Linear Algebra and Introduction to MATLAB

Sabine Eschenhof

Contents

1 Introduction and Motivation

- why to study linear systems?
  - linear equations are the most elementary equations that can arise
  - we can (mostly) calculate explicit solutions
  - when studying non-linear models which cannot be solved explicitly, linear systems can serve as an approximation (calculus, Taylor polynomial)
  - some of the most frequently studied economic models are linear
- most compact way to present linear systems is with matrices
- easiest example is a scalar or a vector
- how to implement with a computer? $\Rightarrow$ MATLAB

2 Vectors in n-dimensional Spaces

- we will use $\mathbb{R}$, the set of real numbers (e.g. $1,\ -2.5,\ \pi$)

Definition 1

A *scalar* is a number $c \in \mathbb{R}$.
A *vector* x is an array of scalars. A *row vector* has one row and n columns, a *column vector* contains of n rows and one column. The number n is called the *dimension* of the vector. In general, if no explicit description is given, a "vector" denotes a column vector.
$\mathbf{0}$ denotes a vector where all entries are 0.

Examples:

- scalar: $c = 1,\ c = -0.25,\ c = \pi$
- row vector: $x' = (a_1,\ a_2,\ \ldots,\ a_n)$ with $a_i \in \mathbb{R}\ \forall i$
- column vector: $x = \begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{pmatrix}$ with $a_i \in \mathbb{R}\ \forall i$

2.1 Elementary Operations

- for scalars $a_i,\ b_i\ \forall i$ and c and vectors $x,\ y$:

$$x := \begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{pmatrix} \qquad \mathbf{0} := \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

$$x + y = \begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{pmatrix} + \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix} = \begin{pmatrix} a_1 + b_1 \\ a_2 + b_2 \\ \vdots \\ a_n + b_n \end{pmatrix} \qquad cx = c \begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{pmatrix} = \begin{pmatrix} ca_1 \\ ca_2 \\ \vdots \\ ca_n \end{pmatrix}$$

- Let $c,\ d$ be scalars and $x,\ y$ and z vectors with the same number of rows. Then

$$\begin{aligned} x + (y + z) &= (x + y) + z \\ x + \mathbf{0} &= x = \mathbf{0} + x \\ x + (-x) &= \mathbf{0} \\ x + y &= y + x \\ c(dx) &= (cd)x \\ 1x &= x \\ c(x + y) &= cx + cy \\ (c + d)x &= cx + dx \\ 0x &= \mathbf{0} \ \wedge \ c\mathbf{0} = \mathbf{0} \\ x + y &= x + z \ \Leftrightarrow \ y = z \\ cx &= \mathbf{0} \ \Leftrightarrow \ c = 0 \ \vee \ x = \mathbf{0} \end{aligned}$$

2.2 Linear Independence, Span and Basis

Definition 2

Two vectors $x,\ y$ in $\mathbb{R}^n$ are called *linearly independent* iff there exists no scalar c such that $x = cy$ or $y = cx$.

For n vectors $x_i,\ i = 1,\ \ldots,\ n$, every vector

$$c_1x_1 + \ldots + c_nx_n = \sum_{i=1}^{n} c_ix_i$$

(c_i scalars) is called a *linear combination* of $x_1,\ \ldots,\ x_n$.
n vectors x_i are *linearly independent* iff there exist no scalars $c_1,\ \ldots,\ c_n$ such that one of the n vectors $x_i,\ i = 1,\ \ldots,\ n$ is a linear combination of the other $n-1$ vectors, i.e., if

$$c_1x_1 + \ldots + c_nx_n = 0 \ \Rightarrow\ c_1 = \ldots = c_n = 0.$$

Note: No set of more than n vectors in $\mathbb{R}^n$ can be linearly independent.

The set of all linear combinations of the vectors $x_i,\ i = 1,\ \ldots,\ m$, i.e., $\{c_1x_1 + \ldots + c_mx_m,\ c_1, \ldots, c_m \in \mathbb{R}\}$ is called the *space spanned* by $x_1,\ \ldots,\ x_m$.

Note: A set of vectors spanning $\mathbb{R}^n$ must contain of at least n vectors.

Let $x_1,\ \ldots,\ x_m$ be m vectors in $\mathbb{R}^n$ spanning the set Ω and being linearly independent, then $x_1,\ \ldots,\ x_m$ form a *basis* of Ω. The number m of vectors forming the basis is called the *dimension* of Ω.

Note: $x_1,\ \ldots,\ x_m$ basis of $\Omega\ \Rightarrow\ x_1,\ \ldots,\ x_m$ spanning Ω, but $x_1,\ \ldots,\ x_m$ spanning $\Omega\ \not\Rightarrow\ x_1,\ \ldots,\ x_m$ basis of Ω in general (spanning can contain linearly dependent vectors!).

Theorem 1

The following statements are equivalent

1. The vectors $x_1,\ \ldots,\ x_n$ are linearly independent.
2. $c_1x_1 + c_2x_2 + \ldots + c_nx_n = 0 \Leftrightarrow c_1 = c_2 = \ldots = c_n = 0$ for $c_i \in \mathbb{R}\ \forall i$.
3. $c_1x_1 + \ldots + c_nx_n = d_1x_1 + \ldots + d_nx_n \Leftrightarrow c_1 = d_1,\ \ldots,\ c_n = d_n$ for $c_i,\ d_i \in \mathbb{R}\ \forall i$.

3 What is MATLAB?

MATLAB (*matrix laboratory*) is a programming language for technical computing. In university environments, it is the standard instructional tool for

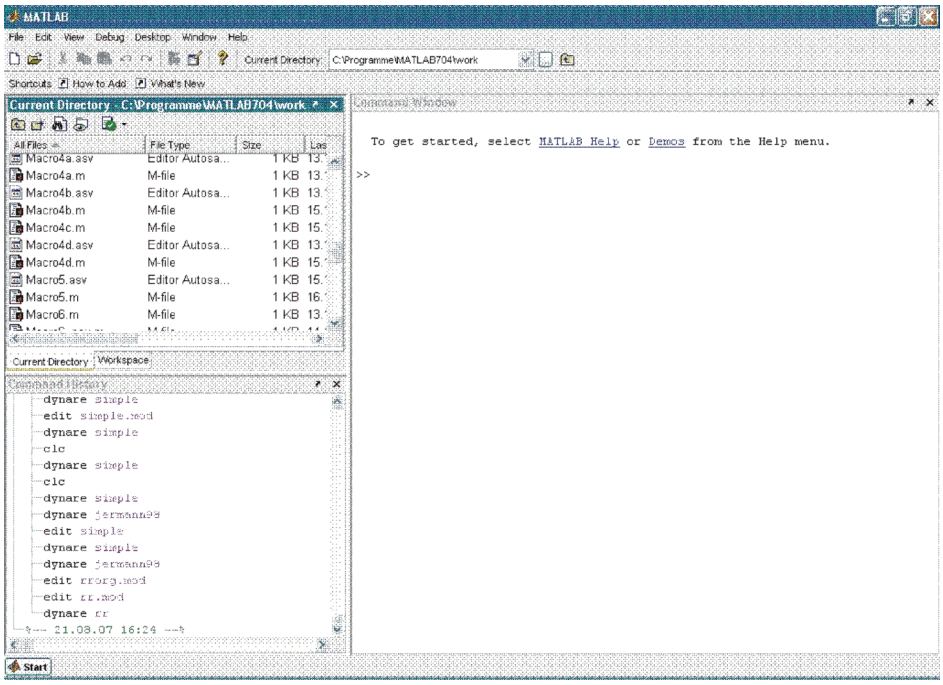
introductory and advanced courses in mathematics, engineering, and science. Basic MATLAB can be used for:

- computations including linear algebra
- data analysis
- polynomials and interpolation
- modeling, simulation and prototyping
- forecasts
- numerical solutions of differential equations
- graphics in 2-D and 3-D including colors and animation
- and a lot of other applications

We will work through some of the applications listed above.
There is a variety of toolboxes which are implemented in MATLAB to solve special classes of problems. Also the program *DYNARE* of M. Juillard uses MATLAB as basis program.

MATLAB's basic data element is an array (i.e., a vector) that does not require dimensioning. This allows us to solve many technical computing problems, especially those with matrix and vector formulations in an easy way.

- To start MATLAB, double-click on the icon on your desktop.
- The *command window* opens and you can enter the commands at the command line prompt >> which are immediately executed after "enter" is pressed.
- The *command history window* is used to view or execute previously run commands and functions.
- The *current directory/workspace window* lists the folders/files in the current directory (=where you are working) or the values and attributes of the variables you have defined.
- The *START* button at the lower left gives you quick access to tools and more.
- The *current directory* line at the top tells you where MATLAB thinks your files are located. This should always point to the folder that you are working in so that your files are saved in your own directory.



– How to open a document depends on the document type:
M-file: Select "File" ⇒ "Open" and select the M-file (filename.m). It opens in the editor/debugger.
Workspace variable: In the workspace browser, double-click the variable. It opens in the array editor.

– To clear the command window, type in the command clc.

3.1 M-Files - An Overview

MATLAB executes a sequence of statements stored in diskfiles, called *M-Files* (because they must have the file type ".m" as the last part of the filename). M-files are created using the editor.
There are two types of M-files: script files and function files.
A script file consists of a sequence of normal MATLAB statements. Suppose we have a file summation.m and type in the command summation, then MATLAB will execute the statements written in the file. Variables in a script file are global and will change the value of variables of the same name in the environment of the current MATLAB session. An M-file can reference other M-files.
In a function file, new functions can be created which will have the same

status as other MATLAB functions. Variables in a function file are local. Here an example:

```
function y = average(x)
% AVERAGE Mean of vector elements
% AVERAGE(x), where x is a column vector, is the mean of vector elements
% Non-vector input results in an error
[m, n] = size(x); % size(x) gives the # m of rows and the # n of columns
if (~ ((n == 1))|(m == 1 & n == 1))
error ('Input must be a column vector!')
end
y=sum(x)/m
```

The % symbol indicates that the rest of the line is a comment, MATLAB ignores the rest of the line.
The first line of the function file has to define the function. It begins with the keyword function, the there follows an output argument y, the function name and in brackets the input argument x. The filename with extension .m has to be equal to the function name average. To call the average function, we can type

```
Z = 1 : 99
average(Z)
```

and get

$$ans = 50.$$

If we type in a statement and press "enter" or "return", MATLAB automatically displays the results in the command window. If we want to suppress parts of the output, this can be done by terminating the statement with a semi-colon. MATLAB performs the computation as usual but does not show up the result. This is often useful for large matrices.

```
>> x = 13; y = 5 + x, z = x^2 + y
   y = -8
   z = 161
```

If a statement does not fit on one line, use "..." and then "return" to indicate that the statement continues on the next line.

Suppose, we have misspelled e.g. the built-in function sqrt as 'sqt', which gives the square root of a number. Then MATLAB responds with

Undefined function or variable 'sqt'.

Instead of retyping the whole command line, we can simply press the ↑ key. Then the former statement is redisplayed. By using the ← key we can correct our statement. Repeated use of ↑ recalls earlier lines. Typing a few characters and then ↑ finds a previous line that begins with those characters.

