CS 435, 2018
Lecture 6, Date: 12 April 2018
Instructor: Nisheeth Vishnoi

Newton's Method

In this lecture we derive and analyze Newton's method which is an example of a second order optimization method. We argue that Newton's method can be seen as gradient descent on a Hessian manifold, which then motivates an affinely invariant analysis of its convergence.

Contents

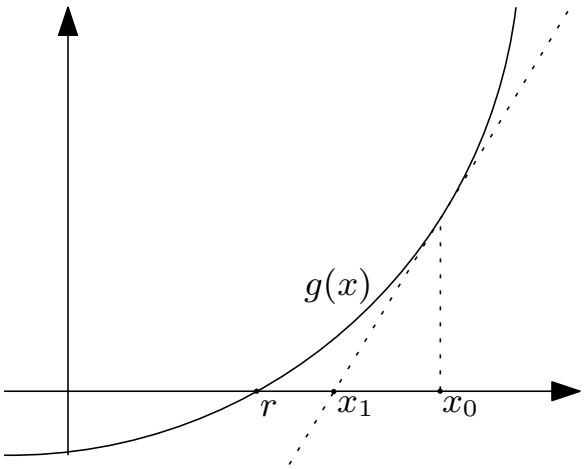

Figure 1: One step of Newton's Method

1 Newton-Raphson method

In numerical analysis, Newton's method (also known as the Newton-Raphson method [3]), named after Isaac Newton and Joseph Raphson, is a method for finding successively better approximations to the roots (or zeroes) of a real-valued function.

1.1 Derivation of the update rule

Suppose we are given a function $g : \mathbb{R} \to \mathbb{R}$ and we want to find its root (or one of its roots). Assume we are given a point x_0 which is likely to be close to a zero of g. Consider the graph of g as a subset of $\mathbb{R} \times \mathbb{R}$ and the point $(x_0, g(x_0))$. We can draw a line through through this point, which is tangent to the graph of g (of course assuming g is regular enough). Let x_1 be the intersection of the line with the x-axis (see Figure 1). Then it is likely to happen (at least if one were to believe the figure above) that by moving from x_0 to x_1 we have made progress in reaching a zero of g. By a simple calculation one can see that x_1 arises from x_0 as follows

$$x_1 = x_0 - \frac{g(x_0)}{g'(x_0)}$$

Thus the following iterative algorithm for computing a root of g naturally follows: start with an $x_0 \in \mathbb{R}$ and use the inductive definition below to compute $x_1, x_2, x_3, \ldots$

$$x_{k+1} := x_k - \frac{g(x_k)}{g'(x_k)} \qquad \text{for all } k \geq 0. \tag{1}$$

Of course we require differentiability of g, in fact we will assume even more – that g is twice continuously differentiable.

An Example

Suppose that for some $a > 0$ we would like to minimize the function

$$f(x) = ax - \log x$$

over all positive $x > 0$. To solve $\min_{x>0} f(x)$ one can first take the derivative $g(x) = f'(x)$ and try to find a root of g. Since f is convex, such a strategy is indeed correct. We have

$$g(x) := f'(x) = a - \frac{1}{x}.$$

It is not a secret that this equation can be solved exactly and the solution is $\frac{1}{a}$, however we would still like to apply Newton's method to it. One reason for that is that we would like to test it on a particular, simple example. Another reason is that it actually has interesting applications! Computing a reciprocal of a number is a rather nontrivial operation and some IBM computers in the past used Newton's method to compute it, because as we will see in a second the Newton's update involves only "simpler" arithmetic operations: addition/subtraction and multiplication.

We initialize the Newton's method at any point $x_0 > 0$, and iterate as follows

$$x_{k+1} = x_k - \frac{g(x_k)}{g'(x_k)} = 2x_k - ax_k^2.$$

Note that computing x_{k+1} from x_k indeed does not use divisions!

Let us now try to analyze the sequence $\{x_k\}_{k\in\mathbb{N}}$ and see when does it converge to $\frac{1}{a}$. By denoting $e_k := 1 - ax_k$ we obtain the following recurrence relation for e_k

$$e_{k+1} = e_k^2.$$

Thus it is now easy to see that whenever $|e_0| < 1$ then $e_k \to 0$. Further, if $|e_0| = 1$ then $e_k = 1$ for all $k \geq 1$ and if $|e_0| > 1$ then $e_k \to \infty$.

In terms of x_0 it means that whenever $0 < x_0 < \frac{2}{a}$ then $x_k \to \frac{1}{a}$. However, if we initialize at $x_0 = \frac{2}{a}$ or x_0, then the algorithm "gets stuck" at 0. And even worse: if we initialize at $x_0 > \frac{2}{a}$ then $x_k \to -\infty$!

This example shows that chosing the right starting point might have a crucial impact on whether Newton's method succeds or fails!

Another interesting aspect is that by modifying the function g for example by taking $g(x) = x - \frac{1}{a}$ or $g(x) = ax - \frac{1}{x^2}$, etc. one obtains different algorithms for compute $\frac{1}{a}$. Some of these algorithms might not make sense (for instance, the iteration $x_{k+1} = \frac{1}{a}$ is not how we would like to compute $\frac{1}{a}$), or might not be practical, yet still we can obtain them *automatically* as instances of the Newton's method.

1.2 Quadratic convergence

Let us now analyze formally how much progress towards a root of g is done in one iteration of Newton's method. When analyzing the example in the previous we encountered a phenomenon that the "distance" to the root was being squared at every iteration – this turns out to hold more generally.

Theorem 1 (Quadratic Convergence for Finding Roots)**.** *Suppose $g : \mathbb{R} \mapsto \mathbb{R}$ is a $\mathcal{C}^2$ function,* [1] *$r \in \mathbb{R}$ is a root of g, $x_0 \in \mathbb{R}$ is a starting point and $x_1 = x_0 - \frac{g(x_0)}{g'(x_0)}$, then*

$$|r - x_1| \leq M|r - x_0|^2$$

where $M = \sup_{\xi \in [r, x_0]} \left| \frac{g''(\xi)}{2g'(x_0)} \right|$.

Proof. We start by expanding g into Taylor series at the point x_0 (using the Mean Value Theorem)

$$g(r) = g(x_0) + (r - x_0)g'(x_0) + \frac{1}{2}(r - x_0)^2 g''(\xi)$$

for some ξ in the interval $[r, x_0]$. From the definition of x_1, we know that

$$g(x_0) = g'(x_0)(x_0 - x_1).$$

Recall also that $g(r) = 0$. Hence, we get:

$$0 = g'(x_0)(x_0 - x_1) + (r - x_0)g'(x_0) + \frac{1}{2}(r - x_0)^2 g''(\xi)$$

which implies that

$$g'(x_0)(r - x_1) = \frac{1}{2}(r - x_0)^2 g''(\xi).$$

This gives us the claimed bound on the distance from x_1 to r in terms of the distance from x_0 to r

$$|r - x_1| = \left| \frac{g''(\xi)}{2g'(x_0)} \right| |r - x_0|^2.$$

□

Assuming that M is a small constant, say $M \leq 1$ (and remains so throughout the execution of this method) and that $|x_0 - r| < 1/2$, we obtain *quadratically* fast convergence of x_k to r. Indeed, after k steps we have

$$|x_k - r| \leq |x_0 - r|^{2^k} \leq 2^{-2^k}$$

[1] The function is twice differentiable and the second derivative is continuous.

and hence, for the error $|x_k - r|$ to become smaller than ε it is enough to take

$$k \approx \log\log 1/\varepsilon.$$

As one can imagine, for this reason Newton's Method is very efficient and powerful. In addition: in practice it turns out to be very robust and converges rapidly even when no bounds on M or $|x_0 - r|$ are available.

