

Probability Theory and Linear Algebra

Benjamin

March 26, 2024

Lecture 1 - Introduction, vectors, linearity

Vectors

Vector is an ordered, finite list of numbers. Vectors are usually written as vertical arrays, surrounded by square or curved brackets:

They can also be written as numbers separated by commas and surrounded by parentheses. In this notation style, the vector above is written as:

$$(-1.1, 0.0, 3.6, -7.2) \tag{1}$$

The elements (or entries, coefficients, components) of a vector are the values in the array. The size (also called dimension or length) of the vector is the number of elements it contains. The vector above, for example, has size four; its third entry is 3.6. A vector of size n is called an n -vector. A 1-vector is considered to be the same as a number, i.e., we do not distinguish between the 1-vector [1.3] and the number 1.3.

We often use symbols to denote vectors. If we denote an n -vector using the symbol a , the i th element of the vector a is denoted a_i , where the subscript i is an integer index that runs from 1 to n , the size of the vector. Two vectors a and b are *equal*, which we denote $a = b$, if they have the same size, and each of the corresponding entries is the same. If a and b are n -vectors, then $a = b$ means $a_1 = b_1, \dots, a_n = b_n$.

The numbers or values of the elements in a vector are called scalars. We will focus on the case that arises in most applications, where the scalars are real numbers. In this case we refer to vectors as real vectors. (Occasionally other types of scalars arise, for example, complex numbers, in which case we refer to the vector as a complex vector.) The set of all real numbers is written as \mathbf{R} , and the set of all real n -vectors is denoted \mathbf{R}^n , so $a \in \mathbf{R}^n$ is another way to say that a is an n -vector with real entries. Here we use set notation: $a \in \mathbf{R}^n$ means that a is an element of the set \mathbf{R}^n ; see appendix A.

Block or stacked vectors. It is sometimes useful to define vectors by *concatenating* or *stacking* two or more vectors, as in

where a , b , c , and d are vectors. If b is an m -vector, c is an n -vector, and d is a p -vector, this defines the $(m + n + p)$ -vector. The numbers or values of the elements in a vector are called scalars. We will focus on the case that arises in most applications, where the scalars are real numbers. In this case we refer to vectors as real vectors. (Occasionally other types of scalars arise, for example, complex numbers, in which case we refer to the vector as a complex vector.) The set of all real numbers is written as \mathbf{R} , and the set of all real n -vectors is denoted \mathbf{R}^n , so $a \in \mathbf{R}^n$ is another way to say that a is an n -vector with real entries. Here we use set notation: $a \in \mathbf{R}^n$ means that a is an element of the set \mathbf{R}^n ; see appendix A.

$$a = (b_1, b_2, \dots, b_m, c_1, c_2, \dots, c_n, d_1, d_2, \dots, d_p) \quad (2)$$

The stacked vector a is also written as $a = (b \, c \, d)$. Stacked vectors can include scalars (numbers). For example if a is a 3-vector, $(1, a)$ is the 4-vector $(1, a_1, a_2, a_3)$.

Subvectors. In the equation above, we say that b , c and d are *subvectors* or *slices* of a , with sizes m , n and p , respectively. *Colon notation* is used to denote subvectors. If a is a vector, then the subvector $a_r : s$ is a vector of size $s - r + 1$, with entries a_r, \dots, a_s :

$$a_r : s = (a_r, \dots, a_s) \quad (3)$$

As a more concrete example, if z is the 4-vector $(1, -1, 2, 0)$, the slice $z_2 : 3$ is $z_2 : 3 = (-1, 2)$. Colon notation is not completely standard, but is gaining traction.

In programming, arrays of length n are indexed from $i = 0$ to $i = n - 1$. In standard mathematical notation, n -vectors are indexed from $i = 1$ to $i = n$, so in this subject/class, vectors will be indexed from $i = 1$ to $i = n$.

Zero vectors. A *zero vector* is a vector with all elements equal to zero. Sometimes the zero vector of size n is written as 0_n , where the subscript denotes the size. But usually a zero vector is denoted just 0 , the same symbol used to denote the number 0. In this case you have to figure out the size of the zero vector from the context. As a simple example, if a is a 9-vector, and we are told that $a = 0$, the 0 vector on the right-hand side must be the one of size 9. Even though zero vectors of different sizes are different vectors, we use the same symbol 0 to denote them. In computer programming this is called *overloading*: The symbol 0 is overloaded because it can mean different things depending on the context (e.g., the equation it appears in).

