(x) : \mathbb{R}^{n_x} \to \mathbb{R}^{n_g}$, $g(x) = [g_1(x), ..., g_{n_g}(x)]^\top$.

Why discuss NLPs in this course?

- Nonlinear Programming = optimization in Euclidian space
- Optimal Control (OC) = optimization in a function space
- NLP techniques are used to solve Optimal Control Problems (OCP)
  - Discrete-time optimal control $\equiv$ NLP

$$\min_{\{x_k\},\{u_k\}} \sum_{k=0}^{N-1} \ell(x_k, u_k) \quad \text{subject to} \begin{cases} \forall\, k \in \{0, \dots, N-1\} \\ x_{k+1} - f(x_k, u_k) = 0 \\ x_0 - \bar{x} = 0 \\ g(x_k, u_k) \le 0 \end{cases}$$

  - Continuous-time dynamics $\rightarrow$ approximate solution obtained via NLPs

Why discuss NLPs in this course?

- Nonlinear Programming = optimization in Euclidian space
- Optimal Control (OC) = optimization in a function space
- NLP techniques are used to solve Optimal Control Problems (OCP)
  - Discrete-time optimal control $\equiv$ NLP

$$\min_{\{x_k\},\{u_k\}} \quad \sum_{k=0}^{N-1} \ell(x_k, u_k) \quad \text{subject to} \begin{cases} \forall\, k \in \{0, \dots, N-1\} \\ x_{k+1} - f(x_k, u_k) = 0 \\ x_0 - \bar{x} = 0 \\ g(x_k, u_k) \leq 0 \end{cases}$$

  - Continuous-time dynamics $\rightarrow$ approximate solution obtained via NLPs

Example – Nonlinear Program

$$\min_{x \in \mathbb{R}^2} (x_1 - 3)^2 + (x_2 - 2)^2$$

subject to

$$g_1(x) = x_1^2 - x_2 - 3 \leq 0$$

$$g_2(x) = x_2 - 1 \leq 0$$

$$g_3(x) = -x_1 \leq 0$$

Example – Nonlinear Program

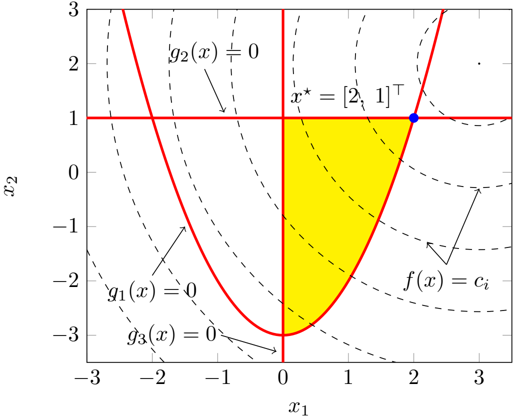
$$\min_{x \in \mathbb{R}^2} (x_1 - 3)^2 + (x_2 - 2)^2$$

subject to

$$g_1(x) = x_1^2 - x_2 - 3 \leq 0$$

$$g_2(x) = x_2 - 1 \leq 0$$

$$g_3(x) = -x_1 \leq 0$$



Feasibility

Definition (Feasible Set)

$$\mathbb{S} := \{x \in \mathbb{R}^{n_x} \,|h(x) = 0 \text{ and } g(x) \leq 0\}$$

- Consider NLP

$$\min_{x \in \mathbb{S}} \quad f(x)$$

with feasible set $\mathbb{S} \subseteq \mathbb{R}^{n_x}$.

$$\mathbb{S} \neq \emptyset \Longleftrightarrow \text{NLP is feasible.}$$

Feasibility

Definition (Feasible Set)

$$\mathbb{S} := \{x \in \mathbb{R}^{n_x} \,|h(x) = 0 \text{ and } g(x) \leq 0\}$$

- Consider NLP

$$\min_{x \in \mathbb{S}} \quad f(x)$$

with feasible set $\mathbb{S} \subseteq \mathbb{R}^{n_x}$.

$$\mathbb{S} \neq \emptyset \iff \text{NLP is feasible.}$$

Definition of Optimality – Infimum

Definition (Infimum)

The infimum of a partially ordered set $\mathbb{S}$, denoted as $\inf \mathbb{S}$, provided it exists, is the greatest lower bound for $z \in \mathbb{S}$, i.e., a real number α satisfying

1. $z \geq \alpha, \forall z \in \mathbb{S}$;
2. $\forall \bar{\alpha} > \alpha, \exists z \in \mathbb{S}$ such that $z < \bar{\alpha}$.

Definition of Optimality – Minimum

Definition

A point $x^* \in \mathbb{S}$ is said to be a (global) *minimizer* of f on $\mathbb{S} \subseteq \mathbb{R}^{n_x}$ if

$$f(x) \geq f(x^*), \quad \forall x \in \mathbb{S},$$

and $f(x^*)$ is called (global) minimum of f on $\mathbb{S}$.

Definition of Optimality – Minimum

Definition

A point $x^* \in \mathbb{S}$ is said to be a (global) *minimizer* of f on $\mathbb{S} \subseteq \mathbb{R}^{n_x}$ if

$$f(x) \geq f(x^*), \quad \forall x \in \mathbb{S},$$

and $f(x^*)$ is called (global) minimum of f on $\mathbb{S}$.

It is said to be a strict (global) minimizer of f on $\mathbb{S} \subseteq \mathbb{R}^{n_x}$ if

$$f(x) > f(x^*), \quad \forall x \in \mathbb{S},\ x \neq x^*,$$

and $f(x^*)$ is called strict (global) minimum of f on $\mathbb{S}$.

Definition of Optimality – Local Minimum

ϵ-ball around $\bar{x}$ (or ϵ-neighborhood):

$$\mathbb{B}_\epsilon(\bar{x}) := \{x \in \mathbb{R}^{n_x} \,|\, \|x - \bar{x}\| \leq \epsilon\} \subset \mathbb{R}^{n_x}$$

Definition (Local minimum)

A point $x^* \in \mathbb{S}$ is said to be local minimizer of f, if

$$\exists\, \epsilon > 0,\ \forall\, x \in \mathbb{B}_\epsilon(x^*) \cap \mathbb{S},\quad f(x) \geq f(x^*).$$

It is said to be a strict local minimizer of f on $\mathbb{S}$ if

$$\exists\, \epsilon > 0,\ \forall\, x \in \mathbb{B}_\epsilon(x^*) \cap \mathbb{S},\quad f(x) > f(x^*).$$

Definition of Optimality – Local Minimum

ϵ-ball around $\bar{x}$ (or ϵ-neighborhood):

$$\mathbb{B}_\epsilon(\bar{x}) := \{x \in \mathbb{R}^{n_x} \,|\, \|x - \bar{x}\| \leq \epsilon\} \subset \mathbb{R}^{n_x}$$

Definition (Local minimum)

A point $x^* \in \mathbb{S}$ is said to be local minimizer of f, if

$$\exists\, \epsilon > 0,\ \forall\, x \in \mathbb{B}_\epsilon(x^*) \cap \mathbb{S},\ \ f(x) \geq f(x^*).$$

It is said to be a strict local minimizer of f on $\mathbb{S}$ if

$$\exists\, \epsilon > 0,\ \forall\, x \in \mathbb{B}_\epsilon(x^*) \cap \mathbb{S},\ \ f(x) > f(x^*).$$

Existence of Minima and Maxima

Theorem (Extreme value theorem)

Let $\mathbb{S}$ *be nonempty and compact and let* f *be continuous on* $\mathbb{S}$*.*

Then, the following problem has a minimizer in $\mathbb{S}$*,*

$$\min_{x \in \mathbb{S}} \quad f(x).$$

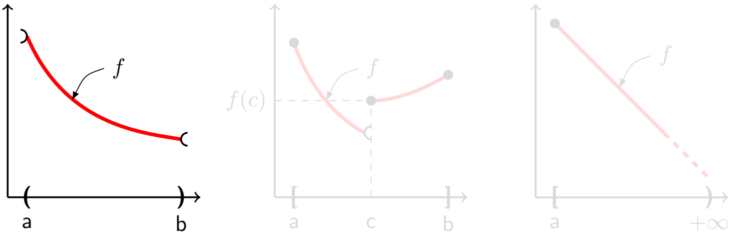
Existence of Minima and Maxima

Theorem (Extreme value theorem)

Let $\mathbb{S}$ be nonempty and compact and let f be continuous on $\mathbb{S}$.

Then, the following problem has a minimizer in $\mathbb{S}$,

$$\min_{x \in \mathbb{S}} \quad f(x).$$



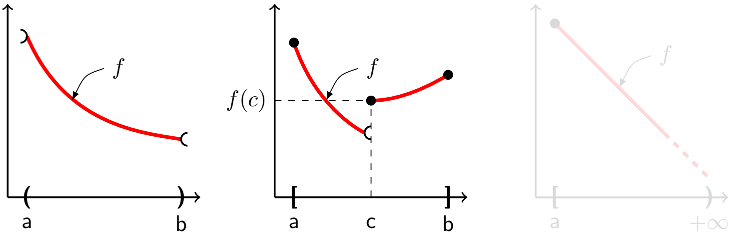
Existence of Minima and Maxima

Theorem (Extreme value theorem)

Let $\mathbb{S}$ be nonempty and compact and let f be continuous on $\mathbb{S}$.

Then, the following problem has a minimizer in $\mathbb{S}$,

$$\min_{x \in \mathbb{S}} \quad f(x).$$



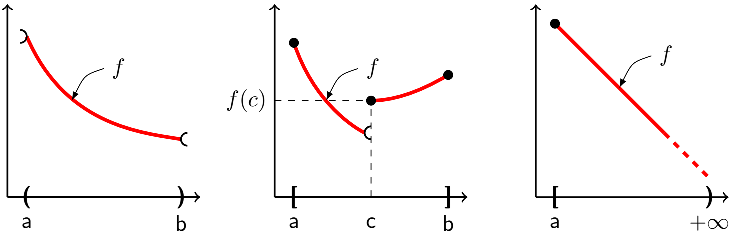
Existence of Minima and Maxima

Theorem (Extreme value theorem)

Let $\mathbb{S}$ be nonempty and compact and let f be continuous on $\mathbb{S}$.

Then, the following problem has a minimizer in $\mathbb{S}$,

$$\min_{x \in \mathbb{S}} \quad f(x).$$



Convex Analysis

Definition (Convex set)

A set $\mathbb{C} \subset \mathbb{R}^{n_x}$ is said to be convex if

$$\forall\, x, y \in \mathbb{C},\ \forall\, \lambda \in [0,1]: \quad z = \lambda x + (1-\lambda) y \in \mathbb{C}.$$

Definition (Convex function)

A function $f : \mathbb{C} \to \mathbb{R}$ is said to be convex on $\mathbb{C}$ if its domain $\mathbb{C}$ is aconvex set and if

$$\forall\, x, y \in \mathbb{C},\ \forall\, \lambda \in [0,1]: \quad f(\lambda x + (1-\lambda) y) \leq \lambda f(x) + (1-\lambda) f(y).$$

Convex Analysis

Definition (Convex set)

A set $\mathbb{C} \subset \mathbb{R}^{n_x}$ is said to be convex if

$$\forall\, x, y \in \mathbb{C},\ \forall\, \lambda \in [0,1]: \quad z = \lambda x + (1-\lambda) y \in \mathbb{C}.$$

Definition (Convex function)

A function $f : \mathbb{C} \to \mathbb{R}$ is said to be convex on $\mathbb{C}$ if its domain $\mathbb{C}$ is aconvex set and if

$$\forall\, x, y \in \mathbb{C},\ \forall\, \lambda \in [0,1]: \quad f(\lambda x + (1-\lambda) y) \leq \lambda f(x) + (1-\lambda) f(y).$$

Convex Program

Definition (Convex program)

Let $\mathbb{C}$ be a nonempty convex set, and let f be convex on $\mathbb{C}$. NLP

$$\min_{x \in \mathbb{C}} \quad f(x)$$

is called a convex program or convex optimization problem.

Theorem

Let x^* be a local minimizer of a convex program, then x^* is a also a global minimizer.

Convex Program

Definition (Convex program)

Let $\mathbb{C}$ be a nonempty convex set, and let f be convex on $\mathbb{C}$. NLP

$$\min_{x \in \mathbb{C}} \quad f(x)$$

is called a convex program or convex optimization problem.

Theorem

Let x^* be a local minimizer of a convex program, then x^* is a also a global minimizer.

Exercises

Given a convex function $f : \mathbb{R}^{n_x} \to \mathbb{R}$ and a non-empty compact set $\mathcal{S} \subseteq \mathbb{R}^{n_x}$. Let $\partial\mathcal{S}$ denote the boundary of the set $\mathcal{S}$.

Which of the following statements are correct? Justify your answers.

a) The minimum of f on $\mathcal{S}$ is unique.

b) The minimizer of f on $\mathcal{S}$ is unique.

c)

$$\arg\min_{x\in\mathcal{S}} f(x) \quad \cap \quad \partial\mathcal{S} \neq \emptyset$$

d)

$$\arg\max_{x\in\mathcal{S}} f(x) \quad \cap \quad \partial\mathcal{S} \neq \emptyset$$

Exercises

Given a convex function $f : \mathbb{R}^{n_x} \to \mathbb{R}$ and a non-empty compact set $\mathcal{S} \subseteq \mathbb{R}^{n_x}$. Let $\partial\mathcal{S}$ denote the boundary of the set $\mathcal{S}$.

Which of the following statements are correct? Justify your answers.

a) The minimum of f on $\mathcal{S}$ is unique.

b) The minimizer of f on $\mathcal{S}$ is unique.

c)

$$\arg\min_{x\in\mathcal{S}} f(x) \quad \cap \quad \partial\mathcal{S} \neq \emptyset$$

d)

$$\arg\max_{x\in\mathcal{S}} f(x) \quad \cap \quad \partial\mathcal{S} \neq \emptyset$$

Exercises

Given a convex function $f : \mathbb{R}^{n_x} \to \mathbb{R}$ and a non-empty compact set $\mathcal{S} \subseteq \mathbb{R}^{n_x}$. Let $\partial\mathcal{S}$ denote the boundary of the set $\mathcal{S}$.

Which of the following statements are correct? Justify your answers.

a) The minimum of f on $\mathcal{S}$ is unique.

b) The minimizer of f on $\mathcal{S}$ is unique.

c)

$$\arg\min_{x\in\mathcal{S}} f(x) \quad \cap \quad \partial\mathcal{S} \neq \emptyset$$

d)

$$\arg\max_{x\in\mathcal{S}} f(x) \quad \cap \quad \partial\mathcal{S} \neq \emptyset$$

Exercises

Given a convex function $f : \mathbb{R}^{n_x} \to \mathbb{R}$ and a non-empty compact set $\mathcal{S} \subseteq \mathbb{R}^{n_x}$. Let $\partial\mathcal{S}$ denote the boundary of the set $\mathcal{S}$.

