MATH 164: Optimization

Lecture Notes, Winter 2026

Yifan Chen
UCLA, Department of Mathematics

March 2026

Abstract

These notes consolidate the lectures of MATH 164 (Optimization) into a single self–contained document (any typos mine; notes are generated based on my hand-written notes, with the help of Claude Code). The course covers four blocks: (i) generic unconstrained optimization (line search, gradient methods, Newton, quasi–Newton, conjugate gradient); (ii) unconstrained quadratic / least–squares problems together with linear systems and the pseudoinverse; (iii) linear programming and the simplex method; (iv) a brief treatment of duality and KKT conditions for constrained problems. The primary reference is E. K. P. Chong and S. Żak, *An Introduction to Optimization* (2nd ed.). A useful secondary reference is J. Nocedal and S. J. Wright, *Numerical Optimization*.

Contents

Part I
Foundations and Unconstrained Optimization

1 Introduction

What is optimization? Quoting our textbook: *"Optimization is central to any problem involving decision making, whether in engineering or economics. The task of decision making entails choosing between various alternatives. This choice is governed by our desire to make the "best" decision. The measure of goodness of the alternatives is described by an* objective function *or* performance index*. Optimization theory and methods deal with selecting the best alternative in the sense of the given objective function."*

In practice, the starting point of an optimization problem is an objective function and a set of constraints, both arising from some application (engineering design, economics, statistics, machine learning, etc.). This course provides the basic theory and the numerical algorithms used to compute (approximate) optimal solutions.

Generic problem. Throughout these notes the generic optimization problem is

$$\min_{x \in \Omega} f(x), \qquad f : \mathbb{R}^n \to \mathbb{R}, \qquad \Omega \subseteq \mathbb{R}^n. \tag{1}$$

The vector $x \in \mathbb{R}^n$ is the *decision variable*, f is the *objective function*, and Ω is the *feasible set* (or *constraint set*). When $\Omega = \mathbb{R}^n$ the problem is called *unconstrained*; otherwise it is *constrained*. In this course constraints typically take the form

$$\Omega \;=\; \big\{x \in \mathbb{R}^n \,:\, g(x) \le 0,\; h(x) = 0\big\},$$

for some functions $g : \mathbb{R}^n \to \mathbb{R}^p$ and $h : \mathbb{R}^n \to \mathbb{R}^q$. A simple constraint that recurs throughout the course is the non–negativity constraint $\Omega = \{x : x_i \ge 0 \text{ for all } i\}$.

1.1 Global and local minimizers

Definition 1.1 (Global minimizer)**.** A point $x^* \in \Omega$ is a *global minimizer* of (1) if $f(x^*) \le f(x)$ for every $x \in \Omega$.

Global minimizers may fail to exist, and even when they exist they need not be unique.

Example 1.2 (Existence and uniqueness pathologies)**.** Three examples that illustrate the issues.

1. $\min (x-1)^2$ over $\mathbb{R}$: unique global minimizer at $x^* = 1$.
2. $\min\big((x-1)(x+1)\big)^2$ over $\mathbb{R}$: two global minimizers, $x^* = \pm 1$.
3. $\min (x-1)^2$ subject to $x > 2$: *no* global minimizer. The candidate value $x = 2$ is plausibly a minimizer but is *not feasible*, because the feasible set $(2, \infty)$ is not closed.

The pathology of (3) is essentially the only pathology one encounters: it is caused by the constraint set failing to be *closed*, i.e. by missing a limit point. The closed interval $[a, b]$ contains its limit points, while (a, b) does not, and one should think of closedness as the rule needed to make "minimum" attainable. This will recur throughout the course.

Remark 1.3 (Global optimization is hard)**.** Finding a global minimizer in full generality is intractable ("not impossible, but only possible for certain structures"). Most algorithms in this course are content with finding a *local* minimizer, defined next.

Definition 1.4 (Local minimizer)**.** A point $x^* \in \Omega$ is a *local minimizer* of (1) if there exists $\delta > 0$ such that $f(x^*) \le f(x)$ for every $x \in \Omega$ with $\|x - x^*\| \le \delta$. In words, x^* is at least as good as any feasible point in some neighborhood.

Plan of the course. Following the syllabus, we will work through, in order:

- generic unconstrained optimization: 1D search, gradient methods, Newton's method, quasi–Newton methods;
- unconstrained quadratic and least–squares problems: conjugate direction methods, linear systems, pseudoinverse;

- linearly constrained problems: linear programming and the simplex method;
- nonlinearly constrained problems: a brief treatment of duality and the KKT conditions.

2 One–dimensional search

The story begins in dimension one. The setting is

$$\min_{x\in[a,b]} f(x), \qquad f:[a,b]\to\mathbb{R} \text{ continuous.}$$

Existence of a minimizer is guaranteed by the extreme value theorem (continuous function on a closed bounded interval), so the question is purely *algorithmic*: by querying f (and possibly its derivatives) at finitely many points, can we approximately locate a minimizer, and how quickly? We measure "how quickly" in terms of the number of function (or derivative) evaluations.

2.1 Derivative–free search: golden ratio

We first consider *derivative–free* algorithms that only query the values of f.

Why one query is not enough. A single value $f(c)$ tells us nothing about where the minimizer lies in $[a,b]$, because the minimum could be at either endpoint or anywhere in between. We need at least two queries.

The two–query bracketing principle. Pick $a<c<d<b$, query $f(c)$ and $f(d)$, and consider the larger of the two values. Intuitively the minimizer "lies on the side closer to the smaller value." The lemma below makes this precise, under the assumption that f is *unimodal*.

Definition 2.1 (Unimodal)**.** A continuous function $f:[a,b]\to\mathbb{R}$ is *unimodal* on $[a,b]$ if it has a unique local minimum on (a,b). Unimodality says that f "goes down then up."

Lemma 2.2 (Bracketing)**.** *Pick $a<c<d<b$ and query $f(c)$, $f(d)$.*

- *If $f(c)\le f(d)$, then $[a,d]$ contains a local minimizer of f on $[a,b]$.*
- *If $f(c)>f(d)$, then $[c,b]$ contains a local minimizer.*

Proof. Without loss of generality assume $f(c)\le f(d)$. Since $[a,d]$ is closed and bounded and f is continuous, f attains its minimum on $[a,d]$ at some $x^*\in[a,d]$. By the assumption $f(c)\le f(d)$, we may take $x^*\neq d$ (we can always pick c in case of a tie), so $x^*\in[a,d)$, which makes x^* a local minimizer of f on $[a,b]$ as required. □

Iterating naively. Setting $a_0 = a, b_0 = b$, applying Lemma 2.2 produces $[a_1, b_1]$ of half the length. Repeating uses two new function evaluations per step. After k steps the interval has shrunk by a factor that depends on *how* we pick c, d at each iteration; with the symmetric choice $c - a = b - d = \rho(b - a)$ for $\rho \in (0, \frac{1}{2})$, the new sub–interval has length $(1-\rho)(b-a)$. To reach an ε–accurate minimizer we need $(1-\rho)^k|b-a| \le \varepsilon$, i.e. $k \approx \log\big(\varepsilon/|b-a|\big)/\log(1-\rho)$ iterations, each costing two new function evaluations — $2k$ evaluations in total.

Reusing information. We can do better by choosing ρ so that one of the new query points coincides with a previously computed query point. Then each iteration costs *one* new function evaluation, halving the total cost. The geometric relation that enforces this reuse is

$$1 \;=\; \rho \;+\; (1-\rho)^2,$$

which is solved by $\rho = (3-\sqrt{5})/2 \approx 0.38197$ and $1-\rho = (\sqrt{5}-1)/2 \approx 0.61803$. The constant $1-\rho$ is the celebrated *golden ratio*.

Theorem 2.3 (Golden Ratio Search)**.** *With the golden–ratio choice $\rho = (3-\sqrt{5})/2$, the iterated bracketing rule produces nested intervals $[a_k, b_k] \subseteq [a, b]$ with*

$$|b_k - a_k| \;=\; (1-\rho)^k\,|b-a|,$$

costing only one *new function evaluation per step after the first. To reach an ε–accurate minimizer it suffices to take $k \;\approx\; \log\big(\varepsilon/|b-a|\big)/\log(1-\rho)$ steps.*

Remark 2.4. The textbook discusses an improvement of golden–ratio search that chooses different ratios at different iterations, but the asymptotic rate is essentially the same.

Two metrics for an algorithm. We always care about two things:

- the *accuracy / convergence rate*: how fast does the error shrink as we iterate?
- the *cost per iteration*: how many function (or derivative) evaluations does each iteration consume?

Golden–ratio search is cheap per iteration (one function value) but slow (linear rate ≈ 0.618). We will see that derivative–based methods can converge much faster, at the cost of extra information per step.

2.2 Derivative–based search: bisection and 1D Newton

Suppose now $f \in C^1([a, b])$. The first–order tools below convert minimization into root finding, since at an interior local minimizer the derivative vanishes.

Theorem 2.5 (FONC, 1D)**.** *If $f \in C^1$ and $x^* \in (a, b)$ is a local minimizer of f, then $f'(x^*) = 0$.*

Proof. Pick $h > 0$ small enough that $x^* \pm h \in (a, b)$. Local minimality gives $f(x^* \pm h) \geq f(x^*)$, hence

$$\lim_{h \to 0^+} \frac{f(x^* + h) - f(x^*)}{h} \;\geq\; 0, \qquad \lim_{h \to 0^+} \frac{f(x^* - h) - f(x^*)}{-h} \;\leq\; 0.$$

Both limits equal $f'(x^*)$, so $f'(x^*) \geq 0$ and $f'(x^*) \leq 0$, i.e. $f'(x^*) = 0$. □

Remark 2.6 (FONC is necessary, not sufficient)**.** Theorem 2.5 only goes one way. For example, take $f(x) = x^3$. Then $f'(0) = 0$ but 0 is neither a local minimum nor a local maximum. For specific classes of functions (unimodal functions, convex functions, ...) the FONC *is* sufficient, but in general it merely identifies candidate points, which we call *first–order stationary points*.

Idea: minimizing ⇔ root finding. By Theorem 2.5, solving $\min f(x)$ amounts to solving the equation $f'(x) = 0$. Two classical algorithms for root finding in 1D give us two algorithms for minimization.

Bisection

Suppose f' is continuous and $f'(a)\, f'(b) < 0$, so by the intermediate value theorem there is a root of f' in (a, b).

Algorithm. Set $a_0 = a$, $b_0 = b$. At iteration k:

1. Set $m = (a_k + b_k)/2$ and query $f'(m)$.
2. If $f'(m) = 0$, declare m a root and stop.
3. Otherwise, if $f'(a_k)f'(m) < 0$ set $a_{k+1} = a_k$, $b_{k+1} = m$; else set $a_{k+1} = m$, $b_{k+1} = b_k$.

Properties. By induction $f'(a_k)f'(b_k) < 0$, so some $x^* \in [a_k, b_k]$ has $f'(x^*) = 0$. Each iteration halves the interval: $|b_k - a_k| = 2^{-k}|b - a|$. After k iterations the algorithm has produced an approximate root within distance at most $2^{-k}|b - a|$ of a true root, at a total cost of $k + 2$ derivative evaluations.

Newton's method (1D)

Idea. In general, $\min f(x)$ is hard, but for *specific* f it can be very easy. The simplest case is when f is quadratic: $\min(\frac{1}{2}ax^2 + bx + c)$ has the closed form $x^* = -b/a$ (assuming $a > 0$). Newton's method exploits this by approximating f near the current iterate x_k by its second order Taylor polynomial,

$$f(x) \;\approx\; p_k(x) \;:=\; f(x_k) + f'(x_k)(x - x_k) + \tfrac{1}{2}f''(x_k)(x - x_k)^2, \tag{2}$$

and *exactly* minimizing p_k to obtain the next iterate: $p_k'(x) = 0$ gives

$$x_{k+1} \;=\; x_k \;-\; \frac{f'(x_k)}{f''(x_k)}. \tag{3}$$

Two facts.

- If f is itself quadratic, the method is *exact in one step* from any starting point (the Taylor model is exact).
- The method requires $f''(x_k) \neq 0$. In particular, for the algorithm to converge to a minimizer x^*, we will need $f''(x^*) \neq 0$.

