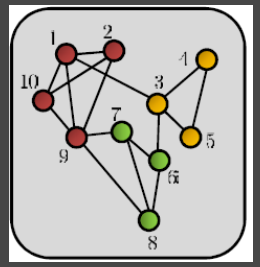


INFERENCE OVER NETWORKS

LECTURE #9: Optimization by Single Agents

Professor Ali H. Sayed
UCLA Electrical Engineering

Course EE210B
Spring Quarter 2015

Proc. IEEE, vol. 102, no. 4, pp. 460-497, April 2014.
Foundations and Trends in Machine Learning, vol. 7, no. 4-5, pp. 311-801, July 2014.

Part II: Single-Agent Adaptation And Learning

Course EE210B
Spring Quarter 2015

Proc. IEEE, vol. 102, no. 4, pp. 460-497, April 2014.
Foundations and Trends in Machine Learning, vol. 7, no. 4-5, pp. 311-801, July 2014.

Reference

Chapter 2 (Optimization by Single Agents, pp. 319-337):

A. H. Sayed, ``Adaptation, learning, and optimization over networks," ***Foundations and Trends in Machine Learning***, vol. 7, issue 4-5, pp. 311-801, NOW Publishers, 2014.

Setting

In this chapter we review the class of gradient-descent algorithms, which are among the most successful iterative techniques for the solution of optimization problems by stand-alone single agents. The presentation summarizes some classical results and provides insights that are useful for our later study of the more demanding scenario of optimization by networked agents. We consider initially the case of real-valued arguments [207] and extend the results to the complex domain as well. We also consider both cases of constant step-sizes and decaying step-sizes.

Risk and Loss Functions

Course EE210B
Spring Quarter 2015

Proc. IEEE, vol. 102, no. 4, pp. 460-497, April 2014.
Foundations and Trends in Machine Learning, vol. 7, no. 4-5, pp. 311-801, July 2014.

Risk and Loss Functions

Thus, let $J(w) \in \mathbb{R}$ denote a real-valued (cost or utility or risk) function of a real-valued vector argument, $w \in \mathbb{R}^M$. It is common in adaptation and learning applications for $J(w)$ to be constructed as the expectation of some loss function, $Q(w; \boldsymbol{x})$, where the ***boldface*** variable $\boldsymbol{x}$ is used to denote some random data, say,

$$J(w) = \mathbb{E}\, Q(w; \boldsymbol{x}) \tag{2.1}$$

and the expectation is evaluated over the distribution of $\boldsymbol{x}$ [208].

Risk and Loss Functions

Following the notation introduced in Appendices A and B, we denote the gradient vectors of $J(w)$ relative to w and w^{T} by the following row and column vectors, respectively, where the first expression is also referred to as the Jacobian of $J(w)$ relative to w:

$$\nabla_w J(w) \triangleq \left[\frac{\partial J(w)}{\partial w_1} \quad \frac{\partial J(w)}{\partial w_2} \quad \ldots \quad \frac{\partial J(w)}{\partial w_M} \right] \tag{2.2}$$

$$\nabla_{w^{\mathsf{T}}} J(w) \triangleq [\nabla_w J(w)]^{\mathsf{T}} \tag{2.3}$$

Risk and Loss Functions

These definitions are in terms of the partial derivatives of $J(w)$ relative to the individual entries of w:

$$w \triangleq \text{col}\{w_1, w_2, \ldots, w_M\} \tag{2.4}$$

Likewise, the Hessian matrix of $J(w)$ with respect to w is defined as the following $M \times M$ symmetric matrix:

$$\nabla_w^2 J(w) \triangleq \nabla_{w^{\mathsf{T}}}[\nabla_w J(w)] = \nabla_w[\nabla_{w^{\mathsf{T}}} J(w)] \tag{2.5}$$

which is constructed from two successive gradient operations.

Example #2.1

Example 2.1 (Mean-square-error costs). Let $\boldsymbol{d}$ denote a zero-mean scalar random variable with variance $\sigma_d^2 = \mathbb{E}\,\boldsymbol{d}^2$ and let $\boldsymbol{u}$ denote a zero-mean $1 \times M$ random vector with covariance matrix $R_u = \mathbb{E}\,\boldsymbol{u}^{\mathsf{T}}\boldsymbol{u} > 0$. The combined quantities $\{\boldsymbol{d}, \boldsymbol{u}\}$ represent the random variable $\boldsymbol{x}$ referred to in (2.1). The cross-covariance vector is denoted by $r_{du} = \mathbb{E}\,\boldsymbol{d}\boldsymbol{u}^{\mathsf{T}}$. We formulate the problem of estimating $\boldsymbol{d}$ from $\boldsymbol{u}$ in the linear least-mean-squares sense or, equivalently, the problem of seeking the vector w^o that minimizes the quadratic cost function:

$$J(w) \triangleq \mathbb{E}\,(\boldsymbol{d} - \boldsymbol{u}w)^2 = \sigma_d^2 - 2r_{du}^{\mathsf{T}}w + w^{\mathsf{T}}R_u w \tag{2.6}$$

Example #2.1

This cost corresponds to the following choice for the loss function:

$$Q(w;\boldsymbol{x}) \triangleq (\boldsymbol{d}-\boldsymbol{u}w)^2 = \boldsymbol{d}^2 - 2\boldsymbol{d}\boldsymbol{u}w + w^{\mathsf{T}}\boldsymbol{u}^{\mathsf{T}}\boldsymbol{u}w \tag{2.7}$$

Such quadratic costs are widely used in estimation and adaptation problems [107, 133, 206, 207, 263]. They are also widely used as quadratic risk functions in machine learning applications [37, 234]. The gradient vector and Hessian matrix of $J(w)$ are easily seen to be:

$$\nabla_w J(w) = 2\left(R_u w - r_{du}\right)^{\mathsf{T}}, \qquad \nabla_w^2 J(w) = 2R_u \tag{2.8}$$

Example #2.1

Figure 2.1 illustrates the mean-square-error cost (2.6) for the two-dimensional case, $M = 2$. The individual entries of $w \in \mathbb{R}^M$ are denoted by $w = \text{col}\{w_1, w_2\}$. The plot is generated by using $\sigma_d^2 = 0.5$, a diagonal covariance matrix, R_u, whose entries are generated randomly from within the interval $[1, 10]$, and a cross-covariance vector, r_{du}, whose entries are also generated randomly within the range $[0, 1]$.

Example #2.1

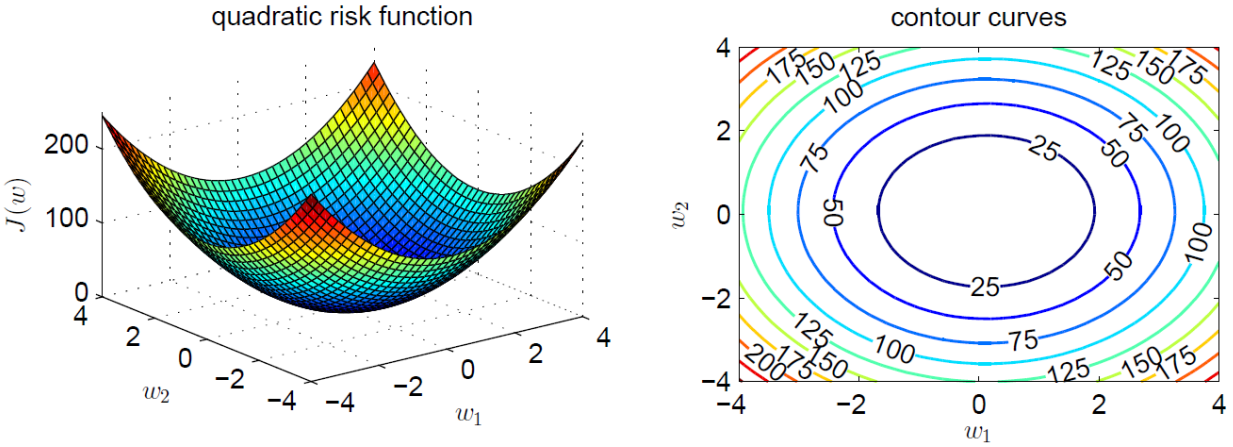


Figure 2.1: Illustration of the mean-square-error cost (2.6) for the two-dimensional case, $M = 2$ (left), along with the corresponding contour curves (right). The plots are generated by using $\sigma_d^2 = 0.5$, and randomly-generated diagonal covariance matrix, R_u, and cross-covariance vector r_{du}.

