Optimization Methods

Fabian Bosshard

June 7, 2025

Contents

1 General formulation of an optimization problem

main components:

- **objective function** $f(\vec{\boldsymbol{x}})$: function to be minimized or maximized.
- **variables** $\vec{\boldsymbol{x}} = (x_1, \ldots, x_n) \in \mathbb{R}^n$: quantities of the problem that can be adjusted and need to be chosen *optimally*.
- **constraint functions** $c_i(\vec{\boldsymbol{x}})$: conditions that limit the possbile values of $\vec{\boldsymbol{x}}$. they can be *equality* or *inequality* constraints.
- **feasible region**: set of all variables $\vec{\boldsymbol{x}}$ satisfying all constraints.

general form of an optimization problem:

$$\boxed{\min_{\vec{\boldsymbol{x}} \in \mathbb{R}^n} f(\vec{\boldsymbol{x}}) \quad \text{subject to} \quad \begin{cases} c_i(\vec{\boldsymbol{x}}) = 0, & i \in \mathcal{E}, \\ c_i(\vec{\boldsymbol{x}}) \geq 0, & i \in \mathcal{I}, \end{cases}} \tag{1}$$

with

- $f : \mathbb{R}^n \to \mathbb{R}$ the objective function;
- $\mathcal{E}$ the set of equality constraints;
- $\mathcal{I}$ the set of inequality constraints.

Remark 1.

- We do not loose generality by restricting our analysis to minimisation problems, since maximizing f is *equivalent to* minimising $-f$.
- We do not loose generality by considering constraints of the form in (1) since:
  - a constraint of the form $c_i(\vec{\boldsymbol{x}}) = b$ can be transformed into $\bar{c}_i(\vec{\boldsymbol{x}}) = 0$ with $\bar{c}_i(\vec{\boldsymbol{x}}) = c_i(\vec{\boldsymbol{x}}) - b$;
  - a constraint of the form $c_i(\vec{\boldsymbol{x}}) \leq 0$ can be transformed into $\bar{c}_i(\vec{\boldsymbol{x}}) \geq 0$ with $\bar{c}_i(\vec{\boldsymbol{x}}) = -c_i(\vec{\boldsymbol{x}})$.
- If $n = 1$, the function f is *univariate* and we also say that the optimization problem is univariate. If $n = 2$, it is a *bivariate* problem, and if $n \geq 2$, it is a *multivariate* problem.

◀

A point $\vec{\boldsymbol{x}} \in \mathbb{R}^n$ satisfying all the constraints is called a *feasible point* and the set of all feasible points is called the *feasible set* (or, feasible region) of the optimization problem. It is formally defined as

$$\Omega = \{\vec{\boldsymbol{x}} \in \mathbb{R}^n : c_i(\vec{\boldsymbol{x}}) = 0,\ i \in \mathcal{E},\ c_i(\vec{\boldsymbol{x}}) \geq 0,\ i \in \mathcal{I}\}\,. \tag{2}$$

This also implies that the problem (1) can be re-written as

$$\min_{\vec{\boldsymbol{x}} \in \Omega} f(\vec{\boldsymbol{x}}).$$

The general formulation of optimization problems includes a large variety of problems and each category of problem requires a specific methodology. There is no universal optimization algorithm that is applicable to all problems. Thus, one needs to understand the specificity of each problem and algorithm to identify the most suitable methodology.

2 Useful definitions and properties

Theorem 1 (Cauchy–Schwarz Inequality)**.** The *Cauchy–Schwarz inequality* states that for any vectors $\vec{\boldsymbol{x}}$, $\vec{\boldsymbol{y}}$ of an inner product space, $|\langle \vec{\boldsymbol{x}}, \vec{\boldsymbol{y}} \rangle|^2 \leq \langle \vec{\boldsymbol{x}}, \vec{\boldsymbol{x}} \rangle \langle \vec{\boldsymbol{y}}, \vec{\boldsymbol{y}} \rangle$, where $\langle \cdot, \cdot \rangle$ is the inner product. Or written in terms of the induced norm $\| \cdot \| := \sqrt{\langle \cdot, \cdot \rangle}$, $|\langle \vec{\boldsymbol{x}}, \vec{\boldsymbol{y}} \rangle| \leq \|\vec{\boldsymbol{x}}\| \|\vec{\boldsymbol{y}}\|$. For $\mathbb{R}^n$ and the dot product, this means

$$|\vec{\boldsymbol{x}} \cdot \vec{\boldsymbol{y}}| \leq \|\vec{\boldsymbol{x}}\| \|\vec{\boldsymbol{y}}\|$$

$$\left| \sum_{i=1}^{n} x_i y_i \right| \leq \sqrt{\sum_{i=1}^{n} x_i^2} \sqrt{\sum_{i=1}^{n} y_i^2}.$$

◁

The dot product can be expressed in terms of the angle θ between the two vectors:

$$\vec{\boldsymbol{x}} \cdot \vec{\boldsymbol{y}} = \vec{\boldsymbol{x}}^T \vec{\boldsymbol{y}} = \|\vec{\boldsymbol{x}}\| \|\vec{\boldsymbol{y}}\| \cos(\theta).$$

Definition 1 (Spectral Norm)**.** The *spectral norm* of a matrix $\underline{\boldsymbol{A}} \in \mathbb{R}^{m \times n}$ is defined as

$$\|\underline{\boldsymbol{A}}\|_2 := \max_{\vec{\boldsymbol{x}} \neq \vec{\boldsymbol{0}}} \frac{\|\underline{\boldsymbol{A}} \vec{\boldsymbol{x}}\|_2}{\|\vec{\boldsymbol{x}}\|_2} = \max_{\|\vec{\boldsymbol{x}}\|_2 = 1} \|\underline{\boldsymbol{A}} \vec{\boldsymbol{x}}\|_2 = \sqrt{\lambda_{\max}(\underline{\boldsymbol{A}}^T \underline{\boldsymbol{A}})},$$

where $\lambda_{\max}(\underline{\boldsymbol{A}}^T \underline{\boldsymbol{A}})$ is the largest eigenvalue of $\underline{\boldsymbol{A}}^T \underline{\boldsymbol{A}} \in \mathbb{R}^{n \times n}$. If $\underline{\boldsymbol{A}}$ is symmetric, then the spectral norm is equal to the largest eigenvalue (in absolute value) of $\underline{\boldsymbol{A}}$,

$$\|\underline{\boldsymbol{A}}\|_2 = \max_{i \in \{1, \ldots, n\}} |\lambda_i(\underline{\boldsymbol{A}})|, \qquad \text{☞ if } \underline{\boldsymbol{A}}^T = \underline{\boldsymbol{A}}$$

where $\lambda_i(\underline{\boldsymbol{A}})$ is the i-th eigenvalue of $\underline{\boldsymbol{A}}$. ✍

Definition 2 (Condition Number)**.** The *condition number* of a matrix $\underline{\boldsymbol{A}} \in \mathbb{R}^{m \times n}$ is defined as

$$\kappa(\underline{\boldsymbol{A}}) := \|\underline{\boldsymbol{A}}\|_2 \cdot \|\underline{\boldsymbol{A}}^{-1}\|_2 \geq 1.$$

If $\underline{\boldsymbol{A}}$ is symmetric positive definite,

$$\kappa(\underline{\boldsymbol{A}}) = \frac{\lambda_{\max}(\underline{\boldsymbol{A}})}{\lambda_{\min}(\underline{\boldsymbol{A}})}. \qquad \text{☞ if } \underline{\boldsymbol{A}} \succ 0$$

✍

3 Classification of Problems

Note that "programming" is used synonymously with "optimization".

3.1 Linear vs Nonlinear Programming

An optimization problem is said to be *linear* if f is linear (or, affine) and the constraints are linear; otherwise, the problem is *nonlinear*.

Definition 3 (Linear Function)**.** A function $f : \mathbb{R}^n \to \mathbb{R}$ is *linear* if:

- for any $\vec{\boldsymbol{x}}, \vec{\boldsymbol{y}} \in \mathbb{R}^n$, $f(\vec{\boldsymbol{x}} + \vec{\boldsymbol{y}}) = f(\vec{\boldsymbol{x}}) + f(\vec{\boldsymbol{y}})$,
- for any $\vec{\boldsymbol{x}} \in \mathbb{R}^n$ and $\lambda \in \mathbb{R}$, $f(\lambda\vec{\boldsymbol{x}}) = \lambda f(\vec{\boldsymbol{x}})$.

Equivalently, f is linear if it can be written as

$$f(\vec{\boldsymbol{x}}) = \vec{\boldsymbol{c}}^T\vec{\boldsymbol{x}} = \sum_{i=1}^{n} c_i x_i,$$

with $\vec{\boldsymbol{c}} \in \mathbb{R}^n$. ✍

Definition 4 (Affine Function)**.** A function $f : \mathbb{R}^n \to \mathbb{R}$ is *affine* if

$$f(\alpha\vec{\boldsymbol{x}} + (1-\alpha)\vec{\boldsymbol{y}}) = \alpha f(\vec{\boldsymbol{x}}) + (1-\alpha) f(\vec{\boldsymbol{y}}), \quad \forall\, \vec{\boldsymbol{x}}, \vec{\boldsymbol{y}} \in \mathbb{R}^n,\ \forall\, \alpha \in [0,1],$$

or equivalently, if there exist $\vec{\boldsymbol{c}} \in \mathbb{R}^n$ and $b \in \mathbb{R}$ such that

$$f(\vec{\boldsymbol{x}}) = \vec{\boldsymbol{c}}^T\vec{\boldsymbol{x}} + b.$$

✍

Example 1. The following minimisation problem is linear:

$$\min_{\vec{\boldsymbol{x}} \in \mathbb{R}^n} \sum_{i=1}^{n} c_i x_i \quad \text{subject to} \quad \sum_{i=1}^{n} x_i = 1,$$

with $c_1, \ldots, c_n \in \mathbb{R}$. ◀

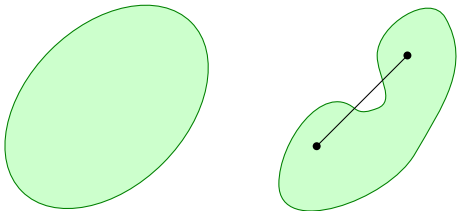
3.2 Continuous vs Discrete Programming

If the components of $\vec{\boldsymbol{x}}$ are real numbers and in an uncountable set, it is a *continuous* optimization problem. Otherwise, it is a *discrete* programming problem (if the variables can only be integers, it is an *integer* programming problem).

Example 2. of a continuous as well as a discrete problem:

- If $x_1, \ldots, x_n$ can take any value in $\mathbb{R}$ or in $[a, b]$ with $a < b$, then the problem is continuous.
- If $x_1, \ldots, x_n$ can take values in $\mathbb{N}$ or $\mathbb{Z}$, then the problem is discrete.

◀



Graphical interpretation of convex sets.

3.3 Convex vs Non-convex Optimization

An optimization problem is said to be *convex* if the objective function f is convex, the equality constraints are linear, and the inequality constraints are concave. Equivalently, the problem is convex if the feasible set Ω is convex.

Definition 5 (Convex Set)**.** A set $S \subset \mathbb{R}^n$ is *convex* if $\forall\, \vec{\boldsymbol{x}}, \vec{\boldsymbol{y}} \in S,\ \forall\, t \in [0,1]$,

$$t\vec{\boldsymbol{x}} + (1-t)\vec{\boldsymbol{y}} \in S.$$

✍

Example 3. A ball defined as

$$\mathcal{B}_r(\vec{\boldsymbol{x}}_0) = \{\vec{\boldsymbol{x}} \in \mathbb{R}^n : \|\vec{\boldsymbol{x}} - \vec{\boldsymbol{x}}_0\| \leq r\}\,,$$

with center $\vec{\boldsymbol{x}}_0 \in \mathbb{R}^n$ and radius r, is convex. ◀

Unless specified otherwise, the norm $\|\cdot\|$ is the Euclidean norm:

$$\|\vec{\boldsymbol{x}}\| = \|\vec{\boldsymbol{x}}\|_2 = \sqrt{\sum_{i=1}^{n} x_i^2}.$$

Definition 6 (Convex Function)**.** A function $f : S \to \mathbb{R}$ is *convex* if its domain S is convex and $\forall\, \vec{\boldsymbol{x}}, \vec{\boldsymbol{y}} \in S\ \forall\, t \in [0,1]$,

$$f\left(t\vec{\boldsymbol{x}} + (1-t)\vec{\boldsymbol{y}}\right) \leq tf(\vec{\boldsymbol{x}}) + (1-t)f(\vec{\boldsymbol{y}}).$$

✍

Concavity is the opposite of convexity: f is *concave* if $-f$ is convex.

Definition 7 (Strictly Convex Function)**.** A function $f : S \to \mathbb{R}$ is *strictly convex* if its domain S is convex and $\forall\, \vec{\boldsymbol{x}}, \vec{\boldsymbol{y}} \in S\ \forall\, t \in [0,1]$,

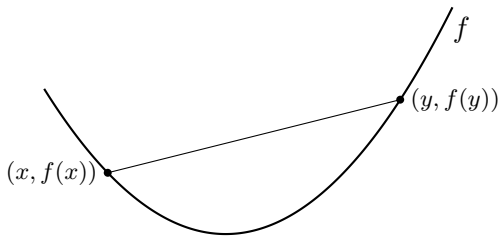
$$f\left(t\vec{\boldsymbol{x}} + (1-t)\vec{\boldsymbol{y}}\right) < t\, f(\vec{\boldsymbol{x}}) + (1-t)\, f(\vec{\boldsymbol{y}}).$$

✍

Definition 8 (μ-Strongly Convex Function)**.** A function $f : S \to \mathbb{R}$ is *μ-strongly convex* (with $\mu > 0$) if its domain S is convex and $\forall\, \vec{\boldsymbol{x}}, \vec{\boldsymbol{y}} \in S\ \forall\, t \in [0,1]$,

$$f\left(t\vec{\boldsymbol{x}} + (1-t)\vec{\boldsymbol{y}}\right) \leq t\, f(\vec{\boldsymbol{x}}) + (1-t)\, f(\vec{\boldsymbol{y}}) - \frac{\mu}{2}\, t(1-t)\|\vec{\boldsymbol{x}} - \vec{\boldsymbol{y}}\|^2.$$

✍



Example of a convex function.

Observe that "strong convexity" $\implies$ "strict convexity" $\implies$ "convexity".

Example 4. The following examples and properties of convex functions are important:

- The exponential function is convex, while the logarithm function is concave.
- Powers of positive numbers x^α with $x > 0$ are convex if $\alpha \geq 1$ or $\alpha \leq 0$, and concave if $0 \leq \alpha \leq 1$.
- Affine functions are both convex and concave.
- Absolute value and norms, e.g., ℓ_p-norms are convex:
  - $\|\vec{\boldsymbol{x}}\|_p = \left(\sum_{i=1}^n |x_i|^p\right)^{1/p}$,
  - $\|\vec{\boldsymbol{x}}\|_\infty = \max\{|x_1|, \ldots, |x_n|\}$
- Positive part (Rectified Linear Unit) $\max\{x, 0\}$ is convex, negative part $\min\{0, x\}$ is concave.
- Positive linear combinations of convex functions are convex.
- If h is convex and g is convex and non-decreasing, then $g\left(h(\vec{\boldsymbol{x}})\right)$ is convex.

◀

3.4 Constrained vs Unconstrained Optimization

An optimization problem is *constrained* if $\mathcal{E}$ OR $\mathcal{I}$ are NOT empty. It is *unconstrained* if both $\mathcal{E}$ AND $\mathcal{I}$ are empty.

3.5 Smooth vs Non-smooth Optimization

An optimization problem is said to be *smooth* if the objective function f is smooth, i.e., f is continuously differentiable and its derivative is Lipschitz continuous; otherwise, the problem is *non-smooth*.