Unit vectors. A (standard) *unit vector* is a vector with all elements equal to zero, except one element which is equal to one. The i th unit vector (of size n) is the unit vector with i th element one, and denoted e_i . For example, the vectors

$$e_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, e_2 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, e_3 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \quad (4)$$

are the three unit vectors of size 3. The notation for unit vectors is an example of the ambiguity in notation noted above. Here, e_i denotes the i th unit vector, and not the i th element of a vector e . Thus we can describe the i th unit n -vector e_i as

![[i]th Unit Vector description](/imgs/SLIAL/ithUnitVectorDesc.png)

for $i, j = 1, \dots, n$. On the left-hand side e_i is an n -vector; $(e_i)_j$ is a number, its j th entry. As with zero vectors, the size of e_i is usually determined from the context.

Ones vector. We use the notation $\mathbf{1}_n$ for the n -vector with all its elements equal to one. We also write $\mathbf{1}$ if the size of the vector can be determined from the context. (Some use e to denote a vector of all ones, but we will not use this notation.) The vector $\mathbf{1}$ is sometimes called the *ones vector*.

Sparsity. A vector is said to be *sparse* if many of its entries are zero; its *sparsity pattern* is the set of indices of nonzero entries. The number of the nonzero entries of an n -vector x is denoted $\mathbf{nnz}(x)$. Unit vectors are sparse, since they have only one nonzero entry. The zero vector is the sparsest possible vector, since it has no nonzero entries. Sparse vectors arise in many applications.

title

Literature: [VMLS], Chapter 1, Section 2.1 + slides Exercises:

Basic vector notation and operations: 1.6, 1.7, 1.10, 1.11, 1.13 Block-vectors: 1.4 Linear combinations: 1.18 Linear functions: 2.1, 2.4 Affine functions: 2.2, 2.7 Regression models: 2.10, 2.12 Polynomials: 2.8

Lecture 1.5 - norm, distance, angle

When: unscheduled! Suggestion: 09.02.2004, afternoon. Literature: [VMLS], Chapter 3

Exercises: Properties of the norm: 3.3, 3.4, 3.18, 3.19 Nearest neighbour: 3.13 Angle: 3.22, 3.23 Chebyshev inequality: 3.7, 3.8 Block-vectors: 3.2

Lecture 2 - Linear independence

Literature: [VMLS], Chapter 5 + Slides

Exercises:

Linear (in)dependence: 5.2, 5.1 Pythagoras theorem: 5.4 Orthogonalization and Gram-Schmidt: 5.5-5.9 (facit -i moodle)

Lecture 3 - Matrices

A matrix is a rectangular array of numbers written between rectangular brackets, as in

$$\begin{bmatrix} 0 & 1 & -2.3 & 0.1 \\ 1.3 & 4 & -0.1 & 0 \\ 4.1 & -1 & 0 & 1.7 \end{bmatrix} \quad (5)$$

It is also common to use large parentheses instead of rectangular brackets, as in

$$\begin{pmatrix} 0 & 1 & -2.3 & 0.1 \\ 1.3 & 4 & -0.1 & 0 \\ 4.1 & -1 & 0 & 1.7 \end{pmatrix} \quad (6)$$

An important attribute of a matrix is its *size* or *dimensions* i.e, the numbers of rows and columns. The matrix above has 3 rows and 4 columns, so its size is 3 x 4.

A matrix of size $m \times n$ is called an $m \times n$ matrix.

The *elements* (or *entries* or *coefficients*) of a matrix are the values in the array.

The i, j element is the value of the i th row and j th column, denoted by double subscripts: the i, j element of a matrix A is denoted A_{ij} (or A_i, j , when i or j is more than one digit or character).

The positive integers i and j are called *indices*.

If A is an $m \times n$ matrix, then the row index runs from 1 to m and the column index j runs from 1 to n . Row indices go from top to bottom, so row 1 is the top row, and row m is the bottom row. Column indices go from left to right, so column 1 is the left column, and column n is the right column.

If the matrix above is B , then we have $B_{13} = -2.3$, $B_{32} = -1$. the row index of the bottom left element (which has value 4.1) is 3; its column index is 1.

