# Appendix A
# Mathematical Preliminaries

In this appendix, mathematical preliminaries on linear algebra, stability theory, probability and stochastic processes, and optimization are introduced. Some classification measures are also defined.

## A.1 Linear Algebra

### Pseudoinverse

**Definition A.1** (*Pseudoinverse*) Pseudoinverse $\mathbf{A}^\dagger$, also called *Moore–Penrose generalized inverse*, of a matrix $\mathbf{A} \in R^{m\times n}$ is unique, which satisfies

$$\mathbf{A}\mathbf{A}^\dagger\mathbf{A} = \mathbf{A}, \tag{A.1}$$

$$\mathbf{A}^\dagger\mathbf{A}\mathbf{A}^\dagger = \mathbf{A}^\dagger, \tag{A.2}$$

$$\left(\mathbf{A}\mathbf{A}^\dagger\right)^T = \mathbf{A}\mathbf{A}^\dagger, \tag{A.3}$$

$$\left(\mathbf{A}^\dagger\mathbf{A}\right)^T = \mathbf{A}^\dagger\mathbf{A}. \tag{A.4}$$

$\mathbf{A}^\dagger$ can be calculated by

$$\mathbf{A}^\dagger = \left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{A}^T \tag{A.5}$$

if $\mathbf{A}^T\mathbf{A}$ is nonsingular, and

$$\mathbf{A}^\dagger = \mathbf{A}^T\left(\mathbf{A}\mathbf{A}^T\right)^{-1} \tag{A.6}$$

if $\mathbf{A}\mathbf{A}^T$ is nonsingular. Pseudoinverse is directly associated with the linear LS problem.



K.-L. Du and M. N. S. Swamy, *Neural Networks and Statistical Learning*,
https://doi.org/10.1007/978-1-4471-7452-3

When $\mathbf{A}$ is a square nonsingular matrix, pseudoinverse $\mathbf{A}^{\dagger}$ reduces to its inverse $\mathbf{A}^{-1}$. For a scalar $\alpha$, $\alpha^{\dagger} = \alpha^{-1}$ for $\alpha \neq 0$, and $\alpha^{\dagger} = 0$ for $\alpha = 0$.

For $n \times n$ identity matrix $\mathbf{I}$ and $n \times n$ singular matrix $\mathbf{J}$, namely, $\det(\mathbf{J}) = 0$, for $a \neq 0$ and $a + nb \neq 0$, we have [14]

$$(a\mathbf{I} + b\mathbf{J})^{-1} = \frac{1}{a}\left(\mathbf{I} - \frac{b}{a+nb}\mathbf{J}\right). \tag{A.7}$$

**Linear Least Squares Problems**

The linear LS or $L_2$-norm problem is basic to many signal processing techniques. It tries to solve a set of linear equations, written in matrix form

$$\mathbf{A}\boldsymbol{x} = \boldsymbol{b}, \tag{A.8}$$

where $\mathbf{A} \in R^{m\times n}$, $\boldsymbol{x} \in R^n$, and $\boldsymbol{b} \in R^m$.

This problem can be converted into the minimization of the squared error function

$$E(\boldsymbol{x}) = \frac{1}{2}\|\mathbf{A}\boldsymbol{x} - \boldsymbol{b}\|_2^2 = \frac{1}{2}(\mathbf{A}\boldsymbol{x} - \boldsymbol{b})^T(\mathbf{A}\boldsymbol{x} - \boldsymbol{b}). \tag{A.9}$$

The solution corresponds to one of the following three situations [7]:

- $\text{rank}(\mathbf{A}) = n = m$. We get a unique exact solution

$$\boldsymbol{x}^* = \mathbf{A}^{-1}\boldsymbol{b} \tag{A.10}$$

  and $E(\boldsymbol{x}^*) = 0$.
- $\text{rank}(\mathbf{A}) = n < m$. The system is overdetermined and has no exact solution. There is a unique solution in the least squares error sense

$$\boldsymbol{x}^* = \mathbf{A}^{\dagger}\boldsymbol{b}, \tag{A.11}$$

  where $\mathbf{A}^{\dagger} = \left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{A}^T$. In this case,

$$E\left(\boldsymbol{x}^*\right) = \boldsymbol{b}^T\left(\mathbf{I} - \mathbf{A}\mathbf{A}^{\dagger}\right)\boldsymbol{b} \geq 0. \tag{A.12}$$

- $\text{rank}(\mathbf{A}) = m < n$. The system is underdetermined, and the solution is not unique. But the solution with the minimum $L_2$-norm $\|\boldsymbol{x}\|_2^2$ is unique

$$\boldsymbol{x}^* = \mathbf{A}^{\dagger}\boldsymbol{b}. \tag{A.13}$$

  Here $\mathbf{A}^{\dagger} = \mathbf{A}^T\left(\mathbf{A}\mathbf{A}^T\right)^{-1}$. We have $E(\boldsymbol{x}^*) = 0$ and $\|\boldsymbol{x}^*\|_2^2 = \boldsymbol{b}^T\left(\mathbf{A}\mathbf{A}^T\right)^{-1}\boldsymbol{b}$.

## Vector Norms

**Definition A.2** (*Vector Norms*) A norm acts as a measure of distance. A vector norm on $R^n$ is a mapping $f : R^n \to R$ that satisfies such properties: For any $\boldsymbol{x}, \boldsymbol{y} \in R^n, a \in R$,

- $f(\boldsymbol{x}) \geq 0$, and $f(\boldsymbol{x}) = 0$ iff $\boldsymbol{x} = 0$.
- $f(\boldsymbol{x} + \boldsymbol{y}) \leq f(\boldsymbol{x}) + f(\boldsymbol{y})$.
- $f(a\boldsymbol{x}) = |a| f(\boldsymbol{x})$.

The mapping is denoted as $f(\boldsymbol{x}) = \|\boldsymbol{x}\|$.

The $p$-norm or $L_p$-norm is a popular class of vector norms

$$\|\boldsymbol{x}\|_p = \left( \sum_{i=1}^{n} |x_i|^p \right)^{\frac{1}{p}} \tag{A.14}$$

with $p \geq 1$. The $L_1$, $L_2$, and $L_\infty$ norms are more useful:

$$\|\boldsymbol{x}\|_1 = \sum_{i=1}^{n} |x_i| , \tag{A.15}$$

$$\|\boldsymbol{x}\|_2 = \sum_{i=1}^{n} \left(x_i^2\right)^{\frac{1}{2}} = \left(\boldsymbol{x}^T \boldsymbol{x}\right)^{\frac{1}{2}} , \tag{A.16}$$

$$\|\boldsymbol{x}\|_\infty = \max_{1 \leq i \leq n} |x_i| . \tag{A.17}$$

The $L_2$-norm is the popular Euclidean norm.

A matrix $\mathbf{Q} \in R^{m \times m}$ is called an *orthogonal matrix* or *unitary matrix* if $\mathbf{Q}^T \mathbf{Q} = \mathbf{I}$. The $L_2$-norm is invariant under orthogonal transforms, that is, for all orthogonal $\mathbf{Q}$ of appropriate dimensions

$$\|\mathbf{Q}\boldsymbol{x}\|_2 = \|\boldsymbol{x}\|_2. \tag{A.18}$$

## Matrix Norms

A matrix norm is a generalization of the vector norm by extending from $R^n$ to $R^{m \times n}$. For a matrix $\mathbf{A} = \left[a_{ij}\right]_{m \times n}$, the most frequently used matrix norms are the Frobenius norm

$$\|\mathbf{A}\|_F = \left( \sum_{i=1}^{m} \sum_{j=1}^{n} a_{ij}^2 \right)^{\frac{1}{2}} , \tag{A.19}$$

and the matrix $p$-norm

$$\|\mathbf{A}\|_p = \sup_{\boldsymbol{x}\neq 0} \frac{\|\mathbf{A}\boldsymbol{x}\|_p}{\|\boldsymbol{x}\|_p} = \max_{\|\boldsymbol{x}\|_p=1} \|\mathbf{A}\boldsymbol{x}\|_p, \tag{A.20}$$

where sup is the supreme operation.

The matrix 2-norm and the Frobenius norm are invariant with respect to orthogonal transforms, that is, for all orthogonal $\mathbf{Q}_1$ and $\mathbf{Q}_2$ of appropriate dimensions

$$\|\mathbf{Q}_1\mathbf{A}\mathbf{Q}_2\|_F = \|\mathbf{A}\|_F, \tag{A.21}$$

$$\|\mathbf{Q}_1\mathbf{A}\mathbf{Q}_2\|_2 = \|\mathbf{A}\|_2. \tag{A.22}$$

### Eigenvalue Decomposition

**Definition A.3** (*Eigenvalue Decomposition*) Given a square matrix $\mathbf{A} \in R^{n\times n}$, if there exists a scalar $\lambda$ and a nonzero vector $\boldsymbol{v}$ such that

$$\mathbf{A}\boldsymbol{v} = \lambda\boldsymbol{v}, \tag{A.23}$$

then $\lambda$ and $\boldsymbol{v}$ are, respectively, called an *eigenvalue* of $\mathbf{A}$ and its corresponding *eigenvector*. All the eigenvalues $\lambda_i, i = 1, \ldots, n$, can be obtained by solving the characteristic equation

$$\det(\mathbf{A} - \lambda\mathbf{I}) = 0, \tag{A.24}$$

where $\mathbf{I}$ is an $n \times n$ identity matrix. The set of all the eigenvalues is called the *spectrum* of $\mathbf{A}$.

If $\mathbf{A}$ is nonsingular, $\lambda_i \neq 0$. If $\mathbf{A}$ is symmetric, then all $\lambda_i$ are real. The maximum and minimum eigenvalues satisfy the Rayleigh quotient

$$\lambda_{\max}(\mathbf{A}) = \max_{\boldsymbol{v}\neq 0} \frac{\boldsymbol{v}^T\mathbf{A}\boldsymbol{v}}{\boldsymbol{v}^T\boldsymbol{v}}, \qquad \lambda_{\min}(\mathbf{A}) = \min_{\boldsymbol{v}\neq 0} \frac{\boldsymbol{v}^T\mathbf{A}\boldsymbol{v}}{\boldsymbol{v}^T\boldsymbol{v}}. \tag{A.25}$$

The trace of a matrix is equal to the sum of all its eigenvalues and the determinant of a matrix is equal to the product of its eigenvalues

$$\mathrm{tr}(\mathbf{A}) = \sum_{i=1}^{n} \lambda_i, \tag{A.26}$$

$$|\mathbf{A}| = \prod_{i=1}^{n} \lambda_i. \tag{A.27}$$

### Singular Value Decomposition

**Definition A.4** (*Singular Value Decomposition*) For a matrix $\mathbf{A} \in R^{m\times n}$, there exist real unitary matrices $\mathbf{U} = [\boldsymbol{u}_1, \boldsymbol{u}_2, \ldots, \boldsymbol{u}_m] \in R^{m\times m}$ and $\mathbf{V} = [\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_n] \in R^{n\times n}$ such that

$$\mathbf{U}^T\mathbf{A}\mathbf{V} = \mathbf{\Sigma}, \tag{A.28}$$

where $\mathbf{\Sigma} \in R^{m\times n}$ is a real pseudodiagonal $m \times n$ matrix with $\sigma_i$, $i = 1, \ldots, p$, $p = \min(m, n)$, $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_p \geq 0$, on the diagonal and zeros off the diagonal. $\sigma_i$'s are called the *singular values* of $\mathbf{A}$, and $\boldsymbol{u}_i$ and $\boldsymbol{v}_i$ are, respectively, called the *left singular vector* and *right singular vector* for $\sigma_i$. They satisfy the relations

$$\mathbf{A}\boldsymbol{v}_i = \sigma_i\boldsymbol{u}_i, \qquad \mathbf{A}^T\boldsymbol{u}_i = \sigma_i\boldsymbol{v}_i. \tag{A.29}$$

Accordingly, $\mathbf{A}$ can be written as

$$\mathbf{A} = \mathbf{U}\Sigma\mathbf{V}^T = \sum_{i=1}^{r} \lambda_i\boldsymbol{u}_i\boldsymbol{v}_i^T, \tag{A.30}$$

where $r$ is the cardinality of the smallest nonzero singular value. In the special case when $\mathbf{A}$ is a symmetric nonnegative definite matrix, $\mathbf{\Sigma} = \text{diag}\left(\lambda_1^{\frac{1}{2}}, \ldots, \lambda_p^{\frac{1}{2}}\right)$, where $\lambda_1 \geq \lambda_2 \geq \ldots \lambda_p \geq 0$ are the real eigenvalues of $\mathbf{A}$, $\boldsymbol{v}_i$ being the corresponding eigenvectors.

SVD is useful in many situations. The rank of $\mathbf{A}$ can be determined by the number of nonzero singular values. The power of $\mathbf{A}$ can be easily calculated by

$$\mathbf{A}^k = \mathbf{U}\mathbf{\Sigma}^k\mathbf{V}^T, \tag{A.31}$$

where $k$ is a positive integer. SVD is extensively applied in linear inverse problems. The pseudoinverse of $\mathbf{A}$ can then be described by

$$\mathbf{A}^\dagger = \mathbf{V}_r\mathbf{\Sigma}_r^{-1}\mathbf{U}_r^T, \tag{A.32}$$

where $\mathbf{V}_r$, $\mathbf{\Sigma}_r$, and $\mathbf{U}_r$ are the matrix partitions corresponding to the $r$ nonzero singular values.

The Frobenius norm can thus be calculated as

$$\|\mathbf{A}\|_F = \left(\sum_{i=1}^{p} \sigma_i^2\right)^{\frac{1}{2}}, \tag{A.33}$$

and the matrix 2-norm is calculated by

$$\|\mathbf{A}\|_2 = \sigma_1. \tag{A.34}$$

SVD requires a time complexity of $O(mn\min\{m,n\})$ for a dense $m \times n$ matrix. Common methods for computing the SVD of a matrix are standard eigensolvers such as QR iteration and Arnoldi/Lanczos iteration.

### QR Decomposition

For the full-rank or overdetermined linear LS case, $m \geq n$, (A.8) can also be solved by using QR decomposition procedure.