■

Example #2.2

Example 2.2 (Logistic or log-loss risks). Let $\boldsymbol{\gamma}$ denote a binary random variable that assumes the values ± 1, and let $\boldsymbol{h}$ denote an $M \times 1$ random (feature) vector with $R_h = \mathbb{E}\,\boldsymbol{h}\boldsymbol{h}^{\mathsf{T}}$. The combined quantities $\{\boldsymbol{\gamma}, \boldsymbol{h}\}$ represent the random variable $\boldsymbol{x}$ referred to in (2.1). In the context of machine learning and pattern classification problems [37, 115, 234], the variable $\boldsymbol{\gamma}$ designates the class that feature vector $\boldsymbol{h}$ belongs to. In these problems, one seeks the vector w^o that minimizes the regularized logistic risk function — see Appendix G:

$$J(w) \triangleq \frac{\rho}{2}\|w\|^2 + \mathbb{E}\left\{\ln\left(1 + e^{-\boldsymbol{\gamma}\boldsymbol{h}^{\mathsf{T}}w}\right)\right\} \tag{2.9}$$

Example #2.2

where $\rho > 0$ is some regularization parameter, $\ln(\cdot)$ is the natural logarithm function, and $\|w\|^2 = w^{\mathsf{T}} w$. The risk (2.9) corresponds to the following choice for the loss function:

$$Q(w; \boldsymbol{x}) \triangleq \frac{\rho}{2}\|w\|^2 + \ln\left(1 + e^{-\boldsymbol{\gamma} \boldsymbol{h}^{\mathsf{T}} w}\right) \tag{2.10}$$

Once w^o is recovered, its value can be used to classify new feature vectors, say, $\{\boldsymbol{h}_\ell\}$, into classes $+1$ or -1. This can be achieved, for example, by assigning feature vectors with $\boldsymbol{h}_\ell^{\mathsf{T}} w^o \geq 0$ to one class and feature vectors with $\boldsymbol{h}_\ell^{\mathsf{T}} w^o < 0$ to another class.

Example #2.2

Assuming the distribution of $\{\boldsymbol{\gamma}, \boldsymbol{h}\}$ is such that it permits the exchange of the expectation and differentiation operations, it can be verified that for the above $J(w)$:

$$\nabla_w J(w) = \rho w^{\mathsf{T}} - \mathbb{E}\left\{\boldsymbol{\gamma}\boldsymbol{h}^{\mathsf{T}}\left(\frac{e^{-\boldsymbol{\gamma}\boldsymbol{h}^{\mathsf{T}}w}}{1+e^{-\boldsymbol{\gamma}\boldsymbol{h}^{\mathsf{T}}w}}\right)\right\} \tag{2.11}$$

$$\nabla_w^2 J(w) = \rho I_M + \mathbb{E}\left\{\boldsymbol{h}\boldsymbol{h}^{\mathsf{T}}\left(\frac{e^{-\boldsymbol{\gamma}\boldsymbol{h}^{\mathsf{T}}w}}{\left(1+e^{-\boldsymbol{\gamma}\boldsymbol{h}^{\mathsf{T}}w}\right)^2}\right)\right\} \tag{2.12}$$

Example #2.2

Figure 2.2 illustrates the logistic risk function (2.9) for the two-dimensional case, $M = 2$, and using $\rho = 10$. The individual entries of $w \in \mathbb{R}^2$ are denoted by $w = \text{col}\{w_1, w_2\}$. The plot is generated by approximating the expectation in (2.9) by means of a sample average over 100 repeated realizations for the random variables $\{\boldsymbol{\gamma}, \boldsymbol{h}\}$. Specifically, a total of 100 binary realizations are generated for $\boldsymbol{\gamma}$, where the values ± 1 are assumed with equal probability, and 100 Gaussian realizations are generated for $\boldsymbol{h}$ with mean vectors $+\mathbb{1}$ and $-\mathbb{1}$ for the classes $\boldsymbol{\gamma} = +1$ and $\boldsymbol{\gamma} = -1$, respectively.

■

Example #2.2

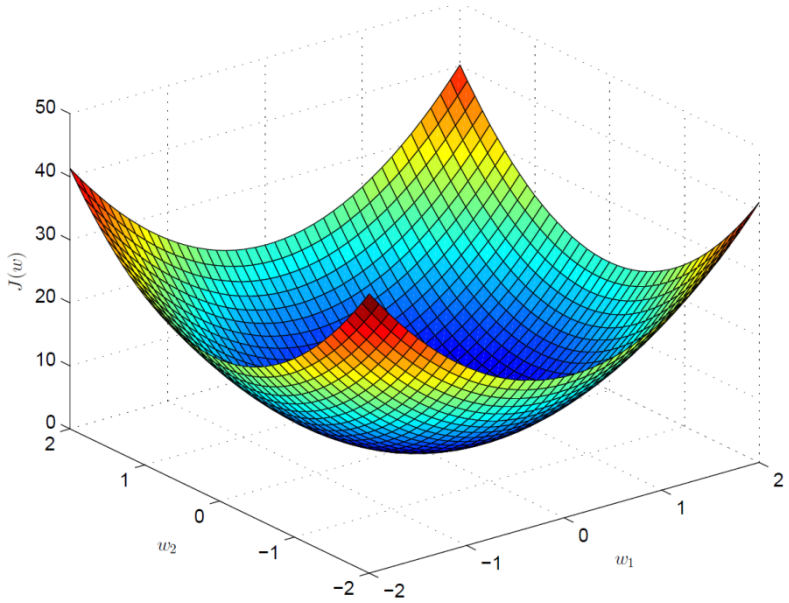


Figure 2.2: Illustration of the logistic risk (2.9) for $M = 2$ and $\rho = 10$. The plot is generated by approximating the expectation in (2.9) by the sample average over 100 repeated realizations for the random variables $\{\boldsymbol{\gamma}, \boldsymbol{h}\}$.

Recall#1: Big and Little-O Notation

$a(i) = O(b(i))$ means $|a(i)| \leq c|b(i)|$ for some constant c and large i. Example:

$$a(i) = O(1/i) \Longrightarrow a(i) \text{ decays asymptotically at a rate comparable to } 1/i$$

$a(i) = o(b(i))$ means that asymptotically the sequence $a(i)$ decays faster than $b(i)$, or $|a(i)|/|b(i)| \to 0$ as $i \to \infty$. Example:

$$a(i) = o(1/i) \Longrightarrow a(i) \text{ decays asymptotically at a faster rate than } 1/i$$

$$\begin{cases} a = O(\mu) \Longrightarrow |a| \text{ is in the order of } \mu \\ a = o(\mu) \Longrightarrow |a| \text{ is some higher power in } \mu \end{cases}$$

Conditions on Risk Function

Course EE210B
Spring Quarter 2015

Proc. IEEE, vol. 102, no. 4, pp. 460-497, April 2014.
Foundations and Trends in Machine Learning, vol. 7, no. 4-5, pp. 311-801, July 2014.