Which of the following statements are correct? Justify your answers.

a) The minimum of f on $\mathcal{S}$ is unique.

b) The minimizer of f on $\mathcal{S}$ is unique.

c)

$$\arg\min_{x\in\mathcal{S}} f(x) \quad \cap \quad \partial\mathcal{S} \neq \emptyset$$

d)

$$\arg\max_{x\in\mathcal{S}} f(x) \quad \cap \quad \partial\mathcal{S} \neq \emptyset$$

Exercises

Given are the following for optimization problems:

a1)

$$\min_{x \in \mathbb{R}} c \cdot x \quad \text{subject to } 0 < x \leq 1$$

with $c \in \mathbb{R}$ arbitrary.

a1) Solution:

- If $c > 0$, the minimizer does not exist.
- If $c = 0$, any x satisfying $0 < x \leq 1$ is a minimizer.
- If $c < 0$, the minimizer is $x = 1$.

Exercises

Given are the following for optimization problems:

a1)

$$\min_{x \in \mathbb{R}} c \cdot x \quad \text{subject to } 0 < x \leq 1$$

with $c \in \mathbb{R}$ arbitrary.

a1) Solution:

- If $c > 0$, the minimizer does not exist.
- If $c = 0$, any x satisfying $0 < x \leq 1$ is a minimizer.
- If $c < 0$, the minimizer is $x = 1$.

Exercises

Given are the following for optimization problems:

a2)

$$\inf_{x \in \mathbb{R}} c \cdot x \quad \text{subject to } 0 < x \leq 1$$

with $c \in \mathbb{R}$ arbitrary.

a2) Solution:

- If $c > 0$, the infimum is 0 and $x \to 0$.
- If $c = 0$, the infimum is 0 with any x satisfying $0 < x \leq 1$.
- If $c < 0$, the infimum is c and $x = 1$.

Exercises

Given are the following for optimization problems:

a2)

$$\inf_{x \in \mathbb{R}} c \cdot x \quad \text{subject to } 0 < x \leq 1$$

with $c \in \mathbb{R}$ arbitrary.

a2) Solution:

- If $c > 0$, the infimum is 0 and $x \to 0$.
- If $c = 0$, the infimum is 0 with any x satisfying $0 < x \leq 1$.
- If $c < 0$, the infimum is c and $x = 1$.

Contents

Notation: Gradients and Partial Derivatives

Consider a function $f : \mathbb{R}^{n_x} \to \mathbb{R}$,

- Partial derivative (the Jacobian) of f

$$\frac{\partial f}{\partial x} = \begin{bmatrix} \frac{\partial f}{\partial x_1} & \frac{\partial f}{\partial x_2} & \dots & \frac{\partial f}{\partial x_{n_x}} \end{bmatrix} \in \mathbb{R}^{1 \times n_x}$$

- Gradient of f

$$\nabla f = \left(\frac{\partial f}{\partial x} \right)^\top \in \mathbb{R}^{n_x}$$

- $f \in \mathcal{C}^n$: f is n-times continuously differentiable on $\mathbb{R}^{n_x}$

Equality Constrained Problem

Consider NLP

$$\mathscr{P}_{\text{eq}}: \quad \min_{x \in \mathbb{R}^{n_x}} \; f(x) \quad \text{subject to } \; h_i(x) = 0, \; i \in \mathcal{E} := \{1, ..., n_h\}$$

Definition (Regular point)

Consider $\mathbb{S} := \{x \in \mathbb{R}^{n_x} \, | h_i(x) = 0, \; i \in \mathcal{E}\}$ with continuously differentiable $h_i : \mathbb{R}^{n_x} \to \mathbb{R}$, $i \in \mathcal{E}$ on $\mathbb{R}^{n_x}$.

A vector $\bar{x} \in \mathbb{S}$ is said to be a regular point if the gradient $\nabla h_i(\bar{x})$, $i \in \mathcal{E}$ are linearly independent, i.e.,

$$\frac{\partial h}{\partial x} \in \mathbb{R}^{n_h \times n_x} \text{ is full row rank.}$$

This is also called linear independence constraint qualification (LICQ).

Equality Constrained Problem

Consider NLP

$$\mathscr{P}_{\text{eq}} : \quad \min_{x \in \mathbb{R}^{n_x}} \quad f(x) \quad \text{subject to} \;\; h_i(x) = 0, \; i \in \mathcal{E} := \{1, ..., n_h\}$$

Definition (Regular point)

Consider $\mathbb{S} := \{x \in \mathbb{R}^{n_x} \, | h_i(x) = 0, \; i \in \mathcal{E}\}$ with continuously differentiable $h_i : \mathbb{R}^{n_x} \to \mathbb{R}$, $i \in \mathcal{E}$ on $\mathbb{R}^{n_x}$.

A vector $\bar{x} \in \mathbb{S}$ is said to be a regular point if the gradient $\nabla h_i(\bar{x})$, $i \in \mathcal{E}$ are linearly independent, i.e.,

$$\frac{\partial h}{\partial x} \in \mathbb{R}^{n_h \times n_x} \text{ is full row rank.}$$

This is also called linear independence constraint qualification (LICQ).

Equality Constrained Problem

Consider NLP

$$\mathscr{P}_{\text{eq}}: \quad \min_{x \in \mathbb{R}^{n_x}} \quad f(x) \quad \text{subject to} \ \ h_i(x) = 0, \ i \in \mathcal{E} := \{1, ..., n_h\}$$

Definition (Regular point)

Consider $\mathbb{S} := \{x \in \mathbb{R}^{n_x} \, | h_i(x) = 0, \ i \in \mathcal{E}\}$ with continuously differentiable $h_i : \mathbb{R}^{n_x} \to \mathbb{R}$, $i \in \mathcal{E}$ on $\mathbb{R}^{n_x}$.

A vector $\bar{x} \in \mathbb{S}$ is said to be a regular point if the gradient $\nabla h_i(\bar{x})$, $i \in \mathcal{E}$ are linearly independent, i.e.,

$$\frac{\partial h}{\partial x} \in \mathbb{R}^{n_h \times n_x} \text{ is full row rank.}$$

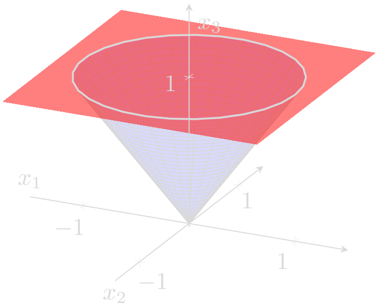
This is also called linear independence constraint qualification (LICQ).

Equality Constraints – Example

$$\mathbb{S} = \left\{ x \in \mathbb{R}^3 : h_i(x) = 0,\ i \in \{1, 2\} \right\}$$

$$h_1(x) = x_3 - (x_1^2 + x_2^2)$$

$$h_2(x) = x_3 - 1$$

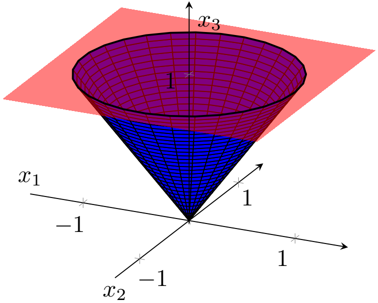


Equality Constraints – Example

$$\mathbb{S} = \left\{ x \in \mathbb{R}^3 : h_i(x) = 0,\ i \in \{1, 2\} \right\}$$

$$h_1(x) = x_3 - (x_1^2 + x_2^2)$$

$$h_2(x) = x_3 - 1$$



Necessary Condition of Optimality

Theorem (1st order optimality condition)

Consider Problem $\mathscr{P}_{\mathrm{eq}}$ *and let* $f : \mathbb{R}^{n_x} \to \mathbb{R}$, $h_i : \mathbb{R}^{n_x} \to \mathbb{R}$, $i \in \mathcal{E}$ *be continuously differentiable on* $\mathbb{R}^{n_x}$*.*

If a local minimizer x^* *is a regular point of the constraints, then there exists a unique vector* $\lambda^* \in \mathbb{R}^{n_h}$ *such that*

$$\nabla f(x^*) + \nabla h(x^*)\lambda^* = 0.$$

Active Constraints and Active Set

Consider generic NLP

$$\mathscr{P}_{\text{ieq}}: \quad \begin{aligned} &\min_{x \in \mathbb{R}^{n_x}} \quad f(x) \\ &\text{subject to} \begin{cases} h_i(x) = 0, \ i \in \mathcal{E} := \{1, ..., n_h\} \\ g_i(x) \leq 0, \ i \in \mathcal{I} := \{1, ..., n_g\} \end{cases} \end{aligned}$$

Definition (Active Constraint)

A constraint g_i is said to be active at $\bar{x}$, if $g_i(\bar{x}) = 0$.

Definition (Active Set)

The active set $\mathcal{A}(\bar{x})$ at any feasible $\bar{x}$ of $\mathscr{P}_{\text{ineq}}$ is denoted by

$$\mathcal{A}(\bar{x}) = \mathcal{E} \cup \{i \in \mathcal{I} | g_i(\bar{x}) = 0\}.$$

Active Constraints and Active Set

Consider generic NLP

$$\mathscr{P}_{\text{ieq}}: \qquad \min_{x\in\mathbb{R}^{n_x}} \; f(x) \qquad \text{subject to} \begin{cases} h_i(x) = 0,\ i \in \mathcal{E} := \{1, ..., n_h\} \\ g_i(x) \leq 0,\ i \in \mathcal{I} := \{1, ..., n_g\} \end{cases}$$

Definition (Active Constraint)

A constraint g_i is said to be active at $\bar{x}$, if $g_i(\bar{x}) = 0$.

Definition (Active Set)

The active set $\mathcal{A}(\bar{x})$ at any feasible $\bar{x}$ of $\mathscr{P}_{\text{ineq}}$ is denoted by

$$\mathcal{A}(\bar{x}) = \mathcal{E} \cup \{i \in \mathcal{I} | g_i(\bar{x}) = 0\}.$$

Active Constraints and Active Set

Consider generic NLP

$$\mathscr{P}_{\text{ieq}}: \qquad \begin{aligned} &\min_{x \in \mathbb{R}^{n_x}} \quad f(x) \\ &\text{subject to} \begin{cases} h_i(x) = 0,\ i \in \mathcal{E} := \{1, ..., n_h\} \\ g_i(x) \le 0,\ i \in \mathcal{I} := \{1, ..., n_g\} \end{cases} \end{aligned}$$

Definition (Active Constraint)

A constraint g_i is said to be active at $\bar{x}$, if $g_i(\bar{x}) = 0$.

Definition (Active Set)

The active set $\mathcal{A}(\bar{x})$ at any feasible $\bar{x}$ of $\mathscr{P}_{\text{ineq}}$ is denoted by

$$\mathcal{A}(\bar{x}) = \mathcal{E} \cup \{i \in \mathcal{I} | g_i(\bar{x}) = 0\}.$$

Regular Points of General NLPs

Definition

Let h_i, $i \in \mathcal{E}$ and g_i, $i \in \mathcal{I}$ be continuously differentiable on $\mathbb{R}^{n_x}$ and let

$$\nabla g_{\mathcal{A}}(\bar{x}) := [\nabla g_i(\bar{x})],\ i \in \mathcal{I} \cap \mathcal{A}(\bar{x})$$

with a feasible point $\bar{x}$ of $\mathscr{P}_{\text{ieq}}$. Then, $\bar{x}$ is said to be a regular point if

$$\operatorname{rank}([\nabla h(\bar{x}),\ \nabla g_{\mathcal{A}}(\bar{x})]^{\top}) = |\mathcal{A}(\bar{x})|.$$

Karush-Kuhn-Tucker (KKT) Conditions

Definition (KKT point)

Let f, h_i, $i \in \mathcal{E}$ and g_i, $i \in \mathcal{I}$ be continuously differentiable on $\mathbb{R}^{n_x}$. Consider Problem $\mathscr{P}_{\text{ieq}}$, any pair (x, λ, κ) with $x \in \mathbb{R}^{n_x}$, $\lambda \in \mathbb{R}^{n_h}$ and $\kappa \in \mathbb{R}^{n_g}$ satisfying

$$
\begin{aligned}
\textsc{Stationarity} \quad & 0 = \nabla f(x) + \sum_{i \in \mathcal{E}} \lambda_i \nabla h_i(x) + \sum_{i \in \mathcal{I}} \kappa_i \nabla g_i(x) \\
\textsc{Primal feasibility} \quad & 0 = h_i(x),\ i \in \mathcal{E},\ 0 \geq g_i(x),\ i \in \mathcal{I} \\
\textsc{Dual feasibility} \quad & 0 \leq \kappa_i,\ i \in \mathcal{I} \\
\textsc{Complementarity} \quad & 0 = \kappa_i g_i(x),\ i \in \mathcal{I}
\end{aligned}
$$

is called a KKT point of $\mathscr{P}_{\text{ieq}}$.

KKT Necessary Conditions of Optimality

Theorem

Consider Problem $\mathscr{P}_{\text{ieq}}$ *and let* f, h_i, $i \in \mathcal{E}$ *and* g_i, $i \in \mathcal{I}$ *be continuously differentiable on* $\mathbb{R}^{n_x}$. *If*

- x^* *is a (local) minimizer of* $\mathscr{P}_{\text{ieq}}$ *and*
- x^* *is a regular point,*

then there exist $\lambda^* \in \mathbb{R}^{n_h}$ *and* $\kappa^* \in \mathbb{R}^{n_g}$ *such that* $(x^*, \lambda^*, \kappa^*)$ *is a KKT point of* $\mathscr{P}_{\text{ieq}}$.

Exercises

Consider NLP

$$\min_{x \in \mathbb{R}^2} \quad \left(x_1 - \frac{3}{2}\right)^2 + (x_2 - t)^4 \quad \text{subject to} \begin{cases} x_1 + x_2 - 1 \leq 0 \\ x_1 - x_2 - 1 \leq 0 \\ -x_1 + x_2 - 1 \leq 0 \\ -x_1 - x_2 - 1 \leq 0 \end{cases}$$

For what value of t does $x^* = [1, 0]^\top$ satisfy the KKT condition?

Exercises

Consider NLP

$$\min_{x \in \mathbb{R}^2} \quad -2x_1 + x_2 \quad \text{subject to} \begin{cases} x_2 - (1 - x_1)^3 \leq 0 \\ 1 - 0.25x_1^2 - x_2 \leq 0 \end{cases}$$

the optimal solution is $x^* = [0, 1]^\top$, questions:

a) Is x^* a regular point?

b) Are the KKT conidtions satisfied?

Contents

Equality Constrained NLP

Consider NLP

$$\min_{x \in \mathbb{R}^{n_x}} \quad f(x) \quad \text{subject to} \;\; h(x) = 0$$

Necessary condition of optimality:

$$\nabla f(x^*) + \nabla h(x^*)\lambda^* = 0$$
$$h(x^*) = 0$$

Question: how does the minimum $f(x^*)$ change for varying constraints $h(x) = c$?