1.3 An extension to multivariate functions

So far we have only considered the univariate version of Newton's method for finding roots of functions $g : \mathbb{R} \to \mathbb{R}$. However, as we demostrate below, it can be extended to solving systems of nonlinear equations in multiple equations. Suppose we would like to find a solution $x \in \mathbb{R}^n$ to the system[2]

$$\begin{cases} g_1(x) = 0, \\ g_2(x) = 0, \\ \dots \\ g_n(x) = 0. \end{cases} \tag{2}$$

where $g_1, g_2, \dots, g_n : \mathbb{R}^n \to \mathbb{R}$. In other words, we have a function $g : \mathbb{R}^n \to \mathbb{R}^n$ of the form $g(x) = (g_1(x), g_2(x), \dots, g_n(x))^\top$ and we would like to find $x \in \mathbb{R}^n$ such that $g(x) = 0$.

Let us now try to find an analogue of Newton's method in this setting by mimicking the update rule (1) for the univariate setting, i.e., $x_1 = x_0 - \frac{g(x_0)}{g'(x_0)}$. Previously, $g(x_0)$ and $g'(x_0)$ were numbers so this expression made sense, however, when going from $n = 1$ to arbitrary $n > 1$, $g(x_0)$ becomes a vector and $g'(x_0)$ (or really $J_g(x_0)$ – the Jacobian of g at x_0) becomes an $n \times n$ matrix. Hence, the "right" extension of (1) to the multivariate setting is

$$x_{k+1} := x_k - J_g(x_k)^{-1} g(x_k) \qquad \text{for all } k \geq 0, \tag{3}$$

where $J_g(x_0)$ is the Jacobian matrix of g at x_0, i.e., the unique matrix that satisfies

$$g(x) = g(x_0) + J_g(x_0)(x - x_0) + o\left(\|x - x_0\|_2\right),$$

where x_0 is fixed and x varies. $J_g(x_0)$ can be also seen as the matrix of partial derivatives $\left[\frac{\partial g_i}{\partial x_j}(x_0)\right]_{1 \leq i,j \leq n}$.

By adjusting the proof of Theorem 1 we can recover an analogous local quadratic convergence rate of Newton's method in the multivariate setting. We do not state a precise

[2]Note that here, for simplicty, we consider the setting where the number of variables is equal to the number of equations in the system. This roughly corresponds to the intuition that if the equations are "independent" then since a point $x \in \mathbb{R}^n$ has "n degrees of freedom", exactly n equations are needed to uniquely specify it. One can easily extend the Newton's method to systems having more equations than variables.

theorem for this variant, but as one might expect the corresponding value M involves the following two quantities: an upper bound on the "magnitude" of the second derivative of g (or in other words a bound on the Lipschitz constant of $x \mapsto J_g(x)$) and a lower bound on the "magnitude" of $J_g(x)$, of the form $1/\|J_g(x)^{-1}\|$, where $\|\cdot\|$ denotes here the spectral norm of a matrix. For more details, we refer to Section 3 where we present a closely related convergence result, which can be also adapted to this setting. For a more thorough discussion of Newton's method we also refer to a book by [1].

2 Newton's method for unconstrained optimization

2.1 From optimization to root finding

How could the benign looking Newton-Raphson method be useful to solve convex programs? The key lies in the observation from the first lecture that the task of minimizing a differentiable convex function in the unconstrained setting is equivalent to *finding a root of its derivative*. In this section we abstract out the method from the previous section and present Newton's method for unconstrained optimization.

Recall that in the unconstrained case, the problem is to find

$$x^\star := \operatorname*{argmin}_{x \in \mathbb{R}^n} f(x)$$

where f is a convex (smooth enough[3]) function. The gradient ∇f of f can be thought as a function $\mathbb{R}^n \mapsto \mathbb{R}^n$ and its Jacobian is the Hessian $\nabla^2 f$ (indeed, we have $\frac{\partial}{\partial x_j}\left[\nabla F(x)\right]_i = \left[\nabla^2 f(x)\right]_{i,j}$). Hence, the update rule (3) for finding a root of a multivariate function g (i.e., $x_{k+1} = x_k - \frac{g(x_k)}{g'(x_k)}$) adjusted to this setting is

$$x_{k+1} := x_k - (\nabla^2 f(x_k))^{-1} \nabla f(x_k) \qquad \text{for all } k \geq 0. \tag{4}$$

For notational convenience we define *the Newton step* at point x to be

$$n(x) := -(\nabla^2 f(x))^{-1} \nabla f(x).$$

Then (4) can be compactly written as

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

2.2 Newton's method as a second order method

Let us now give another interpretation of Newton's method which follows from a very natural optimization perspective. Suppose we would like to find a global minimum of f

[3]Throughout this lecture we will assume f is smooth enough, i.e., thrice-differentiable without stating it explicitly.

and x_0 is our current approximate solution. Let $\widetilde{f}(x)$ denote the second order approximation of $f(x)$ around x_0, i.e.,

$$\widetilde{f}(x) = f(x_0) + \langle x - x_0, \nabla f(x_0) \rangle + \frac{1}{2}(x - x_0)^\top \nabla^2 f(x_0)(x - x_0).$$

A natural idea to compute the new approximation x_1 is then to minimize $\widetilde{f}(x)$ over $x \in \mathbb{R}^n$! Since we hope that $\tilde{f} \approx f$, at least locally, this new point should be an even better approximation to $x^\star$ (the minimizer of f).

To find x_1 we need to solve

$$x_1 := \operatorname*{argmin}_{x \in \mathbb{R}^n} \widetilde{f}(x),$$

assuming that f is convex (or rather that the Hessian of f at x_0 is PSD), this is equivalent to $\nabla \widetilde{f}(x) = 0$, i.e.,

$$\nabla f(x_0) + \nabla^2 f(x_0)(x - x_0) = 0,$$

which translates to (assuming $\nabla^2 f(x_0)$ is invertible)

$$x - x_0 = -\left(\nabla^2 f(x_0)\right)^{-1} \nabla f(x_0).$$

Hence

$$x_1 = x_0 - \left(\nabla^2 f(x_0)\right)^{-1} \nabla f(x_0),$$

and we recover Newton's method. This says that Newton's method is, at every step, minimizing the second order approximation around the current point and takes the minimizer as the next point! For this reason Newton's method is called a second-order method, because it takes advantage of the second order approximation of a function to make a step towards the minimum.