Conditions on Risk Function

Stochastic gradient algorithms are powerful iterative procedures for solving optimization problems of the form

$$w^o = \arg\min_{w} \; J(w) \tag{2.13}$$

Strong Convexity

While the analysis that follows can be pursued under more relaxed conditions (see, e.g., the treatments in [32, 191, 192, 244]), it is sufficient for our purposes to require $J(w)$ to be strongly-convex and twice-differentiable with respect to w. Recall from property (C.18) in the appendix that the cost function $J(w)$ is said to be $\nu-$strongly convex if, and only if, its Hessian matrix is sufficiently bounded away from zero [29, 45, 178, 191]:

$$J(w) \text{ is } \nu-\text{strongly convex} \iff \nabla_w^2 J(w) \geq \nu I_M > 0 \qquad (2.14)$$

for all w and for some scalar $\nu > 0$.

Strong Convexity

Strong convexity is a useful condition in the context of adaptation and learning from streaming data because it helps guard against ill-conditioning in the algorithms; it also helps ensure that $J(w)$ has a *unique* global minimum, say, at location w^o; there will be no other minima, maxima, or saddle points. In addition, as we are going to see later in (2.23), it is well-known that strong convexity endows gradient-descent algorithms with geometric (i.e., exponential) convergence rates in the order of $O(\alpha^i)$, for some $0 \leq \alpha < 1$ and where i is the iteration index [32, 191].

Convexity

For comparison purposes, when the function $J(w)$ is only convex but not necessarily strongly convex, then from the same property (C.18) we know that convexity is equivalent to the following condition:

$$J(w) \text{ is convex} \iff \nabla_w^2 J(w) \geq 0 \tag{2.15}$$

for all w. In this case, while the function $J(w)$ will only have global minima, there can now be multiple global minima. Moreover, the convergence of the gradient-descent algorithm will now occur at the slower rate of $O(1/i)$ [32, 191].

Regularization

In most problems of interest in adaptation and learning, the cost function $J(w)$ is either already strongly convex or can be made strongly convex by means of regularization. For example, it is common in machine learning problems [37, 234] and in adaptation and estimation problems [133, 207] to incorporate regularization factors into the cost functions; these factors help ensure strong convexity automatically. For instance, the mean-square-error cost (2.6) is strongly convex whenever $R_u > 0$. If R_u happens to be singular, then the following regularized cost will be strongly convex:

$$J(w) \;\triangleq\; \frac{\rho}{2}\|w\|^2 \;+\; \mathbb{E}\,(\boldsymbol{d}-\boldsymbol{u}w)^2 \qquad (2.16)$$

Lipschitz Gradient

Besides strong convexity, we also require the gradient vector of $J(w)$ to be $\delta-$Lipschitz, namely, that there exists $\delta > 0$ such that

$$\|\nabla_w J(w_2) - \nabla_w J(w_1)\| \leq \delta \|w_2 - w_1\| \tag{2.17}$$

for all w_1, w_2. It follows from Lemma E.3 in the appendix that for twice-differentiable costs, conditions (2.14) and (2.17) combined are equivalent to

$$0 < \nu I_M \leq \nabla_w^2 J(w) \leq \delta I_M \tag{2.18}$$

Conditions on Risk Function

For example, it is clear that the Hessian matrices in (2.8) and (2.12) satisfy this property since

$$2\lambda_{\min}(R_u)I_M \;\leq\; \nabla_w^2 J(w) \;\leq\; 2\lambda_{\max}(R_u)I_M \tag{2.19}$$

in the first case and

$$\rho I_M \;\leq\; \nabla_w^2 J(w) \;\leq\; (\rho + \lambda_{\max}(R_h))I_M \tag{2.20}$$

in the second case. In summary, we will be assuming the following conditions on the cost function.

Conditions on Risk Function

Assumption 2.1 (Conditions on cost function). The cost function $J(w)$ is twice-differentiable and satisfies (2.18) for some positive parameters $\nu \leq \delta$. Condition (2.18) is equivalent to requiring $J(w)$ to be $\nu-$strongly convex and for its gradient vector to be $\delta-$Lipschitz as in (2.14) and (2.17), respectively.

Summary

Assumptions (can be relaxed):

a) $J(w)$ twice-differentiable

b) $J(w)$ is $\nu-$strongly convex $\iff \nabla_w^2 J(w) \geq \nu I_M > 0$

c) $\nabla_w J(w)$ is $\delta-$Lipschitz $\iff \|\nabla_w J(w_2) - \nabla_w J(w_1)\| \leq \delta \|w_2 - w_1\|$

$$\iff \nabla_w^2 J(w) \leq \delta I_M$$

Example: conditions are satisfied by quadratic or logistic risks.

Single Unified Condition

The conditions can be combined into a **single** statement:

$$0 < \nu I_M \leq \nabla_w^2 J(w) \leq \delta I_M$$

Quadratic risks: $$2\lambda_{\min}(R_u) I_M \leq \nabla_w^2 J(w) \leq 2\lambda_{\max}(R_u) I_M$$

Logistic risks: $$\rho I_M \leq \nabla_w^2 J(w) \leq (\rho + \lambda_{\max}(R_h)) I_M$$

Benefits of Strong Convexity

We will devise iterative procedures for solving:

$$\min_{w} \; J(w)$$

Strong convexity → useful for adaptation & learning:

- Ensures a unique global minimum (denoted by w^o).
- Guards against ill-conditioning in the algorithms.
- Ensures geometric convergence rates, $O(\alpha^i)$, $0 \leq \alpha < 1$.
- Usually guaranteed by regularization.

Comparison with Convexity

In comparison, with convexity alone:

- Multiple global minima can be present.
- Ill-conditioning can occur.
- Convergence occurs at slower rate, $O(1/i)$.

$\epsilon = 10^{-6}$ (desired accuracy)

$O(1/\epsilon) \sim 10^6$ iterations ← (with convexity)

$O(\ln(1/\epsilon)) \sim 6$ iterations ← (with strong convexity)

Gradient-Descent Algorithms

Course EE210B
Spring Quarter 2015

Proc. IEEE, vol. 102, no. 4, pp. 460-497, April 2014.
Foundations and Trends in Machine Learning, vol. 7, no. 4-5, pp. 311-801, July 2014.

Gradient Descent

There are many techniques by which optimization problems of the form (2.13) can be solved. We focus in this work on the important class of gradient descent algorithms. These algorithms require knowledge of the actual gradient vector and take the following form:

$$w_i \;=\; w_{i-1} \;-\; \mu\,\nabla_{w^{\mathsf{T}}} J(w_{i-1}), \quad i \geq 0 \tag{2.21}$$

where $i \geq 0$ is an iteration index (usually time), and $\mu > 0$ is a *constant* step-size parameter. The following result establishes that the successive iterates $\{w_i\}$ converge exponentially fast towards w^o for any step-size smaller than the threshold specified by (2.22).

Geometric Convergence

Lemma 2.1 (Convergence with constant step-size: Real case). Assume the cost function, $J(w)$, satisfies Assumption 2.1. If the step-size μ is chosen to satisfy

$$0 < \mu < \frac{2\nu}{\delta^2} \tag{2.22}$$

then, it holds that for any initial condition, w_{-1}, the gradient descent algorithm (2.21) generates iterates $\{w_i\}$ that converge exponentially fast to the global minimizer, w^o, i.e., it holds that

$$\|\widetilde{w}_i\|^2 \leq \alpha \|\widetilde{w}_{i-1}\|^2 \tag{2.23}$$

Geometric Convergence

where the real scalar α satisfies $0 \leq \alpha < 1$ and is given by

$$\alpha = 1 - 2\mu\nu + \mu^2\delta^2 \tag{2.24}$$

and $\widetilde{w}_i = w^o - w_i$ denotes the error vector at iteration i.