Definition 9 (Continuity)**.** A function $f : \mathbb{R} \to \mathbb{R}$ is *continuous* on $\mathbb{R}$ if for all $x_0 \in \mathbb{R}$,

$$\lim_{x \to x_0} f(x) = f(x_0).$$

✍

Definition 10 (Lipschitz Continuity)**.** A function $f : \mathbb{R} \to \mathbb{R}$ is *L-Lipschitz continuous* with $L > 0$ if for all $x, y \in \mathbb{R}$,

$$|f(x) - f(y)| \leq L|x - y|.$$

✍

Observe that "Lipschitz continuity" $\implies$ "continuity".

Definition 11 (Derivative)**.** The derivative of $f : S \to \mathbb{R}$ at $x \in S \subset \mathbb{R}$ is defined as

$$f'(x) = \lim_{t \to 0} \frac{f(x+t) - f(x)}{t} = \frac{df}{dx}(x).$$

✍

Definition 12 (Differentiability)**.** A univariate function $f : S \to \mathbb{R}$ with domain $S \subseteq \mathbb{R}$ is *differentiable* if S is an open set and the derivative $f'(x)$ exists for every $x \in S$. It is *continuously differentiable* if f' is continuous. ✍

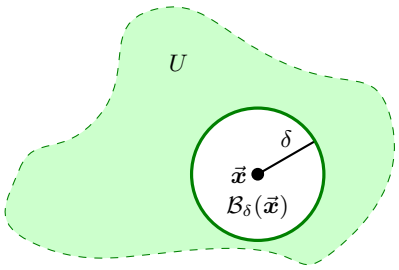


Illustration of an open set.

Definition 13 (Open Set)**.** An open set $U \subseteq \mathbb{R}^n$ is a set that contains an open ball around each point, i.e.,

$$\forall\, \vec{\boldsymbol{x}} \in U, \quad \exists\, \delta > 0, \quad \mathcal{B}_\delta(\vec{\boldsymbol{x}}) = \{\vec{\boldsymbol{y}} \in \mathbb{R}^n : \|\vec{\boldsymbol{y}} - \vec{\boldsymbol{x}}\| < \delta\} \subseteq U.$$

✍

3.6 Global vs Local Optimization

We say that an optimization problem is solved globally if a *global minimizer* is found; otherwise, if only a *local minimizer* is found, the problem is solved locally. Recall that the feasible set is denoted by Ω.

Definition 14 (Global Minimizer)**.** A vector $\vec{\boldsymbol{x}}^* \in \Omega$ is a *global minimizer* of f if

$$f(\vec{\boldsymbol{x}}^*) \leq f(\vec{\boldsymbol{x}}), \quad \forall\, \vec{\boldsymbol{x}} \in \Omega.$$

✍

Definition 15 (Local Minimizer)**.** A vector $\vec{\boldsymbol{x}}^* \in \Omega$ is a *local minimizer* of f if

$$f(\vec{\boldsymbol{x}}^*) \leq f(\vec{\boldsymbol{x}}),$$

for any $\vec{\boldsymbol{x}}$ in a neighborhood of $\vec{\boldsymbol{x}}^*$. ✍

Definition 16 (Strict Local Minimizer)**.** A vector $\vec{\boldsymbol{x}}^* \in \Omega$ is a *strict local minimizer* of f if

$$f(\vec{\boldsymbol{x}}^*) < f(\vec{\boldsymbol{x}}),$$

for any $\vec{\boldsymbol{x}}$ in a neighborhood of $\vec{\boldsymbol{x}}^*$. ✍

Definition 17 (Isolated Local Minimizer)**.** A vector $\vec{\boldsymbol{x}}^* \in \Omega$ is an *isolated local minimizer* of f if it is the only local minimizer in a neighborhood of $\vec{\boldsymbol{x}}^*$. ✍

Remark 2. An example for a neighborhood $\mathcal{N}(\vec{\boldsymbol{x}}^*)$ is the open ball

$$\mathcal{B}_r(\vec{\boldsymbol{x}}^*) := \{\vec{\boldsymbol{x}} \in \mathbb{R}^n : \|\vec{\boldsymbol{x}} - \vec{\boldsymbol{x}}^*\| < r\},$$

with some radius $r > 0$. ◀

Remark 3.

- An isolated local minimizer is always a strict minimizer but the converse is not true: there are strict minimizers that are not isolated.
- A global minimizer can also be strict (by replacing $\leq$ with $<$ in Definition 14). It can also be isolated (if there exists a neighborhood with no local minimizer).

◀

3.7 Stochastic vs Deterministic Optimization

The distinction here refers to the type of algorithms used to solve the optimization problem. A stochastic optimization algorithm incorporates some randomness in its iterations. Such algorithm may find different local minimizers even if it is initialised at the same point.

4 Unconstrained Optimization

4.1 Univariate Objective Functions

The general form of a univariate unconstrained optimization problem is

$$\min_{x \in \mathbb{R}} f(x),$$

with $f : \mathbb{R} \to \mathbb{R}$ the objective function. Since there are no equality or inequality constraints, x may take any real value. In this course we restrict ourselves to smooth functions (see 3.5).

In this chapter (and later ones) we will see that the concept of derivatives is fundamental for solving optimization problems; in particular, derivatives allow us:

1. to construct local approximating models (via Taylor's theorem),
2. to characterise the solutions of the problem (i.e., by deriving optimality conditions).

4.1.1 Taylor Expansions for Univariate Functions

We recall the definitions of the first and second derivatives for a univariate function.

Definition 18 (First Derivative)**.** For a function $f : \mathbb{R} \to \mathbb{R}$ and a point $x \in \mathbb{R}$, the first (or first-order) derivative is defined as

$$f'(x) = \lim_{d \to 0} \frac{f(x+d) - f(x)}{d}.$$

✍

Definition 19 (Second Derivative)**.** The second (or second-order) derivative of f at x is defined as

$$f''(x) = \lim_{d \to 0} \frac{f'(x+d) - f'(x)}{d}.$$

✍

Recall that $f'(x)$ measures the rate of change (the slope of the tangent line) of f at x. Thus for small d one has

$$f(x+d) \approx f(x) + f'(x)\, d.$$

In fact, the tangent line at x is given by

$$t(y; x) = f(x) + f'(x)(y - x), \quad y \in \mathbb{R},$$

or, equivalently, writing $y = x + d$,

$$t(x+d; x) = f(x) + f'(x)\, d.$$

This linear approximation is known as the *first-order Taylor approximation* of f.

By incorporating the second derivative, one can construct a local quadratic model (the second-order Taylor approximation):

$$f(x+d) \approx q(x+d; x) = f(x) + f'(x)\, d + \frac{1}{2} f''(x)\, d^2, \quad d \in \mathbb{R}, \tag{3}$$

or equivalently,

$$f(y) \approx q(y;x) = f(x) + f'(x)(y-x) + \frac{1}{2}f''(x)(y-x)^2, \quad y \in \mathbb{R}. \tag{4}$$

Here, $f''(x)$ is sometimes called the *curvature* of f at x. Note that these approximations are valid in a neighbourhood of x (i.e. for small $d = y - x$).

Taylor's theorem provides an exact expansion.

Theorem 2 (First and Second Order Taylor's Expansion for Univariate Functions)**.** If $f : \mathbb{R} \to \mathbb{R}$ is continuously differentiable, then for any $x \in \mathbb{R}$ and $d \in \mathbb{R}$ there exists $t \in (0,1)$ such that

$$f(x+d) = f(x) + f'(x+t\,d)\,d.$$

Moreover, if f is twice continuously differentiable, then for any $x \in \mathbb{R}$ and $d \in \mathbb{R}$ there exists $t \in (0,1)$ such that

$$f(x+d) = f(x) + f'(x)d + \frac{1}{2}f''(x+t\,d)\,d^2.$$

◁

A useful consequence of Taylor's theorem is a bound on the approximation errors of the linear and quadratic models.

Corollary 3. If f is twice continuously differentiable then for any $x \in \mathbb{R}$ there exist constants $L > 0$ and $\epsilon > 0$ such that

$$|f(y) - t(y;x)| \leq L\,(y-x)^2,$$

for all $y \in \mathbb{R}$ with $|y - x| \leq \epsilon$. If f is three times continuously differentiable, there exist $L > 0$ and $\epsilon > 0$ such that

$$|f(y) - q(y;x)| \leq L\,|y-x|^3,$$

for all $y \in \mathbb{R}$ with $|y - x| \leq \epsilon$. ◁

That is, for $|y - x|$ sufficiently small, the error of the linear model is of order $O\left((y-x)^2\right)$ and the quadratic model approximates f with error of order $O\left(|y-x|^3\right)$.

Actually Taylor's expansion does not stop at the second derivatives. It can also include higher-order derivatives of f (if they exist) in order to construct increasingly better approximation of f:

$$f(x+d) \approx f(x) + \frac{1}{1!}f'(x)d + \frac{1}{2!}f''(x)d^2 + \frac{1}{3!}f^{(3)}(x)d^3 + \cdots + \frac{1}{k!}f^{(k)}(x)d^k$$

Nonetheless in optimization we will essentially need the first and second order approximations.

4.1.2 First and Second Order Optimality Conditions for Univariate Functions

In unconstrained optimisation we can derive "recipes" to characterise solutions of our problem. These "recipes" are optimality conditions.

Theorem 4 (Necessary First-Order Optimality Condition for Univariate Functions)**.** If f is continuously differentiable and x^* is a local minimizer of f, then $f'(x^*) = 0$, i.e. x^* is a *stationary point.* ◁

Proof. (Contradiction) Assume $f'(x^*) \neq 0$. We will use the fact that around x^* the tangent line approximates f so that if its slope is not null, we can decrease f by either decreasing or increasing x^*. Let $d := -f'(x^*)$. Thus $df'(x^*) < 0$ and since f' is continuous there exist t_0 such that $\forall t \in [0, t_0]$

$$df'(x^* + td) < 0$$

Let $t \in [0, t_0]$. By the first part of Taylor's theorem, there exists $\bar{t} \in (0, t)$ such that

$$f(x^* + td) = f(x^*) + tdf'(x^* + \bar{t}d)$$

Since $\bar{t} \leq t \leq t_0$ we have $df'(x^* + \bar{t}d) < 0$ which implies $f(x^* + td) < f(x^*)$. Since one can choose t arbitrarily close to 0 , this proves that x^* is not a local minimizer. □

It is important to note that stationary points may correspond to local minimizers, local maximizers, or saddle points. Therefore, further analysis (using second derivatives) is often required.

Theorem 5 (Necessary Second-Order Optimality Condition)**.** If f is twice continuously differentiable and x^* is a local minimizer of f, then

$$f''(x^*) \geq 0.$$

◁

Proof. (Contradiction) Assume $f''(x^*) < 0$. Note that if $f'(x^*) \neq 0$ we already proved that x^* is NOT a local minimizer so we can assume that $f'(x^*) = 0$. Let $d \neq 0$. Since $f''(x^*) < 0$, we have $f''(x^*) p^2 < 0$. By continuity of f'', there exists $t_0 > 0$ such that

$$f''(x^* + td) d^2 < 0, \quad \forall t \in (0, t_0)$$

Let $t \in (0, t_0)$. By the second part of Taylor's theorem there exists $\bar{t} \in (0, t)$ such that

$$f(x^* + td) = f(x) + \frac{1}{2} f''(x + \bar{t}d) t^2 d^2$$

Since $\bar{t} \leq t \leq t_0$, $f''(x + \bar{t}d) < 0$, which implies that $f(x^* + td) < f(x^*)$. As before we can conclude that x^* is NOT a local minimizer. □

However, the above conditions are necessary but not sufficient. For example, if $f'(x) = 0$ and $f''(x) = 0$ the test is inconclusive.

A sufficient condition is obtained by strengthening the second-order condition.

Theorem 6 (Sufficient Optimality Conditions)**.** If f is twice continuously differentiable, $f'(x^*) = 0$ and $f''(x^*) > 0$, then x^* is a strict local minimizer of f. ◁

Proof. Since f'' is continuous, there exists $r > 0$ such that for all d with $|d| < r$ we have $f''(x^* + d) > 0$. Then by the second-order Taylor expansion, for all such d (with $d \neq 0$) there exists $t \in (0, 1)$ such that

$$f(x^* + d) = f(x^*) + \frac{1}{2} f''(x^* + td) d^2 > f(x^*).$$

Thus x^* is a strict local minimizer. □

Remark 4. Note that the sufficient condition is not necessary; for instance, $f(x) = x^4$ has a strict local minimizer at $x = 0$ despite $f''(0) = 0$. ◀

4.1.3 Minimizers for Convex Univariate Functions

Identifying minimizers is considerably simpler when f is convex (Definition 6). If f is twice differentiable, the following characterisations are equivalent:

$$\begin{aligned} f \text{ is convex} &\iff f''(x) \geq 0, \quad \forall x \in \mathbb{R}, \\ &\iff f(y) \geq f(x) + f'(x)(y - x), \quad \forall x, y \in \mathbb{R}. \end{aligned}$$

Two important properties follow immediately:

Proposition 7. If f is convex, any local minimizer is a global minimizer. ◁

Proposition 8. If f is convex and differentiable, then any stationary point is a global minimizer. ◁

4.2 Multivariate Objective Functions

The general form of a multivariate unconstrained optimization problem is

$$\min_{\vec{\boldsymbol{x}} \in \mathbb{R}^n} f(\vec{\boldsymbol{x}}),$$

with $f : \mathbb{R}^n \to \mathbb{R}$ the objective function, i.e. its argument is a vector with n coordniates:

$$f(\vec{\boldsymbol{x}}) = f(x_1, \ldots, x_n).$$

We will again assume that f is smooth (see 3.5): A function $f : \mathbb{R}^n \to \mathbb{R}$ is said to be L-smooth if it is continuously differentiable and its gradient ∇f is Lipschitz continuous with Lipschitz constant $L > 0$:

$$\boxed{\|\nabla f(\vec{\boldsymbol{x}}) - \nabla f(\vec{\boldsymbol{y}})\| \leq L\|\vec{\boldsymbol{x}} - \vec{\boldsymbol{y}}\|, \quad \forall\, \vec{\boldsymbol{x}}, \vec{\boldsymbol{y}} \in \mathbb{R}^n.} \tag{5}$$

If f is twice continuously differentiable, this is equivalent to $\|\underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}})\|_2 \leq L$ for all $\vec{\boldsymbol{x}} \in \mathbb{R}^n$, where $\|\cdot\|_2$ denotes the spectral norm (Definition 1). Additionally, we assume that f is bounded from below, i.e., there exists $M \in \mathbb{R}$ such that $f(\vec{\boldsymbol{x}}) \geq M$ for all $\vec{\boldsymbol{x}} \in \mathbb{R}^n$.

4.2.1 First Order Derivatives for Multivariate Functions

The first-order derivative of a multivariate function is given by its *gradient*. To define the gradient we first recall partial derivatives and introduce the canonical basis.

We denote by $\{\vec{\boldsymbol{e}}_i\}_{i=1,\ldots,n}$ the canonical basis of $\mathbb{R}^n$ where

$$\vec{\boldsymbol{e}}_i = \begin{pmatrix} 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{pmatrix} \quad \text{(with the } i\text{-th coordinate equal to 1).}$$

Definition 20 (Partial Derivative)**.** Let $f : \mathbb{R}^n \to \mathbb{R}$. The partial derivative of f with respect to the i-th coordinate at a point $\vec{\boldsymbol{x}} \in \mathbb{R}^n$ is defined as

$$\frac{\partial f}{\partial x_i}(\vec{\boldsymbol{x}}) = \lim_{h \to 0} \frac{f(\vec{\boldsymbol{x}} + h\,\vec{\boldsymbol{e}}_i) - f(\vec{\boldsymbol{x}})}{h} = \lim_{h \to 0} \frac{f(x_1, \ldots, x_i + h, \ldots, x_n) - f(x_1, \ldots, x_i, \ldots, x_n)}{h}.$$

✍

Definition 21 (Gradient)**.** Let $f : \mathbb{R}^n \to \mathbb{R}$. The *gradient* of f at $\vec{\boldsymbol{x}}$ is the vector

$$\nabla f(\vec{\boldsymbol{x}}) = \begin{pmatrix} \frac{\partial f}{\partial x_1}(\vec{\boldsymbol{x}}) \\ \vdots \\ \frac{\partial f}{\partial x_n}(\vec{\boldsymbol{x}}) \end{pmatrix} \in \mathbb{R}^n.$$

✍

The gradient gives the direction of steepest increase of f at $\vec{\boldsymbol{x}}$ and each coordinate quantifies the rate of change of f in the corresponding direction. e say that f is differentiable if $\nabla f(\vec{\boldsymbol{x}})$ exists for all $\vec{\boldsymbol{x}}$, and continuously differentiable if ∇f is continuous.