Equality Constrained NLP

Consider NLP

$$\min_{x \in \mathbb{R}^{n_x}} \quad f(x) \quad \text{subject to} \;\; h(x) = 0$$

Necessary condition of optimality:

$$\begin{aligned} \nabla f(x^*) + \nabla h(x^*)\lambda^* &= 0 \\ h(x^*) &= 0 \end{aligned}$$

Question: how does the minimum $f(x^*)$ change for varying constraints $h(x) = c$?

Equality Constrained NLP

Consider NLP

$$\min_{x \in \mathbb{R}^{n_x}} \quad f(x) \quad \text{subject to} \;\; h(x) = 0$$

Necessary condition of optimality:

$$\begin{aligned} \nabla f(x^*) + \nabla h(x^*)\lambda^* &= 0 \\ h(x^*) &= 0 \end{aligned}$$

Question: how does the minimum $f(x^*)$ change for varying constraints $h(x) = c$?

Interpretation of Lagrangian Multipliers

Perturbed problem

$$\mathscr{P}_c: \qquad \min_{x \in \mathbb{R}^{n_x}} \quad f(x) \quad \text{subject to} \quad h(x) = c$$

Assumption

For each c, $\mathscr{P}_c$ has a unique regular solution, i.e.,

$$\xi^\star(c) = \arg\min_x \quad f(x) \quad \textit{subject to} \quad h(x) = c$$

$$\phi^\star(c) = \min_x \quad f(x) \quad \textit{subject to} \quad h(x) = c$$

with $\xi^\star(0) = x^*$ and $\phi^\star(0) = f(x^*)$.

Interpretation of Lagrangian Multipliers

Perturbed problem

$$\mathscr{P}_c : \qquad \min_{x \in \mathbb{R}^{n_x}} \quad f(x) \quad \text{subject to} \;\; h(x) = c$$

Assumption

For each c, $\mathscr{P}_c$ has a unique regular solution, i.e.,

$$\xi^\star(c) = \arg\min_x \quad f(x) \quad \textit{subject to} \;\; h(x) = c$$

$$\phi^\star(c) = \min_x \quad f(x) \quad \textit{subject to} \;\; h(x) = c$$

with $\xi^\star(0) = x^*$ and $\phi^\star(0) = f(x^*)$.

Interpretation of Lagrangian Multipliers

$$h(\xi^\star(c)) = c \;\Rightarrow\; \nabla_x h(\xi^\star(c))^\top \nabla_c \xi^\star(c)^\top = \mathrm{I}$$

$$\begin{aligned}
\nabla_c \phi^\star(c)\Big|_{c=0} &= \nabla_c \xi^\star(0)^\top \nabla_x f(x^*) \\
&= -\underbrace{\nabla_c \xi^\star(0)^\top \nabla_x h(\xi^\star(0))^\top}_{\mathrm{I}} \lambda^* \\
&= -\lambda^*
\end{aligned}$$

Interpretation of Lagrangian Multipliers

$$h(\xi^\star(c)) = c \;\Rightarrow\; \nabla_x h(\xi^\star(c))^\top \nabla_c \xi^\star(c)^\top = \mathrm{I}$$

$$\begin{aligned}
\nabla_c \phi^\star(c)\Big|_{c=0} &= \nabla_c \xi^\star(0)^\top \nabla_x f(x^*) \\
&= -\underbrace{\nabla_c \xi^\star(0)^\top \nabla_x h(\xi^\star(0))^\top}_{\mathrm{I}} \lambda^* \\
&= -\lambda^*
\end{aligned}$$

Interpretation of Lagrangian Multipliers

- The Lagrange multiplier λ^* can be interpreted as the sensitivity of the optimal objective function with respect to changes in the constraint $h(x) = 0$.
- In Economics Lagrange multipliers are used to characterize *marginal values* or *shadow prices*.
- Can be extended to general NLPs with inequality constraints $g(x) \leq c$: multipliers $\kappa^* \approx$ sensitivity of $f(x^*)$ with respect to c.
- Inactive inequality constraints $\kappa_i^* = 0$, $i \in \mathcal{I} \backslash (\mathcal{I} \cap \mathcal{A}(x^*)) \Rightarrow$ no change of optimum for small perturbations.
- Active inequality constraints $\kappa_i^* \geq 0$, $i \in \mathcal{I} \cap \mathcal{A}(x^*) \Rightarrow$ enlarged feasible region, optimal cost cannot increase.

Interpretation of Lagrangian Multipliers

- The Lagrange multiplier λ^* can be interpreted as the sensitivity of the optimal objective function with respect to changes in the constraint $h(x) = 0$.
- In Economics Lagrange multipliers are used to characterize *marginal values* or *shadow prices*.
- Can be extended to general NLPs with inequality constraints $g(x) \leq c$: multipliers $\kappa^* \approx$ sensitivity of $f(x^*)$ with respect to c.
- Inactive inequality constraints $\kappa_i^* = 0$, $i \in \mathcal{I} \backslash (\mathcal{I} \cap \mathcal{A}(x^*)) \Rightarrow$ no change of optimum for small perturbations.
- Active inequality constraints $\kappa_i^* \geq 0$, $i \in \mathcal{I} \cap \mathcal{A}(x^*) \Rightarrow$ enlarged feasible region, optimal cost cannot increase.

Interpretation of Lagrangian Multipliers

- The Lagrange multiplier λ^* can be interpreted as the sensitivity of the optimal objective function with respect to changes in the constraint $h(x) = 0$.
- In Economics Lagrange multipliers are used to characterize *marginal values* or *shadow prices*.
- Can be extended to general NLPs with inequality constraints $g(x) \leq c$: multipliers $\kappa^* \approx$ sensitivity of $f(x^*)$ with respect to c.
- Inactive inequality constraints $\kappa_i^* = 0$, $i \in \mathcal{I} \backslash (\mathcal{I} \cap \mathcal{A}(x^*))$ $\Rightarrow$ no change of optimum for small perturbations.
- Active inequality constraints $\kappa_i^* \geq 0$, $i \in \mathcal{I} \cap \mathcal{A}(x^*)$ $\Rightarrow$ enlarged feasible region, optimal cost cannot increase.

Interpretation of Lagrangian Multipliers

- The Lagrange multiplier λ^* can be interpreted as the sensitivity of the optimal objective function with respect to changes in the constraint $h(x) = 0$.
- In Economics Lagrange multipliers are used to characterize *marginal values* or *shadow prices*.
- Can be extended to general NLPs with inequality constraints $g(x) \leq c$: multipliers $\kappa^* \approx$ sensitivity of $f(x^*)$ with respect to c.
- Inactive inequality constraints $\kappa_i^* = 0$, $i \in \mathcal{I} \backslash (\mathcal{I} \cap \mathcal{A}(x^*)) \Rightarrow$ no change of optimum for small perturbations.
- Active inequality constraints $\kappa_i^* \geq 0$, $i \in \mathcal{I} \cap \mathcal{A}(x^*) \Rightarrow$ enlarged feasible region, optimal cost cannot increase.

Interpretation of Lagrangian Multipliers

- The Lagrange multiplier λ^* can be interpreted as the sensitivity of the optimal objective function with respect to changes in the constraint $h(x) = 0$.
- In Economics Lagrange multipliers are used to characterize *marginal values* or *shadow prices*.
- Can be extended to general NLPs with inequality constraints $g(x) \leq c$: multipliers $\kappa^* \approx$ sensitivity of $f(x^*)$ with respect to c.
- Inactive inequality constraints $\kappa_i^* = 0$, $i \in \mathcal{I} \backslash (\mathcal{I} \cap \mathcal{A}(x^*))$ $\Rightarrow$ no change of optimum for small perturbations.
- Active inequality constraints $\kappa_i^* \geq 0$, $i \in \mathcal{I} \cap \mathcal{A}(x^*)$ $\Rightarrow$ enlarged feasible region, optimal cost cannot increase.

Contents

Algorithm Concepts

Algorithm:

- Given an initial point x^0 compute a sequence $\{x^k\}$ by repeated application of an algorithmic rule.
- Objective: make $\{x^k\}$ converge to a point $\bar{x}$.

Why do we talk about algorithms for NLPs?

- Solvers usually require initial guess and termination criteria $\Rightarrow$ basic understanding of solution algorithms necessary to use solvers.
- Solvers often terminate prematurely $\Rightarrow$ understand and diagnose reasons?

Algorithm Concepts

Algorithm:

- Given an initial point x^0 compute a sequence $\{x^k\}$ by repeated application of an algorithmic rule.
- Objective: make $\{x^k\}$ converge to a point $\bar{x}$.

Why do we talk about algorithms for NLPs?

- Solvers usually require initial guess and termination criteria $\Rightarrow$ basic understanding of solution algorithms necessary to use solvers.
- Solvers often terminate prematurely $\Rightarrow$ understand and diagnose reasons?

Algorithm Concepts

Algorithm:

- Given an initial point x^0 compute a sequence $\{x^k\}$ by repeated application of an algorithmic rule.
- Objective: make $\{x^k\}$ converge to a point $\bar{x}$.

Why do we talk about algorithms for NLPs?

- Solvers usually require initial guess and termination criteria $\Rightarrow$ basic understanding of solution algorithms necessary to use solvers.
- Solvers often terminate prematurely $\Rightarrow$ understand and diagnose reasons?

Global and Local Convergence

Definition (Global convergence)

An algorithm is said to be *globally convergent* if, for any initial point x^0, it generates a sequence of points that converges to a point $\bar{x}$ in the solution set.

Definition (Local convergence)

An algorithm is said to be *locally convergent* if there exists $\rho > 0$ such that for any initial point x^0 with $\|x^0 - \bar{x}\| < \rho$, it generates a sequence of points that converges to a point $\bar{x}$ in the solution set.

Global and Local Convergence

Definition (Global convergence)

An algorithm is said to be *globally convergent* if, for any initial point x^0, it generates a sequence of points that converges to a point $\bar{x}$ in the solution set.

Definition (Local convergence)

An algorithm is said to be *locally convergent* if there exists $\rho > 0$ such that for any initial point x^0 with $\|x^0 - \bar{x}\| < \rho$, it generates a sequence of points that converges to a point $\bar{x}$ in the solution set.

Order of Convergence

Definition

The *order of convergence* of a sequence $\{x^k\}$, with $\lim_{k \to \infty} x^k = \bar{x}$, is the largest non-negative number p such that

$$\lim_{k \to \infty} \frac{\|x^{k+1} - \bar{x}\|}{\|x^k - \bar{x}\|^p} = \beta < \infty.$$

- $p = 1$ and $\beta < 1 \Rightarrow$ linear convergence
- $p = 1$ and $\beta = 0 \Rightarrow$ superlinear convergence
- $p = 1$ and $\beta = 1 \Rightarrow$ sublinear convergence
- $p = 2 \Rightarrow$ quadratic convergence

Order of Convergence

Definition

The *order of convergence* of a sequence $\{x^k\}$, with $\lim_{k\to\infty} x^k = \bar{x}$, is the largest non-negative number p such that

$$\lim_{k\to\infty} \frac{\|x^{k+1} - \bar{x}\|}{\|x^k - \bar{x}\|^p} = \beta < \infty.$$

- $p = 1$ and $\beta < 1 \Rightarrow$ linear convergence
- $p = 1$ and $\beta = 0 \Rightarrow$ superlinear convergence
- $p = 1$ and $\beta = 1 \Rightarrow$ sublinear convergence
- $p = 2 \Rightarrow$ quadratic convergence

Order to Convergence

Definition

The *order of convergence* of a sequence $\{x^k\}$, with $\lim_{k\to\infty} x^k = \bar{x}$, is the largest non-negative number p such that

$$\lim_{k\to\infty} \frac{\|x^{k+1} - \bar{x}\|}{\|x^k - \bar{x}\|^p} = \beta < \infty.$$

Example:

$$x^k = 1 + 0.5^k$$

$$x^k = 1 + k^{-k}$$

$$x^k = 1 + 0.5^{2^k}$$

Newton's Methd for Nonlinear Equations

Given a function $F : \mathbb{R}^n \to \mathbb{R}^n$, search for solutions of the nonlinear equation

$$F(x) = 0 \text{ with } F \in \mathbb{C}^1.$$

Main idea:

- Start with x_0 and solve linear equations

$$F(x^k) + M(x^k)(x^{k+1} - x^k) = 0,\ k \in \{1, 2, \ldots\}.$$

- Matrix $M(x)_k \in \mathbb{R}^{n_x \times n_x}$ chosen in such a way that

$$F(x^k) + M(x^k)(x - x^k) \approx F(x)$$

  is an approximation of F.

- $M(x^k) = \partial F(x^k)$ corresponds to the so called Newton method.

Newton's Methd for Nonlinear Equations

Given a function $F : \mathbb{R}^n \to \mathbb{R}^n$, search for solutions of the nonlinear equation

$$F(x) = 0 \text{ with } F \in \mathbb{C}^1.$$

Main idea:

- Start with x_0 and solve linear equations

$$F(x^k) + M(x^k)(x^{k+1} - x^k) = 0,\ k \in \{1, 2, ...\}.$$

- Matrix $M(x)_k \in \mathbb{R}^{n_x \times n_x}$ chosen in such a way that

$$F(x^k) + M(x^k)(x - x^k) \approx F(x)$$

  is an approximation of F.

- $M(x^k) = \partial F(x^k)$ corresponds to the so called Newton method.

Newton's Methd for Nonlinear Equations

Given a function $F : \mathbb{R}^n \to \mathbb{R}^n$, search for solutions of the nonlinear equation

$$F(x) = 0 \text{ with } F \in \mathbb{C}^1.$$

Main idea:

- Start with x_0 and solve linear equations

$$F(x^k) + M(x^k)(x^{k+1} - x^k) = 0,\ k \in \{1, 2, ...\}.$$

- Matrix $M(x)_k \in \mathbb{R}^{n_x \times n_x}$ chosen in such a way that

$$F(x^k) + M(x^k)(x - x^k) \approx F(x)$$

  is an approximation of F.

- $M(x^k) = \partial F(x^k)$ corresponds to the so called Newton method.

Newton's Methd for Nonlinear Equations

Given a function $F : \mathbb{R}^n \to \mathbb{R}^n$, search for solutions of the nonlinear equation

$$F(x) = 0 \text{ with } F \in \mathbb{C}^1.$$

Main idea:

- Start with x_0 and solve linear equations

$$F(x^k) + M(x^k)(x^{k+1} - x^k) = 0,\ k \in \{1, 2, ...\}.$$

- Matrix $M(x)_k \in \mathbb{R}^{n_x \times n_x}$ chosen in such a way that

$$F(x^k) + M(x^k)(x - x^k) \approx F(x)$$

  is an approximation of F.

- $M(x^k) = \partial F(x^k)$ corresponds to the so called Newton method.

Newton's Method for Nonlinear Equations

If $M(x^k)$ is invertible, the method can be written in the form

$$x^{k+1} = x^k - M(x^k)^{-1} F(x^k),\ k \in \{1, 2, ...\}.$$

- In practice, we usually work with approximations $M(x^k) \approx \partial F(x^k)$.
- If $M(x^k)$ is independent of x^k, we only need to decompose M once (e.g., using LR or QR decomposition).
- Some methods try to update M at every step without re-computing the Jacobian (e.g., BFGS update).

