Real Vector Derivatives, Gradients, and Nonlinear Least-Squares

ECE275A – Lecture Notes

Kenneth Kreutz-Delgado
Electrical and Computer Engineering
Jacobs School of Engineering
University of California, San Diego

Report Number ECE275A-LS2-F17v1.0

October 1, 2017

1. Nonlinear Mappings and Vector Derivatives

Differentiable Nonlinear Mappings. In this note we consider general nonlinear mappings between two finite-dimensional *real* vector spaces

$$h(\cdot) : \mathcal{X} = \mathbb{R}^n \to \mathcal{Y} = \mathbb{R}^m. \tag{1}$$

The *image* of a function $h(\cdot)$ is defined to be the image of the domain $\mathcal{X}$ under the action of the mapping,

$$\text{Im}(h) = h(\mathcal{X}) = \{y \mid y = h(x),\ x \in \mathcal{X}\} \subset \mathcal{Y}.$$

If $h(\cdot)$ happens to be linear, then the image of h is also called the range of h. We will assume throughout this note that the m components of any function, $h(x) \in \mathbb{R}^m$, under consideration,

$$h_i(x) = h_i(x_1, \cdots, x_n),\ i = 1, \cdots, m,$$

are at least twice differentiable with respect to the components, x_j, of the vector $x \in \mathbb{R}^n$.

1.1 The Partial Derivative and Jacobian Operator $\frac{\partial}{\partial x}$

The Partial Derivative and Partial Derivative Operator. Consider, a real-valued function

$$f(\cdot) : \mathcal{X} = \mathbb{R}^n \to \mathbb{R}\,.$$

Given a value of the function, $f(x) \in \mathbb{R}$, evaluated at a particular point in the domain $x \in \mathcal{X}$, often we are interested in determining how to increase or decrease the value of $f(x)$ via local displacements from the nominal point x, $x \mapsto x + dx$.

The differential change in the value of the real-valued function $f(x)$ due to a differential change in the vector x, $x \mapsto x + dx$, is given by

$$\begin{aligned} df &= \frac{\partial f(x)}{\partial x_1} dx_1 + \cdots + \frac{\partial f(x)}{\partial x_n} dx_n \\ &= \begin{pmatrix} \frac{\partial f(x)}{\partial x_1} & \cdots & \frac{\partial f(x)}{\partial x_n} \end{pmatrix} \begin{pmatrix} dx_1 \\ \vdots \\ dx_n \end{pmatrix} \end{aligned}$$

or

$$df = \frac{\partial f(x)}{\partial x} dx \tag{2}$$

where to obtain the last relationship, we define the $1 \times n$ *row-vector* (covariant) partial derivative operator. Note that it is defined without any reference to a metric, and thus exists even in a non-metric space.

$$\frac{\partial}{\partial x} \triangleq \begin{pmatrix} \frac{\partial}{\partial x_1} & \cdots & \frac{\partial}{\partial x_n} \end{pmatrix} \tag{3}$$

and the row-vector (covariant) partial derivative quantity $\frac{\partial f(x)}{\partial x}$ by

$$\frac{\partial f(x)}{\partial x} \triangleq \begin{pmatrix} \frac{\partial f(x)}{\partial x_1} & \cdots & \frac{\partial f(x)}{\partial x_n} \end{pmatrix}. \tag{4}$$

In contrast to the definition of the covariant derivative given in (4), in much of the engineering and elementary mathematics literature it is common to define $\frac{\partial f}{\partial x}$ to be the *transpose* of the right-hand-side of (4) and then to identify it as the gradient of f evaluated at the point x (potentially erroneously, as discussed below). *However,* as discussed in advanced courses on Calculus on Manifolds; Differential Geometry; and Geometrical Physics,[1] it is the *row vector* (4) that is the

[1] See, for example, the following references: Ralph Abraham, Jerrold E. Marsden, & Tudor Ratiu, *Manifolds, Tensor Analysis, and Applications,* 1983, Addison-Wesley; Michael Spivak, *A Comprehensive Introduction to Differential Geometry,* 2nd Edition (5 Volumes), 1979, Publish or Perish Press; Charles Misner, Kip Thorne, & John Wheeler, *Gravitation,* 1973, W.H. Freeman & Co.; Bernard Schutz, *Geometrical Methods of Mathematical Physics,* 1980, Cambridge University Press; Theodore Frankel, *The Geometry of Physics, An Introduction,* 2001 (with corrections and additions), Cambridge University Press. I particularly recommend the last-mentioned text by Frankel for use by Engineers and Physicists. A very accessible introduction to many of the important concepts can also be found in the book by Schutz.

"natural" *most general choice* when defining the quantity $\frac{\partial f}{\partial x}$ and *not* the column vector obtained by taking the transpose of (4).[2] As detailed, for example, in the textbook by Frankel,[3] under a change of basis the column vector of partial derivatives obtained by transposing (4) generally does *not* transform like a vector properly should if it is to have intrinsic geometric meaning.[4] Nonetheless, it *is* the transpose of (4) that is usually called the gradient, or gradient vector, of f in the engineering and elementary calculus literature.[5]

Throughout this note we take $\frac{\partial f(x)}{\partial x}$ to be defined by equation (4) and call it the *partial derivative of f with respect to x*, the *covariant derivative*, the *covariant form of the gradient,* or the *cogradient,* to distinguish it from the commonly used *column-gradient* or *gradient vector* which will instead be noted as $\nabla_x f$ (and described in further detail below).[6]

Consistent with the above discussion, we call the row-operator $\frac{\partial}{\partial x}$ defined by equation (3) the *(row) partial derivative operator*, the *covariant form of the gradient operator*, the *cogradient operator*, the *row-gradient operator*, or the *covariant derivative operator.*

First-Order Necessary Condition for an Optimum. Maxima, minima, and inflection points of the differentiable real-valued function f are so-called *stationary points* of f. These are points, x, in the domain of f at which the value $f(x)$ does not change to first order, given arbitrary first-order

[2]Indeed, in the books by Abraham, Marsden, & Ratiu; Misner, Thorne, & Wheeler; and Schutz (*op. cit.* footnote (1)) the *row-vector* equation (4) is taken to define *the* gradient of f. However, following Frankel, *op. cit.*, we more carefully distinguish between equation (4) as defining the *partial derivative* (i.e., as the covariant form of the gradient which exists even in a non-metric space) and the gradient (the contravariant form of the partial derivative given by equation (24),which requires the existence of a metric).

[3]T. Frankel, *op. cit.* footnote (1), pp. 40-47.

[4]That is, if it is to have a meaning which does *not* depend on the particular, accidental, contingent coordinate system one happens to be using at the moment. The calculus on manifolds is concerned with objects and properties which are *invariant* with respect to *general* coordinate transformations as such objects are likely to have intrinsic properties of the space itself which do not depend spuriously on accidents of perspective.

It is true, however, that if one is only interested in proper orthogonal transformations (i.e., rotations) between Cartesian coordinate systems, then the use of the simple column vector of derivatives as the gradient is correct and well-behaved (as discussed further below). This accounts for the accepted and wide-spread use of the column-vector definition of the gradient in those domains where non-Cartesian spaces are rarely, if ever, encountered. The point is that whether or not the column definition of the gradient is "right" or "wrong" depends on the particular problem domain.

[5]Even though the gradient vector defined by simply transposing (4) is not generally *contravariant* (i.e., *generally* does not properly transform like a vector, except in special cases as mentioned at the end of the previous footnote).

[6]We reiterate the fact that the definition (4) used in this note is contrary to the definitions used in many (perhaps most) electrical engineering textbooks, including: Todd Moon & Wynn Stirling, *Mathematical Methods and Algorithms for Signal Processing,* 2000, Prentice Hall; Steven Kay, *Fundamentals of Statistical Signal Processing: Estimation Theory,* 1993, Prentice Hall; and Simon Haykin, *Adaptive Filter Theory, 3rd Edition,* 1996, Prentice Hall. In these texts $\frac{\partial f}{\partial x} = \nabla_x f =$ transpose of right-hand-side of (4). However, there are useful electrical engineering texts which do use the definition (4), including: Graham Goodwin & Robert Payne, *Dynamic System Identification,* 1977, Academic Press; Rolf Johansson, *System Modeling & Identification,* 1993, Prentice Hall; and Harold Sorenson, *Parameter Estimation,* 1980, Marcel Dekker. The definition (4) is also the standard definition used in virtually all robotics textbooks.

variations, dx, about x. The condition for stationarity at the point x is that

$$df(x) = \frac{\partial f(x)}{\partial x}\, dx = 0$$

for *any* arbitrary variation dx about x. This is true if and only if the partial derivative of f vanishes at x, so that

$$x \text{ is a stationary point of } f \Leftrightarrow \frac{\partial f(x)}{\partial x} = 0\,. \tag{5}$$

This is just the well-known stationarity condition that all partial derivatives of f take the value 0 at the point x, $\frac{\partial f(x)}{\partial x_i} = 0$, $i = 1, \cdots, n$. In particular, this is a necessary condition that $f(x)$ has a local minimum at the point x.[7]

The Jacobian Operator. Given a general differentiable vector-valued m-dimensional nonlinear function $h(x)$ of the form (1), and the "natural" definition of the partial derivative operator $\frac{\partial}{\partial x}$ given by (3), we can naturally extend the action of the partial derivative to $h(x)$ by applying the definition (3) *component-wise* to the elements of $h(x)$,

$$\frac{\partial h(x)}{\partial x} \triangleq \begin{pmatrix} \frac{\partial h_1(x)}{\partial x} \\ \vdots \\ \frac{\partial h_m(x)}{\partial x} \end{pmatrix} = \begin{pmatrix} \frac{\partial h_1(x)}{\partial x_1} & \cdots & \frac{\partial h_1(x)}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial h_m(x)}{\partial x_1} & \cdots & \frac{\partial h_m(x)}{\partial x_n} \end{pmatrix} \in \mathbb{R}^{m \times n}. \tag{6}$$

Equation (6) is called the *Jacobian* or *Jacobian matrix* of $h(x)$.[8] For this reason, we also call the partial derivative operator $\frac{\partial}{\partial x}$ the *Jacobian operator*. The Jacobian of h is often denoted by J_h,

$$J_h(x) \triangleq \frac{\partial h(x)}{\partial x}.$$

Coordinate Transformations. A change of (local) coordinate representation corresponds to a mapping $y = h(x)$ with $m = n$. Often this is denoted (with some danger of confusion) by $y = y(x)$. The Jacobian of this coordinate transformation is denoted variously as

$$J_y(x) = J_h(x) = \frac{\partial y(x)}{\partial x} = \frac{\partial h(x)}{\partial x} \in \mathbb{R}^{n \times n}\,,$$

or even merely as J when there is no possibility of confusion. For $y = h(x)$ to provide a valid change of coordinates, the differentiable function $h(\cdot)$ must be one-to-one and onto (i.e., invertible) and have a differentiable inverse $x = h^{-1}(y) = x(y)$. This condition corresponds to the requirement that the $n \times n$ Jacobian J_y be invertible (i.e., nonsingular). Nonsingularity of the Jacobian

[7]Else one could find an infinitesimal displacement dx such that $f(x + dx) = f(x) + \frac{\partial f(x)}{\partial x} dx < f(x)$, thereby contradicting the claim that x is a local minimum.