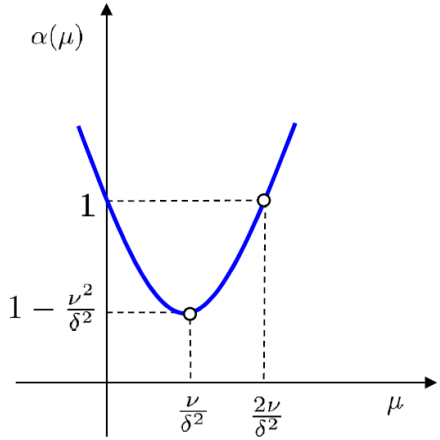
Gradient-Descent Algorithms

Traditional **gradient-descent** algorithm: $\min_{w} \; J(w)$

$$w_i \;=\; w_{i-1} \;-\; \mu\,\nabla_{w^{\mathsf{T}}} J(w_{i-1}), \quad i \geq 0$$

$$\widetilde{w}_i = w^o - w_i$$

Lemma 2.1: For small-enough step-sizes, the error converges exponentially as $\|\widetilde{w}_i\|^2 \;\leq\; \alpha\,\|\widetilde{w}_{i-1}\|^2$, where $\alpha \;=\; 1 - 2\mu\nu + \mu^2\delta^2$.



Recall#2: Convex Functions

Table C.1: Useful properties implied by the convexity, strict convexity, or strong convexity of a real-valued function $g(z) \in \mathbb{R}$ of a *real* argument $z \in \mathbb{R}^M$.

$g(z)$ convex $\Longrightarrow$	$[\nabla_z\, g(z_2) - \nabla_z\, g(z_1)]\,(z_2 - z_1) \geq 0$
$g(z)$ strictly convex $\Longrightarrow$	$[\nabla_z\, g(z_2) - \nabla_z\, g(z_1)]\,(z_2 - z_1) > 0$
$g(z)$ $\nu-$strongly convex $\Longrightarrow$	$[\nabla_z\, g(z_2) - \nabla_z\, g(z_1)]\,(z_2 - z_1) \geq \nu\|z_2 - z_1\|^2$

Recall#3: Mean-Value Theorems

Lemma D.1 (Mean-value theorem: Real arguments). Consider a real-valued and twice-differentiable function $g(z) \in \mathbb{R}$, where $z \in \mathbb{R}^M$ is real-valued. Then, for any $M-$dimensional vectors z_o and Δz, the following increment equalities hold:

$$g(z_o + \Delta z) - g(z_o) = \left(\int_0^1 \nabla_z \, g(z_o + t\,\Delta z) dt \right) \Delta z \qquad \text{(D.8)}$$

$$\nabla_z \, g(z_o + \Delta z) - \nabla_z \, g(z_o) = (\Delta z)^{\mathsf{T}} \left(\int_0^1 \nabla_z^2 \, g(z_o + r\,\Delta z) dr \right) \qquad \text{(D.9)}$$

1st Proof

Proof. We provide two arguments. The first derivation is perhaps more traditional, while the second derivation is based on arguments that are more convenient when we extend the results to optimization over networked agents. We start by subtracting w^o from both sides of (2.21) and use the fact that $\nabla_{w^{\mathsf{T}}} J(w^o) = 0$ to write

$$\widetilde{w}_i \;=\; \widetilde{w}_{i-1} \;+\; \mu\,[\nabla_{w^{\mathsf{T}}} J(w_{i-1}) - \nabla_{w^{\mathsf{T}}} J(w^o)] \tag{2.25}$$

Computing the squared Euclidean norms (or energies) of both sides of the above equality gives

1st Proof

$$
\begin{aligned}
\|\widetilde{w}_i\|^2 &= \|\widetilde{w}_{i-1}\|^2 + \mu^2 \|\nabla_{w^{\mathsf{T}}} J(w_{i-1}) - \nabla_{w^{\mathsf{T}}} J(w^o)\|^2 + \\
&\qquad 2\mu \left[\nabla_w J(w_{i-1}) - \nabla_w J(w^o)\right] \widetilde{w}_{i-1} \\
&\overset{(a)}{\leq} \|\widetilde{w}_{i-1}\|^2 + \mu^2 \left\| \left(\int_0^1 \nabla_w^2 J(w^o - t\widetilde{w}_{i-1}) dt \right) \widetilde{w}_{i-1} \right\|^2 - 2\mu\nu \|\widetilde{w}_{i-1}\|^2 \\
&\overset{(b)}{\leq} \|\widetilde{w}_{i-1}\|^2 + \mu^2 \delta^2 \|\widetilde{w}_{i-1}\|^2 - 2\mu\nu \|\widetilde{w}_{i-1}\|^2 \\
&= \alpha \|\widetilde{w}_{i-1}\|^2
\end{aligned} \tag{2.26}
$$

where step (a) uses the mean-value relation (D.9) and the strong-convexity property (C.17) from the appendices, while step (b) uses the upper bound in (2.18) on the Hessian matrix.

1st Proof

We next verify that condition (2.22) ensures $0 \leq \alpha < 1$. For this purpose, we refer to Figure 2.3, which plots the coefficient $\alpha(\mu)$ as a function of μ. The minimum value of $\alpha(\mu)$, which occurs at the location $\mu = \nu/\delta^2$ and is equal to $1 - \nu^2/\delta^2$, is nonnegative since $0 < \nu \leq \delta$. It is now clear from the figure that $0 \leq \alpha < 1$ for $\mu \in (0, \frac{2\nu}{\delta^2})$.

1st Proof

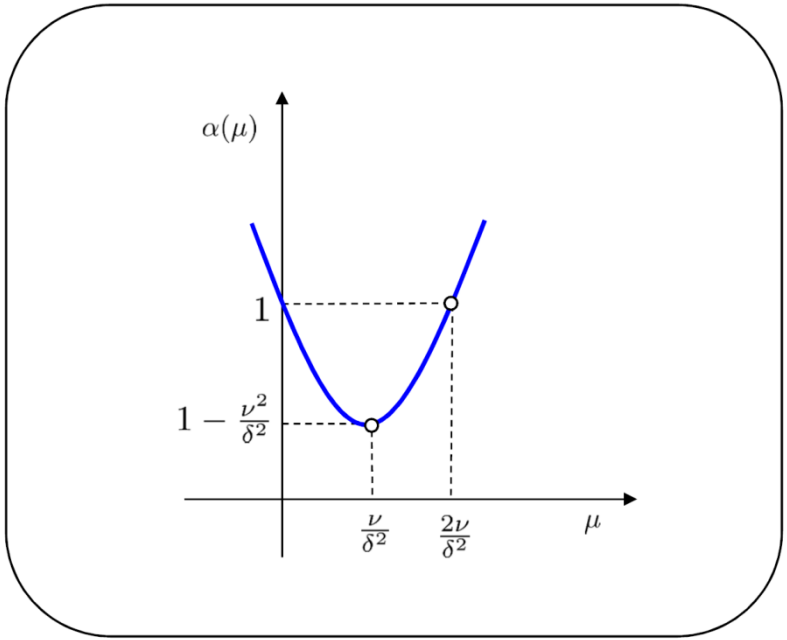


Figure 2.3: Plot of the function $\alpha(\mu) = 1-2\nu\mu+\mu^2\delta^2$ given by (2.24). It shows that the function $\alpha(\mu)$ assumes values below one in the range $0 < \mu < 2\nu/\delta^2$.

Recall#4: Spectral Radius & Norms

Consider an $N \times N$ matrix A. Its spectral radius is defined by:

$$\rho(A) \triangleq \max_{1 \le k \le N} |\lambda_k(A)|$$

The following two properties hold:

(a) For any induced matrix norm:

$$\boxed{\rho(A) \le \|A\|_p, \quad p \ge 1}$$

(b) When A is symmetric:

$$\boxed{\rho(A) = \|A\|_2}$$

2nd Proof

Alternative proof. We can arrive at the same conclusion by using an alternative argument, which may seem to be more demanding at first sight. However, it turns out to be more convenient for scenarios involving optimization by networked agents, as we are going to study in future chapters — see, e.g., the derivation in Sec. 8.4.

