SECTION C: CONTINUOUS OPTIMISATION
LECTURE 3: STEEPEST DESCENT, GRADIENT SEARCH AND NEWTON'S METHOD

HONOUR SCHOOL OF MATHEMATICS, OXFORD UNIVERSITY
HILARY TERM 2005, DR RAPHAEL HAUSER

1. Naive Methods. In Lecture 2 we discussed the fairly general framework of line search descent methods, and we discussed the issue of step size selection in some detail. But the step size selection is only one aspect in which different algorithms may differ from one another. Another important aspect is the choice of the search direction d_k at the intermediate solution x_k.

Theorem 3.8 of Lecture 2 guarantees that a line-search descent method converges as long as the angle θ_k between d_k and $-\nabla f(x_k)$ is strictly smaller than and bounded away from $\pi/2$. This is the only requirement on d_k we identified so far. We will spend the next few lectures to derive and discuss specific choices search directions that satisfy this requirement.

1.1. The Steepest Descent Method. The most elementary descent method is the *steepest descent method* in which the search direction d_k is chosen as $-\nabla f(x_k)$. Since this guarantees that $\theta_k \equiv 0$, we obtain the following convergence result as an immediate corollary of Theorem 3.8 from Lecture 2:

THEOREM 1.1. *Suppose $f \in C^1(\mathbb{R}^n)$ has Lipschitz continuous gradients on $\mathbb{R}^n$ and is bounded below. Then the steepest descent algorithm combined with step lengths that satisfy the Wolfe conditions converges globally to a stationary point of the objective function.*

The steepest descent method seems appealing, since $-\nabla f(x_k)$ is the direction in which the objective function locally decreases at the fastest rate. Furthermore, $-\nabla f(x_k)$ is usually cheap to compute: using automatic differentiation software, one can easily turn a computer progamme that computes function values $f(x)$ into one that computes directional derivatives of f at approximately the same cost, so that computing $\nabla f(x_k$ takes about n times as much time as evaluating $f(x_k)$.

Unfortunately, despite its intuitive appeal, the steepest descent method has serious drawbacks, as it can have excruciatingly slow convergence rates (see Problem 3 of Problem Set 2), and it can be unreliable because roundoff errors cause it to oscillate even for simple convex quadratic objective functions $f(x) = a + c^{\mathrm{T}}x + (1/2)x^{\mathrm{T}}Bx$, where $a \in \mathbb{R}$, $c \in \mathbb{R}^n$ and B is a positive definite symmetric matrix. The steepest descent method behaves so badly in practice that some regard it as the ultimate Mickey Mouse method of optimisation!

1.2. The Coordinate-Search Method. Another naive method that converges even more slowly than the steepest descent method is *coordinate-search* in which the search direction d_k is chosen to coincide with one of the coordinate axes in each iteration. Many variants exist, the most popular being to choose $d_k = e_i$ where e_i is the i-th coordinate vector and $i = 1 + k \mod n$.

Note that evaluating this choice of d_k is even cheaper than computing the steepest-descent direction. Furthermore, a convergence result for the coordinate-search method derives from Theorem 3.8 of Lecture 2 in a similar way as Theorem 1.1.

Despite their problems with convergence, the steepest-descent and coordinate-search methods can be useful in situations where other methods are computationally too expensive to apply: the low cost per iteration of is a virtue of both algorithms.

2. Newton's Method. We will now jump from the discussion of the cheapest-to-compute and slowest-to-converge line-search methods directly to discussing the most-expensive-to-compute and fastest-to-converge method: the Newton–Raphson method, which sets the benchmark for convergence speed, both in theory and in practice. All other search-directions we discuss later can be seen as attempts to strike a balance between the cost per iteration and the number of iterations needed until convergence, with the methods of Section 1 on one end of the spectrum, and the Newton-Raphson method at the other.

2.1. The Newton-Raphson Method for Zero-Finding. Let us first consider how to find zeros of a differentiable univariate function g, that is, suppose we want to find $x^* \in \mathbb{R}$ such that $g(x^*) = 0$. If we are given an approximation x_k of x^* such that $g'(x_k) \neq 0$, then the linear function

$$\varphi : x \mapsto g(x_k) + g'(x_k)(x - x_k)$$

has the unique zero $x_{k+1} = x_k - g(x_k)/g'(x_k)$, see Figure 2.1. But φ is the first order