The gradient can sldo be used to compute the change in any direction. If we define the univariate function

$$g(t) = f\left(\vec{\boldsymbol{x}} + t\,\vec{\boldsymbol{d}}\right), \quad t \in \mathbb{R},$$

for a direction $\vec{\boldsymbol{d}} \in \mathbb{R}^n$, then by the chain rule we have

$$g'(t) = \frac{d}{dt} g(t) = \frac{d}{dt}\left(f\left(\vec{\boldsymbol{x}} + t\,\vec{\boldsymbol{d}}\right)\right) = \nabla f\left(\vec{\boldsymbol{x}} + t\,\vec{\boldsymbol{d}}\right)^T \vec{\boldsymbol{d}}.$$

Thus, the *directional derivative* of f at $\vec{\boldsymbol{x}}$ in the direction $\vec{\boldsymbol{d}}$ is given by

$$g'(0) = \nabla f(\vec{\boldsymbol{x}})^T \vec{\boldsymbol{d}}.$$

Among all unit vectors $\vec{\boldsymbol{d}}$ (i.e. with $\|\vec{\boldsymbol{d}}\| = 1$), the maximum value of $g'(0)$ is attained when $\vec{\boldsymbol{d}}$ is aligned with $\nabla f(\vec{\boldsymbol{x}})$.

4.2.2 First Order Taylor's Expansion for Multivariate Functions

A first-order Taylor expansion (or linear local approximation) of f at $\vec{\boldsymbol{x}}$ is

$$t(\vec{\boldsymbol{y}}; \vec{\boldsymbol{x}}) = f(\vec{\boldsymbol{x}}) + \nabla f(\vec{\boldsymbol{x}})^T (\vec{\boldsymbol{y}} - \vec{\boldsymbol{x}}), \quad \vec{\boldsymbol{y}} \in \mathbb{R}^n.$$

That is,

$$f(\vec{\boldsymbol{x}} + \vec{\boldsymbol{d}}) \approx f(\vec{\boldsymbol{x}}) + \nabla f(\vec{\boldsymbol{x}})^T \vec{\boldsymbol{d}}.$$

Theorem 9 (First Order Taylor's Expansion for Multivariate Functions)**.** Let $f : \mathbb{R}^n \to \mathbb{R}$ be continuously differentiable. Then for any $\vec{\boldsymbol{x}} \in \mathbb{R}^n$ and $\vec{\boldsymbol{d}} \in \mathbb{R}^n$ there exists $t \in (0, 1)$ such that

$$f(\vec{\boldsymbol{x}} + \vec{\boldsymbol{d}}) = f(\vec{\boldsymbol{x}}) + \nabla f(\vec{\boldsymbol{x}} + t\,\vec{\boldsymbol{d}})^T \vec{\boldsymbol{d}}.$$

If we set $n = 1$ we recover the univariate case (Theorem 2). ◁

Corollary 10. If $f : \mathbb{R}^n \to \mathbb{R}$ is twice continuously differentiable, then for any $\vec{\boldsymbol{x}} \in \mathbb{R}^n$ there exist constants $L > 0$ and $\epsilon > 0$ such that for all $\vec{\boldsymbol{y}}$ in

$$\mathcal{B}_\epsilon(\vec{\boldsymbol{x}}) = \left\{\bar{\vec{\boldsymbol{x}}} \in \mathbb{R}^n : \left\|\bar{\vec{\boldsymbol{x}}} - \vec{\boldsymbol{x}}\right\| \leq \epsilon\right\},$$

the error in the linear approximation satisfies

$$\left|f(\vec{\boldsymbol{y}}) - t(\vec{\boldsymbol{y}}; \vec{\boldsymbol{x}})\right| \leq L\|\vec{\boldsymbol{y}} - \vec{\boldsymbol{x}}\|^2.$$

If we set $n = 1$ we recover the univariate case (Corollary 3). ◁

4.2.3 First Order Optimality Conditions for Multivariate Functions

Recall the definitions for local and global minimizers (Definitions 15 and 14).

Theorem 11 (Necessary First-Order Optimality Condition)**.** If f is continuously differentiable and $\vec{\boldsymbol{x}}^*$ is a local minimizer of f then $\nabla f(\vec{\boldsymbol{x}}^*) = \vec{\mathbf{0}}$, i.e. $\vec{\boldsymbol{x}}^*$ is a *stationary point*. If we set $n = 1$ we recover the univariate case (Theorem 4). ◁

Proof. (Contradiction) Assume that $\nabla f(\vec{\boldsymbol{x}}^*) \neq \vec{\mathbf{0}}$. Define the direction $\vec{\boldsymbol{d}} = -\nabla f(\vec{\boldsymbol{x}}^*)$. Consider the univariate function

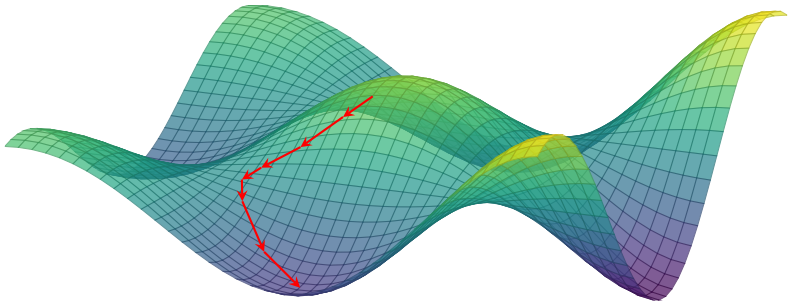
$$g(t) = f(\vec{\boldsymbol{x}}^* + t\,\vec{\boldsymbol{d}}), \quad t \in \mathbb{R}.$$

Then, $g'(0) = \nabla f(\vec{\boldsymbol{x}}^*)^T \vec{\boldsymbol{d}} = -\|\nabla f(\vec{\boldsymbol{x}}^*)\|^2 < 0$. Thus, for sufficiently small $t > 0$ we have $g(t) < g(0)$, meaning

$$f(\vec{\boldsymbol{x}}^* + t\,\vec{\boldsymbol{d}}) < f(\vec{\boldsymbol{x}}^*).$$

This contradicts the local minimality of $\vec{\boldsymbol{x}}^*$. Hence, it must be that $\nabla f(\vec{\boldsymbol{x}}^*) = \vec{\mathbf{0}}$. □

4.3 Gradient Descent



Since the gradient $\nabla f(\vec{\boldsymbol{x}})$ points in the direction of steepest increase, the negative gradient $-\nabla f(\vec{\boldsymbol{x}})$ gives the direction of steepest descent. Recall that for any direction $\vec{\boldsymbol{d}} \in \mathbb{R}^n$ the directional derivative of f at $\vec{\boldsymbol{x}}$ in the direction $\vec{\boldsymbol{d}}$ is $g'(0) = \nabla f(\vec{\boldsymbol{x}})^T \vec{\boldsymbol{d}}$. We can write

$$\nabla f(\vec{\boldsymbol{x}})^T \vec{\boldsymbol{d}} = \cos(\theta) \|\nabla f(\vec{\boldsymbol{x}})\| \|\vec{\boldsymbol{d}}\|$$

where θ is the angle between $\vec{\boldsymbol{d}}$ and $\nabla f(\vec{\boldsymbol{x}})$. Among all unit directions (i.e. $\|\vec{\boldsymbol{d}}\| = 1$), the decrease is maximized when $\cos(\theta) = -1$, i.e. when $\vec{\boldsymbol{d}}$ is collinear with $-\nabla f(\vec{\boldsymbol{x}})$, but points in the opposite direction. Since it is of unit norm, we have

$$\vec{\boldsymbol{d}} = -\frac{\nabla f(\vec{\boldsymbol{x}})}{\|\nabla f(\vec{\boldsymbol{x}})\|},$$

which is called the *direction of steepest descent.*

A gradient descent algorithm then generates a sequence $\{\vec{\boldsymbol{x}}^{(k)}\}$ via

$$\vec{\boldsymbol{x}}^{(k+1)} = \vec{\boldsymbol{x}}^{(k)} - \alpha \nabla f\big(\vec{\boldsymbol{x}}^{(k)}\big),$$

where $\alpha > 0$ is the step size. A common stopping rule is to terminate when $\|\nabla f(\vec{\boldsymbol{x}}^{(k)})\|$ falls below a prescribed tolerance level ϵ or after a maximum number of iterations $k_{\max}$.

4.3.1 Gradient Descent Algorithm

Algorithm 1 Gradient Descent

1: **Input:** Starting point $\vec{\boldsymbol{x}}^{(0)}$, step size $\alpha > 0$, tolerance $\epsilon > 0$, and maximum iterations $k_{\max}$
2: Set $k \leftarrow 0$
3: Compute $\vec{\boldsymbol{g}}^{(0)} = \nabla f\big(\vec{\boldsymbol{x}}^{(0)}\big)$
4: **while** $\|\vec{\boldsymbol{g}}^{(k)}\| > \epsilon$ and $k < k_{\max}$ **do**
5: Update: $\vec{\boldsymbol{x}}^{(k+1)} = \vec{\boldsymbol{x}}^{(k)} - \alpha\, \vec{\boldsymbol{g}}^{(k)}$
6: Set $k \leftarrow k + 1$
7: Compute $\vec{\boldsymbol{g}}^{(k)} = \nabla f\big(\vec{\boldsymbol{x}}^{(k)}\big)$
8: **end while**
9: **Output:** $\vec{\boldsymbol{x}}^{(k)}$

4.3.2 Convergence Analysis

Theorem 12 (Global Convergence of Gradient Descent)**.** Let f be twice continuously differentiable and L-smooth (i.e. its gradient is Lipschitz continuous with constant $L > 0$). Assume that f is

bounded from below by m and let $0 < \alpha < \frac{2}{L}$. Then for any starting point $\vec{\boldsymbol{x}}^{(0)}$, the iterates $\{\vec{\boldsymbol{x}}^{(k)}\}$ produced by Algorithm 1 satisfy

$$\lim_{k\to\infty} \nabla f\big(\vec{\boldsymbol{x}}^{(k)}\big) = \vec{\mathbf{0}}. \tag{6}$$

◁

Proof. Since f is L-smooth, for any $\vec{\boldsymbol{x}}, \vec{\boldsymbol{x}}' \in \mathbb{R}^n$ one has

$$f(\vec{\boldsymbol{x}}') \le f(\vec{\boldsymbol{x}}) + \nabla f(\vec{\boldsymbol{x}})^T(\vec{\boldsymbol{x}}' - \vec{\boldsymbol{x}}) + \frac{L}{2}\|\vec{\boldsymbol{x}}' - \vec{\boldsymbol{x}}\|^2. \tag{7}$$

Apply this inequality with $\vec{\boldsymbol{x}} = \vec{\boldsymbol{x}}^{(k)}$ and $\vec{\boldsymbol{x}}' = \vec{\boldsymbol{x}}^{(k+1)} = \vec{\boldsymbol{x}}^{(k)} - \alpha\,\nabla f\big(\vec{\boldsymbol{x}}^{(k)}\big)$. Then

$$\begin{aligned} f\big(\vec{\boldsymbol{x}}^{(k+1)}\big) &\le f\big(\vec{\boldsymbol{x}}^{(k)}\big) - \alpha\,\|\nabla f\big(\vec{\boldsymbol{x}}^{(k)}\big)\|^2 + \frac{\alpha^2 L}{2}\|\nabla f\big(\vec{\boldsymbol{x}}^{(k)}\big)\|^2 \\ &= f\big(\vec{\boldsymbol{x}}^{(k)}\big) - \underbrace{\left(\alpha - \frac{\alpha^2 L}{2}\right)}_{=:\beta}\|\nabla f\big(\vec{\boldsymbol{x}}^{(k)}\big)\|^2. \end{aligned} \tag{8}$$

Since $\alpha < \frac{2}{L}$, we have $\beta = \alpha - \frac{\alpha^2 L}{2} = \alpha\left(1 - \frac{\alpha L}{2}\right) > 0$. Then,

$$f\big(\vec{\boldsymbol{x}}^{(k+1)}\big) \le f\big(\vec{\boldsymbol{x}}^{(k)}\big) - \beta\|\nabla f\big(\vec{\boldsymbol{x}}^{(k)}\big)\|^2. \tag{9}$$

We can sum these inequalities until iteration k (*accumulation of progress*):

$$f\left(\vec{\boldsymbol{x}}^{(k)}\right) \le f\left(\vec{\boldsymbol{x}}^{(0)}\right) - \beta\sum_{i=0}^{k-1}\left\|\nabla f\left(\vec{\boldsymbol{x}}^{(i)}\right)\right\|^2.$$

Since f is bounded, $m \le f(\vec{\boldsymbol{x}}^{(k)}) \le f(\vec{\boldsymbol{x}}^{(0)}) - \beta\sum_{i=0}^{k-1}\|\nabla f(\vec{\boldsymbol{x}}^{(i)})\|^2$. Therefore $\sum_{i=0}^{k-1}\|\nabla f(\vec{\boldsymbol{x}}^{(i)})\|^2 < \infty$. But this implies that $\lim_{k\to\infty}\|\nabla f\big(\vec{\boldsymbol{x}}^{(k)}\big)\| = 0$, which in turn implies (6). □

Remark 5.

- Using the update inequality (9), the best step size $\alpha^* \in \left(0, \frac{2}{L}\right)$ is obtained by maximizing $\beta(\alpha) = \alpha - \frac{\alpha^2 L}{2}$

$$\beta'(\alpha) = 1 - \alpha L \overset{!}{=} 0 \Rightarrow \alpha^* = \frac{1}{L}.$$

- It is a *global* convergence result, since it is valid for any starting point $\vec{\boldsymbol{x}}^{(0)}$.
- It does NOT necessarily imply that the iterates $\{\vec{\boldsymbol{x}}^{(k)}\}$ converge.
- Often, L is unknown and we need to carefully tune the step size α.

◀

Definition 22 (Global Convergence)**.** An optimization algorithm is globally convergent if, from any starting point $\vec{\boldsymbol{x}}^{(0)}$, its iterates $\vec{\boldsymbol{x}}^{(k)}$ satisfy

$$\nabla f\left(\vec{\boldsymbol{x}}^{(k)}\right) \xrightarrow[k\to\infty]{} \vec{\mathbf{0}}.$$

✍

Definition 23 (Convergence of Iterates)**.** We say that the sequence of iterates $\{\vec{\boldsymbol{x}}^{(k)}\}$ converges to $\vec{\boldsymbol{x}}^*$ if

$$\|\vec{\boldsymbol{x}}^{(k)} - \vec{\boldsymbol{x}}^*\| \xrightarrow[k\to\infty]{} 0.$$

✍

4.3.3 Convergence Rate for Strongly Convex Functions

Lemma 13. If f is differentiable, being μ-strongly convex (Definition 8) is equivalent to

$$! \quad \boxed{f(\vec{\boldsymbol{y}}) \geq f(\vec{\boldsymbol{x}}) + \nabla f(\vec{\boldsymbol{x}})^T(\vec{\boldsymbol{y}} - \vec{\boldsymbol{x}}) + \frac{\mu}{2}\|\vec{\boldsymbol{y}} - \vec{\boldsymbol{x}}\|^2,} \tag{10}$$

and we have the inequality

$$\boxed{2\mu\Big(f(\vec{\boldsymbol{x}}) - f(\vec{\boldsymbol{x}}^*)\Big) \leq \|\nabla f(\vec{\boldsymbol{x}})\|^2, \quad \forall\, \vec{\boldsymbol{x}} \in \mathbb{R}^n,} \tag{11}$$

where $\vec{\boldsymbol{x}}^*$ is the unique global minimizer of f. ◁

Equation (11) says that the gradient norm $\|\nabla f(\vec{\boldsymbol{x}})\|$ grows with the *optimality error* $f(\vec{\boldsymbol{x}}) - f(\vec{\boldsymbol{x}}^*)$. This allows us to quantify how close we are from the global minimum by looking at the gradient norm.