$\mathbf{A}$ is first factorized as

$$\mathbf{A} = \mathbf{QR}, \tag{A.35}$$

where $\mathbf{Q}$ is an $m \times m$ orthogonal matrix, that is, $\mathbf{Q}^T\mathbf{Q} = \mathbf{I}$, and $\mathbf{R} = \begin{bmatrix} \overline{\mathbf{R}} \\ \mathbf{0} \end{bmatrix}$ is an $m \times n$ upper triangular matrix with $\overline{\mathbf{R}} \in R^{n\times n}$.

Inserting (A.35) into (A.8) and premultiplying by $\mathbf{Q}^T$, we have

$$\mathbf{R}\boldsymbol{x} = \mathbf{Q}^T\boldsymbol{b}. \tag{A.36}$$

Denoting $\mathbf{Q}^T\boldsymbol{b} = \begin{bmatrix} \overline{\boldsymbol{b}} \\ \widetilde{\boldsymbol{b}} \end{bmatrix}$, where $\overline{\boldsymbol{b}} \in R^n$ and $\widetilde{\boldsymbol{b}} \in R^{m-n}$, we have

$$\overline{\mathbf{R}}\boldsymbol{x} = \overline{\boldsymbol{b}}. \tag{A.37}$$

Since $\overline{\mathbf{R}}$ is a triangular matrix, $\boldsymbol{x}$ can be easily solved using backward substitution. This is the procedure used in the GSO procedure.

When $\text{rank}(\mathbf{A}) < n$, the rank-deficient LS problem has an infinite number of solutions, QR decomposition does not necessarily produce an orthonormal basis for $\text{range}(\mathbf{A}) = \{\boldsymbol{y} \in R^m : \boldsymbol{y} = \mathbf{A}\boldsymbol{x} \text{ for some } \boldsymbol{x} \in R^n\}$. QR-cp can be applied to produce an orthonormal basis for $\text{range}(\mathbf{A})$.

As a basic method for computing SVD, QR decomposition itself can be computed by means of the Givens rotation, the Householder transform, or GSO.

### Condition Numbers

**Definition A.5** (*Condition Number*) The condition number of a matrix $\mathbf{A} \in R^{m\times n}$ is defined by

$$\text{cond}_p(\mathbf{A}) = \|\mathbf{A}\|_p \left\|\mathbf{A}^\dagger\right\|_p, \tag{A.38}$$

where $p$ can be selected as 1, 2, $\infty$, Frobenius, or any other norm.

The relation, $\text{cond}(\mathbf{A}) \geq 1$, always holds. Matrices with small condition numbers are well conditioned, while matrices with large condition number are poorly con-

ditioned or ill-conditioned. The condition number is especially useful in numerical computation, where ill-conditioned matrices are sensitive to rounding errors.

For the $L_2$-norm,

$$\text{cond}_2(\mathbf{A}) = \frac{\sigma_1}{\sigma_p}, \tag{A.39}$$

where $p = \min(m, n)$.

### Householder Reflections and Givens Rotations

Orthogonal transforms play an important role in the matrix computation such as EVD, SVD, and QR decomposition. The Householder reflection, also termed the *Householder transform*, and Givens rotations, also called the *Givens transform*, are two basic operations in the orthogonalization process. These operations are easily constructed, and they introduce zeros in a vector so as to simplify matrix computations. The Householder reflection is exceedingly efficient for annihilating all but the first entry of a vector, while the Givens rotation is more effective to transform a specified entry of a vector into zero.

Let $\boldsymbol{v} \in R^n$ be nonzero. The Householder reflection is defined as a rank-one modification to the identity matrix

$$\mathbf{P} = \mathbf{I} - 2\frac{\boldsymbol{v}\boldsymbol{v}^T}{\boldsymbol{v}^T\boldsymbol{v}}. \tag{A.40}$$

The Householder matrix $\mathbf{P} \in R^{n\times n}$ is symmetric and orthogonal. $\boldsymbol{v}$ is called a *Householder vector*. The Householder transform of a matrix $\mathbf{A}$ is given by $\mathbf{PA}$. By specifying the form of the transformed matrix, one can find a suitable Householder vector $\boldsymbol{v}$. For example, one can define a Householder vector as $\boldsymbol{v} = \boldsymbol{u} - \alpha\boldsymbol{e}_1$, where $\boldsymbol{u} \in R^m$ is an arbitrary vector of length $|\alpha|$ and $\boldsymbol{e}_1 \in R^m$, wherein only the first entry is unity, all the other entries being zero. In this case, $\mathbf{P}\boldsymbol{x}$ becomes a vector with only the first entry nonzero, where $\boldsymbol{x} \in R^n$ is a nonzero vector.

The Givens rotation $\mathbf{G}(i, k, \theta)$ is a rank-two correction to the identity matrix $\mathbf{I}$. It modifies $\mathbf{I}$ by setting the $(i, i)$th entry as $\cos\theta$, the $(i, k)$th entry as $\sin\theta$, the $(k, i)$th entry as $-\sin\theta$, and the $(k, k)$th entry as $\cos\theta$. The Givens transform $\mathbf{G}(i, k, \theta)\boldsymbol{x}$ applies a counterwise rotation of $\theta$ radians in the $(i, k)$ coordinate plane. One can specify an entry in a vector to zero by applying the Givens rotation and then calculate the rotation angle $\theta$.

### Matrix Inversion Lemma

The matrix inversion lemma is also called the *Sherman–Morrison–Woodbury formula*. It is useful in deriving many iterative algorithms. Assume that the relationship between the matrix $\mathbf{A} \in R^{n\times n}$ at iterations $t$ and $t + 1$ is given as

$$\mathbf{A}(t+1) = \mathbf{A}(t) + \Delta\mathbf{A}(t). \tag{A.41}$$

If $\Delta\mathbf{A}(t)$ can be expressed as $\mathbf{U}\mathbf{V}^T$, where $\mathbf{U} \in R^{n\times m}$ and $\mathbf{V} \in R^{m\times n}$, it is referred to as a *rank-m update*. The matrix inversion lemma gives [7]

$$\begin{aligned}\mathbf{A}^{-1}(t+1) &= \mathbf{A}^{-1}(t) - \Delta\mathbf{A}^{-1}(t) \\ &= \mathbf{A}^{-1}(t) - \mathbf{A}^{-1}(t)\mathbf{U}\left(\mathbf{I} + \mathbf{V}^T\mathbf{A}^{-1}(t)\mathbf{U}\right)^{-1}\mathbf{V}^T\mathbf{A}^{-1}(t),\end{aligned} \tag{A.42}$$

where both $\mathbf{A}(t)$ and $\left(\mathbf{I} + \mathbf{V}^T\mathbf{A}^{-1}(t)\mathbf{U}\right)$ are assumed to be nonsingular. Thus, a rank-$m$ correction to a matrix results in a rank-$m$ correction to its inverse.

Some modifications to the formula are available, and one popular update is given here. If $\mathbf{A}$ and $\mathbf{B}$ are two positive-definite matrices, which have the relation

$$\mathbf{A} = \mathbf{B}^{-1} + \mathbf{C}\mathbf{D}\mathbf{C}^T, \tag{A.43}$$

where $\mathbf{C}$ and $\mathbf{D}$ are also matrices. The matrix inversion lemma gives the inverse of $\mathbf{A}$ as

$$\mathbf{A}^{-1} = \mathbf{B} - \mathbf{B}\mathbf{C}(\mathbf{D} + \mathbf{C}^T\mathbf{B}\mathbf{C})^{-1}\mathbf{C}^T\mathbf{B}. \tag{A.44}$$

**Partial Least Squares Regression**

Partial LS regression [17] is a statistical method for modeling a linear relationship between two datasets $\mathcal{X}$ and $\mathcal{Y}$. It finds projection vectors by maximizing the linear association between two latent components which are the projection of two deflation datasets.

It is a robust, iterative method that avoids matrix inversion for underconstrained datasets by decomposing the multivariate regression problem into successive univariate regressions. Partial LS iteratively chooses its projection directions according to the direction of maximum correlation between the (current residual) input and the output. Computation of each projection direction is $O(d)$ for $d$ dimensions of the data. Successive iterations create orthogonal projection directions by removing the subspace of the input data used in the previous projection. The number of projection directions found by partial LS is bound only by the dimensionality of the data, with each univariate regression on successive projection components further reducing the residual error. Using all $d$ projections leads to ordinary LS regression. If the distribution of the input data is spherical, then partial LS requires only a single projection to optimally reconstruct the output. Partial LS in statistics is equivalent to the CG method [11].

## A.2 Data Preprocessing

### Linear Scaling and Data Whitening

By linear normalization, all the raw data can be brought in the vicinity of an average value. For a one-dimensional dataset, $\{x_i | \, i = 1, \ldots, N\}$, the mean $\hat{\mu}$ and variance $\hat{\sigma}^2$ are estimated by

$$\hat{\mu} = \frac{1}{N} \sum_{i=1}^{N} x_i, \tag{A.45}$$

$$\hat{\sigma}^2 = \frac{1}{N-1} \sum_{i=1}^{N} \left( x_i - \hat{\mu} \right)^2 . \tag{A.46}$$

The transformed data are now defined by

$$\widetilde{x}_i = \frac{x_i - \hat{\mu}}{\hat{\sigma}}. \tag{A.47}$$

The transformed dataset $\{\widetilde{x}_i | \, i = 1, \ldots, N\}$ has zero mean and unit standard deviation.

When the raw dataset $\{\boldsymbol{x}_i | i = 1, \ldots, N\}$ is composed of vectors, accordingly, the mean vector $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$ are calculated by

$$\hat{\boldsymbol{\mu}} = \frac{1}{N} \sum_{i=1}^{N} \boldsymbol{x}_i, \tag{A.48}$$

$$\widehat{\boldsymbol{\Sigma}} = \frac{1}{N-1} \sum_{i=1}^{N} \left( \boldsymbol{x}_i - \hat{\boldsymbol{\mu}} \right) \left( \boldsymbol{x}_i - \hat{\boldsymbol{\mu}} \right)^T . \tag{A.49}$$

Equations (A.46) and (A.49) are, respectively, the unbiased estimates of the variance and the covariance matrix. When the factor $\frac{1}{N-1}$ is replaced by $\frac{1}{N}$, the estimates for $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ are the ML estimates. The ML estimates for variance and covariance are biased.

New input vectors can be defined by the linear transformation

$$\widetilde{\boldsymbol{x}}_i = \Lambda^{-\frac{1}{2}} \mathbf{U}^T \left( \boldsymbol{x}_i - \hat{\boldsymbol{\mu}} \right), \tag{A.50}$$

where $\mathbf{U} = [\boldsymbol{u}_1, \ldots, \boldsymbol{u}_M]$, $\Lambda = \text{diag}\,(\lambda_1, \ldots, \lambda_M)$, $M$ is the dimension of data vectors, and $\lambda_i$ and $\boldsymbol{u}_i$ are the eigenvalues and the corresponding eigenvectors of $\boldsymbol{\Sigma}$, which satisfy

$$\boldsymbol{\Sigma} \boldsymbol{u}_i = \lambda_i \boldsymbol{u}_i. \tag{A.51}$$

The new dataset $\{\widetilde{\boldsymbol{x}}_i\}$ has zero mean, and its covariance matrix is the identity matrix [1]. The above process is also called *data whitening*.

### Gram–Schmidt Orthonormalization Transform

Ill-conditioning is usually measured for a data matrix $\mathbf{A}$ by its condition number $\rho$, defined as $\rho(\mathbf{A}) = \frac{\sigma_{\max}}{\sigma_{\min}}$, where $\sigma_{\max}$ and $\sigma_{\min}$ are, respectively, the maximum and minimum singular values of $\mathbf{A}$. In the batch LS algorithm, the information matrix $\mathbf{A}^T\mathbf{A}$ needs to be manipulated. Since $\rho\left(\mathbf{A}^T\mathbf{A}\right) = \rho(\mathbf{A})^2$, the effect of ill-conditioning on parameter estimation will be more severe. Orthogonal decomposition is a well-known technique to eliminate ill-conditioning.

The GSO procedure starts with QR decomposition of the full feature matrix. Denote

$$\mathbf{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N], \tag{A.52}$$

where the $i$th pattern $\boldsymbol{x}_i = \left(x_{i,1}, x_{i,2}, \ldots, x_{i,J}\right)^T$, $x_{i,j}$ denotes the $j$th component of $\boldsymbol{x}_i$, and $J$ is the dimensions of the raw data. We then represent $\mathbf{X}^T$ by

$$\mathbf{X}^T = \left[\boldsymbol{x}^1, \boldsymbol{x}^2, \ldots, \boldsymbol{x}^J\right], \tag{A.53}$$

where $\boldsymbol{x}^j = \left(x_{1,j}, x_{2,j}, \ldots, x_{N,j}\right)^T$.

QR decomposition is performed on $\mathbf{X}^T$

$$\mathbf{X}^T = \mathbf{QR}, \tag{A.54}$$

where $\mathbf{Q}$ is an orthonormal matrix, that is, $\mathbf{Q}^T\mathbf{Q} = \mathbf{I}_J$, $\mathbf{Q} = \left[\boldsymbol{q}_1, \boldsymbol{q}_2, \ldots, \boldsymbol{q}_J\right]$, $\boldsymbol{q}_i = \left(q_{i,1}, q_{i,2}, \ldots, q_{i,N}\right)^T$, $q_{i,j}$ denoting the $j$th component of $\boldsymbol{q}_i$, and $\mathbf{R}$ is an upper triangular matrix. QR decomposition can be performed by the Householder transform or Givens rotation [7], which is suitable for hardware implementation.

The GSO procedure is given as

$$\boldsymbol{q}_1 = \boldsymbol{x}^1, \tag{A.55}$$

$$\boldsymbol{q}_k = \boldsymbol{x}^k - \sum_{i=1}^{k-1} \alpha_{ik}\boldsymbol{q}_i, \tag{A.56}$$

$$\alpha_{ik} = \begin{cases} \frac{(\boldsymbol{x}^k)^T\boldsymbol{q}_i}{\boldsymbol{q}_i^T\boldsymbol{q}_i}, & \text{for } \; i = 1, 2, \ldots, k-1 \\ 1, & \text{for } \; i = k \\ 0, & \text{for } \; i > k \end{cases}. \tag{A.57}$$

Thus $\boldsymbol{q}_k$ is a linear combination of $\boldsymbol{x}^1, \ldots, \boldsymbol{x}^k$, and the Gram–Schmidt features $\boldsymbol{q}_1, \ldots, \boldsymbol{q}_k$ and the vectors $\boldsymbol{x}^1, \ldots, \boldsymbol{x}^k$ are one-to-one mappings, for $1 \le k \le J$. GSO transform can be used for feature subset selection; it inherits the compactness of

the orthogonal representation and at the same time provides features retaining their original meaning.