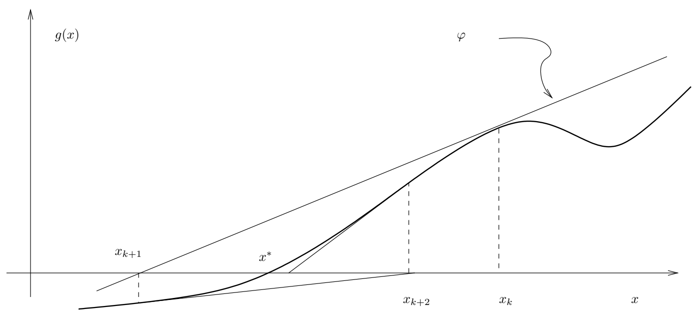

FIG. 2.1. *Newton's method for univariate zero-finding*

Taylor approximation of g around x_k, which locally approximates g well. Therefore, it is reasonable to expect that under benevolent conditions x_{k+1} should be an even better approximation of x^* than x_k, and that $(x_k)_{\mathbb{N}}$ converges to x^* if the same process is applied in an iterative fashion. This is the *Newton-Raphson method* for zero-solving. The "benevolent" conditions we need are simply that $g'(x^*) \neq 0$ and that x_k lies in sufficient proximity to x^*. This will follow from Theorem 2.6 below.

The multivariate version of the Newton-Raphson method is a direct generalisation: we now seek a root (or zero) of a multivariate function $g : \mathbb{R}^n \to \mathbb{R}^n$. Given an approximate root $x_k \in \mathbb{R}^n$, we find the next iterate x_{k+1} as the zero of the first-order Taylor approximation of g around x_k, or in other words, as the root of the linear system of equations $g(x_k) + J_g(x_k)(x - x_k) = 0$, where $J_g(x) = \left[\frac{\partial g_i}{\partial x_j}\right]$ is the Jacobian of g at x. When $J_g(x)$ is nonsingular, this linear system has the unique solution

$$x_{k+1} = x_k - J_g(x_k)^{-1} g(x_k).$$

Note that if $g \in C^1$ and $\det J(x^*) \neq 0$ then $J_g(x)$ is nonsingular for all x_k in a neighbourhood of x^*, and x_{k+1} is well defined.

Note that in this approach a difficult nonlinear problem is replaced by a sequence of easy linear problems. Much of numerical analysis follows similar approaches.

Example 2.1. *Let $A \in \mathbb{R}^{m\times n}$ be a $m \times n$ matrix with full row rank (that is, linearly independent row vectors). Let $b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$, $\mu \in \mathbb{R}_+$, and let $e \in \mathbb{R}^n$ be the vector of all ones*

$$e = \begin{bmatrix} 1 & \dots & 1 \end{bmatrix}^{\mathrm{T}}.$$

At the heart of interior-point methods for linear programming lies the solution of the nonlinear system of equations

$$\begin{aligned} Ax &= b && (2.1)\\ A^{\mathrm{T}}y + s &= c && (2.2)\\ XSe &= \mu e && (2.3)\\ x, s &> 0, && (2.4) \end{aligned}$$

where $x, s \in \mathbb{R}^n$, $y \in \mathbb{R}^m$, $X = \operatorname{diag}(x)$ and $S = \operatorname{diag}(s)$ are the diagonal matrices with x and s on their diagonals, and where $x, s > 0$ means that both vectors have to be component-wise strictly positive.

It can be shown that the system (2.1)-(2.4) has a unique solution (x^*, y^*, s^*). Given a current approximate solution (x, y, s) such that $x, s > 0$, we can compute a Newton step $(\Delta x, \Delta y, \Delta s)$ for the unconstrained system (2.1)-(2.3) which is obtained by solving the linear system of equations

$$\begin{aligned} A\Delta x &= b - Ax\\ A^{\mathrm{T}}\Delta y + \Delta s &= c - A^{\mathrm{T}}y - s\\ S\Delta x + X\Delta s &= \mu e - XSe. \end{aligned}$$

In order to guarantee that (2.4) continues to be satisfied, we use $(\Delta x, \Delta y, \Delta s)$ as a search-direction and determine an updated approximate solution (x_+, y_+, s_+) as follows:

$$\begin{aligned} \alpha^* &= \sup\{\alpha > 0 : \, x + \alpha\Delta x > 0, \, s + \alpha\Delta s > 0\},\\ (x_+, y_+, s_+) &= (x, y, s) + \min(1, 0.99 \times \alpha^*) \times (\Delta x, \Delta y, \Delta s). \end{aligned}$$

It can be shown that the resulting sequence of intermediate solutions converges very efficiently to (x^*, y^*, s^*).