Newton's Method for Nonlinear Equations

If $M(x^k)$ is invertible, the method can be written in the form

$$x^{k+1} = x^k - M(x^k)^{-1} F(x^k),\ k \in \{1, 2, ...\}.$$

- In practice, we usually work with approximations $M(x^k) \approx \partial F(x^k)$.
- If $M(x^k)$ is independent of x^k, we only need to decompose M once (e.g., using LR or QR decomposition).
- Some methods try to update M at every step without re-computing the Jacobian (e.g., BFGS update).

Newton's Method for Nonlinear Equations

If $M(x^k)$ is invertible, the method can be written in the form

$$x^{k+1} = x^k - M(x^k)^{-1} F(x^k),\ k \in \{1, 2, ...\}.$$

- In practice, we usually work with approximations $M(x^k) \approx \partial F(x^k)$.
- If $M(x^k)$ is independent of x^k, we only need to decompose M once (e.g., using LR or QR decomposition).
- Some methods try to update M at every step without re-computing the Jacobian (e.g., BFGS update).

Scaling Properties of Newton's Method

- $F(x^*) = 0 \Rightarrow S \cdot F(x^*) = 0$ with $S \in \mathbb{R}^{n_x \times n_x}$ any (invertible) scaling matrix.
- Applying Newton's method to solve scaled equation

$$\widetilde{F}(x) = S \cdot F(x) = 0$$

  yields iteration $x^{k+1} = x^k - M(x^k)^{-1} S \cdot F(x^k)$.
- Using exact Jacobian $M(x^k) = \partial \widetilde{F}(x^k)$, we have

$$x^{k+1} = x^k - \partial F(x^k)^{-1} F(x^k).$$

- Newton's methods with exact Jacobians is invariant under scaling.

Local Convergence of Newton's Method

Assumption

- *There exists a point x^* with $F(x^*) = 0$.*
- *The initial point x^0 is already in a small neighborhood of x^*.*
- *Matrix $M(x^k)^{-1}\partial F(x)$ is Lipschitz continuous w.r.t. x in a neighborhood of x^* with constant $\omega \geq 0$.*

The basic idea is to estimate the distance of the iterates to x^*:

$$
\begin{aligned}
& \left\| x^{k+1} - x^* \right\| = \left\| x^k - x^* - M(x^k)^{-1} F(x^k) \right\| \\
= \quad & \left\| x^k - x^* - M(x^k)^{-1} \int_0^1 \partial F(x^* + s(x^k - x^*))(x^k - x^*) ds \right\| \\
\leq \quad & \left\| x^k - x^* - M(x^k)^{-1} \partial F(x^k)(x^k - x^*) \right\| + \tfrac{\omega}{2} \left\| x^k - x^* \right\|^2
\end{aligned}
$$

Local Convergence of Newton's Method

Assumption

- *There exists a point* x^* *with* $F(x^*) = 0$.
- *The initial point* x^0 *is already in a small neighborhood of* x^*.
- *Matrix* $M(x^k)^{-1}\partial F(x)$ *is Lipschitz continuous w.r.t.* x *in a neighborhood of* x^* *with constant* $\omega \geq 0$.

The basic idea is to estimate the distance of the iterates to x^*:

$$
\begin{aligned}
& \left\|x^{k+1} - x^*\right\| = \left\|x^k - x^* - M(x^k)^{-1}F(x^k)\right\| \\
= \quad & \left\|x^k - x^* - M(x^k)^{-1}\int_0^1 \partial F(x^* + s(x^k - x^*))(x^k - x^*)ds\right\| \\
\leq \quad & \left\|x^k - x^* - M(x^k)^{-1}\partial F(x^k)(x^k - x^*)\right\| + \frac{\omega}{2}\left\|x^k - x^*\right\|^2
\end{aligned}
$$

Local Convergence of Newton's Method

In summary, we have the estimate

$$\left\|x^{k+1} - x^*\right\| \leq \eta \left\|x^k - x^*\right\| + \frac{\omega}{2} \left\|x^k - x^*\right\|^2$$

as long as $\left\|\mathrm{I} - M(x^k)^{-1} \partial F(x^k)\right\| \leq \eta$. Here, η can be interpreted as a bound on the accuracy of the Jacobian approximation M.

If we have $\eta < 1$ and $\|x^0 - x^*\| < \frac{2}{\omega}(1 - \eta)$, the iterates contract and we have

$$\lim_{k \to \infty} x^k \to x^*.$$

Local Convergence of Newton's Method

In summary, we have the estimate

$$\left\|x^{k+1} - x^*\right\| \leq \eta \left\|x^k - x^*\right\| + \frac{\omega}{2} \left\|x^k - x^*\right\|^2$$

as long as $\left\|\mathrm{I} - M(x^k)^{-1}\partial F(x^k)\right\| \leq \eta$. Here, η can be interpreted as a bound on the accuracy of the Jacobian approximation M.

If we have $\eta < 1$ and $\|x^0 - x^*\| < \frac{2}{\omega}(1-\eta)$, the iterates contract and we have

$$\lim_{k \to \infty} x^k \to x^*.$$

Convergence of Newton's Method

The convergence rate estimate

$$\left\|x^{k+1} - x^*\right\| \leq \eta \left\|x^k - x^*\right\| + \frac{\omega}{2} \left\|x^k - x^*\right\|^2$$

implies that

- if we have $0 < \eta < 1$, the convergence rate is linear.
- if we choose $M(x^k) = \partial F(x^k)$, we have $\eta = 0$ and

$$\left\|x^{k+1} - x^*\right\| \leq \frac{\omega}{2} \left\|x^k - x^*\right\|^2 .$$

  In this case, the convergence rate is quadratic.

Convergence of Newton's Method

The convergence rate estimate

$$\left\|x^{k+1}-x^*\right\| \leq \eta \left\|x^k - x^*\right\| + \frac{\omega}{2}\left\|x^k - x^*\right\|^2$$

implies that

- if we have $0 < \eta < 1$, the convergence rate is linear.
- if we choose $M(x^k) = \partial F(x^k)$, we have $\eta = 0$ and

$$\left\|x^{k+1}-x^*\right\| \leq \frac{\omega}{2}\left\|x^k - x^*\right\|^2 .$$

  In this case, the convergence rate is quadratic.

Convergence of Newton's Method

The convergence rate estimate

$$\left\|x^{k+1} - x^*\right\| \leq \eta \left\|x^k - x^*\right\| + \frac{\omega}{2} \left\|x^k - x^*\right\|^2$$

implies that

- if we have $0 < \eta < 1$, the convergence rate is linear.
- if we choose $M(x^k) = \partial F(x^k)$, we have $\eta = 0$ and

$$\left\|x^{k+1} - x^*\right\| \leq \frac{\omega}{2} \left\|x^k - x^*\right\|^2 .$$

  In this case, the convergence rate is quadratic.

Exercises

Let scalar function $f : \mathbb{R} \to \mathbb{R}$ be three times continuously differentiable with bounded third-order derivative. The first and second derivative of f are denoted by f' and f'', respectively. We additionally assume:

- $f(x^*) = 0$ and $f''(x^*) = 0$ at a point $x^* \in \mathbb{R}$;
- $f'(x^*) \neq 0$.

Prove that the iterates of the exact Newton method, $x^{k+1} = x^k - \frac{f(x^k)}{f'(x^k)}$, converge locally with cubic convergence rate, i.e.,

$$\left|x^{k+1} - x^*\right| \leq \gamma \left|x^k - x^*\right|^3, \;\; \gamma < \infty.$$

Exercises

Solution:

1. Locally, we have

$$\left|x^{k+1}-x^*\right| = \left|x^k - x^* - \frac{f(x^k)}{f'(x^k)}\right| = \left|x^k - x^* - \frac{1}{f'(x^k)}\int_{x^*}^{x^k} f'(z)dz\right|$$

2. For the integral above, we can substitute the Taylor expansion,

$$\begin{aligned} f'(z) =& f'(x^k) + f''(x^k)(z-x^k) + \mathbf{O}(|z-x^k|^2) \\ =& f'(x^k) + \mathbf{O}(|x^k-x^*||z-x^k|) + \mathbf{O}(|z-x^k|^2) \end{aligned}$$

3. Thus, we have

$$\begin{aligned} \left|x^{k+1}-x^*\right| \le& \left|x^k - x^* - \frac{1}{f'(x^k)}\int_{x^*}^{x^k} f'(x^k)dz\right| + \mathbf{O}(|x^k-x^*|^3) \\ =& \mathbf{O}(|x^k-x^*|^3) \end{aligned}$$

Exercises

Solution:

1. Locally, we have

$$\left|x^{k+1} - x^*\right| = \left|x^k - x^* - \frac{f(x^k)}{f'(x^k)}\right| = \left|x^k - x^* - \frac{1}{f'(x^k)}\int_{x^*}^{x^k} f'(z)dz\right|$$

2. For the integral above, we can substitute the Taylor expansion,

$$\begin{aligned} f'(z) =& f'(x^k) + f''(x^k)(z - x^k) + \mathbf{O}(|z - x^k|^2) \\ =& f'(x^k) + \mathbf{O}(|x^k - x^*||z - x^k|) + \mathbf{O}(|z - x^k|^2) \end{aligned}$$

3. Thus, we have

$$\begin{aligned} \left|x^{k+1} - x^*\right| \leq& \left|x^k - x^* - \frac{1}{f'(x^k)}\int_{x^*}^{x^k} f'(x^k)dz\right| + \mathbf{O}(|x^k - x^*|^3) \\ =& \mathbf{O}(|x^k - x^*|^3) \end{aligned}$$

Exercises

Solution:

1. Locally, we have

$$\left|x^{k+1} - x^*\right| = \left|x^k - x^* - \frac{f(x^k)}{f'(x^k)}\right| = \left|x^k - x^* - \frac{1}{f'(x^k)}\int_{x^*}^{x^k} f'(z)dz\right|$$

2. For the integral above, we can substitute the Taylor expansion,

$$\begin{aligned} f'(z) =& f'(x^k) + f''(x^k)(z - x^k) + \mathbf{O}(|z - x^k|^2) \\ =& f'(x^k) + \mathbf{O}(|x^k - x^*||z - x^k|) + \mathbf{O}(|z - x^k|^2) \end{aligned}$$

3. Thus, we have

$$\begin{aligned} \left|x^{k+1} - x^*\right| \le& \left|x^k - x^* - \frac{1}{f'(x^k)}\int_{x^*}^{x^k} f'(x^k)dz\right| + \mathbf{O}(|x^k - x^*|^3) \\ =& \mathbf{O}(|x^k - x^*|^3) \end{aligned}$$

Newton's Method for Unconstrained Optimization

Problem formulation:

$$\min_{x \in \mathbb{R}^{n_x}} \quad f(x)$$

Remark

- *If f is twice Lipschitz-continuously differentiable, a minimizer can be founded by applying Newton's method to*

$$\nabla f(x) = 0.$$

- *If a solution x^* satisfies $\nabla^2 f(x) \succ 0$, it must a local minimizer.*

Newton's Method for Unconstrained Optimization

Newton-type iteration for unconstrained optimization problem

$$x^{k+1} = x^k - M(x^k)^{-1}\nabla f(x^k)$$

with $M(x^k) \approx \nabla^2 f(x^k)$ a suitable Hessian approximation.

- In practice, we often choose a symmetric M.
- If $M(x^k)$ is symmetric and positive definite, the iterate x^{k+1} is the minimizer of the quadratic function

  $$\min_{x^{k+1}} \; f(x^k) + \nabla f(x^k)^\top (x^{k+1} - x^k) + \frac{1}{2}(x^{k+1} - x^k)^\top M(x^k)(x^{k+1} - x^k),$$

  which can be interpreted as a quadratic model of f.

Line Search Methods

So far, we have only analyzed the local convergence properties of Newton-type methods. If we start far from a local solution, Newton type methods are often take "too big" steps and are divergent.

One way to fix this problem is to first compute a step-direction by

$$\Delta x^k = -M(x^k)^{-1} \nabla f(x^k)$$

and update the iterate as

$$x^{k+1} = x^k + \alpha^k \Delta x^k .$$

Here, $\alpha^k \in (0, 1]$ is a so-called step size, which is found by (approximately) solving the scalar optimization problem

$$\min_{\alpha^k \in (0,1]} \quad f(x^k + \alpha^k \Delta x^k).$$

Line Search Methods

So far, we have only analyzed the local convergence properties of Newton-type methods. If we start far from a local solution, Newton type methods are often take "too big" steps and are divergent.

One way to fix this problem is to first compute a step-direction by

$$\Delta x^k = -M(x^k)^{-1} \nabla f(x^k)$$

and update the iterate as

$$x^{k+1} = x^k + \alpha^k \Delta x^k .$$

Here, $\alpha^k \in (0, 1]$ is a so-called step size, which is found by (approximately) solving the scalar optimization problem

$$\min_{\alpha^k \in (0,1]} \quad f(x^k + \alpha^k \Delta x^k).$$

Line Search Methods

So far, we have only analyzed the local convergence properties of Newton-type methods. If we start far from a local solution, Newton type methods are often take "too big" steps and are divergent.

One way to fix this problem is to first compute a step-direction by

$$\Delta x^k = -M(x^k)^{-1} \nabla f(x^k)$$

and update the iterate as

$$x^{k+1} = x^k + \alpha^k \Delta x^k .$$

Here, $\alpha^k \in (0, 1]$ is a so-called step size, which is found by (approximately) solving the scalar optimization problem

$$\min_{\alpha^k \in (0,1]} \quad f(x^k + \alpha^k \Delta x^k).$$

Armijo Linear Search Conditions

In practice the line search optimization

$$\min_{\alpha^k \in (0,1]} \quad f(x^k + \alpha^k \Delta x^k).$$

is not solved exactly (too expensive), but only approximately.

One way to implement this is by using back-tracking until the Armijo condition

$$f(x^k + \alpha^k \Delta x^k) \leq f(x^k) + c \cdot \alpha^k \quad \underbrace{\nabla f(x^k)^\top \Delta x^k}_{\text{DIRECTIONAL DERIVATIVE}}$$

for a constant $c \ll 1$ is satisfied. This condition ensures that the line search parameter is not excessively large, although it is not sufficient to prove convergence in general.

Armijo Linear Search Conditions

In practice the line search optimization

$$\min_{\alpha^k \in (0,1]} \quad f(x^k + \alpha^k \Delta x^k).$$

is not solved exactly (too expensive), but only approximately.

One way to implement this is by using back-tracking until the Armijo condition

$$f(x^k + \alpha^k \Delta x^k) \leq f(x^k) + c \cdot \alpha^k \quad \underbrace{\nabla f(x^k)^\top \Delta x^k}_{\text{DIRECTIONAL DERIVATIVE}}$$

for a constant $c \ll 1$ is satisfied. This condition ensures that the line search parameter is not excessively large, although it is not sufficient to prove convergence in general.

Armijo Linear Search Conditions

In practice the line search optimization

$$\min_{\alpha^k \in (0,1]} \quad f(x^k + \alpha^k \Delta x^k).$$

is not solved exactly (too expensive), but only approximately.