We again subtract w^o from both sides of (2.21) to get

$$\widetilde{w}_i \;=\; \widetilde{w}_{i-1} \;+\; \mu\,\nabla_{w^{\mathsf{T}}} J(w_{i-1}) \tag{2.27}$$

2nd Proof

We then appeal to the mean-value relation (D.9) from the appendix to note that

$$\begin{aligned}\nabla_{w^{\mathsf{T}}} J(w_{i-1}) &= -\left(\int_0^1 \nabla_w^2\, J(w^o - t\widetilde{w}_{i-1})dt\right)\widetilde{w}_{i-1} \\ &\triangleq -H_{i-1}\widetilde{w}_{i-1} \qquad (2.28)\end{aligned}$$

where we are introducing the *symmetric* time-variant matrix H_{i-1}, which is defined in terms of the Hessian of the cost function:

$$H_{i-1} \triangleq \int_0^1 \nabla_w^2\, J(w^o - t\widetilde{w}_{i-1})dt \qquad (2.29)$$

2nd Proof

Substituting (2.28) into (2.27), we get the alternative representation:

$$\widetilde{w}_i \;=\; (I_M - \mu H_{i-1})\widetilde{w}_{i-1} \tag{2.30}$$

Note that the matrix H_{i-1} depends on $\widetilde{w}_{i-1}$ so that the right-hand side of the above recursion actually depends on $\widetilde{w}_{i-1}$ in a nonlinear fashion. However, we can still determine a condition on μ for convergence of $\widetilde{w}_i$ to zero because we can determine a uniform bound on H_{i-1} as follows [191]. Using the sub-multiplicative property of norms, we have

$$\|\widetilde{w}_i\|^2 \;\leq\; \|I_M - \mu H_{i-1}\|^2 \cdot \|\widetilde{w}_{i-1}\|^2 \tag{2.31}$$

2nd Proof

But since $J(w)$ satisfies (2.18), we know that

$$(1-\mu\delta)I_M \;\leq\; I_M - \mu H_{i-1} \;\leq\; (1-\mu\nu)I_M \tag{2.32}$$

for all i. Using the fact that $I_M - \mu H_{i-1}$ is a symmetric matrix, we have that its $2-$induced norm is equal to its spectral radius so that

$$\begin{aligned}
\|I_M - \mu H_{i-1}\|^2 &= [\rho(I_M - \mu H_{i-1})]^2 \\
&\overset{(2.32)}{\leq} \max\{(1-\mu\delta)^2,\ (1-\mu\nu)^2\} \\
&= \max\left\{1 - 2\mu\delta + \mu^2\delta^2,\ 1 - 2\mu\nu + \mu^2\nu^2\right\} \\
&\overset{(a)}{\leq} 1 - 2\mu\nu + \mu^2\delta^2 \\
&= \alpha
\end{aligned} \tag{2.33}$$

2nd Proof

where we used the fact that $\delta \geq \nu$ in step (a). Combining this result with (2.31) we again conclude that (2.23) holds and, therefore, condition (2.22) on the step-size ensures $\widetilde{w}_i \to 0$ as $i \to \infty$.

Actually, the argument that led to (2.33) can be refined to conclude that convergence of $\widetilde{w}_i$ to zero occurs over the wider interval

$$\mu < 2/\delta \tag{2.34}$$

than (2.22). This is because condition (2.34) already ensures

$$\max\{(1-\mu\delta)^2,\ (1-\mu\nu)^2\} < 1 \tag{2.35}$$

We will continue with condition (2.22); it is sufficient for our purposes to know that a small enough step-size value exists that ensures convergence.

□

Example #2.3

Example 2.3 (Optimization of mean-square-error costs). Let us reconsider the quadratic cost (2.6) from Example 2.1. We know from (2.19) that $\delta = 2\lambda_{\max}(R_u)$ and $\nu = 2\lambda_{\min}(R_u)$. Furthermore, if we set the gradient vector in (2.8) to zero, we conclude that the minimizer, w^o, is given by the unique solution to the equations $R_u w^o = r_{du}$. We can alternatively determine this same minimizer in an iterative manner by using the gradient descent recursion (2.21). Indeed, if we substitute expression (2.8) for the gradient vector into (2.21), we find that the iterative algorithm reduces to

$$w_i = w_{i-1} + 2\mu\,(r_{du} - R_u w_{i-1}), \quad i \geq 0 \tag{2.36}$$

Example #2.3

We know from condition (2.22) that the iterates $\{w_i\}$ generated by this recursion will converge to w^o at an exponential rate for any step-size $\mu < \lambda_{\min}(R_u)/\lambda^2_{\max}(R_u)$. Using condition (2.34) instead, we actually have that convergence of w_i to w^o is guaranteed over the wider range of step-size values $\mu < 1/\lambda_{\max}(R_u)$. This conclusion can also be seen from the fact that, in this case, the matrix H_{i-1} defined by (2.29) is constant and equal to $2R_u$ (i.e., it is independent of $\widetilde{w}_{i-1}$). In this way, recursion (2.30) becomes

$$\widetilde{w}_i \;=\; (I_M - 2\mu R_u)\widetilde{w}_{i-1}, \quad i \geq 0 \tag{2.37}$$

from which it is again clear that $\widetilde{w}_i$ converges to zero for all $\mu < 1/\lambda_{\max}(R_u)$.

■

Decaying Step-Sizes

Course EE210B
Spring Quarter 2015

Proc. IEEE, vol. 102, no. 4, pp. 460-497, April 2014.
Foundations and Trends in Machine Learning, vol. 7, no. 4-5, pp. 311-801, July 2014.

Decaying Step-Sizes

It is also possible to employ in (2.21) iteration-dependent step-size sequences, $\mu(i) \geq 0$, instead of the constant step-size μ, and to require $\mu(i)$ to satisfy the two conditions:

$$\sum_{i=0}^{\infty} \mu(i) = \infty, \quad \lim_{i \to \infty} \mu(i) = 0 \tag{2.38}$$

For example, sequences of the form

$$\mu(i) = \frac{\tau}{i+1}, \quad i \geq 0 \tag{2.39}$$

Decaying Step-Sizes

satisfy conditions (2.38) for any finite positive constant τ. It is well-known that, under (2.38), the gradient descent recursion, namely,

$$w_i = w_{i-1} - \mu(i)\,\nabla_{w^{\mathsf{T}}} J(w_{i-1}), \quad i \geq 0 \tag{2.40}$$

continues to ensure the convergence of w_i towards w^o, as explained next [32, 191, 244]. However, the convergence rate will now be slower and in the order of $O(1/i^{2\nu\tau})$. That is, the convergence rate will not be geometric (or exponential) any longer. For this reason, the constant step-size implementation is preferred. Nevertheless, we will still discuss

Decaying Step-Sizes

the decaying step-size case in order to prepare for our future treatment of stochastic gradient algorithms where such step-sizes are more relevant. A second issue with the use of decaying step-sizes is that conditions (2.38) force the step-size sequence to decay to zero; this feature is problematic for scenarios requiring continuous adaptation and learning from streaming data (which will be the main focus of our treatment starting from the next chapter). This is because, in many instances, it is not unusual for the location of the minimizer, w^o, to drift with time. With $\mu(i)$ decaying towards zero, the gradient descent algorithm (2.40) will stop updating and will not be able to track drifts in the solution.

Statement

Lemma 2.2 (Convergence with decaying step-size sequence: Real case)**.** Assume the cost function, $J(w)$, satisfies Assumption 2.1. If the step-size sequence $\mu(i)$ satisfies the two conditions in (2.38), then it holds that for any initial condition, w_{-1}, the gradient descent algorithm (2.40) generates iterates $\{w_i\}$ that converge to the global minimizer, w^o. Moreover, when the step-size sequence is chosen as in (2.39), then the convergence rate is in the order of $\|\widetilde{w}_i\|^2 = O(1/i^{2\nu\tau})$ for large enough i.