Theorem 2.7 (Local quadratic convergence of 1D Newton)**.** *Let $f \in C^3$ in a neighborhood of a local minimizer x^*, with $f''(x^*) \neq 0$. Then for x_k sufficiently close to x^*,*

$$|x_{k+1} - x^*| \;=\; O\big(|x_k - x^*|^2\big)\,.$$

Proof. Subtract x^* from (3) and use $f'(x^*) = 0$:

$$\begin{aligned} x_{k+1} - x^* &= x_k - x^* - \frac{f'(x_k)}{f''(x_k)} \\ &= \frac{f''(x_k)(x_k - x^*) - \big(f'(x_k) - f'(x^*)\big)}{f''(x_k)}. \end{aligned}$$

By Taylor's theorem with remainder, the numerator equals $\frac{1}{2}f'''(\xi)(x_k - x^*)^2$ for some ξ between x_k and x^*. Hence $|x_{k+1} - x^*| \leq C|x_k - x^*|^2$ with $C = \sup|f'''|/(2\inf|f''|)$ on a small neighborhood of x^*, which is finite. □

A sequence with $|x_{k+1} - x^*| = O(|x_k - x^*|^2)$ is said to have *convergence order* 2, or to converge *quadratically*. Quadratic convergence is dramatically faster than the linear convergence of bisection or golden ratio search: the number of correct digits roughly *doubles* per iteration.

The price of fast convergence is two–fold: first, we must be able to compute (or at least estimate) f''; second, x_0 must be sufficiently close to x^* for the local convergence to kick in. This local-vs-global tension will reappear throughout the course.

3 Optimality conditions in $\mathbb{R}^n$

We now move from one dimension to n dimensions. Throughout this section $f : \mathbb{R}^n \to \mathbb{R}$ is at least C^1, and $\Omega \subseteq \mathbb{R}^n$ is the feasible set. We recall that *optimization in high dimensions is hard*: derivative–free search is essentially hopeless, because there are simply too many directions in which f might decrease. Some kind of derivative information is essential.

3.1 The general iterative framework

A generic iterative algorithm for $\min_{x\in\Omega} f(x)$ takes the form

$$x^{(k+1)} = x^{(k)} + \alpha_k\, d^{(k)}, \tag{4}$$

where $d^{(k)} \in \mathbb{R}^n$ is the *search direction* and $\alpha_k > 0$ is the *step size* at iteration k. Designing an algorithm amounts to specifying:

- how to choose the search direction $d^{(k)}$;
- how to choose the step size α_k.

Both questions are guided by *optimality conditions*, which tell us what a candidate minimizer must look like.

3.2 Feasible directions and the constrained FONC

Definition 3.1 (Feasible direction)**.** A vector $d \in \mathbb{R}^n$ is a *feasible direction* at $x^* \in \Omega$ if there exists $\alpha_0 > 0$ such that $x^* + \alpha d \in \Omega$ for every $\alpha \in [0, \alpha_0]$.

Geometric picture. Think of Ω as a region in $\mathbb{R}^2$ and pick two points: an interior point x_1 and a boundary point x_2. At x_1 *every* direction $d \in \mathbb{R}^n$ is feasible (we can move in any direction by a sufficiently small step without leaving Ω). At x_2, only directions "pointing inwards" are feasible.

Theorem 3.2 (FONC, constrained)**.** *Let $f \in C^1$ and let $x^* \in \Omega$ be a local minimizer of (1). Then for every feasible direction d at x^*,*

$$\nabla f(x^*)^\top d \;\geq\; 0. \tag{5}$$

Proof. Let d be feasible at x^*, so $x^* + \alpha d \in \Omega$ for $\alpha \in [0, \alpha_0]$. By local minimality, for α small enough we may also assume $f(x^* + \alpha d) \geq f(x^*)$. Hence

$$0 \;\leq\; \lim_{\alpha \to 0^+} \frac{f(x^* + \alpha d) - f(x^*)}{\alpha} \;=\; \nabla f(x^*)^\top d$$

by the chain rule applied to $\alpha \mapsto f(x^* + \alpha d)$. □

Corollary 3.3 (FONC, unconstrained)**.** *If x^* is an interior point of Ω (in particular, if $\Omega = \mathbb{R}^n$), then $\nabla f(x^*) = 0$.*

Proof. At an interior point every direction $d \in \mathbb{R}^n$ is feasible, so (5) applied to d and $-d$ gives $\nabla f(x^*)^\top d = 0$ for every $d \in \mathbb{R}^n$. Hence $\nabla f(x^*) = 0$. □

3.3 Second order conditions

The FONC merely identifies candidate points (compare Remark 2.6). Second–order information sharpens the picture.

Theorem 3.4 (Second–order conditions)**.** *Let $f \in C^2$ and let $x^* \in \mathbb{R}^n$ be an interior point of Ω.*

- SONC. *If x^* is a local minimizer, then $\nabla f(x^*) = 0$ and $\nabla^2 f(x^*) \succeq 0$ (positive semi–definite, i.e. all eigenvalues non–negative).*
- SOSC. *If $\nabla f(x^*) = 0$ and $\nabla^2 f(x^*) \succ 0$ (positive definite, all eigenvalues strictly positive), then x^* is a strict local minimizer.*

Proof of SONC. The FONC half is Corollary 3.3. For the Hessian condition, fix $d \neq 0$ and Taylor expand at the stationary point:

$$f(x^* + \alpha d) \;=\; f(x^*) + \tfrac{1}{2}\alpha^2\, d^\top \nabla^2 f(x^*)\, d + o(\alpha^2).$$

Local minimality requires $f(x^* + \alpha d) - f(x^*) \geq 0$ for all small α, which forces the coefficient of α^2 to be non–negative: $d^\top \nabla^2 f(x^*) d \geq 0$ for every d, i.e. $\nabla^2 f(x^*) \succeq 0$. For SOSC, the same Taylor expansion gives $f(x^*+\alpha d) - f(x^*) \geq \tfrac{1}{2}\alpha^2 \lambda_{\min}\big(\nabla^2 f(x^*)\big)\, \|d\|^2 - o(\alpha^2) > 0$ for all sufficiently small $\alpha \neq 0$ when $\nabla^2 f(x^*) \succ 0$. □

The SOSC will play an important role when we analyze Newton's method: it is precisely the assumption $\nabla^2 f(x^*) \succ 0$ that makes the Newton step well defined and turns it into a descent direction (Section 5).

4 Gradient methods

4.1 Steepest descent

Suppose $\nabla f(x^{(k)}) \neq 0$. We seek a search direction $d^{(k)}$ that decreases f, and the FONC suggests $d^{(k)} = -\nabla f(x^{(k)})$. This choice is justified by the following variational identity.

Proposition 4.1. *Among unit vectors, the negative gradient is the direction of fastest local decrease of f:*

$$\arg\min_{\|d\|=1} \; \nabla f(x^{(k)})^\top d \;=\; -\frac{\nabla f(x^{(k)})}{\big\|\nabla f(x^{(k)})\big\|}.$$

Proof. By the Cauchy–Schwarz inequality $|\nabla f(x^{(k)})^\top d| \leq \big\|\nabla f(x^{(k)})\big\|\, \|d\|$, with equality precisely when d is a positive multiple of $\pm\nabla f(x^{(k)})$. The minimum is attained by the negative. □

Steepest descent with exact line search. Given the direction $d^{(k)} = -\nabla f(x^{(k)})$, choose the step size α_k as a global minimizer of the 1D function

$$\phi_k(\alpha) \;:=\; f\big(x^{(k)} - \alpha\, \nabla f(x^{(k)})\big).$$

By the FONC applied to ϕ_k,

$$\phi_k'(\alpha_k) \;=\; -\,\nabla f(x^{(k+1)})^\top \nabla f(x^{(k)}) \;=\; 0,$$

so consecutive steepest–descent gradients are orthogonal:

$$\nabla f(x^{(k+1)})^\top \nabla f(x^{(k)}) \;=\; 0. \tag{6}$$

This orthogonality is responsible for the characteristic *zigzag* pattern of steepest descent on ill–conditioned quadratics.

4.2 Convergence of steepest descent on a quadratic

To analyse convergence we specialise to the simplest non–trivial test function: a strictly convex quadratic.

$$f(x) \;=\; \tfrac{1}{2}x^\top Q x \;-\; b^\top x, \qquad Q = Q^\top \succ 0,\;\; b \in \mathbb{R}^n. \tag{7}$$

Direct calculation gives $\nabla f(x) = Qx - b$, hence the unique minimizer is $x^* = Q^{-1}b$, satisfying $\nabla f(x^*) = 0$.

It will be convenient to measure the quality of an iterate by the following "energy".

Lemma 4.2 (Energy reformulation)**.** *Define $V(x) := (x - x^*)^\top Q(x - x^*)$. Then*

$$f(x) - \min_y f(y) \;=\; \tfrac{1}{2}\, V(x).$$

In particular V measures the optimality gap of the objective up to the constant factor $\frac{1}{2}$.

Proof. Expand f around $x^* = Q^{-1}b$:

$$\begin{aligned} f(x) \;&=\; \tfrac{1}{2}x^\top Q x - b^\top x \\ &=\; \tfrac{1}{2}(x - x^*)^\top Q(x - x^*) \;-\; \tfrac{1}{2}x^{*\top} Q x^* \end{aligned}$$

(use $Qx^* = b$ to absorb the cross terms). The term $-\frac{1}{2}x^{*\top}Qx^*$ equals $\min_y f(y)$, so $f(x) - \min f = \frac{1}{2}V(x)$. □

Theorem 4.3 (Steepest descent on a quadratic, exact line search)**.** *Let $g^{(k)} = \nabla f(x^{(k)}) = Qx^{(k)} - b$ and choose the exact line–search step*

$$\alpha_k = \frac{(g^{(k)})^\top g^{(k)}}{(g^{(k)})^\top Q g^{(k)}}.$$

Then

$$V(x^{(k+1)}) \;=\; (1 - r_k)\, V(x^{(k)}), \qquad r_k \;=\; \frac{\big((g^{(k)})^\top g^{(k)}\big)^2}{\big((g^{(k)})^\top Q g^{(k)}\big)\,\big((g^{(k)})^\top Q^{-1} g^{(k)}\big)}.$$

Consequently, with $\kappa(Q) := \lambda_{\max}(Q)/\lambda_{\min}(Q)$ the condition number of Q,

$$V(x^{(k+1)}) \;\le\; \Big(1 - \frac{1}{\kappa(Q)}\Big) V(x^{(k)}).$$

Proof. Compute $V(x^{(k+1)}) - V(x^{(k)})$ from the definition. Using $x^{(k+1)} - x^* = (x^{(k)} - x^*) - \alpha_k g^{(k)}$ and expanding, the cross terms simplify because $g^{(k)} = Q(x^{(k)} - x^*)$. The result, after substituting the line–search optimal α_k, is

$$V(x^{(k+1)}) - V(x^{(k)}) \;=\; -\,\frac{\big((g^{(k)})^\top g^{(k)}\big)^2}{(g^{(k)})^\top Q g^{(k)}}.$$

Combining this with $V(x^{(k)}) = (g^{(k)})^\top Q^{-1} g^{(k)}$ (another easy calculation: $g^{(k)} = Q(x^{(k)} - x^*)$ implies $x^{(k)} - x^* = Q^{-1} g^{(k)}$) yields the displayed ratio.

For the bound on r_k, recall that for a positive definite matrix A, $x^\top A x \le \lambda_{\max}(A)\,\|x\|^2$ and $x^\top A^{-1} x \le \lambda_{\min}(A)^{-1}\,\|x\|^2$. Applying these to the denominator gives $r_k \ge 1/\kappa(Q)$. □

The convergence is therefore *linear* (sometimes called "order 1"): the error contracts by a constant factor $1 - 1/\kappa(Q)$ per iteration. The constant depends only on the *spectrum* of Q. When Q is well conditioned ($\kappa(Q) \approx 1$) the contraction is fast; when Q is ill–conditioned ($\kappa(Q) \gg 1$) it is painfully slow. This is the practical reason ill–conditioned problems are hard for plain gradient methods.

4.3 Gradient descent with constant step size

Exact line search is expensive (it is itself a 1D minimization problem at every iteration). In practice we often replace it by a *constant* step size $\alpha_k = \alpha$ chosen up front.