One way to implement this is by using back-tracking until the Armijo condition

$$f(x^k + \alpha^k \Delta x^k) \leq f(x^k) + c \cdot \alpha^k \quad \underbrace{\nabla f(x^k)^\top \Delta x^k}_{\text{DIRECTIONAL DERIVATIVE}}$$

for a constant $c \ll 1$ is satisfied. This condition ensures that the line search parameter is not excessively large, although it is not sufficient to prove convergence in general.

Quasi-Newton Methods – Preliminaries

One way to represent invertible matrices is by considering matrices of the form

$$A = \underbrace{B}_{\text{EASY-TO-STORE}} + \underbrace{UV^\top}_{\text{LOW RANK}}$$

with $B \in \mathbb{R}^{n\times n}$ and $U, V \in \mathbb{R}^{n\times m}$, $m \ll n$.

If B is easy to invert or B^{-1} is already known, we have A^{-1} as

$$(B + UV)^{-1} = B^{-1} - B^{-1}U(I + V^\top B^{-1}U)^{-1}V^\top B^{-1},$$

which is the so-called "Woodbury's matrix inversion formula".

Quasi-Newton Methods – Preliminaries

One way to represent invertible matrices is by considering matrices of the form

$$A = \underbrace{B}_{\text{EASY-TO-STORE}} + \underbrace{UV^\top}_{\text{LOW RANK}}$$

with $B \in \mathbb{R}^{n\times n}$ and $U, V \in \mathbb{R}^{n\times m}$, $m \ll n$.

If B is easy to invert or B^{-1} is already known, we have A^{-1} as

$$(B + UV)^{-1} = B^{-1} - B^{-1}U(I + V^\top B^{-1}U)^{-1}V^\top B^{-1},$$

which is the so-called "Woodbury's matrix inversion formula".

Quasi-Newton Methods – Broyden's Updates

The Newton-type iterates

$$x^k = x^{k-1} - M(x^{k-1})^{-1}\nabla f(x^{k-1}),\ x^{k+1} = x^k - M(x^k)^{-1}\nabla f(x^k), \ldots$$

We have to compute the gradient ∇f at each iteration such that we can obtain the directional estimate

$$\nabla^2 f(x^{k+1})(x^{k+1} - x^k) \approx \nabla f(x^{k+1}) - \nabla f(x^k).$$

Questions: can we use this relation to improve our next Hessian approximation $M(x^{k+1}) \approx \nabla^2 f(x^{k+1})$?

Quasi-Newton Methods – Broyden's Updates

The Newton-type iterates

$$x^k = x^{k-1} - M(x^{k-1})^{-1}\nabla f(x^{k-1}),\ x^{k+1} = x^k - M(x^k)^{-1}\nabla f(x^k), \ldots$$

We have to compute the gradient ∇f at each iteration such that we can obtain the directional estimate

$$\nabla^2 f(x^{k+1})(x^{k+1} - x^k) \approx \nabla f(x^{k+1}) - \nabla f(x^k).$$

Questions: can we use this relation to improve our next Hessian approximation $M(x^{k+1}) \approx \nabla^2 f(x^{k+1})$?

Quasi-Newton Methods – Broyden's Updates

The Newton-type iterates

$$x^k = x^{k-1} - M(x^{k-1})^{-1}\nabla f(x^{k-1}),\ x^{k+1} = x^k - M(x^k)^{-1}\nabla f(x^k), \ldots$$

We have to compute the gradient ∇f at each iteration such that we can obtain the directional estimate

$$\nabla^2 f(x^{k+1})(x^{k+1} - x^k) \approx \nabla f(x^{k+1}) - \nabla f(x^k).$$

Questions: can we use this relation to improve our next Hessian approximation $M(x^{k+1}) \approx \nabla^2 f(x^{k+1})$?

Quasi-Newton Methods – Broyden's Updates

Define $d^k = x^{k+1} - x^k$ and $y^k = \nabla f(x^{k+1}) - \nabla f(x^k)$, the relation

$$\nabla^2 f(x^{k+1}) d^k \approx y^k$$

motivates to improve our current estimate of $\nabla^2 f$ constructing M^+ by solving

$$\min_{M^+} \ \frac{1}{2} \left\| M^+ - M(x^k) \right\|_F^2 \quad \text{subject to} \ M^+ d^k = y^k$$

with $\| \cdot \|_F$ the Frobenius norms ($\|X\|_F^2 = \mathrm{Tr}(XX^\top)$).

Quasi-Newton Methods – Broyden's Updates

Define $d^k = x^{k+1} - x^k$ and $y^k = \nabla f(x^{k+1}) - \nabla f(x^k)$, the relation

$$\nabla^2 f(x^{k+1}) d^k \approx y^k$$

motivates to improve our current estimate of $\nabla^2 f$ constructing M^+ by solving

$$\min_{M^+} \ \frac{1}{2} \left\| M^+ - M(x^k) \right\|_F^2 \quad \text{subject to} \ \ M^+ d^k = y^k$$

with $\|\cdot\|_F$ the Frobenius norms ($\|X\|_F^2 = \mathsf{Tr}(XX^\top)$).

Quasi-Newton Methods – Broyden's Updates

- Broyden's update

$$M^+ = M(x^k) - \frac{(M(x^k)d^k - y^k)(d^k)^\top}{\|d^k\|_2^2}$$

- Inverse Broyden's update

$$(M^+)^{-1} = M(x^k)^{-1} + \frac{(d^k - M(x^k)^{-1}y^k)(d^k)^\top M(x^k)^{-1}}{(d^k)^\top M(x^k)^{-1}y^k}$$

Remarks:

- both update are rank-1 update.
- we don't need to compute any second order derivatives.
- we can directly compute M^{-1}, no inversion needed.

But: M^+ may be non-symmetric even if $M(x^k)$ was symmetric.

Quasi-Newton Methods – Broyden's Updates

- Broyden's update

$$M^+ = M(x^k) - \frac{(M(x^k)d^k - y^k)(d^k)^\top}{\|d^k\|_2^2}$$

- Inverse Broyden's update

$$(M^+)^{-1} = M(x^k)^{-1} + \frac{(d^k - M(x^k)^{-1}y^k)(d^k)^\top M(x^k)^{-1}}{(d^k)^\top M(x^k)^{-1}y^k}$$

Remarks:

- both update are rank-1 update.
- we don't need to compute any second order derivatives.
- we can directly compute M^{-1}, no inversion needed.

But: M^+ may be non-symmetric even if $M(x^k)$ was symmetric.

Quasi-Newton Methods – Broyden's Updates

- Broyden's update

$$M^+ = M(x^k) - \frac{(M(x^k)d^k - y^k)(d^k)^\top}{\|d^k\|_2^2}$$

- Inverse Broyden's update

$$(M^+)^{-1} = M(x^k)^{-1} + \frac{(d^k - M(x^k)^{-1}y^k)(d^k)^\top M(x^k)^{-1}}{(d^k)^\top M(x^k)^{-1}y^k}$$

Remarks:

- both update are rank-1 update.
- we don't need to compute any second order derivatives.
- we can directly compute M^{-1}, no inversion needed.

But: M^+ may be non-symmetric even if $M(x^k)$ was symmetric.

Quasi-Newton Methods – Broyden's Updates

- Broyden's update

$$M^+ = M(x^k) - \frac{(M(x^k)d^k - y^k)(d^k)^\top}{\|d^k\|_2^2}$$

- Inverse Broyden's update

$$(M^+)^{-1} = M(x^k)^{-1} + \frac{(d^k - M(x^k)^{-1}y^k)(d^k)^\top M(x^k)^{-1}}{(d^k)^\top M(x^k)^{-1}y^k}$$

Remarks:

- both update are rank-1 update.
- we don't need to compute any second order derivatives.
- we can directly compute M^{-1}, no inversion needed.

But: M^+ may be non-symmetric even if $M(x^k)$ was symmetric.

Quasi-Newton Methods – Broyden's Updates

- Broyden's update

$$M^+ = M(x^k) - \frac{(M(x^k)d^k - y^k)(d^k)^\top}{\|d^k\|_2^2}$$

- Inverse Broyden's update

$$(M^+)^{-1} = M(x^k)^{-1} + \frac{(d^k - M(x^k)^{-1}y^k)(d^k)^\top M(x^k)^{-1}}{(d^k)^\top M(x^k)^{-1}y^k}$$

Remarks:

- both update are rank-1 update.
- we don't need to compute any second order derivatives.
- we can directly compute M^{-1}, no inversion needed.

But: M^+ may be non-symmetric even if $M(x^k)$ was symmetric.

Quasi-Newton Methods – BFGS Updates

Broyden-Fletcher-Goldfarb-Shanno Updates

The idea is to maintain the symmetry of the updates by solving

$$\min_{M^+} \ \frac{1}{2}\left\|M^+ - M(x^k)\right\|^2 \quad \text{subject to} \begin{cases} M^+ d^k = y^k \\ (M^+)^\top d^k = y^k \end{cases}$$

with $\left\|M^+ - M(x^k)\right\|^2 := \left\|W^{\frac{1}{2}}(M^+ - M(x^k))W^{\frac{1}{2}}\right\|_{\mathrm{F}}^2 =$

$$\mathsf{Tr}\left(W^{\frac{1}{2}}(M^+ - M(x^k))W(M^+ - M(x^k))W^{\frac{1}{2}}\right).$$

Here, the norm is (mainly for computational reasons) a weighted Frobenius norm by W any symmetric positive definite weighting matrix satisfying

$$Wy^k = d^k.$$

Quasi-Newton Methods – BFGS Updates

Broyden-Fletcher-Goldfarb-Shanno Updates

The idea is to maintain the symmetry of the updates by solving

$$\min_{M^+} \ \frac{1}{2}\left\|M^+ - M(x^k)\right\|^2 \quad \text{subject to} \begin{cases} M^+ d^k = y^k \\ (M^+)^\top d^k = y^k \end{cases}$$

with $\left\|M^+ - M(x^k)\right\|^2 := \left\|W^{\frac{1}{2}}(M^+ - M(x^k))W^{\frac{1}{2}}\right\|_{\mathrm{F}}^2 =$

$$\mathsf{Tr}\left(W^{\frac{1}{2}}(M^+ - M(x^k))W(M^+ - M(x^k))W^{\frac{1}{2}}\right).$$

Here, the norm is (mainly for computational reasons) a weighted Frobenius norm by W any symmetric positive definite weighting matrix satisfying

$$Wy^k = d^k.$$

Quasi-Newton Methods – BFGS Updates

Broyden-Fletcher-Goldfarb-Shanno Updates

The idea is to maintain the symmetry of the updates by solving

$$\min_{M^+} \quad \frac{1}{2}\left\|M^+ - M(x^k)\right\|^2 \quad \text{subject to} \begin{cases} M^+ d^k = y^k \\ (M^+)^\top d^k = y^k \end{cases}$$

with $\left\|M^+ - M(x^k)\right\|^2 := \left\|W^{\frac{1}{2}}(M^+ - M(x^k))W^{\frac{1}{2}}\right\|_{\mathrm{F}}^2 =$

$$\mathsf{Tr}\left(W^{\frac{1}{2}}(M^+ - M(x^k))W(M^+ - M(x^k))W^{\frac{1}{2}}\right).$$

Here, the norm is (mainly for computational reasons) a weighted Frobenius norm by W any symmetric positive definite weighting matrix satisfying

$$Wy^k = d^k.$$

Quasi-Newton Methods – BFGS Updates

Broyden-Fletcher-Goldfarb-Shanno Updates

- BFGS update:

$$M^{+} = M(x^k) - \frac{M(x^k)d^k(d^k)^\top M(x^k)}{(d^k)^\top M(x^k)d^k} + \frac{y^k(y^k)^\top}{(y^k)^\top d^k}$$

- inverse BFGS update:

$$(M^{+})^{-1} = \left(\mathrm{I} - \frac{d^k(y^k)^\top}{(d^k)^\top y^k}\right) M(x^k)^{-1} \left(\mathrm{I} - \frac{d^k(y^k)^\top}{(d^k)^\top y^k}\right) + \frac{d^k(d^k)^\top}{(d^k)^\top y^k}$$

- Both are rank-2 update.

Quasi-Newton Methods – BFGS Updates

Broyden-Fletcher-Goldfarb-Shanno Updates

- BFGS update:

$$M^{+} = M(x^k) - \frac{M(x^k)d^k(d^k)^\top M(x^k)}{(d^k)^\top M(x^k)d^k} + \frac{y^k(y^k)^\top}{(y^k)^\top d^k}$$

- inverse BFGS update:

$$(M^{+})^{-1} = \left(\mathrm{I} - \frac{d^k(y^k)^\top}{(d^k)^\top y^k}\right) M(x^k)^{-1} \left(\mathrm{I} - \frac{d^k(y^k)^\top}{(d^k)^\top y^k}\right) + \frac{d^k(d^k)^\top}{(d^k)^\top y^k}$$

- Both are rank-2 update.

Quasi-Newton Methods – BFGS Updates

Broyden-Fletcher-Goldfarb-Shanno Updates

- BFGS update:

$$M^{+} = M(x^k) - \frac{M(x^k)d^k(d^k)^\top M(x^k)}{(d^k)^\top M(x^k)d^k} + \frac{y^k(y^k)^\top}{(y^k)^\top d^k}$$

- inverse BFGS update:

$$(M^{+})^{-1} = \left(\mathrm{I} - \frac{d^k(y^k)^\top}{(d^k)^\top y^k}\right) M(x^k)^{-1} \left(\mathrm{I} - \frac{d^k(y^k)^\top}{(d^k)^\top y^k}\right) + \frac{d^k(d^k)^\top}{(d^k)^\top y^k}$$

- Both are rank-2 update.

Algorithms for Constrained NLPs

Nonlinear program

$$\min_{x \in \mathbb{R}^{n_x}} \quad f(x) \quad \text{subject to} \begin{cases} h_i(x) = 0, \ i \in \mathcal{E} := \{1, ..., n_h\} \\ g_i(x) \leq 0, \ i \in \mathcal{I} := \{1, ..., n_g\} \end{cases}$$

Convert into unconstrained problem:

- Penalty function method;
- Interior point method.

Solve necessary conditions of optimality:

- Newton-like methods;
- Sequential quadratic programming.

Algorithms for Constrained NLPs

Nonlinear program

$$\min_{x \in \mathbb{R}^{n_x}} \quad f(x) \quad \text{subject to} \begin{cases} h_i(x) = 0, \ i \in \mathcal{E} := \{1, ..., n_h\} \\ g_i(x) \leq 0, \ i \in \mathcal{I} := \{1, ..., n_g\} \end{cases}$$

Convert into unconstrained problem:

- Penalty function method;
- Interior point method.

Solve necessary conditions of optimality:

- Newton-like methods;
- Sequential quadratic programming.