Proof. Rearranging equation (10) with $\vec{\boldsymbol{y}} = \vec{\boldsymbol{x}}^*$ yields

rearrange with $\vec{\boldsymbol{y}} = \vec{\boldsymbol{x}}^*$

$$\begin{aligned} f(\vec{\boldsymbol{x}}) - f(\vec{\boldsymbol{x}}^*) &\leq -\nabla f(\vec{\boldsymbol{x}})^T(\vec{\boldsymbol{x}}^* - \vec{\boldsymbol{x}}) - \frac{\mu}{2}\|\vec{\boldsymbol{x}}^* - \vec{\boldsymbol{x}}\|^2 \\ 2\mu\left(f(\vec{\boldsymbol{x}}) - f(\vec{\boldsymbol{x}}^*)\right) &\leq -2\mu\nabla f(\vec{\boldsymbol{x}})^T(\vec{\boldsymbol{x}}^* - \vec{\boldsymbol{x}}) - \mu^2\|\vec{\boldsymbol{x}}^* - \vec{\boldsymbol{x}}\|^2. \end{aligned}$$

add 0 on the right

Adding $0 = \|\nabla f(\vec{\boldsymbol{x}})\|^2 - \|\nabla f(\vec{\boldsymbol{x}})\|^2$ to the right-hand side, allows us to rewrite the last line as

$$\begin{aligned} 2\mu\left(f(\vec{\boldsymbol{x}}) - f(\vec{\boldsymbol{x}}^*)\right) &\leq \|\nabla f(\vec{\boldsymbol{x}})\|^2 - \|\nabla f(\vec{\boldsymbol{x}})\|^2 - 2\mu\nabla f(\vec{\boldsymbol{x}})^T(\vec{\boldsymbol{x}}^* - \vec{\boldsymbol{x}}) - \mu^2\|\vec{\boldsymbol{x}}^* - \vec{\boldsymbol{x}}\|^2 \\ &= \|\nabla f(\vec{\boldsymbol{x}})\|^2 - \left(\|\nabla f(\vec{\boldsymbol{x}})\|^2 + 2\mu\nabla f(\vec{\boldsymbol{x}})^T(\vec{\boldsymbol{x}}^* - \vec{\boldsymbol{x}}) + \mu^2\|\vec{\boldsymbol{x}}^* - \vec{\boldsymbol{x}}\|^2\right) \\ &= \|\nabla f(\vec{\boldsymbol{x}})\|^2 - \|\nabla f(\vec{\boldsymbol{x}}) + \mu(\vec{\boldsymbol{x}}^* - \vec{\boldsymbol{x}})\|^2 \\ &\leq \|\nabla f(\vec{\boldsymbol{x}})\|^2. \end{aligned}$$

□

Theorem 14. Let f be L-smooth and μ-strongly convex with unique minimizer $\vec{\boldsymbol{x}}^*$. If the step size satisfies $\alpha \in \left(0, \frac{1}{L}\right]$, then for every iteration $k \geq 0$ of gradient descent,

$$f\left(\vec{\boldsymbol{x}}^{(k+1)}\right) - f\left(\vec{\boldsymbol{x}}^*\right) \leq (1 - \alpha\mu)\left(f\left(\vec{\boldsymbol{x}}^{(k)}\right) - f\left(\vec{\boldsymbol{x}}^*\right)\right).$$

◁

Proof. From the smoothness of f we have already derived (Equation 8) that

$$f\left(\vec{\boldsymbol{x}}^{(k+1)}\right) \leq f\left(\vec{\boldsymbol{x}}^{(k)}\right) - \alpha\left(1 - \tfrac{\alpha L}{2}\right)\|\nabla f\left(\vec{\boldsymbol{x}}^{(k)}\right)\|^2.$$

Since $\alpha \leq \frac{1}{L}$, we have $\left(1 - \frac{\alpha L}{2}\right) \geq \left(1 - \frac{1}{2}\right) = \frac{1}{2}$ and therefore

$$f\left(\vec{\boldsymbol{x}}^{(k+1)}\right) - f\left(\vec{\boldsymbol{x}}^*\right) \leq f\left(\vec{\boldsymbol{x}}^{(k)}\right) - f\left(\vec{\boldsymbol{x}}^*\right) - \frac{\alpha}{2}\|\nabla f\left(\vec{\boldsymbol{x}}^{(k)}\right)\|^2.$$

Using Equation (11) from Lemma 13, $2\mu\left(f\left(\vec{\boldsymbol{x}}^{(k)}\right) - f\left(\vec{\boldsymbol{x}}^*\right)\right) \leq \|\nabla f\left(\vec{\boldsymbol{x}}^{(k)}\right)\|^2$, we obtain

$$f\left(\vec{\boldsymbol{x}}^{(k+1)}\right) - f\left(\vec{\boldsymbol{x}}^*\right) \leq f\left(\vec{\boldsymbol{x}}^{(k)}\right) - f\left(\vec{\boldsymbol{x}}^*\right) - \alpha\mu\Big(f\left(\vec{\boldsymbol{x}}^{(k)}\right) - f\left(\vec{\boldsymbol{x}}^*\right)\Big),$$

which simplifies to the stated result. □

Theorem 14 shows that the error in function value decreases by a factor of $1 - \alpha\mu$ at each iteration. In other words, gradient descent converges *linearly* with rate $C = 1 - \alpha\mu$.

Definition 24 (Linear Convergence)**.** Let $\{z^{(k)}\}$ be a sequence with $z^{(k)} \to z^*$ and there exists $C \in [0, 1)$ and $\hat{k}$ such that

$$\|z^{(k+1)} - z^*\| \leq C \|z^{(k)} - z^*\|,$$

for all $k \geq \hat{k}$. Then the sequence converges *linearly* to z^* with rate C. ✍

Remark 6.

- The smaller the linear rate C, the faster is the convergence.
- The rate $1 - \alpha\mu$ is the "worst-case" rate, but it can be faster for certain starting points.
- The best theoretical rate is when $1 - \alpha\mu$ is smallest, i.e., $\alpha^* = \frac{1}{L}$ (we do not always know L!)
- Linear rate is generally considered slow, but unfortunately this is the best "worst-case" rate a first-order method can generally achieve!
- In fact if f is only convex, the worst-case rate is sub-linear, i.e., slower than any linear rate.

◀

For strongly convex and smooth objective functions, we can also determine how many iterations we need until $f\left(\vec{\boldsymbol{x}}^{(k)}\right) - f\left(\vec{\boldsymbol{x}}^*\right) \leq \epsilon$ for a given ϵ.

Since the error decreases

$$f\left(\vec{\boldsymbol{x}}^{(k)}\right) - f\left(\vec{\boldsymbol{x}}^*\right) \leq \Big(1 - \alpha\mu\Big)^k \Big(f\left(\vec{\boldsymbol{x}}^{(0)}\right) - f\left(\vec{\boldsymbol{x}}^*\right)\Big),$$

then to guarantee that

$$f\left(\vec{\boldsymbol{x}}^{(k)}\right) - f\left(\vec{\boldsymbol{x}}^*\right) \leq \epsilon,$$

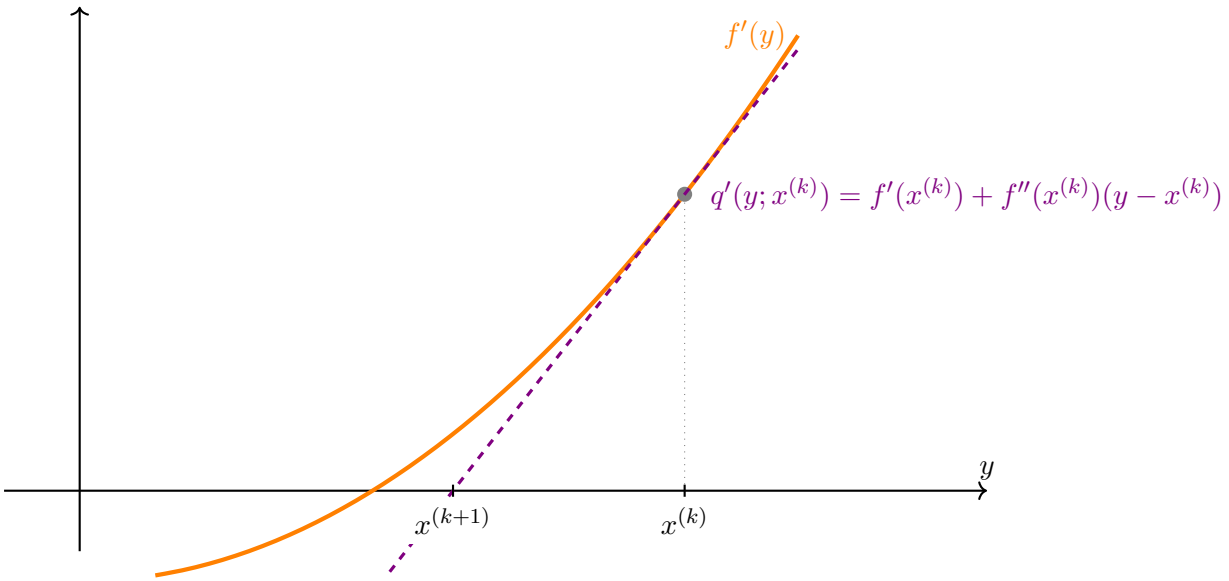
it suffices to have

$$k \geq \frac{1}{-\log(1 - \alpha\mu)} \log\left(\frac{f\left(\vec{\boldsymbol{x}}^{(0)}\right) - f\left(\vec{\boldsymbol{x}}^*\right)}{\epsilon}\right),$$

i.e. the number of iterations is $O\left(\log \frac{1}{\epsilon}\right)$.

4.4 Newton's Method

4.4.1 Newton's Method for Univariate Functions



Recall (Equation 4) the second-order Taylor approximation for a univariate f. Since $f \approx q(y; x^{(k)})$, it makes sense to choose $x^{(k+1)}$ as the minimizer of $q(y; x^{(k)})$, i.e.

$$x^{(k+1)} = \arg\min_y q(y; x^{(k)}) \Rightarrow q'(y; x^{(k)}) = f'(x^{(k)}) + f''(x^{(k)})(y - x^{(k)}) \stackrel{!}{=} 0$$

$$\implies \quad x^{(k+1)} = x^{(k)} - \frac{f'(x^{(k)})}{f''(x^{(k)})} \tag{12}$$

and if $q''(x^{(k+1)}; x^{(k)}) = f''(x^{(k)}) > 0$, then $x^{(k+1)}$ is the global minimizer of $q(\,\cdot\,; x^{(k)})$, since the latter is convex. The direction

$$d_N = -\frac{f'\left(x^{(k)}\right)}{f''\left(x^{(k)}\right)}$$

is called the Newton's direction and the step (12) called the Newton's step (see Algorithm 2).

Algorithm 2 Newton's Algorithm

1: **Input:** Starting point $x^{(0)}$, tolerance $\epsilon > 0$, and maximum iterations $k_{\max}$
2: Set $k \leftarrow 0$
3: **while** $\left|f'(x^{(k)})\right| > \epsilon$ and $k \leq k_{\max}$ **do**
4: Compute $f'(x^{(k)})$ and $f''(x^{(k)})$
5: Update: $x^{(k+1)} = x^{(k)} - \frac{f'(x^{(k)})}{f''(x^{(k)})}$
6: Set $k \leftarrow k + 1$
7: **end while**
8: **Output:** $x^{(k)}$

Remark 7. Newton's algorithm does *not* work when $f''(x^{(k)}) \leq 0$, but there are possible remedies. If f is convex, then $f''(x) \geq 0$ for any x, and if $f''(x^{(k)}) > 0$, the Newton's step is well defined. ◀

Remark 8. Originally, Newton's algorithm is a method for finding the root of a differentiable function, i.e., finding x such that $g(x) = 0$. The form of the iterates comes from the first-order Taylor approximation:

$$g(y) \approx t(y; x^{(k)}) = g(x^{(k)}) + g'(x^{(k)})(y - x^{(k)})$$

The next iterate $x^{(k+1)}$ is chosen as the root of $t(y; x^{(k)})$, i.e., $t(x^{(k+1)}; x^{(k)}) = 0$, which gives

$$x^{(k+1)} = x^{(k)} - \frac{g(x^{(k)})}{g'(x^{(k)})}$$

In optimzation, we use Newton's method to find a stationary point (which *may* be a local minimiser and *is* a global minimiser if f is convex). Therefore, we use Newton's method to find a root of $g = f'$, which corresponds to the update

$$x^{(k+1)} = x^{(k)} - \frac{f'(x^{(k)})}{f''(x^{(k)})}$$

which is the same as (12), obtained from minimizing the second-order Taylor approximation. If the objective is not convex, we need to check that the stationary point found by Newton's method is a local minimizer. ◀

4.4.2 Second Order Taylor's Expansion for Multivariate Functions

Assume $f : \mathbb{R}^n \to \mathbb{R}$ is twice continuously differentiable. The second-order Taylor approximation is

$$\boxed{f(\vec{\boldsymbol{y}}) \approx q(\vec{\boldsymbol{y}}; \vec{\boldsymbol{x}}) = f(\vec{\boldsymbol{x}}) + \nabla f(\vec{\boldsymbol{x}})^T(\vec{\boldsymbol{y}} - \vec{\boldsymbol{x}}) + \frac{1}{2}(\vec{\boldsymbol{y}} - \vec{\boldsymbol{x}})^T \underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}})(\vec{\boldsymbol{y}} - \vec{\boldsymbol{x}}),} \tag{13}$$

where $\underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}})$ is the Hessian matrix defined as

$$\underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}}) = \left(\frac{\partial^2 f}{\partial x_i \partial x_j}\right)_{i,j\in\{1,\dots,n\}} = \begin{pmatrix} \frac{\partial^2 f}{\partial x_1^2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \cdots & \frac{\partial^2 f}{\partial x_n^2} \end{pmatrix} \in \mathbb{R}^{n\times n}.$$

Since f is twice continuously differentiable, the Hessian is symmetric, i.e., $\underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}}) = \underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}})^T$.

