Mathematics for Machine Learning Specialization Formula Sheet

Fady Morris Ebeid

April 30, 2020

Chapter 1 Linear Algebra

1 Vectors

1.1 Vector Operations

$\mathbf{r}+\mathbf{s}=\mathbf{s}+\mathbf{r}$ $\qquad 2\mathbf{r}=\mathbf{r}+\mathbf{r}$

1.2 Dot Product(Inner Product)

$$\mathbf{r}\cdot\mathbf{s}=\sum_i r_i s_i$$

Properties of the Dot Product

- Commutative: $\mathbf{r}\cdot\mathbf{s}=\mathbf{s}\cdot\mathbf{r}$
- Distributive: $\mathbf{r}\cdot(\mathbf{s}+\mathbf{t})=\mathbf{r}\cdot\mathbf{s}+\mathbf{r}\cdot\mathbf{t}$
- Associative: $\mathbf{r}\cdot(a\mathbf{s})=a(\mathbf{r}\cdot\mathbf{s})$
- $\mathbf{r}\cdot\mathbf{r}=\|\mathbf{r}\|^2=\sum_i r_i^2 \qquad \Rightarrow \qquad \|\mathbf{r}\|=\sqrt{\mathbf{r}\cdot\mathbf{r}}$
- The angle between two vectors: $\cos\theta=\dfrac{\mathbf{r}\cdot\mathbf{s}}{\|\mathbf{r}\|\|\mathbf{s}\|}$

1.3 Scalar and Vector Projection

Scalar projection: $\dfrac{\mathbf{r}\cdot\mathbf{s}}{\|\mathbf{r}\|}$ $\qquad$ Vector projection: $\dfrac{\mathbf{r}\cdot\mathbf{s}}{\mathbf{r}\cdot\mathbf{r}}\mathbf{r}$

1.4 Basis

A basis is a set of n vectors that:

1. Are not linear combinations of each other.
2. Span the space.

The space is then n-dimensional.

1.5 Linear Combination and Linear independence

The vector $\mathbf{w}$ is a linear combination of vectors $\mathbf{u}$ and $\mathbf{v}$ if

$$\mathbf{w}=a\mathbf{u}+b\mathbf{v}$$

Two 2D vectors $\mathbf{u}$ and $\mathbf{v}$ are linearly dependent if

$$\mathbf{u}=a\mathbf{v}$$

The three 3D vectors $\mathbf{a}$, $\mathbf{b}$ and $\mathbf{c}$ are linearly independent if

$$\det([\mathbf{a}\mid\mathbf{b}\mid\mathbf{c}])\neq 0$$

2 Matrices

2.1 Identity Matrix

$$I=\begin{bmatrix}1&0\\0&1\end{bmatrix}$$

2.2 Standard Basis

The **standard basis** or **natural basis** is the set of vectors whose coordinates are all zero, except one that equals 1

$$\hat{\mathbf{e}}_1=\begin{bmatrix}1\\0\end{bmatrix} \qquad \hat{\mathbf{e}}_2=\begin{bmatrix}0\\1\end{bmatrix}$$

$$[\hat{\mathbf{e}}_1\mid\hat{\mathbf{e}}_2]=I=\begin{bmatrix}1&0\\0&1\end{bmatrix}$$

2.3 Transforming Space

$$A\mathbf{r}=\mathbf{r}'$$
$$A(n\mathbf{r})=n(A\mathbf{r})=n\mathbf{r}'$$
$$A(\mathbf{r}+\mathbf{s})=A\mathbf{r}+A\mathbf{s}$$

Transforming basis vectors

$$A\hat{\mathbf{e}}_1=\mathbf{e}_1' \qquad A\hat{\mathbf{e}}_2=\mathbf{e}_2'$$

$$\begin{aligned}A(n\hat{\mathbf{e}}_1+m\hat{\mathbf{e}}_2)&=nA\hat{\mathbf{e}}_1+mA\hat{\mathbf{e}}_2\\&=n\mathbf{e}_1'\quad+m\mathbf{e}_2'\end{aligned}$$

2.4 Transformation Matrices and Composition

Clockwise rotation by θ : $R=\begin{bmatrix}\cos\theta&\sin\theta\\-\sin\theta&\cos\theta\end{bmatrix}$

Composition of transformations :

$$A_2A_1\mathbf{r}=\mathbf{r}'$$
$$C=A_2A_1$$
$$C\mathbf{r}=\mathbf{r}'$$

2.5 Determinant and Inverse

Let A be a 2×2 matrix such that $A=\begin{bmatrix}a&b\\c&d\end{bmatrix}$

Determinant :

$$\det(A)=|A|=ad-bc$$

Inverse :

$$A^{-1}=\frac{1}{|A|}\begin{bmatrix}d&-b\\-c&a\end{bmatrix}=\frac{1}{ad-bc}\begin{bmatrix}d&-b\\-c&a\end{bmatrix}$$

If $\det(A)=0$ then A^{-1} doesn't exist (A is not invertible or A is *singular*) and the basis of A are linearly dependent.

Solving Systems of Linear Equations :
A system of linear equations in the form $\mathbf{Ar}=\mathbf{s}$ can be solved for $\mathbf{r}$ using the equation :

$$\mathbf{r}=\mathbf{A}^{-1}\mathbf{s}$$

2.6 Matrix Multiplication

$$\underset{n\times n}{\begin{bmatrix}a_{11}&a_{12}&a_{13}&\dots&a_{1n}\\a_{21}&a_{22}&a_{23}&&a_{2n}\\a_{31}&a_{32}&a_{33}&&a_{3n}\\\vdots&&&\ddots&\\a_{n1}&a_{n2}&a_{n3}&&a_{nn}\end{bmatrix}}\ \underset{n\times n}{\begin{bmatrix}b_{11}&b_{12}&b_{13}&\dots&b_{1n}\\b_{21}&b_{22}&b_{23}&&b_{2n}\\b_{31}&b_{32}&b_{33}&&b_{3n}\\\vdots&&&\ddots&\\b_{n1}&b_{n2}&b_{n3}&&b_{nn}\end{bmatrix}}=\underset{n\times n}{\begin{bmatrix}ab_{11}&ab_{12}&ab_{13}&\dots&ab_{1n}\\ab_{21}&ab_{22}&ab_{23}&&ab_{2n}\\ab_{31}&ab_{32}&ab_{33}&&ab_{3n}\\\vdots&&&\ddots&\\ab_{n1}&ab_{n2}&ab_{n3}&&ab_{nn}\end{bmatrix}}$$

Summation Convention for multiplying matrices A and B :

$$ab_{ik}=\sum_j a_{ij}b_{jk}$$

Einstein summation convention (without big sigma):

$$ab_{ik}=a_{ij}b_{jk}$$

3 Linear Mappings

3.1 Change of Basis

Changing from standard basis to a new basis $\mathbf{b}_1$ and $\mathbf{b}_2$.
The columns of the transformation matrix B are the new basis vectors in the standard coordinate system.

$$B=\begin{bmatrix}\mathbf{b}_1&\mathbf{b}_2\end{bmatrix}$$

$$\mathbf{r}=B\mathbf{r}'$$

Where $\mathbf{r}'$ is the vector in B-basis, and $\mathbf{r}$ is the vector in *standard basis*.

$$\mathbf{r}'=B^{-1}\mathbf{r}$$

The components of the vector $\mathbf{r}'$ are the scalar projections of vectors $\mathbf{b}_1$ and $\mathbf{b}_2$ onto $\mathbf{r}$

$$\mathbf{r}'_x=\text{comp}_{\mathbf{b}_1}\mathbf{r}$$
$$\mathbf{r}'_y=\text{comp}_{\mathbf{b}_2}\mathbf{r}$$

If a matrix $\mathbf{A}$ is *orthonormal* (all the columns are of unit size and orthogonal to each other) then,

$$A^{\mathsf{T}}=A^{-1} \qquad \text{and} \qquad A^{\mathsf{T}}A=I$$

Transformation in a Changed Basis

To do a transformation R_B on a vector $\mathbf{r}'$ in a changed basis B using a transformation R in standard basis

$$R_B=B^{-1}RB$$
$$R_B\mathbf{r}'=B^{-1}RB\mathbf{r}'$$

3.2 Gram-Schmidt Process for Constructing an Orthonormal Basis

Start with n linearly independent basis vectors $\mathbf{v} = \mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n$. Then

$$\mathbf{e}_1 = \frac{\mathbf{v}_1}{\|\mathbf{v}_1\|}$$

$$\mathbf{u}_2 = \mathbf{v}_2 - (\mathbf{v}_2 \cdot \mathbf{e}_1)\mathbf{e}_1 \quad \Rightarrow \quad \mathbf{e}_2 = \frac{\mathbf{u}_2}{\|\mathbf{u}_2\|}$$

$$\mathbf{u}_3 = \mathbf{v}_3 - (\mathbf{v}_3 \cdot \mathbf{e}_1)\mathbf{e}_1 - (\mathbf{v}_3 \cdot \mathbf{e}_2)\mathbf{e}_2 \quad \Rightarrow \quad \mathbf{e}_3 = \frac{\mathbf{u}_3}{\|\mathbf{u}_3\|}$$

$\ldots$ and so on.