Two matrices are equal, if they have the same size, and if all the corresponding entries are equal.

As with vectors, we normally deal with matrices with entries that are real numbers, which will be our assumption unless stated otherwise. The set of real $m \times n$ matrices is denoted $\mathbf{R}^{m \times n}$. But matrices with complex entries, for example, do arise in some applications

Matrix indexing. As with vectors, we in math index from 1. In computer languages, a matrix is typically stored as a 2D array, where we would index the matrix from 0 (unless you use a package that index from 1).

Square, tall and wide matrices. A *square* matrix has an equal number of rows and columns. A square matrix of size $n \times n$ is said to be of *order* n . A *tall* matrix has more rows than columns (size $m \times n$ with $m > n$). A *wide* matrix has more columns than rows (size $m \times n$ with $n > m$).

Column and row vectors. An n -vectors can be interpreted as an $n \times 1$ matrix; we do not distinguish between vectors and matrices with one column. A matrix with only one row, i.e, with size $1 \times n$, is called a *row vector*; to give its size, we can refer to it as an *n-row-vector*. As an example

$$\begin{bmatrix} -2.1 & -30 \end{bmatrix}$$

is a 3-row-vector (or 1×3 matrix). To distinguish them from row vectors, vectors are sometimes called column vectors. A 11 matrix is considered to be the same as a scalar.

Notational conventions. Many people (including us) tend to use capital letters to denote matrices, and lower case letters for (column or row) vectors. But this convention is not standardized, so you should be prepared to figure out whether a symbol represents a matrix, column vector, row vector, or a scalar, from context.

Columns and rows of a matrix. An $m \times n$ A has n columns, given by (the m -vectors)

$$\begin{bmatrix} A_{1j} \\ \vdots \\ A_{mj} \end{bmatrix} \quad (7)$$

for $j = 1, \dots, n$. The same matrix has m rows, given by the (n -row-vectors)

$$A_{i1} \dots A_{in}$$

, for $i = 1, \dots, m$.

As a specific example, the 2×3 matrix

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix} \quad (8)$$

has first row

$$123$$

(which is a row-vector or a 1×3 matrix), and second column

$$\begin{bmatrix} 2 \\ 5 \end{bmatrix} \quad (9)$$

(which is a 2-vector or 2×1 matrix), also written compactly as $(2, 5)$.

Block matrices and submatrices. It is useful to consider matrices whose entries are themselves matrices, as in

$$A = \begin{bmatrix} B & C \\ D & E \end{bmatrix}, \quad (10)$$

where B , C , D and E are matrices. Such matrices are called *block matrices*; the elements B , C , D , and E are called *blocks* or *submatrices* of A . The submatrices can be referred to by their block row and column indices; for example, C is the 1,2 block of A . Block matrices must have the right dimensions to fit together. Matrices in the same (block) row must have the same number of rows (*i.e.*, the same ‘height’); matrices in the same (block) column must have the same number of columns (*i.e.*, the same ‘width’). In the example above, B and C must have the same number of rows, and C and E must have the same number of columns. Matrix blocks placed next to each other in the same row are said to be *concatenated*; matrix blocks placed above each other are called *stacked*. As an example, consider

$$B = [023], C = [-1], D = \begin{bmatrix} 2 & 3 & 1 \\ 1 & 3 & 5 \end{bmatrix}, E = \begin{bmatrix} 4 \\ 4 \end{bmatrix}, \quad (11)$$

Then the block matrix A above is given by

$$A = \begin{bmatrix} 0 & 2 & 3 & -1 \\ 2 & 2 & 1 & 4 \\ 1 & 3 & 5 & 4 \end{bmatrix} \quad (12)$$

We can also divide a larger matrix (or vector) into ‘blocks’. In this context the blocks are called *submatrices* of the big matrix. As with vectors, we can use colon notation to denote submatrices. If A is an $m \times n$ matrix, and p, q, r, s are integers with $1 \leq p \leq q \leq m$ and $1 \leq r \leq s \leq n$ then $A_p : q, r : s$ denotes the submatrix

$$A_p : q, r : s = \begin{bmatrix} A_{pr} & A_{p, r+1} & \dots & A_{ps} \\ A_{p+1, r} & A_{p+1, r+1} & \dots & A_{p+1, s} \\ \vdots & \vdots & \ddots & \vdots \\ A_{qr} & A_{q, r+1} & \dots & A_{qs} \end{bmatrix}. \quad (13)$$

This submatrix has size $(q - p + 1) \times (s - r + 1)$ and is obtained by extracting from A the elements in rows p through q and columns r through s .