This has the following, rather surprising consequence: whenever we apply Newton's method to a strictly convex quadratic function (i.e. of the form $h(x) = \frac{1}{2}x^\top Mx + b^\top x$ for $M \in \mathbb{R}^{n \times n}$ positive definite and $b \in \mathbb{R}^n$). Then, no matter which point we start at, after one iteration we land in the unique minimizer. It is instructive to compare it to the algorithms we looked at in previous lectures. They were all first-order methods and required multiple iterations to reach a point close to the optimizer. Does it mean that Newton's method is a "better" algorithm? On the one hand, the fact that Newton's method also uses the Hessian to perform the iteration makes it clearly more powerful than first-order methods. On the other hand though, this power comes at a cost: computationally one iteration is now more costly, as we need a second order oracle to the function. More precisely, at every step k, to compute x_{k+1} we need to solve a system of n linear equations in n variables, of the form

$$\left(\nabla^2 f(x_k)\right) x = \nabla f(x_k).$$

In the worst case, this takes $O(n^3)$ time using Gaussian elimination (or $O(n^\omega)$ using fast matrix multiplication), unless the Hessian matrix has a special form and faster methods are available (based on Conjugate gradient, Laplacian solvers or more generally for SDD matrices [5]).

2.3 Example: logarithmic barrier and analytic center of a polytope

To illustrate the Newton's method for optimization, consider a polytope

$$P := \{x \in \mathbb{R}^n : \langle a_i, x\rangle \leq b_i, \text{ for } i = 1, 2, \ldots, m\}.$$

We assume that P is a bounded and full-dimensional subset of $\mathbb{R}^n$. Consider the following convex function F over the interior of P called the *logarithmic barrier*

$$F(x) = -\sum_{i=1}^{m} \log(b_i - a_i^\top x).$$

One can think of every its term $-\log(b_i - a_i^\top x)$ as putting a force at the constraint $\langle a_i, x\rangle \leq b_i$, which is the larger, the closer the point x comes to the hyperplane $\{y : \langle a_i, y\rangle = b_i\}$, and is $+\infty$ if the point x lies on the "wrong" side of this hyperplane.

The task of computing the global minimum of F can be seen as finding an "equilibrium" point for all these forces. To see this, let us inspect the gradient ∇F

$$\nabla F(x) = \sum_{i=1}^{m} \frac{a_i}{s_i(x)},$$

where, for brevity we denote the "slack" with respect to the ith constraint by $s_i(x) := b_i - \langle a_i, x\rangle$. Thus the point x with $\nabla F(x) = 0$ has the physical interpretation as a location where all "forces" are in balance. Every term in this sum is a vector a_i (the direction of the force along which the ith force acts) scaled by the magnitude $\frac{1}{s_i(x)}$ of this force, which gets larger as x gets closer to the $\{y : \langle a_i, y\rangle = b_i\}$ hyperplane. For this reason, the unique point $x^\star$ such that $\nabla F(x^\star) = 0$ is sometimes referred to as the *analytic center* of P. Finding $x^\star$ turns out to be a useful primitive – intuitively, this is a point which is relatively far away from all the faces of P, thus for instance it might be used to assess how large the volume of the polytope is. In fact, in the next lecture, we will use this primitive to design a polynomial time algorithm for linear programming.

To apply Newton's method to find $x^\star$ we also need to compute the Hessian of F, this turns out to be

$$\nabla^2 F(x) = \sum_{i=1}^{m} \frac{a_i a_i^\top}{s_i(x)^2}.$$

Note that both $\nabla F(x)$ and $\nabla^2 F(x)$ can be computed efficiently given $a_1, a_2, \ldots, a_m \in \mathbb{R}^n$ and $b_1, b_2, \ldots, b_m \in \mathbb{R}$. In the worst case the gradient $\nabla F(x)$ can be computed in $O(nm)$ time and it takes $O(n^2 m)$ to compute $\nabla^2 F(x)$. Note however, that these are worst case bounds and if for instance a_i's are sparse vectors (have only a constant number of non-zero entries), then the time to compute both $\nabla F(x)$ and $\nabla^2 F(x)$ reduces to $O(m)$! To perform one iteration of Newton's method one is also required to solve a system of n linear equations in n variables, of the form

$$\left(\nabla^2 F(x_0)\right) x = \nabla F(x_0).$$

In the worst case, this takes $O(n^\omega)$ time, but can be sometimes solved significantly faster if the vectors $a_1, a_2, \ldots, a_m$ have additional structure. In fact, if these vectors come from an incidence matrix of a graph, then $\nabla^2 F(x)$ is a graph Laplacian, and a linear system of this form can be solved in $\widetilde{O}(m)$ time, by a result of Spielman and Teng [5].

So far we have discussed the applicability of Newton's method to the problem of minimizing F from the computational perspective: a single iteration can be performed efficiently. However, it is not clear how many iterations are required to reach a close-to-optimal point, if at all this method converges! In the next section we present an analysis of Newton's method which shows that locally it converges rapidly whenever the polytope P is "round enough". In the subsequent section, we provide an improved analysis which allows us to omit the "roundness" assumption. Finally, in the next lecture, using an additional idea, we will be able to go from local convergence to global convergence and as a consequence, derive a polynomial time algorithm for Linear Programming, whose one of the main components is the logarithmic barrier function F (see [4]).

3 First take on the analysis of Newton's method

3.1 Analysis based on the Euclidean norm

Below we present the first take on the analysis of Newton's method. The theorem we state is analogous to Theorem 1 (for univariate root finding) – it says that one step of the Newton's method yields quadratic improvement of the distance to the optimal solution whenever a condition (which we call **NE** for Newton-Euclidean) is satisfied. We start by stating the theorem and proceed to defining and discussing the **NE** condition afterwards.

Theorem 2 (Quadratic Convergence w.r.t. Euclidean Norm)**.** *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a function and $x^\star$ be its minimizer. Let x_0 be an arbitrary starting point and define $x_1 := x_0 + n(x_0)$. If the condition $\mathbf{NE}(M)$ is satisfied, then*

$$\|x_1 - x^\star\|_2 \leq M \|x_0 - x^\star\|_2^2 .$$

It remains to define the condition **NE** formally. Below, we denote by $\|A\|$ (for a matrix $A \in \mathbb{R}^{n\times n}$) the $2 \to 2$ operator norm of A, i.e., $\|A\| := \sup_{x\in\mathbb{R}^n} \frac{\|Ax\|_2}{\|x\|_2}$.