## A.3 Stability of Dynamic Systems

For a dynamic system described by a set of ordinary differential equations, the stability of the system can be examined by Lyapunov's second theorem or the Lipschitz condition.

Lyapunov theorem is a sufficient but not a necessary tool for proving the stability of an equilibrium of a dynamic system. The method is dependent on finding a Lyapunov function for the equilibrium. It is especially important for analyzing the stability of recurrent networks and ordinary differential equations.

**Theorem A.1** (Lyapunov Theorem) *Consider a function $L(\boldsymbol{x})$. Define a region $\Omega$, where any point $\boldsymbol{x} \in \Omega$ satisfies $L(\boldsymbol{x}) < c$ for a constant $c$, with the boundary of $\Omega$ given by $L(\boldsymbol{x}) = c$, such that*

- $\frac{dL(\boldsymbol{x})}{dt} < 0, \forall \boldsymbol{x}, \boldsymbol{x}^* \in \Omega, \boldsymbol{x} \neq \boldsymbol{x}^*$.
- $\frac{dL(\boldsymbol{x}^*)}{dt} = 0$.

*Then, the equilibrium point $\boldsymbol{x} = \boldsymbol{x}^*$ is asymptotically stable, with a domain of attraction $\Omega$.*

**Theorem A.2** (Lyapunov's Second Theorem) *For a dynamic system described by a set of differential equations*

$$\frac{d\boldsymbol{x}}{dt} = \boldsymbol{f}(\boldsymbol{x}), \tag{A.58}$$

*where $\boldsymbol{x} = (x_1(t), x_2(t), \ldots, x_n(t))^T$ and $\boldsymbol{f} = (f_1, f_2, \ldots, f_n)^T$, if there exists a positive-definite function $E = E(\boldsymbol{x})$, called a* Lyapunov function *or* energy function*, such that*

$$\frac{dE}{dt} = \sum_{j=1}^{n} \frac{\partial E}{\partial x_j} \frac{dx_j}{dt} \leq 0 \tag{A.59}$$

*with $\frac{dE}{dt} = 0$ only for $\frac{d\boldsymbol{x}}{dt} = \boldsymbol{0}$, then the system is stable, and the trajectories $\boldsymbol{x}$ will asymptotically converge to stationary points as $t \to \infty$.*

The stationary points are also known as *equilibrium points* and *attractors*. The crucial step in applying the Lyapunov's second theorem is to find a suitable energy function.

**Theorem A.3** (Lipschitz Condition) *For a dynamic system described by (A.58), a sufficient condition that guarantees the existence and uniqueness of the solution is given by the Lipschitz condition*

$$\|\boldsymbol{f}(\boldsymbol{x}_1) - \boldsymbol{f}(\boldsymbol{x}_2)\| \leq \gamma \|\boldsymbol{x}_1 - \boldsymbol{x}_2\|, \tag{A.60}$$

*where $\gamma$ is any positive constant, called* Lipschitz constant*, and $\boldsymbol{x}_1$, $\boldsymbol{x}_2$ are any two variables in the domain of the function vector $\boldsymbol{f}$. $\boldsymbol{f}(\boldsymbol{x})$ is said to be Lipschitz continuous.*

If $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ are in some neighborhood of $\boldsymbol{x}$, then they are said to satisfy the Lipschitz condition locally and will reach a unique solution in the neighborhood of $\boldsymbol{x}$. The unique solution is a trajectory that will converge to an attractor asymptotically and reach it only at $t \to \infty$.

## A.4 Probability Theory and Stochastic Processes

### Conditional Probability

For two statements (or propositions) $A$ and $B$, one writes $A|B$ to denote the situation that $A$ is true subject to the condition that $B$ is true. The probability of $A|B$, called *conditional probability*, is denoted by $P(A|B)$. This gives a measure for the plausibility of the statement $A|B$.

### Gaussian Distribution

The Gaussian distribution, known as the *normal distribution*, is the most common assumption for error distribution. The pdf of the normal distribution is defined as

$$p(x) = \frac{1}{\sigma\sqrt{2\pi}} \mathrm{e}^{-\frac{(x-\mu)^2}{2\sigma^2}}, \quad x \in R, \tag{A.61}$$

where $\mu$ is the mean and $\sigma > 0$ is the standard deviation. For the Gaussian distribution, 99.73 % of the data are within the range of $[\mu - 3\sigma, \mu + 3\sigma]$. The Gaussian distribution has its first-order moment as $\mu$, second-order moment as $\sigma^2$, and higher order moments as zero. If $\mu = 0$ and $\sigma = 1$, the distribution is called the *standard normal distribution*. The pdf is also known as the *likelihood function*. An ML estimator is a set of values $(\mu, \sigma)$ that maximizes the likelihood function for a fixed value of $x$.

The cumulative distribution function (cdf) is defined as the probability that a random variable is less than or equal to a value $x$, that is,

$$F(x) = \int_{-\infty}^{x} p(t)dt. \tag{A.62}$$

The standard normal cdf, conventionally denoted $\Phi$, is given by setting $\mu = 0$ and $\sigma = 1$. The standard normal cdf is usually expressed by

$$\Phi(x) = \frac{1}{2}\left[1 + \text{erf}\left(\frac{x}{\sqrt{2}}\right)\right], \tag{A.63}$$

where the error function $\text{erf}(x)$ is a nonelementary function, which is defined by

$$\text{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x \text{e}^{-t^2} dt. \tag{A.64}$$

When vector $\boldsymbol{x} \in R^n$, the pdf of the normal distribution is then defined by

$$p(\boldsymbol{x}) = \frac{1}{(2\pi)^{\frac{n}{2}} |\boldsymbol{\Sigma}|} \text{e}^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})}, \tag{A.65}$$

where $\boldsymbol{\mu}$ is the mean vector and $\boldsymbol{\Sigma}$ is the covariance matrix.

The Gaussian distribution is only one of the canonical exponential distributions, and it is suitable for describing real-value data. In the case of binary-valued, integer-valued, or nonnegative data, the Gaussian assumption is inappropriate, and a family of exponential distributions can be used. For example, Poisson's distribution is better suited for integer data and the Bernoulli distribution to binary data, and an exponential distribution to nonnegative data.

### Cauchy Distribution

The Cauchy distribution, also known as the *Cauchy–Lorentzian distribution*, is another popular data distribution model. The density of the Cauchy distribution is defined as

$$p(x) = \frac{1}{\pi\sigma\left[1 + \left(\frac{x-\mu}{\sigma}\right)^2\right]}, \quad x \in R, \tag{A.66}$$

where $\mu$ specifies the location of the peak and $\sigma$ specifies the half-width at the half-maximum. When $\mu = 0$ and $\sigma = 1$, the distribution is called the *standard Cauchy distribution*.

Accordingly, the cdf of the Cauchy distribution is calculated by

$$F(x) = \frac{1}{\pi} \arctan\left(\frac{x-\mu}{\sigma}\right) + \frac{1}{2}. \tag{A.67}$$

None of the moments is defined for the Cauchy distribution. The median of the distribution is equal to $\mu$. Compared to the Gaussian distribution, the Cauchy distribution has a longer tail; this makes it more valuable in stochastic search algorithms by searching larger subspaces in the data space.

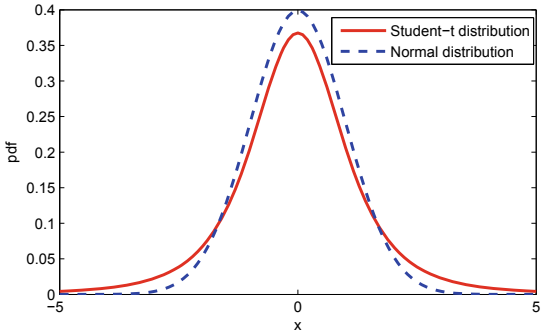


**Fig. A.1** The Student-$t$ distribution with $\nu = 4$ and standard normal distribution

### Student-$t$ Models

The Student-$t$ pdf is given by

$$p(x) = \frac{\Gamma\left(\frac{\nu+1}{2}\right)}{\sqrt{\pi\nu}\Gamma\left(\frac{\nu}{2}\right)} \frac{1}{\left(1+\frac{x^2}{\nu}\right)^{\frac{\nu+1}{2}}}, \tag{A.68}$$

where $\Gamma(\cdot)$ is the Gamma function, and $\nu$ is the degrees of freedom. The Gaussian distribution is a particular $t$ distribution with $\nu = \infty$.

For a random sample of size $n$ from a normal distribution with mean $\mu$, we get the statistic

$$t = \frac{\overline{x} - \mu}{\sigma/\sqrt{n}}, \tag{A.69}$$

where $\overline{x}$ is the sample mean and $\sigma$ is the sample standard deviation. $t$ has a Student-$t$ distribution with $n-1$ degrees of freedom.

The Student-$t$ distribution has a longer tail than the Gaussian distribution. The pdfs of the Student-$t$ distribution and the normal distribution are plotted in Fig. A.1.

### Kullback–Leibler Divergence

Mutual information between two signals $\boldsymbol{x}$ and $\boldsymbol{y}$ is characterized by calculating the cross-entropy, known as *Kullback–Leibler divergence*, between the joint pdf $p(\boldsymbol{x}, \boldsymbol{y})$ of $\boldsymbol{x}$ and $\boldsymbol{y}$ and the product of the marginal pdfs $p(\boldsymbol{x})$ and $p(\boldsymbol{y})$

$$I(\boldsymbol{x}; \boldsymbol{y}) = \int p(\boldsymbol{x}, \boldsymbol{y}) \ln \frac{p(\boldsymbol{x}, \boldsymbol{y})}{p(\boldsymbol{x})p(\boldsymbol{y})} d\boldsymbol{x}d\boldsymbol{y}. \tag{A.70}$$

This may be implemented by estimating the pdfs in terms of the cumulants of the signals. This approach requires the numerical estimation of the joint and marginal densities.

### Cumulants

For random variables $X_1, \ldots, X_4$, second-order cumulants are defined as $\text{cum}(X_1, X_2) = E[\overline{X}_1 \overline{X}_2]$, where $\overline{X}_i = X_i - E[X_i]$, and the fourth-order cumulants are [4]

$$\begin{aligned}\text{cum}(X_1, X_2, X_3, X_4) &= E\left[\overline{X}_1 \overline{X}_2 \overline{X}_3 \overline{X}_4\right] - E\left[\overline{X}_1 \overline{X}_2\right] E\left[\overline{X}_3 \overline{X}_4\right] \\ &\quad - E\left[\overline{X}_1 \overline{X}_3\right] E\left[\overline{X}_2 \overline{X}_4\right] - E\left[\overline{X}_1 \overline{X}_4\right] E\left[\overline{X}_2 \overline{X}_3\right]. \end{aligned} \tag{A.71}$$

The variance and kurtosis of a real random variable $X$ are defined by

$$\text{var}(X) = \sigma^2(X) = \text{cum}(X, X) = E\left[\overline{X}^2\right], \tag{A.72}$$

$$\text{kurt}(X) = \text{cum}(X, X, X, X) = E\left[\overline{X}^4\right] - 3E^2\left[\overline{X}^2\right]. \tag{A.73}$$

They are the second- and fourth-order *autocumulants*. A cumulant having at least two different variables is called a *cross-cumulant*.

### Markov Processes, Markov Chains and Markov-Chain Analysis

Markov processes constitute the best-known class of stochastic processes. A Markov process has a limited memory. Assume a stochastic process $\{X(t) : t \in \mathcal{T}\}$, where $t$ is time, $X(t)$ is a state in the state space $\mathcal{S}$. A Markov process is defined as a stochastic process that satisfies the relation characterized by the conditional distribution

$$\begin{aligned} P\left[X(t_0 + t_1) \leq x \middle| X(t_0) = x_0, X(\tau) = x_\tau, -\infty < \tau < t_0\right] \\ = P\left[X(t_0 + t_1) \leq x \middle| X(t_0) = x_0\right] \end{aligned} \tag{A.74}$$

for any value of $t_0$ and for $t_1 > 0$. The future distribution of the process is determined by the present value of $X(t_0)$ only. This latter property is known as the *Markov property*.

When $\mathcal{T}$ and $\mathcal{S}$ are discrete, a Markov process is called a *Markov chain*. Conventionally, time is indexed using integers, and a Markov chain is a set of random variables that satisfy

$$\begin{aligned} P\left[X_n = x_n \middle| X_{n-1} = x_{n-1}, X_{n-2} = x_{n-2}, \ldots\right] \\ = P\left[X_n = x_n \middle| X_{n-1} = x_{n-1}\right]. \end{aligned} \tag{A.75}$$

This definition can be extended for multistep Markov chains, where a chain state has conditional dependency on only a finite number of its previous states.

For a Markov chain, $P\left[X_n = j | X_{n-1} = i\right]$ is the transition probability of state $i$ to $j$ at time $n-1$. If

$$P\left[X_n = j | X_{n-1} = i\right] = P\left[X_{n+m} = j | X_{n+m-1} = i\right], \quad m \geq 0, \ i, j \in \mathcal{S}, \tag{A.76}$$

the chain is said to be *time homogeneous*. In this case, one can denote

$$P_{i,j} = P\left[X_n = j | X_{n-1} = i\right] \tag{A.77}$$

and the transition probabilities can be represented by a matrix, called the *transition matrix*, $\mathbf{P} = \left[P_{i,j}\right]$, where $i, j = 0, 1, \ldots$. For finite $\mathcal{S}$, $\mathbf{P}$ has a finite dimension. An important property of Markov chains is their time homogeneity, which means that their transition probabilities $p_{ij}$ do not depend on time.

In Markov chain analysis, the transition probability after $k$ step transitions is $\mathbf{P}^k$. The *stationary distribution* or *steady-state distribution* is a vector that satisfies

$$\mathbf{P}^T \boldsymbol{\pi}^* = \boldsymbol{\pi}^*. \tag{A.78}$$

That is, $\boldsymbol{\pi}^*$ is the left eigenvector of $\mathbf{P}$ corresponding to the eigenvalue 1.