For the specific matrix A in , we have

$$A_2 : 3, 3 : 4 = \begin{bmatrix} 1 & 4 \\ 5 & 4 \end{bmatrix}. \quad (14)$$

Column and row representation of a matrix. Using block matrix notation we can write an $m \times n$ matrix A as a block matrix with one block row and n block columns,

$$A = [a_1 a_2 \dots a_n]. \quad (15)$$

where a_j , which is an m -vector, is the j th column of A . Thus, an $m \times n$ matrix can be viewed as its n columns, concatenated.

Similarly, an $m \times n$ matrix A can be written as a block matrix with one block column and m block rows:

$$A = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}, \quad (16)$$

where b_i which is a row n -vector, is the i th row of A . In this notation, the matrix A is interpreted as its m rows, stacked. Literature: [VMLS], Chapters 6,7 + Slides

Exercises: Dimensions: 6.1, 6.3 Matrix-vector multiplication: 6.6, 6.10, 6.11 Graph adjacency/incidence matrices: 6.4, 6.5, 7.7 Linear (in)dependence: 6.17, 7.6 Skew-symmetric matrices: 6.12 Various transformations: 6.13, 7.2, 7.3, 7.4, 7.8, 7.11 Complexity of matrix-vector product: 6.20 Norm of matrix-vector product: 6.14

Lecture 4 - Linear equations. Gaussian elimination

This lecture is about vector-valued linear and affine functions, and systems of linear equations

Linear and affine functions

Vector-valued functions of vectors. The notation $f : \mathbb{R}^n \rightarrow \mathbb{R}^m$ means that f is a function that maps real n -vectors to real m -vectors. The value of the function f , evaluated at an n -vector x , is an m -vector $f(x) = (f_1(x), f_2(x), \dots, f_m(x))$. Each of the components f_i of f is itself a scalar-valued function of x . As with scalar-valued functions, we sometimes write $f_i(x) = f_i(x_1, x_2, \dots, x_n)$ to emphasize that f is a function of n scalar arguments. We use the same notation for each of the components of f , writing $f_i(x) = f_i(x_1, x_2, \dots, x_n)$ to emphasize that f_i is a function mapping the scalar arguments x_1, \dots, x_n into a scalar.

The matrix-vector product function. Suppose A is an $m \times n$ matrix. We can define a function $f : \mathbb{R}^n \rightarrow \mathbb{R}^m$ by $f(x) = Ax$. The inner product

function $f : \mathbf{R}^n \rightarrow \mathbf{R}$, defined as $f(x) = a_x^T$, discussed in Lecture 1, is the special case with $m = 1$.

Superposition and linearity. The function $f : \mathbf{R}^n \rightarrow \mathbf{R}^m$, defined by $f(x) = Ax$, is *linear*, i.e., it satisfies the superposition property:

!Superposition Property[/imgs/SLIAL/SuperpositionProperty.png]

holds for all n -vectors x and y and all scalars α and β . It is a good exercise to parse this simple looking equation, since it involves overloading of notation. On the left-hand side, the scalar-vector multiplications αx and βy involve n -vectors, and the sum $\alpha x + \beta y$ is the sum of two n -vectors. The function f maps n -vectors to m -vectors, so $f(\alpha x + \beta y)$ is an m -vector. On the right-hand side, the scalar-vector multiplications and the sum are those for m -vectors. Finally, the equality sign is equality between two m -vectors.

We can verify that superposition holds for f using properties of matrix-vector and scalar-vector multiplication:

!Verify that superposition holds for f[/imgs/SLIAL/SuperpositionHolds.png]

Thus we can associate with every matrix A a linear function $f(x) = Ax$.

The converse is also true. Suppose f is a function that maps n -vectors to m -vectors, and is linear, i.e., (8.1) holds for all n -vectors x and y and all scalars α and β . Then there exists an $m \times n$ matrix A such that $f(x) = Ax$ for all x . This can be shown in the same way as for scalar-valued functions in 2.1, by showing that if f is linear, then

!8.2[/imgs/SLIAL/8.2.png]

where e_k is the k th unit vector of size n . The right-hand side can also be written as a matrix-vector product Ax , with

!8.2.1[/imgs/SLIAL/8.2.1.png]

Examples of linear functions.