[8]The Jacobian is discussed at great length in robotics textbooks.

corresponds to the requirement that the *Jacobian determinant of the transformation*[9]

$$\det \frac{\partial h(x)}{\partial x} = \det J_h(x) \in \mathbb{R}$$

exists and be nonzero.

Coordinates do not have to be orthogonal, or normalized, or orthonormal.[10] Coordinate systems which are locally, but not globally, orthogonal (but not necessarily normalized) include the standard curvilinear coordinate systems such as cylindrical coordinates, spherical polar coordinates, parabolic coordinates, ellipsoidal coordinates, toroidal coordinates, etc.[11]

With the assumption that the Jacobian describing a transformation between coordinate representations is nonsingular we have

$$J_x(y) = J_y^{-1}(x) \text{ for } y = y(x)\,.$$

Locally, then, a differentiable change of coordinates $y = h(x)$ is given by

$$dy = J_y(x)\,dx = J_x^{-1}(y)\,dx \quad \text{for} \quad y = h(x)\,. \tag{7}$$

Defining the *tangent vectors*[12]

$$v = \frac{dy}{dt} \quad \text{and} \quad w = \frac{dx}{dt}$$

equation (7) yields

$$v = Jw \tag{8}$$

where it is understood that this expression is assumed to hold for a local coordinate systems located at the *point* $x \in \mathcal{X} = \mathbb{R}^n$ and corresponds to a transformation between coordinate systems. The collection of *tangent vectors*, v, to a manifold $\mathcal{X}$ at the *point* x is called the tangent space at the point x and is denoted by $\mathsf{T}_x\mathcal{X}$. In our case, the manifold is equal to the space $\mathbb{R}^n$, $\mathcal{X} = \mathbb{R}^n$, and the tangent space is is given by[13]

$$\mathsf{T}_x\mathcal{X} = \mathsf{T}_x\,\mathbb{R}^n = \mathbb{R}^n_x \triangleq \{x\} + \mathbb{R}^n\,.$$

[9]The absolute value of the Jacobian determinant is used to obtain the probability density function of the transformed random vector, $Y(\omega) = h(X(\omega))$, from the known pdf $p_X(x)$, as discussed in standard courses on probability theory. In older textbooks, the Jacobian determinant of the transformation is often denoted as

$$J\left(\frac{y_1,\cdots,y_n}{x_1,\cdots,x_n}\right) \text{ or } \frac{\partial(y_1,\cdots,y_n)}{\partial(x_1,\cdots,x_n)}\,.$$

[10]Indeed, the concepts of *normalization* or *orthogonality* do not even exist on general spaces. Only when an appropriate metric or inner product is present do these concepts make sense.

[11]See, e.g., George Arfkin, *Mathematical Methods for Physicists,* 2nd Edition, 1970, Academic Press.

[12]Tangent vectors are further discussed is a slightly more rigorous manner later below.

[13]Although slightly notationlly abusive, often we write $\mathbb{R}^n_x = x + \mathbb{R}^n$,

The tangent space $\mathbb{R}^n_x$ is a version of $\mathbb{R}^n$ which has been translated to the point $x \in \mathbb{R}^n$. One can think of $\mathbb{R}^n_x$ as a copy of $\mathbb{R}^n$ with origin now located at the point x.[14]

Equation (8) shows the change of representation between two different representations of the *same* tangent vector in $\mathsf{T}_x\mathcal{X} = \mathbb{R}^n_x$. Quantities which transform like (8) are said to be *vector-like* or *contravariant* and are viewed as two different representations of the same contravariant object. As shown below in equation (26), *the gradient transforms like a vector* and hence is contravariant.

Transformation of the Partial Derivative Operator. Consider a change of coordinates $y = y(x)$ and a differentiable function $f(y)$. Application of the chain-rule at the component level readily yields the fact that

$$\frac{\partial f(y)}{\partial y} = \frac{\partial f(y)}{\partial x}\,\frac{\partial x}{\partial y} = \frac{\partial f(y)}{\partial x}\,J_x(y) = \frac{\partial f(y)}{\partial x}\,J_y^{-1}(x) \quad \text{for} \quad y = h(x)\,. \tag{9}$$

Thus, under a change of coordinates $y = h(x)$, the partial derivative operator transforms as

$$\frac{\partial(\,\cdot\,)}{\partial y} = \frac{\partial(\,\cdot\,)}{\partial x}\,J_x = \frac{\partial(\,\cdot\,)}{\partial x}\,J_y^{-1}\,. \tag{10}$$

Any two row-vector like objects r and s which transform like[15] (10)

$$r = sJ \tag{11}$$

are said to be *covector-like* or *covariant* and are viewed as two different representations of the same covariant object.[16] Equation (9) shows that *the partial derivative is a covariant object.*

Partial Derivative as Linear Functional and Differential Operator. Note that in the relationship (2), the differential variation df is dependent both upon the choice of the point x and the particular differential variation, dx, taken in the domain of the scalar function $f(x)$,

$$df = df_x(dx) = \frac{\partial f(x)}{\partial x}\,dx\,. \tag{12}$$

[14]Note the careful distinction which is kept between the *point* x which locates a tangent space on the manifold, and the *tangent vectors*, v, which live in this tangent space. The fact that $\mathsf{T}_x\mathcal{X} = \{x\} + \mathbb{R}^n$ is an artifact of the fact that the manifold (ambient space) $\mathcal{X}$ just happens, in this case, to be the Euclidean space $\mathcal{X} = \mathbb{R}^n$. More generally we should write $\mathsf{T}_x\mathcal{X} = \{x\} \times \mathbb{R}^n$. For $(x, v) \in \mathsf{T}_x\mathcal{X} = \{x\} \times \mathbb{R}^n$, the component $x \in \mathcal{X}$ is called a *point* (of $\mathcal{X}$) while the component $v \in \mathbb{R}^n$ is referred to as a vector. By convention $\mathsf{T}_x\mathcal{X} = \{x\} \times \mathbb{R}^n \neq \mathsf{T}_{x'}\mathcal{X} = \{x'\} \times \mathbb{R}^n$ when $x \neq x'$. This more careful notation makes it clear that (x, v) and (x', v) refer to two *different* tangent vectors, even though we are denoting both tangent vectors by the same symbol, v, and also clarifies the distinction between points and vectors. In our case where we can make the identification $\{x\} \times \mathbb{R}^n \approx \{x\} + \mathbb{R}^n$, we can view the points x and x' as locating the *origins* of the two different tangent spaces via a displacement of $\mathbb{R}^n$ via amounts x and x', and the vector v as indicating a direction and magnitude relative to those origins.

[15]it is understood that the quantities r, s, and J are all referenced to the same point $x \in \mathcal{X}$

[16]The collection of all covectors located a a point $x \in \mathcal{X}$ comprises the *cotangent space* $\mathsf{T}^*_x\mathcal{X}$ of $\mathcal{X}$ at the point x. A cotangent vector is a linear functional on the tangent space $\mathsf{T}_x\mathcal{X}$.

Note that df is a differential variation in the elementary calculus sense while

$$df_x(\,\cdot\,) \triangleq \frac{\partial f(x)}{\partial x}\,(\,\cdot\,) \tag{13}$$

is a differential operator which produces a specific differential variation df by acting on a specific differential variation dx.

Heuristically, let us denote an arbitrary differential domain-space variation dx by

$$dx = \epsilon\, v \tag{14}$$

where $\epsilon > 0$ is "infinitesimal"[17] and $v \in \mathbb{R}^n_x$ is an arbitrary vector in the domain space $\mathcal{X} = \mathbb{R}^n$ viewed as having been translated so that its origin is now located at the point x.[18] This yields

$$df_x(dx) = \epsilon\, df_x(v) \tag{15}$$

where

$$df_x(v) = \frac{\partial f(x)}{\partial x} v\,. \tag{16}$$

This heuristic statement can be made rigorous by viewing f as a function over a differentiable parameterized curve $x(t)$,[19] and taking $\epsilon = dt$ in equation (15) so that (15) becomes[20]

$$\frac{dx(t)}{dt} = v(t)\,.$$

At a particular parameter value $t = t_0$, this yields

$$\frac{df}{dt}(x(t_0)) = df_x\left(\frac{dx(t_0)}{dt}\right) = \frac{\partial f(x(t_0))}{\partial x}\,\frac{dx(t_0)}{dt}\,.$$

By taking the "velocity vector" $\frac{dx(t_0)}{dt}$ to have the value $v \in \mathbb{R}^n_{x(t_0)}$, $\frac{dx(t_0)}{dt} = v$, we retrieve equation (16) for $x = x(t_0)$. The vector v is known as a *tangent vector*.[21]

The operator $df_x(\cdot)$ is a so-called *linear functional,* linearly mapping vectors $v \in \mathbb{R}^n_x$ to real scalar values $df_x(v) \in \mathbb{R}_{f(x)}$.[22] The operator $df_x(\cdot)$ is often called the *differential operator* of f. Note that the *partial derivative of f*, the *Jacobian of f* and the *differential operator of f* are all essentially the same object but viewed from different perspectives.

[17]The parameter ϵ corresponds to the gradient-descent algorithm step-size parameter α presented in a later section and used, precisely as heuristically presented here, as a "control" to ensure that finite-difference statements are approximately as good as their equivalent differential statements.

[18]$\mathbb{R}^n_x$ is called the tangent space to $\mathbb{R}^n$ at the point x. See Frankel, *op. cit.* footnote (1), page 7.

[19]Note that then a choice of a value for parameter t then fixes a location on the manifold along the curve.

[20]See the references cited in footnote (1).