If $\mathbf{P}$ is irreducible and aperiodic, that is, every state is accessible from every other state and in the process none of the states repeats itself periodically, then $\mathbf{P}^k$ converges elementwise to a matrix each row of which is the unique stationary distribution $\boldsymbol{\pi}^*$, with

$$\lim_{k \to \infty} \left(\mathbf{P}^k\right)^T \boldsymbol{\pi} = \boldsymbol{\pi}^*. \tag{A.79}$$

Many modeling applications are Markovian, and Markov chain analysis is widely used for convergence analysis for algorithms.

*Example A.1* The transition probability matrix corresponding to the graph in Fig. A.2 is given by

$$\mathbf{P} = \begin{bmatrix} 0.3 & 0.5 & 0.2 & 0 \\ 0 & 0 & 0.2 & 0.8 \\ 0 & 0 & 0.3 & 0.7 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

Probabilities of transitions from state $i$ to all other states add up to one, i.e., $\sum_{j=1}^{N} P_{ij} = 1$.

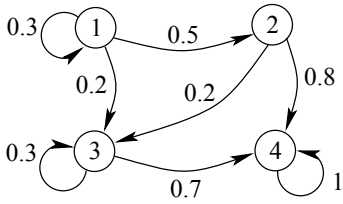


**Fig. A.2** State diagram of a Markov chain

## A.5 Numerical Optimization Techniques

Although optimization problems can be solved analytically in some cases, numerical optimization techniques are usually more powerful and are also indispensible for all disciplines in science and engineering. Optimization problems discussed in this book are mainly unconstrained continuous optimization problems, COPs, and quadratic programming problems. To deal with constraints, the KKT theorem, as a generalization to the Lagrange multiplier method, introduces a slack variable into each inequality constraint before applying the Lagrange multiplier method. The conditions derived from the procedure are known as the *KKT conditions* [5].

### A Brief Taxonomy

Optimization techniques can generally be divided into derivative methods and non-derivative methods, depending on whether or not derivatives of the objective function are required for the calculation of the optimum. Derivative methods can be either gradient-search methods or second-order methods. Gradient-search methods include the gradient-descent method, CG methods, and the natural-gradient method. The gradient descent is also known as *steepest descent*. It searches for a local minimum by taking steps along the negative direction of the gradient of the function. If the steps are along the positive direction of the gradient, the method is known as *gradient ascent* or *steepest ascent*. The gradient-descent method is credited to Cauchy. Examples of second-order methods are Newton's method, the Gauss–Newton method, quasi-Newton methods, the trust-region method, and the LM method. CG methods can also be viewed as a reduced form of the quasi-Newton method, with systematic reinitializations of $\mathbf{H}_t$ to the identity matrix.

Derivative methods can also be classified into *model-based* and *metric-based* methods. Model-based methods improve the current point by a local approximating model. Newton and quasi-Newton methods are model-based methods. Metric-based methods perform a transformation of the variables and then apply a gradient-search method to improve the point. The steepest descent method, quasi-Newton methods, and CG methods belong to this latter category.

The vanilla stochastic gradient-descent method converges slower than the gradient-descent method does. A damped step size is used to guarantee convergence, due to the large variance of stochastic gradient. Nesterov's accelerated gradient descent [9]

uses a proper momentum to accelerate the convergence rate of gradient descent to optimal. For smooth convex optimization, the sequence recovers the minimum at a quadratic convergence rate, with the same computational complexity as the vanilla gradient-descent method.

Coordinate-descent method [6] provides an efficient general solver. The method cyclically chooses one variable at a time and performs a simple analytical update. Stochastic coordinate-descent methods update a single randomly chosen coordinate at a time by moving in the direction of the negative partial derivative. One way to update more coordinates at each iteration is via partitioning the coordinates into blocks and operating on a single randomly chosen block at a time [10]. Theory was developed for methods that update a random subset of blocks of coordinates at a time [12]. Block coordinate-descent method may cycle and stagnate when being applied to solve non-convex problems.

Among quasi-Newton methods, limited memory BFGS reduces the computational complexity in each iteration to the same order as gradient descent. Stochastic quasi-Newton methods such as online BFGS [3, 13] and online limited memory BFGS [13] extend BFGS by using stochastic gradients both as descent directions and constituents of Hessian estimates. A sublinear convergence of online limited memory BFGS for solving optimization problems with stochastic objectives is established in [8].

Typical nonderivative methods for multivariable functions are random-restart hill climbing, simulated annealing, evolutionary algorithms, random search, many heuristic methods, and their hybrids. Hill climbing attempts to optimize a discrete or continuous function for a local optimum. When operating on continuous space, it is called *gradient ascent*. Other nonderivative search methods include univariant search parallel to an axis, sequential simplex method, and acceleration methods in direct search such as Hooke–Jeeves method, Powell's method, and Rosenbrock's method. Hooke–Jeeves method accelerates in distance, Powell's method accelerates in direction, and Rosenbrock's method accelerates in both direction and distance. Interior-point methods represent state-of-the-art techniques for solving linear, quadratic, and nonlinear optimization programs. Standard LP solvers are the simplex algorithm and the interior-point algorithm. LP can be solved using IBM ILOG CPLEX Optimizer (https://www.ibm.com/analytics/data-science/prescriptive-analytics/cplex-optimizer).

### Lagrange Multiplier Method

The Lagrange multiplier method can be used to analytically solve continuous function optimization subject to equality constraints [5]. Let $f(\boldsymbol{x})$ be the objective function and $h_i(\boldsymbol{x}) = 0, i = 1, \ldots, m$, be the constraints. The Lagrange function can be constructed as

$$L(\boldsymbol{x}; \lambda_1, \ldots, \lambda_m) = f(\boldsymbol{x}) + \sum_{i=1}^{m} \lambda_i h_i(\boldsymbol{x}), \tag{A.80}$$

where $\lambda_i$, $i = 1, \ldots, m$, are called the *Lagrange multipliers*.

The constrained optimization problem is converted into an unconstrained optimization problem: Optimize $L(\boldsymbol{x}; \lambda_1, \ldots, \lambda_m)$. By setting

$$\frac{\partial}{\partial \boldsymbol{x}} L(\boldsymbol{x}; \lambda_1, \ldots, \lambda_m) = 0, \tag{A.81}$$

$$\frac{\partial}{\partial \lambda_i} L(\boldsymbol{x}; \lambda_1, \ldots, \lambda_m) = 0, \quad i = 1, \ldots, m, \tag{A.82}$$

and solving the resulting set of equations, we can obtain the $\boldsymbol{x}$ position at the extremum of $f(\boldsymbol{x})$ under the constraints.

### Line Search

The popular quasi-Newton and CG methods implement a line search at each iteration. The efficiency of the line search method significantly affects the performance of these methods.

Bracketing and sectioning are two elementary operations for any line search method. A bracket is an interval $(\alpha_1, \alpha_2)$ that contains an optimal value of $\alpha$. Any three values of $\alpha$ that satisfy $\alpha_1 < \alpha_2 < \alpha_3$ form a bracket when the values of the function $f(\alpha)$ satisfies $f(\alpha_2) \leq \min(f(\alpha_1), f(\alpha_3))$. Sectioning is applied to reduce the size of the bracket at a uniform rate. Once a bracket is identified, it can be contracted by using sectioning or interpolation techniques or their combinations. Popular sectioning techniques are the gloden-section search, the Fibonacci search, the secant method, Brent's quadratic approximation, and Powell's quadratic convergence search without derivatives. The Newton–Raphson search is an analytical line search technique based on the gradient of the objective function. Wolfe's conditions are two inequality conditions for performing inexact line search. Wolfe's conditions enable an efficient selection of the step size without minimizing $f(\alpha)$.

### Semidefinite Programming

For a convex optimization problem, a local solution is the global optimal solution. The semidefinite programming (SDP) problem is a convex optimization problem with a linear objective, and linear matrix inequality and affine equality constraints. It optimizes convex cost functions over the convex cone of positive semidefinite matrices. There exist interior-point algorithms to solve SDP problems with good theoretical and practical computational efficiency. One very useful tool to reduce a problem to an SDP problem is the so-called Schur complement lemma. The SDP problem can be efficiently solved using standard SDP solvers such as a C library for semidefinite programming [2], and the MATLAB packages CVX (http://cvxr.com/cvx/), SeDuMi (http://sedumi.ie.lehigh.edu/) [15], and SDPT3 [16].

Many stability or constrained optimization problems including the SDP problem can be converted into a quasi-convex optimization problem in the form of a linear matrix inequality (LMI)-based optimization problem. The LMI-based optimization problem can be efficiently solved by interior-point methods by using MATLAB LMI Control Toolbox. For verifying the stability of delayed neural networks, a Lyapunov function is usually constructed based on the LMI approach.

The constrained concave–convex procedure (CCCP) of [18] is used for solving the non-convex optimization problem. CCCP essentially decomposes a non-convex function into a convex component and a concave component. At each iteration, the concave part is replaced by a linear function (namely, the tangential approximation at the current point) and the sum of this linear function and the convex part is minimized to get the next iteration.

## A.6 Classification Measures

When dealing with imbalanced datasets, overall accuracy is a biased measure of classifier goodness. Instead, the confusion matrix, and the true positive (TP) and false positive (FP) are better indications of classifier performance. Referred to as matching matrix in unsupervised learning, a confusion matrix provides a visual representation of actual versus predicted class accuracies.

Accuracy is the number of data points correctly classified:

$$Accuracy = \frac{TP + TN}{TP + TN + FN + FP}, \tag{A.83}$$

where the positive class is the class that is important and usually is the minority class, true positive (TP) is the number of data points from the positive class that are correctly classified, false positive (FP) is the number of data points from the negative class that are predicted to be in the positive class, true negative (TN) is the number of data points from the negative class that are correctly classified, and false negative (FN) is the number of data points from the positive class that are predicted to be in the negative class.

Sensitivity, true positive rate (TPR) or recall rate (RR) measures how well a classifier classifies data points in the positive class:

$$Sensitivity = \frac{TP}{TP + FN}. \tag{A.84}$$

Specificity or true negative rate (TNR) measures how well a classifier classifies data points in the negative class:

$$Specificity = \frac{TN}{TN + FP}. \tag{A.85}$$

Precision shows how exactly the classifier predicts the positive data among actual positive data:

$$Precision = \frac{TP}{TP + FP}. \tag{A.86}$$

Fall-out or false positive rate (FPR) measures the rate of false alarm.

$$FPR = \frac{TN}{TN + FP}. \tag{A.87}$$

Miss rate or false negative rate (FNR) measures the rate of missed samples.

$$FNR = \frac{FN}{FN + TP}. \tag{A.88}$$

$F_1$-measure combines precision and sensitivity as

$$F_1 = \frac{2 \times (Precision \times Sensitivity)}{Precision + Sensitivity}. \tag{A.89}$$

Accuracy measure is widely used in clustering. It calculates the largest rate of correct assignments by matching each data to the right cluster. For an imbalanced dataset, accuracy is usually high, while precision and recall are relatively low since the classifier tends to predict all data as majority class.

The receiver operating characteristic (ROC) curve offers another useful graphical representation for a binary classifier. It is a plot of the true positive rate (TPR) against the false positive rate (FPR). A higher area under the curve (AUC) indicates a better classifier.

**Confidence**

Consider a normal distribution $Z$ with mean $m$ and standard error $\sigma$. For a confidence level $\alpha$, the confidence interval is given by $m \pm Z^* \frac{\sigma}{\sqrt{n}}$, where $n$ is the size of each testing. This interval contains the true population mean with a probability of $\alpha$; that is, $\Pr(-Z^* < Z < Z^*) = \alpha$. For a confidence level of 95%, $Z^* = 1.96$.

For a classification accuracy or a proportion, $p$, its standard error is calculated as $\sigma = \sqrt{\frac{p(1-p)}{n}}$. For a confidence level $\alpha$, a normal distribution assumption leads to a confidence interval $p \pm Z^* \sqrt{\frac{p(1-p)}{n}}$. The normal distribution assumption requires $np \geq 10$ and $n(1-p) \geq 10$.

## Problems

**A.1** For nonbinary data, show that $\|\boldsymbol{x}\|_1 > \|\boldsymbol{x}\|_2 > \|\boldsymbol{x}\|_\infty$.

**A.2** Draw the Student-$t$ and Gaussian distributions.

**A.3** Consider the function $f(x) = 10x^3 + 4x^2 + 3x + 12$.
(a) Compute its gradient.
(b) Find all its local and global maxima/minima.

**A.4** Verify the Lipschitzness of the functions:
(a) $f(x) = |x|$.
(b) $f(x) = x^2$.

**A.5** For a classification test, we have the following counts of true labels and output labels:

| | | Output | |
|---|---|---|---|
| | | pos | neg |
| True | pos | 70 | 90 |
| | neg | 50 | 900 |

Calculate the values of precision, recall, sensitivity, and specificity.

**A.6** Suggest a field where:
(a) Precision is more important than recall.
(b) Recall is more important than precision.

**A.7** Assum that a classification accuracy $p$ satisfies the normal distribution and that $p = 0.9$.

(a) Specify a suitable size of the testing set.
(b) Calculate the standard error of the classification accuracy, for testing sets of size $n = 200$.
(c) Give the confidence interval for the 95% confidence level.
(d) Explain the influence of the size of the testing set on the classification confidence.

## References

1. Bishop, C. M. (1995). *Neural networks for pattern recognition*. New York: Oxford Press.
2. Borchers, B. (1999). CSDP: A C library for semidefinite programming. *Optimization Methods and Software, 11*, 613–623.
3. Bordes, A., Bottou, L., & Gallinari, P. (2009). SGD-QN: Careful quasi-newton stochastic gradient descent. *Journal of Machine Learning Research, 10*, 1737–1754.
4. Cardoso, J.-F. (1999). High-order contrasts for independent component analysis. *Neural Computation, 11*, 157–192.