We have already seen the first order Taylor's expansion for multivariate functions (Theorem 9, Corollary 10). Now let's extend this to the second order:

Theorem 15 (Second Order Taylor's Expansion for Multivariate Functions)**.** If $f : \mathbb{R}^n \to \mathbb{R}$ is continuously differentiable, then for any $\vec{\boldsymbol{x}} \in \mathbb{R}^n$ and $\vec{\boldsymbol{d}} \in \mathbb{R}^n$ there exists $t \in (0, 1)$ such that

$$f(\vec{\boldsymbol{x}} + \vec{\boldsymbol{d}}) = f(\vec{\boldsymbol{x}}) + \nabla f(\vec{\boldsymbol{x}} + t\,\vec{\boldsymbol{d}})^T \vec{\boldsymbol{d}}.$$

Moreover, if f is twice continuously differentiable, then for any $\vec{\boldsymbol{x}} \in \mathbb{R}^n$ and $\vec{\boldsymbol{d}} \in \mathbb{R}^n$ there exists $t \in (0, 1)$ such that

$$f(\vec{\boldsymbol{x}} + \vec{\boldsymbol{d}}) = f(\vec{\boldsymbol{x}}) + \nabla f(\vec{\boldsymbol{x}})^T \vec{\boldsymbol{d}} + \frac{1}{2}\vec{\boldsymbol{d}}^T \underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}} + t\,\vec{\boldsymbol{d}})\vec{\boldsymbol{d}}.$$

For $n = 1$, this reduces to the univariate case (Theorem 2). ◁

Corollary 16. If f is three times continuously differentiable, then for any $\vec{\boldsymbol{x}} \in \mathbb{R}^n$ there exists $L > 0$ and $\epsilon > 0$ such that

$$|f(\vec{\boldsymbol{y}}) - q(\vec{\boldsymbol{y}}; \vec{\boldsymbol{x}})| \le L\|\vec{\boldsymbol{y}} - \vec{\boldsymbol{x}}\|^3,$$

for any $\vec{\boldsymbol{y}} \in \mathbb{R}^n$ such that $\|\vec{\boldsymbol{x}} - \vec{\boldsymbol{y}}\| \le \epsilon$. For $n = 1$, this reduces to the univariate case (Corollary 3). ◁

4.4.3 Second Order Optimality Conditions for Multivariate Functions

Theorem 17 (Second Order Necessary Optimality Condition)**.** If f is twice continuously differentiable and $\vec{\boldsymbol{x}}^*$ is a local minimizer of f, then $\underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}}^*) \succeq 0$, i.e., $\vec{\boldsymbol{p}}^T \underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}}^*)\vec{\boldsymbol{p}} \geq 0$ for all $\vec{\boldsymbol{p}} \in \mathbb{R}^n$. ◁

Proof. (Contradiction) Assume $\vec{\boldsymbol{x}}^*$ is a local minimizer of f and $\underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}}^*) \not\succeq 0$. Then there exists $\vec{\boldsymbol{p}} \in \mathbb{R}^n$ such that $\vec{\boldsymbol{p}}^T \underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}}^*)\vec{\boldsymbol{p}} < 0$. Since $\underline{\boldsymbol{H}}_f(\,\cdot\,)$ is continuous there exists $t_0 > 0$ such that for all $t \in (0, t_0)$,

$$\vec{\boldsymbol{p}}^T \underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}}^* + t\vec{\boldsymbol{p}})\vec{\boldsymbol{p}} < 0.$$

Let $t \in (0, t_0)$. If $\nabla f(\vec{\boldsymbol{x}}^*) \neq 0$ we already know that $\vec{\boldsymbol{x}}^*$ cannot be a local minimizer (Theorem 11), so we assume $\nabla f(\vec{\boldsymbol{x}}^*) = 0$. Theorem 15 says there exists $\bar{t} \in (0, t)$ such that

$$f(\vec{\boldsymbol{x}}^* + t\vec{\boldsymbol{p}}) = f(\vec{\boldsymbol{x}}^*) + \frac{1}{2}t^2 \vec{\boldsymbol{p}}^T \underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}}^* + \bar{t}\vec{\boldsymbol{p}})\vec{\boldsymbol{p}},$$

and since $0 < \bar{t} < t < t_0$, we have $\vec{\boldsymbol{p}}^T \underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}}^* + \bar{t}\vec{\boldsymbol{p}})\vec{\boldsymbol{p}} < 0$, which implies $f(\vec{\boldsymbol{x}}^* + t\vec{\boldsymbol{p}}) < f(\vec{\boldsymbol{x}}^*)$, contradicting the assumption that $\vec{\boldsymbol{x}}^*$ is a local minimizer. □

Theorem 18 (Sufficient Optimality Conditions)**.** If f is twice continuously differentiable and $\vec{\boldsymbol{x}}^*$ is such that $\nabla f(\vec{\boldsymbol{x}}^*) = \vec{\boldsymbol{0}}$ and $\underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}}^*) \succ 0$, then $\vec{\boldsymbol{x}}^*$ is a strict local minimizer of f. ◁

Proof. Since $\underline{\boldsymbol{H}}_f(\,\cdot\,)$ is continuous, there exists $r > 0$ such that for any $\vec{\boldsymbol{p}}$ with $\|\vec{\boldsymbol{p}} - \vec{\boldsymbol{x}}^*\| < r$, we have $\underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}}^* + \vec{\boldsymbol{p}}) \succ 0$. If $\vec{\boldsymbol{p}} \neq \vec{\boldsymbol{0}}$, then $\vec{\boldsymbol{p}}^T \underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}}^* + \vec{\boldsymbol{p}})\vec{\boldsymbol{p}} > 0$, and by Theorem 15 there exists $t \in (0, 1)$ such that

$$f(\vec{\boldsymbol{x}}^* + \vec{\boldsymbol{p}}) = f(\vec{\boldsymbol{x}}^*) + \frac{1}{2}\vec{\boldsymbol{p}}^T \underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}}^* + t\vec{\boldsymbol{p}})\vec{\boldsymbol{p}} > f(\vec{\boldsymbol{x}}^*),$$

which shows that $\vec{\boldsymbol{x}}^*$ is a strict local minimizer. □

4.4.4 Minimization of Quadratic Functions

For a multivariate function f, at iteration k of Newton's method (Algorithm 3), we need to find the minimizer $\vec{\boldsymbol{x}}^*$ of the second-order Taylor approximation (Equation 13) at $\vec{\boldsymbol{x}}^{(k)}$ and define the new iterate as $\vec{\boldsymbol{x}}^{(k+1)} = \vec{\boldsymbol{x}}^*$. Equation 13 is a multivariate quadratic function of the form

$$q(\vec{\boldsymbol{x}}) = b + \vec{\boldsymbol{g}}^T \vec{\boldsymbol{x}} + \vec{\boldsymbol{x}}^T \underline{\boldsymbol{Q}} \vec{\boldsymbol{x}} \tag{14}$$

$$= b + \sum_{i=1}^{n} g_i x_i + \sum_{i=1}^{n}\sum_{j=1}^{n} q_{ij} x_i x_j, \tag{15}$$

where $\vec{\boldsymbol{x}} = \vec{\boldsymbol{y}} - \vec{\boldsymbol{x}}^{(k)}$, $b = f(\vec{\boldsymbol{x}}^{(k)})$, $\vec{\boldsymbol{g}} = \nabla f(\vec{\boldsymbol{x}}^{(k)})$, and $\underline{\boldsymbol{Q}} = \frac{1}{2}\underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}}^{(k)})$. We assume that $\underline{\boldsymbol{Q}}$ is symmetric, which is not restricive, since any quadratic model with a non-symmetric matrix can be re-written with a symmetric matrix $\tilde{\underline{\boldsymbol{Q}}} = \frac{1}{2}(\underline{\boldsymbol{Q}} + \underline{\boldsymbol{Q}}^T)$.

Using Equation (15), it is easier to compute the derivatives:

$$\frac{\partial}{\partial x_k} q(\vec{\boldsymbol{x}}) = \sum_{i=1}^{n} \frac{\partial}{\partial x_k} g_i x_i + \sum_{i=1}^{n}\sum_{j=1}^{n} \frac{\partial}{\partial x_k} q_{ij} x_i x_j = g_k + 2\sum_{j=1}^{n} q_{kj} x_j,$$

$$\frac{\partial^2}{\partial x_k \partial x_l} q(\vec{\boldsymbol{x}}) = 2q_{kl},$$

or in vectorized form:

$$\nabla q(\vec{\boldsymbol{x}}) = \vec{\boldsymbol{g}} + 2\underline{\boldsymbol{Q}}\vec{\boldsymbol{x}},$$
$$\underline{\boldsymbol{H}}_q(\vec{\boldsymbol{x}}) = 2\underline{\boldsymbol{Q}}.$$

A stationary point of q thus satisfies

$$\nabla q(\vec{\boldsymbol{x}}^*) = \vec{\boldsymbol{0}} \quad \Longleftrightarrow \quad 2\underline{\boldsymbol{Q}}\vec{\boldsymbol{x}}^* = -\vec{\boldsymbol{g}}$$

so $\vec{\boldsymbol{x}}^*$ is the solution of a linear system of equations. If $\underline{\boldsymbol{Q}}$ is invertible, then $\vec{\boldsymbol{x}}^* = -\frac{1}{2}\underline{\boldsymbol{Q}}^{-1}\vec{\boldsymbol{g}}$.

Remark 9. It is always less computationally expensive to solve a linear system $\underline{\boldsymbol{A}}\vec{\boldsymbol{x}} = \vec{\boldsymbol{b}}$ than inverting $\underline{\boldsymbol{A}}$, i.e., computing $\underline{\boldsymbol{A}}^{-1}\vec{\boldsymbol{b}}$. ◀

Observe that for quadratic functions to be a local minimizer the

- necessary second-order optimality condition is $\underline{\boldsymbol{Q}} \succeq 0$,
- sufficient second-order optimality condition is $\underline{\boldsymbol{Q}} \succ 0 \wedge 2\underline{\boldsymbol{Q}}\vec{\boldsymbol{x}}^* = -\vec{\boldsymbol{g}}$.

Caution 10. Some important matrix properties. Let $\underline{\boldsymbol{A}} \in \mathbb{R}^{n\times n}$.

- The determinant $\det(\underline{\boldsymbol{A}})$ is equal to the product of the eigenvalues of $\underline{\boldsymbol{A}}$.
- The trace $\mathrm{tr}(\underline{\boldsymbol{A}})$ is equal to the sum of the eigenvalues of $\underline{\boldsymbol{A}}$.
- $\underline{\boldsymbol{A}}$ is positive definite ($\underline{\boldsymbol{A}} \succ 0$) if and only if it is symmetric and all its eigenvalues are positive.
- $\underline{\boldsymbol{A}}$ is positive semi-definite ($\underline{\boldsymbol{A}} \succeq 0$) if and only if it is symmetric and all its eigenvalues are non-negative.
- $\underline{\boldsymbol{A}}$ is negative definite ($\underline{\boldsymbol{A}} \prec 0$) if and only if it is symmetric and all its eigenvalues are negative.
- $\underline{\boldsymbol{A}}$ is negative semi-definite ($\underline{\boldsymbol{A}} \preceq 0$) if and only if it is symmetric and all its eigenvalues are non-positive.
- If $\underline{\boldsymbol{A}}$ is symmetric and has both positive and negative eigenvalues, then it is indefinite.

◀

Now consider $\vec{\boldsymbol{x}}^*$ a stationary point of $q(\vec{\boldsymbol{x}}) = b + \vec{\boldsymbol{g}}^T\vec{\boldsymbol{x}} + \vec{\boldsymbol{x}}^T\underline{\boldsymbol{Q}}\vec{\boldsymbol{x}}$, i.e. a point satisfying $2\underline{\boldsymbol{Q}}\vec{\boldsymbol{x}}^* = -\vec{\boldsymbol{g}}$.

- If $\underline{\boldsymbol{Q}} \succ 0$, then q is strictly convex and $\vec{\boldsymbol{x}}^*$ is the global minimizer of q.
- If $\underline{\boldsymbol{Q}} \prec 0$, then q is strictly concave and $\vec{\boldsymbol{x}}^*$ is the global maximizer of q.
- If $\underline{\boldsymbol{Q}}$ has at least one negative eigenvalue then q is unbounded below and has no minimizer.
- If $\underline{\boldsymbol{Q}}$ is indefinite, $\vec{\boldsymbol{x}}^*$ is a saddle point of q.
- If $\underline{\boldsymbol{Q}} \succeq 0$, then q is convex. If $\underline{\boldsymbol{Q}}$ is not positive definite, then it is singular and in this case q is either unbounded or has an infinite number of global minimizers.

4.4.5 Newton's Method for Multivariate Functions

Algorithm 3 Newton's Algorithm (Multivariate)

1: **Input:** Starting point $\vec{\boldsymbol{x}}^{(0)}$, tolerance $\epsilon > 0$, and maximum iterations $k_{\max}$
2: Set $k \leftarrow 0$
3: **while** $\left|\nabla f(\vec{\boldsymbol{x}}^{(k)})\right| > \epsilon$ and $k \leq k_{\max}$ **do**
4: Compute $\nabla f(\vec{\boldsymbol{x}}^{(k)})$ and $\underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}}^{(k)})$
5: Compute $\vec{\boldsymbol{u}}_{\min}$ as the minimizer of $q(\vec{\boldsymbol{u}}; \vec{\boldsymbol{x}}^{(k)})$. ▷ If it exists
6: Update: $\vec{\boldsymbol{x}}^{(k+1)} = \vec{\boldsymbol{u}}_{\min}$ ▷ If it exists: $\vec{\boldsymbol{x}}^{(k+1)} = \vec{\boldsymbol{x}}^{(k)} - (\underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}}^{(k)})^{-1}\nabla f(\vec{\boldsymbol{x}}^{(k)})$
7: Set $k \leftarrow k + 1$
8: **end while**
9: **Output:** $\vec{\boldsymbol{x}}^{(k)}$

In Line 5 of Algorithm 3, we need to find the minimizer of the quadratic local model at $\vec{\boldsymbol{x}}^{(k)}$ given by the second-order Taylor approximation (Section 4.4.2, Equation 13). If we set $\vec{\boldsymbol{y}} = \vec{\boldsymbol{x}}^{(k)} + \vec{\boldsymbol{d}}$ in (13), we obtain

$$q(\vec{\boldsymbol{d}}; \vec{\boldsymbol{x}}^{(k)}) = f(\vec{\boldsymbol{x}}^{(k)}) + \nabla f(\vec{\boldsymbol{x}}^{(k)})^T \vec{\boldsymbol{d}} + \frac{1}{2}\vec{\boldsymbol{d}}^T \underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}}^{(k)})\vec{\boldsymbol{d}}.$$

Denoting $b = f(\vec{\boldsymbol{x}}^{(k)})$, $\vec{\boldsymbol{g}} = \nabla f(\vec{\boldsymbol{x}}^{(k)})$, and $\underline{\boldsymbol{Q}} = \frac{1}{2}\underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}}^{(k)})$, we have a quadratic function of the same form as in (14):

$$q(\vec{\boldsymbol{d}}; \vec{\boldsymbol{x}}^{(k)}) = b + \vec{\boldsymbol{g}}^T \vec{\boldsymbol{d}} + \vec{\boldsymbol{d}}^T \underline{\boldsymbol{Q}}\vec{\boldsymbol{d}}.$$

Thus a stationary point of q satisfies

$$\vec{\boldsymbol{d}}^* = -\frac{1}{2}\underline{\boldsymbol{Q}}^{-1}\vec{\boldsymbol{g}} = -\left(\underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}}^{(k)})\right)^{-1}\nabla f(\vec{\boldsymbol{x}}^{(k)}).$$

If $\underline{\boldsymbol{Q}} \succ 0$, then $\vec{\boldsymbol{d}}^*$ is the global minimizer of $q(\vec{\boldsymbol{d}}; \vec{\boldsymbol{x}}^{(k)})$, and is the *Newton's direction*, denoted $\vec{\boldsymbol{d}}_N$. Hence in the original problem with $\vec{\boldsymbol{y}} = \vec{\boldsymbol{x}}^{(k)} + \vec{\boldsymbol{d}}$, the minimizer $\vec{\boldsymbol{x}}^{k+1}$ is

$$\vec{\boldsymbol{x}}^{(k+1)} = \vec{\boldsymbol{x}}^{(k)} + \vec{\boldsymbol{d}}_N = \vec{\boldsymbol{x}}^{(k)} - \left(\underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}}^{(k)})\right)^{-1}\nabla f(\vec{\boldsymbol{x}}^{(k)}).$$

This update rule is called the *Newton's step*.

Caution 11. Two important remarks:

- If $\underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}}^{(k)})$ is not positive definite, then we cannot compute the Newton's direction. We need to detect this case by checking if all eigenvalues of $\underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}}^{(k)})$ are positive. If they are not, an easy solution is to take a gradient step for that iteration.
- To compute $\vec{\boldsymbol{d}}_N$, we should not invert $\underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}}^{(k)})$, but solve the linear system

  $$2\underline{\boldsymbol{Q}}\vec{\boldsymbol{d}}_N = -\vec{\boldsymbol{g}}, \text{ i.e., } \underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}}^{(k)})\vec{\boldsymbol{d}}_N = -\nabla f(\vec{\boldsymbol{x}}^{(k)})$$

  for $\vec{\boldsymbol{d}}_N$.