[21]It is tangent to the curve $x(t)$ at $t = t_0$ and lives in the tangent space $\mathbb{R}^n_{x(t_0)}$. See ref.'s in footnote (1).

[22]That is, $df_x(\cdot)$ linearly maps vectors from the tangent space to $\mathbb{R}^n$ at the point x to scalar displacements located in the tangent space to $\mathbb{R}$ at the point $f(x)$.

1.2 The Gradient Operator ∇_x

The Metric Tensor. Let us assume that each tangent space, $\mathbb{R}^n_x$ of the space $\mathcal{X} = \mathbb{R}^n$ is an inner-product space with symmetric, positive-definite weighting matrix $\Omega_x = \Omega_x^T > 0$.[23]

The inner-product weighting matrix Ω_x is known as a *metric tensor* in the differential geometry and tensor calculus literature and most generally depends upon the particular point x at which the tangent space $\mathbb{R}^n_x$ is located. The existence of a metric tensor results in the inner-product

$$\langle v_1, v_2 \rangle = v_1^T \Omega_x v_2 \quad v_1, v_2 \in \mathbb{R}^n_x \tag{17}$$

and the corresponding norm

$$\|v\| = \sqrt{v^T \Omega_x v} \quad v \in \mathbb{R}^n_x \, . \tag{18}$$

If Ω_x is a positive definite matrix for all x then the space $\mathcal{X}$ is said to be *Riemannian.* Because a positive definite weighting matrix can always be transformed to a coordinate representation for which the metric is *Cartesian*, $\Omega_x = I$, a Riemannian space is locally Cartesian.[24]

In the theory of calculus on manifolds *scalars* are real (or complex) valued quantities *which are invariant with respect to change of coordinate representations.* Extremely important scalars on a Riemannian space are the *inner product* (17) and the *norm* (18). Their invariance corresponds to the requirement that they have an intrinsic meaning independent of the particular coordinate system that one happens to be using at the moment.

Suppose we have a change of coordinates $y = h(x)$. With the identifications $w = \frac{dy}{dt}$ and $v = \frac{dx}{dt}$, we see from equation (7) that the coordinate transformation $y = h(x)$ induces a change of local coordinates on the tangent space $\mathbb{R}^n_x$ according to

$$w = J_y v = (J_x)^{-1} v \quad \text{or} \quad v = J_x w \, . \tag{19}$$

Equation (19) is *very important* as it shows us how quantities which we can identify as (tangent) vectors[25] *must transform* if important scalar quantities are to remain invariant.[26] If a quantity does

[23]Recall that a finite-dimensional positive-definite inner-product space is a Hilbert space. Our assumption that $\Omega_x > 0$ says that each tangent space is a Hilbert space.

The dependance of the metric tensor Ω_x on the particular point x means that it encodes critical information about the curvature of the space. If for some coordinate system the metric tensor is independent of every location, x, in the space, $\Omega_x = \Omega$, then the space is said to be "flat" or "Euclidean". In this case, the entire space (not just the tangent spaces) is (or can be transformed into) a Hilbert space.

[24]I.e., because Ω_x is assumed positive definite, within each tangent space $\mathbb{R}^n_x$ one can choose to transform to Cartesian coordinates x' for which $\Omega_{x'} = I$. If the space is "flat" (see footnote 23), so that the metric tensor is independent of x, $\Omega_x = \Omega$, then a change of coordinate representation can be performed which will make the space globally Cartesian.

Indefinite metrics can occur for which Ω_x is an indefinite matrix, but for which the resulting inner-product space is nondegenerate in the sense that the only vector orthogonal to every vector is the zero vector. Such spaces are called *pseudo-Riemannian.* Perhaps the best-known pseudo-Riemannian space is the *Minkowski space-time* of Einstein's special theory of relatively, which has the indefinite metrix $\Omega_x = \text{diag}\,(-1, +1, +1, +1)$; see Misner, Thorne, & Wheeler, *op. cit.* footnote (1).

[25]I.e., as vector rates of change or as local vector displacements.

[26]Quantities which transform like vectors, i.e., like equation (19), are said to be *contravariant.*

not transform according to (19) under a change of basis, it is *not* a vector in the sense demanded by considerations of invariance under a change of coordinates.

Invariance of the norm (18) yields,

$$\|v\|^2 = v^T \Omega_x v = w^T J_x^T \Omega_x J_x w = w^T \Omega_y w = \|w\|^2 \, .$$

Since this must be true for all possible tangent vectors v, we obtain the important and useful result that

$$\Omega_y = J_x^T \Omega_x J_x = (J_y)^{-T} \, \Omega_x \, (J_y)^{-1} \ , \tag{20}$$

where $J_y = \frac{\partial y(x)}{\partial x}$. Note that (20) can be viewed as providing two ways to construct the new metric tensor Ω_y from knowledge of the old metric tensor Ω_x and the transformation $y = h(x)$. Specifically, one can determine Ω_y either via the relationship

$$\Omega_y = J_x^T \Omega_x J_x \tag{21}$$

or via

$$\Omega_y = (J_y)^{-T} \, \Omega_x \, (J_y)^{-1} \ . \tag{22}$$

Not surprisingly, it is usually easiest to compute J_x directly (when possible) and construct Ω_y from (21), thereby forgoing the often difficult symbolic inversion involved in computing J_y^{-1} from J_y which is required to compute Ω_y via equation (22).

To determine a metric tensor in new coordinates, usually it is simplest to start with a known Cartesian (orthonormal) basis on $\mathbb{R}^n_x$, for which we have (with the identification $v \sim dx$)

$$\|dx\|^2 = (dx_1)^2 + \cdots + (dx_n)^2 \, .$$

Together with the differential relationships

$$dx_j = \frac{\partial x_j}{\partial y_1} \, dy_1 + \cdots + \frac{\partial x_j}{\partial y_n} \, dy_n \quad j = 1, \cdots, n$$

one can determine the components of the metric tensor Ω_y. Of course this is precisely the procedure suggested by equation (20), taking $w \sim dy$ and $\Omega_x = I$.

Derivation of the Gradient Operator. Given a metric tensor Ω_x we can transform the partial derivative operator into a gradient operator as follows.

With the inner product $\langle v_1, v_2 \rangle = v_1^T \Omega_x v_2$, we note that equation (16) can be written as

$$\begin{aligned} df_x(v) &= \frac{\partial f(x)}{\partial x} \, v \\ &= \left[\Omega_x^{-1} \left(\frac{\partial f(x)}{\partial x} \right)^T \right]^T \Omega_x \, v \\ &= \left\langle \Omega_x^{-1} \left(\frac{\partial f(x)}{\partial x} \right)^T , \, v \right\rangle \end{aligned}$$

or

$$df_x(v) = \langle \nabla_x f(x), v \rangle \tag{23}$$

where

$$\nabla_x f(x) \triangleq \Omega_x^{-1} \left(\frac{\partial f(x)}{\partial x} \right)^T = \Omega_x^{-1} \begin{pmatrix} \frac{\partial f(x)}{\partial x_1} \\ \vdots \\ \frac{\partial f(x)}{\partial x_n} \end{pmatrix} \in \mathbb{R}^n . \tag{24}$$

The quantity $\nabla_x f(x)$ is called the *gradient of f at the point x* and is *uniquely* defined by equation (24).[27] The gradient of f is also known as the *contravariant derivative* of f.

Now let v be an arbitrary unit vector, $\|v\| = 1$. From the Cauchy-Schwarz inequality we have

$$|df_x(v)| \le \|\nabla_x f(x)\| \|v\| = \|\nabla_x f(x)\|$$

with equality holding if and only if v is proportional to $\nabla_x f(x)$, i.e., if and only if the unit vector v is given by

$$v = \pm \frac{\nabla_x f(x)}{\|\nabla_x f(x)\|} .$$

These choices for v give the *directions of steepest change* in the local value of the function f as a function of the choice of $dx = \epsilon\, v$ and they yield[28]

$$df_x(v) = \pm \|\nabla_x f(x)\| = \pm \sqrt{\frac{\partial f(x)}{\partial x}\ \Omega_x^{-1}\ \frac{\partial f(x)}{\partial x}^T} .$$

The choice

$$v \propto \nabla_x f(x)$$

gives the (local) *direction of steepest ascent of f*, while the choice

$$v \propto -\nabla_x f(x)$$

gives the (local) *direction of steepest descent of f*. Note that the concept of "steepest direction" depends on the choice of metric, so that it is no accident that the gradient of f depends on the metric tensor Ω_x.

On a space with a well-defined metric Ω_x, from the partial derivative operator (3) we define the *gradient operator* as

$$\nabla_x \triangleq \Omega_x^{-1} \left(\frac{\partial}{\partial x} \right)^T = \Omega_x^{-1} \begin{pmatrix} \frac{\partial}{\partial x_1} \\ \vdots \\ \frac{\partial}{\partial x_n} \end{pmatrix} . \tag{25}$$

[27] Our result is a special case of a more general result, the *Riesz Representation Theorem,* which says that any bounded linear functional on a general Hilbert space acting on any point, v, in that space, can be represented as an inner product of v with a unique vector in the space which is said to provide a *representation* of the linear functional. (See Arch Naylor & George Sell, *Linear Operator Theory in Engineering and Science,* 1982, Springer-Verlag.) In our case the linear functional is the differential operator $df_x(\cdot)$ (fixing the point x) and the unique representing vector is the gradient of f at the point x, $\nabla_x f$.

[28] Remember that we are working with a weighted inner product and norm with weighting matrix Ω_x.

The gradient operator has meaning on a Riemannian space[29] while the partial derivative operator has meaning in a general vector space setting. Furthermore, even in a Riemannian space the gradient operator can have the simple form $\left(\frac{\partial}{\partial x}\right)^T$ provided that the metric is Cartesian, $\Omega_x = I$, as we shall now discuss.

The Gradient Vector as a "Natural Gradient." Simply taking the transpose of the covector is how the gradient vector is commonly defined in elementary calculus books and in many engineering textbooks and research papers. Let us call the object obtained in this manner, for now, the *naive gradient*,[30]

$$\nabla_x^{\text{naive}} f(x) = \left(\frac{\partial f(x)}{\partial x}\right)^T .$$

Note from (9) that the naive gradient transforms under a change of coordinates $y = h(x)$ according to

$$\nabla_y^{\text{naive}} f(y) = J_y^{-T}\,\nabla_x^{\text{naive}} f(y) \quad \text{where} \quad y = h(x)\,.$$

Note, then, that the naive gradient does *not* generally yield a vector as it does not transform according to equation (19). Only when the Jacobian coordinate transformation corresponds to an orthogonal transformation

$$J_y^{-1} = J_y^T$$

does the naive gradient transform as a vector should.