3.3 Transformation in a Plane or Other Object

1. Transform into the basis referred to the reflection plane, or whichever; E^{-1}
2. Do the reflection or other transformation, in the plane of the object T_E
3. Transform back into the original basis E

So, our transformed vector

$$\mathbf{r}' = ET_E E^{-1}\mathbf{r}$$

4 Eigenvalues and Eigenvectors

Eigen is a German word means *characteristic*.
To investigate the characteristics of the $n \times n$ matrix A, find a solution to the equation,

$$A\mathbf{x} = \lambda\mathbf{x}$$

Where λ is a scalar *eigenvalue* and $\mathbf{x}$ is an *eigenvector*.
An $n \times n$ matrix must have n eigen values $\lambda_1 \ldots \lambda_n$. The n eigen values might be distinct or same.
Eigenvalues will satisfy the following condition

$$(A - \lambda I)\mathbf{x} = \vec{\mathbf{0}}$$

$$\therefore \det(A - \lambda I) = 0$$

For a 2×2 matrix $A = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$

$$\det(\begin{bmatrix} a-\lambda & b \\ c & d-\lambda \end{bmatrix}) = 0$$

$$(a-\lambda)(d-\lambda) - bc = 0$$

$$\therefore \underbrace{\lambda^2 - (a+d)\lambda + ad - bc}_{\text{Characteristic Polynomial of degree 2}} = 0$$

The *characteristic polynomial* is of degree n for $n \times n$ matrix.

4.1 Changing to Eigenbasis

The purpose of this technique is to simplify the computation of transformations that require raising matrices to large powers. Calculating the power of a *diagonal matrix* is easy

$$D = \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{bmatrix} \qquad D^n = \begin{bmatrix} \lambda_1^n & 0 & 0 \\ 0 & \lambda_2^n & 0 \\ 0 & 0 & \lambda_3^n \end{bmatrix}$$

A transformation matrix T^n can be decomposed as follows

$$T^n = CD^nC^{-1}$$

Where C is a matrix with *eigenbasis* and D is a diagonal matrix with *eigenvalues*.

$$D = C^{-1}TC$$

4.2 Page Rank Algorithm

To find the dominant eigenvector of link matrix L, the Power Method can be iteratively applied, starting from a uniform initial vector $\mathbf{r}$.

$$\mathbf{r}^{i+1} = L\mathbf{r}^i$$

A damping factor, d, can be implemented to stabilize this method as follows.

$$\mathbf{r}^{i+1} = dL\mathbf{r}^i + \frac{1-d}{n}$$

Chapter 2
Multivariate Calculus

1 Common Derivatives

$\frac{d}{dx}\left(\frac{1}{x}\right) = -\frac{1}{x^2}$
$\frac{d}{dx}(\sin x) = \cos x$
$\frac{d}{dx}(\cos x) = -\sin x$
$\frac{d}{dx}e^x = e^x$

2 Basic Rules of Differentiation

- **Sum Rule:** $\frac{d}{dx}(f(x) + g(x)) = \frac{d}{dx}(f(x)) + \frac{d}{dx}(g(x))$
- **Power Rule:** Given $f(x) = ax^b$, then $f'(x) = abx^{(b-1)}$
- **Product Rule:** Given $h(x) = f(x)g(x)$, then $h'(x) = f'(x)g(x) + f(x)g'(x)$
- **Chain Rule:** The derivative of the composite function $F(x) = f(g(x))$ is given by :

  $$F'(x) = f'(g(x)) \cdot g'(x)$$

  In *Lebniz notation*, if $y = f(u)$ and $u = g(x)$ then,

  $$\frac{dy}{dx} = \frac{dy}{du}\frac{du}{dz}$$

- **Total derivative:**
  For the function $f(x, y, z, \ldots)$, where each variable is a function of the parameter t, then the total derivative is

  $$\frac{\partial f}{\partial t} = \frac{\partial f}{\partial x}\frac{\partial x}{\partial t} + \frac{\partial f}{\partial y}\frac{\partial y}{\partial t} + \frac{\partial f}{\partial z}\frac{\partial z}{\partial t} + \ldots$$

3 Derivative Structures

3.1 Jacobian

Given $f(x, y, z)$

$$\mathbf{J}_f = \begin{bmatrix} \frac{\partial f}{\partial x} & \frac{\partial f}{\partial y} & \frac{\partial f}{\partial z} \end{bmatrix}$$

Jacobian is a row vector.

Jacobian of a Vector Function

Given $\mathbf{r}(x, y) = \langle u(x, y), v(x, y) \rangle$, where u and v are differntiable functions.

$$\mathbf{J} = \begin{bmatrix} \frac{\partial u}{\partial x} & \frac{\partial u}{\partial y} \\ \frac{\partial v}{\partial x} & \frac{\partial v}{\partial y} \end{bmatrix}$$

If $u(x, y)$ and $v(x, y)$ are linear functions, the Jacobian is a linear transformation matrix between (x, y) vector space and (u, v) vector space, then

$$\mathbf{J} = \begin{bmatrix} \mathbf{r}(1, 0) & \mathbf{r}(0, 1] \end{bmatrix}$$

Jacobian in Polar Coordinates

$x = r\cos\theta, y = r\sin\theta$

$$\mathbf{J} = \begin{bmatrix} \frac{\partial x}{\partial r} & \frac{\partial x}{\partial \theta} \\ \frac{\partial y}{\partial r} & \frac{\partial y}{\partial \theta} \end{bmatrix} = \begin{bmatrix} \cos\theta & -r\sin\theta \\ \sin\theta & r\cos\theta \end{bmatrix}$$

$$|\mathbf{J}| = r(\cos^2\theta + \sin^2\theta) = r$$

Numerical Approximation of Jacobian

$$\mathbf{J} = \begin{bmatrix} \frac{f(x + \Delta x, y) - f(x, y)}{\Delta x} & \frac{f(x, y + \Delta y) - f(x, y)}{\Delta y} \end{bmatrix}$$

3.2 Hessian

$$\mathbf{H}_f = \begin{bmatrix} f_{xx} & f_{xy} & f_{xz} \\ f_{yx} & f_{yy} & f_{yz} \\ f_{zx} & f_{zy} & f_{zz} \end{bmatrix}$$

Hessian matrix is a *symmetric* matrix.

Multivariate Second Derivative Test

If the jacobian $\mathbf{J}_f = \mathbf{0}$ at a point $P = (a, b)$ then

- If $|\mathbf{H}_f| > 0$, then
  - If $f_{xx} > 0$, then the function f has a *local minimum* at P
  - If $f_{xx} < 0$, then the function f has a *local maximum* at P
- If $|\mathbf{H}_f| < 0$, then the point P is a saddle point.