Text strings are entered in MATLAB surrounded by single quotes ' text '. Text strings can be displayed with the function disp. For example:

```
disp( ' This message is displayed. ' )
```

Error messages are best displayed with the function error:

```
error( ' Something is wrong. ' )
```

since when placed in an M-file, it aborts execution of the M-file.

In an M-file the user can be prompted to enter input data with the input function input. When e.g. the statement

```
iter = input( ' Enter the number of iterations: ' )
```

is encountered, the prompt message is displayed and execution pauses while the user keys in the input data. Upon pressing the "return" key, the data is assigned in the variable iter and execution resumes.

To end a MATLAB session, select "File" ⇒ "Exit MATLAB" or type quit in the command window.

Note: A runaway display or computation can be stopped without leaving MATLAB with CTRL-C.

3.2 MATLAB as a Calculator and Formats of Numbers

We can use MATLAB as a calculator: The basic arithmetic operators are $+$, $-$, $*$, $\hat{}$, / and these are used in conjunction with brackets ( ). The symbol $\hat{}$ is used to get powers. $*$, / and $+$, $-$ work from left to right.

There are several types of numbers, one is the "e" notation which is used for very large numbers, e.g. $-1.03e{+}03 = -1030$. The format how numbers are displayed is controlled by the format-command. The format function affects only how numbers are displayed, not how MATLAB computes or saves them. Suppose we want to display π which is a built-in constant in MATLAB:

>> a = π
a = 3.1416

>> format short e

>> a
a = 3.1416e+00

>> format long e

>> a
a = 3.141592653589793e+00

>> format long

>> a
a = 3.14159265358979

>> format short

>> a
a = 3.1416, which is the default format

For the number 4/3, we will get with the default format 1.3333. If we want the number displayed as a fraction, we have to use the command format rat.

3.3 Very First Steps in MATLAB

Like before, we can differentiate between row and column vectors. In MATLAB, the entries within one row are separated by spaces or commas; the entries in one column are separated by semi-colons or returns. The elements of a vector must be enclosed by brackets. The command length shows the number of entries.

Examples:

- For a column vector, type after the prompt >>:

  u = [1; 3
  5]

and you will get the output:

$$u = \begin{matrix} 1 \\ 3 \\ 5. \end{matrix}$$

– For a row vector, type after >>:

v = [1 3, 5]

and you will get the output:

$v = 1 \;\; 3 \;\; 5.$

– Using the function length after >>:

length(v)

you will get

$ans = 3.$

– Be careful with spaces: The command

v2 = [3 + 4 5]

is not equal to the command

v3 = [3 + 4 5].

In the first case, we will get the output

$v2 = 7 \;\; 5,$

in the second

$v3 = 3 \;\; 4 \;\; 5.$

The vector addition and subtraction is straight forward:

v + v3

gives the output

$ans = 4 \quad 7 \quad 10.$

If the length of two vectors v and $v2$ do not coincide, when typing

v+v2,

the error message

error using ==> plus
Matrix dimensions must agree.

appears.

In all vector arithmetic with vectors of the same length, the operations are done element-wise. So is scalar multiplication:
The command

$\mathsf{v5} = 2 * \mathsf{v} - 3 * \mathsf{v3},$

will give

$v5 \quad = \quad 7 \quad -6 \quad 5.$

We can also create a vector by putting existing vectors together. Suppose we already entered

$\mathsf{w} = [1\ 2\ 3],\ \mathsf{z} = [8\ 9].$

Then the vector

$x \quad = \quad (16 \quad 18 \quad -1 \quad -2 \quad -3)$

is created by typing in

$\mathsf{x} = [2 * \mathsf{z},\ -\mathsf{w}].$

The different entries of e.g. the vector w are denoted by $w(i)$ for $i = 1, \ldots, 3$. If we want to change parts of a vector, we can do it the following way: Type in

$\mathsf{w}(2) = -2,\ \mathsf{w}(3).$

We will get:

$$\begin{array}{rcl} w & = & 1 \quad -2 \quad 3 \\ ans & = & 3 \qquad\quad . \end{array}$$

The so called colon operator is a shortcut for creating a vector. Typing

$$1 : 4$$

will give the vector

$$ans \;=\; 1 \quad 2 \quad 3 \quad 4.$$

More generally, a:b:c produces a vector of entries starting with value a, incrementing by value b (can also be negative or no integer) until it gets to c. The default increment is 1.

Extracting parts of a vector or getting the reverse order of a vector can be done by using the colon operator. For the command

$$\mathsf{r} = [1 : 2 : 6, -1 : -2 : -7],$$

which denotes the vector

$$r \;=\; 1 \quad 3 \quad 5 \quad -1 \quad -3 \quad -5 \quad -7,$$

we get:

- $r(3:6) \Rightarrow \; ans \;=\; 5 \quad -1 \quad -3 \quad -5$
- $r(1:2:7) \Rightarrow \; ans \;=\; 1 \quad 5 \quad -3 \quad -7$
- $r(6:-2:1) \Rightarrow \; ans \;=\; -5 \quad -1 \quad 3$

In general, entering a list of elements, MATLAB will store the variable under the chosen variable name. The command who will list the variables currently in the workspace. A variable can be cleared from the workspace with the command clear variable name. The command clear alone will clear all nonpermanent variables.

When we log out or exit MATLAB, all variables are lost. Invoking the command save before exiting causes all variables to be written to a diskfile named "matlab.mat". When one reenters MATLAB, the command load will restore the workspace to its former state.

4 Matrix Algebra

As before $a_i \ \forall i,\ c,\ d,\ \ldots$ denote scalars and $x,\ y,\ \ldots$ denote vectors.

Definition 3

A *real $m \times n$ matrix* (we will only operate with real values) is a rectangular array of scalars with m rows and n columns denoted by a capital letter (convention). We write

$$A := [a_{ij}] = [a_{ij}]_{m\times n} = [a_{ij}]_{\substack{i=1,\ldots,m \\ j=1,\ldots,n}} = \begin{pmatrix} a_{11} & a_{12} & \ldots & a_{1j} & \ldots & a_{1n} \\ a_{21} & a_{22} & \ldots & a_{2j} & \ldots & a_{2n} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{i1} & a_{i2} & \ldots & a_{ij} & \ldots & a_{in} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{m1} & a_{m2} & \ldots & a_{mj} & \ldots & a_{mn} \end{pmatrix}$$

The number a_{ij} is called the *$(i,j)^{th}$ entry* or element of the matrix A. The index i denotes the row and j the column of the entry a_{ij}.

4.1 Special Kinds of Matrices

- column vector: $n = 1$, i.e., number of columns = 1
- row vector: $m = 1$, i.e., number of rows = 1
- zero matrix: an $m \times n$ matrix with all entries being zero
- square matrix: $m = n$, i.e., number of columns = number of rows
- symmetric matrix: $a_{ij} = a_{ji}\ \forall i, j$ (square matrix!), i.e., interchanging rows and columns leaves the matrix unchanged; we write $A = A'$ (matrix equals its transpose)
- diagonal matrix: a square matrix whose only nonzero elements appear on the main diagonal, i.e., $m = n$ and $a_{ij} = 0\ \forall i \neq j$
- scalar matrix: a diagonal matrix with the same value in all diagonal entries (a scalar is a $(1 \times 1$ matrix))
- identity matrix: a scalar matrix with ones along the main diagonal; we write I_n to denote the identity matrix of order n

- upper-triangular matrix: a square matrix where all entries below the main diagonal are zero, i.e., $a_{ij} = 0$ if $i > j$
- lower-triangular matrix: a square matrix where all entries above the main diagonal are zero, i.e., $a_{ij} = 0$ if $j > i$

Examples:

- symmetric matrix: $A = \begin{pmatrix} 1 & 2 & 3 \\ 2 & 5 & 0 \\ 3 & 0 & 7 \end{pmatrix}$
- diagonal matrix: $A = \begin{pmatrix} 1 & 0 \\ 0 & 2 \end{pmatrix}$
- upper triangular matrix: $A = \begin{pmatrix} 1 & 5 & 3 \\ 0 & 5 & 7 \\ 0 & 0 & 3 \end{pmatrix}$

Definition 4

The *transpose* of an $m \times n$ matrix A is the $n \times m$ matrix $A' = A^T$ obtained by interchanging the rows and columns of A:

$$A := \begin{pmatrix} a_{11} & \dots & a_{1n} \\ a_{21} & \dots & a_{2n} \\ \vdots & & \vdots \\ a_{m1} & \dots & a_{mn} \end{pmatrix} \quad \Rightarrow \quad A' := \begin{pmatrix} a_{11} & \dots & a_{m1} \\ a_{12} & \dots & a_{m2} \\ \vdots & & \vdots \\ a_{1n} & \dots & a_{mn} \end{pmatrix}$$

- x is a column vector, whereas x' is a row vector (by convention)
- $(A')' = A$
- if the matrix A is symmetric, then it holds: $A = A'$

4.2 Elementary Operations

4.2.1 Addition of Matrices and Scalar Multiplication

Let $A := [a_{ij}]$ and $B := [b_{ij}]$ be two $m \times n$ matrices and $c \in \mathbb{R}$ a scalar. Then

$A + B := [a_{ij} + b_{ij}]$

$$\begin{pmatrix} a_{11} & \dots & a_{1n} \\ a_{21} & \dots & a_{2n} \\ \vdots & & \vdots \\ a_{m1} & \dots & a_{mn} \end{pmatrix} + \begin{pmatrix} b_{11} & \dots & b_{1n} \\ b_{21} & \dots & b_{2n} \\ \vdots & & \vdots \\ b_{m1} & \dots & b_{mn} \end{pmatrix} = \begin{pmatrix} a_{11} + b_{11} & \dots & a_{1n} + b_{1n} \\ a_{21} + b_{21} & \dots & a_{2n} + b_{2n} \\ \vdots & & \vdots \\ a_{m1} + b_{m1} & \dots & a_{mn} + b_{mn} \end{pmatrix}$$

$cA := [ca_{ij}]$

$$c \begin{pmatrix} a_{11} & \dots & a_{1n} \\ a_{21} & \dots & a_{2n} \\ \vdots & & \vdots \\ a_{m1} & \dots & a_{mn} \end{pmatrix} = \begin{pmatrix} ca_{11} & \dots & ca_{1n} \\ ca_{21} & \dots & ca_{2n} \\ \vdots & & \vdots \\ ca_{m1} & \dots & ca_{mn} \end{pmatrix}$$

- The subtraction of matrices is defined analogously.
- $(A + B)' = A' + B'$
- Note: The addition (and subtraction) of matrices is only defined for matrices with the **same** dimension.
- $(cA)' = cA'$

Example:

$$\left(\begin{pmatrix} 3 & 2 & 1 \\ 0 & 5 & 2 \\ 0 & 1 & 1 \end{pmatrix} + \begin{pmatrix} 1 & 2 & 1 \\ 0 & 1 & 2 \\ 1 & 2 & 0 \end{pmatrix} \right)' = \begin{pmatrix} 4 & 4 & 2 \\ 0 & 6 & 4 \\ 1 & 3 & 1 \end{pmatrix}' = \begin{pmatrix} 4 & 0 & 1 \\ 4 & 6 & 3 \\ 2 & 4 & 1 \end{pmatrix} =$$

$$\begin{pmatrix} 3 & 0 & 0 \\ 2 & 5 & 1 \\ 1 & 2 & 1 \end{pmatrix} + \begin{pmatrix} 1 & 0 & 1 \\ 2 & 1 & 2 \\ 1 & 2 & 0 \end{pmatrix} = \begin{pmatrix} 3 & 2 & 1 \\ 0 & 5 & 2 \\ 0 & 1 & 1 \end{pmatrix}' + \begin{pmatrix} 1 & 2 & 1 \\ 0 & 1 & 2 \\ 1 & 2 & 0 \end{pmatrix}'$$

Theorem 2

$\mathbf{0}$ is the additive identity since $A + \mathbf{0} = A$ and obviously $A - A = \mathbf{0}$.
It follows immediately that matrix addition (subtraction) is associative and commutative, i.e., $(A + B) + C = A + (B + C)$ and $A + B = B + A$.