Exterior Penalty Function Methods

Penalty function

$$\Phi(x) = \sum_{i \in \mathcal{E}} \psi(h_i(x)) + \sum_{i \in \mathcal{I}} \phi(g_i(x)),\ \psi, \phi \in \mathbb{C}^0$$

$$\text{with } \begin{cases} \psi(z) = 0 & \text{if } z = 0 \\ \psi(z) > 0 & \text{else} \end{cases} \text{ and } \begin{cases} \phi(z) = 0 & \text{if } z \leq 0 \\ \phi(z) > 0 & \text{else} \end{cases}$$

Typical choice: $\psi(z) = |z|^p$, $p \in \mathbb{N}_{>0}$ and $\phi(z) = (\max\{0, z\})^p$.

Exterior Penalty Function Methods

Penalty function

$$\Phi(x) = \sum_{i \in \mathcal{E}} \psi(h_i(x)) + \sum_{i \in \mathcal{I}} \phi(g_i(x)),\ \psi, \phi \in \mathbb{C}^0$$

$$\text{with } \begin{cases} \psi(z) = 0 & \text{if } z = 0 \\ \psi(z) > 0 & \text{else} \end{cases} \text{ and } \begin{cases} \phi(z) = 0 & \text{if } z \leq 0 \\ \phi(z) > 0 & \text{else} \end{cases}$$

Typical choice: $\psi(z) = |z|^p$, $p \in \mathbb{N}_{>0}$ and $\phi(z) = (\max\{0, z\})^p$.

Exterior Penalty Function Methods

Unconstrained optimization problem

$$\min_{x \in \mathbb{R}^{n_x}} \quad f(x) + \mu \cdot \Phi(x) \;\; \text{with} \;\; \mu > 0.$$

Remark

- *recovering solution of the original problem $\mu \to \infty$.*
- *ill-conditioned for large μ.*

Exterior Penalty Function Methods

Unconstrained optimization problem

$$\min_{x \in \mathbb{R}^{n_x}} \quad f(x) + \mu \cdot \Phi(x) \ \text{ with } \ \mu > 0.$$

Remark

- *recovering solution of the original problem* $\mu \to \infty$.
- *ill-conditioned for large* μ.

Sequential Unconstrained Optimization

Main idea:

- Start at an initial x^0, update x^k by solving

$$x^{k+1} := \text{arg} \min_{x \in \mathbb{R}^{n_x}} \quad f(x) + \mu^k \cdot \Phi(x).$$

- If $\mu^k \Phi(x^{k+1}) < \epsilon$, stop. Otherwise, update $\mu^{k+1} = \beta \mu^k$ with $\beta > 1$.

Remark

- *Iterates x^k are typically infeasible.*
- *Remedy? $\rightarrow$ interior point methods.*

Sequential Unconstrained Optimization

Main idea:

- Start at an initial x^0, update x^k by solving

$$x^{k+1} := \text{arg} \min_{x \in \mathbb{R}^{n_x}} \quad f(x) + \mu^k \cdot \Phi(x).$$

- If $\mu^k \Phi(x^{k+1}) < \epsilon$, stop. Otherwise, update $\mu^{k+1} = \beta \mu^k$ with $\beta > 1$.

Remark

- *Iterates x^k are typically infeasible.*
- *Remedy? $\rightarrow$ interior point methods.*

Barrier Method

Inequality constrained NLPs

$$\min_{x \in \mathbb{R}^{n_x}} \quad f(x) \quad \text{subject to} \ \ g_i(x) \leq 0, \ i \in \mathcal{I} := \{1, ..., n_g\}$$

Barrier function

$$b(x) = \sum_{i \in \mathcal{I}} \phi(g_i(x)) \ \text{ with} \begin{cases} \phi(z) \geq 0 \ \text{ if } z \leq 0 \\ \lim_{z \to 0^-} = \infty \end{cases}$$

Typical choice

$$\phi(z) = -\ln(-z).$$

Barrier Method

Inequality constrained NLPs

$$\min_{x \in \mathbb{R}^{n_x}} \quad f(x) \quad \text{subject to} \;\; g_i(x) \leq 0,\; i \in \mathcal{I} := \{1, ..., n_g\}$$

Barrier function

$$b(x) = \sum_{i \in \mathcal{I}} \phi(g_i(x)) \;\; \text{with} \begin{cases} \phi(z) \geq 0 \;\; \text{if } z \leq 0 \\ \lim_{z \to 0^-} = \infty \end{cases}$$

Typical choice

$$\phi(z) = -\ln(-z).$$

Barrier Method

Inequality constrained NLPs

$$\min_{x \in \mathbb{R}^{n_x}} \quad f(x) \quad \text{subject to } \; g_i(x) \le 0,\; i \in \mathcal{I} := \{1, ..., n_g\}$$

Barrier function

$$b(x) = \sum_{i \in \mathcal{I}} \phi(g_i(x)) \;\; \text{with} \begin{cases} \phi(z) \ge 0 \;\; \text{if } z \le 0 \\ \lim_{z \to 0^-} = \infty \end{cases}$$

Typical choice

$$\phi(z) = -\ln(-z).$$

Barrier Method

Main idea:

- Start at an initial feasible point x^0 with $g(x^0) < 0$, update x^k by solving

$$x^{k+1} := \text{arg} \min_{x \in \mathbb{R}^{n_x}} \quad f(x) + \mu^k \cdot b(x).$$

- If $\mu^k \cdot b(x^{k+1}) < \epsilon$, stop. Otherwise, update $\mu^{k+1} = \beta \mu^k$ with $\beta \in (0, 1)$.

Remark

- *Iterates x^k are always feasible.*
- *Off-the-shelf solver `Ipopt`.*

Barrier Method

Main idea:

- Start at an initial feasible point x^0 with $g(x^0) < 0$, update x^k by solving

$$x^{k+1} := \arg\min_{x \in \mathbb{R}^{n_x}} \quad f(x) + \mu^k \cdot b(x).$$

- If $\mu^k \cdot b(x^{k+1}) < \epsilon$, stop. Otherwise, update $\mu^{k+1} = \beta\mu^k$ with $\beta \in (0, 1)$.

Remark

- *Iterates x^k are always feasible.*
- *Off-the-shelf solver `Ipopt`.*

Interior Point Method

Inequality constrained NLPs

$$\min_{x \in \mathbb{R}^{n_x}} \quad f(x) \quad \text{subject to} \;\; g_i(x) \leq 0, \; i \in \mathcal{I} := \{1, ..., n_g\}$$

KKT condition:

$$\begin{aligned} \nabla f(x) + \nabla g(x)\kappa &= 0 \\ g(x) &\leq 0 \\ \kappa &\geq 0 \\ \kappa_i \cdot g_i(x) &= 0, \; i \in \mathcal{I} \end{aligned}$$

Perturbed KKT condition:

$$\begin{aligned} \nabla f(x) + \nabla g(x)\kappa &= 0 \\ \kappa_i \cdot g_i(x) &= \mu, \; i \in \mathcal{I} \end{aligned}$$

with $\mu > 0$.

Interior Point Method

Inequality constrained NLPs

$$\min_{x \in \mathbb{R}^{n_x}} \quad f(x) \quad \text{subject to} \;\; g_i(x) \leq 0, \; i \in \mathcal{I} := \{1, ..., n_g\}$$

KKT condition:

$$\begin{aligned} \nabla f(x) + \nabla g(x)\kappa &= 0 \\ g(x) &\leq 0 \\ \kappa &\geq 0 \\ \kappa_i \cdot g_i(x) &= 0, \; i \in \mathcal{I} \end{aligned}$$

Perturbed KKT condition:

$$\begin{aligned} \nabla f(x) + \nabla g(x)\kappa &= 0 \\ \kappa_i \cdot g_i(x) &= \mu, \; i \in \mathcal{I} \end{aligned}$$

with $\mu > 0$.

Interior Point Method

Main Idea:

- Apply Newton's method to deal with nonlinear equations

$$F_\mu(x,\kappa) = \begin{bmatrix} \nabla f(x) + \nabla g(x)\kappa \\ \mathrm{diag}(\kappa) g(x) - \mu \cdot \mathbf{1}_{n_g} \end{bmatrix} = 0$$

- Update μ with $\mu \to 0$, ref. [Chapter 19.3, NW06]
- Linear search is necessary, ref. [Chapter 19.4, NW06]

Remark

Log-barrier based unconstrained problem $\min_x \quad f(x) + \mu \cdot b(x)$ *has KKT conditions equivalent to the perturbed KKT, i.e.,*

$$\nabla f(x) + \sum_{i \in \mathcal{I}} \frac{\mu}{g_i(x)} \nabla g_i(x) = 0 \;\Rightarrow\; \kappa_i = \frac{\mu}{g_i(x)}$$

[NW06] J. Nocedal and S. Wright, Numerical optimization. 2006.

Interior Point Method

Main Idea:

- Apply Newton's method to deal with nonlinear equations

$$F_\mu(x,\kappa) = \begin{bmatrix} \nabla f(x) + \nabla g(x)\kappa \\ \mathsf{diag}(\kappa) g(x) - \mu \cdot \mathbf{1}_{n_g} \end{bmatrix} = 0$$

- Update μ with $\mu \to 0$, ref. [Chapter 19.3, NW06]
- Linear search is necessary, ref. [Chapter 19.4, NW06]

Remark

Log-barrier based unconstrained problem $\min_x \;\; f(x) + \mu \cdot b(x)$ *has KKT conditions equivalent to the perturbed KKT, i.e.,*

$$\nabla f(x) + \sum_{i \in \mathcal{I}} \frac{\mu}{g_i(x)} \nabla g_i(x) = 0 \;\Rightarrow\; \kappa_i = \frac{\mu}{g_i(x)}$$

[NW06] J. Nocedal and S. Wright, Numerical optimization. 2006.

Sequential Quadratic Programming (SQP)

Equality constrained NLP

$$\min_{x\in\mathbb{R}^{n_x}} \quad f(x) \quad \text{subject to} \;\; h(x) = 0$$

1st order optimality conditions

$$F(y) = \begin{bmatrix} \nabla f(x) + \nabla h(x)\lambda \\ h(x) \end{bmatrix} = 0 \;\; \text{with} \;\; y = \begin{pmatrix} x \\ \lambda \end{pmatrix}$$

Main idea: applying Newton's method to solve $F(y) = 0$, i.e.,

$$\begin{bmatrix} H(x) & A(x)^\top \\ A(x) & \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta \lambda \end{bmatrix} = - \begin{bmatrix} \nabla f(x) + \nabla h(x)\lambda \\ h(x) \end{bmatrix}$$

with $H(x) = \nabla_{xx} \left\{ f(x) + \lambda^\top h(x) \right\}$ and $A = \nabla h(x)^\top$.

Sequential Quadratic Programming (SQP)

Equality constrained NLP

$$\min_{x \in \mathbb{R}^{n_x}} \quad f(x) \quad \text{subject to} \;\; h(x) = 0$$

1st order optimality conditions

$$F(y) = \begin{bmatrix} \nabla f(x) + \nabla h(x)\lambda \\ h(x) \end{bmatrix} = 0 \;\; \text{with} \;\; y = \begin{pmatrix} x \\ \lambda \end{pmatrix}$$

Main idea: applying Newton's method to solve $F(y) = 0$, i.e.,

$$\begin{bmatrix} H(x) & A(x)^\top \\ A(x) & \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta \lambda \end{bmatrix} = - \begin{bmatrix} \nabla f(x) + \nabla h(x)\lambda \\ h(x) \end{bmatrix}$$

with $H(x) = \nabla_{xx} \left\{ f(x) + \lambda^\top h(x) \right\}$ and $A = \nabla h(x)^\top$.

Sequential Quadratic Programming (SQP)

Equality constrained NLP

$$\min_{x \in \mathbb{R}^{n_x}} \quad f(x) \quad \text{subject to} \;\; h(x) = 0$$

1st order optimality conditions

$$F(y) = \begin{bmatrix} \nabla f(x) + \nabla h(x) \lambda \\ h(x) \end{bmatrix} = 0 \;\; \text{with} \;\; y = \begin{pmatrix} x \\ \lambda \end{pmatrix}$$

Main idea: applying Newton's method to solve $F(y) = 0$, i.e.,

$$\begin{bmatrix} H(x) & A(x)^\top \\ A(x) & \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta \lambda \end{bmatrix} = - \begin{bmatrix} \nabla f(x) + \nabla h(x) \lambda \\ h(x) \end{bmatrix}$$

with $H(x) = \nabla_{xx} \left\{ f(x) + \lambda^\top h(x) \right\}$ and $A = \nabla h(x)^\top$.

Globalization

How do we measure progress towards a solution?

Recall: in unconstrained minimization, the main idea was to accept the next iterate x^{k+1} if $f(x^{k+1})$ is sufficiently smaller than $f(x^k)$.

In equality constrained optimization we need to measure two things:

1. The objective value $f(x)$ and
2. the constraint violation $\|h(x)\|$

Globalization

How do we measure progress towards a solution?

Recall: in unconstrained minimization, the main idea was to accept the next iterate x^{k+1} if $f(x^{k+1})$ is sufficiently smaller than $f(x^k)$.

In equality constrained optimization we need to measure two things:

1. The objective value $f(x)$ and
2. the constraint violation $\|h(x)\|$

Globalization

How do we measure progress towards a solution?

Recall: in unconstrained minimization, the main idea was to accept the next iterate x^{k+1} if $f(x^{k+1})$ is sufficiently smaller than $f(x^k)$.

In equality constrained optimization we need to measure two things:

1. The objective value $f(x)$ and
2. the constraint violation $\|h(x)\|$

L1-Penalty Functions

One way to measure progress towards a solution is to introduce the L1-penalty function

$$\Psi(x) = f(x) + \sum_{i \in \mathcal{E}} \bar{\lambda}_i |h_i(x)|$$

with $\bar{\lambda}_i$ being sufficiently large constants.

An important property of the function $\Psi(x)$ is that (under mild conditions) we have

$$\Psi(x^*) = f(x^*) \quad \text{but also} \quad \Psi(x) \geq f(x)$$

for all $x \in \mathbb{X}$ within a compact subset $\mathbb{X} \subseteq \mathbb{R}^{n_x}$ and $\bar{\lambda}_i$ are sufficiently large

L1-Penalty Functions

One way to measure progress towards a solution is to introduce the L1-penalty function

$$\Psi(x) = f(x) + \sum_{i \in \mathcal{E}} \bar{\lambda}_i |h_i(x)|$$

with $\bar{\lambda}_i$ being sufficiently large constants.

An important property of the function $\Psi(x)$ is that (under mild conditions) we have

$$\Psi(x^*) = f(x^*) \quad \text{but also} \quad \Psi(x) \geq f(x)$$

for all $x \in \mathbb{X}$ within a compact subset $\mathbb{X} \subseteq \mathbb{R}^{n_x}$ and $\bar{\lambda}_i$ are sufficiently large

Armijo Line Search Conditions

Similar to unconstrained optimization, the line search parameter α^k can be found by using back-tracking until the Armijo condition

$$\Psi(x^k + \alpha^k \Delta x^k) \leq \Psi(x^k) + c \cdot \alpha^k D(\Psi(x^k), \Delta x^k)$$

for a constant $c \ll 1$ is satisfied. Here, $D(\Psi(x^k), \Delta x^k)$ denotes the directional derivative

$$D(\Psi(x^k), \Delta x^k) = \left\| h(x^k) + \nabla h(x^k)^\top \Delta x^k \right\|_1$$

SQP for Equality Constrained NLP

1. Choose initial guesses $x^0 \in \mathbb{R}^{n_x}$ and $\lambda^0 \in \mathbb{R}^{n_h}$, tolerance $\epsilon > 0$.
2. **Repeat:**

   2.1 Choose Hessian approximation $M(x^k) \approx \nabla_{xx} \left\{ f(x^k) + (\lambda^k)^\top h(x^k) \right\}$ and $A(x^k) = \nabla h(x^k)$.