◀

4.4.6 Convergence of Newton's Method

Theorem 19 (Convergence of Newton's Method)**.** If $f : \mathbb{R}^n \to \mathbb{R}$ is twice continuously differentiable, $\vec{\boldsymbol{x}}^*$ is a point such that $\nabla f(\vec{\boldsymbol{x}}^*) = \vec{\mathbf{0}} \wedge \underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}}^*) \succ 0$, and $\underline{\boldsymbol{H}}_f(\,\cdot\,)$ is L-Lipschitz continuous with $L > 0$ in a neighborhood of $\vec{\boldsymbol{x}}^*$, then there exists $\epsilon > 0$ such that for any $\vec{\boldsymbol{x}}^{(0)}$ with $\|\vec{\boldsymbol{x}}^{(0)} - \vec{\boldsymbol{x}}^*\| \leq \epsilon$, at each iteration $k \geq 0$ of Newton's method (Algorithm 3),

$$\left\|\vec{\boldsymbol{x}}^{(k+1)} - \vec{\boldsymbol{x}}^*\right\| \leq L \left\|\left(\underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}}^*)\right)^{-1}\right\| \left\|\vec{\boldsymbol{x}}^{(k)} - \vec{\boldsymbol{x}}^*\right\|^2, \tag{16}$$

i.e., the iterates $\{\vec{\boldsymbol{x}}^{(k)}\}$ converge quadratically to $\vec{\boldsymbol{x}}^*$. ◁

Remark 12. Theorem 19 shows that if Algorithm 3 is initialized sufficiently close to a local minimizer $\vec{\boldsymbol{x}}^*$ at which the second-order sufficient optimality conditions (Theorem 18) hold, then the iterates converge quadratically to $\vec{\boldsymbol{x}}^*$. It is thus a *local convergence* result and not a global convergence result. ◀

Definition 25 (Convergence Rates)**.** A sequence of vectors $\{\vec{\boldsymbol{z}}^{(k)}\}$ with $\vec{\boldsymbol{z}}^{(k)} \to \vec{\boldsymbol{z}}^*$ converges

- *linearly* if there exists $c \in (0, 1)$ and $\hat{k} \geq 0$ such that
$$\|\vec{\boldsymbol{z}}^{(k+1)} - \vec{\boldsymbol{z}}^*\| \leq c\|\vec{\boldsymbol{z}}^{(k)} - \vec{\boldsymbol{z}}^*\| \text{ for all } k \geq \hat{k}.$$
- *quadratically* if there exists $c \in (0, \infty)$ and $\hat{k} \geq 0$ such that
$$\|\vec{\boldsymbol{z}}^{(k+1)} - \vec{\boldsymbol{z}}^*\| \leq c\|\vec{\boldsymbol{z}}^{(k)} - \vec{\boldsymbol{z}}^*\|^2 \text{ for all } k \geq \hat{k}.$$
- *sublinearly* if there exists $\{c^{(k)}\} \subset (0, 1)$ with $c^{(k)} \to 1$ and $\hat{k} \geq 0$ such that
$$\|\vec{\boldsymbol{z}}^{(k+1)} - \vec{\boldsymbol{z}}^*\| \leq c^{(k)}\|\vec{\boldsymbol{z}}^{(k)} - \vec{\boldsymbol{z}}^*\| \text{ for all } k \geq \hat{k}.$$
- *superlinearly* if there exists $\{c^{(k)}\} \subset (0, \infty)$ with $c^{(k)} \to 0$ and $\hat{k} \geq 0$ such that
$$\|\vec{\boldsymbol{z}}^{(k+1)} - \vec{\boldsymbol{z}}^*\| \leq c^{(k)}\|\vec{\boldsymbol{z}}^{(k)} - \vec{\boldsymbol{z}}^*\| \text{ for all } k \geq \hat{k}.$$

✍

Remark 13. Observe that: quadratic $\implies$ superlinear $\implies$ linear $\implies$ sublinear. ◀

Theorem 19 says that when Algorithm 3 is initialized sufficiently close to the local minimizer x^*, we have $C = \|(\underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}}^*))^{-1}\| > 0$ such that

$$\|x^{(k+1)} - x^*\| \leq C\,\|x^{(k)} - x^*\|^2. \tag{17}$$

Denoting the error at iteration k by

$$\texttt{err}_k = \|x^{(k)} - x^*\|,$$

we then can rewrite (17) as

$$\texttt{err}_{k+1} \leq C\,(\texttt{err}_k)^2. \tag{18}$$

Recursively applying this estimate leads to

$$\texttt{err}_K \le C\Bigg(\underbrace{C\Big(\cdots C\Big(\underbrace{C\texttt{err}_0^2}_{\ge \texttt{err}_1}\Big)^2\cdots\Big)^2}_{\ge \texttt{err}_2}\Bigg)^2 = C^{2^K-1}\left(\texttt{err}_0\right)^{2^K}.$$
$$\underbrace{\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad}_{\ge \texttt{err}_{K-1}}$$

For simplicity, assume $\texttt{err}_0 = 1$. To achieve an error $\texttt{err}_K \le \epsilon$ with $0 < \epsilon \ll 1$, it suffices that

$$C^{2^K-1} \le \epsilon.$$

Taking logarithms gives

$$\left(2^K - 1\right)\log C \le \log \epsilon.$$

Since $\log C < 0$ (because $C < 1$ in the quadratic regime), we can rearrange to obtain

$$2^K \ge 1 + \frac{\log \epsilon}{\log C}$$

and taking logarithms once more yields

$$K \ge \log\left(1 + \frac{\log \epsilon^{-1}}{\log C^{-1}}\right) = O\Big(\log\log \epsilon^{-1}\Big).$$

Thus, to reduce the optimality error to order ϵ we need only $O(\log\log \epsilon^{-1})$ iterations. This is significantly fewer than the $O(\log \epsilon^{-1})$ iterations typically required by gradient descent.

In its standard form, Algorithm 3 is not globally convergent, but it can be made so with a slight variation of the update rule using *line search*. Also, computing second-order derivatives is computationally expensive, but also this can be avoided by using *quasi-Newton methods* while keeping some of the benefits of Newton's method.

4.5 Line Search Algorithms

To make Newton's algorithm globally convergent from any starting point we need more flexibility in the update rule. One possibility is to take a smaller step in the same direction, i.e.,

$$\vec{\boldsymbol{x}}^{(k+1)} = \vec{\boldsymbol{x}}^{(k)} + \alpha\,\vec{\boldsymbol{d}}_N = \vec{\boldsymbol{x}}^{(k)} + \alpha\underbrace{\left(-\left(\underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}}^{(k)})\right)^{-1}\nabla f(\vec{\boldsymbol{x}}^{(k)})\right)}_{\vec{\boldsymbol{d}}_N},$$

with $\alpha \in (0,1)$ a step size. This is similar to Gradient Descent where instead of $\vec{\boldsymbol{d}}_N$, we have $\vec{\boldsymbol{d}}_S = -\nabla f(\vec{\boldsymbol{x}}^{(k)})$. In fact, both Gradient Descent and Newton's algorithms (and their variants) fall in the class of *line search* algorithms where finding a "good" step size is a sub-routine. A line search method has an update rule of the form

$$\vec{\boldsymbol{x}}^{(k+1)} = \vec{\boldsymbol{x}}^{(k)} + \alpha^{(k)}\,\vec{\boldsymbol{d}}^{(k)},$$

where:

- $\vec{\boldsymbol{d}}^{(k)} \in \mathbb{R}^n$ is a search direction;
- $\alpha^{(k)} > 0$ is a chosen step size.

Performing a line search specifically refers to the sub-routine of choosing the best possible (or at least, good enough) $\alpha^{(k)}$, given $\vec{\boldsymbol{d}}^{(k)}$. At a minimum we require that we achieve a decrease of f, i.e.,

$$f\big(\vec{\boldsymbol{x}}^{(k+1)}\big) = f\big(\vec{\boldsymbol{x}}^{(k)} + \alpha^{(k)}\,\vec{\boldsymbol{d}}^{(k)}\big) < f\big(\vec{\boldsymbol{x}}^{(k)}\big).$$

This is always satisfied for sufficiently small $\alpha^{(k)}$ if $\vec{\boldsymbol{d}}^{(k)}$ is a descent direction. Recall that if we define

$$g(\alpha) = f(\vec{\boldsymbol{x}}^{(k)} + \alpha\,\vec{\boldsymbol{d}}^{(k)}),$$

then its derivative is $g'(\alpha) = \nabla f(\vec{\boldsymbol{x}}^{(k)} + \alpha\,\vec{\boldsymbol{d}}^{(k)})^T\vec{\boldsymbol{d}}^{(k)}$, with $g'(0) = \nabla f(\vec{\boldsymbol{x}}^{(k)})^T\vec{\boldsymbol{d}}^{(k)}$. Since

$$g(\alpha) \approx f\big(\vec{\boldsymbol{x}}^{(k)}\big) + g'(0)\,\alpha,$$

we can guarantee $g(\alpha) < f\big(\vec{\boldsymbol{x}}^{(k)}\big)$ for sufficiently small $\alpha > 0$ if

$$g'(0) = \nabla f\big(\vec{\boldsymbol{x}}^{(k)}\big)^T\vec{\boldsymbol{d}}^{(k)} < 0,$$

i.e., if $\vec{\boldsymbol{d}}^{(k)}$ is indeed a descent direction. Unfortunately, this minimal requirement is not enough to choose a good step size in practice.

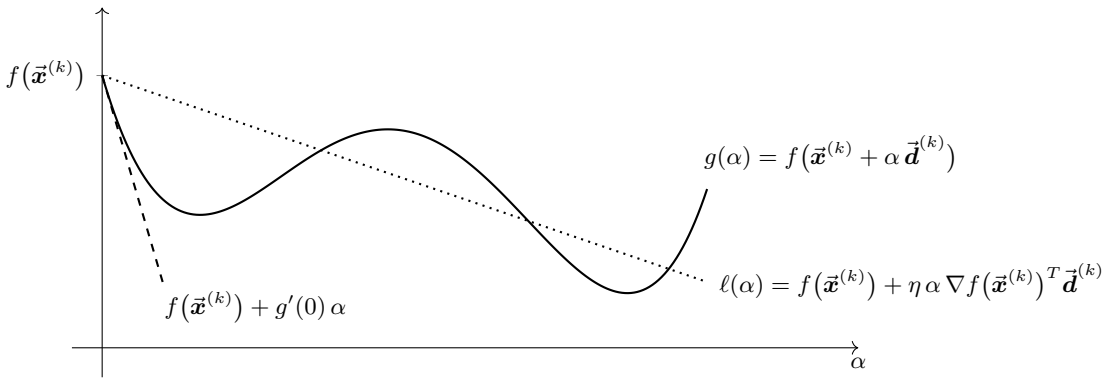
One possibility is to choose the step size that minimizes $g(\alpha)$, i.e.,

$$\min_{\alpha>0}\; g(\alpha) \quad \text{with} \quad g(\alpha) = f\big(\vec{\boldsymbol{x}}^{(k)} + \alpha\,\vec{\boldsymbol{d}}^{(k)}\big), \tag{19}$$

which is called *exact line search*; however, solving this subproblem can be expensive. Therefore, in practice, $\alpha^{(k)}$ is chosen to satisfy less strong requirements, for instance the Wolfe conditions.

Remark 14. Whenever $\alpha^{(k)}$ is *not* chosen by solving the minimization problem (19) exactly, we say it is an *inexact line search* method. ◀

4.5.1 Wolfe Conditions



The Wolfe conditions are two conditions that guide the choice of a "good enough" step size $\alpha^{(k)}$ at each iteration of a line search method to achieve global convergence. Consider the affine function

$$\ell(\alpha) = f\big(\vec{\boldsymbol{x}}^{(k)}\big) + \eta\,\alpha\,\nabla f\big(\vec{\boldsymbol{x}}^{(k)}\big)^T\vec{\boldsymbol{d}}^{(k)},$$

with $\eta \in (0,1)$ a relaxation parameter (sometimes called the *relaxed tangent*). The first Wolfe condition (also called the *Armijo* or *sufficient decrease condition*) stipulates that $g(\alpha^{(k)})$ should be no larger than $\ell(\alpha^{(k)})$. Hence the Armijo condition requires that the decrease in f is at least proportional to

both $\alpha^{(k)}$ and $\nabla f\big(\vec{\boldsymbol{x}}^{(k)}\big)^T \vec{\boldsymbol{d}}^{(k)}$. In practice one typically chooses η to be rather small (e.g., $\eta = 10^{-4}$ or 10^{-2}).

The second Wolfe condition (also called the *curvature condition*) discards step sizes that are too small, thereby avoiding very slow progress.

Definition 26 (First Wolfe Condition: Armijo or Sufficient Decrease Condition)**.** Given a point $\vec{\boldsymbol{x}}^{(k)}$, a direction $\vec{\boldsymbol{d}}^{(k)}$ and a parameter $\eta \in (0,1)$, the step size $\alpha^{(k)} > 0$ in a line search method should verify

$$\boxed{g\big(\alpha^{(k)}\big) = f\big(\vec{\boldsymbol{x}}^{(k)} + \alpha^{(k)}\,\vec{\boldsymbol{d}}^{(k)}\big) \le f\big(\vec{\boldsymbol{x}}^{(k)}\big) + \eta\,\alpha^{(k)}\,\nabla f\big(\vec{\boldsymbol{x}}^{(k)}\big)^T \vec{\boldsymbol{d}}^{(k)} = \ell\big(\alpha^{(k)}\big).} \tag{20}$$

✍

Definition 27 (Second Wolfe Condition: Curvature Condition)**.** Given a point $\vec{\boldsymbol{x}}^{(k)}$, a descent direction $\vec{\boldsymbol{d}}^{(k)}$, and a parameter $\bar{\eta} \in (\eta, 1)$, the step size $\alpha^{(k)}$ should verify

$$\boxed{g'\big(\alpha^{(k)}\big) = \nabla f\big(\vec{\boldsymbol{x}}^{(k)} + \alpha^{(k)}\,\vec{\boldsymbol{d}}^{(k)}\big)^T \vec{\boldsymbol{d}}^{(k)} \ge \bar{\eta}\,\nabla f\big(\vec{\boldsymbol{x}}^{(k)}\big)^T \vec{\boldsymbol{d}}^{(k)} = \bar{\eta}\,g'\big(0\big).} \tag{21}$$

✍

Note that the left-hand side of the above inequality is $g'\big(\alpha^{(k)}\big)$ while the right-hand side is $\bar{\eta}\,g'(0)$ (and $g'(0) < 0$ since $\vec{\boldsymbol{d}}^{(k)}$ is a descent direction). In other words, the curvature condition ensures that the derivative of g at $\alpha^{(k)}$ is not too small in magnitude; otherwise, one could (and should) take a longer step.

The Armijo and curvature conditions together are known as the *Wolfe conditions*. Although they give criteria for selecting a "good" $\alpha^{(k)}$, they do not, by themselves, indicate how to find such an $\alpha^{(k)}$. One popular approach is the backtracking line search algorithm.

4.5.2 Backtracking Line Search

The main idea behind backtracking line search (Algorithm 4) is to start from a "large" initial value $\bar{\alpha}$ (e.g. $\bar{\alpha} = 10$) and then decrease it until the Armijo condition is met.

Algorithm 4 Backtracking Line Search

1: **Given:** Current point $\vec{\boldsymbol{x}}^{(k)}$, descent direction $\vec{\boldsymbol{d}}^{(k)}$
2: **Parameters:** Initial trial step size $\bar{\alpha} > 0$, relaxation parameter $\eta \in (0,1)$
3: Initialize: Set $\alpha^{(k)} \leftarrow \bar{\alpha}$
4: **while** $g(\alpha^{(k)}) > \ell(\alpha^{(k)})$ **do**
5: $\quad \alpha^{(k)} \leftarrow \rho\,\alpha^{(k)}$ ▷ with $\rho \in (0,1)$; often $\rho = \frac{1}{2}$ is used
6: **end while**
7: **Output:** $\alpha^{(k)}$

The backtracking line search algorithm can be incorporated into both Newton's and Gradient Descent algorithms to choose the step size at each iteration. A general line search procedure is given in Algorithm 5. In this algorithm, $\vec{\boldsymbol{d}}^{(k)} = -\nabla f(\vec{\boldsymbol{x}}^{(k)})$ (Gradient Descent; Algorithm 1) or $\vec{\boldsymbol{d}}^{(k)} = -\big(\underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}}^{(k)})\big)^{-1}\nabla f(\vec{\boldsymbol{x}}^{(k)})$ (Newton's Method; Algorithm 3) or any other descent direction.