For the quadratic (7),

$$x^{(k+1)} \;=\; x^{(k)} - \alpha(Qx^{(k)} - b),$$

so subtracting $x^* = Q^{-1}b$ from both sides,

$$x^{(k+1)} - x^* \;=\; (I - \alpha Q)\big(x^{(k)} - x^*\big).$$

Taking norms,

$$\left\|x^{(k+1)} - x^*\right\| \;\le\; \|I - \alpha Q\|\;\left\|x^{(k)} - x^*\right\|.$$

When does this contract? For symmetric $Q \succ 0$ with eigenvalues $\lambda_1 \le \cdots \le \lambda_n$, the spectral norm is

$$\|I - \alpha Q\| \;=\; \max_i\, |1 - \alpha\lambda_i|.$$

The condition $\|I - \alpha Q\| < 1$ becomes $|1 - \alpha\lambda_i| < 1$ for every i, i.e.

$$0 \;<\; \alpha \;<\; \frac{2}{\lambda_{\max}(Q)}.$$

The convergence is linear with ratio $\|I - \alpha Q\|$.

Best constant step size. The optimal α minimizes $\max_i |1 - \alpha\lambda_i|$ and is given by

$$\alpha^* \;=\; \frac{2}{\lambda_{\max}(Q) + \lambda_{\min}(Q)},$$

yielding contraction factor $\|I - \alpha^* Q\| = \dfrac{\kappa(Q) - 1}{\kappa(Q) + 1}$, again a condition–number–limited rate. This rate is comparable to (and sometimes slightly worse than) the exact–line–search rate of Theorem 4.3.

Beyond constant step size. In practice we may not know $\lambda_{\max}(Q)$ or, indeed, may not have a quadratic f at all. Two more general strategies are:

- line–search algorithms that pick α_k on the fly to guarantee descent (we will see Armijo backtracking in Section 5);
- adaptive ("learning rate") schedules that update α_k based on observed progress.

A full convergence theory beyond the quadratic case is outside the scope of this course.

5 Newton's method

Gradient methods use only *first*–order derivative information. Newton's method goes one order higher: it uses the Hessian $\nabla^2 f$.

5.1 Derivation in $\mathbb{R}^n$

Idea: approximate f by a quadratic. Just as in 1D, we approximate f near the current iterate by its second–order Taylor polynomial,

$$f(x) \;\approx\; p_k(x) \;:=\; f(x^{(k)}) + \nabla f(x^{(k)})^\top (x - x^{(k)}) + \tfrac{1}{2}(x - x^{(k)})^\top \nabla^2 f(x^{(k)})(x - x^{(k)}), \quad (8)$$

and minimize p_k exactly to get $x^{(k+1)}$. Setting $\nabla p_k(x) = 0$ gives

$$\nabla f(x^{(k)}) \;+\; \nabla^2 f(x^{(k)})(x - x^{(k)}) \;=\; 0,$$

hence

$$x^{(k+1)} \;=\; x^{(k)} - \left[\nabla^2 f(x^{(k)})\right]^{-1} \nabla f(x^{(k)}). \quad (9)$$

For this to make sense we need $\nabla^2 f(x^{(k)})$ to be invertible.

Exact for quadratics. For $f(x) = \frac{1}{2} x^\top Q x - b^\top x$ with $Q \succ 0$, the Taylor model (8) is *equal* to f, so a single Newton step from any starting point $x^{(0)}$ produces the exact minimizer:

$$x^{(1)} \;=\; x^{(0)} - Q^{-1}(Q x^{(0)} - b) \;=\; Q^{-1} b \;=\; x^*.$$

This is the multidimensional analog of the 1D fact: Newton is exact for quadratics in one step.

5.2 Local quadratic convergence

Theorem 5.1 (Local convergence of Newton in $\mathbb{R}^n$)**.** *Let $f \in C^3$ on a neighborhood of x^*, with $\nabla f(x^*) = 0$ and $\nabla^2 f(x^*) \succ 0$. Then there exists $\delta > 0$ such that, whenever $\left\|x^{(k)} - x^*\right\| \le \delta$, the Newton iterate (9) satisfies*

$$\left\|x^{(k+1)} - x^*\right\| \;\le\; C \left\|x^{(k)} - x^*\right\|^2$$

for a constant C depending only on local bounds on $\nabla^2 f$ and its derivatives.

Proof. Subtract x^* from (9) and use $\nabla f(x^*) = 0$:

$$x^{(k+1)} - x^* \;=\; x^{(k)} - x^* - \big[\nabla^2 f(x^{(k)})\big]^{-1} \nabla f(x^{(k)}).$$

Multiply through by $\nabla^2 f(x^{(k)})$:

$$\nabla^2 f(x^{(k)})\big(x^{(k+1)} - x^*\big) \;=\; \nabla^2 f(x^{(k)})(x^{(k)} - x^*) - \big(\nabla f(x^{(k)}) - \nabla f(x^*)\big).$$

By the mean–value form of Taylor's theorem applied to ∇f,

$$\nabla f(x^{(k)}) - \nabla f(x^*) \;=\; \nabla^2 f(\xi)\big(x^{(k)} - x^*\big)$$

for some ξ on the segment between $x^{(k)}$ and x^*. Hence

$$\left\|x^{(k+1)} - x^*\right\| \;\le\; \|[\nabla^2 f(x^{(k)})]^{-1}\| \, \|\nabla^2 f(x^{(k)}) - \nabla^2 f(\xi)\| \, \left\|x^{(k)} - x^*\right\|.$$

Lipschitz continuity of $\nabla^2 f$ near x^* gives $\left\|\nabla^2 f(x^{(k)}) - \nabla^2 f(\xi)\right\| \le L \left\|x^{(k)} - \xi\right\| \le L \left\|x^{(k)} - x^*\right\|$. Local positive definiteness of $\nabla^2 f$ keeps $\left\|[\nabla^2 f(x^{(k)})]^{-1}\right\|$ bounded. Putting the pieces together gives $\left\|x^{(k+1)} - x^*\right\| \le C \left\|x^{(k)} - x^*\right\|^2$. □

Comparison with the 1D case. The structure is identical to the proof of Theorem 2.7; the only new ingredient is the mean–value form of Taylor's theorem in $\mathbb{R}^n$, which keeps the algebra clean. Crucially, the convergence is again *quadratic* but only *locally*: we need $x^{(k)}$ to be already close to x^* for the analysis to apply.

5.3 Globalization 1: Newton as a search direction

When we are far from the optimum, Newton's method may not even be defined (the Hessian may be singular or indefinite) and need not descend. A standard fix is to use the Newton vector as a *direction* only, and choose the step size separately:

$$d^{(k)} \;=\; -[\nabla^2 f(x^{(k)})]^{-1} \nabla f(x^{(k)}), \qquad x^{(k+1)} \;=\; x^{(k)} + \alpha_k \, d^{(k)}. \tag{10}$$

The full Newton step corresponds to $\alpha_k = 1$; near the optimum we will indeed take $\alpha_k = 1$ and recover the quadratic convergence rate.

Proposition 5.2 (Newton direction is descent)**.** *Suppose $\nabla^2 f(x^{(k)}) \succ 0$ and $\nabla f(x^{(k)}) \neq 0$. Then the Newton direction (10) is a* descent direction*, in the sense that*

$$\nabla f(x^{(k)})^\top d^{(k)} \;<\; 0.$$

Consequently there exists $\bar\alpha > 0$ such that $f\big(x^{(k)} + \alpha d^{(k)}\big) < f(x^{(k)})$ for all $\alpha \in (0, \bar\alpha]$.

Proof. Compute

$$\nabla f(x^{(k)})^\top d^{(k)} \;=\; -\nabla f(x^{(k)})^\top \big[\nabla^2 f(x^{(k)})\big]^{-1} \nabla f(x^{(k)}) \;<\; 0,$$

since the inverse of a positive definite matrix is positive definite, and $\nabla f(x^{(k)}) \neq 0$. The second statement follows from a Taylor expansion of $\alpha \mapsto f(x^{(k)} + \alpha d^{(k)})$ at $\alpha = 0$: the first-order term has slope $\nabla f(x^{(k)})^\top d^{(k)} < 0$, so for α small, f strictly decreases. □

5.4 Globalization 2: Levenberg–Marquardt

What if $\nabla^2 f(x^{(k)})$ is *not* positive definite? Then Newton's direction need not descend, and may not even be defined. The *Levenberg–Marquardt* (LM) idea is to replace the Hessian by a modified version that *is* positive definite:

$$\nabla^2 f(x^{(k)}) \;\longrightarrow\; \nabla^2 f(x^{(k)}) + \mu_k\, I,$$

where $\mu_k > 0$ is chosen large enough that the sum is positive definite. Concretely, taking

$$\mu_k \;>\; -\lambda_{\min}\big(\nabla^2 f(x^{(k)})\big)$$

suffices, since $\lambda_{\min}\big(\nabla^2 f(x^{(k)}) + \mu_k I\big) = \lambda_{\min}\big(\nabla^2 f(x^{(k)})\big) + \mu_k > 0$.

The *LM direction* is

$$d^{(k)} \;=\; -\big(\nabla^2 f(x^{(k)}) + \mu_k I\big)^{-1} \nabla f(x^{(k)}). \tag{11}$$

Two limiting cases give it a clean interpretation:

- As $\mu_k \to 0$, $d^{(k)}$ tends to the Newton direction.
- As $\mu_k \to \infty$, $d^{(k)} \to -\mu_k^{-1} \nabla f(x^{(k)})$, which is parallel to the gradient descent direction.

The LM direction therefore *interpolates* between Newton (small μ_k, fast local convergence, no globalization) and gradient descent (large μ_k, slow but always descending). The schedule for μ_k is chosen so that μ_k shrinks as we get closer to the optimum, recovering Newton's quadratic rate.

5.5 Step size by line search: Armijo's rule

Even after fixing a descent direction, the choice of step size matters. A practical and theoretically clean rule is the *Armijo* (or backtracking) condition:

$$f\big(x^{(k)} + \alpha\, d^{(k)}\big) \;\le\; f(x^{(k)}) + c\,\alpha\, \nabla f(x^{(k)})^{\top} d^{(k)}, \tag{12}$$

for a fixed parameter $c \in (0,1)$ (typical choice $c = 10^{-4}$). The condition demands that the actual decrease in f is at least a fixed fraction c of the decrease predicted by the linear model.

Algorithm.

1. Set the trial step $\alpha = 1$.
2. If (12) holds, accept this α and stop.
3. Otherwise, replace α by $\beta\alpha$ for a fixed $\beta \in (0,1)$ (e.g. $\beta = 0.5$) and go back to step 2.

Proposition 5.3 (Termination of Armijo backtracking)**.** *If $d^{(k)}$ is a descent direction (i.e. $\nabla f(x^{(k)})^\top d^{(k)} < 0$), then the Armijo condition (12) holds for all sufficiently small $\alpha > 0$, so the backtracking loop terminates.*

Proof. Taylor expand:

$$f\big(x^{(k)} + \alpha d^{(k)}\big) \;=\; f(x^{(k)}) + \alpha\, \nabla f(x^{(k)})^\top d^{(k)} + O(\alpha^2).$$

Subtract $f(x^{(k)}) + c\alpha\, \nabla f(x^{(k)})^\top d^{(k)}$:

$$f(x^{(k)} + \alpha d^{(k)}) - f(x^{(k)}) - c\alpha\, \nabla f(x^{(k)})^\top d^{(k)} \;=\; (1-c)\alpha\, \nabla f(x^{(k)})^\top d^{(k)} + O(\alpha^2).$$

Since $c < 1$ and $\nabla f(x^{(k)})^\top d^{(k)} < 0$, the leading α–term is strictly negative, so for α small the sum is also negative, which is exactly (12). □

6 Conjugate direction and conjugate gradient methods

Newton–type methods are powerful but expensive: each iteration needs the Hessian (an $n \times n$ matrix) and the solution of a linear system, costing $O(n^3)$ in general. For very large n (common in modern applications) this is prohibitive. We now develop a class of methods that work for the quadratic test problem (7) *without* forming the Hessian, and which are guaranteed to terminate in at most n steps.

6.1 Conjugate directions

Throughout this section $Q = Q^\top \succ 0$.