4.2.2 Vector Multiplication

Definition 5

The *inner product* or scalar product of two vectors x and y of the same dimension n is a scalar with:

$$\langle x, y\rangle = x'y = (a_1 \cdots a_n) \begin{pmatrix} b_1 \\ \vdots \\ b_n \end{pmatrix} = a_1b_1 + a_2b_2 + \ldots + a_nb_n = \sum_{i=1}^{n} a_ib_i$$

Theorem 3

The inner product has the following properties for any vectors $x,\ y,\ z\ \in \mathbb{R}^n$ and a scalar $c \in \mathbb{R}$

1. symmetry: $\langle x, y\rangle = \langle y, x\rangle$, i.e., $x'y = y'x$
2. bilinearity:

$$\begin{aligned} \langle x, y + z\rangle &= \langle x, y\rangle + \langle x, z\rangle & \qquad \langle x, cy\rangle &= c\langle x, y\rangle \\ \langle x + y, z\rangle &= \langle x, z\rangle + \langle y, z\rangle & \qquad \langle cx, y\rangle &= c\langle x, y\rangle \end{aligned}$$

3. positivity: $\langle x, x\rangle \geq 0$ with equality iff $x = 0$

Definition 6

The *length* of a vector x is defined as the square root of the inner product of the vector:

$$||x|| := \sqrt{\langle x, x\rangle} = \sqrt{a_1^2 + \ldots + a_n^2}$$

($||x||$ is also called the *Euklidean norm* or *2-norm* of a vector x.)
The *distance* of the two vectors x and y is defined as

$$d := ||x - y|| = \sqrt{(x - y)'(x - y)}$$

Theorem 4

The length has the following properties for any vectors $x,\ y\ \in \mathbb{R}^n$ and a scalar $c \in \mathbb{R}$:

1. $||x|| \geq 0$ with equality iff $x = 0$

2. $||cx|| = |c| \cdot ||x||$
3. $||x + y|| \leq ||x|| + ||y||$
4. The Cauchy-Schwartz Inequality holds: $|x'y| \leq \sqrt{x'x}\sqrt{y'y} = ||x||\ ||y||$

Definition 7

γ denotes the *angle* between two vectors x and y in $\mathbb{R}^n$ where:

$$\langle x,\ y\rangle = ||x|| \cdot ||y|| \cdot \cos\gamma$$

x and y are *orthogonal* if $\langle x,\ y\rangle = 0$ as $\cos\ 90° = 0$.
x and y are *parallel* and point in the same direction if $\langle x,\ y\rangle = ||x|| \cdot ||y||$ as $\cos\ 0° = 1$ and point in different directions if $\langle x,\ y\rangle = -||x|| \cdot ||y||$ as $\cos\ 180° = -1$.

4.2.3 Matrix Multiplication

Definition 8

The *product* of an $m \times n$ matrix $A = [a_{ij}]$ and an $n \times p$ matrix $B = [b_{jk}]$ is defined as the $m \times p$ matrix $C := AB = [c_{ik}]$ whose $(i,k)^{th}$ entry is the inner product of the i^{th} row of A and the k^{th} column of B:

$$c_{ik} = a_i' b_k = (a_{i1} \cdots a_{in}) \begin{pmatrix} b_{1k} \\ \vdots \\ b_{nk} \end{pmatrix} = a_{i1}b_{1k} + ... + a_{in}b_{nk} = \sum_{h=1}^{n} a_{ih}b_{hk}$$

- The product of two matrices A and B is only defined iff the number of columns of A is equal to the number of rows of B.
- The product of square matrices of the same dimension is always defined.
- $AB \neq BA$ in general, even if both products are defined. So the matrix multiplication is not commutative. Two matrices C and D are *equal* iff they have the same dimension and each element of C equals the corresponding element of D, i.e.: $C = D \ \Leftrightarrow\ c_{ij} = d_{ij}\ \forall i,\ j$.
- Note: Due to the non commutativity we have to distinguish pre- and post-multiplication (from left, from right).
- $(AB)' = B'A'$, $(ABC)' = C'B'A'$
- $A \cdot \mathbf{0} = \mathbf{0}$ if well defined.
- The matrix A is *idempotent* if $A^2 = A \cdot A = A$.

Example:

$A = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}$ is idempotent: $\begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}\begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}$

Theorem 5

The identity matrix I_n is the multiplicative identity for A, B: $A \cdot I_n = A$ if $A_{m\times n}$ and $I_n \cdot B = B$ if $B_{n\times p}$.
The matrix multiplication is associative and fulfills the distributive law: $(AB)C = A(BC)$ and $(A+B)C = AC + BC \wedge A(B+C) = AB + AC$.
For a scalar c and matrices A and B, it holds: $(cA)B = c(AB) = A(cB)$.

4.2.4 Some Notation

Consider the general $m \times n$ matrix

$$A = \begin{pmatrix} a_{11} & \dots & a_{1j} & \dots & a_{1n} \\ \vdots & \ddots & & & \vdots \\ a_{i1} & \dots & a_{ij} & \dots & a_{in} \\ \vdots & & & \ddots & \vdots \\ a_{m1} & \dots & a_{mj} & \dots & a_{mn} \end{pmatrix}$$

Sometimes it is convenient to have a notation for the i^{th} row of this matrix. As by convention a vector is usually a column vector, we will sometimes denote by a_i the column vector formed by the transpose of row i: $a_i = (a_{i1} \cdots a_{ij} \cdots a_{in})'$. There are authors who define a_i as a row vector! Be careful!

4.3 Matrices in MATLAB

MATLAB works with only one kind of object: a rectangular array, which contains of numbers. So all variables are matrices. A 1×1 matrix is interpreted as a scalar and an $n \times 1$ or a $1 \times n$ matrix as a vector.

Matrices can be introduced into MATLAB in several ways:

- Entered by an explicit list of elements.
- Generated by so called built-in statements and functions.

- Created by M-files (MATLAB scripts).
- Loaded by external data.

We will start by entering matrices as a list of its elements. The following basis conventions have to be taken into account:

- The elements of a row are separated by commas or spaces.
- A semi-colon indicates the end of each row.
- The entire list of elements has to be surrounded by brackets [ ].

If we type in

```
A = [3, 5, 7, 1; 9, 3, 2, 5; 2, 3, 1, 1; 4, 2, 5, 7]
```

MATLAB displays the matrix:

$$A = \begin{matrix} 3 & 5 & 7 & 1 \\ 9 & 3 & 2 & 5 \\ 2 & 3 & 1 & 1 \\ 4 & 2 & 5 & 7 \end{matrix}$$

Once a matrix is entered, it is automatically remembered in the MATLAB workspace. We can then simply refer to it as A.
`[m, n] = size(A)` returns the number of rows m and columns n of A. If A is square, `n = size(A)` returns the dimension of A.

4.3.1 Sum, Transpose and Diag

If we type in

```
sum(A)
```

MATLAB replies with

$$ans = \begin{matrix} 18 & 13 & 15 & 14 \end{matrix}$$

When we do not specify an output variable, MATLAB will automatically use the variable *ans*. `Sum` computes a row vector containing the sum of the columns of A. MATLAB has a preference working with the columns of a matrix; to get the sum of the rows of a matrix, we should take the transpose, then compute

the sum and finally transpose the result. The transpose in MATLAB is denoted by an apostrophe-dot operator '. So

```
A.'
```

produces

$$ans = \begin{matrix} 3 & 9 & 2 & 4 \\ 5 & 3 & 3 & 2 \\ 7 & 2 & 1 & 5 \\ 1 & 5 & 1 & 7 \end{matrix}$$

The command `diag(A)` produces a column vector containing the elements of the main diagonal of A. Typing

```
diag(A)
```

MATLAB returns

$$ans = \begin{matrix} 3 \\ 3 \\ 1 \\ 7 \end{matrix}$$

4.3.2 Subscripts

The element in row i and column j of A is denoted by $A(i, j)$. To compute the sum of the elements in the fourth column of A, we have to type

```
A(1,4) + A(2,4) + A(3,4) + A(4,4)
```

This gives

$$ans = 14$$

but this is not the most elegant way of summing a simple column. We can also use the colon operator introduced before. Type

```
sum(A(1:4,4))
```

and MATLAB will return the same sum as in explicit formulation of the sum. There is another way. The colon operator refers to all elements in a row or column of a matrix, the keyword `end` refers to the last row or column. So we can write

```
sum(A(:,end))
```

to achieve once again 14 as a result.

4.3.3 Mathematical Expressions

MATLAB provides mathematical expressions which involve entire matrices. We can distinguish between variables, numbers, operators and functions.

MATLAB does not require any type declaration or dimension statements with variables. When we type in a new variable name, MATLAB creates automatically the variable and allocates the amount of storage needed. If the variable already exists, MATLAB changes its contents and adjusts the storage. If once a variable is created, entering the variable name returns the matrix assigned to the variable. In general, variable names consist of a letter followed by other letters, numbers, ...

Note: MATLAB is case sensitive; typing uppercase and lowercase letters is not the same!

The function genvarname creates unique and valid variable names. The different formats are already described in section 3.

For numbers, MATLAB uses conventional decimal notation. All numbers are stored internally using the long format.

As operators, we can use

+	*Addition*
−	*Subtraction*
*	*Multiplication*
/	*Division*
\	*LeftDivision*
^	*Power*
'	*Transpose*

For a list of elementary mathematical functions implied in MATLAB type in

```
help elfun
```

We have for example abs, sqrt, exp and sin. Some of the functions are built-in functions, we cannot see the code and they are really efficient. For other functions we can see the code and we can modify it. Some constants are also already implied in MATLAB, e.g. pi, i (for the imaginary unit of a complex number, but we concentrate only on real numbers in this whole course), Inf for

infinity and NaN (not-a-number). It is possible to overwrite any of the built-in constants with a new value, such as

Inf = 1.