5. Fletcher, R. (1991). *Practical methods of optimization*. New York: Wiley.
6. Friedman, J., Hastie, T., Hofling, H., & Tibshirani, R. (2007). Pathwise coordinate optimization. *Annals of Applied Statistics, 1*(2), 302–332.
7. Golub, G. H., & van Loan, C. F. (1989). *Matrix computation* (2nd Edn.). Baltimore, MD: Johns Hopkins University Press.
8. Mokhtari, A., & Ribeiro, A. (2015). Global convergence of online limited memory BFGS. *Journal of Machine Learning Research, 16* 3151–3181.
9. Nesterov, Y. (1983). A method of solving a convex programming problem with convergence rate $O(1/k^2)$. *Soviet Mathematics Doklady, 27*, 372–376.
10. Nesterov, Y. (2012). Efficiency of coordinate descent methods on huge-scale optimization problems. *SIAM Journal on Optimization, 22*(2), 341–362.
11. Phatak, A., & de Hoog, F. (2002). Exploiting the connection between PLS, Lanczos methods and conjugate gradients: Alternative proofs of some properties of PLS. *Journal of Chemometrics, 16*(7), 361–367.
12. Richtarik, P., & Takac, M. (2015). Parallel coordinate descent methods for big data optimization. *Mathematical Programming*, 1–52.
13. Schraudolph, N. N., Yu, J., & Gunter, S. (2007). A stochastic quasi-newton method for online convex optimization. In *Proceedings of the 11th International Conference on Artificial Intelligence and Statistics* (pp. 436–443). San Juan, Puerto Rico.
14. Searle, S. R. (1982). *Matrix algebra useful for statistics*. New York: Wiley-Interscience.
15. Sturm, J. F. (1999). Using Sedumi 1.02: a MATLAB toolbox for optimization over symmetric cones. *Optimization Methods and Software, 11*, 625–653.
16. Toh, K. C., Todd, M. J., & Tutuncu, R. H. (1999). SDPT3 – A MATLAB software package for semidefinite programming. *Optimization Methods and Software, 11*, 545–581.
17. Wold, H. (1975). Soft modeling by latent variables: The nonlinear iterative partial least squares approach. In J. Gani (Ed.), *Perspectives in probability and statistics: Papers in honor of M. S. Bartlett*. London: Academic Press.
18. Yuille, A. L., & Rangarajan, A. (2003). The concave-convex procedure. *Neural Computation, 15*, 915–936.

# Appendix B
# Benchmarks and Resources

In this appendix, we provide some benchmarks and resources for machine learning, pattern recognition, and data mining.

## B.1 Face Databases

The face image data have been standardized as ISO/IEC JTC 1/SC 37 N 506 (Biometric Data Interchange Formats, Part 5: Face Image Data).

**AT&T Olivetti Research Laboratory (ORL) face recognition database** (http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html) includes 400 images from 40 individuals. Each individual has 10 images: five for training and five for testing. The face images in ORL only contain pose variation and are perfectly centralized/localized. All the images are taken against a dark homogeneous background but vary in sampling time, illuminations, facial expressions, facial details (glasses/no glasses), scale, and tilt. Each image with 256 gray scales is in the size of $92 \times 112$.

**California Institute of Technology (CIT) Face Database** (http://www.vision.caltech.edu/Image_Datasets/faces/) has 450 color images, the size of each being $320 \times 240$ pixels, and contains 27 different people and a variety of lighting, backgrounds, and facial expressions.

**MIT CBCL Face Database** (http://cbcl.mit.edu/cbcl/software-datasets/FaceData2.html) has 6, 977 training images (with 2,429 faces and 4,548 nonfaces) and 24,045 test images (472 faces and 23,573 nonfaces). All images are captured in grayscale at a resolution of $19 \times 19$ pixels, but rather than use pixel values as features.

**Face Recognition Grand Challenge (FRGC) Database** [7] consists of 1,920 images, corresponding to 80 individuals selected from the original collection. Each individual has 24 controlled or uncontrolled color images. The faces are automatically detected and normalized through a face detection method and an extraction method. FRGC dataset provides high-resolution images and 3D face data.



K.-L. Du and M. N. S. Swamy, *Neural Networks and Statistical Learning*,
https://doi.org/10.1007/978-1-4471-7452-3

**FRGC v2 Database** is the largest available database of 3D face images composed of 4,007 images with different facial expressions from 466 subjects with different facial expressions. All images have resolution of 640 × 480, acquired by a Minolta Vivid 910 laser scanner. The face images have frontal pose and several types of facial expression: neutral, happy, sad, disgusting, surprised, and puffy cheek. Moreover, some images present artifacts: stretched or distorted images, nose absence, holes around nose, or waves around mouth.

**Carnegie Mellon University (CMU) Multi-PIE (pose, illumination, and expression) Face Database** (http://www.flintbox.com/public/project/4742/) contains 337 subjects with more than 750,000 face images under various viewpoints, illuminations, and expressions. The face images are of size 32 × 32 pixels, captured as under 13 poses, 43 illumination conditions, and 4 expressions.

**Yale Face Database** (https://www.cvc.yale.edu/projects/yalefaces/yalefaces.html) contains 165 grayscale images in GIF format of 15 individuals. There are 11 images of 64 × 64 pixels per subject, one per different facial expressions or configurations: center light, w/glasses, happy, left light, w/no glasses, normal, right light, sad, sleepy, surprised, and wink.

**Yale Face Database B** (https://www.cvc.yale.edu/projects/yalefacesB/yalefacesB.html) contains 2,414 frontal face images of 38 persons in 9 poses/facial expressions and under 64 illumination conditions, with approximately 64 images for each person. The images were cropped to 192 × 168 pixels.

**AR Face Database** (http://www2.ece.ohio-state.edu/~aleix/ARdatabase.html) contains over 4,000 color images corresponding to 126 people's faces (70 men and 56 women). Images feature frontal view faces with different facial expressions (anger, smiling and screaming), illumination conditions (left and/or right light on), and occlusions (sunglasses and scarf).

**Oulu Physics-Based Face Database** (www.ee.oulu.fi/research/imag/color/pbfd.html) contains faces of 125 different individuals, each in 16 different camera calibrations and illumination conditions, an additional 16 if the person has glasses. Faces are in frontal position, captured under horizon, incandescent, fluorescent, and daylight illuminant. The database includes three spectral reflectances of skin per person measured from both cheeks and forehead, and contains RGB spectral response of camera used and spectral power distribution of illuminants.

**Sheffield (previously UMIST) Face Database** (https://www.sheffield.ac.uk/eee/research/iel/research/face) consists of 575 images of 20 individuals (mixed race/gender/appearance). Each individual is shown in a range of poses from profile to frontal views. The files are all in PGM format, approximately 220 × 220 pixels with 256-bit grayscale.

**University of Notre Dame 3D Face Database** (http://www.nd.edu/~cvrl/CVRL/Data_Sets.html) includes a total of 275 subjects, among which 200 subjects participated in both a gallery acquisition and a probe acquisition. The time lapse between the acquisitions of the probe image and the gallery image for any subject ranges between 1 and 13 weeks. The 3D scans in the database were acquired using a Minolta Vivid 900 range scanner. All subjects were asked to display a neutral facial expression and to look directly at the camera. The result is a 640 × 480 array of range data.

**FG-NET Aging Database** (http://www.fgnet.rsunit.com) contains 1,002 high-resolution color or grayscale face images of 82 subjects at different ages, with the minimum age being 0 and the maximum age being 69, with large variation of lighting, pose, and expression. The ages (0–69) are divided into six ranges: 0–9, 10–19, 20–29, 30–39, 40–49, and 50+.

**MORPH Data Corpus** (http://www.faceaginggroup.com/projects-morph.html) is a face aging dataset. It has two separate databases: Album1 and Album2. Album1 contains 1,690 images from 625 different subjects. Album2 contains more than 20,000 images from more than 4,000 subjects whose metadata (age, sex, ancestry, height, and weight) are also recorded.

**Iranian Face Aging Database** (http://kiau.ac.ir/bastanfard/IFDB_index.htm) contains digital images of people from 1 to 85 years of age. It is a large database that can support studies of the age classification systems. It contains over 3,600 color images.

**Bosphorus Database** (http://bosphorus.ee.boun.edu.tr/Home.aspx) is intended for research on 3D and 2D human face processing tasks including expression recognition, facial action unit detection, facial action unit intensity estimation, face recognition under adverse conditions, deformable face modeling, and 3D face reconstruction. There are 105 subjects and 4,666 faces in the database.

**XM2VTS Face Video Database** (http://www.ee.surrey.ac.uk/CVSSP/xm2vtsdb/) contains four recordings of 295 subjects taken over a period of 4 months. The BioID face detection database is available at http://support.bioid.com/downloads/facedb/index.php. Some other face databases are given at http://www.face-rec.org/databases/.

**Yahoo! News Face Dataset** was constructed from about half a million captioned news images collected from the Yahoo! News website by crawling from the web [1]. It consists of a large number of photographs taken in real-life conditions. As a result, there are a large variety of poses, illuminations, expressions, and environmental conditions. There are 1,940 images, corresponding to 97 largest face clusters, in which each individual cluster has 20 images. Faces are cropped from the selected images using the face detection and extraction methods.

**CUHK Face Sketch FERET (CUFSF) Dataset** (http://mmlab.ie.cuhk.edu.hk/archive/cufsf/) is used to evaluate photo-sketch face recognition. It contains 1,194 subjects with lighting variations, where examples in this dataset come from photo and sketch.

### Databases for Facial Recognition in the Wild

**WIDER FACE Dataset** (http://mmlab.ie.cuhk.edu.hk/projects/WIDERFace/) is a face detection benchmark, whose images are selected from the publicly available WIDER dataset.

**MS-Celeb-1M** (https://www.msceleb.org/celeb1m/dataset) is a public face recognition dataset, containing more than 10 million labeled face images of the top 100,000

distinct identities from the 1 million celebrity list with significant pose, illumination, occlusion, and other variations.

**CelebA Dataset** (http://mmlab.ie.cuhk.edu.hk/projects/CelebA.html) is annotated with 40 face attributes and 5 keypoints by a professional labeling company for 202,599 face images of over 10,000 subjects.

**MegaFace Dataset** (http://megaface.cs.washington.edu/) is used to test the robustness of face recognition algorithms in the open-set setting with 1 million distractors. The dataset has two parts: the first part allows the use of any external training datasets and the other provides 4.7 million face images of 672,000 subjects.

**YouTube Faces Dataset** (https://www.cs.tau.ac.il/~wolf/ytfaces/) contains 3,425 videos of 1,595 different subjects and is the standard dataset used to evaluate video-face recognition algorithms.

### Databases for Facial Expression Recognition

**Japanese Female Facial Expression (JAFFE) Database** (http://www.kasrl.org/jaffe.html) contains 213 images of seven facial expressions (six basic facial expressions + one neutral) posed by 10 Japanese female models. Each image has been rated on 6 emotion adjectives by 60 Japanese subjects.

**Binghamton University BU-3DFE Database** is a database of annotated 3D facial expressions [10]. There are a total of 100 subjects in the database, 56 females and 44 males. A neutral scan was captured for each subject, and then they were asked to perform six expressions: happiness, anger, fear, disgust, sad, and surprise. The expressions vary according to four levels of intensity (low, middle, high, and highest). Thus, there are 25 3D facial expression models per subject. A set of 83 manually annotated facial landmarks is associated to each model. These landmarks are used to define the regions of the face that undergo specific deformations due to muscle movements when conveying facial expression.

**Audio Visual Emotion Challenge (AVEC 2011) Database** (https://avec-db.sspnet.eu/), which is based on **SEMAINE Database**, contains spontaneous emotional states in naturalistic situations. It consists of 95 videos recorded at 49.979 frames per second. Binary labels along the four affective dimensions (activation, expectation, power, and valence) are provided for each video frame. **SEMAINE Database** allows to study interpersonal dynamics between speakers and listeners.

**EmotiW Database** [11] is a collection of short video clips collected from some popular movies, where the actor is expressing one of seven emotions (anger, disgust, fear, happy, neutral, sad, and surprise) under near real-world conditions. It contains realistic challenges like pose variations, various illumination conditions, occlusions, and spontaneous emotions. EmotiW consists of 380 training, 396 validation, and 312 testing video clips, respectively.

## B.2 UCI Machine Learning Repository

Some popular datasets from UCI machine learning repository (http://archive.ics.uci.edu/ml/) are listed below.

- **HouseVotes Dataset** contains the 1,984 congressional voting records for 435 representatives voting on 17 issues. Votes are all three-valued: yes, no, or unknown. For each representative, the political party is given; this dataset is typically used in a classification setting to predict the political party of the representative based on the voting record.
- **Mushroom Dataset** contains physical characteristics of 8,124 mushrooms, as well as whether each mushroom is poisonous or edible. There are 22 physical characteristics for each mushroom, all of which are discrete.
- **Adult Dataset** has 48,842 patterns of 15 attributes, including eight categorical attributes, six numerical attributes, and one class attribute. The class attribute indicates whether the salary is over 50,000. In the dataset, 76% of the patterns have the value of $\leq$50,000. The goal is to predict whether a household has an income greater than $50,000.
- **Iris Dataset** has 150 data samples from three classes (setosa, versicolor, and virginica) with four measurements (sepal length, sepal width, petal length, and petal width).
- **Wisconsin Diagnostic Breast Cancer Data (WDBC)** contains 569 samples, each with 30 features. The samples are grouped into two clusters: 357 samples for benign and 212 for malignant.
- **Boston Housing Dataset** consists of 516 instances with 12 input variables (including a binary one) and an output variable representing the median housing values in suburbs of Boston.
- **Microsoft Web Training Data (MSWeb)** contains 32,711 instances of users visiting the www.microsoft.com website on one day in 1996. For each user, the data contain a variable indicating whether or not that user visited each of the 292 areas of the site.
- **Image Segmentation Database** consists of samples randomly drawn from a database of seven outdoor images. The images were hand segmented to create a classification for every pixel. Each sample has a $3 \times 3$ region and 19 attributes. There are a total of 7 classes, each having 330 samples. The attributes were normalized to lie in $[-1, 1]$.
- **Internet Advertisement Dataset** from UCI machine learning repository consists of 3,279 examples including 459 ads images (positive examples) and 2,820 non-ads images (negative examples). The first view describes the image itself (words in the image's URL, alt text and caption), while the other view contains all other features (words from the URLs of the pages that contain the image and the image points to).
- **Zoo Dataset** covers 101 animals with 17 Boolean-valued attributes, where the attributes contain hair, feathers, eggs, milk, legs, tail, etc. These animal data are grouped into seven classes.