Definition 6.1 (Q–conjugate vectors)**.** Vectors $d^{(0)}, d^{(1)}, \ldots, d^{(k)} \in \mathbb{R}^n$ are *Q–conjugate* if

$$(d^{(i)})^\top Q\, d^{(j)} \;=\; 0 \qquad \text{for all } i \neq j.$$

Lemma 6.2. *Nonzero Q–conjugate vectors are linearly independent.*

Proof. Suppose $\sum_i \lambda_i d^{(i)} = 0$. Multiply both sides on the left by $(d^{(j)})^\top Q$ to get $\lambda_j (d^{(j)})^\top Q d^{(j)} = 0$. Since $Q \succ 0$ and $d^{(j)} \neq 0$, $(d^{(j)})^\top Q d^{(j)} > 0$, so $\lambda_j = 0$ for each j. □

In particular, n nonzero Q–conjugate vectors form a basis of $\mathbb{R}^n$.

6.2 The conjugate direction iteration

Suppose we are given Q–conjugate vectors $d^{(0)}, \ldots, d^{(n-1)}$. Define the iteration

$$x^{(k+1)} \;=\; x^{(k)} \;+\; \alpha_k\, d^{(k)}, \tag{13}$$

where α_k is chosen by exact line search: $\alpha_k = \arg\min_\alpha f(x^{(k)} + \alpha d^{(k)})$. Setting the derivative to zero,

$$\big(\nabla f(x^{(k)} + \alpha_k d^{(k)})\big)^\top d^{(k)} \;=\; \big(Q(x^{(k)} + \alpha_k d^{(k)}) - b\big)^\top d^{(k)} \;=\; 0,$$

which solves to

$$\alpha_k \;=\; -\frac{(g^{(k)})^\top d^{(k)}}{(d^{(k)})^\top Q d^{(k)}}, \qquad g^{(k)} := \nabla f(x^{(k)}) = Qx^{(k)} - b.$$

Theorem 6.3 (Finite–step convergence)**.** *For the quadratic* (7)*, the conjugate direction iteration* (13) *satisfies* $x^{(n)} = x^*$*. In words: the algorithm converges* exactly *in at most* n *steps.*

Proof. We show by induction on k that the error $x^{(k+1)} - x^*$ is Q–conjugate to all the directions $d^{(0)}, \dots, d^{(k)}$.
Base case $k = 0$. Compute, using $g^{(0)} = Q(x^{(0)} - x^*)$:

$$\begin{aligned}(x^{(1)} - x^*)^\top Q d^{(0)} \;&=\; \big(x^{(0)} + \alpha_0 d^{(0)} - x^*\big)^\top Q d^{(0)} \\ &=\; (x^{(0)} - x^*)^\top Q d^{(0)} + \alpha_0 (d^{(0)})^\top Q d^{(0)} \\ &=\; (g^{(0)})^\top d^{(0)} + \alpha_0 (d^{(0)})^\top Q d^{(0)} \;=\; 0,\end{aligned}$$

by the choice of α_0.
Induction step. Assume $(x^{(k)} - x^*)^\top Q d^{(i)} = 0$ for $i < k$. For $i < k$,

$$(x^{(k+1)} - x^*)^\top Q d^{(i)} = (x^{(k)} - x^*)^\top Q d^{(i)} + \alpha_k (d^{(k)})^\top Q d^{(i)} = 0 + 0,$$

the first term by the induction hypothesis and the second by Q–conjugacy. For $i = k$, repeat the calculation of the base case.
After n steps, $x^{(n)} - x^*$ is Q–conjugate to the basis $d^{(0)}, \dots, d^{(n-1)}$ of $\mathbb{R}^n$, hence $Q(x^{(n)} - x^*) = 0$, hence (since Q is invertible) $x^{(n)} = x^*$. □

Proposition 6.4 (Subspace optimality / global effect)**.** *The iterate* $x^{(k+1)}$ *is the minimizer of* f *over the affine subspace* $x^{(0)} + \operatorname{span}\{d^{(0)}, \dots, d^{(k)}\}$*. Equivalently,*

$$\nabla f(x^{(k+1)})^\top d^{(i)} \;=\; 0 \qquad \text{for } i = 0, 1, \dots, k.$$

Proof. Consider the constrained problem $\min_{(\alpha_0,\dots,\alpha_k)} f\big(x^{(0)} + \sum_i \alpha_i d^{(i)}\big)$. By the FONC of this problem, the optimal coefficients $(\alpha_0^*, \dots, \alpha_k^*)$ satisfy $\nabla f(x^{(0)} + \sum_i \alpha_i^* d^{(i)})^\top d^{(i)} = 0$ for each i. The proof of Theorem 6.3 shows that the coefficients produced by the conjugate direction iteration satisfy exactly this condition. Since f is strictly convex quadratic, the FONC is also sufficient, so the iterate equals the constrained optimum. □

6.3 Conjugate gradient: building Q–conjugate directions on the fly

The remaining question is: *how do we obtain Q–conjugate directions in practice?* The conjugate gradient (CG) method generates them on the fly using only gradient information, avoiding the matrix Q altogether (other than implicitly through one matrix–vector product per iteration).

Recipe. Take the first direction to be the negative gradient: $d^{(0)} = -g^{(0)}$. At step $k+1$, build $d^{(k+1)}$ as the negative gradient *corrected* so that it is Q–conjugate to the previous direction:

$$d^{(k+1)} \;=\; -\,g^{(k+1)} + \beta_k\, d^{(k)}. \tag{14}$$

Imposing $(d^{(k+1)})^\top Q d^{(k)} = 0$ gives

$$0 \;=\; -(g^{(k+1)})^\top Q d^{(k)} + \beta_k (d^{(k)})^\top Q d^{(k)},$$

hence

$$\beta_k \;=\; \frac{(g^{(k+1)})^\top Q\, d^{(k)}}{(d^{(k)})^\top Q\, d^{(k)}}. \tag{15}$$

Theorem 6.5 (CG produces Q–conjugate directions)**.** *The directions* (14) *produced by CG satisfy* $(d^{(i)})^\top Q d^{(j)} = 0$ *for all* $i \neq j$, $i, j \in \{0, 1, \ldots, n-1\}$. *Moreover the residuals (gradients) are mutually orthogonal:*

$$(g^{(i)})^\top g^{(j)} \;=\; 0 \qquad \text{for } i \neq j.$$

Proof sketch. By construction $(d^{(k+1)})^\top Q d^{(k)} = 0$. For $j < k$ we use the identity $Q d^{(j)} = (g^{(j+1)} - g^{(j)})/\alpha_j$ together with the subspace optimality of Proposition 6.4, which gives $(g^{(k+1)})^\top d^{(j)} = 0$. Substituting,

$$(d^{(k+1)})^\top Q d^{(j)} \;=\; \big(-g^{(k+1)} + \beta_k d^{(k)}\big)^\top Q d^{(j)} \;=\; -(g^{(k+1)})^\top Q d^{(j)},$$

and the right hand side equals $-(g^{(k+1)})^\top (g^{(j+1)} - g^{(j)})/\alpha_j$, which is zero by subspace optimality applied to $g^{(k+1)}$ and the directions $d^{(0)}, \ldots, d^{(k)}$. Orthogonality of the gradients then follows since $g^{(k+1)} = -d^{(k+1)} + \beta_k d^{(k)}$ is a linear combination of vectors orthogonal (or Q–conjugate) to each $g^{(j)}$ with $j \leq k$. □

6.4 Three equivalent formulas for β_k

Direct use of (15) requires knowing Q (or at least applying it). In the quadratic case the following three formulas all agree with (15), but they only need *gradients*, no matrix:

$$\text{Hestenes–Stiefel:} \quad \beta_k^{\mathrm{HS}} \;=\; \frac{(g^{(k+1)})^\top (g^{(k+1)} - g^{(k)})}{(d^{(k)})^\top (g^{(k+1)} - g^{(k)})}, \tag{16}$$

$$\text{Polak–Ribière:} \quad \beta_k^{\mathrm{PR}} \;=\; \frac{(g^{(k+1)})^\top (g^{(k+1)} - g^{(k)})}{(g^{(k)})^\top g^{(k)}}, \tag{17}$$

$$\text{Fletcher–Reeves:} \quad \beta_k^{\mathrm{FR}} \;=\; \frac{(g^{(k+1)})^\top g^{(k+1)}}{(g^{(k)})^\top g^{(k)}}. \tag{18}$$

Equivalence on quadratics. The Hestenes–Stiefel formula uses $Q d^{(k)} = (g^{(k+1)} - g^{(k)})/\alpha_k$ to rewrite the numerator and denominator of (15) without Q. Polak–Ribière is obtained by using $(d^{(k)})^\top (g^{(k+1)} - g^{(k)}) = -(g^{(k)})^\top g^{(k)}$ (a consequence of orthogonality of the gradients and the recursion $d^{(k)} = -g^{(k)} + \beta_{k-1} d^{(k-1)}$). Fletcher–Reeves further simplifies the numerator using $(g^{(k+1)})^\top g^{(k)} = 0$.

Behavior for non–quadratic f. For non–quadratic objectives the formulas (16), (17), (18) are no longer equivalent, and the generated directions need not stay Q–conjugate. In practice the Fletcher–Reeves formula is the simplest and is the most common default. The line search must be done numerically (e.g. Armijo). A common safeguard is to *restart* CG every n iterations (setting $\beta_k = 0$) to recover from accumulated loss of conjugacy. Refinements are discussed in textbook Chapter 14.

7 Quasi–Newton methods

Newton's method is fast (locally quadratic) but expensive: at each iteration we need to compute and invert the Hessian. Quasi–Newton methods replace the inverse Hessian by a symmetric positive definite matrix $H_k \succ 0$, which is updated cheaply from one iteration to the next using only first–order information. The iteration is

$$x^{(k+1)} \;=\; x^{(k)} + \alpha_k d^{(k)}, \qquad d^{(k)} \;=\; -H_k\, g^{(k)}, \tag{19}$$

with $g^{(k)} := \nabla f(x^{(k)})$ and α_k chosen by line search. The design question is: *how to update H_k?*

7.1 The quasi–Newton (secant) condition

Set $\Delta x^{(k)} := x^{(k+1)} - x^{(k)}$ and $\Delta g^{(k)} := g^{(k+1)} - g^{(k)}$. For a quadratic objective $f(x) = \frac{1}{2}x^\top Q x - b^\top x$ we have $\nabla^2 f = Q$ and $g^{(k)} = Qx^{(k)} - b$, so

$$\Delta g^{(k)} \;=\; Q\,\Delta x^{(k)}, \qquad Q^{-1}\Delta g^{(k)} \;=\; \Delta x^{(k)}.$$

We would like our approximation H_{k+1} of Q^{-1} to mimic this relation:

$$H_{k+1}\,\Delta g^{(k)} \;=\; \Delta x^{(k)}. \tag{20}$$

This is the *quasi–Newton* (or *secant*) condition. More ambitiously, in the spirit of the conjugate–direction analysis, we might demand the stronger "finite memory" condition

$$H_{k+1}\,\Delta g^{(i)} \;=\; \Delta x^{(i)} \qquad \text{for all } i \le k, \tag{21}$$

which links H_{k+1} to *all* previously generated information.

Theorem 7.1 (Quasi–Newton is a conjugate direction method on quadratics)**.** *Suppose $\{H_k\}_{k\ge 0}$ are symmetric positive definite, satisfy (21), and the iteration (19) uses exact line search. Then on a strictly convex quadratic the generated directions $d^{(0)}, \dots, d^{(n-1)}$ are Q–conjugate. In particular, the algorithm terminates in at most n steps.*

Proof sketch. Show $(d^{(k)})^\top Q d^{(0)} = 0$ by direct computation: substitute $d^{(k)} = -H_k g^{(k)}$, use $Q\Delta x^{(0)} = \Delta g^{(0)}$, the secant relation $H_k \Delta g^{(0)} = \Delta x^{(0)}$, and the fact that exact line search at step 0 gives $(g^{(k)})^\top d^{(0)} = 0$ for $k \ge 1$ (Proposition 6.4). Iterate this argument by induction. □

7.2 Constructing H_k: rank–one and rank–two updates

The standard recipe is to start with $H_0 = \eta I$ for some $\eta > 0$ and then add a low–rank correction at each step:

$$H_{k+1} \;=\; H_k \;+\; R_k.$$

Different quasi–Newton methods differ in how R_k is built.