To restore the original value we have to type in

clear Inf

4.3.4 Generating Matrices

We will now show explicitly how to generate matrices listed at the beginning of section 4.3. So far we entered matrices as a list of elements. Here is a short explanation of the second possibility: There are six functions that generate basic matrices in MATLAB:

zeros	*All zeros*
ones	*All ones*
rand	*Uniformely distributed random elements*
randn	*Normally distributed random elements*
triu	*Upper triangular part of a matrix*
tril	*Lower triangular part of a matrix*

E.g., if we type in

Z = zeros(2, 4),

MATLAB generates a 2×4 matrix with all entries equal to zero.

The function eye(n) returns an $n \times n$ square identity matrix, i.e., a matrix of dimension n with ones along the main diagonal and zeros elsewhere.

We can also create a matrix by M-files. M-files are text files containing MATLAB code. With the MATLAB editor, we can create a file containing the same statements we have typed so far at the command line, e.g., to generate the matrix A. When we save this document under a name "filename.m", typing filename reads the file and creates the variable A.

Finally the load function reads binary files containing matrices generated by earlier MATLAB sessions or reads text files containing numeric data. The text

file (or excel file) has to be organized as a rectangular table of numbers, separated by blanks (in Excel, different columns) with one row per line and an equal number of elements in each row. The text file has to be stored under the name "filename.dat". Then the statement

```
load filename.dat
```

reads the file and creates the variable "filename".
The command for loading the data of an excel file will be in two steps:

1.) [w] = xlsread(' filename.xls '); % creates the matrix w
2.) y = w(:, 1) % denotes the first column of the data matrix w with y

Another way of creating matrices is called "concatenation". This is a process of joining small matrices to make bigger ones. For example look again at our first matrix A. If we type in

```
B = [A  A + 3; A + 8  A + 1],
```

MATLAB returns

$$B = \begin{matrix} 3 & 5 & 7 & 1 & 6 & 8 & 10 & 4 \\ 9 & 3 & 2 & 5 & 12 & 6 & 5 & 8 \\ 2 & 3 & 1 & 1 & 5 & 6 & 4 & 4 \\ 4 & 2 & 5 & 7 & 7 & 5 & 8 & 10 \\ 11 & 13 & 15 & 9 & 4 & 6 & 8 & 2 \\ 17 & 11 & 10 & 13 & 10 & 4 & 3 & 6 \\ 10 & 11 & 9 & 9 & 3 & 4 & 2 & 2 \\ 12 & 10 & 13 & 15 & 5 & 3 & 6 & 8 \end{matrix}$$

We can also delete rows and columns from a matrix using a pair of brackets [ ]. For

```
A(:, 2) = [ ]
```

we delete the second column of A:

$$A = \begin{matrix} 3 & 7 & 1 \\ 9 & 2 & 5 \\ 2 & 1 & 1 \\ 4 & 5 & 7 \end{matrix}$$

If we want to do arithmetic operations element by element (so handle the matrix like an array), we can use these operators:

$+$ *Addition*
$-$ *Subtraction*
$.*$ $Element - by - element\ multiplication$
$./$ $Element - by - element\ division$
$.\backslash$ $Element - by - element\ left\ division$
$.\hat{}$ $Element - by - element\ power$
$.'$ *Array transpose*

Addition and subtraction are the same for matrices and arrays, but the multiplicative operations are different. MATLAB uses a dot as part of the notation for multiplicative array operations.

4.3.5 Addition, Subtraction, Vector Products and Transpose

Addition and subtraction require both matrices to have the same dimension or one of them to be a scalar. If the dimensions are incompatible, the same error message as in the vector case shows up. Suppose, we have the two matrices

$$A = \begin{pmatrix} 1 & 2 & 3 \\ 2 & 2 & 1 \\ 5 & 1 & 2 \end{pmatrix} \text{ and } B = \begin{pmatrix} 3 & 2 & 3 \\ 1 & 2 & 2 \\ 5 & 4 & 2 \end{pmatrix}$$

Summing A and B in a matrix $\mathsf{X} = \mathsf{A} + \mathsf{B}$ gives

$$X = \begin{matrix} 4 & 4 & 6 \\ 3 & 4 & 3 \\ 10 & 5 & 4 \end{matrix}$$

When we multiply vectors, we get a scalar, the inner product, or a matrix, the outer product.
Suppose we have the two vectors $x,\ y$ with

$$x = \begin{pmatrix} 3 \\ 1 \\ 4 \end{pmatrix}, \quad y = (2 \quad 0 \quad -1)$$

For $\mathsf{z} = \mathsf{y} * \mathsf{x}$, MATLAB returns:

$z = 2$

For Z = x ∗ y, MATLAB returns:

$$Z = \begin{matrix} 6 & 0 & -3 \\ 2 & 0 & -1 \\ 8 & 0 & -4 \end{matrix}$$

If the vectors x and y are both real column vectors, the product $x * y$ is not defined, but the two products

x′ ∗ y and y′ ∗ x

are valid and produce the same scalar.

The transpose operator interchanges a_{ij} and a_{ji}. For

```
X=Z'
```

we get

$$X = \begin{matrix} 6 & 2 & 8 \\ 0 & 0 & 0 \\ -3 & -1 & -4 \end{matrix}$$

4.3.6 Matrix Multiplication

If A is $m \times p$ and B is $p \times n$, their product C is $m \times n$. The product can be defined using MATLAB's for loop, colon notation and vector scalar product:

```
for i = 1 : m
    for j = 1 : n
      C(i, j) = A(i, :) * B(:, j);
    end
end
```

Matrix multiplication is denoted by a single asterisk *. For

$$A = \begin{pmatrix} 3 & 5 & 7 & 1 \\ 9 & 3 & 2 & 5 \\ 2 & 3 & 1 & 1 \\ 4 & 2 & 5 & 7 \end{pmatrix}, \quad B = \begin{pmatrix} -5 & 2 & 8 & 0 \\ 7 & -3 & 1 & 2 \\ 1 & -9 & 1 & 2 \\ 3 & 1 & 1 & 1 \end{pmatrix}$$

```
X=A*B
```

gives

$$X = \begin{array}{rrrr} 30 & -71 & 37 & 25 \\ -7 & -4 & 82 & 15 \\ 15 & -13 & 21 & 9 \\ 20 & -36 & 46 & 21 \end{array}$$

A matrix can be multiplied on the right by a column vector x and on the left by a row vector y. Suppose

$$x = \begin{pmatrix} 1 \\ 5 \\ 4 \\ -1 \end{pmatrix}, \quad y = (2,\ 0,\ 1,\ 1)\,.$$

Then

```
z1 = A * x
```

$$z1 = \begin{array}{r} 55 \\ 27 \\ 20 \\ 27 \end{array}$$

and

```
z2 = y * B
```

$$z2 = \begin{array}{rrrr} -6 & -4 & 18 & 3. \end{array}$$

Suppose we have the matrix A from above and the following matrix C:

$$C = \begin{pmatrix} 3 & 1 & 1 & -1 \\ 1 & 2 & -9 & 0 \\ 5 & 4 & 7 & 3 \end{pmatrix}$$

Typing `Y = A*C`, MATLAB will return

error using ==> mtimes
Inner matrix dimensions must agree.

4.3.7 Vector Norm

The 2-norm of a vector x

$$||x|| = \left(\sum \mid x_i \mid^2\right)^{1/2}$$

is computed by norm(x, 2) or simply norm(x), because 2 is the default value of the norm function implemented in MATLAB. There exist also other norms, so called *p-norms* which have the general form

$$||x||_p = \left(\sum \mid x_i \mid^p\right)^{1/p}.$$

These are not discussed in more detail in this course.

4.4 Trace, Rank and Determinant of a Matrix

Definition 9

The *trace* of an $n \times n$ matrix A is the sum of its (main) diagonal terms, i.e., $tr(A) := \sum_{i=1}^{n} a_{ii}$.

- Let A be $m \times n$ and B be $n \times m$, then AB is $m \times m$ and BA $n \times n$ and we have: $tr(AB) = tr(BA)$.
- $x'Ax$ with $x_{n\times 1}$ and $A_{n\times n}$ is a scalar and $x'Ax = tr(x'Ax) = tr(Axx')$.
- $tr(AB) \neq tr(A)tr(B)$

4.4.1 Excursus: Lines, Planes and Hyperplanes

We will discuss these issues (in point-normal representation) by using a simple example namely that of a budget line:

- *lines* in $\mathbb{R}^2$:
  take the budget line $p_1x_1 + p_2x_2 = m$ and write it in vector notation as $p'x = m$; with p and m given, this equation describes a subset of $\mathbb{R}^2$. Take any x_0 satisfying the equation, i.e., $p'x_0 = m$, then: $p'x_0 = p'x \Leftrightarrow p'(x - x_0) = 0$.
  Hence the line (that are the points satisfying $p'(x - x_0) = 0$) is precisely

the set of points for which $x - x_0$ is orthogonal to p; p is called *normal* vector.

- *planes* in $\mathbb{R}^3$:
  $p_1x_1 + p_2x_2 + p_3x_3 = m$ in vector notation again $p'(x - x_0) = 0$ nothing changed but in 3 dimensions now!
- *hyperplanes* in $\mathbb{R}^n$:
  $p_1x_1 + p_2x_2 + ... + p_nx_n = m$ in vector notation $p'(x - x_0) = 0$; this is the natural extension of lines in $\mathbb{R}^2$ and planes in $\mathbb{R}^3$ which are actually special cases of a hyperplane.
  A *hyperplane* in $\mathbb{R}^n$ is a set of the form $\{x \in \mathbb{R}^n : p'(x - x_0) = 0,\ p \in \mathbb{R}^n \backslash \{0\},\ x_0 \in \mathbb{R}^n\}$

Definition 10

The *column space* of a matrix is the vector space spanned by its column vectors. The *column rank* is the dimension of the column space, i.e., the column rank is the largest number of linearly independent columns of a matrix.
Analogously, the *row space* of a matrix is the vector space spanned by its row vectors. The *row rank* is the dimension of the row space, i.e., the largest number of linearly independent rows of a matrix.

Theorem 6

For any matrix A the column rank equals the row rank, hence the dimension of the row space and the column space of a matrix is the same.

- If A is $m \times n$ and $rank(A) = n$, we have full column rank; if $rank(A) = m$ we have full row rank; notice however that $rank(A) \leq \min(n,\ m)$.
- $rank(A) = rank(A')$
- $rank(AB) \leq \min(rank(A),\ rank(B))$
- If A is $m \times n$ with rank $r \leq m$, then
  - the columns of A span $\mathbb{R}^m$ if $r = m$
  - and do not span $\mathbb{R}^m$ if $r < m$!

Definition 11

The *determinant* of a 2×2 matrix A is defined as

$$\det A := |A| = \det \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} = a_{11}a_{22} - a_{21}a_{12}$$

The *determinant* of a 3×3 matrix A is defined (Sarrus rule) as

$$\det \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} = \begin{array}{ccccccc} a_{11}a_{22}a_{33} & + & a_{12}a_{23}a_{31} & + & a_{21}a_{32}a_{13} \\ -a_{31}a_{22}a_{13} & - & a_{32}a_{23}a_{11} & - & a_{21}a_{12}a_{33} \end{array}$$

To derive the *determinant* for a general $n \times n$ matrix A, we need the following definitions:
Let A be an $n \times n$ matrix and let A_{ij} be the $(n-1) \times (n-1)$ *submatrix* obtained by deleting row i and column j from A. Then the scalar $M_{ij} \equiv \det A_{ij}$ is called the $(i,j)^{th}$ *minor* of A.
The scalar $C_{ij} = (-1)^{i+j} M_{ij}$ is called the $(i,j)^{th}$ *cofactor*.