---

4 Multivariate Chain Rule

Given a function $f(\mathbf{x}(\mathbf{u}(t)))$, where

$$f(\mathbf{x}) = f(x_1, x_2)$$

$$\mathbf{x}(\mathbf{u}) = \begin{bmatrix} x_1(u_1, u_2) \\ x_2(u_1, u_2) \end{bmatrix}$$

$$\mathbf{u}(t) = \begin{bmatrix} u_1(t) \\ u_2(t) \end{bmatrix}$$

$$\frac{df}{dt} = \frac{\partial f}{\partial \mathbf{x}} \frac{\partial \mathbf{x}}{\partial \mathbf{u}} \frac{d\mathbf{u}}{dt}$$

$$= \begin{bmatrix} \frac{\partial f}{\partial x_1} & \frac{\partial f}{\partial x_2} \end{bmatrix} \begin{bmatrix} \frac{\partial x_1}{\partial u_1} & \frac{\partial x_1}{\partial u_2} \\ \frac{\partial x_2}{\partial u_1} & \frac{\partial x_2}{\partial u_2} \end{bmatrix} \begin{bmatrix} \frac{du_1}{dt} \\ \frac{du_2}{dt} \end{bmatrix}$$

---

5 Neural Networks

5.1 Activation Functions

The activation function denoted $\sigma(x)$ can be any of the following :

- Hyperbolic tangent :

  $\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$

  $\frac{d}{dx}\tanh(x) = \frac{1}{\cosh^2(x)} = \frac{4}{(e^x + e^{-x})^2}$
- Logistic function:

  $f(x) = \frac{1}{1 + e^{-x}}$

  $\frac{d}{dx} f(x) = \frac{e^x}{(1 + e^x)^2} = f(x)(1 - f(x)) = f(x)f(-x)$
- Rectified Linear Unit(ReLU):

  $f(x) = \begin{cases} 0 & \text{for} \quad x < 0 \\ x & \text{for} \quad x \geq 0 \end{cases}$

  $f'(x) = \begin{cases} 0 & \text{for} \quad x < 0 \\ 1 & \text{for} \quad x \geq 0 \end{cases}$

5.2 Feed-Forward Propagation

$$\mathbf{a}^{(L)} = \sigma(\mathbf{W}^{(L)}\mathbf{a}^{(L-1)} + \mathbf{b}^{(L)})$$

$$a_i^{(L)} = \sigma\left[\sum_{j=0}^{n} W_{ij}^{(L)} a_{ij}^{(L-1)} + b\right]$$

5.3 Cost Function

$$C = \frac{1}{m}\|\mathbf{a}^{(N)} - \mathbf{y}\|^2$$

Where N is the output layer, m is the total number of training samples.

5.4 Backpropagation

$$\frac{\partial C}{\partial \mathbf{W}^{(i)}} = \frac{\partial C}{\partial \mathbf{a}^{(N)}} \underbrace{\frac{\partial \mathbf{a}^{(N)}}{\partial \mathbf{a}^{(N-1)}} \frac{\partial \mathbf{a}^{(N-1)}}{\partial \mathbf{a}^{(N-2)}} \cdots \frac{\partial \mathbf{a}^{(i+1)}}{\partial \mathbf{a}^{(i)}}}_{\text{from layer } N \text{ to layer } i} \frac{\partial \mathbf{a}^{(i)}}{\partial \mathbf{z}^{(i)}} \frac{\partial \mathbf{z}^{(i)}}{\partial \mathbf{W}^{(i)}}$$

$$\frac{\partial C}{\partial \mathbf{b}^{(i)}} = \frac{\partial C}{\partial \mathbf{a}^{(N)}} \underbrace{\frac{\partial \mathbf{a}^{(N)}}{\partial \mathbf{a}^{(N-1)}} \frac{\partial \mathbf{a}^{(N-1)}}{\partial \mathbf{a}^{(N-2)}} \cdots \frac{\partial \mathbf{a}^{(i+1)}}{\partial \mathbf{a}^{(i)}}}_{\text{from layer } N \text{ to layer } i} \frac{\partial \mathbf{a}^{(i)}}{\partial \mathbf{z}^{(i)}} \frac{\partial \mathbf{z}^{(i)}}{\partial \mathbf{b}^{(i)}}$$

Where $\frac{\partial \mathbf{a}^{(n)}}{\partial \mathbf{a}^{(n-1)}}$ can itself be expanded to :

$$\frac{\partial \mathbf{a}^{(n)}}{\mathbf{a}^{(n-1)}} = \frac{\partial \mathbf{a}^{(n)}}{\partial \mathbf{z}^{(n)}} \frac{\partial \mathbf{z}^{(n)}}{\partial \mathbf{a}^{(n-1)}}$$

5.5 Neural Network Partial Derivatives

$$\frac{\partial C}{\partial \mathbf{a}^{(N)}} = \frac{2}{m}(\mathbf{a}^{(N)} - \mathbf{y})$$

$$\frac{\partial \mathbf{a}^{(i)}}{\partial \mathbf{z}^{(i)}} = \sigma'(\mathbf{z}^{(i)})$$

$$\frac{\partial \mathbf{z}^{(i)}}{\partial \mathbf{a}^{(i-1)}} = \mathbf{W}^{(i)}$$

$$\frac{\partial \mathbf{z}^{(i)}}{\partial \mathbf{W}^{(i)}} = \mathbf{a}^{(i-1)}$$

$$\frac{\partial \mathbf{z}^{(i)}}{\partial \mathbf{b}^{(i)}} = \vec{\mathbf{1}}$$

---

6 Taylor Series

6.1 Univariate

$$f(x) = \sum_{n=0}^{\infty} \frac{f^{(n)}(c)}{n!}(x - c)^n$$

$$= f(c) + f'(c)(x - c) + \frac{f''(c)}{2!}(x - c)^2 + \frac{f'''(c)}{3!}(x - c)^3 + \dots$$

Linearization

$$f(x + \Delta x) = \sum_{n=0}^{\infty} \frac{f^{(n)}(x)}{n!}\Delta x^n$$

$$= f(x) + f'(x)\Delta x + \frac{f''(x)}{2!}\Delta x^2 + \frac{f'''(x)}{3!}\Delta x^3 + \dots$$

$f(x + \Delta x) = f(x) + f'(x)(\Delta x) + O(\Delta x^2)$

$g_1(x + \Delta x) = f(x) + f'(x)(\Delta x)$

$$f'(x) = \frac{f(x + \Delta x) - f(x)}{\Delta x} - f''(x)\Delta x - f^{(3)}(x)\Delta x^2 - \dots$$

$$= \frac{f(x + \Delta x) - f(x)}{\Delta x} + O(\Delta x)$$

6.2 Multivariate

$\mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix}$

$$\begin{aligned} f(\mathbf{x} + \Delta\mathbf{x}) = \; & f(x + \Delta x, y + \Delta y) \\ = \; & f(x, y) \\ & + \partial_x f(x, y)\Delta x + \partial_y f(x, y)\Delta y \\ & + \frac{1}{2}\Big[\partial_{xx} f(x, y)\Delta x^2 \\ & + 2\partial_{xy} f(x, y)\Delta x \Delta y + \partial_{yy} f(x, y)\Delta y^2\Big] \\ = \; & f(x, y) \\ & + \begin{bmatrix} \partial_x f(x, y) & \partial_y f(x, y) \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} \\ & + \frac{1}{2}\begin{bmatrix} \Delta x & \Delta y \end{bmatrix} \begin{bmatrix} \partial_{xx} f(x, y) & \partial_{xy} f(x, y) \\ \partial_{yx} f(x, y) & \partial_{yy} f(x, y) \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} \\ = \; & f(\mathbf{x}) + \mathbf{J}_f \Delta\mathbf{x} + \frac{1}{2}\Delta\mathbf{x}^t \mathbf{H}_f \Delta\mathbf{x} + \dots \end{aligned}$$

$$f(\mathbf{x}) = f(\mathbf{c}) + \mathbf{J}_f(\mathbf{c})(\mathbf{x} - \mathbf{c}) + \frac{1}{2}(\mathbf{x} - \mathbf{c})^t \mathbf{H}_f(\mathbf{c})(\mathbf{x} - \mathbf{c}) + \dots$$

---

7 Optimization and Vector Calculus

7.1 Newton-Raphson Method

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}$$

7.2 Gradient

$$\nabla f = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \\ \frac{\partial f}{\partial z} \end{bmatrix}$$