Symmetric rank–one (SR1)

The simplest choice is a rank–one symmetric matrix $R_k = \alpha_k\, z^{(k)}(z^{(k)})^\top$ for some scalar α_k and vector $z^{(k)}$. Imposing the secant condition $H_{k+1}\Delta g^{(k)} = \Delta x^{(k)}$ gives

$$\alpha_k\, z^{(k)}\big((z^{(k)})^\top \Delta g^{(k)}\big) \;=\; \Delta x^{(k)} - H_k \Delta g^{(k)}.$$

Hence $z^{(k)}$ must be a multiple of $\Delta x^{(k)} - H_k\Delta g^{(k)}$, and a short calculation (multiply both sides by $(\Delta g^{(k)})^\top$ to fix the scalar) yields the *symmetric rank–one update*

$$H_{k+1} \;=\; H_k \;+\; \frac{\big(\Delta x^{(k)} - H_k\Delta g^{(k)}\big)\big(\Delta x^{(k)} - H_k\Delta g^{(k)}\big)^\top}{(\Delta g^{(k)})^\top\big(\Delta x^{(k)} - H_k\Delta g^{(k)}\big)}. \tag{22}$$

Properties. The SR1 update satisfies the secant condition by construction, and on the quadratic test problem one can in fact verify that H_{k+1} satisfies the stronger condition (21), i.e. it is consistent with *all* previously gathered $\Delta g^{(i)}, \Delta x^{(i)}$ pairs. *Drawback:* H_{k+1} may fail to be positive definite even when H_k is. This can cause the algorithm to break down on a non–quadratic objective.

Rank–two updates: DFP and BFGS

To preserve positive definiteness one uses rank–*two* updates. The two classics are:

DFP (Davidon–Fletcher–Powell):

$$H_{k+1} \;=\; H_k \;+\; \frac{\Delta x^{(k)}(\Delta x^{(k)})^\top}{(\Delta g^{(k)})^\top \Delta x^{(k)}} \;-\; \frac{H_k\,\Delta g^{(k)}(\Delta g^{(k)})^\top H_k}{(\Delta g^{(k)})^\top H_k \Delta g^{(k)}}. \tag{23}$$

BFGS (Broyden–Fletcher–Goldfarb–Shanno): obtained by applying DFP to the inverse of H_k. BFGS is the de facto industry standard for quasi–Newton methods and is the backbone of many production optimization solvers (LBFGS being its limited–memory variant).

Both updates preserve positive definiteness, in the sense that if $H_k \succ 0$ and the curvature condition $(\Delta g^{(k)})^\top \Delta x^{(k)} > 0$ holds (which is automatic with exact line search on a strictly convex objective), then $H_{k+1} \succ 0$. The proofs are computational; consult the textbook.

8 Nonlinear least squares: Gauss–Newton

Setting. Least–squares problems arise whenever we want to fit a parametric model to data. We are given N data pairs (x_i, y_i), with x_i in some space $\mathcal{X}$ and $y_i \in \mathbb{R}$, plus a model function $g(x;\theta)$ parameterised by $\theta \in \mathbb{R}^p$. We want to choose θ so that $g(x_i;\theta) \approx y_i$ for every i. The least–squares formulation is

$$\min_{\theta \in \mathbb{R}^p} \; f(\theta) \;=\; \tfrac{1}{2} \sum_{i=1}^{N} \big(y_i - g(x_i;\theta)\big)^2 \;=\; \tfrac{1}{2}\, \|r(\theta)\|^2 , \tag{24}$$

where the residual map $r : \mathbb{R}^p \to \mathbb{R}^N$ stacks the per–point residuals $r_i(\theta) = y_i - g(x_i;\theta)$.

Example 8.1 (Linear model: line fitting)**.** Take $g(x;\theta) = \theta_1 x + \theta_2$, so the residual is $r_i = y_i - \theta_1 x_i - \theta_2$. This is a *linear* least squares problem and is the topic of Section 9.

Example 8.2 (Nonlinear model)**.** Take $g(x;\theta) = A \sin(\omega x + \phi)$ with parameters $\theta = (A, \omega, \phi) \in \mathbb{R}^3$. Now r_i depends nonlinearly on θ, and we are in the nonlinear least squares setting. Concrete examples are spread throughout the textbook.

8.1 Gradient and Hessian via the Jacobian

Let $J(\theta) := \partial r / \partial \theta \in \mathbb{R}^{N \times p}$ be the Jacobian of r. By the chain rule

$$\nabla f(\theta) \;=\; J(\theta)^{\top} r(\theta),$$

which is cheap to compute once J is known. Differentiating again,

$$\nabla^2 f(\theta) \;=\; J(\theta)^{\top} J(\theta) \;+\; S(\theta), \qquad S(\theta) \;=\; \sum_{i=1}^{N} r_i(\theta)\, \nabla^2 r_i(\theta),$$

where $S(\theta)$ collects the second derivatives of the *residuals*, weighted by the residuals themselves.

8.2 Gauss–Newton: drop S

A pure Newton step requires $J^{\top}J + S$ and its inverse. Two issues:

- The matrix S requires second derivatives of the residuals, which can be expensive or unavailable.
- Even when S is computable, $J^{\top}J + S$ may fail to be positive definite (because S is signed).

The *Gauss–Newton* idea is to drop S entirely and work with $J^{\top}J$ alone:

$$\theta^{(k+1)} \;=\; \theta^{(k)} \;-\; \alpha_k \big(J(\theta^{(k)})^{\top} J(\theta^{(k)})\big)^{-1} J(\theta^{(k)})^{\top} r(\theta^{(k)}). \tag{25}$$

Why this is reasonable. $J^\top J$ is always positive semi–definite ($x^\top J^\top J x = \|Jx\|^2 \geq 0$), and is positive definite as soon as J has full column rank. Moreover, near a good fit (small residuals) the dropped term S is small and $J^\top J$ is an excellent approximation of the true Hessian.

Combining with Levenberg–Marquardt. If we are not yet near a good fit, $J^\top J$ may still be poorly conditioned. Adding a regularising μI as in Section 5 gives the *Levenberg–Marquardt* direction

$$d^{(k)} \;=\; -\big(J(\theta^{(k)})^\top J(\theta^{(k)}) + \mu_k I\big)^{-1} J(\theta^{(k)})^\top r(\theta^{(k)}),$$

which is the standard tool for nonlinear least squares fitting and is implemented in essentially every scientific computing library.

Part II
Linear Systems, Least Squares, and the Pseudoinverse

9 Linear least squares

We now turn from generic objectives to the special structure of *linear systems*. Let $A \in \mathbb{R}^{m\times n}$, $b \in \mathbb{R}^m$, and consider the equation

$$Ax \;=\; b, \qquad x \in \mathbb{R}^n.$$

There are three possible regimes:

- **Square ($m = n$, A invertible).** Standard linear algebra: $x = A^{-1}b$.
- **Overdetermined ($m > n$).** More equations than unknowns; in general no exact solution. Use a *least squares* formulation to find the best approximate solution.
- **Underdetermined ($m < n$).** More unknowns than equations; infinitely many exact solutions. Use a *minimum norm* formulation to single out a canonical one.

Both non–square cases are handled by optimization, and we treat them in turn.

9.1 The overdetermined case

Assume $m \geq n$. Since $Ax = b$ is generally not solvable, we relax to

$$\min_{x\in\mathbb{R}^n} \; f(x) \;:=\; \|Ax - b\|^2 .$$

Theorem 9.1 (Least–squares solution)**.** *Suppose* $\operatorname{rank}(A) = n$ *(in particular* $m \geq n$*). Then the unique minimizer of* $\|Ax - b\|^2$ *is*

$$x^* \;=\; (A^\top A)^{-1} A^\top b. \tag{26}$$

Proof. Expand:

$$f(x) \;=\; (Ax-b)^\top (Ax-b) \;=\; x^\top A^\top A\, x \;-\; 2b^\top A\, x \;+\; b^\top b.$$

The function f is a convex quadratic in x, so the FONC is sufficient. Setting $\nabla f(x^*) = 2A^\top A x^* - 2A^\top b = 0$ gives the *normal equation*

$$A^\top A\, x^* \;=\; A^\top b.$$

By Lemma 9.2 below, $A^\top A$ is invertible under our rank assumption, so $x^* = (A^\top A)^{-1} A^\top b$. □

Lemma 9.2. *For any* $A \in \mathbb{R}^{m\times n}$, $\operatorname{rank}(A^\top A) = \operatorname{rank}(A)$. *In particular,* $A^\top A$ *is invertible if and only if* A *has full column rank.*

Proof. We show $\ker(A^\top A) = \ker(A)$, which implies the rank statement.

- ($\supseteq$) If $Ay = 0$, then $A^\top A y = A^\top 0 = 0$.
- ($\subseteq$) If $A^\top A y = 0$, then $\|Ay\|^2 = y^\top A^\top A y = 0$, so $Ay = 0$.

Hence the kernels coincide. By the rank–nullity theorem, $\operatorname{rank}(A^\top A) = n - \dim\ker(A^\top A) = n - \dim\ker(A) = \operatorname{rank}(A)$. □

9.2 Geometric interpretation

Write $A = [a_1 \;\cdots\; a_n]$ with columns $a_i \in \mathbb{R}^m$. Then $Ax = \sum_i x_i a_i$ is a linear combination of the columns, and so Ax ranges over $\operatorname{Col}(A)$, the column space. The least squares problem

$$\min_x \|Ax - b\|$$

is the same as: *find the linear combination of* $a_1, \ldots, a_n$ *that is closest, in the Euclidean distance sense, to* b. Geometrically, the optimal Ax^* is the orthogonal projection of b onto the subspace $\operatorname{Col}(A)$, so the residual $b - Ax^*$ is orthogonal to every column a_i:

$$a_i{}^\top (b - Ax^*) \;=\; 0 \qquad \text{for } i = 1, \ldots, n,$$

which compactly reads $A^\top(b - Ax^*) = 0$, i.e. the normal equation $A^\top A x^* = A^\top b$. This is the geometric "derivation" of the normal equations.

Example 9.3 (Line fitting)**.** Fit a line $y = mt + c$ to the data

$$(t_1, y_1) = (2, 3), \quad (t_2, y_2) = (3, 4), \quad (t_3, y_3) = (4, 5).$$

We seek $\theta = (m, c)$ minimizing $\sum_i (y_i - mt_i - c)^2$. Set up

$$A = \begin{pmatrix} t_1 & 1 \\ t_2 & 1 \\ t_3 & 1 \end{pmatrix} = \begin{pmatrix} 2 & 1 \\ 3 & 1 \\ 4 & 1 \end{pmatrix}, \qquad b = \begin{pmatrix} 3 \\ 4 \\ 5 \end{pmatrix}.$$

Compute

$$A^\top A = \begin{pmatrix} 29 & 9 \\ 9 & 3 \end{pmatrix}, \qquad A^\top b = \begin{pmatrix} 38 \\ 12 \end{pmatrix},$$

and solve $A^\top A\theta = A^\top b$ to get $\theta = (m, c) = (1, 1)$, i.e. the fitted line is $y = t + 1$. (This is textbook Example 12.2.)

Remark 9.4 (Recursive least squares)**.** When data arrive sequentially we may not want to recompute $(A^\top A)^{-1}$ from scratch. Suppose first we solve $\min \|A_0 x - b_0\|^2$, and then a new row ${a_1}^\top$ and value b_1 arrive. The new least–squares solution corresponds to

$$\min \left\| \begin{bmatrix} A_0 \\ {a_1}^\top \end{bmatrix} x - \begin{bmatrix} b_0 \\ b_1 \end{bmatrix} \right\|^2.$$

A recursive formula updates $({A_0}^\top A_0)^{-1}$ to $\left({A_0}^\top A_0 + a_1 {a_1}^\top\right)^{-1}$ via the Sherman–Morrison identity, avoiding a full inversion. The details are in the textbook.

9.3 The underdetermined case: minimum–norm solution

Now suppose $m < n$ and $\operatorname{rank}(A) = m$. Then the system $Ax = b$ has infinitely many solutions: any x_0 satisfying $Ax_0 = b$ can be shifted by any element of $\ker(A)$ to give another solution. We single out the *minimum norm* solution by solving

$$\min_{x \in \mathbb{R}^n} \ \tfrac{1}{2} \|x\|^2 \quad \text{subject to} \quad Ax = b. \tag{27}$$

A 1D warm–up. Take $m = 1$, $n = 2$, so $A = a^\top \in \mathbb{R}^{1\times 2}$, $b \in \mathbb{R}$, and the constraint is the single line $a^\top x = b$. The minimum–norm point on this line is the orthogonal projection of 0 onto the line, which is $x^* = (b/\|a\|^2)\, a$. Note that $x^* = a^\top (aa^\top)^{-1} b$ in this notation, foreshadowing the general formula.