!Examples of linear functions[/imgs/SLIAL/ExamplesLinearFunctions.png]

!Examples of linear functions 2[/imgs/SLIAL/ExamplesLinearFunctions2.png]

Examples of functions that are not linear here we list some examples of functions f that map n -vectors x to n -vectors $f(x)$ that are not linear. In each case we show a superposition counterexample.

!Examples of nonlinear functions[/imgs/SLIAL/ExamplesNonlinearFunctions.png]

Affine functions. A vector-valued function $f : \mathbf{R}^n \rightarrow \mathbf{R}^m$ is called *affine* if it can be expressed as $f(x) = Ax + b$, where A is an $m \times n$ matrix and b is an m -vector. It can be shown that a function $f : \mathbf{R}^n \rightarrow \mathbf{R}^m$ is affine if and only if

$$f(\alpha x + \beta y) = \alpha f(x) + \beta f(y) \quad (17)$$

holds for all n -vectors x, y , and all scalars

α, β ,

that satisfy

$\alpha + \beta = 1$.

In other words, superposition holds for affine combinations of vectors. (For linear functions, superposition holds for any linear combinations

of vectors.) The matrix A and the vector b in the representation of an affine function as $f(x) = Ax + b$ are unique. These parameters can be obtained by evaluating f at the vectors $0, e_1, \dots, e_n$, where e_k is the k th unit vector in R^n . We have

$$A = [f(e_1) - f(0), f(e_2) - f(0), \dots, f(e_n) - f(0)], b = f(0). \quad (18)$$

Just like affine scalar-valued functions, affine vector-valued functions are often called linear, even though they are linear only when the vector b is zero.

Linear function models

Linear function models

Many functions or relations between variables that arise in natural science, engineering, and social sciences can be approximated as linear or affine functions. In these cases we refer to the linear function relating the two sets of variables as a model or an approximation, to remind us that the relation is only an approximation, and not exact.

Taylor approximation Suppose $f : \mathbf{R}^n \rightarrow \mathbf{R}^m$ is differentiable, i.e., has partial derivatives, and z is an n -vector. The first-order Taylor approximation of f near z is given by

!["First-order Taylor approximation"](/imgs/SLIAL/FirstOrderTaylorApproximation.png)

for $i = 1, \dots, m$. (This is just the first-order Taylor approximation of each of the scalar-valued functions f_i , described in Lecture One.) For x near z $\hat{f}(x)$ is a very good approximation of $f(x)$. We can express this approximation in compact notation, using matrix-vector multiplication, as

!["Taylor approximation using compact notation"](/imgs/SLIAL/TaylorApproximationCompactNotation.png)

where the mn matrix $Df(z)$ is the derivative or *Jacobian* matrix of f at z (see §C.1). Its components are the partial derivatives of f ,

!["Derivative or Jacobian matrix"](/imgs/SLIAL/JacobianMatrix.png)

As in the scalar-valued case, Taylor approximation is sometimes written with a second argument as $\hat{f}(x; z)$ to show the point z around which the approximation is made. Evidently the Taylor series approximation \hat{f} is an affine function of x . (It is often called a linear approximation of f , even though it is not, in general, a linear function.)

Regression model

Taking the regression model:

!["The regression model"](/imgs/SLIAL/RegressionModel.png)

where the n -vector x is a feature vector for some object, ta is an n -vector of weights, v is a constant (the offset), and \hat{y} is the (scalar) value of the regression model prediction. Now suppose we have a set of N objects

(also called *samples* or *examples*), with feature vectors $x^{(1)}, \dots, x^{(N)}$. The regression model predictions associated with the examples are given by

!The regression model predictions associated with given examples](/imgs/SLIAL/RegressionModel

These numbers usually correspond to predictions of the value of the outputs or responses. If in addition to the example feature vectors $x^{(i)}$ we are also given the actual value of the associated response variables, $y^{(1)}, \dots, y^{(N)}$, then our *prediction errors* or *residuals* are

!Regression model prediction errors (residuals)](/imgs/SLIAL/RegressionModelResiduals.png)

Literature: [VMLS], Chapter 8 + slides. You may find a much more detailed explanation of row reduction with examples in Sections 1.1-1.2 of [Lay]=”Linear Algebra and Its Applications” by Lay, Lay, McDonald (attached). Note: I only discuss the (non-unique) echelon form in my slides, after which we run a backward substitution algorithm, whereas Lay also talks about the reduced echelon form. Basically we do not explicitly perform ”Step 5” of the reduction algorithm on p. 17 in Lay’s book.