Definition 3 (Condition **NE**)**.** *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a function, $x^\star$ be its minimizer and x_0 be an arbitrary point. Denote by $H(x)$ the Hessian of f at the point $x \in \mathbb{R}^n$. We say that the condtion $\mathbf{NE}(M)$ is satisfied for some $M > 0$ if there exists a Euclidean ball $B(x^\star, R)$ of radius r around $x^\star$, containing x_0, and two constants $h, L > 0$ such that $M \geq \frac{L}{2h}$ and*

- *for every $x \in B(x^\star, R)$, $\left\|H(x)^{-1}\right\| \leq \frac{1}{h}$,*
- *for every $x, y \in B(x^\star, R)$, $\|H(x) - H(y)\| \leq L \|x - y\|_2$.*

Given the above condition, one can present a rough analogy between Theorem 2 and Theorem 1. There, for the method to have quadratic convergence, $|g'(x)|$ should be large

(relatively to $|g''(x)|$). Here, the role of g is played by the gradient ∇f of f. The first condition on $H(x)$ says basically that the "magnitude" of the second derivative of f is "big". The second condition may be a bit more tricky to decipher, it says that $\nabla^2 f(x)$ is Lipschitz-continuous, and upper-bounds the Lipschitz constant. Assuming f is thrice continuously differentiable, this essentially gives an upper bound on the magnitude of $D^{(3)} f$. This intuitive explanation is not quite formal, however we only wanted to emphasize that the spirit of Theorem 2 still remains the same as Theorem 1.

The proof of Theorem 2 is similar to the proof of Theorem 1 and thus is moved to the Appendix A.

3.2 The problem with the convergence in Euclidean norm

The statement and the proof of Theorem 2 are rather natural extensions of Theorem 1, however, as it turns out, the fact that it is stated with respect to quantities based on the Euclidean norm $\|\cdot\|_2$ makes it hard or even impossible to apply in many important cases. We will see in the next section that there is a much more natural choice of a norm to work with in the context of Newton's method – *the local norm*. However, before we introduce the local norm, let us first see that there is indeed a problem with the use of the Euclidean norm by studying a particular example where Theorem 2 fails to give reasonable bounds.

Consider the logarithmic barrier F for the polytope $P = [-K_1, K_1] \times [-1/K_2, 1/K_2] \subseteq \mathbb{R}^2$ defined in Section 2.3, where both $K_1, K_2 > 0$ should be thought of as a large constants. This means that P is very "wide" (w.r.t. the x_1 axis) and very short (w.r.t. the x_2 axis). The logarithmic barrier takes the following form

$$F(x_1, x_2) = -\log(K_1 - x_1) - \log(K_1 + x_1) - \log\left(\frac{1}{K_2} - x_2\right) - \log\left(\frac{1}{K_2} + x_2\right),$$

and its Hessian is

$$H(x_1, x_2) = \begin{pmatrix} \frac{1}{(K_1 - x_1)^2} + \frac{1}{(K_1 + x_1)^2} & 0 \\ 0 & \frac{1}{\left(\frac{1}{K_2} - x_2\right)^2} + \frac{1}{\left(\frac{1}{K_2} + x_2\right)^2} \end{pmatrix}.$$

We would like to find estimates on the parameters h and L (and in consequence on M) for the condition $\mathbf{NE}(M)$ to hold in a close neighborhood of the optimal point $(x_1^\star, x_2^\star) = (0, 0)$. As we will show, the M parameter is always prohibitively large, so that Theorem 2 cannot be applied to reason about convergence of Newton's method in this case. However, nevertheless, as we argue in a later section, Newton's method works very well when applied to the logarithmic barrier function, no matter what P is!

Let us start by observing that the first condition in $\mathbf{NE}(M)$ asks for an $h > 0$ such that $hI \preceq H(x_1, x_2)$ in the neighborhood of $x^\star$. Even at $x^\star$ we already have $h \leq \frac{2}{K_1^2}$, i.e., we can make it arbitrarily small by just changing K_1. The second condition asks to establish a bound on the Lipschitz constant of the Hessian of $H(x)$. To see that L is quite large as

well, consider the point $\widetilde{x} = (0, 1/K_2^2)$. Note that

$$\|H(\widetilde{x}) - H(x^\star)\| = \left\| \begin{pmatrix} 0 & 0 \\ 0 & \Theta(1) \end{pmatrix} \right\| = \Theta(1).$$

This estalishes a lower bound of

$$\frac{\|H(\widetilde{x}) - H(x^\star)\|}{\|\widetilde{x} - x^\star\|} \approx K_2^2$$

on the Lipschitz constant L.

Thus the constant $M = \frac{L}{2h}$ which determines the quadratic convergence of Newton's method is at least $\Omega(K_1^2 K_2^2)$ and thus in particular even when we initialize it at $\widetilde{x}$, which is relatively close to the analytic center $x^\star$, the guarantee in Theorem 2 is too weak to imply that in one step the distance to the analytic center will drop. In fact, we get that if we set $x_0 = \widetilde{x}$, then the next point x_1 satisfies

$$\|x_1\| \leq M \|x_0\|^2 \leq M \frac{1}{K_2^4} = \Omega\left(\frac{K_1^2}{K_2^2}\right),$$

thus whenever K_1 is at least a constant, Theorem 2 does not even imply that the distance drops.

As we show later, this issue disappears when we state the convergence guarantee of Newton's in terms of the "right" norm! In fact Newton's method converges rapidly to $x^\star$ when initialized at $\widetilde{x}$ (this can be also checked directly – by hand, for this simple 2-dimensional example).

3.3 Affine invariance of Newton's method

One of the important features of Newton's method is its *affine invariance*: if we consider an affine change of coordinates $y := \phi(x) = Ax + b$ (where $A \in \mathbb{R}^{n \times n}$ is an invertible matrix and $b \in \mathbb{R}^n$) then the Newton's method will proceed over the same sequence of points in the $x-$ and y-coordinates.

Formally, consider the function $f : \mathbb{R}^n \to \mathbb{R}$ written in the y-coordinates and $\tilde{f} : \mathbb{R}^n \to \mathbb{R}$ written in x-coordinates, i.e., $\tilde{f}(x) = f(Ax + b)$. Then, if $x_0 \in \mathbb{R}^n$ moves to $x_1 \in \mathbb{R}^n$ by applying one step of Newton's method with respect to $\tilde{f}$, then $y_0 = \phi(x_0)$ moves to $y_1 = \phi(x_1)$ by applying one step of Newton's method with respect to f. This property does not hold for gradient descent or mirror descent, or any of the first-order methods which we have studied before. For this reason it is sometimes possible to improve the convergence rate of gradient descent by a certain *preconditioning* (change of coordinates).

Note that this gives another reason why the bound obtained in Theorem 2 is not quite satisfactory. Indeed, it depends on quantities which are not affinely invariant, and thus changing the coordinates in which the function f is written, even though the Newton's

method still takes the same trajectory, causes the parameters L and h in condition $\mathbf{NE}(M)$ and thus also the final bound in Theorem 2 to change!

To overcome these issues, in the next section we present a different interpretation of Newton's method – in terms of the gradient flow on a Riemannian manifold – and finally in the last section we obtain an analysis of Newton's method based only on affinely invariant quantities.