Theorem 9.5 (Minimum–norm formula)**.** *Assume* $\operatorname{rank}(A) = m$*. The unique solution of* (27) *is*

$$x^* \ = \ A^\top (AA^\top)^{-1} b. \tag{28}$$

Proof. First, x^* is feasible: $Ax^* = AA^\top (AA^\top)^{-1} b = b$, using that $AA^\top$ is invertible (an analogous lemma to Lemma 9.2). Now let x be any other feasible point. Decompose

$$\|x\|^2 = \|x - x^* + x^*\|^2 = \|x - x^*\|^2 + \|x^*\|^2 + 2(x - x^*)^\top x^*.$$

Compute the cross term:

$$(x - x^*)^\top x^* \ = \ (x - x^*)^\top A^\top (AA^\top)^{-1} b \ = \ \big(A(x - x^*)\big)^\top (AA^\top)^{-1} b \ = \ 0,$$

because $Ax = Ax^* = b$ implies $A(x - x^*) = 0$. Therefore

$$\|x\|^2 \ = \ \|x - x^*\|^2 + \|x^*\|^2 \ \geq \ \|x^*\|^2,$$

with equality iff $x = x^*$. □

9.4 Computing without the inverse: the Kaczmarz iteration

Inverting $AA^\top$ is wasteful when m is large (e.g. when A is sparse). The *Kaczmarz* algorithm offers a cheap iterative alternative.

Algorithm. Initialise $x^{(0)}$ (typically $x^{(0)} = 0$). At step k, pick an index $j_k \in \{1, \dots, m\}$ (cyclically or at random) and update

$$x^{(k+1)} \;=\; x^{(k)} \;+\; \mu\, \frac{b_{j_k} - {a_{j_k}}^\top x^{(k)}}{\|a_{j_k}\|^2}\, a_{j_k}, \qquad \mu \in (0,2), \tag{29}$$

where ${a_j}^\top$ is the jth row of A. Each iteration only touches the row a_{j_k} and the scalar b_{j_k}, so the cost per iteration is $O(n)$ for sparse rows.

Geometric interpretation. The constraint set $\{x : Ax = b\}$ is the intersection of m affine hyperplanes $\{x : {a_j}^\top x = b_j\}$. For $\mu = 1$, the update (29) is precisely the orthogonal projection of $x^{(k)}$ onto the hyperplane $\{x : {a_{j_k}}^\top x = b_{j_k}\}$. So the original Kaczmarz iteration is recursive *projection onto the constraint hyperplanes one at a time.* For $\mu \in (0,2)$, $\mu \neq 1$, it is a relaxed projection.

Choice of order. In practice the choice of j_k matters. Two standard choices:

- *Cyclic order:* $j_k = 1 + (k \bmod m)$ (the textbook default);
- *Randomised:* pick j_k uniformly at random, or proportional to $\|a_{j_k}\|^2$ (Strohmer–Vershynin).

Theorem 9.6 (Convergence of Kaczmarz)**.** *Starting from $x^{(0)} = 0$, the iterates of (29) converge to the minimum–norm solution $x^* = A^\top(AA^\top)^{-1}b$ of (27). Starting from a general $x^{(0)} \neq 0$, the iterates converge to the minimizer of $\|x - x^{(0)}\|$ subject to $Ax = b$, which is given by*

$$x^* \;=\; A^\top(AA^\top)^{-1}\big(b - Ax^{(0)}\big) + x^{(0)}.$$

Sketch. Each Kaczmarz update keeps $x^{(k+1)} - x^{(0)}$ in $\operatorname{span}\{a_1, \dots, a_m\} = \operatorname{Col}(A^\top)$. Therefore any limit lies in $x^{(0)} + \operatorname{Col}(A^\top)$. Combined with the feasibility condition $Ax = b$, linear algebra shows that the unique such point is the one in the theorem (when $x^{(0)} = 0$, this is the minimum–norm solution by Theorem 9.5; otherwise, apply the same result to the shifted variable $y = x - x^{(0)}$, which solves $Ay = b - Ax^{(0)}$). □

9.5 Summary so far

For $A \in \mathbb{R}^{m\times n}$ and $b \in \mathbb{R}^m$:

- $m = n$, $\operatorname{rank}(A) = n$: $x^* = A^{-1}b$ (square, invertible case).
- $m > n$, $\operatorname{rank}(A) = n$: $x^* = (A^\top A)^{-1}A^\top b$ (least squares).

- $m < n$, $\operatorname{rank}(A) = m$: $x^* = A^\top (AA^\top)^{-1} b$ (minimum norm), reachable iteratively by Kaczmarz.

The next section unifies all three formulas via the *pseudoinverse*, defined through the singular value decomposition.

10 The Singular Value Decomposition and the Pseudoinverse

10.1 Singular value decomposition

Theorem 10.1 (SVD)**.** *For every matrix $A \in \mathbb{R}^{m\times n}$ there exist orthogonal matrices $U \in \mathbb{R}^{m\times m}$ and $V \in \mathbb{R}^{n\times n}$, and positive numbers $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r > 0$ where $r = \operatorname{rank}(A)$, such that*

$$A \;=\; U\,\Sigma\,V^\top, \qquad \Sigma \;=\; \begin{pmatrix} \operatorname{diag}(\sigma_1, \ldots, \sigma_r) & 0 \\ 0 & 0 \end{pmatrix} \in \mathbb{R}^{m\times n}.$$

The numbers σ_i are the singular values *of A and are unique.*

Notation reminder. "U orthogonal" means $UU^\top = U^\top U = I$, and similarly for V. Orthogonal matrices preserve Euclidean lengths: $\|Ux\| = \|x\|$ for any vector x, because $\|Ux\|^2 = x^\top U^\top U x = x^\top x = \|x\|^2$.

Geometric meaning. Given an SVD $A = U\Sigma V^\top$, we can read off the action of A as a chain of three simple maps:

$$x \;\xrightarrow{V^\top}\; \text{(rotate)} \;\xrightarrow{\Sigma}\; \text{(stretch by } \sigma_i\text{)} \;\xrightarrow{U}\; \text{(rotate)}\,.$$

Every linear map between Euclidean spaces is the composition of a rotation, a coordinate–wise stretch, and another rotation.

An aside: SVD and compression. The SVD is the basis of low–rank approximation. Truncating to the top–k singular values (and corresponding columns of U, V) gives the best rank–k approximation of A in Frobenius and spectral norm (Eckart–Young). This is widely used to compress matrices — see for instance the SVD image compression demo on the course website.

10.2 The Moore–Penrose pseudoinverse

Definition 10.2 (Pseudoinverse via SVD)**.** Given $A = U\Sigma V^\top$ as in Theorem 10.1, define

$$A^+ \;:=\; V\,\Sigma^+\,U^\top, \qquad \Sigma^+ \;:=\; \begin{pmatrix} \operatorname{diag}(\sigma_1^{-1}, \ldots, \sigma_r^{-1}) & 0 \\ 0 & 0 \end{pmatrix} \in \mathbb{R}^{n\times m}.$$

A^+ is the *Moore–Penrose pseudoinverse* of A.

Example 10.3. For $\Sigma = \left(\begin{smallmatrix} 5 & 0 & 0 \\ 0 & 3 & 0 \end{smallmatrix}\right)$, the pseudoinverse is $\Sigma^+ = \left(\begin{smallmatrix} 1/5 & 0 \\ 0 & 1/3 \\ 0 & 0 \end{smallmatrix}\right)$.

Theorem 10.4. *Let $A \in \mathbb{R}^{m\times n}$ have rank $r \le \min(m,n)$. Then $x^* := A^+ b$ is the unique vector that*

1. *minimizes $\|Ax - b\|^2$, and*
2. *among all minimizers of (1), minimizes $\|x\|^2$.*

In particular, A^+ unifies the three regimes:

- *$m = n$, $\operatorname{rank}(A) = n$: $A^+ = A^{-1}$, recovering $x^* = A^{-1}b$.*
- *$m > n$, $\operatorname{rank}(A) = n$: $A^+ = (A^\top A)^{-1}A^\top$, recovering least squares.*
- *$m < n$, $\operatorname{rank}(A) = m$: $A^+ = A^\top(AA^\top)^{-1}$, recovering minimum norm.*

Sketch. Use the SVD $A = U\Sigma V^\top$ to substitute $\tilde{y} = V^\top x$, $\tilde{b} = U^\top b$. Then $\|Ax - b\|^2 = \left\|\Sigma\tilde{y} - \tilde{b}\right\|^2$ and $\|x\|^2 = \|\tilde{y}\|^2$ (by orthogonal invariance). Both problems decouple over the coordinates of $\tilde{y}$, where the minimization is immediate: $\tilde{y}_i = \tilde{b}_i/\sigma_i$ for $i \le r$ and $\tilde{y}_i = 0$ for $i > r$ (the second choice is forced by the minimum norm condition). Returning to the original variables gives $x^* = V\Sigma^+U^\top b = A^+ b$. □

Verification in the rank–full overdetermined case. Let $A \in \mathbb{R}^{m\times n}$ with $m \ge n = \operatorname{rank}(A)$. Then in the SVD,

$$\Sigma = \begin{pmatrix} \operatorname{diag}(\sigma_1, \dots, \sigma_n) \\ 0 \end{pmatrix} \in \mathbb{R}^{m\times n}.$$

A direct computation gives $A^\top A = V \operatorname{diag}(\sigma_i^2)\, V^\top$, so $(A^\top A)^{-1} = V \operatorname{diag}(\sigma_i^{-2})\, V^\top$, and

$$(A^\top A)^{-1}A^\top = V \operatorname{diag}(\sigma_i^{-2})\, V^\top V\Sigma^\top U^\top = V \operatorname{diag}(\sigma_i^{-2})\Sigma^\top U^\top = V\Sigma^+U^\top = A^+.$$

The underdetermined case is similar.

Remark 10.5. The textbook gives an axiomatic definition of A^+ via the *four Moore–Penrose conditions* (Definition 12.1.1), and proves that the SVD construction is equivalent (Exercise 12.1). The SVD definition has the advantage of being computationally explicit and visually transparent.

Part III
Linear and Constrained Optimization

11 Linear programming: motivation and standard form

We now turn to a different special structure: *linear* objective and *linear* constraints. This class of problems — the *linear programs* (LPs) — is so important that it has its

own dedicated theory and algorithms (the simplex method, due to Dantzig in the 1940s, is the prototype).

11.1 Two motivating examples

Example 11.1 (Production planning)**.** A factory produces two goods A and B. Producing one unit of A uses 5 units of resource and one unit of B uses 6 units; the total resource available is 30. A second constraint is $3x_1 + 2x_2 \leq 12$. Profits per unit are 1 for A and 5 for B. How many units of each should we make to maximize profit?

The model: let x_1 be the number of units of A and x_2 the number of units of B. Then we want

$$\max_{x_1, x_2 \geq 0} \; x_1 + 5x_2 \quad \text{s.t.} \quad 5x_1 + 6x_2 \leq 30, \;\; 3x_1 + 2x_2 \leq 12.$$

The objective and the constraints are all *linear* in (x_1, x_2).

Geometric solution. The feasible set is a polygon in $\mathbb{R}^2$ (the intersection of finitely many half–planes). Level sets of the objective $x_1 + 5x_2$ are parallel lines, and the optimum is reached by sliding such a line as far as possible in the direction of increasing objective while still touching the polygon. In this example the optimum is reached at a *vertex* of the polygon, namely $(x_1, x_2) = (0, 5)$, with objective value 25.

Example 11.2 (Diet problem)**.** Suppose there are n food types and m nutrients. Food i contains a_{ij} units of nutrient j and costs c_i dollars per unit. We need at least b_j units of nutrient j per day. Find the cheapest healthy diet:

$$\min_{x \in \mathbb{R}^n} \; c^\top x \quad \text{s.t.} \quad A^\top x \geq b, \;\; x \geq 0.$$

Two recurring observations. Both examples share two structural features:

1. the feasible set is a *polytope* — the intersection of finitely many half–spaces. Polytopes have "corners" which we will later call *extreme points* or *vertices*;

2. an optimal solution can always be found at one of these extreme points. This is a remarkable algorithmic gift: it reduces the problem from an infinite search to a finite one.

We will make both points precise.

11.2 Many faces of an LP

In the wild, LPs come in many forms: min vs max, equality vs inequality constraints, bounded vs free variables, etc. For a clean theoretical and algorithmic treatment we standardise to a single form.