Proof

Proof. We first establish the convergence result for step-size sequences satisfying (2.38). The argument that led to (2.26) will similarly lead to

$$\|\widetilde{w}_i\|^2 \quad \leq \quad \alpha(i)\,\|\widetilde{w}_{i-1}\|^2 \tag{2.41}$$

where now $\alpha(i) = 1 - 2\nu\mu(i) + \delta^2\mu^2(i)$. We split $2\nu\mu(i)$ into the sum of two factors and write

$$\alpha(i) \quad = \quad 1 - \nu\mu(i) - \nu\mu(i) + \delta^2\mu^2(i) \tag{2.42}$$

Now, since $\mu(i) \to 0$, we conclude that for large enough $i > i_o$, the sequence $\mu^2(i)$ will assume smaller values than $\mu(i)$. Therefore, a large enough time index, i_o, exists such that the following two conditions are satisfied:

Proof

$$\nu\mu(i) \geq \delta^2\mu^2(i), \quad 0 < 1 - \nu\mu(i) \leq 1, \quad i > i_o \tag{2.43}$$

It follows that

$$\alpha(i) \leq 1 - \nu\mu(i), \quad i > i_o \tag{2.44}$$

and, hence,

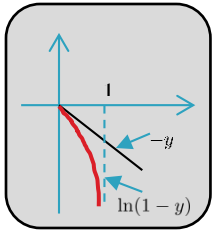
$$\|\widetilde{w}_i\|^2 \leq (1 - \nu\mu(i)) \, \|\widetilde{w}_{i-1}\|^2, \quad i > i_o \tag{2.45}$$

Iterating over i we can write (assuming a finite i_o exists for which $\|\tilde{w}_{i_o}\| \neq 0$, otherwise the algorithm would have converged) [165, 206]:

Proof

$$\lim_{i\to\infty}\left(\frac{\|\widetilde{w}_i\|^2}{\|\widetilde{w}_{i_o}\|^2}\right) \le \prod_{i=i_o+1}^{\infty}(1-\nu\mu(i)) \tag{2.46}$$

or, equivalently,



$$\lim_{i\to\infty}\ln\left(\frac{\|\widetilde{w}_i\|^2}{\|\widetilde{w}_{i_o}\|^2}\right) \le \sum_{i=i_o+1}^{\infty}\ln(1-\nu\mu(i)) \tag{2.47}$$

Now using the following easily verified property for the natural logarithm function:

$$\ln(1-y) \le -y, \quad \text{for all } 0\le y<1 \tag{2.48}$$

Proof

and letting $y = \nu\mu(i)$, we have that

$$\ln(1-\nu\mu(i)) \;\leq\; -\nu\mu(i), \quad i > i_o \tag{2.49}$$

so that

$$\sum_{i=i_o+1}^{\infty} \ln(1-\nu\mu(i)) \;\leq\; -\sum_{i=i_o+1}^{\infty} \nu\mu(i) \;=\; -\nu\left(\sum_{i=i_o+1}^{\infty} \mu(i)\right) \;=\; -\infty \tag{2.50}$$

since the step-size series is assumed to be divergent in (2.38). We conclude that

$$\lim_{i\to\infty} \ln\left(\frac{\|\widetilde{w}_i\|^2}{\|\widetilde{w}_{i_o}\|^2}\right) = -\infty \tag{2.51}$$

so that $\widetilde{w}_i \to 0$ as $i \to \infty$.

Proof

We now examine the convergence rate for step-size sequences of the form (2.39). Note first that these sequences satisfy the following two conditions

$$\sum_{i=0}^{\infty}\mu(i)=\infty, \qquad \sum_{i=0}^{\infty}\mu^2(i) \;=\; \tau^2\left(\sum_{i=1}^{\infty}\frac{1}{i^2}\right) \;=\; \frac{\tau^2\pi^2}{6}<\infty \tag{2.52}$$

Again, since $\mu(i)\to 0$ and $\mu^2(i)$ decays faster than $\mu(i)$, we know that for some large enough $i>i_1$, it will hold that

$$2\nu\mu(i)\geq\delta^2\mu^2(i) \tag{2.53}$$

and, hence,

$$0<\alpha(i)\leq 1, \quad i>i_1 \tag{2.54}$$

Proof

We can now repeat the same steps up to (2.51) using $y = 2\nu\mu(i') - \delta^2\mu^2(i')$ to conclude that

$$\begin{aligned}
\ln\left(\frac{\|\widetilde{w}_i\|^2}{\|\widetilde{w}_{i_1}\|^2}\right) &\leq \sum_{i'=i_1+1}^{i} \ln\left(1 - 2\nu\mu(i') + \delta^2\mu^2(i')\right) \\
&\leq -\sum_{i'=i_1+1}^{i} \left[2\nu\mu(i') - \delta^2\mu^2(i')\right] \\
&= -2\nu\left(\sum_{i'=i_1+1}^{i} \mu(i')\right) + \delta^2\left(\sum_{i'=i_1+1}^{i} \mu^2(i')\right)
\end{aligned}$$

Proof

$$
\begin{aligned}
&\leq -2\nu\left(\sum_{i'=i_1+1}^{i}\mu(i')\right)+\frac{\delta^2\tau^2\pi^2}{6}\\
&= -2\nu\tau\left(\sum_{i'=i_1+2}^{i+1}\frac{1}{i'}\right)+\frac{\delta^2\tau^2\pi^2}{6}\\
&\overset{(a)}{\leq} -2\nu\tau\left(\int_{i_1+2}^{i+2}\frac{1}{x}dx\right)+\frac{\delta^2\tau^2\pi^2}{6}\\
&= 2\nu\tau\ln\left(\frac{i_1+2}{i+2}\right)+\frac{\delta^2\tau^2\pi^2}{6}\\
&= \ln\left(\frac{i_1+2}{i+2}\right)^{2\nu\tau}+\frac{\delta^2\tau^2\pi^2}{6} \qquad (2.55)
\end{aligned}
$$

Proof

where in step (a) we used the following integral bound, which reflects the fact that the area under the curve $f(x) = 1/x$ over the interval $x \in [i_1 + 2, i + 2]$ is upper bounded by the sum of the areas of the rectangles shown in Figure 2.4:

$$\int_{i_1+2}^{i+2} \frac{1}{x} dx \leq \sum_{i'=i_1+2}^{i+1} \frac{1}{i'} \tag{2.56}$$

Proof

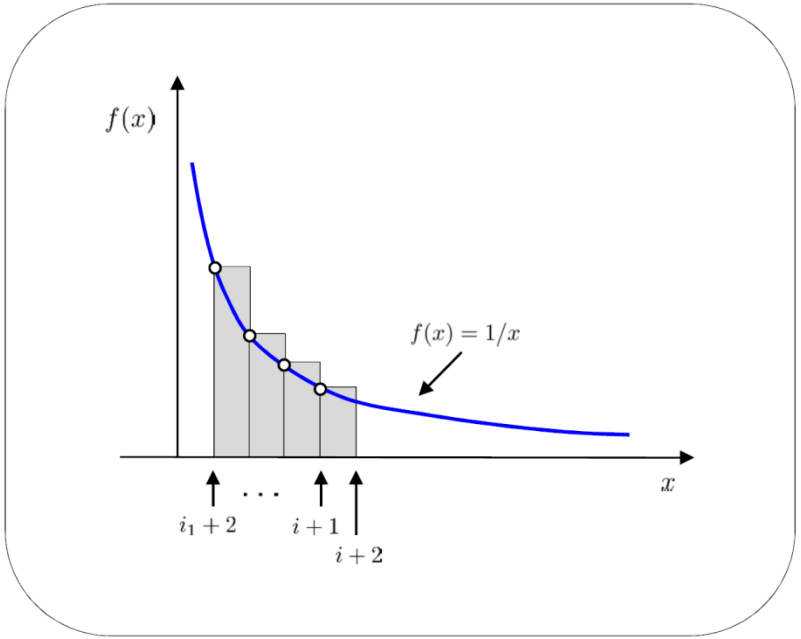


Figure 2.4: The area under the curve $f(x) = 1/x$ over the interval $x \in [i_1 + 2, i + 2]$ is upper bounded by the sum of the areas of the rectangles shown in the figure.