4 Newton's method as gradient descent on a Riemannian manifold

4.1 Motivational example

Suppose we would like to minimize the quadratic function $f : \mathbb{R}^2 \to \mathbb{R}$ given by

$$f(x_1, x_2) = x_1^2 + Kx_2^2,$$

where $K > 0$ is a large constant. It is clear that the minimum is $x^\star = 0$, yet still we would like to see how we can find it using iterative methods. The gradient descent method takes a starting point, say $\widetilde{x} = (1, 1)$ and goes to

$$x' = \widetilde{x} - \eta \nabla f(\widetilde{x})$$

for some step size $\eta > 0$. In this case

$$\nabla f(\widetilde{x}) = (2, 2K)^\top.$$

Note now that if we take η of constant size, then the new point x' will be of distance roughly $\Omega(K)$ from the optimum, hence instead of getting closer to $x^\star$, we get far away from it. The reason is that our step size η is too large. Indeed, to make the distance to $x^\star$ drop, when going from $\widetilde{x}$ to x' we are forced to take a step size of roughly $\eta \approx 1/K$. This, clearly makes the convergence process slow and inefficient.

How can we fix this issue of the gradient not pointing in the "right direction", i.e., not towards the optimum? Recall that in the gradient descent algorithm we chose to follow the negative gradient direction because it gave the largest drop among all directions of Euclidean length at most one. However, since the role of coordinates x_1 and x_2 is not at all symmetric in f, we should use a different norm (rather than Euclidean) to measure lengths of such vectors. To counter this effect let us consider the norm

$$\|(u_1, u_2)\|_\circ := \sqrt{u_1^2 + Ku_2^2},$$

and see what does the principle of maximum drop gives us with respect to this norm.

Recall (from our discussion of gradient descent) that the optimal direction at x with respect to the new norm is the solution to the following optimization problem

$$\max_{\|u\|_\circ \leq 1} \left[\lim_{\delta \to 0^+} \frac{f(x) - f(x + \delta u)}{\delta} \right].$$

We have

$$\lim_{\delta \to 0^+} \frac{f(x) - f(x + \delta u)}{\delta} = -2(x_1 u_1 + K x_2 u_2),$$

hence we would like to solve

$$\max_{u_1^2 + K u_2^2 \leq 1} -2(x_1 u_1 + K x_2 u_2).$$

By the Cauchy-Schwarz inequality we obtain that the optimal solution is

$$u_{opt} = -\frac{x}{\|x\|_\circ},$$

thus in fact, up to rescaling, this strategy suggests to go to $x^\star = 0$ along a straight line. The update is

$$x' = \widetilde{x} - \eta \widetilde{x}.$$

This new search direction is actually perfect, as it points directly towards the minimum of f and thus we are not forced to take small steps anymore. Indeed, for $\eta = 1$ we recover the minimum exactly.

This demostrates that changing the norm which we use to measure the lengths of possible "directions to follows" from a point x might be helpful in determining the optimal search direction.

Note also that this observation extends immediately to convex quadratic functions of the form $h(x) = x^\top A x + b^\top x$ (where A is positive definite). Instead of using the gradient direction $\nabla h(x) = 2Ax + b$, it is better to determine a new search direction by solving

$$\min_{\|u\|_A \leq 1} \left[\lim_{\delta \to 0^+} \frac{h(x) - h(x + \delta u)}{\delta} \right] = \min_{u^\top A u \leq 1} \langle \nabla h(x), u \rangle ,$$

here the rationale behind using the A-norm is again (as in the 2-dimensional example) to counter the effect of "stretching" caused by the quadratic term $x^\top A x$ in the objective. This turns out to be indeed the right norm to work with, as the optimal vector u (up to scaling) is equal to $u_{opt} = A^{-1} \nabla h(x) = 2x - A^{-1} b = 2(x - x^\star)$, hence again, pointing perfectly towards the optimum.

4.2 Riemannian manifolds

When discussing the example above we have in fact constructed (a very simple) instance of a Riemannian manifold. A manifold is a topological space Ω: think of it as $\mathbb{R}^n$ or a "nice" subet of $\mathbb{R}^n$ such as the sphere or a torus, such that the neighborhood around every point x in Ω is just (a slightly deformed) Euclidean space. Note that this is clearly the case for the circle

$$S^1 = \{x \in \mathbb{R}^2 : \|x\|_2 = 1\},$$

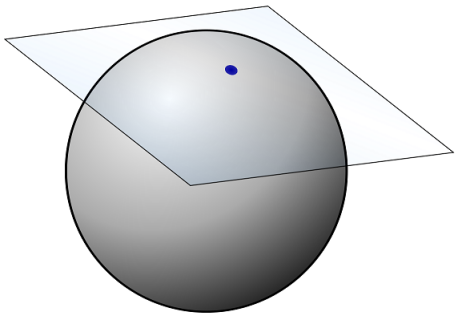

Figure 2: The tangent space of a point on the 2-dimensional sphere S^2.

which around every point looks just like a one-dimensional interval, even though globally it is not same as $\mathbb{R}^1$. At every point $x \in \Omega$ there is a notion of a "tangent space" at x, denoted typically by $T_x\Omega$ which is a linear space of all directions along which we can move "tangent" to Ω, see Figure 2 for an illustration of this notion. If we are in $\mathbb{R}^n$, the tangent space at every point is again $\mathbb{R}^n$, however in general it might be a non-trivial object. For the example of a circle S^1 the tangent space is a 1-dimensional subspace of $\mathbb{R}^2$ of the form $T_xS^1 = \{u \in \mathbb{R}^2 : \langle u, x\rangle = 0\}$.

In a Riemannian manifold, to every point $x \in \Omega$ there is an inner product $\langle \cdot, \cdot \rangle_x$ (or equivalently a norm $\|u\|_x := \sqrt{\langle u, u\rangle_x}$) assigned which acts on the tangent space $T_x\Omega$ and changes smoothly with x. This is useful in capturing the fact that even though Ω lives in $\mathbb{R}^n$ or is just $\mathbb{R}^n$, in reality it might be an embedding of a complicated subset of a much higher-dimensional Euclidean space $\mathbb{R}^m$. To capture the non-trivial geometry of Ω in a higher dimension, we might want to measure inner products on the tangent spaces in a non-standard way so that it matches what happens to Ω in $\mathbb{R}^m$ Intuitivelly, this is to make the embedding "isometric".

In the previous section the local norm $\|\cdot\|_x$ we used to derive better search directions for a convex quadratic was $\|\cdot\|_A$ and was the same for every point $x \in \mathbb{R}^n$. What we will see next is that in general, we might need to vary this norm with x.

4.3 Gradient descent on Riemannian manifold

From now on we will assume that $f : \mathbb{R}^n \to \mathbb{R}$ is a strictly convex function, i.e., the Hessian $\nabla^2 f(x)$ is positive definite at every point $x \in \mathbb{R}^n$. Moreover, for brevity, we denote the Hessian $\nabla^2 f(x)$ by $H(x)$.