Algorithm 5 General Line Search Algorithm

1: **Given:** Starting point $\vec{\boldsymbol{x}}^{(0)}$
2: Initialize: Set $k \leftarrow 0$
3: **while** $\|\nabla f(\vec{\boldsymbol{x}}^{(k)})\| > \epsilon$ and $k \leq k_{\max}$ **do**
4: Compute search direction $\vec{\boldsymbol{d}}^{(k)}$ ▷ e.g. using $-\nabla f(\vec{\boldsymbol{x}}^{(k)})$ or $-\big(\underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}}^{(k)})\big)^{-1}\nabla f(\vec{\boldsymbol{x}}^{(k)})$
5: Find $\alpha^{(k)}$ using the backtracking line search (Algorithm 4)
6: Update: $\vec{\boldsymbol{x}}^{(k+1)} = \vec{\boldsymbol{x}}^{(k)} + \alpha^{(k)}\,\vec{\boldsymbol{d}}^{(k)}$,
7: Increment: $k \leftarrow k+1$
8: **end while**
9: **Output:** $\vec{\boldsymbol{x}}^{(k)}$

4.5.3 Global Convergence Theorem

With the addition of the line search sub-routine, it is possible to prove that Newton's algorithm is globally convergent. In fact, a more general result holds for any line search method.

Theorem 20 (Zoutendijk's Theorem)**.** Assume that:

- $f : \mathbb{R}^n \to \mathbb{R}$ is continuously differentiable, L-smooth with $L > 0$, and bounded from below;
- For each iteration k, the search direction $\vec{\boldsymbol{d}}^{(k)}$ is a descent direction and the step size $\alpha^{(k)}$ satisfies the Wolfe conditions (20) and (21).

Then

$$\sum_{k\geq 0} \cos^2\theta^{(k)}\,\|\nabla f(\vec{\boldsymbol{x}}^{(k)})\|^2 < \infty,$$

where $\theta^{(k)}$ is the angle between $\vec{\boldsymbol{d}}^{(k)}$ and $-\nabla f(\vec{\boldsymbol{x}}^{(k)})$, i.e.,

$$\cos\theta^{(k)} = \frac{\nabla f(\vec{\boldsymbol{x}}^{(k)})^T\vec{\boldsymbol{d}}^{(k)}}{\|\nabla f(\vec{\boldsymbol{x}}^{(k)})\|\,\|\vec{\boldsymbol{d}}^{(k)}\|}.$$

◁

Proof. Let $k \geq 0$. Since $\alpha^{(k)}$ satisfies the curvature condition for some $0 < \bar{\eta} < 1$, we have

$$\nabla f\big(\vec{\boldsymbol{x}}^{(k+1)}\big)^T\vec{\boldsymbol{d}}^{(k)} = \nabla f\big(\vec{\boldsymbol{x}}^{(k)} + \alpha^{(k)}\,\vec{\boldsymbol{d}}^{(k)}\big)^T\vec{\boldsymbol{d}}^{(k)} \geq \bar{\eta}\,\nabla f\big(\vec{\boldsymbol{x}}^{(k)}\big)^T\vec{\boldsymbol{d}}^{(k)},$$

which can be rewritten as

$$\Big(\nabla f\big(\vec{\boldsymbol{x}}^{(k+1)}\big) - \nabla f\big(\vec{\boldsymbol{x}}^{(k)}\big)\Big)^T\vec{\boldsymbol{d}}^{(k)} \geq (\bar{\eta}-1)\,\nabla f\big(\vec{\boldsymbol{x}}^{(k)}\big)^T\vec{\boldsymbol{d}}^{(k)},$$

and, since ∇f is L-Lipschitz and by the Cauchy–Schwarz inequality, we have

$$\Big(\nabla f\big(\vec{\boldsymbol{x}}^{(k+1)}\big) - \nabla f\big(\vec{\boldsymbol{x}}^{(k)}\big)\Big)^T\vec{\boldsymbol{d}}^{(k)} \leq L\,\alpha^{(k)}\,\|\vec{\boldsymbol{d}}^{(k)}\|^2.$$

Combining the two inequalities, we obtain

$$\alpha^{(k)} \geq \frac{(\bar{\eta}-1)\,\nabla f\big(\vec{\boldsymbol{x}}^{(k)}\big)^T\vec{\boldsymbol{d}}^{(k)}}{L\,\|\vec{\boldsymbol{d}}^{(k)}\|^2}.$$

Since $\vec{\boldsymbol{d}}^{(k)}$ is a descent direction and $\alpha^{(k)}$ satisfies the Armijo condition for some $\eta \in (0, \bar{\eta})$, it follows that

$$\begin{aligned} f\big(\vec{\boldsymbol{x}}^{(k+1)}\big) &= f\big(\vec{\boldsymbol{x}}^{(k)} + \alpha^{(k)}\, \vec{\boldsymbol{d}}^{(k)}\big) \\ &\leq f\big(\vec{\boldsymbol{x}}^{(k)}\big) + \eta\, \alpha^{(k)}\, \nabla f\big(\vec{\boldsymbol{x}}^{(k)}\big)^T \vec{\boldsymbol{d}}^{(k)} \\ &\leq f\big(\vec{\boldsymbol{x}}^{(k)}\big) - \frac{\eta\,(\bar{\eta}-1)}{L}\, \frac{\big|\nabla f\big(\vec{\boldsymbol{x}}^{(k)}\big)^T \vec{\boldsymbol{d}}^{(k)}\big|^2}{\|\vec{\boldsymbol{d}}^{(k)}\|^2} \\ &= f\big(\vec{\boldsymbol{x}}^{(k)}\big) - \frac{\eta\,(\bar{\eta}-1)}{L}\, \cos^2\theta^{(k)}\, \|\nabla f(\vec{\boldsymbol{x}}^{(k)})\|^2. \end{aligned}$$

If we denote $c = \frac{\eta\,(\bar{\eta}-1)}{L}$ and sum these inequalities over k, we obtain

$$f\big(\vec{\boldsymbol{x}}^{(K)}\big) \leq f\big(\vec{\boldsymbol{x}}^{(0)}\big) - c \sum_{k=0}^{K} \cos^2\theta^{(k)}\, \|\nabla f(\vec{\boldsymbol{x}}^{(k)})\|^2.$$

Since f is bounded from below and $c > 0$, it follows that

$$\sum_{k\geq 0} \cos^2\theta^{(k)}\, \|\nabla f(\vec{\boldsymbol{x}}^{(k)})\|^2 < \infty.$$

□

An immediate consequence of Theorem 20 is that

$$\cos^2\theta^{(k)}\, \|\nabla f(\vec{\boldsymbol{x}}^{(k)})\|^2 \to 0.$$

If the cosine terms are bounded away from zero (i.e., there exists $\delta > 0$ such that $\cos\theta^{(k)} \geq \delta$ for all k), then this implies $\|\nabla f(\vec{\boldsymbol{x}}^{(k)})\| \to 0$; in other words, the iterates are attracted to a stationary point. For example:

- In Gradient Descent, where $\vec{\boldsymbol{d}}^{(k)} = -\nabla f(\vec{\boldsymbol{x}}^{(k)})$, we have $\cos\theta^{(k)} = -1$.
- In Newton's method, when
$$\vec{\boldsymbol{d}}^{(k)} = -\big(\underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}}^{(k)})\big)^{-1} \nabla f(\vec{\boldsymbol{x}}^{(k)}),$$
one may show that
$$\cos\theta^{(k)} \geq \frac{\lambda_{\min}\big(\underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}}^{(k)})\big)}{\lambda_{\max}\big(\underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}}^{(k)})\big)},$$
where $\lambda_{\min}\big(\underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}}^{(k)})\big)$ and $\lambda_{\max}\big(\underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}}^{(k)})\big)$ denote the smallest and largest eigenvalues of $\underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}}^{(k)})$. Hence if the condition number
$$\kappa\big(\underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}}^{(k)})\big) = \frac{\lambda_{\max}\big(\underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}}^{(k)})\big)}{\lambda_{\min}\big(\underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}}^{(k)})\big)}$$
is uniformly bounded, then $\cos\theta^{(k)}$ is bounded away from zero.

Thus, under appropriate assumptions, the line search methods (whether steepest descent, Newton's method with line search, or quasi-Newton methods with line search) yield

$$\|\nabla f(\vec{\boldsymbol{x}}^{(k)})\| \to 0.$$

Note, however, that this global convergence result does not necessarily imply that the iterates converge to a local minimizer without additional assumptions on the Hessian of f.

4.6 Variations of Newton's Method

Even with the addition of a line search, there remain two main drawbacks of Newton's algorithm:

- The update is well defined only if the Hessian matrix $\underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}}^{(k)})$ is positive definite,
- It requires the computation of second-order derivatives, which can be computationally expensive.

We now discuss possible solutions.

4.6.1 Newton with Hessian Correction

Recall that if $\underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}}^{(k)})$ is not positive definite, then the Newton direction

$$\vec{\boldsymbol{d}}^{(k)} = -\big(\underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}}^{(k)})\big)^{-1}\nabla f(\vec{\boldsymbol{x}}^{(k)}),$$

may not be well defined or may fail to be a descent direction. A common remedy is to *correct* the Hessian by adding a multiple of the identity matrix. That is, one seeks $\lambda > 0$ such that

$$\underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}}^{(k)}) + \lambda \underline{\boldsymbol{I}},$$

is positive definite.

A practical method for checking whether a symmetric matrix is positive definite is to attempt its Cholesky decomposition. If the decomposition fails, then the matrix is not positive definite. Recall that the Cholesky decomposition of a positive definite matrix $\underline{\boldsymbol{H}}$ is a factorization of the form

$$\underline{\boldsymbol{H}} = \underline{\boldsymbol{L}}\,\underline{\boldsymbol{L}}^T,$$

where $\underline{\boldsymbol{L}}$ is a lower triangular matrix (the Cholesky factor). This decomposition is widely used to solve linear systems $\underline{\boldsymbol{H}}\,\vec{\boldsymbol{x}} = \vec{\boldsymbol{b}}$ (by first solving $\underline{\boldsymbol{L}}\,\vec{\boldsymbol{v}} = \vec{\boldsymbol{b}}$ via forward substitution and then $\underline{\boldsymbol{L}}^T\,\vec{\boldsymbol{x}} = \vec{\boldsymbol{v}}$ via backward substitution) and to compute the inverse by solving $\underline{\boldsymbol{A}}\,\underline{\boldsymbol{X}} = \underline{\boldsymbol{I}}$.

Algorithm 6 describes a procedure for finding λ (the Hessian correction parameter) using the Cholesky decomposition.

Algorithm 6 Hessian Correction

Given: Hessian matrix $\underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}})$
Parameters: Initial correction parameter $\bar{\lambda} > 0$, increase factor $c \in (0, 1)$
Try to compute the Cholesky factor $\underline{\boldsymbol{L}}$ of $\underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}})$
if successful **then**
    Return $\underline{\boldsymbol{L}}$
end if
Set $\lambda \leftarrow \bar{\lambda}$
Try to compute the Cholesky factor $\underline{\boldsymbol{L}}$ of $\underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}}) + \lambda \underline{\boldsymbol{I}}$
while not successful **do**
    Increase $\lambda \leftarrow c\,\lambda$
    Try to compute the Cholesky factor $\underline{\boldsymbol{L}}$ of $\underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}}) + \lambda \underline{\boldsymbol{I}}$
end while
Output: $\underline{\boldsymbol{L}}$, the Cholesky factor of the corrected Hessian

Using the Hessian correction subroutine above, one can incorporate it into Newton's method. The overall algorithm with line search and Hessian correction is given in Algorithm 7.

Algorithm 7 Newton's Algorithm with Line Search and Hessian Correction

Given: Starting point $\vec{\boldsymbol{x}}^{(0)}$
Initialize: $k \leftarrow 0$
while $\|\nabla f(\vec{\boldsymbol{x}}^{(k)})\| > \epsilon$ and $k \leq k_{\max}$ **do**
    Compute $\vec{\boldsymbol{g}}^{(k)} = \nabla f(\vec{\boldsymbol{x}}^{(k)})$ and $\underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}}^{(k)})$
    Compute the Cholesky factor $\underline{\boldsymbol{L}}$ of $\underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}}^{(k)})$ using the Hessian Correction subroutine (Algorithm 6)
    Solve $\underline{\boldsymbol{L}}\,\vec{\boldsymbol{v}} = -\vec{\boldsymbol{g}}^{(k)}$ (forward substitution)
    Solve $\underline{\boldsymbol{L}}^T\,\vec{\boldsymbol{d}}^{(k)} = \vec{\boldsymbol{v}}$ (backward substitution)
    Find $\alpha^{(k)}$ using a line search procedure (e.g., Algorithm 4)
    Update: $\vec{\boldsymbol{x}}^{(k+1)} = \vec{\boldsymbol{x}}^{(k)} + \alpha^{(k)}\,\vec{\boldsymbol{d}}^{(k)}$
    Increment: $k \leftarrow k + 1$
end while
Output: $\vec{\boldsymbol{x}}^{(k)}$

4.6.2 Quasi-Newton Methods

Quasi-Newton methods aim to capture some of the advantages of Newton's method while avoiding the expensive computation of second-order derivatives. The main idea is to replace the Hessian $\underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}}^{(k)})$ by an approximation that is updated using gradient information only.

A natural starting point is the first-order Taylor expansion of ∇f at $\vec{\boldsymbol{x}}^{(k)}$:

$$\nabla f\big(\vec{\boldsymbol{x}}^{(k+1)}\big) \approx \nabla f\big(\vec{\boldsymbol{x}}^{(k)}\big) + \underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}}^{(k)})\,\Big(\vec{\boldsymbol{x}}^{(k+1)} - \vec{\boldsymbol{x}}^{(k)}\Big),$$

which can be rearranged to yield the *secant equation*

$$\underline{\boldsymbol{H}}_f(\vec{\boldsymbol{x}}^{(k)})\,\underbrace{\Big(\vec{\boldsymbol{x}}^{(k+1)} - \vec{\boldsymbol{x}}^{(k)}\Big)}_{\vec{\boldsymbol{s}}^{(k)}} \approx \nabla f\big(\vec{\boldsymbol{x}}^{(k+1)}\big) - \nabla f\big(\vec{\boldsymbol{x}}^{(k)}\big) \triangleq \vec{\boldsymbol{y}}^{(k)}.$$

That is, we wish to have

$$\tilde{\underline{\boldsymbol{H}}}^{(k+1)}\,\vec{\boldsymbol{s}}^{(k)} = \vec{\boldsymbol{y}}^{(k)},$$

where $\tilde{\underline{\boldsymbol{H}}}^{(k+1)}$ is a symmetric approximation of the Hessian. Since a symmetric $n \times n$ matrix has $\frac{n(n+1)}{2}$ free parameters, the system is underdetermined and there exist infinitely many solutions. One usually chooses the update so that $\tilde{\underline{\boldsymbol{H}}}^{(k+1)}$ is positive definite (to guarantee that the resulting search direction is a descent direction).

One of the earliest quasi-Newton methods is the DFP (Davidon–Fletcher–Powell) method. Its update is given by

$$\tilde{\underline{\boldsymbol{H}}}^{(k+1)} = \left(\underline{\boldsymbol{I}} - \frac{\vec{\boldsymbol{y}}^{(k)}\,\vec{\boldsymbol{s}}^{(k)T}}{\rho^{(k)}}\right)^T \tilde{\underline{\boldsymbol{H}}}^{(k)} \left(\underline{\boldsymbol{I}} - \frac{\vec{\boldsymbol{y}}^{(k)}\,\vec{\boldsymbol{s}}^{(k)T}}{\rho^{(k)}}\right) + \frac{\vec{\boldsymbol{y}}^{(k)}\,\vec{\boldsymbol{y}}^{(k)T}}{\rho^{(k)}},$$

with $\rho^{(k)} = \vec{\boldsymbol{y}}^{(k)T}\vec{\boldsymbol{s}}^{(k)}$.