Exercises:

[VMLS]: 8.1, 8.3, 8.4, 8.6, 8.8, 8.9, 8.14, 8.16 [Lay]: 1.1.1, 1.1.6. Run backward substitution: 1.1.7, 1.1.9. Determine consistency: 1.1.15, 1.1.17. Find all solutions to the system: 1.2.7, 1.2.9, 1.2.11, 1.2.13. Existence/uniqueness: 1.2.15, 1.2.19

Lecture 4.5 - matrix multiplication

Literature: [VMLS], Chapter 10

Exercises:

Matrix multiplication: 10.1, 10.2, 10.3, 10.4, 10.6, 10.8, 10.12, 10.14, 10.23, 10.25, 10.26, 10.33, 10.34 Orthogonal matrices: 10.35, 10.38, 10.40 (there is an obvious typo in the definition of Q_i) Computational complexity: 10.42, 10.43, 10.44

Lecture 5 - Workshop 01

When: 27.02.2024, 12:30

Lecture 6 - Matrix inverses

Left and right inverses

Recall that for a number a , its (multiplicative) inverse is the number x for which $xa = 1$, which we usually denote as $x = 1/a$ or (less frequently) $x = a^{-1}$. The inverse x exists provided a is nonzero. For matrices the concept of inverse is more complicated than for scalars; in the general case, we need to distinguish between left and right inverses. We start with the left inverse.

Left inverse. A matrix X that satisfies $XA = I$ is called a *left inverse* of A . The matrix A is said to be *left-invertible* if a left inverse exists. Note that if A has size mn , a left inverse X will have size nm , the same dimensions as A^T .

Literature: [VMLS], Chapter 11 + Slides

Exercises:

11.1, 11.2, 11.3, 11.4, 11.7, 11.12, 11.16, 11.17, 11.18, 11.22, 11.24 Block-matrices: 11.5 (hint: show that the inverse matrix is also symmetric; assume it has the same block-structure as A), 11.6

Lecture 7 - Least-squares problems

When: 08.03.2024, 12:30. This is a self-study with help. Literature: [VMLS], Chapter 12 + slides

Exercises:

12.1, 12.2, 12.4, 12.9, 12.14, 12.16, 13.3

Lecture 8 - Workshop 02

When: 12.03.2024, 12:30

Lecture 9 - Introduction, optimality conditions

When: 18.03.2024, 10:15. Lecture followed up by exercises. Literature: [AEP], from chapter 7 up to and including section 8.2.2 (pp 197-214) + section 10.2 (up and including Theorem 10.6 w/o proof, pp 243-248).

Exercises (from chapter 9 *exercises.pdf*, see below). I recommend solving the practice problem, and at least one of the other problems.

Solve the practice problems 1-4 on p.532 Solve graphically: exercises 9.2.7-9.2.10. I recommend trying to solve both the minimization and the maximization problem for each exercise. Rewrite to standard, not canonical form: exercises 9.2.3-9.2.6. Randomize each problem by: randomly selecting minimize/maximize; then for each constraint randomly select from $=$, \leq , and \geq ; finally for each of the variables randomly select, whether it should be 0, ≥ 0 , or "free"/no sign restricted. Duality: for each problem you stated in the standard form in the previous category, formulate the dual problem. Modelling: exercises 9.2.1-9.2.2, 9.2.15.

Lecture 10 - Duality in linear optimization

When: 22.03.2024, 12:30. This is a self-study with help. Literature: [AEP], from chapter 7 up to and including section 8.2.2 (pp 197-214) + section 10.2 (up and including Theorem 10.6 w/o proof, pp 243-248).