Definition 11.3 (Standard form LP)**.** A linear program in *standard form* is

$$\min_{x\in\mathbb{R}^n} \; c^\top x \quad \text{subject to} \quad Ax = b, \;\; x \geq 0, \tag{30}$$

with data $A \in \mathbb{R}^{m\times n}$, $b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$.

Proposition 11.4 (Reduction to standard form)**.** *Every linear program can be put into standard form by the following moves.*

- ***Inequality*** $\leq$**.** $Ax \leq b$ *becomes* $Ax + y = b$, $y \geq 0$, *where* y *is a* slack variable.
- ***Inequality*** $\geq$**.** $Ax \geq b$ *becomes* $Ax - y = b$, $y \geq 0$, *where* y *is a* surplus variable.
- ***Free variable.*** *A free variable* $x_i \in \mathbb{R}$ *is replaced by the difference* $x_i = u_i - v_i$ *of two non–negative variables.*
- ***Maximization.*** $\max c^\top x$ *becomes* $\min(-c)^\top x$.

Example 11.5 (A messy LP put into standard form)**.** Consider

$$\max \; x_1 - x_2 \quad \text{s.t.} \quad 3x_1 = x_2 - 5, \;\; |x_2| \leq 2, \;\; x_1 \geq 0.$$

Steps:

1. Flip the objective: $\min \; -x_1 + x_2$.
2. Replace $x_2 = u - v$ (free variable) with $u, v \geq 0$. The equality becomes $3x_1 + u - v = 5$, $x_1 \geq 0$.
3. Replace $|x_2| \leq 2$ by $-2 \leq x_2 \leq 2$, then by $x_2 + x_3 = 2$ and $-x_2 + x_4 = 2$ with $x_3, x_4 \geq 0$. After substituting $x_2 = u - v$: $u - v + x_3 = 2$, $-u + v + x_4 = 2$.

The final standard form is

$$\min \; (-x_1 - u + v) \quad \text{s.t.} \quad \begin{aligned} 3x_1 + u - v &= 5, \\ u - v + x_3 &= 2, \\ -u + v + x_4 &= 2, \\ x_1, u, v, x_3, x_4 &\geq 0. \end{aligned}$$

(See HW5 for more examples.)

12 Basic feasible solutions and the fundamental theorem

The key conceptual move from *geometric* (extreme points) to *algebraic* (basic feasible solutions) is what makes the simplex method possible. Throughout this section we assume the LP is in standard form (30) and that $\operatorname{rank}(A) = m \leq n$.

12.1 Basis and basic feasible solution

Definition 12.1 (Basis, basic solution, BFS, degenerate BFS)**.**

- A *basis* is an $m \times m$ submatrix B of A formed by selecting m columns of A that are linearly independent (so B is invertible).
- Reorder the columns so that $A = [B\ D]$ and $x = (x_B, x_D)$ correspondingly. The vector x defined by
$$x_B = B^{-1}b, \qquad x_D = 0$$
is called a *basic solution* of $Ax = b$.
- If furthermore $x_B \geq 0$, the basic solution is called a *basic feasible solution (BFS)* of the LP.
- A BFS is *degenerate* if at least one entry of x_B is zero. Otherwise it is non–degenerate.

A basic solution has at most m nonzero coordinates (those in x_B, and possibly fewer if degenerate). Conversely, given a candidate point $x \geq 0$ satisfying $Ax = b$, the support $J = \{j : x_j > 0\}$ has $|J| \leq m$ if x is a BFS, and one can build a basis B by extending $\{a_j : j \in J\}$ to a maximal linearly independent set of columns.

12.2 Extreme points

Definition 12.2 (Extreme point)**.** A point x in a convex set Ω is an *extreme point* of Ω if it cannot be written as a non–trivial convex combination $x = \frac{1}{2}y + \frac{1}{2}z$ with $y, z \in \Omega$ and $y \neq z$. Equivalently, the only way to write x as the midpoint of a segment in Ω is the trivial one $y = z = x$.

For a polytope, the extreme points are exactly the geometric "corners." The polygon in Example 11.1 has finitely many extreme points (the vertices of the polygon), and we saw geometrically that the optimum is attained at one of them. The fundamental theorem below promotes that observation to a theorem for arbitrary LPs.

12.3 The fundamental theorem of LP

Theorem 12.3 (Fundamental theorem of linear programming)**.** *Consider the standard–form LP* (30) *with* $\operatorname{rank}(A) = m$*.*

1. *If a feasible solution exists, then a BFS exists.*
2. *If an optimal solution exists, then an optimal BFS exists.*
3. *x is a BFS if and only if x is an extreme point of the feasible polytope $\{x : Ax = b,\ x \geq 0\}$.*

Proof of (3). ($\Rightarrow$, BFS implies extreme point.) Let x be a BFS with support $J = \{j : x_j > 0\}$. By definition the columns $\{a_j : j \in J\}$ are linearly independent (they extend, possibly with the addition of zero–coordinate columns, to a basis B). Suppose for contradiction that $x = \frac{1}{2}y + \frac{1}{2}z$ with y, z feasible and $y \neq z$. For $j \notin J$ we have $x_j = 0$ and $y_j, z_j \geq 0$, so the average can only be zero if $y_j = z_j = 0$. Hence y and z are also supported on J. Restricted to the columns of J, $Ay = Az = b$ and the restricted matrix is injective (linear independence), so $y = z = x$. Contradiction.

($\Leftarrow$, extreme point implies BFS.) Let x be an extreme point and J as above.
Case (a): The columns $\{a_j : j \in J\}$ are linearly independent. Then $|J| \leq m$, and we may extend $\{a_j : j \in J\}$ to a set of m linearly independent columns to form a basis B. This makes x a (possibly degenerate, if $|J| < m$) BFS.
Case (b): The columns $\{a_j : j \in J\}$ are linearly dependent. Then there exist scalars y_j for $j \in J$, not all zero, with $\sum_{j \in J} y_j a_j = 0$. Extend y by zero outside J. For sufficiently small $\varepsilon > 0$ the perturbed points $x \pm \varepsilon y$ have $A(x \pm \varepsilon y) = Ax = b$ and remain non–negative (since $x_j > 0$ on J and $y_j = 0$ off J). Hence both $x + \varepsilon y$ and $x - \varepsilon y$ are feasible, and $x = \frac{1}{2}((x + \varepsilon y) + (x - \varepsilon y))$ is a non–trivial midpoint representation, contradicting extremality. □

Algorithmic consequence. The fundamental theorem reduces the LP to a finite combinatorial search: there are at most $\binom{n}{m}$ choices of basis, and an optimal solution sits at one of them. Brute force search would already give a finite algorithm, but $\binom{n}{m}$ is enormous in practice. The simplex method is an organised walk through the bases that decreases the objective at every step, visiting only a small number of bases before finding the optimum.

13 The simplex algorithm

13.1 Reduced cost and the optimality test

Fix a basis B with non–basis D, and write $x = (x_B, x_D)$, $c = (c_B, c_D)$. Since $Ax = b$ gives

$$Bx_B + Dx_D = b \quad \Longleftrightarrow \quad x_B = B^{-1}b - B^{-1}D\,x_D,$$

we can substitute into the objective:

$$c^\top x \;=\; {c_B}^\top x_B + {c_D}^\top x_D \;=\; {c_B}^\top B^{-1}b \;+\; \left({c_D}^\top - {c_B}^\top B^{-1}D\right) x_D \;=\; v + {r_D}^\top x_D, \quad (31)$$

where:

- $v := {c_B}^\top B^{-1}b$ is the objective value at the current BFS;
- ${r_D}^\top := {c_D}^\top - {c_B}^\top B^{-1}D$ is the *reduced cost* vector for the non–basic variables.

The interpretation: $r_{D,j}$ is the marginal change in the objective per unit increase in the non–basic variable x_j (holding the constraints satisfied by adjusting x_B).

Proposition 13.1 (Optimality test)**.** *If $r_D \geq 0$, the current BFS is optimal. If some entry $r_q < 0$, then increasing x_q from 0 strictly decreases $c^\top x$, so the current BFS is not optimal and we can do better.*

Proof. For the first part: increasing any $x_D \geq 0$ from zero leaves $c^\top x = v + {r_D}^\top x_D \geq v$. For the second: setting $x_q = \varepsilon > 0$ and the other non–basic variables to zero decreases $c^\top x$ by $-r_q\varepsilon > 0$. □

13.2 The pivot

Suppose $r_q < 0$ for some $q \notin B$ (we are entering the column a_q into the basis). Express a_q in the current basis:

$$a_q \;=\; \sum_{i=1}^{m} y_{i,q}\, a_{(i)}, \qquad y_{,q} = B^{-1} a_q.$$

Setting $x_q = \varepsilon > 0$ forces, by $Ax = b$, the basic variables to change as $x_{(i)} \mapsto x_{(i)} - \varepsilon\, y_{i,q}$. Each basic variable must remain non–negative, so the largest admissible ε is

$$\varepsilon^* \;=\; \min\Big\{ \frac{x_{(i)}}{y_{i,q}} \;:\; y_{i,q} > 0 \Big\},$$

attained at some index p. At $\varepsilon = \varepsilon^*$, the basic variable $x_{(p)}$ becomes zero — it leaves the basis — while x_q enters the basis. This is one *pivot step*: swap the columns $a_{(p)}$ and a_q in the basis. In the absence of degeneracy, the new BFS has strictly smaller objective.

The simplex algorithm consists in iterating pivot steps until the reduced cost vector is non–negative, at which point the current BFS is optimal.

13.3 Tableau implementation

In practice the pivot is implemented by row operations on the *tableau*

$$T \;=\; \left[\begin{array}{c|c} A & b \\ \hline c^\top & 0 \end{array}\right].$$

Bringing the basis columns to identity columns (via row reduction) makes the BFS, the reduced costs, and the next pivot all readable directly from the tableau. After each pivot, one applies row operations to (i) clear the pivot column to a unit vector and (ii) clear the bottom (cost) row in the pivot column.

Workflow.

1. Compute reduced costs (the bottom row). If they are all ≥ 0, the current BFS is optimal. Stop.

2. Otherwise, pick an entering column q with $r_q < 0$.

3. Compute the ratios $x_{(i)}/y_{i,q}$ for $y_{i,q} > 0$ and choose the leaving column p as the index attaining the minimum.

4. Pivot on entry (p, q): scale row p so that the pivot becomes 1, and use row operations to zero out the rest of column q (including the bottom row).

5. Go to step 1.

Example 13.2 (Textbook 16.2)**.** Solve

$$\max\ 2x_1 + 5x_2 \quad \text{s.t.} \quad x_2 \le 4,\ \ x_2 \le 6,\ \ x_1 + x_2 \le 8,\ \ x_1, x_2 \ge 0.$$

Standard form. Introduce slacks $x_3, x_4, x_5 \ge 0$:

$$\min\ -2x_1 - 5x_2 \quad \text{s.t.} \quad x_2 + x_3 = 4,\ \ x_2 + x_4 = 6,\ \ x_1 + x_2 + x_5 = 8,\ \ \text{all} \ge 0.$$

Initial tableau (basis $\{a_3, a_4, a_5\}$, BFS $x = (0, 0, 4, 6, 8)$, $r = (-2, -5, 0, 0, 0)$):

$$\left[\begin{array}{ccccc|c} 0 & 1 & 1 & 0 & 0 & 4 \\ 0 & 1 & 0 & 1 & 0 & 6 \\ 1 & 1 & 0 & 0 & 1 & 8 \\ \hline -2 & -5 & 0 & 0 & 0 & 0 \end{array}\right].$$

The most negative reduced cost is $r_2 = -5$, so x_2 enters. The minimum ratio test among $\{4/1, 6/1, 8/1\}$ gives 4, so the leaving variable is x_3 (row 1).

Pivot 1. After clearing the rest of column 2 (subtract row 1 from rows 2 and 3, add 5·row 1 to the bottom row), the tableau becomes

$$\left[\begin{array}{ccccc|c} 0 & 1 & 1 & 0 & 0 & 4 \\ 0 & 0 & -1 & 1 & 0 & 2 \\ 1 & 0 & -1 & 0 & 1 & 4 \\ \hline -2 & 0 & 5 & 0 & 0 & 20 \end{array}\right].$$

Now $r_1 = -2 < 0$, so x_1 enters. Ratios on positive entries in column 1: only $x_{(3)}/y_{3,1} = 4/1 = 4$. Pivot on $(3, 1)$.