- **Wine Quality Dataset** can be used for ordinal regression.
- **Isolet Dataset** contains acoustic features of isolated spoken letters from "A" to "Z".

## B.3 Some Machine Learning Databases

**KEEL Dataset Repository** (http://www.keel.es/dataset.php) provides imbalanced datasets, multi-instance datasets, and multi-label datasets for evaluating algorithms.

**CaliforniaDBpedia Dataset** is a dataset for spatio-textual query. It is a synthesized dataset which combines the spatial data in California and a real collection of article categories from DBpedia.

**MovieLens 10M** (http://www.grouplens.org/) and **Netflix Prize Dataset** (http://www.netflixprize.com/) are two large publicly available collaborative filtering datasets. The $1M Netflix prize competition dataset is a training dataset of 100,480,507 ratings that 480,189 users gave to 17,770 movies. Each training rating is a quadruplet <user, movie, date of grade, grade>. The user and movie fields are integer IDs, while grades are from 1 to 5 (integral) stars.

**AMI Meeting Corpus** is a dataset for understanding human multimodal behaviors during social interactions. It contains 100 h of video recordings of meetings, all fully transcribed and annotated.

### EEG Datasets

**EEGLAB** (http://sccn.ucsd.edu/eeglab/) is an interactive MATLAB toolbox for processing continuous and event-related EEG, MEG and other electrophysiological data incorporating ICA, time/frequency analysis, artifact rejection, event-related statistics, and several useful modes of visualization of the averaged and single-trial data. The Sleep-EDF database gives sleep recordings and hypnograms in European data format (EDF).

**The EEG Dataset From BCI Competition 2003** (http://www.bbci.de/competition/) has 28 channel input recorded from a single subject performing a self-paced key typing, that is, pressing with the index and little fingers corresponding keys in a self-chosen order and timing. **BCI Competition IV-2a** provides data from 9 subjects performing 4 imagery movement tasks, namely, right hand (RH), left hand (LH), both feet (F), and tongue (T), during 2 days of recordings. Each day the subjects performed 72 trials of each task (3 seconds per trial), and the EEG data were recorded using 20 electrodes. **Physiobank** (http://www.physionet.org/physiobank/database) contains EEG data from 109 subjects performing various combinations of real and imagined movements in one day of recordings.

### Image Databases

**Columbia Object Image Library (COIL-20)** (http://www.cs.columbia.edu/CAVE/software/softlib/coil-20.php) contains the images of 20 different three-dimensional objects. The objects represent cups, toys, drugs, and cosmetics. For each object 72 training samples are available. The size of the images is $32 \times 32$ grayscale images viewed from varying angles.

**CIFAR-10 and CIFAR-100 Datasets** (https://www.cs.toronto.edu/~kriz/cifar.html) are labeled subsets of the 80 million tiny images dataset. CIFAR-10 dataset consists of 60,000 $32 \times 32$ color images in 10 classes, with 6000 images per class: 50000 for training and 10000 for testing. CIFAR-100 dataset is just like CIFAR-10, except it has 100 classes containing 600 images each: 500 for training and 100 for testing per class. The 100 classes in the CIFAR-100 are grouped into 20 superclasses.

**Microsoft Research Asia Internet Multimedia Dataset 1.0 (MSRA-MM 1.0)** explored the query log of Microsoft Bing Image Search and selected a set of representative ones. The ground truth of queries is given. MSRA-MM 2.0 image dataset adds 1, 097 frequently used queries. These queries are manually classified into nine categories, i.e., Animal, Cartoon, Event, Object, NamedPerson, PeopleRelated, Scene, Time, and Misc. The total image number is 1,011,738. Each concept has approximately 500–1,000 images. Seven low-level features were extracted for each image.

**Corel Images Dataset** from UCI repository consists of 34 categories, each with 100 JPEG images of $384 \times 256$ or $256 \times 384$ resolution.

Corel and MSRA image data are very representative and frequently used in many tasks of multiple-instance learning.

**NUS-WIDE Dataset** (http://lms.comp.nus.edu.sg/research/NUS-WIDE.htm) is also a large-scale annotated web image dataset publicly available to researchers. It consists of 269,648 images collected from the website of Flickr and their ground truth annotations for 81 concepts.

**Caltech-101 Dataset** (http://www.vision.caltech.edu/Image_Datasets/Caltech101/) contains 101 categories of object images. It contains 8,677 images from 101 categories and 467 images from an additional background category.

**Pascal Visual Object Classes (VOC) Challenge Dataset** (http://host.robots.ox.ac.uk/pascal/VOC/) is a benchmark for image classification, detection, and segmentation. There are large variations in pose, view, scale, appearance, and clustered background. Pascal VOC 2012 contains 22,534 images including a trainval set of 11,540 images and a test set of 10,994 images. 5,717 images of the trainval set are used for training and the remaining 5,823 images for validation. Annotations of the test set are not publicly available.

**ImageNet** (http://image-net.org/) is a real-world image database containing roughly 15 million images organized according to the WordNet hierarchy. Currently, over 20 thousand noun synsets in WordNet are indexed and each synset has over 500 images on average. **ILSVRC 1000 (ImageNet Large Scale Visual Recognition Challenge 2010) Dataset** has 1,000 categories and 1,261,406 images. **ILSVRC 2013 Object Detection Set** is constructed following the style of PASCAL VOC but

contains more images and categories: 200 basic-level categories, 395, 909 images for training, 20,121 images for validation, and 40,152 images for testing.

**KITTI Vision Benchmark Suite** (http://www.cvlibs.net/datasets/kitti/) provides datasets for developing automatic driving systems. The object detection dataset in the suite can be used for detecting cars in images taken "in the wild." It has three levels of difficulty: easy, moderate, and hard. There are 7,481 images for training and 7,518 for testing. The testing images have no ground truth.

**Richly Annotated Pedestrian (RAP) Dataset** (http://rap.idealtest.org/) contains 84,928 images with 72 types of attributes and additional tags of viewpoint, occlusion, body parts, and 2,589 person identities. It was collected from a high-definition ($1,280 \times 720$) surveillance network at an indoor shopping mall. Other examples of datasets for pedestrian detection are **MIT Pedestrian Dataset** (http://cbcl.mit.edu/software-datasets/PedestrianData.html) and **INRIA Person Dataset** (http://pascal.inrialpes.fr/data/human/).

Some benchmark image databases are **Berkeley Image Segmentation Database** (http://www.eecs.berkeley.edu/Research/Projects/CS/vision/bsds/), brain MRI images from **BrainWeb Database** (http://www.bic.mni.mcgill.ca/brainweb/), and 3D shape objects from **NORB Dataset** (https://cs.nyu.edu/~ylclab/data/norb-v1.0/index.html). In the Berkeley segmentation database, a natural color image (Training Image #124084) is a flower, which contains four dominant colors. A collection of datasets for the annotations of the video sequence is available at http://www.vision.ee.ethz.ch/~bleibe/data/datasets.html.

There are several standard benchmarks for single image super-resolution (i.e., **Set5**, **Set14** and **BSD200** Datasets) and for super-resolution (i.e., **Videoset4 Dataset**) [2].

### Image Motion Detection Databases

**CDnet Dataset** (http://www.changedetection.net/) is a real-world region-level motion detection benchmark. The 2012 dataset contains 31 video sequences that are divided into 6 video categories: baseline, dynamic background, intermittent object motion, thermal, camera jitter, and shadows, with 4 to 6 video sequences in each category. The resolution of the videos also varies from $320 \times 240$ to $480 \times 720$ with hundreds to thousands of frames. The 2014 dataset contains 11 video categories with 4 to 6 video sequences in each category.

**Hopkins-155 Dataset** (http://www.vision.jhu.edu/data/hopkins155/) is a benchmark for motion segmentation. It contains 120 two-motion and 35 three-motion videos.

### Image Retrieval Databases

**INRIA Holidays Dataset** (http://lear.inrialpes.fr/~jegou/data.php#holidays) is collected from personal holiday albums. It has 1, 491 images composed of 500 groups of similar images. Each image group has 1 query, totaling 500 query images.

**Ukbench Dataset** consists of 10,200 images of various contents, such as objects, scenes, and CD covers. The images are divided into 2,550 groups, each having 4 images of the same object/scene, under various angles, illuminations, and translations. Each image is taken as a query, thus 10,200 queries in total.

**Oxford Buildings Dataset** (http://www.robots.ox.ac.uk/~vgg/data/oxbuildings/) consists of 5,062 images collected from Flickr by searching for particular Oxford landmarks using the names of 11 different landmarks in Oxford. The dataset defines 5 queries for each landmark by hand-drawn bounding boxes, totaling 55 query regions of interest (ROI). Each database image is assigned one of the four labels, good, OK, junk, or bad.

**Flickr 100k Dataset** (http://www.robots.ox.ac.uk/~vgg/data/oxbuildings/flickr100k.html) contains 100, 071 high-resolution images crawled from Flickr's 145 most popular tags.

**Paris Dataset** (http://www.robots.ox.ac.uk/~vgg/data/parisbuildings/) is featured by 6,412 images collected from Flickr by searching from 11 queries on particular Paris landmarks. Each landmark has 5 queries, so there are also 55 queries with bounding boxes. The database images are annotated with the same four types of labels as Oxford Buildings Dataset.

**Pittsburgh Dataset** (http://www.ok.sc.e.titech.ac.jp/~torii/project/repttile/) is a geotagged image database for visual place recognition. It is formed by 254,064 perspective images, generated from 10,586 Google Street View panoramas of the Pittsburgh area.

**Places Dataset** (http://places2.csail.mit.edu/) contains more than 10 million images comprising 400+ unique scene categories. The dataset features 5,000–30,000 training images per class, consistent with real-world frequencies of occurrence.

**FM2 Dataset** (http://www.zemris.fer.hr/~ssegvic/datasets/unizg-fer-fm2.zip) for traffic scene recognition contains 6,237 images from eight classes: highway, road, tunnel, tunnel exit, settlement, overpass, toll booth, and dense traffic.

**Cityscapes Dataset** (https://www.cityscapes-dataset.com/) focuses on semantic understanding of urban street scenes. It consists of a diverse set of street scene photos taken from 50 different cities by car-carried cameras: 5,000 photos with high-quality pixel-level annotations and 20,000 photos with coarse annotations. The pictures belong to 30 semantic classes, such as road, car, pedestrian, and bicycle, which are grouped into eight categories, i.e., flat, nature, object, sky, construction, human, and vehicle, and void.

**Biometric Databases**

**University of Notre Dame Iris Image Dataset** (https://sites.google.com/a/nd.edu/public-cvrl/data-sets) contains 64,980 iris images obtained from 356 subjects (712 unique irises) between January 2004 and May 2005.

**ATVS-FIr Database** (http://atvs.ii.uam.es/fir_db.html) is an iris database from ATVS Biometric Recognition Group. The samples are taken from 50 random users of Biosec Baseline Iris Subcorpus. It contains iris samples of both eyes. Four samples of each iris were captured in 2 acquisition sessions. Fake samples were also acquired from high-quality printed images of the original samples. There are 800 real and 800 fake image samples.

**CASIA-IrisV2 and CASIA-IrisV4 Databases** (http://biometrics.idealtest.org/) are provided by Institute of Automation of the Chinese Academy of Sciences (CASIA). CASIA-IrisV2 includes two subsets, each including 1,200 images from 60 classes. CASIA-IrisV4 comprises six subsets.

**IIITD Contact Lens Iris Database** (http://www.iab-rubric.org/resources.html) is provided by Image Analysis and Biometrics Lab of IIIT, Delhi, India. It is composed of 6,570 iris images coming from 101 subjects. Both left and right iris images of each subject were captured, and therefore there are 202 iris classes.

**Hong Kong Polytechnic University (PolyU) Palmprint Database** (https://www4.comp.polyu.edu.hk/~biometrics/) includes 600 palmprint images with the size of $128 \times 128$ from 100 individuals, with six images from each.

**AMI Ear Dataset**, which was collected at the University of Las Palmas, consists of 700 images of a total of 100 distinct subjects in the age group of 19–65 years.

**Annotated Web Ears (AWE) Dataset** contains images collected from the web and is a dataset for ear recognition gathered in the wild. AWE MATLAB toolbox (http://awe.fri.uni-lj.si) contains tools for generating performance metrics and graphs, and for research in ear recognition. The dataset contains 1,000 ear images of 100 subjects. Each image in the dataset was annotated according to gender, ethnicity, accessories, occlusion, head pitch, head roll, head yaw, and head side.

**USTB (University of Science and technology in Beijing) Ear Image Databases** (http://www1.ustb.edu.cn/resb/en/visit/visit.htm) offers four collections of 2D ear and face profile images, and **UND (University of Notre Dame) Databases** (https://cvrl.nd.edu/projects/data/) offers five databases of 2D ear images.

Some other biometric databases are **PolyU Finger-Knuckle–Print Databases** (http://www4.comp.polyu.edu.hk/~biometrics/FKP.htm) and **CASIA Gait Database** (http://www.cbsr.ia.ac.cn/english/Gait%20Databases.asp).

**Cambridge Gesture Dataset** (https://labicvl.github.io/ges_db.htm) consists of 900 image sequences of nine hand gesture classes, which are divided into three primitive hand shapes and three primitive motions. Each class contains 100 image sequences including five illumination backgrounds, and each of the sequences was recorded in front of a fixed camera which roughly isolated gestures in space and time.

For human action recognition, **Weizmann Action Dataset** and **Ballet Dataset** (http://www.cs.sfu.ca/research/groups/VML/semilatent/) are video sequences of different actions of many subjects.

#### Datasets for One-Class Classification

#### Datasets for One-Class Classification

**Intrusion Detection Dataset** (http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html) consists in binary TCP dump data from 7 weeks of network traffic. Each original pattern has 34 continuous features and seven symbolic features. The training set contains 4,898,431 connection records, which are processed from about four gigabytes of compressed binary TCP dump data from 7 weeks of network traffic. Another 2 weeks of data produced the test data with 311,029 patterns. The dataset includes a wide variety of intrusions simulated in a military network environment. There are a total of 24 training attack types, and additional 14 types that appear in the test data only.

**Promoter Database** (from UCI Repository) consists of 106 samples, 53 for promoters, while the others for nonpromoters.

#### Datasets for Handwriting Recognition

The well-known real-world OCR benchmarks are the USPS dataset, the MNIST dataset, and the UCI Letter dataset (from UCI Repository).