A more popular approach is the BFGS (Broyden–Fletcher–Goldfarb–Shanno) method, which instead updates an approximation $\underline{\boldsymbol{B}}^{(k)}$ of the inverse Hessian. By considering the first-order Taylor expansion of ∇f, one may wish to satisfy

$$\underline{\boldsymbol{B}}^{(k+1)}\,\vec{\boldsymbol{y}}^{(k)} = \vec{\boldsymbol{s}}^{(k)},$$

and the BFGS update is given by

$$\underline{\boldsymbol{B}}^{(k+1)} = \underline{\boldsymbol{B}}^{(k)} - \frac{\underline{\boldsymbol{B}}^{(k)}\,\vec{\boldsymbol{s}}^{(k)}\,\vec{\boldsymbol{s}}^{(k)T}\,\underline{\boldsymbol{B}}^{(k)}}{\vec{\boldsymbol{s}}^{(k)T}\underline{\boldsymbol{B}}^{(k)}\vec{\boldsymbol{s}}^{(k)}} + \frac{\vec{\boldsymbol{y}}^{(k)}\,\vec{\boldsymbol{y}}^{(k)T}}{\rho^{(k)}},$$

with $\rho^{(k)} = \vec{\boldsymbol{y}}^{(k)T}\vec{\boldsymbol{s}}^{(k)}$.

5 Constrained Optimization

Recall Equation (1), the general formulation of a constrained optimization problem, as well as Equation (2), the definition of the feasible set.

Definition 28 (Active Set)**.** The active set $\mathcal{A}$ at any feasible $\vec{\boldsymbol{x}}$ consists of the equality constraint indices from $\mathcal{E}$ together wiwth the indices of the inequality constraints i for which $c_i(\vec{\boldsymbol{x}}) = 0$; that is,

$$\mathcal{A} = \mathcal{E} \cup \{i \in \mathcal{I} \mid c_i(\vec{\boldsymbol{x}}) = 0\}.$$

At a feasible point $\vec{\boldsymbol{x}}$, the inequality constraint i is said to be *active* if $c_i(\vec{\boldsymbol{x}}) = 0$ and *inactive* if the strict inequality $c_i(\vec{\boldsymbol{x}}) > 0$ is satisfied. ✍

6 Exam Topics

- In particular, revise:
  - Optimality conditions
  - Definitions of directional derivative and descent directions
  - Definitions and properties of convex, strongly convex, Lipschitz and smooth functions
  - Properties of multivariate quadratic functions
  - Wolfe conditions
- Do you know how to recognise a convex function? A smooth function?
- Do you know how to recognise that a point is a local minimiser?
- Do you know what is a descent direction?
- Do you know what is a quadratic function and in which cases it has a global minimiser?
- Do you know how to apply Gradient Descent and Newton's algorithms?
- Do you know how to apply the Wolfe conditions?
- Is a square matrix with all entries equal to 0 positive-semidefinite? Positive-definite?
- What are the eigenvalues of the following matrix? Its condition number? Its spectral norm? Its inverse? The eigenvalues of its inverse?

$$\underline{\boldsymbol{A}} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 2.5 & 0 \\ 0 & 0 & 0.5 \end{pmatrix}$$

Answers

Q1: What are the optimality conditions?

A:

- **First-Order Necessary Condition:** For an unconstrained problem

$$\min_{\vec{\boldsymbol{x}}} f(\vec{\boldsymbol{x}}),$$

  a necessary condition for $\vec{\boldsymbol{x}}^*$ to be a local minimiser is

$$\nabla f(\vec{\boldsymbol{x}}^*) = 0.$$

- **Second-Order Necessary Condition:** If $\nabla f(\vec{\boldsymbol{x}}^*) = 0$, a necessary condition is that the Hessian is positive semidefinite:

$$\nabla^2 f(\vec{\boldsymbol{x}}^*) \succeq 0.$$

- **Second-Order Sufficient Condition:** If $\nabla f(\vec{\boldsymbol{x}}^*) = 0$ and

$$\nabla^2 f(\vec{\boldsymbol{x}}^*) \succ 0 \quad \text{(positive definite)},$$

  then $\vec{\boldsymbol{x}}^*$ is a strict local minimiser.

Q2: What are the definitions of the directional derivative and descent directions?

A:

- **Directional Derivative:** For a function $f : \mathbb{R}^n \to \mathbb{R}$, the directional derivative at $\vec{\boldsymbol{x}}$ in the direction $\vec{\boldsymbol{d}}$ is defined as

$$f'(\vec{\boldsymbol{x}}; \vec{\boldsymbol{d}}) = \lim_{t \to 0} \frac{f(\vec{\boldsymbol{x}} + t\,\vec{\boldsymbol{d}}) - f(\vec{\boldsymbol{x}})}{t},$$

  provided the limit exists.

- **Descent Direction:** A vector $\vec{\boldsymbol{d}}$ is a descent direction at $\vec{\boldsymbol{x}}$ if

$$\nabla f(\vec{\boldsymbol{x}})^T \vec{\boldsymbol{d}} < 0.$$

  This implies that for sufficiently small $t > 0$, $f(\vec{\boldsymbol{x}} + t\,\vec{\boldsymbol{d}}) < f(\vec{\boldsymbol{x}})$.

Q3: What are the definitions and properties of convex, strongly convex, Lipschitz, and smooth functions?

A:

- **Convex Function:** f is convex if for all $\vec{\boldsymbol{x}}, \vec{\boldsymbol{y}}$ and $\lambda \in [0, 1]$,

$$f(\lambda\,\vec{\boldsymbol{x}} + (1-\lambda)\,\vec{\boldsymbol{y}}) \leq \lambda\, f(\vec{\boldsymbol{x}}) + (1-\lambda)\, f(\vec{\boldsymbol{y}}).$$

  For differentiable functions, this is equivalent to:

$$f(\vec{\boldsymbol{y}}) \geq f(\vec{\boldsymbol{x}}) + \nabla f(\vec{\boldsymbol{x}})^T(\vec{\boldsymbol{y}} - \vec{\boldsymbol{x}}) \quad \forall\, \vec{\boldsymbol{x}}, \vec{\boldsymbol{y}}.$$

- **Strongly Convex Function:** f is strongly convex with parameter $\mu > 0$ if

$$f(\vec{\boldsymbol{y}}) \geq f(\vec{\boldsymbol{x}}) + \nabla f(\vec{\boldsymbol{x}})^T(\vec{\boldsymbol{y}} - \vec{\boldsymbol{x}}) + \frac{\mu}{2}\|\vec{\boldsymbol{y}} - \vec{\boldsymbol{x}}\|^2.$$

  Equivalently, its Hessian satisfies

$$\nabla^2 f(\vec{\boldsymbol{x}}) \succeq \mu \underline{\boldsymbol{I}} \quad \text{for all } \vec{\boldsymbol{x}}.$$

- **Lipschitz Continuity:** A function f (or its gradient) is Lipschitz continuous if there exists a constant $L \geq 0$ such that

$$\|f(\vec{\boldsymbol{x}}) - f(\vec{\boldsymbol{y}})\| \leq L\,\|\vec{\boldsymbol{x}} - \vec{\boldsymbol{y}}\|,$$

  or for gradients,

$$\|\nabla f(\vec{\boldsymbol{x}}) - \nabla f(\vec{\boldsymbol{y}})\| \leq L\,\|\vec{\boldsymbol{x}} - \vec{\boldsymbol{y}}\|.$$

- **Smooth Function:** A function is L-smooth if it is differentiable and its gradient is Lipschitz continuous with constant L; that is,

$$f(\vec{\boldsymbol{y}}) \leq f(\vec{\boldsymbol{x}}) + \nabla f(\vec{\boldsymbol{x}})^T(\vec{\boldsymbol{y}} - \vec{\boldsymbol{x}}) + \frac{L}{2}\|\vec{\boldsymbol{y}} - \vec{\boldsymbol{x}}\|^2.$$

Q4: What are the properties of multivariate quadratic functions?

A:

- **General Form:** A quadratic function in $\mathbb{R}^n$ can be expressed as

$$f(\vec{\boldsymbol{x}}) = \frac{1}{2}\,\vec{\boldsymbol{x}}^T \underline{\boldsymbol{A}}\,\vec{\boldsymbol{x}} + \vec{\boldsymbol{b}}^T\,\vec{\boldsymbol{x}} + c,$$

  where $\underline{\boldsymbol{A}}$ is a symmetric matrix, $\vec{\boldsymbol{b}}$ is a vector, and c is a scalar.

- **Convexity:** The function is convex if $\underline{\boldsymbol{A}} \succeq 0$ (i.e., $\underline{\boldsymbol{A}}$ is positive semidefinite) and strictly convex if $\underline{\boldsymbol{A}} \succ 0$ (i.e., $\underline{\boldsymbol{A}}$ is positive definite).

- **Global Minimiser:**

  - If $\underline{\boldsymbol{A}} \succ 0$, then the unique global minimiser is

$$\vec{\boldsymbol{x}}^* = -\,\underline{\boldsymbol{A}}^{-1}\,\vec{\boldsymbol{b}}.$$

- If $\underline{\boldsymbol{A}} \succeq 0$ but is singular, a global minimiser exists if and only if $\vec{\boldsymbol{b}}$ lies in the range (column space) of $\underline{\boldsymbol{A}}$; however, the minimiser may not be unique.

Q5: What are the Wolfe conditions?

A: The Wolfe conditions are used in line search methods to choose a step size α and consist of:

- **Armijo (Sufficient Decrease) Condition:**

$$f(\vec{\boldsymbol{x}}_k + \alpha\, \vec{\boldsymbol{d}}_k) \leq f(\vec{\boldsymbol{x}}_k) + c_1\, \alpha\, \nabla f(\vec{\boldsymbol{x}}_k)^T \vec{\boldsymbol{d}}_k,$$

  where $0 < c_1 < 1$.

- **Curvature Condition:**

$$\nabla f(\vec{\boldsymbol{x}}_k + \alpha\, \vec{\boldsymbol{d}}_k)^T \vec{\boldsymbol{d}}_k \;\geq\; c_2\, \nabla f(\vec{\boldsymbol{x}}_k)^T \vec{\boldsymbol{d}}_k,$$

  with $c_1 < c_2 < 1$.

Q6: How do you recognise a convex function? A smooth function?

A:

- **Convex Functions:** Verify that the line segment between any two points on the graph lies above the graph; for differentiable functions, a nonnegative second derivative (or positive semidefinite Hessian) confirms convexity.
- **Smooth Functions:** These are functions that are continuously differentiable, often with a Lipschitz continuous gradient.

Q7: How do you recognise that a point is a local minimiser?

A: A point $\vec{\boldsymbol{x}}^*$ is a local minimiser if:

- **First-Order Test:** $\nabla f(\vec{\boldsymbol{x}}^*) = 0$.
- **Second-Order Test:** The Hessian $\nabla^2 f(\vec{\boldsymbol{x}}^*)$ is positive definite (or at least positive semidefinite with additional conditions to rule out saddle points).
- There exists a neighborhood U of $\vec{\boldsymbol{x}}^*$ such that $f(\vec{\boldsymbol{x}}^*) \leq f(\vec{\boldsymbol{x}})$ for all $\vec{\boldsymbol{x}} \in U$.

Q8: What is a descent direction?

A: A descent direction $\vec{\boldsymbol{d}}$ at a point $\vec{\boldsymbol{x}}$ satisfies:

$$\nabla f(\vec{\boldsymbol{x}})^T \vec{\boldsymbol{d}} < 0,$$

ensuring that a sufficiently small move in the direction $\vec{\boldsymbol{d}}$ decreases the function value.

Q9: What is a quadratic function and in which cases does it have a global minimiser?

A:

- **Quadratic Function Form:**

$$f(\vec{\boldsymbol{x}}) = \frac{1}{2}\, \vec{\boldsymbol{x}}^T \underline{\boldsymbol{A}}\, \vec{\boldsymbol{x}} + \vec{\boldsymbol{b}}^T\, \vec{\boldsymbol{x}} + c.$$

- **Global Minimiser Existence:**
  - If $\underline{\boldsymbol{A}} \succ 0$, the function is strictly convex and the unique global minimiser is

$$\vec{\boldsymbol{x}}^* = -\, \underline{\boldsymbol{A}}^{-1}\, \vec{\boldsymbol{b}}.$$

- If $\underline{\boldsymbol{A}} \succeq 0$ (but not positive definite), a global minimiser exists if $\vec{\boldsymbol{b}}$ lies in the range of $\underline{\boldsymbol{A}}$; otherwise, the function is unbounded below. The minimiser may not be unique in that case.

Q10: How do you apply Gradient Descent and Newton's algorithms?

A:

- **Gradient Descent:** Update iteratively via

$$\vec{\boldsymbol{x}}_{k+1} = \vec{\boldsymbol{x}}_k - \alpha_k \nabla f(\vec{\boldsymbol{x}}_k),$$

  where the step size α_k can be determined by a line search (often using Wolfe or Armijo conditions).

- **Newton's Method:** Update using second-order information:

$$\vec{\boldsymbol{x}}_{k+1} = \vec{\boldsymbol{x}}_k - \left[\nabla^2 f(\vec{\boldsymbol{x}}_k)\right]^{-1} \nabla f(\vec{\boldsymbol{x}}_k).$$

  This method exhibits quadratic convergence near the minimiser provided the Hessian is positive definite and the initial guess is close enough.

Q11: How do you apply the Wolfe conditions during a line search?

A:

1. Choose an initial step size α and compute the trial point $\vec{\boldsymbol{x}}_k + \alpha\, \vec{\boldsymbol{d}}_k$.
2. Verify the **Armijo condition**:

$$f(\vec{\boldsymbol{x}}_k + \alpha\, \vec{\boldsymbol{d}}_k) \leq f(\vec{\boldsymbol{x}}_k) + c_1\, \alpha\, \nabla f(\vec{\boldsymbol{x}}_k)^T \vec{\boldsymbol{d}}_k.$$

3. Check the **Curvature condition**:

$$\nabla f(\vec{\boldsymbol{x}}_k + \alpha\, \vec{\boldsymbol{d}}_k)^T \vec{\boldsymbol{d}}_k \;\geq\; c_2\, \nabla f(\vec{\boldsymbol{x}}_k)^T \vec{\boldsymbol{d}}_k.$$

4. If both conditions are met, accept α; otherwise, adjust (typically reduce) α and repeat.

Q12: Is a square matrix with all entries equal to 0 positive-semidefinite? Positive-definite?

A:

- **Positive-Semidefinite:** Yes, because for any vector $\vec{\boldsymbol{x}}$,

$$\vec{\boldsymbol{x}}^T \underline{\mathbf{0}}\, \vec{\boldsymbol{x}} = 0 \geq 0.$$

- **Positive-Definite:** No, since for any nonzero $\vec{\boldsymbol{x}}$,

$$\vec{\boldsymbol{x}}^T \underline{\mathbf{0}}\, \vec{\boldsymbol{x}} = 0 \quad \text{(not strictly greater than 0)}.$$

Q13: Given the matrix $\underline{\boldsymbol{A}} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 2.5 & 0 \\ 0 & 0 & 0.5 \end{pmatrix}$, what are its eigenvalues, condition number, spectral norm, inverse, and the eigenvalues of its inverse?

A:

- **Eigenvalues:** They are the diagonal entries:

$$\lambda_1 = 1, \quad \lambda_2 = 2.5, \quad \lambda_3 = 0.5.$$

- **Condition Number:** The 2-norm condition number is given by

$$\kappa(\underline{\boldsymbol{A}}) = \frac{2.5}{0.5} = 5.$$

- **Spectral Norm:** This is the largest absolute eigenvalue:

$$\|\underline{\boldsymbol{A}}\|_2 = 2.5.$$

- **Inverse of $\underline{\boldsymbol{A}}$:** Since $\underline{\boldsymbol{A}}$ is diagonal,

$$\underline{\boldsymbol{A}}^{-1} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \frac{1}{2.5} & 0 \\ 0 & 0 & \frac{1}{0.5} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0.4 & 0 \\ 0 & 0 & 2 \end{pmatrix}.$$

- **Eigenvalues of $\underline{\boldsymbol{A}}^{-1}$:** These are the reciprocals of the eigenvalues of $\underline{\boldsymbol{A}}$:

$$1, \quad 0.4, \quad 2.$$