Theorem 7

If A is an $n \times n$ matrix, then for any row i (expansion along row i)

$$\det A := \sum_{j=1}^{n} (-1)^{i+j} a_{ij} \det A_{ij}$$

and for column j (expansion along column j)

$$\det A := \sum_{i=1}^{n} (-1)^{i+j} a_{ij} \det A_{ij}$$

- The determinant is only defined for a square matrix.
- The determinant of a diagonal matrix is the product of its diagonal entries.
- $\det I_n = 1$
- The determinant of triangular matrices is the product of its diagonal entries.
- $\det A = \det A'$
- If one column of A contains only zeros, then $\det A = 0$.

- The addition of a scalar multiple of one column to another column of A does not change the value of the determinant of A.
- Changing two columns of A only changes the sign of the determinant of A, but not the value.
- The same is true if we change the word "columns" with "rows" (because the columns of A are the rows of A').
- $|A||B| = |A \cdot B|$ but $|A + B| \neq |A| + |B|$ in general.
- The previous result implies $|A^{-1}| = \dfrac{1}{|A|}$ if $|A| \neq 0$.
  proof: $\det A \cdot \det A^{-1} = \det(AA^{-1}) = \det I = 1$

4.5 The Adjoint and the Inverse of a Matrix

Definition 12

The *adjoint matrix* of an $n \times n$ matrix A is defined as the matrix of the cofactors C_{ij} of A:

$$adj\ A := \begin{pmatrix} C_{11} & C_{12} & \dots & C_{1n} \\ C_{21} & C_{22} & \dots & C_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ C_{n1} & C_{n2} & \dots & C_{nn} \end{pmatrix}$$

Theorem 8

$(adj\ A)' \cdot A = \det A \cdot I_n$

Definition 13

An $n \times n$ matrix A is called *invertible*, if there exists an $n \times n$ matrix B such that $AB = I = BA$. The *inverse* B is denoted by $B := A^{-1}$.
If A is not invertible, it is called *singular*. Invertible matrices are also called *regular*.

- It follows from the definition that only square matrices can have an inverse.
- Not all square matrices have an inverse.
- A matrix A can have at most one inverse.

- If A is invertible, then B is unique.
  proof: if $\exists\ C$ with $AC = I = CA$, then
  $B = BI = B(AC) = (BA)C = IC = C$
- I is invertible with $I = I^{-1}$
- $(AB)^{-1} = B^{-1}A^{-1}$ if both A and B are invertible!
- $(ABC)^{-1} = C^{-1}B^{-1}A^{-1}$ if A, B, C are invertible!
- $(A^{-1})^{-1} = A$, $(A')^{-1} = (A^{-1})'$

Examples:

- The zero matrix $\mathbf{0}$ is not invertible.
- For $A = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$ it holds: $AA = I$. A is invertible and $A^{-1} = A$.
- If one column or row of A contains only zeros, then A is not invertible.
- The 2×2 matrix $A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$ is invertible iff $ad - cb \neq 0$. Then

$$A^{-1} = (ad - cb)^{-1} \begin{pmatrix} d & -b \\ -c & a \end{pmatrix}.$$

Theorem 9

The following statements about an $n \times n$ matrix A are equivalent

1. A has an inverse.
2. $rank\ A = n$
3. $\det A \neq 0$
4. The columns of A are linearly independent.
5. A is nonsingular.

Theorem 10

Let A be an $n \times n$ matrix with $|A| \neq 0$. Then A has an inverse given by

$$A^{-1} = \frac{1}{\det A} \cdot (adj\ A)'.$$

4.6 Quadratic Forms and Definiteness

Definition 14

A *quadratic form* on $\mathbb{R}^n$ is a real valued function

$$Q(x_1,\ x_2,\ \ldots,\ x_n) := \sum_{i\leq j}^{n} a_{ij}x_ix_j.$$

Each quadratic form can be represented as $Q(x) = x'Ax$, where A is a (unique) symmetric matrix:

$$\begin{pmatrix} a_{11} & \frac{1}{2}a_{12} & \ldots & \frac{1}{2}a_{1n} \\ \frac{1}{2}a_{12} & a_{22} & \ldots & \frac{1}{2}a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{1}{2}a_{1n} & \frac{1}{2}a_{2n} & \ldots & a_{nn} \end{pmatrix}$$

Example

For $Q(x) = 2x_1^2 + 4x_1x_2 - x_2^2$: $A = \begin{pmatrix} 2 & 2 \\ 2 & -1 \end{pmatrix}$.

Definition 15

Let A be an $n \times n$ symmetric matrix; then A is

1. *positive definite* if $x'Ax > 0\ \forall\ x \neq 0$ in $\mathbb{R}^n$
2. *negative definite* if $x'Ax < 0\ \forall\ x \neq 0$ in $\mathbb{R}^n$
3. *positive semidefinite* if $x'Ax \geq 0\ \forall\ x \neq 0$ in $\mathbb{R}^n$
4. *negative semidefinite* if $x'Ax \leq 0\ \forall\ x \neq 0$ in $\mathbb{R}^n$
5. *indefinite* if $x'Ax > 0$ for some $x \neq 0$ in $\mathbb{R}^n$ and $x'Ax < 0$ for some $x \neq 0$ in $\mathbb{R}^n$

Theorem 11

The quadratic form $Q(x) = ax_1^2 + 2bx_1x_2 + cx_2^2$ on $\mathbb{R}^2$ is

1. positive definite $\Leftrightarrow \det A = ac - b^2 > 0$ and $a > 0$.
2. negative definite $\Leftrightarrow \det A = ac - b^2 > 0$ and $a < 0$.
3. indefinite $\Leftrightarrow \det A = ac - b^2 < 0$.

Definition 16

Let A be an $n \times n$ matrix. The $k \times k$ submatrix of A formed by deleting $n-k$ columns and the same $n-k$ rows from A is called a *k^{th} order principal submatrix* of A. The determinant of a $k \times k$ principal submatrix is called a *k^{th} order principal minor* of A. The k^{th} order principal submatrix A_k obtained by deleting the last $n-k$ rows and columns of A is called the *k^{th} order leading principal submatrix* of A, its determinant the *k^{th} order leading principal minor* of A.

Theorem 12

Let A be an $n \times n$ symmetric matrix, then

1. A is positive definite iff all its n leading principal minors are strictly positive.
2. A is negative definite iff all its n leading principal minors alternate in sign as follows: $|A_1| < 0,\ |A_2| > 0,\ |A_3| < 0,\ \ldots$ The k^{th} order leading principal minor should have sign $(-1)^k$.
3. A is positive semidefinite iff every principal minor of A is non negative.
4. A is negative semidefinite iff every principal minor of odd order is non positive and of even order is non negative.

Theorem 13

A continuous twice differentiable function f on an open, convex subset U of $\mathbb{R}^n$ is concave (convex) in U iff the Hessian $D^2 f(x)$ is negative (positive) semidefinite for all x in U.

4.7 Eigenvalues and Eigenvectors

Definition 17

Let A be an $n \times n$ matrix, v an $n \times 1$ vector and λ a scalar. We call v an *eigenvector* of matrix A with *eigenvalue* λ if $v \neq 0$ and $Av = \lambda v$.
We can rewrite the above equation as $(A - \lambda I) \cdot v = 0$. As $v \neq 0$ this has a solution only if $(A - \lambda I)$ is singular. Hence we could define an *eigenvalue* alternatively as the number λ that when subtracted from each diagonal entry of the $n \times n$ matrix A, transforms A into a singular matrix. Hence we can determine the eigenvalues of A by solving $\det(A - \lambda I) = 0$. This equation is

called the *characteristic polynomial*. The eigenvectors are then obtained from the definition above.

- The diagonal entries of a diagonal matrix A are eigenvalues of A (also for upper and lower triangular matrices).
- A square matrix A is singular iff zero is an eigenvalue of A.
- The trace of an $n \times n$ matrix A equals the sum of its eigenvalues.
- The determinant of an $n \times n$ matrix A equals the product of its eigenvalues.
- For an $n \times n$ matrix A the characteristic equation is an n^{th} order polynomial in λ. Its solution may be n distinct values or may contain repeated roots (it may contain zeros as well). Moreover there is no guarantee that the eigenvalues are real, they may be complex as well.

Definition 18

Let A be an $n \times n$ matrix with eigenvalues $\lambda_1, \ldots, \lambda_n$ and corresponding eigenvectors $v_1, \ldots, v_n$. Then if the matrix $V = [v_1, \ldots, v_n]$ is invertible:

$$V^{-1}AV = \Lambda$$

where Λ is a diagonal matrix with the eigenvalues along the main diagonal. This method is called the *diagonalization* of A.

Theorem 14

For a symmetric $n \times n$ matrix A, it holds:

1. If A is an $n \times n$ symmetric matrix with only real entries, then all its eigenvalues are real and the eigenvectors corresponding to different eigenvalues are orthogonal, that is if v_1 and v_2 are eigenvectors corresponding to eigenvalues $\lambda_1 \neq \lambda_2$, then $v_1' v_2 = 0$.
2. Even if the symmetric A has multiple eigenvalues there still exist n distinct and orthogonal eigenvectors.
3. Hence for a symmetric $n \times n$ matrix we have $AV = V\Lambda$ with $V = [v_1, \ldots, v_n]$ and $\Lambda = diag(\lambda_1, \ldots, \lambda_n)$ and since $v_i' v_j = 0$ for $i \neq j$ we get $V'V = I$. Hence $V' = V^{-1}$ (we call such a matrix *orthonormal*) and it follows that $V'AV = \Lambda$ (Note: we have assumed that all eigenvectors are normalized to unit length).

4. Using the above diagonalization result we have that the rank of a symmetric matrix equals the number of nonzero eigenvalues it contains.
5. A symmetric matrix is positive (negative) definite iff $\lambda_i > 0$ ($\lambda_i < 0$) $\forall\ i = 1,\ \ldots,\ n$. It is positive (negative) semidefinite iff $\lambda_i \geq 0$ ($\lambda_i \leq 0$) $\forall\ i = 1,\ \ldots,\ n$.

4.7.1 QR Decomposition

Suppose A is an $m \times n$ matrix with $rank(A) = n$, i.e., the columns $a_1,\ \ldots,\ a_n$ are linearly independent. Then there exists a decomposition

$$A = QR$$

where Q is an $m \times n$ matrix whose columns $v_1,\ \ldots,\ v_n$ build an orthonormal system, R is an $n \times n$ upper triangular matrix.