7.3 Directional Gradient

$$\nabla f \cdot \hat{\mathbf{r}}$$

7.4 Gradient Descent

$$\mathbf{s}_{n+1} = \mathbf{s}_n - \gamma \nabla f(\mathbf{s}_n)$$

7.5 Lagrange Multipliers

$\mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix}$

This method is used to optimize a function $f(\mathbf{x})$ subject to a constraint $g(\mathbf{x}) = 0$.
Therefore,

$$\nabla f(\mathbf{x}) = \lambda \nabla g(\mathbf{x})$$

$$\begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix} = \lambda \begin{bmatrix} \frac{\partial g}{\partial x} \\ \frac{\partial g}{\partial y} \end{bmatrix}$$

This equation, along with $g(\mathbf{x}) = 0$ can be put into a single vector equation

$$\mathcal{L}(x, y, \lambda) = f(\mathbf{x}) - \lambda g(\mathbf{x})$$

$$\nabla \mathcal{L}(x, y, \lambda) = \begin{bmatrix} \frac{\partial f}{\partial x} - \lambda \frac{\partial g}{\partial x} \\ \frac{\partial f}{\partial y} - \lambda \frac{\partial g}{\partial y} \\ -g(\mathbf{x}) \end{bmatrix} = 0$$

We can solve the previous equaiton using root finding methods, such as the Newton-Raphson method.

7.6 Linear Regression

Simple Linear Regression

$\mathbf{a} = \begin{bmatrix} m \\ c \end{bmatrix}$

$$y = y(x; a_i) = mx_i + c$$
$$r_i = y_i - mx_i - c$$

$$\chi^2 = \sum_i r_i^2 = \sum_i (y_i - mx_i - c)^2$$

$$\nabla \chi^2 = \begin{bmatrix} \frac{\partial \chi^2}{\partial m} \\ \frac{\partial \chi^2}{\partial c} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} = \begin{bmatrix} -2 \sum_i x_i (y_i - mx_i - c) \\ -2 \sum_i (y_i - mx_i - c) \end{bmatrix}$$

$$c = \bar{y} - m\bar{x} \qquad \sigma_c \approx \sigma_m \sqrt{\bar{x}^2 + \frac{1}{n} \sum_i (x - \bar{x})^2}$$

$$m = \frac{\sum (x - \bar{x}) y}{\sum (x - \bar{x})^2} \qquad \sigma_m^2 \approx \frac{\chi^2}{\sum (x - \bar{x})^2 (n - 2)}$$

Normalizing x_i, the previous equations become:

$y = (m \pm \sigma_m)(x - \bar{x}) + (b \pm \sigma_b)$

$$m = \frac{\sum (x - \bar{x}) y}{\sum (x - \bar{x})^2} \qquad \sigma_m^2 \approx \frac{\chi^2}{\sum (x - \bar{x})^2 (n - 2)}$$

$$b = \bar{y} \qquad \sigma_b^2 \approx \frac{\chi^2}{n(n - 2)}$$

Generalized non-linear least squares χ^2 minimization

$$\chi^2 = \sum_i^n \frac{[y_i - y(x_i; a_k)]^2}{\sigma_i} \qquad \begin{matrix} k = 1 \dots m \\ i = 1 \dots n \end{matrix}$$

Criterion : $\nabla \chi^2 = 0$

$$\begin{aligned} a_{\text{next}} &= a_{\text{cur}} - \gamma \nabla \chi^2 \\ &= a_{\text{cur}} + \gamma \sum_i^n \frac{[y_i - y(x_i; a_k)]}{\sigma_i} \frac{\partial y}{\partial a_k} \end{aligned}$$

Chapter 3
Principal Component Analysis

1 Statistics

1.1 Mean Values of Higher-Dimensional Data Sets

Given a dataset $\mathcal{D} = \{\mathbf{x}_1 \dots \mathbf{x}_N\}, \mathbf{x}_n \in \mathbb{R}^D$, we compute the mean of the dataset as

$$\boldsymbol{\mu} = \mathbf{E}[\mathcal{D}] = \frac{1}{N} \sum_{n=1}^{N} \mathbf{x}_n \tag{3.1}$$

1.2 Variance

1D Data Sets

Given a dataset $\mathcal{D} = \{x_1, \dots, x_N\}, x_n \in \mathbb{R}$, we compute the variance of the dataset as

$$\mathbb{V}[\mathcal{D}] = \frac{1}{N} \sum_{n=1}^{N} (x_n - \mathbf{E}[\mathcal{D}])^2 = \frac{1}{N} \sum_{n=1}^{N} x_n^2 - \mathbf{E}[\mathcal{D}]^2$$

$$\sigma^2 = \frac{1}{N} \sum_{n=1}^{N} (x_n - \mu)^2 = \frac{1}{N} \sum_{n=1}^{N} x_n^2 - \mu^2 \tag{3.2}$$

where $\sigma^2 = \mathbb{V}[\mathcal{D}]$ is the variance of the dataset.

1.3 Higher-Dimentional Data Sets

Given a dataset $\mathcal{D} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}, \mathbf{x}_n \in \mathbb{R}^D$, we compute the variance of dataset as

$$\mathbb{V}[\mathcal{D}] = \frac{1}{N} \sum_{n=1}^{N} (\mathbf{x}_n - \boldsymbol{\mu})(\mathbf{x}_n - \boldsymbol{\mu})^\mathsf{T} \in \mathbb{R}^{D \times D}$$

where $\boldsymbol{\mu} \in \mathbb{R}^D$ is the mean value of the dataset.

Example

If every element $\mathbf{x} \in \mathbb{R}^n$ of the dataset $\mathcal{D}$ is an n-dimensional vector $\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$

the covariance matrix will look as follows:

$$\mathbb{V}[\mathcal{D}] = \begin{bmatrix} \text{var}[x_1] & \text{cov}[x_1, x_2] & \dots & \text{cov}[x_1, x_n] \\ \text{cov}[x_2, x_1] & \text{var}[x_2] & \dots & \text{cov}[x_2, x_n] \\ \vdots & \vdots & \ddots & \\ \text{cov}[x_n, x_1] & \text{cov}[x_n, x_2] & & \text{var}[x_n] \end{bmatrix}$$

where
$\text{cov}[x_i, x_j] = \mathbf{E}[(x_i - \mathbf{E}[x_i])(x_j - \mathbf{E}[x_j])]$

2 Effect of Linear Transformations

2.1 1D Data Sets

Consider a dataset $\mathcal{D} = \{x_1, \dots, x_N\}, x_n \in \mathbb{R}$

$$\begin{aligned} \mathbf{E}[\ \mathcal{D} + b] &= \ \mathbf{E}[\mathcal{D}] + b \\ \mathbf{E}[a\mathcal{D}] &= a\mathbf{E}[\mathcal{D}] \\ \therefore \mathbf{E}[a\mathcal{D} + b] &= a\mathbf{E}[\mathcal{D}] + b \end{aligned}$$

$$\begin{aligned} \mathbb{V}[\ \mathcal{D} + b] &= \ \mathbb{V}[\mathcal{D}] \\ \mathbb{V}[a\mathcal{D}] &= a^2 \mathbb{V}[\mathcal{D}] \\ \therefore \mathbb{V}[a\mathcal{D} + b] &= a^2 \mathbb{V}[\mathcal{D}] \end{aligned}$$

2.2 Higher-Dimentional Data Sets

Consider a dataset $\mathcal{D} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}, \mathbf{x}_n \in \mathbb{R}^D$, with

$$\mathbf{E}[\mathcal{D}] = \boldsymbol{\mu}$$
$$\mathbb{V}[\mathcal{D}] = \boldsymbol{\Sigma}$$

If we transform every $\mathbf{x}_i \in \mathcal{D}$ according to

$$\mathbf{x}_i' = \mathbf{A}\mathbf{x}_i + \mathbf{b}$$

for a gievn $\mathbf{A}, \mathbf{b}$, then

$$\mathbf{E}[\mathcal{D}'] = \mathbf{A}\boldsymbol{\mu} + \mathbf{b}$$
$$\mathbb{V}[\mathcal{D}'] = \mathbf{A}\boldsymbol{\Sigma}\mathbf{A}^\mathsf{T}$$

where $\mathcal{D}' = \{\mathbf{x}_1', \dots, \mathbf{x}_N'\}$

3 The Dot Product

3.1 Definition

$$\mathbf{x} \cdot \mathbf{y} = \mathbf{x}^\mathsf{T}\mathbf{y} = \sum_{i=1}^{N} x_i y_i, \qquad \mathbf{x}, \mathbf{y} \in \mathbb{R}^N$$