In contrast, equations (9), (24), and (22) yield,

$$\begin{aligned}
\nabla_y f(y) &= \Omega_y^{-1}\left(\frac{\partial f(y)}{\partial y}\right)^T \\
&= \left[(J_y)^{-T}\,\Omega_x\,(J_y)^{-1}\right]^{-1} J_y^{-T}\left(\frac{\partial f(y)}{\partial x}\right)^T \\
&= J_y\,\Omega_x^{-1}\left(\frac{\partial f(y)}{\partial x}\right)^T \\
&= J_y\,\nabla_x f(y) \quad \text{where} \quad y = h(x)
\end{aligned} \tag{26}$$

showing that the gradient defined by (24) transforms in the manner of equation (19), i.e., like a vector (contravariantly).[31] Thus the magnitude and the direction of the gradient (which we know to be the direction of steepest ascent of the function $f(x)$) is invariant with respect to coordinate transformations.[32]

[29]More generally the gradient operator has meaning on pseudo-Riemannian spaces (i.e., on a space with a nondegenerate metric) such as the Minkowski space-time of special relativity. See footnote (24).

[30]Later, we will see that it corresponds to the *Cartesian* gradient when certain conditions are satisfied.

[31]This justifies our earlier comment that the gradient of f is also called the contravariant derivative of f.

[32]This is a consequence of the invariance of the inner product (so that angle of direction is invariant) and the norm (so that magnitude is invariant) for vectors transforming like (19) and a metric tensor transforming like (20).

Now from equation (24) one might claim that the naive gradient is acceptable provided that the metric tensor is Cartesian, $\Omega_x = I$, and this would be correct in the sense that the gradient would give the correction direction of steepest ascent. However this still does not rectify the fact that the naive gradient does not *generally* transform like a vector should.[33] *However, it is the case that in elementary treatments of calculus one usually only considers changes of coordinates which are pure rotations*[34] *between Cartesian spaces and for this situation the naive gradient does behave appropriately.* Thus as long as one works in this elementary setting, the distinction between the partial derivative and the gradient vanishes. However, this restriction disallows the use of curvilinear coordinates and other coordinate systems having non-Cartesian metric tensors, $\Omega_x \neq I$.

For this reason, we will now refer to the "naive gradient" as the "Cartesian gradient," under the assumption that the user is not naive, but rather is well aware of the assumptions (usually implicit) which underly the valid use of this gradient. Of course, one might intentionally use the Cartesian gradient "naively," (as is commonly done in descent algorithms for ease of computation, as discussed below) even in the more general setting. In this case, however, one cannot claim that the "gradient" is truly the gradient, i.e., that it provides the direction of steepest ascent. Furthermore, the magnitude of the naive gradient and its direction (whatever it might mean) are not invariant under a general change of coordinates.

Because the gradient (24) gives the direction of steepest descent in *any* coordinate system, it is sometime referred to as the "natural gradient" in the sense that via its dependence on the metric tensor, Ω_x, it conforms to the natural (coordinate-invariant) level-set structure which a functional induces on a Riemannian space.[35] For this reason, it has been argued in some of the recent learning theory literature that the use of the gradient (24) is preferable to the (possibly naive) use of the Cartesian gradient in gradient descent-based learning and parameter estimation.[36]

To summarize, we have seen that the concept of a gradient as a vector and as an object defining a (local) direction of steepest ascent of the values of a functional only makes sense in an inner product space with a well-defined metric tensor. In general, then, the "gradient" formed from simply transposing the partial derivative is generally *not* a well-defined geometric object.[37] Even in a Riemannian space, the naive gradient is a true gradient if and only the metric is Cartesian (i.e., if and only if $\Omega_x = I$) *and* we restrict ourselves to orthogonal coordinate transformations.

[33]A general transformation $y = h(x)$ can destroy the Cartesian assumption, resulting in $\Omega_y \neq I$.

[34]Recall that a pure rotation is a proper orthogonal transformation.

[35] See Shun-ichi Amari, "Natural Gradient Works Efficiently in Learning," *Neural Computation,* 10:251-76, 1998; or Shun-ichi Amari & Hiroshi Nagaoka, *Methods of Information Geometry,* 2000, American Mathematical Society/Oxford University Press. Amari and his coworkers have argued extensively for the importance of applying differential geometric ideas and tools to problems of statistical inference. Using the tools of differential geometry they have developed algorithms and insights into important domains such as turbo coding, density estimation, blind source separation, adaptive filtering, etc.

[36]In particular, this has been argued forcefully by Shun-ichi Amari and his research colleagues, *op. cit.* footnote (35). We will discuss the Natural Gradient algorithm of Amari in a later section.

[37]Indeed, *it may not even be the gradient,* if the gradient is to be understood in the correct sense of it providing the direction of steepest ascent.

The true, coordinate transformation-invariant gradient which gives the actual direction of steepest ascent on a Riemannian space is the "natural" gradient given in equation (24).

1.3 Vector Derivative Identities

Comparing the partial derivative and the gradient, it is evident that the partial derivative is the more fundamental object as it exists and is well defined even when a space is not an inner product space. Furthermore, because of its simple structure it is usually straightforward to construct operator identities involving the partial derivative. For this reason, even if one is primarily interested in applications involving the gradient, to obtain gradient identities it is usually a good policy to first prove the equivalent identities involving partial derivatives and then transform these identities into gradient identities via equation (24).

The following very useful partial derivative identities can be readily shown to be true by component-level and other considerations:

Partial Derivative Identities

$$\frac{\partial\, c^T x}{\partial x} = c^T \quad \text{for an arbitrary vector } c \tag{27}$$

$$\frac{\partial\, Ax}{\partial x} = A \quad \text{for an arbitrary matrix } A \tag{28}$$

$$\frac{\partial\, g^T(x)h(x)}{\partial x} = g^T(x)\frac{\partial\, h(x)}{\partial x} + h^T(x)\frac{\partial\, g(x)}{\partial x}, \quad g(x)^T h(x) \text{ scalar} \tag{29}$$

$$\frac{\partial x^T A x}{\partial x} = x^T A + x^T A^T \quad \text{for an arbitrary matrix } A \tag{30}$$

$$\frac{\partial x^T \Omega x}{\partial x} = 2x^T\Omega \quad \text{when } \Omega = \Omega^T \tag{31}$$

$$\frac{\partial h(g(x))}{\partial x} = \frac{\partial h}{\partial g}\frac{\partial g}{\partial x} \qquad \text{(Chain Rule)} \tag{32}$$

Note that one can readily transform these partial derivative identities into equivalent gradient identities, particularly in the simple case when the Hilbert space containing x is Cartesian.[38] Identity (27) can be easily shown to be true at the component level. Identity (28) follows from application of Identity (27) to $b^T Ax$ for arbitrary b. Identity (29) can be proved by component-level considerations. Identity (30) is a special case of (29) via the identifications $g(x) = x$ and $h(x) = Ax$.

[38] A Variety of such identities for $\frac{\partial}{\partial x}$ defined to be a row-vector, as in done in this note, can be found in Phoebus Dhrymes, *Mathematics for Econometrics,* 4rd Edition, 2013, Springer. A very large number of such identities for $\frac{\partial}{\partial x'}$ defined to be a row vector and $\frac{\partial}{\partial x}$ defined to be a column vector can be found in Helmut Lütkepohl, *Handbook of Matrices,* 1996, Wiley.

Identity (31) follows from (30) via the identification $A = \Omega$. Identity (32) can be proved by component-level considerations.

Finally, note that the partial derivative Identity (32) is a statement about Jacobians and can be restated in an illuminating manner as

$$J_{h\circ g} = J_h \, J_g \,, \tag{33}$$

which says that "the linearization of a composition is the composition of the linearizations."

2. Application to Nonlinear Least-Squares Problems

2.1 Nonlinear Least-Squares Loss Function

As an *important example* of the framework outlined above and the usefulness of the partial derivative identities (27)-(32), consider the derivation of the vector derivative of the *nonlinear weighted least-squares loss function*,[39]

$$\ell(x) = \frac{1}{2}\|y - h(x)\|_W^2 = \frac{1}{2}\|e(x)\|_W^2 = \frac{1}{2}e(x)^T W e(x)\,, \quad e = y - h(x)\,, \tag{34}$$

with W an $m \times m$ positive-definite matrix W.[40] We have, as a straight-forward application of the partial derivative identities,

$$\frac{\partial}{\partial x}\,\ell(x) = \frac{\partial \ell}{\partial e}\,\frac{\partial e}{\partial x} = e^T W\,\frac{\partial e}{\partial x} = -e^T W\,\frac{\partial}{\partial x} h(x) = -\left(y - h(x)\right)^T W\,\frac{\partial}{\partial x} h(x)\,. \tag{35}$$

Comment. We will usually assume in the sequel that that x belongs to a Cartesian space (the Euclidean space $\mathbb{R}^n$ with the Cartesian metric $\Omega_x = I$). The resulting gradient, then will be equivalent to the Cartesian gradient discussed above.

This assumption is made in order to conform with the standard gradient-descent theory which is based on the use of the Cartesian gradient, even though the extension to the case $\Omega_x \neq I$ is very straightforward[41] In any event, the restriction of the central development to the Cartesian case $\Omega_x = I$ in the end will not result in any loss of

[39]The factor $\frac{1}{2}$ is added merely for convenience. Recall that W must be symmetric, positive-definite and corresponds to the use of a weighted inner-product on $\mathcal{Y} = \mathbb{R}^m$.

[40]With the assumption that $e = y - h(x)$ is small, we can view e as essentially living in the tangent space $\mathbb{R}^m_y$ showing that we can take $W = W_y$ to be the metric tensor on $\mathbb{R}^m_y$. However, because y is kept constant, there is no need to explicitly show the y-dependence of the metric tensor if it should exist.

Alternatively, we could take $W = W_{\hat{y}}$, where $\hat{y} = h(\hat{x})$, with $\hat{x}$ a current estimate of a solution to the nonlinear least-squares problem. However, as the required changes to the algorithms derived below are straightforward, we choose to keep W constant to simplify the notation.