Proof

We therefore conclude from (2.55) that

$$\begin{aligned}
\|\widetilde{w}_i\|^2 &\leq \left(e^{\left\{\ln\left(\frac{i_1+2}{i+2}\right)^{2\nu\tau}+\frac{\delta^2\tau^2\pi^2}{6}\right\}}\right)\|\widetilde{w}_{i_1}\|^2, \quad i>i_1 \\
&= e^{\frac{\delta^2\tau^2\pi^2}{6}}\cdot\|\widetilde{w}_{i_1}\|^2\cdot\left(\frac{i_1+2}{i+2}\right)^{2\nu\tau} \\
&= O(1/i^{2\nu\tau}) \qquad (2.57)
\end{aligned}$$

as claimed.

□

Complex Domain

Course EE210B
Spring Quarter 2015

Proc. IEEE, vol. 102, no. 4, pp. 460-497, April 2014.
Foundations and Trends in Machine Learning, vol. 7, no. 4-5, pp. 311-801, July 2014.

Recall#5: Real vs Complex

Consider a function $J(w)$, with $w = x + jy$, $v = \text{col}\{x, y\}$:

$$
\begin{aligned}
H(v) &= \nabla_v^2 J(v) = D^* \left[\nabla_w^2 J(w)\right] D \\
\frac{1}{2}\left[\nabla_v J(v)\right] D^* &= \begin{bmatrix} \nabla_w J(w) & (\nabla_{w^*} J(w))^{\mathsf{T}} \end{bmatrix} \\
D &= \begin{bmatrix} I_M & jI_M \\ I_M & -jI_M \end{bmatrix} \\
DD^* &= 2I_{2M}
\end{aligned}
$$

$$
\underbrace{\begin{bmatrix} I_M & jI_M \\ I_M & -jI_M \end{bmatrix}}_{=D} \underbrace{\begin{bmatrix} x \\ y \end{bmatrix}}_{=v} = \begin{bmatrix} w \\ (w^*)^{\mathsf{T}} \end{bmatrix}
$$

Recall#6: Real vs Complex

Consider a $\nu-$strongly convex function $J(w)$:

$$J(\alpha w_1 + (1-\alpha)w_2) \leq \alpha J(w_1) + (1-\alpha)J(w_2) - \frac{\nu}{2}\alpha(1-\alpha)\|w_1 - w_2\|^2$$

$$\Longleftrightarrow \begin{cases} w \text{ real}: & \nabla_w^2 J(w) \geq \nu I_M > 0 \\ w \text{ complex}: & \nabla_w^2 J(w) \geq \dfrac{\nu}{2} I_{2M} > 0 \end{cases}$$

Recall#7: Real vs Complex

Consider a convex function $J(w)$:

$$\begin{cases} w \text{ real}: & \nabla_w^2 J(w) \leq \delta I_M \iff \|\nabla J(w_1) - \nabla J(w_2)\| \leq \delta \|w_1 - w_2\| \\ w \text{ complex}: & \nabla_w^2 J(w) \leq \dfrac{\delta}{2} I_{2M} \iff \|\nabla J(w_1) - \nabla J(w_2)\| \leq \dfrac{\delta}{2} \|w_1 - w_2\| \end{cases}$$

Complex Domain

We now extend the results of the previous two sections to the case in which the argument $w \in \mathbb{C}^M$ is complex-valued while $J(w) \in \mathbb{R}$ continues to be real-valued. We again focus on the case of strongly-convex functions, $J(w)$, for which the minimizer, w^o, is unique. It is explained in (C.44) in the appendix that, in the complex case, condition (2.14) is replaced by

$$J(w) \text{ is } \nu-\text{strongly convex} \iff \nabla_w^2 J(w) \geq \frac{\nu}{2} I_{2M} > 0 \qquad (2.58)$$

Strong Convexity

with a factor of $\frac{1}{2}$ multiplying ν, and with I_M replaced by I_{2M} since the Hessian matrix is now $2M \times 2M$. Note that we can capture conditions (2.14) and (2.58) simultaneously in a single statement for both cases of real or complex-valued arguments by writing

$$J(w) \text{ is } \nu-\text{strongly convex} \iff \nabla_w^2 J(w) \geq \frac{\nu}{h} I_{hM} > 0 \qquad (2.59)$$

where the variable h is an integer that denotes the type of the data:

$$h \triangleq \begin{cases} 1, & \text{when } w \text{ is real} \\ 2, & \text{when } w \text{ is complex} \end{cases} \qquad (2.60)$$

Notation: Type Factor

Observe that h appears in two locations in (2.59); in the denominator of ν and in the subscript indicating the size of the identity matrix. We shall frequently employ the data-type variable, h, throughout our presentation, and especially in future chapters, in order to permit a uniform treatment of the various algorithms regardless of the type of the data.

Likewise, the Lipschitz condition (2.17) is replaced by

$$\|\nabla_w J(w_2) - \nabla_w J(w_1)\| \leq \frac{\delta}{h}\|w_2 - w_1\| \tag{2.61}$$

for all w_1, w_2, where again a factor of $h = 2$ would appear on the right-hand-side in the complex case.

Lipschitz Condition

It follows from the result of Lemma E.7 in the appendix that for twice-differentiable costs, conditions (2.59) and (2.61) combined are equivalent to

$$0 < \frac{\nu}{h} I_{hM} \leq \nabla_w^2 J(w) \leq \frac{\delta}{h} I_{hM} \tag{2.62}$$

We therefore assume that the cost function $J(w)$ is twice-differentiable and satisfies (2.62) for some positive parameters $\nu \leq \delta$.

Example #2.4

Example 2.4 (Complex Hessian matrices). Let us reconsider the context of Example 2.1 with complex data. Let $\boldsymbol{d}$ be a scalar zero-mean random variable with variance $\sigma_d^2 = \mathbb{E}\,|\boldsymbol{d}|^2$ and let $\boldsymbol{u}$ be a $1 \times M$ zero-mean random vector with covariance matrix $R_u = \mathbb{E}\,\boldsymbol{u}^*\boldsymbol{u} > 0$. The cross-correlation vector between $\boldsymbol{d}$ and $\boldsymbol{u}$ is denoted by $r_{du} = \mathbb{E}\,\boldsymbol{d}\boldsymbol{u}^*$. The mean-square-error cost function is now defined as

$$\begin{aligned} J_k(w) &\stackrel{\Delta}{=} \mathbb{E}\,|\boldsymbol{d} - \boldsymbol{u}w|^2 \\ &= \sigma_d^2 - (r_{du})^* w - w^* r_{du} + w^* R_u w \end{aligned} \tag{2.63}$$

Example #2.4

The complex gradient vectors of $J(w)$ relative to w and w^* are given by — see Example A.3 in the appendix:

$$\nabla_w J(w) = (R_u w - r_{du})^*, \quad \nabla_{w^*} J(w) = R_u w - r_{du} \tag{2.64}$$

and the $2M \times 2M$ Hessian matrix of $J(w)$ can be verified to be block diagonal in this case and given by — see (B.36) in the appendix:

$$\nabla_w^2 J(w) = \begin{bmatrix} R_u & 0 \\ 0 & R_u^{\mathsf{T}} \end{bmatrix} \tag{2.65}$$

It is instructive to compare expressions (2.64) and (2.65) with (2.8) in the real case.