3.2 Lengths and Distances

The **length** of a vector $\mathbf{x}$ is

$$\|\mathbf{x}\| = \sqrt{\mathbf{x}^\mathsf{T}\mathbf{x}} = \sqrt{\sum_{i=1}^{N} x_i^2}$$

The distance between two vectors $\mathbf{x}, \mathbf{y}$ is given by

$$d(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\| = \sqrt{(\mathbf{x} - \mathbf{y})^\mathsf{T}(\mathbf{x} - \mathbf{y})}$$

3.3 Angles

The angle θ between two vectors $\mathbf{x}, \mathbf{y}$ can be computed via

$$\cos\theta = \frac{\mathbf{x}^\mathsf{T}\mathbf{y}}{\|\mathbf{x}\|\|\mathbf{y}\|}$$

4 The Inner Product

4.1 Definition

Consider a vector space V. A function that is a *positive definite*, *symmetric bilinear* mapping $\langle \cdot, \cdot \rangle : V \times V \to \mathbb{R}$ is called an **inner product** on V. This function is

- **Symmetric** : For all $\mathbf{x}, \mathbf{y} \in V$ it holds that $\langle \mathbf{x}, \mathbf{y} \rangle = \langle \mathbf{y}, \mathbf{x} \rangle$
- **Positive definite** For all $\mathbf{x} \in V \setminus \{\vec{\mathbf{0}}\}$ it holds that

$$\langle \mathbf{x}, \mathbf{x} \rangle > 0, \qquad \langle \vec{\mathbf{0}}, \vec{\mathbf{0}} \rangle = 0$$

- **Bilinear** For all $\mathbf{x}, \mathbf{y}, \mathbf{z} \in V, \lambda \in \mathbb{R}$

$$\langle \lambda\mathbf{x} + \mathbf{y}, \mathbf{z} \rangle = \lambda\langle \mathbf{x}, \mathbf{z} \rangle + \langle \mathbf{y}, \mathbf{z} \rangle$$

$$\langle \mathbf{x}, \lambda\mathbf{y} + \mathbf{z} \rangle = \lambda\langle \mathbf{x}, \mathbf{y} \rangle + \langle \mathbf{x}, \mathbf{z} \rangle$$

$\langle \mathbf{x}, \mathbf{y} \rangle = \mathbf{x}^\mathsf{T}\mathbf{I}\mathbf{y}$ (dot product)

The function defined by

$$\begin{aligned}\langle \mathbf{x}, \mathbf{y} \rangle &= \mathbf{x}^\mathsf{T}\mathbf{A}\mathbf{y} \\ &= \begin{bmatrix} x_1 & x_2 \end{bmatrix} \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} \\ &= ax_1y_1 + bx_1y_2 + cx_2y_1 + dx_2y_2 \\ &= \text{trace}\left(\mathbf{y}\mathbf{x}^\mathsf{T}\mathbf{A}\right)\end{aligned}$$

is an inner product, if and only if, it is symmetric, positive definite and bilinear.

4.2 Lengths and Distances

Consider a vector space V with an inner product $\langle \cdot, \cdot \rangle$.

- The **length** or **norm** of a vector $\mathbf{x} \in V$ is

$$\|\mathbf{x}\| = \sqrt{\langle \mathbf{x}, \mathbf{x} \rangle}$$

- The **distance** between two vectors $\mathbf{x}, \mathbf{y} \in V$ is given by

$$d(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\| = \sqrt{\langle \mathbf{x} - \mathbf{y}, \mathbf{x} - \mathbf{y} \rangle}$$

  If the inner product is a dot product, the distance is the *euclidean distance*

Properties of the *norm* :

- $\|\lambda\mathbf{x}\| = |\lambda|\|\mathbf{x}\|$
- **Triange Inequality:** $\|\mathbf{x} + \mathbf{y}\| \leq \|\mathbf{x}\| + \|\mathbf{y}\|$
- **Cauchy–Schwarz inequality:** $|\langle \mathbf{x}, \mathbf{y} \rangle| \leq \|\mathbf{x}\|\|\mathbf{y}\|$

4.3 Angles

Consider a vector space V with an inner product $\langle \cdot, \cdot \rangle$.
The angle θ between two vectors $\mathbf{x}, \mathbf{y} \in V$ can be computed via

$$\cos\theta = \frac{\langle \mathbf{x}, \mathbf{y} \rangle}{\|\mathbf{x}\|\|\mathbf{y}\|} = \frac{\langle \mathbf{x}, \mathbf{y} \rangle}{\sqrt{\langle \mathbf{x}, \mathbf{x} \rangle \langle \mathbf{y}, \mathbf{y} \rangle}}$$

where the *length/norm*

$$\|\mathbf{x}\| = \sqrt{\langle \mathbf{x}, \mathbf{x} \rangle}$$

is defined via the inner product.

5 Inner Products of Functions and Random Variables

5.1 Continuous Functions

$$\langle u, v \rangle = \int_a^b u(x)v(x)\, dx$$

If that integral evaluates to 0, then the two functions are *orthogonal* to each other.

5.2 Random Variables

For two random variables $\mathbf{x}$ and $\mathbf{y}$ wich are uncorrelated then

$$\text{var}[\mathbf{x} + \mathbf{y}] = \text{var}[\mathbf{x}] + \text{var}[\mathbf{y}]$$

Definition

If

$$\langle \mathbf{x}, \mathbf{y} \rangle = \text{cov}[\mathbf{x}, \mathbf{y}]$$

The covariance is

- **Symmetric:** $\text{cov}[\mathbf{x}, \mathbf{y}] = \text{cov}[\mathbf{y}, \mathbf{x}]$
- **Positive Definite:**
- **Bilinear:**

$$\text{cov}[\lambda\mathbf{x} + \mathbf{y}, \mathbf{z}] = \lambda\text{cov}[\mathbf{x}, \mathbf{z}] + \text{cov}[\mathbf{y}, \mathbf{z}]$$

$$\text{cov}[\mathbf{x}, \lambda\mathbf{y} + \mathbf{z}] = \lambda\text{cov}[\mathbf{x}, \mathbf{y}] + \text{cov}[\mathbf{x}, \mathbf{z}]$$

Length

$$\|\mathbf{x}\| = \sqrt{\langle \mathbf{x}, \mathbf{x} \rangle} = \sqrt{\text{cov}[\mathbf{x}, \mathbf{x}]} = \sqrt{\text{var}[\mathbf{x}]} = \sigma[\mathbf{x}]$$

Angle Between Two Random Variables

$$\cos\theta = \frac{\langle \mathbf{x}, \mathbf{y} \rangle}{\|\mathbf{x}\|\|\mathbf{y}\|} = \frac{\text{cov}[\mathbf{x}, \mathbf{y}]}{\sqrt{\text{var}[\mathbf{x}]\text{var}[\mathbf{y}]}}$$

It evaluates to zero when $\text{cov}[\mathbf{x}, \mathbf{y}] = 0$ (when the two random variables are uncorrelated).