Such a strictly convex function f naturally induces a Riemannian metric on $\mathbb{R}^n$. Indeed, at every point $x \in \mathbb{R}^n$ an inner product $\langle \cdot, \cdot \rangle_x$ can be defined as

$$\forall u, v \in \mathbb{R}^n \quad \langle u, v\rangle_x := u^\top H(x) v,$$

and a corresponding norm

$$\forall u \in \mathbb{R}^n \quad \|u\|_x := \sqrt{u^\top H(x) u}.$$

The above are called the *local inner product* and *local norm* respectively. They can be used to measure angles or distances between vectors u, v in the "tangent space" at x, which here happens to be, simply $\mathbb{R}^n$. Nevertheless, whenever a local norm of a vector u at x is considered, one should think of u being a "direction" to move from x. Whenever we refer to the "local norm" in the future, the underlying function f will be clear from the context.

Such a Riemannian metric is really a new "geometry" on the space $\mathbb{R}^n$. Therefore, it might be interesting to revisit the derivation of the gradient descent algorithm, which relied on the use of Euclidean norm $\|\cdot\|_2$, and see what this new geometry yields.

When deriving the gradient descent algorithm, we decided to greedily pick the "best direction" to go, or in other words, the direction which provides the largest drop in the objective value. Formally, this led us to the following optimization problem

$$\max_{\|u\| \leq 1} \left[\lim_{\delta \to 0^+} \frac{f(x) - f(x + \delta u)}{\delta} \right] = \max_{\|u\| \leq 1} \left[- \langle \nabla f(x), u \rangle \right]. \tag{5}$$

By taking the optimal direction u_{opt} with respect to the Euclidean norm, i.e., $\|\cdot\| = \|\cdot\|_2$ we obtained $u_{opt} \propto -\nabla f(x)$ and arrived at the *gradient flow* $\frac{dx}{dt} = -\nabla f(x)$.

What if we now instead maximize over all u of local norm at most 1? Then (5) becomes

$$\max_{\|u\|_x \leq 1} \left[- \langle \nabla f(x), u \rangle \right] = \max_{u^\top H(x) u \leq 1} \left[- \langle \nabla f(x), u \rangle \right]. \tag{6}$$

The rationale behind the above is clear given our discussion on the quadratic case – we would like to capture the "shape" of the function f around a point x with our choice of the norm, and now, our best guess for that is the "quadratic term" of f around x which is given by the Hessian! Again, by a simple derivation using the Cauchy-Schwarz inequality we obtain

$$u_{opt} \propto -H(x)^{-1} \nabla f(x)$$

which leads straight to the so called *Newton's flow*

$$\frac{dx}{dt} = - \left(\nabla^2 f(x) \right)^{-1} \nabla f(x),$$

and gives an alternative way of deriving Newton's method – as a discretization of Newton's flow, or in other words – as a gradient descent on a certain Riemannian manifold! The type of Riemannian structures we deal with here is very special as the inner product at every point is given by a Hessian of a convex function – these are typically referred to as Hessian manifolds.

In the next section we will use local norms to design a new, affinely invariant analysis of the Newton's method for optimization which gets around issues arising when restricting ourselves to the use of the Euclidean norm, as in Theorem 2.

5 Analysis in terms of the local norm

The bound provided in Theorem 2 was stated with respect to the Euclidean distance from the current iterate x_k to the optimal solution $x^\star$. As already discussed, such a quantity is not affinely invariant and hence we would ideally like to replace it by a different notion of progress.

5.1 A new potential function

Since we deal with convex functions, a natural quantity which can tell us how close or how far are we from the optimum is the norm of the gradient $\|\nabla f(x)\|$. However, as observed in the previous section, in Newton's method we really work over the Hessian manifold induced by $H(x)$ hence the notion of a gradient of f is now $H(x)^{-1}\nabla f(x)$ and we should use the local norm instead of the generic Euclidean norm! Consequently, we arrive at the following quantity to track:

$$\|n(x)\|_x = \left\|H(x)^{-1}\nabla f(x)\right\|_x = \sqrt{(\nabla f(x))^\top H(x)^{-1}\nabla f(x)}.$$

It is now easily verified that $\|n(x)\|_x$ is indeed affinely invariant. We note that $\|n(x)\|_x$ has also a different interpretation: $\frac{1}{2}\|n(x)\|_x^2$ is the gap between the current value of f and the minimum value of the second order quadratic approximation of f at x. To see this, let us take an arbitrary point x_0 and let $x_1 = x_0 + n(x_0)$. We consider the quadratic approximation $\widetilde{f}$ of f at x_0

$$\widetilde{f}(x) = f(x_0) + \langle \nabla f(x_0), x - x_0\rangle + \frac{1}{2}(x - x_0)^\top \nabla^2 f(x)(x - x_0).$$

From our discussion in the previous section we know that x_1 minimizes $\widetilde{f}$, hence indeed

$$\begin{aligned}\widetilde{f}(x_0) - \widetilde{f}(x_1) &= -\langle \nabla f(x_0), n(x_0)\rangle - \frac{1}{2}n(x_0)^\top \nabla^2 f(x_0) n(x_0)\\ &= \frac{1}{2}\|n(x_0)\|_{x_0}^2.\end{aligned}$$

At this point let us now revisit the "bad example" studied in Section 3.2. Let us first see that our "potential" $\|n(\widetilde{x})\|_{\widetilde{x}}$ is "small" at the point $\widetilde{x} = (0, 1/K_2^2)$. Indeed, by thinking of K_2 as large and suppressing lower order terms we obtain

$$n(\widetilde{x}) = -H(\widetilde{x})^{-1}\nabla F(\widetilde{x}) \approx -\begin{pmatrix}0 & 0\\ 0 & K_2^{-2}\end{pmatrix}\cdot\begin{pmatrix}0\\ 2\end{pmatrix} = -\begin{pmatrix}0\\ K_2^{-2}.\end{pmatrix}$$

Finally

$$\|n(\widetilde{x})\|_{\widetilde{x}} \approx \frac{1}{K_2}.$$

If fact, more generally, $\|n(x)\|_x$ corresponds to the (Euclidean) distance from x to 0 as if the polytope P was scaled (along with point x) to become the square $[-1,1] \times [-1,1]$. As our next theorem says, the error measured as $\|n(x)\|_x$ decays quadratically fast in the Newton's method, hence we obtain convergence when initialized at $\widetilde{x}$.

5.2 Statement of the bound in the local norm

Note that the theorem we present below bears significant resemblance to Theorem 2, however crucially instead ot measuring the Euclidean distance to the optimal solution, we state the improvement with respect to the potential $\|n(x)\|_x$. This in turn allows us to use a much more convenient and natural condition **NL** instead of **NE** as in Theorem 2.