2.2. Newton's Method for Unconstrained Minimisation. Let us now go back to the unconstrained optimisation problem

$$\min_{x\in\mathbb{R}^n} f(x). \tag{2.5}$$

The first order necessary optimality condition of Theorem 2.1 (i) of Lecture 2 tells us that (2.5) can be replaced by the multivariate root-finding problem

$$\nabla f(x) = 0. \tag{2.6}$$

When we apply the Newton-Raphson method to the zero-finding problem (2.6), we obtain the updating rule

$$x_{k+1} = x_k - \left(D^2 f(x_k)\right)^{-1} \nabla f(x_k), \tag{2.7}$$

which is well-defined as long as $D^2 f(x_k)$ is nonsingular. We call

$$n_f(x_k) := -\big(D^2 f(x_k)\big)^{-1} \nabla f(x_k)$$

the *Newton-step.*

ALGORITHM 2.2 (Newton-Raphson method). *Let $f \in C^2(\mathbb{R}^n)$ and let x_0 be an initial approximation of a local minimiser x^* of f. The Newton-Raphson method for solving* (2.5) *consists in computing the sequence of points $(x_k)_{\mathbb{N}}$ defined by*

$$x_{k+1} = x_k + n_f(x_k).$$

This is an excellent method for minimising convex functions, since $D^2 f(x_k) \succ 0$ implies

$$\langle n_f(x_k), \nabla f(x_k) \rangle = -\nabla f(x_k)^{\mathrm{T}} D^2 f(x_k) \nabla f(x_k) < 0,$$

that is, $n_f(x_k)$ is a descent direction. Furthermore, Theorem 2.6 below shows that the Newton-Raphson method converges to a stationary point x^* Q-quadratically, and since the local minimisers of a convex function are exactly characterised by the optimality condition $\nabla f(x^*) = 0$ (see Lecture 1), x^* must be a local minimiser.

Regrettably, when f is not convex, the Newton-Raphson method is not guaranteed to converge to local minimiser. Cycling can occur (i.e., $x_{k+j} = x_k$ for some $k, j \in \mathbb{N}$) but is unlikely in practice, but more importantly, the method can converge to a local maximiser or a saddle point of f.

EXAMPLE 2.3. *Consider the univariate objective function $f(x) = -x^2$. In a neighbourhood of the local maximiser $x^* = 0$, the Newton-Raphson method is attracted to x^*. In particular, if $x_k = 1$, then $d_k = -f''(1)^{-1} f'(1) = -(-2)/(-2) = -1$ is a direction of* ascent *rather than descent.*

This phenomenon is not surprising, as the algorithm is designed as a zero-finding method for $\nabla f(x) = 0$, and as $n_f(x_k)$ may fail to be a descent direction when $D^2 f(x_k)$ is not positive definite.

Two obvious ways to overcome these drawbacks are

i) either to use the Newton-Raphson method only in the final phase of an algorithm that otherwise consists of a line-search descent method (this is motivated by the fact that in a neighbourhood of a local minimiser f looks convex, and often strictly convex),

ii) or to use $n_f(x_k)$ as a search-direction which might have to be reversed if it fails to be a descent-direction.

We formalise the second approach in the following algorithm:

ALGORITHM 2.4 (Damped Newton method). *Let $f \in C^2(\mathbb{R}^n)$ and x_0 an initial approximation of a local minimiser of f. The damped Newton method uses*

$$d_k := \begin{cases} n_f(x_k) & \textit{if } \langle n_f(x_k), \nabla f(x_k) \rangle < 0, \\ -n_f(x_k) & \textit{otherwise} \end{cases}$$

as a search-direction and computes a sequence $(x_k)_{\mathbb{N}}$ via the line-search descent algorithm of Lecture 2 (Algorithm 3.2).

In order for this algorithm to pick up the superb convergence rates of the Newton-Raphson method, the line-search step lengths α_k have to approach the value 1 asymptotically, corresponding to *full Newton-Raphson steps.*

The next example shows that computing a Newton-Raphson step is not always trivial, because thinking of the problem space as a Euclidean space may not be that straight-forward:

EXAMPLE 2.5. *Let us consider the vector space $S^n \subset \mathbb{R}^{n\times n}$ of $n \times n$ symmetric matrices. We endow S^n with the Euclidean inner product $(X, S) \mapsto \langle X, S\rangle$ which is obtained by multiplying the matrices X and S component-wise and taking the sum of these products. A more convenient and equivalent way of thinking about this inner product is given by the identity*

$$\langle X, S\rangle = \operatorname{tr}\bigl(X^{\mathrm{T}} S\bigr),$$

where $\operatorname{tr}(A) = a_{11} + \cdots + a_{nn}$ denotes the trace of an $n \times n$ matrix A. We write $X \succ 0$ if $X \in S^n$ is positive definite. Then

$$S^n_{++} := \bigl\{X \in S^n : X \succ 0\bigr\}$$

is a convex open set in S^n, and the log-barrier function

$$\begin{aligned} f : S^n_{++} &\to \mathbb{R},\\ X &\mapsto -\ln\det X \end{aligned}$$

is well-defined. How can one compute a Newton-Raphson update for this objective function?