   2.2 Solve subQP

$$\begin{aligned} \min_{\Delta x^k \in \mathbb{R}^{n_x}} \quad & \frac{1}{2}(\Delta x^k)^\top H(x^k) \Delta x^k + \nabla f(x^k)^\top \Delta x^k \\ \text{subject to} \quad & h(x^k) + A(x^k)^\top \Delta x^k = 0 \qquad \mid \lambda^{\mathrm{QP}} \end{aligned}$$

   2.3 Terminate if $\left| \nabla f(x^k)^\top \Delta x^k \right| + \sum_{i \in \mathcal{E}} |\lambda_i| \, |h_i(x)| \leq \epsilon$.

   2.4 Choose a line-search parameter $\alpha^k \in (0, 1]$ and set $x^{k+1} = x^k + \alpha^k \Delta x^k$ and $\lambda^{k+1} = \lambda^k + \alpha^k (\lambda^{\mathrm{QP}} - \lambda^k)$.

SQP for Inequality Constrained NLP

- Include linearized inequality constraints in subQPs, i.e.,

$$\begin{aligned}\min_{\Delta x^k \in \mathbb{R}^{n_x}} \quad & \frac{1}{2}(\Delta x^k)^\top H(x^k)\Delta x^k + \nabla f(x^k)^\top \Delta x^k \\ \text{subject to} \quad & h(x^k) + A(x^k)^\top \Delta x^k = 0 \\ & g(x^k) + B(x^k)^\top \Delta x^k \leq 0\end{aligned}$$

  with $B(x^k) = \nabla g(x^k)$.

- Use the following L1-penalty function for linear search

$$\Psi(x) = f(x) + \sum_{i\in\mathcal{E}} |\bar{\lambda}_i||h_i(x)| + \sum_{i\in\mathcal{I}} |\bar{\kappa}_i|(\max\{0, g_i(x)\})$$

  with sufficiently large $\bar{\lambda}_i$ and $\bar{\kappa}_i$.

Numerical Implementation

- SubQP infeasible $\Rightarrow$ relax the constraints
- Hessian regularization, e.g., $M(x^k) = \nabla_{xx}\{f(x) + \lambda^\top h(x)\} + \sigma \mathrm{I} \succ 0$.
- Rank-deficient constraints $\Rightarrow$ reformulated the constraints, e.g.,

$$\min_x \ x \quad \text{subject to} \ x^2 = 0$$

  with $x^* = 0$ but we cannot find a λ^* since

$$0 = \nabla f(x^*) + \nabla h(x^*)\lambda^* = 1$$

  is wrong. Replacing $x^2 = 0$ by $x = 0$ can avoid this degeneracy.
- Constraint Jacobian ill-conditioned $\Rightarrow$ scaling, e.g., Ruiz equilibration.

Numerical Implementation

- SubQP infeasible $\Rightarrow$ relax the constraints
- Hessian regularization, e.g., $M(x^k) = \nabla_{xx}\{f(x) + \lambda^\top h(x)\} + \sigma \mathrm{I} \succ 0$.
- Rank-deficient constraints $\Rightarrow$ reformulated the constraints, e.g.,

$$\min_x \; x \quad \text{subject to} \; x^2 = 0$$

  with $x^* = 0$ but we cannot find a λ^* since

$$0 = \nabla f(x^*) + \nabla h(x^*)\lambda^* = 1$$

  is wrong. Replacing $x^2 = 0$ by $x = 0$ can avoid this degeneracy.
- Constraint Jacobian ill-conditioned $\Rightarrow$ scaling, e.g., Ruiz equilibration.

Numerical Implementation

- SubQP infeasible $\Rightarrow$ relax the constraints
- Hessian regularization, e.g., $M(x^k) = \nabla_{xx}\{f(x) + \lambda^\top h(x)\} + \sigma \mathrm{I} \succ 0$.
- Rank-deficient constraints $\Rightarrow$ reformulated the constraints, e.g.,

$$\min_x \ x \quad \text{subject to} \ x^2 = 0$$

with $x^* = 0$ but we cannot find a λ^* since

$$0 = \nabla f(x^*) + \nabla h(x^*)\lambda^* = 1$$

is wrong. Replacing $x^2 = 0$ by $x = 0$ can avoid this degeneracy.

- Constraint Jacobian ill-conditioned $\Rightarrow$ scaling, e.g., Ruiz equilibration.

Numerical Implementation

- SubQP infeasible $\Rightarrow$ relax the constraints
- Hessian regularization, e.g., $M(x^k) = \nabla_{xx}\{f(x) + \lambda^\top h(x)\} + \sigma \mathrm{I} \succ 0$.
- Rank-deficient constraints $\Rightarrow$ reformulated the constraints, e.g.,

$$\min_x \quad x \quad \text{subject to} \quad x^2 = 0$$

  with $x^* = 0$ but we cannot find a λ^* since

$$0 = \nabla f(x^*) + \nabla h(x^*)\lambda^* = 1$$

  is wrong. Replacing $x^2 = 0$ by $x = 0$ can avoid this degeneracy.
- Constraint Jacobian ill-conditioned $\Rightarrow$ scaling, e.g., Ruiz equilibration.

Contents

Why do we need to compute derivatives?

Motivation

$$\min_{x \in \mathbb{R}^{n_x}} \quad f(x) \quad \text{subject to} \begin{cases} h(x) = 0 \\ g(x) \leq 0 \end{cases}$$

- Derivatives of objectives and constraints (gradients);
- Sensitivities of ODE or DAEs (needed later);
- Jacobians to solve implicit equations (e.g. implicit ODEs, DAEs, ...).

Main Possibilities

- Numerical differentiation
- Algorithmic differentiation

Why do we need to compute derivatives?

Motivation

$$\min_{x \in \mathbb{R}^{n_x}} \quad f(x) \quad \text{subject to} \begin{cases} h(x) = 0 \\ g(x) \leq 0 \end{cases}$$

- Derivatives of objectives and constraints (gradients);
- Sensitivities of ODE or DAEs (needed later);
- Jacobians to solve implicit equations (e.g. implicit ODEs, DAEs, ...).

Main Possibilities

- Numerical differentiation
- Algorithmic differentiation

Numerical Differentiation – Finite Differences

The derivative of a twice continuously differentiable function $f : \mathbb{R} \to \mathbb{R}$ can be approximated by finite differences:

$$\frac{df(x)}{dx} \approx \frac{f(x+h) - f(x)}{h}$$

- The mathematical approximation error, given by

$$\left| \frac{f(x+h) - f(x)}{h} - \frac{df(x)}{dx} \right| \approx \frac{h}{2} \left| \frac{d^2 f(x)}{dx^2} \right| = \mathbf{O}(h)$$

  tends to 0 with $h \to 0$.

- How to choose increment h?

$$h = \sqrt{\text{eps}} \quad \Rightarrow \quad \textbf{Limited accuracy} \sqrt{\text{eps}}$$

Numerical Differentiation – Finite Differences

The derivative of a twice continuously differentiable function $f : \mathbb{R} \to \mathbb{R}$ can be approximated by finite differences:

$$\frac{df(x)}{dx} \approx \frac{f(x+h) - f(x)}{h}$$

- The mathematical approximation error, given by

$$\left| \frac{f(x+h) - f(x)}{h} - \frac{df(x)}{dx} \right| \approx \frac{h}{2} \left| \frac{d^2 f(x)}{dx^2} \right| = \mathbf{O}(h)$$

  tends to 0 with $h \to 0$.

- How to choose increment h?

$$h = \sqrt{\text{eps}} \quad \Rightarrow \quad \textbf{Limited accuracy} \sqrt{\text{eps}}$$

Numerical Differentiation – Finite Differences

The derivative of a twice continuously differentiable function $f : \mathbb{R} \rightarrow \mathbb{R}$ can be approximated by finite differences:

$$\frac{df(x)}{dx} \approx \frac{f(x+h) - f(x)}{h}$$

- The mathematical approximation error, given by

$$\left| \frac{f(x+h) - f(x)}{h} - \frac{df(x)}{dx} \right| \approx \frac{h}{2} \left| \frac{d^2 f(x)}{dx^2} \right| = \mathbf{O}(h)$$

  tends to 0 with $h \rightarrow 0$.

- How to choose increment h?

$$h = \sqrt{\mathrm{eps}} \quad \Rightarrow \quad \textbf{Limited accuracy} \sqrt{\mathrm{eps}}$$

Numerical Differentiation – Central Differences

In order to reduce the mathematical approximation error, we can use central differences:

$$\frac{df(x)}{dx} \approx \frac{f(x+h) - f(x-h)}{2h}$$

to approximate the derivative of f.

- The mathematical approximation error is now

$$\left| \frac{f(x+h) - f(x-h)}{2h} - \frac{df(x)}{dx} \right| \leq \mathbf{O}(h^2)$$

- How to choose increment h?

$$h = \sqrt[3]{\mathrm{eps}} \quad \Rightarrow \quad \textbf{Limited accuracy}(\sqrt[3]{\mathrm{eps}})^2$$

Numerical Differentiation – Central Differences

In order to reduce the mathematical approximation error, we can use central differences:

$$\frac{df(x)}{dx} \approx \frac{f(x+h) - f(x-h)}{2h}$$

to approximate the derivative of f.

- The mathematical approximation error is now

$$\left| \frac{f(x+h) - f(x-h)}{2h} - \frac{df(x)}{dx} \right| \leq \mathbf{O}(h^2)$$

- How to choose increment h?

$$h = \sqrt[3]{\mathrm{eps}} \quad \Rightarrow \quad \textbf{Limited accuracy}(\sqrt[3]{\mathrm{eps}})^2$$

Numerical Differentiation – Central Differences

In order to reduce the mathematical approximation error, we can use central differences:

$$\frac{df(x)}{dx} \approx \frac{f(x+h) - f(x-h)}{2h}$$

to approximate the derivative of f.

- The mathematical approximation error is now

$$\left| \frac{f(x+h) - f(x-h)}{2h} - \frac{df(x)}{dx} \right| \leq \mathbf{O}(h^2)$$

- How to choose increment h?

$$h = \sqrt[3]{\text{eps}} \quad \Rightarrow \quad \textbf{Limited accuracy}(\sqrt[3]{\text{eps}})^2$$

Numerical Differentiation – Imaginary Trick

The derivative of a continuously differentiable function $f : \mathbb{R} \to \mathbb{R}$ can be approximated by

$$\frac{df(x)}{dx} \approx \frac{\mathrm{Im}(f(x + i \cdot h))}{h}, \; i^2 = -1.$$

- The mathematical approximation error is same as central differences, i.e.,

$$\left| \frac{\mathrm{Im}(f(x + i \cdot h))}{h} - \frac{df(x)}{dx} \right| \leq \mathbf{O}(h^2)$$

  but the computation is cheap.

- Sketch Proof:

$$f(x + i \cdot h) = f(x) + i \cdot \frac{df(x)}{dx} h - \frac{1}{2} \frac{d^2 f(x)}{dx^2} h^2 - \mathbf{O}(i \cdot h^3)$$

- Easy to implement in `Matlab`

Numerical Differentiation – Imaginary Trick

The derivative of a continuously differentiable function $f : \mathbb{R} \to \mathbb{R}$ can be approximated by

$$\frac{df(x)}{dx} \approx \frac{\mathrm{Im}(f(x + i \cdot h))}{h},\ i^2 = -1.$$

- The mathematical approximation error is same as central differences, i.e.,

$$\left| \frac{\mathrm{Im}(f(x + i \cdot h))}{h} - \frac{df(x)}{dx} \right| \leq \mathbf{O}(h^2)$$

  but the computation is cheap.

- Sketch Proof:

$$f(x + i \cdot h) = f(x) + i \cdot \frac{df(x)}{dx} h - \frac{1}{2} \frac{d^2 f(x)}{dx^2} h^2 - \mathbf{O}(i \cdot h^3)$$

- Easy to implement in `Matlab`

Numerical Differentiation – Imaginary Trick

The derivative of a continuously differentiable function $f : \mathbb{R} \to \mathbb{R}$ can be approximated by

$$\frac{df(x)}{dx} \approx \frac{\mathrm{Im}(f(x + i \cdot h))}{h}, \ i^2 = -1.$$

- The mathematical approximation error is same as central differences, i.e.,

$$\left| \frac{\mathrm{Im}(f(x + i \cdot h))}{h} - \frac{df(x)}{dx} \right| \leq \mathbf{O}(h^2)$$

  but the computation is cheap.

- Sketch Proof:

$$f(x + i \cdot h) = f(x) + i \cdot \frac{df(x)}{dx} h - \frac{1}{2} \frac{d^2 f(x)}{dx^2} h^2 - \mathbf{O}(i \cdot h^3)$$

- Easy to implement in `Matlab`

Numerical Differentiation – Imaginary Trick

The derivative of a continuously differentiable function $f : \mathbb{R} \to \mathbb{R}$ can be approximated by

$$\frac{df(x)}{dx} \approx \frac{\mathrm{Im}(f(x + i \cdot h))}{h}, \ i^2 = -1.$$

- The mathematical approximation error is same as central differences, i.e.,

  $$\left| \frac{\mathrm{Im}(f(x + i \cdot h))}{h} - \frac{df(x)}{dx} \right| \leq \mathbf{O}(h^2)$$

  but the computation is cheap.

- Sketch Proof:

  $$f(x + i \cdot h) = f(x) + i \cdot \frac{df(x)}{dx} h - \frac{1}{2} \frac{d^2 f(x)}{dx^2} h^2 - \mathbf{O}(i \cdot h^3)$$

- Easy to implement in `Matlab`

Factorable Functions

Many (but not all) functions of our interest can be composed into a finite list of atom operations from a given library L, e.g., $L = \{+, -, *, \sin, \cos, \exp, ...\}$.

Example

- The function $f(x) = \sin(x_1 * x_2) + \cos(x_1)$ will (internally) be evaluated as

$$\begin{aligned} x_3 &= x_1 * x_2 \\ x_4 &= \sin(x_3) \\ x_5 &= \cos(x_1) \\ x_6 &= x_4 + x_5 \\ f(x) &= x_6 \end{aligned}$$

  Here, the memory for $x_3, ..., x_5$ is (usually) allocated temporarily.

Factorable Functions

Many (but not all) functions of our interest can be composed into a finite list of atom operations from a given library L, e.g., $L = \{+, -, *, \sin, \cos, \exp, ...\}$.