Theorem 4 (Quadratic Convergence w.r.t Local Norm)**.** *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a strictly convex function satisfying* **NL**, *$x_0 \in \mathbb{R}^n$ be any point and $x_1 := x_0 + n(x_0)$. If $\|n(x_0)\|_{x_0} < \frac{1}{6}$ then*

$$\|n(x_1)\|_{x_1} \leq 3 \|n(x_0)\|_{x_0}^2 .$$

To complete the picture we still need to define the condition **NL** (which stands for Newton-Local) mentioned in the above Theorem[4].

Definition 5 (Condition **NL**)**.** *Let f be a function; we say that f satisfies the* **NL** *condition if*

$$\forall_{x,y} \|y - x\|_x = \delta < 1 \qquad (1 - 3\delta)H(x) \preceq H(y) \preceq (1 + 3\delta)H(x),$$

where $H(x) := \nabla^2 f(x)$ and $\|\cdot\|_x$ denotes the local norm.

First, it is worth noting that the **NL** condition is indeed affinely invariant. This means that $x \mapsto f(x)$ satisfies this condition if and only if $x \mapsto f(\phi(x))$ satisfies it, for any affine change of variables ϕ.

Let us now inspect the condition **NL** in more detail. To this end, assume for simplicity that $f(x)$ is a univariate function. Then **NL** says, roughly that

$$|H(x) - H(y)| \leq 3 \|x - y\|_x |H(x)|$$

whenever $\|x - y\|_x$ is small enough. In other words

$$\frac{|f''(x) - f''(y)|}{\|x - y\|_x} \cdot \frac{1}{|f''(x)|} \leq 3.$$

Note that the first term, intuitively, corresponds to bounding the third derivative of f in the "local norm". Thus really the above says something very similar to "$M \leq 3$" where M is the quantity from Theorem 1 (recall that $g(x)$ corresponds to $f'(x)$ here). The difference is though that the quantities we consider here are computed with respect to the local norm, as opposed to the Euclidean norm, as in Theorem 1 or Theorem 2. The constant "3" is by no means crucial to the definition of **NL**, it is only chosen for the convenience of our future calculations.

[4] A closely related condition called *selfconcordance* was introduced and studied in [2].

5.3 Proof of convergence in the local norm

Before proving Theorem 4 we need to establish one simple yet important lemma first. It says that if condition **NL** is satisfied then the local norms of close-by points are also "similar", or more formally they have low "distortion" with respect to each other! If we go from x to y and their local distance is a (small) constant, then the local norms at x and y differ just by a factor of 2.

Lemma 6 (Low distortion of close-by norms)**.** *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a strictly convex function satisfying condition* **NL**. *Whenever $x, y \in \mathbb{R}^n$ are such that $\|y - x\|_x \leq \frac{1}{6}$ then for every $u \in \mathbb{R}^n$ we have*

- $\frac{1}{2}\|u\|_x \leq \|u\|_y \leq 2\|u\|_x$, *and*
- $\frac{1}{2}\|u\|_{H(x)^{-1}} \leq \|u\|_{H(y)^{-1}} \leq 2\|u\|_{H(x)^{-1}}$.

Recall that by $\|u\|_A$ for $u \in \mathbb{R}^n$ and a positive definite matrix $A \in \mathbb{R}^{n\times n}$ we denote the A-norm of the vector u, i.e, $\|u\|_A := \sqrt{u^\top A u}$. We remark that in the above, $\|\cdot\|_{H(x)^{-1}}$ is really dual to the local norm $\|\cdot\|_x = \|\cdot\|_{H(x)}$, i.e., $\|\cdot\|_{H(x)^{-1}} = \|\cdot\|_x^\star$.

Proof of Lemma 6: From condition **NL** we have

$$\frac{1}{2}H(y) \preceq H(x) \preceq 2H(y),$$

and hence also

$$\frac{1}{2}H(y)^{-1} \preceq H(x)^{-1} \preceq 2H(y)^{-1}.$$

The lemma follows now by the definition of the PSD ordering. □

Before we proceed to the proof of Theorem 4 let us state a simple fact on the relation between the PSD ordering and spectral norms of symmetric matrices which will be helpful to us in the proof.

Fact 1. *Suppose $A \in \mathbb{R}^{n\times n}$ is a symmetric positive definite matrix and $B \in \mathbb{R}^{n\times n}$ is symmetric such that $-\alpha A \preceq B \preceq \alpha A$ for some $\alpha \geq 0$, then*

$$\left\|A^{-1/2}BA^{-1/2}\right\| \leq \alpha.$$

For completeness we provide a proof of the fact in Appendix B. We are now well equipped to prove Theorem 4.

Proof of Theorem 4: Recall that our goal is to prove that $\|n(x_1)\|_{x_1} \leq 3\|n(x_0)\|_{x_0}^2$. At this point note that $\|n(x_1)\|_{x_1}$ can be also written as $\|\nabla f(x_1)\|_{H(x_1)^{-1}}$ and similarly:

$\|n(x_0)\|_{x_0} = \|\nabla f(x_0)\|_{H(x_0)^{-1}}$. Therefore, using the fact that the local norms at x_1 and x_0 are same up to a factor of 2 (see Lemma 6) it is enough to prove that

$$\|\nabla f(x_1)\|_{H(x_0)^{-1}} \leq \frac{3}{2} \|\nabla f(x_0)\|^2_{H(x_0)^{-1}}. \tag{7}$$

Towards this goal we first write the gradient $\nabla f(x_1)$ in the form $A(x_0)\nabla f(x_0)$ where $A(x_0)$ is a certain explicit matrix. Later we will show that the norm of $A(x_0)$ is (in a certain sense) small, which in turn will allow us to establish the goal (7). We start by writing $\nabla f(x_1)$ in a convenient form.

$$\begin{aligned}
\nabla f(x_1) &= \nabla f(x_0) + \int_0^1 H(x_0 + t(x_1 - x_0))(x_1 - x_0)dt \\
&\quad \text{(by the Fundamental Theorem of Calculus applied to } \nabla f : \mathbb{R}^n \to \mathbb{R}^n) \\
&= \nabla f(x_0) - \int_0^1 H(x_0 + t(x_1 - x_0))H(x_0)^{-1}\nabla f(x_0)dt \\
&\quad \text{(by rewriting } (x_1 - x_0) \text{ as } -H(x_0)^{-1}\nabla f(x_0)) \\
&= \nabla f(x_0) - \left[\int_0^1 H(x_0 + t(x_1 - x_0))dt\right] H(x_0)^{-1}\nabla f(x_0) \\
&\quad \text{(by linearity of } \textstyle\int) \\
&= \left[H(x_0) - \int_0^1 H(x_0 + t(x_1 - x_0))dt\right] H(x_0)^{-1}\nabla f(x_0) \\
&\quad \text{(by writing } \nabla f(x_0) \text{ as } H(x_0)H(x_0)^{-1}\nabla f(x_0)) \\
&= M(x_0)H(x_0)^{-1}\nabla f(x_0).
\end{aligned}$$