At first we need to compute gradient and Hessian of f, which is nontrivial. The function f is infinitely differentiable on S^n_{++}, since ln is C^∞ on $(0, \infty)$, and because $\det X$ is a polynomial in the components of X and takes values only in $(0, \infty)$ when $X \succ 0$. Note that

$$\det(I + tY) = 1 + t(\operatorname{tr} Y) + O(t^2). \tag{2.8}$$

This is because in the development of the determinant as a sum of monomials, whenever there appears a nondiagonal entry of $I + tY$ in a monomial there is at least one other nondiagonal entry that appears in the same monomial and hence all such monomials are of order $O(t^2)$. Moreover,

$$\det\bigl(I + t\,\mathrm{Diag}(Y)\bigr) = 1 + t(\operatorname{tr} Y),$$

where $\mathrm{Diag}(Y)$ is the diagonal matrix obtained by setting all off-diagonal elements of Y to zero. Substituting (2.8) into the first order Taylor development of the function $\ln(1 + x) = x + O(x^2)$ we obtain

$$\ln\det(I + tY) = t\operatorname{tr} Y + O(t^2).$$

Therefore,

$$\begin{aligned} \frac{d}{dt}\Big|_{t=0} \ln\det(I + tY) &= \lim_{t\to 0} \frac{1}{t}\left(\ln\det(I + tY) - \ln\det I\right)\\ &= \lim_{t\to 0} (\operatorname{tr} Y + O(t)) = \operatorname{tr} Y = \operatorname{tr}(I^{\mathrm{T}} Y), \end{aligned}$$

which shows that $\nabla f(I) = I$. It follows that

$$\begin{aligned}\frac{d}{dt}|_{t=0} \ln\det(X+tY) &= \lim_{t\to 0}\frac{1}{t}\left(\ln\det(X+tY) - \ln\det X\right)\\ &= \lim_{t\to 0}\frac{1}{t}\left(\ln\bigl(\det(X^{-1})\det(X+tY)\bigr)\right)\\ &= \lim_{t\to 0}\frac{1}{t}\left(\ln\det(I+tX^{-1}Y)\right) = \operatorname{tr}(X^{-1}Y),\end{aligned}$$

and this shows that $\nabla f(X) = X^{-T} = X^{-1}$ for all $X \in S^n_{++}$. In order to compute $D^2 f$ it suffices to determine the directional derivatives of $\nabla f(X)$. Arguing similarly as above, one obtains

$$\lim_{t\to 0}\frac{1}{t}\left(\nabla f(X+tS) - \nabla f(X)\right) = X^{-1}SX^{-1}.$$

Therefore, for all $0 \neq S \in S^n$ we have

$$\begin{aligned}D^2 f(X)\bigl(S,S\bigr) &= \operatorname{tr}\bigl(X^{-1}SX^{-1}S\bigr)\\ &= \operatorname{tr}\bigl((X^{-1/2}SX^{-1/2})^{\mathrm{T}}(X^{-1/2}SX^{-1/2})\bigr)\\ &= \|X^{-1/2}SX^{-1/2}\|^2 > 0,\end{aligned}$$

where $X^{1/2}$ is the unique positive definite symmetric matrix such that $X^{1/2}X^{1/2} = X$ (see problem set). This proves that $D^2 f(X)$ is positive definite for all $X \in S^n_{++}$, and we know from Lecture 1 that this implies that f is strictly convex on S^n_{++}. The Newton equation for the update matrix D is

$$D^2 f(X)[D] = -\nabla f(X).$$

The formulas derived above allow us now to rewrite this equation as

$$X^{-1}DX^{-1} = -X^{-1}.$$

This yields $D = -X$. Note that $-\nabla f(X) = -X^{-1}$ is the steepest-descent direction.