6 Orthogonal Projections

6.1 Projection onto 1D subspaces

Consider a n-dimensional vector space V with the dot product at the inner product and a 1-dimensional subspace U of V. With a basis vector $\mathbf{b} \in \mathbb{R}^n$ of U, we obtain the **orthogonal projection** of any vector $\mathbf{x} \in V$ onto U via

$$\pi_U(\mathbf{x}) = \lambda\mathbf{b}, \qquad \lambda = \frac{\mathbf{b}^\mathsf{T}\mathbf{x}}{\mathbf{b}^\mathsf{T}\mathbf{b}} = \frac{\mathbf{b}^\mathsf{T}\mathbf{x}}{\|\mathbf{b}\|^2}$$

where λ is the 1D **coordinate** of $\pi_U(\mathbf{x})$ with respect to $\mathbf{b}$.
The **projection matrix** $\mathbf{P}_\pi \in \mathbb{R}^n$ is

$$\mathbf{P}_\pi = \frac{\mathbf{b}\mathbf{b}^\mathsf{T}}{\mathbf{b}^\mathsf{T}\mathbf{b}} = \frac{\mathbf{b}\mathbf{b}^\mathsf{T}}{\|\mathbf{b}\|^2}$$

such that

$$\pi_U(\mathbf{x}) = \mathbf{P}_\pi\mathbf{x}$$

for all $\mathbf{x} \in V$

Properties

1. $\langle \mathbf{b}, \pi_U(\mathbf{x}) - \mathbf{x} \rangle = 0$ (orthogonality)
2. $\pi_U(\pi_U(\mathbf{x})) = \pi_U(\mathbf{x})$

In the special case that $\mathbf{b}$ is normalized:

$$\mathbf{P}_\pi = \mathbf{b}\mathbf{b}^\mathsf{T}$$

$$\lambda = \mathbf{b}^\mathsf{T}\mathbf{x} \tag{3.3}$$

6.2 Projection Onto k-Dimensional Subspaces

Consider an n-dimensional vector space V with the dot product at the inner product and a k-dimensional subspace U of V. With basis vectors $\mathbf{b}_1, \ldots, \mathbf{b}_k$ of U, we obtain the **orthogonal projection** of any vector $\mathbf{x} \in V$ onto U via

$$\pi_U(\mathbf{x}) = \mathbf{B}\boldsymbol{\lambda}, \qquad \boldsymbol{\lambda} = (\mathbf{B}^\mathsf{T}\mathbf{B})^{-1}\mathbf{B}^\mathsf{T}\mathbf{x}$$

where

$$\mathbf{B} = [\mathbf{b}_1 | \ldots | \mathbf{b}_k] \in \mathbb{R}^{n \times k} \qquad \boldsymbol{\lambda} = \begin{bmatrix} \lambda_1 \\ \vdots \\ \lambda_k \end{bmatrix} \in \mathbb{R}^k$$

where $\boldsymbol{\lambda}$ is the **coordinate vector** of $\pi_U(\mathbf{x})$ with respect to the basis $\mathbf{b}_1, \ldots, \mathbf{b}_k$ of U.
The **projection matrix** $\mathbf{P}_\pi \in \mathbb{R}^n$ is

$$\mathbf{P}_\pi = \mathbf{B}(\mathbf{B}^\mathsf{T}\mathbf{B})^{-1}\mathbf{B}^\mathsf{T}$$

such that

$$\pi_U(\mathbf{x}) = \mathbf{P}_\pi\mathbf{x} \tag{3.4}$$

for all $\mathbf{x} \in V$

Notes

- In the special case of $\mathbf{B}$ is an **orthonormal basis**:

$$\mathbf{P}_\pi = \mathbf{B}\mathbf{B}^\mathsf{T} \tag{3.5}$$

$$\boldsymbol{\lambda} = \mathbf{B}^\mathsf{T}\mathbf{x} \tag{3.6}$$

- The value of $\boldsymbol{\lambda}$ can be obtained by Gaussian elimination of the **normal equation**

$$\mathbf{B}^\mathsf{T}\mathbf{B}\boldsymbol{\lambda} = \mathbf{B}^\mathsf{T}\mathbf{x}$$

- $\mathbf{P}_\pi^n = \mathbf{P}_\pi$
- $\pi_U(\mathbf{x})^\mathsf{T}(\pi_U(\mathbf{x}) - \mathbf{x}) = 0$ (orthogonality)

6.3 Least squares regression

Assuming that we have a prediction model in the way such that

$$\hat{y}_i = f(\boldsymbol{x}_i) = \boldsymbol{\theta}^\mathsf{T}\boldsymbol{x}_i = \boldsymbol{x}_i^\mathsf{T}\boldsymbol{\theta}$$

where $\boldsymbol{x}$ is the feature vector, y is the label and $\hat{y}$ is the prediction.
If we collect the dataset into a (N, D) data matrix $\boldsymbol{X}$, we can write down our model like this:

$$\begin{bmatrix} \boldsymbol{x}_1^\mathsf{T} \\ \vdots \\ \boldsymbol{x}_N^\mathsf{T} \end{bmatrix} \boldsymbol{\theta} = \begin{bmatrix} \hat{y}_1 \\ \vdots \\ \hat{y}_2 \end{bmatrix},$$

i.e.,

$$\boldsymbol{X}\boldsymbol{\theta} = \hat{\boldsymbol{y}}.$$

Note that the data points are the *rows* of the data matrix, i.e., every column is a dimension of the data.
Our goal is to find the best $\boldsymbol{\theta}$ such that we minimize the following objective (least square).

$$\begin{aligned} &\sum_{i=1}^{n} \|\hat{y}_i - y_i\|^2 \\ &= \sum_{i=1}^{n} \|\boldsymbol{\theta}^\mathsf{T}\boldsymbol{x}_i - y_i\|^2 \\ &= (\boldsymbol{X}\boldsymbol{\theta} - \boldsymbol{y})^\mathsf{T}(\boldsymbol{X}\boldsymbol{\theta} - \boldsymbol{y}). \end{aligned}$$

If we set the gradient of the above objective to $\mathbf{0}$, we have

$$\begin{aligned} \nabla_\theta(\boldsymbol{X}\boldsymbol{\theta} - \boldsymbol{y})^\mathsf{T}(\boldsymbol{X}\boldsymbol{\theta} - \boldsymbol{y}) &= \vec{\mathbf{0}} \\ \nabla_\theta(\boldsymbol{\theta}^\mathsf{T}\boldsymbol{X}^\mathsf{T} - \boldsymbol{y}^\mathsf{T})(\boldsymbol{X}\boldsymbol{\theta} - \boldsymbol{y}) &= \vec{\mathbf{0}} \\ \nabla_\theta(\boldsymbol{\theta}^\mathsf{T}\boldsymbol{X}^\mathsf{T}\boldsymbol{X}\boldsymbol{\theta} - \boldsymbol{y}^\mathsf{T}\boldsymbol{X}\boldsymbol{\theta} - \boldsymbol{\theta}^\mathsf{T}\boldsymbol{X}^\mathsf{T}\boldsymbol{y} + \boldsymbol{y}^\mathsf{T}\boldsymbol{y}) &= \vec{\mathbf{0}} \\ 2\boldsymbol{X}^\mathsf{T}\boldsymbol{X}\boldsymbol{\theta} - 2\boldsymbol{X}^\mathsf{T}\boldsymbol{y} &= \vec{\mathbf{0}} \\ \boldsymbol{X}^\mathsf{T}\boldsymbol{X}\boldsymbol{\theta} &= \boldsymbol{X}^\mathsf{T}\boldsymbol{y} \end{aligned}$$

The solution that gives zero gradient solves (which we call the maximum likelihood estimator) the following equation:

$$\boldsymbol{X}^\mathsf{T}\boldsymbol{X}\boldsymbol{\theta} = \boldsymbol{X}^\mathsf{T}\boldsymbol{y}.$$

This is exactly the same as the normal equation we have for projections.
This means that if we solve for $\boldsymbol{X}^\mathsf{T}\boldsymbol{X}\boldsymbol{\theta} = \boldsymbol{X}^\mathsf{T}\boldsymbol{y}$. we would find the best $\boldsymbol{\theta} = (\boldsymbol{X}^\mathsf{T}\boldsymbol{X})^{-1}\boldsymbol{X}^\mathsf{T}\boldsymbol{y}$, i.e. the $\boldsymbol{\theta}$ which minimizes our objective.