Definition 19

For $v_1,\ \ldots,\ v_n$ building an *orthonormal system* it holds that

1. All vectors are different from zero and pairwise orthogonal, i.e.,

$$\langle v_i,\ v_i \rangle \neq 0\ \forall i;\ \langle v_i,\ v_j \rangle = 0\ \forall i \neq j.$$

2. All vectors are normalized to unity, i.e.,

$$\langle v_i,\ v_i \rangle = 1\ \forall i.$$

To calculate the entries for Q and R, we have to use the Gram-Schmidt orthogonalizing method. Suppose the linear independent $a_1,\ \ldots,\ a_n$ are given. We will go in two steps. At first, we get an orthogonal system of vectors $u_i,\ i = 1,\ \ldots,\ n$, then we normalize the calculated vectors to get v_i:

$$\begin{array}{ll}
u_1 := a_1 & v_1 := \frac{u_1}{||u_1||} \\
u_2 := a_2 - \frac{\langle u_1,\ a_2 \rangle}{\langle u_1,\ u_1 \rangle} u_1 & v_2 := \frac{u_2}{||u_2||} \\
u_3 := a_3 - \frac{\langle u_1,\ a_3 \rangle}{\langle u_1,\ u_1 \rangle} u_1 - \frac{\langle u_2,\ a_3 \rangle}{\langle u_2,\ u_2 \rangle} u_2 & v_3 := \frac{u_3}{||u_3||} \\
\vdots & \vdots \\
u_n := a_n - \sum_{i=1}^{n-1} \frac{\langle u_i,\ a_n \rangle}{\langle u_i,\ u_i \rangle} u_i & v_n := \frac{u_n}{||u_n||}
\end{array}$$

These vectors $v_1, \ldots, v_n$ are the columns of Q. For R choose

$$R = \begin{pmatrix} ||u_1|| & \langle v_1,\ a_2\rangle & \ldots & \langle v_1,\ a_n\rangle \\ 0 & ||u_2|| & \ldots & \langle v_2,\ a_n\rangle \\ \vdots & & \ddots & \vdots \\ 0 & \ldots & 0 & ||u_n|| \end{pmatrix}$$

Example

Suppose given the matrix

$$A = \begin{pmatrix} 1 & 3 & 3 \\ -1 & 1 & -1 \\ 1 & -1 & -1 \\ -1 & -3 & -1 \end{pmatrix}.$$

We can calculate

$\langle v_i,\ a_i\rangle$	a_1	a_2	a_3
v_1	2	2	2
v_2	0	4	2
v_3	0	0	2

and finally get

$$Q = \frac{1}{2}\begin{pmatrix} 1 & 1 & 1 \\ -1 & 1 & -1 \\ 1 & -1 & -1 \\ -1 & -1 & 1 \end{pmatrix},\ R = \begin{pmatrix} 2 & 2 & 2 \\ 0 & 4 & 2 \\ 0 & 0 & 2 \end{pmatrix}.$$

4.8 Higher Matrix Algebra in MATLAB

To solve for the trace of a matrix A, we have to type in

`trace(A)`.

The rank of a A is computed with the command

`rank(A)`.

4.8.1 Inverses and Determinants

The two equations $AB = I$ and $BA = I$ have the same solution B, the inverse of A (A square and nonsingular), denoted by A^{-1}. B is computed by the function inv. Assume we have the matrix

$$A = \begin{pmatrix} 3 & 5 & 7 & 1 \\ 9 & 3 & 2 & 5 \\ 2 & 3 & 1 & 1 \\ 4 & 2 & 5 & 7 \end{pmatrix}$$

Then typing

```
B = inv(A)
```

returns

$$B = \begin{matrix} 0.0566 & 0.1887 & -0.2075 & -0.1132 \\ -0.0480 & -0.0995 & 0.5094 & 0.0051 \\ 0.1732 & 0.0017 & -0.3019 & 0.0172 \\ -0.1424 & -0.0806 & 0.1887 & 0.1938 \end{matrix}$$

To compute the determinant of a square matrix A, MATLAB uses the function det. So typing

```
d = det(A)
```

for our example matrix A from above gives

$d = 583.$

Note: The function det has some bad scaling and roundoff error properties. So be careful in using it.

If A is square and nonsingular, then without roundoff errors B = inv(A)*X would theoretically be the same as B = A\ X and Y = X*inv(A) the same as Y = X/A. But the computations involving the backslash and the slash operators are preferable because they require less computer time, less memory and have better error detection properties.

4.8.2 Eigenvalues and Eigenvectors

Let λ denote an eigenvalue of a square matrix A and let v denote an eigenvalue of A. If we have the diagonal matrix Λ with all eigenvalues of A on the main diagonal and the nonsingular matrix V with the eigenvectors corresponding to the eigenvalues of A forming the columns of V, then A can be decomposed as

$$A = V\Lambda V^{-1}.$$

A is said to be diagonalizable.
The statement

lambda = eig(A)

produces a column vector containing the eigenvalues. With two output arguments, eig computes the eigenvectors and stores the eigenvalues in a diagonal matrix. So typing

[V, D] = eig(A)

returns a matrix V containing eigenvectors of A and a diagonal matrix D containing the eigenvalues on the main diagonal.
If a matrix B is not diagonalizable, e.g.

$$B = \begin{pmatrix} 6 & 12 & 19 \\ -9 & -20 & -33 \\ 4 & 9 & 15 \end{pmatrix},$$

typing

[V, D] = eig(B)

produces

$$V = \begin{matrix} 0.4741 & 0.4082 & -0.4082 \\ -0.8127 & -0.8165 & 0.8165 \\ 0.3386 & 0.4082 & -0.4082 \end{matrix}$$

$$D = \begin{matrix} -1.000 & 0 & 0 \\ 0 & 1.000 & 0 \\ 0 & 0 & 1.000 \end{matrix}$$

There is a double eigenvalue at $\lambda = 1$. The second and the third column of V are not linearly independent. They are different normalizations of the single eigenvector corresponding to $\lambda = 1$. So whenever one eigenvalue appears more than once, no full set of linearly independent eigenvectors exists.

To get the coefficients in the characteristic polynomial of B, we have to type

poly(B).

MATLAB returns the coefficient as a row vector

$ans = 1.0000 \quad -1.0000 \quad -1.0000 \quad 1.0000$

which denotes the polynomial

$\lambda^3 - \lambda^2 - \lambda + 1.$

5 Systems of Linear Equations

Consider a general system of m linear equations in n unknowns $Ax = b$ where A is $m \times n$, x is $n \times 1$ and b is $m \times 1$:

$$\begin{array}{ccccccccc} a_{11}x_1 & + & a_{12}x_2 & + & \dots & + & a_{1n}x_n & = & b_1 \\ a_{21}x_1 & + & a_{22}x_2 & + & \dots & + & a_{2n}x_n & = & b_2 \\ \vdots & & & \vdots & & & \vdots & & \vdots \\ a_{m1}x_1 & + & a_{m2}x_2 & + & \dots & + & a_{mn}x_n & = & b_m. \end{array}$$

Definition 20

We call A the *coefficient matrix* and the matrix formed by adding b as an additional column to A the *augmented matrix* $\hat{A}$:

$$\hat{A} := \left(\begin{array}{cccc|c} a_{11} & a_{12} & \dots & a_{1n} & b_1 \\ a_{21} & a_{22} & \dots & \vdots & \vdots \\ \vdots & \vdots & & \vdots & \vdots \\ a_{m1} & \dots & \dots & a_{mn} & b_m \end{array} \right)$$

Theorem 15

Performing any of the following three *elementary row operations* on the matrix $\hat{A}$ results in a new augmented matrix which represents a system of linear equations equivalent to the system represented by $\hat{A}$:

1. interchanging two rows of the matrix
2. changing a row by adding to it a multiple of another row
3. multiplying each element in a row by the same nonzero number

5.1 The Gauss-Jordan Elimination Algorithm

To solve the system of linear equations and to decide if there is no, one or infinitely many solutions, we have to bring the described system $Ax = b$ in row echelon form:

1. Change the system by using the three elementary row operations to get

$$\begin{array}{ccccccccc} a'_{11}x_1 & + & a'_{12}x_2 & + & \dots & + & a'_{1n}x_n & = & b'_1 \\ & & a'_{22}x_2 & + & \dots & + & a'_{2n}x_n & = & b'_2 \\ & & \vdots & & & & \vdots & & \vdots \\ & & a'_{m2}x_2 & + & \dots & + & a'_{mn}x_n & = & b'_m. \end{array}$$

   This can be done by

   - If $a_{11} \neq 0$, do not change the first equation and subtract
     $\frac{a_{21}}{a_{11}}$ times the first equation from the second equation
     $\vdots$
     $\frac{a_{m1}}{a_{11}}$ times the first equation from the last equation
   - If $a_{11} = 0$, search an equation, where $a_{i1} \neq 0$, change the first row with the i^{th} row and proceed like before.
   - If $a_{i1} = 0 \; \forall \; i = 1, \; \dots, \; m$, then there is nothing to do.

2. Leave the first equation of the transformed system unchanged and proceed for the left $m - 1$ equations described in 1.

3. After at most $m-1$ steps, we will end in a system with echelon form:

$$\begin{array}{ccccccccccccc}
c_{1j_1}x_{j_1} & + & c_{1j_2}x_{j_2} & + & \dots & + & c_{1j_r}x_{j_r} & + & \dots & + & c_{1n}x_n & = & d_1 \\
 & & c_{2j_2}x_{j_2} & + & \dots & + & c_{2j_r}x_{j_r} & + & \dots & + & c_{2n}x_n & = & d_2 \\
 & & & & & & \vdots & & & & \vdots & & \vdots \\
 & & & & & & c_{rj_r}x_{j_r} & + & \dots & + & c_{rn}x_n & = & d_r \\
 & & & & & & & & & & 0 & = & d_{r+1} \\
 & & & & & & & & & & \vdots & & \vdots \\
 & & & & & & & & & & 0 & = & d_m
\end{array}$$

Definition 21

The scalars $c_{1j_1},\ c_{2j_2},\ \dots,\ c_{rj_r}$ are different from zero; they are called the *pivot entries*. The corresponding variables $x_{j_1},\ x_{j_2},\ \dots,\ x_{j_r}$ are called *pivot variables*. The number r for which the left hand side of the equations is not equal to zero, is called the *rank* of the system of linear equations. It is equal to the rank of the matrix A. So the rank of a matrix A is equal to the number of nonzero rows in its echelon form. If the i^{th} column of a row echelon matrix contains a pivot, we call x_i a *basic variable*. If the i^{th} column does not contain a pivot we call x_i a *free or nonbasic variable*. Each basic variable is either unambiguously determined or a linear expression of the free variables.

5.2 No, one or infinitely many Solutions

- The echelon form implies that $r \leq n$.
- We transform the system of linear equations $Ax = b$ into an echelon form system of linear equations $Cx = d$, i.e., we triangularize the matrix A, obtaining C.
- **case 1:** One $d_i,\ i \in r+1,\ \dots,\ m$ is unequal zero. Then the equations contradict themselves and there exists no solution of the system of linear equations.
- **case 2:** $d_{r+1},\ \dots,\ d_m = 0$. Then the system of linear equations has at least one solution. We can find the solutions in the following way:
  1.) Choose an arbitrary value for all $n-r$ variables x_i, which are no pivot variables, say $x_i = a_i$.
  2.) Solve for the values of the pivot variables by beginning with the last equation. Solve for x_{j_r}. Then solve for $x_{j_{r-1}}$ and so on until you achieve the solutions for all pivot variables.