**MNIST handwritten Digits Database** (http://yann.lecun.com/exdb/mnist/) consists of 60,000 training samples from approximately 250 writers and 10,000 test samples from a disjoint set of 250 other writers. It contains 784-dimensional nonbinary sparse vectors which resembles $28 \times 28$ pixel gray-level images of the handwritten digits.

**US Postal Service (USPS) handwritten digit database** (http://www.cs.nyu.edu/~roweis/data.html) contains 7,291 training and 2,007 images of handwritten digits, size $16 \times 16$.

**Pendigits Dataset** (from UCI Repository) contains 7,494 training digits and 3,498 testing digits represented as vectors in 16-dimensional space. The digit database collects 250 samples from 44 writers. The samples written by 30 writers are used for training, and the digits written by the other 14 are used for testing.

## B.4 Datasets for Data Mining

**Reuters-21578 Corpus** (http://www.daviddlewis.com/resources/testcollections/reuters21578/) is a set of 21,578 economic news published by Reuters in 1987. Each article is typically designated into one or more semantic categories such as

"earn", "trade", and "corn", where the total number of categories is 114. The commonly used ModApte split filters out duplicate articles and those without a labeled topic, and then uses earlier articles as the training set and later articles as the test set.

**20 Newsgroups Dataset** (http://people.csail.mit.edu/jrennie/20Newsgroups/) is a collection of approximately 20,000 newsgroup documents, partitioned (nearly) evenly across 20 different newsgroups. This corpus contains 26,214 distinct terms after stemming and stop word removal. Each document is then represented as a term frequency vector and normalized to one.

**CMU WebKB Knowledge Base** (http://www.cs.cmu.edu/afs/cs/project/theo-11/www/wwkb/) is a collection of 8,282 web pages obtained from 4 academic domains. The web pages in the WebKB set are labeled using two different polychotomies. The first is according to topic, and the second is according to web domain. The first polychotomy consists of 7 categories: course, department, faculty, project, staff, student, and other.

**OHSUMED Dataset** (http://ir.ohsu.edu/ohsumed/ohsumed.html) is a clinically oriented MEDLINE subset formed by 348,566 references of 270 medical journals published between 1987 and 1991. It consists of 348,566 references and 106 queries with their respective ranked results. The relevance degrees of references with regard to the queries are assessed by humans, on three levels: definitely, possibly, or not relevant. Totally, there are 16,140 query–document pairs with relevance judgments.

**tr41 Dataset** is derived from the TREC-5, TREC-6, and TREC-7 collections (http://trec.nist.gov). It includes 210 documents belonging to seven different classes. The dimension of this dataset is 7,454.

**Spam Dataset** (from UCI Repository) contains 4,601 examples of e-mails, roughly 39% of which are classified as spam. There are 57 attributes for each example, most of which represent how frequently certain words or characters appear in the e-mail.

## B.5 Databases and Tools for Speech Recognition and Audio Classification

**YOHO Speaker Verification Database** consists of sets of four combination lock phrases spoken by 168 speakers. This database can be purchased from Linguistic Data Consortium as LDC94S16.

**Isolet Spoken Letter Recognition Database** (from the UCI Repository) contains 150 subjects who spoke the name of each letter of the alphabet twice. The speakers are grouped into sets of 30 speakers each and are referred to as isolets 1 through 5.

**TIMIT Acoustic-Phonetic Continuous Speech Corpus** contains a total of 6,300 sentences, 10 sentences spoken by 630 speakers selected from eight major dialect regions of the United States. 70% of the speakers are male, and 30% are female. It can be purchased from Linguistic Data Consortium as LDC93S1. The speech was labeled at both phonetic and lexical levels.

**Oregon Graduate Institute Telephone Speech (OGI-TS) Corpus** is a multilingual speech corpus for LID experiments. The OGI-TS speech corpus contains the speech from 11 languages. It includes recorded utterances from about 2,052 speakers.

**CALLFRIEND Telephone Speech Corpus** (http://www.ldc.upenn.edu/Catalog/) is a collection of unscripted conversations for 12 languages recorded over telephone lines. It is used in the NIST language recognition evaluations (http://www.itl.nist.gov/iad/mig/tests/lang/) tasks, which are performed as language detection: Given a segment of speech and a language hypothesis, the task is to decide whether that target language was spoken in the given segment. OGI-TS corpus and CALLFRIEND corpus are widely used in language identification evaluation.

HMM Tool Kit (http://htk.eng.cam.ac.uk/) is a de facto standard toolkit in C for training and manipulating HMMs in speech research. The HMM-based speech synthesis system (HTS) (http://hts-engine.sourceforge.net/) adds to HMM Tool Kit various functionalities in C for HMM-based speech synthesis. Some speech synthesis systems are Festival (http://www.cstr.ed.ac.uk/projects/festival/), Flite (Festival-lite) (http://www.speech.cs.cmu.edu/flite/), and MARY text-to-speech system (http://mary.dfki.de/).

**CMU_ARCTIC Databases** (http://festvox.org/cmu_arctic/) are phonetically balanced, U.S. English, single-speaker databases designed for speech synthesis research. The HTS recipes for building speaker-dependent and speaker-adaptive HTS voices use these databases.

Some open-source speech processing systems are Speech Signal Processing Toolkit (http://sp-tk.sourceforge.net/), STRAIGHT and STRAIGHTtrial (http://www.wakayama-u.ac.jp/~kawahara/STRAIGHTadv/index_e.html), and Edinburgh Speech Tools (http://www.cstr.ed.ac.uk/projects/speech_tools/).

auDeep (https://github.com/auDeep/auDeep) is a Python toolkit for deep unsupervised learning from acoustic data. It is based on a recurrent sequence to sequence autoencoder approach to learn representations of time series data. It provides a command-line interface. auDeep can be used for audio classification tasks, such as acoustic scene classification, environmental sound classification, and music genre classiffication.

Benchmarks for audio classification tasks are **TUT Acoustic Scenes 2017 Dataset** (http://www.cs.tut.fi/sgn/arg/dcase2017/challenge/task-acoustic-scene-classification) for acoustic scene classification, **ESC-50 Dataset** (https://github.com/karoldvl/ESC-50) for environmental sound classiffication (ESC), and **GTZAN Dataset** (http://opihi.cs.uvic.ca/sound/genres.tar.gz) for music genre classiffication.

## B.6 Datasets for Microarray and for Genome Analysis

**Yeast Sporulation Dataset** (http://cmgm.stanford.edu/pbrown/sporulation) is a microarray dataset on the transcriptional program of sporulation in budding yeast. A DNA microarray containing 97 % of the known and predicted genes is used. The total number of genes is 6, 118. During the sporulation process, the mRNA levels

were obtained at seven time points 0, 0.5, 2, 5, 7, 9, and 11.5 h. The ratio of each gene's mRNA level (expression) to its mRNA level in vegetative cells before transfer to the sporulation medium is measured.

**Human Fibroblasts Serum Dataset** (http://www.sciencemag.org/feature/data/984559.shl) contains the expression levels of 8,613 human genes. It has 13 dimensions. A subset of 517 genes whose expression levels changed substantially across the time points has been chosen.

**Rat Central Nervous System Dataset** (http://faculty.washington.edu/kayee/cluster) examines the expression levels of a set of 112 genes during rat central nervous system development over nine time points.

**Yeast Cell Cycle Dataset** (http://faculty.washington.edu/kayee/cluster) was extracted from a dataset that shows the fluctuation of expression levels of approximately 6,000 genes over two cell cycles (17 time points). Out of these 6,000 genes, 384 genes have been selected to be cell cycle regulated.

**ELVIRA Biomedical Dataset Repository** (http://leo.ugr.es/elvira/DBCRepository/index.html) includes high-dimensional biomedical datasets, including gene expression data, protein profiling data, and genomic sequence data that are related to classification. The colon cancer dataset consists of 62 samples of colon epithelial cells from colon cancer patients. The samples consist of tumor biopsies collected from tumors (40 samples), and normal biopsies collected from healthy part of the colons (22 samples) of the same patient. The number of genes in the dataset is 2,000.

**Global Cancer Map** (http://www.broadinstitute.org/cgi-bin/cancer/datasets.cgi) is a gene expression dataset consisting of 198 human tumor samples spanning 14 different cancer types.

### General Databases

**GenBank** (http://www.ncbi.nlm.nih.gov/Genbank/index.html) is the NIH genomic database, an annotated collection of all publicly available DNA sequences. It contains all annotated nucleic acid and amino acid sequences. Apart from presenting and annotating sequences, these databases offer many functions related to searching and browsing sequences.

**Rfam Database** (http://rfam.sanger.ac.uk/) is a collection of RNA families, each represented by multiple sequence alignments, consensus secondary structures, and covariance models.

**EMBL Nucleotide Sequence Database (EMBL-Bank)** (http://www.ebi.ac.uk/embl/) constitutes Europe's primary nucleotide sequence resource.

**Stanford Microarray Database** (http://genome-www5.stanford.edu/) and gene expression omnibus are the two most famous and abundant gene expression databases in the world. Gene expression omnibus is a database including links to microarray-based experiments measuring mRNA, genomic DNA, and protein abundances, as well as non-array techniques such as serial analysis of gene expression and mass spectrometric proteomic data.

**Analysis Tools**

Some websites for genome analysis are Human Genome Project (http://www.ornl.gov/sci/techresources/Human_Genome/home.shtml), Ensembl Genome Browser (http://www.ensembl.org/index.html), and UCSC Genome Browser (http://genome.ucsc.edu/).

MeV (http://www.tm4.org/mev.html) is a versatile microarray tool, incorporating sophisticated algorithms for clustering, visualization, classification, statistical analysis, and biological theme discovery.

For sequence analysis, BLAST (http://blast.ncbi.nlm.nih.gov/Blast.cgi) finds regions of similarity between biological sequences, and ClustalW2 (http://www.ebi.ac.uk/Tools/clustalw/) is a general-purpose multiple sequence alignment program for DNA or proteins.

SignatureClust (http://infos.korea.ac.kr/sigclust.php) is a tool for landmark gene-guided clustering that enables biologists to get multiple views of the microarray data.

## B.7 Software

Stuttgart Neural Network Simulator (http://www.ra.cs.uni-tuebingen.de/SNNS/) is a software simulator for neural networks on Unix systems. The simulator kernel is written in C, and it provides X graphical user interface. The simulator supports the following network architectures and learning procedures that are discussed in this book: online BP, BP with momentum term and flat spot elimination, batch BP, Quickprop, Rprop, generalized RBF network, ART 1, ART 2, ARTMAP, cascade correlation, dynamic LVQ, BPTT, Quickprop through time, SOM, TDNN with BP, Jordan networks, Elman networks, and associative memory.

SHOGUN (http://www.shogun-toolbox.org) is an open-source toolbox in C++ that runs on UNIX/Linux platforms and interfaces to MATLAB. It provides a generic interface to 15 SVM implementations (among them are SVMlight, LibSVM, GPDT, SVMLin, LibLinear, SVM SGD, SVMPegasos and OCAS, kernel ridge regression, SVR), multiple kernel learning, Naive Bayes classifier, $k$-NN, LDA, HMMs, $C$-means, and hierarchical clustering. SVMs can be combined with more than 35 different kernel functions. One of the SHOGUN's key features is the combined kernel to construct weighted linear combinations of multiple kernels that may even be defined on different input domains.

Dlib-ml (http://dclib.sourceforge.net) provides a similarly rich environment for developing machine learning software in C++. It contains an extensible linear algebra toolkit with built-in BLAS support. It also houses implementations of algorithms for performing inference in Bayesian networks and kernel-based methods for classification, regression, clustering, anomaly detection, and feature ranking. MLPACK (http://www.mlpack.org) is a scalable, multi-platform C++ machine learning library offering a simple, consistent API, high performance, and flexibility.

LRSLibrary (https://github.com/andrewssobral/lrslibrary) provides a collection of low-rank and sparse decomposition algorithms in MATLAB. It was designed for motion segmentation in videos but can be used for other computer vision problems. LRSLibrary offers more than 100 algorithms based on matrix and tensor methods, including robust PCA, subspace tracking, matrix completion, low-rank recovery, three-term decomposition, NMF, nonnegative tensor factorization, and tensor decomposition.

Netlab (http://www1.aston.ac.uk/eas/research/groups/ncrg/resources/netlab/) is another neural network simulator implemented in MATLAB.

ThunderSVM (https://github.com/zeyiwen/thundersvm) is an open-source SVM software toolkit which exploits the high performance of GPUs and multi-core CPUs. ThunderSVM supports all the functionalities of LibSVM. It is generally an order of magnitude faster than LibSVM while producing identical SVMs.

DOGMA (http://dogma.sourceforge.net) is a MATLAB toolbox for discriminative online learning. The library focuses on linear and kernel online algorithms, mainly developed in the relative mistake bound framework. Examples are perceptron, passive-aggressive, ALMA, NORMA, SILK, projectron, RBP, and Banditron.

**Some resources for implementing RBF networks are**: ELM (http://www.ntu.edu.sg/home/egbhuang/), optimally pruned ELM (https://research.cs.aalto.fi/aml/software/OPELM.zip), and the improved Levenberg–Marquardt algorithm for RBF networks (http://www.eng.auburn.edu/~wilambm/nnt/index.htm).

**A MATLAB toolbox for implementing several PCA techniques** is available at http://research.ics.tkk.fi/bayes/software/index.shtml. Some NMF tools are NMF-Pack (MATLAB,http://www.cs.helsinki.fi/u/phoyer/software.html), NMF package (C++, http://nmf.r-forge.r-project.org), and bioNMF (MATLAB, C, http://bionmf.cnb.csic.es).

**Some resources for implementing ICA are** JADE (http://www.tsi.enst.fr/icacentral/Algos/cardoso/), FastICA (http://www.cis.hut.fi/projects/ica/fastica/), efficient FastICA (http://itakura.kes.tul.cz/zbynek/downloads.htm), RADICAL (http://www.eecs.berkeley.edu/~egmil/ICA), and denoising source separation (http://www.cis.hut.fi/projects/dss/).