Exercises (from chapter9 *exercises.pdf*, see below). I recommend solving the practice problem, and at least

Solve the practice problems 1-4 on p.532 Solve graphically: exercises 9.2.7-9.2.10. I recommend trying to solve both the minimization and the maximization problem for each exercise. Rewrite to standard, not canonical form: exercises 9.2.3-9.2.6. Randomize each problem by: randomly selecting minimize/maximize; then for each constraint randomly select from $=$, \leq , and \geq ; finally for each of the variables randomly select, whether it should be 0, ≥ 0 , or "free"/no sign restricted. Duality: for each problem you stated in the standard form in the previous category, formulate the dual problem. Modelling: exercises 9.2.1-9.2.2, 9.2.15.

Lecture 11 - Simplex algorithm

Basic feasible solutions and the Representation Theorem

In this section we introduce the concept of basic feasible solution and show the equivalence between extreme point and basic feasible solution. From this we can draw the conclusion that if there exists an optimal solution then there exists an optimal solution among the basic feasible solutions. This fact is crucial in the simplex method which searches for an optimal solution among the basic feasible solutions.

Consider a linear program in standard form,

$$\begin{aligned} & \text{minimize } z = c^T x, \\ & \text{subject to } Ax = b, \\ & x \geq 0^n \end{aligned} \tag{19}$$

where $A \in \mathbb{R}^m \times n$ with $\text{rank } A = \text{rank}(A, b) = m$ (otherwise, we can always delete rows), $n > m$, and $b \geq 0^m$. A point \hat{x} is a *basic solution* of 8.6 if

- the equality constraints are satisfied at \hat{x} , that is, $A\hat{x} = b$; and
- the columns of A corresponding to the non-zero components of \hat{x} are linearly Independent.

Literature: [AEP]: Section 8.2.2 up to Theorem 8.7 (you do not have to read the proof)+Chapter 9.

Exercises:

Go through the full solutions of example1 (only Phase II) and example 2 (Phase I + II) on moodle [AEP]: Exercises 9.1, 9.2(a) Exercises from [chapter9 *exercises*], section 9.3 : 9–14, 1–2. *Ignore everything about the "simplex tableau", just use sim* 8.6

Lecture 12 - Workshop 03

When: 04.04.2024, 12:30.

Lecture 13 - Introduction to probability theory

When: 09.04.2024, 12:30. Lecture followed up by exercises. Literature: [Rosen] Sections 7.1-7.2

Exercises:

Compute the probabilities [7.1]: 3, 7, 9, 11 Compare the probabilities [7.1]: 37, 45 Probability distributions [7.2]: 1, 3, 5

Lecture 13.5 - mean value, variance

When: unscheduled! Suggestion: 12.04.2024, afternoon Literature: [Rosen] Section 7.4 + slides

Exercises:

Compute the expected value: [7.4] 3, 5, 7, 11 Independence: [7.4] 19, 21 Variance: [7.4] 27, 29

Lecture 14 - Conditional probability. Bayes' theorem

When: 16.04.2024, 12:30. This is a self-study with help. Literature: [Rosen], Sections 7.2-7.3 + slides

Exercises:

Compute the probabilities: [7.2] 7, 10, 29, 30 Conditional probabilities: [7.2] 23, 24, 25 Independent events: [7.2] 26, 27 Bernoulli trials: [7.2] 34, 35 Bayes' thm: [7.3]: 1, 3, 5, 9, 13, 15, 23

Lecture 14.5 - eigenvalues and eigenvectors

When: unscheduled! Suggestion: 19.04.2024, afternoon Literature: Section 5.1 of [Lay]="Linear Algebra and Its Applications" by Lay, Lay, McDonald (attached) + slides. Lay gives the definitions and explains the way of computing eigenvectors given the knowledge of an eigenvalue. QR (and other) algorithms are used in practice to compute eigenvalues.

Exercises:

[Lay]: 5.1.2, 5.1.5, 5.1.7, 5.1.17, 5.1.19, 5.1.21

Lecture 15 - Linear dynamical systems, Markov chains

When: 23.04.2024, 12:30. This is a self-study with help. Literature: [Lay], Section 4.9 + slides. Skim through [VMLS], Chapter 9 for some examples/applications


Exercises:

[Lay 4.9]: 1, 11, 7, 9, 10, 19, 20. Use QR algorithm to approximate the matrix eigenvalues in the exercise 7. [VMLS]: 9.4, 9.3

Lecture 16 - Workshop 04

When: 30.04.2024, 12:30.

Other notes:

Reduction Remember, in order to isolate a variable:  (/imgs/SLIAL/IsolatingAVaria