Example

- The function $f(x) = \sin(x_1 * x_2) + \cos(x_1)$ will (internally) be evaluated as

$$\begin{aligned} x_3 &= x_1 * x_2 \\ x_4 &= \sin(x_3) \\ x_5 &= \cos(x_1) \\ x_6 &= x_4 + x_5 \\ f(x) &= x_6 \end{aligned}$$

  Here, the memory for $x_3, ..., x_5$ is (usually) allocated temporarily.

Factorable Functions

Consider a given factorable function $f : \mathbb{R}^n \to \mathbb{R}^m$, we define

- augmented state by

$$s_0 = x = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}, \; s_1 = \begin{bmatrix} x_1 \\ \vdots \\ x_{n+1} \end{bmatrix}, ..., s_m = \begin{bmatrix} x_1 \\ \vdots \\ x_{n+m} \end{bmatrix}$$

- augmented elementary function by $\Phi_i : \mathbb{R}^{n+i} \to \mathbb{R}^{n+i+1}$

$$\Phi_i(s_i) = \begin{bmatrix} x_1 \\ \vdots \\ x_{n+i} \\ \phi_i(x_1, .., x_{n+i}) \end{bmatrix}, \; s_{i+1} = \Phi_i(s_i)$$

- Representation of f given by $f(x) = C \cdot \Phi_{m-1}(\Phi_{m-2}(\cdots \Phi_1(\Phi_0(x))))$ with selection matrix $C = [\mathbf{0}_{m \times n}, \mathrm{I}_m]$

Factorable Functions

Consider a given factorable function $f : \mathbb{R}^n \to \mathbb{R}^m$, we define

- augmented state by

$$s_0 = x = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}, \ s_1 = \begin{bmatrix} x_1 \\ \vdots \\ x_{n+1} \end{bmatrix}, ..., s_m = \begin{bmatrix} x_1 \\ \vdots \\ x_{n+m} \end{bmatrix}$$

- augmented elementary function by $\Phi_i : \mathbb{R}^{n+i} \to \mathbb{R}^{n+i+1}$

$$\Phi_i(s_i) = \begin{bmatrix} x_1 \\ \vdots \\ x_{n+i} \\ \phi_i(x_1, .., x_{n+i}) \end{bmatrix}, \ s_{i+1} = \Phi_i(s_i)$$

- Representation of f given by $f(x) = C \cdot \Phi_{m-1}(\Phi_{m-2}(\cdots \Phi_1(\Phi_0(x))))$ with selection matrix $C = [\mathbf{0}_{m \times n}, \mathrm{I}_m]$

Factorable Functions

Consider a given factorable function $f : \mathbb{R}^n \rightarrow \mathbb{R}^m$, we define

- augmented state by

$$s_0 = x = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}, \; s_1 = \begin{bmatrix} x_1 \\ \vdots \\ x_{n+1} \end{bmatrix}, ..., s_m = \begin{bmatrix} x_1 \\ \vdots \\ x_{n+m} \end{bmatrix}$$

- augmented elementary function by $\Phi_i : \mathbb{R}^{n+i} \rightarrow \mathbb{R}^{n+i+1}$

$$\Phi_i(s_i) = \begin{bmatrix} x_1 \\ \vdots \\ x_{n+i} \\ \phi_i(x_1, .., x_{n+i}) \end{bmatrix}, \; s_{i+1} = \Phi_i(s_i)$$

- Representation of f given by $f(x) = C \cdot \Phi_{m-1}(\Phi_{m-2}(\cdots \Phi_1(\Phi_0(x))))$

  with selection matrix $C = [\mathbf{0}_{m \times n}, \mathrm{I}_m]$

Algorithmic Differentiation – Forward Mode

- Recall: the representation of a factorable function $f : \mathbb{R}^n \to \mathbb{R}^m$,

$$f(x) = C \cdot \Phi_{m-1}(\Phi_{m-2}(\cdots \Phi_1(\Phi_0(x))))$$

The Jacobian $J_f = \frac{\partial f}{\partial x}$ can be written as

$$J_f = C \cdot J_{m-1} \cdot J_{m-2} \cdots J_1 \cdot J_0 \quad \text{with } J_i = \frac{\partial \Phi_i}{\partial s_i}$$

- Main idea: the directional derivative $J_f p$ with seed $p \in \mathbb{R}^n$ given by

$$J_f p = C \cdot (J_{m-1} \cdot (J_{m-2} \cdots (J_1 \cdot (J_0 p))))$$

we define $p = \tilde{s}_0 = [\tilde{x}_1, ..., \tilde{x}_n]^\top$ such that

$$\tilde{s}_{i+1} = J_i(s_i)\tilde{s}_i,\ i = 1, ..., m-1$$

with $\tilde{s}_{i+1} = [\tilde{s}_i^\top, \tilde{x}_{n+i}]^\top$.

Algorithmic Differentiation – Forward Mode

- Recall: the representation of a factorable function $f : \mathbb{R}^n \to \mathbb{R}^m$,

$$f(x) = C \cdot \Phi_{m-1}(\Phi_{m-2}(\cdots \Phi_1(\Phi_0(x))))$$

The Jacobian $J_f = \frac{\partial f}{\partial x}$ can be written as

$$J_f = C \cdot J_{m-1} \cdot J_{m-2} \cdots J_1 \cdot J_0 \quad \text{with } J_i = \frac{\partial \Phi_i}{\partial s_i}$$

- Main idea: the directional derivative $J_f p$ with seed $p \in \mathbb{R}^n$ given by

$$J_f p = C \cdot (J_{m-1} \cdot (J_{m-2} \cdots (J_1 \cdot (J_0 p))))$$

we define $p = \tilde{s}_0 = [\tilde{x}_1, ..., \tilde{x}_n]^\top$ such that

$$\tilde{s}_{i+1} = J_i(s_i)\tilde{s}_i, \; i = 1, ..., m-1$$

with $\tilde{s}_{i+1} = [\tilde{s}_i^\top, \tilde{x}_{n+i}]^\top$.

Algorithmic Differentiation – Forward Mode

Example: $f(x) = \sin(x_1 * x_2) + \cos(x_1)$:

$x_3 = x_1 * x_2$	$\tilde{x}_3 = x_1 * \tilde{x}_2 + \tilde{x}_1 * x_2$
$x_4 = \sin(x_3)$	$\tilde{x}_4 = \cos(x_3)\tilde{x}_3$
$x_5 = \cos(x_1)$	$\tilde{x}_5 = -\sin(x_1)\tilde{x}_1$
$x_6 = x_4 + x_5$	$\tilde{x}_6 = \tilde{x}_4 + \tilde{x}_5$

Result: $\tilde{x}_6 = \tilde{s}_0^\top \nabla f(x)$.

$$\mathrm{Cost}(J_f) \text{ *in forward mode* } \leq 2n \cdot \mathrm{Cost}(f)$$

Algorithmic Differentiation – Backward Mode

- Recall: the representation of a factorable function $f : \mathbb{R}^n \to \mathbb{R}^m$,

$$f(x) = C \cdot \Phi_{m-1}(\Phi_{m-2}(\cdots \Phi_1(\Phi_0(x))))$$

The Jacobian $J_f = \frac{\partial f}{\partial x}$ can be written as

$$J_f = C \cdot J_{m-1} \cdot J_{m-2} \cdots J_1 \cdot J_0 \quad \text{with } J_i = \frac{\partial \Phi_i}{\partial s_i}$$

- Main idea: the adjoint directional derivative $\lambda^\top J_f$ with seed $\lambda \in \mathbb{R}^m$ given by

$$\lambda^\top J_f = (((((\lambda^\top C) \cdot J_{m-1}) \cdot J_{m-2}) \cdots J_1) \cdot J_0)$$

we define $C^\top \lambda = \bar{s}_m$ such that

$$\bar{s}_i = J_i(s_i)^\top \bar{s}_{i+1},\ i = m-1, ..., 0$$

with $\bar{s}_{i+1} = [\bar{s}_i^\top, \bar{x}_{n+i}]^\top$.

Algorithmic Differentiation – Backward Mode

- Recall: the representation of a factorable function $f : \mathbb{R}^n \to \mathbb{R}^m$,

$$f(x) = C \cdot \Phi_{m-1}(\Phi_{m-2}(\cdots \Phi_1(\Phi_0(x))))$$

  The Jacobian $J_f = \frac{\partial f}{\partial x}$ can be written as

$$J_f = C \cdot J_{m-1} \cdot J_{m-2} \cdots J_1 \cdot J_0 \quad \text{with } J_i = \frac{\partial \Phi_i}{\partial s_i}$$

- Main idea: the adjoint directional derivative $\lambda^\top J_f$ with seed $\lambda \in \mathbb{R}^m$ given by

$$\lambda^\top J_f = (((((\lambda^\top C) \cdot J_{m-1}) \cdot J_{m-2}) \cdots J_1) \cdot J_0)$$

  we define $C^\top \lambda = \bar{s}_m$ such that

$$\bar{s}_i = J_i(s_i)^\top \bar{s}_{i+1},\ i = m-1, ..., 0$$

  with $\bar{s}_{i+1} = [\bar{s}_i^\top, \bar{x}_{n+i}]^\top$.

Algorithmic Differentiation – Backward Mode

Example: $f(x) = \sin(x_1 * x_2) + \cos(x_1)$:

$$\begin{aligned} x_3 &= x_1 * x_2 \\ x_4 &= \sin(x_3) \\ x_5 &= \cos(x_1) \\ x_6 &= x_4 + x_5 \end{aligned}$$

$$\begin{array}{lcl} \bar{x}_6 &=& 1,\ \bar{x}_i = 0,\ i = 1, ..., 5 \quad \text{define seed} \\ \hline \bar{x}_4 &=& \bar{x}_4 + \bar{x}_6 \\ \bar{x}_5 &=& \bar{x}_5 + \bar{x}_6 \\ \hline \bar{x}_1 &=& \bar{x}_1 - \sin(x_1)\bar{x}_5 \\ \hline \bar{x}_3 &=& \bar{x}_3 + \cos(x_3)\bar{x}_4 \\ \hline \bar{x}_1 &=& \bar{x}_1 + x_2 * \bar{x}_3 \\ \bar{x}_2 &=& \bar{x}_2 + x_1 * \bar{x}_3 \end{array}$$

Result: $\nabla f(x) = [\bar{x}_1, \bar{x}_2]^\top$.

$$\textit{Cost}(J_f) \textit{ in backward mode } \leq 3m \cdot \textit{Cost}(f)$$

Exercise

Consider function $f : \mathbb{R}^3 \to \mathbb{R}$,

$$f(x) = \sin(x_1 x_2) + \exp(x_1 x_2 x_3)$$

with $x = [x_1, x_2, x_3]^\top$. Write down

- its factorable form;
- the forward algorithmic differentiation;
- the backward algorithmic differentiation;

Exercise

Solution:

$$
\begin{array}{rcl|rcl}
x_4 & = & x_1 * x_2 & \tilde{x}_4 & = & x_1 * \tilde{x}_2 + \tilde{x}_1 * x_2 \\
x_5 & = & \sin(x_4) & \tilde{x}_5 & = & \cos(x_4)\tilde{x}_4 \\
x_6 & = & x_3 * x_4 & \tilde{x}_6 & = & x_3 * \tilde{x}_4 + \tilde{x}_3 * x_4 \\
x_7 & = & \exp(x_6) & \tilde{x}_7 & = & \exp(x_6)\tilde{x}_6 \\
x_8 & = & x_5 + x_7 & \tilde{x}_8 & = & \tilde{x}_5 + \tilde{x}_7
\end{array}
$$

Exercises

Solution:

INITIALIZE SEED

$\bar{x}_i = 0, \; i = 1, ..., 7$

$\bar{x}_8 = 1$

DIFFERENTIATION OF $x_8 = x_5 + x_7$

$\bar{x}_5 = \bar{x}_5 + \bar{x}_8$

$\bar{x}_7 = \bar{x}_7 + \bar{x}_8$

DIFFERENTIATION OF $x_7 = \exp(x_6)$

$\bar{x}_6 = \bar{x}_6 + \exp(x_6)\bar{x}_7$

DIFFERENTIATION OF $x_6 = x_3 * x_4$

$\bar{x}_3 = \bar{x}_3 + \bar{x}_4 * \bar{x}_6$

$\bar{x}_4 = \bar{x}_4 + \bar{x}_3 * \bar{x}_6$

DIFFERENTIATION OF $x_5 = \sin(x_4)$

$\bar{x}_4 = \bar{x}_4 + \cos(x_4)\bar{x}_5$

DIFFERENTIATION OF $x_4 = x_1 * x_2$

$\bar{x}_1 = \bar{x}_1 + x_2 * \bar{x}_4$

$\bar{x}_2 = \bar{x}_2 + x_1 * \bar{x}_4$

Summary

- Nonlinear programming = optimization in real-valued vector spaces.
- KKT-Conditions $\leftrightarrow$ first-order necessary conditions of optimality.
- Extendable to sufficient second-order conditions (not discussed here).
- Non-differentiable functions can cause numerical problems.
- Number of active constraints and correct identification of the active set is more important than the total number of constraints.

Initialization is key for solving non-convex problems.

Efficient derivative computation enables fast and reliable solutions.

Line search (and globalization) are crucial for global convergence.

Summary

- Nonlinear programming = optimization in real-valued vector spaces.
- KKT-Conditions $\leftrightarrow$ first-order necessary conditions of optimality.
- Extendable to sufficient second-order conditions (not discussed here).
- Non-differentiable functions can cause numerical problems.
- Number of active constraints and correct identification of the active set is more important than the total number of constraints.

Initialization is key for solving non-convex problems.

Efficient derivative computation enables fast and reliable solutions.

Line search (and globalization) are crucial for global convergence.

Literature and References

- B. Chachuat. Nonlinear and Dynamic Optimization - From Theory to Practice. EPFL, 2009. URL:https://infoscience.epfl.ch/record/111939/files/Chachuat_07(IC32).pdf
- D. Bertsekas. Nonlinear Programming. 2nd. Athena Scientific, Belmont, Massachusetts, 1999
- S.P. Boyd and L. Vandenberghe. Convex Optimization. University Press, Cambridge, 2004. URL: https://web.stanford.edu/~boyd/cvxbook/bv_cvxbook.pdf
- J. Nocedal and S. Wright. Numerical Optimization. 2nd Edition, 2006. URL: http://www.apmath.spbu.ru/cnsa/pdf/monograf/Numerical_Optimization2006.pdf
- A. Griewank and A. Walther. Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation. SIAM, 2008
- W. Squire and G. Trapp. Using complex variables to estimate derivatives of real functions. In: SIAM Review 40.1 (1998), pp. 110–112
- J.R.R.A. Martins, P. Sturdza, J.J. Alonso. The complex-step derivative approximation. In: ACM Transactions on Mathematical Software (TOMS) 29.3 (2003), pp. 245–262
- S. Gros and M. Diel. Numerical Optimal Control(draft). 2020. URL: https://www.syscop.de/files/2020ss/NOC/book-NOCSE.pdf