**Some resources for implementing clustering are** SOM_PAK and LVQ_PAK (http://www.cis.hut.fi/~hynde/lvq/), Java applets for TSP based on SOM and Kohonen network (http://sydney.edu.au/engineering/it/~irena/ai01/nn/tsp.html, http://www.sund.de/netze/applets/som/som2/), Java applet implementing several competitive learning-based clustering algorithms (http://www.sund.de/netze/applets/gng/full/GNG-U_0.html), C++ code for minimum sum-squared residue co-clustering algorithm (http://www.cs.utexas.edu/users/dml/Software/cocluster.html), and C++ code for single-pass fuzzy $C$-means and online fuzzy $C$-means (http://www.csee.usf.edu/~hall/scalable).

**Some resources for implementing LDA are** uncorrelated LDA and orthogonal LDA (http://www-users.cs.umn.edu/~jieping/UOLDA/), neighborhood component analysis (http://www.cs.berkeley.edu/~fowlkes/software/nca/), local LDA (http://sugiyama-www.cs.titech.ac.jp/~sugi/software/LFDA/), and semi-supervised local

Fisher discriminant analysis (http://sugiyama-www.cs.titech.ac.jp/~sugi/software/SELF).

**Some resources for implementing SVMs are** Lagrangian SVM (http://www.cs.wisc.edu/dmi/lsvm), potential SVM (http://ni.cs.tu-berlin.de/software/psvm), LASVM (http://leon.bottou.com/projects/lasvm, http://www.neuroinformatik.rub.de/PEOPLE/igel/solasvm), LS-SVM (http://www.esat.kuleuven.ac.be/sista/lssvmlab/), $2\nu$-SVM (dsp.rice.edu/software), Laplacian SVM in the primal (http://sourceforge.net/projects/lapsvmp/), SimpleSVM (http://sourceforge.net/projects/simplesvm/), decision-tree SVM (http://ocrwks11.iis.sinica.edu.tw/dar/Download/WebPages/DTSVM.htm), core vector machine (http://c2inet.sce.ntu.edu.sg/ivor/cvm.html), OCAS and OCAM in LIBOCAS (http://cmp.felk.cvut.cz/~xfrancv/ocas/html/) and as a part of the SHOGUN toolbox, Pegasos (http://ttic.uchicago.edu/~shai/code), EnsembleSVM (http://homes.esat.kuleuven.be/~claesenm/ensemblesvm/), MSVMpack in C (http://www.loria.fr/~lauer/MSVMpack/), and BMRM in C++ (http://users.cecs.anu.edu.au/~chteo/BMRM.html).

**Some resources for implementing kernel methods are** regularized kernel discriminant analysis (http://www.public.asu.edu/~jye02/Software/DKL/), $L_p$-norm multiple kernel learning (http://doc.ml.tu-berlin.de/nonsparse_mkl/, implemented within the SHOGUN toolbox), TRON and TRON-LR in LIBLINEAR (http://www.csie.ntu.edu.tw/~cjlin/liblinear), FaLKM-lib (http://disi.unitn.it/~segata/FaLKM-lib), SimpleMKL (http://asi.insa-rouen.fr/enseignants/~arakotom/code/mklindex.html), HessianMKL (http://olivier.chapelle.cc/ams/), LevelMKL (http://appsrv.cse.cuhk.edu.hk/~zlxu/toolbox/level_mkl.html, SpicyMKL (http://www.simplex.t.u-tokyo.ac.jp/~s-taiji/software/SpicyMKL), generalized kernel machine toolbox (http://theoval.cmp.uea.ac.uk/~gcc/projects/gkm), and JKernelMachines (in Java https://github.com/davidpicard/jkernelmachines) for SVM and learning with kernels.

GPML (http://www.gaussianprocess.org/gpml/code/matlab/doc/) and GPstuff (http://research.cs.aalto.fi/pml/software/gpstuff/) are MATLAB toolboxes for Gaussian processes, which are Bayesian nonparametric models using a prior on functions. GPML toolbox implements approximate inference algorithms for Gaussian processes for a wide class of likelihood functions for both regression and classification. GPstuff toolbox is a versatile collection of Gaussian process models and computational tools required for inference. GPflow (http://github.com/GPflow/GPflow) is a Gaussian process library that uses TensorFlow for core computations and Python for its front end. It uses variational inference as the primary approximation method, provides concise code through using automatic differentiation, and is able to exploit GPU hardware.

A selected collection of tutorials, publications, computer codes for Gaussian processes, mathematical programming, SVM, and kernel methods can be found at http://www.kernel-machines.org.

**For Bayesian Networks**

XMLBIF (XML-based BayesNets Interchange Format) is an XML-based format that is very simple to understand and yet can represent DAGs with probabilistic relations, decision variables, and utility values. The XMLBIF format is implemented in the JavaBayes (http://www.cs.cmu.edu/~javabayes/) and GeNie (http://genie.sis.pitt.edu/) systems.

FastInf (http://compbio.cs.huji.ac.il/FastInf) is a C++ library for propagation-based approximate inference methods in large-scale discrete undirected graphical models. Various message-scheduling schemes that improve on the standard synchronous or asynchronous approaches are included. FastInf includes exact inference by the junction-tree algorithm [3], loopy belief propagation, generalized belief propagation [9], tree re-weighted belief propagation [8], propagation based on convexification of the Bethe free energy [4], variational Bayesian, and Gibbs sampling. All methods can be applied to both sum and max product propagation schemes, with or without damping of messages.

libDAI (http://www.libdai.org) is an open-source C++ library that provides implementations of various exact and approximate inference methods for graphical models with discrete-valued variables. libDAI uses factor graphs. Apart from exact inference by brute force enumeration and the junction-tree method, libDAI offers the following approximate inference methods for calculating partition sums, marginals, and MAP states: mean field, (loopy) belief propagation, tree expectation propagation [5], generalized belief propagation [9], loop-corrected belief propagation [6], a Gibbs sampler, and several other methods. In addition, libDAI supports parameter learning of conditional probability tables by ML or EM (in case of missing data).

**Some resources for implementing Bayesian and probabilistic networks are**: Murphy's Bayes Network Toolbox (in MATLAB, http://code.google.com/p/bnt/), Probabilistic Networks Library (http://sourceforge.net/projects/openpnl), GRMM (http://mallet.cs.umass.edu/grmm), Factorie (http://code.google.com/p/factorie), Hugin (http://www.hugin.com), and an applet showcasing common Markov chain algorithms (http://www.lbreyer.com/classic.html).

**For Reinforcement Learning**

RL-Glue (http://glue.rl-community.org) is a language-independent software for reinforcement learning experiments; it provides a common interface for a number of software and hardware projects in the reinforcement learning community. RL-Glue has been ported to a number of languages including C/C++/Java/MATLAB via sockets.

Libpgrl (http://code.google.com/p/libpgrl/) implements both model-free reinforcement learning and policy search algorithms, though not any model-based learning. Libpgrl is efficient in a distributed reinforcement learning environment. Libpgrl is a fast C++ implementation that has abstract classes to model a subset of reinforcement learning.

MATLAB Markov Decision Process Toolbox (http://www.inra.fr/mia/T/MDPtoolbox/) implements only a few basic algorithms such as tabular Q-learning, SARSA, and dynamic programming. Some resources on reinforcement learning are available at http://www-all.cs.umass.edu/rlr/.

### IoT Platforms

ThingSpeak (https://thingspeak.com) is an open cloud platform that connects things and people. It includes real-time data collection and storage, MATLAB analytics and visualizations, alerts, scheduling, device communication, open API, and geolocation data.

NIMBITS (https://www.nimbits.com/index.jsp) is an open-source IoT platform for connecting people, sensors, and devices on the cloud.

EVRYTHNG (https://evrythng.com) manages billions of intelligent IoT identities on the cloud, giving each a persistent, addressable web presence. SensorCloud (https://www.sensorcloud.com) is a sensor data storage, visualization, and remote management platform based on the cloud.

Xively (https://xively.com) offers a PAAS that allows IoT devices to connect to the cloud.

### Other Resources

CVX (http://cvxr.com/cvx/) is a MATLAB-based modeling system for convex optimization.

SDPT3 (http://www.math.nus.edu.sg/~mattohkc/sdpt3.html) is an SDP solver. The MATLAB function `fmincon` is an SQP solver with a quasi-Newton approximation to the Hessian of the Lagrangian using the BFGS method.

SparseLab(http://sparselab.stanford.edu/) is a MATLAB software package for sparse solutions to systems of linear equations.

Resources on random forests are available at http://www.math.usu.edu/~adele/forests/. WEKA machine learning archive (http://www.cs.waikato.ac.nz/ml/weka/) offers a Java implementation of random forests. The classification results for bagging and boosting can be obtained using WEKA on identical training and test sets.

MultiBoost package (http://www.multiboost.org/) provides a fast C++ implementation of multiclass/multi-label/multitask boosting algorithms.

C5.0 (http://www.rulequest.com/see5-info.html) is a sophisticated data mining tool in C for discovering patterns that delineate categories, assembling them into classifiers, and using them to make predictions.

Resources on GPU can be found at http://www.nvidia.com, http://www.gpgpu.org/.

SIFT descriptors for an image can be generated by using open-source C libraries such as openSIFT library (http://robwhess.github.io/opensift/) or ezSIFT (https://github.com/robertwgh/ezSIFT).

Pylearn2 (http://deeplearning.net/software/pylearn2) is a python machine learning library.

Megaman (https://github.com/mmp2/megaman) is a Python package for scalable manifold learning.

MLweb (http://mlweb.loria.fr/lalolab/) is an open-source JavaScript software toolkit for machine learning on the web. All computations are performed on the client side without the need to send data to a third-party server.

SAMOA (scalable advanced massive online analysis, https://github.com/abifet/samoa) is an open-source Java platform for mining big data streams. It provides a collection of distributed streaming algorithms for the most common data mining and machine learning tasks. Its pluggable architecture allows it to run on distributed stream processing engines such as Storm, S4, and Samza.

Gesture recognition toolkit (https://github.com/nickgillian/grt) is a cross-platform open-source C++ library designed to make real-time machine learning and gesture recognition.

SPMF (http://www.philippe-fournier-viger.com/spmf/) is an open-source Java library of more than 55 data mining algorithms. It is specialized for discovering patterns in transaction and sequence databases such as frequent itemsets, association rules, and sequential patterns.

MEKA project (http://waikato.github.io/meka/) provides an open-source Java implementation of dozens of methods for multi-label learning and evaluation. MEKA is based on WEKA machine learning toolkit.

Apache Spark is a popular open-source platform for large-scale data processing that is well suited for iterative machine learning tasks. MLlib is Spark's open-source distributed machine learning library.

scikit-learn (https://scikit-learn.org/stable/) is a popular open-source machine learning library in Python. scikit-multilearn (http://scikit.ml) is a Python library for performing multi-label classification.

Related to the WEKA project, MOA (https://moa.cms.waikato.ac.nz/, in Java) is a popular open-source machine learning framework for data stream mining. Built on scikit-learn, MOA and MEKA, scikit-multiflow (https://github.com/scikit-multiflow, in Python) is a framework for learning from data streams and multi-output learning.

TensorLy (https://github.com/tensorly) is a Python library that provides a high-level API for tensor methods and deep tensorized neural networks. They can be scaled on multiple CPU or GPU machines.

Imbalanced-learn (https://github.com/scikit-learn-contrib/imbalanced-learn) is an open-source Python toolbox providing a wide range of methods to cope with the problem of imbalanced datasets.

Tensor Toolbox for MATLAB (http://www.tensortoolbox.org/) provides functions for manipulating dense, sparse, and structured tensors.

OpenXBOW (https://github.com/openXBOW/) is an open-source Java toolkit for generating bag-of-words (BoW) representations from multimodal input.

SnapVX (http://snap.stanford.edu/snapvx) is a fast and scalable python solver for large convex optimization problems defined on networks. It is based on the alternating direction method of multipliers (ADMM).

BlockBench (https://github.com/ooibc88/blockbench) is a benchmarking framework for quantitatively evaluating private blockchains.

## References


1. Berg, T. L., Berg, A. C., Edwards J., Maire M., White, R., Teh, Y.-W., Learned-Miller, E., & Forsyth, D. A. (2004). Names and faces in the news. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition* (Vol. 2, pp. 848–854).
2. Dhall, A., Goecke, R., Joshi, J., Wagner, M., & Gedeon, T. (2013). Emotion recognition in the wild challenge 2013. In *Proceedings of the 15th ACM on International Conference on Multimodal Interaction* (pp. 509–516). Sydney, Australia.
3. Hayat, K. (2018). Multimedia super-resolution via deep learning: A survey. *Digital Signal-Processing, 81*, 198–217.
4. Lauritzen, S. L., & Spiegelhalter, D. J. (1988). Local computations with probabilities on graphical structures and their application on expert systems. *Journal of the Royal Statistical Society Series B, 50*(2), 157–224.
5. Meshi, O., Jaimovich, A., Globerzon, A., & Friedman, N. (2009). Convexifying the bethe free energy. In *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence (UAI)*. Montreal, Canada.
6. Minka, T. (2001). *Expectation propagation for approximate Bayesian inference*. Doctoral dissertation, MIT Media Lab.
7. Mooij, J., & Kappen, H. (2007). Sufficient conditions for convergence of the sum-product algorithm. *IEEE Transactions on Information Theory, 53*, 4422–4437.
8. Phillips, P. J., Flynn, P. J., Scruggs, T., Bowyer, K. W., Chang, J., Hoffman, K., Marques, J., Min, J., & Worek, W. (2005). Overview of the face recognition grand challenge. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition* (Vol. 1, pp. 947–954).
9. Wainwright, M.J., Jaakkola, T. S., & Willsky, A.S. (2005). A new class of upper bounds on the log partition function. *IEEE Transactions on Information Theory, 51*(7), 2313–2335.
10. Yedidia, J. S., Freeman, W. T., & Weiss, Y. (2005). Constructing free energy approximations and generalized belief propagation algorithms. *IEEE Transactions on Information Theory, 51*, 2282–2312.
11. Yin, L., Wei, X., Sun, Y., Wang, J., & Rosato, M. J. (2006). A 3D facial expression database for facial behavior research. In *Proceedings of the 7th International Conference on Automatic Face and Gesture Recognition* (pp. 211–216).

# Index





K.-L. Du and M. N. S. Swamy, *Neural Networks and Statistical Learning*,
https://doi.org/10.1007/978-1-4471-7452-3

**M**



**N**

**Z**