Let us now analyse the convergence speed of the Newton-Raphson method:

Theorem 2.6. *Let $f \in C^2(\mathbb{R}^n, \mathbb{R})$ have Λ-Lipschitz continuous Hessian and let $x^* \in \mathbb{R}^n$ be a stationary point of f. If $D^2 f(x^*)$ is nonsingular then there exists a neighbourhood $B_\rho(x^*)$ of x^* such that choosing $x_0 \in B_\rho(x^*)$ as a starting point and applying the Newton-Raphson method generates a sequence $(x_k)_{\mathbb{N}}$ that is is well-defined, lies in $B_\rho(x^*)$ and converges to x^* Q-quadratically.*

Proof. Since $D^2 f(x^*)$ is nonsingular and $D^2 f : D \to \mathbb{R}^{n\times n}$ is a continuous mapping, there exists a radius $\bar\rho > 0$ such that $D^2 f(x)$ is nonsingular for all $x \in B_{\bar\rho}(x^*)$. Moreover, $X \mapsto X^{-1}$ is a continuous mapping on $Gl_n(\mathbb{R})$ (the set of nonsingular $n \times n$ matrices). Thus, we may choose $\bar\rho$ sufficiently small so that

$$\|(D^2 f(x))^{-1}\| \le 2\|(D^2 f(x^*))^{-1}\| =: \beta, \tag{2.9}$$

where $\|\cdot\|$ denotes the canonical operator norm. Note that the choice of $\bar\rho$ guarantees that (2.7) is well-defined for $\|x_k - x^*\| < \bar\rho$. Subtracting x^* from both sides of (2.7) yields

$$(x_{k+1} - x^*) = (x_k - x^*) - (D^2 f(x_k))^{-1}\nabla f(x_k). \tag{2.10}$$

Using $\nabla f(x^*) = 0$ we find that

$$\begin{aligned}\nabla f(x_k) &= \nabla f(x_k) - \nabla f(x^*)\\ &= \int_{t=0}^{1} D^2 f(tx^* + (1-t)x_k)(x_k - x^*)dt\end{aligned}$$

Substituting this expression into (2.10), we obtain

$$(x_{k+1} - x^*) = \left(D^2 f(x_k)\right)^{-1} S\,(x_k - x^*), \tag{2.11}$$

where

$$\begin{aligned}S &= D^2 f(x_k) - \int_{t=0}^{1} D^2 f(tx^* + (1-t)x_k)dt\\ &= \int_{t=0}^{1} D^2 f(x_k) - D^2 f(tx^* + (1-t)x_k)dt.\end{aligned}$$

Taking norms on both sides of (2.11), we obtain

$$\|x_{k+1} - x^*\| \leq \|(D^2 f(x_k))^{-1}\| \times \|S\| \times \|x_k - x^*\|. \tag{2.12}$$

The Lipschitz continuity of $D^2 f$ implies that

$$\begin{aligned}\|S\| &\leq \int_{t=0}^{1} \|D^2 f(x_k) - D^2 f(tx^* + (1-t)x_k)\|dt\\ &\leq \int_{t=0}^{1} \Lambda t \|x_k - x^*\|dt = \frac{\Lambda}{2}\|x_k - x^*\|.\end{aligned}$$

Using this bound and (2.9) in (2.12) we find

$$\|x_{k+1} - x^*\| \leq \frac{\beta\Lambda}{2}\|x_k - x^*\|^2. \tag{2.13}$$

Finally, for $\rho := \min(\bar{\rho}, 2(\beta\Lambda)^{-1})$, (2.13) shows that

$$x_k \in B_\rho(x^*) \Rightarrow x_k \in B_\rho(x^*),$$

so that the entire sequence $(x_k)_{\mathbb{N}}$ is well defined as long as $x_0 \in B_\rho(x^*)$. $\square$

Theorem 2.6 shows that the Newton-Raphson method produces an output sequence $(x_k)_{\mathbb{N}}$ in which every x_{k+1} approximates x^* with roughly twice as many correct digits as x_k, as long as the process is started within $B_\rho(x^*)$. In applications it is a drawback that the radius ρ depends on the Hessian $D^2 f(x^*)$, which is of course unknown because x^* is unknown. The ball $B_\rho(x^*)$ is called the *domain of quadratic attraction* of x^*. During the 1990's, interesting new theories have been developed (Nesterov & Nemirovskii's theory of self–concordant functions, Smale's α-theory) which show that for certain classes of objective functions there exist mathematical criteria which make it possible to detect that x_k lies in the domain of quadratic attraction without knowing the location of x^*. The ensuing interior-point and homotopy methods based on these principles have revolutionised the way in which many important optimisation problems are solved in practice, including linear programming!

The Newton-Raphson method plays indeed a uniquely important role as a basic element in many practical optimisation algorithms. However, its usefulness has limitations in situations where computing the Hessian $D^2 f$ and solving the linear system $D^2 f(x_k)d_k = -\nabla f(x_k)$ to determine the Newton step is computationally too expensive. In some applications the problem dimension is so large that it is not possible to keep all the entries of $D^2 f$ in the main memory of a computer.