Where in the last equation we denoted

$$M(x_0) := H(x_0) - \int_0^1 H(x_0 + t(x_1 - x_0))dt.$$

By taking $\|\cdot\|_{H(x_0)^{-1}}$ on both sides of the above derived equality we obtain

$$\begin{aligned}
\|\nabla f(x_1)\|_{H(x_0)^{-1}} &= \left\|M(x_0)H(x_0)^{-1}\nabla f(x_0)\right\|_{H(x_0)^{-1}} \\
&= \left\|H(x_0)^{-1/2}M(x_0)H(x_0)^{-1}\nabla f(x_0)\right\|_2 \\
&\quad \text{(by the fact that } \|u\|_A = \left\|A^{-1/2}u\right\|_2) \\
&\leq \left\|H(x_0)^{-1/2}M(x_0)H(x_0)^{-1/2}\right\| \cdot \left\|H(x_0)^{-1/2}\nabla f(x_0)\right\|_2 \\
&\quad \text{(since } \|Au\|_2 \leq \|A\|\,\|u\|_2) \\
&= \left\|H(x_0)^{-1/2}M(x_0)H(x_0)^{-1/2}\right\| \cdot \|\nabla f(x_0)\|_{H(x_0)^{-1}} \\
&\quad \text{(by the fact that } \|u\|_A = \left\|A^{-1/2}u\right\|_2)
\end{aligned}$$

Thus, to conclude the proof it remains to show that the matrix $M(x_0)$ is "small" in the following sense:

$$\left\| H(x_0)^{-1/2} M(x_0) H(x_0)^{-1/2} \right\| \leq \frac{3}{2} \left\| \nabla f(x_0) \right\|_{H(x_0)^{-1}}.$$

For that, by Fact 1 it is enough to show that

$$-\frac{3}{2}\delta H(x_0) \preceq M(x_0) \preceq \frac{3}{2}\delta H(x_0), \tag{8}$$

where for brevity $\delta := \left\| \nabla f(x_0) \right\|_{H(x_0)^{-1}}$. This in turn follows from the **NL** condition. Indeed, since $\delta = \left\| x_1 - x_0 \right\|_{x_0}$, from **NL**, for every $t \in [0,1]$ we obtain

$$-3t\delta H(x_0) \preceq H(x_0) - H(x_0 + t(x_1 - x_0)) \preceq 3t\delta H(x_0).$$

By integrating this inequality from $t = 0$ to $t = 1$, (8) follows. □

A Convergence proof based on Euclidean norm

Proof of Theorem 2. The basic idea of the proof is the same as in 1. We need a similar tool as the Taylor expansion used in the previous chapter. To obtain such, we consider the function $\phi : [0,1] \to \mathbb{R}^n$, $\phi(t) = \nabla f(x + t(y - x))$. Applying the fundamental theorem of calculus to ϕ (to every coordinate separately) yields:

$$\begin{aligned} \phi(1) - \phi(0) &= \int_0^1 \nabla \phi(t) dt \\ \nabla f(y) - \nabla f(x) &= \int_0^1 H(x + t(y - x))(y - x) dt. \end{aligned} \tag{9}$$

We start by writing $x_1 - x^\star$ in a convenient form

$$\begin{aligned} x_1 - x^\star &= x_0 - x^\star + n(x_0) \\ &= x_0 - x^\star - H(x_0)^{-1} \nabla f(x_0) \\ &= x_0 - x^\star + H(x_0)^{-1} (\nabla f(x^\star) - \nabla f(x_0)) \\ &= x_0 - x^\star + H(x_0)^{-1} \int_0^1 H(x_0 + t(x^\star - x_0))(x^\star - x_0) dt \\ &= H(x_0)^{-1} \int_0^1 (H(x_0 + t(x^\star - x_0)) - H(x_0))(x^\star - x_0) dt. \end{aligned}$$

Now take norms

$$\begin{aligned} \left\| x_1 - x^\star \right\|_2 &\leq \left\| H(x_0)^{-1} \right\| \int_0^1 \left\| (H(x_0 + t(x^\star - x_0)) - H(x_0))(x^\star - x_0) \right\| dt \\ &\leq \left\| H(x_0)^{-1} \right\| \left\| x^\star - x_0 \right\|_2 \int_0^1 \left\| (H(x_0 + t(x^\star - x_0)) - H(x_0)) \right\| dt. \end{aligned} \tag{10}$$

We use the Lipschitz condition on H to bound the integral:

$$\begin{aligned}\int_0^1 \|(H(x_0 + t(x^\star - x_0)) - H(x_0))\| \, dt &\leq \int_0^1 L \, \|t(x^\star - x_0)\|_2 \, dt \\ &\leq L \, \|x^\star - x_0\|_2 \int_0^1 t dt \\ &= \frac{L}{2} \, \|x^\star - x_0\|_2 \, .\end{aligned}$$

Together with (10) this implies:

$$\|x_1 - x^\star\|_2 \leq \frac{L \left\|H(x_0)^{-1}\right\|}{2} \|x^\star - x_0\|_2^2 \tag{11}$$

which completes the proof. We can take $M = \frac{L\left\|H(x_0)^{-1}\right\|}{2} \leq \frac{L}{2h}$. □

B Proof of the Fact

Proof of Fact 1: Since $A^{-1/2}BA^{-1/2}$ is a symmetric matrix, we have

$$\left\|A^{-1/2}BA^{-1/2}\right\| = \sup_{u \neq 0} \frac{\left|u^\top A^{-1/2}BA^{-1/2}u\right|}{u^\top u}.$$

Since $A^{-1/2}$ is invertible we can set $v := A^{-1/2}u$ and obtain

$$\sup_{u \neq 0} \frac{\left|u^\top A^{-1/2}BA^{-1/2}u\right|}{u^\top u} = \sup_{v \neq 0} \frac{\left|v^\top Bv\right|}{v^\top Av}.$$

The last expression is upper-bounded by α from our assumption. □

References

[1] A. Galntai. The theory of newton's method. *Journal of Computational and Applied Mathematics*, 124(1):25 – 44, 2000. Numerical Analysis 2000. Vol. IV: Optimization and Nonlinear Equations.

[2] Y. Nesterov and A. Nemirovskii. *Interior-Point Polynomial Algorithms in Convex Programming*. Society for Industrial and Applied Mathematics, 1994.

[3] Joseph Raphson. *Analysis aequationum universalis seu ad aequationes algebraicas resolvendas methodus generalis, & expedita, ex nova infinitarum serierum methodo, deducta ac demonstrata*. typis Tho. Braddyll, prostant venales apud Johannem Taylor.

[4] James Renegar. A polynomial-time algorithm, based on newton's method, for linear programming. *Math. Program.*, 40(1-3):59–93, 1988.

[5] Daniel A. Spielman and Shang-Hua Teng. Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems. In *STOC'04: Proceedings of the 36th Annual ACM Symposium on the Theory of Computing*, pages 81–90, 2004.