[41]As we shall make clear via a series of running footnotes on the simple modifications required to deal with the case $\Omega_x \neq I$.

generality because we will later consider *generalized* gradient descent-algorithms, a class of algorithms which includes "natural" gradient-descent algorithms[42] as special cases.

With the assumption that $\Omega_x = I$, the (Cartesian) gradient of the weighted least-squares loss function is given by

$$\nabla_x \ell(x) = \left(\frac{\partial}{\partial x}\ell(x)\right)^T = -\left(\frac{\partial h(x)}{\partial x}\right)^T W\,(y - h(x)) = -\left(\frac{\partial h(x)}{\partial x}\right)^* (y - h(x))\;, \tag{36}$$

where

$$\left(\frac{\partial h(x)}{\partial x}\right)^* = \left(\frac{\partial h(x)}{\partial x}\right)^T W$$

is the adjoint operator of the Jacobian matrix $\frac{\partial h(x)}{\partial x}$ with respect to the weighted inner product on $\mathcal{Y} = \mathbb{R}^m$.[43]

2.2 Multivariate Taylor Series Expansion

Using the partial derivative notation developed above, we can denote the first-order Taylor series expansion of a *vector-valued* function $h(x)$ about a point x_0 as,

$$h(x) = h(x_0 + \Delta x) = h(x_0) + \frac{\partial h(x_0)}{\partial x}\Delta x + \text{higher order terms}\,, \tag{37}$$

where $\Delta x \triangleq x - x_0$. Note in particular that if we set

$$\Delta y \triangleq h(x) - h(x_0) = h(x_0 + \Delta x) - h(x_0)$$

we have

$$\Delta y = \frac{\partial h(x_0)}{\partial x}\,\Delta x + \text{higher order terms}\,,$$

showing that the Jacobian matrix,

$$H(x) \triangleq \frac{\partial h(x)}{\partial x}\,, \tag{38}$$

provides the *linearization* of $h(x)$,

$$\Delta y \approx H(x_0)\Delta x\,.$$

The differential statement is, of course, exact

$$dy = \frac{\partial h(x_0)}{\partial x}\,dx = H(x_0)\,dx\,.$$

[42]That is, gradient descent algorithms based on the use of the gradient (24) as suggested by S. Amari, *op. cit.* footnote (35).

[43]Note that if we had used the natural gradient for the case $\Omega_x \neq I$, we would have obtained $\left(\frac{\partial h(x)}{\partial x}\right)^* = \Omega_x^{-1}\left(\frac{\partial h(x)}{\partial x}\right)^T W$ in equation (36) from the fact that $\nabla_x \ell(x) = \Omega_x^{-1}\left(\frac{\partial h(x)}{\partial x}\right)^T$.

The Taylor series expansion of a *scalar-valued* function $f(x)$ to second-order can be written as

$$f(x_0 + \Delta x) = f(x_0) + \frac{\partial f(x_0)}{\partial x}\Delta x + \frac{1}{2}\Delta x^T \mathcal{H}(x_0)\Delta x + \text{h.o.t.} \tag{39}$$

where $\mathcal{H}(x)$ denotes the *Hessian matrix* of second-order partial derivatives,[44]

$$\mathcal{H}(x) \triangleq \frac{\partial^2 f(x)}{\partial x^2} \triangleq \frac{\partial}{\partial x}\left(\frac{\partial f(x)}{\partial x}\right)^T = \left(\frac{\partial^2 f(x)}{\partial x_i x_j}\right).$$

Therefore we can approximate the scalar-function $f(x)$ to quadratic order about a point x_0 as

$$\begin{aligned} f(x) &\approx f(x_0) + \frac{\partial f(x_0)}{\partial x}\Delta x + \frac{1}{2}\Delta x^T \mathcal{H}(x_0)\Delta x & (40)\\ &= f(x_0) + \left(\nabla_x f(x_0)\right)^T \Delta x + \frac{1}{2}\Delta x^T \mathcal{H}(x_0)\Delta x\,. & (41)\end{aligned}$$

Note that in this subsection only the last step depends on the assumption that x belongs to a Cartesian space, $\Omega_x = I$, allowing us to take $\nabla_x f(x_0) = \left(\frac{\partial f(x_0)}{\partial x}\right)^T$.[45]

2.3 The Nonlinear Least-Squares Problem

Suppose we want to solve the *nonlinear inverse problem*

$$y \approx h(x)$$

for a given nonlinear function $h(\cdot) : \mathcal{X} \to \mathcal{Y}$. We assume that $h(\cdot)$ is (locally) one-to-one[46] but generally not onto, $\text{Im}(h) = h(\mathcal{X}) \neq \mathcal{Y}$.[47] We continue to make the assumption that the metric tensor is Cartesian, $\Omega_x = I$, while on the codomain $\mathcal{Y}$ the inner-product weighting matrix (metric tensor on $\mathcal{Y}$) is taken to be an arbitrary symmetric positive-definite $m \times m$ matrix W. Defining the discrepancy between y and $h(x)$ by the error $e(x) = y - h(x)$, a rational way to proceed is to find a value for x which minimizes the nonlinear least-squares loss function defined above in equation (34).

[44] It is straightforward to show that the Hessian is symmetric, $\mathcal{H}^T = \mathcal{H}$. However, in general, the Hessian is not guaranteed to be positive definite.

[45] Note however that we could just as easily have written the quadratic expansion of $f(x)$ in terms of the natural gradient because the linear term can be alternatively written as

$$\frac{\partial f(x_0)}{\partial x}\Delta x = \langle \nabla_x f(x_0),\, \Delta x\rangle$$

where here $\nabla_x f$ is the natural gradient and the inner product is the weighted inner product associated with the metric tensor Ω_x.

[46] That is, $h(x)$ is one-to-one in a neighborhood of the point x.

[47] Note that "generally not onto" means that we *might* have $h(\cdot)$ onto. We want to have a generally enough development that we can handle both the case where $h(\cdot)$ is onto as well as the case where it is not onto.

As discussed earlier in equation (5) *et seq.*, a necessary condition for x_0 to be a minimum for the loss function is that the partial derivative of the loss function evaluated at x_0 vanish. Of course, as we see from equation (24), an entirely equivalent condition is that the gradient vanish,

$$\nabla_x \ell(x_0) = 0 \,,$$

which from equation (36) above results in the *nonlinear normal equations*

$$H^*(x_0)\,(y - h(x_0)) = 0 \tag{42}$$

where $H(x) = \frac{\partial h(x)}{\partial x}$ is the Jacobian matrix of $h(x)$ and $H^*(x) = H^T(x)W$ is the adjoint operator to the Jacobian $H(x)$ assuming the inner-product weighting matrices $\Omega_x = I$ and W on the domain and codomain respectively.[48]

One can interpret the condition (42) geometrically as indicating that the error $e(x) = y - h(x)$ must be orthogonal to the tangent hyperplane spanned by the columns of the Jacobian $H(x)$ at an optimal point x_0. Under the assumed condition that $h(x)$ is locally one-to-one at the point x it must be the case that the Jacobian matrix $H(x)$ is one-to-one (has full column rank)[49] and therefore the $n \times n$ matrices $H(x)^T H(x)$ and $H^*(x)H(x) = H^T(x)WH(x)$ are invertible.

Note that if $h(x)$ happens to be linear, so that $h(x) = Ax$, then $H(x) = A$, $H^*(x) = A^T W$ and the condition (42) becomes the standard linear least-squares normal equations,

$$A^T W A\, x_0 = A^T W y \,.$$

It is well-known that a *sufficient* condition for x_0 to be a local minimum for the loss function $\ell(x)$ is that the stationarity condition (42) holds at x_0 *and* the Hessian matrix of $\ell(x)$ be positive definite at x_0,

$$\mathcal{H}(x_0) = \frac{\partial}{\partial x}\left(\frac{\partial \ell(x_0)}{\partial x}\right)^T = \left(\tfrac{\partial^2 \ell(x_0)}{\partial x_i \partial x_j}\right) > 0 \,. \tag{43}$$

To compute the Hessian, note from (35) that

$$\left(\frac{\partial \ell(x)}{\partial x}\right)^T = -\left(\frac{\partial h(x)}{\partial x}\right)^T W\,(y - h(x)) = -\sum_{i=1}^{m}\left(\frac{\partial h_i(x)}{\partial x}\right)^T [W\,(y - h(x))]_i$$

where $h_i(x)$ is the i-th component of the vector function $h(x)$ and $[W\,(y - h(x))]_i$ denotes the i-th

[48]For the general case $\Omega_x \neq I$ one obtains the nonlinear normal equations (42) but with the adjoint Jacobian given by $H^*(x) = \Omega_x^{-1} H^T W$.

[49]The linearization $H(x)$ being one-to-one in the neighborhood of the point x is a necessary and sufficient condition for the nonlinear function $h(x)$ to be locally one-to-one in a neighborhood of x. Similarly, $h(x)$ is a locally onto function if and only if $H(x)$ is onto for all points in a neighborhood of x.

component of $W\,(y - h(x))$. Then

$$\begin{aligned}\mathcal{H}(x) &= \frac{\partial}{\partial x}\left(\frac{\partial \ell(x)}{\partial x}\right)^T \\ &= \left(\frac{\partial h(x)}{\partial x}\right)^T W \left(\frac{\partial h(x)}{\partial x}\right) - \sum_{i=1}^{m} \frac{\partial}{\partial x}\left(\frac{\partial h_i(x)}{\partial x}\right)^T [W\,(y - h(x))]_i \\ &= H(x)^T W\, H(x) - \sum_{i=1}^{m} \frac{\partial}{\partial x}\left(\frac{\partial h_i(x)}{\partial x}\right)^T [W\,(y - h(x))]_i\end{aligned}$$

or

$$\mathcal{H}(x) = H^T(x) W H(x) - \sum_{i=1}^{m} \mathcal{H}^{(i)}(x)\,[W\,(y - h(x))]_i \tag{44}$$

where

$$\mathcal{H}^{(i)}(x) \triangleq \frac{\partial}{\partial x}\left(\frac{\partial h_i(x)}{\partial x}\right)^T$$

denotes the Hessian of the the i-th scalar-valued component of the vector function $h(x)$.[50]

Evidently, the Hessian matrix of the least-squares loss function $\ell(x)$ can be quite complex. Also note that because of the second term on the right-hand-side of (44), the Hessian $\mathcal{H}(x)$ can become singular or indefinite, further complicating numerical algorithms (such as the Newton method discussed below) which are based on inverting the Hessian or assumptions of Hessian positive-definiteness.