7 Principal Component Analysis

7.1 Orthogonal Complements

- If we look at an n-dimensional vector space V and a k-dimensional subspace $W \subset V$, then the orthogonal complement $W^\perp$ is an $(n-k)$-dimensional subspace of V and contains all vectors in V that are orthogonal to every vector in W
- Every vector $\mathbf{x} \in V$ can be (uniquely) decomposed into

$$\mathbf{x} = \sum_{i=1}^{k} \lambda_i \mathbf{b}_i + \sum_{j=1}^{n-k} \psi_j \mathbf{b}_j^\perp, \qquad \lambda_i, \psi_j \in \mathbb{R} \tag{3.7}$$

,where $\mathbf{b}_1, \dots, \mathbf{b}_k$ is a basis of W and $\mathbf{b}_1^\perp, \dots, \mathbf{b}_{n-k}^\perp$ is a basis of $W^\perp$

7.2 PCA Derivation

Problem Setting

For the data matrix $\mathbf{X} = [\mathbf{x}_1 | \dots | \mathbf{x}_N]^\mathsf{T}, \quad \mathbf{x}_n \in \mathbb{R}^D$
Given *orthonormal basis* $[\mathbf{b}_1 | \dots | \mathbf{b}_D]$, then

1. Each $\mathbf{x}_n$ can be represented as a linear combination of basis $\mathbf{b}_i$

$$\mathbf{x}_n = \sum_{i=1}^{D} \beta_{in}\mathbf{b}_i \tag{3.8}$$

2. The orthogonal projection of $\mathbf{x}_n$ onto the one-dimensional subspace spanned by the ith basis vector is given by:

$$\beta_{in} \overset{(3.3)}{=} \mathbf{x}_n^\mathsf{T}\mathbf{b}_i$$

3. If we choose a subset of basis $[\mathbf{b}_1 | \dots | \mathbf{b}_D]$ to form a new lower-dimensional basis

$$\mathbf{B} = [\mathbf{b}_1, \dots, \mathbf{b}_M]$$

where $M < D$,
The orthogonal projection of $\mathbf{x}$ onto the subspace spanned by the M basis vectors can be written as

$$\tilde{\mathbf{x}}_n \overset{(3.6)}{=} \mathbf{B}\underbrace{\mathbf{B}^\mathsf{T}\mathbf{x}_n}_{\text{code}}$$

where $\mathbf{B}^\mathsf{T}\mathbf{x}_n$ is the *coordinates* or *code*.

We also assume that

- The data is centered

$$\mathbf{E}[\mathbf{X}] = \vec{\mathbf{0}}$$

- The basis $\mathbf{b}_1, \dots, \mathbf{b}_D$ is *orthonormal basis* (ONB)

Equation 3.8 can be split into

$$\mathbf{x}_n \overset{(3.7)}{=} \sum_{i=1}^{M} \beta_{in}\mathbf{b}_i + \sum_{i=M+1}^{D} \beta_{in}\mathbf{b}_i \in \mathbb{R}^D \tag{3.9}$$

$$\tilde{\mathbf{x}}_n = \sum_{i=1}^{M} \beta_{in}\mathbf{b}_i \in \mathbb{R}^M \tag{3.10}$$

$\mathbf{b}_1, \dots, \mathbf{b}_M$ span the *principal subspace*. β_{in} is called the code or coordinates of $\tilde{\mathbf{x}}_n$ with respect to the basis vectors.

Average Squared Reconstruction Error

The average squared reconstruction error can be written as:

$$J = \frac{1}{N}\sum_{n=1}^{N} \|\mathbf{x}_n - \tilde{\mathbf{x}}_n\|^2 \tag{3.11}$$

Our objective is to minimize the error J. We compute the partial derivatives of J with respect to the parameters β_{in} and $\mathbf{b}_i$, then set them to zero and solve for the optimal parameters.

$$\frac{\partial J}{\partial\{\beta_{in}, \mathbf{b}_i\}} = \frac{\partial J}{\partial \tilde{\mathbf{x}}_n}\frac{\partial \tilde{\mathbf{x}}_n}{\partial\{\beta_{in}, \mathbf{b}_i\}}$$

$$\frac{\partial J}{\partial \tilde{\mathbf{x}}_n} = -\frac{2}{N}(\mathbf{x}_n - \tilde{\mathbf{x}}_n)^\mathsf{T} \tag{3.12}$$

Finding Optimal Projection Parameters

$$\begin{aligned} \frac{\partial J}{\partial \beta_{in}} &= \frac{\partial J}{\partial \tilde{\mathbf{x}}_n} \cdot \frac{\partial \tilde{\mathbf{x}}_n}{\beta_{in}} \\ &\overset{(3.12)}{=} -\frac{2}{N}(\mathbf{x}_n - \tilde{\mathbf{x}}_n)^\mathsf{T}\mathbf{b}_i \\ &\overset{(3.10)}{=} -\frac{2}{N}\left(\mathbf{x}_n - \sum_{j=1}^{M}\beta_{jn}\mathbf{b}_j\right)^\mathsf{T}\mathbf{b}_i \\ &\overset{\text{ONB}}{=} -\frac{2}{N}\left(\mathbf{x}_n^\mathsf{T}\mathbf{b}_i - \beta_{in}\underbrace{\mathbf{b}_i^\mathsf{T}\mathbf{b}_i}_{=1}\right) \\ &= -\frac{2}{N}\left(\mathbf{x}_n^\mathsf{T}\mathbf{b}_i - \beta_{in}\right) \end{aligned}$$

$$\frac{\partial J}{\partial \beta_{in}} = -\frac{2}{N}(\mathbf{x}_n^\mathsf{T}\mathbf{b}_i - \beta_{in}) = 0 \quad \Leftrightarrow \quad \beta_{in} \overset{(3.3)}{=} \mathbf{x}_n^\mathsf{T}\mathbf{b}_i \tag{3.13}$$

Reformulation of The Objective Function

$$\begin{aligned}
\tilde{\mathbf{x}}_n &\overset{(3.10)}{=} \sum_{j=1}^{M} \beta_{jn}\mathbf{b}_j \\
&\overset{(3.13)}{=} \left(\mathbf{x}_n^\mathsf{T}\mathbf{b}_j\right)\mathbf{b}_j \\
&= \sum_{j=1}^{M} \mathbf{b}_j\left(\mathbf{b}_j^\mathsf{T}\mathbf{x}_n\right) = \underbrace{\left(\sum_{j=1}^{M}\mathbf{b}_j\mathbf{b}_j^\mathsf{T}\right)}_{\text{projection matrix}}\mathbf{x}_n
\end{aligned}$$

$$\mathbf{x}_n = \left(\sum_{j=1}^{M}\mathbf{b}_j\mathbf{b}_j^\mathsf{T}\right)\mathbf{x}_n + \left(\sum_{j=M+1}^{D}\mathbf{b}_j\mathbf{b}_j^\mathsf{T}\right)\mathbf{x}_n$$

$$\begin{aligned}
\therefore \mathbf{x}_n - \tilde{\mathbf{x}}_n &= \left(\sum_{j=M+1}^{D}\mathbf{b}_j\mathbf{b}_j^\mathsf{T}\right)\mathbf{x}_n \\
&= \sum_{j=M+1}^{D}(\mathbf{b}_j^\mathsf{T}\mathbf{x}_n)\mathbf{b}_j \qquad (3.14)
\end{aligned}$$

$$\begin{aligned}
J &\overset{(3.11)}{=} \frac{1}{N}\sum_{n=1}^{N}\|\mathbf{x}_n - \tilde{\mathbf{x}}_n\|^2 \\
&\overset{(3.14)}{=} \frac{1}{N}\sum_{n=1}^{N}\left\|\sum_{j=M+1}^{D}\left(\mathbf{b}_j^\mathsf{T}\mathbf{x}_n\right)\mathbf{b}_j\right\|^2 \\
&\overset{\text{ONB}}{=} \frac{1}{N}\sum_{n=1}^{N}\sum_{j=M+1}^{D}\left(\mathbf{b}_j^\mathsf{T}\mathbf{x}_n\right)^2 \\
&= \frac{1}{N}\sum_{n}\sum_{j}\mathbf{b}_j^\mathsf{T}\mathbf{x}_n\mathbf{x}_n^\mathsf{T}\mathbf{b}_j \\
&= \sum_{j=M+1}^{D}\mathbf{b}_j^\mathsf{T}\underbrace{\left(\frac{1}{N}\sum_{n=1}^{N}\mathbf{x}_n\mathbf{x}_n^\mathsf{T}\right)}_{\mathbf{S}}\mathbf{b}_j \\
&= \sum_{J=M+1}^{D}\mathbf{b}_j^\mathsf{T}\mathbf{S}\mathbf{b}_j \qquad (3.15) \\
&= \text{trace}\left(\left(\sum_{J=M+1}^{D}\mathbf{b}_j\mathbf{b}^\mathsf{T}\right)\mathbf{S}\right)
\end{aligned}$$

where $\sum_{J=M+1}^{D}\mathbf{b}_j\mathbf{b}^\mathsf{T}$ is the projection matrix that takes our covariance matrix $\mathbf{S}$ and project it onto the orthgonal compliment of the principal subspace.
So, the *loss function* can be reformulated as the variance of the data projected onto the subspace we ignore.