■

Gradient Descent

In the complex case, the gradient descent algorithm (2.21) is replaced by

$$w_i = w_{i-1} - \mu \nabla_{w^*} J(w_{i-1}), \quad i \geq 0 \tag{2.66}$$

in terms of the complex gradient vector relative to w^*. Since $J(w)$ is real-valued, it holds that

$$\nabla_{w^*} J(w_{i-1}) = [\nabla_w J(w_{i-1})]^* \tag{2.67}$$

Gradient Descent

Comparing with (2.21) we see that transposition of the gradient vector is replaced by complex conjugation. The above recursion can be motivated from (2.21) as follows. We express the complex variable w in terms of its real and imaginary components as

$$w = x + jy \tag{2.68}$$

We then treat $J(w)$ as the function $J(v)$ of the $2M \times 1$ extended real variable:

$$v = \mathrm{col}\{x, y\} \tag{2.69}$$

Gradient Descent

and consider instead the equivalent optimization problem

$$\min_{v \in \mathbb{R}^{2M}} \; J(v) \tag{2.70}$$

We already know from (2.21) that the gradient descent recursion for minimizing $J(v)$ over v, using the step-size $\mu' = \mu/2$, has the form:

$$v_i \;=\; v_{i-1} \;-\; \frac{1}{2}\mu \, \nabla_{v^{\mathsf{T}}} J(v_{i-1}), \quad i \geq 0 \tag{2.71}$$

Gradient Descent

The reason for introducing the factor of $\frac{1}{2}$ into μ' will become clear soon. We can rewrite the above recursion in terms of the components of $v_i = \text{col}\{x_i, y_i\}$ as follows:

$$\begin{bmatrix} x_i \\ y_i \end{bmatrix} = \begin{bmatrix} x_{i-1} \\ y_{i-1} \end{bmatrix} - \frac{1}{2}\mu \begin{bmatrix} \nabla_{x^\mathsf{T}} J(x_{i-1}, y_{i-1}) \\ \nabla_{y^\mathsf{T}} J(x_{i-1}, y_{i-1}) \end{bmatrix} \tag{2.72}$$

Gradient Descent

where we used relation (C.29) from the appendix to express the gradient vector of $J(v)$ in terms of the gradients of the same function $J(x,y)$ relative to x and y. Now, if we multiply the second block row of (2.72) by jI_M, add both block rows, and use $w_i = x_i + jy_i$, we can rewrite (2.72) in terms of the complex variables $\{w_i, w_{i-1}\}$:

$$\begin{aligned} w_i &= w_{i-1} - \frac{1}{2}\mu\left[\nabla_{x^{\mathsf{T}}} J(x_{i-1},y_{i-1}) + j\,\nabla_{y^{\mathsf{T}}} J(x_{i-1},y_{i-1})\right] \\ &\overset{(C.31)}{=} w_{i-1} - \mu\,\nabla_{w^*} J(w_{i-1}) \end{aligned} \tag{2.73}$$

Gradient Descent

The second relation above agrees with the claimed form (2.66); it is seen that the factor of $1/2$ is used in transforming the combination of gradient vectors relative to x and y into the gradient vector relative to w. The next statement establishes the convergence of (2.66); in the statement, we employ the data-type variable, h, so that the conclusion encompasses both the real and complex-valued domains.

Geometric Convergence

Lemma 2.3 (Convergence with constant step-size: Complex case). Assume the cost function $J(w)$ satisfies (2.62). If the step-size μ is chosen to satisfy

$$\frac{\mu}{h} < \frac{2\nu}{\delta^2} \tag{2.74}$$

then, it holds that for any initial condition, w_{-1}, the gradient descent algorithm (2.66) generates iterates that converge exponentially fast to the global minimizer, w^o, i.e., it holds that

$$\|\widetilde{w}_i\|^2 \leq \alpha^i \|\widetilde{w}_{i-1}\|^2 \tag{2.75}$$

Geometric Convergence

where the real scalar α satisfies $0 \leq \alpha < 1$ and is given by

$$\alpha = 1 - 2\nu\left(\frac{\mu}{h}\right) + \delta^2\left(\frac{\mu}{h}\right)^2 \tag{2.76}$$

Proof

Proof. We are only interested in establishing the above results in the complex case, which corresponds to $h = 2$, since we already established these same conclusions for the real case in Lemma 2.1. Rather than establish the claims by working directly with recursion (2.66) in the complex domain, we instead reduce the problem to one that deals with the equivalent function $J(v)$ of the extended *real* variable $v = \text{col}\{x, y\}$ and then apply the result of Lemma 2.1.

Proof

To begin with, we already know from (E.39) in the appendix that if $J(w)$ is $\nu-$strongly convex, then $J(v)$ is $\nu-$strongly convex as well. We also know from (E.22) and (E.56) in the same appendix that the gradient vector function of $J(v)$ is Lipschitz with factor δ when the gradient vector function of $J(w)$ is Lipschitz with factor $\delta/2$. We further know from (2.71)–(2.73) that a gradient descent recursion in the $w-$domain (as in (2.73)) is equivalent to a gradient descent recursion in the $v-$domain (as in (2.71)) if we use $\mu' = \mu/2$:

$$v_i = v_{i-1} - \mu' \nabla_{v^{\mathsf{T}}} J(v_{i-1}), \quad i \geq 0 \tag{2.77}$$

Proof

Lemma 2.1 then guarantees that the real-valued iterates $\{v_i\}$ will converge to v^o when $\mu' < 2\nu/\delta^2$. Consequently, the gradient descent algorithm (2.66) will converge for $\mu < 4\nu/\delta^2$, which is condition (2.74) with $h = 2$ in the complex case. We note that from the argument that led to (2.34) we can conclude that convergence actually occurs over the wider interval $\mu' < 2/\delta$ or, equivalently, $\mu/h < 2/\delta$. Either way, we find that relation (2.75) holds by noting that $\|\widetilde{w}_i\|^2 = \|\widetilde{v}_i\|^2$ and using the result from Lemma 2.1 to conclude that

$$\|\widetilde{v}_i\|^2 \;\leq\; \alpha^i \, \|\widetilde{v}_{i-1}\|^2 \tag{2.78}$$

where $\widetilde{v}_i = v^o - v_i$ and

$$\alpha = 1 - 2\mu'\nu + (\mu')^2\delta^2 \tag{2.79}$$

□

Decaying Step-Sizes

We can also study gradient descent recursions with decaying step-size sequences satisfying (2.38), namely,

$$w_i = w_{i-1} - \mu(i)\,\nabla_{w^*} J(w_{i-1}), \quad i \geq 0 \tag{2.80}$$

Decaying Step-Sizes

Lemma 2.4 (Convergence with decaying step-size: Complex case)**.** Assume the cost function $J(w)$ satisfies (2.62). If the step-size sequence $\mu(i)$ satisfies (2.38), then it holds that for any initial condition, w_{-1}, the gradient descent algorithm (2.80) generates iterates $\{w_i\}$ that converge to the global minimizer, w^o. Moreover, when the step-size sequence is chosen as in (2.39), then the convergence rate is in the order of $\|\widetilde{w}_i\|^2 = O(1/i^{(2\nu\tau/h)})$ for large enough i.

Proof

Proof. We apply Lemma 2.2 to the following recursion in the $v-$domain:

$$v_i \;=\; v_{i-1} \;-\; \mu'(i)\,\nabla_{v^{\mathsf{T}}} J(v_{i-1}), \quad i \geq 0 \tag{2.81}$$

where $\mu'(i) = \mu(i)/2$.

□

End of Lecture

Proc. IEEE, vol. 102, no. 4, pp. 460-497, April 2014.
Foundations and Trends in Machine Learning, vol. 7, no. 4-5, pp. 311-801, July 2014.