However, in the special case when $h(x)$ is *linear*, $h(x) = Ax$, we have that $H(x) = A$ and $\frac{\partial H_i(x)}{\partial x} = 0$, $i = 1, \cdots n$, yielding,

$$\mathcal{H}(x) = H^T(x) W H(x) = A^T W A\,,$$

which for full column-rank A and positive W is always symmetric and invertible.

The conditions (42) and (43) tell us when we have found a locally optimal solution. The question still remains as to how we actually find one. Immediately below we discuss two iterative techniques for finding an optimum. Specifically, we present the Newton Method and the Gauss-Newton Method and discuss how they can be interpreted as special cases of Generalized Gradient Descent, where the Generalized Gradient Descent methods are a general family of methods which generalize the standard gradient descent method of nonlinear optimization and include new proposed methods, such as the Natural Gradient Method of Amari.

The Newton Method. The *Newton method* is based on reiteratively minimizing the quadratic approximation (40) of the loss function $\ell(x)$ evaluated at a current estimate, $\hat{x}_k$ of the unknown

[50]Note that all terms on the right-hand-side of (44) are symmetric, as required if $\mathcal{H}(x)$ is to be symmetric.

optimal solution x_0,[51]

$$\ell(x) = \ell(\hat{x}_k + \Delta x_k) \approx \hat{\ell}^{\text{Newton}}(\Delta x_k) \triangleq \ell(\hat{x}_k) + \frac{\partial \ell(\hat{x}_k)}{\partial x}\Delta x_k + \frac{1}{2}\Delta x_k^T \mathcal{H}(\hat{x}_k)\Delta x_k \,, \tag{45}$$

where $\Delta x_k = x - \hat{x}_k$.[52] To minimize the approximation (45) with respect to Δx_k, we solve the necessary condition

$$\frac{\partial}{\partial \Delta x_k}\hat{\ell}^{\text{Newton}}(\Delta x_k) = 0 \,, \tag{46}$$

for an optimum "update" Δx_k. Using the vector derivative identities given above and assuming the invertibility of the Hessian matrix $\mathcal{H}(\hat{x}_k)$,[53] the necessary condition (46) yields,

$$\Delta x_k = \mathcal{H}^{-1}(\hat{x}_k) H^T(\hat{x}_k) W \left(y - h(\hat{x}_k) \right) \,, \tag{47}$$

where the last step follows from equations (36) and (38). Note that the Hessian matrix

$$\frac{\partial^2}{\partial (\Delta x_k)^2}\hat{\ell}^{\text{Newton}}(\Delta x_k)$$

of the *approximation* (45) is equal to $\mathcal{H}(\hat{x}_k)$.[54]

Once we have determined the correction, we can then obtain an improved estimate of the optimal solution as

$$\hat{x}_{k+1} = \hat{x}_k + \Delta x_k = \hat{x}_k + \mathcal{H}^{-1}(\hat{x}_k) H^T(\hat{x}_k) W \left(y - h(\hat{x}_k) \right) \,. \tag{48}$$

Note that if $\hat{x}_k$ is already an optimum solution, then $\nabla_x \ell(\hat{x}_k) = 0$ yielding $\Delta x_k = 0$.

Equation (48) provides an iterative method for generating estimates of an unknown optimum solution x_0. As discussed below, the iterative algorithm (48) is usually slightly generalized by including a positive, real-valued step-size parameter α_k in the correction term

$$\Delta x_k \to \alpha_k \Delta x_k = \alpha_k \, \mathcal{H}^{-1}(\hat{x}_k) H^T(\hat{x}_k) W \left(y - h(\hat{x}_k) \right) \,, \tag{49}$$

yielding the

[51]Note that the approximation (45) is equivalent to assuming that the loss function $\ell(x)$ is well-modelled locally as a quadratic function of x.

[52]Which is equivalent to $x = \hat{x}_k + \Delta x_k$.

[53]Note that it is important therefore that the Hessian be numerically well-posed which, as discussed earlier, can be problematic. Amari, *op. cit.* footnote (35), has argued that this can be a reason to prefer the Natural Gradient method to the Newton Method.

[54]Also note that nowhere have we used any assumptions on the nature of Ω_x.

Newton Algorithm:[55]

$$\boxed{\hat{x}_k + \alpha_k \mathcal{H}^{-1}(\hat{x}_k) H^T(\hat{x}_k) W \left(y - h(\hat{x}_k)\right)} \tag{50}$$

Often the step size is taken to be constant, $\alpha_k = \alpha \leq 1$. The simple choice $\alpha = 1$ is sometimes called the *Newton step* as this choice retrieves the pure Newton algorithm (48).

With an appropriate choice of the step-sizes α_k,[56] the algorithm (50) can usually be stabilized and the iteratively produced estimates $\hat{x}_k$ converge to a locally optimal solution x_{opt},

$$\hat{x}_\infty \triangleq \lim_{k\to\infty} \hat{x}_k = x_{\text{opt}}$$

assuming that $\mathcal{H}(\hat{x}_k$ is positive-definite for all $\hat{x}_k$ and at the solution point $\hat{x}_\infty = x_{\text{ opt}}$. However, the initial condition $\hat{x}_0$ used in (50) often results in different locally optimal solutions, so that a variety of initializations are usually tried. The resulting locally optimal solutions are then compared and the most optimal of them is kept as the final solution.[57]

As discussed in textbooks on optimization theory, the Newton method usually has robust and fast convergence properties (particularly if the step-size α_k is optimized to enhance the rate of convergence). Unfortunately, the method is also usually difficult to implement as the construction of the Hessian via equation (44) and its subsequent inversion at each iteration-step is usually difficult and time consuming. For these reasons other methods, such as the Gauss-Newton method discussed next, are more often used.

The Gauss-Newton Method. Whereas the Newton method is based on reiteratively approximating the loss function about a current estimate $\hat{x}_k$ to quadratic order, the *Gauss-Newton method* is based on *reiteratively linearizing the inverse problem* about the current estimate. Note from equation (37) that an expansion of $h(x)$ about a current estimate $\hat{x}_k$ yields

$$e(x) = y - h(x) \approx y - h(\hat{x}_k) - H(\hat{x}_k)\Delta x_k = \Delta y_k - H(\hat{x}_k)\Delta x_k \,,$$

[55] Note that in the derivation of the Newton algorithm we have nowhere made the assumption that $\Omega_x = I$. Thus there is no need to modify the algorithm in the case $\Omega_x \neq I$. Note that in terms of the Cartesian gradient

$$\nabla_x^{\text{Cart}} \ell(x) = \frac{\partial \ell(x)}{\partial x}^T = H^T(x) W \left(y - h(x)\right)$$

the Newton algorithm can be written as

$$\hat{x}_k + \alpha_k \mathcal{H}^{-1}(\hat{x}_k) \nabla_x^{\text{Cart}} \ell(\hat{x}_k) \,.$$

[56] Quite often the Newton step $\alpha_k = 1$ will suffice. However, a vast literature exists on appropriate choices of the step-size α_k that will not only ensure stability but will provide the fastest rates of convergence. Good references in this regard are D. Luenberger, *Introduction to Linear and Nonlinear Programming,* 1984, Addison-Wesley, and D. Luenberger, *Optimization by Vector Space Methods,* 1969, Wiley.

[57] The question of if and when the true globally optimal solution has been found is generally a difficult one to answer.

and therefore

$$\ell(x) = \ell(\hat{x}_k + \Delta x_k) = \frac{1}{2}\|e(x)\|_W^2 \approx \hat{\ell}^{\text{Gauss}}(\Delta x_k) \triangleq \frac{1}{2}\|\Delta y_k - H(\hat{x}_k)\Delta x_k\|_W^2 \,. \qquad (51)$$

Note that this loss function provides a weighted least-squares solution to the *lineararized inverse problem*

$$\Delta y_k \approx H(\hat{x}_k)\Delta x_k \,. \qquad (52)$$

Recalling that $h(x)$ is locally one-to-one if and only $H(x)$ is one-to-one, the loss-function approximation (51) can be minimized with respect to Δx_k yielding the unique correction

$$\begin{aligned} \Delta x_k &= \left(H^T(\hat{x}_k)WH(\hat{x}_k)\right)^{-1} H^T(\hat{x}_k)W\Delta y_k \\ &= \left(H^T(\hat{x}_k)WH(\hat{x}_k)\right)^{-1} H^T(\hat{x}_k)W\left(y - h(\hat{x}_k)\right) \\ &= -\left(H^T(\hat{x}_k)WH(\hat{x}_k)\right)^{-1} \nabla_x \ell(\hat{x}_k)\,, \end{aligned} \qquad (53)$$

where the last step follows from equations (36) and (38). The Hessian matrix $\frac{\partial^2}{\partial(\Delta x_k)^2}\hat{\ell}^{\text{Gauss}}(\Delta x_k)$ of the loss-function approximation (51) is equal to $H^T(\hat{x}_k)WH(\hat{x}_k)$.

As expected, the correction (53) is equivalent to

$$\Delta x_k = H^+(\hat{x}_k)\,\Delta y_k \qquad (54)$$

and provides the solution to the linearized inverse problem (52), where

$$H^+(\hat{x}_k) = \left(H^*(\hat{x}_k)H(\hat{x}_k)\right)^{-1} H^*(\hat{x}_k) = \left(H^T(\hat{x}_k)WH(\hat{x}_k)\right)^{-1} H^T(\hat{x}_k)W$$

is the pseudoinverse of $H(\hat{x}_k)$ with respect to the W-weighted inner-product on $\mathcal{Y} = \mathbb{R}^m$.[58]

Once we have determined the correction (53)-(54), we can obtain an improved estimate of the optimal solution as

$$\begin{aligned} \hat{x}_{k+1} &= \hat{x}_k + \Delta x_k = \hat{x}_k + H(\hat{x}_k)^+\Delta y_k && (55) \\ &= \hat{x}_k + \left(H^T(\hat{x}_k)WH(\hat{x}_k)\right)^{-1} H^T(\hat{x}_k)W\left(y - h(\hat{x}_k)\right)\,. && (56) \end{aligned}$$

Note that if $\hat{x}_k$ is already an optimum solution, then $\nabla_x \ell(\hat{x}_k) = 0$ yielding $\Delta x_k = 0$ as a consequence of equations (36) and (38). Equation (56) provides an iterative method for generating estimates of an unknown optimum solution x_0.