Pivot 2. Add 2·row 3 to the bottom row to clear r_1:

$$\left[\begin{array}{ccccc|c} 0 & 1 & 1 & 0 & 0 & 4 \\ 0 & 0 & -1 & 1 & 0 & 2 \\ 1 & 0 & -1 & 0 & 1 & 4 \\ \hline 0 & 0 & 3 & 0 & 2 & 28 \end{array}\right].$$

All reduced costs are ≥ 0, so the algorithm stops. The optimal BFS is $x = (4, 4, 0, 2, 0)$, with $\max\ 2x_1 + 5x_2 = 2(4) + 5(4) = 28$. (In fact, the textbook example has slightly different data; the workflow above is faithful to the lecture treatment.)

Example 13.3 (Textbook 16.3)**.** Solve

$$\max 7x_1 + 6x_2 \quad \text{s.t.} \quad 2x_1 + x_2 \le 3, \;\; x_1 + 4x_2 \le 4, \;\; x_1, x_2 \ge 0.$$

Standard form. Add slacks $x_3, x_4 \ge 0$:

$$\min -7x_1 - 6x_2 \quad \text{s.t.} \quad 2x_1 + x_2 + x_3 = 3, \;\; x_1 + 4x_2 + x_4 = 4.$$

Starting from the slack basis $\{a_3, a_4\}$ (BFS $(0, 0, 3, 4)$, reduced cost $(-7, -6, 0, 0)$), the most negative entry is $r_1 = -7$, so x_1 enters. The ratio test gives $\min(3/2, 4/1) = 3/2$, so x_3 leaves. After the pivot the bottom row becomes $(0, -5/2, 7/2, 0)$, so x_2 enters. A second pivot brings the bottom row to be entirely ≥ 0 and the algorithm terminates. The optimal BFS reads off the final tableau.

Practical tip. *Always clear the basis columns to identity by row operations.* Once you do that, the BFS is the right column, the reduced costs are the bottom row, and the next pivot decision is immediate. Without this normalisation, you end up doing extra arithmetic.

Two issues we do not cover.

- *Degeneracy.* When the BFS is degenerate, the pivot step may produce zero progress, and in pathological cases the algorithm can cycle. Anti–cycling rules (Bland's rule, lexicographic ordering) prevent this; see the textbook.
- *Initial BFS.* When the LP has $\le$ constraints with $b \ge 0$, the slack basis is an obvious initial BFS. In general, finding an initial BFS is itself an LP, solved by the *two–phase simplex method* or the *big–M* method.

14 Duality

Every LP has an associated *dual* LP, whose theory provides both new structural insight and practical certificates of optimality.

14.1 The dual LP

Throughout this section the primal LP is

$$\text{(P)} \qquad \min c^\top x \quad \text{s.t.} \quad Ax = b, \;\; x \ge 0, \tag{32}$$

and the associated dual LP is

$$\text{(D)} \qquad \max \lambda^\top b \quad \text{s.t.} \quad A^\top \lambda \le c, \tag{33}$$

where $\lambda \in \mathbb{R}^m$ is unrestricted in sign.

14.2 Min–max derivation

The cleanest way to motivate (33) is via a min–max reformulation of the primal.

Step 1: enforce $Ax = b$ with a Lagrange multiplier. Introduce a multiplier $\lambda \in \mathbb{R}^m$ for the equality constraint and consider

$$\min_{x \geq 0} \; \max_{\lambda} \; c^\top x \; - \; \lambda^\top (Ax - b).$$

For any fixed x, the inner maximization over λ is

$$\max_{\lambda} \; -\lambda^\top (Ax - b) \; = \; \begin{cases} 0 & \text{if } Ax = b, \\ +\infty & \text{otherwise.} \end{cases}$$

So the outer minimization will only consider x satisfying $Ax = b$, and on those x the inner expression equals $c^\top x$. The min–max problem is therefore equivalent to the primal (32).

Step 2: swap min and max. Interchange min and max:

$$\max_{\lambda} \; \min_{x \geq 0} \; \left(c^\top - \lambda^\top A\right) x \; + \; \lambda^\top b.$$

For a fixed λ, the inner minimization is a linear function of $x \geq 0$. If *any* component of $c^\top - \lambda^\top A$ is negative, the inner min is $-\infty$. If $c^\top - \lambda^\top A \geq 0$ componentwise, the minimum is attained at $x = 0$ and equals $\lambda^\top b$. Hence

$$\min_{x \geq 0} \, (c^\top - \lambda^\top A)\, x + \lambda^\top b \; = \; \begin{cases} \lambda^\top b & \text{if } A^\top \lambda \leq c, \\ -\infty & \text{otherwise.} \end{cases}$$

The outer maximization is therefore precisely the dual (33).

14.3 Weak and strong duality

Theorem 14.1 (Weak duality)**.** *For any primal–feasible x and dual–feasible λ, $\lambda^\top b \leq c^\top x$. In particular, $D^* \leq P^*$, where P^* and D^* are the optimal values of (P) and (D), respectively.*

Proof. $\lambda^\top b = \lambda^\top (Ax) = (A^\top \lambda)^\top x \leq c^\top x$, where the inequality uses $A^\top \lambda \leq c$ and $x \geq 0$ component–wise. □

Corollary 14.2 (Optimality certificate)**.** *Suppose x_0 is primal–feasible, λ_0 is dual–feasible, and $c^\top x_0 = {\lambda_0}^\top b$. Then x_0 is optimal for (P) and λ_0 is optimal for (D).*

Proof. For any primal–feasible x, $c^\top x \geq {\lambda_0}^\top b = c^\top x_0$, so x_0 is optimal. Symmetric for λ_0. □

Theorem 14.3 (Strong duality)**.** *If (P) has an optimal solution, then $P^* = D^*$.*

The proof is via the simplex termination analysis (we omit the details and refer to the textbook).

14.4 KKT conditions for an LP

Combining strong duality with the form of (D), we obtain the following compact optimality characterisation.

Theorem 14.4 (KKT for the LP)**.** *A primal–dual pair (x, λ) is optimal for (32)–(33) if and only if*

$$Ax = b, \;\; x \geq 0, \;\; A^\top \lambda \leq c \qquad \textit{(primal and dual feasibility)}, \tag{34}$$

$$(c^\top - \lambda^\top A)\, x = 0 \qquad \textit{(complementary slackness)}. \tag{35}$$

Proof. ($\Rightarrow$) Optimality plus strong duality gives $c^\top x = \lambda^\top b = \lambda^\top A x = (A^\top \lambda)^\top x$, hence $(c^\top - \lambda^\top A)\, x = 0$.
($\Leftarrow$) From (35), $c^\top x = (A^\top \lambda)^\top x = \lambda^\top A x = \lambda^\top b$. Then Corollary 14.2 declares both points optimal. □

14.5 Duality for the asymmetric / symmetric form

The derivation above gave the dual of the standard–form LP (the *asymmetric* pairing). A frequently encountered variant is the symmetric form: starting from

$$\min \; c^\top x \quad \text{s.t.} \quad Ax \geq b, \;\; x \geq 0,$$

we may convert to standard form by adding surplus variables and then re–apply the recipe to obtain the symmetric dual

$$\max \; \lambda^\top b \quad \text{s.t.} \quad A^\top \lambda \leq c, \;\; \lambda \geq 0.$$

The two forms are interchangeable.

15 KKT conditions for general constrained problems

We close with a brief, conceptual treatment of *nonlinear* constrained optimization. Consider

$$\min_{x \in \mathbb{R}^n} \; f(x) \quad \text{s.t.} \quad g(x) \leq 0, \;\; h(x) = 0, \tag{36}$$

where $f : \mathbb{R}^n \to \mathbb{R}$, $g : \mathbb{R}^n \to \mathbb{R}^p$, $h : \mathbb{R}^n \to \mathbb{R}^q$ are C^1.

15.1 Lagrangian and min–max form

Introduce multipliers $\lambda \in \mathbb{R}^p_{\geq 0}$ for the inequality constraints and $\mu \in \mathbb{R}^q$ for the equality constraints, and form the *Lagrangian*

$$\mathcal{L}(x, \lambda, \mu) \;=\; f(x) \;+\; \lambda^\top g(x) \;+\; \mu^\top h(x).$$

The primal can be rewritten as

$$\min_{x} \; \max_{\lambda \geq 0,\, \mu} \; \mathcal{L}(x, \lambda, \mu).$$

(Just as in the LP case: if any constraint is violated, the inner max is $+\infty$, so the outer min avoids such x.) Swapping the order of min and max produces the *Lagrange dual*

$$\max_{\lambda \geq 0,\, \mu} \; \min_{x} \; \mathcal{L}(x, \lambda, \mu).$$

Weak duality (the dual value $\leq$ the primal value) holds in general; strong duality requires additional structure (e.g. convexity and a constraint qualification).

15.2 KKT conditions

Theorem 15.1 (KKT necessary conditions)**.** *Let x^* be a local minimizer of (36) satisfying a constraint qualification (so the active constraint gradients are linearly independent). Then there exist multipliers $\lambda^* \geq 0$ and μ^* such that*

$$\nabla f(x^*) + \sum_{i=1}^{p} \lambda_i^* \, \nabla g_i(x^*) + \sum_{j=1}^{q} \mu_j^* \, \nabla h_j(x^*) \;=\; 0 \qquad \textit{(stationarity)}, \tag{37}$$

$$g(x^*) \leq 0, \;\; h(x^*) = 0, \;\; \lambda^* \geq 0 \qquad \textit{(feasibility)}, \tag{38}$$

$$\lambda_i^* \, g_i(x^*) \;=\; 0 \quad \forall i \qquad \textit{(complementary slackness)}. \tag{39}$$

Specialisation to the LP. For an LP, every ∇g_i and ∇h_j is constant (since the constraints are linear), so the stationarity condition $c - A^\top \lambda = 0$ collapses to a feasibility condition for the dual. Likewise, the complementary slackness conditions for the primal non–negativity constraints become $\lambda_i^* \, x_i^* = 0$, which is exactly (35). KKT for the LP is therefore nothing more than the assertion that strong duality plus complementary slackness characterises optimality, just as in Theorem 14.4. KKT for general problems is the natural generalisation of the FONC of Theorem 3.2.

A Course summary

Generic unconstrained optimization.

- *1D search:* golden ratio (derivative–free, linear rate ≈ 0.618); bisection (uses f'); Newton's method (uses f', f'', locally quadratic).
- *General dimension:* steepest descent / gradient descent (with exact line search or constant step size), Newton's method, Levenberg–Marquardt, conjugate gradient, quasi–Newton.
- *Special structure:* Gauss–Newton and LM for nonlinear least squares.

Linear systems and least squares.

- $m = n$, $\operatorname{rank}(A) = n$: $\quad x^* = A^{-1} b$ (square, invertible).
- $m > n$, $\operatorname{rank}(A) = n$: $\quad x^* = (A^\top A)^{-1} A^\top b$ (least squares; minimizes $\|Ax - b\|^2$).

- $m < n$, $\operatorname{rank}(A) = m$: $x^* = A^\top (AA^\top)^{-1} b$ (minimum norm; minimizes $\|x\|^2$ subject to $Ax = b$). Reachable iteratively by Kaczmarz's projection algorithm.
- General rank: $x^* = A^+ b$ via the SVD; this simultaneously minimizes $\|Ax - b\|^2$ and, among such minimizers, $\|x\|^2$.

Linear programming.

- Standard form: $\min c^\top x$ s.t. $Ax = b$, $x \geq 0$.
- *Geometric view:* the feasible set is a polytope, and the optimum is attained at an extreme point (vertex).
- *Algebraic view:* the optimum is attained at a *basic feasible solution* (BFS). Theorem: BFS $\Leftrightarrow$ extreme point.
- *Algorithm:* simplex method, implemented as row operations on the tableau; pivot through bases until reduced costs are non–negative.
- *Duality:* weak and strong duality, derived via the min–max reformulation; KKT conditions for the LP collapse to complementary slackness.

General constrained problems.

- Lagrangian and min–max reformulation.
- KKT conditions: stationarity, primal/dual feasibility, and complementary slackness.

Reference. E. K. P. Chong and S. Żak, *An Introduction to Optimization*, 2nd ed. Supplementary: J. Nocedal and S. J. Wright, *Numerical Optimization*.