Finding Optimal Basis Vectors

Solving equation 3.15 subject to a constraint $\mathbf{b}_j^\mathsf{T}\mathbf{b}_j = 1$ using the method of Lagrange multipliers.

$$\begin{aligned}
\mathbf{b}_j^\mathsf{T}\mathbf{b}_j &= 1 \qquad (3.16)\\
\mathbf{b}_j^\mathsf{T}\mathbf{b}_j - 1 &= 0
\end{aligned}$$

$$\begin{aligned}
\mathcal{L}(\mathbf{b}_j,\lambda_j) &= \mathbf{b}_j^\mathsf{T}\mathbf{S}\mathbf{b}_j - \lambda_j(\mathbf{b}_j^\mathsf{T}\mathbf{b}_j - 1) \\
&= \mathbf{b}_j^\mathsf{T}\mathbf{S}\mathbf{b}_j + \lambda_j(1 - \mathbf{b}_j^\mathsf{T}\mathbf{b}_j)
\end{aligned}$$

$$\frac{\partial\mathcal{L}}{\partial\lambda} = 1 - \mathbf{b}_j^\mathsf{T}\mathbf{b}_j = 0 \qquad \Leftrightarrow \qquad \mathbf{b}_j^\mathsf{T}\mathbf{b}_j = 1$$

$$\frac{\partial\mathcal{L}}{\partial\mathbf{b}_j} = 2\mathbf{b}_j^\mathsf{T}\mathbf{S} - 2\lambda_j\mathbf{b}_j^\mathsf{T} = 0 \qquad \Leftrightarrow \qquad \mathbf{S}\mathbf{b}_j = \lambda_j\mathbf{b}_j \qquad (3.17)$$

Equation 3.17 is an eigenvalue-eigenvector equation.
substituting equation 3.17 in 3.15

$$\begin{aligned}
J &\overset{(3.15)}{=} \mathbf{b}_j^\mathsf{T}\mathbf{S}\mathbf{b}_j \\
&\overset{(3.17)}{=} \mathbf{b}_j^\mathsf{T}\mathbf{b}_j\lambda_j \overset{\text{ONB}}{=} \lambda_j
\end{aligned}$$

The average reconstruction error is minimised if we choose the basis vectors that span the ignored subspace to be the eigenvectors of the data covariance matrix that belong to the smallest eigenvalues.

$$J = \sum_{J=M+1}^{D}\lambda_j$$

7.3 Steps of PCA

Assume that we have a dataset $\mathbf{X} = [\mathbf{x}_1|\dots|\mathbf{x}_N]^\mathsf{T} \in \mathbb{R}^{N\times D}$ with N data points and each data point is D-dimensional. We want to perform PCA on the dataset for M principal components, where $M < D$.

Data normalization

1. Compute the mean $\boldsymbol{\mu}$ and the standard deviation $\boldsymbol{\sigma}$ of the data matrix $\mathbf{X}$

$$\boldsymbol{\mu} \overset{(3.1)}{=} \frac{1}{N}\sum_{n=1}^{N}\mathbf{x}_n \qquad \boldsymbol{\sigma} \overset{(3.2)}{=} \sqrt{\frac{1}{N}\sum_{n=1}^{N}\mathbf{x}_n^2 - \boldsymbol{\mu}^2}$$

2. Then normalize the data matrix $\mathbf{X}$ to obtain $\bar{\mathbf{X}}$ by mean subtraction then dividing by the standard deviation of each data point.

$$\bar{\mathbf{x}}_n = \frac{\mathbf{x}_n - \boldsymbol{\mu}}{\boldsymbol{\sigma}}$$

PCA algorithm (If $N > D$):

1. Solve the eigenvector/eigenvalue equation
$$\mathbf{S}\mathbf{b}_i \overset{(3.17)}{=} \lambda_i\mathbf{b}_i$$
by computing the orthonormal eigenvectors $\mathbf{b}_i$ of the data covariance matrix $\mathbf{S} = \frac{1}{N}\bar{\mathbf{X}}^\mathsf{T}\bar{\mathbf{X}} \in \mathbb{R}^{D\times D}$
2. Choose the eigenvectors $\mathbf{b}_i$ associated with the M largest eigenvalues to be the basis of the principal subspace.
3. Collect these eigenvectors in a matrix $\mathbf{B} = [\mathbf{b}_1|\dots|\mathbf{b}_M]$
4. Compute the orthogonal projection of the data onto the principal axes
$$\tilde{\mathbf{X}} \overset{(3.4)}{=} \mathbf{P}_\pi\mathbf{X}$$
where $\mathbf{P}_\pi \overset{(3.5)}{=} \mathbf{B}\mathbf{B}^\mathsf{T}$ is the projection matrix.

PCA for High-Dimensional Datasets (If $N < D$):

computing the covariance matrix for high-dimensional data is very computationally expensive. So, the computations can be simplified by the following:

1. Solve the eigenvector/eigenvalue equation
$$\frac{1}{N}\bar{\mathbf{X}}\bar{\mathbf{X}}^\mathsf{T}\mathbf{c}_i = \lambda_i\mathbf{c}_i \qquad (3.18)$$
where $\mathbf{c}_i = \bar{\mathbf{X}}\mathbf{b}_i$, by computing the eigenvectors $\mathbf{c}_i$ of the matrix $\frac{1}{N}\bar{\mathbf{X}}\bar{\mathbf{X}}^\mathsf{T} \in \mathbb{R}^{N\times N}$.
2. Choose the eigenvectors $\mathbf{c}_i$ associated with the M largest eigenvalues.
3. Recover the original orthonormal eigenvectors $\mathbf{b}_i$ of the data covariance matrix $\mathbf{S} = \frac{1}{N}\bar{\mathbf{X}}^\mathsf{T}\bar{\mathbf{X}}$ by left-multiplying the eigenvector equation (3.18) by $\bar{\mathbf{X}}^\mathsf{T}$, which yields
$$\underbrace{\frac{1}{N}\bar{\mathbf{X}}^\mathsf{T}\bar{\mathbf{X}}}_{\mathbf{S}}\bar{\mathbf{X}}^\mathsf{T}\mathbf{c}_i = \lambda_i\bar{\mathbf{X}}^\mathsf{T}\mathbf{c}_i$$
and we recover $\bar{\mathbf{X}}^\mathsf{T}\mathbf{c}_i$ as an eigenvector of $\mathbf{S}$ with (the same) eigenvalue λ_i
4. To perform PCA make sure to normalize the recovered eigenvectors so that $\|\bar{\mathbf{X}}^\mathsf{T}\mathbf{c}_i\| = 1$.
$$\mathbf{b}_i = \frac{\bar{\mathbf{X}}^\mathsf{T}\mathbf{c}_i}{\|\bar{\mathbf{X}}^\mathsf{T}\mathbf{c}_i\|}$$
5. Collect these eigenvectors in a matrix $\mathbf{B} = [\mathbf{b}_1|\dots|\mathbf{b}_M]$
6. Compute the orthogonal projection of the data ontho the principal axis
$$\tilde{\mathbf{X}} \overset{(3.4)}{=} \mathbf{P}_\pi\mathbf{X}$$
where $\mathbf{P}_\pi \overset{(3.5)}{=} \mathbf{B}\mathbf{B}^\mathsf{T}$ is the projection matrix.

https://github.com/FadyMorris/formula-sheets
DOI: 10.5281/zenodo.3986728