In practice, the iterative algorithm (55) is usually stabilized by including a positive, real-valued step-size parameter α_k

$$\Delta x_k \to \alpha_k \Delta x_k = \alpha_k\, H^+(\hat{x}_k)\Delta y_k = \alpha_k\, \left(H^T(\hat{x}_k)WH(\hat{x}_k)\right)^{-1} H^T(\hat{x}_k)W\Delta y_k\,, \qquad (57)$$

[58]Note (because of the full column-rank assumption on the Jacobian $H(x)$) that the same result holds even if $\Omega_x \neq I$. This shows that the Gauss-Newton method does not have to be modified in the case of a non-Cartesian domain space. However, consistent with his comments regarding the Newton method, Amari (*op. cit.* footnote (35)) argues that this still does not mean that the correction is necessarily along a good local direction in the space $\mathcal{X}$.

yielding the

Gauss-Newton Algorithm:

$$\boxed{\hat{x}_{k+1} = \hat{x}_k + \Delta x_k = \hat{x}_k + \alpha_k \, H(\hat{x}_k)^+ \Delta y_k} \tag{58}$$

or, equivalently,

$$\boxed{\hat{x}_{k+1} = \hat{x}_k + \alpha_k \, \left(H^T(\hat{x}_k) W H(\hat{x}_k)\right)^{-1} H^T(\hat{x}_k) W \left(y - h(\hat{x}_k)\right)} \tag{59}$$

Often the step size is taken to be constant, $\alpha_k = \alpha \leq 1$ with the value $\alpha = 1$ called the *Gauss-Newton step* as this choice retrieves the pure Gauss-Newton algorithm (56)

With an appropriate choice of the step-sizes α_k, the algorithm (59) can usually be stabilized[59] so that the iterative estimates converge to a locally optimal solution x_0,

$$\hat{x}_\infty \triangleq \lim_{k \to \infty} \hat{x}_k = x_0 \, .$$

However, the initial condition $\hat{x}_0$ used in (59) often results in different locally optimal solutions, so that a variety of initializations are usually tried. The resulting locally optimal solutions are then compared and the most optimal of them is kept as the final solution.[60]

A comparison of equations (44), (47) and (53) indicates that the Gauss-Newton method can be viewed as an approximation to the Newton method corresponding to the assumption that the second term on the right-hand-side of (44) evaluated at $\hat{x}_k$ is negligible,

$$\sum_{i=1}^{m} \mathcal{H}^{(i)}(x) \left[W \left(y - h(x)\right)\right]_i \approx 0 \tag{60}$$

so that

$$\mathcal{H}(\hat{x}_k) \approx H^T(\hat{x}_k) W H(\hat{x}_k) = H^*(\hat{x}_k) H(\hat{x}_k) \, . \tag{61}$$

Another way to see this is to expand the definition of $\hat{\ell}^{\text{Gauss}}(\Delta x_k)$ given on the right-hand-side of (51) and then compare the result to the definition of $\hat{\ell}^{\text{Newton}}(\Delta x_k)$ given in (45).

Consideration of the condition (60) allows us to conclude that if $h(\cdot)$ is onto, or if we have other reasons to believe that the approximation error $e(\hat{x}_k) = y - h(\hat{x}_k)$ can be made small, then the difference between the Newton and Gauss-Newton algorithms will become negligible as $e(\hat{x}_k) \to 0$. Not surprisingly then, the simpler Gauss-Newton method is more often implemented than the Newton method. However the Gauss-Newton method still requires a matrix inverse at each iteration step, which can be computationally prohibitive. As discussed below, an even simpler algorithm is provided by the gradient descent algorithm.[61]

[59]Quite often the Gauss-Newton step $\alpha = 1$ will suffice.

[60]As with the Newton method, the question of if and when the true globally optimal solution has been found is generally a difficult one to answer.

[61]However the increase in simplicity usually comes at the price of a significant degradation in the rate of convergence.

Generalized Gradient Descent. Consider the nonlinear least-squares loss function (34). We have that

$$d\ell = \frac{\partial \ell(x)}{\partial x} dx \,,$$

so that if we linearize about a current estimate $\hat{x}_k$ we can claim that

$$\Delta\ell(\alpha_k \, \Delta x_k) \triangleq \ell(\underbrace{\hat{x}_k + \alpha_k \, \Delta x_k}_{\hat{x}_{k+1}}) - \ell(\hat{x}_k) \approx \alpha_k \frac{\partial \ell(\hat{x}_k)}{\partial x} \Delta x_k = \alpha_k \left(\nabla_x \ell(\hat{x}_k)\right)^T \Delta x_k \qquad (62)$$

to a high degree of accuracy *provided that* $\alpha_k \, \Delta x_k = x - \hat{x}_k$ *is "small enough."* It is the requirement that this approximation be valid which leads us to introduce the step-size parameter α_k. The step-size α_k is chosen to have a small value *precisely* in order to keep the correction $\alpha_k \, \Delta x_k$ "small enough" to preserve the validity of equation (62), and (as we shall see below) this is done in order to guarantee stability and convergence of the resulting algorithm.

Assuming the validity of equation (62), let the correction be given by

$$\alpha_k \, \Delta x_k = -\alpha_k \, Q(\hat{x}_k) \, \nabla_x \ell(\hat{x}_k) \; = \alpha_k \, Q(\hat{x}_k) \, H^T(\hat{x}_k) W \left(y - h(\hat{x}_k)\right), \qquad (63)$$

where $\alpha_k > 0$ and $Q(x) = Q^T(x) > 0$ is an arbitrary positive-definite, symmetric matrix-valued function of x.[62] We call the term $Q(\hat{x}_k) \, \nabla_x \ell(\hat{x}_k)$ a *generalized gradient* and the resulting correction (63) a *generalized gradient-descent correction.*

Recall that the gradient $\nabla_x \ell(\hat{x}_k)$ defines the direction of *steepest ascent* of the function $\ell(x)$ at the point $x = \hat{x}_k$ and the negative gradient, $-\nabla_x \ell(\hat{x}_k)$, gives the direction of *steepest descent* at the point $\hat{x}_k$. For the case $Q(x) = I$, the correction is directly along the direction of steepest descent and the resulting algorithm is known as a *gradient descent algorithm.*[63] For the case of a general positive-definite matrix-valued function $Q(x) \neq I$, we can potentially improve the descent direction by using the negative of the *generalized* gradient as a descent direction, and the resulting algorithm for an arbitrary $Q(x)$ is called a *generalized gradient descent algorithm.* Thus the generalized gradient descent algorithms include the standard gradient descent algorithm as a special case.

With the correction (63) we obtain the

Generalized Gradient Descent Algorithm:

$$\boxed{\hat{x}_{k+1} = \hat{x}_k + \alpha_k \, \Delta x_k = \hat{x}_k + \alpha_k \, Q(\hat{x}_k) \, H^T(\hat{x}_k) W \left(y - h(\hat{x}_k)\right)} \qquad (64)$$

Important special cases:

$$\begin{aligned}
Q(x) &= I && \textbf{Gradient Descent Method} && (65)\\
Q(x) &= \left(H(x)^T W H(x)\right)^{-1} && \textbf{Gauss-Newton Method} && (66)\\
Q(x) &= \mathcal{H}^{-1}(x) && \textbf{Newton Method} && (67)\\
Q(x) &= \Omega_x^{-1} && \textbf{Natural Gradient Method} && (68)
\end{aligned}$$

[62]Note that if the approximation (62) is not valid for this particular choice of Δx_k, we can reduce the size of α_k until it is.

[63]Recall that we are assuming that $\Omega_x = I$ so that the Cartesian gradient is the true gradient.

In particular:

$$\begin{aligned}
\textbf{Gradient Descent:} \quad \hat{x}_{k+1} &= \hat{x}_k + \Delta x_k = \hat{x}_k + \alpha_k\, H^*(\hat{x}_k)\ (y - h(\hat{x}_k)) && (69)\\
\textbf{Natural Gradient:} \quad \hat{x}_{k+1} &= \hat{x}_k + \Delta x_k = \hat{x}_k + \alpha_k\, \widetilde{H}^*(\hat{x}_k)\ (y - h(\hat{x}_k)) && (70)\\
\textbf{Gauss-Newton:} \quad \hat{x}_{k+1} &= \hat{x}_k + \Delta x_k = \hat{x}_k + \alpha_k\, H^+(\hat{x}_k)\, (y - h(\hat{x}_k)) && (71)\\
&= \hat{x}_k + \Delta x_k = \hat{x}_k + \alpha_k\, \widetilde{H}^+(\hat{x}_k)\, (y - h(\hat{x}_k)) && (72)
\end{aligned}$$

where

$$\widetilde{H}^*(x) = \Omega_x^{-1} H^*(x) = \Omega_x^{-1} H^T(x) W$$

is the “natural” adjoint operator of the jacobian $H(x)$ and

$$\widetilde{H}^+(x) = \left(\widetilde{H}^*(x) H(x)\right)^{-1} \widetilde{H}^*(x) = \left(H^*(x) H(x)\right)^{-1} H^*(x) = H^+(x)$$

is the “natural” pseudoinverse of the jacobian $H(x)$, assuming metric tensors on the domain and codomain of $\Omega_x \neq I$ and W respectively.[64]

The increase in complexity involved in moving from a gradient descent algorithm to a Gauss-Newton algorithm (or a Newton algorithm) is quite significant, however this move also generally results in a significant improvement in performance. Although the classical Gradient Descent method is the easiest of the methods to implement, it tends to have very slow convergence compared to the Gauss-Newton and Newton methods.

The Natural Gradient Algorithm. The Natural Gradient algorithm is equivalent to gradient descent using the natural gradient in order to obtain a correction along the *true* direction of steepest descent as determined from the metric tensor appropriate for the space $\mathcal{X}$. Of course, to implement the Natural Gradient method requires the actual determination of an appropriate metric tensor Ω_x on $\mathcal{X}$.

Shun-ichi Amari and his coworkers first proposed the use of the Natural Gradient method and they have determined metric tensors for a variety of important statistical learning theory applications, including neural network learning, parametric density estimation, and blind source separation.[65] As they have noted, it can be very difficult to construct the metric tensor for a parameter space of interest and the resulting algorithms can be significantly computationally demanding. However, there are applications where the implementation of the Natural Gradient method is surprisingly straightforward and even computationally simple.[66]

[64]Note that the Newton and Gauss-Newton algorithms are “natural” in the sense that their derivations are independent of the value of Ω_x. However, Amari (op. cit footnote (35)) argues that this does not mean that they necessarily give good correction directions as compared to the “natural gradient” algorithm. Furthermore to the degree that the (naive or Cartesian) gradient descent algorithm purports to perform a correction along the direction of steepest descent, it is evident that it fails in this goal to the degree that the Cartesian gradient used naively is not along the correct direction of steepest descent, which is given by the natural (true) gradient.