Theorem 16

In the following cases, the system of linear equations has at least one solution:

- $r = m$, i.e., the number of equations is equal to the rank of the system: in this case, no left hand side of an equation disappears.
- The system of equations is *homogeneous*, i.e., $b_1 = \ldots = b_m = 0$. A homogeneous system has always a solution, namely $x_1 = \ldots = x_n = 0$, the so called *trivial* solution.

Consider the system $Ax = b$. Then

- If $m < n$:
  1.) $Ax = 0$ has infinitely many solutions.
  2.) For any given b, $Ax = b$ has infinitely many or no solutions.
  3.) If $rank\ A = r = m$, $Ax = b$ has infinitely many solutions for every b.
- If $m > n$:
  1.) $Ax = 0$ has one or infinitely many solutions.
  2.) For any given b, $Ax = b$ has no, one or infinitely many solutions.
  3.) If $rank\ A = r = n$, $Ax = b$ has no or one solution for every b.
- If $m = n$:
  1.) $Ax = 0$ has one or infinitely many solutions.
  2.) For any given b, $Ax = b$ has no, one or infinitely many solutions.
  3.) If $rank\ A = r = m = n$, $Ax = b$ has exactly one solution for every b.
- If the rank of a system of m linear equations in n unknowns is r, then there exists no solution or the value of $n - r$ variables can be chosen arbitrary.
- There exists a solution of the system $Ax = b \Leftrightarrow rank\ A = rank\ \hat{A}$.
- Let the vectors $a_1,\ a_2,\ \ldots,\ a_n$ in $\mathbb{R}^m$ be the columns of the $m \times n$ matrix A; then these vectors are linearly independent iff the only solution to $Ax = 0$ is $x = 0$.
- Let $a_1,\ \ldots,\ a_m$ be a set of m vectors in $\mathbb{R}^m$ and $b \in \mathbb{R}^m$. Form the matrix $A = (a_1,\ \ldots,\ a_m)$. Then b lies in the space spanned by $a_1,\ \ldots,\ a_m$ iff $Ax = b$ has a solution x. Moreover, $a_1,\ \ldots,\ a_m$ span $\mathbb{R}^m$ iff $Ax = b$ has a solution for every $b \in \mathbb{R}^m$.
- $Ax = b$ has a solution iff $rank\ A = rank(A|b)$.

Example

Now consider the following augmented matrices:

$$A = \left(\begin{array}{ccc|c} 1 & 2 & 3 & 4 \\ 0 & 1 & 2 & 3 \\ 0 & 0 & a & 0 \end{array}\right) \qquad B = \left(\begin{array}{ccc|c} 1 & 2 & 3 & 4 \\ 0 & 1 & 2 & 3 \\ 0 & 0 & b & 2 \end{array}\right)$$

First, look at A: If $a = 0$, x_3 is a free variable and the system has infinitely many solutions. If $a \neq 0$, $x_3 = 0,\ x_2 = 3$ and $x_1 = -2$, i.e., we have a unique solution.
Second, look at B: if $b = 0$, there does not exist a solution. If $b \neq 0$ there exists a unique solution depending on the value of b.

5.3 Calculation of an Inverse with Elementary Operations

Theorem 17

To get the inverse of a matrix A, we can also think of "transforming a system of linear equations". We know:
For A and B being two $n \times n$ matrices, B is the inverse of A iff

$$Ax = y \ \Leftrightarrow \ x = By \ \forall \ x,\ y \in \mathbb{R}^n.$$

Using the Gauss-Jordan elimination algorithm, we can decide whether an inverse exists: If we get a row such that all coefficients of the left hand side of $Ax = y$ are zero, then A is not invertible. Otherwise A is invertible and $x = By$ delivers $B = A^{-1}$. Precisely, we obtain the inverse by
1.) augmenting the $n \times n$ matrix A by the $n \times n$ identity matrix, i.e., forming the $n \times 2n$ matrix $(A|I_n)$.
2.) using elementary row operations to bring this augmented matrix in the echelon form $(I_n|B)$ with $B = A^{-1}$.
Note: If it is not possible to perform such row operations, A does not have an inverse.

Theorem 18

For any $n \times n$ matrix A, the following statements are equivalent

1. A is invertible.

2. A has full rank n ($rank\ A = n$), i.e., the columns of A are linearly independent.
3. A is nonsingular.
4. The system $Ax = b$ has a unique solution for every b given by $x = A^{-1}b$.
5. There is no all-zero row in the row echelon form of A.
6. The only vector for which $Ax = 0$ is $x = 0$.

Theorem 19

For an $n \times n$ matrix A, the following statements are equivalent:

1. The homogeneous system $Ax = 0$ has a solution $x \neq 0$ (a *nontrivial* solution).
2. A is not invertible.
3. $rank\ A < n$, i.e., the columns of A are linearly dependent.

5.4 Basis and Fundamental Systems of Solutions

Recall

- The number of vectors forming a basis of a set B is called the dimension of B.
- The column space of an $m \times n$ matrix A is the subspace of $\mathbb{R}^m$ spanned by its column vectors and the column rank is the dimension of the column space.
- The row space of an $m \times n$ matrix A is the subspace of $\mathbb{R}^n$ spanned by its row vectors and the row rank is the dimension of the row space.

Definition 22

Let A be an $m \times n$ matrix with echelon form A_r:
A column of A is a *basic column* if the corresponding column of A_r contains a pivot variable. While the subspaces $Col(A)$ and $Col(A_r)$ are generally different, the basic columns of A form a *basis* for $Col(A)$. One basis for $Col(A)$ is the set of those columns of A_r containing a pivot variable.

Theorem 20

For the $m \times n$ matrix A with echelon form matrix A_r, it holds: The subspace $Row(A)$ is the same as the subspace $Row(A_r)$. The nonzero row vectors of A_r are a basis for $Row(A)$ and hence the dimension of $Row(A)$ is equal to the rank of A. So:
$\dim Col(A) = \dim Row(A) = rank\ A$

Definition 23

Let A be an $m \times n$ matrix. The set of solutions to the homogeneous system $Ax = 0$ is a subspace of $\mathbb{R}^n$ called the *nullspace* $Null(A)$ of A with dimension $\dim Null(A) = n - rank\ A$. For $q := n - rank\ A$, every basis of $Null(A)$ contains of q vectors. Every basis $v_1, \ldots, v_q$ of $Null(A)$ is called a *fundamental system of solutions* of $Ax = 0$. The general solution of the homogeneous system of linear equations is given by all linear combinations $x_H = c_1v_1 + \ldots + c_qv_q$ with $c_1, \ldots, c_q \in \mathbb{R}$ given and $v_1, \ldots, v_q \in \mathbb{R}^n$.

5.4.1 How to get a Fundamental System of Solutions to $Ax = 0$?

By the definition of a fundamental system, we have to find q linear independent solutions $v_1, \ldots, v_q$ with $q = n - rank\ A$ ($\equiv$ number of free variables $\equiv$ $n-$ number of basic variables) to solve the homogeneous system. To get the solutions, do the following:

1. Convert A with elementary row operations into an echelon form.
2. Let $x_{j_1}, \ldots, x_{j_q}$ be the free variables of the system and let $q = n - rank\ A$. To find v_1 choose $x_{j_1} = 1,\ x_{j_2} = 0, \ldots, x_{j_q} = 0$ and get the solution for v_1.
3. Repeat step 2. to get $v_2, \ldots, v_q$.

Example

$$\begin{array}{rcrcrcrcl} x_1 & + & 2x_2 & - & x_3 & + & x_4 & = & 0 \\ 2x_1 & & & + & x_3 & - & 2x_4 & = & 0 \\ -x_1 & + & 2x_2 & - & 2x_3 & + & 3x_4 & = & 0 \end{array} \qquad \begin{array}{rrrr} 1 & 2 & -1 & 1 \\ 2 & 0 & 1 & -2 \\ -1 & 2 & -2 & 3 \end{array}$$

$$\begin{array}{rcrcrcrcl} x_1 & + & 2x_2 & - & x_3 & + & x_4 & = & 0 \\ & - & 4x_2 & + & 3x_3 & - & 4x_4 & = & 0 \\ & & & & & & 0 & = & 0 \end{array} \qquad \begin{array}{rrrr} 1 & 2 & -1 & 1 \\ 0 & -4 & 3 & -4 \\ 0 & 0 & 0 & 0 \end{array}$$

Pivot variables: $x_1,\ x_2$
Free variables: $x_3,\ x_4$

Choose $x_3 = 1,\ x_4 = 0$ to get $v_1 = \begin{pmatrix} -\frac{1}{2} \\ \frac{3}{4} \\ 1 \\ 0 \end{pmatrix}$.

Choose $x_3 = 0,\ x_4 = 1$ to get $v_2 = \begin{pmatrix} 1 \\ -1 \\ 0 \\ 1 \end{pmatrix}$.

The fundamental system of solutions is the given by:

$$x = c_1 \begin{pmatrix} -\frac{1}{2} \\ \frac{3}{4} \\ 1 \\ 0 \end{pmatrix} + c_2 \begin{pmatrix} 1 \\ -1 \\ 0 \\ 1 \end{pmatrix} \text{ with } c_1,\ c_2 \in \mathbb{R}$$

5.4.2 How to get a Fundamental System of Solutions to $Ax = b$?

The solution set to the non-homogeneous system $Ax = b$ is not a subspace of $\mathbb{R}^n$ but has the same dimension as $Null(A)$, namely $q := n - rank\ A$.
Suppose, there exists a special solution x_S to the inhomogeneous system $Ax = b$, then the general solution is of the form:

$$x = x_S + x_H = x_S + c_1 v_1 + \ldots + c_q v_q,\ c_i \in \mathbb{R}$$

where x_S is a special solution for $Ax = b$ and $v_1,\ \ldots,\ v_q$ is a fundamental system of solutions for $Ax = 0$.

So we have to find a special solution (we can get a special solution by setting all free variables equal to zero and solving then for the basic variables) and finally find a fundamental system of solutions for the homogeneous system to get the general solution for $Ax = b$.

5.5 Cramer's Rule for Solving a System $Ax = b$

Cramer's rule describes a way of solving a system of linear equations $Ax = b$ by using the determinant of the matrix A.

Theorem 21

Let A be an $n \times n$ nonsingular matrix, i.e., $|A| \neq 0$. Then the unique solution $x^* = (x_1^*, ..., x_n^*)$ to the linear system of n equations in n unknowns $Ax = b$ is given by

$$x_i^* = \frac{\det D_i}{\det A}$$

where D_i is the matrix obtained by replacing the i^{th} column of A with the right hand side $b = b_1, \ \ldots, \ b_n$.

Example

Consider the following matrix $A = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}$ of the system

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 &= b_1 \\ a_{21}x_1 + a_{22}x_2 &= b_2 \end{aligned}$$

Then

$$x_1 = \frac{\begin{vmatrix} b_1 & a_{12} \\ b_2 & a_{22} \end{vmatrix}}{\det A} \quad \text{and} \quad x_2 = \frac{\begin{vmatrix} a_{11} & b_1 \\ a_{21} & b_2 \end{vmatrix}}{\det A}$$

5.6 Solving Linear Equation Systems in MATLAB

Given two matrices A and B, does there exist a unique vector x so that $Ax = B$ or $xA = B$?