[65]Shun-ichi Amari, *op. cit.* footnote (35).

[66]For the classical problem of parametric density estimation by finding the maximum likelihood estimate of the parameter, one often obtains the parameter estimate via gradient descent on the negative log-likelihood function. It can be shown that the metric tensor for the space of density parameters is the classical Fisher information metric.

Both theoretical and experimental results exist which show that the Natural Gradient method can have excellent asymptotic learning properties and show significant improvement over naive gradient descent.

Importance of the Step-size Parameter for Ensuring Algorithm Stability. From equations (63) and (64), and *assuming the validity of* (62), we obtain

$$\Delta\ell(\alpha_k \Delta x_k) = \ell(\hat{x}_{k+1}) - \ell(\hat{x}_k) \approx -\alpha_k \left(\nabla_x \ell(\hat{x}_k)\right)^T Q(\hat{x}_k) \nabla_x \ell(\hat{x}_k) = -\alpha_k \|\nabla_x \ell(\hat{x}_k)\|^2_{Q(\hat{x}_k)} < 0\,,$$

whenever $\nabla_x \ell(\hat{x}_k) \neq 0$,[67], so that we expect that the algorithm (64) will result in a strictly decreasing loss function,

$$\ell(\hat{x}_{k+1}) < \ell(\hat{x}_k)\,.$$

This will ***not*** *be the case if the step-size α_k is chosen too large so that the key approximation (62) is invalid.* Furthermore, a proper choice of step-size can greatly improve the speed of convergence of the algorithm. Thus we see that a proper choice of step-size is key to ensuring good performance and, indeed, the step-size issue is usually discussed at great length in course on mathematical optimization.[68]

A Naive Method for Step-Size Determination. The Generalized Gradient Descent algorithm is of the form

$$\widehat{x}_{k+1} = \widehat{x}_k + \alpha_k\, \Delta x_k$$

where

$$\alpha_k\, \Delta x_k = -\alpha_k\, Q(\widehat{x}_k)\, \nabla_x \ell(\widehat{x}_k)$$

corresponds to a step in the opposite direction to the generalized gradient $Q(\widehat{x}_k)\, \nabla_x \ell(\widehat{x}_k)$[69] and the size of this step is controlled by the value of the step-size parameter $\alpha_k > 0$.

As mentioned, the choice of step size can be very critical for ensuring convergence of a generalized gradient descent algorithm and is a major topic of concern in advanced textbooks on optimization theory. If the simple choice of a constant value of α_k is used, generally the smaller the value of α_k the more likely it is that the algorithm will converge but also that the rate of convergence will be slow. Fortunately, the Newton and Gauss-Newton methods tend to be quite stable and the choice of $\alpha_k = 1$ or α_k equal to a constant value slightly less than one often works well in practice.

More generally, the step size choice can be chosen dynamically.[70] Note that we need $\alpha_k \to 0$ more slowly than the generalized gradient goes to zero in order to avoid turning off the update step

[67]If $\nabla_x \ell(\hat{x}_k) = 0$ then $\hat{x}_k$ satisfies the necessary condition for a local optimum and the algorithm has converge to a solution.

[68]D. Luenberger *op. cit.*

[69]The fact that $Q(\widehat{x}_k)$ is positive definite ensures that the angle between the gradient $\nabla_x \ell(\widehat{x}_k)$ and the generalized gradient $Q(\widehat{x}_k)\, \nabla_x \ell(\widehat{x}_k)$ is less than 90^o so that a movement along the generalized gradient direction always has a component along the gradient direction.

[70]And even optimized–see D. Luenberger *op. cit.*

before we have learned the unknown parameter vector x. In principle we require that $\alpha_\infty > 0$ but practically we can have $\alpha_\infty = 0$ provided that the generalized gradient converges to the zero vector before the step-size parameter has converged to zero. A simple (but very naive) dynamic choice for α_k is given by

$$\alpha_k = \gamma + \alpha_0 \beta^k \,, \quad k = 0, 1, 2, \cdots$$

for $\alpha_0 > 0$, $0 < \beta < 1$, and $0 < \gamma \ll 1$. More sophisticated ways to dynamically adjust the step-size are discussed in the textbooks on optimization theory.[71]

A simple way to enforce convergence[72] is to choose a step-size that guarantees that the loss function $\ell(\widehat{x}_k)$ is decreased at each iteration time-step k as follows:

```
Begin

Choose values for $\alpha_0 > 0$ and $1 > \beta > 0$

Set $\ell_k = \ell(\widehat{x}_k)$ and $j = 0$

    Loop Until  $\ell_{k+1} < \ell_k$
        $\alpha = \alpha_0 \, \beta^j$

        $\widehat{x}_{k+1} = \widehat{x}_k - \alpha \, Q(\widehat{x}_k) \, \nabla_x \ell(\widehat{x}_k)$

        $\ell_{k+1} = \ell(\widehat{x}_{k+1})$

        $j \leftarrow j + 1$
    End Loop

End
```

This will ensure that $\ell(\widehat{x}_{k+1}) < \ell(\widehat{x}_k)$ but at the potential cost of several expensive update and loss function evaluations at each iteration step k of the generalized gradient descent algorithm.

2.4 Constrained Optimization and Lagrange Multipliers

Method of Lagrange Multipliers. Assume, for simplicity, that both the domain and codomain are Cartesian, $\Omega = I$, $W = I$. If there are p independent equality constraint conditions,[73] they can be represented as

$$g(x) \equiv 0 \,, \tag{73}$$

[71] D. Luenberger *op. cit.*

[72] But perhaps at the expense of the speed of convergence.

[73] The p constraints $g(x) = 0$ are linearly independent at the point x if and only if the Jacobian matrix of $g(x)$, $\frac{\partial g(x)}{\partial x}$, has full row-rank at the point x.

where $g(\cdot) : \mathbb{R}^n \to \mathbb{R}^p$. The constraint condition (73) defines a locally p-dimensional smooth surface in $\mathcal{X} = \mathbb{R}^n$ which we call the *constraint manifold.*

Because the constraint condition (73) holds identically, the differential variations dx *cannot* be freely chosen in an arbitrary manner (as was done for the derivation of the unconstrained stationarity condition (5)) and it must be the case that admissible differential variations satisfy the constraint condition,

$$\frac{\partial g(x)}{\partial x}\, dx = 0\,.$$

Thus, admissible variations dx must be in the nullspace of $\frac{\partial g(x)}{\partial x}$, $dx \in \mathcal{N}\left(\frac{\partial g(x)}{\partial x}\right)$, which is just the condition that dx lies in the tangent space of the constraint manifold at the point x.

Let $\ell(x)$ be a general, arbitrary loss function.[74] A necessary condition to have minimized the loss function $\ell(x)$ on the constraint manifold is that the projection of its gradient $\nabla_x \ell(x)$ into the nullspace of $\frac{\partial g(x)}{\partial x}$ is zero. Equivalently, we require that $\nabla_x \ell(x) \perp \mathcal{N}(\frac{\partial g(x)}{\partial x})$. Thus

$$\nabla_x \ell(x) \in \mathcal{N}\left(\frac{\partial g(x)}{\partial x}\right)^{\perp} = \mathcal{R}\left(\frac{\partial g(x)}{\partial x}^T\right)\,,$$

and therefore

$$\left(\frac{\partial}{\partial x}\ell(x)\right)^T = \nabla_x \ell(x) = -\frac{\partial g(x)}{\partial x}^T \lambda \tag{74}$$

for some vector $\lambda \in \mathbb{R}^p$.[75] Equation (74) is a necessary condition for the point x to minimize $\ell(x)$ on the constraint manifold.

The necessary optimality condition (74) can be rewritten as the equivalent necessary condition,

$$0 = \frac{\partial}{\partial x}\left(\ell(x) + \lambda^T g(x)\right) = \frac{\partial}{\partial x}\mathcal{L}(x;\lambda)\,, \tag{75}$$

where

$$\mathcal{L}(x;\lambda) = \ell(x) + \lambda^T g(x) \tag{76}$$

is the *lagrangian function.* Notice that the stationarity condition

$$\frac{\partial}{\partial \lambda}\,\mathcal{L}(x;\lambda) = 0 \tag{77}$$

retrieves the equality constraint condition (73), while the stationarity condition

$$\frac{\partial}{\partial x}\,\mathcal{L}(x;\lambda) = 0 \tag{78}$$

retrieves the necessary condition (74).

The two necessary conditions (73) and (74) are together to be solved for the optimal point x.

Thus, the lagrangian (76) provides a complete encoding of the information needed to solve for a solution to the necessary conditions (73) and (74).

[74]*Not* limited only to the least squares loss function described earlier.

[75]The elements of the vector λ are called *lagrange multipliers.*

Linearly Constrained Quadratic Optimization. As an example, consider the linearly constrained quadratic optimization problem

$$\min_x \frac{1}{2}\|x\|_\Omega^2 \quad \text{subject to } Ax = y$$

where the $m \times n$ matrix A is onto. The constraint condition can be written as

$$g(x) = y - Ax = 0\,,$$

and the lagrangian as

$$\mathcal{L} = \frac{1}{2}\|x\|_\Omega^2 + \lambda^T\left(y - Ax\right)\,.$$

We have that

$$0 = \frac{\partial}{\partial x}\mathcal{L} = x^T\Omega - \lambda^T A\,,$$

which can be solved to yield the condition[76]

$$\hat{x} = \Omega^{-1}A^T\lambda\,.$$

Applying the condition $Ax = y$ to this equation allows us to solve for the lagrange multipliers as

$$\lambda = \left(A\Omega^{-1}A^T\right)^{-1} y$$

so that finally

$$\hat{x} = \Omega^{-1}A^T\left(A\Omega^{-1}A^T\right)^{-1} y = A^*(AA^*)^{-1}y = A^+y\,,$$

which is equivalent to the solution obtained using geometric methods.

[76]Note that this condition is equivalent to the geometric condition that $\hat{x} = A^*\lambda$, i.e. the condition that $\hat{x} \in \mathcal{R}(A^*)$.