Consider the following example:

$$7x = 21$$

The solution to this problem is not obtained by computing the inverse of 7; MATLAB solves in general such equations without computing the inverse of a matrix in the system of linear equations.

Although it is not standard in mathematics, MATLAB uses the division terminology to describe the solution of a general system of linear equations.

$x = A\backslash B$ denotes the solution to the equation $Ax = B$.
$x = B/A$ denotes the solution to the equation $xA = B$.

The dimension compatibility conditions for $x = A\backslash B$ require the two matrices A and B to have the same number of rows. The solution x then has the same number of columns as B and its row dimension is equal to the column dimension of A. For $x = B/A$, the role of rows and columns are interchanged.

In general, the backslash operator is used far more frequently than the slash operator because, mostly systems of linear equations of the form $Ax = B$ are considered. So we will describe the following properties with the backslash operator. To get similar properties with the slash operator, we can draw conclusions, knowing the following identity holds true:

$$(B/A)' = (A'\backslash B')$$

Suppose from now on, we have a system $Ax = B$. We will look at the following three types of systems. Let A be an $m \times n$ matrix:

1. $m = n$ $\Rightarrow$ square system $\Rightarrow$ unique solution
2. $m > n$ $\Rightarrow$ overdetermined system $\Rightarrow$ at least one solution
3. $m < n$ $\Rightarrow$ underdetermined system $\Rightarrow$ trivial solution with at most m nonzero components

5.6.1 Square System

Suppose, we have a square matrix A and a single right-hand side column vector b. The solution $x = A\backslash b$ is the same size as b. If A and B are square and the same size, then $x = A\backslash B$ is also that size.

If A is singular, then the solution to $Ax = B$ either does not exist or is not unique. The backslash operator $A\backslash B$ issues a warning if A is nearly singular and raises an error condition if exact singularity is detected.

Example

Suppose, the following system $Ax = b$

$$A = \begin{pmatrix} 1 & 5 & 3 \\ 0 & 0 & 0 \\ -2 & 3 & 7 \end{pmatrix}, \quad b = \begin{pmatrix} 3 \\ 0 \\ 5 \end{pmatrix}$$

is given. Typing

```
x=A\b
```

MATLAB returns

$$Warning : Matrix\ is\ singular\ to\ working\ precision.$$

5.6.2 Overidentified Systems

We often have overidentified systems in various kinds of curve fitting to experimental data. Lets look at an example on OLS-procedures. Doing least squares fit means minimizing the sum of the squares of the deviations of the data from the model.
Suppose, a rectangular matrix A does not have linearly independent columns (*rank deficient*). In that case, the least squares solution to $Ax = B$ is not unique. The backslash operator $A\backslash B$ issues a warning if A is rank deficient and produces a basis solution that has as few nonzero elements as possible.

5.6.3 Underdetermined Systems

Underdetermined systems involve more unknowns than equations. When they are accompanied by additional constraints, they are the purview of linear programming. By itself, the backslash operator deals only with the unconstraint system. The solution is never unique. MATLAB finds a basic solution which has at most m nonzero components, but even this may not be unique. (For completeness: The particular solution computed is determined by the so called QR factorization with column pivoting.)

Example

Suppose, we have

$$A = \begin{pmatrix} 6 & 8 & 7 & 3 \\ 3 & 5 & 4 & 1 \end{pmatrix}, \quad b = \begin{pmatrix} 1 \\ 2 \end{pmatrix}$$

and we choose

```
format rat
p = A\b
```

The we get

$$p = \begin{matrix} 0 \\ \frac{5}{7} \\ 0 \\ -\frac{11}{7} \end{matrix}$$

This is a special solution to the system. The complete solution to this underdetermined system can be characterized as adding an arbitrary vector from the null space, which can be found using the null function with the option requesting a rational basis:

```
Z = null(A, 'r')
```

$$Z = \begin{matrix} -\frac{1}{2} & -\frac{7}{6} \\ -\frac{1}{2} & \frac{1}{2} \\ 1 & 0 \\ 0 & 1 \end{matrix}$$

It can be confirmed that A*Z is zero and any vector of the form

$$x = p + \sum_{i=1}^{n} q_i z_{.i}$$

for any combination of arbitrary scalars $q_i \in \mathbb{R}$ with n the number of columns of Z and $z_{.i}$ the ith column of Z satisfies $Ax = b$.

5.6.4 Row Echelon Form and Inverse

Suppose, we have the matrix

$$A = \begin{pmatrix} 16 & 3 & 2 & 13 \\ 5 & 10 & 11 & 8 \\ 9 & 6 & 7 & 12 \\ 4 & 15 & 14 & 1 \end{pmatrix}$$

The determinant of this matrix is zero, A is singular:

```
d = det(A)
```
$$d = 0$$

The reduced form echelon form of A is not the identity and has a zero row:

```
R = rref(A)
```

gives

$$R = \begin{matrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & -3 \\ 0 & 0 & 1 & 3 \\ 0 & 0 & 0 & 0 \end{matrix}$$

Since the matrix is singular, it does not have an inverse. If you try to compute the inverse with

```
B = inv(A)
```

you will get the warning message:

Warning : Matrix is close to singular or badly scaled.

6 Vector and Matrix Differentiation - an Introduction

Definition 24

Define the *gradient vector* of a function f from $\mathbb{R}^n$ to $\mathbb{R}$ as the column vector formed by its partial derivatives i.e., for $f(x_1, \ldots, x_n) = f(x)$ where $x = (x_1, \ldots, x_n)'$ is an $n \times 1$ vector we can write the *gradient* as

$$\frac{\partial f(x)}{\partial x} = \begin{pmatrix} \partial f(x)/\partial x_1 \\ \vdots \\ \partial f(x)/\partial x_n \end{pmatrix} = (Df(x))' = \nabla f(x)$$

Note that the gradient is a column vector, the shape of the derivative is determined by the denominator of the derivative.
When we differentiate the gradient with respect to the row vector $x' = (x_1, \ldots, x_n)$, we obtain the *Hessian*

$$H(x) = \frac{\partial^2 f(x)}{\partial x' \partial x} = \frac{\partial(\partial f(x)/\partial x)}{\partial x'} = \left[\frac{\partial(\partial f(x)/\partial x)}{\partial x_1}, \cdots, \frac{\partial(\partial f(x)/\partial x)}{\partial x_n}\right] = D^2 f(x)$$

Note that each column of H is the derivative of the gradient with respect to the corresponding variable in x'.

6.1 Taylor Approximation in $\mathbb{R}^n$

We hence can write the *first order Taylor approximation* of $f(x)$ around x_0 as:

$$f(x) \approx f(x_0) + \left(\frac{\partial f(x_0)}{\partial x}\right)'(x - x_0) = f(x_0) + Df(x_0)(x - x_0)$$

The *second order Taylor approximation* is given by

$$\begin{aligned} f(x) &\approx f(x_0) + \left(\frac{\partial f(x_0)}{\partial x}\right)'(x - x_0) + \frac{1}{2}(x - x_0)' H(x_0)(x - x_0) \\ &= f(x_0) + \underbrace{Df(x_0)(x - x_0)}_{inner\ product} + \underbrace{\frac{1}{2}(x - x_0)' D^2 f(x_0)(x - x_0)}_{quadratic\ form} \end{aligned}$$

6.2 Differentiation of Inner Products

We are searching for a vector expression of the derivative of an inner product. Recall that for two $n \times 1$ vectors $x' = (x_1, \ldots, x_n)$ and $y' = (y_1, \ldots, y_n)$ the inner product is defined as

$$y'x = x'y = \sum_{i=1}^{n} x_i y_i.$$

It then holds:

$$\frac{\partial(y'x)}{\partial x} = \begin{pmatrix} \partial(y'x)/\partial x_1 \\ \vdots \\ \partial(y'x)/\partial x_n \end{pmatrix} = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix} = y = \frac{\partial(x'y)}{\partial x}$$

Note that $b = y'x$ is simply a linear function.

In a set of linear functions $b = Ax$ we have the following where A is $m \times n$, x is $n \times 1$ and b is $m \times 1$:

$$b = \begin{pmatrix} b_1 \\ \vdots \\ b_m \end{pmatrix} = Ax = \begin{pmatrix} \sum_{i=1}^{n} a_{1i} x_i \\ \vdots \\ \sum_{i=1}^{n} a_{mi} x_i \end{pmatrix}$$

Hence:

$$\frac{\partial b}{\partial x} = \frac{\partial(Ax)}{\partial x} = (\frac{\partial b_1}{\partial x} \cdots \frac{\partial b_m}{\partial x}) = A'$$

and

$$\frac{\partial b}{\partial x'} = \frac{\partial(Ax)}{\partial x'} = A = \frac{\partial(x'A)}{\partial x}$$

6.3 Differentiation with respect to a Vector in MATLAB

MATLAB can differentiate a column vector b with respect to a row vector x and uses for this the implemented Symbolic Math Toolbox. The command

jacobian(b,x) computes the Jacobian of b with respect to x. b is a symbolic scalar expression or a symbolic column vector, x is a symbolic row vector. The $(i,j)^{th}$ entry of the result is $\frac{\partial b_i}{\partial x_j}$. As a first step, symbolic variables have to be created with the command syms. Then we can use the command jacobian to derive the differentiation result.

Example

```
syms k l m
b = [k*l*m; l; k+m]
x = [k,l,m]
J = jacobian(b,x)
```

returns

```
J =
[l*m, k*m, k*l]
[  0,   1,   0]
[  1,   0,   1]
```

6.4 Differentiation of Quadratic Forms

Recall that a quadratic form can be written as

$$y = x'Ax = \sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij}x_i x_j$$

where A is an $n \times n$ matrix and x is an $n \times 1$ vector.

The differentiation results of quadratic forms can be summarized as

- $\dfrac{\partial x'Ax}{\partial x} = (A + A')x$
- If A is symmetric, then $\dfrac{\partial x'Ax}{\partial x} = 2Ax$.
- 2nd derivative: $\dfrac{\partial^2 x'Ax}{\partial x \partial x'} = A + A'$

- $\dfrac{\partial x'Ax}{\partial a_{ij}} = x_i x_j$, hence $\dfrac{\partial x'Ax}{\partial A} = xx'$
- $\dfrac{\partial y'Ax}{\partial A} = yx'$

References

[1] Hoffmann F. , (2006), Lecture Notes on Linear Algebra

[2] Chiang A., (1984), Fundamental Methods of Mathematical Economics

[3] Dhrymes P., (2000), Mathematics for Econometrics

[4] Martin A., (2002), Lineare Algebra I und II, TU Darmstadt

[5] Sigmon K., (1993), MATLAB Primer

[6] Mathworks, Getting Started with MATLAB
http://www.mathworks.com/access/helpdesk/help/techdoc/matlab.shtml

[7] Graham I., (2005), MATLAB Manual and Introductory Tutorials